IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 11, November 2016

Android Joystick

Hiral Rayani[1], Vinaya Sawant[2]

Student, Information Technology, Dwarkadas J. Sanghvi College of Engineering, Mumbai, India [1]

Assistant Professor, Information Technology, Dwarkadas J. Sanghvi College of Engineering, Mumbai, India [2]

Abstract: The project idea presents the design and implementation of turning Android smart phones into computer remote controllers, which allow users to wirelessly operate a computer via Bluetooth connectivity. For user's comfort, the joystick proposed is customizable according to the need of application. The application provides users with different keys of varied colors, shapes and sizes. In comparison to the existing systems, the proposed solution has an advantage of being more user-friendly, provides a stable bluetooth connection and has minimum delay in performing remote control operations.Another advantage of the proposed system is its distributed support- multiple android devices can get connected to the server at the same time, this is very useful for multiplayer gaming.

Keywords: Android, remote controller, Bluetooth, joystick.

I. INTRODUCTION

Personal computers are a backbone for our daily work and also provide recreational use as per our interests. They have become an inseparable part of our quotidian life. This is also applicable to the smart phones, which have transformed into multifunctional devices with almost same features as computers have. Remote control of a desktop computer is a convenient feature that could be performed by handheld devices.

The proposed application can ease the use of many applications giving the user the ability to control them from any place in the room wirelessly. It is suitable for implementing advanced interaction techniques. Furthermore the proposed application is a convenient substitution for the costly wired joysticks. The customization of the joystick provides flexibility and ease of access for the user controlling the desktop remotely.

II. SYSTEM ARCHITECTURE

The proposed application relies on a smart device running Android OS (smart phone or tablet) and a Windows-based PC and offers a virtual joystick that send commands to the desktop OS in real time. Most of the commercially available desktop controllers have fixed keyboard layouts for operating the desktops which confines the user to the standardized options which the companies avail.

However the proposed android application allows the users to customize their virtual joystick according to their needs without placing any standard layout constraints on the users. In a distributed environment this application serves best for multiplayer gaming. Java is the programming language used on both the desktop and the mobile devices, thus giving the opportunity for easier porting to other desktop operating systems. The server application needs to be installed on the desktop or personal computer that is to be controlled or operated. The client android application needs to be installed on the device that will control the remote pc.

The software application proposed in this work consists of two main parts - client (Android application) and server (desktop application). They communicate with a simple text-based protocol via Bluetooth.

A. Server Application

The server application software resides on the desktop PC. When the server is started it awaits connection through Bluetooth. Java SE API is used to provide remote control functionality. The architecture follows the standard practices - a business logic layer that comprises of Bluetooth stack implementation, key conversion and remote control commands presented to the user with a simple GUI.

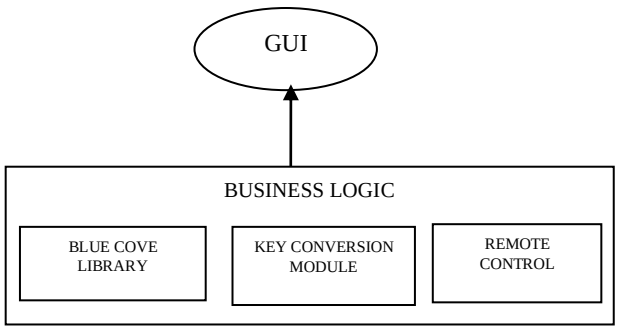

Fig. 1. Server Architecture

B. Client Application

The client application is launched by the user on the Android device. To remotely control the desired pc, the application searches Bluetooth instances. In terms of architecture –it has a user interface that interacts with the business logic layer which covers a small set of different modules.

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 11, November 2016

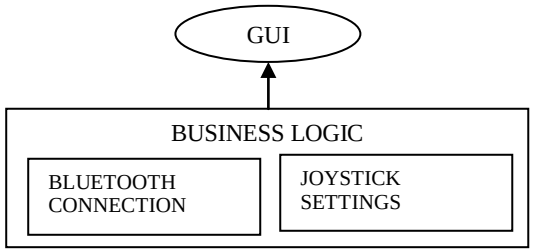

Fig. 2. Client Architecture

C. Client- Server Interaction

During remote communication, the Android device sends Echo requests to find a controllable computer. Once the desktop with the server receives the echo request, it sends an echo reply. Thus the client and server interact via different control commands when sending different keyboard events.

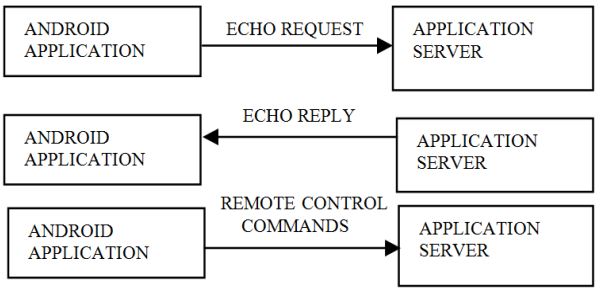

Fig 3 Client-Server Interaction

III. DESIGN AND IMPLEMENTATION

A. Protocol

The protocol used for client-server communication describes three types of events:

- Echo Request
- Echo Reply
- Keyboard events

Implementing the protocol package results in generating and parsing the messages. Advantage of using this protocol is the smaller size of messages results in faster parsing. This protocol is very easy and convenient to implement as compared to other complex methods. In case if the message is not delivered, it will be rejected. However the message isn't authenticated (which isn't a requirement in the application and therefore can be ignored).

B. Client Application

The user interface of the application is very easy and flexible, making the Android device an excellent touchpad with keyboard input ability. The multi-touch capabilities of the device are also utilized by the virtual joystick. The client application is distributed through a standard Android application package file (APK). The minimum Android version required is 4.1. (API level 16). Android devices with touch screen are mandatory to install the client application.

• Bluetooth Connectivity

A perquisite for successful client-server communication is the pairing of both the devices prior to remote control operations. Because the server is found by the client through the bluetooth discovery process and the echo request/reply part of the protocol, the client application sends only events (messages) through Bluetooth. A error handling mechanism is used to notify the client when the connection is lost. It is provided as a a high-level API[1] which gives information about the availability of the hardware, the list of paired devices and interface for radio frequency communication (RFCOMM).

• Client Interface

The client is provided with the facility to design his/her own keyboard. The application also provides different features such as button sizes, colors, shapes along with the flexibility to place them anywhere the clients wish to according to their comfort and need of the application. The key control activity runs in full screen mode and also keeps the screen always on for excellent user experience. Based on the current button state (button that is pressed) a message with the corresponding key code is constructed and sent to the remote device.

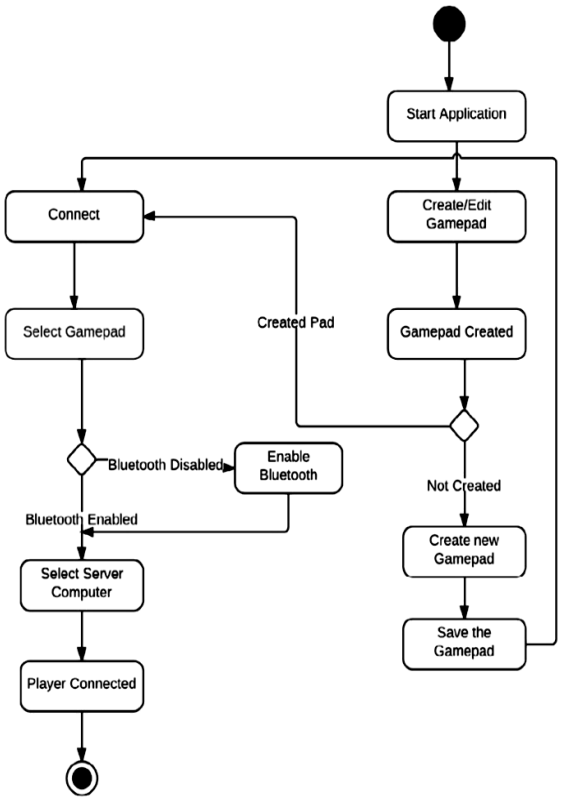

Fig 4 Client Implementation

The application has five screens – Initial application screen, creating a gamepad screen, control buttons settings screen, help screen and about application screen.

The Initial screen provides the client with various options- selecting an already created joystick, creating a new

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 11, November 2016

joystick, connecting to the server or reading the instructions from the help menu. The create joystick screen exposes the essential functionality of the application. It allows the user to customize the joystick according to the need of operation to be performed. The settings also allow the user to choose the size, shape and color of the buttons.

• Permissions Required
In order to communicate with the remote device, the application uses permissions for Bluetooth/Bluetooth admin for Bluetooth connection handling.

C. Server Application
The Server application is intended to run on the controllable desktop to receive client's remote commands via bluetooth. The server application is distributed as an executable file. Implementation details of the main modules of the server are enlisted bellow:

• Bluetooth Connectivity
Java Platform, Standard Edition doesn't provide an Application Programming Interface for controlling a Bluetooth device. Therefore for the client-server Bluetooth communication implementation, Bluecove[2]- a open-source library is used. It is an implementation of Java Specification Request 82 (JSR-82). Implementation steps are as follows:

First the Bluetooth device is made discoverable. The DiscoveryAgent.GIAC (General/Unlimited Inquiry Access Code) constant means that all remote devices (i.e. all the clients) will be able to find the server device.

Then createRFCOMMConnection() method is called to create a RFCOMM[1] connection notifier for the server, with the given UUID and name. Both parameters will be used by the client to find the server.

• Key-code conversion
There is no algorithm for converting between the two sets of key codes defined by the Android platform to the key codes found in the Java SE platform. Therefore to avail the communication between the Android device and Java server, a key translation module is created. The key translation module uses a table with key-value pairs - the keys are the Android key codes and the values are the Java SE key code equivalent.

• User Interface
The remote control in Java SE is implemented using the Robot API[3] . This class is used to generate native system input events for the purposes of test automation, self-running demos, and other applications where control of the mouse and keyboard is needed. The Server class provides binding between the user interface and the Bluetooth discoverable threads. The server begins to listen for Bluetooth connections once it's started. The server user interface is a simple single screen with graphical controls, and also depicts the status of the messages sent and received.

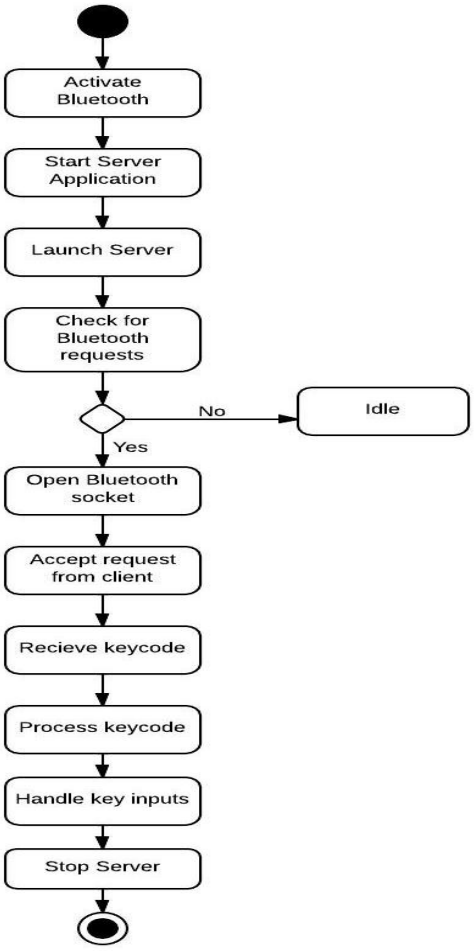

Fig.5. Server Implementation

IV. GRAPHICAL USER INTERFACE

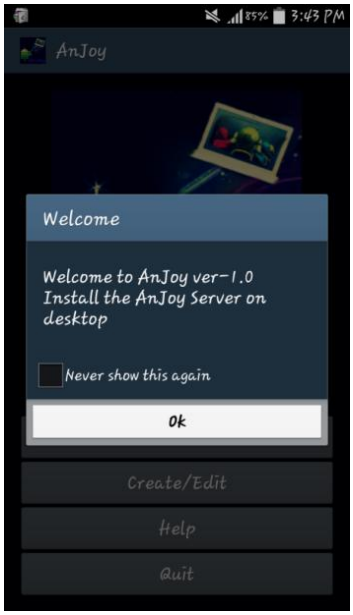

Fig. 6

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 11, November 2016

Fig. 6 shows the Initial dialog box once the application is launched.

Fig. 7

Fig. 7 shows the Initial screen of the android application once it's launched.

Fig. 8

Fig. 8 shows the various options available for customizing your keyboard be selecting different shapes, colors, sizes and keys.

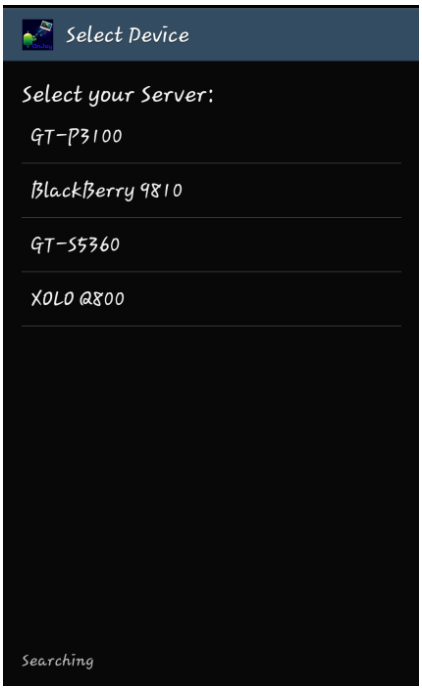

Fig. 9

Fig. 9 shows the screen for finding the server i.e the remote device that needs to be controlled.

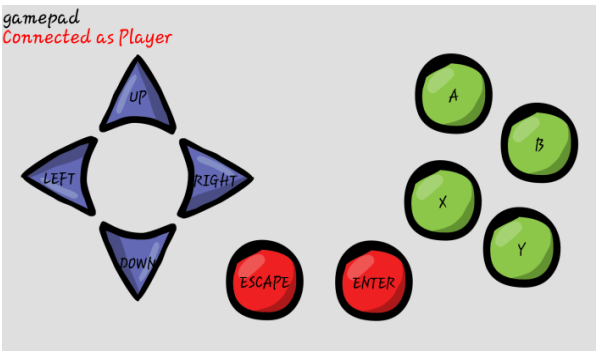

Fig. 10

Fig. 10 shows a sample keyboard created using different keys according to the application of remote control.

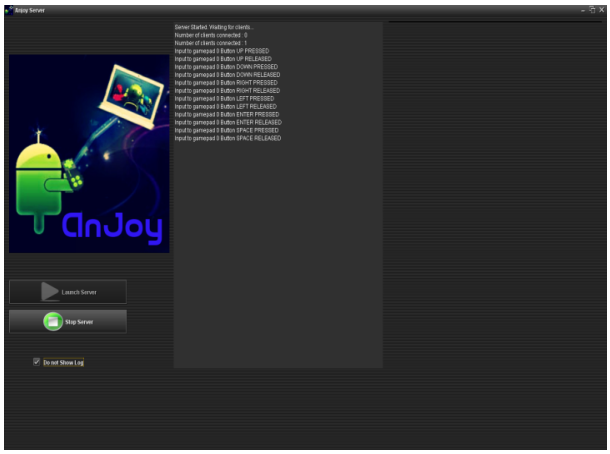

Fig. 11

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 11, November 2016

Fig. 11 shows the single Sever screen that displays the list of key events and is installed on the remote desktop.

V. CONCLUSION

In conclusion the developed application serves as a bridge to connect the android device and the desktop for remote control. The application aims at providing flexibility and intuitive experience to the users whose actions are seen on the remote desktop with minimal delay. Customization of the keyboard is the greatest advantage of the proposed solution and thus it finds application in various areas. Bluetooth implementation is stable and compatible with all major Bluetooth stacks. Multiple users can get connected to the server at the same time.

REFERENCES

[1] Microsoft (n.d), Bluetooth RFCOMM-API. Retrieved October 15, 2016, from https://msdn.microsoft.com/en-us/windows/uwp/devices-sensors/send-or-receive-files-with-rfcomm
[2] Intel Research. (n.d.). Bluecove - Java library for Bluetooth. Retrieved October 15, 2016, from https://code.google.com/p/bluecove/; http://bluecove.org/
[3] Oracle (Sun Microsystems). (n.d.). JavaDoc - Robot. Retrieved October 15, 2016, from http://docs.oracle.com/javase/6/docs/api/java/awt/Robot.html