ASEE's 123rd Annual • Conference & Exposition • New Orleans, LA • June 26-29, 2016

New Orleans JAZZED about Engineering Education

Paper ID #15901

Android-Based Remote Robot Control System

Dr. Yujian Fu P.E., Alabama A&M University

Dr. Yujian Fu is an associate professor of computer science department at Alabama A&M University. Her research interests fall in formal verification of cyber physical systems, behavioral analysis of mobile security, software architecture and design analysis of safety-critical and mission-critical systems. Her projects are supported by NSF, Air Force and DoD. She have several publications regarding to the research and educational projects.

Mr. Md Hossain Shuvo, Alabama A&M University

Graduate Student, Dept. of Computer Science Alabama A&M University

Android Based Remote Robot Control System

Abstract

In this paper, we present an automatic robot control system in Android platform. This research work is aims at designing and developing a robust, reliable, and correct software intensive system for pursuing top level, incomparable and preeminent research objectives by consolidating several techniques and methodologies to make our system more responsive, user friendly and more communicative with the robot. To fulfill the project requirements and reach research objectives, the UML diagrams are simply adopted for the structural and behavioral modeling. As a de facto standard visual representation of object oriented software system, UML provides a seamless connection between the system implementation and design model in both robotics and Android platform. Therefore, it is straightforward to show our implementation has a consistent relation to the design model. Other than this, here we are destined to achieve the high degree of portability by using this mobile platform which can be carried in any Android supported device. Moreover, tracking the robot is a big issue as well as a challenge for our System because of its varying accuracy and robot's frequent movement on the ground and Android is highly capable of detecting the pin point location by finding the absolute position of objects as it supports very strong GPS positioning API. To ensure the above system properties in the robustness and reliability, a systematic specification based unit testing methodology was adopted and implemented on the Android controlled robotics system. The assertions are defined on the structural and behavioral model of UML diagram for the purpose of validating the prospective system. Fundamentally, finding out the possible crash points, potential execution fault, conventional mistakes in robotic software development and efficacious solutions to those problems are our broader research objectives from this project.

Keywords

Robotics, Android platform, UML, software testing, remote control.

Introduction

The growing trend of mobile and pervasive computing towards the development of several embedded system has brought about an evolutionary, dramatic, spectacular and unprecedented change in the field of technology specially in the mobile and robotics field which has provided the opportunity to lead a commodious life for the people of all categories as well as pioneered a door for the researcher to jump into an endless research arena. Several computing platforms have been evolved so far, several versatile devices are now at the shore, several research fields have been opened but there is still lack of ample research minds in some specific area. Meanwhile, applying well-designed projects in the new emerging areas in mobile computing to the software engineering and object oriented programming and design classroom can dramatically motivate students and improve learning outcome toward the conceptual learning and context understanding.

Mobile platform especially Android platform is such a novel system unveiled in 2007 by Google[12] which is still lack of research to reveal its capability to control and manipulate several embedded systems like several robotic systems. Being motivated by the technology trend and the idea of project based learning (PBL), this Android based remote robot control system is mainly aspired to utilize the enormous capability of Android system and to enhance the flexibility of the robot control system remotely and analyze several behavior more efficiently. It is well recognized that LEGO Mindstorm robotic kit provide an ideal on device JAVA firmware compilation platform. In addition, as Java is the supported programming language by Android operating system, there is a seamless synchronization of development platform between LEGO and Android system.

As aforementioned, in this paper, the focus is to realize the degree of potentiality of mobile platform specially the Android platform in handling error and security hazard over the network by operating this robot control system. This is because, robotic platform is complex in terms of its functional architecture and many aspects need to be considered while operating a robot using mobile based applications.

In addition to this, as mobile platform especially Android is a growing trend in the field of software design and development as well as a great attraction in STEM learning to the students, from this work, students can learn several aspects of mobile application development. Though, this is basically on Android platform but it will help them to grab the fundamental concept of mobile application development and make them aware of various development facts. As the Robot Control System works based on both Bluetooth and Wi-Fi communication, they will grab the idea about several communication protocols. Besides, the software testing area of this work will have a fruitful impact on developing their skill for dealing with software bugs which can be applicable not only in mobile platform but also in any software development platform. By this, they will be able to learn several software testing methodologies. Moreover, the use of software design models in this work will aid them to develop software in proper way. Eventually, it will be helpful in senior designing and practical hand on software development.

Pedagogical Significance

The advancing tendency of the mobile systems and the growing interest in the mobile robotics cause the gradual switch from the conventional computing system which eventually attract the students as well as the researchers to look more into this field of embedded system and explore the unexplored facts about it, with the purpose of learning and advancement. With the consideration of this fact as well as with the intent of providing a compact, informative and research based material, this work have been accomplished. The coherent organization, consistent designing approach and advanced development procedure have made this paper strongly favorable for the deployment in educational aspect and in STEM learning. The use of standard procedure and methods for the design and development of the system, will assist the students to grab the basic exploitation of the overall architecture. By having the knowledge of implementation on this particular mobile platform will enable them to employ it in diverse mobile development platform. Moreover, by grabbing this sequential and exhaustive methods will enable them to detect the possible phenomenon toward the failure of application and make them adapt to build applications in mobile platform by avoiding those severe possibilities. Furthermore, the comprehensive software testing methodologies along with the implementation techniques using various testing platform will serve them the ideology of assuring the quality of software both from the functional and nonfunctional perspectives. Other than this, the implementation of each part of the overall system can be used as an effective material for classroom teaching for diverse graduate as well as undergraduate courses that will enhance their capability for applying the acquired knowledge in high level practical development. Therefore, the entire implementation of the system bear a dandy signification in the field of education.

Related works

A couple of researches have been done in robot control system in diverse mobile platform as well robotic platform and these growing researches on mobile platform indicate its efficiency in controlling robots. Some positive and convenient features of using mobile platform for controlling robot including mobility, cost effectiveness and feasibility of using different mode of communication have been revealed from the existing researches .Researchers tried to control robot remotely as much efficiently as possible. Some works tried to use third party hardware[2, 7, 8] including microcontroller along with the Android mobile device to control the robot. Some works also tried to add new feature of controlling robot using mobile device. Gourav and Prasad[7] tried to control the robot by speech, passed by the Android supported device. Besides in some works, researchers focuse on specific mode of remote communication. In some works [2, 6, 9], Ritika and Narender, Mrumal and Javed, et.al and Abinaya and Jayakumar emphasis on Bluetooth communication for communicating with robot remotely. Also in some works[5,7] of robot control system emphasis on specific programming paradigm to make the remote robot control system effective. Some works[3, 13] shows the cost effectiveness of using Android platform for controlling robot for various purposes.

Some implementations have been done also in Lego Mindstorm robotic kit. Andre and Nogueira[10] implemented their robot control system using Android device on Lego Mindstrom NXT. They used Bluetooth communication to communicate with the robot. Stephan, Ruben, Simon and Albert[14] also implemented their robot control system on Lego Mindstorm NXT.

In addition to this, the extra ordinary capability of this (Mindstorm) robotic kit has made it a perfect platform for students' hands on robotics. With this consideration, a lot of works[23-27] have been done by focusing on the educational aspects of this robotic kit to make student adapt with robotics.

All these works of implementing robot control system on mobile platform, demonstrate the capability of mobile platform specially the Android platform which bear the testimony that mobile platform can be made efficient in controlling robot.

Preliminaries

i. UML

Unified Modeling Language[28] widely known as UML is a software engineering tool used for modeling software systems. Fundamentally it is used as a tool for analyzing, designing and implementing software intensive systems. UML provides a visual representation of the system which reflects the standard and interactive organization or system's elements. From the beginning till now, there are several versions of UML have been evolved and UML 2.0 is used for the modeling of our system. UML offers two types of system modeling, one is structural or static modeling which require the presence of all of the elements those required for the system and another one is behavior or dynamic modeling which is basically concerned with the functionalities or actions of the system which represents the action sequences of each element of the system. Class diagram is used for static modeling of the system whereas use-case diagram, state chart diagram are used for dynamic modeling of the system. Above all, it renders a rich graphical notations for the efficient, comprehensive and interactive depiction of the system.

ii. Lego Mindstorm EV3

EV3, which is followed by its two descendants RCX and NXT, is a robotic kit developed by

Lego Mindstorm. It is purposed to provide an education and research material to the learners and researcher in the field of robotics. Moreover, it has great impact in STEM learning and by adopting it, students are building their background strong in robotics. Apart from this, it provides vast freedom of using programming languages because of its diversity of supporting several programming languages including Python, Perl, Lua, Java, and Objective C. The robot used in this system, is a three wheeled ground robot with an ultrasonic sensor tied in front of it as shown in Fig 1. Its compactness, robustness, user friendliness, the synchronization (both Android and EV3) in terms of programming paradigm and necessarily its cost effectiveness have made it the ideal robotic platform for the implementation of this project.

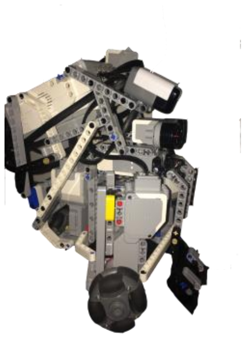


Fig 1: Structure of the robot

iii. Android platform

Android is an innovative as well as a noble and open development platform launched in 2007 by Google, based on the Linux carnal. Besides enhancing the flexibility in using mobile devices and promoting user experience, it has facilitates the application developer to develop all-purpose including intelligent mobile applications. Moreover, it is capable of controlling various embedded systems by its powerful features. For the implementation of the overall project Android OS V4.4.2 (KitKat) is used. The multitasking capability, activeness in handling error, powerful API's and its open source development platforms have made to choose Android among all other mobile platforms.

iv. LeJOS

The entire programming for the implementation is done in Java. Since the system based on Android, hence the programming API based on Java for Android is adopted. Likewise, as the system is for controlling an embedded system which is operated on Java based virtual machine, LeJOS EV3 API for controlling the EV3 robot is adopted with the integration of Android. LeJOS is a Java based virtual machine which was embedded with Lego EV3 in 2013. LeJOS EV3 is an API for manipulating LeJOS firmware which facilitates to use object oriented Java programming language and that eventually facilitates to use all the powerful features of pure Java.

System Architecture

Major development architecture for Robot Control System is divided into three parts, one is *System Designing* which is concerned with the representation of the overall system using modeling language, another one is *System Development* which is concerned about the implementation of the system and the other one is System Testing which is concerned about the functional and underlying quality assurance and activeness of the system. Since the system is aimed at controlling an embedded system which is Lego Mindstrom EV3, hence for avoiding synchronization problem with Android platform in terms of programming, LeJOS EV3 API is used. It facilitates to program the whole system using Java without the integration of any other programming environment. Immediately after the implementation of the system, it is tested for its functional validity and programming correctness.

As aforementioned and as mapped in the system architecture, each component of the overall lifecycle of the system implementation can be applied in classroom teaching for CS curriculum for both undergrad and graduate courses. Constituents of each implementation stage of system design, development and testing can also be used as examples in labs for respective courses. Fundamentally the entire architecture shows how the knowledge from this system can be deployed from the beginner's level in programming to the advanced development in CS curriculum.

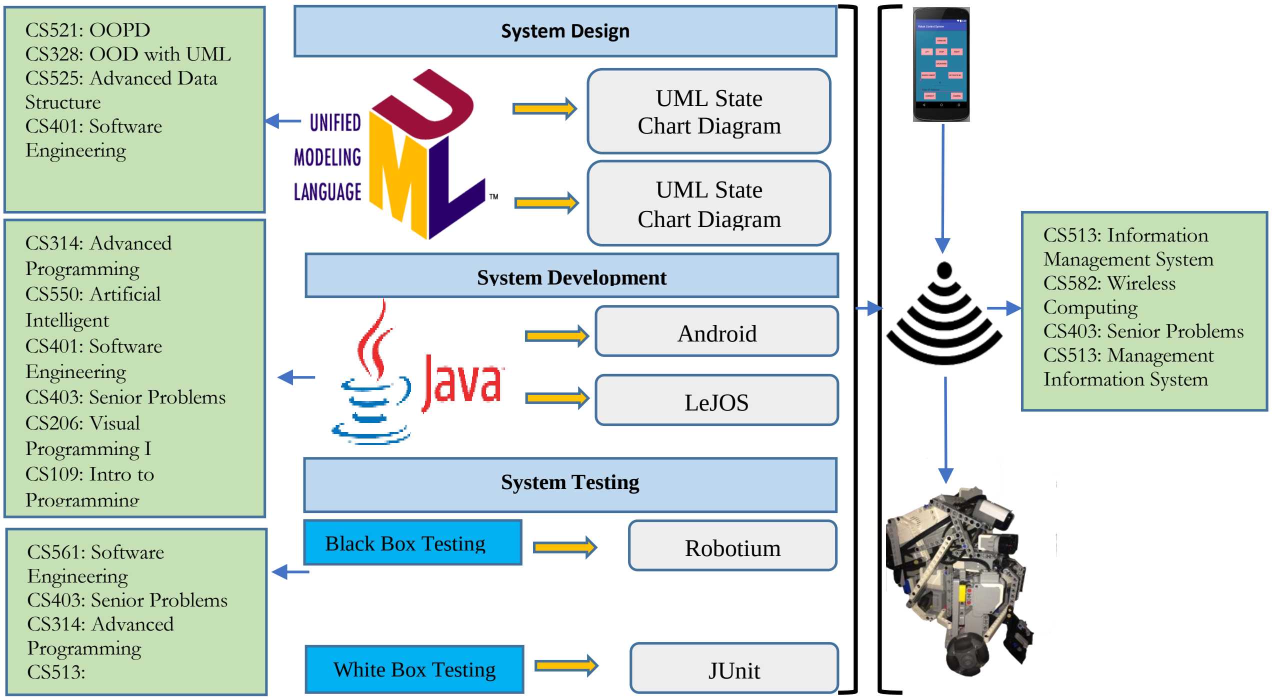

CS521: OOPD
CS328: OOD with UML
CS525: Advanced Data Structure
CS401: Software Engineering
System Design
UNIFIED MODELING LANGUAGE
UML State Chart Diagram
UML State Chart Diagram
CS314: Advanced Programming
CS550: Artificial Intelligent
CS401: Software Engineering
CS403: Senior Problems
CS206: Visual Programming I
CS109: Intro to Programming
System Development
Java
Android
LeJOS
System Testing
CS561: Software Engineering
CS403: Senior Problems
CS314: Advanced Programming
CS513:
Black Box Testing
Robotium
White Box Testing
JUnit
CS513: Information Management System
CS582: Wireless Computing
CS403: Senior Problems
CS513: Management Information System

Fig 2: System Architecture of Robot Control System

System Development

The overall system has been implemented throughout the standard software development procedure to ensure the validity as well as the quality of the system. The implementation of the system is done in a consecutive steps which fundamentally focus on the software system development cycle from requirement, to design, implementation and testing. As a class project, system requirement is simplified and documented at the beginning. We started with system design in the following.

a. System Design

UML provides a comprehensive insight about the overall system. For the modeling of the system using UML diagram, the static or structural and the dynamic or behavioral view have been considered so that the system can be represented graphically from both point of view using UML class diagram and UML state diagram consecutively.

i. UML class diagram

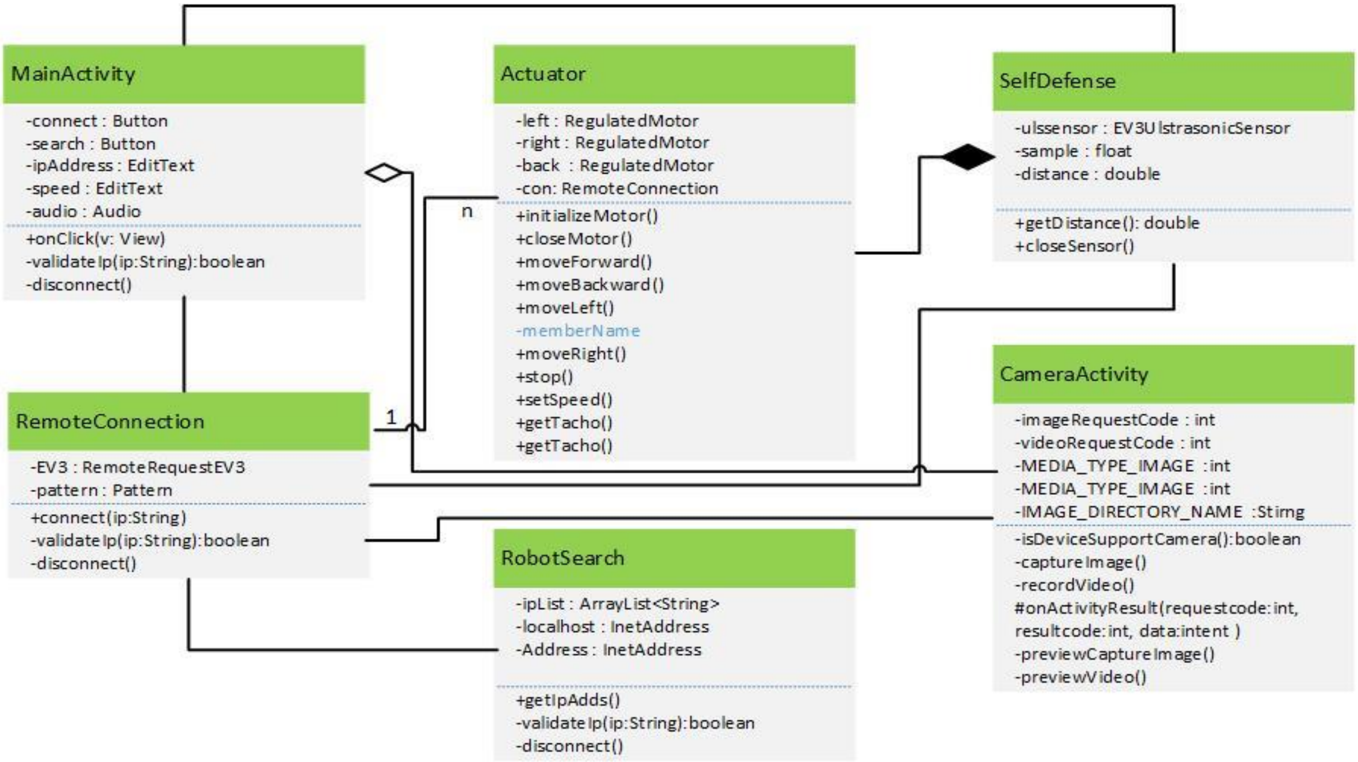


Fig 3: UML class diagram for the system

UML class diagram (Fig. 3) shows the static structure of the Robot Control System by presenting the classes along with their properties and methods. It offers a compact view of overall implementation of the system. Moreover it provides the visibility and scope of each property of the class. In addition to this, it exhibits the dependency of one class on another by demonstrating the coherent relationship among the classes.

In Fig. 3, *RemoteConnection, SelfDefence, and RobotSearch* classes are associated with the *MainActivity* class. Besides, *MainActivity* is aggregated with the *CameraActivity* class and exposes an aggregation relationship between those two classes. The *SelfDefence* class is composed with the Actuator class because of its entire action dependency on that class. This class diagram basically represents the dependencies and the relationship among the classes build for Robot Control System.

ii. UML State Diagram

The dynamic behavior of the system is depicted using UML state diagram where the change of the state based on the triggered actions by the robot controller is represented. Unlike the static diagram or the class diagram, this state diagram reflects the execution flow from one state to another depending on the user action or defined constraints. For the robot control system, it represents the dynamic nature of each states starting from the creation of that states to the termination or transition to another state. Eventually, it represents the lifetime of each states during the execution of the system. State diagram for Robot controller system is shown in Fig. 4

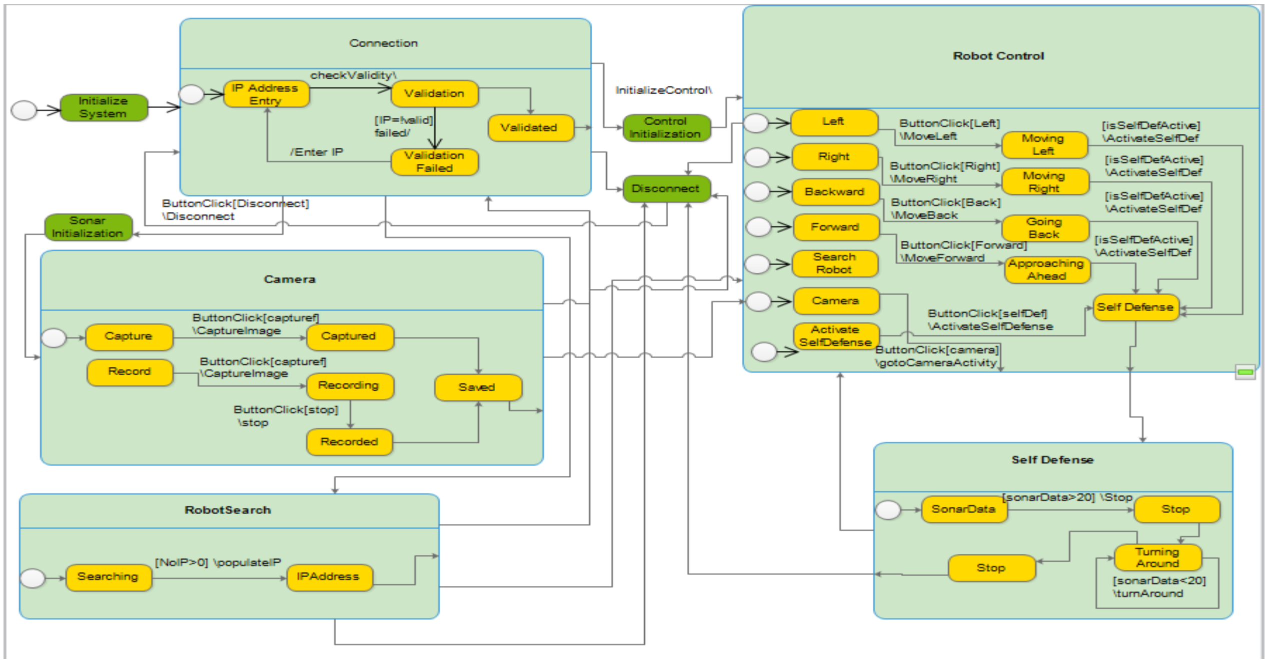


Fig 4: State chart diagram for Robot Controller System

System Development

As aforementioned, as the system is in Android, the system's functionality is developed entirely depending on Java. As the system is not solely bound to an Android device rather it has to control robot, for the synchronization between the devices. Besides Java, an API named LEJOS that supports Java has been adopted. Furthermore, the programming of UI is not is not based on Java in Android platform like the conventional Java application rather the UI is programmed by XML. The system has been developed by programming it in a manner so that the complexity of the code can be reduced and attempted to be objective as much as possible. Moreover the possible error avoidance measures have been taken for the proper control of the robot. For the development of the environment, the official Android development IDE, Android Studio has been used. Besides, since the application is wireless and network packet are transacted while the application executes, hence, during and after the development, the system is analyzed using a robust packet analyzer called *WireShark.*

i. Programming the UI

User Interface is the front end of user friendly software, and plays a significant role for the operation of any system properly. For this, an UI has been programmed in Android Studio using XML. The interface has multiple windows for operating multiple operations related to the robot control. The overall interface is consisted of three activities—**Control, Search and Camera** as shown in Fig 5. The control panel is adorned with several UI components including Buttons, SeekBar, EditText and Label for the convenience of operating the robot.For moving the robot around it has four buttons left, right, backward and forward. For speeding up or down and to show the current speed, a SeekBar and an EditText are used consecutively. For connecting the robot either using Wi-Fi or Bluetooth there is another EditText for writing the IP address shown in the Fig 5a. As the minimum and the maximum running value for the robot is 0 and 720rpm respectively, the max and min values of SeekBar has been defined earlier so that no speed less than 0 and greater than 720 can be inputted in the system to avoid malfunctioning.

For the Camera functionality, it has two Buttons one is used for capturing videos and one is used of images and the panel preview shows the video and the images after it has been captured as shown in Fig 5b.

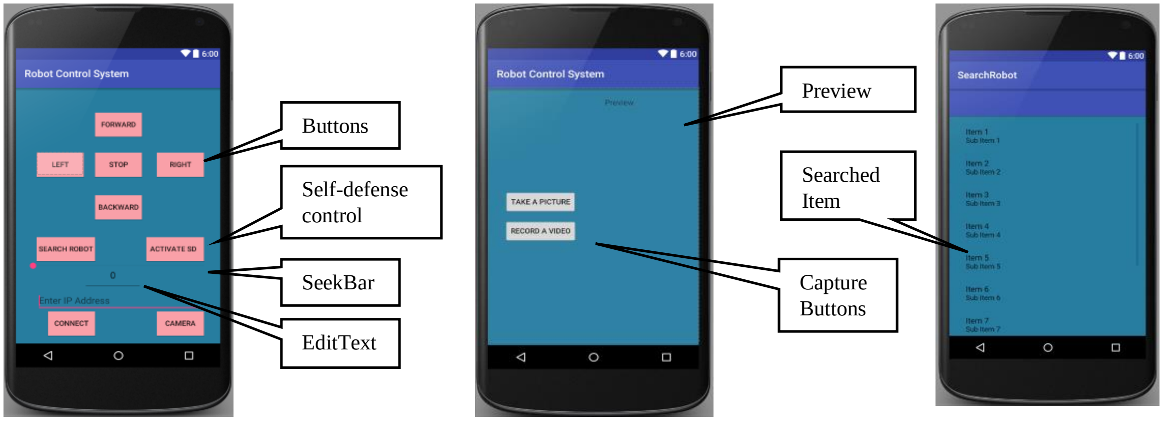


Fig 5a: Layout of the Control

Fig 5b: Layout of the camera functionality

Fig 5c: Layout of the search functionality

Fig 5: Layout of the entire application

Finally, for searching the IP address to connect with the system there is another search panel (Fig 5c) that search for the available IP address in a certain range.

ii. Embedding the Functionalities

For the robot control system, we have embedded various functionalities. The functionalities we have are-

- Connecting the robot with the system using its IP address
- Moving the robot around essentially in left, right, backward and forward direction
- Adjusting the speed while moving
- Searching for available IP address
- Capturing photo or videos while it moves
- Activating self-defense capability

The four buttons Left, Right, Backward and Forward are used for moving the robot around. To move the robot, every button is assigned with an action listener and when they are pressed the command is send to the robot which has also been implemented using LEJOS API. Before moving the robot, the actuators of the robot needs to be initialized. As our robot has three motors- A, B, C which are marked as left, right and back motors and all three motors needs to be initialized before moving.

```
left = con.ev3.createRegulatedMotor("A", 'L');
right =con.ev3.createRegulatedMotor("B", 'L');
back = con.ev3.createRegulatedMotor("C", 'L');
```

After that, the robot is moved in different directions as follows

```
public void moveForward()
{     left.forward();
    right.forward();
}
public void moveBackward()
{     left.backward();
    right.backward();}
```

```
public void moveLeft()
{     left.backward();
    right.forward();}
public void moveRight()
{     left.forward();
    right.backward();}
```

Table 1: Code snippet for moving robot around

iii. Capturing the Photos and Videos

While the robot moves, the system is able to capture photo or record videos of the surrounding environment. The photos or the videos is visible on the preview panel (Fig 5c) .The photos and the videos is immediately saved to the robot's SD card and can be fetched for further use. The code snippet for this module is as follow:

```
private void captureImage() {
    Intent intent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
    fileUri = getOutputMediaFileUri(MEDIA_TYPE_IMAGE);
    intent.putExtra(MediaStore.EXTRA_OUTPUT, fileUri);
    startActivityForResult(intent, CAMERA_CAPTURE_IMAGE_REQUEST_CODE);}
private void recordVideo() {
    Intent intent = new Intent(MediaStore.ACTION_VIDEO_CAPTURE);
    fileUri = getOutputMediaFileUri(MEDIA_TYPE_VIDEO);
    intent.putExtra(MediaStore.EXTRA_VIDEO_QUALITY, 1);
    intent.putExtra(MediaStore.EXTRA_OUTPUT, fileUri);
    startActivityForResult(intent, CAMERA_CAPTURE_VIDEO_REQUEST_CODE);}
```

Table 4: Code snippet for camera module

iv. Self-Defense Capability

The robot control system has the capability to activate or deactivate the self-defense capability. This functionality adds a semiautonomous feature to the robot triggered by the Android device which demonstrates the integration capability of several functionalities in one activity of Android device. To implement this, the Sonar sensor has been used that calculates the distance from any possible collision in the way of its movement only when this feature is activated and force stop the robot moving further if any obstacle is detected. At the same time, the control is completely transferred from the outsider to the robot and the robot makes a turn to right or left to make a clear path.

The implemented functionalities based on the powerful features provided by the Android platform made the Robot Control System, systematically stable.

System Testing

For the testing of the system, several software testing methodologies have been adopted[15]. The system has been passed through a series of testing for the validation of several components and functionalities of the system. Successive steps have been followed for the testing of Robot Control System. The fundamental architecture for the overall testing procedure for Robot Control System is depicted in Fig 8.

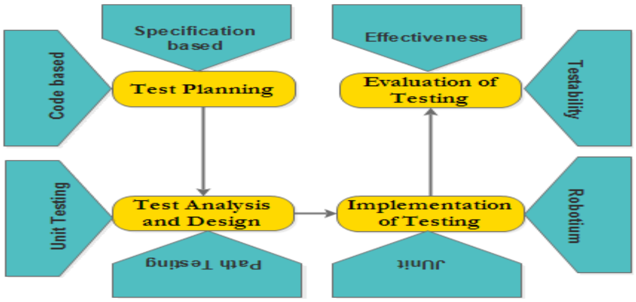


Fig 8: System testing procedure

Therefore, the entire test procedure is carried out based on the planning, designing, implementation and evaluation.

i. Test Case Generation

An apprize planning for conducting the testing have been demonstrated for Robot Control System. The possible risk of the system, the method of testing, the entry and exit criteria have been determined before the implementation of the testing.

For this system, we have attributed several risks those are the reasons for the vulnerability for the system. The risks have been categorized from three major aspects based on this system – possible risk due to the incorrect outsider entry to the system. This is because, as the system is wireless and followed by several network protocol, for noisy environment, the system may have the control to the unexpected host. Moreover due to the multiple requests to the system seeking control over the robot, may throw exceptions because of the requirement of closing the actuators of the robot and prepare for a new connection. Finally, as the robot's nature is stochastic, the inaccurate data flow to the range of sensors and the actuators may result in the collision of the robot with the obstacle.

For the identification of the test cases, both the specification based approach and the code based approach have been followed. For the specification based testing the functionalities of the Robot Control System have been considered. This specification based testing fundamentally refer to the black box testing.

On the other hand the internal coding scheme has been tested. This is followed by the code based approach. The implementation of this test necessarily tests whether the code of Robot Control System are working as expected or not. Basically, it takes into account the internal mechanism of the overall application and this code based testing fundamentally refer to the white box testing.

As the system is based on the Android platform, for the specification based or black box testing, the system is tested using a robust test framework Robotium[16]**.** On the other hand, for code based or white box testing of Robot Control System, another frameworks, JUnit has been adopted which tests the in-depth Activity. In fact, through this method of testing the internal code has been tested which directly get access to the methods and the operators of classes rather than restraining at UI level.

In accordance to the test plan, the testing of Robot Control System is done by adopting two test procedures. The brief description of the test is demonstrated in this section.

ii. Specification Based or Black Box Testing Design and Analysis

For the testing of the system, several functional facts have been taken into consideration that may cause the system to crash. Several specifications have been tested using Robotium. Basically, the overall test those are done using Robotium for the Robot Control System can be attributed as Input/output tests, control test and UI test.

a. Input/ Output test

For this the valid IP format as well as the IP those are connectable or in range is tested when it is attempted to make a connection between the device and the robot. Moreover the testing of the valid speed has been demonstrated.

b. *Control Test*

This testing is crucial in some cases especially for some functionalities of remote controlled application where IP is used and can be considered differently by the system. For this, the blank entry is tested. The submission of blank field for connection may interpret by the system as null which may eventually considered as 0.0.0.0 by the system which is also a valid IP which is similar as 127.0.0.0 in some cases[17, 18, 19]. This creates a possibility for the system to remain in risky state. To avoid this possibility this module has been tested.

c. *UI Test*

To check the functionality of every UI widget, to check it correct operability and response to specific event, UI test has been accomplished using Robotium.

iii. *White Box Testing*

a. *Basis Path testing*

For the implementation of code based testing, basis path testing has been adopted where the testing is based on source code rather than specification. For this testing, the code written for controlling several accessibility to the system depending on the satisfaction of condition have been tested. The flow graph and the directed graph below show the brief view of one of the path testing where the functional capability of the source code is tested.

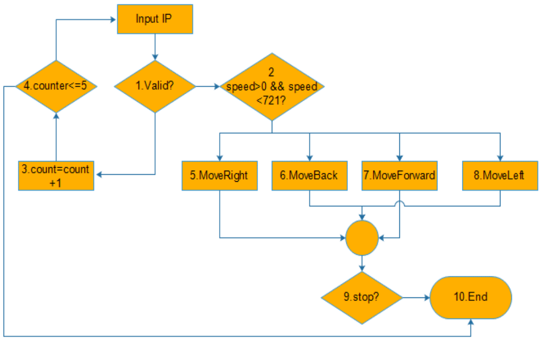


Fig 9b: Flow graph for path testing

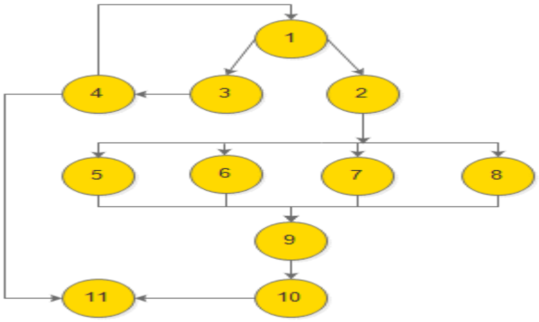
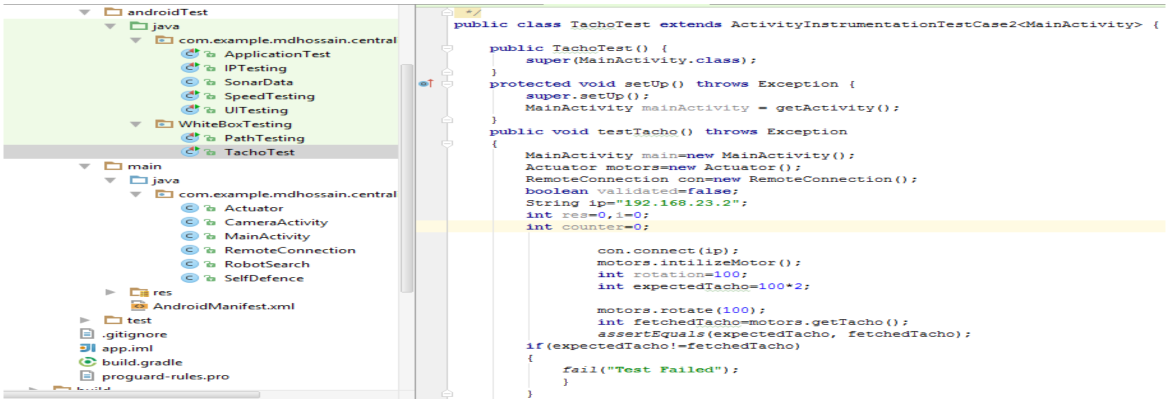


Fig 9c: directed graph for path testing

b. *TachoTest*

This test has been done to test the consistency between the expected moving distance and the actual moving using the EV3's default odometer data. This test purposed to find the drift in the EV3 servo motors. The code snippet is depicted in the following figure.



```
 */
public class TachoTest extends ActivityInstrumentationTestCase2<MainActivity> {

    public TachoTest() {
        super(MainActivity.class);
    }
    protected void setUp() throws Exception {
        super.setUp();
        MainActivity mainActivity = getActivity();
    }
    public void testTacho() throws Exception
    {
        MainActivity main=new MainActivity();
        Actuator motors=new Actuator();
        RemoteConnection con=new RemoteConnection();
        boolean validated=false;
        String ip="192.168.23.2";
        int res=0,i=0;
        int counter=0;

                con.connect(ip);
                motors.intilizeMotor();
                int rotation=100;
                int expectedTacho=100*2;

                motors.rotate(100);
                int fetchedTacho=motors.getTacho();
                assertEquals(expectedTacho, fetchedTacho);
        if(expectedTacho!=fetchedTacho)
        {
            fail("Test Failed");
            }
        }
```

Fig 10: Tacho Test for Robot Control System

All the test tests cases are written using Robotium and JUnit. The main class of Robotium *Solo* is used for testing which offers different methods including *clearEditText, enterText, clickOnButton, wait* and many more which facilitates to write powerful and robust automatic test cases. On the other hand the testing of the internal code has been tested using JUnit. The overall implementation of testing procedure using Robotium and JUnit is depicted in Fig 11.

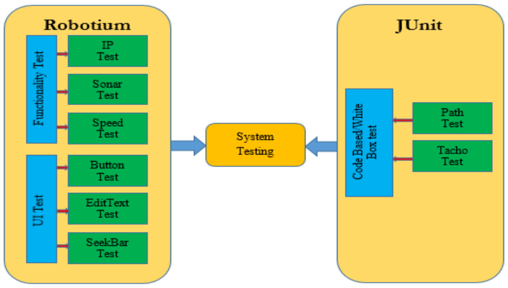
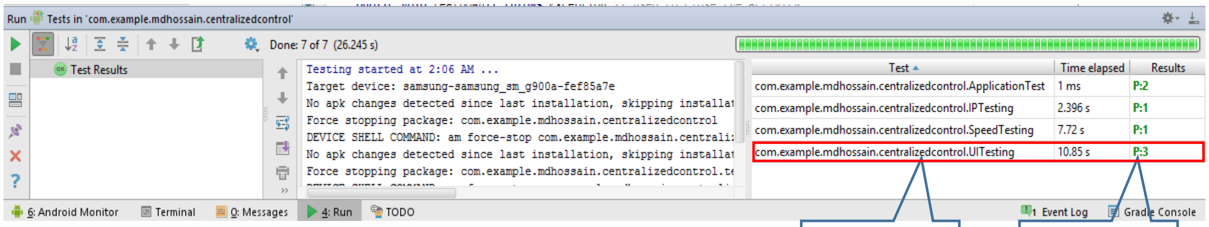


Fig 11: System testing architecture

Result and Discussion

The system is examined by inspecting its various implemented functionalities separately to observe the expected actions of each individual component as well as by its testing result. Even though it is possible to run these type of mobile based applications on emulator but for getting the best result and to have a better interaction between the robot and the system, it is run on the physical device to control the robot. The result is fetched from the two standpoints, one is from the system testing and another one from the run time environment.

The testing is done in several phases which has been demonstrated earlier. The UI test using Robotium has been done to check the functional activeness of each UI widget, response to the event, organization in order and to check the proper assignment of attributes. The passing of test cases those are written to inspect those criteria, stands for the consistent organization and operation of all the UI components. All the UI components are tested with the pragmatic interaction with the robotic system. The test results are as follows

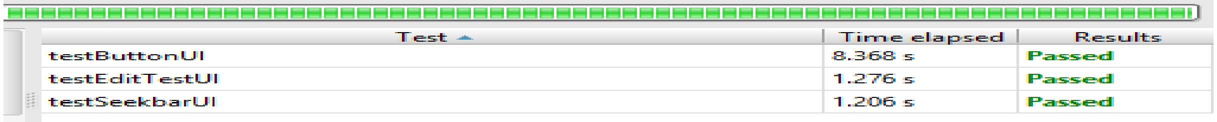


```
Testing started at 2:06 AM ...
Target device: samsung-samsung_sm_g900a-fef85a7e
No apk changes detected since last installation, skipping installat
Force stopping package: com.example.mdhossain.centralizedcontrol
DEVICE SHELL COMMAND: am force-stop com.example.mdhossain.centrali:
No apk changes detected since last installation, skipping installat
Force stopping package: com.example.mdhossain.centralizedcontrol.te
```

Test	Time elapsed	Results
com.example.mdhossain.centralizedcontrol.ApplicationTest	1 ms	P:2
com.example.mdhossain.centralizedcontrol.IPTesting	2.396 s	P:1
com.example.mdhossain.centralizedcontrol.SpeedTesting	7.72 s	P:1
com.example.mdhossain.centralizedcontrol.UITesting	10.85 s	P:3

Fig 12: Test result for UI test

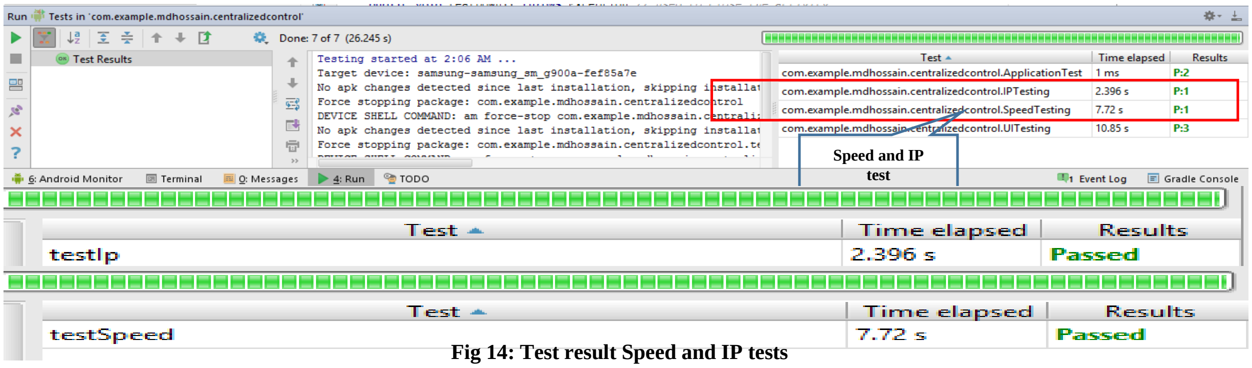
The detailed test result of individual test cases is depicted below



Test	Time elapsed	Results
testButtonUI	8.368 s	Passed
testEditTestUI	1.276 s	Passed
testSeekbarUI	1.206 s	Passed

Fig 13: Detailed UI test result

Other than this, two crucial functionalities including connection and speed of the robot are also tested which demonstrated the correctness of all the specification of the system. The test result for speed an IP is depicted in Fig 14.



Run Tests in 'com.example.mdhossain.centralizedcontrol'

Done: 7 of 7 (26.245 s)

Test Results

```
Testing started at 2:06 AM ...
Target device: samsung-samsung_sm_g900a-fef85a7e
No apk changes detected since last installation, skipping installat
Force stopping package: com.example.mdhossain.centralizedcontrol
DEVICE SHELL COMMAND: am force-stop com.example.mdhossain.centrali:
No apk changes detected since last installation, skipping installat
Force stopping package: com.example.mdhossain.centralizedcontrol.te
```

Test	Time elapsed	Results
com.example.mdhossain.centralizedcontrol.ApplicationTest	1 ms	P:2
com.example.mdhossain.centralizedcontrol.IPTesting	2.396 s	P:1
com.example.mdhossain.centralizedcontrol.SpeedTesting	7.72 s	P:1
com.example.mdhossain.centralizedcontrol.UITesting	10.85 s	P:3

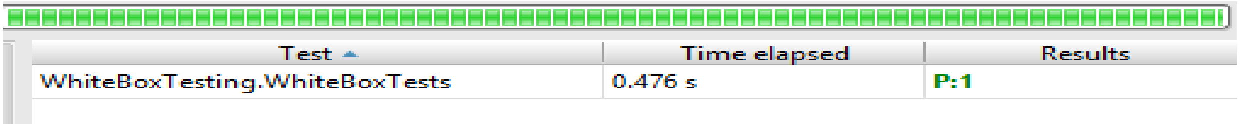
Speed and IP test

Test	Time elapsed	Results
testIp	2.396 s	Passed

Test	Time elapsed	Results
testSpeed	7.72 s	Passed

Fig 14: Test result Speed and IP tests

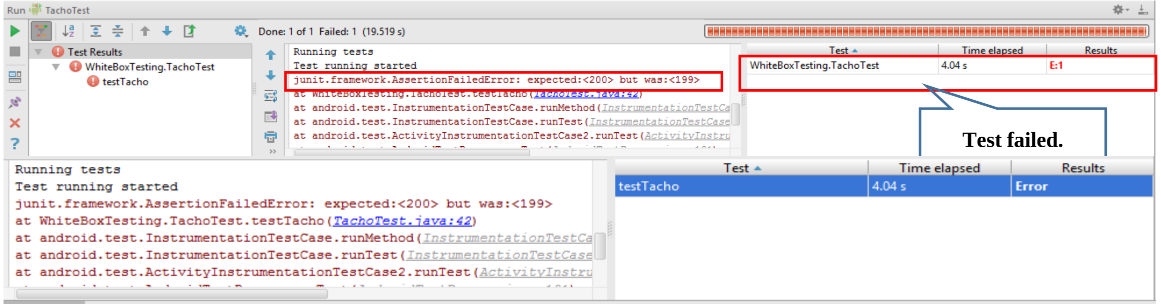
As aforementioned, in addition to this specification based tests, the code based or white box test has been performed to check the validity of the system code. For this, basis path testing has been adopted where the sequence of functionalities depending on the assigned constrained have been adopted directly accessing the source code of the system. The result of this test is as follows



Test	Time elapsed	Results
WhiteBoxTesting.WhiteBoxTests	0.476 s	P:1

Fig 15: Basis path testing

Moreover, to test the inconsistency between the desired moving distance and the actual distance *Tachotest* is done. Practically, it is hard and almost impossible to have the desired and actual distance value exactly accurate because of the drift of motors odometer, but the main purpose to do this test is to justify the difference between those two values. The result of this test (where the set Tacho is 200 but its 199 from the robot's odometer. As it is for two motors so the overall drift for each one is 0.5 rpm) is as follows



Run TachoTest

Done: 1 of 1 Failed: 1 (19.519 s)

Test Results
WhiteBoxTesting.TachoTest
testTacho

```
Running tests
Test running started
junit.framework.AssertionFailedError: expected:<200> but was:<199>
at WhiteBoxTesting.TachoTest.testTacho(TachoTest.java:42)
at android.test.InstrumentationTestCase.runMethod(InstrumentationTestCa
at android.test.InstrumentationTestCase.runTest(InstrumentationTestCase
at android.test.ActivityInstrumentationTestCase2.runTest(ActivityInstru
```

Test	Time elapsed	Results
WhiteBoxTesting.TachoTest	4.04 s	E:1

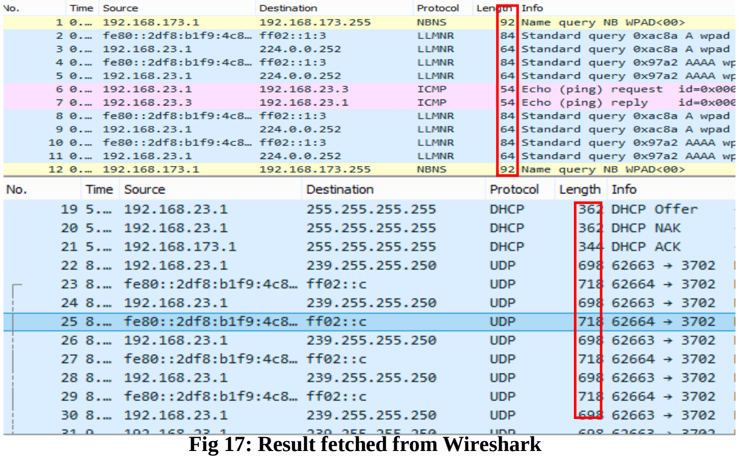
Test failed.

```
Running tests
Test running started
junit.framework.AssertionFailedError: expected:<200> but was:<199>
at WhiteBoxTesting.TachoTest.testTacho(TachoTest.java:42)
at android.test.InstrumentationTestCase.runMethod(InstrumentationTestCa
at android.test.InstrumentationTestCase.runTest(InstrumentationTestCase
at android.test.ActivityInstrumentationTestCase2.runTest(ActivityInstru
```

Test	Time elapsed	Results
testTacho	4.04 s	Error

Fig 16: Tacho testing

It is obvious that the system interacts with the robot as expected but it does not provide the sufficient information for the correctness of implementation. As early stated the, as the system is entirely remote based and communicate with any entity remotely, securing the communication channel is a bigger concern to ensure the durability of the system and to preserve logical coherence. While running, the system's interaction with the network as well as with the robot has been examined using *Wireshark*- a packet analyzer. Since various precautionary measures have been considered while developing this system as stated before, while analyzing using *Wireshark*, the system exhibits some of its strong features including its uniformity in sending and receiving packets for each specific as depicted by the following result fetched from the *Wireshark*.



No.	Time	Source	Destination	Protocol	Length	Info
1	0.…	192.168.173.1	192.168.173.255	NBNS	92	Name query NB WPAD<00>
2	0.…	fe80::2df8:b1f9:4c8…	ff02::1:3	LLMNR	84	Standard query 0xac8a A wpad
3	0.…	192.168.23.1	224.0.0.252	LLMNR	64	Standard query 0xac8a A wpad
4	0.…	fe80::2df8:b1f9:4c8…	ff02::1:3	LLMNR	84	Standard query 0x97a2 AAAA wp
5	0.…	192.168.23.1	224.0.0.252	LLMNR	64	Standard query 0x97a2 AAAA wp
6	0.…	192.168.23.1	192.168.23.3	ICMP	54	Echo (ping) request id=0x000
7	0.…	192.168.23.3	192.168.23.1	ICMP	54	Echo (ping) reply id=0x000
8	0.…	fe80::2df8:b1f9:4c8…	ff02::1:3	LLMNR	84	Standard query 0xac8a A wpad
9	0.…	192.168.23.1	224.0.0.252	LLMNR	64	Standard query 0xac8a A wpad
10	0.…	fe80::2df8:b1f9:4c8…	ff02::1:3	LLMNR	84	Standard query 0x97a2 AAAA wp
11	0.…	192.168.23.1	224.0.0.252	LLMNR	64	Standard query 0x97a2 AAAA wp
12	0.…	192.168.173.1	192.168.173.255	NBNS	92	Name query NB WPAD<00>

No.	Time	Source	Destination	Protocol	Length	Info
19	5.…	192.168.23.1	255.255.255.255	DHCP	362	DHCP Offer
20	5.…	192.168.23.1	255.255.255.255	DHCP	362	DHCP NAK
21	5.…	192.168.173.1	255.255.255.255	DHCP	344	DHCP ACK
22	8.…	192.168.23.1	239.255.255.250	UDP	698	62663 → 3702
23	8.…	fe80::2df8:b1f9:4c8…	ff02::c	UDP	718	62664 → 3702
24	8.…	192.168.23.1	239.255.255.250	UDP	698	62663 → 3702
25	8.…	fe80::2df8:b1f9:4c8…	ff02::c	UDP	718	62664 → 3702
26	8.…	192.168.23.1	239.255.255.250	UDP	698	62663 → 3702
27	8.…	fe80::2df8:b1f9:4c8…	ff02::c	UDP	718	62664 → 3702
28	8.…	192.168.23.1	239.255.255.250	UDP	698	62663 → 3702
29	8.…	fe80::2df8:b1f9:4c8…	ff02::c	UDP	718	62664 → 3702
30	8.…	192.168.23.1	239.255.255.250	UDP	698	62663 → 3702

Fig 17: Result fetched from Wireshark

This result from both testing and the Wireshark, states the strong performance of the overall system in controlling robot and self-protective nature in the network which makes it an ideal Robot Control System.

Conclusion and Future Plan

This work is fundamentally accomplished to demonstrate the capability of mobile platform in controlling embedded systems as well as the potentiality in securing remote communication. Moreover, since the overall task has been accomplished in both mobile and robotic platform, hence it will provide a rigorous, comprehensive and interactive notion to the learners and researchers. Apart from this, though it is implemented in specific mobile and robotic platform but it can be implemented in any of the mobile and robotic platform by adopting the realization that has been implemented here. Besides the adoption of UML models for the designing of system's underlying architecture as well as the behavior will facilitate to design, develop and analyze software systems using standard modeling approach.

Though the system is completely capable of controlling robot but for exploring more and digging into the deeper of the mobile platform, the image processing feature for detecting and analyzing images based on the feature detection using Android built in camera will be implemented. It will demonstrate the strong image processing capability with the consolidation of robotic platform. At this stage of implementation, this system is capable of controlling only EV3 robot in this mobile platform. But to realize the immense capability of mobile platform in the heterogeneous robotic environment will be attempted to be accomplished where the robot control system will be able to control the robots from diverse platforms including NAO humanoid robot, LEGO wheeled robot, IRIS unmanned aerial system etc. This task[21] is in progress in computer platform with the title "Trusted distributed centralized control architecture for heterogeneous robotic system".

Acknowledgement

This project is supported by NSF awards No. 1419295, No. 1225654 and partially supported by Air Force Research Lab under award No. FA87501520106. We would like to thank all reviewers for their valuable inputs.

References

1. M.Selvam, (2014). Smart Phone Based Robotic Control For Surveillance Applications. International Journal of Research in Engineering and Technology IJRET, 03(03), pp. 229-232.
2. Yeole, A. R., Bramhankar, S. M., Wani, M. D., & Mahajan, M. P. (2015). Smart Phone Controlled Robot Using ATMEGA328 Microcontroller.IJIRCCE International Journal of Innovative Research in Computer and Communication Engineering, 03(01), 352-356.
3. Loïc Frund, "Interfacing an Android-based phone-rob ot with Webots", 2012, Master Thesis Project Report, Ecole polytechnique fédérale de Lausanne (EPFL)
4. Christoph, Reinhard, "Android-Based Low-Cost Robot Controller", Practical Robotics Institute Austria (PRIA), 2012

5. Muhammad, Mirza, “WG11 Android based Surveillance Robot Control System Using Socket Programming with Implementation”, 2015, International Journal Of Multidisciplinary Sciences And Engineering, Vol. 6, No. 3, March 2015
6. Ranjit, B.Santhos ,“Android Based Robot Implementation For Pick and Retain of Objects”, 2014, *International Journal of Engineering Trends and Technology (IJETT) – Volume 16 Number 3 – Oct ? 2014*
7. *Gaurab, Prasad, “*Robotic Control using Speech Recognition and Android*”, 2015,* International Journal of Engineering Research and General Science Volume 3, Issue 1, January-February, 2015
ISSN 2091 -2730
8. Naik, Chiranjivi, M. Deshpande, Ravija.R., Rashmi , “Android Controlled Spy-Robot”, International Association of Scientific Innovation and Research (IASIR), ISSN (Print): 2279-0063, PP. 54-57
9. Mrumal., Javed, Aarushi, Reshma, Raunak ,“Robot Control Design Using Android Smartphone”, 2015, Journal Of Business Management And Economics 3 : 2 February (2015) 31 - 33.
10. Andr´e, Nogueira, “Autonomous Mobile R ob ot N avigation using Smartphones”, 2008, Extended abstract
11. bunaya, Jayakumar, “Unmanned Ground Vehicle Remote Controlled Robot for Security Applications”, 2005, International Journal of Trend in Research and Development, Volume 2(4), ISSN 2394-9333, pp.61-63
12. Android operating system, Retrieved November 25, 2015, from https://en.wikipedia.org/wiki/Android_(operating_system)
13. Chinmay, Shuhas, et.al. “Surveillance Robot Using Arduino Microcontroller, Android APIs and the Internet”, 2014, 2014 First International Conference on Systems Informatics, Modelling and Simulation, pp.83-87.
14. Goebel S, Jubeh R, Raesch S-L and Zuendorf A (2011), *"Using the Android Platform to control Robots"*, In Proceedings of 2nd International Conference on Robotics in Education (RiE 2011). Vienna, Austria, September, 2011. , pp. 135-142. INNOC - Austrian Society for Innovative Computer Sciences.
15. Jovanović,Irena“Software Testing Methods and Techniques”Transaction on internet research, 2008, pp. 30-41
16. Robotium-Testing Android UI. Retrieved January 24, 2016, from http://www.methodsandtools.com/tools/robotium.php
17. 0.0.0.0, Retrieved January 23, 2016, from https://en.wikipedia.org/wiki/0.0.0.0
18. What is the Difference Between 127.0.0.1 and 0.0.0.0? Retrieved January 23, 2016, from http://www.howtogeek.com/225487/what-is-the-difference-between-127.0.0.1-and-0.0.0.0/
19. 127.0.0.1 – What Are its Uses and Why is it Important? Retrieved January 23, 2016, from http://www.tech-faq.com/127-0-0-1.html
20. Solo (Robotium :: Solo 5.4.2-SNAPSHOT API),Retrieved January 23, 2016, from https://robotium.googlecode.com/svn/doc/com/robotium/solo/Solo.html
21. Md Hossain Shuvo, “Trusted distributed central architecture for heterogeneous robotic system”, Submitted abstract, SRC 2016, Austin, TX, May 14-22, 2016.
22. William, Tony, Natasha, Chrish, “Physics With Robotics Using LEGO® MINDSTORMS® in High School Education”, 2010 AAAI Spring Symposium Series.
23. Charlotte, Penn, “Using LEGO MINDSTORMS in a Control Systems Lab to impact next generation engineers (Works in Progress)”, 2015, 122 ASEE Annual Conference & Exposition, June 14-15,2015, Seattle,WA.
24. Danial, “Experiences with the LEGO MindstormsTM throughout the Undergraduate Computer Science Curriculum”, 2006, 36th ASEE/IEEE Frontiers in Education Conference, October 28 – 31, 2006, San Diego, CA
25. Paulo, Jose, “Learning Mobile Robotics Using Lego Mindstorms”, *Polytechnic Institute of Bragan¸caSchafer, “*Hands-on Artificial Intelligence Education Using LEGO Mindstorms: Lessons Learned*”,* Department of Computer Science University of Northern Iowa
26. Vallance, Martin, Wiz Schaik, “LEGO Mindstorms™ for informed metrics in virtual worlds”, 2009, HCI 2009 – People and Computers XXIII – Celebrating people and technology
27. Gómez-de-Gabriel, J.M., Mandow, A., Fernández-Lozano, J., García-Cerezo, A.J. (2011) "Using LEGO NXT Mobile Robots With LabVIEW for Undergraduate Courses on Mechatronics," IEEE Transactions on Education, vol.54, no.1, pp.41-47, Feb. 2011.
28. Unified Modeling Language, Retrieved January 27, 2016, from https://en.wikipedia.org/wiki/Unified_Modeling_Language