Course Project Documentation
CS-308 Project

Remote Robot Controller using TCP/IP connection

TEAM : 3
Saif Hasan, 09005003
Chinmay Chauhan, 09005010
Sagar Chordia, 09005013
Hemant Gangolia, 09005015

Table of Contents

1. Introduction

The project aims at controlling a robot remotely through TCP/IP connection. In this project we have used FireBird V as robot, Android phone as means of establishing TCP/IP connection.

We have implemented one application. “Burglary Detection Robot” from this generalized communication framework. Robot is put on surveillance with its camera on. Whenever it detects motion it sends SMS to owner. Owner then can watch video remotely coming from camera of robot.

User can also move robot remotely, he can turn on buzzer on robot remotely and this is constantly aided by video feed.

2. Requirements

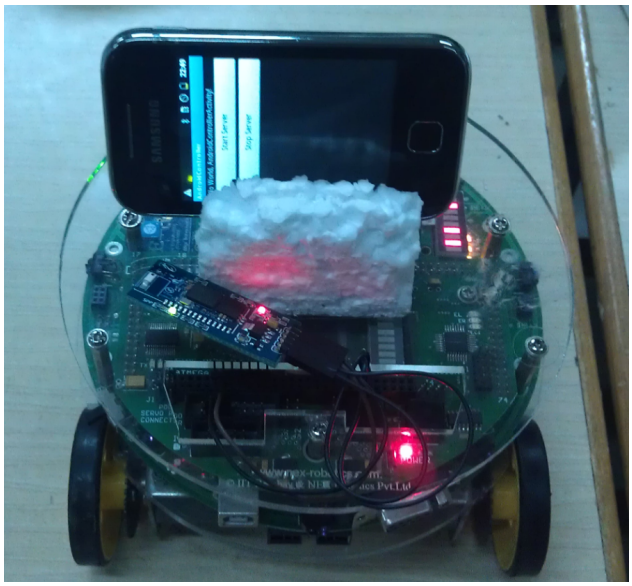
(A) Hardware Requirements

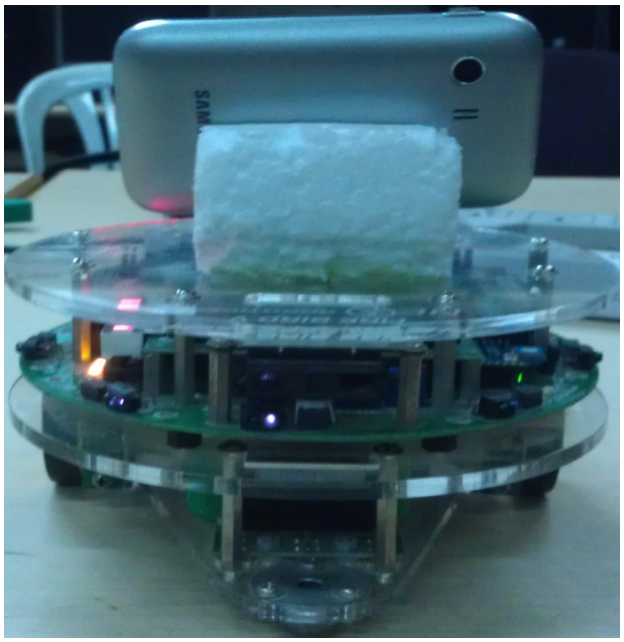
- Robot having following functionalities:
  - Processor to manage various tasks.
  - Capable of moving in all directions
  - Wifi module to establish TCP/IP connection
  - Buzzer to signal alarm
- Android Phone having WiFi to control robot remotely
- Another Android phone having Bluetooth and Camera
- Bluetooth Module

(B) Software Requirements

- AVR Studio 4
- AVR Bootloader
- Eclipse SDK
- Android ADK

3. Robot Assembly

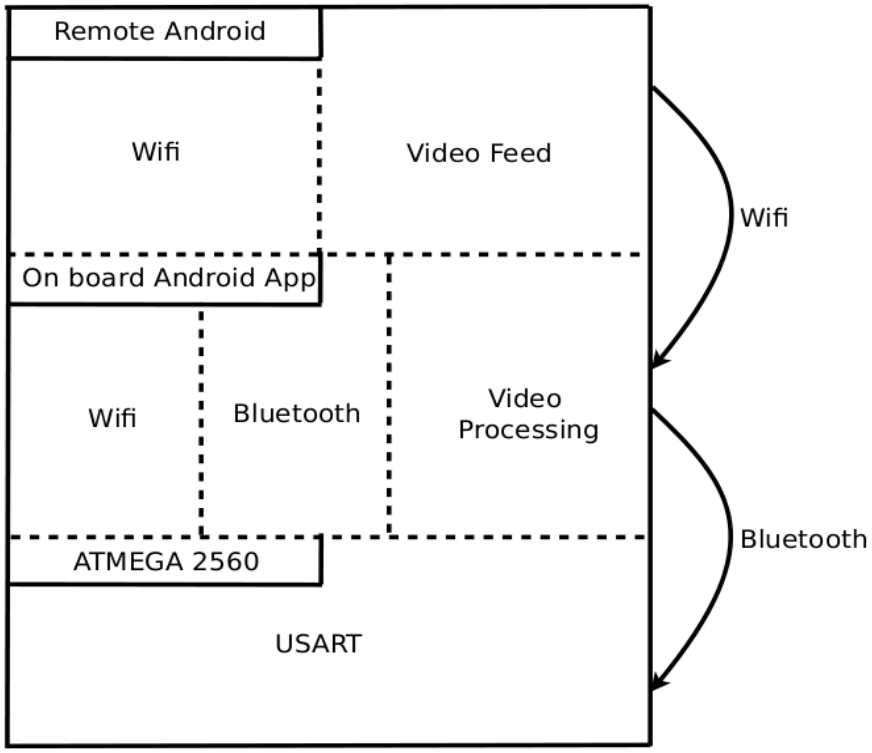




- Robot - Android mounted on FB5 acts as master controller whereas Atmega on FB5 acts as slave.
    - FireBird V
        - supports motions, buzzer functionality and asynchronous communication using Atmega 2560 attached on robot.

    - We have mounted Android phone on FireBirdV.
    - camera on mobile phone is used for video feed
    - wifi module in mobile is used to establish TCP/IP connection
    - Processor on mobile is used to manage all tasks
- Android phone to control this robot remotely.
- Android mobile phone is mounted on this robot.

4. Implementation

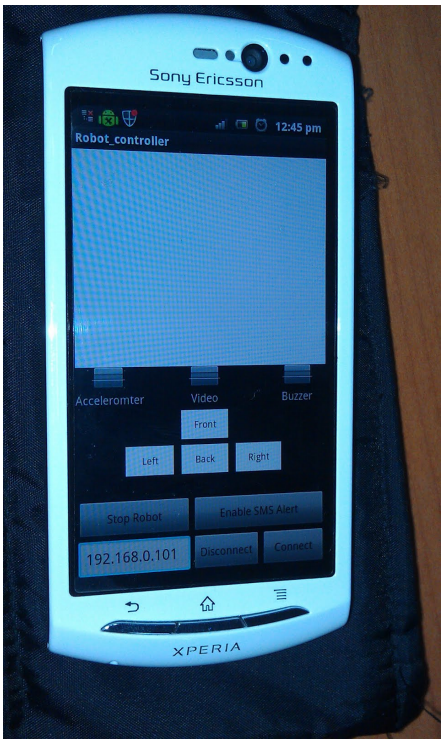


STATE CHART

We basically have 3 modules of functionality which can be described as follows :

1. **Remote Android Application**
2. **On board Android Application**
3. **Atmega 2560**

1. Remote Android Application

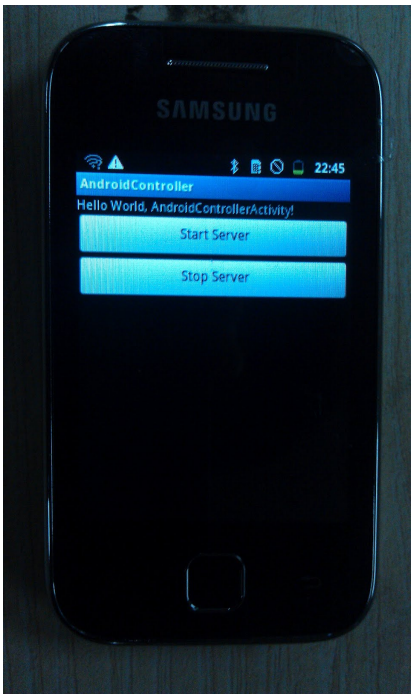


This Android application connects to the server via TCP/IP connection and provides the live video feed from the bot.

This application also has the functionality to provide user the option of receiving a SMS on his mobile phone on detection of motion in the bot's environment. This serves a theft detection application. This is achieved by Image Processing on the Android phone mounted on the bot which sends the control signal to send the SMS.

2. On Board Android Application

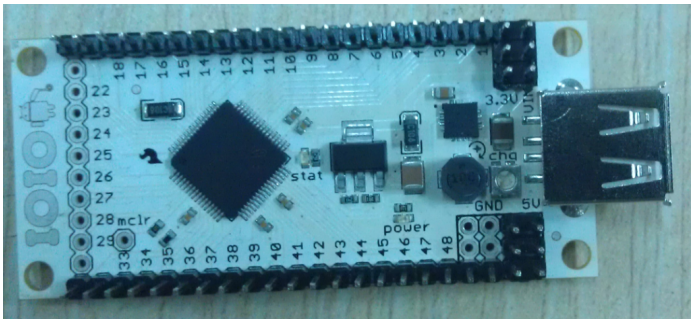
This is also connected to the server through communicates to the remote Android phone. The bot is mounted with a Bluetooth module which serves a communication medium between Firebird and the Wifi and Android device. The Firebird serves a slave by receiving control signals from the Android device. The communication is via USART serial communication.



3. Atmega 2560

This is basically the bot which receives the control signals and moves accordingly. This

functionality has been coded in C and burnt on Firebird V robot.



We initially started with IOIO board from SparkFun for communication between phone and Firebird but had to drop out because the board got shorted on the very day of the demo. This was a much better solution as there wouldn't have been any lag in the communication as this would have provided a direct and faster link with the bot.

5. Algorithm

- We have added reliability over normal UDP protocol for sending of video packets. For this, we have marked each packet with sequence numbers. This will ensure inorder transfer of packets end-to-end.
- We fragment a video packet into chunks before sending it to avoid the overloading the channel and also it avoids the network congestion.
  E.g. A video packet of 100 kb would be divided into 10 small packets of 10 kb each before sending.
- Control signals are sent via TCP protocol ensuring reliable transfer.

6. Testing

- We have extensively tested on Eclipse emulator Logcat for messages transfers.
- We also tested on Wireshark for verifying packets transfer between the Android devices.
- Parallel testing on Firebird done sequentially as the code progressed.
- For the android applications both on the remote side and on the firebird side, we have tested it on the following hardware platforms with respective android OS versions and all of them have been found to run smoothly.
    - Sony Xperia Neo V (Android 2.3.4)
    - HTC Incredible S (Android 2.3.5)
    - Samsung Galaxy Y (Android 2.3.6)

7. Future Work

- The code we have developed is highly modular. We have the following modules
    a. **Communication module** between the **on-board Android** and **Remote Android**. This module is for sending and receiving the video feed (could be any data) via UDP to the remote Android and via TCP/IP for sending control messages, since control messages need to be reliably sent and received.
    b. **Bluetooth communication interface** between the on-board Android and Firebird. We can send and receive data via bluetooth between Firebird and the on-board Android.
    c. A simple **Firebird API** burnt on the Bot for performing various simple functions like controlling motors, toggling buzzer, motion sensing etc.
- Because of the modularity of our code, one could easily work on various aspects that we have touched in our project and easily extend the modules which itself could form a major course project.

Examples of how our project can be further extended:

- **Terrain Exploration :** Since we already have a remote controlling capability along with video feed, we could extend the project by adding more functionality on the firebird side like Terrain Exploration, and while receiving the video feed do some processing on the remote Android processor to see if we have found something interesting during our exploration.
- **Firebird API :** As of now we have a very simple API running on the firebird, but one could add more complex functionalities for exploiting various features of firebird like its sensors, adaptive cruise control, line following capability etc. and this coupled with the video feed we have one could do interesting stuff like image processing etc. on the remote side.
- **Object Finder :** We could add functionality to the android application to allow us to upload an image of an object and then ask the bot to do some sort of generic search (like A*) in a specific area. On the

Android side, we can then constantly process the video feed we receive, to see if our bot has found the necessary object.

- **Audio Feed :** Along with video feed, the project could be extended to also include audio feed, which could be done on similar lines as we have done with the video.
- **Different Communication Module :** Instead of using the Bluetooth module, one could easily use any module for communication between Firebird and on-board Android. This could be based on user specific requirements. For example, if the application is such that there should be reliability in communication between on-board Android and the Firebird, then we could use IOIO board instead of the Bluetooth for communication. We personally have an experience of shifting from IOIO to Bluetooth in 24 hours. So we see that almost any user-specific module can fit-in.
- **Remote Android Application :** In our project, due to limitation of time we have implemented message sending using buttons, but we can also use an accelerometer for motion control. Also there is a lot of scope for improvement in the functionality and aesthetics of the Android application.

8. References

- *Android* Open Accessory Development Kit | *Android* Developers
  developer.**android**.com/guide/topics/usb/**adk**.html
- IOIO for Android - SparkFun Electronics
  http://www.sparkfun.com/products/10585
  - E-Yantra
    http://www.e-yantra.org/home/index.php
- Atmega 2506 Firebird V Hardware and Software Manual
  - *Android SDK* | Android Developers
    developer.**android**.com/**sdk**/