Research Paper E-ISSN NO : 2455-295X | VOLUME : 4 | ISSUE : 12 | DECEMBER 2018

UNLIMITED RANGE SURVEILLANCE ROBOT CONTROLLED BY ANDROID APPLICATION THROUGH THE INTERNET

AVISHEK BISWAS [1] | KAUSTAV NANDY [2] | PROF RATNA CHAKRABARTY [3]

[1] ELECTRONICS & COMMUNICATIONS ENGINEERING DEPARTMENT, INSTITUTE OF ENGINEERING AND MANAGEMENT, KOLKATA, INDIA.

[2] ELECTRONICS & COMMUNICATIONS ENGINEERING DEPARTMENT, INSTITUTE OF ENGINEERING AND MANAGEMENT, KOLKATA, INDIA.

[3] PROFESSOR, ASSISTANT H.O.D, ELECTRONICS AND COMMUNICATIONS DEPARTMENT, INSTITUTE OF ENGINEERING AND MANAGEMENT, KOLKATA, INDIA.

ABSTRACT

This is aproject on developing a range-less robotic car with a surveillance camera. The movement of this robot can be controlled from anywhere in the world at anytime through the internet. The robot will have an Android phone attached to it, which will be used as it's camera source. An Android application will be developed in this phone so that it streams real time footage to a media server. Another client-based Android Application will be developed to retrieve the live footage from the aforementioned media server and also send commands to the robot's microcontroller via an MQTT server. The microcontroller unit attached to the robot will receive and process these commands and move the robot accordingly.

The movement of the robot and the live streaming is not affected by the distance between the controller Android device and the robot as long as both devices have internet access.

KEYWORDS: INTERNET OF THINGS, HOME AUTOMATION, ARDUINO, ROBOTICS, RTSP, REAL TIME STREAMING, ANDROID APPLICATIONS, MQTT, SURVEILLANCE ROBOTICS.

I. INTRODUCTION

Due to marked rise in international terrorist threat in recent times, the need of unmanned surveillance is dire at this point of time to prevent senseless loss of human lives without sacrificing the lives of soldiers for the same. This robot is mainly built for accomplishing surveillance purposes from a distance at a location that is impossible for humans to always access.

For most wirelessly controlled robots nowadays, the distance between the robot and the controlling device should be maintained within a certain limit. But our robot is not restricted by this range limitation. Since this robot is controlled over the internet, the concept of distance limitation itself is virtually eliminated and the range can be anything, restricted only by the availability of internet connectivity to both the robot and the controller (the Android phone in this case). Thus, it can be controlled even across countries and continents if necessary. Since the MQTT server used for this project is fast and lightweight, the communication will be fast even across countries. Thus, this robot will enable the surveillance of some area remotely from virtually any distance (given that internet is made available to the robot via a router or a dongle)

II. VARIOUS HARDWARE AND SOFTWARE COMPONENTS USED AND THEIR FUNCTIONS

Let us start by presenting the block diagram of the project and discuss the role and details about each component one by one.

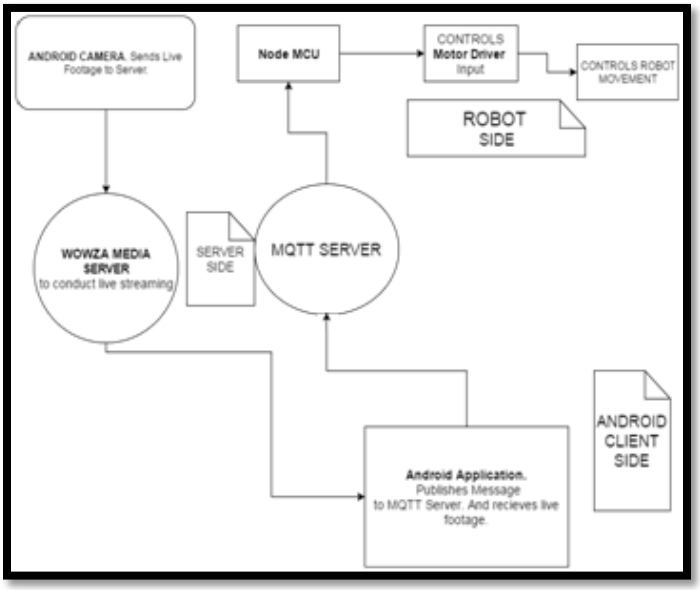

FIG 1: BASIC BLOCK DIAGRAM OF THECLIENT-SERVER-ROBOT SYSTEM

NodeMCU: The NodeMCU development board is the main microcontroller that we use in this project. It is based on the ESP8266 wifi module that generally acts as an added wifi module to other microcontrollers for imparting wifi connectivity to them. But the ESP8266 itself possesses some processing power which is sufficient for our project. The NodeMCU development board incorporates the

ESP8266 with other components like the CH340 USB to serial chip (needed to connect the NodeMCU development board to the computer for programming it) and the SPX3819 low voltage LDO voltage regulator(to regulate the input power supply voltage of +5V to +3.3V which is the input voltage of the ESP8266 core) to male it ready for processing purposes.

It operates at an input power of +5V and it consists of 30 pins including 11 general purpose input output pins (GPIO pins). Its pin configuration is as follows:

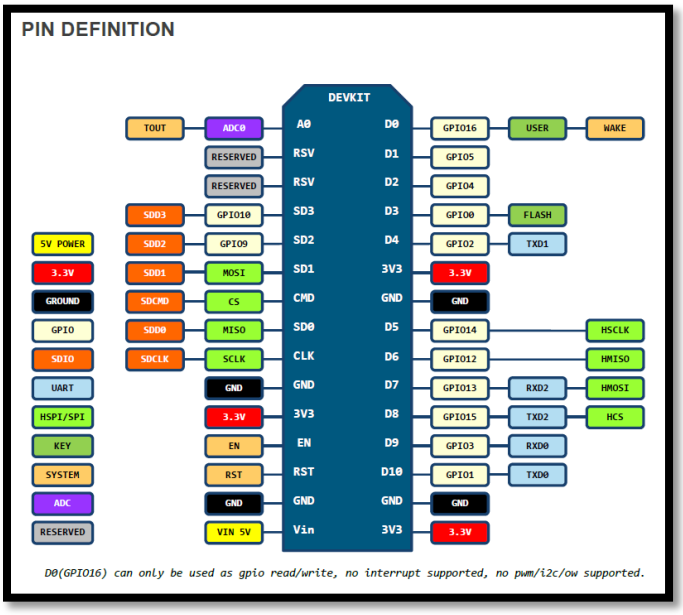

FIGURE 2: NODE MCU PIN DEFINITION

The function of the NodeMCU board in this project is to connect to the MQTT server, check for signals from the server and give the signals to the motor driver through the GPIO pins accordingly on receiving different messages from the server amd processing them based on the program that is stored in it.

The coding for the NodeMCU was done using the ESPlorer IDE, which supports .lua script coding.

Motor Driver: The motor driver unit used in this project is the L293D motor driver board. The NodeMCU cannot provide the amount of current required by the motors to operate. The motors we used required 12V for their operations. So, the operation of the motors were handled by the motor driver. It took the outputs of the NodeMCU board as input and accordingly sent power to the motors to operate as per the requirement specifications passed to it from the NodeMCU development board.

The pin diagram of an L293D board is as follows:

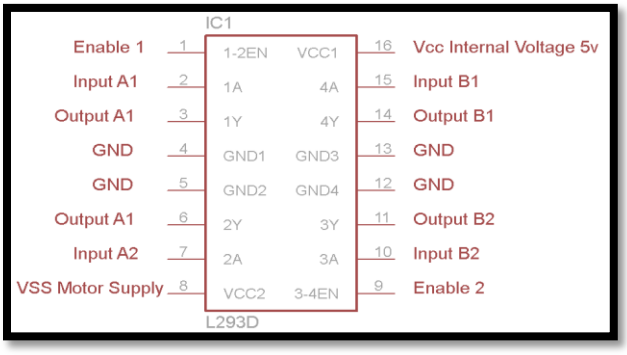

FIGURE 3: L293D MOTOR DRIVER PIN DIAGRAM

MQTT server: This is the internet server across which the controller sends instructions to the robot.MQTT is a light weight publish/subscribe messaging transport protocol, particularly well suited to event-oriented interactions. It was specifically designed for constrained environments such as those found in Machine to Machine (M2M) and Internet Of Things (IoT) contexts where a small code footprint is required and/or network bandwidth is at a premium.

In an MQTT server, the messages are always sent under 'topics' with a specific name. For two clients connected to the server to communicate, the sender client must 'publish' the message under a topic (identified by its topic name) and the receiving client must subscribe to the same topic and thus, receive the message when it is published under that topic by the sending client. Here, publishing means sending the message to the MQTT server under a particular topic and subscribing means monitoring that topic constantly for any message published under it. This protocol of messaging makes the communication simple and is ideal for being used in out project.
For using the MQTT server you have to reserve gain access to the server of a particular company (We used the MQTT service provided by CloudMQTT for our project) with an optional username and password. Then, after making the making the server access available, we connect to the server from the LUA MQTT client running in the NodeMCU board using the userbame and password if applicable. Then the

required topic under which the messages are being published by the controller application is subscribed to, in order to receive the required instructions for the robot.

Wowza Media Server: A cloud-based media server needs to be set up to facilitate our Client-Server streaminginteractions.WOWZA Media Server fulfills the purposes of live streaming.Wowza media server is one of the most popular and critically acclaimed media server choice in the world right now, having won the Streaming Media Reader's Choice Award 2015 and the Cloud-based Streaming Server Award in the same year. It is a robust, customizable media server software that powers reliable high-quality video and audio streaming. It takes in any

format, transcodes it once, and reliably delivers it in multiple formats to any device, anywhere, anytime. Wowza software fits with nearly every streaming architecture, and provides a flexible pricing model and deployment options allows users to scale easily as needs change. Wowza supports RTSP live source streaming and RTMP streaming to flash players as well.

After setting up a cloud server, a customizable stream is created.The following information about source connection is visible in the streaming home page after the stream has been aptly set up.

- The cloud server address with port (normally 1935)
- Workspace folder name (user assigned)
- Stream name (user assigned)
- Authentication details (username and password -optional)

The stream for feeding RTSP (Real Time Streaming Protocol) stream is accessed by using rtsp://server-ip:port/workspace_name/stream_name

Real Time Streaming Protocol (RTSP) is used to connect from android camera fixed on our robot chassis to stream live footage to server; and Real Time Messaging Protocol (RTMP) is used to retrieve the live feed (in real-time)from the server to a Flash-based video player in our client app.

Eclipse IDE, Java Development Kit (JDK) and Android Development Tools (ADT): For developing the Android Applications to a) Stream live footage from Android Camera fixed in Robot chassis to server, and b) Retrieve the live footage and send MQTT commands to the NodeMCU to facilitate the movement of the robot, a proper a Integrated Development Environment is required. Eclipse IDE was used for our purposes. The latest JDK (1.7) and ADT (23.0.7) plugin for Eclipse were used.Three open source GitHub project libraries were used to simplify complex coding. RTSP live streaming was achieved by the **libstreaming** library, RTMP streaming was done using the **Vitamio** library, MQTT publishing and subscribing were done using the **MQTT-paho** libraries (**org.eclipse.paho.client.mqttv3-1.0.2.jar** and the **org.eclipse.paho.android.service-1.0.2.jar).**

All the libraries that were used in this project were open-source projects from reliable sources (GitHub). Their use was legal and authenticated by the developer.

III. SETTING UP AND READYING THE ROBOT FOR WORK

The basic steps followed to setup the robot and readying it are as follows:

1. The driver for the CH340 USB to serial converter, namely the CH340G driver, was downloaded.
2. The NodeMCU development board was flashed with the latest NodeMCU firmware. Flashing the NodeMCU board means burning into the the NodeMCU flash memory firmware that allows you to program the ESP8266 modules with LUA script. For this the NodeMCU flasher tool software and the latest NodeMCU firmware was downloaded.
3. The NodeMCU board was connected to the computer and was programmed using Esplorer which is an integrated development environment (IDE) for ESP8266 and other boards based on it. The programming language used for this was Lua which is a lightweight language most ideal for this project. The program was such that the NodeMCU board connects to the wifi and then to the MQTT server on startup and sets the required output pins high based on the instructions sent by the controller application through the MQTT server. After writing the program, it was uploaded to the NodeMCU board.
4. The connections and wirings were made. First the motor driver and the motors were set up.The L293D Motor Driver IC that we discussed in the previous chapter also comes in module form. There are two input nodes (four terminals) and two output nodes (four terminals). There is a power terminal that needs a 12V power supply to run. The two input terminals of a single node were connected with Pin 2(Input A1) and Pin 7(Input A2) to facilitate a single motor. The other input terminals were connected to pins 10 (Input B1) and 15 (Input B2). The inputs of these terminals were derived from our ESP8266 NodeMCU module.The two output nodes of the Module were connected with the respective output pins of the motor driver IC. Output terminals 1 and 2 were connected with pin 3 (Output A1) and pin 6 (Output A2), and output terminals 3 and 4 were connected with pin 11 (Output B1) and pin 14 (Output B2). This output terminals were connected with the motors of the robot.The power node was connected with a 12V power supply. The supply voltage was achieved by connecting eight 1.5V cells in series using two four-battery holders. Then the NodeMCU development board was connected to a +5V input power (a power bank in this case).

The following is a representation of the final circuit:

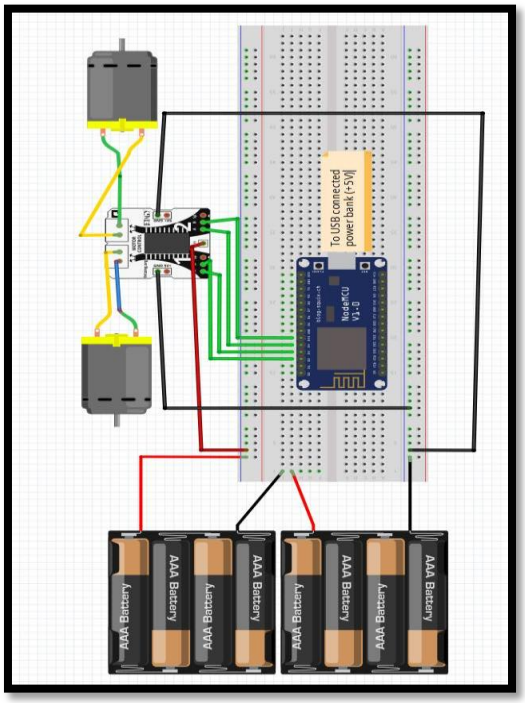

FIGURE 4: NODE MCU AND MOTOR DRIVER CIRCUIT CONNECTION

5. An android application is developed to stream live footage from our android camera to our Wowza Media Server. This camera was fixed on the chassis of the robot. The live stream application was developed with the help of the open source github Android project library called "libstreaming". The app captures video through camera, converts the video to a specified resolution and quality, and performs live streaming via RTSP protocol. The target of this stream is given by
rtsp://wowza_server_ip:port/wowza_workspace/wowza_streamname
The username and password should be provided in the code if Source Authentication was enabled while setting up the channel in the back-end media server.
6. Another android application was developedforthe client-end that does the following work:

- Publish MQTT messages to our topic so that it controls of the motion of the robot.
- Enable user to view live footage streamed from our media server
- Create an intuitive and simple User Interface to fulfill our goals.

To help tackle the MQTT problem, the excellent GitHub open source project MQTT-PAHO was employed. The two .jar files, namely the **org.eclipse.paho.client.mqttv3-1.0.2.jar** and the **org.eclipse.paho.android.service-1.0.2.jar**were used as Referenced Libraries**.**

An intuitive UI design is created that has four buttons pointing Up, Down, Left, Right. Whenever the user holds down one of the four buttons, the respective message gets published in the MQTT server under the relevant topic (the one which the Node MCU lua script had subscribed to previously). When the user releases a button, the message that invokes the "rest" state of the robot is published.

The following messaging system was used:
UP: Publish "w"
DOWN : Publish "s"

LEFT : Publish "a"

RIGHT : Publish "d"

HALT: Publish "h"

To retrieve the live footage from the Wowza Server via RTMP (Real Time Messaging Protocol), a Flash player is hosted in Android. Since Android has officially depreciated Flash-based players, the open source **VITAMIO** library was used to set up the Flash-based environment. The video is displayed on a video-view located just above the button layout in the UI design (as shown in Figure 5).With the help of the Vitamio Library, a connection is made to our stream channel in Wowza Server and the client's Android device.

The stream link is as given: rtmp://wowza_server_ip:port/wowza_workspace/wowza_streamname

Figure 5: Android Client-Application UI

IV. HOW THE SYSTEM WORKS

The following table shows the relationship between the

MQTT message with respect to the robot movement, via the NodeMCU and the Motor Driver.

Mqtt mess-age (publi shed from App)	NodeM CU Pin2/M otor Driver Pin 3	Node MCU Pin3/ Motor Driver Pin 7	Node MCU Pin4/ Motor Driver Pin 15	NodeMC U Pin5/Mot or Driver Pin 10	Left Wh-e el *	Right Whe-e l *	Robot Move ment
w	H	L	H	L	C	C	Forw ard
s	L	H	L	H	CC	CC	Back
a	L	H	H	L	CC	C	Left
d	H	L	L	H	C	CC	Right
h	L	L	L	L	Idle	Idle	Halt

*C: Clockwise, CC: Counterclockwise

FIGURE 6: TABLE SHOWING THE RELATION BETWEEN THE MQTT SERVER MESSAGES AND CORRESPONDING ROBOT MOVEMENT

When the up button is pressed on the client-app, the message "w" is published in the relevant MQTT topic of our server. Similarly, for pressing left, right, back the messages "a","d" and "s" are published. When the user releases any of the buttons, the "h" message is published which makes the robot halt.

The live streaming is done through the Wowza Media server. One needs to activate the source input for the stream from the server, and load up the camera application in the android phone fixed to our robot. The client will receive the RTMP footage by clicking the "play" button in the android application.

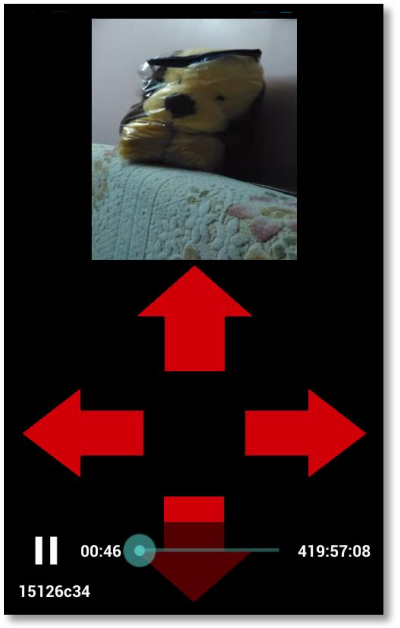

FIGURE 7: ANDROID CLIENT APP LIVE STREAMING FOOTAGE

This is a sample screenshot taken from the Client Android Application. The arrow keys are used to publish the messages "w","s","a","d" or "h" to the MQTT server. The picture of the teddy bear is the live footage obtained from the robot camera through the Wowza Cloud Media server. The playback controls at the bottom of the screen shows the controls for the live footage (pause/play).The ID 15126c34 in the bottom left of the image denotes the encoded string for our stream name.

The detailed flowchart of the system is as shown.

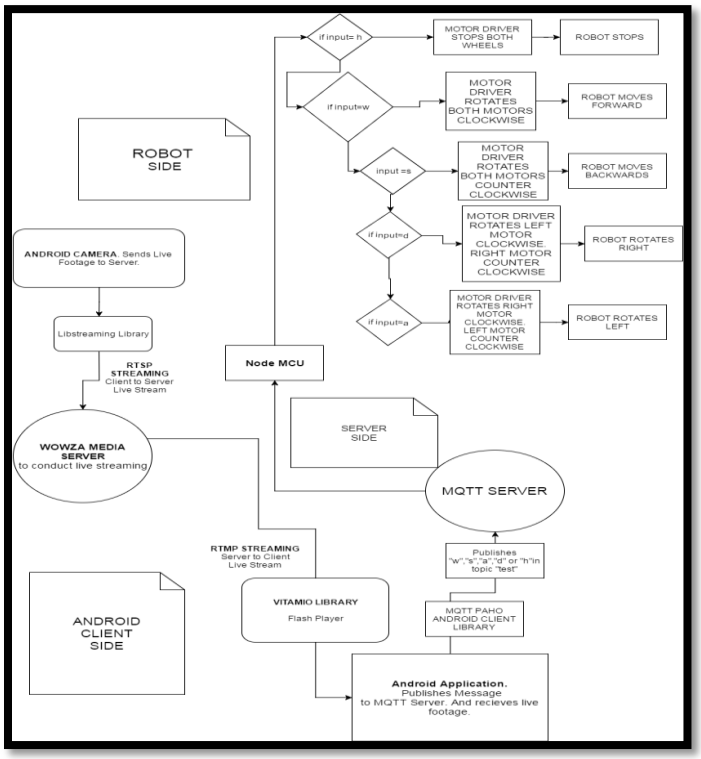

FIGURE 8: DETAILED BLOCK DIAGRAM SHOWING THE VARIOUS ASPECTS OF THE CLIENT-SERVER-ROBOT SYSTEM

V. CONCLUSION

Thus, our aim of building a surveillance robot that is not restricted by range is achieved. As discussed, the robot is controlled through an android application from any range within the earth as long as there is an internet connection available both at the site of the robot and the site of the controller. This enables the controller to keep a watch over areas that was earlier impossible or difficult to reach and stay at by humans. With the increased rise in global terrorism we have today, this robot may find applications in monitoring areas with imminent terrorist threat without sacrificing the lives of any soldier or police worker in order to protect civilian lives. This could be a huge help to the military for keeping watch over remote and inaccessible areas because unlike most robots, the controlling person need not be anywhere near the robot. Furthermore, the robot could be endowed with extra abilities as upgrades that will enable it to accomplish more tasks over simple surveillance. The source code for the project can be found at
https://github.com/shekavi/Rangeless-Android-controlled-Surveillance-Robot

VI. FUTURE SCOPE

The aim of this project was to build a robot for the purpose of surveillance that could be controlled and monitored

from any range. Our robot has fulfilled this aim. But its applications do not stay permanently restricted to that. If needed, the robot can even be upgraded to do more tasks like picking up objects, pushing objects, cutting wires and lifting objects using mechanical arms driven by extra motors and motor drivers that may be installed if needed. This would make the robot capable of not only surveillance but also simple human-like tasks and repair simple repair works that may be required at different locations. Thus, upon demand, this robot can be turned into a complete package of a task following surveillance robot controllable from any distance.

VII. ACKNOWLEDGEMENTS

I would like to thank my parents, Dr. Amitava Biswas and Mrs. Tapashi Biswas for their undying support and well wishes. I would be nowhere without them. Being part of such a project was a great learning curve for me, and I am really excited about the future application of this effort.Without the tutelage and guidance of our mentor, Professor Ratna Chakraborty, and contributions of my partner Kaustav Nandy, this project won't be possible. Lastly, I would like to thank my friends Foram Joshi, Abhishek Dey, and Sudip Chakrabarty for continuously inspiring me and motivating me to try my best at everything I attempt.

-- Avishek Biswas

I would like to thank my parents, Mr Shayamal Nandy and Mrs Bhaswati Nandy for supporting and motivating me. This project would have been very difficult for us to do if not for ur mentor, Professor Ratna Chakrabarty, who helped us with every little problem we faced. My partner Avishek Biswas was excellent during the project and I thank him for his contributions.

--Kaustav Nandy

REFERENCES

1. Hsiang Wen Chen ; Fuchun Joseph Lin; "Converging MQTT Resources in ETSI Standards Based M2M Platform" - Sep 2014

2. 2014 IEEE International Conference on Internet of Things (iThings), and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom)

3. Yuvraj Upadhyay ; Amol Borole ; D. Dileepan: "MQTT based secured home automation system" - Sep 2016

4. Shanzhi Chen ; Hui Xu ; Dake Liu ; Bo Hu ; Hucheng Wang: "A Vision of IoT: Applications, Challenges, and Opportunities With China Perspective" - Aug 2014

5. Lalit Mohan Satapathy, Samir Kumar Bastia, Nihar Mohanty: "Arduino based home automation using Internet of things (IoT)"

6. Rampeesa Vijay, Thotakura Sainag, Vamsee Krishna : "A Smart Home Wireless Automation Technology using Arduino based on IOT" - Oct-Dec 2017

7. H. Schulzrinne, A. Rao, R. Lanphier: "Real Time Streaming Protocol (RTSP)" – 1998.