Wireless Generic Real-Time system for Sensors and Actuators
Clustered Robots using generic MCU's

[1]Ashay V Sant, [2]Krushant Chalthanwala, [3]Ashwini Pandhare, [4]Suraj Balwan
[1,2,3]Engineering, [3]Assistant Professor
[1,2,3,4]Department of Computer Engineering,
19, Late Prin. V.K. Joag Path,Wadia College Campus, Pune – 411001

Abstract : To create a generic system composed of Arduino, Raspberry Pi, ESP8266 and various custom made Printed circuit boards for resistor based GPIO interfacing with MCU's. The system is designed with a global multidimensional perspective for any requirement needed. The system features an interface for I2C devices, furthermore I2C devices can work in master slave mode, SPI devices working on MISO, MOSI, SCK pulses from either the Arduino or Raspberry Pi. Raspberry Pi revision 2 or Raspberry Pi revision 3 may be interchangeably used for the system and support for RALink Wi-Fi Adapter (RT2501/RT2573) on Rpi version 2 is provided. Furthermore easy pulse phase modulation based Trans receiver is coupled for close range accurate communication. Actuation is based on Servos and BO motors. The system features an onboard Camera and Gripper. Camera actuates in both X and Y axis, same with the Gripper. The system is powered by a 3055 transistor and multiple hybrid combinations of Li-Po and Li-Ion batteries ranging from 7.4V to 11.1V.

Index Terms **- RaLink, Wi-Fi, Raspberry Pi, Actuation, Revision, multidimensional, I2C, SPI.**

I. INTRODUCTION

USB camera based security surveillance system is created. The camera is interfaced with raspberry Pi using USB interface. Light Switches are controlled over Raspberry Pi GPIO. A program is written to monitor surroundings using camera and motion is detected.

A python environment is created for email based home automation system. Raspberry Pi is connected to a modem for internet based connectivity and relay driver is used to control various devices. Performance is evaluated using LED's. When signals are sent over email, LED's get controlled.

Alternative to Raspberry Pi is NVIDIA's Jetson TK1 developer kit. Power of the Graphics processing unit combined is embedded systems is a pure enhancement in both "cost" and performance. Using NVidia Kepler architecture and CUDA platform, an easy access to OpenGL 4.4 is provided. Enhancement in OpenCV is seen. Trade off between high cost and performance is done.

NVidia know for Kepler, Maxwell and recently 14nm Pascal and futures Volta, a giant in GPU manufacturing develops AI technology using neural networks on a technology known as NVIDIA DGX systems. Tied up with Toyota NVIDIA creates an autonomous AI navigation system.

Raspberry Pi based remote monitoring system is created to monitor ECG. A raspberry Pi version B is explained along with booting, current capacity, voltage etc. Also a GSM module is connected to raspberry Pi and controlled using AT commands. Coding is done in Python.

Introduction to students regarding embedded microcontrollers is done in labs. An Arduino board is connected to a speaker to make music. Also a LCD display is connected to Arduino to display and trigger actions using the LCD Display and button combo circuit.

HIH-5030 for sensing Humidity, TC1046VNB for sensing temperature along with pressure sensor and shock sensor are utilized in a home automation system. The heater is implemented by using a 25W power resistor to heat up the room and illustrate usage of vent for the cooling purpose. Lab VIEW software is used for software level implementation providing built in graphics and graphs and also state diagrams. A HC04 ultrasound sensor is also configured to calculate distance using sound signals.

1.1 Hardware

The system is meant to be featuring a rich set of controllers and peripherals. The main MCU's are Raspberry Pi and Atmega 328p based Arduino. Specifically, Arduino Nano is emphasized over Arduino Uno. Arduino Nano is deployed on a shield featuring direct connections for peripherals which include Signal, VCC and ground pins in a single row arranged in columnar fashion. This completes the majority part of main MCU's.

Secondly the ESP8266 is also placed for receiving signals over the Wi-Fi. The ESP8266 module is just meant to communicate with an Arduino via serial interface so that the ESP8266-Arduino pair can be together connected to Wi-Fi and help in easy 5V coupling to other peripherals.

Thirdly the peripheral section includes

1.2.1 1602 Display

The 1602 display is a 2 row 16 column display. Each column has 2 dot matrices and each row features in total of 16 dot matrices. Thus total representable characters using dot matrices at a time are 32 (16 X 2).

1.2.2 PCA9685

The PCA9685 is a servo driver meant to control up-to 16 servos on a single device. The device is an I2C bus controlled slave. The frequency of operation is 24Hz to 1526Hz.

1.2.3 L293D H-bridge

The L293D are mounted on custom made PCB for driving DC motors up-to 12V. Two L293D are used so as to distribute current load in parallel combination. The Enable pins are given PWM pulses so as to regulate the speeds of the driven DC motors.

Fourthly the actuators used are

1.2.4 Servo Motors

The servo motors used are SG90 servo of the type "Micro" because of small torque rating of 2.5 kg-cm.

1.2.5 BO motors

BO motors are brushed DC motors operating anywhere from 5V to 12V DC. With enhanced PWM pulses from MCU, speeds on such motors can be regulated.

1.2 Software

The software to be used includes

1.2.1 Dnsmasq

For masquerading connections i.e. from ppp interface to Wi-Fi (Wlan) interface. It acts as a DHCP server for our system. Our system is placed in a subnet of 192.168.3.0 so as to lease ip addresses in the 192.168.3.0 subnet.

1.2.2 Iptables

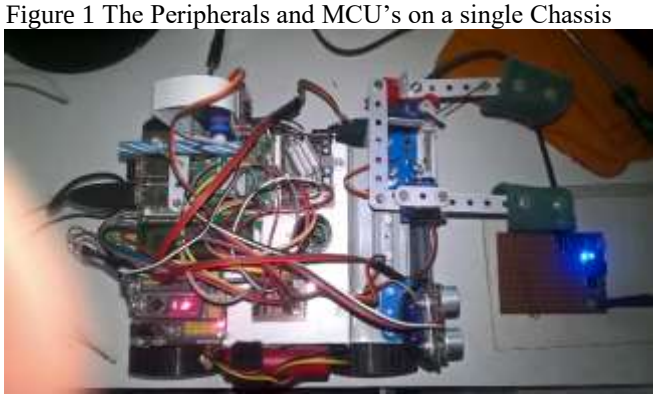
User level program allowing the user to create, modify the low level OS level firewall rules. Here only a packet forwarding is enabled at first and then the forwarding rules are added.

1.2.3 Hostapd

Make raspberry a hotspot on any Wlan interface. WPA2 security is used as a convention and nl80211 kernel level drivers are used for Wi-Fi adapter.

Further simple codes are written in C to listen to incoming connections via network, writing to display, getting device's IP address etc. These programs are converted to header files for ease of use. Furthermore shared object perspective is used in python to enable C and Python hybrid coding.

Figure 1 The Peripherals and MCU's on a single Chassis



II. SYSTEM ARCHITETCURE

The system is focusing on use of various MCU's, there integration, communication and interfacing with each other and various peripherals. The entire system is Internet controlled thus the made system is a part of *"IOT"*.

The system aims at a generic perspective so as to give following functionalities

1. Remote Monitoring
2. Security Purposes
3. Overview of farmstead
4. Post Disaster view
5. A remote connection point
6. Router

The system deploys a Raspberry Pi at its core along with Arduino, ESP and other slave MCU's.

2.1 Inter MCU Communication

Inter MCU communication takes place via the serial interface.

Let the Rx and Tx pins of MCU 1 be labelled Rx0 and Tx0 resp. Similarly labelling Rx and Tx pins of MCU 2 as Rx1 and Tx1. So for MCU 1 to communicate with MCU 2 and vice versa we need to connect Rx0 to Tx1 and Rx1 to Tx0.

Thus signals are sent on one end and received on the other end. Thus 2 MCU's intercommunicate by using Rx and Tx i.e. serial interface.

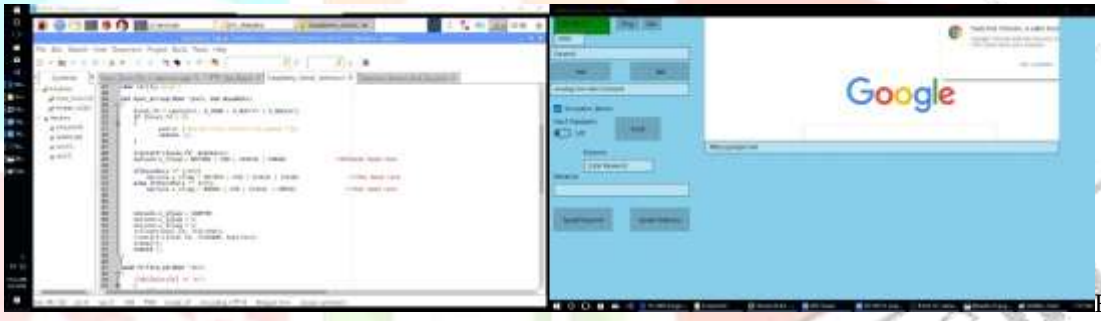


Figure 2 MCU communication via RX and TX i.e Serial interface

2.2 MCU to peripheral communication

MCU to peripheral communication takes place via a self-made resistor interface or a 5V to 3.3V TTL converter self-made interface. Direct attachment of peripherals is avoided so as to protect the MCU's from voltage and current spikes.

The resistors used vary on type of connection, but mostly 100 ohms to 330 ohms are used, sometimes resistors up to 10 Ohms are also used wherever current requirement is less.

2.3 Master MCU The Raspberry Pi version 3

The master MCU i.e. the central MCU for entire system is a 64bit Raspberry Pi revised version 3. The MCU is customized, a heatsink is added on top of MCU so as to dissipate heat generated during heavy operation. Pi version 3 has a built in Wi-Fi adapter hence the effect of heat is seen as compared to pi version 2

2.4 Slave MCU's

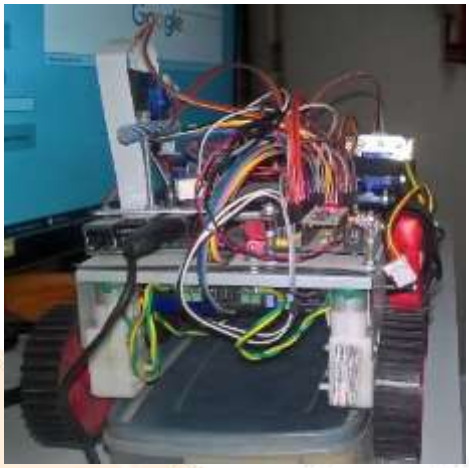
2.4.1 Ardunio

Acting as a sole 5V MCU in our entire IOT based system is an Atmega 328p chip communicating with the peripherals operating at 5V logic levels.

2.4.2 ESP8266

ESP8266 is a Wi-Fi enabled MCU able to connect to Raspberry Pi's server and serving web pages on request and communicating via peripherals via serial interface.

Figure 3 The Actuating Wheels and Peripherals



III. WORKING

The raspberry pi is configured to act as a router. For this purpose built in linux firewall is used i.e Iptables. Packet forwarding rule is set in Iptables so as to forward packets over WLAN or PPP interface to other interface say WLAN or Ethernet. Thus packets aimed at Raspberry pi's IP over internet are forwarded to other interfaces thereby indirectly enabling other devices to be controlled over internet provided that the slave MCU's are connected to raspberry Pi via WLAN hotspot.

So overall, Raspberry Pi is our router and ESP8266's are connected to router's WI-Fi interface over wlan interface and packets are forwarded via a simple precompiled low level C program or alternatively a high level implementation providing multithreading is done in python.

3.1 Connection to Raspberry over Internet

3.2.1 Java Program

A java program is written to create a webview so as to allow viewing of web pages in a single window. The window is overlaid with toolbars, featuring some buttons which actually connect to Raspberry Pi over Internet using low level socket interface.

If a button say 'Mover Forward' is clicked then the corresponding action on sender i.e. client is

- Create a Socket
- Connect the Socket to Rpi
- Send Message to Rpi

Corresponding action on server is

- Server is listening to incoming connections
- Server receives a request from our client
- Server attaches the client request to a specific port

Communication over TCP channel is established

3.2.2 C# APP

A universal windows phone app is written in C# language using .NET framework for all Windows 10 based machines including Desktop versions and mobile phones. The app has a functionality for speech based control for communicating with said IOT system. Also manual typing of actions in text boxes is allowed.

In our C# App, multicore and multithreaded program is readily and natively supported thereby making all our socket based synchronous calls asynchronous opposing our Java implementation. Thus all calls are non-blocking in C# program, "await" directive is used wherever required sync operation.

Figure 4 The C# App in connecting Phase

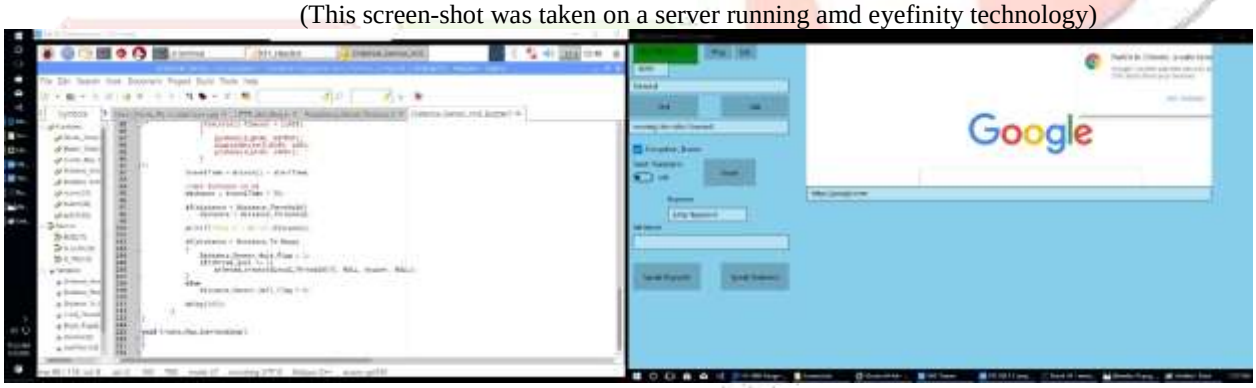
3.2 Hardware Actuation

Hardware Actuation is carried out over the GPIO interface of Raspberry Pi, Arduino. The raspberry pi is a 3.3V compatible MCU whereby the other peripherals like the servo driver, Display are 5V compatible. Hence a 3.3V to 5V adapter is put in place to interface GPIO of raspberry to peripheral devices. Arduino is directly connected to H-bridge and ultrasound distance sensor as it is 5V compatible.

3.2.1 C Programs

C programs are developed for hardware actuation to facilitate easy testing and low level system access. Via C programs, using the library WiringPi and other I2C device libraries easy access to peripherals is achieved. Also multithreading is exploited using pthread library for putting various sensors in different software threads.

Figure 5 The distance sensor program using pthread libraries(left) and C# app command forward(right)
(This screen-shot was taken on a server running amd eyefinity technology)



3.2.2 Python C hybrid program

Also, some high level implementation is done in python for ease of development and rich set of functions, and also to minimize need of compilation. Jumps are made to C code via python using shared objects analogous to Dynamic link libraries in Windows system.

Figure 6 C shared object creation using gcc compiler directives

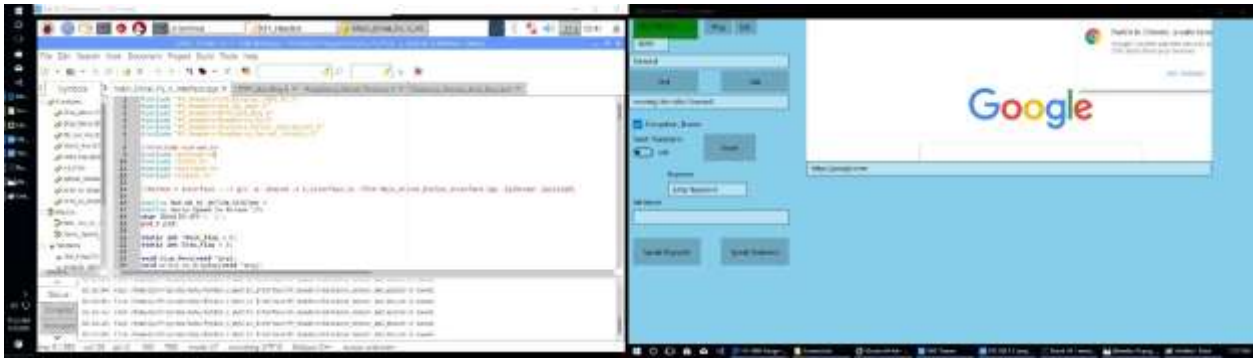


Figure 7 The Lipo Battery 11.1V (3X(3.7V) at 2.2A) 2.2A for powering Wheels

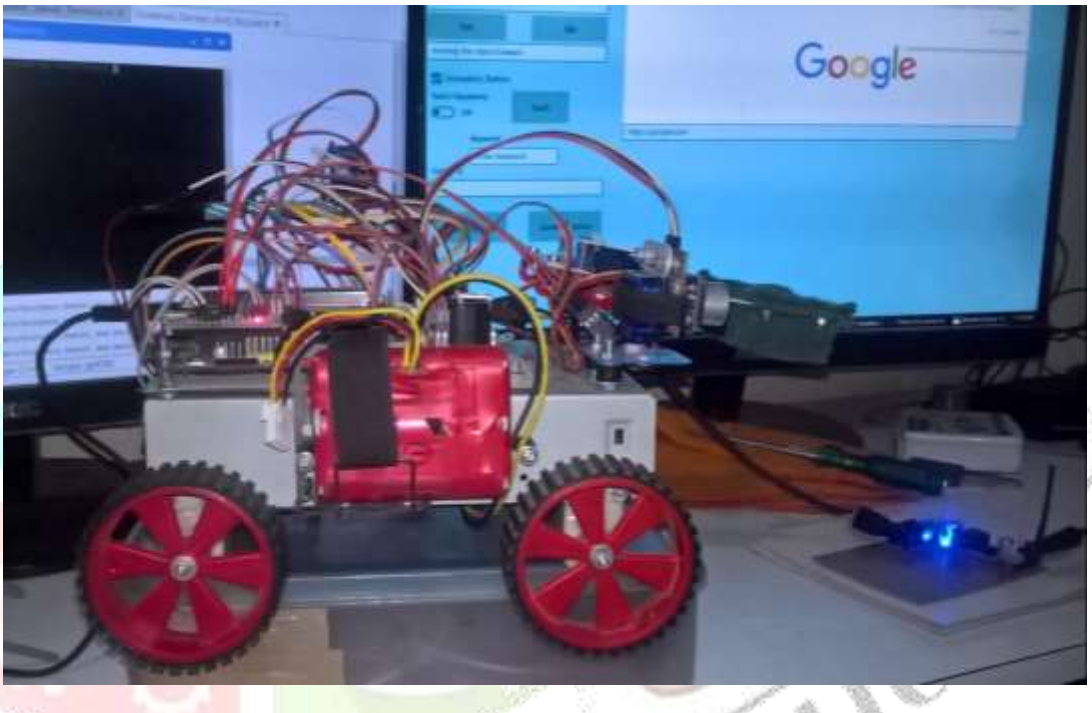
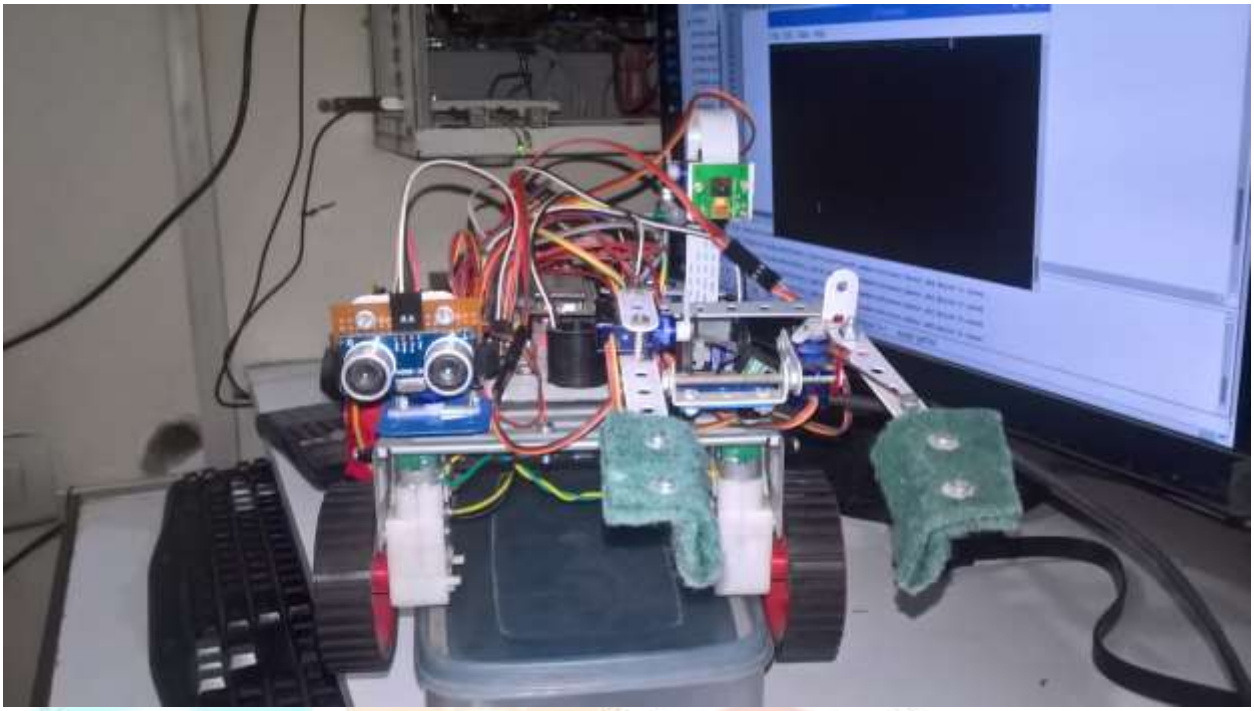


Figure 8 The Gripper and Ultrasound Distance detector



IV. TESTS

Real-time tests are carried out for said system in a controlled environment.

4.1 Internet Connection Test

Internet is connected using Tata Photon+ USB dongle using wvdial dial-up connection software. Also usb-modeswitch, a Linux utility to switch between modem mode and data mode is required. Once connection is established our system gets a dynamic IP i.e. IP valid for said instance.

4.2 Gripper Actuation Test

Table 1

Opening span	9 cm approx.
Closing span	0.5 cm approx.
Max Lifted heighted of Gripper (from mounted chassis)	7 cm approx.
Max height flat (from mounted chassis)	2.5cm approx.
Max length of Gripper	11cm approx.

4.3 Robot Actuation Test

4 Motors provide us the required actuation.
Modes supported are

- Simple Forward with controllable speed
- Simple Reverse with controllable speed
- Simple Left turn with controllable speed (Left motors stopped)
- Simple Right turn with controllable speed (Right motors stopped)

- Fast Right turn (Right motors reversed, left motors forward)
- Fast Left turn (Left motors reversed, Right motors forward)

4.4 Java and C# client Application Test

Java and C# application are tested regressively after every minor changes incorporated. The C# package needs to installed and certificate to be authenticated prior to running on any windows platform (Requires Windows 10 anniversary update). Java application is exported to a runnable jar and requires java jre version 8 for running flawlessly on any platform.

4.5 Camera Actuation Test

Currently, camera supports a full 180 degree actuation in x and y plane. Furthermore in actual final product, according to the camera used the degree of actuation will be cut off for overcoming camera cable limitations.

4.6 C v/s Python Test

A program that uses for loops for introducing appropriate delays for generating PWM pulses in our raspberry system was written in both C and Python and tested using the same interface i.e the "wiringPi" GPIO.

Table 2

	C Program	Python Program
Shell command used for calculating time since epoch	time	time
GPIO library	wiringPi	wiringPi
Run as	Super User	Super User (for wiringPi access)
Time Real	15.951 seconds	1minute 15.914 seconds
Time User	15.930 seconds	1minute 14.540 seconds
Time sys	0.000 seconds	1.310 seconds

Thus deducing that the same interpreted python program is far slower for hardware interfacing a low level implementation is done using C language.

V. FUTURE SCOPE

A raspberry pi is used for the entire project. It acts as the core of the system. Simple berg wires are used for connectivity. When coming to actual deployment, these components will be substituted by industry grade static high quality cables and mostly a single PCB can be fabricated. The cost will be a bit elevated but the custom double track design and discrete component addition will help saving costs.

Thus overall enhancements are possible to the model at industry level. Techniques like a fully covered high quality insulated chassis can be used, high torque motors, high voltage batteries can be accommodated and varied based on one's affordability.

More cross platform connectivity apps can be developed on various platforms can be implemented in different languages. Our program on server side employs a socket based technique, hence connectivity can be established using socket in any language.

VI. CONCLUSION

The aimed system is a "generic" in nature. The main purpose of the system is remote monitoring, security and farmstead live monitoring with the help of a camera. The said system is created on a metal chassis with high ground clearance and high strength. The system as a whole is a self-made router based IOT system with customized peripherals and MCU tweaking's to suit the particular application.

The system is meant for normal users with ease of access, multiple access methods and high throughput. The system's speed however is network limited as of now, a 3 GHz Tata photon plus is used to get a dynamic IP for the system. This enables us to deploy

a system anywhere in the open physical world, however throughput varies based on the proximity of network tower and minor weather changes.

If these constraints are taken into consideration by the user, a user can enhance the throughput of the system by himself selecting the right location and right wireless USB Dongle for deployment. The system is assembled on a single chassis with the help of nut-bolts hence any damage/breakage can be repaired or remounting can be done easily. Also system is composed of discrete components, in case of failure, only the failed component can be replaced thereby reducing maintenance cost.

REFERENCES

[1] Virginia Menezes, Vamsikrishna Patchava, M. Surya Deekshith Gupta. Surveillance and Monitoring System Using Raspberry Pi and SimpleCV. 2015 International Conference on Green Computing and Internet of Things (ICGCIoT).

[2] Sarthak Jain, Anant Vaibhav, Lovely Goyal, 2014. Raspberry Pi based Interactive Home Automation System through E-mail. 2014 International Conference on Reliability, Optimization and Information Technology

[3] https://www.nvidia.com/en-us/autonomous-machines/embedded-systems-dev-kits-modules/

[4] https://www.nvidia.com/en-us/self-driving-cars/

[5] M. Surya Deekshith Gupta, Vamsikrishna Patchava, Virginia Menezes. Healthcare based on loT using Raspberry Pi. 2015 International Conference on Green Computing and Internet of Things.

[6] Gordana Laštovicka-Medin, Marko Petric. Embedded Lab: Arduino Projects in Science Lessons. 4th Mediterranean Conference on Embedded Computing.

[7] Silviu Folea, Daniela Bordencea, Casiana Hotea, Honoriu Valean. Smart Home Automation System Using Wi-Fi Low Power Devices. Automation Quality and Testing Robotics (AQTR), 2012 IEEE International Conference.