Multiple Motion Control System of Robotic Car Based on IoT to Produce Cloud Service

Maliha Rahman Mishi, Rabeya Bibi, Tanveer Ahsan
Dept. of Computer Science and Engineering
International Islamic University Chittagong
Chittagong- 4203, Bangladesh
cse.mishi29@gmail.com, rabeya.cse950@gmail.com, tanveerahsan@gmail.com

***Abstract*- The world of control is an exciting field that has exploded with new technologies where the Internet of Things (IoT) vision becomes reality. This paper proposes a multiple motion controlling mechanism of a robotic car using Raspberry Pi which works as master and Arduino UNO which works as slave. Each device is uniquely identifiable by the controlling software which is the core concept of IoT. Client manages the activities of the car from remote or distant places over the internet by voice commands and Universal Windows Application and also able to get data and feedback. The main contribution of this paper is that it leverages the efficiency of robot's motion controlling system because robotic car can receive direct commands at a time from multiple sources which make the maneuvering system more efficient. Both device and client do not need to be online at the same time. Commands and data are stored in cloud service which delivers them when the device is ready to receive. A GPS system is incorporated thus clients can trace the car. The system has ultrasonic distance sensor for avoiding obstacles coming in between its path. We present the architecture and design of the Raspberry Pi and Arduino communication software and illustrate how to control the car by means of commands and application.**

***Keywords* - Raspberry Pi, Arduino, Ultrasonic distance sensor, GPS, motion control, cloud service, IoT.**

I. INTRODUCTION

Having lesser limitations, more accuracy and being more reliable are what make an automaton more preferable. The controlling mechanism of these systems makes them more outstanding. Multiple control system ensures that a collection of independent computers appears to users as a single controlling system. It uses decentralized elements or subsystems to control distributed processes. They offer flexibility, extended equipment life, simplicity of new equipment integration, and centralized maintenance when used in an industrial environment [6]. Several advanced control systems of robots have been developed based on existing control techniques or new control techniques that have been build on purposes [7]. As a result, for efficient and flexible processing, the multiple control mechanism is more than a necessity. The accessibility and availability of inexpensive credit card sized single board computer such as Raspberry Pi has enabled the creation of numerous automated and controlling system that has low power consumption, faster processing ability at a lower cost. The multiple control system of robots proposed in this paper integrates the use of affordable instruments, connectivity, wireless communication and efficiency of controlling mechanism.

978-1-5090-5627-9/17/$31.00 ©2017 IEEE

A. *Background study and overview*

This section provides a descriptive summary of some methods that have been implemented and tested for controlling system of robots and devices by Raspberry Pi and Arduino. Vladimir Vujovic et al. described a Raspberry Pi home automation system where Raspberry Pi works as a sensor web node for controlling appliances in home automation which makes it the perfect platform for interacting with many different devices. Here Raspberry Pi is not just a sensor node but a controller [1]. Yet the controlling mechanism only includes data collection and updating and works only in indoor environment. Another device controlling mechanism of Raspberry Pi is described in a Raspberry Pi based home automation system through E-mail [2]. The contribution of this paper is, Raspberry Pi can read out the commands of users through E-mail and the devices to be controlled are interfaced with Raspberry Pi using relay driver [2]. However clients can only control the switching state of the appliances, no other controlling system is included.

Jaroslav Sobota et al. [3], proposes extremely inexpensive and flexible control platform using Raspberry Pi and Arduino running the REX control system which is an open system for embedded control [3]. On the other hand, REX platform is not standard enough and unable to control a large number of devices at a time.

Another real- time monitoring system has been implemented in developing a fire alarm system using Raspberry Pi and Arduino [4]. In this paper it is described how Raspberry Pi controls the situation based on sensors. However it has not incorporated any user controlled interaction and is only a sensor based module. Anita Sabo et al. described a controlling mechanism of robotic arm using Raspberry Pi through the internet in a research paper [12]. In spite of its advantages, it has some limitations. It only incorporates the controlling mechanism through web service and the client is unable to detect its location. Further, there is no feedback system and so the client has no way to be sure of effective execution of command, which is a must-have feature in any system connected to the internet.

The solution for the problems from the previous researches as stated above is to develop a multiple controlling system that allows clients to control robots from distant places through voice commands and client application over the internet. Wireless connection is considered here. In this paper, the motion control system of robotic car is considered. Initially the commands include: move forward, move backward, turn left, turn right, rotate left, rotate right, activate obstacle detection, and deactivate obstacle detection. These commands can be given via voice commands and/or user application. It is possible to locate the car continuously in the UI and get feedback and data regarding to the car. Also the ultrasonic distance sensor helps the robot to avoid collision with objects coming in between its path.

II. SYSTEM CONFIGURATION

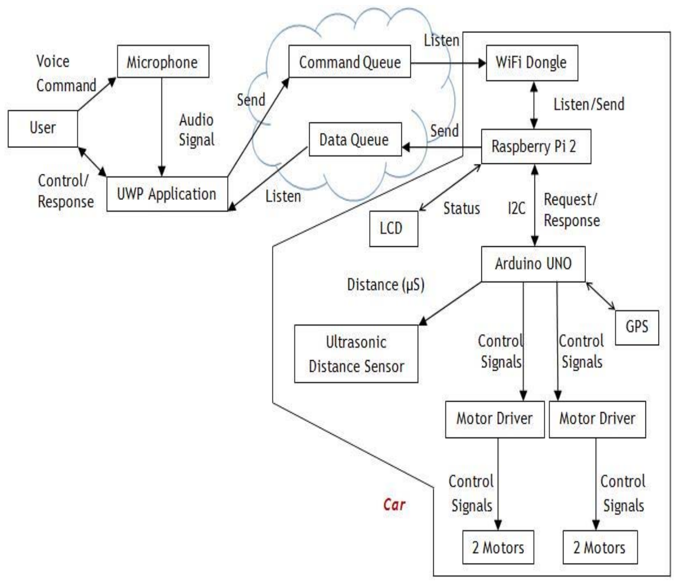

Fig. 1 System architecture

In this section the system workflow described in details. The working procedure is divided into seven major parts and they are as follows:

A. Sending command

There are two modes of sending command to the car: voice command or manual clicking of buttons visible in the user interface. The possible words or commands that might be spoken by users are listed in a XML grammar file. Users can also control the car directly from the interface of UWP application and send any command same as before.

B. Checks for command validation

In order to recognize speech, a Speech Recognizer object is created. A XML grammar file is fed to that object which uses this file to decode speech from signal after proper processing. On successful decoding the dedicated event handlers take care of the rest of the task. But on unsuccessful decoding the client is requested to generate any command from the set of valid commands. This request is in actual a message displayed on the user interface of the application.

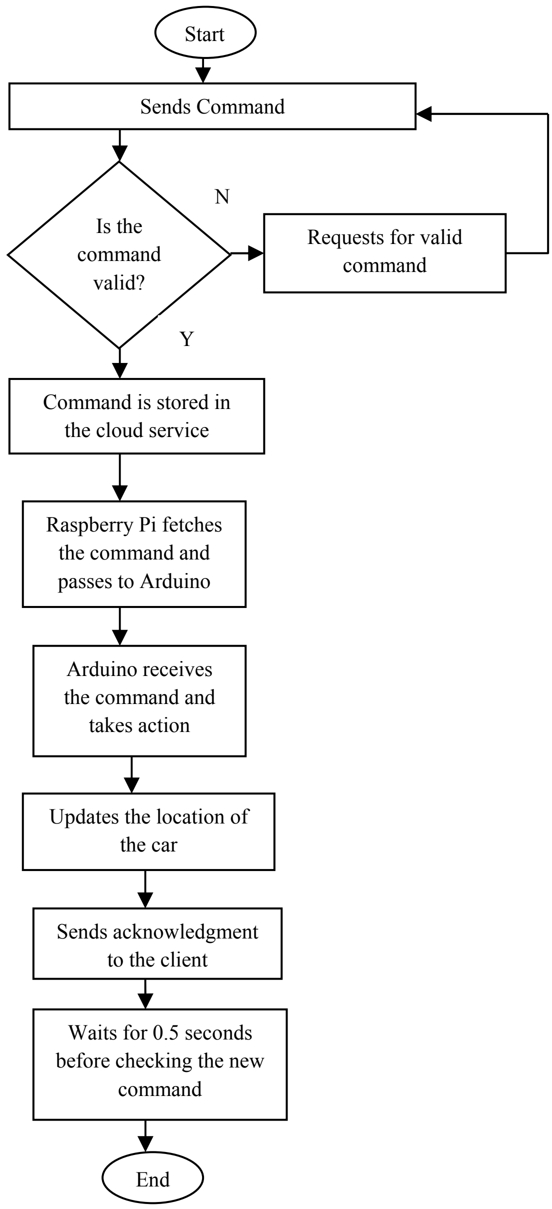

Fig. 2 Workflow of the Proposed System

C. Stores commands in a cloud service

The UWP application stores command in a Queue of the cloud service hosted by Azure IoT hub. Queue provides a well defined and flexible service to this system. As both car information and commands were needed to be transferred at the desired places or devices and at the same time, so two queues were used- one for data and another for command. The Raspberry Pi in the car listens to the *Command Queue* and it sends data to the *Data Queue*. On the other hand the UWP application in the controller end listens to the *Data Queue* and it sends command to the *Command Queue*. As both the end system- components need not be connected to the Azure IoT hub at the same time so it provides the system more guaranteed performance.

D. Raspberry Pi collects the command and passes to the Arduino

Raspberry Pi checks for commands and fetches them from the command queue of cloud service in every 0.5 seconds. Raspberry Pi and Arduino communicates with each other through the communication protocol called I^2C (Inter-Integrated Circuit). They have different unique addresses to identify them. There are basically three modes of command signals that the Arduino UNO receives from the Raspberry Pi. These are: 1) To send GPS sensor values acquired from the GPS, 2) To send the data received from the obstacle detector and 3) For maneuvering the car's direction of motion or state according to the command signal sent by the Raspberry Pi.

E. Arduino takes action according to the command

Based on the command received Arduino takes appropriate action. For example: acquiring GPS sensor value, acquiring obstacle distance sensor reading and maneuvering the car's direction of motion or state. The GPS sensor continuously pings for getting the actual location of the car. Arduino also pings the ultrasonic distance sensor for distance of obstacle before the car. Based on the commands, Arduino changes the direction and speed of the motors using the motor controllers. A total number of four motors considered here.

Table 1: Differential steering method

Left Motors	Right Motors	Outcome
Forward	Forward	Forward
Forward	Static	Left
Static	Forward	Right
Backward	Backward	Backward
Forward	Backward	Rotate Right
Backward	Forward	Rotate Left

F. Updates GPS position of the car

Whenever the Robotic Car is commanded to change its position, Arduino UNO polls the GPS sensor to get the updated GPS position and then when it is commanded to send the GPS position then this location is sent to the Data queue of the cloud service bus. This data is later received by the UWP application which updates the UI accordingly.

G. Sends acknowledgement to user

Whatever may be the command sent to the car, for every command there is a specific response which either represents that the command has been fulfilled or it has failed to fulfill the command. Usually the acknowledgement is represented as 1 or 0. This is a crucially important feature in case of systems like this which is based on the Internet of Things paradigm as without this feature client is unable to realize whether the system has actually performed in the desired way or not.

Figure 3 shows the real system of robotic car. In figure 4, screen shot of client application is shown. Users can control the car by continuous speech recognition system built for the voice commands or just using the moving direction indicator buttons in UWP. Any update or information regarding to the car is shown in the user application. Users can leave commands and the system will work by fetching the commands from the Queue in first- in first-out method.

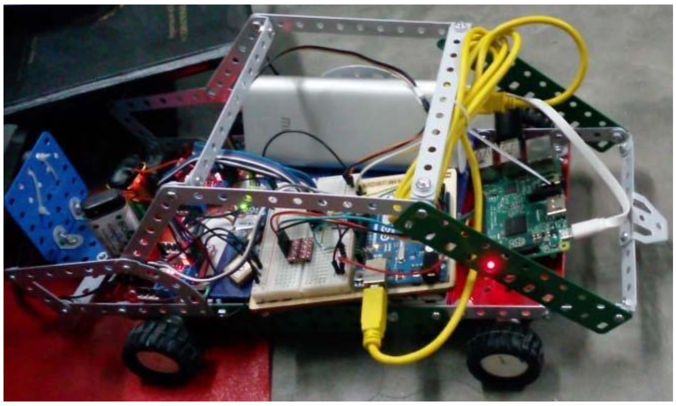

Fig. 3 Picture of robotic car

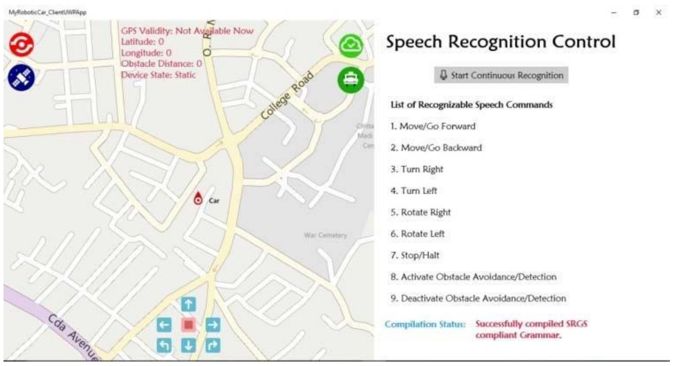

Fig. 4 UWP Application's User-Interface

III. PERFORMANCE EVALUATION

Table 2: Performance analysis

Distance (m)	Time (ms)			
	Commands received by car in	Stored in cloud	Perform actions in	Client gets updates
10	5 ms	5 ms	8ms	11 ms
20	5 ms	5 ms	9 ms	12 ms
35	5 ms	5 ms	8 ms	11 ms
45	6 ms	6 ms	10 ms	13 ms
60	6 ms	6 ms	10 ms	12 ms
95	6 ms	6 ms	8 ms	11 ms
100	7 ms	7 ms	10 ms	13 ms

For creating wireless communication, a wireless router with a broadband connection was used. The experiment was conducted in a normal environment with sound level ranging from 60dB to 80dB. From the experimental result it is seen that the signal of router reaches up to 100m. The results are based on all eight commands listed in the XML grammar file. Depending on the distance between the robotic car and the router, there are variations in time of receiving the commands. On an average it takes only 6 ms to receive a command. At the same time, commands are stored in the cloud service. According to the mode of commands, the Arduino take actions and users get updated with an average timing of 9 ms and 12 ms respectively. But the system is crucially dependent on the performance of router, the broadband connection and the WiFi dongle attached to the Raspberry Pi. The performance is comparatively better than previous research results of any motion controlling systems of robotic car. The working environment, surface of robotic car and signal availability are kept in consideration here.

IV. CONCLUSION AND FUTURE SCOPE

In this paper an efficient approach of multiple control system is incorporated with IoT. Controlling multiple devices in multiple ways makes causes more convenience in handling a system. The cloud service helps the system to reduce memory load. Stored messages are automatically removed after a certain amount of time. The performance results prove that if the incorporation is efficient enough, multiple controlling methods have less effect on time and performance compared to single way of control system.

Yet, the system has some limitations. No video surveillance system has been incorporated. The wireless range is too small. It can be efficient if GPRS, zigbee module is used for wireless medium. Including object detection method is one of the main future works that needs to be implemented.

REFERENCES

[1] V. Vujovi, M. Maksimovi, "Raspberry Pi as a Sensor Web node for home automation," *Comput. Electr. Eng.*, vol. 44, pp. 153–171, 2014.

[2] S. Jain, A. Vaibhav, and L. Goyal, "Raspberry Pi based Interactive Home Automation System through E-mail LED Switch," no. 2002, pp. 277–280, 2014.

[3] J. Sobota, R. Pĭsl, P. Balda, and M. Schlegel, "Raspberry pi and arduino boards in control education," *IFAC Proc. Vol.*, vol. 10, no. PART 1, pp. 7–12, 2013.

[4] M. S. Bin Bahrudin, R. A. Kassim, and N. Buniyamin, "Development of Fire alarm system using Raspberry Pi and Arduino Uno," *2013 Int. Conf. Electr. Electron. Syst. Eng. ICEESE 2013*, pp. 43–48, 2013.

[5] K. Krasovitskaya, E. Cherkashin, and S. Gorunchik, "Expert System for Structural Analysis of Electrocardiograms," *Int. Conf. Appl. Internet Inf. Technol.*, pp. 220–230, 2016.

[6] "Distributed Control System (DCS) Information | Engineering 360." [Online]. Available: http://www.globalspec.com/learnmore/networking_communication_equipment/networking_equipment/distributed_control_system_dcs.

[7] "Theory of Robot Control - Google Books." [Online]. Available: https://books.google.com.bd/books?hl=en&lr=&id=5NnUBwAAQBAJ&oi=fnd&pg=PR12&dq=robot+control+&ots=WRcW3pf_p&sig=hGTxIllfyxS082pmtHmmksqMvEg&redir_esc=y#v=onepage&q=robotcontrol&f=false.

[8] "Azure Service Bus | Microsoft Azure." [Online]. Available: https://azure.microsoft.com/enus/documentation/articles/service-bus-fundamentals-hybrid-solutions.

[9] "Arduino UltraSonic Range Finder: The best interfacing tutorial!" [Online]. Available: http://diyhacking.com/arduino-ultrasonic-range finder/.

[10] "How to Change the Direction of Rotation of a DC Motor? | StudyElectrical | Online Electrical Engineering Study Site." [Online]. Available: http://www.studyelectrical.com/2015/07/how-to-change-direction-of-dc-motor.html.

[11] X. Huang and L. Deng, "An overview of modern speech recognition", *Handb.Nat.Lang.*, pp.339–367, 2010.

[12] A. R. Krishna, G. S. Bala, A. Sastry, B. B. Sarma and G. S. Alia, "Design And Implementation Of A Robotic Arm Based On Haptic Technology," International Journal of Engineering Research and Applications (IJERA), vol. 2, no. 3, pp. 3098- 3103, 2012.

[13] A. Abdullah, O. Sidek, N. A. Amran, U. N. Za'bah, F. Nikmat, H. Jafar and M. A. Hadi, "Development of Wireless Sensor Network for Monitoring," 2012, International Conference on Advanced Computer Science and Information Systems.