Future Computing and Informatics Journal

Volume 3
Issue 2 *Vol 3,Issue 2,2018*

Article 15

2018

A low cost autonomous unmanned ground vehicle

Leckraj Nagowah
Department of Software and Information Systems, Faculty of Information, Communication and Digital Technologies, University of Mauritius, Réduit, Mauritius, l.nagowah@uom.ac.mu

Available online at www.sciencedirect.com

ScienceDirect

Future Computing and Informatics Journal 3 (2018) 304–320
http://www.journals.elsevier.com/future-computing-and-informatics-journal/

A low cost autonomous unmanned ground vehicle

Christopher Kwet Young Lam Loong Man [a], Yogesh Koonjul [a], Leckraj Nagowah [b,*]

[a] *Department of Information Communication Technology, Faculty of Information, Communication and Digital Technologies, University of Mauritius, Réduit, Mauritius*

[b] *Department of Software and Information Systems, Faculty of Information, Communication and Digital Technologies, University of Mauritius, Réduit, Mauritius*

Received 18 January 2018; accepted 18 October 2018
Available online 1 November 2018

Abstract

The aim of this project is to design and implement a low cost Autonomous Unmanned Ground Vehicle (AUGV), a vehicle that can be controlled remotely without an onboard human presence. The AUGV is also able to move autonomously while automatically detecting and avoiding obstacles. The vehicle also reads directions from QR codes, calculates the shortest path to its destination and autonomous move towards its final destination. A Raspberry Pi 3 has been used as the brain of the vehicle together with other components such as DC and Servo motors, Ultrasonic and Infrared sensors, webcam, batteries, power bank, motor controller and a smartphone. Python, Java and PHP have been used to implement the prototype which currently focusses on indoor navigation. There exists several potential practical applications of the UAGV such as an autonomous wheel chair for handicapped persons allowing them to move around autonomously without relying on any other persons. The idea can be extended to fit into the untapped indoor commercial market such as malls, hotels, banks, nursing homes, hospitals, offices, stores, schools, museums and many more.

Keywords: Autonomous; Unmanned vehicle; Robots; Sensors

1. Introduction

Robotics have helped humans greatly in achieving everyday tasks. Robots are designed to work in any environment and perform task on behalf of humans. They operate under real-world and real-time constraints where sensors and effectors with specific physical characteristics have to be controlled [1]. In many cases, those robots are controlled manually to move from one destination to another. However a number of studies have been carried out on autonomous robots leading to a whole panoply of potential applications of these autonomous robots [2–4]. An Unmanned Ground Vehicle (UGV) is a vehicle that operates while in contact with the ground and without an onboard human presence. This type of vehicle is controlled remotely. In the same context, an Autonomous Ground Vehicle (AGV) is a vehicle that is capable of sensing its environment and navigating from one place to another without any human intervention [5].

2. Motivation

There are many potential benefits of autonomous vehicles [6] such as saving lives as autonomous vehicles can significantly reduce the number of crashes, increased mobility for young or disabled persons, reducing cost of congestion since the occupants can perform other activities during traffic jams, improving land use as the autonomous vehicles could drop their passengers in urban areas and automatically drive to satellite car parks. A number of researchers are focusing on potential commercial applications of automated vehicles and include, but not limited to, autonomous wheelchair for elderly and disabled persons [7], autonomous unmanned aerial

* Corresponding author.
E-mail address: l.nagowah@uom.ac.mu (L. Nagowah).
Peer review under responsibility of Faculty of Computers and Information Technology, Future University in Egypt.

https://doi.org/10.1016/j.fcij.2018.10.001

vehicles for agricultural applications [8], autonomous underwater vehicles [9] and autonomous driving in urban environments [10].

While a number of autonomous vehicles and robots rely on complex sensors and hence are costly to implement, the following research questions arise: Firstly, is it possible to build a low cost unmanned ground vehicle? Secondly, can the unmanned ground vehicle exhibit reliable autonomous decision making to move from one place to another? Thirdly, can the vehicle detect obstacles, process this information to calculate the next best path to follow to reach its destination?

3. Methodology

To answer the above research questions, the methodology followed during the course of this research work has consisted of the following phases:

- Literature Review: An in-depth literature review has been carried out so as to review the related works and understand the techniques and algorithms used in the design of autonomous vehicles.
- Critical Analysis: The different techniques for objection detection and avoidance have been critically analysed.
- System Architecture: After having done a thorough study and analysis of related approaches, the low cost Autonomous Unmanned Ground Vehicle (AUGV) has been designed.
- Implementation and Testing: A prototype has been developed and was tested based on a number of scenarios.
- Evaluation: A proper evaluation of the prototype has been carried out.
- Lessons Learnt: A list of lessons learnt in the implementation of the low cost AUGV has been identified.

The rest of the paper is structured as follows: Literature review, similar systems and the techniques and algorithms used in autonomous vehicles have been summarized in section 4. A critical analysis of the different techniques and algorithms is presented in section 5. The system architecture of the proposed system is presented in section 6. Section 7 describes the implementation and testing of the autonomous vehicle. The system has been evaluated in section 8. Section 9 details a list of the lessons learnt in building this low cost AUGV. Finally, conclusion and future works are presented in section 10.

4. Literature review

This section gives an overview of existing similar systems and highlights the techniques and algorithms used in autonomous vehicles. These vehicles have been in the attention of a number of researchers for quite some time. Some of their works have been briefly summarized below.

Shahdib et al. [11] created an autonomous vehicle that used the range of distances collected by an ultrasonic sensor combined with the image captured by the camera for object detection and object size measurement. The angle of vision of the camera was already known and using the distance obtained from the ultrasonic sensor, the horizontal viewing length on 2D plane could be obtained. Combining that information with the pixels of the image, and using the same aspect ratio, the actual size of object was calculated which helped in the identification of the obstacle.

This vehicle of Dumbre et al. [12] aimed at enhancing security surveillance due to increasing number of terrorist attacks. It was controlled remotely via the internet and hence there was no limitation on the distance between the user and the vehicle. A webcam connected to a Raspberry Pi took a series of images which were then streamed over the internet to the web browser at the client side. Depending on what the user wanted to do, he could either input controls from keyboard or webpage directly and the robot would start moving.

A lot of deaths are caused by road accidents. The main objective of the project of Ujjainiya and Chakravarthi [13] was to decrease that number. A real time system was developed for analyzing the inroad conditions and densely populated areas to detect the presence of obstacles in the way of the moving vehicle. Whenever an obstacle was detected, the system immediately informed the driver, who could act accordingly.

Self-driving cars is the future. People love to travel, but they get frustrated when they get blocked in traffic. Hasdak [14] aimed at creating autonomous vehicles to solve traffic jams and also to decrease road accidents and delayed journeys to make peoples' life easier. An android phone, used for its GPS capabilities, was connected to a Raspberry Pi which controlled the motors. With the help of an ultrasonic sensor, the car was able to navigate from one point to another while avoiding any obstacles along the way.

McConnell et al. [15] aimed at using robots to transport items within a building. Their system used a USB camera and a Raspberry Pi to identify its waypoints to navigate from one point to another. The images captured from the webcam were used to identify QR codes at a distance of over 4.2 m away. The robot would first rotate on itself and capture a series of images. These images would be scanned for QR codes and once the software found one, it would orientate the robot to that specific direction.

This project of Pannu et al. [16] aimed at developing an outdoor monocular vision autonomous car using a Raspberry Pi. It used an HD webcam and ultrasonic sensors to capture data from the world. Using the data, the car was capable of reaching the destination safely and intelligently and thus avoiding risk of human error. The vehicle used the webcam for lane detection purposes such that the robot would be able to drive itself and make turns whenever necessary.

Hospital are huge and difficult to navigate. Regardless of how many times patients have visited the hospital, they always end up getting lost. That is why Shafer et al. [7] developed an autonomous wheelchair navigation system that was able to transport patients to their destinations with only the push of a few buttons. The system used the map of the hospital and RFID tags placed everywhere to provide orientation information to the wheelchair. The user input the room number he wished to go to, and the system would generate the optimal

path to the destination and then sent the commands to drive the wheelchair.

There exist a number of algorithms and techniques that can be used for detecting obstacles which is an important component of autonomous navigation. These algorithms and techniques are briefly surveyed below.

The ranged-based and appearance-based obstacle detection technique [11] uses a combination of an ultrasonic sensor for distance measurement and a camera to calculate the size of the object. A camera, just as a human eye has a fixed field of view. Everything the camera sees will be squeezed into the image captured. Depending on how far an object is from the camera, will surely impact on its pixel size on the image. The ultrasonic sensor calculates the distance of the obstacle and using that distance combined with the angle of view of the webcam and the image resolution, the obstacle size is calculated.

The Sobel Edge Detection operator is a discrete differentiation operator, computing the approximation of the gradient of the image intensity function [17]. It is separated in two, the x and y directions kernel that are both 3×3 size matrices. To check for edges, both matrices are applied to each pixel in the image. The gradient of the image intensity at each point is calculated, giving the direction of the largest possible increase from light to dark and the rate of change in that direction. The result therefore shows how the image changes and how likely it is that this part of the image represents an edge and its orientation [18]. When multiplying the matrices, the result will be positive or negative values. The range of values will then be stretched between 0 and 255, which is a grey scale image. The result will be an image with black on one side of the edge and white on the other side. Finally to obtain the final result, the values are combined to give the gradient magnitude, which results in the edges of the image.

The Canny Edge Detector is an edge detection operator that uses a multi-stage algorithm to localize and minimize multiple responses to a single edge [19]. The input of the Canny operator is the output obtained from Sobel operator. After the gradient of each pixel and direction of edge have been obtained, the next step is to relate the edge direction to a direction that can be traced in an image. For a pixel p, there are only four possible directions when describing the surrounding pixels. It can be in the horizontal direction, along the positive diagonal, in the vertical direction or along the negative diagonal. Finally non-maximum suppression is used to trace along the edge in the edge direction and remove all the unnecessary pixels alongside the edge that make the latter thick in size into just one pixel width. A thin line will be given as an output image.

The Infrared Sensor is a device that has an IR transmitter and receiver [20]. It emits an infrared light using the transmitter, and when an obstacle is very close to the sensor, the light will bounce off the object and into the receiver. Depending on the distance between the obstacle and the sensor, the reflected light will have a different intensity. The module has on board potentiometer that lets user adjust detection range [21]. However different objects of different color or material will have a different effect on the reflected light even if they are all placed at the same distance from the sensor. In addition, there is no way to calculate the distance of the obstacle from the sensor and also, since this sensor uses infrared, the heat from sunlight will have an impact on the detection of the sensor, thus the sensor is not very accurate to be used outdoors but can perform well in indoor environments.

The Ultrasonic Sensor is a device that can calculate the distance from an object by emitting a sound wave and measuring the time taken for it to bounce on the obstacle and come back [22,23]. The distance can be obtained using the formula in (1).

$$\text{distance} = (\text{speed of sound} * \text{time taken})/2 \quad (1)$$

5. Critical analysis

This section presents a critical analysis of the different techniques and algorithms that can be used for object detection.

The ranged-based and appearance-based obstacle algorithms might encounter difficulties if there are shiny floors, boundaries between carpets, or shadows. The algorithm makes use of a reference area, i.e. the bottom ten rows, which makes the system very adaptive. However, this approach requires the reference area to always be free of obstacles. Unfortunately, the reference area may not always be sufficiently representative for pixels higher up in the image which are observed at different angles.

Moreover Gradient-based algorithms such as the Sobel edge detector have a major drawback of being very sensitive to noise. The size of the kernel filter and coefficients are fixed and cannot be adapted to a given image. An adaptive edge-detection algorithm is necessary to provide a better solution that can adapt to the different levels to help distinguish valid image contents from visual artifacts introduced by noise.

Furthermore Canny edge detection algorithm is computationally more expensive compared to Sobel operator. However, the Canny edge detection algorithm performs better than Sobel under almost all scenarios. However in real life scenarios, Canny Edge detects all edges in the image but cannot differentiate between the foreground and the background of an image. Thus the results obtained might be very difficult to use for obstacle detection and avoidance.

Hardware sensors are very reliable to detect obstacles. The Infrared Sensor can detect obstacles easily but only if they are very close range. The Ultrasonic Sensor on the other hand can detect obstacles that are not in close range and can also calculate the distance of the obstacle from the sensor. This information can become really useful to define a threshold value such that if the distance is below that value, then the vehicle can assume that there is an obstacle on its path and performs alternative actions.

However it is also important to note that the sonar sensor cannot detect all obstacles. This is because some objects are shaped in such a way that the sound wave emitted bounces off the object but does not return back to the sensor. Small objects will also not reflect enough sound waves back to the sensor,

and others such as cloth might just absorb the sound wave such that there is no way for the sensor to detect them accurately. Similar to the infrared sensor, the ultrasonic sensor may not be very accurate when used outdoors but can perform quite well in indoor environments.

Table 1 below gives an overview of the advantages and disadvantages of using a specific technique for determining obstacles in the path of the autonomous vehicle.

6. System architecture

Fig. 1 below shows the high level design of the application for the Autonomous Unmanned Ground Vehicle (AUGV). In order to remotely control the AUGV, both the smartphone and the AUGV vehicle should be connected to an access point. From there, the mobile application is able to access the vehicle through its IP address and send actions to the vehicle. The AUGV will have a raspberry Pi inside the vehicle which will receive those instructions, perform the necessary actions and send feedback that will be received and displayed on the mobile application.

The proposed design will allow the AUGV to operate in two modes: firstly, the remote controlled mode, and secondly, the autonomous mode. In the remote controlled mode, the user will have complete control over the vehicle which will obey the instructions received through the mobile application. In the autonomous mode, the user will use the mobile application to select the source and destination of the AUGV. The AUGV will automatically compute the shortest path to its final destination. It will use sensors to move towards its destination. Using a camera, the AUGV shall automatically scan for QR codes to read the directions towards its final destination. A motor will rotate the camera in case the QR code is not in range. Based on the QR code, the AUGV will decide whether to continue to move forward, turn left or turn right. Obstacles will be detected through the use of sensors. When obtacles are detected, the AUGV will rotate 180°, hence moving in the opposite direction until it reaches another QR code which will help it to calculate another path to its destination.

6.1. The AUGV components

Fig. 2 shows the internal electronic components of the AUGV.

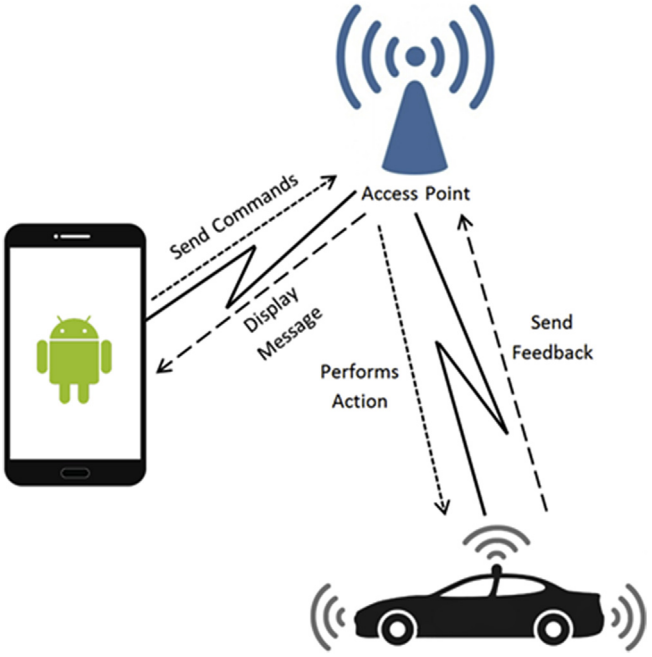

Fig. 1. Architectural design of AUGV.

The components necessary for the implementation of the AUGV are listed below:

- DC Motors:

Two DC Motors will be used. They are fast - the more the power supplied, the faster they will rotate. Two such motors shall be used, combined with a Steel Caster Round Ball, which will allow the vehicle to be balanced and to move in any direction.

- H-Bridge Controller (L298n):

A connection needs to be made between the two DC Motors and the Pi such that the latter can control the motors. The L298n module will allow this connection. The functionality of this module is that is able to invert the positive and negative terminal of power so as to rotate the DC Motors in clockwise or anti-clockwise direction. It can support up to 50 V of power.

Table 1
Advantages and disadvantages of existing obstacle detection techniques.

Method	Advantages	Disadvantages
Ranged based and appearance based obstacle algorithm	• Very quick to compute	• May be subject to false positives and false negatives
Sobel Edge detection	• Simple detection of edges and their orientation	• Sensitive to noise • May be inaccurate
Canny edge detection	• Improved signal to noise ratio with a better detection with noise	• Complex and time-consuming computation false zero crossing time
Infrared sensor	• Quick response time • Not affected by noise	• Accuracy affected by sunlight • Cannot detect distant objects
Ultrasonic sensor	• Quick response time • Detect object at distance	• May not detect all objects • Produces false reading

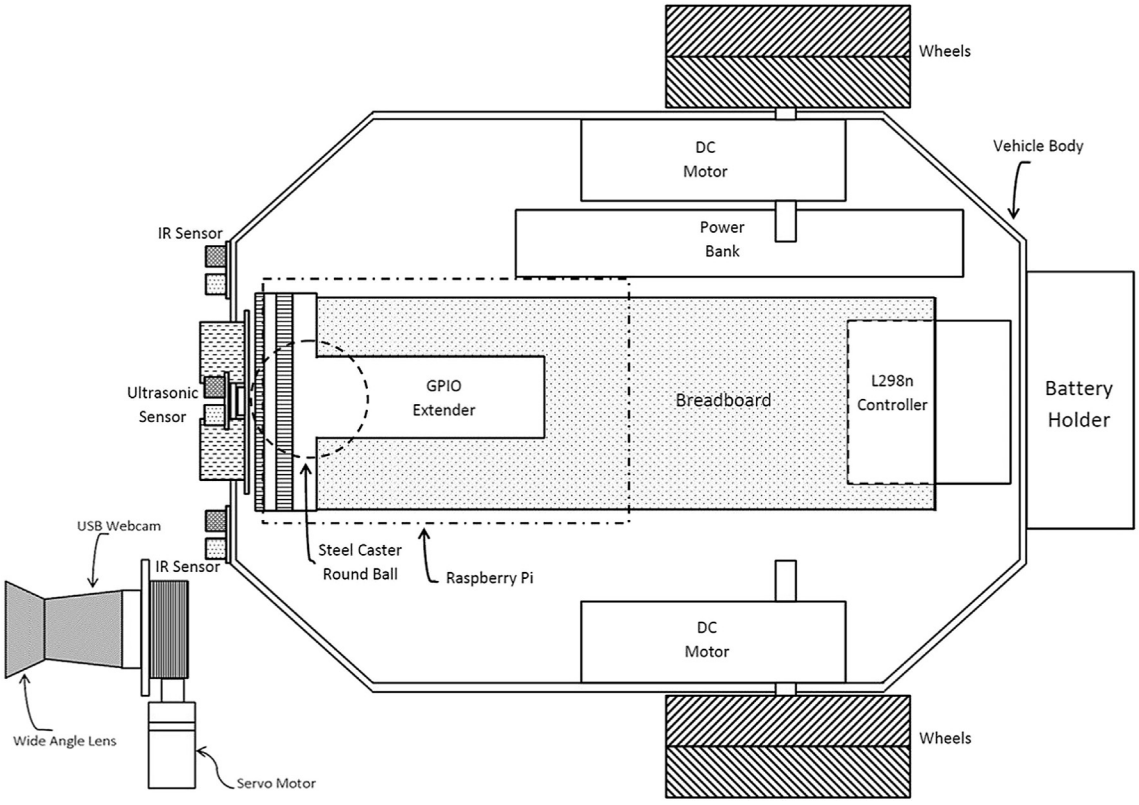

Fig. 2. Design of AUGV.

• Power Bank:

The Pi 3 will be powered using a 3400 mAh Power Bank. The latter is quite small and not too heavy which will help reducing the total weight of the vehicle. The Pi 3 is a very power efficient minicomputer, thus the power bank should last for quite some time on a single charge.

• GPIO Extender:

The Pi has 40 GPIO pin and since these are very small, it will be very difficult to distinguish between the different pins and it can happen that wrong connections are made. This GPIO extender takes all the pins on the Pi and extends it to the Breadboard for easier access and labels each of the pins.

• USB Webcam:

This camera will be used to detect and decrypt QR codes that will tell the AUGV in which direction to turn depending on the destination to reach. It will also be used to provide live streaming video feed such that the vehicle can be controlled remotely.

• IR Sensors (FC-51):

The IR transmitter emits infrared radiation and gets reflected and captured by the receiver. Black Line has the ability to absorb any kind of radiation, thus this property can be used to enable the vehicle to navigate from one place to another. Consequently, three IR sensors will be used; the left and right IR will be used to make the vehicle stay between the line and the middle IR to detect whether the vehicle has been lost, i.e. if the vehicle is not on any line for a given amount of time, it will be assumed to be lost.

• Ultrasonic Sensor (HC-SR04):

This module will be used to check whether there is an obstacle in front of the AUGV while the vehicle is moving. A threshold value will be set and if the distance obtained from the sensor is below this threshold, the vehicle will stop and perform designated tasks.

• Servo Motor (SG90):

USB Webcams do not provide very wide viewing angle. Thus there will be cases where the part of the QR code will not be found inside the webcam's field of view. Thus the Servo Motor will be used to rotate the webcam up and down slowly to get the whole QR code in frame.

Fig. 3 shows the different Raspberry Pi's GPIO pins that will be used to connect and control the different components of the AUGV.

6.2. AUGV flowchart

Fig. 4 below shows the different steps involved to enable the vehicle to travel from a source to a destination

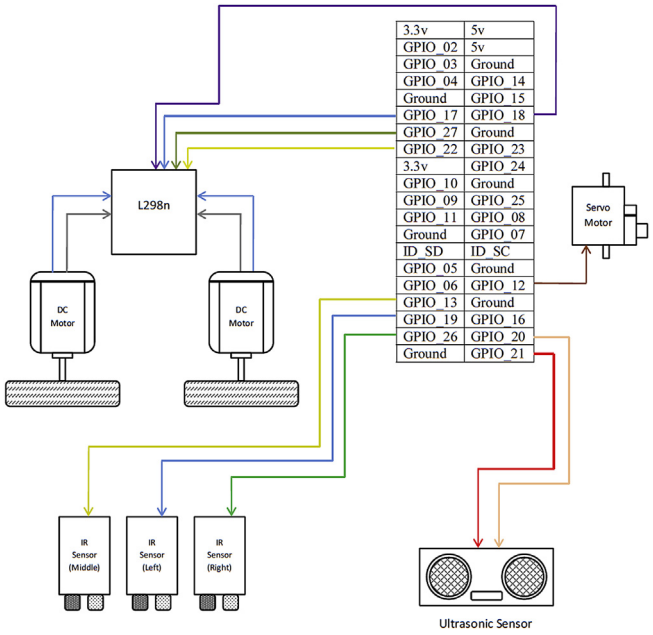

Fig. 3. Prototype GPIO layout.

autonomously. The user manually inputs the source and the destination and the script computes the shortest path to reach the destination. The vehicle will follow the line using the IR sensors and at any intersection, it will scan and decode the QR Code. Depending on the destination to reach, the vehicle will know in which direction to turn from the information obtained in the QR code. If an obstacle is detected, the script will automatically compute another path from its current location to the same destination and will take that other route.

There are several python scripts that will created to achieve autonomous navigation.

1. Script to calculate road path from source to destination.
2. Script to follow line
3. Script to read QR code and perform direction change.

All those scripts will be called inside a bash script as in Fig. 5 below. When the user inputs the source and destination on the mobile application, the bash script will be executed with the two variables.

The bash script will first call the road calculation python script with the two variables. The latter will already have the graph of the map together with their respective distances (in millimeters), and will calculate the shortest path using dijskra's algorithm [24]. An example of such a graph is illustrated in Fig. 6. Finally the results will be stored in a road path text file.

The bash script will then check if the file is empty, where empty will mean that the destination has been reached. While the file is not empty, the latter will create a loop calling the line following algorithm and the QR script successively.

The line following algorithm will follow the line using the left and right IR sensors. If the left IR has detected a black line, the vehicle will rotate left, if the right IR has detected the black line, the vehicle will rotate right, else the vehicle will move forward. While the vehicle is moving, the ultrasonic sensor will also be used to calculate distance in front of the vehicle. If the distance is less than a threshold value, i.e. an obstacle has been detected, the vehicle will rotate 180° and calculate a new road path from its current location to the destination. The algorithm will then overwrite the road path text file with the new path obtained. When both sensors have detected a black line, it will mean that there is an intersection with a QR code ahead. The AUGV will then stop and the script will exit.

After the line following script has terminated. The QR scan script will be launched by the bash script. The QR scan script will retrieve the path from the first line of the file, and then activate the webcam. The webcam will start rotating downwards using the servo motor until the QR is in frame. If the servo has reached the maximum downward position, it will be reinitialized to the maximum top position, and the latter can start to rotate downwards step by step again. Depending on the value of source and destination variable obtained, the algorithm will determine in which direction to turn from the QR code decoded. The vehicle will move in the appropriate direction and update the road path text file by removing the first line of text. The script will then exit.

The bash script will continue to loop between the two scripts 1 and 2 until the road path text file is empty. When the file is empty, the bash script will exit, and the vehicle will already be at the destination.

7. Prototype implementation and testing

The list below shows the set of programming tools and languages as well as libraries that were used to implement the system:

1. Android Studio – version 2.1.3
2. Python 3 (Libraries - OpenCV2, zbarlight, Rpi.GPIO)
3. Bash Shell
4. PHP5
5. Websocketd
6. MJPG-Streamer
7. Wiring Pi

Fig. 7 shows the AUGV prototype, built from the components listed in Table 2. The total cost of building the AUGV amounted to approximately one hundred US dollars.

Once the prototype has been built, a number of test scenarios were devised in order to ensure that the AUGV performs as expected. The 6 test scenarios are as follows:

1. Manually controlling the AUGV
2. Autonomously follow a route using IR sensors
3. Autonomously follow a route using IR sensors and with obstacle detection
4. AUGV reading directions from QR code
5. AUGV reading directions from QR code and with obstacle detection
6. AUGV Lost

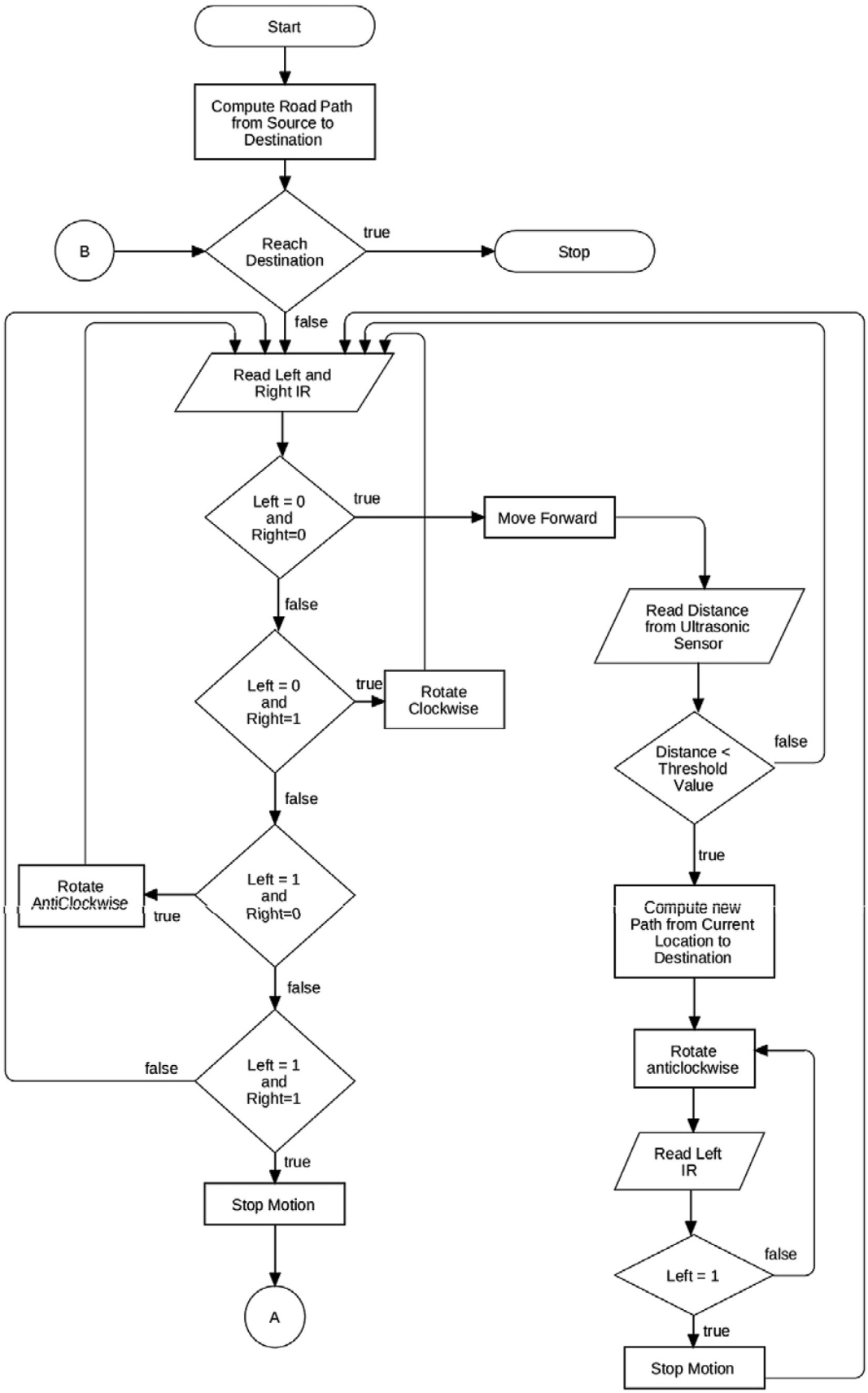

Fig. 4 AUGV flowchart.

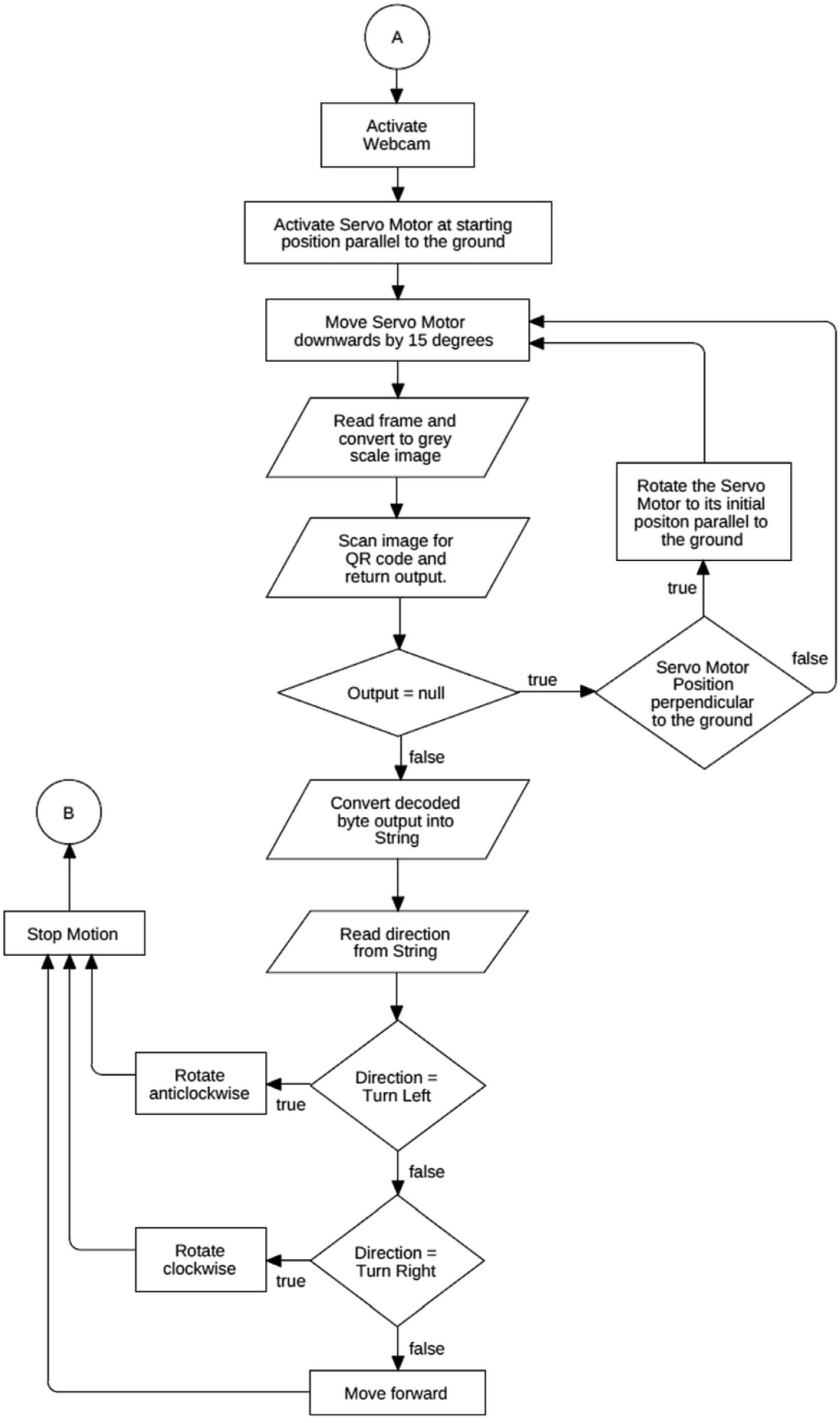

Fig. 4 (*continued*)

```
Compute road path from source to destination
Insert Results in text file
While text file is not empty
  Execute script to Follow Line
  Execute script to Scan QR Code
End While
```

Fig. 5. Autonomous bash script.

```
graph ={'br2': {'kit':945, 'gue':545},
        'gue': {'br2':545, 'off':645, 'gym':540},
        'gym': {'gue':540},
        'gam': {'liv':545, 'kit':540, 'off':300},
        'kit': {'br2':945, 'gam':540},
        'liv': {'gam':545},
        'off': {'gam':300, 'gue':645} }
```

Fig. 6. Graph of autonomous map.

7.1. Manually controlling the AUGV

The Raspberry Pi hosts a webserver that is ready to listen to incoming HTTP request and respond to them promptly. The process for remotely controlling the vehicle is as follows:

1. Press and hold directional button on the mobile application
2. Mobile application sends HTTP request to the Pi Apache Server
3. Pi receives request and executes the appropriate python script that will turn the wheels.
4. Vehicle starts moving in the appropriate direction.

When the button previously pressed is released, another python script is executed to stop the motion of the car.

Fig. 8 shows the mobile application which allows the user to remotely control the vehicle over a distance of 10 m which is the range of the Pi's WiFi component. The webcam live streaming can be activated and deactivated, when required, using the Camera Toggle Button.

Table 2
AUGV components and pricing.

#	Component	Price ($)
1	Raspberry PI 3 Model B A1.2 GHz 64-bit quad-core ARMv8 CPU, 1 GB RAM	41.50
	Pi Case	2.90
2	DC Motors with wheels (x2) - unbranded	4.50
3	L298n H-Bridge Controller	1.90
4	3400 mAh Power Bank	10.00
5	40 pin GPIO Extender	3.50
6	USB Webcam – A4Tech PK-30F	15.00
7	FC-51 IR Sensors (x 5)	2.60
8	HC-SR04 Ultrasonic Sensor	0.80
9	SG90 Servo Motor	1.25
10	Battery Holder	1.00
11	Jumper Cables	3.50
12	Ice Cream container	1.00
13	Batteries	8.00
Total		97.45

When any directional button is pressed, the vehicle starts to moving immediately. It continues to move in the desired direction as long as the user holds down the button. If the user presses the forward or backward and quickly releases it, the vehicle moves in the respective direction for about 10 cm. As soon as the user releases the button the vehicle stops all actions, but still continue to move due to momentum. If the user presses the left or right button and quickly releases it, the vehicle will rotate clockwise or anticlockwise approximately 90°. Hence it was observed that rotating the vehicle for less than 90° was difficult. The user had to rotate more than 360° until the desired angle is reached. E.g. a rotation of 45° was achieved by rotating the vehicle 405°.

Since the vehicle moves very fast, there is a slight delay of approximately 1 s when the user starts controlling the vehicle and when it is being shown on the live stream which is a very acceptable case scenario. Fig. 9 shows the user controlling the AUGV via the mobile application.

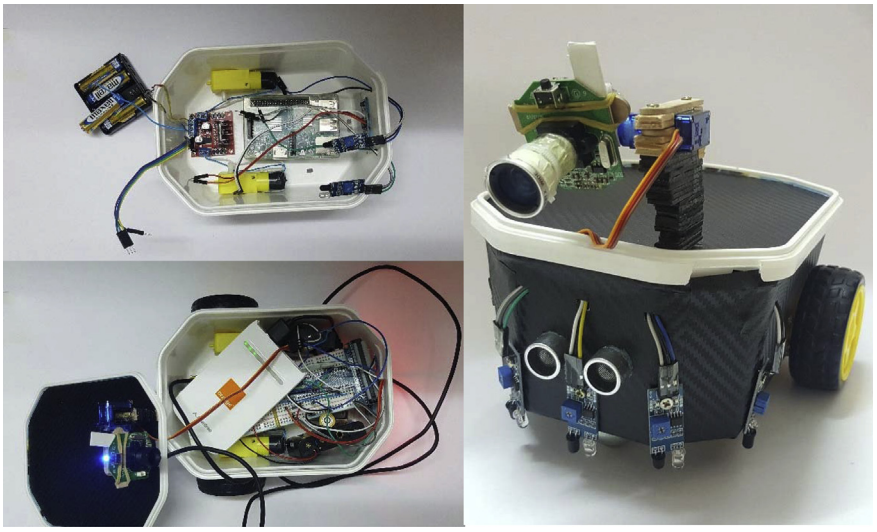

Fig. 7. AUGV prototype.

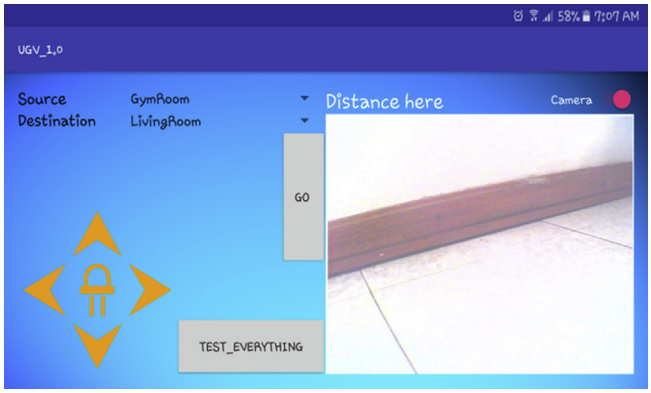

Fig. 8. Mobile application layout.

Fig. 9. Controlling the AUGV from the mobile application.

7.2. Autonomously follow a route using IR sensors

The vehicle follows a route using IR sensors. Fig. 10 shows an example of a route made from black electrical adhesive tape taped on the floor.

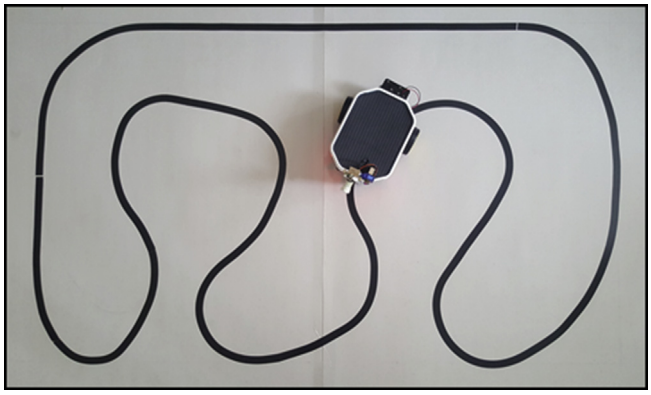

Fig. 10. IR sensor map.

If the right IR detects a black line, the vehicle rotates right, and if the left IR detects a black line, the vehicle rotates left, else the vehicle moves forward. These simple steps are used to make the AUGV follow any route.

If the motors' power are used at hundred percent, the vehicle moves so fast that even if the IR sensors have detected the line and instructed the vehicle to turn in a specific direction, the momentum of the vehicle is so high that the vehicle continues to move in its previous direction even when the motors are off. Thus, the vehicle gets lost on the path leaving the route shown by the electrical adhesive tape, as shown in Fig. 11.

For the vehicle to follow the line perfectly, the power of the motors had to be reduced to about 30%. Fig. 12 shows that, with this adjustment, the AUGV autonomously follows the route without it getting off the track.

However, as the AUGV is being operated, the energy level of its batteries decreases, resulting in the vehicle having less power that needed. Thus, sometimes the vehicle will need a little kick back to start moving and at other times, the batteries will be so low that the vehicle will not move at all.

7.3. Autonomously follow a route using IR sensors and with obstacle detection

The vehicle perfectly follows the route using IR sensors. At the same time, the Ultrasonic sensor scans the area in front of the vehicle for obstacles. If an obstacle is detected, the vehicle rotates 180° and continues to follow the path.

For the reading from the ultrasonic sensor to be as accurate as possible, there must be a small time interval between the transmissions of the signal. Else, the readings obtained are not reliable. However when this ultrasonic code is combined with the line following algorithm, this small time interval disrupts the line following algorithm resulting in the vehicle to sometimes getting lost. As a result, the small time sleep interval was omitted which made the ultrasonic sensors produce false reading sometimes. So, the distance is checked twice before instructing the vehicle to turn 180°.

When it is turning on itself (Fig. 13), it uses the IR sensors to detect the line and make the vehicle stop rotating any further than required and then the line following algorithm continues to execute until another obstacle is detected and the action is repeated again.

However, even with the double checking, it was observed that, in some rare instances, the vehicle seemed to detect an obstacle in front of it even if there were none. Thus, it might happen that the vehicle turns 180° even if there are actually no obstacle on its path. Fig. 14 shows an instance when the vehicle rotates even if no obtacles were found.

7.4. AUGV reading directions from QR code

If the AUGV is placed in front of a QR code, the webcam detects the QR code, interprets the code and instructs the vehicle to turn in the appropriate direction to reach its destination.

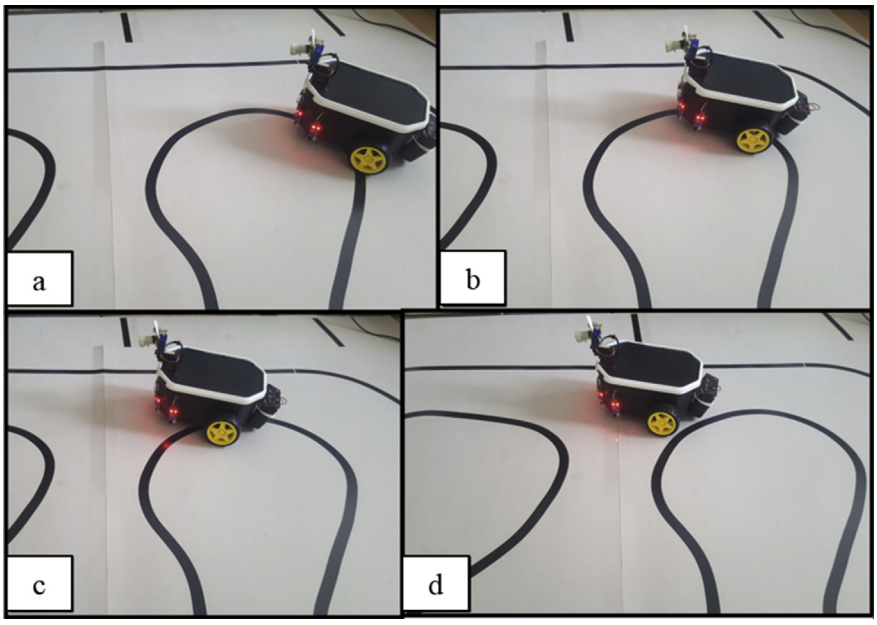

Fig. 11. Vehicle Follow Line and get Lost.

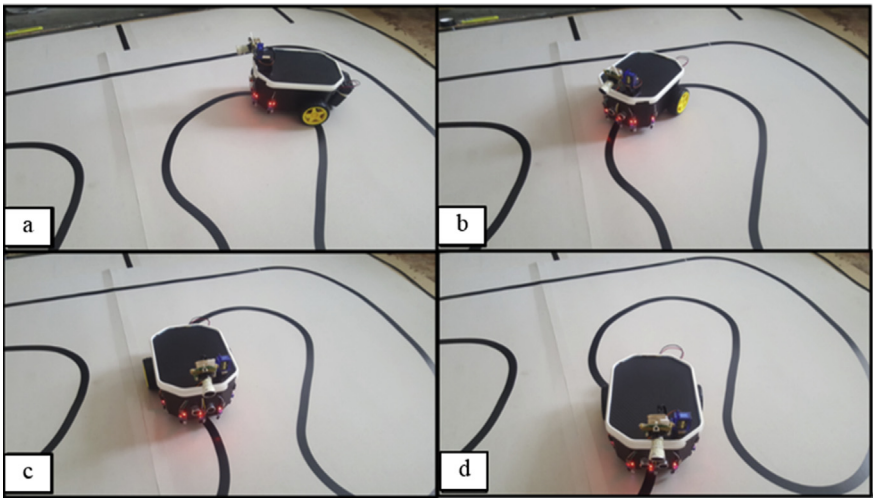

Fig. 12. Vehicle Follow Line without getting lost.

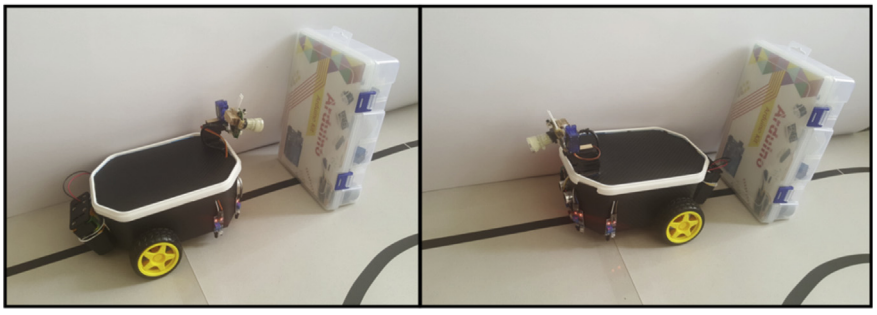

Fig. 13. Vehicle detects Obstacle and Turn 180 Degrees.

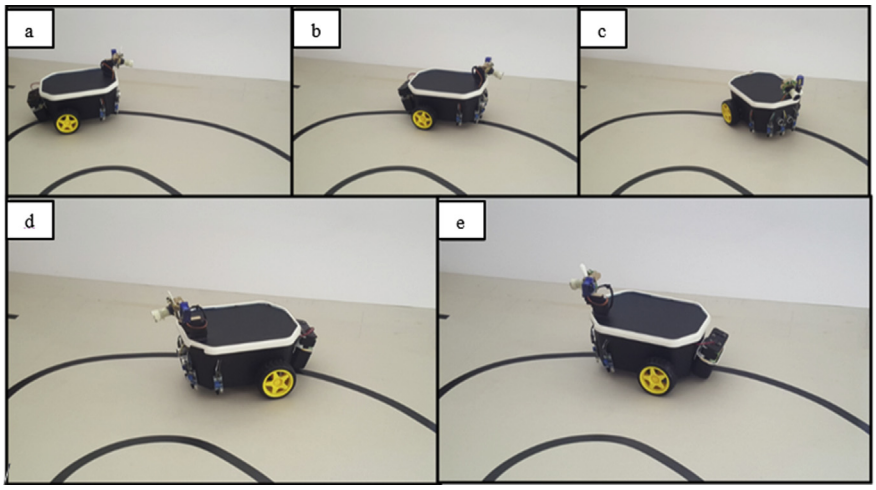

Fig. 14. AUGV detects no obstacles but still turns 180°.

If the vehicle must turn left or right, the motors moves forward for 0.5 s and then rotates in the respective direction until one of the IR sensors detect a line, and the vehicle stops its turning motion, then the line following algorithm takes over making the vehicle follow the line until another QR code is found. Fig. 15 shows the AUGV turning right at an intersection.

However, the time (0.5 s) that the motors moves forward has been hardcoded. Due to the inconsistencies of the power of the batteries and the motors, the perfect timing of the motors to move forward is very difficult to program.

7.5. AUGV reading directions from QR code and with obstacle detection

In this scenario, the AUGV travels from a source to a destination as shown on the map in Fig. 16. The source and destination have been entered through the mobile application.

When the script is launched, the program first calculates the shortest path from the source to the destination and saves this information into a text file. Each time, the vehicle reaches a

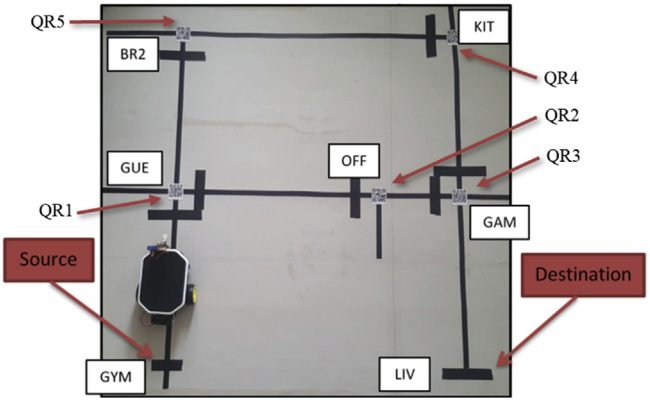

Fig. 16. Autonomous map.

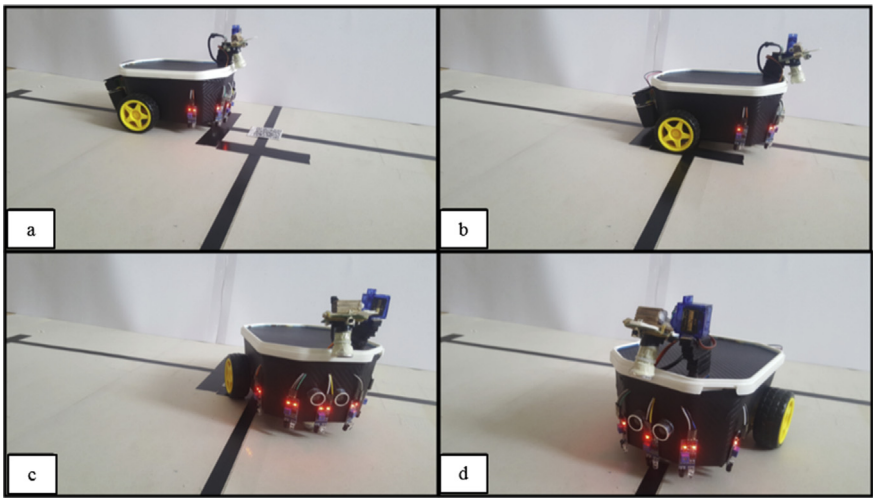

Fig. 15. AUGV making a turn at Road Intersection.

QR code; it scans the code and checks whether it is in the right location. Then the program reads the first road from the text file, and checks the QR string for instructions on where it should go next. Table 3 below shows the paths on the QR code based on the map in Fig. 16.

In this scenario, the AUGV has to move from the Gym Room to the Living Room. The AUGV automatically calculates the shortest path in terms of nodes.

Source: Gym Room
Destination: Living Room
Path: [1]-[7]-[11]

The vehicle starts to move from node to node. If there are no obstacles along the way, the vehicle reaches the destination using the shortest path as shown in Fig. 17.

Table 3
Paths and instructions on QR codes.

QR Number		Source	Destination	Instruction
1	[1]	Gym Room	Office	Turn Right
	[2]	Gym Room	Bedroom 2	Forward
	[3]	Bedroom 2	Office	Turn Left
	[4]	Bedroom 2	Gym Room	Forward
	[5]	Office	Bedroom 2	Turn Right
	[6]	Office	Gym Room	Turn Left
2	[7]	Guest Room	Game Room	Forward
	[8]	Guest Room	Guest Room	Forward
3	[9]	Living Room	Kitchen	Forward
	[10]	Living Room	Office	Turn Left
	[11]	Office	Living Room	Turn Right
	[12]	Office	Kitchen	Turn Left
	[13]	Kitchen	Living Room	Forward
	[14]	Kitchen	Office	Turn Right
4	[15]	Bedroom 2	Game Room	Turn Right
	[16]	Game Room	Bedroom 2	Turn Left
5	[17]	Guest Room	Kitchen	Turn Right
	[18]	Kitchen	Guest Room	Turn Left

If the vehicle detects an obstacle on its way to its destination, the vehicle rotates 180° and takes another path, as shown in Fig. 18.

Obstacle detected at: Office
New Source: Office
Destination: Living Room
New Path: [5]-[17]-[15]-[13]

7.6. AUGV lost

At some point in time especially when the power level of the batteries are high, the AUGV may be lost, i.e. it deviates from a route. If this is the case, the AUGV automatically notices that it is not following any line (Fig. 19), it stops all actions and sends a pop up message to the mobile application (Fig. 20) informing the user that the vehicle has been lost and that the user should manually take control of the vehicle.

8. Evaluation

The low cost AUGV Project has been evaluated against the seven existing similar systems, named project 1 to 7, based on the criteria listed in Table 4.

Project 1: Object Detection and Object Size Measurement for Autonomous Mobile Robot using Sensor [11].

1. Does not incorporate any kind of remote vehicle control capability.
2. It uses distance obtained from ultrasonic sensor when an obstacle is detected to perform other computations.
3. Vehicle does support a webcam which is used to perform some computation, but it does not offer proper live streaming capability.
4. Does not support any kind of autonomous driving, this vehicle's main purpose is to calculate the size of an obstacle in front of it.

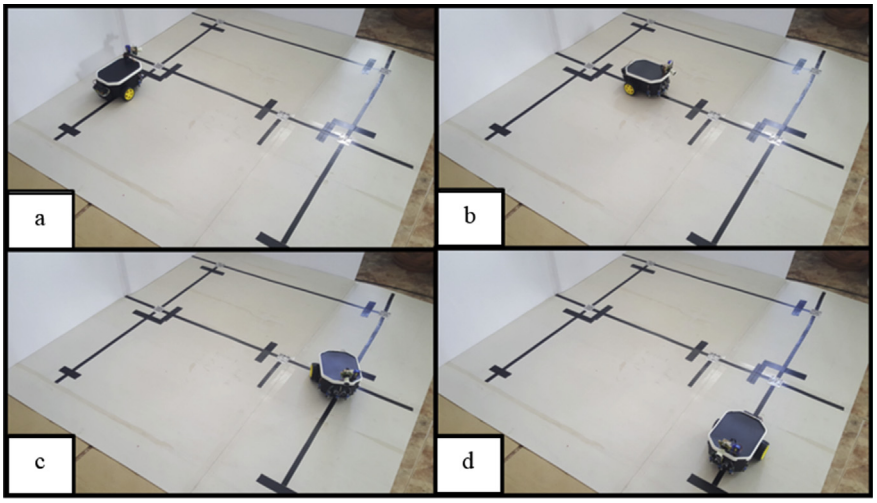

Fig. 17. AUGV taking shortest path to destination.

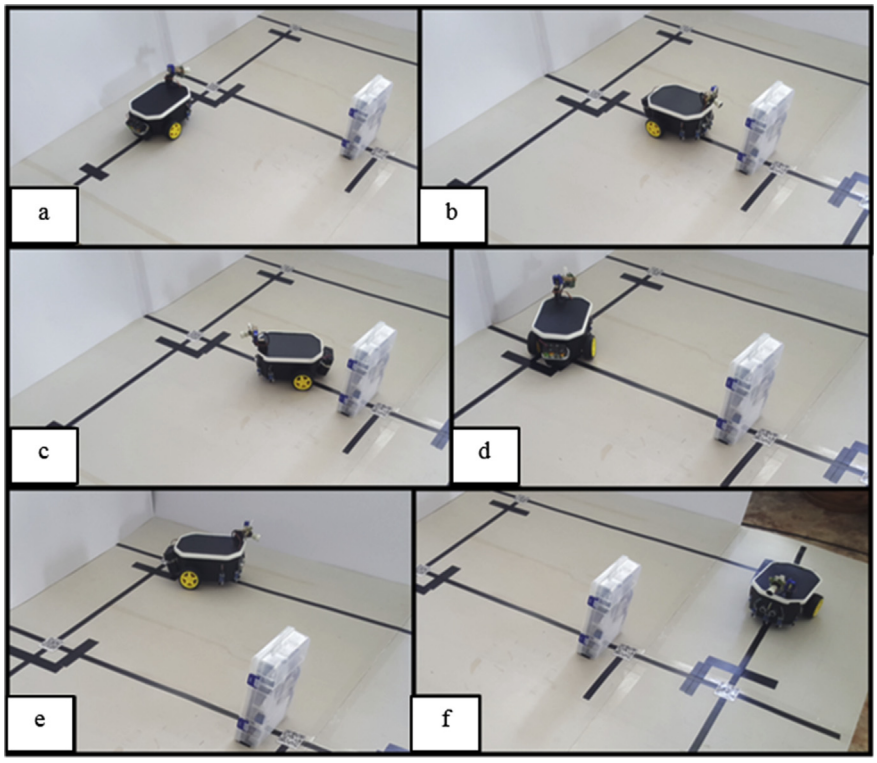

Fig. 18. AUGV detects obstacle and takes alternative route.

Project 2: Robotic Vehicle Control using Internet via Webpage and Keyboard [12].

1. Yes this vehicle can be controlled via the internet.
2. No obstacle detection functionality was implemented.
3. Yes vehicle offers live streaming via a webpage on the internet.
4. No autonomous driving functionality.

Project 3: Raspberry-Pi based Cost Effective Vehicle Collision Avoidance System using Image Processing [13].

1. This vehicle cannot be controlled from an external device.
2. It uses a webcam to detect obstacles, but does not give the actual distance of the vehicle from the obstacle.
3. Even though the vehicle supports a webcam, live streaming has not been implemented.

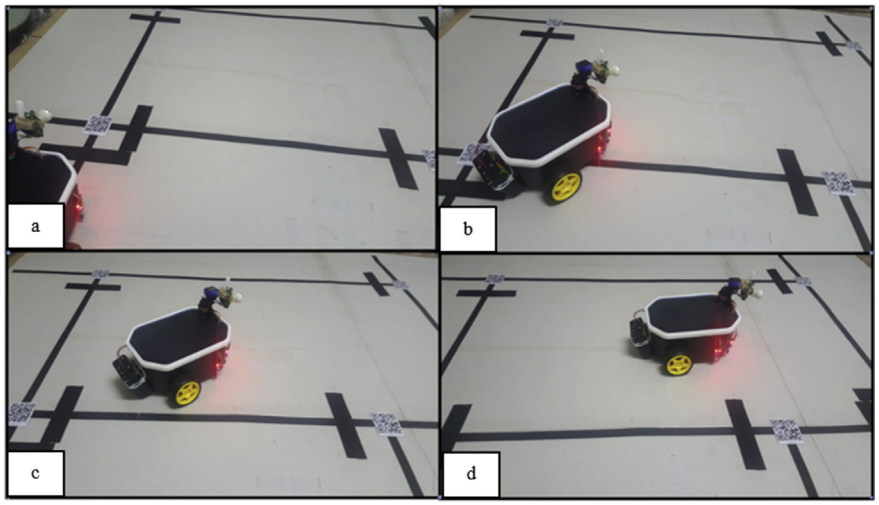

Fig. 19. Vehicle no longer following the path.

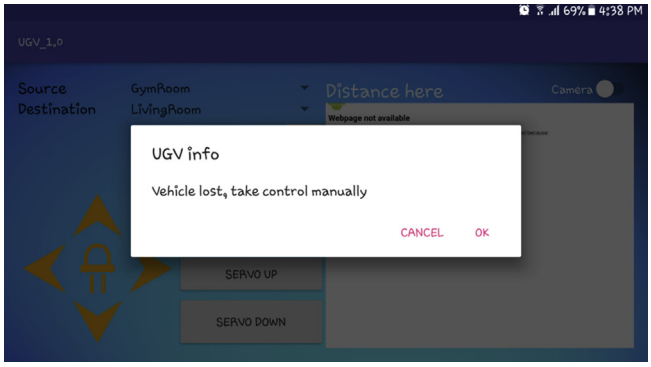

Fig. 20. Pop Up Message obtained when Vehicle is Lost.

Table 4
Evaluation criteria table.

Criteria Number	Criteria
1	Control Vehicle from External Device
2	Obstacle Detection
3	Live Video Computation and Streaming Capability
4	Autonomous Driving Functionality

4. No autonomous driving, the prototype's aim is to alert the user whenever an obstacle is detected.

Project 4: Programming a Self-Driving Car [14].

1. This vehicle cannot be controlled from an external device.
2. It uses an ultrasonic sensor for obstacle detection.
3. It does not support a webcam.
4. Yes the vehicle does perform outdoor autonomous driving; it uses the GPS functionality of a smartphone connected to the internet to be able to drive from source to destination. If ever an obstacle is detected, the vehicle will deviate it, and continue its way.

Project 5: Design of an Autonomous Robot for Indoor Navigation [15].

1. This vehicle cannot be controlled from an external device.
2. No obstacle detection functionality was implemented.
3. No live streaming functionality, but the vehicle does use the webcam to perform QR code detection and decryption.
4. This vehicle is able to move from one place to another. Depending on the destination to reach, the vehicle determine its route by scanning the whole room it is located in and search for QR codes which will tell the vehicle in which direction to go next.

Project 6: Design and Implementation of an Autonomous Car using Raspberry Pi [16].

1. The vehicle can be controlled from an external device.
2. The vehicle uses an ultrasonic sensor for obstacle detection.
3. The vehicle uses a webcam to find turns in road and react accordingly, but does not provide any streaming functionality.
4. The vehicle is able to drive autonomously using the information obtained from the webcam scanning the environment surrounding it.

Project 7: Robotics Based Autonomous Wheelchair Navigation [7].

1. This vehicle cannot be controlled from an external device.
2. No obstacle detection functionality was implemented.
3. No live streaming functionality and no live video computation.
4. The vehicle is able to travel autonomously using a static map and RFID tags as a guide to travel from current location to patient's room number.

AUGV Project.

1. This vehicle can be controlled externally by a mobile application.
2. An ultrasonic sensor is used for obstacle detection and avoidance.
3. The webcam is used to either provide for live streaming capability or used to scan and decrypt QR codes.
4. This vehicle exhibits indoor autonomous driving by following a line drawn on the floor. At each intersection, a

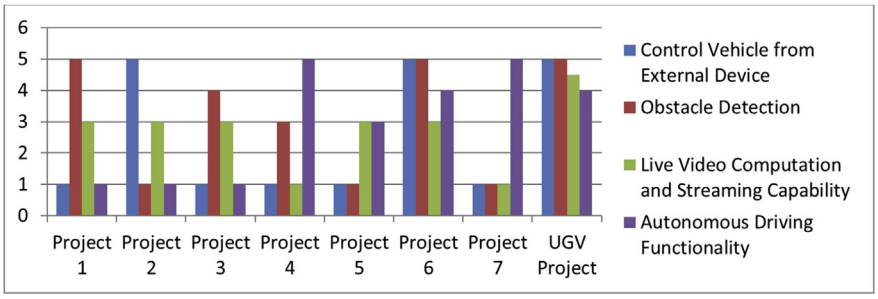

Fig. 21. Evaluation chart of AUGV project.

QR code is read and depending on destination to reach, the vehicle will turn in the appropriate direction.

Fig. 21 summarizes the evaluation of the low cost AUGV project with some related works. As it can be observed from the graph, our AUGV is quite interesting in an indoor environment.

9. Lessons learnt

Based on the experience in building the low cost Autonomous Unmanned Ground Vehicle (AUGV), this section details the lessons learnt as a list of necessary electronic components and important functions and features that need to be implemented if ever researchers need to build a low cost AUGV.

The electronic components required to build a low cost AUGV are:

1. A single board computer, e.g. Raspberry Pi, Arduino, BeagleBone etc… to act as the brain of the AUGV.
2. DC Motors to allow the vehicle to move swiftly. Stepper motors should be avoided as these would require micro-stepping in order for the AUGV to move smoothly.
3. A Power Bank to power most of the electronic components. Batteries will also required to power the DC motors.
4. GPIO Extender to take the pins on the Pi and extend it to the Breadboard for easier access.
5. USB Webcam for live video streaming and scanning QR codes.
6. Servo Motor which when connected to the webcam, can be used to rotate the webcam so that it is able to capture several views of the environment.
7. Infrared Sensors to ensure that the AUGV is following the route accordingly.
8. Ultrasonic Sensors to allow the AUGV to detect and avoid obstacles.

Functions and features that are vital for an AUGV are listed below:

1. To activate the proper GPIO pins in order to drive the motors in a specific direction.
2. To reduce the power of the motors to control the speed at which the vehicle is moving.
3. To activate the ultrasonic sensor, retrieve the data and convert the data into distance.
4. To retrieve data from the IR sensors, interpret the information and make the vehicle react accordingly.
5. To activate the webcam and stream live video.
6. To rotate the webcam through a servo motor so that the camera can detect and decode a QR code.
7. To interpret information from a QR code and make the vehicle move in the indicated direction.
8. To calculate the shortest path and directions that the vehicle must take in order to arrive at its destination.
9. To make the vehicle rotate whenever it encounters an obstacle.
10. To calculate alternative routes and directions if ever the vehicle has encountered obstacles on its path.
11. Listener function on the mobile application to activate the proper script to make the vehicle move in the direction indicated by the user.
12. Function on the mobile application to activate and deactivate the live stream video.

10. Conclusion

In this paper, a low cost Autonomous Unmanned Ground Vehicle (AUGV) has been implemented using a Raspberry Pi, DC motors, a webcam, infrared, servo and ultrasonic sensors. The AUGV operates in two modes: the remote controlled mode, and the autonomous mode. In the remote controlled mode, the user has complete control over the vehicle and move the vehicle similar to a remote controlled car with live video streaming on a mobile application. In the autonomous mode, the mobile application is used to select the source and destination of the AUGV. The AUGV automatically computes the shortest path to its final destination. It uses the infrared sensors to move towards its destination by following routes sketched using a black adhesive tape on the floor. The AUGV follows both straight line and curves by scanning the information obtained from the infrared sensors. The AUGV automatically reads the directions towards its final destination with the help of a webcam which scans for QR codes whenever required. A servo motor rotates the webcam in case the QR code is not in range of the webcam. Based on the QR code, the AUGV decides whether to continue to move forward, turn left or turn right. Obstacles are detected through the use of ultrasonic sensors. When detecting obtacles, the AUGV rotates 180° and move in the opposite direction until it reaches another QR code which will help it to calculate another path to its destination.

The total cost is approximately one hundred US dollars which is, in our opinion, quite cheap for building an autonomous unmanned ground vehicle. At the end of this project, the three research questions presented in section 2 can be answered: Yes, it is possible to build a low cost unmanned ground vehicle; Yes, unmanned ground vehicle can exhibit reliable autonomous decision making while moving from one place to another; Yes, the vehicle can detect obstacles, process this information and calculate the next best path to follow to reach its destination.

While some minor problems were encountered in this project, the work can be further enhanced by installing an orientation sensor that will enable the vehicle rotate to the nearest degree as instructed in the QR code and the vehicle will always rotate the right amount irrespective of the power level. The batteries used in this project had a short lifetime. Lithium polymer batteries can be used to improve the low power problem. An additional servo motor can be installed in the AUGV, together with the existing servo to allow horizontal tilting. Also, algorithms in the computer vision field can be used to enhance the system by scanning the images received,

from the webcam, for object detection and avoidance and for autonomous navigation.

References

[1] Kramer J, Scheutz M. Development environments for autonomous mobile robots: a survey. Aut Robots 2007;22(2):101−32.
[2] Van Henten EJ, Hemming J, Van Tuijl BAJ, Kornet JG, Meuleman J, Bontsema J, et al. An autonomous robot for harvesting cucumbers in greenhouses. Aut Robots 2002;13(3):241−58.
[3] Meier L, Tanskanen P, Heng L, Lee GH, Fraundorfer F, Pollefeys M. PIXHAWK: A micro aerial vehicle design for autonomous flight using onboard computer vision. Aut Robots 2012;33(1−2):21−39.
[4] Srinivasa SS, Ferguson D, Helfrich CJ, Berenson D, Collet A, Diankov R, et al. HERB: a home exploring robotic butler. Aut Robots 2010;28(1):5−20.
[5] Al-Mayyahi A, Wang W, Birch P. Adaptive neuro-fuzzy technique for autonomous ground vehicle navigation. Robotics 2014 Nov 19;3(4): 349−70.
[6] Anderson JM, Nidhi K, Stanley KD, Sorensen P, Samaras C, Oluwatola OA. Autonomous vehicle technology: a guide for policymakers. Rand Corporation; 2014.
[7] Shafer A, Turney M, Ruiz F, Mabon J, Paruchuri V, Sun Y. Robotics based autonomous wheelchair navigation. J Commun Comput 2016;13: 319−28.
[8] Vroegindeweij BA, van Wijk SW, van Henten E. Autonomous unmanned aerial vehicles for agricultural applications. In: Proceedings of international conference of agricultural engineering, Zurich; 2014.
[9] Hernández JD, Vidal E, Vallicrosa G, Pairet E, Carreras M. Simultaneous mapping and planning for autonomous underwater vehicles in unknown environments. In: OCEANS 2015-Genova. IEEE; 2015. p. 1−6.
[10] Wei J, Snider JM, Kim J, Dolan JM, Rajkumar R, Litkouhi B. Towards a viable autonomous driving research platform. In: Intelligent vehicles symposium (IV). IEEE; 2013. p. 763−70.
[11] Shahdib F, Bhuiyan MWU, Hasan MK, Mahmud H. Obstacle detection and object size measurement for autonomous mobile robot using sensor. Int J Comput Appl 2013;66(9).
[12] Dumbre K, Ganeshkar S, Dhekne A. Robotic vehicle control using internet via webpage and keyboard. Int J Comput Appl 2015;114(17).
[13] Ujjainiya L, Chakravarthi MK. Raspberry−pi based cost effective vehicle collision avoidance system using image processing. ARPN J Eng Appl Sci 2015;10(7):3001−5.
[14] Hasdak O. Programming a self-driving car [Doctoral dissertation]. BRAC University; 2015.
[15] McConnell M, Chionuma D, Wright J, Brandt J, Zhe L. Design of an autonomous robot for indoor navigation. In: International telemetering conference proceedings. International Foundation for Telemetering; 2013.
[16] Pannu GS, Ansari MD, Gupta P. Design and implementation of autonomous car using Raspberry Pi. Int J Comput Appl 2015;113(9).
[17] Sobel I, Feldman G. A 3x3 isotropic gradient operator for image processing. In: A talk at the stanford artificial project in; 1968. p. 271−2.
[18] Vincent OR, Folorunso O. June. A descriptive algorithm for sobel image edge detection. In: Proceedings of Informing Science & IT Education Conference (InSITE). 200940; 2009. p. 97−107.
[19] Canny J. A computational approach to edge detection. IEEE Trans Pattern Anal Mach Intell 1986;6:679−98.
[20] Yan J. Infrared sensor. U.S. Patent 7,408,157. 2008.
[21] Forino C. Play embedded. 2016 [Online] Available at: www.playembedded.org/blog/en/2016/01/08/detecting-obstacle-with-ir-sensor-and-arduino/ [Accessed 16 November 2017].
[22] Matsuo K. Murata manufacturing Co., Ltd. Ultrasonic sensor. U.S. Patent 7,728,486. 2010.
[23] King T. ArduinoInfo. 2017 [Online] Available at: www.arduino-info.wikispaces/UltrasonicDistance [Accessed 14 November 2017].
[24] Skiena S. Dijkstra's algorithm. Implementing discrete mathematics: combinatorics and graph theory with mathematica. Reading, MA: Addison-Wesley; 1990. p. 225−7.