DOI: http://dx.doi.org/10.21123/bsj.2020.17.3(Suppl.).1049

RMSRS: Rover Multi-purpose Surveillance Robotic System

Maad Issa AL-TAMEEMI

Department of Computer Engineering, College of Engineering, University of Baghdad
E-mail: maad.al.tameemi@coeng.uobaghdad.edu.iq, maadesaa@gmail.com
ORCID ID:https://orcid.org/0000-0003-4991-8886

Received 27/6/2019, Accepted 15/1/2020, Published 8/9/2020

This work is licensed under a Creative Commons Attribution 4.0 International License.

Abstract:

The development of the internet of things (IoT) and the internet of robotics (IoR) is becoming increasingly involved with our daily lives. It serves a variety of tasks; some of them are essential in human life like real-time remote monitoring to avoid hazards in dangerous situations. The main objective of our robot is to develop a surveillance system for detecting suspicious and target places without any loss in human life. This paper shows the design and implementation of the robotic surveillance platform during real-time monitoring with the help of image processing. The robotic live streaming is provided by two cameras, the first one is fixed straight on the road, and the second is dynamic with tilt-pan ability. Both of the two cameras have image processing capabilities to analyze, detect, and track objects plus a few other graphical functions. These components are built on the top of the four-wheel vehicle system with high torque to provide mobility on rough terrain. This work is based on Raspberry Pi and can be controlled through Wi-Fi locally or publicly over the internet with TCP protocol. The results show high potential, relatively low price robot with lots of features and functions that can perform multiple tasks simultaneously; all are crucial to surveillance and monitoring problems, controlled by a user from far distances and for a longer time.

Key words: IoT, IoR, Object detection, Object tracking, Raspberry Pi, Surveillance.

Introduction:

The Internet of Things (IoT) is the interconnection of the internet with objects, places, and physical environments. The name refers to an increasing number of objects connected to the internet, thus allowing communication between our so-called physical stuff and their digital existences. Lately, the rapid advancements in robotics which aim to perform different functions in different areas lead to augment in the needs for more robotics platforms significantly. Some of the new required functions are achieved easily by the Raspberry Pi, which we based our project on. Earlier the robots were controlled through wire networks but they are now framed to make robots easily controlled by users. In this work, we do not consider distance limitation issues, and the robot can work everywhere where there is a wireless connection. The robot can be controlled over Wi-Fi on TCP protocol with socket programming to exploit the Wi-Fi's high bandwidth compared with other mediums.

Related Works:

In recent days, the robotics is growing rapidly in the world and become more integrated with IoT for sensing, controlling, and for the management of the hardware (1) (2). Aponte and Michael design and construct a mobile robotic platform which works as explorer and observer for risky and dangerous situations. For this reason, this type of robot is exploited in many fields like natural disasters, difficult access places, and for military forces for examination, inspection, and rescue (1). The internet is a global network composed of several identifiable networks (public IP addresses) and reachable thanks to the communication protocol standard (TCP / IP). The data is sent and received at the same time between sensors in a wireless sensor network by avoiding the hidden and exposed problems using the protocol of the order identification time division multiple access (3). The IoT is a new generation of the internet that allows devices and services to communicate over the internet to achieve some objectives (2) (4). A reliable system is implemented in the IoT cloud computing to make load-balance for the data received from IoT devices through the internet by using two types of protocols: Message Queuing Telemetry Transport (MQTT) and Hypertext

Transfer Protocol (HTTP) (5). Some applications need a developable board that has a very effective controller and many features which apply the principles of the IoT like Raspberry Pi (6). As an example, we can mention a Multiple-Privileges Access E-Door System based on raspberry pi using the IoT and technologies supporting it to close and open the door automatically by RFID technology (7). As an another example, a RC car was modified to manage the following functions: track self-driving, traffic light and stop sign detection, and avoid front collision by using a pi camera module and HC-SR04 ultrasonic sensor, which are connected to the Raspberry Pi (model B+) to collect the input data. The process is done by sending the input data to the computer via a local Wi-Fi connection; which then send instructions to Arduino through a USB connection to drive the RC car (8). However, this rover dose not handle object tracking. A wireless surveillance robot system is based on Arduino microcontroller, it is controlled by using a smartphone android system, the connection between this system and the mobile is done through Zigbee protocol Wi-Fi which leads to a limited connection coverage range (9). Also, this surveillance robot is modified to work over Wi-Fi connection remotely using the internet where the coverage range is unlimited (9). The wireless surveillance robot is provided by real-time audio and video streaming, it is developed to monitor specific areas. The brain of this robot is the Arduino Uno R3, which represents the microcontroller, that controls every input and output data (10). Wireless e-surveillance robot system is developed, in which it is controlled by the internet through a webpage, for monitoring by using a moveable camera that is connected with the Raspberry Pi, which is considered as a main part of this robot to perform image processing. Also, this robot can detect the live bodies depending on the Pyroelectric Passive Infrared (PIR) sensor, which sends a notification through wireless communication to the controller to make an alarm when a foreign person enters secured areas (11). NASA Jet Propulsion Laboratory (JPL) designed a mini-rover based on the Raspberry Pi for its high potentials like versatility, accessibility, simplicity, ability to add and upgrade the modifications. The rover uses USB cameras with the facility of real time video streaming which are associated with this Raspberry Pi for touring classrooms, museums, and public engagement activities also to inform the curious about the use of the rover in the space (12). Most of the rovers mentioned above are not supported path drawing, collision avoidance, motion detection, object detection and tracking, and image processing. However, this work proposed rover surveillance robotics system to do all these operations.

Proposed System:

In this section, we present a design and implementation of a rover multi-purpose surveillance robotic system for real-time monitoring with the help of image processing, in which this rover can make remote surveillance, object detection, object tracking, motion detection, path drawing, image tracking, and collision avoidance.

System Design

In the system design process, we design an overall block diagram of the RMSRS, as shown in Fig. 1, this RMSRS can be controlled from long range over the internet or local network. It can also provide live stream videos from two cameras in real-time as explained in detailed in the system implementation section. Wireless communication through TCP/IP, using Wi-Fi 2.4 GHz with TCP/IP protocol enables long communication range through public IP or virtual private network. The medium-range communication can be done by the internet or by using the local network through routers or using Wi-Fi Hotspot devices. The programming is done with Java and Python in order to combine the Java OOP paradigm and the Python’s capabilities for image processing with OpenCV library. OpenCV library is also available for Java but adds additional code complexity and provides less documentation and recourses.

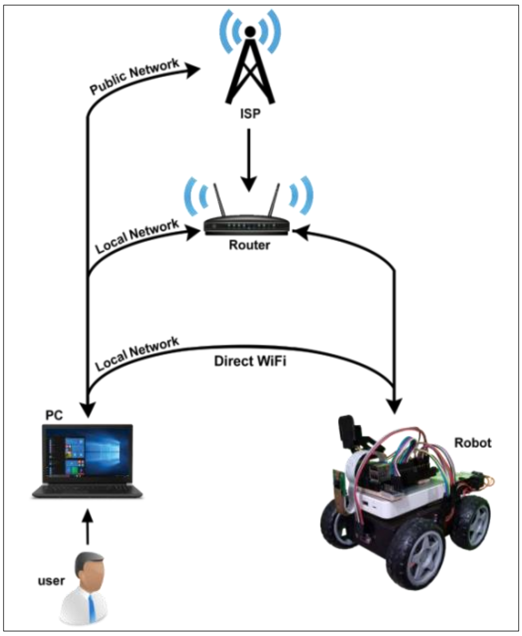


Figure1. The RMSRS design

System Structure

The system specifications used in the robot can rely on IoT. So, we designed and selected suitable modules to develop the RMSRS, as shown in Tables 1 and 2.

Table 1: presents the hardware components summary. These components are used and implemented in the RMSRS.

Table 1. Hardware Components Summary

Component	Module	Qty.	Specification
Embedded Computer	Raspberry Pi 3 model B	1	It is the controller for all robots' components which provides control to the sensors and actuators while sending and receiving instructions from User-Side through wireless network.
Actuators	DC Motor	4	Each DC motor is used with gear box to increase torque providing mobility to the surveillance system.
	Servo Motor	1	It is a rotary actuator that allows precise control of angular or linear position. It is used to control the titling of the second camera up to 90° in the vertical axis.
	Stepper Motor	1	It is a brushless DC electric motor that divides a full rotation into a number of equal steps to control the panning of the second camera up to 360° in the horizontal axis.
Cameras	Pi Camera	1	The v2 Camera module is equipped with an 8-megapixel Sony IMX219 sensor, which can be used to capture high-definition videos as well as pictures. This is the first fixed position camera mounted on the robot.
	USB Camera	1	USB cameras are imaging cameras that transmit picture information using USB technology, which is identified to work on Raspberry Pi 3 model B. This is the second dynamic position camera mounted on the robot.
Batteries	5V power bank	1	To power the Raspberry Pi and the sensors.
	Li-Ion Batteries	4	NCR 18650 3200mAh rechargeable batteries are used to power the actuators, with 4 of these batteries at 3.7v where each two of them are connected in series and connected the two series groups in parallel to provide high capacity with 6400mAh and 7.4v.
Sensors	Distance Sensor	1	The GYVL53L0X sensor is a new generation laser-ranging module which provides accurate distance measurement regardless of the target reflectance. This module is used to avoid collisions.
	Speed Sensor	1	The LM393 Speed Sensor is used to calculate the rotation speed of the robot wheels. It gives a pulse when tires rotate in order to update the user interface which displays the path of the robot on a map.
Driver Circuits	L298n Dual H Bridge	1	The dual H-Bridge motor driver allows speed and direction control of 4 DC motors at the same time.
	ULN2003	1	ULN2003 stepper motor driver provides a direct drive interface between the controller and stepper motor, by which four inputs are provided for connection to the controller. Used to drive the stepper motor for panning the second camera up to 360°
Battery Indictor	level indicator	1	This Light panel lithium battery indicator is used to monitor the battery charge level.

The software is divided into two devices: the PC which is used to control and view information about the robot, and the robot itself which contains the software that controls the actuators and sensors.

Table 2. Software Components Summary

	Component	Language	Specification
Robot-Side	Raspberry Pi	Java	Server-Client program receiving instruction commands and sending information to the User-Side.
		Python	Client program used to stream frames from the two cameras to the User-Side.
User-Side	PC	Java	Server-Client program operated by the user to control the robot by sending commands and receiving information from the Robot-Side. Server program receiving information from the python program for tracking and detecting objects.
		Python	Server-Client program receiving and processing streaming frames from robot side in order to detect and track objects and sending their information to the Java program to make decisions.

System Implementation

We divided the implementation process into three sections, the surveillance system with image processing, vehicle system, and development of Raspberry Pi, as shown in Fig. 2.

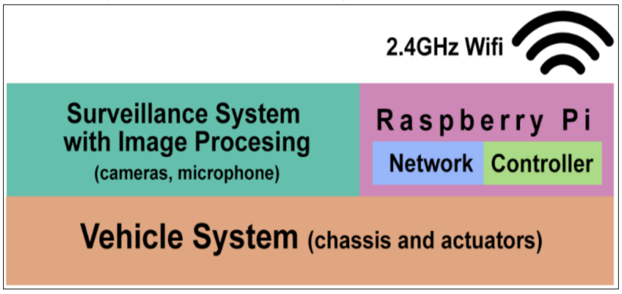

2.4GHz Wifi
Surveillance System with Image Procesing
(cameras, microphone)
Raspberry Pi
Network
Controller
Vehicle System (chassis and actuators)

Figure 2. Basic Architecture of RMSRS

The Surveillance System

It consists of 2 cameras, the first camera is fixed straight forward to the road ahead, its main purpose is to record the front of the robot at all-time and can be used to detect or track desired objects. The second camera is dynamic and is mounted on a pan-tilt platform and can see at 360 degrees (full circle) to prevent any blind spots around it with a tilt of 180-degree to see below and above its position.

The two cameras can provide live footage with a variety of tasks such as recording and capturing, and with the image processing capabilities, more tasks can be accomplished such as motion detection, face detection, object detection, and tracking.

Image Processing:

One of the most important features in the surveillance system is the image processing capabilities. Image processing is used to extract some useful information from an image (frame) through the use of algorithms and operations. Every frame is processed with OpenCV trough a python program that exploits the cameras to detect objects of any kind such as faces, cars, plate number and different sorts of signs and objects.

Object Detection

In this section, the main algorithm used in object detection is based on Haar -feature-based-cascade classifiers (13), which is an efficient object detection method. The algorithm was proposed by Viola and Jones in 2001 (13). It is a machine-learning-based approach where a cascade function is trained from a lot of positive and negative images. Then, it is used to detect objects in other images. As for the object tracking, the object tracking algorithm uses the detection method to track the object by using the information taken from the detected objects such as location and size in the image. Initially, the algorithm needs a lot of positive images (images of a chosen object) and negative images (images without the chosen object) to train the classifier. Then we need to extract features from it. Features are numerical information extracted from the images that can be used to distinguish one image from the another; for example, a histogram is one of the features that can be used to define several characteristics of an image even without looking at it, such as the brightness, the intensity range of the image, contrast, edges, corners, blobs, ridges and so on. Fig. 3 shows some of the features used.

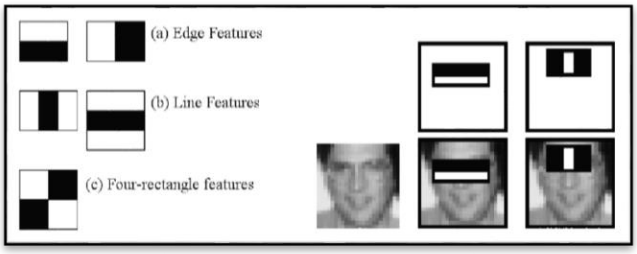


Figure 3. Few types of Haar features (13)

To implement the mentioned above algorithms, we use the library OpenCV which has dedicated functions for Haar Cascades. The OpenCV comes with a trainer as well as a detector. The training algorithm and detection flowchart are explained below.

Algorithm 1: Training

```
1. Data: positve and nagative images
2. Result: xml file
3. vector=opencv_createsamples(positives_info)
4. xml file=opencv_traincascade(vector,negatives)
```

The algorithm 1 is used for training, where each folder should contain a text file named info, each line of this file corresponds to an image. The first element of the line is the filename, followed by the number of object annotations, and followed by numbers describing the coordinates of the objects bounding rectangles (x, y, width, height). According to the detection flowchart, every frame is converted to grayscale and detectMultiScale() function is applied to find desired objects and remove the false positive one. Then, we check if the object stays on at least 3 consistent and consecutive frames to decide that this is a truly positive and puts every detected object into an array as an (x,y) coordinates and a width and height (w,h). Thus, for every object we draw a rectangle and a text representing coordinates. Finally, we export it to the scene window with imshow() function, as shown in Fig. 4.

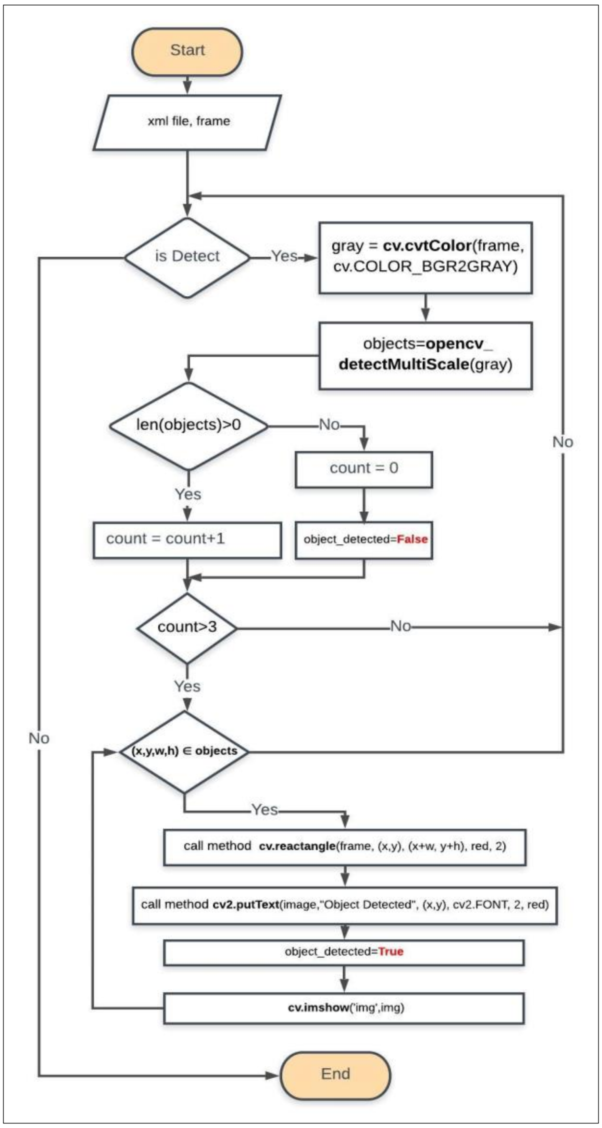

Start
xml file, frame
is Detect
Yes
gray = cv.cvtColor(frame, cv.COLOR_BGR2GRAY)
objects=opencv_ detectMultiScale(gray)
len(objects)>0
No
count = 0
No
Yes
count = count+1
object_detected=False
count>3
No
Yes
No
(x,y,w,h) ∈ objects
Yes
call method cv.reactangle(frame, (x,y), (x+w, y+h), red, 2)
call method cv2.putText(image,"Object Detected", (x,y), cv2.FONT, 2, red)
object_detected=True
cv.imshow('img',img)
End

Figure 4. Block diagram of object detection

Tracking

After detecting the object, a tracking option is available to the user, which enables the camera to look on the object. The whole vehicle will move with the object to keep the object in the middle of the captured framed. We check if the object in the middle by checking (x) coordinates. (y) coordinates are neglected because there is no importance for the object's vertical position in the frame. Also, the area of the object is checked by (width*height) in order to know if the object is far away or close to the robot.

Example

We assume that the screen is 800x600 pixels and the object is 200x200 pixels, the middle being (800/2-w/2), and w is the width of the object in which the object is located at the center correctly (400-100=300). So, the x must be 300 for the object to be in the middle. Therefore, we set the middle range from (800/2-100-w/2) to (800/2+100-w/2) so the range of x will be (200 to 400) and the area should be in a range from (40,000 to 90,000).

The possible cases for the example mentioned above can include as follows:

- object = 100 < rangeX, the vehicle should move to the left
- object = 300 ∈ rangeX, do nothing
- object = 600 > rangeX, the vehicle should move to the right
- objectArea = 20000 < rangeArea, the vehicle should move forward
- objectArea = 60000 ∈ rangeArea, do nothing
- objectArea = 90000 > rangeArea, the vehicle should move backward

The algorithm 2 shows how the operation is done assuming the ranges explained in the example above.

Algorithm 2: Object Tracking

```
Data: x, w, h of a detected object
Result: action (left, right, forward, backward)
while (on) do
    if(x<200) do
        send(left)
    else if (x>400)
        send(right)
    else
        send(stop)
    end
    if ((w*h)<40000) do
        send(forward)
    else if (x>90000)
        send(backward)
    else
        send(stop)
    end
end
```

The algorithm of tracking is also trying to make the object in the middle of the frame by driving and steering the robot so the object centers in the middle. Hence, if any condition in this algorithm is true, the pc control client sends an instruction to the robot to steer it in the right direction and to center the object with the appropriate distance as shown in Fig. 5.

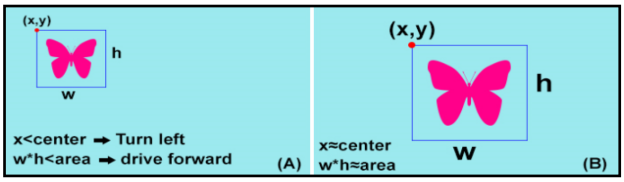


Figure 5. Steering modes: A) Object detected, B) Object tracked

The Robot Vehicle System

It consists of four geared DC motors, two motors on each side, the four-wheel-drive system provides mobility to the heavy surveillance system. Driving forward/backward by giving identical angular velocity value and direction to both sets on each side. A couple of LM393 Speed sensors are attached to the Dc motors one on each side. It used to map the robot's orientation and speed as it moves to create a path. That will help the robot to return to his original starting place or explore other areas without making repetitive turns.

Development of Raspberry Pi Controller

Raspberry Pi is a Single-board computer with many features that make it a suitable controller for our work. The most important features are the Built-in wireless LAN, GPU, and 40-pin general-purpose input/output (GPIO) header that allows Raspberry Pi to connect with GPIO devices. Raspbian is an Debian based operating system running on the Raspberry Pi. It runs two program: the first one was written in Java to control the actuators and sensors, and the second one is written in python for camera streaming. Both programs can receive and send data through TCP network in order to communicate between user-side and robot-side.

Java program controls all the inputs and outputs devices connected to the Raspberry Pi except the cameras, where each device uses a different strategy. DC motors, stepper motor, servo motor, VL53L0X time-of-flight distance sensor module are connected to the GPIO pins to perform the functions explained in Table 1.

Python program running in the Raspberry Pi is simpler than Java's; its only task is to stream frames from the two cameras connected to the Raspberry Pi and to the network. The first camera is a **Pi camera v2 module** high quality 8 megapixel that goes to a dedicated slot on the board, and the second camera is a **USB camera** connected to one of the USB ports of the Raspberry Pi. Both cameras are used for real-time surveillance to capture and record features. The **Pi camera** also tracks and detects objects while the **USB camera** is only for face recognition.

Network

Either local network (LAN) or a public network (internet) can be used for communication between the Robot-side and the User-Side.

- In the case of a local network, the two sides must be on the same network and their local IPs must be known to establish a TCP connection.
- In the case of a public network, the public address should be known, Public IP address is an external facing IP address that is provided by Internet Service Providers (ISPs) and is accessible by anyone on the internet. Unlike with local network, port forwarding must be done to redirect connections from public IP to local IP to make a local device accessible from the internet. The TCP protocol (Transmission Control Protocol) is used for this connection. It is a connection-oriented protocol where a connection is established and maintained until the application programs at each end have finished exchanging data. It determines how to split the data of the application into packets that networks can supply, these packets are then sent and accepted from the network layer, manages flow control, and also handles retransmission of broken packets. TCP utilizes three-way handshaking to connect, this 3-way handshake process is designed so that both ends can initiate and negotiate separate TCP socket connections at the same time. Being able to negotiate multiple TCP socket connections in both directions at the same time allows a single physical network interface, such as Ethernet, to be multiplexed to transfer multiple streams of TCP data simultaneously. When the server starts, it creates a server socket with a port number and then enters a wait state to listen for client requests. The client uses the servers IP address and port number to establish a connection to the server. Thus, the server accepts the request and a stream is open between the two ends for data transfer, as shown in Fig. 6.

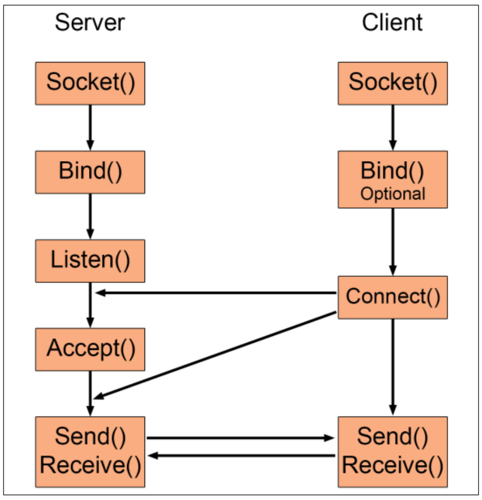


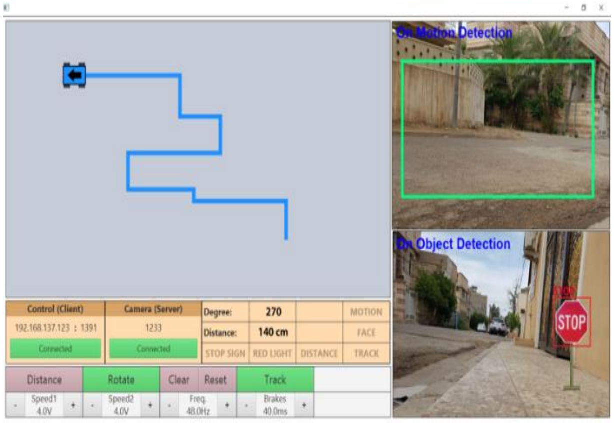
Figure 6. Flows of Events for a Connection-Oriented Socket

Results and Discussion:

Surveillance robots are not a new technology, for years it is used for many military and civilian applications to provide live video with recording abilities, all this is done remotely through a wireless connection. In this work, Wi-Fi 2.4 GHz with TCP/IP protocol is used primarily because of its high bandwidth to handle the live cameras stream. One problem with Wi-Fi is that it consumes more power than others but we solve this problem by adding high capacity Li-ion batteries. The transmitted stream of frames is processed in user-side PC because the PC is more powerful than the Raspberry Pi so it has more computing power, thus we can decrease the delay rate and consume less bandwidth, allowing us full use of the camera's live feed for more benefits. RMSRS accomplish all of the mentioned above with high performance and speed. However, we added few functions to make it stand out from the ordinary surveillance robots. The functions include the use of image processing with object detection and tracking to make the robot more automated, path drawing, collision avoidance, motion detection, live video streaming. RMSRS functions can be stated as follows:

Main User Interface

Starting the Java application in pc will show the main interface as shown in Fig 7.



Control (Client) 192.168.137.123 : 1391 Connected

Camera (Server) 1233 Connected

Degree: 270

Distance: 140 cm

Figure 7. User interface

Figure 7 shows the interface controlling all the operations of the robot such as Map area, Control-client, Camera-server, Distance, Rotate, Clear, Reset, Frequency, and Track.

Path Drawing

We draw the path taken by the vehicle as shown in Fig 7 using speed sensors (LM393 module) mounted on the motors. Each time the user drives the vehicle, a dot is left behind the vehicle icon in the map area. The continuous dots make a path that the user can use to get back to its starting place or to prevent returning to already explored areas.

Collision Avoidance

Collision Avoidance is an intelligent system which can automatically sense the obstacle that is located in front of the vehicle as well as avoid collisions by stopping the vehicle automatically and disable forward movement control. The distance measurement method is calculated by using GY-VL53L0X World's smallest Time-of-Flight (TOF) for ranging and gesture detection sensor, in which the distance value between object and sensor is easy to get by calculating the time difference or phase difference between emission and reflection, so does the in-depth information. Table 3 shows the measurement accuracy of the GY-VL53L0X distance sensor.

Table 3 Accuracy Measurement of GY-VL53L0X Distance Sensor.

Measured Distance	Distance Sensor(Dark)	Distance Sensor(Light)	Accuracy
100cm	93cm	Same	%93
75cm	74cm	Same	%98.6
50cm	52cm	Same	%96.1
25cm	29cm	Same	%86.2
		Overall Accuracy	%93.6

From the table above, the accuracy is determined to be %93.6 in which the lighting of the environment does not affect the outcome.

Camera Functions

- **Object Detection**

When running one of cameras RMSRS, object detection is activated. The detected object must be pre-trained in Haar cascade training algorithm to be detected and feedback is then returned to the user interface through labels. For now, this work trained to detect a stop sign, a traffic light and faces. When an object is detected one of the labels corresponding to the object is turned red and action is done for example when the stop sign is detected the label "STOP SIGN" turns red and the forward control to the vehicle is disabled preventing the user from moving forward as shown in Fig. 8.

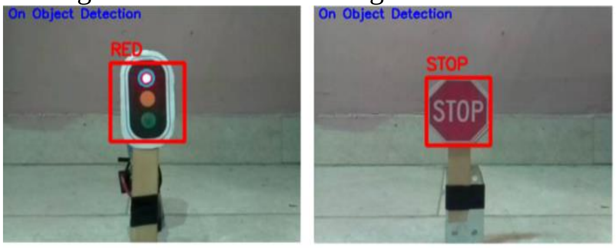


Figure 8. Object detection example

- **Motion Detection**

When the camera motion detection is activated, the green square shown in Fig. 9 saves the pixels for testing with the pixels in the next frame. Then, the motion is detected when the x, y location of the box is moved over a certain threshold.

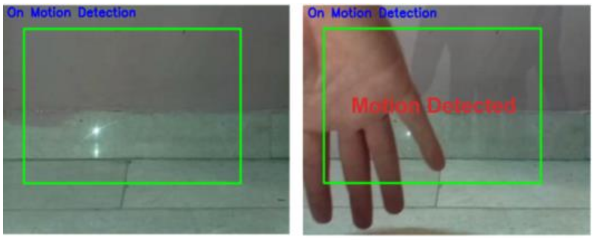


Figure 9. Motion detection example

- **Object Tracking**

When the first camera is activated for the tracking algorithm, it detects the tracked object and checks the x, y and area of the object to drive and steer the vehicle. The label “TRACK SIGN” turns blue indicating that an object is detected as shown in Fig. 10.

Table of comparison of selected related robots:

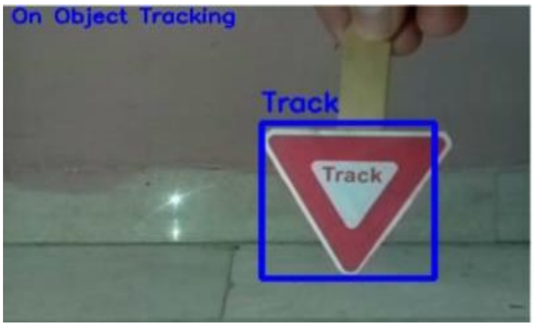


Figure 10. Object Tracking example

In addition, there are other functions that can also be performed using RMSRS cameras, which are image capturing, video recording, and color tracking.

Table 4. comparison of selected related work robots

	RMSRS	**Wi-Fi Surveillance Bot** (10)	**Zheng Wang Robot** (8)	**Nasa Scaled Down Curiosity** (12)
Controller	Raspberry Pi	Arduino Uno R3	Raspberry Pi, Arduino	Raspberry Pi
Program Languages	Java, Python	C++	Python, C++	Python
Camera	2 Cameras	1 Camera, Audio	1 Camera	1 Camera
Connection	Wi-Fi	Wi-Fi	Wi-Fi, RF	Wi-Fi, Bluetooth
Image Processing	YES	NO	YES	NO
Controller	PC	Smart Android Phones	Automatic	Smart Phone, Android Devices, else
Features	4 Wheeled, Pan-Tilt, Camera, Object Detection, Object Tracking, Motion Detection, Path Drawing, Image Tracking, Color Tracking, Collision Avoidance, Customizable	4 Wheeled, Camera, obstacles avoidance	4 Wheeled, Camera, Object Detection, Self-Driving, Collision Avoidance	Six Wheels, Camera

According to the results mentioned in the previous sections and the comparison table with other types of surveillance robots, we can observe that RMSRS is more efficient than others in term of cost, remote connection, battery life, path drawing, collision avoidance, and image processing.

Conclusion and Future Work:

The decision to use the Raspberry Pi 3 came because it supports embedded wireless control and graphical processing capabilities. In addition, the Raspberry Pi’s support for multiple programming languages and its flexibility led to easier programming. Wi-Fi is used primarily because of its high bandwidth to handle the live camera stream. The use of four Wheels Chassis with four DC motors provides mobility to the heavy surveillance system components. Powerful batteries handle the power consumption of this system, necessary for providing speed and torque that allowing the robot to drive smoothly on rough terrains. In conclusion, our work leads to a high potential, relatively low price robot with lots of features and functions that can perform multiple crucial tasks simultaneously to perform surveillance and monitoring, object detection and tracking. The battery life is up to 10 hours and the robot is controlled by a user from far distances and for a long time. There are multiple paths for future work. One path is to control our surveillance robot from mobile phones or tablets and connect it to a cloud environment to save drawn paths and objects detected for configuration and monitoring purposes. This robot can also be developed to control the new near devices which are receiving commands from this robot involve running new functions over Wi-Fi connection.

Author's declaration:

- Conflicts of Interest: None.
- I hereby confirm that all the Figures and Tables in the manuscript are mine. Besides, the Figures and images, which are not mine, have been given the permission for re-publication attached with the manuscript.
- Ethical Clearance: The project was approved by the local ethical committee in University of Baghdad.

References:

1.Castañeda GE MDAJAODacoamtrrp. Design and construction of a mobile type rover robotics platform. In IX Latin American Robotics Symposium and IEEE Colombian Conference on Automatic Control; 2011 Oct 1. p. (1-6).

2.Atzori L IAMG. The Internet of Things: A survey. Computer Networks. 2010 Oct 28;54(15):2787-805.

3.AL-TAMEEMI MI. OIdTDMA: Order Identification Time Division Multiple Access Switching and Routing for Wireless Sensor Networks. International Journal of Computer Networks and Wireless Communications. 2016 Feb; 6, No 2.

4.Whitmore A,AAaDXL. The Internet of Things—A survey of topics and trends. Information Systems Frontiers. 2015 Apr 1;17(2):261-74.

5.Marwa M. Abed MFY. Developing Load Balancing for IoT - Cloud Computing Based on Advanced Firefly and Weighted Round Robin Algorithms. Baghdad Science Journal. 2019 Mar; 16(1).

6.Stephen Chin JLW. Raspberry Pi with Java: programming the Internet of things (IoT)(Oracle Press).: McGraw-Hill Education; 2015.

7.Mohammed Issam Younis MIATMSH. MPAES: A Multiple-Privileges Access E-Door Baghdad: 1st International Conference on Recent Trends of Engineering Sciences and Sustainability by IEEE; 2017.

8.zhengludwig. zhengludwig.wordpress.com. [Online].; 2018. Available from: https://zhengludwig.wordpress.com/projects/self-driving-rc-car/.

9.K. Damodhar BVaKS. A Surveillance Robot For Real Time Monitoring And Capturing Controlled Using Android Mobile. Middle-East Journal of Scientific Research. 24 (S1): 155-166, 2016.

10.Diksha Singh PZN. Wi-Fi Surveillance Bot with Real Time Audio &Video Streaming Through Android Mobile. In 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT).; 2017.

11.Dixit DSKaDMS. Design and implementation of e-surveillance robot for video monitoring and living body detection. International Journal of Scientific and Research Publications. 2014; 4(4), pp.2250-3153.

12.Scaled-Down Model of Curiosity Rover by NASA JPL. NASA Jet Propulsion Laboratory. Build your own NASA Curiosity rover. [Online].; 2018. Available from: https://www.raspberrypi.org/blog/build-nasa-curiosity-rover/.

13.Viola P&JM. Rapid Object Detection using a Boosted Cascade of Simple Features. In ; 2001: IEEE Conf. Computer Vis Pattern recognition. 1. I-511. 10.1109/CVPR.2001.990517.

RMSRS: نظام مراقبة روبوتي متنقل متعدد الاغراض

معاد عيسى التميمي

قسم هندسة الحاسبات، كلية الهندسة، جامعة بغداد، بغداد، العراق.

الخلاصة:

أصبح تطوير إنترنت الأشياء (IoT) وإنترنت الروبوتات (IoR) أكثر وأكثر مشاركة في حياتنا اليومية. إنه يخدم مجموعة متنوعة من المهام بعضها مهم في الحياة البشرية مثل المراقبة في الوقت الفعلي عن بعد لتجنب المخاطر في الاماكن الخطرة . الهدف الرئيسي من نظام المراقبة المتنقل الذكي هو تطوير نظام مراقبة للكشف عن الأماكن المشبوهة والمستهدفة للمستخدمين دون أي خسائر في الأرواح البشرية. تعرض هذه الورقة تصميم وتنفيذ منصة مراقبة آلية للمراقبة في الوقت الفعلي بمساعدة معالجة الصور ، والتي يمكن أن تستكشف أماكن الوصول الصعب أو المخاطرة العالية. يتدفق البث المباشر الآلي عبر كاميرتين، الأولى ثابتة مباشرة على الطريق والثانية ديناميكية مع إمكانية الإمالة. كلتا الكامرتين لديها قدرات المعالجة الصورية لتحليل وكشف وتعقب الكائنات بالإضافة إلى عدد قليل من الوظائف الرسومية. المكونات المذكورة أعلاه مبنية على قمة نظام المركبات الرباعي مع عزم دوران عالي لتوفير القدرة على الحركة في المناطق الوعرة. يستند هذا العمل إلى الراسبيري باي ويمكن التحكم فيه عبر الواي فاي محليًا أو عالميا عبر الإنترنت. تظهر النتائج إنشاء روبوت ذو إمكانات عالية ومنخفض الكلفة نسبيًا مع الكثير من الميزات والوظائف التي يمكن أن تؤدي مهام متعددة في وقت واحد ، وكلها مهمة للغاية بالنسبة لمشاكل المراقبة ، والتي يتحكم فيها المستخدم من مسافات بعيدة ولفترة طويلة.

الكلمات المفتاحية: إنترنت الأشياء، انترنت الروبوتات، مراقبة، كشف الكائن، تتبع الكائن، راسبيري باي