Implementing a Test Bed for Space Exploration Using Internet of Things

Urvi Joshi and Zesheng Chen

Department of Computer Science, Purdue University Fort Wayne

Abstract

The Internet of Things (IoT) has found many applications in industries from retail to smart homes. However, IoT technologies have not yet been applied to space research. The goal of this research is to implement a test bed that demonstrates how IoT technologies can be applied to space exploration. Such a test bed enables a robot to sense information about its terrain and send that information back to users. Specifically, the test bed is built using a Raspberry Pi microcontroller, a Raspberry Pi camera (called picamera), and a movable robot. A user can remotely control the movements of the robot by accessing a web application hosted on the Raspberry Pi. Moreover, the user can also view the live stream of the picamera and take pictures and videos of the stream. The main contribution of this research is a basic IoT test bed that can demonstrate how IoT can be used to facilitate space exploration. Specifically, the prototype for this test bed can potentially demonstrate how visual terrain information on a planet (e.g., Mars) is sent to the end-user application (e.g., a mobile phone) on Earth through the technologies of IoT. Moreover, this research can also help illuminate the capabilities and the limitations of using IoT in space; for example, the camera quality and the live stream speed.

1. Introduction

The Internet of Things (IoT) is defined as "*a network of items – each embedded with sensors – which are connected to the Internet.*" [1] The IoT is valuable because it enables these sensors to transmit important information from remote areas. IoT technologies have already been used in several industries, including healthcare, smart buildings, smart cities, retail, energy, manufacturing, mobility and transportation, logistics, and smart homes [2]. However, space exploration is one area where IoT technologies have yet to be implemented.

Applying IoT to space exploration can yield several useful benefits. First, IoT technologies can be used to remotely monitor and control a movable robot on a different planet (e.g., Mars). Second, the information (e.g., visual terrain) from the planet, taken from sensors (e.g., through a camera), can be transmitted to users on Earth using a user application (e.g., web application). Finally, transmitted data can be processed in the IoT middleware so that the quality of communications between the Earth and a different planet can be improved.

The overall goal of this project is to implement a test bed that demonstrates how IoT technologies can be applied to space research and exploration. The test bed consists of a movable robot, a Raspberry Pi 3 Model B microcontroller [3], and a picamera [4]. Both the robot and the picamera are connected to the Raspberry Pi, which sits atop the movable robot. The robot carries both the Raspberry Pi and the picamera by piggyback when moving around a terrain. A website is hosted on the Raspberry Pi. The function of the website is to capture and send a live stream video, taken by the picamera, of the surrounding terrain to remote users through the web framework built in the Raspberry Pi. That is, users can view the real-time video of the terrain from the test bed in a web page. Moreover, this robot can also be controlled remotely through this website. Various Python scripts have been built in the Raspberry Pi to control the movement of the robot, such as moving the robot forwards and backwards, turning the robot, and adjusting the robot's speed of movement. In principle, we have applied Web of Things (WoT) [5], an IoT framework, in our test bed's design.

As previously mentioned, IoT technologies have been applied to many applications. However, unlike other applications such as a smart home [2], which is a stationary system, in this research the mobility of the robot needs to be considered. That is, we explore how to use the IoT to instruct the robot to move around and take pictures and videos of any part of its terrain.

The paper is structured as follows. First, we will describe the design of the IoT system for space exploration. Following that, we will detail the implementation of the test bed. Next, we will point out the main challenges and our solutions found in the research. Finally, we will discuss the lessons learned and give the conclusions.

2. Design

The overall goal of this project is to develop and implement a test bed where we can investigate the possibility of applying the IoT framework and technologies to planetary exploration. Specifically, Figure 1 shows a diagram of the IoT framework applied to space exploration. Typically, an IoT system consists of edge devices (e.g., smart sensors and actuators), a microcontroller (MCU), a middleware (also called a server or a broker), and an end-user application. The sensors detect the surrounding parameters, such as pressure, humidity, temperature, and images, and send the data to the middleware through the MCU and the Internet. The middleware communicates with the end-user application for any action to be taken or for any information to be displayed. The action to be taken will be transmitted back to the actuators through the middleware, the Internet, and the MCU. When applied to space exploration, IoT can use sensors to sense useful data (e.g., visual terrain information) on the planet (e.g., Mars) and transmit the data to the end-user application (e.g., a mobile application) on Earth.

The IoT framework can potentially facilitate network communications for space exploration, with a relatively low cost. For example, the data sensed by sensors on Mars can be transmitted to the middleware and processed there. A user on Earth can view the data or send a command to the actuators on Mars by simply visiting a web page hosted on the middleware, which can be on Earth or in a satellite.

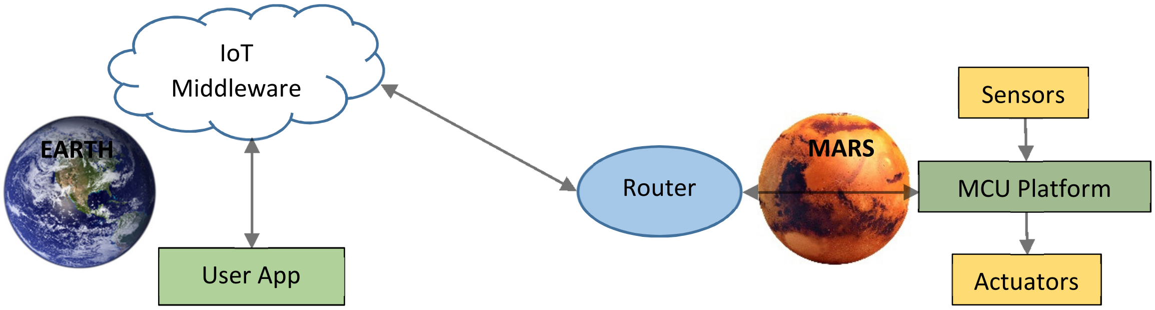

Figure 1. IoT Framework for Space Exploration

3. Implementation

In our designed test bed, we use a camera as the sensor, the robot (e.g., a motor driver and a servo motor) as the actuators, a Raspberry Pi as the MCU, and a web browser as the end-user application. We do not use the IoT middleware in our prototyping. Instead of hosting the web service in the middleware, we implement the web application directly inside the Raspberry Pi (i.e., the MCU). In the following, we detail the implementation of the main components of the test bed. The physical connections between the robot, the Raspberry Pi, and the camera are shown in Figure 2.

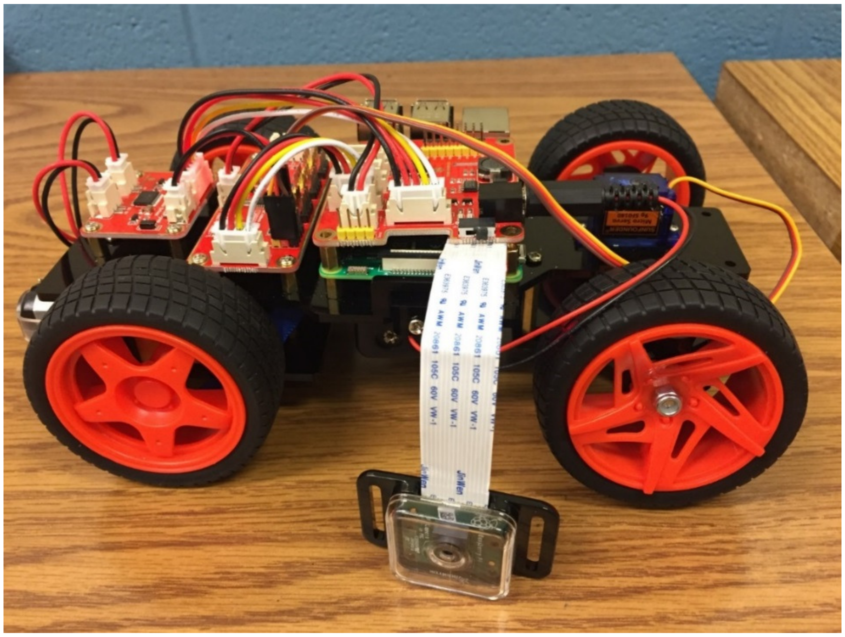

Figure 2. Physical Connections between the Robot, the Raspberry Pi, and the Camera in the Test Bed

3.1 Robot

The platform for the robot is built using a SunFounder Smart Sensor Car Kit [6]. The robot is controlled by a Raspberry Pi 3 Model B (i.e., the MCU), which is running a Linux distribution called Raspbian Jesse as the operating system [7]. On the Raspberry Pi, there are several Python scripts that can move the robot forwards or backwards, turn the robot, and increase or decrease the robot's speed. The Python scripts utilize the SunFounder library to control the motors of the robot's wheels, in order to move the robot in various directions [8]. In addition, the Raspberry Pi uses a TB6612 motor driver [9] to control the back wheels and a servo motor [10] to control the front wheels. The algorithm for the python scripts to move the robot is shown in Algorithm 1.

Algorithm 1: Moving robot

```
Instantiate back/forward wheels
Set user defined speed or angle
Run a Sunfounder Library function to move forwards/backwards/turn
```

3.2 Website

The website is built using HTML, CSS, JavaScript, Ajax, and PHP. Specifically, HTML is used for the page structure, CSS is used for page styling, and JavaScript is used to handle the user's events. Ajax is applied for the communication between the client and the server, whereas PHP is the programming language used for implementing the back-end web service. Moreover, we use the JQuery library to facilitate the page events, especially for the button event handler. The website is hosted on the Raspberry Pi using an Apache web server [11]. Figure 3 shows a snapshot of the web page where a user can control the robot, as well as view the live video stream.

When a user clicks and holds a button (e.g., "Move Forward") in the web page, that message is communicated to the server on the Raspberry Pi, which then runs an appropriate script. The web server uses "mouseDown" and "mouseUp" events to know how long to run a specific script. For example, if a user clicks the "Move Forward" button, a PHP script in the back-end web server is triggered. The PHP script instructs the Raspberry Pi to run the "moveForward.py" script, so that the robot moves forward. As long as the user holds down the "Move Forward" button, the robot keeps moving forward. Whenever the button is released, the same PHP script gets the notification from the event of mouse up and can invoke another Python script in the Raspberry Pi to stop the robot.

In addition to the buttons displayed on the web page, the user can also turn the robot's front wheels to a certain angle and set the speed of the robot, as shown in Figure 3. The speed is saved between page refreshes.

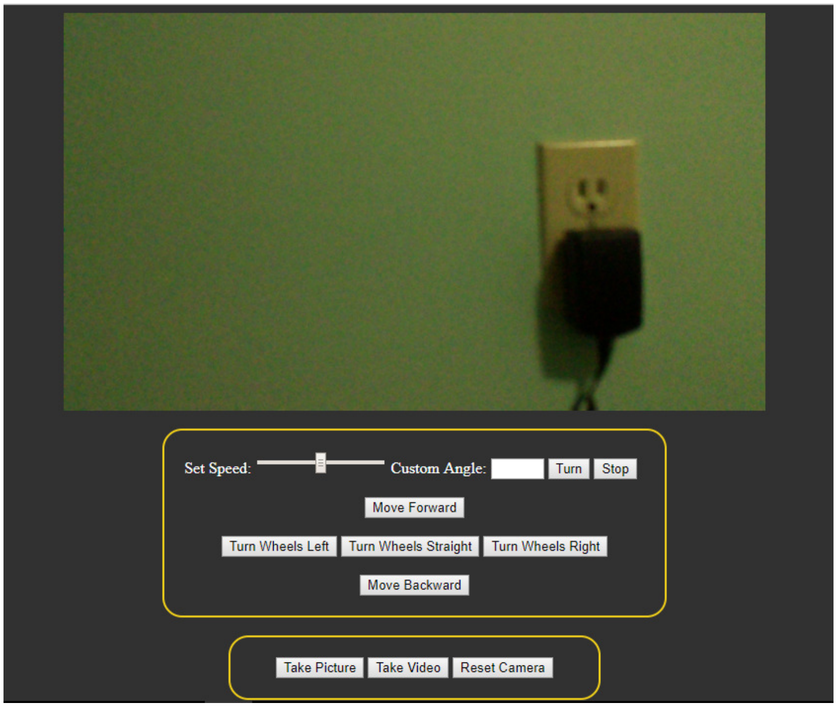

Figure 3. Web Page for a User to Control the Robot and View Live Video Stream

In summary, the algorithm for the web service to control the robot is shown in Algorithm 2.

Algorithm 2: Controlling the robot

```
Input: button_ pressed, speed
if mouseDown event then
   if button_ pressed == button_name then
      Execute a Python script for that action (e.g., moveForward.py, turn.py, etc.)
   end if
else if mouseUp event then
      Stop running currently executing script
end if
```

3.3 Camera

The camera used in our test bed is a Raspberry Pi camera, i.e., picamera [4]. Moreover, we utilize the motion JPEG (i.e., MJPEG) streamer [12] for picamera video streams. The Raspberry Pi has Python scripts that can take pictures and videos from the live stream. When a user clicks “Take Picture” or “Take Video,” as shown in Figure 3, the Raspberry Pi runs a script that takes a picture or a video, and then emails that picture or video to the user. We refer to the process proposed in [13] to implement such functionalities.

4. Challenges and Our Solutions

Currently, IoT is an active research topic [14, 15]. During the construction of the IoT test bed for space exploration, we encountered several challenges. In the following, we list the main challenges and our proposed solutions:

- Streaming pause: When applying the MJPEG software [12], we found that if there is no movement in the terrain of the test bed for a certain amount of time, the live stream would be cut. To remedy this, a “Reset Camera” button has been implemented, as shown in Figure 3. When a user clicks this button, the MJPEG software is restarted, the page is refreshed, and the camera feed starts streaming again.
- Live stream delay: There is a delay between when a video stream is taken and when it is shown to a user in the web page. We have researched several software and techniques for live streaming [16, 17]. Specifically, we found that the Motion streaming software results in a delay of three or four seconds, whereas the MJEPG streaming software leads to a delay of two seconds. Therefore, we utilized the MJPEG software in our implementation.
- Access from a different network domain: Streaming video across different networks was also a challenge. We have investigated different possible solutions. For example, the Message Queuing Telemetry Transport (MQTT) [18] protocol is normally used to transmit data between different networks. However, video data is too large to transfer using MQTT. Other protocols, such as Constrained Application Protocol (CoAP) [19], have the same problem as MQTT and can also result in data loss. One solution to this problem is to stream video to YouTube. We have tested this solution and were able to host streaming video in a YouTube page, referring to the instructions from [17, 20].

5. Lessons Learned and Conclusions

There are several lessons learned in this project. First, this is a great learning experience for us to learn IoT technologies and apply them to space exploration. We came to appreciate how WoT [5] can be applied to a wide range of applications based on the IoT framework. Second, we acquired knowledge of how to use streaming software for live video streams and what the benefits and the drawbacks of different streaming software are. Furthermore, we gained a deep understanding on the tradeoff between design decisions. For example, we found that we can use YouTube to host the live video stream to overcome the issue of access from a different network. However, the delay for hosting video in a YouTube page is about six or seven seconds. Thus, there is a clear tradeoff between the convenience and the delay.

In this project, a test bed was built, which shows that the IoT technologies can be used for sensing visual terrain information and controlling the movement of a robot remotely. Our main contribution is a basic IoT test bed that can demonstrate how IoT can be used to facilitate space exploration.

As our on-going work, we plan to add more sensors, such as a motion sensor and a temperature sensor. Moreover, we will study other streaming techniques that can reduce the stream lag time to below two seconds. Additionally, we will address the potential security issues in the test bed.

ACKNOWLEDGMENTS

This work was supported in part by the 2018 IPFW IRSC Faculty Course Release Grant and Indiana Space Grant Consortium (INSGC).

References

[1]. "Special Report: The Internet of Things," *IEEE Institute*, March 2014, available at: http://theinstitute.ieee.org/static/special-report-the-internet-of-things (January/2019 accessed).

[2]. U. Joshi, A. Dills, E. Biazo, C. Cook, Z. Chen, and G. Wang, "Designing and Implementing an Affordable and Accessible Smart Home Based on Internet of Things," Investigations: The Journal of Student Research @ IPFW, June 2018 [Online]. Available: https://investigations.ipfw.edu/2018/06/12/ijsr007/ (January/2019 accessed).

[3]. Raspberry Pi, a Small and Affordable Computer [Online]. Available: https://www.raspberrypi.org/ (January/2019 accessed).

[4]. Raspberry Pi Camera Module V2 [Online]. Available: https://www.amazon.com/Raspberry-Pi-Camera-Module-Megapixel/dp/B01ER2SKFS (January/2019 accessed).

[5]. D. D. Guinard and T. V. Trifa, "Building the Web of Things," Manning, 2016. ISBN: 9781617292682.

[6]. SunFounder Raspberry Pi Car DIY Robot Kit for Kids and Adults, Visual Programming with Ultrasonic Sensor Light Following Module and Tutorial [Online]. Available: https://www.amazon.com/SunFounder-Raspberry-Following-Ultrasonic-Electronic/dp/B06XYZRBNJ/ref=as_li_ss_tl?ie=UTF8&qid=1493188659&sr=8-6&keywords=motor+driver&linkCode=sl1&tag=intorobo-20&linkId=79a1b59c8266773c28f3890763c73bc8 (January/2019 accessed).

[7]. Raspbian, the Foundation's Official Supported Operating System [Online]. Available: https://www.raspberrypi.org/downloads/raspbian/ (January/2019 accessed).

[8]. Sunfounder_PiCar on GitHub [Online]. Available: https://github.com/sunfounder/SunFounder_PiCar/tree/master/picar (January/2019 accessed).

[9]. TB6612 Motor Driver [Online]. Available: https://www.sparkfun.com/products/14451 (January/2019 accessed).

[10]. Servo Motor, Wikipedia [Online]. Available: https://en.wikipedia.org/wiki/Servomotor (January/2019 accessed).

[11]. The Apache Software Foundation, The Apache Software Foundation, available at: https://apache.org/ (January/2019 accessed).

[12]. Motion JPEG, Wikipedia [Online]. Available: https://en.wikipedia.org/wiki/Motion_JPEG (January/2019 accessed).

[13]. A. Orr, "Use Python to Build a Raspberry Pi - Powered Home Security Camera," [Online]. Available: https://medium.com/@andyorr/use-python-to-build-a-raspberry-pi-powered-home-security-camera-for-50-84ab7e344e2d (January/2019 accessed).

[14]. J. A. Stankovic, "Research Directions for the Internet of Things," *IEEE Internet of Things Journal*, vol. 1, no. 1, Feb. 2014, pp. 3 - 9.

[15]. H. Ma, L. Liu, A. Zhou, and D. Zhao, "On Networking of Internet of Things: Explorations and Challenges," *IEEE Internet of Things Journal*, vol. 3, no. 4, Aug. 2016, pp. 441-452.

[16]. How to Make Raspberry Pi Webcam Server and Stream Live Video || Motion + Webcam + Raspberry Pi [Online]. Available: https://www.instructables.com/id/How-to-Make-Raspberry-Pi-Webcam-Server-and-Stream-/ (January/2019 accessed).

[17]. How to Live Stream to Youtube or Ustream with Raspberry Pi [Online]. Available: https://www.youtube.com/watch?v=u1DgsvjYKS4 (January/2019 accessed).

[18]. MQTT, a Machine-to-Machine (M2M) / "Internet of Things" Connectivity Protocol [Online]. Available: http://mqtt.org/ (January/2019 accessed).

[19]. CoAP, RFC 7252 Constrained Application Protocol [Online]. Available: http://coap.technology/ (January/2019 accessed).

[20]. Live Stream to YouTube With a Raspberry Pi [Online]. Available: https://www.makeuseof.com/tag/live-stream-youtube-raspberry-pi/ (January/2019 accessed).