Empowering Robots via Cloud Robotics: Image Processing and Decision Making BoeBots

Siavash Rastkar, Diego Quintero, Diego Bolivar, Sabri Tosunoglu
Department of Mechanical and Materials Engineering
Florida International University
10555 W Flagler St
Miami, Florida, USA
954-592-8007, 786-877-5916, 305-215-4405, 305-348-1091
srast002@fiu.edu, dquin024@fiu.edu, dboli004@fiu.edu, tosun@fiu.edu

ABSTRACT
Technological advancements have skyrocketed ever since the invention of the computer. While the computer power increases, restrictions on the hardware in use still exist. These constraints may vary from shortened battery life to limited software capabilities. Cloud computing, and subsequently cloud robotics have risen as an alternative to offer solutions to the expanding needs in robotics. Rather than limiting the robot to use only the onboard software, cloud robotics offers access to vast resources; in most cases through wireless internet to complete computational needs remotely. This paper shows an example of how a very restricted robot platform such as the BoeBot mobile platform will benefit from cloud robotics. We will use a local network and an external camera to control the robot and perform image recognition utilizing the cloud system. Furthermore, through this analysis, a conclusion of its economic implications will be reached. Economically speaking, cloud robotics minimizes the need of hardware, which traditionally translates to lower prices for products that are highly technologically advanced. Cloud robotics is additionally appealing due to the simple way in which it can be continuously improved and easily shared.

Keywords

1 INTRODUCTION
Development in the field of robotics is strongly linked to advances in electronics. Code written into microcontrollers is usually restricted to the capacity of their drives, making the function of the robotic parts it controls somewhat limited. In the past few years cloud computing has revolutionized the way robots and humans interact. Cloud computing functions by sharing codes and other resources over a network, with the creation of the Internet and wireless connections, robots can now be controlled from any distance and with almost no limit on its programming size or source of the codes. This is called cloud robotics. This system takes robots to a level where, for instance, artificial intelligence can be implemented with little onboard hardware. The purpose of this paper is to show how a very simple platform such as the Parallax BoeBot can perform computationally demanding tasks by utilizing the cloud robotics concept.

1.1 Parallax BoeBot
The BoeBot robots used in this project are produced by the Parallax Company. These platforms have an aluminum structure, which holds the electronic and mechanical components. Each robot has two ultrasonic sensors in front and back, two servos and two whiskers on the left and right sides, and also two photo-resistor sensors can be mounted on it (Fig. 1). The Basic-Stamp software is used to program the platform, and the debugged code is transferred to the BoeBot by using a USB connection.

Figure 1. A Boebot with an ultrasonic sensor in front
1.2 Cloud Robotics

In the 1960s, after the invention of computers, the need for computers as a public service was realized. It was not until 30 years later, with the creation of the Internet, that allowing launching cloud computing to what it is today (Fig. 2).

Cloud computing refers to sharing of computational resources over a network. Although seemingly a simple concept, it relieves remote devices from the burden of carrying out extensive computations and complex decision-making procedures. It also allows even the simplest devices to have access to an unlimited supply of software resources [1].

Cloud robotics (Fig. 3), on the other hand, builds on this concept and includes the possibility of sharing hardware resources in addition to software. For instance, decisions on the availability of hardware and the most suitable configuration for a given task can be identified relatively quickly. This resource is used to minimize the amount hardware embedded on devices and lower the consumption of power. It also saves time by being able to send the same codes and programs to many robots [2].

In conventional robotics, every task such as moving a foot or grasping an object is programmed and executed on the processor which is carried by robot. However, by taking advantage of cloud robotics, CPU-heavy tasks can be offloaded to remote servers and robot simply sends its workspace data to the cloud and receives executable commands from the cloud [3].

A noteworthy effort has been initiated at Stanford University. The ROS (Robotic Operating System) is an open-source platform, which is a spinoff of the Stanford AI Robot (Quigley). The ROS software is developed on a Unix-like platform although an experimental version for the Mac OS exists [4].

Another platform for cloud computing is the MS Cloud OS known as Windows Azure, which is a commercial platform, backed by Microsoft. It is originally envisioned as a general-purpose cloud-computing platform [5].

Generally we can say that the benefits of using a cloud server are:

- Sending and receiving data to and from external servers.
- Complicated processes can be handled by the server allowing the user to employ smaller and less expensive robots with less power consumption.
- Robots can be easily controlled remotely.
- Cloud server can provide a platform for robots to collaborate with each other.
- Software maintenance is significantly eased. Updating of the software and drivers can be handled easily and uniformly across all robots.

In this paper, Microsoft Windows operating system is used to perform the role of cloud server; a communication channel between the mobile platform and the server is developed with a wireless hub. A special MATLAB code developed in this work captures workspace data sent by the moving platform and makes decisions for the platform and guides it.

2 CLOUD ROBOTICS FRAMEWORK

In this research, tasks are divided into three steps: (1) Establishing a wireless connection between moving platform and the server, (2) Controlling the moving
platform through a website, and (3) performing an image processing task to show data transferring and processing capabilities of the developed cloud robotics framework. In each step, two different types of codes and programs are required, one to be run on the platform and the other to control the server.

2.1 Wireless Connection
Wireless connection between the server and the platform is established by using a wireless hub and wireless USB adapter\(^1\). As figure 4 illustrates, the wireless hub is mounted on top of the BoeBot.

![Figure 4. BoeBot with wireless hub and camera](image)

Three small brackets are attached to BoeBot’s front panel to provide a holder for wireless camera. A specific ‘COM’ port is dedicated to the wireless USB adapter and this port is used both on the server and the platform codes as the communication channel. For sending videos from the mobile platform to the server, an iPhone is mounted on the platform and by using an application called “Mini Web” from the Apple Store, we were able to send the information to a local host. This is exactly same as what IP (Internet Protocol) cameras can do.

2.2 Remote Controlling the BoeBot
A Basic Stamp code is loaded to the platform which can give different motion commands (forward, backward, left, right, stop) based on the value of a string variable which will be read through the ‘COM’ port. Also, this code can send sensor readings through the same ‘COM’ port to our server. Figure 5 shows a section of the Basic Stamp code. As shown in this figure, the SEROUT command gets, and the SERIN command sends data through a serial port. This is used to communicate with the external server. In these commands, the serial pin number of the BoeBot, Baud mode rate and target variables are to be specified.

The first line sends out the readings of the ultrasonic sensor and the second line monitors the serial port for a string variable as motion command.

![Figure 5. Sample basic stamp code](image)

\(\text{SEROUT} \text{Pin, Baud, [DEC Distance, LF]}\)
\(\text{SERIN} \text{Pin, Baud, [command]}\)

Five different subroutines are developed in our Basic Stamp code. Each generates appropriate commands for servo motors to provide either forward, backward, left or right turn motion or the stop command.

Generally, it can be said that the platform gathers data from its workspace and sends them to the server. Server does all the required processing on the raw data and returns a suitable motion command to BoeBot. This motion command sends a signal to the platform’s actuators to execute the desired motion.

A website is designed which contains a panel for controlling the platform (Fig. 6). In the camera address bar of this website, the user can enter the IP address of the wireless camera installed on the platform. By pressing each button, a motion command is stored in a text file in website’s root folder. The remote controlling code, written in MATLAB, continuously monitors this file and whenever a new command is sent to it, MATLAB acquires that and

---

\(^1\) IOGEAR GUWH104 USB wireless hub
by using the developed wireless communication channel, transfers it to the BoeBot.

In Figure 6 a sample MATLAB code is shown. In this code, the first line opens the developed wireless channel, the second line sends the data to this channel and the last line closes the port and shows that the communication is completed.

The website (Fig. 6) can be easily accessed wherever an internet connection is available and the user can remotely control the BoeBot’s motion and also receive an online video from its workspace. Another advantage of using a website as a remote control medium is that it can be accessed via smart phones.

2.3 Image Processing

The whole idea of developing the cloud framework is to be able to accomplish tasks which require high processing power by a robot which has a simple onboard processor. Image processing can be a very good example of such a purpose; it can very well test both data transfer capabilities (by streaming online video) and processing power (by processing the online video) of our cloud system.

This section demonstrates the power of cloud robotics by accomplishing the image processing on the cloud instead of utilizing the onboard microprocessor of the mobile platform. For this purpose, the BoeBot is placed in a workspace which has objects with different colors.

In MATLAB, images are stored as 3D matrices with three layers and each layer shows the values of red, green and blue channels of each pixel’s color [6]. So, for finding a target with a specific color, a maximum and minimum threshold for each color channel is defined and when the BoeBot starts streaming video, MATLAB will capture frames of the video in predefined time intervals.

Then, colors of all of the pixels in that image are compared against the reference value. If a pixel has our desired values, its color will be changed to white; else it is going to be transformed to a black pixel. After this step, the group of white pixels are attached together to form an object, noises are filtered from the processed image and probable holes in the white region are patched.

As we mentioned earlier, performing an image processing task here is to demonstrate the extra capabilities that a simple platform like a BoeBot can gain after being

Figure 7. Sample MATLAB code

```
open(portname);
fprintf(portname,command);
close(portname);
```

Then BoeBot will stream an online video from the environment and this video will be processed by MATLAB. An object with a specific color is defined as the target (here we are looking for the green object), BoeBot scans its workspace until the program on the server computer\(^2\) says that it has found the target and then the same program will guide the platform toward it.

In MATLAB, images are stored as 3D matrices with three layers and each layer shows the values of red, green and blue channels of each pixel’s color [6]. So, for finding a target with a specific color, a maximum and minimum threshold for each color channel is defined and when the BoeBot starts streaming video, MATLAB will capture frames of the video in predefined time intervals.

Then, colors of all of the pixels in that image are compared against the reference value. If a pixel has our desired values, its color will be changed to white; else it is going to be transformed to a black pixel. After this step, the group of white pixels are attached together to form an object, noises are filtered from the processed image and probable holes in the white region are patched.

As we mentioned earlier, performing an image processing task here is to demonstrate the extra capabilities that a simple platform like a BoeBot can gain after being

---

\(^2\)Intel Core 2 Duo CPU T5550 @ 1.83 GHz / 2.00 GB RAM
connected to a cloud server. Hence, for simplicity it is always assumed that there is one and only one target with desired color in our workspace.

In the next step, centroid of the white object is calculated and it is compared with the center of the picture frame. If the object is in the right or left half of the picture, then a command is sent to the BoeBot to turn to the right or left, respectively. If the object is in the center of the picture, then a command is sent to the BoeBot to keep its current direction. The amount of the rotation is defined as a function of the distance between white object’s centroid and the picture’s center.

Figure 6. A frame captured from online video

After BoeBot aligns itself with the target, forward motion command is sent to it and the platform starts to move forward. While BoeBot is in motion, MATLAB grabs new picture frames from the wireless camera in predetermined time intervals and checks BoeBot’s direction. If any correction is required, a left or right rotation command is sent to the BoeBot. This process continues until the reading of front ultrasonic sensor says that the BoeBot is reached within 1 inch distance of the target. At this point, the server sends a stop command to the BoeBot to end the task.

Figure 8 shows a frame capture from the online video and Figure 9 illustrates the processed image. As shown in Figure 9, all of the green pixels are converted to white and the rest is colored black – as our stated goal was to identify the color green in this test. Based on the distance between the white object’s centerline (“Target’s Center” in Figure 9) and picture frame’s centerline (“Picture’s Center in Figure 9), BoeBot aligns itself towards the target.

3 CONCLUSION
The idea of this research was built on the concept that includes the possibility of sharing hardware resources in addition to software. By a relatively modest modification on a simple robot like the BoeBot, we were able to connect it to a more powerful processing and analyzing resources.

At the first step, a webpage was developed and a connection between this page and the platform was established which gave us the capability of controlling the platform via the web. The main advantage of this method is that it can be accessed through smart phones and also it can be easily developed for controlling more than one robot.

Furthermore, to exemplify how a very restricted robot platform such as the BoeBot will benefit from cloud robotics. A local network and an external camera were used to control the robot and perform image recognition.

Economically speaking, cloud robotics minimizes the need of hardware, which traditionally translates to lower prices for products that are highly technologically advanced, as it was observed throughout the research. With very minimal expense, in relation to the robot’s hardware, relatively complex tasks were successfully completed as a benefit of using cloud robotics. In other words, BoeBot was able to execute tasks that it could never attempt without the cloud robotics infrastructure. It must also be noted that cloud robotics is additionally appealing due to the simple way in which it can be improved on and easily shared.

4 REFERENCES
