IoT Waste Management Project

CMPT 496

Author:
Walter Chelliah
chelliahw@mymacewan.ca

Project Supervisors:
Dr. Cameron Macdonell, Ph.D.
cameron.macdonell@macewan.ca

Dr. Nicholas M. Boers, Ph.D.
boersn@macewan.ca

I. Introduction

Waste management at MacEwan University is a voluminous task. Thousands of students each day dispose of waste into every garbage bin on campus on a daily basis. To keep them from overflowing, these garbage bins require regular monitoring and collection services on a regular basis by maintenance staff on campus. This requires a vast amount of resource expenditure on behalf of MacEwan University in terms of labour and capital.

According to MacEwan University sustainability statistics, an estimated 63% of waste was diverted from the landfill in 2012 and 2013 [1]. In 2014, MacEwan University diverted 65% of their trash from the landfill and by 2020, they hope to reach a 90% diversion rate from the landfill [2].

Maximizing waste management efficiency and increased sustainability has become a primary focus of many educational institutions and businesses all across North America. At the forefront of this trend, is the implementation of information technology and intelligent systems into the waste management process. Automated and SMART processes allow human resources to be more efficiently deployed on other tasks and save money for the administrators by reducing labour costs. One such method is remote monitoring of garbage bin capacity.

One method of remote monitoring of garbage bin capacity involves deploying proximity sensors inside the garbage bin which can detect the capacity of garbage in that specific bin. These sensors are connected to a node, which can transmit sensor data to an external server where these sensor readings can be interpreted, logged and displayed to a client located off site via wireless networking. Our IoT waste management project is based precisely on this model.

This paper is structured as follows: Section II outlines the architecture of the project and illustrates the deployment in a live environment. Section III details the hardware components that were deployed. Section IV illustrates the sensor calibration task that was carried out to provide usable data from the sensor. Section V describes the power requirements of the hardware components. Section VI outlines the security measures that were implemented in the project. Section VII concludes with some future considerations for the project and a summarizing conclusion.

II. Architecture

The architecture of the working IoT waste management system is based on a client-server relationship where the client initiates communication on a network and the server reciprocates the communication to provide a service. Attached to each client node is a proximity sensor that detects any changes in distance when an object is placed in front of it. In our model, one proximity sensor is mounted on the top of a garbage bin to detect any changes in capacity as the garbage bin is being filled. The proximity sensor continuously sends this distance information to a Raspberry Pi client node. A Python script on the client relays this information to a server through a wide area network via TCP/IP socket programming.

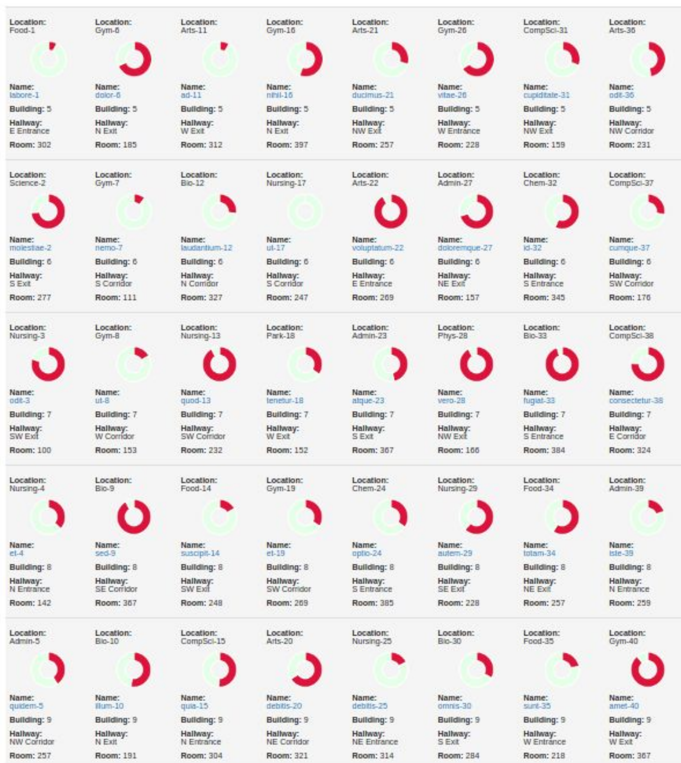


Figure 1. Visual Table Of Current Capacity Of 40 Garbage Bins

Another Python script on the server awaits any incoming connection from the client and stores the incoming distance data into a MySQL database. The server also displays associated client node details such as which client node sent this information, which building was this client node located and what time this information was received. The server also sends confirmation to the client that the data has been successfully received and continues to await new incoming connections from the client node.

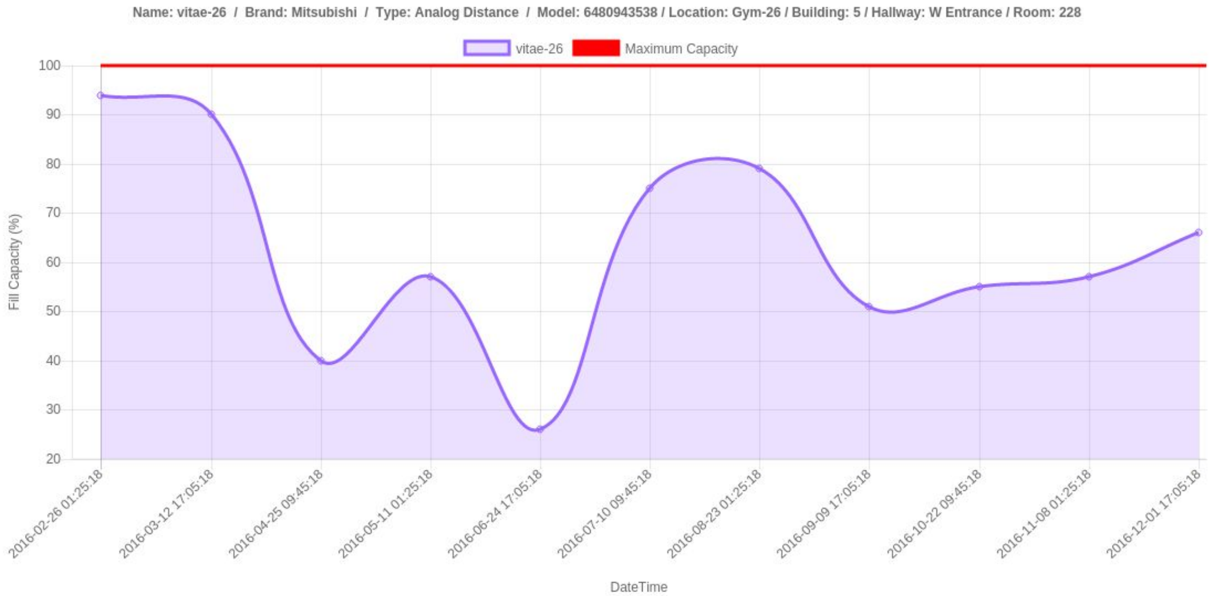


Figure 2. Line Chart For Specific Garbage Bin

The information in this database is accessible to administrators through a php web frontend utilizing the Laravel framework. This framework allows for sorting, analyzing and logging all data through various graphical user interfaces. Various javascript and CSS utilities such as charts, graphs and statistics can be utilized and shared to track trends and patterns to better utilize waste management resources.

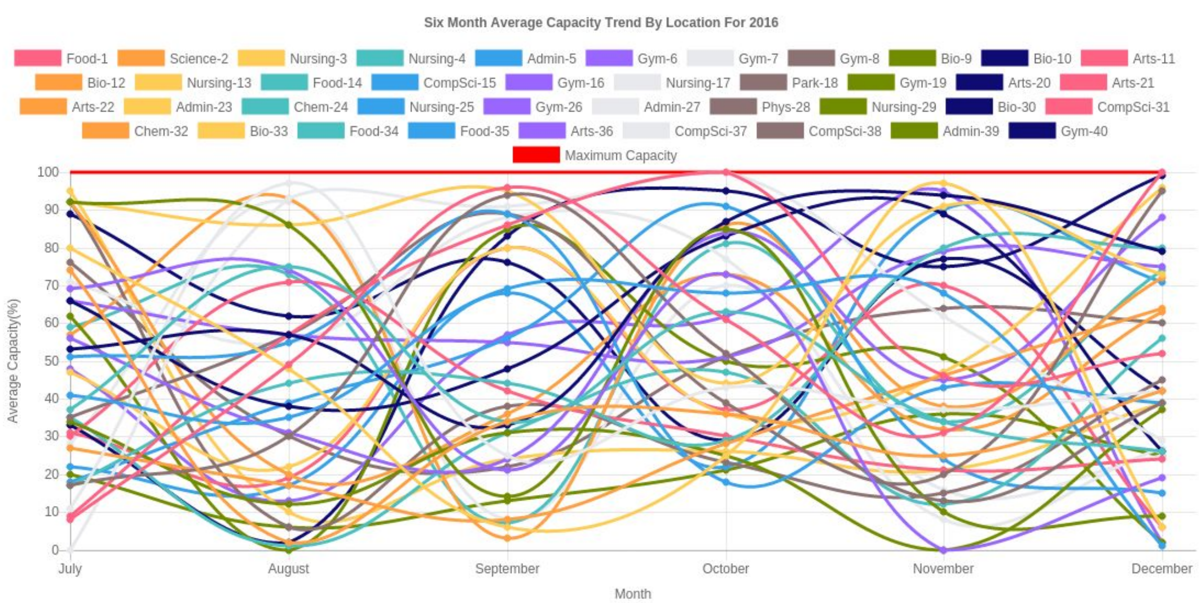


Figure 3. Six Month Average Trend Of All Monitored Garbage Bins

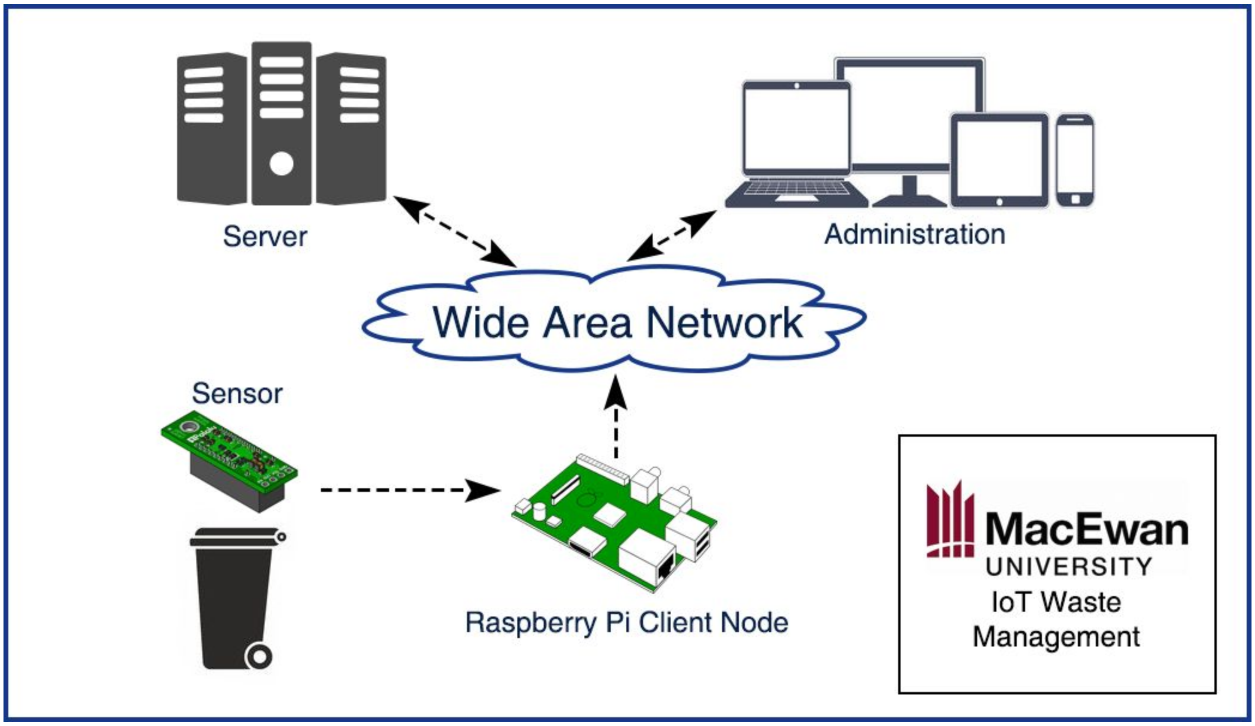
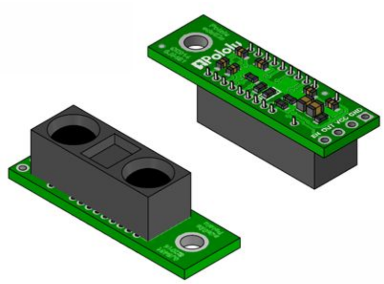
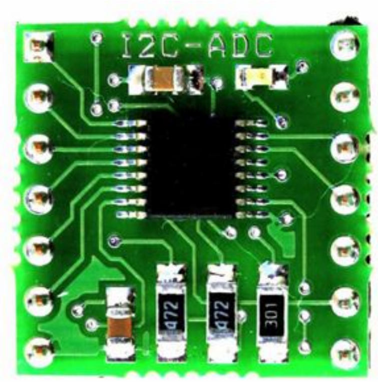
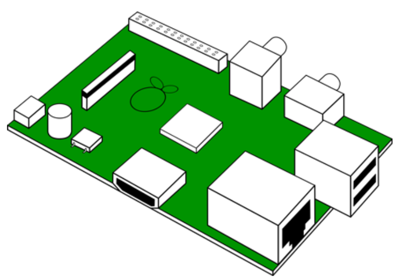


Figure 4. MacEwan University IoT Waste Management Architecture Diagram

III. Hardware

The hardware selected for this project was based on certain criteria: low cost, efficiency and portability. Selecting a low cost combination that would work in remote areas using minimal energy resources that required very little to no maintenance was the priority consideration. The following pieces of hardware fit this criteria and were used in this project:

Name	Image	Description
Pololu Carrier with Sharp GP2Y0A60SZLF Analog Distance Sensor 10-150 cm, 3.3V		The Sharp GP2Y0A60SZ distance sensor offers a wide detection range of10 cm to 150 cm and an update rate of 60 Hz. The distance measured is indicated by an analog voltage, so a single analog input is required to interface with the module. The sensor ships installed

		on a compact carrier board by Pololu, which allows the sensor to be more easily integrated into any project. This sensor board is configured for 3V mode.[3]
Gravitech I2C-ADC 12-Bit, 8-CH Analog-to-Digital Converter (contains Texas Instruments ADS7828 12-Bit, 8-Channel Sampling Analog-to-Digital Converter Chip with I2C Interface)		This I2C-ADC board is a 14-pin CMOS device that provides an 8-CH, 12-bit Analog to Digital Converter (ADC) using I2C bus. This board features innovations like on-board I2C address jumpers, pull-up resistors, power LED and a 2.5V reference. The main processor on the I2C-ADC is a Texas Instruments ADS7828 Chip.[4]
Raspberry Pi Model 2B		This second generation Raspberry Pi Model 2B+ delivers 6 times the processing capacity of previous models. It has an upgraded Broadcom BCM2836 processor, which is a powerful ARM Cortex-A7 based quad-core processor that runs at 900MHz. The board also features an increase in memory capacity to 1Gb.[5]

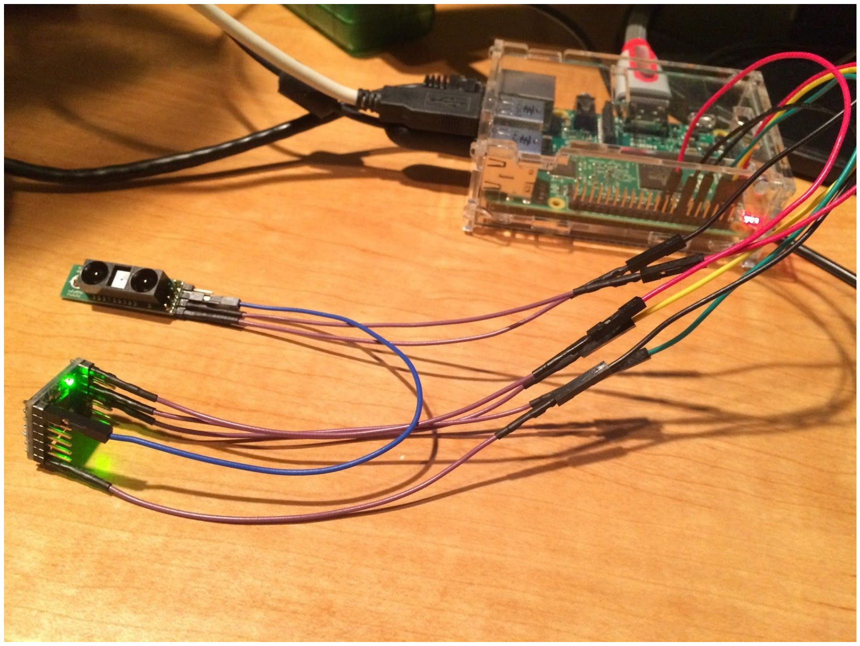


Figure 5. Preliminary Working Hardware Configuration

IV. Sensor Calibration

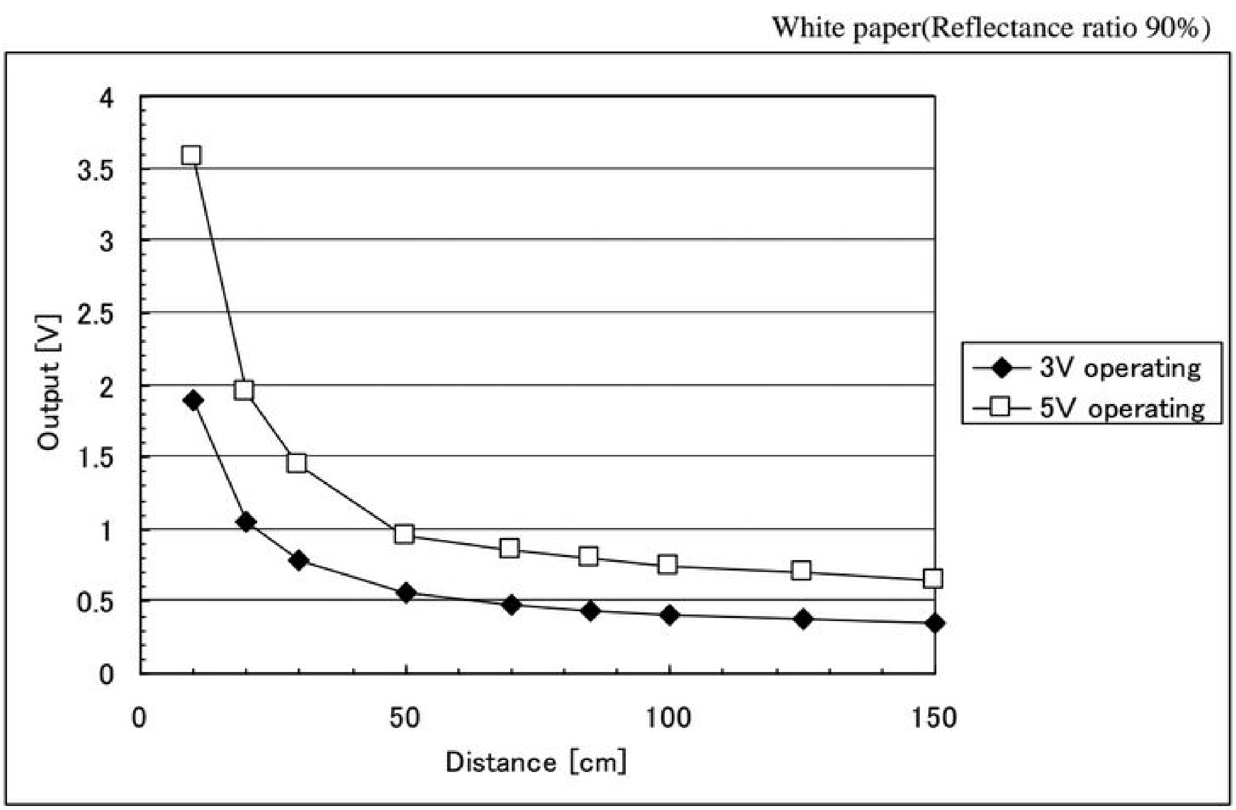


Figure 6. Analog Sensor Output Voltage vs Distance From The Sharp GP2Y0A60SZLF Data Sheet [6]

The data sheet for the Sharp GP2Y0A60SZLF analog distance sensor contains a plot of analog output voltage as a function of the distance [6]. This information only reveals that there is a relation between these two variables but it does not provide any information on how to calculate a measure of distance from the analog sensor reading. In order to get usable data from the Sharp GP2Y0A60SZLF analog distance sensor, it had to be manually calibrated. This process involved mounting the analog distance sensor onto a static location where it would not be disturbed. A meter stick was then mounted 15cm behind the sensor and elevated approximately 5cm above the sensor so that the 0cm mark on the meter stick lined up with the front face of the sensor lens. The meter stick was positioned this way so as not to interfere with the sensor's readings.

Although the Sharp GP2Y0A60SZLF analog distance sensor is rated to read distances 10cm to 150cm from the sensor, preliminary tests carried out during calibration found that the sensor detects as close as 5cm from it's lens so that was the starting point for our calibration task. Initially, a solid black box was used to calibrate the sensor but this was providing inconsistent data through multiple calibrations runs. Once the box was wrapped in white paper, consistent output data was achieved.

In order to effectively calibrate the sensor, the output of the sensor was measured 5 times for each required distance and the mean sensor value was calculated from those readings. The analog sensor was made to output values from 5cm to 100cm with an approximate 25 cm increment. The 150 cm is equivalent to an infinity reading when nothing is placed in front of the sensor within its detectable range.

Mean Sensor Values:

Distance From Sensor (cm)	Analog Sensor Output (integer)
5cm	3750
25cm	1469
50cm	950
75cm	830
100cm	698
150cm	575

Figure 7. Sharp GP2Y0A60SZLF Mean Sensor Voltage Values vs. Distance From Calibration Process (Table)

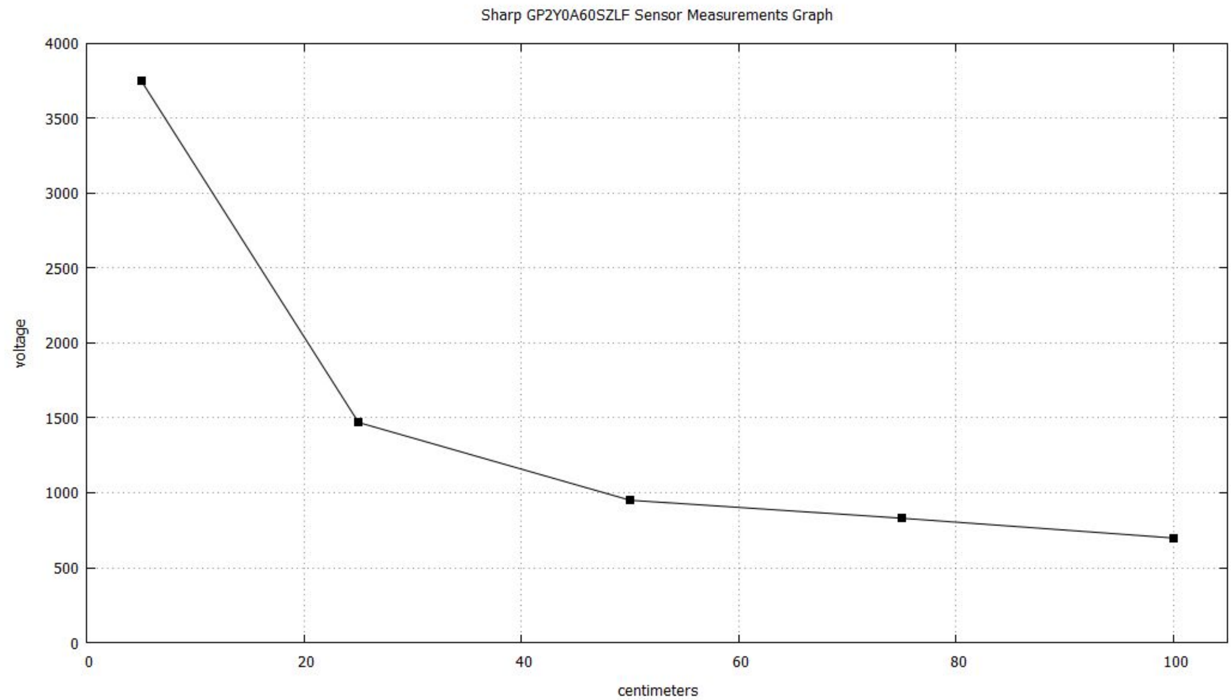


Figure 8. Sharp GP2Y0A60SZLF Mean Sensor Voltage Values vs. Distance From Calibration Process (Graph)

The measured values were added to an Excel spreadsheet and a power trendline was generated to give an equation that would calculate distance from the analog 12-bit sensor value. This equation gave only estimated measurements as only 6 data points were provided during calibration.

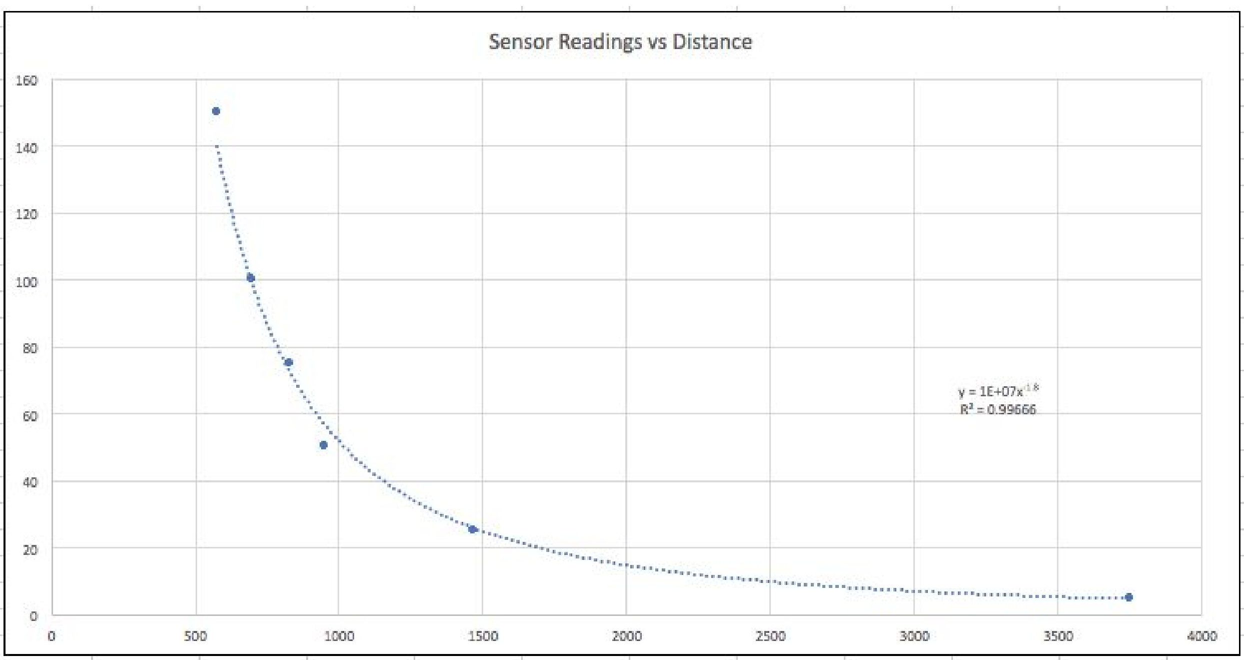


Figure 9. Sensor Voltage Values vs. Distance Power Trendline From Excel

V. Power

The power requirements of the Sharp GP2Y0A60SZLF Analog Distance Sensor consists of a minimum operating voltage of 2.7V and a maximum operating voltage of 5.5V and requires 33mA of supply current. However, this sensor draws current in large, short bursts so there could be fluctuations in current draw.

The power requirements of the Gravitech I2C-ADC Analog-to-Digital Converter consists of a minimum operating voltage of 2.7V and a maximum operating voltage of 5V. The convertor also contains an internal 2.5V reference and an external 5V reference.

Power consumption measurements were conducted by through the use of a multimeter set to measure current at 20mA. The sensor drew approximately 21.1 mA of power when enabled and this number did not seem to vary depending on the power draw required when performing readings.

The analog-digital convertor had two modes which affected power consumption and both modes were required to be enabled to get proper reading information from the sensor. When the the power reference mode and the analog to digital conversion mode were enabled, the power draw was measured to be 4.81 mA and only dropped to 3.96mA when both modes were disabled.

Since the 3.3V VCC pin that was initially used to power the analog-digital convertor was not controllable through the Raspberry Pi GPIO interface, the analog-digital convertor was powered through one of the standard GPIO pins instead. This switch allowed the analog to digital convertor to be programmatically controlled through the client node's Python script by utilizing the Raspberry Pi GPIO module [7]. When power to the analog-digital convertor was enabled through the GPIO pin, power consumption was measured to be 3.57 mA. However, when the GPIO pin was disabled, a measurement of -1.00 mA was observed. The reason for this negative reading is suspected to be that the digital GPIO pin on the Raspberry Pi was sinking current when disabled. Sinking current can occur when a ground and voltage source are required to create a circuit [8]. Since we are utilizing both ground and VCC connections on the Raspberry Pi and the analog to digital convertor, we have a sourcing I/O connected to a sinking I/O which could potentially draw current. Finally, when the analog to digital convertor was enabled and the sensor was performing a reading, there was a significant increase in power draw from 3.57mA to 4.30mA.

	Sensor	**Analog - Digital Convertor** (Power Reference & Analog-Digital Conversion)	**Analog - Digital Convertor** (VCC through GPIO pin)	**Analog - Digital Convertor** (VCC through GPIO pin) - while Sensor is reading
Enabled	21.1	4.81	3.57	4.30
Disabled	0.03	3.96	-1.00	-1.00

Figure 10. Measured Analog-Digital Convertor & Sensor Current Values In mA (milliamps)

The Raspberry Pi Model 2B has a Micro USB socket port that requires 2A @ 5V of power. The 27 GPIO pins on the Raspberry Pi provide 3.3V of power and there are additional VCC pins to power additional 5V and 3.3V accessories.

The minimum supply power draw of the Raspberry Pi Model 2B is 200mA. Adding on WiFi will add on an additional 170mA while plain wired ethernet will add only 40mA. With the computational and hardware requirements of this project, it is estimated that the power draw of the Raspberry Pi Model 2B will be a minimum of around 650mA @ 5V of power but this is a lowest case scenario [5].

Since the minimum power requirements of the Raspberry Pi Model 2B for this project are estimated to be 650mA, we can double it to get a safe typical estimate and times it by the required voltage to give us a measurement in watts [9].

$P_{(Watts)} = I_{(Amps)} \times V_{(Volts)}$

2 * (650 mA * 5 V) = 6.5 W

A typical battery pack has a capacity of 10000mAh.

The watt-hours can be calculated by multiplying the battery capacity by the battery voltage and dividing by 1000 [10].

$E_{(Wh)} = Q_{(mAh)} \times V_{(Volts)} / 1000$

Wh = (10000mAh x 3.7) /1000 = 37 Wh

It should be noted that the mAh rating of a battery usually refers to its internal cell voltage of 3.7 volts, not to its output voltage of 5 volts.

Therefore, if you add a 10000mAh battery pack to the Raspberry Pi Model 2B, it can run for:

37 Wh / 6.5 W = about 5.7 hours

Since the analog-digital converter and sensor would be powered down between readings, the typical power draw would be less than 650mA for the majority of this time, thus expected battery life could possibly be longer than calculated.

VI. Security

Security was a component of the project as the data from the client node travelling to the server could be compromised if left unprotected. Some of the possible consequences of leaving these components unsecured include loss of data, theft of data or manipulation of data which would have negative consequences for waste management potentially affecting scheduling and collection operations.

The data connection between the client node and the server was protected through AES encryption employing the cypher block chaining(CBC) method.

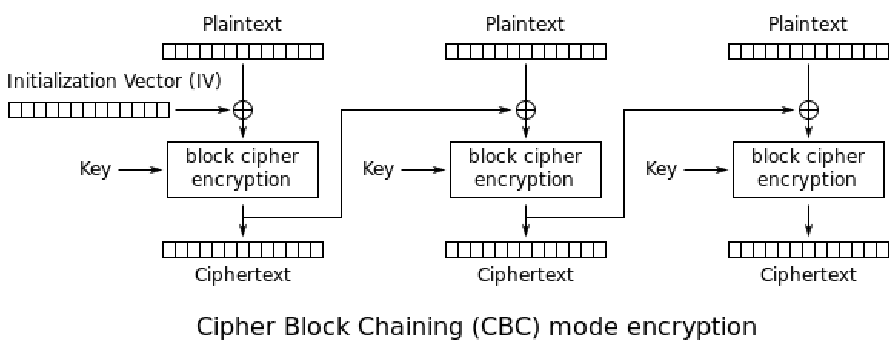


Figure 11. Cipher Block Chaining (CBC) Encryption Process [11]

The CBC method of AES encryption involves using an initialization vector in the first block to make the encrypted message unique. Each block after is then XORed with the previous ciphertext block prior to encryption. This forms a dependency on all plaintext blocks that have been previously processed.

The specific Python encryption package that was used in this project was PyCrypto. The plaintext data from the client was encoded using Base64 and then padded to match the CBC requirement of a block size of 16 bytes. On the server, the encrypted data was decrypted bit by bit and decoded using Base64. Finally, the Python chr and ord functions were used to encode and decode the plaintext padding in ascii format which was then added on or stripped off respectively.

Other modes of AES encryption such as cipher-feedback (CFB), output-feedback (OFB) and counter (CTR) modes may be preferable to use with their enhanced security features and the lack of a requirement for padding but for the scope of this project, the CBC method was more than secure enough.

VII. Conclusion And Future Works

For this project, our goal was to provide a working model to maximize waste management efficiency and increase sustainability at MacEwan University. Our model successfully demonstrated a working sensor inside of a garbage can that could monitor capacity levels and transmit that data over a network where it was interpreted, logged and displayed to a remote administrative client. However, as the project progressed, there were some components of the project that required further investigation and possibly, additional resources to maximize efficiency.

Additional client nodes could be implemented, each with its own cluster of sensors deployed to garbage bins in an assigned area. This would extend the garbage bin monitoring exponentially utilizing minimal resources as the same client scripts, network and server would still be in place. In essence, the administrative server side would require no significant increase in energy resources while the client node side would only require additional low cost hardware. This could be repeated as necessary until all garbage bins on campus can be monitored remotely.

Along with additional client nodes would be the additional requirement for battery power for nodes in outside or remote locations that could operate with minimal maintenance required. Measurements only need to occur a few times daily to effectively monitor garbage capacity so the client nodes, analog-digital converter and sensors could be powered off between cycles. In addition, client node hardware could be minimized further by utilizing smaller Arduino Feathers with embedded LoRaWAN (Long Range Wide Area Network) radios. These Arduino Feathers are lower cost and require less energy to run than Raspberry Pi Model 2B's [12]. The addition of LoRaWAN radios permit these client nodes to communicate over a radio frequency that allows for communication for many kilometres without the need of a wireless network [13].

The trendline could have been more accurate if there were more calibration marks implemented in a controlled environment such as a laboratory setting. The current method of calibration used only 25 cm increments but a more accurate trendline could be achieved by measuring 1 cm increments through the sensor detection range of

5 to 150 cm. A controlled method of achieving exact 1 cm measurements should be implemented to get the most accurate distance measurements from the sensor.

Finally, the model used for this project can be ported for other uses besides garbage bin capacity monitoring. Some other examples of where proximity sensors can be used is in security monitoring and automation. From monitoring windows to automated pet food dispensing, proximity sensors will be a useful tool in many upcoming IoT projects.

References:

[1] MacEwan University Sustainability Progress Report 2012-2014, http://www.macewan.ca/contribute/groups/public/documents/document/sustaindocs_reports_progress14.pdf, accessed on 17 November 2016.

[2] MacEwan University Sustainability Map, http://www.macewan.ca/contribute/groups/public/documents/document/sustaindocs_tools_map.pdf, accessed on 17 November 2016.

[3] Pololu Carrier with Sharp GP2Y0A60SZLF Analog Distance Sensor 10-150cm, 3V https://www.pololu.com/product/2476, accessed 17 November 2016.

[4] Gravitech I2C 12-Bit, 8-CH Analog-to-Digital Converter http://www.gravitech.us/i2c128anco.html, accessed 17 November 2016.

[5] Introducing The Raspberry Pi Model 2B, https://learn.adafruit.com/introducing-the-raspberry-pi-2-model-b/, accessed 17 November 2016.

[6] Sharp GP2Y0A60SZLF Distance Measuring Sensor Unit Data Sheet, https://www.pololu.com/file/download/gp2y0a60szxf_e.pdf?file_id=0J812, accessed on 17 November 2016.

[7] Raspberry Pi GPIO 0.6.3 Module, https://pypi.python.org/pypi/RPi.GPIO, accessed on 17 November 2016.

[8] What Is the Difference Between the Terms Sinking and Sourcing?, http://digital.ni.com/public.nsf/allkb/A10122C63A7F5CFE86256B4C007491DD, accessed on 17 November 2016.

[9] How to convert amps to watts, http://www.rapidtables.com/convert/electric/Amp_to_Watt.htm, accessed on 17 November 2016.

[10] How to convert mAh to Wh, http://www.rapidtables.com/convert/electric/mah-to-wh.htm, accessed on 17 November 2016.

[11] Block Cipher Mode Of Operation, https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation, accessed 17 November 2016.

[12] Arduino Feather Power Management, https://learn.adafruit.com/adafruit-feather-32u4-basic-proto/power-management, accessed 17 November 2016.

[13] LoRaWan Limitations https://www.thethingsnetwork.org/wiki/LoRaWAN/Limitations, accessed 17 November 2016.