Paper ID #35861

Autonomous Inventory Tracking

Maria Katerina Apostle, Wentworth Institute of Technology

Current sophomore at Wentworth Institute of Technology pursuing a bachelor's in Computer Engineering and a minor in Physics. My interests include robotics and embedded systems.

saurav basnet, WentWorth Institute of Technol
Luke Clarke Bassett

Autonomous Inventory Tracking

Maria Apostle, Luke Bassett, Saurav Basnet
Computer Engineering Program
School of Engineering
Wentworth Institute of Technology
Boston, MA, 02115, USA
apostlem@wit.edu, bassettl@wit.edu, basnets@wit.edu

Abstract— Inventory management is a key part of the success of many businesses and organizations. Several methods to track inventory exist, the most common of which being manual tracking using items such as hand scanners. However, manual input may be more prone to error. Some methods of autonomous tracking currently exist, however these are very large-scale and are expensive to produce and maintain. This project proposes a low-cost alternative for autonomous inventory tracking with a small-scale QR code-based inventory management robot using a Raspberry Pi and Arduino. The robot utilizes a USB camera to detect product QR codes on a shelf while using several QTI (Charge Transfer Infrared) sensors to follow a black line along the ground bordered at its edges to indicate the start and end point. On each run, the robot will move forward along the path, reversing when it detects the first border and finally stopping when it detects a second border. Meanwhile, as the robot moves along the path, the camera detects the QR codes along the shelves and records the data in a CSV (Comma Separated Values) file for easy exporting into databases and spreadsheet programs such as Excel. In practice, the robot functions in most rooms, even in those with lower lighting. Additionally, the robot can accurately and consistently read codes from a large range of distances, allowing for flexibility in its placement in a warehouse or store. Lastly, its cost-effectiveness allows for implementation in most businesses. Currently, more features are being implemented to improve the bot, including allowing it to read codes on higher shelves, and improving camera quality and detection, researching alternatives to further reduce costs.

Keywords—*QR Code, Database, Vision Processing, Line Following, Inventory Management*

I. Introduction

In the retail field, efficient inventory management is a crucial aspect of a business' success. Presently, most retail stores and warehouses do not have a fully automatic system of inventory management. Items must be scanned manually, or there are people who are required to update spreadsheets or databases to track inventory. Manual inventory management can occupy a lot of the company's time, resources, and money [1]. In addition to the aforementioned issues, manual error is a major factor in the problem of inaccurate or inefficient management of inventory. The addition of an automated inventory system would increase a company's efficiency and free up resources to invest in other aspects of the business.

Currently, there are no available systems that fully automate the inventory tracking process. There have only been prototypes and research projects, but none have been released to the public. The solutions proposed in research utilize a wide variety of methods, including RFID (Radio Frequency Identification) and OpenCV (Open Computer Vision) to track inventory [2,3,5]. These systems are useful but are not able to reach the market because of their cost, as well as being in early stages of development. Additionally, some are not as precise in their item or tag tracking.

More specifically, the downside to RFID technology is that its range can easily be weakened by interference from metal objects, other RFID tags, and nearby electronics [3]. RFID is also significantly more expensive than competing technologies.

As for computer vision robots, an extensive dataset of images is required and they may have a difficult time identifying specific items due to their limited ability to read text on items which can vary in font, sizing, and other physical attributes [2].

The solution in this project proposes a line-following robot that drives along shelves to read QR codes. The purpose of this project is to develop an inventory management robot and examine how effective the prototype is at driving, reading QR codes, and storing that data in a database in a small form factor and cost efficient manner.

II. Application

The overall design approach was to build a prototype robot to drive along the side of a shelf and scan QR codes. Upon pushing a button, this robot would execute one full run in which it would follow a black line, return upon detecting a perpendicular black border, then stop upon detecting the second border at the starting position. Meanwhile, the camera would detect and scan QR codes while passing by the shelf. Ultimately, this data would then be uploaded into a CSV file and transferred onto the user's computer wirelessly via SSH (Secure Shell) for inventory tracking.

A. *Design*

I) *Electrical Circuit Design*

Two different microcontrollers operated separately as a part of the overall system: an Arduino (Uno R3, arduino.cc), and a Raspberry Pi (Model 3B+, raspberrypi.org). Their circuit diagrams can be seen in Figures 1 and 2 respectively.

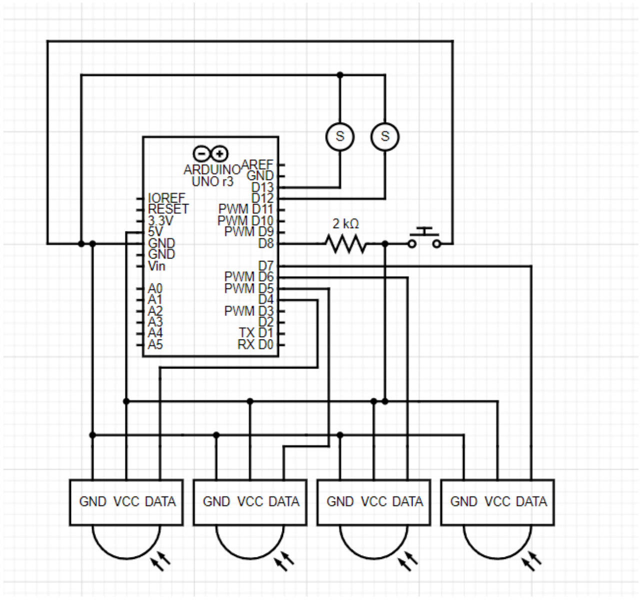

Figure 1. Arduino Circuit Diagram

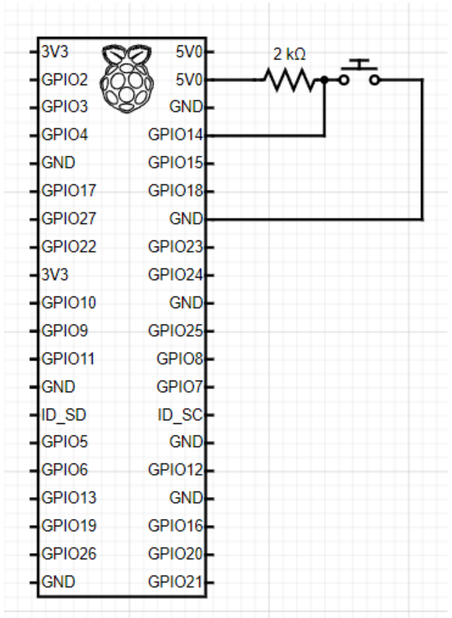

Figure 2. Raspberry Pi Circuit Diagram

The Arduino was utilized for motor control and surface lighting detection. Two servo motors were used for driving the robot, and four QTI sensors were used for detecting the lighting of the surface underneath it. Additionally, a button was connected to allow the user to manually turn on the robot. To allow this part to work, a 2kΩ resistor was also connected to the button to create a pull-up resistor.

The Raspberry Pi managed the QR code detection and data processing. To read these codes, a camera was connected via the Pi's USB port. In order to start the program, the Pi is connected to the same button as the Arduino. The connections are placed similarly to the Arduino, as shown in its respective circuit diagram.

II) *Mechanical Design*

A preexisting robot was used as a base for the prototype. This consisted of Parallax's Arduino Robot Shield Kit (SKU 32335, Parallax Inc.) combined with the QTI Line Follower AppKit (SKU 28108, Parallax Inc.), which had been built according to the instructions. Spare parts provided in the kit were used to attach several other components needed for the prototype; this will be further elaborated on in the Fabrication and Assembly section.

Aside from the provided mechanical parts, there were two essential parts that needed to be manually designed: a mount for the camera and the portable battery. These parts were developed in SolidWorks (Dassault Systèmes) and are shown in Figures 3 and 4. The camera mount is designed to fit the camera's square-shaped clamp, which attaches to the front of the mount. On each side, one hole is left to fit through the spacers on the front of the robot. The portable battery mount includes a flat surface with mounting points for the Raspberry Pi, with a perpendicular slot for the portable charger on the side.

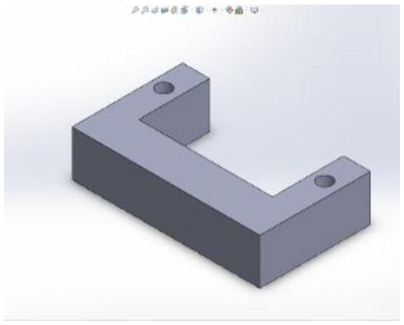
Figure 3. 3D Printed Camera Mount

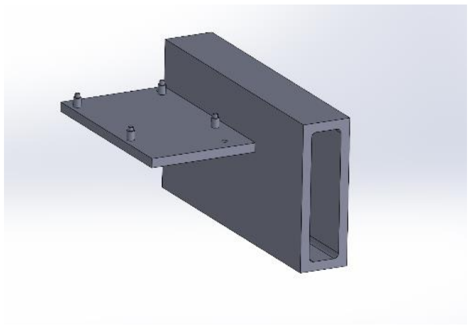
Figure 4. 3D Printed Battery Mount

III) *Software Design*

Two separate programs will be discussed in this section: the Arduino and the Raspberry Pi. Their respective flowcharts are shown in Figures 5 and 6.

Before writing the Arduino program, the appropriate servo values for each line following state needed to be determined by finding the values for which the bot would move forwards, backwards, and turn left and right. Once these values were determined, they could be used to control the motion of the robot. When the start button is pressed, an internal counter is reset to zero. The code then enters a while loop that only terminates when the counter equals two. Each QTI sensor returns a value of 1 if it reads a black surface, or 0 otherwise. For each combination of 1s and 0s produced by the four sensors, the robot adjusts its position to follow the black line accordingly. If the sensors read 1111, this indicates that a border has been reached and the counter increments. When the counter is 1, the bot begins to move backwards, again using the same line following technique. After reading 1111 for a second time, the counter increments to 2. The robot then drives forward for one second to recenter itself over the line and then stops, having completed one full loop.

On the Raspberry Pi, a button press triggers a loop that creates a new CSV File and prompts the camera to continuously check if a QR code is within view using OpenCV. Once a QR code is detected, a box is placed around it and data is displayed beneath it. If the data is successfully read, it will be organized into categories previously specified by the user. Data will continue to be appended to the existing file until either the button is pressed again or the camera has not detected a new code for 30 seconds – whichever comes first. When the user wants to export data from the Pi, they can remotely connect via SSH using their PC and navigate to the correct file location to view the data in each of the CSV files.

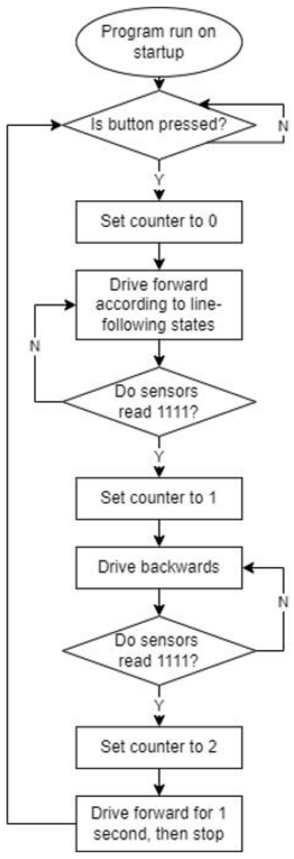

Figure 5. Arduino Flowchart

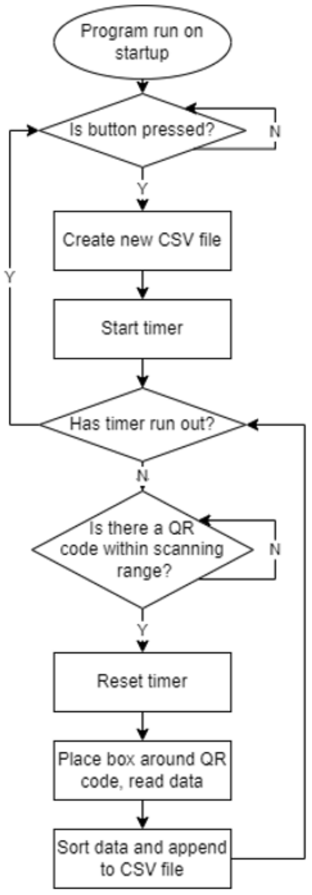
Program run on startup
Is button pressed?
N
Y
Create new CSV file
Y
Start timer
Has timer run out?
N
Is there a QR code within scanning range?
N
Y
Reset timer
Place box around QR code, read data
Sort data and append to CSV file

Figure 6. Raspberry Pi Flowchart

B. *Fabrication and Assembly*

The prebuilt robot had already included a 5V battery case, mini breadboard and the Arduino with the Parallax Shield mounted on top. As part of the modifications, 10 spacers, 1 washer, 2 nuts, and 8 screws were used to mount the Raspberry Pi on top of the Parallax Shield and through the base of the battery mount. The camera mount was attached to the front of the robot through the front spacers as shown in the diagram.

On the Arduino, each leg of the button was connected to: 5V via a 2 kΩ resistor, GND (Ground), and digital output pin 8. The QTI sensors were connected to 5V, GND, and digital output pins 4-7. The 2 servo motors were connected to the two servo slots on the Parallax shield, which correspond to output pins 12 and 13 on the Arduino.

The button was also connected to the Raspberry Pi in a similar manner: 5V via a 2 kΩ resistor, GND, and GPIO pin 14. The camera and battery were both connected via USB ports on the Pi (USB-A for the camera, and micro-USB for the battery). Figure 7 shows the fully assembled prototype. Additionally, a cost breakdown for the components is provided in Table 1.

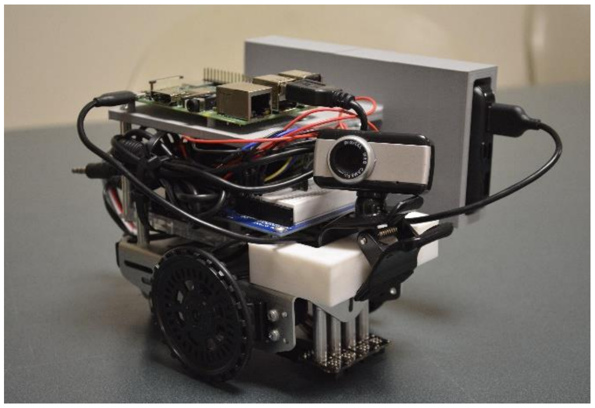
Figure 7. Inventory Management Robot Prototype

Table 1: Cost Breakdown

Part	**Cost**
Sumobot Kit (includes all mechanical and electrical components, plus Arduino and Parallax shield)	$44.95
Raspberry Pi	$35
USB camera	$12
External battery (for Raspberry Pi)	$11
Total Cost	**$102.95**

C. *Testing Procedures*

The first test to be performed was a lighting test on the QTI sensors using a lux meter. The objective of this was to determine the lowest light level at which the QTI sensors could properly distinguish light from dark surfaces. At every interval (each one spaced 5 lux apart), the bot was run three times on the test track. If the bot failed to complete one lap properly across the surface, this was considered negative, whereas a successful run was considered positive.

The second test evaluated the QR scanning ability of the camera. Specifically, the goal was to determine its effective scanning distance. To test this, a ruler was laid out on the floor from the QR code towards the bot, starting from outside of its effective range and slowly moving the bot towards the QR code until it was detected by the camera.

Lastly, to test the general effectiveness and accuracy of the prototype, the same test track from the first test was utilized. This track consisted of a 35 cm black line bordered by 3 cm thick black strips at the edges, with 3 QR codes attached to a nearby wall about 15 cm away. The bot was powered on at the start and run 5 times to determine how consistently the bot would read the codes as it traveled across the surface.

III. Results

As a result of the first test, the QTI sensors were found to be reliably effective at lux levels equal to or greater than 35. At a lux of 30, the sensors failed one out of the three tests, thus rendering them only 66% effective at that level. However, a room with a lux of that level is quite dim. Thus, for all practical purposes, the sensors would work effectively in any warehouse or store, which is usually well-lit and far above 30 lux.

The effective scanning distance of the camera was determined to be between 8 and 52 cm. This is a reasonable range for the camera, as the bot should neither be too close to the shelf (so as to not bump into the products), nor too far away (so as to not occupy too much space in an aisle). Furthermore, the position of the QR code on the camera did not affect its ability to detect it, so long as the entire code was in the camera's frame. However, the camera was unable to detect multiple QR codes in frame at once. Moreover, items should be placed with this in consideration, and the ideal camera distance should be adjusted so that only one code is read at a time.

As for the test runs, it was found that the bot consistently read all the QR codes in every trial. Thus, for that specific small-scale demo, it can be said that the robot was 100% effective. However, more varied tests will need to be considered for a more accurate representation of the prototype's effectiveness.

IV. Conclusions and Future Directions

In summary, a functional proof-of-concept was created; the prototype successfully achieved the task of being able to move along shelves, detect QR codes, and record and export the data to the user's computer in a CSV file. The cost was also kept within a reasonable range to accommodate most budgets. However, there were several issues that were encountered over the course of the project which resulted in certain objectives not being met, but that could be improved upon in the future.

The main issue with the design was that the Arduino and Raspberry Pi had no integration with each other, which lead to issues when deciding how the Pi would know when to stop adding data to the current file and create a new one for a new run. Ideally the Pi would be able to detect when the Arduino had stopped moving, but this was not possible due to the aforementioned issue. Because of this, the Pi depends on the button and also a hard-coded timer that deletes the previous data if a QR code is not scanned for a certain interval (in the code, this is set to 30 seconds by default). In practice, this approach works since the user can change the hard-coded value, but it is not as user-friendly.

Another issue is the lack of vertical scanning range. The bot has no way of reaching higher shelves as the camera is at a fixed height. Additionally, the robot lacks a remote or automatic run option, as the button needs to be manually pressed in order for the robot to start up.

To improve the robot for future applications, all the aforementioned issues can be addressed in a variety of ways. First of all, the Raspberry Pi and Arduino could be integrated to create a computer vision processing-based approach, removing the need for lines to be drawn on the ground. This would involve using the Raspberry Pi’s camera to give position feedback to the Arduino. In regard to vertical scanning range, an elevator system, such as a rack-and-pinion mechanism, could be added to the robot. As for the remote or automatic startup, the robot could either be programmed to run at specific intervals set by the user or run manually via a mobile app that connects to the Arduino via Bluetooth. Finally, proper housing could be designed and 3D printed in order to conceal the wiring and make the button more easily accessible to the user. Overall, though the prototype achieved its intended purpose as a proof-of-concept, it is far from complete and can be improved upon to create a bigger and better product in the future.

Acknowledgment

The authors would like to thank the lab manager, Joseph Diecidue, for providing the materials necessary for this project, as well as Professor Filip Cuckov for his technical advice and suggestions for Future Directions in particular.

References

[1] S. Islam, A. Pulungan, and A. Rochim, “Inventory management efficiency analysis: A case study of an SME company”, *Journal of Physics Conference Series, 2019*, vol. 1402, no. 2. doi:10.1088/1742-6596/1402/2/022040.

[2] M. Paolanti, M. Sturari, A. Mancini, P. Zingaretti, and E. Frontoni, “Mobile robot for retail surveying and inventory using visual and textual analysis of monocular pictures based on deep learning,” Sep. 2017, pp. 1–6, doi: 10.1109/ECMR.2017.8098666.

[3] I. Ehrenberg, C. Floerkemeier, and S. Sarma, “Inventory Management with an RFID-equipped Mobile Robot,” Sep. 2007, pp. 1020–1026, doi: 10.1109/COASE.2007.4341838.

[4] M. Pakdaman, M. M. Sanaatiyan and M. R. Ghahroudi, "A line follower robot from design to implementation: Technical issues and problems," in *2010 the 2nd International Conference on Computer and Automation Engineering* (ICCAE), 2010, . DOI: 10.1109/ICCAE.2010.5451881.

[5] R. Holý, P. Bílek and L. Vopařil, "Electronic inventory in the university environment and automation using RFID technology," in *2014 International Conference on Intelligent Green Building and Smart Grid (IGBSG), 2014*. DOI: 10.1109/IGBSG.2014.6835274.

[6] M. E. Çoban et al, "Raspberry pi based robot application using QR code: QR-robot," in *2019 4th International Conference on Computer Science and Engineering (UBMK), 2019*. DOI: 10.1109/UBMK.2019.8907129.