Autonomous Trash-Collecting Robot

Trịnh Thị Khánh Nguyên
Toán 21-24, VNU-HCM High School for the Gifted
Vũ Minh Đăng
Toán 2 22-25, VNU-HCM High School for the Gifted

Instructors: **Dr. Phùng Mạnh Dương** (*Fulbright University Vietnam*), **Dr. Đoàn Nhật Minh** (*Fulbright University Vietnam*), and **Dr. Lê Quân** (*Fulbright University Vietnam*)

Abstract

In the contemporary context, the escalation of environmental pollution due to plastic waste has become a pressing concern with far-reaching consequences. Unfortunately, existing automatic trash-collecting robots in the market are characterized by limited accessibility and complexity. Akshay Badkar, in the study titled "Advantages and Disadvantages of Automation" (July 4, 2023), highlights that previous attempts at deploying automated systems have not sufficiently addressed optimization in terms of cost and hardware. The objective of this research is to enhance and develop a more optimized robotic system by simulating obstacle avoidance algorithms through GazeboSim operated by Jetson Nano and retraining the object detection model using the OpenCV module. The initial steps involved constructing a mechanical design and deploying electronics. Subsequently, motion algorithms were implemented to facilitate the core functions of the robot. Despite challenges such as the unresolved simultaneously localizing problem, our robot exhibits the capability to detect, collect bottles, and navigate while avoiding obstacles. Key components of the system, including capturing real-time video, bottle recognition, obstacle avoidance using Lidar, and serial communication between the Jetson Nano and the Arduino Uno, have been successfully established. Notably, the bottle detection program has been optimized through the successful training of the TinyYOLOv3 model, resulting in enhanced detection speed and efficiency. Additionally, a bespoke formula for approximating the distance to a target bottle has been developed as part of the optimization process.

Introduction

In the contemporary era, the escalating pollution stemming from plastic waste poses profound environmental ramifications. However, the existing landscape of automatic trash-collecting robots is characterized by limited accessibility and inherent complexity. As indicated in the study titled "Advantages and Disadvantages of Automation" (July 4, 2023) authored by Akshay Badkar, automated systems, although previously deployed, have yet to undergo optimization concerning cost and hardware considerations.
The overarching objective of this research is to enhance and devise a meticulously optimized robotic system. This endeavor involves the meticulous design of the physical frame utilizing Fusion 360 and AutoCAD software, followed by assembly through 3D printing and CNC machines. Subsequently, the research methodology encompasses rigorous testing and simulation of obstacle avoidance algorithms facilitated by the Robot Operating System package, all executed under the auspices of Jetson Nano. Finally, the object detection model undergoes retraining through a preconfigured dataset for object recognition, supported by OpenCV and employing transfer learning methodologies.

Methodology

Part 1: Mechanical Design

1.1 Measurements

Initially, precise measurements of the length and width of essential components for the robotic system were undertaken. The objects subjected to measurement encompassed:

- Water bottles
- A rechargeable (portable) charger
- Two battery holders, each containing two 18650 batteries
- A Jetson Nano compute
- A1M8 Lidar
- DC motors
- An Arduino Uno board
- A USB Camera
- A rotating front blade to collect water bottles

Detailed measurements of the above components can be found in this repository.

1.2 Design

Following the acquisition of data pertaining to the requisite components for integration into the robotic system, the subsequent step involved the initiation of the design process for a three-dimensional (3D) model delineating the framework of the robot. The 3D model file can be found in this repository.

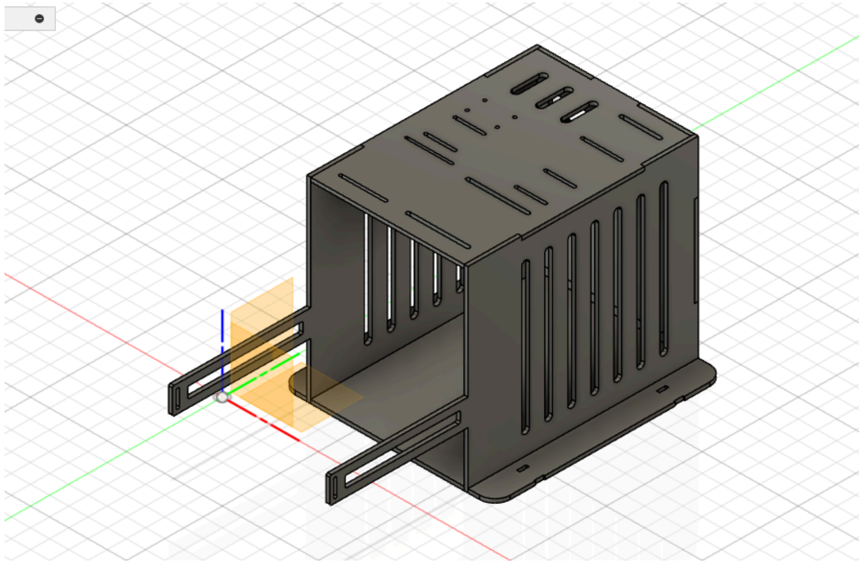

Figure 1.2. The robot's frame

Part 2: Electronics

In this section, each electronic component utilized in our robotic system will be introduced, accompanied by a detailed explanation justifying the selection of each component.

2.1 Overview

The robotic system incorporates two processing units: a Jetson Nano computer and an Arduino Uno board. The locomotion system features six direct current (DC) motors with a voltage range of 3V–9V, connected to the Arduino Uno through two L298N drivers. Specifically, four motors are allocated for the control of the robot's wheels, while the remaining two govern the front blade's movement. Additionally, a lidar system is integrated to facilitate obstacle avoidance, accompanied by a strategically positioned camera designed for efficient bottle detection. The selection of each component is grounded in a meticulous consideration of their individual functionalities, aligning with the specific operational requirements of the robotic system.

	Component	Function
Controllers	Jetson Nano	Process sensor's information and send commands to Arduino Uno via serial communication
	Arduino Uno	Receive serial commands and control motors' direction
Drivers	L298N motor driver	Control motors' speed via pulse-width modulation
Actuators	DC motors	Run the robot's wheels and rotating blade
Sensors	SLAMTEC RPLidar A1M8	Locate obstacles and record their data
	Logitech C270 Webcam	Capture real-time video

2.2 Controllers

In the operational framework of the robot, the Jetson Nano functions as the central hub for receiving sensor information, processing the acquired data, and subsequently transmitting commands to the Arduino Uno through serial communication. Following this communication, the Arduino Uno assumes the responsibility of processing these received commands to initiate the requisite motor actions within the system. This collaborative interaction between the Jetson Nano and Arduino Uno ensures the seamless execution of motor control based on the processed sensor information.

2.2.1 Jetson Nano

In the preliminary stages of selecting the computer for the robot, a deliberation ensued between opting for a Raspberry Pi 3 or an NVIDIA Jetson Nano. Subsequent to delineating the robot's objectives and specifying a training model, the decision favored the adoption of the Jetson Nano as the controller. A comparative analysis reveals that the Raspberry Pi 3 features a 4x ARM Cortex A53 CPU processor clocked at 1.4 GHz and a Broadcom VideoCore IV GPU operating at 250 MHz. Conversely, the Jetson Nano is equipped with a Quad-Core ARM Cortex-A57 64-bit CPU running at 1.4 GHz, complemented by a 128-core NVIDIA Maxwell architecture-based GPU. Benchmarks conducted by Arnab Kumar Das substantiate the discernible superiority of the Jetson Nano in terms of computational capabilities.:

Model	Application	Framework	NVIDIA Jetson Nano	Raspberry Pi 3
ResNet-50 (224×224)	Classification	TensorFlow	36 FPS	1.4 FPS
MobileNet-v2 (300×300)	Classification	TensorFlow	64 FPS	2.5 FPS
SSD ResNet-18 (960×544)	Object Detection	TensorFlow	5 FPS	DNR
SSD ResNet-18 (480×272)	Object Detection	TensorFlow	16 FPS	DNR
SSD ResNet-18 (300×300)	Object Detection	TensorFlow	18 FPS	DNR
SSD Mobilenet-V2 (960×544)	Object Detection	TensorFlow	8 FPS	DNR
SSD Mobilenet-V2 (480×272)	Object Detection	TensorFlow	27 FPS	DNR
SSD Mobilenet-V2 (300×300)	Object Detection	TensorFlow	39 FPS	1 FPS
Inception V4 (299×299)	Classification	PyTorch	11 FPS	DNR
Tiny YOLO V3 (416×416)	Object Detection	Darknet	25 FPS	0.5 FPS
OpenPose (256×256)	Pose Estimation	Caffe	14 FPS	DNR
VGG-19 (224×224)	Classification	MXNet	10 FPS	0.5 FPS
Super Resolution (481×321)	Image Processing	PyTorch	15 FPS	DNR
Unet (1x512x512)	Segmentation	Caffe	18 FPS	DNR

Source: Nvidia Jetson Nano Review and Benchmark (arnabkumardas.com)

Given its superior processing speed, particularly for tasks involving image processing, object detection, and segmentation, the Jetson Nano exhibits heightened efficacy in executing code programs, particularly those related to bottle detection. The enhanced computational capabilities of the Jetson Nano render it well-suited for optimizing and executing algorithms associated with image analysis and recognition, thereby contributing to superior performance in tasks requiring intricate visual processing, such as bottle detection.

2.2.2 Arduino Uno

The selection of the Arduino Uno for controlling the direction of the robot's motors and facilitating serial communication with the Jetson Nano is underpinned by several strategic considerations. The Arduino Uno is adept at managing high power requirements, exhibiting broad compatibility with diverse drivers and motor control circuits. Additionally, its utilization is fortified by an extensive and user-friendly support library embedded within the Arduino Integrated Development Environment (IDE). These attributes collectively make the Arduino Uno an optimal choice, aligning with the specific operational demands of the robot's motor control system.

2.3 Drivers

The chosen driver serves the purpose of regulating motor speed through pulse-width modulation (PWM). The ubiquitous L298N drivers have been selected for their capacity to manage high voltages, incorporate circuit protection mechanisms, and seamlessly integrate with the Arduino Uno. These attributes collectively make the L298N drivers well-suited for the task of motor speed control within the robotic system, ensuring efficient and reliable modulation through PWM techniques.

2.4 Actuators

The 3V–9V DC motors used in this robot are widely used in automatic robots, are easy to use, and are compatible with the L298N drivers.

2.5 Sensors

2.5.1 Lidar

For the purposes of obstacle detection and avoidance, a lidar system has been employed. After careful consideration of the task's low complexity and the resources at our disposal, the SLAMTEC RPLidar A1M8 was chosen for its attributes of stable accuracy, straightforward construction, and cost-effectiveness. This selection was made based on a judicious assessment of the specific requirements of the robotic system, where the SLAMTEC RPLidar A1M8 is deemed suitable for providing reliable and affordable lidar capabilities for obstacle detection and avoidance.

2.5.2 Camera

The detection of plastic bottles is facilitated through the integration of a Logitech C270 webcam, known for its standard specifications, capable of capturing up to 30 frames per second (fps). This webcam transmits information to the Jetson Nano via USB connectivity. The selection of the Logitech C270 is grounded in its compatibility, reliable performance, and the requisite frame capture capabilities essential for accurate detection of plastic bottles within the operational framework of the robotic system.

2.6 Batteries

2.6.1 Circuit power supply batteries

To power the entire circuit, a configuration of four rechargeable 18650 AA batteries has been employed.

2.6.2 Jetson Nano computer power supply

The Jetson Nano computer is powered through a micro USB port utilizing a portable charger.

Part 3: Software and Algorithms

In this section, the primary objectives for the robot encompass three main tasks: recognizing water bottles, avoiding obstacles, and orchestrating the robot's operations. The codes pertinent to these tasks are accessible in the designated repository.

3.1 Booting the Jetson Nano

To initialize the Jetson Nano for integration into the robot, the following installations were executed:

- Ubuntu Bionic Beaver operating system on the Jetson Nano
- ARM64 architecture Visual Studio Code IDE
- Arduino IDE 1
- ROS Melodic, RPlidar package, Hector Slam, and Turtlebot 3 Simulation
- Python 2.7 (used for ROS Melodic), Python 3.8 (used for plastic bottle recognition), OpenCV, ImageAI, and other computer vision supporting packages:

ImageAI/requirements.txt at master · OlafenwaMoses/ImageAI (github.com)
ImageAI/requirements_gpu.txt at master · OlafenwaMoses/ImageAI (github.com)

3.2 Obstacle Avoidance Algorithms

Initially, two approaches were considered for leveraging Lidar technology in obstacle avoidance:

1. Pre-scanning and Algorithmic Navigation:

The first approach involved the pre-scanning of the area map using Lidar technology. Subsequently, an algorithm was employed to guide the robot through the pre-established map.

2. Real-time Lidar Scanning with Dynamic Obstacle Avoidance:

The second approach utilized Lidar to dynamically scan data in real-time as the robot moved. Simultaneously, an obstacle avoidance algorithm was implemented to respond dynamically to the evolving environmental conditions encountered during the robot's movement.

3.2.1 Mapping the area

To facilitate the scanning and storage of the area's map in the .tiff file format, the installation of ROS Melodic, inclusive of the RVIZ and Hector SLAM packages, was undertaken. These packages play a pivotal role in visualizing input data from the lidar and preserving it for subsequent use.

For remote control of the robot, RealVNC software was installed to enable access to the Jetson Nano. This allowed for the transmission and reception of data from a remote location. The robot's movements were remotely managed through designated functions such as forward(), backward(), turnLeft(), turnRight(), rotateLeft(), and rotateRight(). Following the acquisition of the area's map, Gazebo, integrated into ROS, was employed for simulating obstacle avoidance algorithms within cost map layers.

This methodology proves to be optimal on various fronts, allowing for the overlay of multiple cost map layers to navigate and avoid obstacles within a predefined area.

3.2.2 Getting real-time data

For real-time lidar data acquisition, the iter_scans() function from the rplidar package is employed. This function provides distance information to the nearest obstacle across angles spanning from 0 degrees to 360 degrees. The algorithm is executed directly while concurrently scanning the data, allowing for swift deployment. Furthermore, this approach enables seamless integration within the same program as the bottle detection code through multi-threading.

3.2.3 Algorithm

Ultimately, live data collection was opted due to its time-optimized nature, yielding results with a satisfactory level of accuracy. The lidar angle error is constrained within 1 degree, while the distance error is limited to 2 cm. The specific areas where data is collected and the subsequent robot actions, particularly in response to obstacles within those areas, are elucidated in the accompanying diagram:

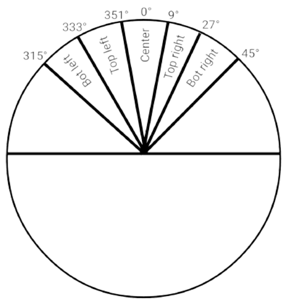

Figure 3.2a. Angle range divisions

Upon segmenting the obstacle detection areas, the robot will execute the obstacle avoidance algorithm in accordance with the following flow chart:

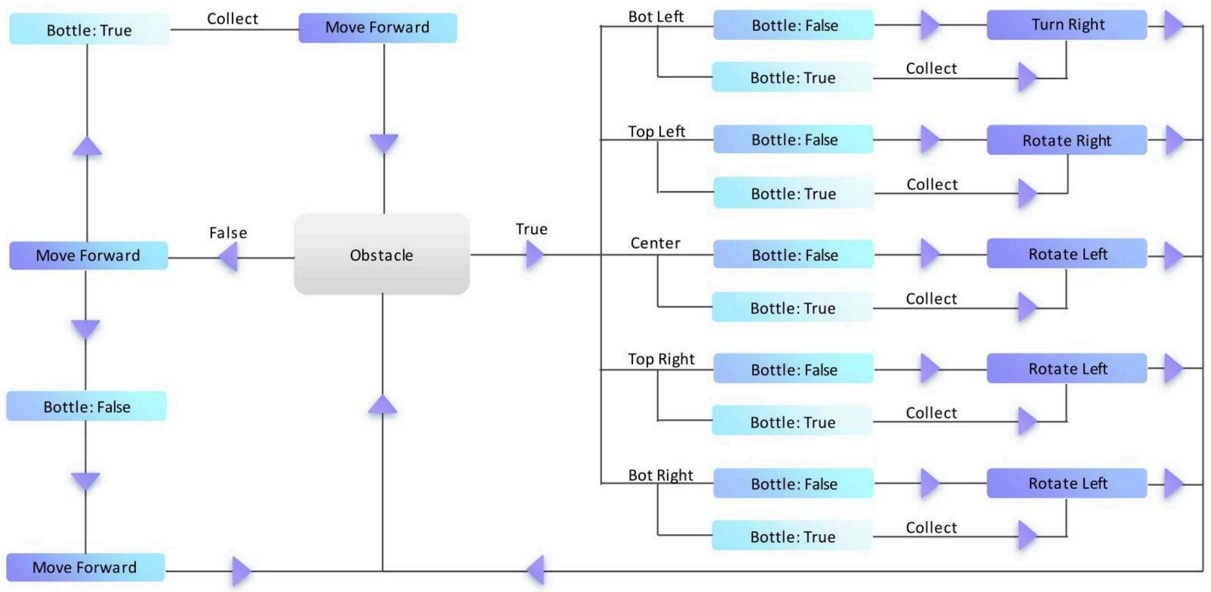

Figure 3.2b. Algorithm flow chart

3.3 Bottle Detection Algorithms

A distinctive feature of the robotic system lies in its ability to detect bottles through real-time video input from the camera. This functionality is achieved through the utilization of the OpenCV library for video capture, coupled with the ImageAI library and the YOLO object detection model for the recognition of bottles in the video feed.

3.3.1 Capturing Real-time Video from the Camera

Real-time video capture from the camera is facilitated by the OpenCV library in Python. The camera is interfaced using VideoCapture(), and frames are systematically captured through the looped read() function for individual frame processing. The imshow() function is employed to display the frames within a computer window, providing a convenient means to monitor camera activity.

3.3.2 Object Detection Models

For the task of water bottle detection, the ImageAI library is enlisted, employing the YOLO object detection model. Initial assessments on a personal computer using YOLOv3 achieved a recognition rate of 4 frames per second (fps) and a confidence range of 93% to 98%. However, upon attempting YOLOv3 execution on the Jetson Nano, the detection speed experienced a notable decrease to 0.6 fps, resulting in a noticeable delay of approximately 9-15 seconds. Due to its suboptimal detection speed and consequential latency, this model is deemed impractical for integration into our robotic system.

In pursuit of improved performance, TinyYOLOv3, recognized as a lighter version of YOLOv3, underwent testing on a personal computer. The respective detection rates and file sizes for each model are presented below:

Model	Detection rate (fps)	File size (MB)
YOLOv3	49	237
TinyYOLOv3	277	34

This model attains a recognition speed of up to 30 frames per second (fps) on a personal computer. However, the detection probability experiences a reduction to the range of 60% to 72%. In a concerted effort to enhance accuracy, we further trained the TinyYOLOv3 model utilizing a dataset composed of water bottle images obtained from Roboflow, curated by Vnu (December 2022). The resultant trained model maintains a comparable detection rate of 25-30 fps, with a noteworthy improvement in the detection probability to 75%-81%. This model effectively strikes a balance between the speed and accuracy required for the water bottle recognition task.

Subsequently, the trained TinyYOLOv3 model was executed on the Jetson Nano, where it achieved a detection speed of 12-18 fps—marginally lower than its performance on a personal computer. Nevertheless, the recognition probability remains stable within the range of 75%-81%. This model is deemed optimal for the specific demands of the bottle detection task on the Jetson Nano.

Model	Detecting rate (fps)	Precision (%)
YOLOv3	4	93 - 98
YOLOv3 - trained	0,6	90 - 97
TinyYOLOv3	30	60 - 72
TinyYOLOv3 – trained (final)	12 - 18	75 - 81

3.3.3 Calculating the Distance from the Robot to the Bottle

Upon successfully detecting a water bottle, it becomes imperative for the robot to ascertain the distance to the bottle, crucial for estimating the time required for the robot to reach the target. Given the fixed position of the camera on the robot, the image captured by the camera serves as a basis for distance calculation.

Initially, the image projection method was employed. By calibrating the camera, distortion parameters of the image were determined and subsequently applied to the distance calculation formula. However, the outcomes yielded an error margin of up to 60 centimeters (cm). Consequently, the team decided to devise a bespoke method for calculating the approximate distance, encompassing the following steps:

- Create a coordinate plane on the camera frame: Take the horizontal line at the bottom edge of the frame as the *x*-axis; the line dividing the frame vertically in half as the *y*-axis; and the intersection of the two lines is the origin O, which is considered the center of the robot.
- Divide the frame into 10 equal parts along the *y*-axis, and find the real-life length of the parts by placing a ruler right at the dividing line. After doing this, we put the data into the function equation finder tool to find d_y, the real-life distance from the object to the real point of O if the object is placed on the *y*-axis,
$$d_y = \frac{(309.059 \cdot e^{1.04215y} + 130.13)}{10}\ cm$$
where y is the distance in pixels from the object image to the *y*-axis.
- We can calculate the angle between the object's image and the origin O by using the equation
$$\alpha = tan^{-1}\left(\frac{y}{x}\right)$$
where x is the distance in pixels from the object image to the *x*-axis.
- From there, we find the approximate distance, d, from the object to the robot's center
$$d = \frac{d_y}{cos(\alpha)}\ cm$$
This method gives optimistic results with an error bound of less than 10cm. The coordination axes, variables, and calculations of the above formula can be visualized with the following diagrams:

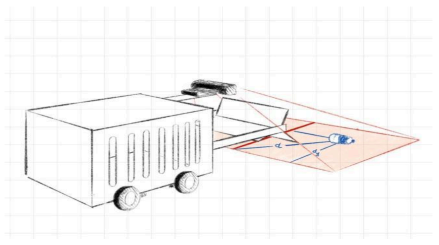

Figure 3.3a. Camera's capture area

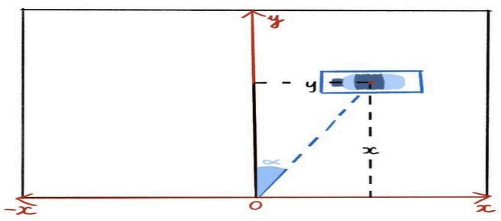

Figure 3.3b. A bottle's picture on the camera's frame

3.4 Multi-threading and Controlling Motors

3.4.1 Multi-threading

In order to concurrently execute the bottle detection and obstacle avoidance programs, both functionalities have been compartmentalized into threads, and multi-threading has been implemented. This decision is grounded in the recognition that our tasks can be effectively performed collectively on a single CPU core. To optimize processing time, the choice was made to implement multithreading, thereby facilitating parallel execution of the two programs.

3.4.2 Serial Communication with Arduino Uno

Jetson Nano and Arduino Uno communicate via the 9600 baud rate serial port. Below is a list of commands we used for serial communication between the Jetson Nano and Arduino Uno:

Command	Sent from	Action
forward, backward	Jetson Nano	Moving forward, backward Moving forward, backward
collect	Jetson Nano	Moving forward and activating the blade to collect bottle
turnLeft, turnRight	Jetson Nano	Turn left, right
rotateLeft, rotateRight	Jetson Nano	Rotate left, right
stop	Jetson Nano	Stop current action
done executing	Arduino Uno	(For functions in seconds) alerting that action is finished

Results

1. Efficiency of the Autonomous Robotic System

A novel robotic system was engineered from the ground up to autonomously manage waste retrieval in public spaces. The sturdy design facilitated the collection of debris of diverse sizes across various surfaces. Utilizing sensor technology and camera-based image processing, the robot could navigate, evade obstacles, and detect waste efficiently.

A comparative table was conducted on YOLOv3 and Tiny-YOLOv3 computer vision models. The final model accurately categorized all detected waste objects within the frame as 'Trash,' providing precise localization and confidence metrics.

Upon waste detection, the robot adeptly navigated to the target, employing circum-navigation techniques to bypass obstacles. Future endeavors entail prototyping and system testing with expanded datasets within a university campus setting.

To further enhance the project, several features could be incorporated. Following the collection of non-biodegradable waste, garbage segregation could be refined into distinct categories such as glass, metal, plastic, cardboard, and paper. This would facilitate tailored compression techniques for streamlined recycling operations, enabling the robot to handle larger volumes of waste per unload cycle. Additionally, programming the robot to collect both biodegradable and non-biodegradable waste, followed by additional segregation if feasible, would broaden its waste management capabilities.

Moreover, continued training of the deep-learning model on substantial datasets while mitigating overfitting could enhance performance. Enhanced software algorithms could be implemented to bolster obstacle detection capabilities, ensuring the robot's adaptability to dynamic environments.

Considering reinforcement learning for future iterations could empower the robot to autonomously learn and make intuitive decisions based on environmental cues. Furthermore, enabling the robot's front and rear wheels to pivot on their axes would facilitate omnidirectional movement while stationary, optimizing navigation efficiency. Implementing trajectory tracking mechanisms would further enhance efficiency by minimizing redundant visits to the same area, redirecting the robot to unexplored regions.

2. Cost-effectiveness Analysis

Upon completion of the internship period, the implementation of the autonomous robotic system was successfully achieved. The associated costs for the key components of the robot are presented as follows:

- Jetson Nano: \$185 to \$205 (4,500,000 to 5,000,000 VND)
- RPLidar A1M8: \$103 to \$111 (2,500,000 to 2,700,000 VND)
- Arduino Uno, DC motors, L298N drivers, wires: \$29 to \$41 (700,000 to 1,000,000 VND)
- Logitech C270 Webcam: \$12 to \$21 (300,000 to 500,000 VND)
- Frame material (mica): \$12 to \$21 (300,000 to 500,000 VND)

In contrast, commercial automatic cleaning robots available in the market are priced at approximately \$22,000 (530,000,000 VND), as evidenced by the Driverless Intelligent Cleaning Robot Commercial Industrial Floor Washer Machine Mop Vacuum Sweeping Cleaning Robot on alibaba.com. The total cost of our developed robotic system ranges from only \$341 to \$400 (8,300,000 to 9,700,000 VND). This cost optimization renders our system more economically accessible compared to existing trash collection alternatives in the market.

3. Comparison

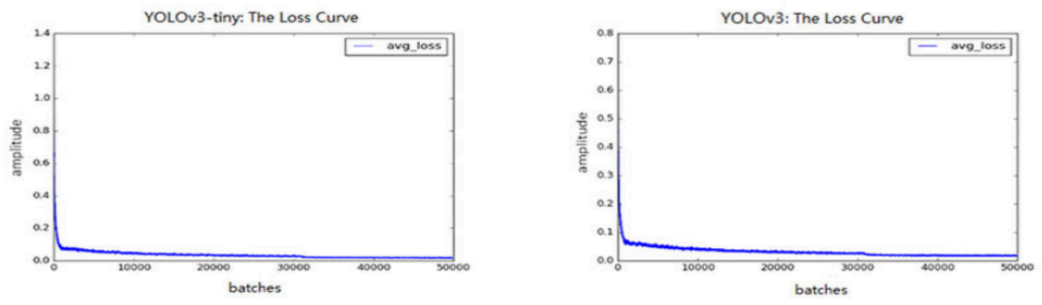

Figure 3: Loss Comparison

Upon completion of model training on the laptop, it was migrated to the Jetson Nano. The Jetson Nano's utilization of the trained model, measured by frame rate (frames per second), operated at 8% (12.25 times slower) compared to the computer.

YOLOv3 models outperformed Tiny-YOLOv3 in detection speed due to their single-shot detection nature. Additionally, between the YOLO models utilized, YOLOv3 exhibited swiffer outcomes than Tiny-YOLOv3, owing to its lighter architecture. Hence, Tiny-YOLOv3 was integrated into this project, where rapid software results were essential for the robot to function comprehensively. The model performed adequately during real-world tests, with the frames per second (fps) capped at 18 to ensure consistent performance on the Jetson Nano.

The adopted path-planning algorithm enabled the robot to operate in a continuous loop, encompassing search, detection, and trash collection. This loop could be time-bounded, with a specified duration for the robot's operation. Additionally, an emergency stop button could be incorporated on the robot, allowing the program monitor to halt its operation in response to unforeseen events.

Discussion

1. **Experiment design**

The primary objective of the robot is to navigate enclosed spaces, such as classrooms and offices, to address the issue of littering, particularly with a focus on water bottles and plastic waste. The robot has been designed to autonomously collect, classify, and process plastic waste.

2. **Note**

- It is imperative to design a 3D file with a minimum division of 4 mm suitable for CNC machines.
- Proper determination of the source voltage and current intensity is crucial when supplying power to the circuit to prevent potential hazards such as damage, fire, explosion, or short circuits.
- To optimize the processing speed of the Jetson Nano, it is advisable not to use a TF memory card for running the Ubuntu operating system. Initialization of programs from the eMMC memory on the Jetson Nano board can enhance processing efficiency.

3. **Issues**

After the completion of the robot, several drawbacks have been identified, and potential resolutions are proposed for future projects:

1. SLAM Problem Resolution: The Simultaneous Localization and Mapping (SLAM) problem is acknowledged as an area for further research over an extended period.
2. Obstacle Scanning Limitation: The robot encounters difficulty scanning obstacles that are not at the same height. To address this, the obstacle avoidance method within a pre-scanned and limited area map can be employed. Alternatively, integrating an additional ultrasonic sensor communicating with the Arduino can identify obstacles at varying heights.
3. Frame Design Aesthetics and Safety: The current robot frame design presents aesthetic and safety concerns as internal components are exposed and lack protection. Potential

resolutions include redesigning the robot frame or implementing a cover to shield the components.

4. Image Detection Limitation: The current model has undergone a retraining and optimization process tailored to accommodate limited hardware capabilities, specifically designed for IoT applications. This approach acknowledges the constraints inherent in the hardware environment and seeks to maximize the efficiency and performance of the image detection functionality within these limitations. This adaptation ensures that the image detection process aligns with the practical requirements and resource constraints of the intended application in the Internet of Things (IoT) domain.

These considerations underscore the iterative nature of robotics projects, emphasizing the importance of continuous improvement and adaptation to address evolving challenges.

Conclusion

1. Demand for Garbage Cleaning Robots

As the environmental concerns associated with plastic waste escalate, the demand for automated trash cleaning robot systems has become increasingly pronounced. The overarching goal of this research is to develop robots capable of autonomously addressing and resolving the issue of environmental pollution caused by plastic waste.

2. Optimizing Garbage Cleaning Robots

The existing automatic trash cleaning robots available in the market encounter challenges in terms of accessibility and deployment. Consequently, this research is dedicated to enhancing and optimizing robotic systems with a focus on cost reduction and operational efficiency.

3. Object Detection Model’s Performance

A significant facet of this research revolves around the training and enhancement of the object detection model integrated into the robot. Employing the Tiny YOLOv3 model, notable strides have been made in improving both the speed and accuracy of bottle detection. This emphasizes a commitment to refining the technological components critical for the successful execution of the robot's waste-cleaning objectives.

Acknowledgment

We would like to express our sincere gratitude to all those who have contributed to the completion of this project.

Firstly, we truly value the guidance and aid given to us during our project by our instructors, Mr. Phung Manh Duong, Mr. Doan Nhat Minh, and Mr. Le Quan. Their insightful feedback and suggestions helped to shape the project's scope. Secondly, we would also like to express gratitude to Mr. Nguyen Thanh Long and Mr. Nguyen Quoc An, our mentors, for their unwavering assistance whenever our project faced technical issues. Additionally, we would like to thank Makerspace - Fulbright University Vietnam for giving us the supplies, environment, and resources that were crucial to the completion of our project. Lastly, we wish to acknowledge the PII coordination team for their continuous supervision and assistance during our participation in the program, as well as the PII organizing team for developing such a valuable learning and experience program.

References

Akshay Badkar. (2023, July 4). Advantages and Disadvantages of Automation. Retrieved from https://www.simplilearn.com/advantages-and-disadvantages-of-automation-article

Arnab Kumar Das (2019, May 26). Nvidia Jetson Nano Review and Benchmark. Retrieved from https://www.arnabkumardas.com/platforms/nvidia/nvidia-jetson-nano-review-and-benchmark/

Bradski, G. (2000). The OpenCV Library. *Dr. Dobb's Journal of Software Tools.*

OpenCV. (2014). The OpenCV Reference Manual (2.4.13.7). Retrieved April, 2014, from http://docs.opencv.org/

Vnu. (2022, December). *Plastic_Bottle Dataset* [Open Source Dataset]. *Roboflow Universe*. Roboflow. Retrieved from https://universe.roboflow.com/vnu-jozz8/plastic_bottle-lxw9d

Moses. (2018). ImageAI, an open-source Python library built to empower developers to build applications and systems with self-contained Computer Vision capabilities [Software]. Retrieved from https://github.com/OlafenwaMoses/ImageAI

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 779–788). Retrieved from https://doi.org/10.1109/CVPR.2016.91

Jiang, Zicong & Zhao, Liquan & Li, Shuaiyang & Jia, Yanfei. (2020). Real-time object detection method based on improved YOLOv4-tiny.

Without the contributions and assistance of all those mentioned above, this project would not have been possible. Thank you all for your valuable contributions and support.

Appendix

This research project is carried out within the PTNK Innovation Initiative (PII) Summer Research Internship Program under the topic "4d. Smart Robot" at Makerspace - Fulbright University Vietnam from June 15, 2023, to September 5, 2023.

This research project has achieved significant milestones, including:

Time period	Milestone
21/06 - 30/06/2023	Preparing and designing the robot
03/07 - 08/07/2023	Getting to know Arduino and the Ubuntu operating system on Jetson Nano
10/07 - 27/07/2023	Working on bottle detection and obstacle avoidance programs
31/07 - 14/08/2023	Optimizing the robot

With the aim of sharing knowledge, aligned with the spirit of continuous learning and accumulation in science, we would like to share the tools, knowledge, and useful keywords that have been beneficial to us during the course of this project, including:

- Fusion 360, a 3D design software
- The ImageAI library
- Depth Camera
- Lidar

At the end of the program, we have gained valuable experiences for ourselves, and we are ready to continue our path of continuous learning and research. We hope that the results we share will make a meaningful contribution to the community and will be continuously built upon.