The GOAT

Grabber Of All Trash

Raquel Gould and Abi Sullivan
6th Grade
Central Middle School
San Carlos School District

SAN MATEO COUNTY OFFICE OF EDUCATION

Abstract

Our STEM Fair project is a Proof-Of-Concept. This GOAT is nowhere near complete, it is a prototype that shows that a fully functional GOAT is possible. The purpose of the GOAT is to pick up trash, to eventually help decrease the world's litter problem. Some of the things we needed the GOAT to do were to consistently identify trash, accurately grab the trash, dump the trash into a bin, and if time permitted, move using the wheels. We built the GOAT by breaking it down into 5 modules, eventually connecting them. The modules are the controller module, which tells everything what to do, the vision module, which captures images, the detection module, which detects trash with AI, and the motion module, which controls the wheels, and which we haven't gotten to working on yet. While testing the GOAT, we found a lot of bugs along the way, but they got fixed (mostly). At the end of the day, our GOAT works. There are many things we could improve upon, but it works well enough to meet our criteria.

Identify the Need for your Prototype

The end goal for our GOAT, the Grabber Of All Trash, was for it to be a robot that would roll around along the sides of streets, and pick up trash as a way of reducing Earth's litter problem. After doing research, we came across a robot that sifts through sand to clean beaches, and a raspberry-picking robot, but nothing quite like our GOAT. For this project, we only built a proof-of-concept robot, but we plan on improving the GOAT until it is fully functional.

Design Criteria and Constraints

Some criteria we had for the GOAT were that it needed to be able to:

- Consistently identify trash
- Grab trash
- Dump trash
- If time permits, have working wheels

Some of the constraints we had for the GOAT:

- Limited time
- Limited money
- Our skill levels
- Technology we had access to

Evaluate Alternative Designs

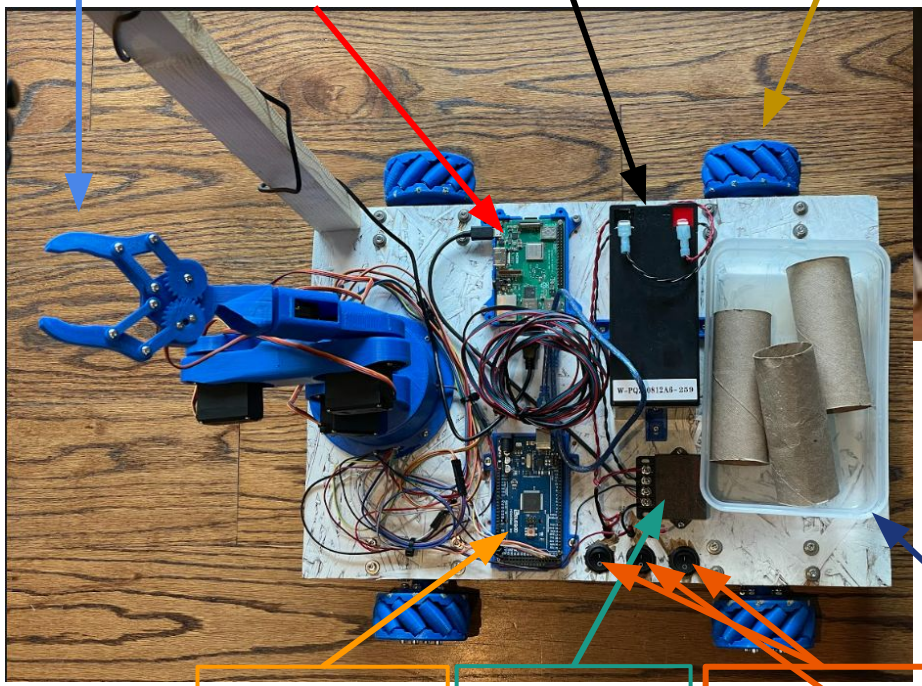
After looking into what other people have done, we discovered a robot that sifts through sand to remove plastic from beaches, a robot that picks up cigarette butts from beaches, and a raspberry-picking robot. Neither of those quite fit our criteria, so we found different already-created modules we could use to make an original GOAT.

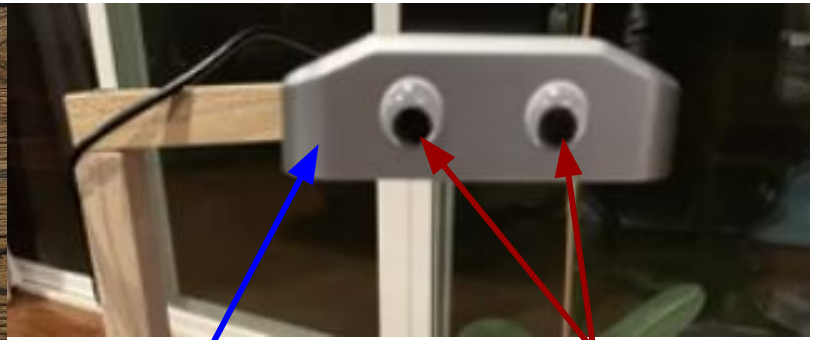
- Arm: Considered four existing designs. Chose one for its range of motion and precision (controlled by servos), and was simple to 3D print and assemble. We modified the design slightly to meet our needs.
- Wheels: Considered motors, motors with encoders, and stepper motors. We chose the mecanum wheels because they were easy to 3D print and assemble, and because they can go in any of 8 directions easily, which could work in our favor once we improve on the GOAT. We chose servos because they can go to an exact specified position.
- Controllers: There is both a Raspberry Pi minicomputer and an Arduino Mega 2560 controller on the GOAT because Raspberry Pis can handle AI and such, and Arduinos are better at interacting with the real world, and doing movement-related things.
- AI: We decided to use TensorFlow Lite for our trash detection because there were examples we could follow, and because it ran on a Raspberry Pi.

Your Prototype

The GOAT (Grabber Of All Trash)
Video Link

Your Prototype (labeled)





Links to:
Bill of Materials
GOAT code
Communication protocol

Communicate the Design

We broke the design into modules when first planning the GOAT, which are the:

- Controller module: on Raspberry Pi minicomputer, which tells everything what to do and when, and communicates with the camera, AI, and other modules on the Arduino controller. It also converts from pixels (camera image) to centimeters (arm position).
- Vision module: on Raspberry Pi, which uses the camera to capture the image of what the GOAT is seeing.
- Detection module: on Raspberry Pi, using TensorFlow Lite object detection AI, which detects if there's trash in the image captured by the vision module, and if so, where in the image there's trash. We trained it by taking over 100 pictures of trash (mainly toilet paper rolls) and annotating them to indicate where the trash was in those pictures, and then trained and validated the AI model.
- Arm module: on Arduino and Raspberry Pi, which controls the robot arm with servos and grabs the trash, and then dumps the trash into the bin. After building it, we calibrated the servo angles. It uses an existing inverse kinematics library to convert X, Y, Z coordinates into servo angles to control arm positions.
- Motion module: on Arduino and Raspberry Pi, which we did not get to. It will control the wheels, so that the GOAT can see and reach trash further away. We plan to get the wheels to move at a later date.

Some frustrations we had were:

- Assembling the wheels, which we had to put tape over the holes before gluing so that the rods didn't fall out.
- Attempting to use trash images from the TACO Dataset, which did not work for us. Instead, we took our own pictures.
- Object detection performance was not initially very good, so we'll need to take more pictures. We also did not add enough pictures of trash beside toilet paper rolls.
- The power cable to the Raspberry Pi was too thin, with too high resistance. We replaced it with a thicker one.
- 3D printing dimensions were not accurate, requiring additional sanding, drilling, and gluing.
- The inverse kinematics library seemed confusing, and we had to just go with it and it worked out.

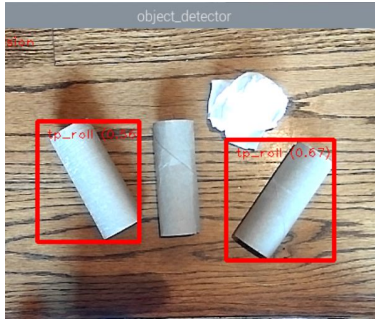
Your Test Plan

To test the GOAT, we ran different parts of the code to test them out individually.

- First we tried to use another dataset to train our AI model (we tried to use the TACO dataset), but the validation failed, and did not recognize our trash, so we trained an AI detection model on our own pictures, and tested that multiple times.
- Then we tested the communication between the Raspberry Pi and the Arduino controller.
- Then, we tested out our Arduino code to make sure we could tell each individual arm servo where to move to.
- Next, we tested IKPY (Inverse Kinematics Python Library), which found the angles that the arm needed to grab a piece of trash, by writing code telling the arm to go to a specified set of coordinates, pick up the trash, and dump it in the container.
- Finally, we tested the vision module, and the connections between everything, by running the GOAT.

Your Test Results

1. Object detection

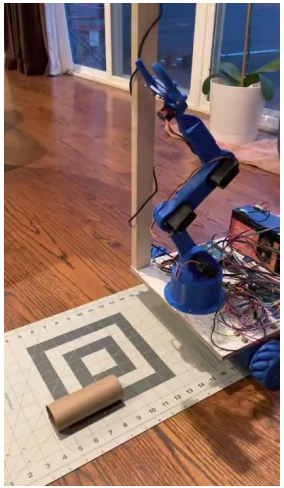


Toilet paper rolls detected in some positions, not others. Tissue not detected.

2. Arm angle control

Video Link

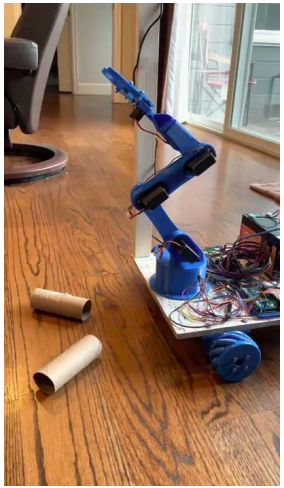
3. Inverse kinematics to pick up trash



A bug made it drop trash at wrong time.

Video Link

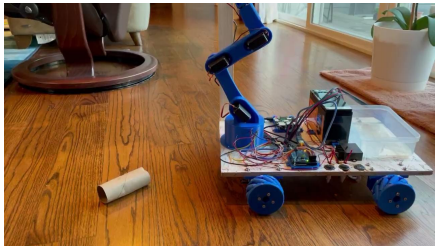
4. Picking up items



Missed trash because Y-axis was off.

Video Link

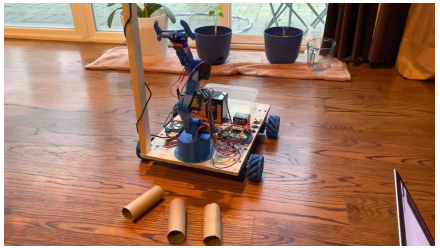
5. Picking up multiple items



A bug got it stuck in a loop.

Video Link

6. Picking up multiple items successfully



Video Link

Analysis of your Test Results

We discovered a lot of bugs while testing the GOAT.

- The TACO Dataset did not work, so we didn't use it.
- The AI only recognized toilet paper rolls that were at angles, not straight up and down, and it didn't recognize tissues very often at all. This bug is still unfixed.
- The servo calibration on the arm's servos was a bit off. For example, the position that was marked as 90° on one servo was actually the 110° position, so we correctly calibrated the servos in the code.
- The IKPY (Inverse Kinematics Python Library) was just a little bit odd when it came to the things we had to put into the code to make it work. We ended up just following the instructions of RoTechnic, and it ended up working.
- We had the arm open the claw when it went straight up, so the claw opened too early, so we had the "up" position include the claw being closed.
- The y-axis was 5 centimeters off from where it should have been, so the arm grabbed too close to the base. This happened because we miscalculated the y-axis. We fixed this by changing some of the math in the code.
- The vision took too long to update, so it kept seeing toilet paper rolls that were no longer there, so we shut down the vision after the end of a loop, and restarted it at the beginning.
- The object detection also needed to be run 5 times before it worked. We have no idea why, so we just ran it 5 times in our "run GOAT" code.
- And finally, there was a point at which the Raspberry Pi did not know what to do, because we had spelled "degrees" as "degreees." (This is Abi's favorite bug.)

Conclusions

Overall, the GOAT proof-of-concept was successful. Though we didn't have time to complete the motion module, the GOAT still works fairly well. If we were to improve on the GOAT so that it would be able to drive around on the sides of roads, picking up trash, there are some things that should be changed. For example, we could:

- Use wheels that are more suited to outdoor terrain
- Lengthen the links of the arm so that it can grab trash from farther away
- Stabilize the camera so that it is less finicky
- Take more pictures of trash, and train the AI on more types of trash
- Use a better bin to dump the trash into
- Build a closed box around the GOAT's electronics so that it is less vulnerable to damage

Acknowledgements

First and foremost, thank you to all the parents involved for funding the GOAT.

Thank you to Raquel's dad, for helping us find resources, providing supervision and helping us with drilling things, and providing support and advice whenever it was needed.

Thank you to Abi's parents for providing the base for our GOAT.

Thank you to many people for providing bits (or lots) of code for the GOAT.

Thank you to Michael Hong, who gifted us the Raspberry Pi.

Thank you to Laura, Lisa, and Connor McGann for gifting their old 3D printer.

Thank you to Clem Tillier for loaning his Dremel tool for the wheels.

Bibliography and Sources

Khan and the Tensorflow Authors: object detection AI and example code.
Dejan from How To Mechatronics: robot arm design.
Pierre Manceron: IKPY inverse kinematics library
RoTechnic: showed us how to use IKPY in his video.
The Robotics Back-End: helped us figure out how to make Raspberry Pi and Arduino communicate with each other through a USB serial connection.
Dejan from How To Mechatronics: wheels design.
Fernando Costa: 3D-print motor mount design.
All3DP: tutorials on how to make basic changes to 3D files.
Blender: free 3D modeling software, which helped us with basic changes to the arm.
Raspberry Pi and Arduino: devices and documentation.
The Raspberry Pi Guide by Jolle Jolles: additional help and tutorials.
GitHub: keeping our code safe and shareable.
TACO Dataset: images of trash that we tried to use