Design Report:
Memory-Efficient Object Detection Using Ultrasonic Rangefinders On The DE2Bot

Sushanth Penta
Eric Rafalovsky
Antony Samuel
Leo Weng

ECE 2031, Digital Design Lab
Section L01

Georgia Institute of Technology
School of Electrical and Computer Engineering

Submitted
December 6, 2016

Abstract

This paper details a memory-efficient method of detecting and tagging an unknown number of objects within a 12 foot by 8 foot arena and the process of implementing this algorithm using a DE2Bot. The DE2Bot is able to detect these objects by first traversing a centerline that splits a given arena into two equally sized rectangles. While traveling this line, ultrasonic rangefinders are used to detect objects on the left, right and directly ahead of the robot. Traversing this centerline minimizes the distance from the ultrasonic rangefinders on the DE2Bot to objects in the arena. Immediately after detecting an object, the DE2Bot will turn 90° and tag the object by touching it (or stays straight in the case of an object directly in front of the DE2Bot). After the object has been tagged, the bot returns to the home square using its internal odometry and is manually reset to avoid accumulation of errors between runs. The implementation of this algorithm is moderately successful in completing the project objectives: The robot fails to return to the home square multiple times due to imprecise 90° turning and the robot does not properly account for the conical nature of the sensors for objects at far distances - the DE2Bot would detect an object slightly off track as directly in the bot's line of motion.

Memory-Efficient Object Detection Using Ultrasonic Rangefinders On The DE2Bot

Introduction

Project Specification

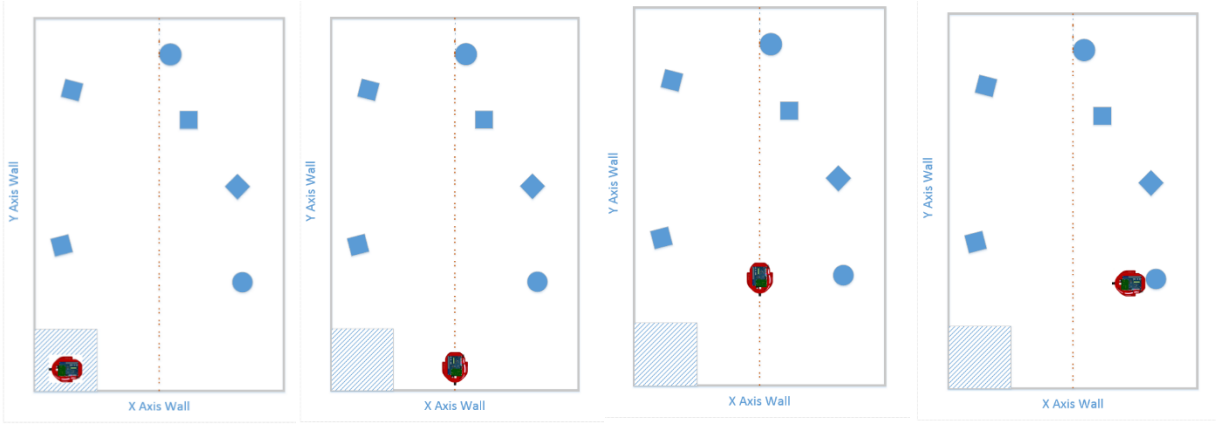
An important challenge present in robotics both small and large is object detection. Georgia Tech's ECE 2031: Digital Design Laboratory class explores this challenge by utilizing a DE2Bot, an AmigoBot with its electronics removed and replaced with custom hardware and a programmable Altera DE2 FPGA development board, to explore an unfamiliar arena. The project objectives are to detect objects randomly placed within the arena, to "tag" each object by touching it, and to travel from the tagged object back to a predetermined home square.

The arena is a 12 foot by 8 foot unobstructed rectangle besides four to six objects the DE2Bot will detect. The arena is walled on two adjacent sides and open on the other two sides with a two foot by two foot home square, an area where the robot starts and returns after tagging each object, in its walled corner. The objects placed in the arena are between 6 inches and 24 inches and will be placed at least 24 inches away from each other, 12 inches away from the boundaries, and 24 inches away from the home square. Each object is manually removed from the arena after the bot tags it.

Design Description

A software solution that runs on the Simple Computer (SCOMP) defined in the DE2 FPGA hardware is used. To find objects in the arena, the DE2Bot first traverses from the home square to the centerline that bisects the arena into two 12 foot by 4 foot rectangles while checking that no objects are in its way. To find objects in the arena, the robot travels along this centerline and then uses the ultrasonic rangefinders (sonars) to detect objects on the left, right and directly ahead of the robot. Immediately after detecting an object, the DE2Bot will turn 90° degrees and tag the object by lightly touching it. This

process is demonstrated in Figure 1. After the object has been tagged, the DE2Bot uses its internal odometry to exactly retrace its path back into the home square. After the DE2Bot returns to the home square after touching an object, the robot is then reset - a process which is repeated until all objects are tagged. Since objects are removed after being tagged, they are not tagged multiple times, and the DE2Bot avoids having to map the arena to keep track of already tagged objects. To minimize odometry error, the robot only travels parallel to the walls of the arena and only makes 90° degree turns. This ensures that odometry error is equally applied to both wheels. The robot continues to minimize odometry error by lightly tagging the objects rather than pushing through them to avoid any wheel slippage by decelerating before the robot tags a detected object.



a b c d

Figure 1. An example of object detection and tagging, including the 12 foot by 8 foot arena. Six example objects are depicted. The orange line, henceforth referred to as the center line, splits the arena into two equal 12 foot by 4 foot rectangles. In (a), the DE2Bot is initialized in the home square with its center 1 foot from the y-axis wall and 6 inches from the x-axis wall. In (b), the robot traverses to the bottom center point and turns left 90°. In (c), the robot travels along the center line until it detects an object on its left, right or immediately ahead of it. In (d), the robot turns 90° to the right, in the direction of the detected object. In (c), the robot proceeds forward after turning 90° to the right until it has tagged the object.

The design was mostly successful: while the robot was very effective in detecting and tagging objects, it struggled to return back to the home square after tagging each object. Inconsistency in turning hindered the robot's ability to return to the home square and the not accounting for the conical nature of

the sensors at far distances prevented an optimal run - the DE2Bot would detect an object slightly off track as directly in the bot's line of motion.

General Methodology

Design Choices

In the early stages of the project, different approaches were brainstormed and design choices were evaluated. The above approach was chosen because of little reliance on odometry, a minimum average distance between the robot and detected objects, and low memory requirements. Use of mapping techniques such as occupancy grids were evaluated but were rejected due to unneeded system complexity. Project specifications that dictated that objects are removed after tagging eliminated the need for mapping.

Design Implementation

The above approach was implemented through software running on the DE2Bot's Simple Computer. Assembly code was written, compiled into an Altera memory initiation file and loaded into the provided Simple Computer VHDL file. The design was implemented primarily through several subroutines which were independently implemented and later integrated. Three primary subroutines, Search, Tag and Return Home, each corresponding to the phases of the design were used. The operation of the design's subroutines is depicted in Figure 2. Information from the DE2Bot manual including the robots constant deceleration rate, was used to inform the proper implementation of the design. No external code other than the provided robot initialization code was utilized.

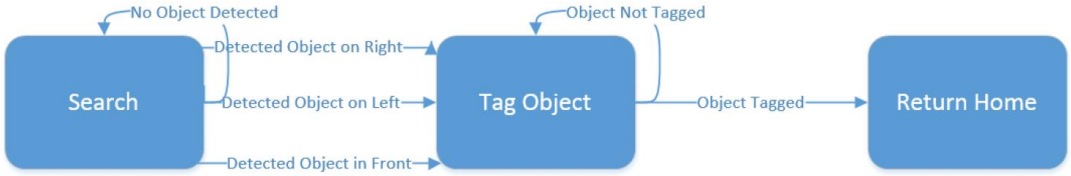


Figure 2. The three primary subroutines used in implementing the proposed design. The three main steps are: Search, Tag Objects, and Return Home.

Design Testing

Individual subroutines, as well as the final system, were tested. When possible, subroutines were tested independently prior to integration. Correct operation of each function was confirmed through visual inspection, LED output and/or beep as appropriate. Throughout debugging the program, aids such as printing data to the LCD using I/O, blinking of the LEDs and beeping of the robot were extensively used. The search subroutine was tested by placing the robot into the home square and indicating object detection 90° to the left or right with a beep. The tag subroutine was tested by aligning the robot with an arena object and testing if the robot would stop once it collided with the object. The return home subroutine was tested last, after the previous two subroutines were integrated, and was deemed successful when the robot was able to return to the home square after tagging an object using the previous two subroutines. Following subroutine testing and integration, the completed system was tested within the test space specified by the project description. The robot system was tested with objects in a variety of valid locations and orientations within the area. The system was also tested on several robots to account for robot specific tendencies. System testing resulted in minor modifications to threshold and distance parameters. While system testing was thorough, an untested edge case led to unexpected system behavior during the final demonstration.

Design Modifications

Few changes were made between the design proposal and final demonstration. Code was added to detect objects between the home square and the beginning of the search sequence. This made the design more robust and accounted for objects that could be located between the home square and the start of the search area. In the case that an object was detected, the robot would drive to it and them immediately proceed backwards into the home square. Besides this additional functionality, the speed of the robot during the search sequence was reduced from .4 m/s to .35 m/s to increase the probability of object detection.

Project Management

The project was completed by four team members. Appendix A depicts the project timeline in the form of a Gantt chart. Sushanth Penta contributed to the code as well as the design proposal. Eric Rafalovsky took the lead on programming the robot, writing the technical portions of group documents, and created the presentation. Antony Samuel also programmed the robot and wrote introductions for the design proposal and design report. Leo Weng managed the team's paperwork and contributed to the writing assignments. Github was used for code collaboration.

Technical Results

Results of the final demonstration are shown below in Table 1.

TABLE 1
FINAL DEMONSTRATION RESULTS

	Run 1	Run 2	Average
Objects Tagged	4/4	0/6	2
Returned Home	0	0	0
Pickups	4	2	3

Collisions	1	0	0.5
Remaining Time (sec)	2	90	46

During the first demonstration run, all four objects were successfully tagged, however the robot was unable to return home successfully during any of the runs. One of the four times the robot failed to recognize that it had tagged an object and the return home sequence was never initiated. The remaining three times the robot successfully was able to return to the 8 foot wall but imprecise turning as depicted in Figure 3 resulted in the robot crashing into the wall before it could return home. The group picked up the robot in these cases and manually returned it to the home square.

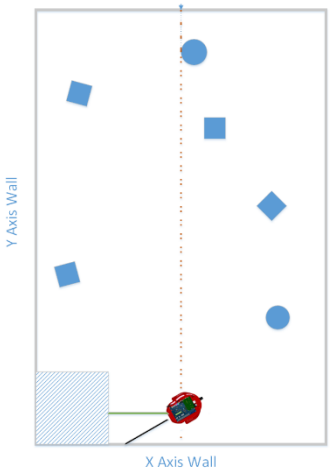


Figure 3. The robot collided with the wall when attempting to return home due to overturning. The green line indicates intended robot orientation caused by overturning.

The second run resulted in no tagged objects and no successful return homes. An untested object orientation resulted in the robot moving forward from the base, going out of bounds, instead of proceeding on the planned path. The robot drove out of the arena time before the group terminated the demonstration trial.

Conclusions

Design Strengths

Taking a non-direct path back to the home square proved advantageous. Because the robot returned home along a path it had already traveled, it could travel the path at high speed without the need for object detection or collision. Further, during testing and the first demonstration trial, object detection was shown to be very effective. This is a direct result of searching while traveling the center of the arena. This results in the minimum average distance between the sonar and arena objects and therefore minimizes sonar inaccuracy. Lastly, as the team anticipated during the brainstorming phase, the project was implemented in very little memory. Of the 2048 words of memory in the given SCOMP, only, about 15%, 303 words were used in the implementation of the design. This factor is important as mobile robotic systems such as the DE2Bot are resource constrained systems.

Design Weaknesses

The demonstration as well as testing revealed several weaknesses with the design and implementation. The design is inefficient in that non-direct paths are always used to tag objects and return home. This is a result of not mapping and exploring the arena each run without prior knowledge. When returning home the non-direct path described in the design has a path length up to 38% longer than a straight line path back home.

In addition to design weaknesses, implementation errors in turning and initially sensing objects, hindered demonstration performance. Turning 90° was implemented by running the left and right motors in opposite directions for a fixed length of time as determined by the DE2Bot timer. This turning system was imprecise and did not account for deviation caused by different positions of the rear caster. Imprecise turning led to the robot crashing into the wall rather than successfully returning home. A more precise turning method could have been implemented using the odometry information of the DE2Bot.

Further, a poor implementation of an initial check of the arena resulted in poor performance during the second demonstration trial. A failure to account for the conical shape of the sonar led the robot to think that there was an object in its path when in fact there wasn't. This led the robot to drive forward, intending to tag an object, out of the arena.

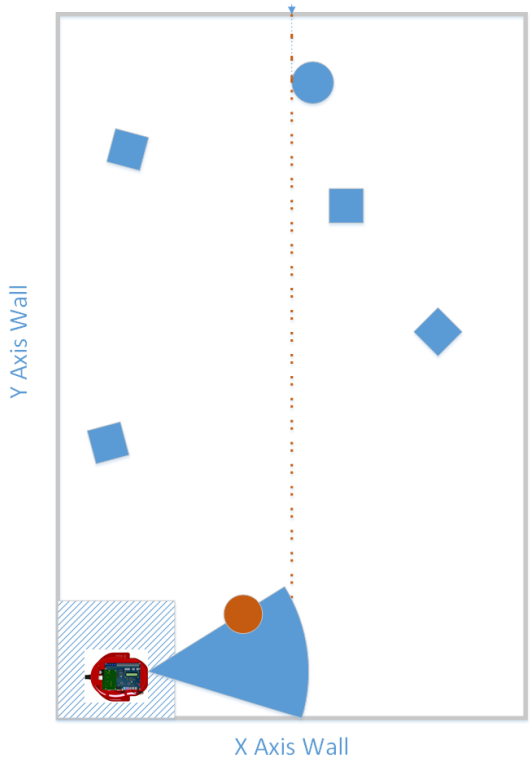


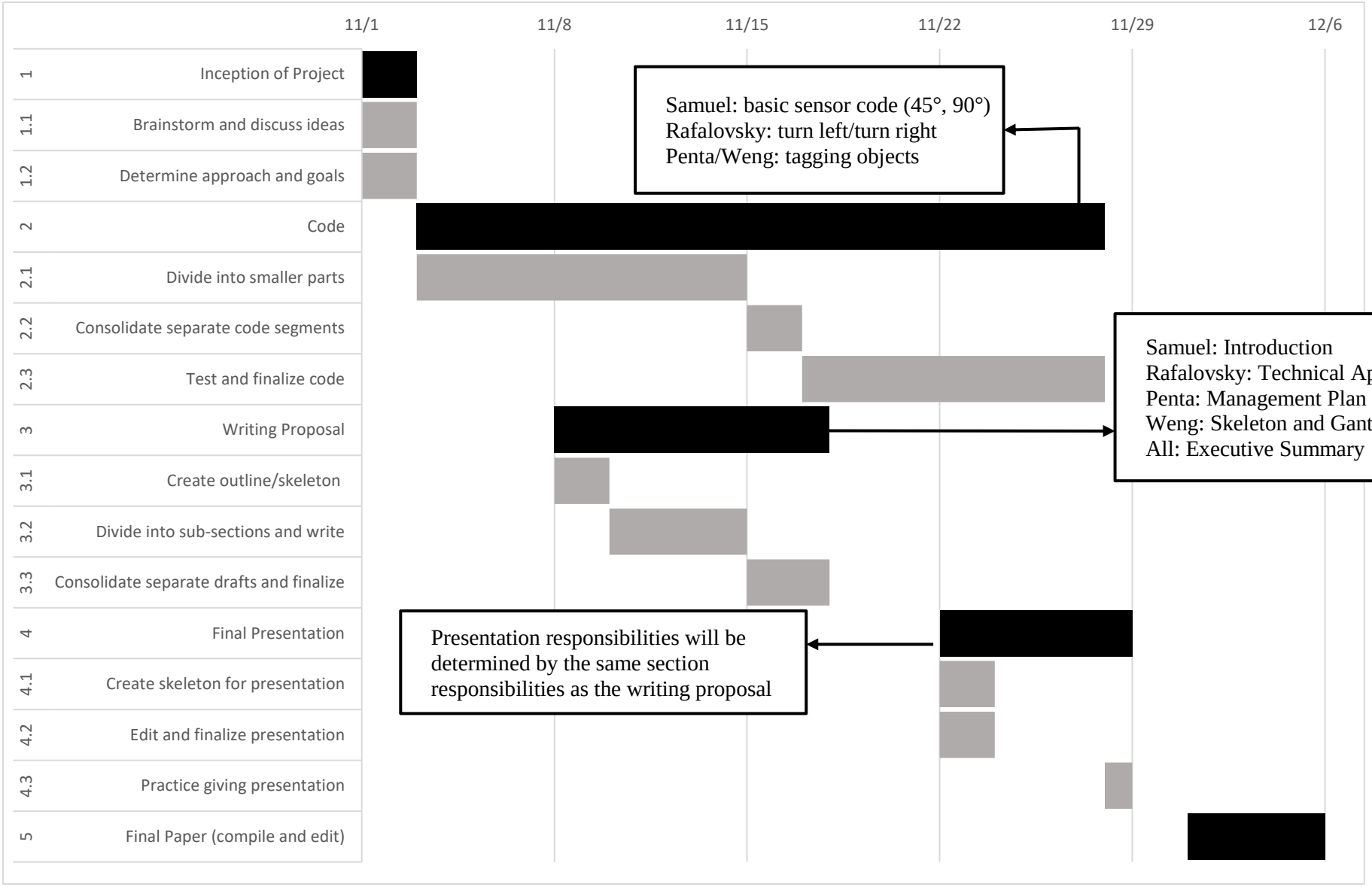
Figure 4. During the final demonstration, the robot incorrectly believed an object was in front of it due to the conical shape of the sonar. The robot drove forward attempting to tag the object but drove out of the arena boundaries.

Future Work

The demonstration confirmed that sonars on the DE2Bot can be successfully used to detect and tag objects in an arena. Techniques used to avoid odometry error proved effective should be used in other designs where odometry is relied upon. Use of the DE2Bot I/O proved extremely helpful for debugging and should be extensively utilized by those implementing systems on the DE2Bot. The demonstration also indicated opportunities for future work. Implementation of basic mapping of the arena would allow

for more direct object tagging and a more direct return home path. Storing locations of detected objects would allow the robot to avoid searching in some runs and directly proceed to tag the object. Further, taking a straight line path from the tagged object back to the home square would result in significant time savings.

Appendix A: Gantt Chart


11/1
11/8
11/15
11/22
11/29
12/6
1 Inception of Project
1.1 Brainstorm and discuss ideas
1.2 Determine approach and goals
2 Code
2.1 Divide into smaller parts
2.2 Consolidate separate code segments
2.3 Test and finalize code
3 Writing Proposal
3.1 Create outline/skeleton
3.2 Divide into sub-sections and write
3.3 Consolidate separate drafts and finalize
4 Final Presentation
4.1 Create skeleton for presentation
4.2 Edit and finalize presentation
4.3 Practice giving presentation
5 Final Paper (compile and edit)
Samuel: basic sensor code (45°, 90°)
Rafalovsky: turn left/turn right
Penta/Weng: tagging objects
Samuel: Introduction
Rafalovsky: Technical Approach
Penta: Management Plan
Weng: Skeleton and Gantt Chart
All: Executive Summary
Presentation responsibilities will be determined by the same section responsibilities as the writing proposal

Appendix B: Project Logbook
(Brainstorming Sheets and Individual Logs)