Matlab Differential-Drive Robotics Simulation Environment

James S Smith, *Member, IEEE*

Abstract—**Visualization and Simulation are key strategies for quick and effective algorithm development and problem solving. In the realm of differential-drive robotics (DDR), new path-planning and navigation algorithms are quickly emerging as intelligent systems are reimagined. In order to speed the development process and produce low-cost, realistic visualization for path-planning and navigation algorithms for DDR, a novel Matlab differential-drive robotics simulation environment is created which combines realistic simulation conditions and simplistic deployment.**

Index Terms—**differential-drive, path planning, navigation, simulation, visualization**

I. Introduction

THE recent advancements in intelligent systems brings an onset of new path-planning and navigation algorithms for differential-drive robots (DDR). As these algorithms are implemented, researchers normally develop and test strategies in a simulation environment to reduce development time and improve development effectiveness by receiving immediate yet realistic feedback. Strong visualization and simulation strategies early on reward researchers in later stages of development.

Matlab is a language with strong capabilities that is used by many fields for its ease of use. Matlab has an optional robotics toolbox; however, this toolbox is difficult to learn for the novice wanting to test a new idea for path-planning and navigation algorithms.

For a fast-paced field seeing an entrance of experts in intelligent systems and other advancing fields, difficult and complicated simulation environments (such as the Matlab robotics toolbox) hurt potential from newcomers. It is for this reason that this paper presents a novel robotics simulation environment for DDR that combines both the realistic simulation conditions needed for validation and simplistic deployment needed for students and newcomers.

II. Specifications

The program contains the following: navigation courses with obstacles and navigation points; navigation course creation tool; differential-drive robot with settings for radius, axle track, maximum wheel velocity, initial global pose, length, and width; object collision detection; sensor readings with definable sensor noise; easy way to develop/test algorithms for robot tracking, robot path planning, and robot navigation. The rest of this section will further discuss a selection of these items.

A. Courses

Courses can easily be created with a modifiable script, where the course size, navigation points (including start and finish), and obstacles (blocks) are specified. The environment comes with pre-made courses, such as below.

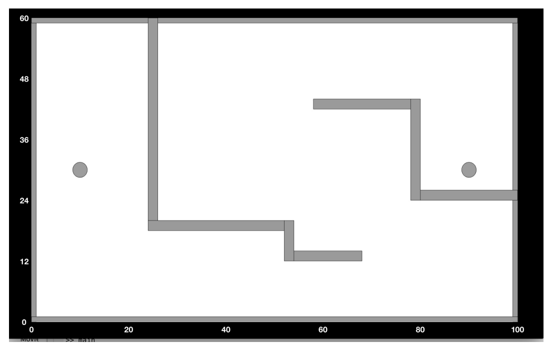

Fig. 1. Sample Course

B. Object-Object Collision

The Separating Axis Theorem (SAT), which practically states that if a line can be drawn between two polygons, they

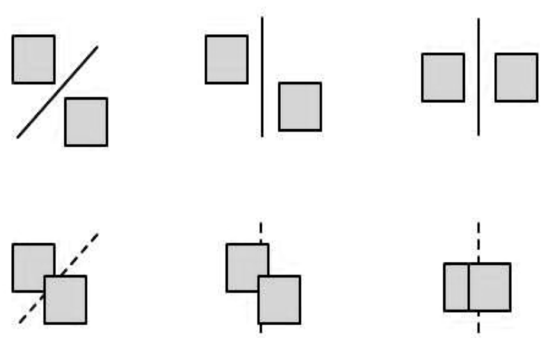

Fig. 2. Visualization of Separating Axis Theorem (SAT) – Image provided at https://gamedevelopment.tutsplus.com/tutorials/collision-detection-using-the-separating-axis-theorem--gamedev-169

James S Smith is with the Electrical and Computer Engineering Department, Auburn University, Auburn, AL 36830 USA. (e-mail jss0036@auburn.edu)

must not be colliding, is used to check each move for robot collisions. Choosing a robust method for object-object collision is important retain high speed.

C. Sensor Readings

The Separating Axis Theorem is also used to find line – rectangle collisions between the sensor paths and objects. As shown below, a user-definable number of sensors can be placed on the robot to be used for tracking, planning, and navigation algorithms. After finding the intersection point between the sensor line and obstacle, the Euclidean Distance is calculated as the sensor reading. An optional amount of noise may be added to improve realism.

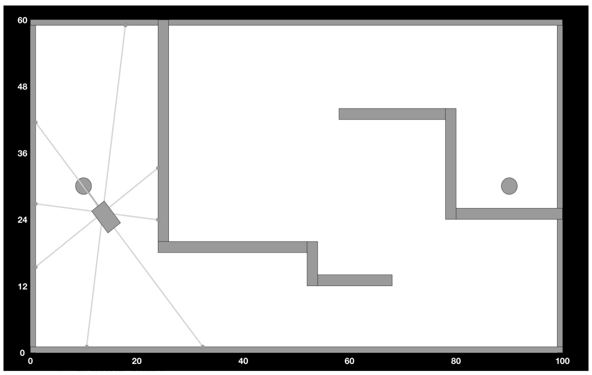

Fig. 3. Sensor Visualization for DDR

D. Simple Function Testing for Tracking, Planning, and Navigating

Finally, the environment is set up with pre-made functions for tracking, planning, and navigation in respective folders contained at the top level of the project. Templates for creating new functions are also included. One needs only worry about these functions and the main script for simple testing.

III. Instructions

From the top-level project folder, open the "main" script. Adjust parameters for the robot and simulation environment. If a new course is to be created, open the "createCourse: script and edit the existing template. If the robot location is to be determined in a way other than truth-based, open the "localize" function and select a function from the Localization folder. Finally, open the "navigation" function and pick the appropriate planning and navigation functions. To create a new function, one need simply to open the appropriate folder and edit the contained "template" file of the respective process.

For class instruction, the project should be saved with desired settings so that only one of the above files need be edited. In an ideal class assignment, students will need only edit a single file and then run the "main" script.

IV. Conclusion

The presented Matlab Simulation Environment for Differential-Drive Robotics speeds the development process and produces low-cost, realistic visualization for path-planning and navigation algorithms. The environment is ideal for students, instructors, and experts from a different field looking to test their ideas in a simple yet deterministic manner. For further inquiries, the environment should be acquired and explored.

References

[1] G.O. Young Siegwart, Rolant, Illah R. Nourbakhsh, and Davide Scaramuzza, *Introduction to Autonomous Mobile Robots* Cambridge MA, MIT Press, 2011.

James S. Smith (M'14) received the B.S. degree in electrical engineering from Auburn University, Auburn, AL, USA in 2017. He is currently pursuing the Ph.D. degree in electrical engineering at Auburn University.

In the summer of 2014 he was an Engineering Intern at Troy 7, Inc. In the summer of 2015 he was a Research Fellow at the Naval Research Laboratories. In the summer of 2016 he was a Engineering Intern at Dynetics, Inc. From 2016 – current, he conducts research at Auburn University in Computational Intelligence.

Mr. Smith is also a member of Etta Kappa Nu, Tau Beta Pi, and Phi Kappa Phi.