Zeno - An Interactive, Open Source, and Social Educational Robotics Platform

Amay Saxena[1], Aaryaman Sen[1], Raghav Jalan[1]

Abstract
Zeno is an interactive robotics platform with the objective of providing, to the otherwise uninitiated, as comprehensive an introduction to the full gamut of robotic motion and kinematics as possible. The platform comes equipped with a vectoring 3WD omni-directional drive base, along with a control system with two levels of hardware abstraction- gyroscopes and accelerometers for the background functioning of the robot, and a set of light sensors, which will be accessible and programmable by the users- all encased in a compact semi-spherical shell. The kit provides to the users a set of mazes with increasing difficulty, each meant to introduce the student to a different aspect of robot control. We also give students the ability to design their own mazes and to open source their designs to an online platform, from where they can also access and compete on mazes designed by other students. Additional sensor capabilities are also provided, and other skills introduced include omni-directional control, line following, object detection and avoidance, and as a possible advanced concept, control algorithms such as PID.

[1] *University of California, Berkeley*

Contents

Introduction

Zeno (named after the philosopher and teacher) aims to provide a comprehensive introduction to the power of robotic control. The overall learning objectives of the final product will be to get students comfortable with writing code that can interact with the physical world, and to show them just how powerful the synthesis of code and mechanics can be. To this end, we shall equip the robotic platform with a 3WD omni directional drive base as opposed to the traditional 2 wheel standard drive base. This will allow the base to drive in any direction and will generally give to far more degrees of freedom than a traditional drive base. Of course, most of the control system will be abstracted away from the students by the software, but we hope to retain the essence of the control system, such that the students can appreciate the elegance of the design they are working with. The robot will be controlled with an open-source Arduino micro-controller, and the product shall come equipped with software packages that include easy to use methods to control the functionality of the robot. These libraries shall abstract away all of the low-level pin set-up and control algorithms used in the background, and allow students to use simple commands to control the high-level functionality of their robot.

We are still experimenting with various media to implement the mazes that the robots shall compete on. The objective is to make maze-design entirely open-source, and to allow the students to upload their maze designs to an online platform, and compete on other student's maze-designs. In order to facilitate these sharing capabilities, we have a few different designs in mind- on the simpler end, supplying an erasable marker that works on most hard surfaces along with a simple to use upload-design interface, and on the complex end, projection onto a surface, with IR based touch sensing capabilities, and direct upload in real-time. We have provided, in this document, the general outline for both these designs.

1. Robotic Platform Design

The platform will be, as established earlier, a 3WD omni-wheel drive base, along with a micro-controller, a motor-controller, a 9V DC power source, a set of gyroscopes and accelerometers, and a custom PCB to serve as the physical interface for simple and modular sensor placement.

1.1 Drive Base

The drive base will sit on 3 38mm diameter omni-wheels (Fig. 1), which will be driven by a set of three 9V DC motors, placed at angles of 120^o. See Fig. 2 for an example of such a drive-base. This will give the robot the ability to move in any direction without needing to turn, effectively giving it infinite degrees of freedom even without turning, as opposed to the traditional drive base's 1 degree of freedom (forwards and backwards).

The robot will have all the capabilities of a traditional robot, with the ability to control it in an identical fashion if one so chooses (drive forward, backwards, and turn), along with the added capabilities of an omni-directional platform. We shall provide software libraries which allow both of these control topologies. This design decision was largely made in order to introduce another layer of kinematics, and highlight the capabilities of good mechanical design.

The entire body will be encased in a 3D printed casing.

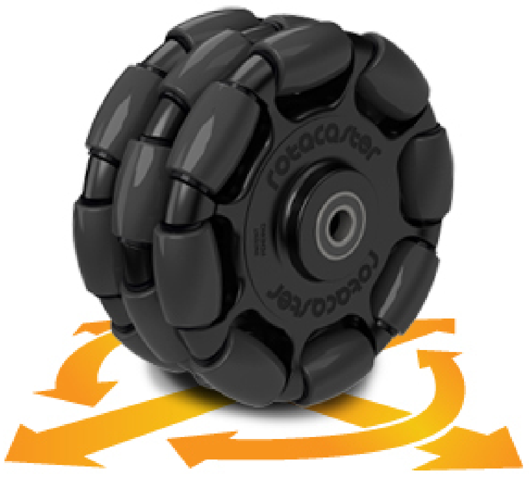

Figure 1. An Omni-Directional Wheel

1.2 On-Board Electronics

The platform shall carry, on-board, an open-source Arduino UNO micro-controller, used to control all functionality of the robot, an AdaFruit motor controller, a battery of sensors including an InvenSense MPU-6050 combined MEMS gyroscope and accelerometer, one TCS34725 RGB color sensor, and at least one custom configuration of a TSL2561 lux sensor.

In order to aid assembly and to avoid unnecessary complexity discouraging or detracting from the learning experience, we shall abstract much of the electronics, by combining the micro-controller and the motor-controller into one element, and by providing a custom PCB with easy to use cable connections for the sensors. In general, the users will not need to interact with low-level electronics.

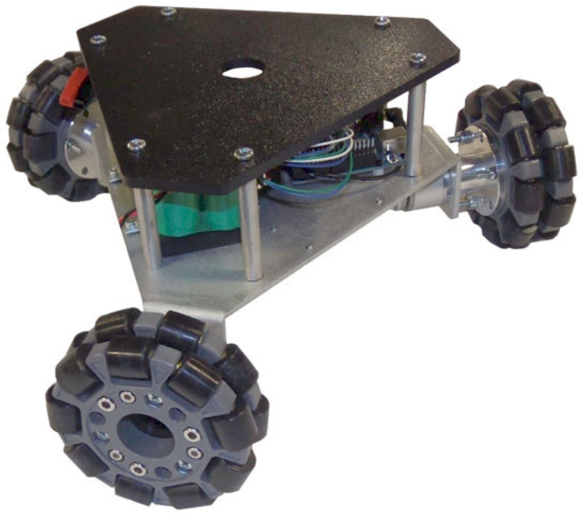

Figure 2. Example of a 3WD omni-directional drive-base

2. Software

In this section we outline all of the software that will be included with the final product.

2.1 User Interface

Zeno uses an Android app to interact with the user and walk them through different technical challenges. The user creates a profile on the network with their name, a picture and a small blurb about themselves. The app has a social feature, which lets the user create their own mazes, scan them with the phone in one step and upload them to an online platform. A global play mode matches them with another user, and they try beating this opponent's score on the maze they uploaded. Both users can see each other's details, thus humanizing the competitive robotics experience and adding a sense of perspective to the game. Upon potential large scale deployment, it has the ability to transcend geographical and socioeconomic barriers to being children together through technology.

2.2 Control API

Zeno's code libraries are designed to be accessible to students with little to no experience with programming. Most of the details of the Arduino code are abstracted away and the user is presented with a set of functions described in natural English (moveForward(time) et al). As the student gains traction in the game, we could also experiment with letting the student slightly further into the weeds and having them add to the code base by teaching them to create their own functions.

In the background, each of our motion methods (moveForward(); etc) will use a PID loop via the gyroscope in order to maintain consistent motion and reduce the possibility of technical irregularities in the motors affecting performance on the user's end

3. Maze-Implementations

We are still experimenting with various ways of implementing the physical maze that robots will compete on. The following are two possibilities that we are considering.

3.1 Markers and Upload Interface

This is a simple, somewhat brute force design. The idea is to provide with the final product a modified dry-erase marker designed to work on any hard surface. We then have internal code that interprets different colors as different maze objects (green interpreted as an obstacle, for instance). This part of the code is not accessible to the students, and is only included in order to simulate the presence of actual obstacle, since we find it inefficient and inelegant to include physical objects solely for the purpose of being obstacles. We would rather have as compact of a final product as possible. We then also include, in the app, an interface to enter their maze design and upload it directly to the platform.

This design has a few clear flaws- first off, the act of having to enter the maze design into a separate interface is a tad inelegant and adds yet another step that the student must perform. Moreover, it may or may not be a challenge to calibrate the sensors to pick up the marker. On the other hand, this design is quite simple to implement, and the markers are a convenient medium since they are familiar and drawing with them is easy.

3.2 Projection and Virtual Touch Screen

This is a much more ambitious approach. This design involves using a small projector at about a foot off the ground on a small tripod to project the maze onto the surface. We will also include, adjacent to the projector, an array of infrared distance sensors, together forming a 2D virtual touch screen. The implementation of the touch screen alone is not too difficult. The student can then draw out the maze wit his finger, and the data collected by the virtual touch screen is translated in real-time to the projector. This design obviously is much more involved, but in the ideal situation, we can have the virtual touch screen upload the maze design directly to the social platform, without any extra effort on the part of the student.

More testing needs to be done before we can finalize one implementation, in particular, we must ascertain exactly how feasible the virtual touch screen approach will be.

4. Maze Progression

Zeno will come with a progression of pre-planned mazes. Each maze in this sequence is meant to introduce the student to a different aspect of programming.

The very first maze in this sequence will simply be a straight line. Here the objective is simply to get the student acquainted with the platform, and comfortable with the idea of controlling a robot with code. This will be followed by a series 2 or 3 of increasingly serpentine mazes, each of which can be solved without the need for any sensors or special control structures (like if-else blocks or loops). Then we progress to the loop mazes. This set of 2 mazes will introduce the user to the idea of a loop. The first of these will be a square, with the objective being to make the robot travel in an infinite loop around the square. The second, will be a circle, which is a much more involved kinematic task and will also introduce the student to the diffTurn(float difference); method.

Finally we move onto the more complex tasks. At this point the students should be comfortable with basic control, and can get to the interactive part, and start competing with their friends and against other users on the online platform. They can start playing around with designing their own mazes, or try using some of our more advanced libraries like the ones for object avoidance or line following to get an edge-up on the competition.

5. Bill of Materials

- 1x Arduino UNO micro-controllers
- 1x AdaFruit motor controller
- 1x InvenSense MPU-6050 combined MEMS gyroscope and accelerometer
- 1x TCS34725 RGB color sensor
- 1x TSL2561 lux sensor
- 3x 38mm Omni-Directional wheels
- 3x 9V DC Motors

References

[1] AdaFruit. *Adafruit Motor/Stepper/Servo Shield for Arduino v2*. https://www.adafruit.com/product/1438

[2] InvenSense. *InvenSense MPU-6050 combined MEMS gyroscope and accelerometer*. http://playground.arduino.cc/Main/MPU-6050

[3] InvenSense. *MPU-9150 Accelerometer + Gyro + Magnetometer (compass)*. http://playground.arduino.cc/Main/MPU-9150

[4] AdaFruit. *TCS34725 RGB color sensor*. https://www.adafruit.com/product/1334

[5] AdaFruit. *TSL2561 lux sensor*. https://learn.adafruit.com/tsl2561/overview

[6] RobotShop *38mm Aluminum Omni Wheel*. http://www.robotshop.com/en/38mm-aluminum-omni-wheel.html