LalaProfiling

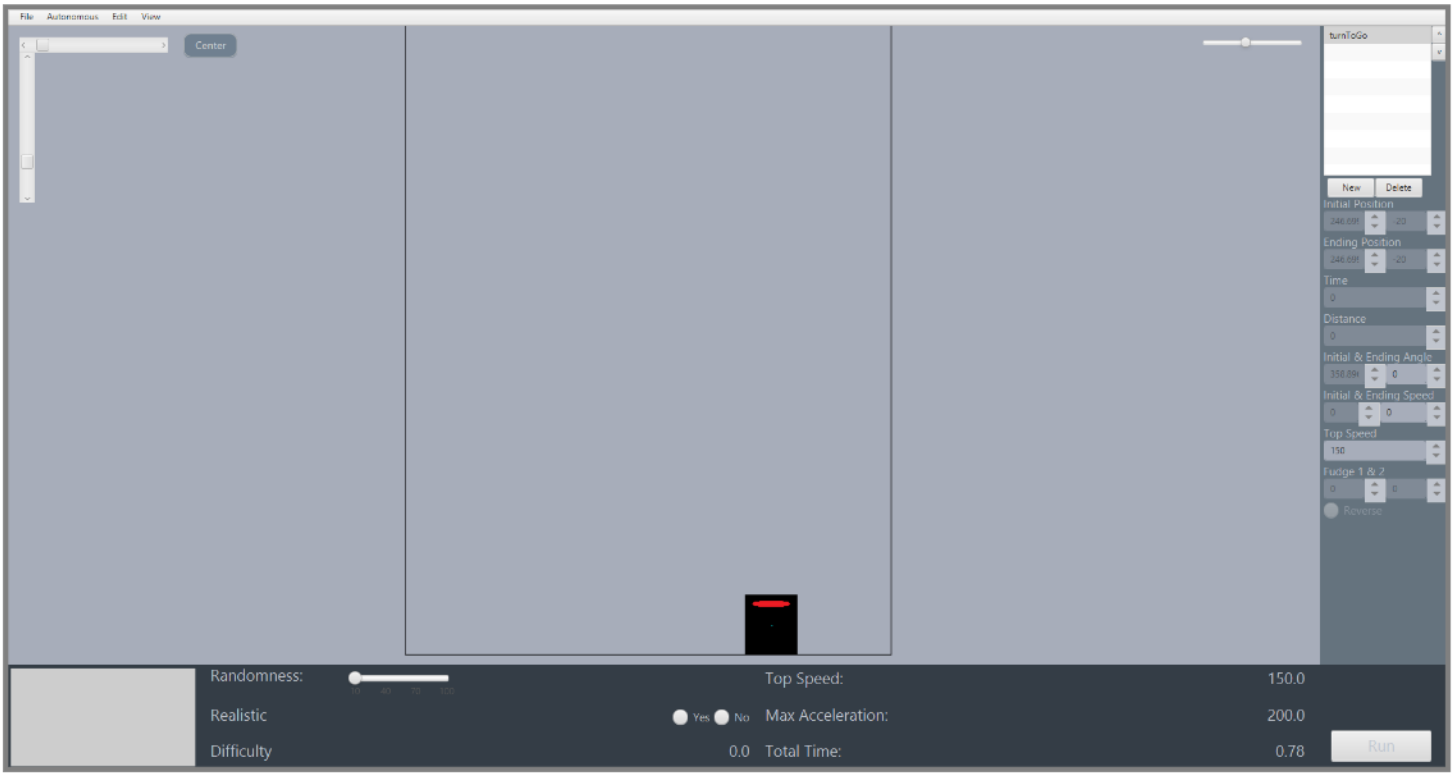
MIT Maker Portfolio

By: Ruben Castro

LalaProfiling

Overview

LalaProfiling is an interactive, graphical, easy to use desktop application written in Java that facilitates the creation of robot trajectories. Within minutes, you can create a trajectory based on a set of movements (lines, curves, rotations, stills) that are modular and easily customizable. LalaProfiling also includes a set of libraries to upload to robot/drivetrain code that will allow the robot to move in the desired pattern after a few tests to determine necessary constants.



LalaProfiling

Movements

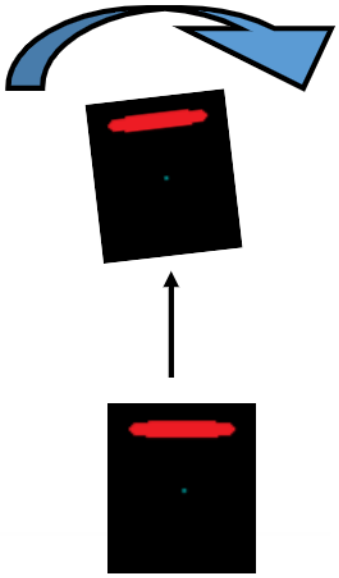
There is currently five types of movements embedded in LalaProfiling: rotations, lines, stills, cubic Bezier curves, and quartic Bezier curves. Each movement is meant to give you the best agility possible when conducting an autonomous movement routine, and minimizing time necessary to move from point A to point B.

Still Movement

Still movements require the amount of time you wish to remain still.

Rotation Movement:

Calculated using initial constraints, ending angle, and top speed.

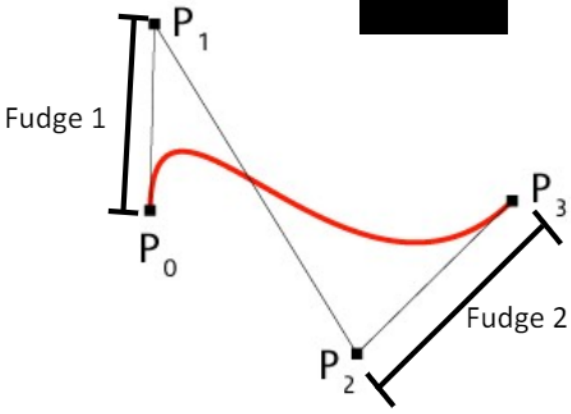


Line Movement

Taking in a distance value, the program calculates the ending position based on initial position and initial angle.

Cubic & Quartic Bezier Curve

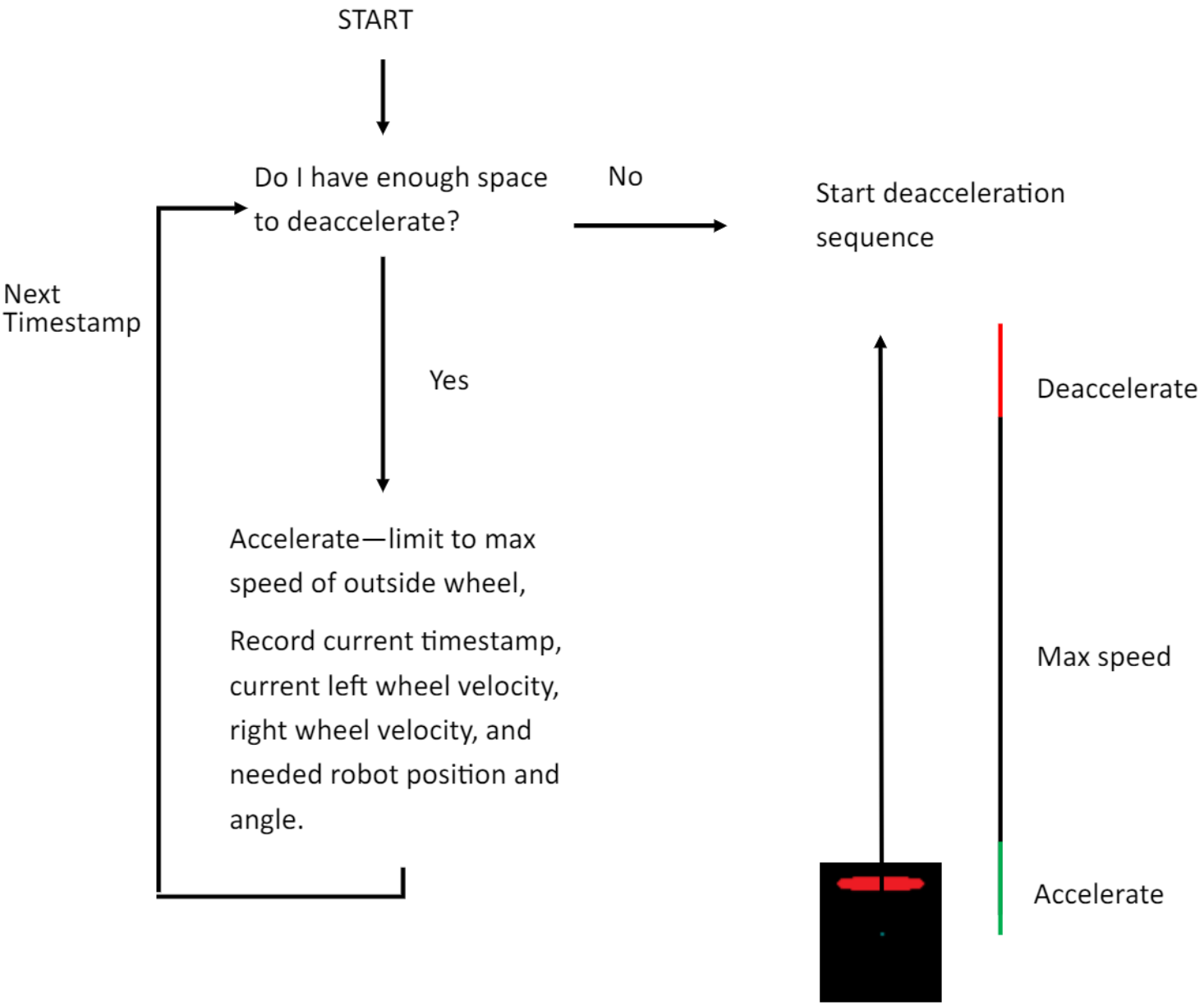
These Bezier curves require ending position, ending angle, and two "fudge" values. Quartic Bezier curves require an extra control point to manipulate curve for further detail. To calculate exact angle at each moment in the curve, derivatives of the curve polynomials are taken.



LalaProfiling

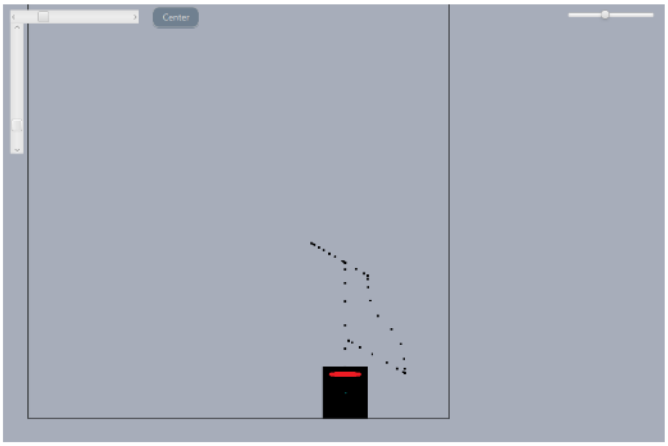
Trajectory-Builder

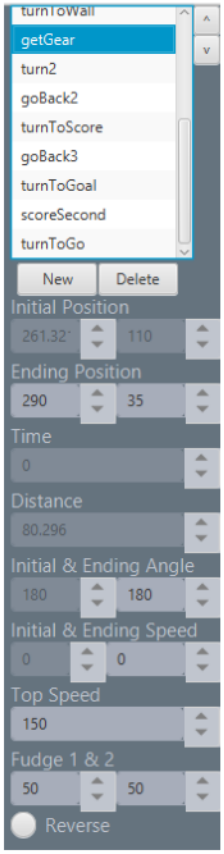
The algorithm that builds the actual trajectory with timestamps, left & right wheel velocities, and expected angle & position can be decomposed as follows.



LalaProfiling

GUI





Stage

The stage brings the robot and the trajectories to life. With real-time playback, control over stage size, and background image, you can see how your robot will perform.

Movement Editor

The movement editor provides a graphical interface which the user can manipulate to easily choose and change a movement's parameters, from ending position, to top speed.

Autonomous Controller

The autonomous controller, depicts useful data such as difficulty of the trajectory, total time estimate, max acceleration, top speed, and other useful information.

LalaProfiling

Robot library

LalaProfiling contains a set of classes to implement in your robot/drive train code. While these are mostly concerned with motion planning, there is other classes included from other projects including: object recognition, a double interpolator, PID control loops, sensor wrappers, and interfaces. Each class is meant to be modular, able to be adapted to any motor controller and sensor type by using standard java interfaces found in java.util.Function.

The most important classes are:

RDriveTrain.java

Drivetrain class that implements open-loop and closed-loop drivetrain control. Capable of keeping robot moving at desired speeds, limiting acceleration, and maintaining robot at a certain angle by using several control loops.

TrajectoryInterpreter.java

Class that takes in a set of trajectories, and starts a loop that moves robot in the desired trajectory.

TrajectoryIO.java

Simple IO class that facilitates the interaction between application and robot. It reads what LalaProfiling app exports, and converts it into the required trajectories when the robot boots up.