CSCI–4190 Introduction to Robotic Algorithms, Spring 2006
Lab 2: out Tuesday February 21, due Thursday March 9

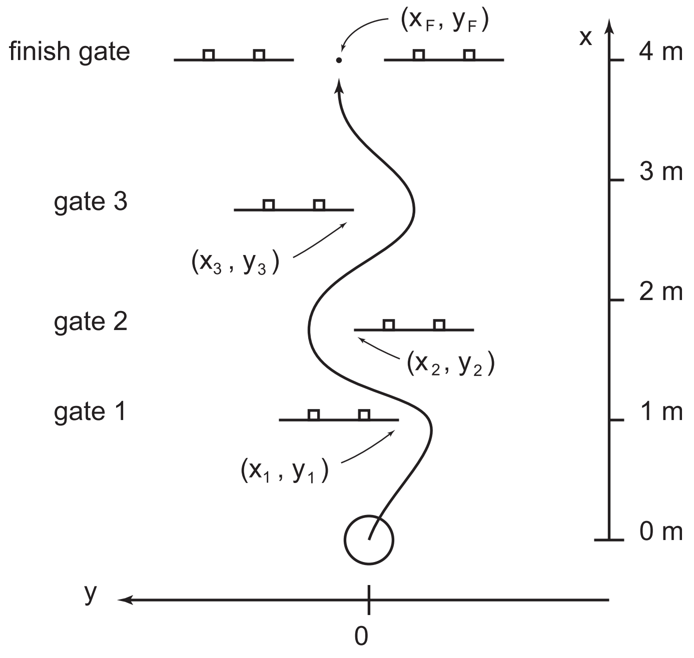
Robot Slalom

For this lab, you will program the robot to run a slalom course. Your grade on this lab will be determined in part by the performance of your program.

There will be a maximum linear velocity for the robot, so the key to a good time on a course is to steer the robot accurately. You can use the SONAR and IR sensor data as well as odometry to guide the robot. If the robot comes in contact with one of the gates, there will be a penalty added to the time for that run, so you will want to turn around the gates closely but not too closely.

Course description

The robot will always start at the same position in the lab, but the position of gates may differ (within certain constraints) for different courses. Your program must run without any information on the specific course; instead in must use its sensor data to negotiate the course. Here is an example course which I will use to describe the parameters for gate placement:



Robot starting configuration

Assume that the world coordinate frame has its origin at the robot's start point and the x axis points in the forward direction of the robot.

Gates

The gates will be short cardboard walls, approximately 1 meter wide, which can be sensed with both SONAR and IR sensors.

- All gates will be approximately parallel to the world y axis.

- There are two small blocks on the back of each wall which are considered part of the gate, so the robot should not come into contact with them. These blocks are at least 25 cm (10 in) from each end.
- There may be as few as 2 gates: Gate 1 and the Finish Gate, or as many as 4 gates: Gates 1–3 and the Finish Gate. Each gate will consist of a single cardboard wall except for the Finish Gate which will consist of two colinear walls with a gap between them
- The first gate may require either a left or a right turn around the gate. Subsequent gates will require alternating left and right turns until the finish gate is reached.

Gate location & spacing

For purposes of describing constraints on the gates, I will describe the position of each gate by its *turning point*, the endpoint of the gate about which the robot must turn. For example, if the robot must make a left turn around Gate 1 (as in the figure), then the Gate 1 position will be described by the coordinates (x_1, y_1) of the right endpoint of the wall. The Finish Gate, however, is described by the position of the point in the middle of the gate.

- Gate 1 will always be at $x_1 = 1$
- If Gate 1 requires a left turn around the gate, then the right side of Gate 1 will cross the $y = 0$ line by no more than 30 cm, i.e., $0 \geq y_1 \geq -0.3$.
- If Gate 1 requires a right turn, then the left side of Gate 1 will cross the $y = 0$ line by no more than 30 cm, i.e., $0.3 \geq y_1 \geq 0$.
- The spacing between gates in the x direction will be at least 0.75 m, i.e. $x_{i+1} - x_i \geq 0.75$. This also applies to the last numbered gate and the Finish Gate. (Note that the robot is approximately 0.4 m in diameter.)
- The turning points of two consective gates may differ by at most 0.5 m in the y direction, i.e., $|y_{i+1} - y_i| \leq 0.5$. This also applies to the turning point of the last numbered gate and the center point of the Finish Gate. Note that this means that two gates need not overlap in the y direction.
- The course length will be between 3 and 4 m, i.e. $3 \leq x_F \leq 4$.
- The gap in the Finish Gate will be 0.75 m wide.

Please note that all measurements are approximate, i.e., as best as we can align the gates, so expect variations of a few cm.

Rules

- The robot may be set to have a maximum linear velocity of 0.4 m/s. (However, you may not want it to travel this fast all the time.)
- The time to run the course will be from the moment a "go" command is given until the robot passes completely through the Finish Gate.
- There will be a 10 s penalty added to the robot's time on a course for each gate that the robot (accidentally) contacts during that run. (Your program should not intentionally cause the robot to contact a gate; we may apply additional penalty time or disqualify runs to discourage any such strategy.)
- You may not give your program any information about the specific course, e.g. the direction of the first turn, the number of gates, etc. This means that your program must sense the gate positions and navigate accordingly.

- The robot should stop after it has passed through the Finish Gate, though currently there is no bonus or penalty for doing or not doing so. (This could change.)
- Your program should not be affected by spectators standing (or moving about) on the sides of the course or sufficiently far behind the Finish Gate.

Suggested approaches

For the non-performance part of this lab, a "principled" approach to this problem will be viewed more favorably than an "ad-hoc" approach. You are free to use any approach you wish, but here are some ideas and suggestions:

- You will find the IR sensors most useful for this lab, but you may want to make some use of the SONAR sensors.
- We have covered a number of techniques for obstacle avoidance and local navigation. You may wish to implement one of these or a variation thereof.
- We have also covered a number of control techniques, in particual the pure pursuit approach to following a path. (We will soon be covering wall/hall following which will be similar to pure pursuit.) You may wish to implement this sort of approach.
- You have the background to compute an estimate of the turning point of a gate based on sensor readings. This could be used to compute the location of a subgoal for each gate.
- You may find it helpful to use some sort of estimation techniques to process your sensor data. You probably don't need to use the Kalman filter (or the EKF) — simple merging and compounding should suffice.
- You may need to calibrate the sensors or measure an error model for them.
- The robot will keep track of its odometry data, but the orientation of its world frame will drift as the robot gets further into the course. One approach to solving this problem would be to correct this drift by using the fact that the gates are parallel to the world y axis. Another approach would be to do all your computations in a local frame that need not be aligned with the axes of the robot's world frame.

To be turned in

- Your code.
- A written report describing the approach you took to solving this problem, including what techniques you used and any relevant models or calculations for estimation or for controlling the robot.

  Further details on the content of the written report will be available on the Lab 2 information page.