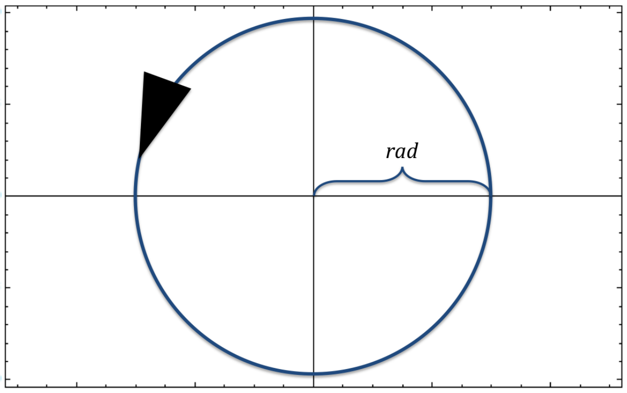
Lab 3 Robots on Racetracks
15-424/15-624 Foundations of Cyber-Physical Systems

Test Due Date: Friday, 10/14/13, worth 20 points
Final Due Date: Wednesday, 10/18/13, worth 80 points
Course TA: Sarah Loos (sloos+fcps@cs.cmu.edu)



In this lab, you will design three controllers to drive your robots around a circular racetrack centered at the origin and with radius *rad*. In each question, you will design a controller that satisfies the safety specifications given. The properties you prove are safety properties (i.e. stay on the track and don't hit the robot in front of you), but you should also keep in mind the efficiency of your controller (i.e. follow the lead robot closely). So, for example, a robot that just stays still will satisfy all the safety properties, but wouldn't get any points.

1. **Lap time (empty racetrack).** In this question, your robot should be able to drive *counter-clockwise* on the track without leaving it. In other words, the robot should always be on the circle with radius *rad* centered at the origin. The racetrack is completely empty, so the robot may choose to travel at any speed. Your HP should include a loop so that the controller may execute more than once.

   (a) [test] Design a hybrid program to keep your robot on the track, then create a formula which, if proved, would guarantee that your controller never causes the robot to drift off the circle. You will need to determine appropriate initial conditions to make this property provable; however, try to prove the strongest property you can by making the initial conditions as general as possible. Save this formula in `test1_username.key`. Make sure that your file will load (parse) in KeYmaera before you submit.

   (b) [final] Use KeYmaera to prove that your controller keeps the robot on the track. Submit the .key file you proved and the proof file as `final1_username.key` and `final1_username.proof`.

   (c) [final] **Question:** Solve the differential equation you used to model the physical dynamics of the system. Could you rewrite an equivalent hybrid program using this solution and the assignment operator instead of a continuous evolution? Why or why not? Add your solution to `lab3_username.txt`.

2. **Stationary robot (static obstacle).** Now modify the first question to include another robot that ran out of battery and is just stopped at a predefined position on the track. You may decide how to represent the position of the stationary robot, but it should be symbolic. Of course, you can't start your robot at 100 mph just feet behind the stationary robot and expect the system to still be safe, so you may define some initial conditions on the distance between your robot and the relative position of the stationary robot. If these initial conditions are satisfied, your controller should be able to bring your robot to a stop before the stationary robot. The bounds on acceleration and braking for your controlled robot are $A > 0$ and $-B < 0$. Your controller should be time-triggered.

   (a) [test] Design a hybrid program to keep your robot on the track and stop before the position of the stationary robot. Create a formula which, if proved, would guarantee that your controller satisfies these safety properties. Save this formula in `test2_username.key`. Make sure that your file will load (parse) in KeYmaera before you submit.

   (b) [final] Use KeYmaera to prove the safety properties. Submit the .key file you proved and the proof file as `final1_username.key` and `final1_username.proof`.

3. **Pace car (dynamic obstacle).**

   Instead of a stationary robot, we now include a robot that can drive on the track by accelerating at rate $A > 0$, braking at rate $-B < 0$ or keeping a constant velocity. Your controller should follow the leader as closely as possible, but you will only have to prove safety (i.e. that your robot does not collide with this new robot). You may again define some initial conditions on the distance between your robot and the relative position of the stationary robot. If these initial conditions are satisfied, your controller should be able to bring your robot to a stop before the stationary robot. The bounds on acceleration and braking of your robot are $A > 0$ and $-B < 0$. Your controller should be time-triggered. You should model the other robot such that it also only travels counter-clockwise around the track. Since you are verifying the safety of the controlled robot, it is not considered unsafe if the other robot causes a collision (i.e. if the other robot is behind you on the track and crashes into you).

   (a) [test] Design a hybrid program to keep your robot on the track and avoid collisions with the other robot. Create a formula which, if proved, would guarantee that your controller satisfies these safety properties. Save this formula in `test2_username.key`. Make sure that your file will load (parse) in KeYmaera before you submit.

   (b) [**BONUS**] Use KeYmaera to prove the safety properties. Submit the .key file you proved and the proof file as `final1_username.key` and `final1_username.proof`.

4. **Submission checklist.**

   Test submission (Due 10/14):
   `test1_username.key`
   `test2_username.key`
   `test3_username.key`

   Final submission (Due 10/18):
   `final1_username.key`
   `final1_username.proof`
   `final2_username.key`
   `final2_username.proof`
   `lab3_username.txt`
   BONUS (submitted with final submission):
   `final3_username.key`
   `final3_username.proof`