Lab 5: Robots vs. Rogue-bots (Dynamic Obstacle Avoidance)
15-424/15-624 Foundations of Cyber-Physical Systems

Test Due Date: Friday, 11/8/13, worth 20 points
Final Due Date: Friday, 11/15/13, worth 80 points
Course TA: Sarah Loos (sloos+fcps@cs.cmu.edu)

1. In this lab you will design a controller which may move freely (non-deterministically) on a plane, rather than just around a fixed circular track. The robot should be able to move anywhere, but it must safely interact (i.e. the robot should not cause a collision) with a rogue-bot, which moves at constant velocity.

   This lab is an extension of lab4, except that the obstacle is now a rogue-bot which drives around with constant velocity rather than a stationary obstacle.

   To help you get started with this assignment, you may download a template for the key file here: `http://symbolaris.com/course/fcps13/template5.key`. This file contains the annotations which are required for simulation. We've included some variables which are not strictly necessary for modeling or proving, but which you may uncomment. If you want to add more variables about the state of the system, please post your request on Piazza so that we can create a standard naming scheme.

   - You **may not** have discrete changes in the position, direction, or linear velocity of the robot (or any other variable which would implicitly cause such a discrete change).
   - You may (and should) discretely control the track radius (which can be negative to change the direction of travel from clockwise to counter-clockwise) and acceleration of the robot.
   - The robot should have a non-deterministic controller (i.e. by suitable resolution of the nondeterminism, it should be able to drive anywhere that does not cause it to come too close to the obstacle).
   - The robot should be time-triggered, with the control executing within time $T > 0$.
   - Since you are freely moving in 2D, your robot has a shape, which can be over-approximated by a circle of radius $r > 0$.
   - The rogue-bot's shape can be over-approximated as a circle of radius $rogr > 0$ (hint: you may still model your system using points, so long as a sufficient buffer is kept between the robot's point and the rogue-bot's point).
   - The robot should never turn with a turning radius less than $minr > 0$ (i.e. it can't spin in place or turn too sharply).
   - The acceleration and braking of the robot are bounded by $A > 0$ and $B > 0$ respectively.
   - The rogue-bot maintains constant velocity $rogv \geq 0$.
   - The rogue-bot's steering should be completely non-deterministic; this means that the rogue-bot can drive freely around the 2D plane, regardless of where it is in relation to your robot.

   Suggestions and hints:

   - You may simplify your model to represent an infinitesimal point (x, y) for the position of the robot and point $(rogx, rogy)$ as the point of the rogue-bot **provided** you ensure that the two never get within a reasonable symbolic *buffer* distance of each other.
   - Guarded non-deterministic assignments will be very useful for modeling a range of steering choices.
   - If you found using the Max function to be difficult during proving, you may find it easier to reformulate those statements using disjunction instead.
   - Use the provided template (available at `http://symbolaris.com/course/fcps13/template5.key`).

1.1 [test] Fill in the missing parts of the provided template to model the hybrid program above and verify that it is safe. Remember that safety with moving obstacles will not be the same as safety with static obstacles. Submit this file as `test_username.key`.

1.2 [final] Use KeYmaera to prove that your hybrid program is safe. Submit the resulting proof and corresponding .key file as `final_username.proof` and `final_username.key`.

1.3 [final] **Question:** Discuss the advantages and disadvantages of the definition of safety that you proved. Suppose now that you were to design the controller of the rogue-bot. How would you design a rogue-bot to make your robot perform poorly? Is this related to your definition of the safety requirement?

2. **Submission checklist.**

Test submission (Due 11/8):

- `test_username.key`

Final submission (Due 11/15):

- `final_username.key`
- `final_username.proof`
- `lab5_username.txt`