16x62: Lab 8
Cooperation City

Due: ***Tuesday Week 12***

Introduction:

You have all begun to play the Game, congratulations! You know how to solve the problem of the competitive 1-on-1 Game. At least nonoptimally. Not that you've actually competed against someone else's robot, but that's besides the point. Your task now is to conduct research and development into the problem of solving the problem of cooperative Game playing.

First of all, here are the assumptions & capabilities you need to know about for the Cooperative Game:

- Each robot receives, at runtime, a map of the entire world and its position and the position of *all* gold and *all of your team's* cargo bays. Each robot only receives the position of itself and NOT the position of its teammate. So this might be a useful thing for your robots to begin by communicating with one-another. Each robot also knows how many robots are on its team.
- ALL of your cargo bays will be accessible to both of your robots.
- There may be maze cells that your team cannot access.
- Therefore, there may be gold that your team cannot access!
- Your teammates can communicate using the wavelan. No further rules to communication content exist. You may choose to use Netcomm. Or you may choose to use your own comm. system.

To solve this problem, you will have to start simple. Some people get PhD's for figuring out how robots should negotiate and communicate. Design your cooperative strategy fully, including how you plan to divvy up gold, how you plan to solve bottleneck problems, and what information at runtime your robots will communicate. Try to implement a simple protocol for Tuesday that works. You will get a good grade on this assignment so long as you have *some* cooperation occurring on Tuesday. We suggest that you concentrate on the somewhat smaller two-player mazes in Games for this week's assignment.

To solve this problem, you WILL want to make a high-level simulator, so that you can have (a) two robots on one screen on one computer to test your cooperation strategy, and (b) two notebooks communicating over Netcomm for example but without robots, just playing on make-believe mazes on the screen. These simulators are easy to create, and only with these in hand will you be able to debug and tweak high-level code.

In less than two weeks from Tuesday, on Monday and Tuesday, you will be playing REAL full-blown games. These rounds will determine your seeding values and, therefore, how hard of a team you play in the qualifying rounds just after thanksgiving. Budget your time carefully from here out! After this assignment, you will still have much cooperation work to do, and furthermore it will be time to ensure that you can write code to pick up the golds literally using the magnets.

Assignment 8.0:

Write your cooperative robot controller. You will read in a GameWorld file (described below).

Good example mazes in Games include g4sm and g5sm.
We will test you using games like these!
Large-scale games, like the ones you should simulate on and you may see in the real rounds, include g6 and g7. Of course, we do not have room to use these in REL.

WARNING: from here forth, be sure to also plan (at least a little) for the contingency that, if our wavelan fails or if some robots break, we will be doing a 1-on-1 contest rather than a 2-on-2 contest. This is highly unlikely but make sure you can reasonably easily compete in such a game.

Startup Protocol:

When you are running a game, your interface should have more than one button. Specifically, there should be one or more buttons that:

1) test the camera connection, ensuring you can talk to it OK
2) test the robot connection, ensuring you are communicating with it OK
3) turn on the sonars, once and for all
4) connect to your teammate notebook, ensuring that network comm is working OK
5) Load up the game file, displaying the correct gameworld on-screen
6) A test button that twitches your robot, providing a quick check that indeed the notebook-robot connection is still good, and the estop is off.

Because once people are set up we’re often waiting 3 minutes for the other team to finish setting up, this step (6) is incredibly important in verifying, from time to time, that things are Good.

Of course there should then be another “Play” button that, when hit, causes the robot to start playing the game (assuming all the above has been done OK).