CSE-571 Project 1 (warm up part)

1. **Patrol robot**

   The purpose of this exercise is to get more familiar with Saphira behavioral actions and the LPS. The expected result is to get the robot to patrol continuously between two goal positions. The goals will be two points specified in the robot's internal coordinate system. The robot should move between these points, even when the points are perturbed from their original positions. This exercise requires 4 components: A behavioral action schema that, when invoked, will guide the robot to a goal position (the goal position should be an argument to the schema). A Colbert activity to sequence the behavioral actions into the patrol activity. A function, written as part of a loadable file and callable from Colbert, that can change the position of the goal points. Graphical display of the goal positions in the LPS.

   Some tips: You can't rely on the turn commands to turn the robot exactly, or on the position commands to move the robot exactly. Also, a goal position may move while the robot is going towards it. Therefore, the behavioral action needs to continuously calculate the turn angle and forward speed of the robot. The tutorial on actions has an example of both a behavioral action and a Colbert activity. The directory tutor/movit has the relevant sample files, as well as a makefile for compilation.

2. **Bump and Go**

   The idea is to make the robot go forward until it contacts an object (based on wheel stall), and then get it to back out and continue in another direction. The key is to design a movement that will keep the robot from getting stuck no matter what the environment. Debugging on the simulator is fine, but lots of experiments on the real robots will help to design a foolproof movement. It would be nice to use the subsumption architecture for this purpose, and implement it using Saphira activities, but this would be much too complicated for a two-week exercise. So just use a Saphira activity or activities as you see fit, keeping track of state (no contact, initial contact, backing up, etc.) to implement the movement. Be aware that the motors time out for about 2 seconds after stalling, so that no motion will occur during this period. However, if you issue a motion command during this time, such as move, it will be executed as soon as the motors come on line again.

   Some hints: try all of the above for robustness. See if you can put together several different strategies, especially for getting unstuck, and vary them if the robot does get stuck. You may also have to recognize when the robot is stuck, which may not be easy. Don't assume the robot can back up safely, or that the motor stall indicators will always correctly signal which side the problem is on. We'll test this one by trying "escape from the chair." There is a sample bump-and-go Colbert activity in colbert/bump.act. This file should serve as an example for how to write a simple bump-and-go activity, but it has some obvious performance problems.

3. **Friendly robot**

   Write an activity that will turn the robot, looking for nearby objects, and then approach the found object. After a short while, the robot should get bored and go

looking for another object. This exercise needs the sonar sensors. Since we won't cover these sensors until the next project set, you can use some simple supplied functions to test if there are any sonar readings within a given polar or rectangular area. This exercise requires 4 components:

(a) The behavioral action schema developed in Part 1.

(b) A Colbert activity to sequence turning and go-to behaviors, while monitoring for objects.

(c) Functions to determine the presence of objects in different areas around the robot.

(d) Graphical display of a found object position in the LPS.