CMPUT 609 Robot Project
Fall 2013 – Sutton

Joshua Charles "1367084" Campbell
Ping Jin

December 6, 2013

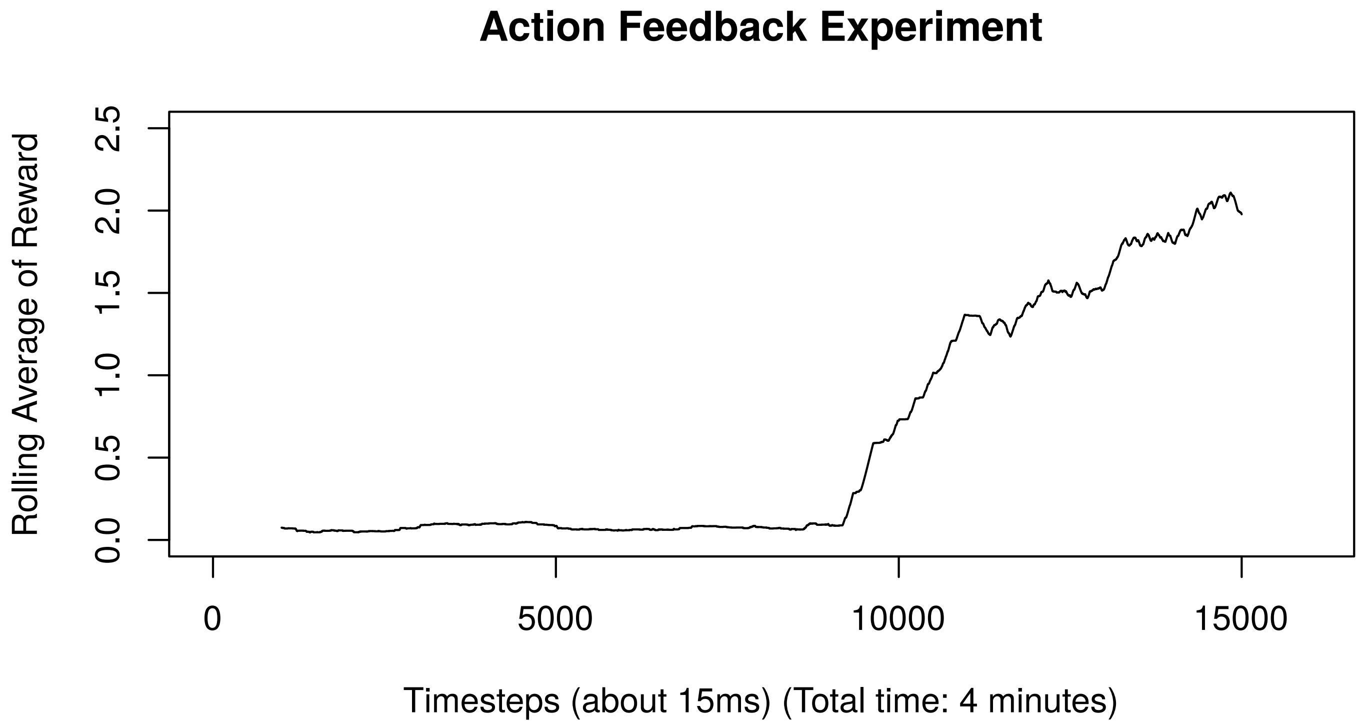
Experiment 1: We performed an experiment to determine the feasibility of using whiteboard markers. We used the provided cliff calibration software to see if the sensors could adequately (such that we could set a constant threshold for each sensor) distinguish between the whiteboard and the whiteboard with black marker on it. It could not: furthermore, it was impossible to position the sensors over the marker marks.

Experiment 2: We performed an experiment to determine if the robot could learn to follow the walls of the micro-world. This experiment involved 16 states, of which only 8 were useful, and 9 actions. The states were determined by the state of the two bump sensors (front left and front right) of the iRobot Create, one cliff sensor (thresholded to 420/4095) and the wall sensor (thresholded to 2/4095). The algorithm used was discounting-reward epsilon-greedy SARSA with eligibility tracing. The parameters were set as follows: $\alpha = 0.1$,$\gamma = 0.98$, $\varepsilon = 0.01$, $\lambda = 0.9$. The SARSA timestep was at least one tenth of second. We saw after multiple runs that the robot was reliably able to learn to use the wall sensor to perform the following motion, shown in this video: http://youtu.be/rxXYQPYi-rU. This wasn't quite the wall-following action we had hoped that it would learn, however, it does go forward and turn when it gets near a wall. Code for this demo is also available: https://github.com/orezpraw/createProject/blob/WorkingVersion1/sarsa.c. The robot in this scenario is rewarded going forward, minus the amount it went backwards. It is also punished severely for touching the walls (triggering the bump sensor).

Experiment 2b: We performed further experimentation following our success to see if adding the previous requested action (after applying any reflexes) to the state space, decreasing the SARSA timestep to 15ms, and replacing the useless cliff sensor data with a "memory bit" would improve performance. The state space increased to $16 \times 9 = 72$ in order to record the previous action and the action space increased to 10 actions (the 10th not being recordable). The tenth action merely flipped a bit of the state space. The action stored as the "previous action" in the state was decoded from the last request sent to the physical robot: this means that it was the action the robot was actually taking (driving forward, etc.) after applying any reflexes or restrictions. This version of the code is available at: https://github.com/orezpraw/createProject/blob/CheatingDemo/sarsa.c

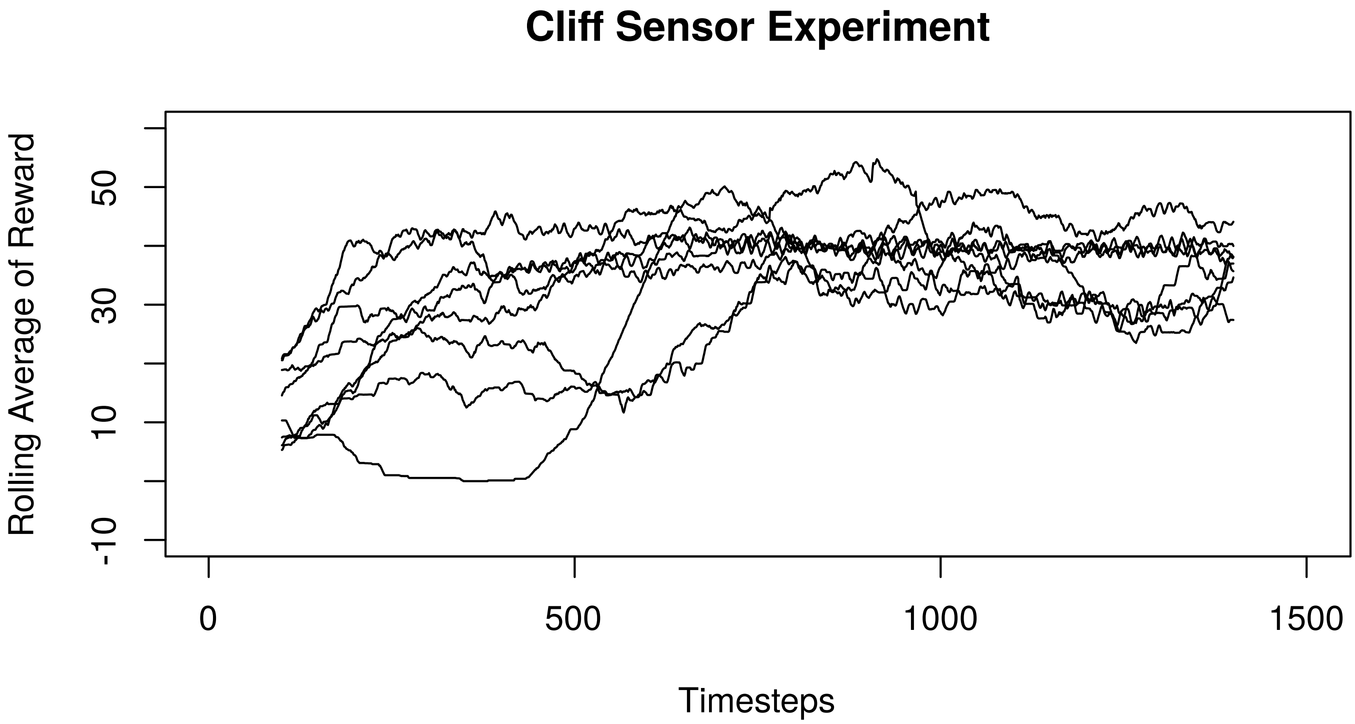
The robots behavior following these changes was very interesting. Most times we tried it the robot would learn how to "cheat" by sending alternating actions to one wheel. This cause that wheel to not trigger the encoders as often as sending a continuous forward signal.

In this video https://www.youtube.com/watch?v=DKycFaNPWrU it is circling by continuously driving one wheel forward while alternating the other wheel between stopped and backwards, which yields a lot of forward reward. Eventually it combines the wall sensor with this behavior to really maximize its reward. A moving average of the reward per timestep (15ms) during the above video is shown:



During one experiment it learned to “hop” backwards: the robot could pulse both wheels backwards and then stop to slide backwards without being punished as much as continually driving backwards. Then it would drive forward continuously to get the most reward. Unfortunately, we were unable to replicate this behavior later.

Experiment 3: We performed an experiment to make the cliff sensors more usable by taping a black mouse-pad on the whiteboard surface on one side. We modified the rewards to be multiplied by the number (0-4) of cliff sensors that detected “on the mouse-pad” as determined by the cliff sensor threshold calibration software provided. In most cases the robot determined how to move forward around the edge of the pad as demonstrated by this YouTube video: https://www.youtube.com/watch?v=7-ktqKlfL2M. The following graph shows the reward over time (smoothed by a moving average filter) for 8 different runs of this program starting from random values. In this experiment, the timestep was also about 100ms, so this represents data from the first 2 minutes and 30 seconds of each trial.



Cliff Sensor Experiment
Rolling Average of Reward
-10
10
30
50
0
500
1000
1500
Timesteps