Jacob Clouse - ICSI 551 - Final Project Report

Abstract

Applying Bayesian inference and nested sampling within automation allows us to demonstrate the effectiveness of Bayes Algorithm in solving real world problems. In the past, Professor Knuth worked with colleagues on a project using Lego Robotics to detect a circle using matlab code and nested sampling. For my Final Project, I have decided to pick up the original code that Professor Knuth wrote about 15 years ago and adapt it so it can be used with the newer model of the EV3.

Introduction and Setup

When I initially started on this project, I needed to do some introductory research into what the Lego Mindstorm EV3 product was and how to communicate with it. The product was several years old and it was listed as a retired product on Lego's product page (which meant that there was no current support for the platform). I needed to find a method of connecting to the EV3 control brick and sending commands to it.

There had been support for controlling the Lego EV3 bricks from matlab as Dr. Knuth had done this with his previous paper, these scripts were several years old and I needed to confirm that the functionality still existed. Before I started on programming, I needed to make sure that the EV3 was able to connect with and be detected by my computer. I originally attempted to set up a bluetooth connection from my computer to the EV3, but the link never established and the Lego Mindstorm software never was able to connect with the EV3 module. I was able to take an old mini-usb cable to connect the EV3 with a wire to my computer and this was able to create a usable connection within the EV3 software. I then wanted to make sure that I was able to send commands to the EV3 with the basic controls of the Lego Mindstorm software (with its rudimentary programming blocks). I created a sample to make a few beeps to show it was connecting and executing, and this worked as intended. I knew that there were plugins that worked with Lego hardware inside of matlab, but I would need to receive all the hardware before I could move forward using them.

Now I wanted to move towards creating more complex programs with my EV3 and needed to expand beyond the simple Lego Mindstorm software. I then talked with Dr. Knuth and picked up the original, larger Lego robot that I would be working with (along with the attached peripherals like motors, cameras, pressure sensors, etc.). This Lego robot had an older control unit attached to it called the NXT. The NXT looked very similar to the EV3 unit, it had all the connection ports and control surfaces as the EV3, but it appeared to be a few years older. As it was the original unit attached to the full robot, I had initially decided to try and use this as my control surface. The problem was that it was not recognized by the EV3 software that I had previously used, even when connected via a cable. It did not show up inside of my computer devices and I was also unable to connect over bluetooth. I had looked online to see if I could find the original control software Lego created to issue commands to it, but I was unable to make the software work. I looked online and found potential methods of controlling the NXT unit with Python and C, but the documentation for both was vague and had few examples. After discussing with Dr. Knuth, we had decided to simply swap the connections from the NXT module to the EV3 module in the larger robot and then use the EV3 as the main control unit with the NXT peripherals.

Method

I then turned my attention to getting the matlab IDE working directly with the EV3 unit via the use of plugins. I installed the **MATLAB Support Package for LEGO MINDSTORMS EV3 Hardware** inside of matlab and all the subsequent hardware dependencies within the package manager. I wrote a few test scripts using the plugin's documentation to articulate the attached motors, print data to the ev3's LCD and make beeps from the speaker on the device. I was able to successfully execute these tests using the matlab code and this confirmed that the device was functional and I could progress with coding my solution. However, I soon ran into an issue with the color sensor on the robot and how it was identified in Matlab. I attempted to create a sample program to query the color sample from the color sensor using the plugin documentation, but it kept giving me errors. I attempted using the original color sensor and the new one included with the new EV3 brick, but both gave the same unidentified peripheral error. I printed out the connected peripherals to see what sensors were attached, and I was expecting to see “motor”, “color”, “motor” and “motor”. However, the print outs were nothing like this, instead it printed out {0×0 char}, {'NXT_LIGHT'}, {0×0 char}, and {0×0 char}. The motors were working correctly, but it appears that the color sensor was being identified as the “NXT_LIGHT” and this was causing an issue that prevented me from obtaining data from the sensor. Before proceeding, I talked with Professor Knuth about the issue that I was experiencing and asked him how to proceed. We both agreed that the best course of action would be to continue adjusting the code as if the sensor was working until we could figure out how to get it to work. With this knowledge in hand, I was able to move forward with my work to adapt the older NXT nested sampling code to work with the new EV3 module. The first thing I did was to adjust the inputs from using the older, NXT sample to using the EV3 syntax from the matlab plugin. I retained as much of the original sampling code as I could from Professor Knuth's, I wanted it to be integrated as tightly as it could be with the existing infrastructure. I mainly worked on adapting the aquireDATA.m function as it was the code that seemed to most directly interface with the NXT. I moved through the other existing functions attached and reviewed them to see if the nested sampling code would still work with the new modifications. I was unable to check my code with the EV3 robot as I was still unable to get the color sensor to work and could not use the NXT robot either to test it.

Results

Unfortunately, I had issues with the hardware and could not get full results from the EV3 robot due to the previously noted color sensor. I did have several test scripts that were able to articulate the motors and test the speakers. Once we address the communication issue, we can move on testing, analyzing results and fine tuning. Upon the advice of the professor, I hardcoded the values for the circle radius and center to see what sample results would look like for the project. We were able to get a good idea of the outputs and it confirmed that the matlab code was still valid. This can be further expanded upon to provide better sampling once we get the robot working.

Conclusions

There is great promise in this endeavor to utilize lego components in utilizing nested sampling on the Lego EV3 platform. I feel as though if I had a few more weeks dedicated to solely working on this project, I could make great headway addressing the color sensor error, adapting the existing code and getting testing to work. I do feel as though this would benefit from having another developer working on it alongside me for testing and refinement. I do believe that there are real possiblities to extend the service life of lower cost and lower quality hardware with bayesian approaches and targeted algorithms. We have the potential to save thousands of pieces of “obselete hardware” and repurpose them in ways seldom thought possible previously.

References

- **Interact with EV3 Brick Peripherals, Read Sensor Values, and Control Motors: https://www.mathworks.com/help/supportpkg/legomindstormsev3io/ref/interact-with-ev3-brick-peripherals-read-sensor-values-and-control-motors.html

- MINDSTORMS® EV3 Core Set Tutorials: https://education.lego.com/en-us/lessons/ev3-tutorials/

- "Having One LEGO Mindstorms EV3 Brick Connect To Another": https://www.youtube.com/watch?v=SydYVqXgGak

- MATLAB Support for LEGO MINDSTORMS EV3: https://www.youtube.com/watch?v=rp3ChAZi1Aw

- [IMPORTANT] MATLAB Meets MINDSTORMS: How to Control LEGO NXT Robots Using MATLAB for Educational Purposes: https://www.mathworks.com/videos/matlab-meets-mindstorms-how-to-control-lego-nxt-robots-using-matlab-for-educational-purposes-90418.html

- Is LEGO MINDSTORMS NXT supported in previous releases?: https://www.mathworks.com/matlabcentral/answers/95216-is-lego-mindstorms-nxt-supported-in-previous-releases

- RWTH - Mindstorms NXT Toolbox: https://www.mathworks.com/matlabcentral/fileexchange/18646-rwth-mindstorms-nxt-toolbox

- EV3 connection with bluetooth: https://www.ev3dev.org/docs/tutorials/connecting-to-the-internet-via-bluetooth/

- Simulink toolbox download: https://www.mathworks.com/matlabcentral/fileexchange/45075-simulink-support-package-for-lego-mindstorms-ev3-hardware

- EV3 software linux: https://www.ev3dev.org/

- PixyMon v2: https://pixycam.com/downloads-pixy2/

- Getting Started with LEGO MINDSTORMS EV3 Hardware: https://www.mathworks.com/help/releases/R2023b/supportpkg/legomindstormsev3/ref/getting-started-with-lego-mindstorms-ev3-hardware.html

- Connect a Host Computer Running Windows to an EV3 Brick Using Bluetooth: https://www.mathworks.com/help/supportpkg/legomindstormsev3io/ug/connect-to-an-ev3-brick-over-bluetooth-using-windows-1.html

- [Troubleshooting] Cannot Connect to the EV3 Brick: https://www.mathworks.com/help/supportpkg/legomindstormsev3io/ug/cannot-connect-ev3brick-lego.html

- PDF Documentation for MATLAB Support Package for LEGO MINDSTORMS EV3 Hardware: https://www.mathworks.com/help/pdf_doc/supportpkg/legomindstormsev3io/index.html