Wall Following With a Subsumption Architecture

Paul Johnson — PJ247

April 11, 2014

1 Introduction

Since the 1950's, the 'traditional' approach to building artificial intelligence systems has been a symbolic one. These systems focus on converting the real world into a complete and explicit symbolic representation with which a computer can reason. Representation based reasoning agents have two key problems to solve:

- **The transduction problem** of translating the world into an accurate symbolic description .
- **The representation problem** of representing information symbolically in real time.

These problems and others, lead to some researchers rejecting the assumptions under which such approaches are based(Wooldridge, 2009, p85). In the late 1980's researchers began to focus their attention on building intelligent systems which reject the necessity of a symbolic representation. These researchers suggested that intelligent behaviour is not disembodied from an environment, but is a product of the interactions the system maintains with the environment(Brooks, 2009, p139). The key premise, was that intelligent behaviour *emerges* from the interaction of simple behaviours.

To prove the viability of intelligence without representation, we have built a Lego Mindstorm robot (fig.1) which implements a representationless, behaviour based and reactive architecture, known as a subsumption architecture. The goal of the robot is to navigate a room by following the walls and avoid becoming trapped.

2 Approach

The subsumption architecture is composed of a number of task achieving layers, whereby each layer is composed of a fixed-topology network of simple finite state machines (fig.2). Each layer, or *behaviour* has access to the sensor data and works independently of the other layers. The lower down the

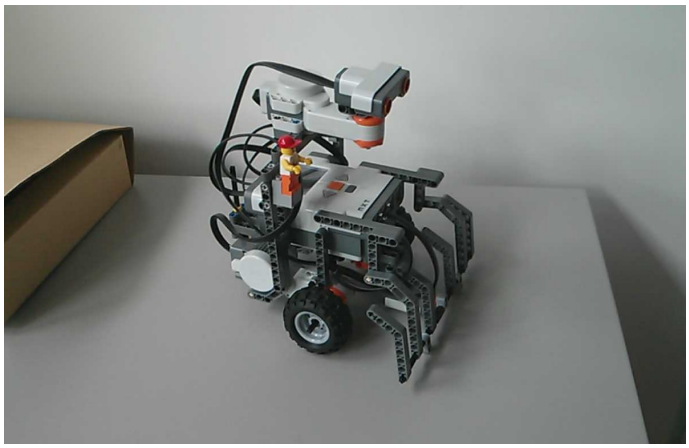

Figure 1: The Lego robot used to test a subsumption architecture

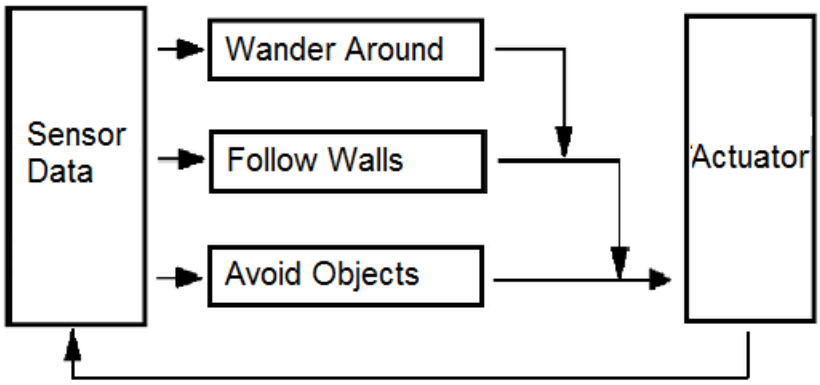

Figure 2: Simplified visualisation of subsumption architecture

hierarchy the layer is, the more fundamental/primitive the behaviour is, and the higher a priority this action takes. For example, if the bottom layer detects that the robot will imminently collide with an object, and a higher layer is attempting to take the robot towards a location, the bottom layer will take priority and execute an action to divert the robot from the collision path.

The robot (built jointly by Alex Aiton and myself) was designed with two motorised and independently turning front wheels, with a central, unmotorised rear wheel. The robot relies on two sensors for environmental input: a front mounted touch sensor bumper, and a rotating ultrasound distance sensor, located on top of the robot, above the control unit.

The reactive subsumption architecture was implemented via the leJos subsumption library. We first created an arbitrator, which controls which behaviour object has priority. To this arbitrator, we added a number of leJos behaviour interfaces, which consist of i) an action, ii) a suppressor and iii) an activator function. The activator function defines the pre-conditions required for the behaviour to become active; the action function determines which actions the behaviour should attempt to execute, and the suppress function is called to override the action function and force the behaviour to exit.

The two stipulations Brooks strongly advocates, when building a subsumption architecture are: "We must incrementally build up the capabilities of the intelligent systems" and "At each step we should build complete intelligent systems that we let loose in the real world with real sensing and real action"(Brooks, 2009, p139). To this end, we fully tested the robot after each behaviour was implemented.

The first capabilities we programmed the robot with, were the random walk and forward collision detection behaviours. These two behaviours represented opposite ends of the hierarchy, with the collision detection behaviour taking high priority, and the random walk behaviour being executed when the robot had nothing else to do. The layer above collision detection was a proximity detection behaviour. If the ultrasound sensor was facing forward and detected a wall was close, the robot would turn. The next behaviour implemented was the follow wall behaviour. Once a wall was detected the robot follows the wall by moving in a straight line, and periodically rotating its ultrasound sensor to detect whether the wall was still close to its left. If a wall was lost, the follow wall behaviour would attempt a left turn, to try to find the wall it lost. The follow wall behaviour also had the ability to align the robot with the wall, by moving 10 degrees towards or away from the wall, if it senses it is a little too close or too far.

Figure 3: Simple course designed with only right angle turns

Figure 4: More complex course designed to test robot in uneven environment

3 Result

The first test environment (fig.3) was run at full speed and half speed 5 times each. At full speed the robot managed to navigate in an average of 51.42 seconds, with standard deviation 5.52 and an average of 2.4 collisions. At half speed, the robot navigated the course in an average of 58.92 seconds, with standard deviation 1.88 but only one collision over the 5 tests (0.2 average).

The second test environment (fig.4) was designed to test the robot's ability to follow uneven walls at half speed. Although the robot was capable of completing the circuit in an average of 85.2 seconds, the standard deviation was 22.4, with the average number of collisions being 3.

The robot's ability to navigate obstacle courses where the walls were lower than its ultrasound sensor was tested in two environments; one with a text book added to the side of environment 1 and another where the robot navigated around tables (fig.5). In both cases the robot managed to navigate the course successfully, with the assistance of its bumper touch sensor.

Figure 5: Tables used to test the robot in a real world scenario

4 Discussion

This paper's primary objective was to display the benefits of the representationless subsumption architecture. To prove a subsumption architecture is 'superior' to a symbolic one, would require recreating the conditions of this experiment and comparing move calculation times of the two architectures, along with the 'intelligence' of their respective action decisions. However, this was beyond the scope of the paper.

The robot had a number of limitations when demonstrating the speed at which it made decisions, mainly the rotation speed of the sensors. This limitation is clearly displayed by the large increase in collisions of the robot going at full speed and also the larger standard deviation, which shows the robot to be less reliable since it often takes different routes. The full speed robot would often collide with objects in the time it took for the sensor to rotate back to the front. This limitation could possibly be counteracted by equipping the robot with a secondary distance sensor attached to the side of the robot and removing the need to rotate the sensors.

5 Conclusion

Using a simple subsumption hierarchy, the robot we built successfully navigated each of the environments it was subjected to and without much digression from its objective (going too far off course). The robot stored no information on the environment and each decision it made was based on the outcomes of each finite state machine layer of the architecture. The results demonstrate that a representationless architecture can indeed be used to build an intelligent robot.

References

M. Wooldridge. (2009). *Introduction to Multiagent Systems.*2nd Ed. John Wiley & Sons, Inc., P49-85.

Brooks, R. A. (1991). *Intelligence without representation.* Artificial Intelligence, 47(1–3):139–159.