Honours Project Literature Survey

Leslie Luyt
Supervisor: Dr. Karen Bradshaw
Department of Computer Science
Rhodes University

Grahamstown, South Africa
April 2009

1 Introduction

This paper introduces and defines the main concepts and issues related to creating a framework for programming a robot with artificial intelligence. It also includes background concepts on programming artificial intelligence in general, as well as a description and explanation of the robotics kit that shall be used.

The process of creating the artificial intelligence programming framework consists firstly of implementing artificial intelligence methods. Secondly, the artificial intelligence needs to be reconfigured to take robotic movement input, or data from the sensors; and produce output that is sensible for robotic processing. Finally, it is required that all of the above is encapsulated inside of a programming framework. This framework should reduce the difficulty of implementing artificial intelligence when working with robotics, and should be versatile enough such that most tasks can be easily accomplished using it.

2 Artificial Intelligence

2.1 Neural Network Approach

2.1.1 What is a Neural Network?

An artificial neural network is a computational model defined by a directed graph; where each node, or vertex, in the graph has inputs and outputs. These inputs and outputs form the edges of the graph and are associated with a weight function, or a value. This network provides a practical method for learning real, discrete, and vector valued functions from examples. After the learning phase has been completed the associated weights will provide the required output for all training data. However, these weights may be difficult to explain and interpret, and all that can be inferred is just a numerical relationship between two nodes. [3, 9]

2.1.2 How does it learn?

The artificial neural network learns by taking in a list of example input and the corresponding correct output for each input. For every input given, it uses the current network

to compute an output. It then calculates the error between the given output and the current output, and adjusts the weights by backward propagation to minimise the error. After the weights have been adjusted it re-computes the input example. It follows this loop until the network correctly produces the required output for each input, or the learning process returns with an error stating that the current network configuration cannot correctly simulate the function that maps the given input to the given output. The system will then have to alter the structure of the network, by adding or deleting more nodes or layers as is appropriate; and then re-attempt performing the learning process.

Once the learning process has been completed successfully, we have a correct neural network structure and weights for the function we wish to emulate. We can then proceed to feed in inputs for which we do not know the output and observe what the neural network produces. The resultant output produced by the neural network may not necessarily be correct, as there may be missing cases in the example input and output which lead to only a partial solution being formed. [3, 9]

2.1.3 Relevant Research

Yang and Meng suggest that a neural network is sufficient for real-time collision-free path planning in a dynamic workspace, claiming that the model does not suffer from either the "too close" or "too far" problem. [18]
Na and Oh proposed a hybrid control architecture based on reactive navigation, that works by using a neural network to classify sixteen topological maps roughly describing the environment. [11]

2.2 Bayesian Network Approach

2.2.1 What is a Bayesian Network?

A Bayesian network is a probabilistic graphic model defined by a directed acyclic graph; where each node represents a random variable and the edges between nodes represent probabilistic dependencies between the random variables associated with the connected nodes. After the learning phase has been completed the probabilities associated with the edges will provide the required solution. These probabilities tend to be relatively easier to interpret than the neural network weights. [2, 3, 9]

2.2.2 How does it learn?

When the structure of a Bayesian network is unknown at the outset, it is necessary that we learn the structure from the given training data. This problem is known as the BN learning problem, which can be stated informally as follows: Given training data and prior information, estimate the network structure and the parameters of the joint probability distribution in the BN. Once this initial estimation has been done, the system falls into one of the four cases shown in Table 1:

Table 1: Four cases of BN learning problems

Case	BN structure	Observability	Proposed learning method
1	Known	Full	Maximum-likelihood estimation
2	Known	Partial	EM (or gradient ascent) MCMC
3	Unknown	Full	Search through model space
4	Unknown	Partial	EM + search through model space

For case one, the training can be performed by maximising the log-likelihood of each node independently.
For case two, the expectation maximisation algorithm can be used to find a locally optimal maximum-likelihood estimate of the parameters.
For case three, the goal is to find a directed acyclic graph that explains the data. An approach to solving this NP-Hard problem is to take the naïve approach and assume that each node is conditionally independent, thereby simplifying the problem.
For the fourth case, finding the directed acyclic graph is an intractable problem. Thus, an approach is to use asymptotic approximation to the posterior called Bayesian information criterion. This approach will marginalise out the hidden nodes and parameters. [2, 3, 9]

2.2.3 Relevant Research

Olivier Lebeltel et al. propose a new method of programming robots called BRP (Bayesian Robot Programming), which uses a Bayesian Network to control the decision making of the robot. [7] ***expand section***

3 Programming Frameworks

3.1 What is a Programming Framework?

A programming framework is an abstraction of complicated, or simple, tasks that reduce the time it takes for a programmer to perform these tasks. A programming framework should be completely reusable and encapsulate any required resources, such as: dynamic shared libraries; nib files; image files; localised strings; header files; and reference documentation in a single package. The programming framework should be relatively intuitive and fairly easy to understand, without the developer needing to read the source code to get the system working. [1, 15, 20]

3.2 Why use an Object-oriented Framework?

Frameworks designed using the Object-oriented approach promise higher productivity for the developer, by increasing the ability for code reuse and better design. As a result a shorter time, in comparison to non-framework based approaches, is taken from the beginning of production to when the product is finally marketed. [1, 15]

3.3 Components of a Programming Framework

Generally, a programming framework will model a solution to a specific problem or a specific domain. Thus, they are geared toward providing methods and abstractions that aid in working in the specified domain or solving the given problem.

A programming framework is generally composed of the following:

- Generic setup and instantiation methods
- Specific problem solving methods
- Groupings of methods that problem solve using a specific algorithm [15, 20]

4 FischerTechnik Robotic Kit

4.1 FischerTechnik Kit Contents

The FischerTechnik set used is the Robo Mobile Kit with additional sensors as listed below. [5]

The components available include:

- Lego-like pieces to construct a robot.
- The fischertechnik ROBO Interface.
- 2x 9V DC Motors
- Sensors included in the set are: 2x Light sensor; and 4x Touch sensors.
- Additional sensors: 2x Ultra-sonic Distance sensors; and 1x Colour sensor.
- ROBO Pro software for the ROBO Interface.
- A 9V DC/1A rechargeable battery.
- A 9V DC/1A power converter to recharge the rechargeable battery.
- An instruction booklet containing building instructions for 8 different models buildable with the set.

4.2 Controller Specifications

The 16-bit micro-controller has both USB and serial interfaces, and comes with 128 kBytes of flash memory for the download of two separate programs. These programs are retained in the flash memory when the power supply is disconnected. The board also has: four 9V/250mA (max. 1A) motor outputs now with variable speed control; eight digital inputs; two analog inputs for resistances from 0-5kO; two analog inputs for voltages from 0-10V and two inputs for digital distance sensors.

Connections are available for the following add-on modules: one ROBO I/O Extension expansion module; one ROBO RF Data Link radio interface; and an interface to infrared transmitter from the IR Control Set.

All the inputs and outputs run through a 26-pole male connector strip, for convenient connection of finished models through a single 26-pole connector. [5]

4.3 Supported Languages

Apart from the ROBO Pro software provided by FischerTechnik for the kit; there exists a custom built DLL for the FischerTechnik sets, such that it is compatible with the following major programming languages:

- Delphi 4 (or up to Delphi 7)
- Java 2 (SDK 1.2.2 or higher)
- JScript
- MSWLogo (32bit, engl.)
- Microsoft CSharp 2005 .NET 2.0
- Microsoft Visual C++ 2005 .NET 2.0
- Microsoft Visual Basic .NET 2005
- Perl
- Python
- Visual Basic 6 (includes VBA, VBS)

This functionality can also be extended to any other programming language that can reference a C++ complied DLL if you desire, but will require you to write the wrapper class or code to reference the functions contained inside. [5, 10]

Programming can also be achieved in the Microsoft Visual Studio 2008 Development Suite, by simply updating all the calling methods to be compatible with the updated language and compilers. This can generally be done by simply changing and updating what is specified in the change-log between Microsoft Visual Studio 2005 and Microsoft Visual Studio 2008.

5 Programming Languages

5.1 Evaluation of ROBO Pro

5.1.1 Description of ROBO Pro

The ROBO Pro software is a flowchart programming environment provided by FischerTechnik for the ROBO Interface. Being a flowchart programming environment, it is easy for beginners to use as they can string blocks together without knowing the inner workings. In addition, program operation is easier to understand as all the details are abstracted away by the flowchart blocks in an object-oriented fashion.

The ROBO Pro software consists of various software modules, which can interchange data between themselves or even subroutines using not only variables but also graphical or flowchart connections. Subroutines are stored in a library and can be used without the necessity of understanding the internal workings of the subroutine.

The graphical programming language ROBO Pro provides all the key elements of a modern programming language, such as arrays, functions, recursion, objects, asynchronous events and quasi-parallel processing. All programs are translated directly into machine language for efficient execution on the ROBO Interface. These facts make it a useful tool even for professional programmers.

When running the ROBO Pro software in online mode, it is possible to control multiple ROBO Pro interfaces in parallel for large-scale models; and to make custom control panels which include switches, controllers and display elements. [5]

5.1.2 ROBO Pro Limitations

Given the wide functionality and seemingly limitlessness of the ROBO Pro language, one would think this would be the ideal programming language for any development using the ROBO Interface. However, this is not the case as the ROBO Pro language lacks portability and only runs on Microsoft Windows operating systems. The ROBO Pro software is also not translatable, and has no translation tools to other programming languages. Since ROBO Pro software is not freeware or open source, if one wants to make use of anything developed using this software one must first purchase a licence before using the application. Due to it being untranslatable, it is not ideal for research; as the

research becomes severely limited and only relevant to those with a ROBO Pro licence. However, the most important factor is that the ROBO Pro language is not framework compatible. There is no way to create a further level of abstraction upon the one in which it currently operates. [17]

5.2 Evaluation of C++

5.2.1 Description of C++

C++ is a standardised, procedural, cross-platform, dynamic, object-oriented, high-level programming language. The C++ language is, in essence, an extension of the C language to include the Object Orientated Design approach to programming. C++ has a large collection of standard libraries that can be imported and used, and has high community support due to its extensive use. [4, 19]

5.2.2 Benefits

C++ has great portability making it possible to compile the same C++ code on almost any type of computer and operating system, without making any changes. Code written in C++ tends to be very short in comparison with other procedural languages, since the use of special characters are preferred to long key words. This preference of special characters decreases the amount of time necessary for the programmer to spend typing, and thus increases their productivity.

C++ is designed as a modular language, with an application's body often being made up of several source code files that are compiled separately before being linked together. This saves in compile time when modifying large applications, as it is not necessary to recompile the complete application but only the file that has been changed. In addition, this characteristic allows the linking of C++ code with code produced in other languages, such as Assembler or C.

C++ compiles to native byte-code for a specific architecture, and thus the resulting code is very efficient and can be run on the processor without any interpretation or translation required. The efficiency is also improved by the fact that C++ functions as both a high-level and low-level language, and by the reduced size of the language itself; the produced byte-code is well optimised for the system is was compiled on. [19]

5.2.3 Limitations

Although the conciseness of C++ is an advantage for programmers, combined with the wealth of operators available to the programmer; it is also a disadvantage as it can make code difficult to follow. Given this and C++'s freedom of expression with regards to pointers, can make very obfuscated code that is impossible to follow and debug. [12]

5.3 Evaluation of Python

5.3.1 Description of Python

Python is an interpreted, interactive, cross-platform, dynamic object-oriented language, very-high-level programming language (VHLL) that provides strong integration and support when combining with other languages. Python is simpler, faster to process, and more regular than classic high-level languages. Python also comes with an extensive set of standard libraries, providing basic functionality in many areas by simply using and including the correct library. Python is stated as boasting a multitude of features, such as: exception-based error handling; high portability; full modularity; supporting hierarchical packages; intuitive object orientation; very clear, readable syntax; very high level dynamic data types. [8, 13, 16]

5.3.2 Benefits

Python has been designed to be modular, with a small kernel that is extended by importing extension modules, or packages. The standard Python distribution includes a diverse library of extensions for operations ranging from string manipulations and regular expression handling, to Graphical User Interface generators, operating system services, and debugging and profiling tools. New Python extension modules can be created to extend the language, and can be written in Python, C or C++. [8, 14, 16]

Python, as with most other scripting languages, supports numerous high-level constructs and data structures that enable you to write shorter programs than those, with similar functionality, written in C, C++, C#, or Java. Where Java requires a loop, Python may require only a single line statement to perform the same operation. This significantly increases the maintainability and readability of code, thereby producing a better end product than possible with more traditional languages. [6, 14, 16]

5.3.3 Limitations

Since Python is an interpreted language, some performance loss is to be expected. This is a direct result of hardware-independent byte-code being converted to hardware-dependant byte-code, by being run through an interpreter, before execution can occur. However, Python code runs with reasonable performance, sufficient not to be a significant loss; and in most cases will only run marginally slower than C code for the same tasks. The Python coding style used can also alleviate this problem by using extensions, such as the Numeric extension, which provides good performance when working with large arrays of numbers. [6, 8, 14, 16]

6 Conclusion

Artificial intelligence techniques were evaluated and it was found that both the neural networks approach and the Bayesian networks approach are very powerful techniques that can be very useful tools in learning applications for robotics. Both of these approaches have sufficient power to adapt to changing environments, and after the initial learning phase is completed will perform well in most situations.

The Fischertechnik kit was explored, and each of its components investigated to find the full extent of their usability. The micro-controller included with the set was also sufficiently researched such that it could be understood exactly what the kit is capable of.

The essence of creating a programming framework was also researched, which included: what to put into the framework; how to structure the framework; and how best to create a framework useful to the end-user.

Finally, both the Python and C++ programming languages were evaluated, and were found to be very powerful languages. The clean and simple syntax of Python, with its advanced built-in data types and the expansive standard libraries, make it an attractive programming language for beginners and experienced programmers alike. Whereas the powerful features, and speed of C++ combined with the extensive libraries available make it a good candidate for any advanced development. Both Python and C++ are highly extensible with a powerful modular system for easily adding extra functionality or libraries. All the above features make Python and C++ both highly suitable languages for use in creating an artificial intelligence framework extension.

7 Plan of Action

My plan of action for the vacation is the following:

- Begin design of the Programming Framework
- Become more comfortable programming the robot
- Start gathering together resources and working paper algorithms for implementing Artificial Intelligence

My plan of action for after the vacation is as follows:

- Begin implementing the Artificial Intelligence algorithms
- Start building the programming framework
- Slowly integrate framework and algorithms

References

[1] Apple Inc. Framework Programming Guide. Online, May 2009. Available from: `http://developer.apple.com/documentation/MacOSX/Conceptual/BPFrameworks/Concepts/WhatAreFrameworks.html`.

[2] Ben-Gal, I. E. *Encyclopedia of Statistics in Quality and Reliability*. John Wiley & Sons, 2007.

[3] Bender, E. A. *Mathematical Methods in Artificial Intelligence*. IEEE Computer Society Press, 1996.

[4] Cprogramming.com. Cprogramming.com - Your Resource for C and C++ Programming. Online, June 2009. Available from: `http://www.cprogramming.com/`.

[5] FischerTechnik. Fischertechnik Official Webpage. Online, May 2009. Available from: `http://www.fischertechnik.de/en/`.

[6] Langtangen, H. P. *Python Scripting for Computational Science*. Springer, 2008.

[7] Lebeltel, O., Bessire, P., Diard, J., and Mazer, E. Bayesian robot programming. *Autonomous Robots 16* (2004), 49–79.

[8] Martelli, A. *Python in a Nutshell, (2nd Edition).* O'Reilly, 2006.

[9] Mitchell, T. M. *Machine Learning.* McGraw-Hill Book Company, 1997.

[10] Müller, U. ftComputing. Online, May 2009. Available from: `http://www.ftcomputing.de/index.htm`.

[11] Na, Y.-K., and Oh, S.-Y. Hybrid control for autonomous mobile robot navigation using neural network based behavior modules and environment classification. *Autonomous Robots 15* (2003), 193–206.

[12] Prata, S. *C Primer Plus*, 5th ed. SAMS, 2005.

[13] Python.org. About Python. Online, May 2009. Available from: `http://www.python.org/`.

[14] Python.org. Python FAQ. Online, May 2009. Available from: `http://www.python.org/doc/faq/`.

[15] Riehle, D. *Framework Design: A Role Modeling Approach.* PhD thesis, Swiss Federal Institute Of Technology Zurich, 2000.

[16] Sanner, M. F. Python: A programming language for software integration and development. *J. Mol. Graphics Mod 17* (1999), 57–61.

[17] Schreuders, P. D. Engineering modeling using a design paradigm: A graphical programming-based example. *Journal of Technology Education 19*, 17 (Fall 2007), –.

[18] Simon X. Yang, Max Meng. Real-time collision-free path planning of robot manipulators using neural network approaches. *Autonomous Robots 9*, 1 (August 2000), 27–39.

[19] Soulie, J. C++ : A brief description. Online, May 2009. Available from: `http://www.cplusplus.com/info/description/`.

[20] Zend Technologies Inc. Zend Framework Programmer's Reference Guide. Online, May 2009. Available From: `http://framework.zend.com/manual/en/`.