Criteria For Selection of Software Development Environment For Construction Robotic Systems

Khaled Zied and Derek Seward
Engineering department, Lancaster University, Lancaster, LA1 4YR, UK
k.zied@lancaster.ac.uk & d.seward@lancaster.ac.uk

ABSTRACT: The selection process for a suitable programming environment for construction robotic systems should satisfy a range of requirements identified from both a users and systems point of view. In the present work two different object-oriented programming environments are chosen for comparison, namely MATLAB as an example of text-based programming and LabVIEW as iconic-based programming. The selection of the appropriate development environment is performed using the AHP process for decision-making. Several criteria and sub-criteria are identified and used for the selection process. A complete hierarchy of the problem is constructed and priority vectors are identified. Sensitivity analysis on the results is performed to identify the factors affecting the final decision. For the entered values of the priority vectors, the obtained result shows a preference for LabVIEW over MATLAB as a software development environment for construction robotic systems.

KEYWORDS: Robotics, Software, AHP, Decision-making

1. INTRODUCTION

The development of robotic systems in construction advances very slowly owing to several challenges, [Garas]. One of the obstacles is the development of the required software components. This is a major obstacle because of the requirement for highly trained programmers and expert software engineers. Software represents a substantial part of the overall complexity of a robotic system. In most development systems, software is the component on the critical path and is usually blamed for system development problems, [Stevens].

In a robotic system, software plays a vital role in many sub-systems, including the controller, sensors and user interfaces [Seward & Zied]. Hence a robotic system can be considered as computer-based system CBS. A CBS is a mixture of software, hardware and people but the software is considered as the core of the system and the key element for cost, added value and risk [Thome]. The need to apply Systems Engineering principles is clear because the design of software needs decomposition, risk management, interface control and integration, which are the elements of Systems Engineering as described by [Stevens] and [Martin].

The software development process requires a powerful programming environment that can produce functional and reliable software to satisfy the end user needs, as well as the developers' needs. The characteristics of a robotic system oblige us to select a powerful software development environment that enables modularity, easy integration and reusability. For example, in the present project, the Starlifter robot –See Figure (1) [Zied 2001], the robot controller, ATC is working on Windows 3.1 operating systems which uses 16-bit data format, the operating software for the RotoScan, laser scanner for range measurements [Seward 2002] is a DOS based program etc. This illustrates one of the difficulties involved in integrating these components of software into one user interface The need for a programming environment capable of dealing with these problems is obvious. This environment must allow modular design and easy interfacing with other software written in different programming languages etc. Another problem arises from the fact that these systems are invariably one-offs or low volume products, and so the resources that can be invested in software are severely limited. Object oriented programming environments fit the above requirements; it is however difficult to start from scratch in the software development process which implies the need for ready-made components that reduce the development time and cost.

Figure (1) The Starlifter robot

In the present work two software development environments are considered for comparison, MATLAB which is a text-based programming environment (TPE) and LabVIEW which is an iconic-based programming environment or graphical programming environment (GPE). Capabilities of both environments are examined to reach a final decision as to which one is appropriate for the development of software for construction robots.

2. THE SELECTION PROCESS OF SOFTWARE DEVELOPMENT ENVIRONMENT

Most researchers choose a software development environment according to their personal preference and the skills they already have in programming. Current personal skills and the availability of the development environment affect the decision of whether or not to use a certain development environment. For example people familiar with MATLAB or VC++ tend to choose them in the first place regardless of the capabilities of the environments and the development time that these environments will consume.

From the authors' personal experience, firstly, it was decided to use MATLAB as the software development environment because of past experience in programming. However, after spending some time MATLAB programming it was found that excessive programming resources would be required to complete the system. VC++ was tried for creating user interfaces but it was found difficult and time consuming. Eventually it was decided to use LabVIEW because of the availability of the hardware that required interfacing with the robot.
In the following section, selection criteria are presented to enable developers to choose the right software development environment from different points of view. The selection process is based on the AHP process [Saaty].

The starting point of the selection process is to make a hierarchy of the problem showing the goal of the problem and the alternatives.

2.1 The Problem Hierarchy

Whitley and Blackwell [Whitley] presented a comprehensive survey-based study on visual programming versus textual programming. In this study they compared two types of languages, textual and iconic.. This comparison is based on surveys between programmers who use these two types of languages. It is important to identify that they used the term visual programming to represent iconic programming, which in the present study is referred to as Graphical Programming. The results of these surveys showed that there is a great difference in opinion between academics and professionals regarding the capabilities of programming languages in general. "*Researchers have ambitious theories regarding the influence that new programming languages can exert on mental processes of the programmers"* Professional programmers are mainly concerned with productivity, which is represented by reusability, and they prefer their existing tools.

Graphical Programming users admitted that the visual representation of functions is more advantageous than re-usability in LabVIEW.
[Whitely] presented several criteria, to compare textual and graphical programming. These criteria are regrouped in the present work into four main criteria. These criteria in addition to other criteria related to personal experience are used to select a software development environment. The main criteria are:

1. General criteria
2. Technical (Beginners) criteria
3. Technical (Advanced) criteria
4. Practical criteria

Figure (2) shows the hierarchy of the problem, the first level is the objective or the goal, which is the selection of a software development environment. The second level is the selection criteria and the third level is the alternatives, which in the present study are, LabVIEW and MATLAB.

2.2 Priorities Setting

The AHP process consists of two main steps; the first step is the pairwise comparison between criteria at the same level i.e. comparing the relative importance between the main criteria in the A-level and between the sub-criteria in each main criteria. The second step is comparing the preference of one alternative over the other relative to the individual sub-criterion.

In the present work all of the data supplied to the process are based on the authors personal judgement and past experience. The process analysis is performed using Expert Choice 2000 software (EC2000). EC2000 is based on the principles of the AHP process developed by [Saaty]. The data obtained is verified manually to confirm the correctness of the output data. The following section illustrates examples of the output from the AHP process:

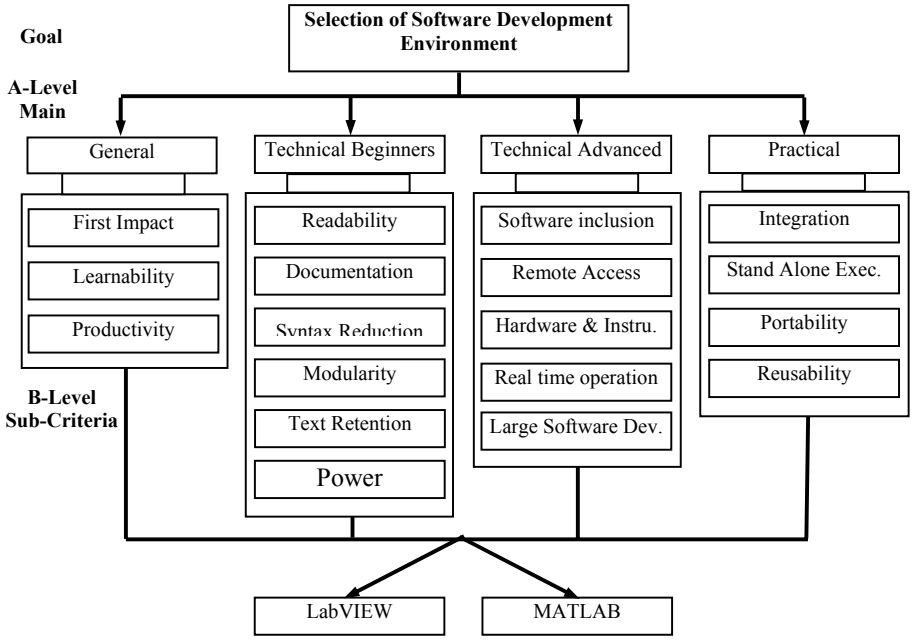

Figure (2) The selection problem hierarchy

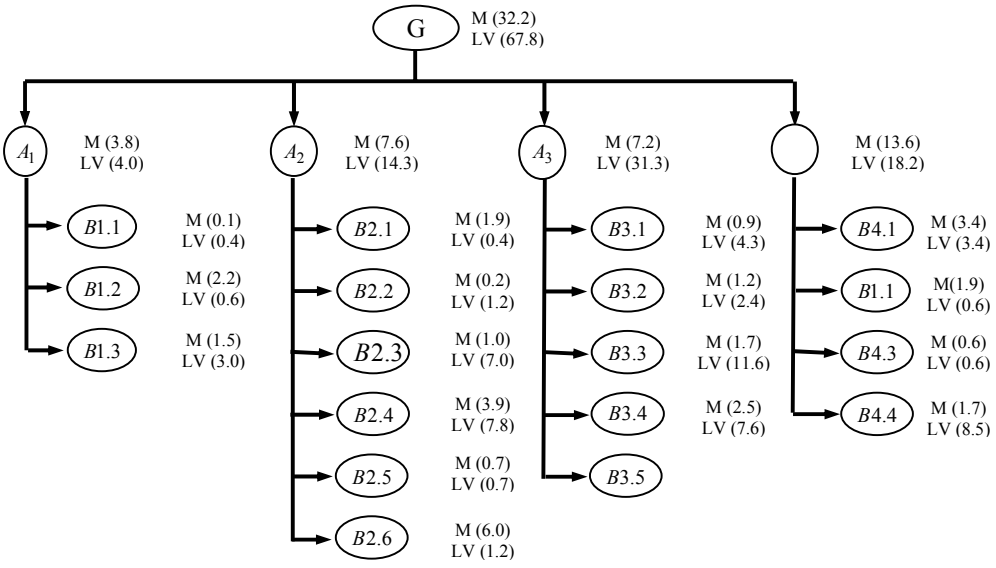

Figure (3) Percentage contribution of sub-criteria for MATLAB (M) and LabVIEW (LV) at different levels of the hierarchy in the decision process

2.2.1 Level A: Main criteria

Compare the relative importance between the main criteria

		A_1	A_2	A_3	A_4	Weight Vector
General	A_1	1	1/5	1/5	1/2	0.078
Technical Beginners	A_2	5	1	1/2	2	0.317
Technical Advanced	A_3	5	2	1	1	0.387
Practical	A_4	2	1/2	1	1	0.218

Inconsistency 0.08

2.2.2 Level B: Sub-criteria

Compare the importance between the sub criteria relative to A1 = General criteria

		$B1.1$	$B1.2$	$B1.3$	PV	WV (B1)
Impact	$B1.1$	1	1/5	1/5	0.065	006
Learnability	$B1.2$	5	1	1/2	0.361	.032
Productivity	$B1.3$	5	2	1	0.574	.045

WV Weight vector and PV Priority Vector

2.2.3 Level C: Alternatives

Analysis with respect to B1.1= Impact
Weight of B1.1=0.0057, refer to weight vector B1

		$C1$	$C2$	Priority Vector	Weighted Vector
LabVIEW	$C1$	1	3	0.75	0.004257
MATLAB	$C2$	1/3	1	0.25	0.001425

2.2.4 The aggregate vectors

Calculation for the alternatives relative to the A1 = General

	Analysis with respect to:			
Criteria A1	$B1.1$	$B1.2$	$B1.3$	Aggregate Vector
LabVIEW C1	0.004257	0.00564	0.02988	0.039777
MATLAB C2	0.001425	0.02256	0.01494	0.038925

2.2.5 The aggregate matrix

	C1	C2
A_1	0.143	0.074865
A_2	0.31473	0.0718
A_3	0.18128	0.1353
A_4	0.039777	0.038925
Sum	0.679	0.321

2.2.6 The final priority vector

This vector shows the preference of LabVIEW (67.9%) over MATLAB (32.1%)

$$\begin{bmatrix} C1 \\ C2 \end{bmatrix} = \begin{bmatrix} 0.679 \\ 0.321 \end{bmatrix}$$

2.3 Analysis of Results

Structuring the problem in the way the AHP process required not only makes the problem formulation easy but also makes it clear which criteria or sub-criteria influence the final decision. The final decision, which can be extracted from the final priority vector, shows the preference for LabVIEW over MATLAB according to the entered judgements at each level. The judgements in the lowest level of the hierarchy influence greatly the final decision however; the relative importance between criteria in each level increases or decreases the contribution of the initial judgements in the lowest level. Figure (3) shows the percentage contribution of each criterion in the final decision. It is clear from this figure that the influence of the technical advanced criteria has more effect than the other criteria in which the percentage contribution reaches about 47% of the total percentage of the LabVIEW preference. The lowest contribution in the final decision in the A level criteria is shown by the general criteria for the LabVIEW preference which is the same for the MATLAB preference. The practical criteria contribution is the highest towards the MATLAB preference.

2.4 Sensitivity Analysis

Because the judgements made here are based on personal experience it is necessary to show how sensitive the preference for LabVIEW over MATLAB is to the change in the priority vector at each level. Any change in the priority vector means a change in the inherent judgements entered by the user. Therefore, identifying the relative importance of the entered values must follow any change in the priority vector.

Figure (4) shows the gradient graphs of the priority vectors of the main criteria and the effect of changes have been made to the preference of MATLAB and LabVIEW. It is obvious from this figure that there is no break-even point which indicates that changes to the individual priority vectors will not provide a preference for MATLAB over LabVIEW. However the general criterion gives a break-even point when it is the only criterion at this level. This shows that it is necessary to go to a lower level to change the priority vector.

Figure (5) shows example gradient graphs for B1 criteria, it is obvious that a break-even point can be achieved for different sub-criteria, which implies the possibility of changing the decision at this level. A break-even point for certain criteria means an equal priority of the two alternatives. The change in one of the elements of the priority vector means proportional changes in the other elements to keep the judgements consistent. Analysis of the results shows that any positive change in the priority vector for the B2 criteria increases the preference of MATLAB over LabVIEW. Keeping the modularity sub-criteria priority, increasing the weight for the power and computability and reducing the weight for syntax reduction can achieve the preference for MATLAB over LabVIEW. This increase in MATLAB preference is due to the original preference of MATLAB in the sub-criteria B2.6 in which MATLAB is 5 times preferred to LabVIEW. Sensitivity analysis is a good tool for advising on the required change in priorities for a proper selection of tradeoffs.

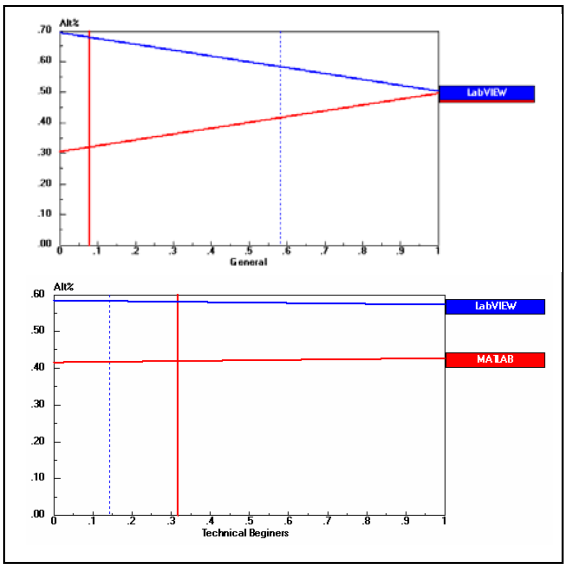

Figure (4) Gradient graphs of the A-level priority vectors

3. NON-COMPARABLE FEATURES OF MATLAB AND LABVIEW

As the above analysis is based on pairwise comparisons it is necessary to identify other advantages that exist in one environment and not in the other. These advantages may support the use of one environment over the other if it is essential for the development.

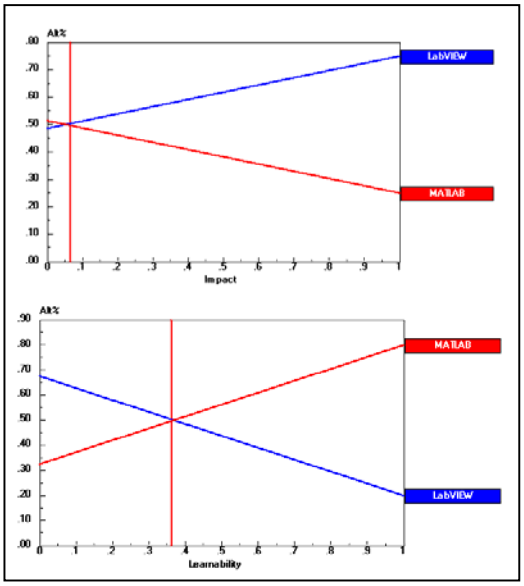

Figure (5) Gradient graphs of the B1 level priority vectors (B1.1)

3.1 MATLAB Advantages:

1. *Specialist toolboxes* - this feature enables the use of specially designed functions and to implement them directly in the program. For example, the robotic toolbox [Corke], the system identification toolbox, and the control toolbox contain powerful functions to reduce programming time. LabVIEW contains functions similar to some toolboxes such as signal processing, data acquisition etc.

2. MATLAB C compiler, this feature enables the user to convert MATLAB code into a C-code executable or dynamic link library (DLL). This enables real time operation for real time critical systems and facilitates stand-alone applications.

3.2 LabVIEW Advantages

1. The code interface nodes CIN, this feature enables the user to bring all the features of other programming languages inside LabVIEW. For example, MATLAB, HiQ and C interface nodes, which allow the implementation of existing c code inside programmes made with LabVIEW. The good thing in using this feature is for example using the powerful MATLAB toolboxes inside LabVIEW.

2. Another form of interface node is the dynamic link library interface node, which allows the interface of other software libraries. This in conjunction with the MATLAB c-compiler can bring the real time functionality of the MATLAB toolboxes inside LabVIEW.

3. Polymorphism is a feature in LabVIEW that allows the automatic change of data types without conflict.

4. DISCUSSION

For software development, the concepts of SE provide solutions to most of the interfacing and integration problems however the tools to carryout these concepts need to be identified. Two programming environments allow the use of these concepts namely, MATLAB and LabVIEW. The selection process for the most appropriate environment can be carried out using the AHP process, which requires the establishment of selection criteria. Pairwise comparison of criteria with respect to suggested alternatives provides a systematic way of decision-making. Pairwise comparisons can be obtained from users of the two environments or from other supporting information. A sensitivity analysis is needed to identify the change in the decision if any of the priority vectors change. The output from the AHP shows a preference for LabVIEW over MATLAB for the values considered in this study. Practically, Graphical Programming allows logical top-down architectural design and the decomposition of the software components. For ready-made components, which require the use of a specific data format, it is possible to use them directly thanks to the polymorphism of the data types in LabVIEW. It is possible to use top-down decomposition without the worry of the interfacing or changing the logical top down architecture. A crucial property available in Graphical Programming is modularisation of the software package components. Modular design requires clear interfaces between the system modules; the versatility of Graphical Programming provides different interfaces, which allow easy integration of the software package components. For example, in case of the Starlifter controller software, the MATLAB robotics toolbox is used for kinematics calculations.

5. CONCLUSIONS

The AHP process provides a good systematic tool for decision-making in case of multiple criteria problems such as software development. A software development environment should satisfy robotic systems development principles, which are based on systems engineering principles. Architectural design, modularisation and prototyping are important concepts that should be employed in the software development process. Selecting an appropriate software development environment involves many issues, which depend mainly on experience. However using the suggested criteria helps in the decision to select a particular programming environment. Sensitivity analysis on the priority vector provides a good tool for supporting the final decision by examining the shift in decision caused by the preference of one criterion over others. The results obtained from the AHP process reflect the practical situation in which it was found that graphical programming provided powerful capabilities to assist in the rapid development of software for construction robots.

6. REFERENCES

[Corke] Corke, P.I., 1996, "A Robotics Toolbox for MATLAB", *IEEE Robotics and Automation Magazine,* v 3(1), p 24-32.

[Garas] Garas, F., 1996,"Automation of the UK construction Industry-Current status and key issues". *Proceedings of the 13th ISARC*, p 43-50, June 11-13, Tokyo, Japan.

[Martine] Martine, James N., 1997, "Systems guidebook, a process for developing systems and products". Lucent Technologies, Florida, USA, ISBN 0-8493-7837-0.

[Saaty], Saaty, 1980, The Analytic Hierarchy Process, McGraw Hill, New York.

[Seward & Zied] Seward, D. W. and Zied, K., 2004, "Graphical Programming and the Development of Construction Robots, Computer-Aided Civil and Infrastructure Engineering, 18.1,

[Seward 2002] Seward, D. Quayle, S., Zied, K. and Pace, C., 2002, "Data interpretation from Leuze Rotoscan sensor for robot localization and environment mapping",19th ISARC, Sept. 2002, p343-349, NIST Gaithersberg, USA.

[Stevens] Stevens, R., Brook, P. Jackson, K. and Arnold, S., 1998, "Systems engineering, coping with complexity". *Prentice Hall Europe*, ISBN, 0-13-095085-8, London.

[Thome], Thome, B., 1991,"Systems engineering, principles and practice of computer-based systems engineering", Wiley , Wiley series in software based systems.

[Whitley] Whitley, K.N. & Blackwell, A.F. 1997. Visual programming: The outlook from academia and industry. In S. Wiedenbeck & J. Scholtz (Eds.), *Proceedings of the 7th Workshop on Empirical Studies of Programmers,* pp. 180-208.

[Zied et al], 2000, "The development of a robotic system for tool deployment in hazardous environment, ISARC 17, Taipei, Taiwan, p179-184.