End User Robot Programming via Visual Languages

James P Diprose
Department of Computer Science
University of Auckland
Auckland, New Zealand
jdip004@aucklanduni.ac.nz

Abstract—**The world has an increasing population of robots whose end users could benefit from being able to give them new tasks. Visual languages are a possible medium to accomplish this. We have taken a first step towards this through the realisation of Ruru, a visual language that enables novice programmers to create simple robot behaviours. It also addresses some inherent issues associated with robot software development. We plan to explore other domains, such as healthcare and agriculture, to facilitate the development of an end user robot programming language that can express more realistic real world robot tasks.**

Keywords-end user robot programming; visual language; physics of notations; domain experts;

I. Introduction

The International Federation of Robotics (IFR) estimates that at the end of 2009 the world's robot population totaled 8.7 million units and grow by 11.4 million units by the end of 2013 [2]. The creators of this vast population of robots could not possibly have predicted and therefore programmed every task that end users will want their robots to perform. There are also many diverse domains where autonomous, mobile robots are not utilized well yet e.g. healthcare and agriculture. However, end users in these domains may wish to take advantage of autonomous mobile robots in the future. Both cases involve the creation of new robot behaviors that accomplish end user needs e.g. a farmer might want a robot to find and spray the weeds on his farm so that he does not have to do it himself. Ideally, end users should be able to create new robot behaviors by themselves, or at the very least, they should be able to communicate better with programmers when new robot behaviors are being created e.g. by using a programming language that end users can read and understand [14].

Visual languages (VL's) are a possible solution to this problem. They use graphical symbols rather than text to represent semantic constructs and have advantages that make them suitable for end user robot programming (EURP). Firstly, they make the task of programming accessible to a wider audience than professional programmers [3], they increase the speed of programming [3], the use of graphics can afford more meaning than text [4] and they can aid comprehension and recall [4]. There are two main issues that arise when designing a VL; choosing the right semantic constructs to represent and deciding what the VL's visual notation should like.

The semantic constructs and relationships from the end user robotics domain can be acquired using an iterative development cycle based on the *Three Examples Pattern* [7]. In the context of EURP VL's this involves creating meta-models for three different representative robotics domains including; novice robot programming (see Section II), healthcare, and agriculture. Robots are currently being applied as healthcare assistants by a cross disciplinary team of researchers at the University of Auckland [15]. For instance, a robot might visit a number of patients throughout the day reminding them to take their medication [15]. This is an ideal domain for research into EURP as the Psychologists on the team are interested in specifying robot behavior. Agriculture is also an appropriate domain to research real world EURP because farmers could make use of robots that perform repetitive real world tasks on their farms, such as finding and spraying weeds. The common abstractions that are found between the meta-models of novice programming, healthcare and agriculture are the semantic concepts that should be represented in the final meta-model [7].

Deciding what the VL should look like can be informed by tools such as Moody's Physics of Notations (PON), which is a set of principles for VL notational design [8]. This enables the notational designer to give an explicit design rationale for a visual notation, transforming notational design from a craft into a design discipline [8].

II. Ruru

There is much existing work on VL's for robotics and novice programming. We reviewed nine of these VL's using Moody's PON and found that they had a number of deficiencies [6]. Additionally, the robotics related VL's fail to address problems associated with robot development as shown in Figure 1. This was the motivation for our own robot VL, Ruru, which is targeted at novice robot programmers (a type of end user) [6].

Ruru was designed as a first example in the approach mentioned in the introduction. Its semantic constructs were derived by finding common abstraction between typical novice robot programs (extracted from the book Learning computing with Robots) [9], robot program examples included in the Player Project [10], the Player robotics API [10] and robotics related literature [11] & [12]. The design of Ruru's visual notation was heavily influenced by Moody's PON principles [6].

Ruru's visual notation also addresses some of the problems associated with robot software development (Figure 1). Firstly, Ruru is live i.e. its visualisations are animated in real time and can be edited in real time while the robot is operating. This addresses the problem of the robot operating in a real time, dynamic environment. Second, the visual representations of robot sensors in Ruru are based on the physical forms of the underlying data being represented [13]. This addresses problems associated with the robot operating in 3D space. Lastly, in contrast to other VL's designed for multiple robots, e.g. Microsoft Visual Programming Language (MVPL) and the LabVIEW robotics module, Ruru abstracts away differences between instances of particular robot sensors e.g. different laser range finder models. This addresses some of the problems associated with the heterogeneity of the robot input devices.

The nature of the robot environment:

- The environment is dynamic, asynchronous and real time.
- Unexpected environmental variations cause non repeatable behavior.

The nature of the robot being programmed:

- Robots are mobile – they move away from the programmer.
- Robot hardware is heterogeneous i.e. there is a large and increasing number of input, output and storage devices.
- There is a lack of standardization of robot hardware and software interfaces.

The nature of mobile robot tasks:

- Robot tasks emphasize geometry and 3D space.
- Some tasks will be disrupted if they are interrupted e.g. a robot carrying a heavy object might drop it if the program controlling it is stopped for inspection.
- The programmer has to manage parallel unrelated activities occurring on multiple inputs and outputs.

Figure 1. Challenges robot programmers face. Adapted from [1] & [5].

We performed a preliminary qualitative user study on the efficacy of Ruru [6]. The participants considered themselves novice programmers and were from fields other than computer science and engineering. They found it motivating and understandable, indicating it is a good bridge for them into an understanding of computational concepts [6].

III. WHY MORE WORK IS NEEDED

Ruru can only express robot programs that are appropriate for novice robot programmers. Examples include those enabling a robot to: avoid obstacles, follow a colored object and follow/avoid light. Additionally, emphasis is placed on illustrating the elementary concepts behind these programs as opposed to making them "robust". The requirements of end users in real world domains are more demanding. They need their robots to accomplish more meaningful tasks and do them robustly e.g. spraying weeds autonomously and reliably. Other robot VL's potentially have the semantic expressiveness to create real world end user robot programs e.g. MVPL and LabVIEW robotics module. However, they do not address the challenges robot programmers face (Figure 1) and they have deficiencies in their visual notations [6]. Although Ruru has been explicitly designed to solve some of the robot programming problems illustrated in Figure 1, it does not attempt solve them all. This makes is why new tools for real world EURP need to be created.

IV. NEXT STEPS

My own and others existing work has raised some questions within the larger research question: "how should end users program robots?" In particular:

- What types of robot programs do real world end users want to construct?
- What would a stable robot domain model look like [1]?
- How can the problems associated with robot programming be fully addressed in an EURP VL environment?
- How can an appropriate level of semantic expressiveness be maintained in an EURP VL whilst still having a cognitively manageable notation?
- Would the EURP VL need to be supplemented with other techniques in order to maintain semantic expressiveness and cognitive manageability? E.g. using robot programming by demonstration.
- What is an appropriate visual representation for an EURP VL?

REFERENCES

[1] D. Brugali, A. Agah, B. MacDonald, I. A. D. Nesnas and W. D. Smart, "Trends in Robot Software Domain Engineering," in *Software Engineering for Experimental Robotics*, D. Brugali, Ed. Berlin: Springer-Verlag, pp. 3–8, 2007.

[2] International Federation of Robotics. "Executive Summary of World Robotics 2010 Service Robots," IFR Statistical Department, 2010.

[3] M. Burnett, "Visual Programming," *Encyclopaedia of Electrical and Electronics Engineering*, Vol. 4, pp. 275-283, New York: John Wiley & Sons Inc, 1999.

[4] N. C. Shu, "Visual programming: Perspectives and approaches," *IBM Systems Journal*, vol. 28, no. 4, pp. 199-221, 1999.

[5] B. MacDonald, G. Biggs and T. Collett, "Software Environments f*or Robot Programming," in Software Engineering for Experimental Robotics*, D. Brugali, Ed. Berlin: Springer-Verlag, 2007, pp. 107–124.

[6] J. Diprose, B. MacDonald, J. Hosking, "Ruru: A spatial and interactive visual programming language for novice robot programming," in Proc VLHCC 2011, In Press.

[7] D. Roberts, and R. Johnson, "Evolving frameworks: A pattern language for developing frameworks," *in Pattern Languages of Program Design 3*, D. Reiehle, F. Buschmann, & R.C Martin, Eds. Addison-Wesley, 1997.

[8] D. Moody. "The "Physics" of Notations: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering," *IEEE TSE*., vol 35, no. 6, pp. 756-779, December 2009.

[9] D. Kumar, Eds., *Learning Computing With Robots*. Institute for Personal Robots in Education, 2009.

[10] Player Project. "Player Project," Retrieved March 2011, http://playerstage.sourceforge.net/.

[11] D. Brageul, "An Intuitive Interface for Programming by Demonstration," M.S. thesis, Department of Computer and Electrical Engineering, University of Auckland, Auckland, New Zealand, 2008.

[12] T. Collett, "Augmented reality visualisation for player," Ph.D. thesis, Department of Computer and Electrical Engineering, University of Auckland, Auckland, NZ, 2007.

[13] T. H. J. Collet, B. A. MacDonald, "An Augmented Reality Debugging System for Mobile Robot Software Engineers," *Journal of Software Engineering for Robotics*, vol. 1, no. 1, pp 18-32, Jan 2010.

[14] M. Fowler, R. Parsons, "Domain-Specific Languages," Boston: Pearson Education, Sep 2010.

[15] E. Broadbent, R. Tamagawa., N. Kerse, B. Knock, A. Patience, B. MacDonald, Uniservices. "Retirement home staff and residents' preferences for heathcare robots" In Proc. IEEE Symposium on Robot and Human Interactive Communication 2009, pp 645 – 650.