# Convergence of Programming Development Tools for Autonomous Mobile Research Robots

**József Tick**

Institute for Software Engineering, John von Neumann Faculty of Informatics, Budapest Tech
Bécsi út 96/B, H-1034 Budapest, Hungary
tick@bmf.hu

*Abstract: This paper examines the general services and features of computing development environments of autonomous mobile research robots. It outlines the levels of software development support, the parameters of accessible development systems and the convergence of these systems to a standard solution. This paper presents a possible leading development environment of the future, the Microsoft Robotics Studio (MRS). It also examines the advantages and disadvantages of MRS and the specifications of its application.*

*Keywords: robotics, robot controlsystem, programming development tools*

## 1 Introduction

Robots are applied in a wide range of professional, educational and research fields. This paper examines the group of autonomous robots in the family of mobile ones focusing on a special application of research mobile robots. The usage of robots for research and educational purposes has a lot of advantages. It is especially true in the field of education. The positive feedback, seeing the robots running after long hours of programming, definitely raises motivation among students to a large extent. In this paper such robots are scrutinized that are only used for research, educational and development purposes and not applied for industrial purposes.

Common features of these robots are the following:

- not too large size, in general the diagonal is between 5 and 50 cms so does the height,
- relatively simple mechanical structure enabling mobility to a high extent,

- capable of computer capacity ranging from the average to a specially well developed one (from microprocessor control to on-board ITX),
- good software system facilities, adequate on-board functionality,
- good development environment that can be run on host computers,
- good communication solution, usually wireless communication with the partner machines (other robots, host computers),
- relatively well equipped robots with sensors (collision sensor, distance sensor, cameras),
- poorly equipped with manipulator (in this category the manipulator is usually not included),
- autonomous operation enabling two to four hour free movements without extra charging.

There are several examples for autonomous mobile research robots. Some typical examples are listed below without attempting to cover the total set:

One extreme example of these robots is the '**Alice**' micro robot developed according to Swiss watch making tradition. It is a 22*21*20 mm sized robot that weighs 11 grams, consumes only 12-18mW and is capable to operate autonomously for about 10 hours. The hardware is based on RISC processor, it is programmed in C, it has 4 sensors and can be equipped with a camera [1]. Processing can be done in the host computer via infra and wireless communication, therefore the robot can be used successfully in research and education, especially in multi-agent robot applications, for example behaviour examination in case of many robots.

A good example in the group of middle category autonomous mobile research robots is the popular, widely used **Rug Warrior** developed by MIT Artificial Intelligence Laboratory. The robot is based on Motorola 68HC11 microprocessor and equipped with mechanical, photo and voice sensors. It communicates with the environment via standard interface [2]. It can be programmed in Interactive C (IC), which is supported by a compiler that assembles to P-Code running on the host machine. The on-board computer includes P-Code-Interpreter, which interprets and executes the programme.

The high level category robots include some PC compatible computer as their on-board computers. The built-in sensors are of several types, Bluetooth and WiFi solutions are general communication possibilities. The **PC-BOT-914** is a good example, which, apart from the above described function, is equipped with higher level functions like vision-based navigation, object recognition, speech synthesis and speech recognition. The robot is a robust, 55 cm tall robot capable of autonomous operation for three and a half hours.

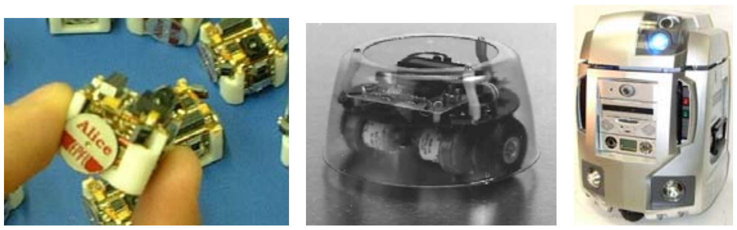

Figure 1
Robots described above: ALICE, Rug Warrior, PC-BOT-914

On the one hand, this paper examines the services of the developing systems enabling the development of the control system of these types of robots and, on the other hand, the convergence of these systems.

## 2 Programming Development Tools popular in Autonomous Mobile Research Robots

### 2.1 The Different Levels of Language Support

The software support of the control software development of these robots ranges on a large scale. The various levels of language support are presented in the following by giving typical examples.

#### 2.1.1 Low Level Language Support: Development by a Standard Programming Language

The lowest level of support for control software development of robots means the low level software development. In this case control system software is developed in a common language. The language does not include special instructions and the assembler does not have special directives. The only support means the various libraries, out of which different special subroutines, procedures and functions can be used. Simple development tools are available, which include a certain interpreter like C and a word processing environment like Programmers Notepad. Translation as well as linking result in immediately executable code, which can be downloaded to the on-board computer of the robot with simple software. The development languages used are those typical in embedded systems: ASSEMBLY, C, C++, JAVA, LISP. At this level the most popular language support is C.

### 2.1.2 Middle Level Language Support: Development by an Extended Standard Programming Language

In case of middle level language support the general programming language is supplemented by elements built in the language itself. The special additions support robot controlling, event-management, and task management and communication protocols. A good example of this category is Task Description Language (TDL) developed by Carnegie Mellon University (CMU) and supported by NASA. This language extends the standard C++ to be able to handle tasks (asynchronous constrained procedures). TDL provides syntactic support for hierarchical task-level control functions, including task decomposition, synchronization, execution monitoring, and also exception handling. The development of robot control programs is supported by TDLC, which is the TDL Compiler written in JAVA. TDLC translates TDL code to pure C++ code, which includes calls to a platform-independent task-control-management library [3]. TDL is successfully used for mobile robots, aircrafts and satellites controls.

### 2.1.3 High Level Language Support: Development by Special Robot Control Programming Language

The highest level of language support means that a language has been developed especially for controlling robots. Several languages have already been developed for a large number of applications. Each language supports the given application in a different way. Let us see some good examples of these languages without attempting to list all of them: ROLL (Robot Learning Language), CURL (Cambridge University Robot Language), RPL (Reactive Plan Language), RAP (Reactive Action Packages), ESL (Executive Support Language); the logic-based action language GOLOG, RoboML (Robotic Markup Language), XRCL (Extensible Robot Control Language), CCL (Computation and Control Language).

## 2.2 Support by Programming Development Environment

It seems more efficient if the development work is supported by a total development environment. There is a large number of these systems, each of them supports development in a different way. Some typical development environments will be presented below in order to describe the situation:

**MobilEye** is a good example of program development system with enhanced graphical interface enabling effective visualisation as well as supporting the easy planning of movements, tasks and routes of robots. Large and complex units are available in the process of control system planning, allowing easy and quick programme development. In this system automatic route calculations are supported to such a large extent that it is enough to click anywhere on the screen where the robot ought to go and the toolkit plans the route the robot should make, it downloads the instructions to the on-board computer of the robot and executes the task. It makes fast, easy and simple programme development possible in case of tasks based on visualisation.

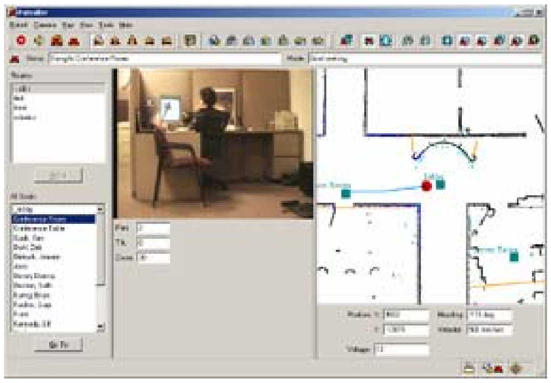

Figure 2
The Concept and the Graphical User Interface of the MobilEye Development System

**MARIE** is a quite different answer to the same problem. MARIE (Mobile and Autonomous Robotics Integration Environment) is an advanced program development environment allowing multiple applications, programs and tools, to effectively operate on one or multiple machines and work together on a mobile robot implementation. [5] MARIE's first goal is to create a programming environment at a system level, facilitating reusability of applications, tools and programming environments in an integrated and coherent system. [5]

**RobotFlow** is another development toolkit based on enhanced graphical visualisation. It is a mobile robotics toolkit based on the FlowDesigner project on the University of Sherbrooke. FlowDesigner is a data-flow oriented architecture, similar to Simulink (Matlab) or Labview. The visual programming interface provided in the FlowDesigner project will help people to better visualize & understand what is really happening in the robot's control loops, sensors, actuators, by using graphical probes and debugging in real-time. The very effective graphical interface of RoboFlow is shown in Figure 3.

The examples above well demonstrate that programme development systems for autonomous mobile research robot control have been developed using quite a number of approaches. However, a certain convergence can be noted in the evolution of these environments. In the following chapter these features will be examined.

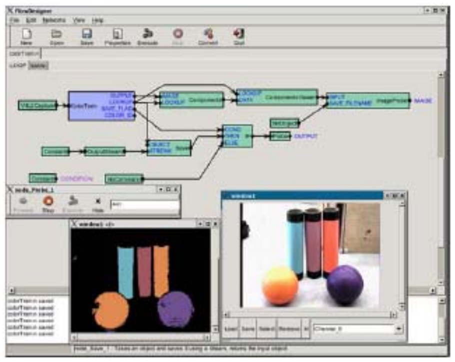

Figure 3
The concept and the Graphical User Interface of RoboFlow

## 3 The Convergence of Programming Development Tools for Autonomous Mobile Research Robots

The convergence of software systems is a general process as a result of development and improvement. Taking the process into account, the process can rather be called selection as used in biology instead of convergence. In the field of software development this applies to programming languages, operation systems and programming paradigms as well. The same process can be noted in case of programming languages, environments and systems supporting robot control system development. In the beginning, there were pioneers then such systems came out in bulk, each of them being specific, however, a kind of standardisation and convergence can be noticed by now.

### 3.1 Signs of Convergence

First of all, the integration of systems ought to be highlighted. The developers of each ‘more serious’ system strive to develop a consistent development environment by putting the tools on a unified platform, therefore making the development job much easier and much more comfortable.

An emphasis shift can be noticed towards the nice realisation of user interface. The application of Graphical User Interface has become typical exploiting its advantages at the same time. As a result, the systems have become not only user friendly but also the tasks could be defined much more easily and the executions could be put under control.

The exploitation of GUI to a larger extent has enabled the establishment of a certain visual control in case of several environments. In this case the visual denotation of the route is possible and the execution of the task as well as the feedback is done in a visual way, which makes the development job much easier for non computer experts as well.

An increase in the complexity of the systems can also be highlighted as a next common feature, which means partly an expansion in functionality and partly an increase in the level of services of certain functions.

It is unambiguous from the system development side that features of the development systems moved towards a general usage feature. As a result, control systems of several different robots can be developed on the same platform and/or the ready software products can be reused in case of turning to another robot or robot family.

In order to fulfil the requirements of the 21st century different development environments are getting integrated more and more into the operation systems of the client side, which, in most cases, are windows- or unix-based operation system.

As a common characteristic, the platform free feature ought to be highlighted i.e. defining a target-platform on which the development is made. This requires that on the one hand the on-board computers embedded in the robots accommodate downwards to the physical architecture of the robot, and, on the other hand, present a unified interface towards the development system.

## 3.2 The Microsoft Robotics Studio

The development of small- and middle-sized informatics systems in the last thirty years proved that if Microsoft enters a segment of the market, it strives to become a dominant party in that field. It can be seen for example in the operation system market: MS-DOS, Windows, and it can also be seen in case of the supplementary tools for operation systems: e.g. Office. This tendency can be noted in case of language development environments as well, e.g. Visual systems, .NET Studio, etc.

On 20 June, 2006 at RoboBusiness Conference and Exposition 2006, Microsoft showcased Microsoft Robotics Studio (MRS), a new Windows-based environment for academic, hobbyist and commercial developers to easily create robotic applications for a wide variety of computing platforms. Since June some companies, universities and research institutes adopted this platform to their products. Such as: Kuka, Robosoft, Whitebox Robotics, Coroware, etc.

MRS includes an effective visual programming tool, making it easy to develop and debug robot applications. It enables developers to generate modular services for hardware and software, allowing users to interact with robots through Web-based or Windows-based interfaces. Developers can also simulate robotic applications using realistic 3-D models. MRS provides a lightweight services-oriented runtime. Using a .NET-based concurrency library, it makes asynchronous application development simple.

MRS includes all the features that have been described earlier as the features of convergence. Its introduction to and proliferation in the market will definitely give an impetus and shows the future direction for Programming Development Tools for Autonomous Mobile Research Robots.

## Conclusions

The standardisation and convergence of Programming Development Tools for Autonomous Mobile Research Robots is a natural consequence of evolution. MRS could have an influence on and could determine the further development of these systems in the future and seems to be able to force programme developers to come out with competitive products with similar capabilities, capacities and facilities.

## References

[1] G. Caprari, P. Balmer, R. Piguet, R. Siegwart: The Autonomous Micro Robot ALICE: a Platform for Scientific and Commercial Applications. in Proceedings of the International Symposium on Micromechatronics and Human Science, Nagoya, Japan, November 25-28, 1998

[2] A. R. Flynn, J. Jones: Mobile Robots - From Inspiration to Implementation. A. K. Peters, Wellesley, Mass., 1993, ISBN 1568810113

[3] R. Simmons & D. Apfelbaum: A Task Description Language for Robot Control., in Proceedings of the Conference on Intelligent Robots and Systems, Vancouver, Canada, 1998

[4] D. J. Miller, R. C. Lennox: An Object-Oriented Environment for Robot System Architectures, IEEE International Conference on Robotics and Automation. Cincinatti, OH, USA, May 1990

[5] C. Cote, D. Létourneau, F. Michaud, J.-M. Valin, Y. Brosseau, C. Raievsky, M. Lemay, V. Tran: Code Reusability Tools for Programming Mobile Robots, in Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2004