Prolog-Based Real-Time Intelligent Control of the Hexor Mobile Robot*

Piotr Matyasik[1] and Grzegorz J. Nalepa[1] and Piotr Zięcik[1]

Institute of Automatics,
AGH University of Science and Technology,
Al. Mickiewicza 30, 30-059 Kraków, Poland
ptm@agh.edu.pl, gjn@agh.edu.pl, kosmo@agh.edu.pl

Abstract. The paper presents a concept of an intelligent control platform for the Hexor mobile robot, based on the XTT knowledge representation method for rule-based systems. The control systems is implemented in Prolog, with use of the Embedded Prolog Platform. The paper presents real-time control capabilities provided by this solution.

1 Introduction

The paper presents a research in the area of intelligent control of embedded real-time systems. A knowledge-based hierarchical control platform has been developed for the *Hexor* mobile robot. The platform combines a high-level control logic expressed with use of the XTT-based representation [1], and an embedded runtime environment using the *Embedded Prolog Platform* [2]. In the paper practical enhancements of the XTT towards effective control of reactive systems in real-time are proposed.

2 The Hexor Platform

HexorII is an autonomous 6-legged intelligent robot, developed by *Stenzel* (`www.stenzel.com.pl`) company as a didactic platform. It has a modular construction and can be easily extended by additional components (e.g. compass, laser sensors, etc.). The company provides a simple Basic-based software development environment. However, *HexorII* lacks some features needed for advanced control algorithm development. The transmission protocol implemented in Hexor does not allow to send data independently. Environmental information can only be pooled from the robot by the host software. Writing microcontroller software with a single control loop in Basic is easy. Unfortunately, this approach results in performance loss and *domino effect* while modifying subsystems.

Because of the software platform limitations, a new Hexor's internal controlling software architecture is proposed. Simple Basic program with one control loop was replaced by a real-time embedded operating system. High-level control logic is knowledge-based with use of the XTT representation.

* The paper is supported by the Hekate Project funded from 2007–2009 resources for science as a research project.

3 New Knowledge-Based Control Platform Architecture

Several possibilities where taken into consideration while developing the new Hexor low-level software. The first one was a dedicated application running directly on hardware. This one was abandoned because of problems with controlling whole hardware with a complicated state machine for each task, running off an interrupt timer. Thus embedded operating system became an obvious choice.

New Hexor controlling software is written on top of FreeRTOS (`www.freertos.org`). It is an open source portable hard real-time operating system. It can run with small memory footprint (700 bytes for OS, depending on configuration). The new *HexorNG* software architecture (Fig. 1) features: a multi-layer, easy extensible design, an intelligent knowledge-based control, based on the XTT rule-based representation, ability to distribute computations between many processing units, reliable real-time operating system based hardware control.

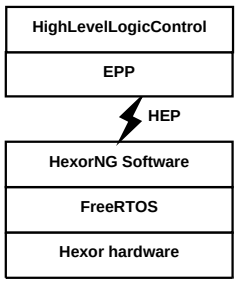

Fig. 1. HexorNG system architecture

HexorNG software substitutes all of the functions of the original one (low-level hardware control, movement execution, sensors reading, communication). The intelligent behavior of the controlled system can be described using a high-level, rule-based model. Following assumption are taken into consideration in constructing the knowledge base model: a top down approach, visual form, hierarchical structure, ability to verify system properties, high-level of abstraction from the hardware, distinguishing *time* as a special attribute for time-oriented verification and time-related behavior.

The knowledge base is described using the *XTT* (*eXtended Tabular Trees*) method ([1]). It provides implementation agnostic approach for rule-based systems [3,4], and allows for fast prototyping of the knowledge-based models with Prolog. A very important feature of the *XTT* method is the visual, hierarchical form of representation. XTT-based development allows for fast implementation of new, high level control algorithms. Moreover, with its ability to verify knowledge during the design phase, it allows for avoiding problems related to system completeness, determinism, or optimization. Development of knowledge-based control system in XTT-based environment is presented on Fig. 2. *XTT* representation allows for generating a Prolog-based control logic prototype. In order

to execute this logic, a Prolog interpreting environment must be provided. In this approach the *Embedded Prolog Platform* is used [2].

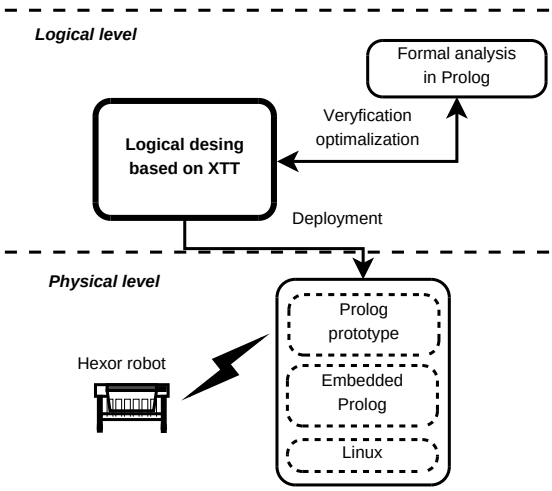

Fig. 2. XTT-based development cycle

4 XTT Enhancements and Real-Time Control Features

XTT lacks some features needed for control applications like event handling and interchanging information with controlled object. Originally *XTT* execution starts from a single entrance point. Handling multiple events forced to change this feature. Below an extension to *XTT* is introduced based on the following postulates: every event is represented as *tabular tree*, every event is processed separately but may share attributes with the others, events can be of two kinds: asynchronous (coming from environment, for example if sensor finds obstacle), and synchronous (generated by the timers). Events are executed according to the assigned priorities, events with higher priorities are executed before those with lover ones, events with the same priorities are queued with FIFO policy.

Extended graphical notation is presented on Fig. 3. Program consist of two kinds of *XTT* table sets. The first one is responsible for handling the timed events (above). It is represented by marking a starting point with a clock icon. After the slash in event name a priority is specified. Second one handles the asynchronous events. Starting points of those events are marked with an arrow and also contain name and priority information.

Embedded Prolog Platform (EPP) [2] introduces real-time capabilities and hardware drivers for Prolog. *EPP* consists of a Prolog layer, with an interpreter for executing control logic code, a supervising middleware, for connecting the interpreter with the operating system, and an operating system itself, general

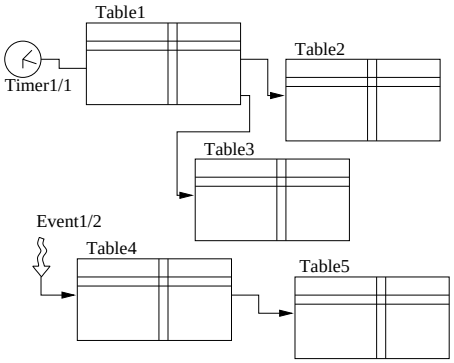

Fig. 3. XTT event handling control logic view

purpose or embedded for interfacing hardware, possibly with real-time extensions. EPP it implements events as asynchronous calls to Prolog, that can be used in similar way as the interrupts. But events in EPP can carry data, and provide bi-directional communication with Prolog.

5 Future Work

The research presented in this paper should be considered a work in progress. Future work will be concentrated on improving the visual representation of the knowledge in editor, and more EPP integration. The original contribution of this paper includes the concept of using an embedded Prolog-based logic for real-time control of reactive systems, and the practical enhancements of the XTT knowledge representation method towards effective control of such systems.

References

1. Nalepa, G.J., Ligęza, A.: A graphical tabular model for rule-based logic programming and verification. Systems Science **31**(2) (2005) 89–95
2. Nalepa, G.J., Zięcik, P.: Integrated embedded prolog platform for rule-based control systems. In Napieralski, A., ed.: MIXDES 2006 : MIXed DESign of integrated circuits and systems : proceedings of the international conference : Gdynia, Poland 22–24 June 2006, Łodź, Technical University Lodz. Department of Microelectronics and Computer Science (2006) 716–721
3. Liebowitz, J., ed.: The Handbook of Applied Expert Systems. CRC Press, Boca Raton (1998)
4. Ligęza, A.: Logical Foundations for Rule-Based Systems. Springer-Verlag, Berlin, Heidelberg (2006)