Chapter 5
A Generic Guide for Using Membrane Computing in Robot Control

ABSTRACT

As various theoretical and practical details of using membrane computing models have been presented throughout the book, certain details might be hard to find at a later time. For this reason, this chapter provides the reader with a set of checkmark topics that a developer should address in order to implement a robot controller using a membrane computing model. The topics discussed address areas such as: (1) robot complexity, (2) number of robots, (3) task complexity, (4) simulation versus real world execution, (5) sequential versus parallel implementations. This chapter concludes with an overview of future research directions. These directions offer possible solutions for several important concerns: the development of complex generic algorithms that use a high level of abstraction, the design of swarm algorithms using a top-down (swarm-level) approach and ensuring the predictability of a controller by using concepts such as those used in real-time operating systems.

INTRODUCTION

The use of membrane computing in robot control has gained momentum since the introduction of membrane controllers in (Buiu, Vasile, & Arsene, 2012). New models of P systems amenable to robot control, such as enzymatic P systems, have been proposed. New applications have been approached, such as the control of collective robot systems, including swarms of robots. Furthermore, these approaches have

DOI: 10.4018/978-1-5225-2280-5.ch005
been compared with conventional ones, such as finite-state machines, and some advantages and disadvantages of using membrane computing models in robot control, have been provided.

The aim of this chapter is to offer the curious robotics engineer or intrepid researcher interested in new ways to control robots a guide for selecting the proper membrane computing models and simulators, and what problems could they face in implementing these control models for real-life robots and applications.

This chapter will take the form of a set of guidelines organized along a number of important topics such as:

- Abstraction level;
- Robot complexity;
- The level of interaction between robots;
- Task complexity;
- The type of execution environment, simulated or real-world;
- The level of parallelism.

The majority of the issues are presented in a comparative manner and represent separate concerns (a software engineering principle introduced in (Ghezzi, Jazayeri, & Mandrioli, 2002)). These separate concerns are important in that a developer should take each one into account when intending to use membrane computing models for robot control.

The following discussion is generic in the sense that it is not centered around a particular scenario but rather tries to accommodate a membrane computing robot controller to different structural and functional requirements.

**ABSTRACTION LEVEL**

The first fact that must be considered when designing a robot controller is the target level of control or in other words, where will the controller be placed in the control application. This control level is proportional to the abstraction level in the sense that a high level decision application (path planner, role assignment, …) uses only abstract elements as opposed to an embedded device controller (driver) that uses mainly numerical values in order to interface with the hardware. Several example programs are presented graphically in Figure 1.

This type of stacked architecture is often used for operating systems where hardware devices (CPU, memory, …) reside at the bottom of the stack, followed by
Improving Dependability of Robotics Systems: Analysis of Sequence-Dependent Failures
www.igi-global.com/chapter/improving-dependability-of-robotics-systems/212062?camid=4v1a

Design of a Highly Dynamic Hydraulic Actuator for Active Damping Systems in Machine Tools
www.igi-global.com/article/design-highly-dynamic-hydraulic-actuator/74807?camid=4v1a