Chapter 16
The LPZROBOTS Simulator

Abstract: In this chapter we describe our robot simulator. We start with the overall structure of the software package containing the controller framework, the physics simulator and external tools. The controller framework makes it very easy to develop and test our algorithms, be it in simulations or with real robots. The physics simulator can handle rigid bodies with fixed geometric representation that are connected by actuated joints. Particular efforts have been undertaken to develop an elaborated treatment of physical object interactions including friction, elasticity, and slip. The chapter also briefly discusses the generation of virtual creatures, the user interface and the most important features of the LPZROBOTS simulation environment.

Realistic computer simulations are very important not only for experimental scientists but also for theoretic studies. They allow one to quickly check hypotheses and algorithms and verify generalizations and approximations that have been done in the course of analytical derivations. This is especially fitting for robotics, where the hardware is normally error-prone and requires rather intense maintenance. However, many argue that robot experiments must be performed with real robots only. This harsh opinion is rooted in the fact that software controllers tested in simulations often have not been able to reproduce the same results in reality. In our case we have a different setting. We do not develop static controllers, but instead aim at the self-organizing of behavior. If self-organization works with complex simulated systems then it will also work in real systems. The control algorithms adapt to the specific hardware during the behavior. In this way the behavior will be quantitatively different on a real robot but the qualitative properties will be the same. Fortunately, the gap between reality and simulation is also shrinking because we can nowadays perform physically realistic computer simulations, as you have seen in the numerous videos and experiments so far.

We developed a simulation software called LPZROBOTS that comes with a realistic physics simulation and is particularly suitable for our research. There are several distinct features as we will discover below. One of them is the support for custom materials and their proper interaction. Let us now take a closer look at the back-stage of our virtual world of playful machines. In the next section we focus on the overall structure of the simulator. Afterwards the framework for developing con-
controllers is introduced, which is independent of the physics simulator that is described in Sect. 16.4. It follows a list of highlighted features (Sect. 16.5). A comprehensive documentation with technical details and the source code is available on the project website [107].

16.1 Structure

The simulation software is divided into the following parts:

**Controller framework:** Implementation of the controllers, neural networks, matrix library, introspection, and support functions (SELFORG).

**Physics simulator:** Rigid body dynamics and graphical rendering (ODEROBOTS).

**External tools:** Visualization tools GUILOGGER and MATRIXVIZ.

The control algorithms are located in an independent package (SELFORG) that can be used in other simulators or with hardware robots. For the development of our algorithms it is important to be able to observe the evolution of internal parameters online and to change some control parameters like learning rates during the runtime. For that reason the framework allows for quick controller development and their flexible connection to robotic systems as well as the ability to connect to external tools. The next section will introduce this framework. Since the software is written in C++ we used the concepts of object-oriented programming. To understand the following text only a few terms need to be known, such as *class* that refers to an object type, *interfaces* which is an abstract object type to specify only the signature (available functions), and *subclass* or *inheritance* for the mechanism to define a more specific class based on an existing one.

16.2 Controller Framework

The SELFORG framework is designed for connecting a controller to any system, be it a real robot, a simple academic program, or our full-fledged robot simulator. A controller is a class that has essentially a single function that receives a vector of sensor values and returns a vector of motor values. The most important part in the framework is the *wired controller*, consisting of a controller and a *wiring* paired with some utilities to log, plot and configure the system. The wiring allows for the preprocessing of sensor and motor values, making the connection to different systems very easy. For example a subset of sensor values can be selected or the derivative of a sensor value can be added. Likewise the motor values can be preprocessed, e.g. to protect the robot hardware from dangerous configurations. The

---

1 The modification of parameters is only necessary during the test phase. In the robot experiments we report on the parameters are not changed manually, except states otherwise.