ELSEVIER

Neurocomputing 44–46 (2002) 1043 – 1048

NEUROCOMPUTING

www.elsevier.com/locate/neucom

IQR: a distributed system for real-time real-world neuronal simulation

Ulysses Bernardet[1], Mark Blanchard[1], Paul F.M.J. Verschure*[,1]

Institute of Neuroinformatics, ETH-University Zurich, Winterthurerstrasse 190, CH-8057 Zurich, Switzerland

Abstract

IQR is a new simulator which allows neuronal models to control the behaviour of real-world devices in real-time. Data from several levels of description can be combined. IQR uses a distributed architecture to provide real-time processing. We present the key features of IQR and highlight successful projects which have used this simulator. © 2002 Elsevier Science B.V. All rights reserved.

Keywords: Simulator; Real-time; Behaviour; Neural model; Distributed computing

1. Introduction

The brain is organised at multiple levels ranging from single cells, to circuits and systems giving rise to behaviour. To understand these multiple levels of neural organisation and their interactions is at the core of current research. However, to directly measure and manipulate all different elements of complex neuronal systems using standard experimental techniques is exceedingly difficult. Hence, simulation techniques are useful to fill this gap in our study of the nervous system.

Multi-level simulations have the advantage that they circumvent the fundamental indeterminacy problem confronted by any theoretical study of an input–output system. When we conceptualise a modelling study as the attempt to fit a curve to a set of data points it becomes clear that many curves might provide a possible fit.

* Corresponding author.

E-mail addresses: ulysses@ini.phys.ethz.ch (U. Bernardet), jmb@ini.phys.ethz.ch (M. Blanchard), pfmjv@ini.phys.ethz.ch (P.F.M.J. Verschure).

[1] Supported by SPP, Swiss National Science Foundation.

PII: S0925-2312(02)00412-5

Hence, the modeller confronts the problem of making sure that the particular solution chosen is unique. We have argued that by including multiple levels of description in simulation studies, the problem of indeterminacy can be effectively addressed [6].

This method of convergent validation, however, requires dedicated software tools that allow the simulation of neuronal systems at multiple levels including the real-time control of behaving systems. Hence, we have developed IQR [5]. IQR allows a wide variety of simulations, ranging from large-scale neuronal models controlling real-world artifacts, i.e. robots, to more detailed biophysical models of neuronal circuits. This paper shortly describes some of the key elements of IQR and provides some examples of its application.

2. Features of IQR

2.1. Real-time processing

In order to control the behaviour of a real-world device, simulations must run in real-time. IQR meets this requirement by utilising the distributed architecture shown in Fig. 1. Models are divided into *processes*, which represent different functional units (e.g. vision, audition, motor control). These processes are assigned to different host computers according to either their device requirements or computational complexity, and communicate via Ethernet. Furthermore this design allows complex simulations to be performed using low-cost hardware.

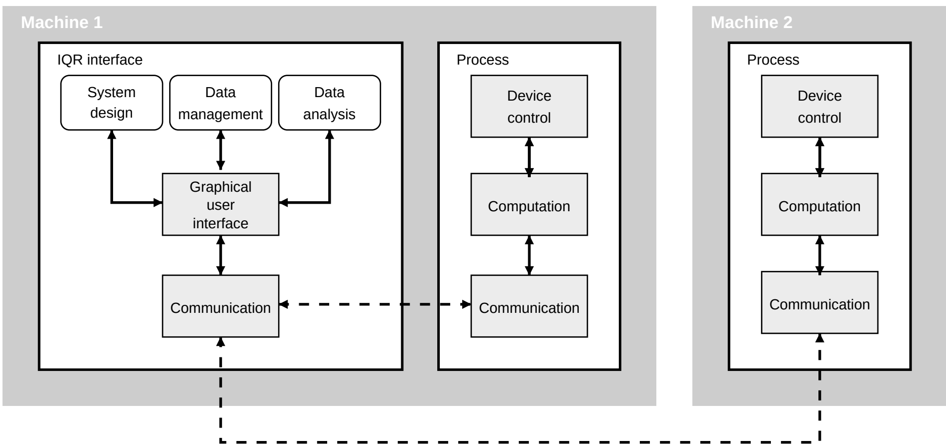

Fig. 1. Summary of the distributed architecture of IQR. Two processes are shown, which comprise computation, communication, and device I/O. The first process is shown running on the same machine as the user interface, the second runs on a remote machine via Ethernet. The two processes might represent, for example, the motor control of a mobile robot and visual processing of the view seen by a robot-mounted video camera.

2.2. Modelling language

Each *process* is defined by a neuronal circuit which comprises *groups* of neurons and the *connections* between these *groups*. Each *group* is an array of neurons of the same type. The model neurons provided by IQR are commonly used abstract cell types such as integrate-and-fire and linear threshold. These simple models offer the best compromise between accuracy and computation complexity, in keeping with the need to run in real-time. A range of topographic *connection* types are provided, and an interface is provided for specifying other non-uniform connection patterns (e.g. point-to-point, region-to-region).

2.3. Interfaces to external devices

IQR models are interfaced to external devices by specifying mappings between the state of *groups* of neurons and device-specific variables. For input, the value of an external sensor (e.g. a video camera) is mapped onto the state of a group of neurons; for output, the state of a group is used to set the value of a control parameter of the effector (e.g. the speed of a mobile robot). Predefined interfaces are provided for video cameras, microphones and mobile robots.

2.4. Extensibility

IQR can be extended by means of user defined modules, i.e. new neuron types and device interfaces. An interface is provided which allows the user to write the required routines in C++ and access these routines from within their models.

2.5. User interface

IQR has a graphical user interface which is used for model design, control of running simulations, and visualisation of the internal states of the model. Real-time displays include plots for whole groups, time traces for single cell states and group statistics, as well as online correlation and spectrum analysers. All model parameters can be changed on-the-fly and the effects of these changes seen immediately.

In addition, the interface offers a number of more advanced tools. The data manager allows the user to select states from any process for storage and allows this recorded data to be played back and analysed. For long experiments, the protocol manager allows a sequence of steps to be defined and executed automatically. For example, a systematic study of the effect of a parameter change on system behaviour can be performed, including the required data sampling and storage.

The user interface is decoupled from the computation, and can be run on a separate host computer to ensure that it has no impact on the overall system performance.

2.6. Implementation

IQR was originally developed for UNIX platforms using C and the Motif library [5]. At present, IQR is being redeveloped under Linux operating system using C++, XML and the cross-platform widget set Qt (TrollTech A.S., Oslo, Norway). This choice of technology will allow the new version of IQR to be made available for multiple operating systems.

3. Example projects

IQR has already been used successfully in a number of projects which range from abstract neuronal models of learning and problem solving applied to robots, to models that include a high-degree of biophysical detail. Below, we describe one model in detail in order to illustrate the IQR modelling approach. Other projects are described briefly.

- We are using IQR in a study of the properties of the locust lobula giant movement detector (LGMD) system. The LGMD responds selectively to objects which approach the animal, and behavioural experiments have shown that locusts react to approaching objects with escape jumps or avoidance steering during flight. It is thought that these reactions are triggered by the responses of the LGMD. We use IQR and a mobile robot equipped with a video camera to study this system. The IQR model used for these experiments (Fig. 2) comprises three processes. The

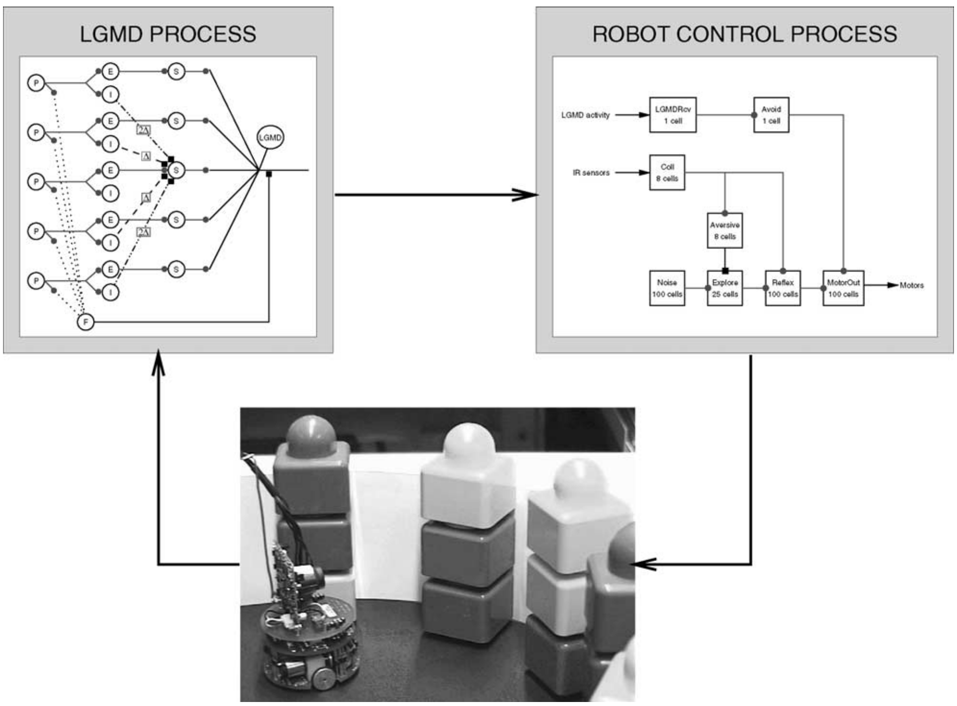

Fig. 2. IQR system used in LGMD experiments. The LGMD model was broken into two IQR processes. The first process received input from the robot-mounted video camera and processed this input using a model of the LGMD system. The LGMD responses from this process were passed to the second process, which controlled the robot. If the spike rate from the LGMD was high, this process triggered escape reactions, allowing the robot to avoid collisions with the surrounding obstacles.

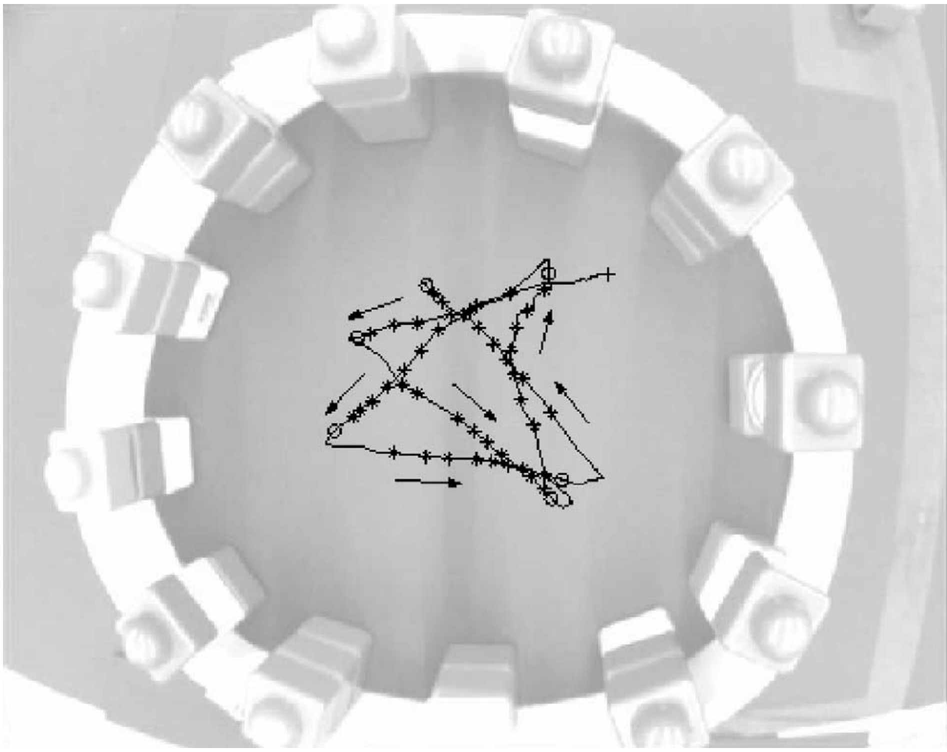

Fig. 3. Responses of LGMD model can be combined with tracking data during experiments, allowing direct comparison between the neuronal dynamics and the behaviour of the robot. Open circles indicate LGMD spikes, the line indicates the path of the robot.

first process passes the video input through a model of the LGMD system. The spike train from the model LGMD is passed to a second process, which controls the robot. When the LGMD spike rate is high, which indicates an approaching obstacle, an avoidance reaction is triggered and the robot turns away from the obstacle. A third process is used to track the position of the robot using an overhead camera.

IQR allows the states of these three processes to be monitored simultaneously, allowing direct comparisons to be made between the responses of the neuronal model and the behaviour of the robot. Fig. 3 shows the relationship between the responses of the LGMD and the position of the robot during an experiment. Our experiments have shown that the responses of the LGMD can trigger escape reactions reliably [1,2]

- The Distributed Adaptive Control (DAC) series of models [9] were built using IQR. In these models of learning and memory, a mobile robot formed associations about colour patterns during exploration of an arena. The memories formed allowed the robot to navigate to target locations reliably.

- IQR has also been used for models of classical conditioning which include a model of learning in the auditory cortex [4]. This model used the temporal relation between pre- and postsynaptic action potentials to adjust synaptic weights and was tested using a range of sound sources via a microphone input. In addition, a model of the cerebellar circuitry underlying classical conditioning was tested using a mobile robot [8].
- IQR has also been applied to models of the mammalian olfactory bulb in combination with an artificial nose [3] and to the dynamic modulation of dendritic integration properties in the binding and segmentation of visual scenes in a model of the primary visual cortex [7].

4. Summary

We have presented IQR, a new simulator which offers a new approach to modelling neuronal systems. Data from different levels of description can be combined, and the behaviour produced by models can be studied in the real-world. The value of this style of modelling has already been demonstrated in a number of successful projects.

References

[1] M. Blanchard, F.C. Rind, P.F.M.J. Verschure, Collision avoidance using a model of the locust LGMD neuron, Robotics Autonomous Systems 30 (1–2) (2000) 17–38.

[2] M. Blanchard, F.C. Rind, P.F.M.J. Verschure, How accurate need sensory coding be for behaviour? Experiments using a mobile robot, Neurocomputing 38–40 (2001) 1113–1119.

[3] T.C. Pearce, P.F.M.J. Verschure, J. White, J.S. Kauer, Stimulus encoding in the early stages of olfactory processing: a modeling study using an artificial olfactory system, Neurocomputing 38–40 (2001) 299–306.

[4] M.A. Sanchez-Montanes, P. König, P.F.M.J. Verschure, Learning sensory maps with real-world stimuli in real time using a biophysically realistic learning rule, IEEE Trans. Neural Networks, in press.

[5] P.F.M.J. Verschure, Xmorph: a software tool for the synthesis and analysis of neural systems, Technical Report, Institute of Neuroinformatics, June 1997.

[6] P.F.M.J. Verschure, Synthetic epistemology: the acquisition, retention, and expression of knowledge in natural and synthetic systems, in: Proceedings of World Conference on Computational Intelligence 1998, Anchorage, IEEE Press, New York, 1998, pp.147–153.

[7] P.F.M.J. Verschure, P. König, On the role of biophysical properties of cortical neurons in binding and segmentation of visual scenes, Neural Comput. 11 (1999) 1113–1138.

[8] P.F.M.J. Verschure, M. Mintz, A real-time model of the cerebellar circuitry underlying classical conditioning: a combined simulation and robotics study, Neurocomputing 38–40 (2001) 1019–1024.

[9] P.F.M.J. Verschure, T. Voegtlin, A bottom up approach towards the acquisition and expression of sequential representations applied to a behaving real-world device: distributed adaptive control III, Neural Networks 11 (1998) 1531–1549.