Parallel implementation of a biologically inspired model of figure-ground segregation: Application to real-time data using MUSIC

Ali Nazem[1,2], Gert Kootstra[1], Danica Kragic[1], Mikael Djurfeldt[2,3]
[1] CVAP, CSC, KTH, 100 44 Stockholm, Sweden
[2] PDC, CSC, KTH, 100 44 Stockholm, Sweden
[3.] INCF, Karolinska Institutet, Nobels väg 15A, 171 77 Stockholm, Sweden
E-mail: djurfeldt@incf.org

Background

MUSIC, the multi-simulation coordinator, supports communication between neuronal-network simulators, or other (parallel) applications, running in a cluster super-computer [1,4]. Here, we have developed a class library that interfaces between MUSIC-enabled software and applications running on computers outside of the cluster. Specifically, we have used this component to interface the cameras of a robotic head to a neuronal-network simulation running on a Blue Gene/L supercomputer [2]. Additionally, we have developed a parallel implementation of a model for figure ground segregation based on neuronal activity in the Macaque visual cortex [3]. The interface enables the figure ground segregation application to receive real-world images in real-time from the robot. Moreover, it enables the robot to be controlled by the neuronal network.

Methods

I. A special purpose TCP/IP based communication interface has been implemented in C++ as an extendible class library. The architecture of the interface is shown in Figure 1. The client end of the interface is specifically designed to meet the requirements for operating on the Blue Gene /L, as well as being MUSIC-aware. The server side component resides in the outside world providing the client with streaming real-time data. We designed an inheritable class called CSerializable that defines the unit of data and marshals the data across the route from the source to the final destination in the parallel application. Based on CSerializable, we implemented the entities CCommand and CRawImage which are required for transmission of control commands and images, respectively, between the parallel application and the robot (Figure 1). A single process application called MusicGate, connects the client side of the interface and the MUSIC-enabled component together. The parallel application sends a command to the server requesting a stream of images. As soon as one frame of the data sent from the server is available in MusicGate, it will be directly transferred from the read buffer to the parallel application by the MUSIC library, hence avoiding redundant internal memory operations. The implemented architecture performs the communication IO operations in parallel with the neural processing in order to minimize the idle time in compute nodes. The idle time is a function of communication latency, and the processing load of the neuronal-network simulation.

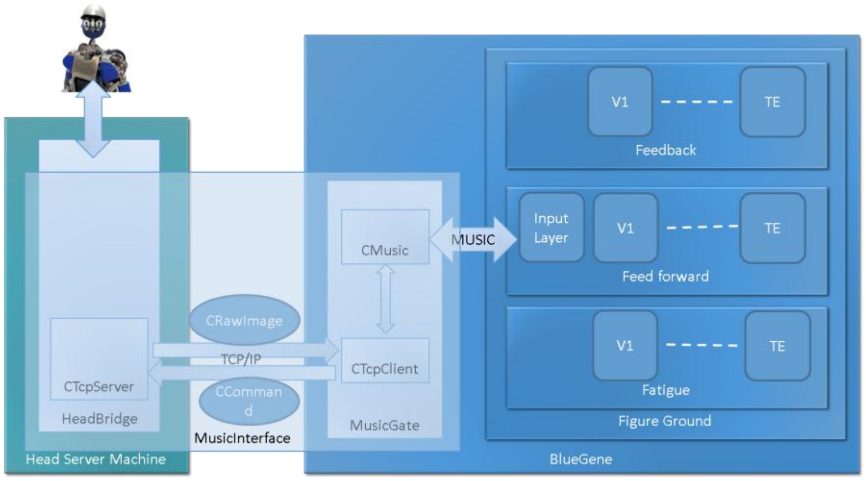

Figure 1 - The MUSIC interface architecture.

II. Having the communication interface, we developed a parallel implementation of a model for figure-ground segregation in the form of a recurrent neuronal network. The implementation defines structures such as neurons, neuronal layers, and neuronal networks in a way suitable for a parallel platform. Each processor is carrying the computational load of a number of similar neurons. We call this structure a *tile* of neurons. Each neuronal layer contains a number of tiles.

Results

We have successfully implemented an interface between a neuronal-network simulator running on a parallel platform and a robot in the real world. We have also developed a parallel implementation for a figure ground segregation model based on a data from Macaque visual cortex. The latency and transfer rate of the entire model makes real-time figure-ground segregation possible. Since the client side of the interface provides a standard MUSIC port towards the parallel application, it can also be used to connect the robotic head to a generic MUSIC-enabled neuronal-network simulator. We demonstrate this for the NEST simulator [4].

References

1. Örjan Ekeberg, and Mikael Djurfeldt. **MUSIC - Multi-Simulation Coordinator User's Manual.** 2009.
2. Gary L. Mullen-Schultz. **Blue Gene/L: Application Development**. *IBM*, 2006.
3. Pieter R. Roelfsema, Victor A. F. Lamme, Henk Spekreijse, and Holger Bosch. **Figure–Ground Segregation in a Recurrent Network Architecture.** *Journal of Cognitive Neuroscience* 2002, **14:4** 525-537.
4. Mikael Djurfeldt, Johannes Hjorth, Jochen M. Eppler, Niraj Dudani, Moritz Helias, Tobias C. Potjans, Upinder S. Bhalla, Markus Diesmann, Jeanette Hellgren Kotaleski, and Örjan Ekeberg. **Run-Time Interoperability Between Neuronal Network Simulators Based on the MUSIC Framework.** *Neuroinform*, 2010.