30th Annual International IEEE EMBS Conference
Vancouver, British Columbia, Canada, August 20-24, 2008

# BMI Cyberworkstation: Enabling Dynamic Data-Driven Brain-Machine Interface Research through Cyberinfrastructure

Ming Zhao†, Prapaporn Rattanatamrong†, Jack DiGiovanna†, Babak Mahmoudi†, Renato J. Figueiredo††, Justin C. Sanchez††, José C. Príncipe‡, José A. B. Fortes‡
(† IEEE Student Member, †† IEEE Member, ‡ IEEE Fellow)

*Abstract*—**Dynamic data-driven brain-machine interfaces (DDDBMI) have great potential to advance the understanding of neural systems and improve the design of brain-inspired rehabilitative systems. This paper presents a novel cyberinfrastructure that couples *in vivo* neurophysiology experimentation with massive computational resources to provide seamless and efficient support of DDDBMI research. Closed-loop experiments can be conducted with in vivo data acquisition, reliable network transfer, parallel model computation, and real-time robot control. Behavioral experiments with live animals are supported with real-time guarantees. Offline studies can be performed with various configurations for extensive analysis and training. A Web-based portal is also provided to allow users to conveniently interact with the cyberinfrastructure, conducting both experimentation and analysis. New motor control models are developed based on this approach, which include recursive least square based (RLS) and reinforcement learning based (RLBMI) algorithms. The results from an online RLBMI experiment shows that the cyberinfrastructure can successfully support DDDBMI experiments and meet the desired real-time requirements.**

## I. Introduction

Brain-machine interfaces (BMIs) are key to helping paralyzed patients and others with motor disabilities regain autonomy by using brain signals to directly control prosthetic limbs. To realize this vision, computer systems are fundamental in understanding brain function and designing motor control through modeling. In a particularly important paradigm, many different BMI models can be executed concurrently based on dynamic brain signals and sensorial feedback, where their results are selected to adaptively control the motor in the optimal way [3]. This dynamic data-driven BMI (DDDBMI) computing system has great potential to advance the state of art of research on BMIs.

There are several critical challenges to building such a DDDBMI system. First, the concurrent execution of many BMI models requires a tremendous amount of computing power and storage capacity. Thus the system needs to efficiently aggregate resources for model computation and integrate with onsite signal sampling and robotic movement. Second, effective brain-inspired motor control has stringent timing requirement because of the need for low latency between brain signaling and sensory feedback. Hence, it is important for the system to provide the necessary guarantee for the timing from signal acquisition, modeling, to motor control. Last but not least, a successful DDDBMI system needs to provide seamless support for its users, the scientists from domains such as signal processing and neurophysiology. It is desirable that the system be able to hide its complexity and present easy-to-use interfaces for users to deploy their algorithms and conduct research.

This paper presents a novel cyberinfrastructure, named BMI Cyberworkstation, to address the above challenges and support advanced DDDBMI research. It consolidates the distributed computing, data, communication, and instrument resources, and allows BMI experiments to be conducted in a closed-loop manner, including data acquisition, model computation, and robot control. Computing resources are managed through virtualization by using virtual machines [5] to dynamically allocate resources and satisfy both resource demand and timing constraints. Virtual machines also allow for flexible customization of execution environments, which facilitates transparent deployment of BMI models. In addition, a Web-based portal is provided to help users interact with the cyberworkstation, through which they can conveniently control and monitor the experiments, as well as visualize and analyze the results.

New BMI models are developed and tested based on this cyberworkstation, including both supervised learning based (RLS [2]) and reinforcement learning based (RLBMI [1]) algorithms. The cyberworkstation allows for these studies in two broad scenarios: in the online scenario, real-time computation and control are provided for in vivo experiments; in the offline scenario, data from past experiments can be replayed with various configurations to analyze and train models. An online RLBMI experiment is also reported in this paper to demonstrate the effectiveness of the proposed cyberworkstation.

The rest of this paper is organized as follows: Section II presents the architecture of the cyberworkstation; Section III describes two BMI models supported by this approach; Section IV discusses an experimental evaluation and Section V concludes the paper.

Manuscript received April 16, 2008. This work is supported in part by the NSF under Grant No. CNS-0540304 and the BellSouth Foundation.

Ming Zhao, Prapaporn Rattanatamrong, Renato J. Figueiredo, José A. B. Fortes, and José C. Príncipe are with the Dept. of Electrical and Computer Engineering; Jack DiGiovanna and Babak Mahmoudi are with the Dept. of Biomedical Engineering; Justin C. Sanchez is with the Dept. of Pediatrics, University of Florida, Gainesville, Florida, USA. (e-mail: {mingzhao, rattanat, renato, fortes, principe, jfd134, babakm, jcs77}@ufl.edu)

## II. ARCHITECTURE

The BMI cyberworkstation consolidates the software and hardware resources across two collaborating labs (Neuroprosthetics Research Group, NRG and Advanced Computing and Information Systems Lab, ACIS) at the University of Florida (Fig. 1). The novel aspect of this system is its middleware which manages data, resources, and jobs to support DDDBMI experiments. During an online experiment, in vivo brain signals are acquired at NRG and sent across a campus network for model computation at ACIS; the results are fed back over network to control the robotic arm at NRG. Offline study takes the data acquired and stored during previous online experiments and conducts more extensive and time-consuming training and analysis for the models. The rest of this section describes how each phase of this closed loop is supported by the cyberworkstation in detail.

### *A. Data Acquisition*

During an online experiment, brain signals are sampled from a live animal through a multi-channel digital signal processing (DSP) device. It is directly connected to a computer server where the middleware polls for data from the DSP and sends them for model computation. The latency of acquiring data from multiple channels can be considerably high and may violate the timing constraint if the channels are polled sequentially from the DSP. In order to support real-time experiments, parallel polling is employed in which the multi-channel data are prepared in a buffer in the DSP and polled by the server with a single operation. In addition, the server is tuned to minimize interference with the data acquisition by removing unnecessary services and processes and giving the polling process the highest scheduling priority.

### *B. Network Transfer*

Data are transferred from NRG to ACIS over a campus network. Because of its unreliable and shared nature, data loss and unexpected delay can happen at any time during the communication. Reliable data transfer must be provided to overcome these problems and support reliable and real-time experiments. Using TCP protocol to transfer data can achieve reliability via transport-layer timeout and retransmission, but it is difficult to control these mechanisms with the desired recovery policies and timing constraints. Therefore, the cyberworkstation builds reliable data transfer upon the unreliable UDP protocol and delivers reliability at application-layer through the middleware.

The middleware on the data acquisition server monitors the elapsed time after sending out the data. A timeout happens when it detects that it is unlikely to get the computation results back in time to meet the deadline. It then stores this failed data sample in a circular first-in-first-out (FIFO) buffer, and starts a new closed-loop cycle by polling for the next sample. During the new cycle, it queues the newly acquired sample in the buffer and retries the transmission of the previously failed sample. The transmission can be retried in the following cycles in a similar way until it is successful or

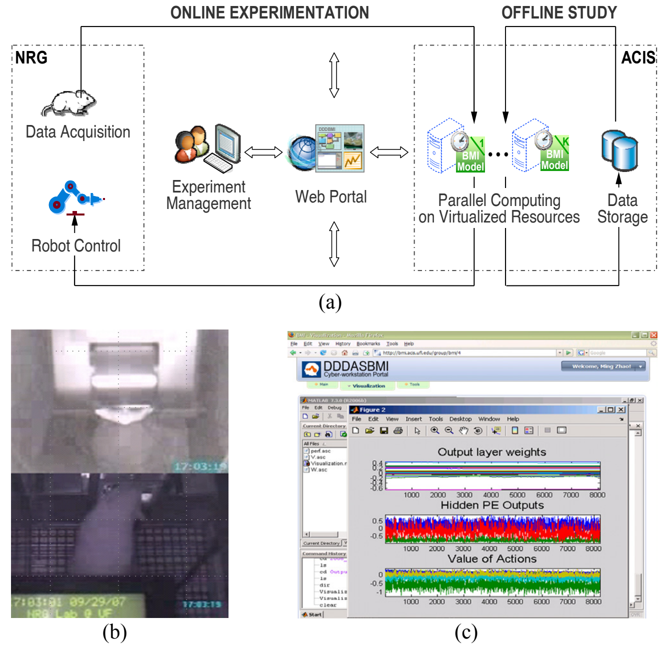


Fig. 1. (a) The architecture of the cyberworkstation which supports closed-loop online and offline experiments and provides a Web portal for experiment management. (b) A photo showing that a rat was controlling the movement of a robotic arm using its brain signals through the cyberworkstation. (c) A snapshot of the Web portal where results from an experiment were visualized and analyzed.

the failed sample is evicted from the buffer because of its limited size. A delay between neural activity and robot action will occur because of this buffering. However, the extent of this delay is decided by the size of the buffer, which is adjustable based on model requirement and user preference.

### *C. Model Computation*

In DDDBMI, brain signal data are used to drive different motor control models, each seeking the best control action independently, where their results are aggregated to find the optimal robotic movement. The cyberworkstation supports this paradigm by computing these models in parallel using the message passing interface (MPI).

This parallel computation is conducted upon resources provided through virtualization to provide efficient utilization of resources and seamless support of BMI models. Virtualization allows many virtual machines to be created upon a single physical machine and transparently share its computing and storage resources. Each virtual machine can be customized with the necessary execution environment, including operating system and libraries, to support seamless deployment of a BMI model. Multiple models can run concurrently with their dedicated virtual machines, where resources are dynamically provisioned to them according to their demands and timing requirements.

For offline BMI study, virtual machines allow the model computation to share resources with other jobs in an isolated manner. These jobs are managed via a cluster management system [6] that provides queuing and scheduling of job executions on the virtual machines. On the other hand, an online experiment has strict timing requirement for the model computation. Thus, a set of physical resources is reserved in advance to prepare a cluster of virtual machines for the online computation. The existing jobs on these reserved resources can be transparently suspended or relocated to other resources by suspending or migrating their VMs [4].

### *D. Robot Control*

The results from the parallel model computation are sent back from ACIS to NRG across the network, again through the aforementioned reliable data transfer mechanism, to move the robotic arm in real-time. The robotic arm is also directly connected to the data acquisition server and controlled by the middleware. Two types of control are supported to meet different model needs. Incremental control moves the arm one step at a time, where each movement has to finish within the closed-loop cycle. With point-to-point control, the arm continuously travels a relatively long distance, which may take several cycles to complete.

The former is needed for dynamically adjusting the movement, whereas the latter can be used to directly reach the destination when certain condition holds. To support point-to-point movement without violating timing constraints, its control is implemented in a separate thread which runs concurrently with the main closed-loop thread. In this way, the main thread does not have to wait for a point-to-point movement but can still manage it through event-based coordination with the control thread. Both types of robot control are employed to support various models in the cyberworkstation as further explained in Section III.

### *E. Web Portal*

The complexity of the above introduced techniques for supporting the closed loop is mostly hidden from the users by the cyberworkstation, whereas only the needed functionalities are exposed through an easy-to-use Web-based portal [8]. The portal facilitates the integration of data and resources to enable and catalyze collaborative research on DDDBMI. Users can access the portal from anywhere with an Internet connection, conveniently conducting research and collaborating with colleagues. The functionalities of the portal are provided through AJAX-based portlets [7], which allows for flexible customization of portal interfaces and responsive, asynchronous update of portal contents.

The experiment management portlet enables users to dynamically configure and control online experimentation as well as offline study. It cooperates with the other components of the cyberworkstation to realize the management of the entire closed loop. The monitoring portlet provides the static and dynamic information of computing resources as well as BMI jobs. The visualization portlet allows users to dynamically visualize and analyze data using the tools they are familiar with (e.g., MATLAB). Other portlets such as message boards, instant messaging tools, and wiki are also provided to foster collaboration upon the cyberworkstation.

## III. BMI MODELS

The cyberworkstation supports BMI models with distinct paradigms, including supervised learning and reinforcement learning, which demonstrates its feasibility and flexibility of facilitating DDDBMI research. (More details about the algorithms can be found in [1] and [2].)

### *A. Recursive Least Square based Model (RLS)*

RLS [2] uses the recursive least square method to model the brain function based on available data and can update the model on a sample-by-sample basis online. The algorithm is specifically optimized to satisfy the real-time experiment's need. Recursive model update is employed to reduce the amount of computation for generating the model during each closed-loop cycle. A sliding training window is used to further reduce the computation intensity and improve the online training time. It uses multidimensional inputs and outputs and separated modules for auto- and cross-correlation recursion, so that the same signal can be used to train different models at low cost. Based on the learned model, the robot control action is computed according to the acquired brain signals. Point-to-point robot movement is then utilized to reach the goal as decided by the computation.

### *B. Reinforcement Learning based Model (RLBMI)*

A fundamental limitation of the supervised learning based input-output modeling approach, such as RLS, is that paralyzed patients are unable to train their own models because they cannot move their limbs. To overcome this limitation, a semi-supervised approach (RLBMI [1]) based on reinforcement learning is designed to control the robot's movement. Using co-adaptation a computer agent finds the mapping between neural activity and behavior by maximizing the reward of completing a goal directed task. In this paradigm, one has access to not only the real animal brain but also the spatio-temporal activation of brain states (indirectly related to the environment) as the animal seeks a goal or reward. Incremental robot control is used to continuously exploit and explore the space.

## IV. EVALUATION

### *A. Setup*

This section uses an online RLBMI experiment to evaluate the cyberworkstation's effectiveness in supporting DDDBMI research and its ability of meeting the desired timing requirements. A 100ms deadline was imposed on each cycle of the closed loop consisting of the aforementioned four phases: data acquisition, network transfer, parallel model computation, and robot control. A total of 32 channels of brain signals were sampled through a DSP device. The robotic arm has 5 degrees of freedom. The data acquisition/robot control server has dual 2.4HGz Xeon processors and runs Windows Server 2003. The computation was conducted on a cluster of VMware ESX server 3.0 [5] based virtual machines. They are hosted on several dual 3.2GHz Xeon servers. Each virtual machine has 1GB RAM and runs Ubuntu Linux 7.04. The experiment was submitted, controlled, and monitored through the Web portal.

### B. Results

Fig. 2 shows the histogram of the closed-loop cycle time obtained from a 15-minute-long online RLBMI experiment with 8,151 iterations. This result clearly demonstrates that the cyberworkstation can provide a high-performance computing environment for researchers to run their real-time experiments as there is no missed deadline. Table 1 reports on the timing results in more detail, including the statistics of the entire cycle time as well as each individual phase's latency.

### C. Discussion

The cyberworkstation is built upon best-effort software and hardware resources in that they cannot provide any hard timing guarantee for the computation and I/O operations. The real-time closed-loop needed by DDDBMI experiments is supported through the middleware-level techniques explained in Section II. Nonetheless, as shown in Table I, a certain level of variances exists for the latencies due to the best-effort nature of the underlying resources.

Specifically, the operating systems deployed on the data acquisition/robot control server and the virtual machines are designed for general-purpose usage. Hence, even though the cyberworkstation has taken all the possible techniques to improve the timing for the experiment, relatively high variances are observed from the latencies of data acquisition, model computation, and robot control. To address this, our ongoing research is investigating special-purpose operating systems for stronger real-time support.

Another factor that also contributes to the variance of computation time is that the amount of computation varies across different iterations of the experiment. In particular, only the iterations during an active trial require intensive computation, whereas the others involve negligible amount of processing. Consequently, the active-trial iterations have much higher computation time than the others.

The variance of network transfer time is strongly related to the particular network setup of the experiment. For security reasons, the virtual machines are deployed in an isolated private network and the access to them are forwarded through a gateway server at ACIS. This setup increases the network latency as well as its variance since the gateway server is also responsible for forwarding other unrelated network traffic. Our future investigation will consider improving the network transfer latency by reserving resources on the gateway server for forwarding the BMI-specific network traffic.

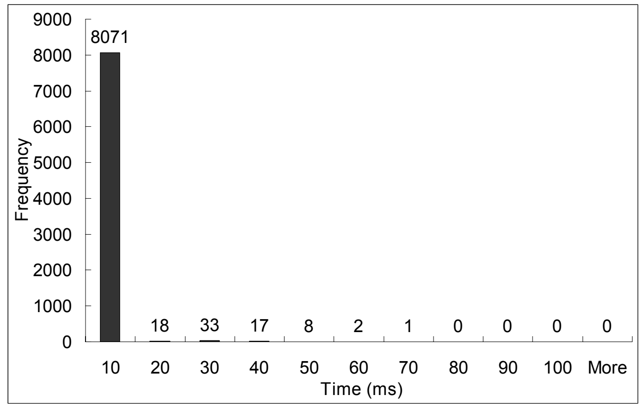


Fig. 2. The histogram of the closed-loop cycle time from an online RLBMI experiment with 8,151 iterations.

TABLE I
TIMING STATISTICS OF AN ONLINE RLBMI EXPERIMENT

| Latency | Min (ms) | Max (ms) | Average (ms) | Stdev (ms) |
|---|---|---|---|---|
| Entire closed loop | 1.071 | 64.265 | 1.943 | 2.830 |
| Data acquisition | 0.913 | 61.746 | 8.090 | 10.184 |
| Network transfer | 0.940 | 49.212 | 1.426 | 1.104 |
| Model computation | 0.145 | 17.756 | 0.310 | 0.222 |
| Robot control | 0.037 | 63.439 | 2.541 | 6.582 |

## V. CONCLUSION

This paper has presented an application-centric and user-oriented cyberinfrastructure for DDDBMI research. Online and offline BMI experiments can be performed on it in a closed-loop manner that includes in vivo data acquisition, reliable network transfer, parallel model computation, and real-time robot control. Scientists can conveniently deploy their algorithms on the cyberinfrastructure and conduct research through its Web portal. Two interesting models based on recursive least square (RLS [2]) and reinforcement learning (RLBMI [1]) are successfully developed and tested using this approach. Our future work will further improving this cyberinfrastructure to support DDDBMI with automatic model integration, complex experiment workflow, and Quality of Service driven resource management.

## REFERENCES


[1] J. DiGiovanna, B. Mahmoudi, J. Mitzelfelt, J. C. Sanchez, and J. C. Principe, "Brain-Machine Interface Control via Reinforcement Learning," in Proc. 3rd Intl. IEEE EMBS on Neural Engineering, 2007.

[2] J. DiGiovanna, L. Marchal, P. Rattanatamrong, M. Zhao, S. Darmanjian, B. Mahmoudi, J. C. Sanchez, J. C. Principe, L. Hermer-Vazquez, R. J. Figueiredo, and J. A. B. Fortes, "Towards Real-Time Distributed Signal Modeling for Brain Machine Interfaces," in Proc. International Conference on Computational Science, 2007.

[3] J. Fortes, R. Figueiredo, L. Hermer-Vazquez, J. Principe, and J. Sanchez, "A New Architecture for Deriving Dynamic Brain-Machine Interfaces", in Proc. Int. Conf. on Computational Science, 2006.

[4] M. Zhao and R. J. Figueiredo, "Experimental Study of Virtual Machine Migration in Support of Reservation of Cluster Resources," Proc. 2nd Int. Workshop on Virtualization Technology in Distributed Computing (held in conjunction with SC07), 2007.

[5] VMware ESX. URL : http://www.vmware.com/products/vi/esx

[6] Torque Resource Manager. URL: http://www.clusterresources.com/pages/products.php

[7] JSR 168: Portlet Specification. URL: http://www.jcp.org/en/jsr/detail?id=168

[8] DDDBMI Cyberworkstation. URL : http://bmi.acis.ufl.edu