Neural Networks 24 (2011) 950–960

Contents lists available at SciVerse ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

2011 Special Issue

Real-time simulation of a spiking neural network model of the basal ganglia circuitry using general purpose computing on graphics processing units

Jun Igarashi [a,b,*,1], Osamu Shouno [b,1], Tomoki Fukai [a,c], Hiroshi Tsujino [b]

[a] Computational Science Research Program, RIKEN, Japan

[b] Honda Research Institute Japan Co., Ltd, Japan

[c] Laboratory for Neural Circuit Theory, Brain Science Institute, RIKEN, Japan

ARTICLE INFO

Keywords:
Realistic neural networks
Real-time simulation
GPGPUs
Basal ganglia
High-performance computing

ABSTRACT

Real-time simulation of a biologically realistic spiking neural network is necessary for evaluation of its capacity to interact with real environments. However, the real-time simulation of such a neural network is difficult due to its high computational costs that arise from two factors: (1) vast network size and (2) the complicated dynamics of biologically realistic neurons. In order to address these problems, mainly the latter, we chose to use general purpose computing on graphics processing units (GPGPUs) for simulation of such a neural network, taking advantage of the powerful computational capability of a graphics processing unit (GPU). As a target for real-time simulation, we used a model of the basal ganglia that has been developed according to electrophysiological and anatomical knowledge. The model consists of heterogeneous populations of 370 spiking model neurons, including computationally heavy conductance-based models, connected by 11,002 synapses. Simulation of the model has not yet been performed in real-time using a general computing server. By parallelization of the model on the NVIDIA Geforce GTX 280 GPU in data-parallel and task-parallel fashion, faster-than-real-time simulation was robustly realized with only one-third of the GPU's total computational resources. Furthermore, we used the GPU's full computational resources to perform faster-than-real-time simulation of three instances of the basal ganglia model; these instances consisted of 1100 neurons and 33,006 synapses and were synchronized at each calculation step. Finally, we developed software for simultaneous visualization of faster-than-real-time simulation output. These results suggest the potential power of GPGPU techniques in real-time simulation of realistic neural networks.

1. Introduction

With the benefit of exponential growth in computer performance, simulations of large-scale realistic neural networks have recently been undertaken to explore the mechanisms underlying information processing in the brain (Ananthanarayanan, Esser, Simon, & Modha, 2009; Esser, Hill, & Tononi, 2009; Izhikevich & Edelman, 2008; Morrison, Mehring, Geisel, Aertsen, & Diesmann, 2005). In most cases, the simulation studies have been performed in an offline state in which neural networks are separated from real environments. However, biological intelligence is assumed to emerge through real-time interactions between an organism and a real environment (Pfeifer & Scheier, 1999). In order to study how biological intelligence emerges, a detailed, biologically accurate model of the brain and its real-time interaction with the environment through a sensorimotor loop are essential. In a sensorimotor loop, the neural network model processes sensory information from the environment in real-time and affects the environment through its actuator, resulting in changes in the environment that in turn provides updated input to the model.

While real-time simulation of neural circuits with biologically realistic neural networks provides a very powerful tool for studying brain function, its computational cost is very large for two reasons: (1) the vast number of neurons and synaptic connections between them in neural networks of biologically realistic size, and (2) the complicated dynamics of a detailed, biologically accurate model of neurons. Therefore, the size and operation speed of a network model are often limited by a finite amount of both computational power and memory, making it difficult to simulate large-scale, biologically realistic neural network models in real-time.

A general purpose computing on graphics processing unit (GPGPU) is a highly effective approach to speeding up simulations of neural networks. While general central processing units (CPUs)

* Corresponding address: 2-1, Hirosawa, Wako, Saitama, 351-0198, Japan. Tel.: +81 48462 1111x7464; fax: +81 48 467 6899.
E-mail address: igarashi@brain.riken.jp (J. Igarashi).

[1] These authors contributed equally to this work.

0893-6080/$ – see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.neunet.2011.06.008

have several floating-point units (FPUs) at most, the latest graphics processing unit (GPU) has hundreds of FPUs, which enables massively parallel computation. A GPU delivers very high performance on graphics applications by means of data-parallel computation in which all threads execute identical operations on different data. Recently, several works have shown that GPU data-parallel computation is also very effective at simulating spiking neural networks, achieving much faster simulation of large-scale spiking neural networks than CPU implementations (Nageswaran, Dutt, Krichmar, Nicolau, & Veidenbaum, 2009; Nowotny, 2010; Scorcioni, 2010; Yudanov, Shaaban, Melton, & Reznik, 2010).

In these GPU implementations, each neuron is mapped on a thread and executed in parallel. In order to fully take advantage of the GPU's powerful data-parallel computation, all neuronal dynamics in a network should be modeled by an identical computational object, that is, an identical set of differential equations, as seen in previous studies. For example, Nageswaran et al. (2009) implemented a spiking neural network that consists of regular spiking neurons and inhibitory neurons. Although the two types of neurons are different in their membrane dynamics, each type of neuron is described by an identical set of two differential equations, known as Izhikevich simple model (Izhikevich, 2003).

However, biological neurons in the brain show a wide variety of neuronal dynamics, and reproducing unique neuronal dynamics with biological realism require custom-tuned, detailed conductance-based models that consist of various sets of differential equations. As a consequence, a biologically realistic, detailed neural network model is thus often heterogeneous in terms of computational object. Naïve GPU implementation of 'heterogeneous' networks is performed in a task-serial and data-parallel manner in which data-parallel computations for computationally distinct neuron models are serially processed. This type of implementation should result in computation time that increases linearly with the number of neuron models.

Additionally, detailed conductance-based neuron models incur much greater computational costs than simpler neuron models such as the Izhikevich simple model (Izhikevich, 2003) that was used frequently in previous studies. For example, even the Hodgkin–Huxley model, which is a basic conductance-based neuron model, requires more than 92 times the computational cost of the Izhikevich simple model (Izhikevich, 2003). Using these detailed, but computationally heavy neuron models make it difficult to simulate large-scale neural network models in real-time.

The goals of the current study were to efficiently implement 'heterogeneous' neural networks and to robustly simulate the neuron models with high computational costs in a given time. To accomplish this we introduced task-parallel computation on GPUs, in which different tasks (operations) can be performed in parallel and computational resources can be freely assigned to different tasks according to their computational costs to minimize overall computation time.

As the target for real-time simulation of a heterogeneous neural network on GPUs, we adopted a model of the basal ganglia circuitry developed by Shouno, Takeuchi, and Tsujino (2009). The basal ganglia model is a spiking neural network model that consists of heterogeneous populations of single-compartment, spiking neuron models (total 370 neurons), including two distinct, computationally heavy, conductance-based models in addition to computationally light neuron models such as Izhikevich neuron models and leaky integrate-and-fire neuron models. These various neuron models are connected by biological synapses (total 11,002 synapses) that are described by computationally heavy alpha or exponential decay functions and some of which are endowed with short-term plasticity.

The basal ganglia model demonstrates action selection with trial-by-trial variability. A real-time simulation of the basal ganglia model has not yet been performed using general computing servers with general CPUs. If the real-time simulation is realized, then the development of a robot with the basal ganglia model would lead to a study of action selection in interaction with changing real environments.

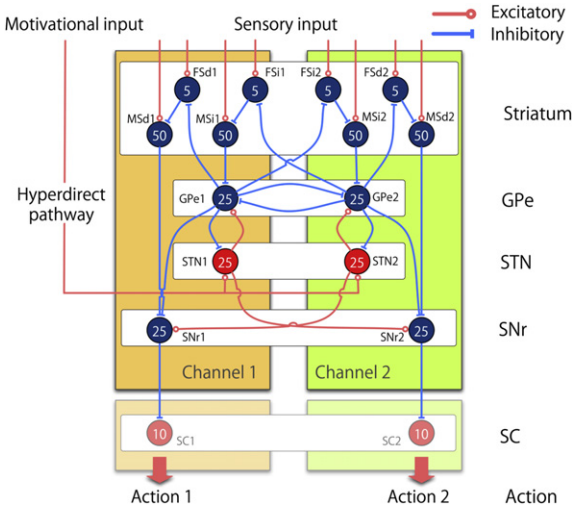

Fig. 1. Schematic model architecture. Circles represent neuron populations.

In this study, we demonstrated that mixture of data-parallel and task-parallel GPGPU computing robustly performed faster-than-real-time simulation of the basal ganglia model.

2. A model for the basal ganglia circuitry

The cortical-basal ganglia circuitry has been implicated in action selection and action initiation (Hollerman, Tremblay, & Schultz, 1998; Lee & Assad, 2003; Pasupathy & Miller, 2005; Samejima, Ueda, Doya, & Kimura, 2005; Tremblay, Hollerman, & Schultz, 1998; Turner & Anderson, 2005). Shouno et al. (2009) proposed a model for the basal ganglia circuitry that generates trial-by-trial variations in the selection and timing of actions. Fig. 1 shows the network structure of the basal ganglia model.

The cortico-basal ganglia circuitry has been assumed to be a collective of loops spanning these regions, called channels. In this model, there are two competing channels, each of which is assumed to be selective for the generation of a distinct action. Each channel consists of four nuclei, namely, the striatum, the external segment of the globus pallidus (GPe), the subthalamic nucleus (STN) and the substantia nigra pars reticulate (SNr). In the model striatum there are two types of neurons, medium spiny output neurons (MS) and fast-spiking GABAergic interneurons (FS). Striatal neurons are subdivided into direct pathway neurons (MSd and FSd) and indirect pathway neurons (MSi and FSi).

The basal ganglia model features the GPe–STN network that support two self-sustained, network activity patterns, repetitive firing and rhythmic burst, by virtue of strong post-excitatory rebound excitation of STN cells and stuttering behavior of GPe cells (Plenz & Kitai, 1999, Terman, Rubin, Yew, & Wilson, 2002). Additionally, the GPe–STN network is able to switch its activity pattern, from repetitive firing to rhythmic burst, in activity-dependent manner by virtue of short-term facilitation at GPe-to-STN GABAergic synapses. Taking advantage of the characteristics of the GPe–STN network activity, the basal ganglia model is able to generate exploratory actions, that is necessary for reinforcement learning, by random selection of an action and by variable timing of an action, without long-term synaptic changes at cortico-striatal synapses. Note that the cortico-striatal input to the model does not have any special spatiotemporal patterns, such as spatial

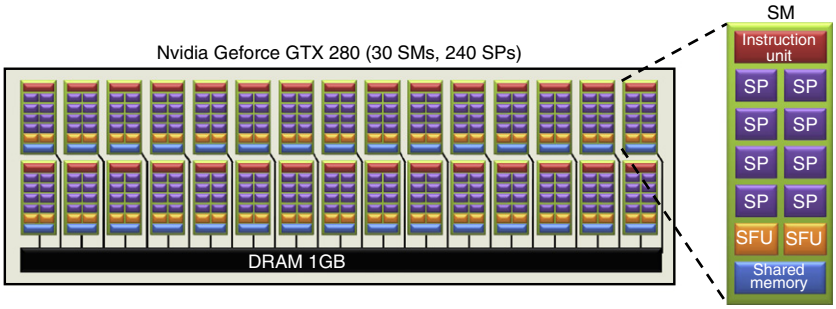

Fig. 2. Schematic diagram of GPU architecture.

bias between channels, ramping intensity, abrupt increase, etc., and is modeled as Poisson spike trains whose rate parameters are constant throughout a simulation. Concerning how the basal ganglia model works, see Section 5.1.

To realize the characteristic GPe–STN network activity patterns, detailed conductance-based models are required for GPe and STN cells. Neuronal dynamics of STN and GPe neurons are modeled using single-compartment, conductance-based models (Otsuka, Abe, Tsukagawa, & Song, 2004; Terman et al., 2002, Appendix, Tables 2 and 3). However, the striatum and SNr models are simplified as far as the basal ganglia model can show exploratory behaviors, in order to highlight the critical roles of the GPe and STN and save computation times. We modeled each type of striatal neuron using the Izhikevich simple model, and SNr neurons using the leaky integrate-and-fire neuron model (Izhikevich, 2007, Appendix). In this sense, the composition of neuron models in the basal ganglia model could contribute to real-time simulations in some extent. However, note that the original purpose of adopting this composition is not for the real-time simulations. Synaptic currents between neurons are modeled using a biological synapse model: alpha function for MSi→GPe, FSd→MSd, FSi→MSi, GPe→FSd, GPe→FSi, GPe→GPe, GPe→STN, STN→GPe; exponential decay function for MSd→SNr, STN→SNr, GPe→SNr (Appendix, Table 4). Several connections (GPe→FSd, GPe→FSi, GPe→STN and GPe→GPe) are endowed with short-term synaptic plasticity (Dayan & Abbott, 2002, Appendix, Table 5).

Details on implementing the basal ganglia model on a GPGPU and associated calculation times are explained below.

3. Computational environment—GPU architecture and CUDA

We used an Intel Xeon Quad Core CPU (2.0 GHz) desktop computer equipped with a NVIDIA GeForce GTX 280 GPU card and 2.75 GB DRAM memory (Table 1). The NVIDIA GeForce GTX 280 GPU card has 30 Stream Multiprocessors (SM) each of which consists of eight Stream Processors (SPs), 16 KB Shared Memory, a Super Function Unit (SFU), and an Instruction Unit (Fig. 2). The GPU has a theoretical performance of 933 GFLOPS for single-precision floating-point operation and 78 GFLOPS for double-precision floating-point operation. Each SP can execute a single-precision floating-point multiply–add instruction per cycle (NVIDIA, 2008). The graphics card has 1.0 GB of GPU-dedicated on-card DRAM memory.

As a GPGPU environment, we used CUDA 2.0 provided by NVIDIA. CUDA is an integrated environment of the C language for GPGPU (NVIDIA, 2008). In CUDA programming, there is a hierarchical structure consisting of thread, block, and grid, which are mapped on the SP, SM, and device, respectively. A thread is the lowest level process in the hierarchical structure. A block is a group of threads. A grid is the highest level structure and consists of blocks. A single SM can execute 32 threads in one block per four clock cycles. In CUDA, each group of 32 parallel threads is called a "warp". Each thread belonging to the same grid executes the same C function, called a kernel. In CUDA, basically, same operation is executed by different threads for different data in parallel (data-parallel computing). However, different operations can be also executed by different blocks in parallel using conditional branching (task-parallel computing). For example, when all the threads in block "A" are assigned to the branch of equation "A", and all the threads in block "B" are assigned to the branch of equation "B", the two different blocks execute the two different equations in parallel.

Table 1
Calculation environments.

	GPU	Control
CPU	Intel Xeon Quad Core (2.0 GHz)	Intel Xeon Quad (3.2 GHz)
Memory	2.75 GB	
GPU	NVIDIA GeForce GTX 280 (240 SP, 30 SM, Graphic memory 1 GB)	–
OS	Windows Xp (32 bit)	Mac OS X 10.5.7
Etc.	CUDA 2.0	NEST simulator (Ver. 1.9, Intel Compiler 10.1.014 — O2)

4. GPGPU implementation of the basal ganglia model

4.1. Parallelization of the basal ganglia model

To effectively simulate the basal ganglia model on a GPU card, we had to determine which part of the model had a high computational cost and how the model should be divided and performed in parallel. We investigated the breakout of computation time. The model was divided into 26 types of processes that could be calculated in parallel: seven types of model neurons (STN, GPe, MSd MSi, FSd, FSi and SNr), 12 types of synaptic conductances, and seven types of synaptic background noises (STN, GPe, MSd MSi, FSd, FSi and SNr). We measured the computation time for updating the state of only one neuron or one synapse using one thread executed on one SP of one SM. Fig. 3 shows the computation times of representative processes for conductance-based neuron models (STN, GPe), a non-conductance-based neuron model (FS), a synapse (MSd to FSd), and synaptic background noise (FS). When we used single-precision floating-point operation, the computation time for each process was shorter than the duration required to complete the process in a biological setting (10 s). We also tested computation with double precision for STN. The computation time increased 10.4 times longer than that with the single precision. The ratio of the computation times between both precisions was close to the ratio of the theoretical performances between both precisions (see Section 4.4). It would be impossible to realize real-time simulation of the model using double-precision floating-point operation, because the computation time of one process substantially exceeded the limit of real-time simulation. Although computation time of each process with single-floating point operation was within the limit of real-time simulation, summation of the calculation times of all updating functions

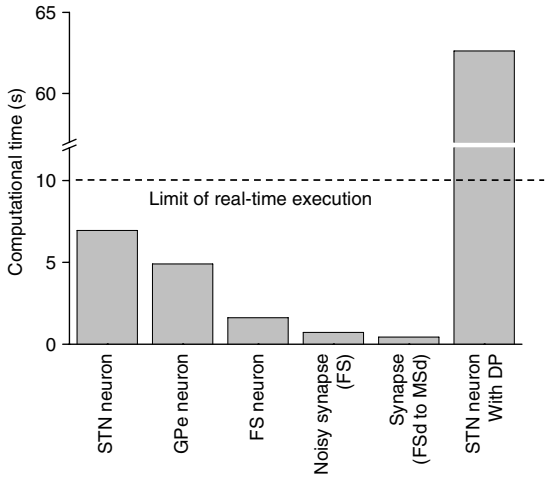

Fig. 3. Computational time for completion of representative processes of 10 s in biological time.

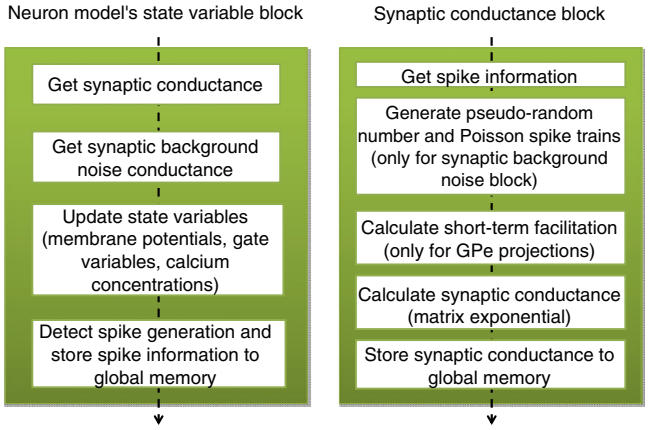

Fig. 5. Flowchart of computation executed within blocks of each neuron's state variables and synaptic conductance.

exceeded the limitation of biological time (10 s). We therefore concluded that different types of update functions should be performed in parallel, not in serial, to achieve real-time simulation of the entire basal ganglia model using single-precision floating-point operation.

Our simulation strategy used a mix of data-parallel and task-parallel GPU computing. Basically, in the computing, similar types of update functions in one channel were executed on the same SM in a data-parallel fashion, and different types of update functions were executed on different SMs in a task-parallel fashion. The combination of these computation approaches on a GPU for real-time simulation of realistic and heterogeneous neural networks is an original feature of our study.

The number of neurons in each neuron group in the model, excluding FSd and FSi, occurred in multiples of 25. Execution of a block with less than 32 threads on one SM is faster than that of a block with more than 32 threads due to the CUDA "warp" mechanism (Section 3). In terms of efficient usage of computational resource on a GPU, execution of more than several warps on one SM is better because the threads of more than several warps can mask memory access latency. However, minimizing computation time in a given period is critical in real-time simulation. Therefore, we arranged 25 threads in a block.

Based on a consideration of computational cost, SMs were assigned to the update functions so as to both keep overall computation time below biological time and to achieve efficient allocation of resources (Fig. 4). The task assignment to different SMs was performed by conditional branching using the block index number. Calculation of conductance-based neuron models of the STN neuron and the GPe neuron took a much longer time than the other processes. One SM was assigned to 25 STN neurons or 25 GPe neurons in one channel. The calculation time for the other neuron models (MSd, MSi, FSd, FSi and SNr) was much shorter than that required for the STN and GPe neuron models. One SM was assigned to the update functions of the computationally light neuron models in each channel, and these functions were executed in a serial manner on the SM. The calculation times for synaptic background noise and each synaptic conductance were much shorter than those for the neuron models. Twelve types of synaptic connections were arranged in three groups: synapse 1 (MSd→SNr, MSi→GPe, STN→SNr, SNr→SC), synapse 2 (FSd→MSd, FSi→MSi, GPe→FSd, GPe→FSi, GPe→SNr, and GPe→GPe), and synapse 3 (GPe→STN, STN→GPe). Three SMs were assigned to the computation of each synapse group. One SM was assigned to the synaptic background noise of all neurons. Finally, the calculation processes of the basal ganglia model were pulled together into 10 groups as shown in Fig. 4. Within an SM, the computational flows for updating the state variables of neurons or synaptic conductances are shown in Fig. 5. We confirmed that each group was calculated faster than real-time. The 10 groups were synchronized at time steps of 0.05 ms using flags stored in the GPU-dedicated DRAM memory that is called global memory.

4.2. Pipeline parallel processing of synaptic conductance using conduction delay

Synaptic conductance was calculated in parallel with the state variables of neuron models (membrane potentials, gate variables and calcium concentrations) by taking advantage of uniform conduction delay throughout the model. Fig. 6 shows the processing flow of synaptic conductance and the variables of neuron models. When the basal ganglia model was calculated on the GPU, conduction delay was 0.1 ms and time step was 0.05 ms.

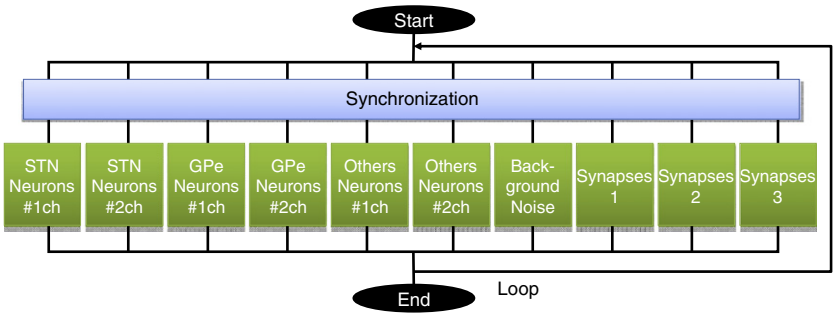

Fig. 4. Schematic diagram of parallelization and processing flow. Ten middle squares are processes calculated by one block. All processes were synchronized at each time step.

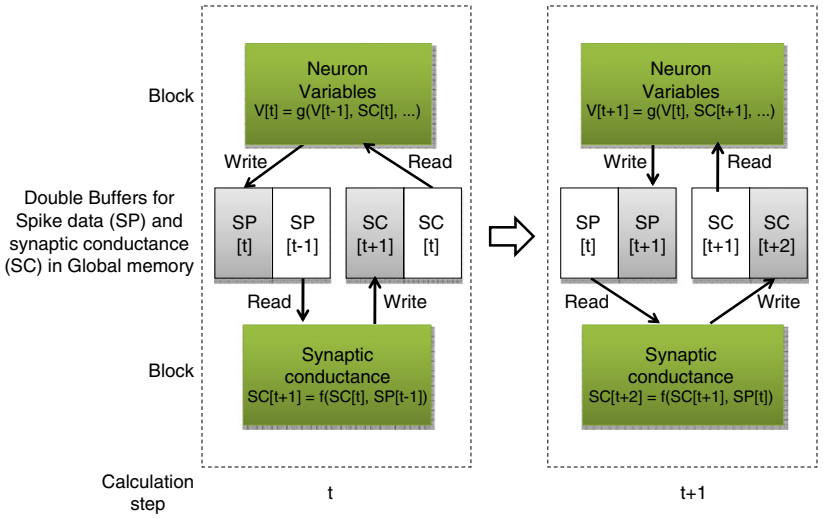

Fig. 6. Schematic diagram of pipeline parallel calculation of synaptic conductance and neurons' state variables taking advantage of uniform conduction delays. Spike history and synaptic conductance data were transferred through double buffering in global memory.

There was a gap of two time steps between the generation of a spike in presynaptic neuron and the induction of synaptic currents in postsynaptic neurons by the spike. Therefore, when neuron variables were calculated at step t, synaptic conductance at step $t + 1$ could be simultaneously calculated according to information about the spike at step $t - 1$ in pipeline parallel fashion. Spike histories and synaptic conductances were transferred between synaptic conductance blocks and neuron variable blocks through global memory. We implemented double buffering for spike history and synaptic conductance on global memory because loading and storing spike history or synaptic conductance should be simultaneously executed in pipeline parallel fashion. The pipeline parallel processing enabled to hide calculation time of synaptic conductance behind the calculation time of conductance-based neuron models.

4.3. Optimization of memory usage

Threads executed on an SM can access shared memory in the SM within several clock cycles that is comparable to general L1 cache (Fig. 2, right). Users can freely manage shared memory. However, the capacity of shared memory was not large enough (16 KB per SM) to store all variables in the basal ganglia model. In contrast, GPU-dedicated DRAM memory, called "global memory", has a large capacity of 1 GB; however, it takes 400–600 clock cycles to access. Assignment of data to shared memory is critical for efficient GPU computing. The state variables of neuron models such as membrane potentials, gate variables and calcium concentration were stored in shared memory because they were used frequently and repetitively. Shared memory is also useful for data transfer between threads within a block. Synaptic conductances calculated by each thread within a block were summed up through shared memory. All other variables and parameters were stored in global memory.

4.4. Numerical algorithms

The fourth-order Runge–Kutta method with a time step of 0.05 ms was used for updating the states of all neuron models except for the GPe model. The Runge–Kutta–Fehlberg method with a 0.025 ms time step was applied to the GPe neuron model although synchronization with the other processes and renewal of synaptic conductance values were performed every 0.05 ms in the same manner as the other neurons. The matrix exponential method (Rotter & Diesmann, 1999) was used for the calculation of synaptic conductance and short-term facilitation (Dayan & Abbott, 2002). In the block associated with synaptic background noise, spikes were generated by a Poisson process using pseudorandom numbers generated by the Xorshift method (Marsaglia, 2003).

The original CPU implementation of the basal ganglia model used double-precision floating-point operation with calculation time steps of 0.1 ms. However, double-precision floating-point operations are theoretically about 12 times less efficient than single-precision floating-point operations in an NVIDIA Geforce GTX 280 GPU. Although single-precision operations require a smaller simulation time step than double-precision operations to reproduce neural activity in the basal ganglia model, the computational performance of single-precision operation in a GPU still had advantages compared with double-precision operation. Therefore, we used single-precision floating-point operations for the GPGPU implementation.

We used the intrinsic functions executed on the SFU for the exponential function and the division.

4.5. CPU tasks

The CPU in the host computer mainly performed processes related to the preparation and termination of simulation, and did not perform numerical calculation. Before simulation, the CPU generated pseudorandom number seeds, started up the GPU, and allocated the memory of the host computer and the GPU. When the neural activity of the basal ganglia model was displayed on-screen simultaneously with simulation, OpenGL drawing functions were called from the CPU. At the end of simulation, the CPU released both host and GPU memory.

5. Results

5.1. Reproduction of action selection with trial-by-trial variations of the basal ganglia model

The basal ganglia model implemented on GPU reproduced action selection and action initiation with trial-by-trial variations. Fig. 7 shows the reproduced neural activity of the basal ganglia model. The activity of the model started with the silent state of P0 (Fig. 7(A), (B)). The SNr cells are tonically active enough to exert inhibition on downstream SC cells to shut them down.

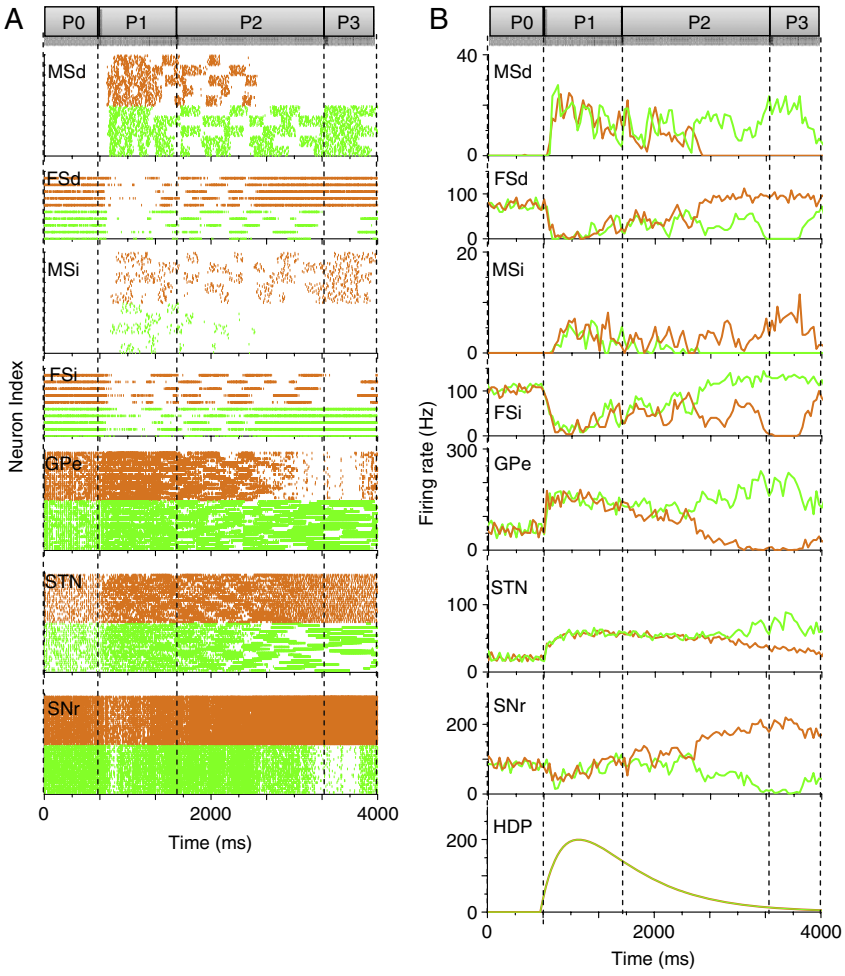

Fig. 7. Neural activity of the basal ganglia model reproduced by GPU. (A) Spike raster plots for FSd, FSi, MSd, MSi, GPe, STN, and SNr. Upper half: channel 1, Lower Half: channel 2. (B) Corresponding population firing rates for the spike raster plots and mean rate of the spike input through the hyper-direct pathway (HDP) to STN.

GPe and STN cells exhibit spontaneous firing (∼80 Hz, and ∼20 Hz, respectively). Striatal FS neurons show tonic activity in response to excitatory cortico-striatal inputs. Although striatal MS neurons also receive excitatory cortico-striatal inputs that exceed their threshold, MS neurons are silent because they receive tonic inhibition from FS neurons that fire more readily than MS neurons. When the excitatory Poisson spike inputs through the hyper-direct pathway are given to STN neurons during the P1 period (see panels HDP in Fig. 7(B)), STN neurons, and then GPe neurons elevate their firing rates that lead to increasing inhibitory inputs on STN neurons by short-term facilitation, and eventually post-inhibitory rebound excitations of STN neurons are triggered and the GPe–STN network activity switch its patterns from repetitive firing to rhythmic bursts. The elevated GPe activations also shut down striatal FS neurons through their inhibitory projections on FS neurons (Bevan, Booth, Eaton, & Bolam, 1998), and, thus, open temporal windows through which MS neurons are released from the feedforward FS inhibition. As a consequence, the MS neurons are episodically activated. Furthermore, the elevated GPe activations increase mutual inhibitions between GPe populations of competing channels by the short-term facilitation at GPe-to-GPe inhibitory synapses, and instantiate the winner-take-all dynamics between the GPe–STN networks of different channels. The competition between channels eventually results in diverged activities between channels: the GPe and STN neurons in the loser channel (in this example, channel 1) are decreasing their activity while the GPe and STN neurons in the winner channel (in this example, channel 2) sustain their rhythmic burst activity (the P2 period). As a result of the selection of the GPe–STN network of the channel, the winner SNr neurons decrease their activity due to the increased inhibition from the GPe in the same channel and decreased excitation from the STN in the competing channel, while the loser SNr elevate their activity due to decreasing the GPe inhibition from the same channel and increasing STN excitation from the competing channel. In parallel, the striatal MS neurons also show diverging activities between channels: the winner MSd neurons continue their episodic firing, while others become silent (the P2 period). Once the amount of inhibitory inputs from the GPe neurons and MSd neurons on the winner SNr exceeds the threshold, the SNr neurons are shut down. Consequently, downstream SC neurons are released from the inhibition from the SNr, and trigger an action (the P3 period). These network activities were quantitatively consistent with the results computed by CPUs (Shouno et al., 2009).

The observed marked variance in timing of action initiation in GPU simulations was qualitatively similar with that observed in CPU simulations. Fig. 8(A) shows distributions of action execution times for two channels when the same cortico-MSi synaptic strengths were applied for both channels, which corresponded to $x = 0$ in Fig. 8(B). The two distribution did not differ significantly (channel 1: 5427 ± 1345 ms, channel 2: 5248 ± 1175, t-test, $P > 0.05$). GPU simulations showed that the selection of channel was probabilistic, and the selection probabilities are varied with increasing difference between channels in the cortico-MSi synaptic strengths, consistent with probabilistic nature of the action selection of the model observed in CPU simulations

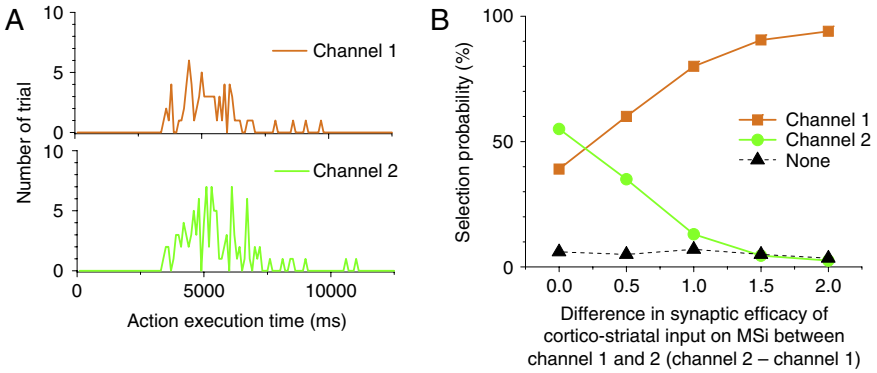

Fig. 8. Action execution of the basal ganglia model reproduced on GPU. (A) Distributions of action execution times for channel 1 (upper) and channel 2 (lower). $t = 0$ corresponds to the onset of the task. (B) Selection probability is dependent on interchannel differences in the synaptic efficacy of cortico-striatal inputs to MSi neurons. The zero point of the horizontal axis corresponds to the condition described in Fig. 7.

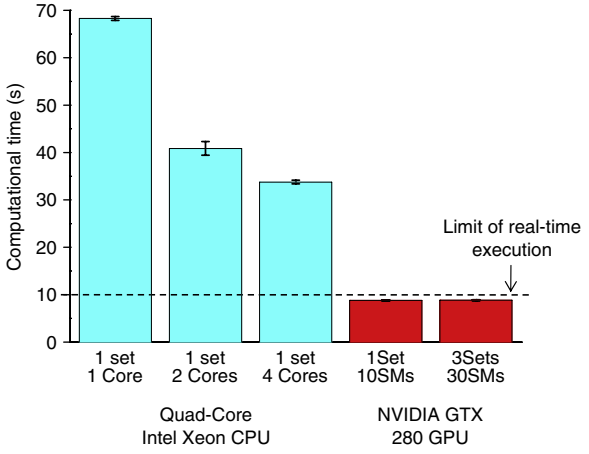

Fig. 9. Calculation time of the basal ganglia model for 10 s of biological time using GPU and CPU. Gray bar: CPU, White bar: GPU.

(Fig. 8(B)). Getting together, GPU implementation reproduced complex and functional behaviors of the basal ganglia model.

5.2. Faster-than-real-time emulation of the basal ganglia model on the GPU

First, we tested the basal ganglia simulation performance of a general computing server with an Intel Quad Core Xeon CPUs (Table 1). Simulation was performed by NEST simulator on the CPU (Gewaltig & Diesmann, 2007). The calculation time step was 0.1 ms and double-precision floating-point operation were used. We measured the time it took to simulate 10 s of biological time. Performing the simulation using 1–4 CPU cores, the time required for computation ranged from 68.3 ± 0.4 to 33.8 ± 0.4 s (200 trials), respectively, which is approximately three times longer than biological time (Fig. 9).

When simulation was performed on the NVIDIA Geforce GTX 280 GPU, actual computation time was 8.8 ± 0.01 s (200 trials), which was shorter than the 10 s of biological time (Fig. 9). Therefore, faster-than-real-time simulation was performed on the GPU with low variation in calculation time. Ten SMs, comprising one-third of the total resources in the GPU, performed about 3.8 times faster than the four cores of the CPU. In order to investigate whether the faster-than-real-time computation on the GPU was maintained at every moment, calculation times were measured every 50 ms through 10 s of simulation. Despite changes in firing rate in the model over time, faster-than-real-time emulation on the GPU was sustained throughout the simulation. These results suggest that GPGPU may be effective for robustly performing real-time simulation of realistic neural network models.

When the basal ganglia model was simulated using 10 of the GPU SMs, 20 SMs that comprised two-thirds of the GPUs computational resources were not used. Next, three instances of basal ganglia models were calculated using 30 SMs. The models had exactly the same network structure as described above, and were synchronized at each calculation step of 0.05 ms. There were no synaptic connections between the three instances. The calculation time for the three instances of the model was 8.84 ± 0.01 s, which was shorter than 10 s of biological time, although the calculation time for the three instances increased by ∼0.04 s compared to that for one instance due to the additional synchronization processes (Fig. 9). These results suggest that the GPGPU approach has an advantage over CPUs for extending the size of neural networks that can be performed in real-time.

Finally, activity of the single basal ganglia model was visualized on-screen (Fig. 10; supplementary video can be viewed online at doi:10.1016/j.neunet.2011.06.008). Calculation time, animation of model activity, spike raster and population firing rate were displayed and renewed every 50 ms of biological time. Simultaneous execution of visualization and simulation was performed faster than real-time.

6. Discussion

In the present study, faster-than-real-time simulation of the basal ganglia model was robustly realized using only 10 SMs in an NVIDIA Geforce GTX 280 GPU. Further, faster-than-real-time simulation of three instances of the model, synchronized at each calculation step, was also robustly realized using all 30 SMs of the GPU. Finally, simultaneous execution of visualization and simulation was performed faster than real-time.

There are several simulation studies of neural networks using GPGPU (Mutch, Knoblich, & Poggio, 2010; Nageswaran et al., 2009; Nowotny, 2010; Scorcioni, 2010; Yudanov et al., 2010). Our work differed from the previous reports in its use of combined data-parallel and task-parallel computations; previous studies used data-parallel and task-serial computations. The difference originated mainly from the difference in complexity of the neuron models used in the simulations. For example, the Izhikevich model, which was often used in previous studies, needs 13 floating-point number operations for simulation of 1 ms of biological time, while the STN and GPe neuron model used in the current study needs 8860 and 9920 floating-point number operations for the simulation, respectively. The computational cost of the conductance-based models is 682–763 times larger than that of the Izhikevich model. From the point of view of real-time simulation,

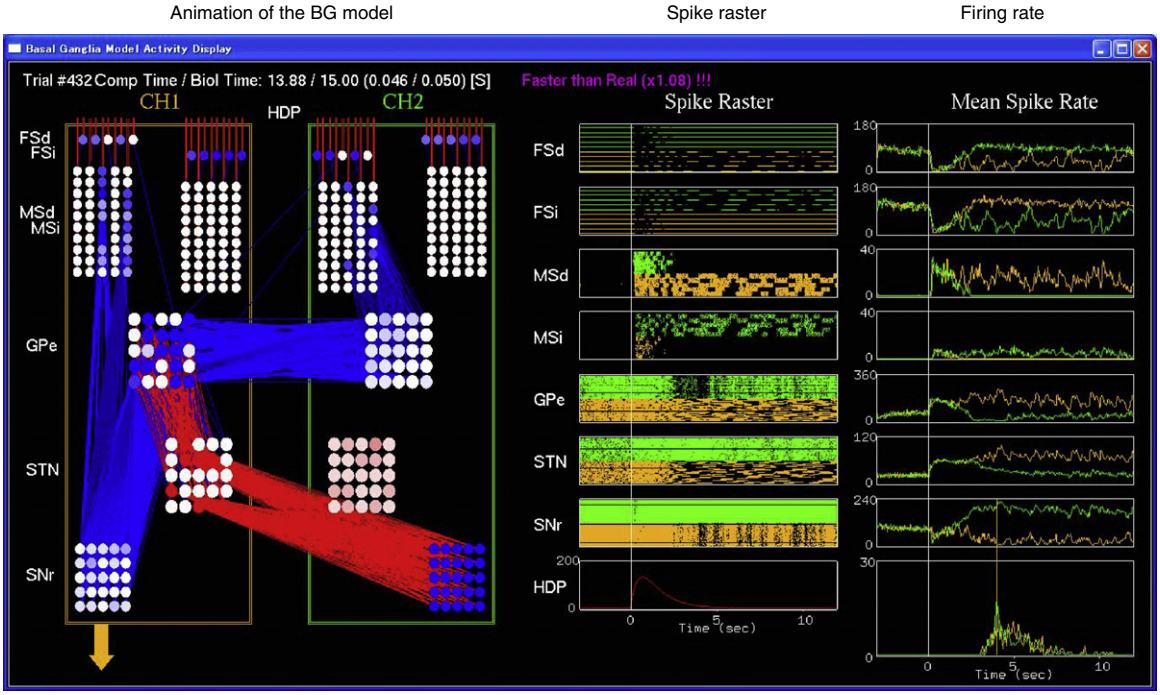

Fig. 10. Screenshot of the display window for the basal ganglia model during faster-than-real-time emulation. Left: Animation of neural activity. Middle: Spike rasters as shown in Fig. 7(A). Right: Population firing rates as shown in Fig. 7(B).

the computational time for the conductance-based model is comparable to biological time and there is little, remaining time for other computations. With task-serial computation, however, the computation occurring at each step in heterogeneous neural networks requires time equal to the product of "the number of neuron types" and "computational time of each neuron", and thus real-time simulation would be difficult. Using task-parallel computation, the total computational time is comparable to that of most heavy computation processes, that is, the computation of conductance-based neurons models, although there is an additional synchronization cost between processes in task-parallel computation.

We used single-precision floating-point operation for the computation of the basal ganglia model on GPU. Even if the basal ganglia model is implemented with double precision, it would be impossible to realize real-time simulation of the model although we did not implement the full size of the model due to the limitation of the amount of shared memory. In the new generation GPU released by NVIDIA, which is called Fermi architecture, the theoretical performance for double-precision floating-point operation is improved as compared with the previous generation GPUs. There is possibility to perform real-time simulation of the basal ganglia model with double precision using the new architecture GPU.

Related studies have reported implementations of neural networks by field-programmable gate arrays or application-specific integrated circuits that target large-scale simulation and real-time emulation (Khan et al., 2008; Merolla, Arthur, Shi, & Boahen, 2007; Vogelstein, Mallik, Culurciello, Cauwenberghs, & Etienne-Cummings, 2007). Compared to these works, the advantages of GPGPU implementation is that the hardware cost is cheap, the technical barrier is relatively low, and changing the structure of the neural network is easy and fast.

The faster-than-real-time simulation on the GPU was performed with very low variability in calculation times. The low variability arose from the pipeline parallel computation. The GPGPU technique was implemented so that computation time for synaptic conductance was kept shorter than that of conductance-based neuron models whose computation time should be relatively stable despite changes in the model's firing rate. As a result, the changing computational time for synaptic conductance was hidden behind the stable computation time of conductance-based neuron models. The low variability in computational time is a favorable feature for real-time emulation because significant variability could result in inconsistency between the model time and the environment time, which could interfere with seamless interaction between the model and environment.

In this study, we parallelized computations of the updating neuron's state variables and synaptic conductance in pipeline parallel fashion by taking advantage of uniform conduction delay in all synaptic connections. Even if the heterogeneous conduction delays are introduced, we can still parallelize the computations in pipeline parallel fashion by adding new process of "conduction delay manager" to the synapse conductance block in the current GPGPU implementation (Fig. 11). The conduction delay manager performs (1) calculation of the onset time of synaptic conductance at a receiver neuron from conduction delay and spike-generated times at a sender neuron, (2) setting schedule buffer of the onset times, and (3) searching the schedule buffer for onset time that matches the current time.

Faster-than-real-time simulation of one instance of the basal ganglia model was performed simultaneously with visualization, although total processing time increased slightly due to the overhead for image processing. Such visualization is useful for providing an intuitive understanding of the complicated activity of neural networks like the basal ganglia model.

GPGPU implementation allowed for the size extension of the basal ganglia model due to parallel computing performance. In the current study, we showed that three instances of the basal ganglia model synchronized at each calculation step could be simulated in faster than real-time. If synaptic connections between the three instances were added, a three-fold larger basal ganglia model consisting of six channels could be performed faster than real-time. However, the size extension achieved was still not enough to simulate a basal ganglia model of the same scale as the real basal ganglia. The total number of neurons in the rat basal ganglia

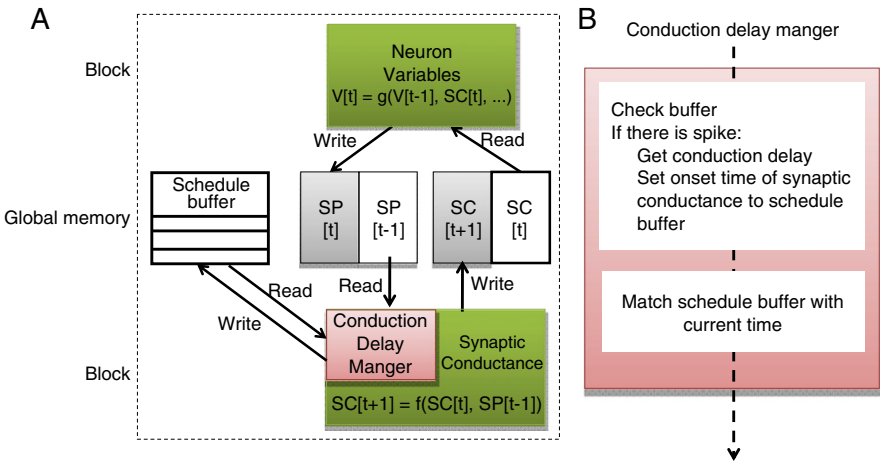

Fig. 11. Pipeline parallel processing neurons' state variables and synaptic conductance with conduction delay manger that handles variable conduction delays. (A) Schematic diagram of pipeline parallel processing with conduction delay manager. (B) Flowchart of processing in conduction delay manager.

has been estimated at around 3000,000 (Oorschot, 1996), while the current GPGPU implementation has 1110 neurons and 33,006 synapses. To realize a simulation of the same size as the real basal ganglia, we would need to extend the size of the current implementation more than 27,000-fold. GPU clustering may help with further scaling the size of the neural network. However, there may be barriers to be solved for the GPU clustering. The amount of computation per node of GPU cluster can be kept constant so that each node can perform real-time execution. However, the amount of spike data transferred among the GPU nodes may increase with scaling up neural networks, which may become bottleneck in the computation. Further, communication performance among the GPU nodes is generally lower than that among computing units within one GPU because GPU nodes send data through relatively slow data paths between host PC and GPU, such as PCI-express and DRAM memory in the host PC, and computer networking technologies connecting host PCs, such as gigabit Ethernet or InfiniBand with Message Passing Interface. However, the problems of data transfer might be avoided by introducing the simultaneous execution of computation and communication that is expected to hide latency for communication among GPU nodes. In order to perform real-scale simulation of the basal ganglia, it is estimated that the GPU cluster with more than 27,000 GPUs are required. If conductance-based neuron models were used for all neurons in the basal ganglia model instead of simple model neurons that accounted for 72% (270 simple neurons/370 neurons in 2 channels) in the current model, then, the substituted 270 conductance-based model neurons per 2 channels would require additional 10 SMs for computation. Therefore, totally, 20 SMs that are two times larger than 10 SMs in the original model would be needed for computation of 2 channels. According to this expectation of double computational resources, 54,000 GPUs would be needed for real-scale simulation. However, such a vast GPU cluster whose scale has never appeared in the world is not realistic in terms of economic cost and power consumption. If the Moore's low is sustained in the next 10 years and GPU architecture exists in the future, then, we may get an increase of computational performance of GPUs in ∼100 times, according to the history of semiconductor. Therefore, after 10 years, the GPU cluster consisting of 540 GPUs, which is a feasible size to be built, might be able to execute a full scale of the basal ganglia model. The diversity of action selection performed by the basal ganglia is thought to be proportional to its population size. Further scaling up the size of the basal ganglia model would be an important subject for a future study so as to reproduce more realistic and complex action selection.

Another step to be performed in the future is to implement the basal ganglia model in a robot with actuators and to study the interaction between the neural network and a real environment. In order to study the development of biological intelligence, we should introduce additional features, such as sensors receiving real inputs, cerebral cortical circuit models that process input signals and send their output to the basal ganglia, and long-term synaptic plasticity. These features may also be suitable for implementation by GPGPU techniques because they possess parallel structures that are typical in other processes in the brain.

Acknowledgments

This study was partly supported by 'The Next-Generation Integrated of Living Matter', as part of the Development and Use of the Next-Generation Supercomputer Project of the Ministry of Education, Culture, Sports, Science and Technology (MEXT), and also by Grant-in-Aids for Scientific Research on Innovative Areas (no. 22115013).

Appendix

STN neuron model

$$C_m \frac{dv}{dt} = -I_{Na} - I_K - I_A - I_L - I_T - I_{Ca\text{-}K} - I_l$$

$$I_{Na} = g_{Na} m^3 h (v - v_{Na})$$

$$I_K = g_K n (v - v_K)$$

$$I_A = g_A a^2 b (v - v_K)$$

$$I_L = g_L c^2 d_1 d_2 (v - v_{Ca})$$

$$I_T = g_T p^2 q (v - v_{Ca})$$

$$I_{Ca\text{-}K} = g_{Ca\text{-}K} r^2 (v - v_{Ca})$$

$$I_l = g l (v - v_l)$$

$$\frac{dw}{dt} = \frac{w_\infty(v) - w}{\tau_w} \quad (w : a, b, c, d_1, d_2, h, m, n, p, q)$$

$$w_\infty = \frac{1}{1 + \exp((v - \theta_w)/k_w)}$$

$$\tau_w = \tau_x^0 + \tau_x^1 / \{1 + \exp[-(v - \theta_x^\tau)/\sigma_x]\} \quad (I_{Na}, I_A)$$

$$\tau_w = \tau_x^0 + \tau_x^1 / \{\exp[-(v - \theta_x^{\tau 1})/\sigma_x^1] + \exp[-(v - \theta_x^{\tau 2})/\sigma_x^2]\} \quad (I_{others})$$

$$d[Ca]_i = -\alpha I_{Ca} - K_{Ca}[Ca]_i$$

Table 2
Basic set of parameter values for the STN model neuron.

Parameter	
g_{Na}	49
g_K	57
g_{IA}	5
g_L	15
g_T	5
$g_{Ca\text{-}K}$	1
g_l	0.35
v_{Na}	60
v_K	−90
v_l	−60
θ_a	−45
θ_b	−90
θ_c	−30.6
θ_{d_1}	−60
θ_{d_2}	0.1
θ_m	−40
θ_h	−45.5
θ_n	−41
θ_p	−56
θ_q	−85
θ_r	0.17
k_a	−14.7
k_b	7.5
k_c	−5
k_{d_1}	7.5
k_{d_2}	0.02
k_r	−0.08
α	5.182×10^{-3}
$\tau_a^0,\ \tau_a^1$	1, 1
$\tau_b^0,\ \tau_b^1$	0, 200
$\tau_c^0,\ \tau_c^1$	45, 10
$\tau_{d1}^0,\ \tau_{d1}^1$	400, 500
$\tau_m^0,\ \tau_m^1$	0.2, 3
$\tau_h^0,\ \tau_h^1$	0, 24.5
$\tau_n^0,\ \tau_n^1$	0, 11
$\tau_p^0,\ \tau_p^1$	5, 0.33
$\tau_q^0,\ \tau_q^1$	0, 400
θ_a^τ	−40
$\theta_b^{\tau 1},\ \theta_b^{\tau 2}$	−60, −40
$\theta_c^{\tau 1},\ \theta_c^{\tau 2}$	−27, −50
$\theta_{d1}^{\tau 1},\ \theta_{d1}^{\tau 2}$	−40, −20
θ_m^τ	−53
$\theta_h^{\tau 1},\ \theta_h^{\tau 2}$	−50, −50
$\theta_n^{\tau 1},\ \theta_n^{\tau 2}$	−40, −40
$\theta_p^{\tau 1},\ \theta_p^{\tau 2}$	−27, −102
$\theta_q^{\tau 1},\ \theta_q^{\tau 2}$	−50, −50
σ_a	−0.5
$\sigma_b^1,\ \sigma_b^2$	−30, 10
$\sigma_c^1,\ \sigma_c^2$	−20, 15
$\sigma_{d1}^1,\ \sigma_{d1}^2$	−15, 20
σ_m	−0.7
$\sigma_h^1,\ \sigma_h^2$	−15, 16
$\sigma_n^1,\ \sigma_n^2$	−40, 50
$\sigma_p^1,\ \sigma_p^2$	−10, 15
$\sigma_q^1,\ \sigma_q^2$	−15, 16

v is membrane potential. a, b, c, d_1, d_2, h, m, n, p, q and r are gate variables. C_m is membrane capacitance. See Table 2 for a list of parameters.

GPe neuron model

$$C_m\frac{dv}{dt} = -I_l - I_K - I_{Na} - I_T - I_{Ca} - I_{AHP} - I_{syn}$$
$$I_l = g_l(v - v_l)$$
$$I_{Na} = g_{Na}m^3h(v - v_{Na})$$
$$I_T = g_Ta^3r(v - v_{Ca})$$
$$I_{Ca} = g_{Ca}s^2(v - v_{Ca})$$
$$I_{AHP} = g_{AHP}(v - v_K)([Ca]/([Ca] + k_1))$$
$$\frac{dw}{dt} = \phi_x\frac{w_\infty(v) - w}{\tau_x(v)} \quad (w : n, h, r)$$

Table 3
Basic set of parameter values for GPe neuron.

Parameter	
g_{Na}	120.0
g_K	30.0
g_l	0.1
g_T	0.5
g_{Ca}	0.15
g_{AHP}	30.0
v_{Na}	55.0
v_K	−80.0
v_{Ca}	120.0
v_l	−55.0
θ_a	−57.0
θ_b	−35.0
θ_g	20.0
θ_h	−58.0
θ_m	−37.0
θ_n	−50.0
θ_r	−70.0
θ_s	−35.0
ϕ_h	0.05
ϕ_n	0.05
ϕ_r	1.0
τ_h^0	0.05
τ_n^0	0.05
τ_h^1	0.27
τ_n^1	0.27
τ_r	30.0
θ_h^τ	−40.0
θ_n^τ	−40.0
σ_a	2.0
σ_m	10.0
σ_h	−12.0
σ_n	14.0
σ_r	−2.0
σ_s	2.0
σ_h^τ	−12.0
σ_n^τ	−12.0
k_l	30.0
k_{Ca}	20.0
ε	1.0×10^{-4}

$$w(v) = 1/[1 + \exp[-(v - \theta_w)/\sigma_w]] \quad (w : n, m, h, a, r, s)$$
$$\tau_w = \tau_w^0 + \tau_w^1/\{1 + \exp[-(v - \theta_w^\tau)/\sigma_w^\tau]\} \quad (w : n, m, h, a, r, s)$$
$$d[Ca]_i = \varepsilon I_{Ca} - I_T - K_{Ca}[Ca]_i$$

v is membrane potential. a, h, m, r, and s are gate variables. C_m is membrane capacitance. See Table 3 for a list of parameters.

FS neuron model

$$20\dot{v} = (v + 55)(v + 40) - u + I$$
$$\dot{u} = 0.2\{U(v) - u\}, \quad \text{if } v > 25, \text{ then } v \leftarrow -45$$
$$U(v)\begin{cases} = 0 & (v < v_b) \\ = 0.025(v - v_b)^3 & v \geq v_b \end{cases}.$$

MS neuron model

$$50\dot{v} = (v + 80)(v + 25) - u + I$$
$$0.01\dot{u} = \{-20(v + 80) - u\},$$
$$\text{if } v \geq 40, \text{ then } v \leftarrow -55,\ u \leftarrow u + 150.$$

Synaptic conductance

Alpha function

$$g_s = \frac{g_{max}t}{\tau_s}\exp(1 - t/\tau_s).$$

Exponential decay

$$g_s = g_{max}\exp(-t/\tau_s).$$

The matrix exponential method (Rotter & Diesmann, 1999) was used for the calculation of synaptic conductance. See Tables 4 and 6 for parameters of synaptic connections.

Table 4
Maximum conductances of synaptic connections.

Connection	g_max
FSd→MSd	8.0
FSi→MSi	8.0
MSi→GPe	0.004
MSd→SNr	1.2
GPe→STN	5.0
GPe→GPe	0.013
GPe→SNr	1.1
GPe→FSd	3.0
GPe→FSi	3.0
STN→GPe	0.055
STN→SNr (different Ch.)	3.0
STN→SNr (same Ch.)	0.5

Table 5
Basic set of parameter values for short-term plasticity.

Connection	τ	Increment factor
GPe→FSi	200	0.02
GPe→FSd	200	0.02
GPe→STN	500	0.0001
GPe→GPe	200	0.0098

Table 6
Basic set of parameter values for back ground noise.

	Mean rate (Hz)	g_max
FSd	1000	3.2
FSi	1000	3.2
MSd	1000	2.2
MSi	1000	2.2
STN	200	0.04
GPe	150	0.025
SNr	2250	2.866

Short-Term Plasticity

$$\tau_P \frac{dp_{\mathrm{rel}}}{dt} = P_0 - P^{rel}.$$

Short-term facilitation rule (Dayan & Abbott, 2002) was set to synaptic connections GPe→FSd, GPe→FSi, GPe→STN, and GPe→GPe. The matrix exponential method (Rotter & Diesmann, 1999) was used for the calculation of short-term facilitation (Table 5).

References

Ananthanarayanan, R., Esser, S. K., Simon, H. D., & Modha, D. S. (2009). The cat is out of the bag: cortical simulations with 10^9 neurons and 10^13 synapses. In *Supercomputing 09: proceedings of the ACM/IEEE SC2009*.

Bevan, M. D., Booth, P. A. C., Eaton, S. A., & Bolam, J. P. (1998). Selective innervation of neostriatal interneurons by a subclass of neuron in the globus pallidus of the rat. *Journal of Neuroscience*, *18*, 9438–9452.

Dayan, P., & Abbott, L. F. (2002). *Theoretical neuroscience: computational and mathematical modeling of neural systems*. Cambridge, MA: The MIT Press, (Chapter 5).

Esser, S. K., Hill, S., & Tononi, G. (2009). Breakdown of effective connectivity during slow wave sleep: investigating the mechanism underlying a cortical gate using large-scale modeling. *Journal of Neurophysiology*, *102*, 2096–2111.

Gewaltig, M. O., & Diesmann, M. (2007). NEST. *Scholarpedia*, *2*, 1430.

Hollerman, J. R., Tremblay, L., & Schultz, W. (1998). Influence of reward expectation on behavior-related neuronal activity in primate striatum. *Journal of Neurophysiology*, *80*, 947–963.

Izhikevich, E. M. (2003). Simple model of spiking neurons. *IEEE Transactions on Neural Networks*, *14*, 1569–1572.

Izhikevich, E. M. (2007). *Dynamical systems in neuroscience*. The MIT Press (Chapter 8).

Izhikevich, E. M., & Edelman, G. M. (2008). Large-scale model of mammalian thalamocortical systems. *PNAS*, *105*, 3593–3598.

Khan, M., Lester, D., Plana, L., Rast, A., Jin, X., & Painkras, E. (2008). SpiNNaker: mapping neural networks onto a massively-parallel chip multiprocessor. In *IEEE international joint conference on neural networks* (pp. 2849–2856).

Lee, I. H., & Assad, J. A. (2003). Putaminal activity for simple reactions or self-timed movements. *Journal of Neurophysiololgy*, *89*, 2528–2537.

Marsaglia, G. (2003). Xorshift RNGs. *Journal of Statistical Software 8*, *6*.

Merolla, P., Arthur, J., Shi, B. E., & Boahen, K. (2007). Expandable networks for neuromorphic chips. *IEEE Transactions on Circuits and Systems I*, *54*, 301–311.

Morrison, A., Mehring, C., Geisel, T., Aertsen, A, & Diesmann, M. (2005). Advancing the boundaries of high connectivity network simulation with distributed computing. *Neural Computation*, *17*, 1776–1801.

Mutch, J., Knoblich, U., & Poggio, T. (2010). CNS: a GPU-based framework for simulating cortically-organized networks. Computer Science and Artificial Intelligence Laboratory. Technical Report, MIT-CSAIL-TR-2010-013 / CBCL-286, Cambridge, MA.

Nageswaran, J. M., Dutt, N., Krichmar, J. L., Nicolau, A., & Veidenbaum, A. V. (2009). A configurable emulation environment for the efficient emulation of large-scale spiking neural networks on graphics processors. *Neural Networks*, *22*, 791–800.

Nowotny, T. (2010). Parallel implementation of a spiking neuronal network model of unsupervised olfactory learning on Nvidia CUDA. In *WCCI 2010 IEEE world congress on computational intelligence* (pp. 3238–3245).

NVIDIA, (2008). CUDA Programming Guide 2.0.

Oorschot, D. E. (1996). Total number of neurons in the neostriatal, pallidal, subthalamic, and substantia nigral nuclei of the rat basal ganglia: a stereological study using the cavalieri and optical disector methods. *Journal of Computer Neurology*, *366*, 580–599.

Otsuka, T., Abe, T., Tsukagawa, T., & Song, W. J. (2004). Conductance-based model of the voltage-dependent generation of a plateau potential in subthalamic neurons. *Journal of Neurophysiology*, *92*, 255–264.

Pasupathy, A., & Miller, E. K. (2005). Different time courses of learning-related activity in the prefrontal cortex and striatum. *Nature*, *433*, 873–876.

Pfeifer, R., & Scheier, C. (1999). *Understanding intelligence*. Cambridge, MA: The MIT Press.

Plenz, D., & Kitai, S. (1999). A basal ganglia pacemaker formed by the subthalamic nucleus and external globus pallidus. *Nature*, *400*, 677–682.

Rotter, S., & Diesmann, M. (1999). Exact digital emulation of time-invariant linear systems with applications to neuronal modeling. *Biological Cybernetics*, *81*, 381–402.

Samejima, K., Ueda, Y., Doya, K., & Kimura, M. (2005). Representation of action-specific reward values in the striatum. *Science*, *310*, 1337–1340.

Scorcioni, R. (2010). GPGPU implementation of a synaptically optimized, anatomically accurate spiking network simulator. In *Biomedical sciences and engineering conference*.

Shouno, O., Takeuchi, J., & Tsujino, H. (2009). A spiking neuron model of the basal ganglia circuitry that can generate behavioral variability. *The Basal Ganglia IX*, *58*, 191–200.

Terman, D., Rubin, J. E., Yew, A. C., & Wilson, C. J. (2002). Activity patterns in a model for the subthalamopallidal network of the basal ganglia. *Journal of Neuroscience*, *22*, 2963–2976.

Tremblay, L., Hollerman, J. R., & Schultz, W. (1998). Modifications of reward expectation-related neuronal activity during learning in primate striatum. *Journal of Neurophysiology*, *80*, 964–977.

Turner, R. S., & Anderson, M. E. (2005). Context-dependent modulation of movement-related discharge in the primate globus pallidus. *Journal of Neuroscience*, *25*, 2965–2976.

Vogelstein, R. J., Mallik, U., Culurciello, E., Cauwenberghs, G., & Etienne-Cummings, R. (2007). A multichip neuromorphic system for spike-based visual information processing. *Neural Computation*, *19*, 2281–2300.

Yudanov, D., Shaaban, M., Melton, R., & Reznik, L. (2010). GPU-based simulation of spiking neural networks with real-time performance & high accuracy In *WCCI 2010 IEEE world congress on computational intelligence* (pp. 2143–2150).