Using Games to Embody Spiking Neural Networks for Neuromorphic Hardware

Corey M. Thibeault *
HRL Laboratories LLC. Malibu, CA, 90265 USA

Frederick C. Harris, Jr. †
University of Nevada at Reno, Reno, NV 89557 USA

Narayan Srinivasa *
HRL Laboratories LLC. Malibu, CA, 90265 USA

Abstract

Adding value to action-selection through reinforcement-learning provides a mechanism for modifying future decisions of real and artificial entities. This behavioral-level modulation is vital for performing in complex and dynamic environments. In this paper we focus on three classes of biologically inspired feed-forward spiking neural networks capable of action-selection via reinforcement-learning embodied in a minimal virtual agent. Their ability to learn two simple games through reinforcement and punishment is explored under varying levels of noise and feedback. There is no bias or understanding of the task inherent to the networks and all of the dynamics emerge based on environmental interactions. Value of an action takes the form of reinforcement and punishment signals assumed to be provided by the environment or a user. The variation in the four classes arises from different levels of network complexity based on differences in network architecture, the nature of network interactions including the interplay between excitation, inhibition and reinforcement, and the degree of bio-fidelity of the model. A novel aspect of these models is that they obey the constraints of neuromorphic hardware that are currently under development, including the DARPA SyNAPSE neuromorphic chips for very low power spiking model implementations. The simulation results demonstrate the performance of these models for a variant of classic pong as well as a first-person view selection task. Embodying models in games allows for the creation of environments with varying levels of detail that are ideal for testing spiking neural networks. In addition, the performance results suggest that these models could serve as building blocks for the control of more complex robotic systems that are embodied in both virtual and real environments.

Key Words: Embodied modeling, neurorobotics, spiking neural networks, action-selection, reinforcement-learning, neuromorphic hardware.

*Center for Neural and Emergent Systems, Information and System Sciences Laboratory, Email: {cmthibeault, nsrinivasa}@hrl.com
†Department of Computer Science and Engineering. Email: Fred.Harris@cse.unr.edu

1 Introduction

The combination of action-selection and reinforcement-learning in biological entities is essential for successfully adapting and thriving in complex environments. This is also important for the effective operation of intelligent agents. However, strategies for embedding artificial intelligence have resulted in agents with limited demonstrable emergent properties. Because of this, it is still unreasonable to deploy a neurorobotic entity and expect it to learn from and perform in its environment the same way biological entities can. Similarly, neural models require complex and varied input signals in order to accurately replicate the activity observed experimentally. One strategy for creating this complex stimuli is through immersing a model in a real or virtual environment capable of providing the feedback necessary for the model to extract value and interact appropriately. These are part of the motivations for the DARPA SyNAPSE program [13, 21]. Through the creation of low-power neuromorphic architectures both suitable for efficient remote operation and capable of replicating many of the biologically salient features of neural systems, the program can reduce the technological and theoretical barriers of embodied modeling.

Embodied modeling can be described as the coupling of computational biology and engineering. This can be accomplished in many different ways but games are one of the most beneficial for exploring those. The varying levels of complexity combined with quantifiable performance result in environments appropriate for testing many different levels of biological fidelity. Two of the most basic aspects of playing those games are action-selection and reinforcement learning. These are important for making decisions based on past experience to achieve the desired outcomes.

Action selection is the appropriate negotiation of competing signals. In the mammalian nervous system the complex circuitry of the Basal Ganglia (BG) is active in gating the information flow in the frontal cortex by appropriately selecting between input signals. This selection mechanism can affect simple action all the way up to complex behaviors and cognitive processing [8]. Although overly simplified, it can be helpful to relate the BG to a circuit multiplexer, actively connecting inputs

to outputs based on the current system state.

Reinforcement or reward learning (RL) is the reinforcement of actions or decisions that maximizes the positive outcome of those choices. This is similar to instrumental conditioning where stimulus-response trials result in reinforcement of responses that are rewarded and attenuation of those that are not [7]. Reinforcement-learning in a neural network is an ideal alternative to supervised learning algorithms. Where supervised learning requires an intelligent teaching signal that must have a detailed understanding of the task, reinforcement learning can develop independent of the task without any prior knowledge. Only the quality of the output signal in response to the input signal and current contextual state of the network is needed.

In this work we focus on three different classes of small biologically inspired feed-forward spiking networks capable of action-selection and reinforcement-learning while immersed in a virtual environment. Each is suitable for realization on the neuromorphic hardware developed under the SyNAPSE project and provides a theoretical framework for testing future novel reinforcement-learning algorithms. These networks are embodied in a minimal virtual agent and their ability to learn a simple ping-pong game through reinforcement and punishment is explored. There is no bias or understanding of the task inherent to the network and all of the dynamics emerge based on interactions with the environment. Value of an action takes the form of simple reinforcement and punishment signals. This concept is then extended by exploring how these networks can be combined to perform more complex actions. Towards this goal, a first-person view environment was developed. A model combining multiple RL networks was then constructed and trained to identify the most appropriate object in its environment.

In addition to supporting hardware validation, the resulting models are ideal for simple robotic embodiments and are capable of demonstrating reinforcement-learning and action-selection in different ways. Similarly, the two games developed for testing these networks illustrate the utility of embodied modeling in competitive environments.

There have been a number of research efforts aimed at utilizing games to explore action-selection and reward learning. For instance, Wiles *et al.* [25] developed a spiking neural model to control a rat animate performing phototaxis. The network was constructed to perform the task similar to a Braitenberg vehicle. Burgsteiner *et al.* [5] created a liquid state machine using a recurrent network with fixed internal synapses and plastic output synapses that learned a similar task.

The model of Arena *et al.* [1] consisted of three layers of Izhikevich neurons to control a virtual robot with several sensory modalities. The networks were constructed with an initial understanding of how to process low-level sensor input such as proximity and contact sensors as well as visual cues. These were used to direct the robot through the environment. Simultaneously, the network learns to perform this navigation using range-finding sensors. The inherent low-level sensors essentially train the network on how to respond to the high-level sensors.

Florian *et al.* [9] evolved a fully recurrent spiking neural network to control a simple virtual agent to seek out, push and release balls in its environment. An evolutionary algorithm was used to calculate the synaptic weights of the network to accomplish the task.

Barr *et al.* [3] implemented a mode of the basal ganglia on a neural processor array. Although not directly demonstrated in the hardware presentation, the original software neural model was capable of performing action selection. However, there are no inherent mechanisms for reinforcement-learning and the micro-channels of the basal ganglia were predefined by the network.

Merolla *et al.* [13] presented a neuromorphic processor capable of playing a game of pong against a human opponent. This description was later extended by Arthur *et al.* [2]. The network was constructed off-line and once programmed on the hardware remained static. In that, a neural network, consisting of 224 neurons, that could also play a pong style game was created. The network was constructed off-line and was demonstrated on a neuromorphic processing core. Training involved teaching the network to predict different patterns of motion by the puck. Rather than simply tracking it, like the networks here, the model would plan where the paddle must be placed. The resulting networks, however, are specialized for that task and can not adapt to changing environments once embodied in hardware.

2 Design and Methods

2.1 Neuron Model

The neural model supported by the initial SyNAPSE hardware is the Leaky-Integrate and Fire (LIF) neuron. The LIF model is defined by

$$C_m \frac{dV}{dt} = -g_{leak}(V - E_{rest}) + I. \quad (1)$$

where

C_m	is the membrane capacitance.
I	is the sum of external and synaptic currents.
g_{leak}	conductance of the leak channels.
E_{leak}	is the reversal potential for the background leak currents.

As the current input into the model neuron is increased the membrane voltage will proportionally increase until a threshold is reached. At this point an action potential is fired and the membrane voltage is reset to the resting value. The neuron is placed in a refractory period for two milliseconds where no changes in the membrane voltage are allowed. If the current is removed before reaching the threshold, the voltage will decay to E_{rest}. The LIF model is one of the least computationally

intensive neural models but is still capable of replicating many aspects of neural activity [6].

The connections between neurons are modeled by conductance-based synapses. The general form of that influence is defined as

$$I_{syn} = g_{max} \cdot g_{eff} \cdot (V - E_{syn}). \quad (2)$$

where

g_{max}	is the maximum conductance for the class of synapse.
g_{eff}	is the current synaptic efficacy between $[0, 1]$.
E_{syn}	is the reversal potential for that particular class of synapse.

Although the synapses are conductance based the buffering and reuptake of neurotransmitter is treated as a pulse event lasting one time step. In that way it is similar to current based synapse. For numerical integration An Euler method is used with time step $\tau = 1ms$.

Learning at the synaptic level is achieved through the spike-timing dependent plasticity rules defined by Song *et al.* [19]:

$$\dot{g}_{eff} = P_{ij} X_i(t) - D_{ij} X_j(t - \Delta_{ij}) \quad (3)$$

$$\dot{P}_{ij} = -\frac{P_{ij}}{\tau_+} + A_+ X_j(t - \Delta_{ij}) \quad (4)$$

$$\dot{D}_{ij} = -\frac{D_{ij}}{\tau_-} + A_- X_i(t), \quad (5)$$

where $X_j(t)$ is the spike train of neuron j defined as a sum of Dirac functions over the action potential times $t_j^{\mathrm{AP}_k}$ equal to $\sum_k \delta(t - t_j^{\mathrm{AP}_k})$, P_{ij} is the potentiation, modeling the influence of incoming spikes, and D_{ij} is the depression value, tracking the influence of outgoing spikes. The global parameter values used in this study are presented in Table 1.

Table 1: Global model parameters

Parameter	Value	
C_m	1.	(pF)
E_{exc}	0.	(mV)
E_{inh}	$-80.$	(mV)
V_{rest}	0.	(mV)
A_+	0.025	
A_-	0.026	
τ_+	20.	(ms)
τ_-	20.	(ms)

The spiking neural networks were simulated using the HRLSim™ package [14]. HRLSim™ is a distributed CPU and GPGPU spiking neural simulation environment. HRLSim™ was developed to support the modeling aspects of the SyNAPSE project. It has also been effective in general neural simulation studies [15, 20, 22, 23]. The experiments in HRLSim™ are defined in C++—providing higher performance as well as compile and run time optimizations. In addition, embodying of models can be accomplished using different mechanisms; including compiling the environment directly into the experiment.

2.2 Networks

Three possible embodied networks are presented here. Initially, each of these networks have no knowledge or inherent understanding of their environment. The behavior is learned through feedback from the environment in the form of reward and punishment signals that can be encoded as either random or structured spike events. These strengthen or weaken the synaptic connections between neurons.

2.2.1 Excitatory Only Network. The first model explored was a simple feed-forward network that consists of 70 excitatory neurons, Figure 1. The input and output layers are divided into channels represented by populations of neurons. The connections between the layers are diffuse and are randomly established— resulting in no intentional bias between channels. The parameters are presented in Table 2. Note that each output neuron receives a maximum of 16 inputs.

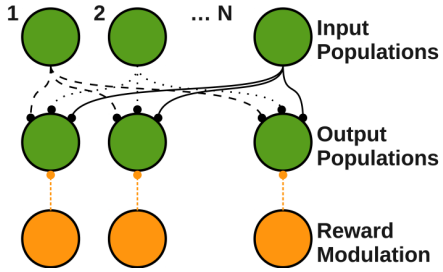

Figure 1: Excitatory neuron only network

Table 2: Parameters for the excitatory only network

A. Neuron parameters

Neural Region	Neurons Per Channel
Input	3
Output	3
Reward	1

B. Connections

Source → Destination	Synaptic Conductance $(g_{max}) \cdot (g_{eff})$	Number of Incoming Connections (total)
Input → Output	$(10.0) \cdot (0.25)$	15
Reward → Input	$(10.0) \cdot (1.0)$	1

2.2.2 Lateral-Inhibition Network. As an extension of this idea, lateral inhibition between the output populations is added,

as shown in Figure 2. These diffuse connections create an on-center off-surround activity pattern where the most active population suppresses the other output populations. The parameters of this model are included in Table 3.

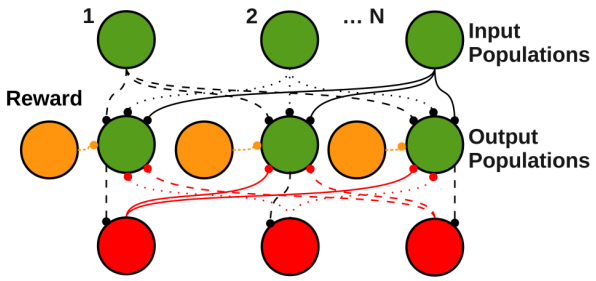

Figure 2: Lateral-inhibition network

Table 3: Parameters for the lateral-inhibition network

A. Neuron parameters

Neural Region	Neurons Per Channel
Input	3
Output	3
Inhibition	3
Reward	1

B. Connections

Source → Destination	Synaptic Conductance $(g_{max}) \cdot (g_{eff})$	Number of Incoming Connections (total)
Input → Output	$(10.0) \cdot (0.25)$	15
Output → Inhibition	$(10.0) \cdot (1.0)$	15
Inhibition → Output	$(10.0) \cdot (1.0)$	15
Reward → Input	$(10.0) \cdot (1.0)$	1

2.2.3 Basal Ganglia Direct Pathway. The third network presented is an implementation of the direct pathway of the BG, Figure 3. This network emulates the physiological activity of the rodent BG direct pathway where the neurons of the SNr are tonically active, firing around 30 Hz. This basal activity is suppressed by the inhibitory afferents of the striatum, resulting in a disinhibitory mechanism of action. Learning occurs between the cortex and the neurons of the striatum to develop the appropriate input-output channel combinations.

Physiologically neurons in the SNr are tonically active, however, the LIF neuron is not capable of replicating that spontaneous activity. To compensate, a Poisson random excitatory input is introduced into the SNr populations. In addition, low-level uniform random noise is injected into the membrane voltage of the neurons in the network. The parameters of this model are included in Table 4.

2.2.4 Combined Reward Learning Network. Although simple, these networks are capable of distinguishing competing inputs under noisy conditions. They can also be used as building blocks to perform more complex tasks. To illustrate this concept we combine three of the lateral-inhibition networks of Section 2.2.2. Each network is divided into multiple channels with the outputs of two of the channels directly connecting to the corresponding input channel of the third, Figure 4. The connections are made one-to-one at a weight of 0.5, with output channel 1 connected to input channel 1, output channel 2 to input channel 2, and so on.

As illustrated in Figure 4, each network receives a different input signal. Through reinforcement the networks can learn to appropriately respond to different combinations of inputs. Here, these are used to dynamically solve a first-person view identification task, described below.

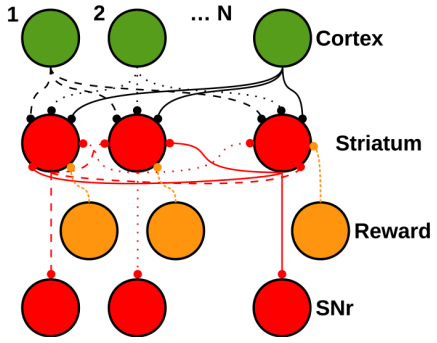

Figure 3: basal ganglia direct pathway network.

Table 4: Parameters for the basal ganglia direct pathway

A. Neuron parameters

Neural Region	Neurons Per Channel
Cortex (Ctx)	4
Striatum (Str)	3
Substania Nigra pars reticulata (SNr)	3
Excitatory	9
Reward	6

B. Connections

Source → Destination	Synaptic Conductance	Number of Incoming Connections (per channel)
Ctx → Str	0.1	4
Str → Str (diffuse)	10.0	3
Excitatory → SNr	0.08	3
Str → SNr	10.0	3
Reward → Str	10.0	6

2.2.5 Stimulus Learning. Learning in the experiments is driven by a conditioned stimulus injection. Stereotyped spiking signals are sent to an input population and the reward populations. The timing of the signal is delayed for the target channel so the synaptic weights between the input populations and the desired output populations are potentiated, while all

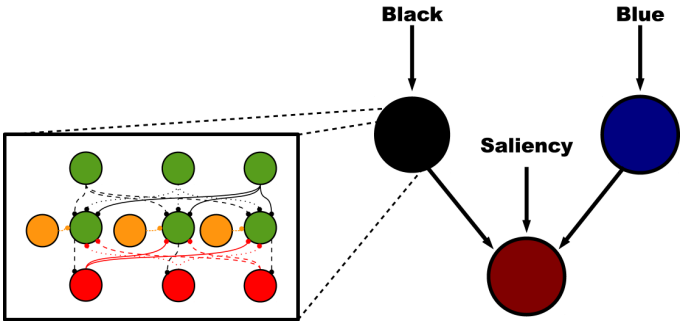

Figure 4: First-person view identification control network. Three lateral-inhibition networks from Section 2.2.2 are combined to perform more complex tasks

other channels are depressed. The stimulus period lasts for either 300 or 500 ms depending on the experiment.

2.3 Games

2.3.1 Pong. To illustrate the capabilities of these networks a pong style virtual environment was implemented. This version of the game has a single player controlling the paddle at the bottom of the board. The puck bounces off of the left, right and top walls with minimal physics that change the speed of the puck based on the angle of incidence with the wall. The player has to move the paddle to block the puck from falling through the bottom of the game board.

The game was developed in different stages. First, A mock-up of the game was created in Python using PyGame [18]. A game controller was then developed in C++. However, the controller has no visualization capabilities. It compiles directly into the HRLSim experiment and provides the virtual environment for the networks. The output of the environment is recorded by the controller and can then be played back by the Python visualizer.

The position of the puck in the game space is encoded and sent to a number of discretized neural channels. Each of these channels represents a vertical column of the game board. The input signal is Poisson random spike events with a rate determined by a Gaussian curve, described below. This provides a noisy input signal with overlap between channels. The networks signal, through a winner-takes-all mechanism, the position of the paddle.

The location of the puck on the map determines the peak amplitude and center of a Gaussian function defined as

$$f_{X_c}(b) = ae^{-\left((X_c-b)^2/2c^2\right)} \tag{6}$$

where

a	Peak amplitude of the Gaussian function,
b	Center of the Gaussian function,
c	Spatial width or σ of the Gaussian function,
X_c	The non-dimensional location of the channel.

The peak amplitude and Gaussian center are defined as

$$a = Y^* \cdot R_{max} \tag{7}$$

$$b = X^* \tag{8}$$

where

Y^*	Non-dimensional location of the puck in the y dimension,
R_{max}	Maximum input stimulus in $^{Spikes}/_{s}$,
X^*	Non-dimensional location of the puck in the x dimension.

This is visualized in Figure 5 for a spatial width of $c = 0.035$. The reward or punishment to the network arrives when the puck reaches the bottom of the game board.

2.3.2 Pong Controller. The paddle is controlled by a simple proportional controller. The environment receives discrete locations from the neural network. The location on the screen that the paddle has to move to is calculated based on these discrete locations. Its velocity in the X direction is defined by

$$V_x = V_{max} \cdot P. \tag{9}$$

The variable P is the output of the proportional controller defined by

$$P = k \cdot e. \tag{10}$$

Where k is the gain variable and e is the error between the target and current locations

$$e = X_{Location} - X_{Target} \tag{11}$$

The output of the proportional controller, P, is a piecewise linear function that is dependent on the distance from the target.

$$P = \begin{cases} -1 & -e < -\frac{1}{k} \\ 1 & e > \frac{1}{k} \\ e - k & |e| \leq \frac{1}{k} \end{cases}$$

This ensures that the speed of the paddle does not exceed the maximum defined velocity. The pivot point $\frac{1}{k}$ is calculated by setting $k \cdot e = 1$. In addition, the proportionality constant k is less than 1 to ensure that the paddle slows down as it gets closer to its target.

2.3.3 First-Person View Identification Task. The first-person view (FPV) identification task is modeled after some of the original first-person games; such as [12]. In this implementation the player moves along a specified path controlling its forward movement, where its attention is focused laterally, and selecting which object in its view is important. Similar to the pong environment described above this was implemented at different levels of abstraction. The game engine and visualization was developed in Python, with the latter using PyGame [18] and the Pyggel library [16]. The

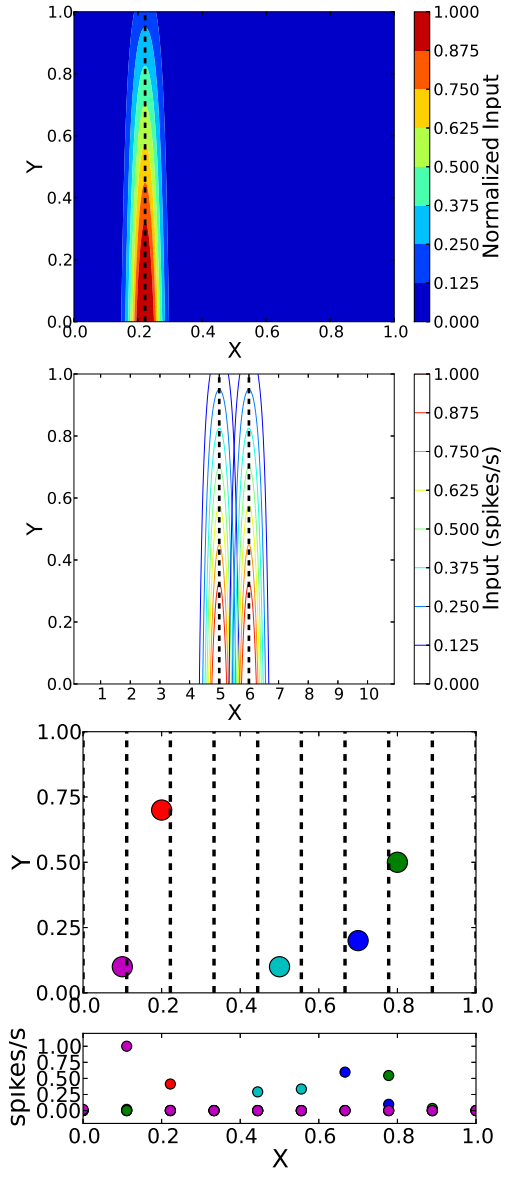

Figure 5: Pong game board disrectizations for a 10 channel network. The spatial width, c, is 0.035. In the top row is the stimulus maps for channel 2. The middle row illustrates the overlap between two consecutive channels. The bottom row shows how the location of the puck (top) translates to input stimulus for each of the 10 channels (bottom)

game engine is separated from the visualization to facilitate faster simulations and communication with the simulations is provided through a socket server. The engine and the simulation are synchronized—performance is determined by the slowest component.

There are two types of game elements in the current version—black blocks that the player must target and blue blocks the player must ignore. Each of these creates a different input into the black and blue channels respectively. It is assumed that a separate mechanism identifies the element and determines which channel is stimulated. The arena is a square track with equal width, Figure 6. As the player moves through the environment game elements enter into the view of the player. Elements in the players POV are picked up and their location in that view creates the input stimulus injected into the saliency channels of the network.

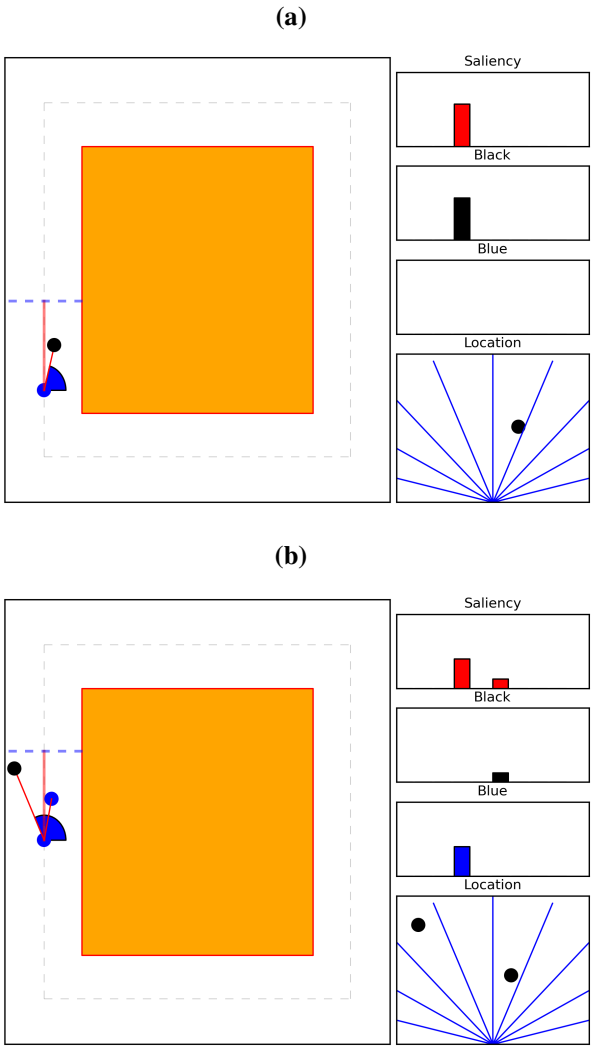

Figure 6: Example stimulus encoding and FPV game board. (a) Target element only. (b) Target and benign elements

The game board is discretized based on the player's perspective. The hemispherical point-of-view (POV) for the player is partitioned into a rectangular region, Figure 7a. The POV is then segmented into discrete channels with centers at equally spaced angles along the hemisphere, Figure 7b. This defines the center for each of the channels that are represented by the network, Figure 7c. The channels create a pie-shaped region of interest, Figure 7d, which have arc lengths with a 10 percent overlap between channels, Figure 7e. Each of the segments defines that channels stimulus map, which is described

by a Gaussian function, Figure 7f.

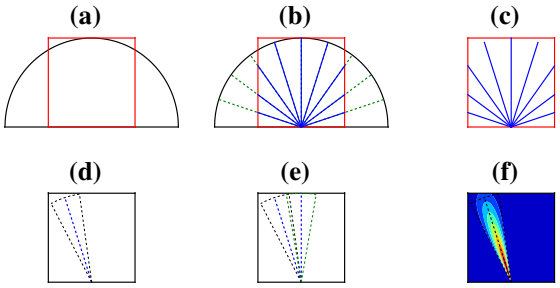

Figure 7: FPV Discretization. (a) A rectangular frame is taken from the hemispherical point-of-view (POV) of the player. (b) The POV space is discretized into equal segments (channels). (c) The resulting frame segments the players view of the world. (d) Each of the channels is centered along equal angular steps about the space with arc-lengths defining the stimulus regime for that channel. (e) The channels are constructed with overlapping stimulus regions to create a noisier environment for the networks to negotiate. (f) The stimulus space for a single channel is defined by a Gaussian function that is railed to the segment boundary

$$f_{\Theta_c}(b) = ae^{-\left((\Theta_c - b)^2/2c^2\right)} \tag{12}$$

where

a	Peak amplitude of the Gaussian function.
b	Center of the Gaussian function.
c	Spatial width or σ of the Gaussian function.
Θ_c	The non-dimensional angular location of the channel.

The peak amplitude and Gaussian center are defined as

$$a = r^* \cdot R_{max} \tag{13}$$

$$b = \Theta^* \tag{14}$$

Where

r^*	Non-dimensional location of the element in the radial dimension.
R_{max}	Maximum input stimulus in $Spikes/s$.
Θ^*	Non-dimensional angle of the element relative to the player.

For this implementation the game engine directs the stimulus. Figure 6a illustrates the stimulus for a black element in the players POV. Figure 6b illustrates what the stimulus for the two different game elements would be.

3 Results

Initially, all of the networks are randomly constructed and, in the absence of feedback, the synaptic weights change randomly. When immersed in the game environments the weights form input-output pairs with maximal synaptic efficacy. How rapidly those pairs are formed as well as how effective the network is under these tasks will affect how it performs in the environment.

3.1 Pong Performance Analysis

There are a number of additional factors that determine the network's performance in the game tasks. The first is the spatial width of the Gaussian stimulus curve, c. This affects the overlap between channels—the larger the value of c the larger the overlap. For testing we use three spatial widths: 0.025, 0.035, 0.045. The next factor is the peak of the Gaussian stimulus curve—the larger the value the more active the input channels become. Two input peaks, R_{max}, are used, 10 Hz and 40 Hz. Finally, the length of reward is an important factor. This determines how long a feedback stimulus lasts and can affect the magnitude of the change in synaptic efficacy. Two values are chosen for this: 300 ms and 500 ms. For each combination of these parameters 5 simulations of 500 seconds were completed. The accuracy, $(saves/opportunities) \cdot 100$, is computed for 25 second windows and the average of 5 simulations is presented.

3.1.1 Excitatory Only Network. Figure 8 presents the results of the pong performance of the excitatory network. The network struggles when the input stimulus is too high (40 Hz results). However, for the lower activity stimulus the performance is similar throughout the spatial widths. This network benefits most from a longer reward period and in those cases not only does the variability in the 10 Hz peak input results go down but the overall performance is better throughout the spatial widths. As the peak input stimulus is raised, the results for $c = 0.025$ are comparable to the lower input stimulus. However, this performance is lost again as the spatial width is increased.

3.1.2 Lateral-Inhibition Network. The interneurons in the lateral-inhibition (LI) network suppress the overall activity of the output populations resulting in lower rates and more control than the excitatory only network. This directly affects the changes in the output weights, which are much smaller than in the excitatory case (not shown). Figure 9 illustrates the performance of the network as it learns the rules of the game.

Figure 10 presents the pong performance results. For the 10 Hz stimulus the network performs considerably better than the excitatory network results of Figure 8. The variability in the standard deviation is much lower and the overall performance is higher (not shown). However, when the peak input stimulus is higher the performance drops considerably. When the reward time is increased to 500 ms the overall performance throughout the parameter space is surprisingly consistent. The slopes in

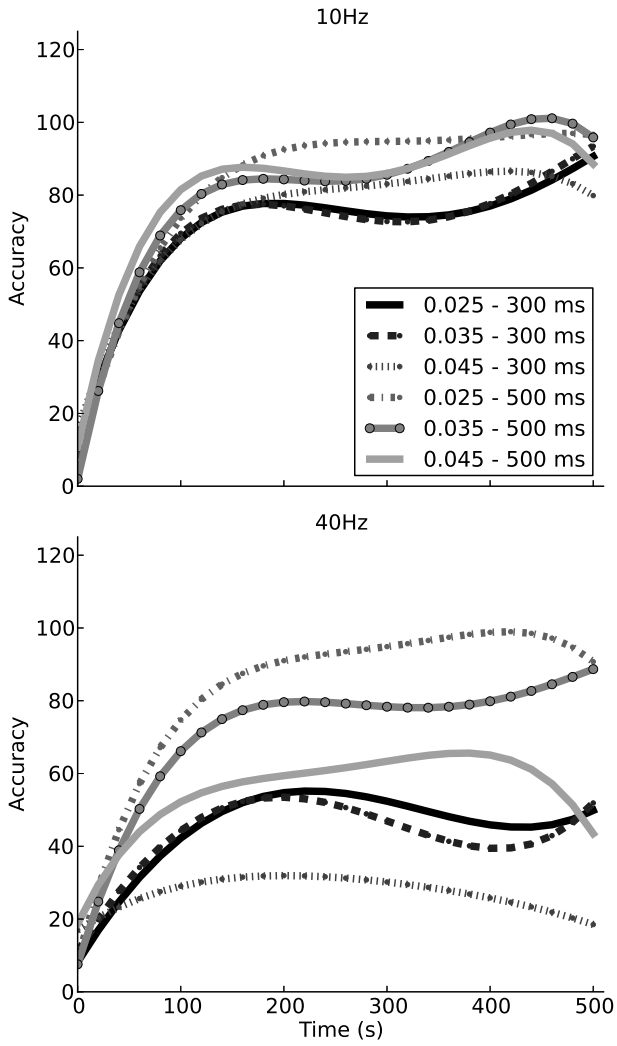

Figure 8: Excitatory network pong performance. Top: 10 Hz stimulus peak. Bottom: 40 Hz stimulus peak. The y-axis is the accuracy of the network. A value of 100 means the network blocked all of the pucks in that 25 second block

the accuracy curves are slightly different but all approach an accuracy of 100 percent.

3.1.3 Basal Ganglia Direct Pathway. In the BG direct pathway network a channel is activated through disinhibition facilitated by the Str neurons. As a consequence of this increased activity, the peak value of the input stimulus for the BG direct network needs to be higher than the other networks to sufficiently activate the desired channel. Because of this the lower input rate is set to 50 Hz and the higher rate is set to 80 Hz. With a 300 ms reward period the BG direct network performs well for small spatial widths, Figure 11. As the overlap is increased however, the performance degrades quickly. Increasing the reward period to 500 ms improves the performance of the network for the first two spatial widths, Figure 11. When the spatial width is 0.045 the performance is improved, but it fails to perform as well as the LI network.

3.1.4 Learning Capabilities. An important characteristic of this class of networks is the ability to not only learn arbitrary pairs but then later learn new ones. The rules of the game can be changed and through the same feedback mechanisms the networks will adjust to the new rules. This scenario is illustrated by the spiking activity presented in Figure 12. The stages, marked by the letters in the center are:

A. The network is initialized with all input/output connections have a synaptic USE value of 0.25; as illustrated in Figure 12a by the heat map of the average weights between input/output populations.
B. A Poisson random input is injected into consecutive channels for 10 seconds to establish the basal activity of the network. The resulting average synaptic weight matrix is shown in Figure 12b.
C. Alternating reward signals are sent to establish single input/output pairs. The weight matrix is now dominated by the diagonal shown in Figure 12c.
D. The repeated Poisson input signals from B are injected for 10 seconds. After this, the weight matrix shown in Figure 12d demonstrates further potentiation of the established input/output pairs and a continued depression of the other connections.
E. An opposite set of input/output associations are established using alternating reward signals. For stable retraining of the network the reward protocol needs to be about twice as long as the original training. The new weight matrix is shown in Figure 12e.
F. 10 seconds of the repeated Poisson inputs illustrate the newly established input/output pairs, Figure 12f.

The importance of this capability should not be overlooked. Adapting in changing environments is essential for an entity to thrive. This adaptation is similarly vital for artificial agents and for the successful deployment of neuromorphic models.

3.2 First-Person View Identification

The combination of three LI networks allows for more complex decision making. The individual networks can learn to weight different classes of input information based on reward feedback and the results can be combined to perform different tasks. In the network presented here each of the subnetworks has 9 channels, with the Black and Blue subnetworks both feeding into the action selection (AS) subnetwork. The AS subnetwork also receives saliency information from the environment.

Using the same stereotyped reward mechanisms, the FPV network can be trained to perform more complex action selection tasks, Figure 13. In this case the Black and AS subnetworks have learned a one-to-one correlation, while the Blue subnetwork has been effectively disconnected. The result is that the saliency information alone is not enough to cause the

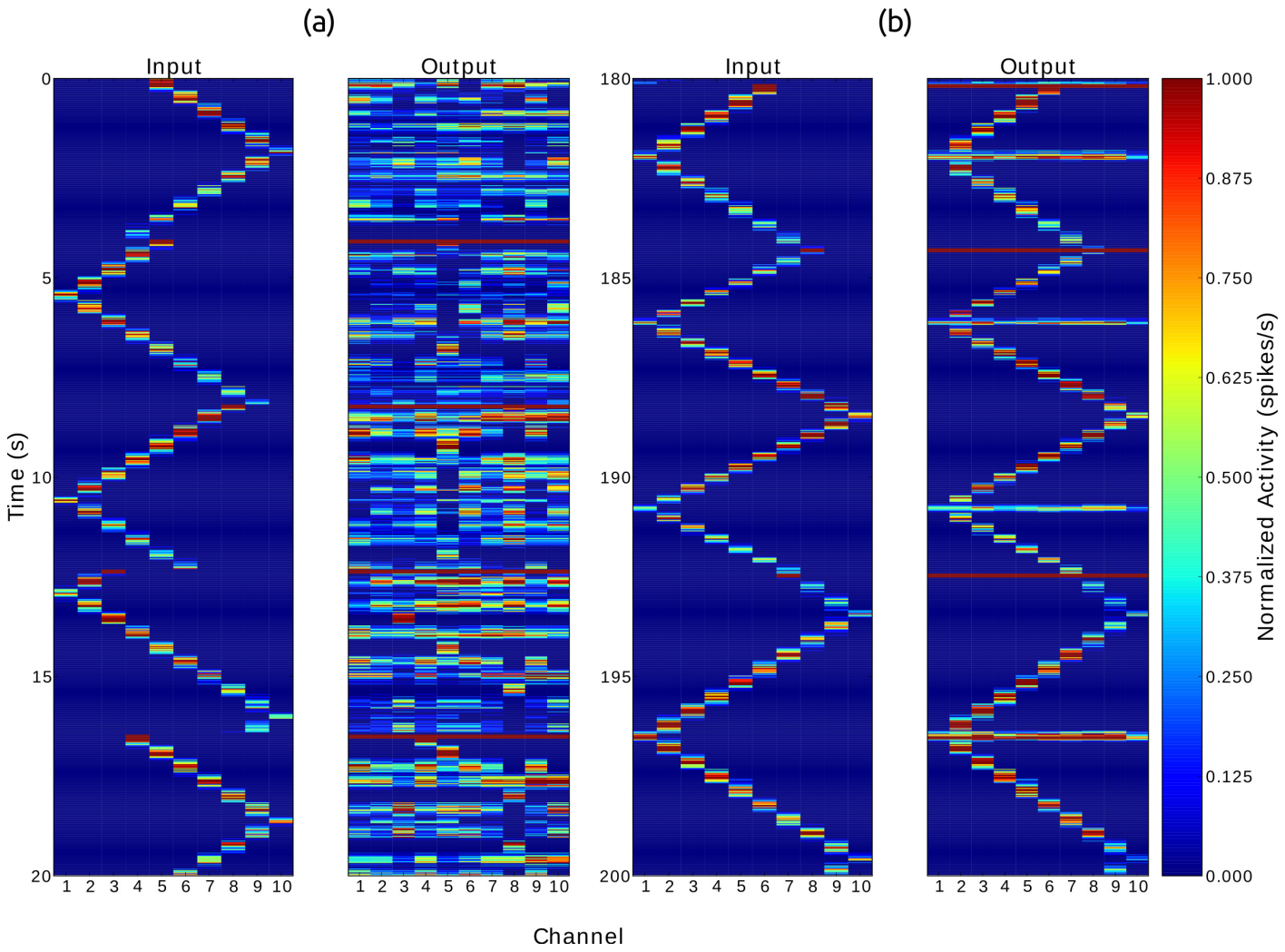

Figure 9: Lateral Inhibition network pong game play. The input signal corresponds to the position of the puck. Notice that the time course runs along the y-axis from top to bottom. The x-axis is the corresponding channel. (a) $0-20$ seconds. The network has minimal responds. (b) $180-200$ seconds. The pair correlations become more defined and the output directly follows the input

AS network to cross the selection threshold. A complementary input is required from one of the other subnetworks, in this instance only a black game element can contribute, Figure 14. The resulting network learns to ignore the blue elements while focusing on the black ones.

When placed in the FPV environment the network can move through the board and select the appropriate elements when in view. In addition, when presented with both types of game elements, the network can appropriately select the black element, even when a blue one is closer to the player, Figure 15.

4 Conclusion

4.1 Pong

The learning of channel associations is somewhat arbitrary in the examples presented here. The correlation between input and output populations can in fact be engineered to have more complex relationships than a simple pair. As illustrated by the FPV network results, other combinations can be created as well as mechanisms for more intricate information processing.

The tracking of the puck in the pong networks is reactive, with movements made based on the current position in the game. In the future this concept will be extended to include predictive control of the paddle. A recurrent network capable of learning these kinds of associations could be included along side the reactive networks presented here to achieve this. Initially all of the weights would be random. Through the feedback mechanisms demonstrated here the reactive networks can be trained to track the position of the puck. This learned behavior can then be used as an training signal to the predictive networks.

4.2 First-Person View Identification

First-person games have been extremely popular in Artificial Intelligence (AI) research. The complex interactions between

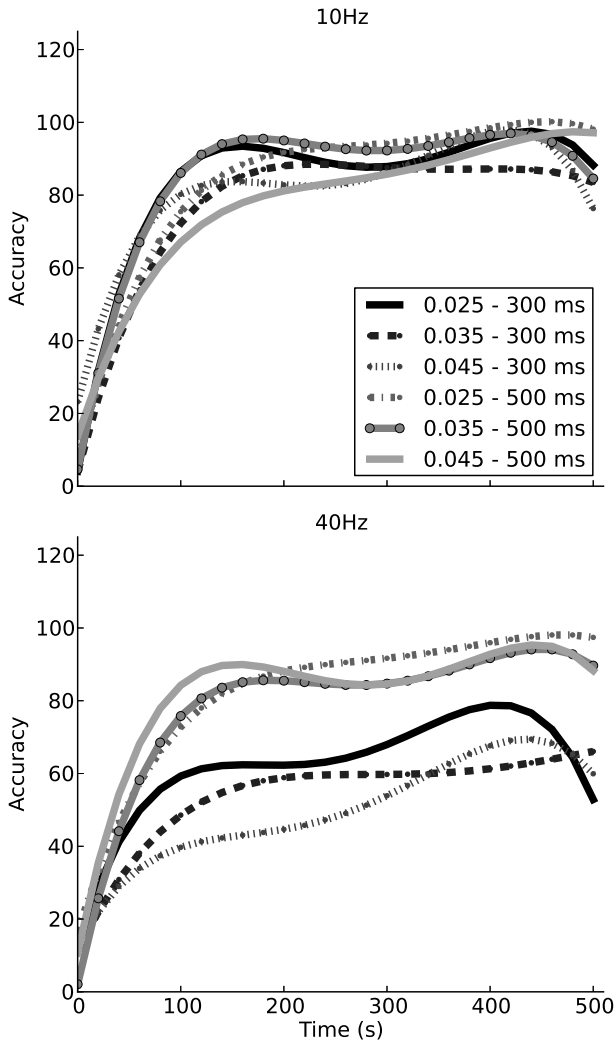

Figure 10: LI network pong performance for 300 ms reward. Top: 10 Hz stimulus peak. Bottom: 40 Hz stimulus peak. The y-axis is the accuracy of the network. A value of 100 means the network blocked all of the pucks in that 25 second block

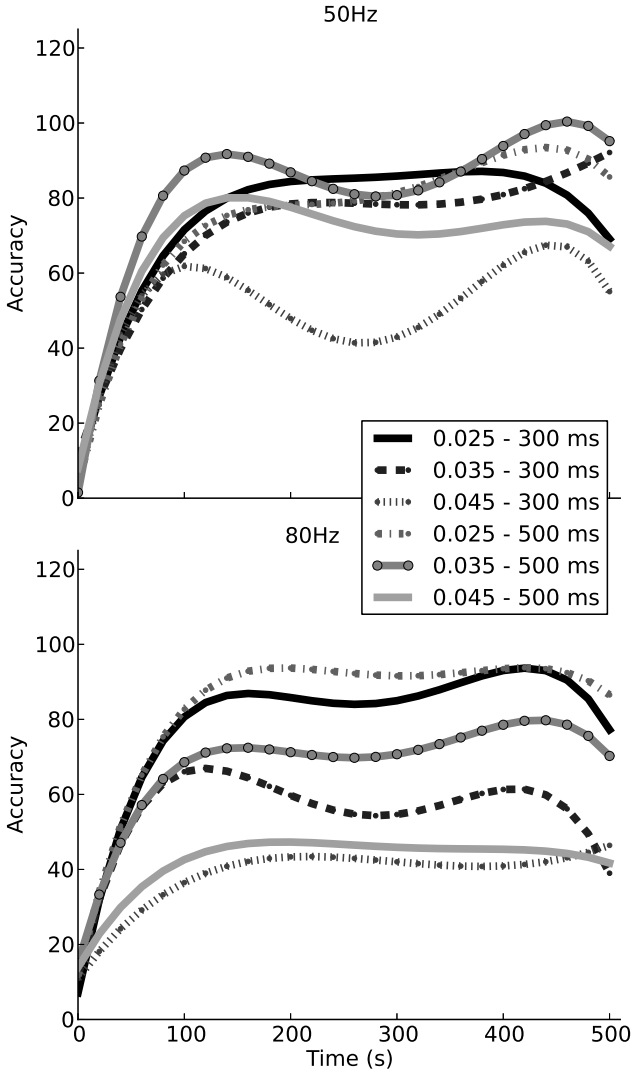

Figure 11: BG Direct pong performance. Top: 10 Hz stimulus peak. Bottom: 40 Hz stimulus peak. The y-axis is the accuracy of the network. A value of 100 means the network blocked all of the pucks in that 25 second block

the environment, game elements and multiple players, challenge non-player controllers in unique ways. This popularity has even led to competitions, such as the Botprize, where the goal is to create the most "human like" AI controller [4].

Due to the different strategies required to successfully play a modern FPV game, traditional AI domains have dominated [17, 24]. It is the complexity of the task that makes it attractive to embodied modeling. The approach taken here relies on abstracting some of that complexity away. As the networks become more capable other aspects of the FPV paradigm can be added.

4.3 Future Work

These simple feed-forward networks are a satisfactory start to employing the SyNAPSE neuromorphic architecture in embodied modeling. Alone they can be utilized as configurable controllers but their real potential lies in their use as building blocks in more complex control systems. We have already demonstrated how these can be connected together in a simple configuration but in the future these will be combined with more sophisticated networks. For example, recurrent networks can provide, through feedback, state information of the system. This basic form of short-term memory can process the temporal aspects of a system's inputs and allow for more intelligent processing.

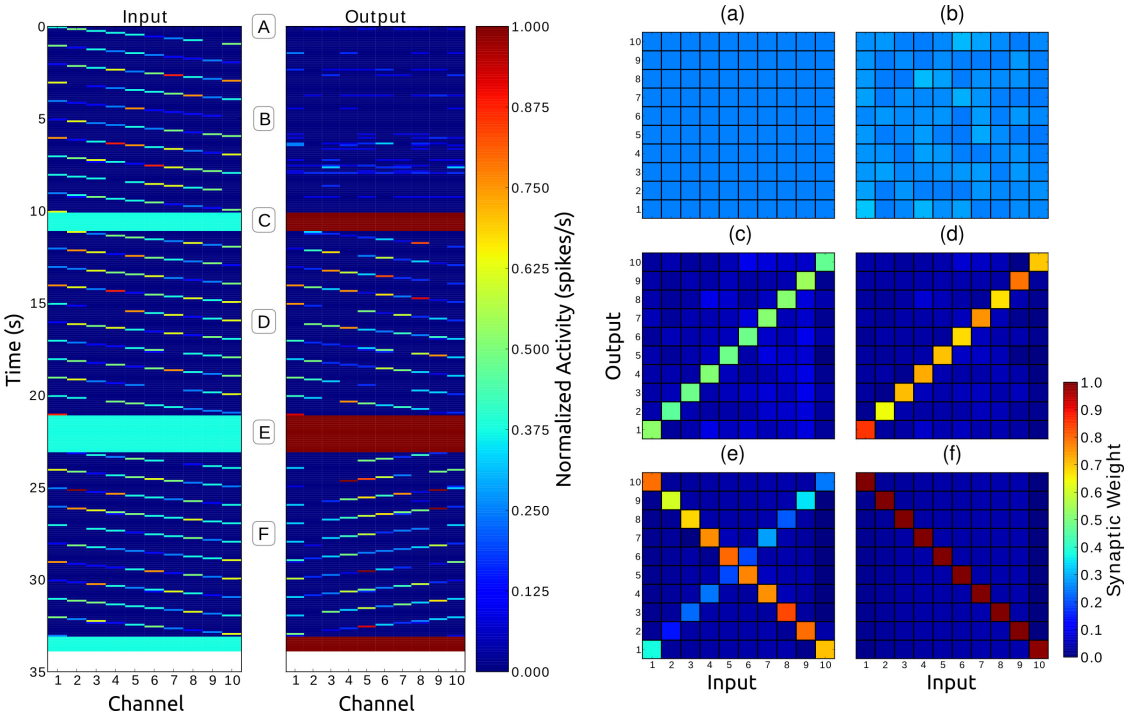

Figure 12: Left: Example lateral inhibition network reward-learning scenario. Activity rate map of the example scenario. Activity was calculated using a moving Gaussian weighted window. Right: Average and maximum synaptic weights between input/output pairs after learning. (a) 0 sec. (b) 10 sec. (c) 11 sec. (d) 21 sec. (e) 22 sec. (f) 33 sec

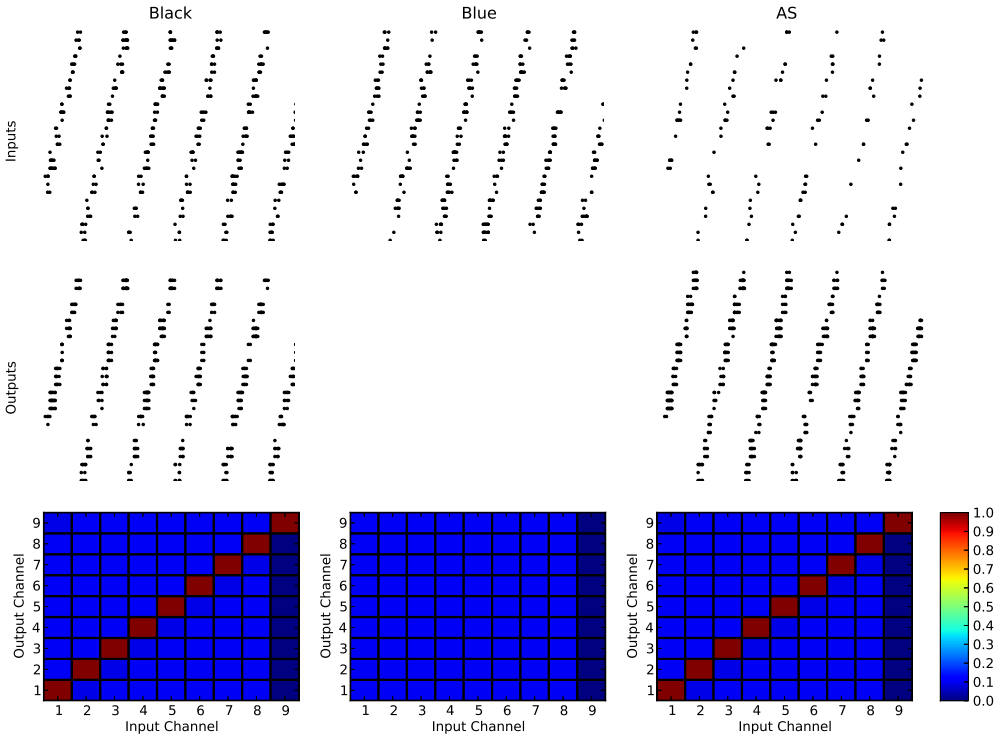

Figure 13: FPV network learning capabilities

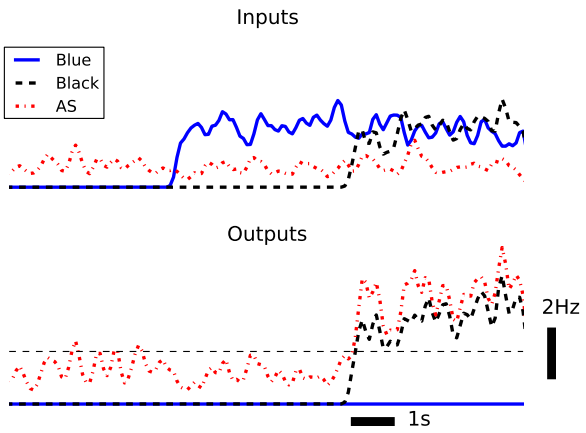

Figure 14: FPV single channel activity after training. The saliency input alone is not enough to push the AS subnetwork above the selection threshold (dashed gray line). The addition of a blue stimulus is ignored and thus does not contribute to the AS subnetwork activity. When a black element stimulus is added the activity of the AS subnetwork is driven passed the selection limit and that channel is selected

Although the performance of the BG direct pathway is slightly lower than the LI network for the tasks presented here, it is still an extremely useful building-block for future models. Physiologically, the mammalian basal ganglia achieves action-gating by removing its inhibitory influence on thalamocortical relay neurons. This allows information from higher cortical areas to pass through the thalamus to other brain areas. This type of action-gating is replicated by the BG model presented here and can perform a similar function in larger neural models.

Finally, the feedback for these networks was dependent on conditioned input stimulus to the reward modulation populations. The games played the role of the critic. In the future, more sophisticated reward and punishment signals, such as those in Florian *et al.* [10] and Friedrich *et al.* [11], will be implemented to find a generic reward critic and more efficient controllers.

Acknowledgments

The authors gratefully acknowledge the support for this work by Defense Advanced Research Projects Agency (DARPA) SyNAPSE grant HRL0011-09-C-001. This work is approved for public release and distribution is unlimited. The views, opinions, and/or findings contained in this article are those of the authors and should not be interpreted as representing the official views or policies, either expressed or implied, of the DARPA or the Department of Defense.

References

[1] P Arena, L Fortuna, M Frasca, and L Patane. Learning Anticipation via Spiking Networks: Application to Navigation Control. *Neural Networks, IEEE Transactions on*, 20(2):202 –216, Feb. 2009.

[2] J Arthur, P Merolla, F Akopyan, R Alvarez, A Cassidy, S Chandra, S Esser, N Imam, W Risk, D Rubin, R Manohar, and D Modha. Building Block of

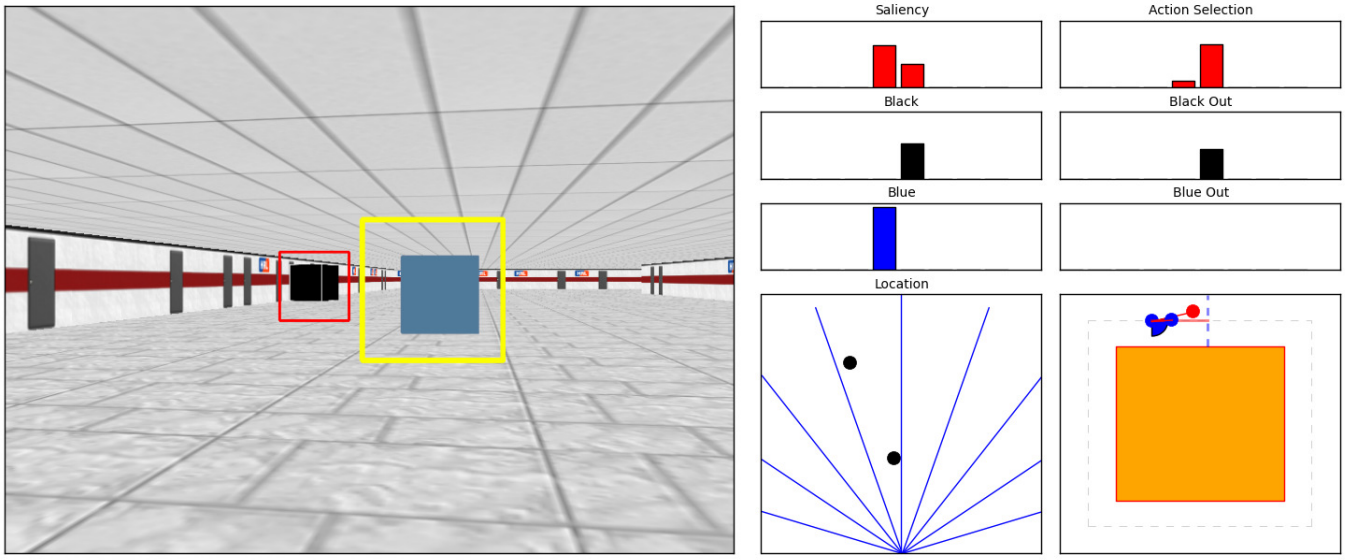

Figure 15: FPV Game Play. Although the blue cube is more salient in the player's field of vision the network appropriately selects black cube

A Programmable Neuromorphic Substrate: A Digital Neurosynaptic Core. In *Neural Networks (IJCNN), The 2012 International Joint Conference on*, pp. 1 –8, June 2012.

[3] D Barr, P Dudek, J Chambers, and K Gurney. Implementation of Multi-Layer Leaky Integrator Networks On A Cellular Processor Array. In *Neural Networks, 2007. IJCNN 2007. International Joint Conference on*, pp. 1560 –1565, Aug. 2007.

[4] BotPrize. The 2K BotPrize. `http://botprize.org/index.html`, 2012. Accessed: 10/16/2012.

[5] H Burgsteiner. Imitation Learning With Spiking Neural Networks and Real-World Devices. *Engineering Applications of Artificial Intelligence*, 19(7):741 – 752, 2006.

[6] N Burkitt. A Review of the Integrate-and-fire Neuron Model: I. Homogeneous Synaptic Input. *Biol. Cybern.*, 95:1–19, June 2006.

[7] V Chakravarthy, D Joseph, and R Bapi. What Do The Basal Ganglia Do? A Modeling Perspective. *Biological Cybernetics*, 103:237–253, 2010.

[8] M. X Cohen and M. J Frank. Neurocomputational Models of Basal Ganglia Function in Learning, Memory and Choice. *Behavioural Brain Research*, 199(1):141 – 156, 2009.

[9] R Florian. Spiking Neural Controllers for Pushing Objects Around. In S Nolfi, G Baldassarre, R Calabretta, J Hallam, D Marocco, J.-A Meyer, O Miglino, and D Parisi, editors, *From Animals to Animats 9*, volume 4095 of *Lecture Notes in Computer Science*, pp. 570–581. Springer Berlin Heidelberg, 2006.

[10] R. V Florian. Reinforcement Learning Through Modulation of Spike-Timing-Dependent Synaptic Plasticity. *Neural Computation*, 19(6):1468–1502, 6 2007.

[11] J Friedrich, R Urbanczik, and W Senn. Spatio-Temporal Credit Assignment in Neuronal Population Learning. *PLoS Comput Biol*, 7(6), June 2011.

[12] id Software. Wolfenstein 3D. `http://www.idsoftware.com/games/wolfenstein/wolf3d`, 2012. Accessed: 11/2/2012.

[13] P Merolla, J Arthur, F Akopyan, N Imam, R Manohar, and D Modha. A Digital Neurosynaptic Core Using Embedded Crossbar Memory With 45pJ per Spike In 45nm. In *Custom Integrated Circuits Conference (CICC), 2011 IEEE*, pp. 1 –4, sept. 2011.

[14] K Minkovich, C Thibeault, M O'Brien, A Nogin, Y Cho, and N Srinivasa. HRLSim: A High Performance Spiking Neural Network Simulator for GPGPU Clusters. *Neural Networks and Learning Systems, IEEE Transactions on*, 25(2):316–331, Feb 2014.

[15] M. J O'Brien and N Srinivasa. A Spiking Neural Model for Stable Reinforcement of Synapses Based on Multiple Distal Rewards. *Neural Computation*, 25:123–156, 2013.

[16] Pyggel group. Pyggel. `http://www.pygame.org/project-PYGGEL-968-.html`, 2012. Accessed: 10/22/2012.

[17] J Schrum and R Miikkulainen. Evolving Agent Behavior in Multiobjective Domains Using Fitness-Based Shaping. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, pp. 439–446. ACM, 2010.

[18] P Shinners. PyGame. `http://pygame.org/`, 2012. Accessed: 10/22/2012.

[19] S Song, K. D Miller, and L. F Abbott. Competitive Hebbian Learning Through Spike-timing-dependent Synaptic Plasticity. *Nature Neuroscience*, 3(9):919–926, 2000.

[20] N Srinivasa and Y Cho. Self-Organizing Spiking Neural Model for Learning Fault-Tolerant Spatio-Motor Transformations. *Neural Networks and Learning Systems, IEEE Transactions on*, 23(10):1526–1538, Oct 2012.

[21] N Srinivasa and J Cruz-Albrecht. Neuromorphic Adaptive Plastic Scalable Electronics: Analog Learning Systems. *Pulse, IEEE*, 3(1):51 –56, jan. 2012.

[22] N Srinivasa and Q Jiang. Stable Learning of Functional Maps in Self-Organizing Spiking Neural Networks with Continuous Synaptic Plasticity. *Frontiers in Computational Neuroscience*, 7(10), 2013.

[23] C. M Thibeault and N Srinivasa. Using a Hybrid Neuron in Physiologically Inspired Models of The Basal Ganglia. *Frontiers in Computational Neuroscience*, 7(88), 2013.

[24] N van Hoorn, J Togelius, and J Schmidhuber. Hierarchical controller learning in a First-Person Shooter. In *Computational Intelligence and Games, 2009. CIG 2009. IEEE Symposium on*, pp. 294 –301, sept. 2009.

[25] J Wiles, D Ball, S Heath, C Nolan, and P Stratton. Spike-Time Robotics: A Rapid Response Circuit for a Robot that Seeks Temporally Varying Stimuli. *Australian Journal of Intelligent Information Processing Systems*, 11(1), 2010.

Corey M. Thibeault received the B.S. degree in Computer Engineering from the Rochester Institute of Technology, Rochester, NY, USA, in 2004, the B.S. and M.S. degrees in Mechanical Engineering from Minnesota State University, Mankato, MN, USA, in 2006 and 2009, respectively, and the M.S. degree in Computer Engineering and the Ph.D. degree in Biomedical Engineering from the University of Nevada, Reno, NV, USA, in 2012. He is currently a Post-Doctoral Researcher with the Center for Neural and Emergent Systems, HRL Laboratories, Malibu, CA, USA. He is a member of IEEE and ISCA. His current research interests include computational neuroscience and biology, high-performance computing, and intelligent systems.

Frederick C. Harris, Jr. is currently a Professor in the Department of Computer Science and Engineering and the Director of the High Performance Computation and Visualization Lab and the Brain Computation Lab at the University of Nevada, Reno, USA. He received his BS and MS in Mathematics and Educational Administration from Bob Jones University in 1986 and 1988 respectively, his MS and Ph.D. in Computer Science from Clemson University in 1991 and 1994 respectively. He is a member of ACM (Senior Member), IEEE, and ISCA (Senior Member). His research interests are in parallel computation, computational neuroscience, computer graphics and virtual reality.

Narayan Srinivasa received the Ph.D. degree in Mechanical Engineering from the University of Florida, Gainesville, FL, USA, in 1994.

He is a Principal Research Scientist and Manager for the Center for Neural and Emergent Systems, Information and System Sciences Department, HRL Laboratories LLC, Malibu, CA, USA. He is currently the Program Manager and Principal Investigator for three DARPA projects. The DARPA SyNAPSE and Physical Intelligence Programs attempt to develop a theoretical foundation inspired by brain science and physics to engineer electronic systems that exhibit intelligence. The DARPA UPSIDE Program seeks to develop emerging device technologies for highly energy efficient implementations for real-world applications, including image processing. He has managed several projects for GM and Boeing solving real-world problems in the areas of sensing and control. He was a Beckman Fellow with the Beckman Institute, University of Illinois at Urbana-Champaign, Urbana, IL, USA, from 1994 to 1997. He has published 83 technical papers and holds 32 U.S. patents.

Dr. Srinivasa received the HRL Distinguished Inventor Award, the GM Most Valuable Colleague Award, the HRL Outstanding Team Award, and the HRL Chairman Award. He is a member of IEEE, INNS and AAAS.