2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Spiking Neurons Ensemble for Movement Generation in Dynamically Changing Environments

Kaname Favier, Shogo Yonekura, and Yasuo Kuniyoshi

Abstract— **Spiking neurons might play a larger role than simply as an efficient signal transmitter. Several studies have demonstrated how movements can be generated using networks of spiking neurons. However, the complexity of spiking neural networks makes their implementation difficult, and the use of spiking neurons in robotics has remained largely impractical. In this paper, we show that the addition of a single layer of spiking neurons can help improve performance on stabilization tasks in dynamically changing environments. In a one-dimensional inverted pendulum stabilization task, the spiking neurons seem to expand the space of usable parameters of the controller. Using a robot arm in 3-D space, the additional layer of spiking neurons suffices to improve performance up to 30% on an inverted pendulum stabilization task. We expect this technique to enhance performance in most stabilization tasks but also tasks that are essentially similar such as reaching tasks and posture control. We also expect the effects of this layer to be greatest when the optimal tuning of control parameters is difficult, such as when the environment is unpredictable and dynamic.**

I. INTRODUCTION

Cognition and movement generation in animals and humans are one of the fundamental touchstones against which we assess progress for many tasks in the field of robotics and machine learning. One of the building blocks of the nervous system, the spiking neuron, has been at the center of research aiming at uncovering the mechanisms responsible for some of the feats we have been observing from these biological systems.

This line of research includes neuromorphic platforms[1], which are very-large-scale integration systems used to simulate large scale networks of spiking neurons. Simulation of models of neurons can easily become computationally expensive, but through event-based computations and substantial architectural parallelism, neuromorphic platforms offer an efficient alternative to more traditional Von Neumann architectures[2], [3]. However, the applications of such platforms are oftentimes more abstract, cognitive tasks such as image processing, constraint satisfaction problems or simultaneous localization and mapping[4], [5], and remain limited in terms of movement generation.

While not necessarily implemented on neuromorphic platforms, there are multiple studies on movement generation using networks of spiking neurons. It was shown that reaching movements can be generated from a fairly large generic spiking neural network (not tuned for the specific task) by simply training a linear read-out that generates commands from the firing activity of the network[6]. Spiking neural networks can also be used to generate movement trajectories towards a goal while avoiding obstacles[7], [8]. This path planning can even be taken a step further to control a real robot arm[8].

These studies shed light on some of the computing principles that govern movement generation in a biological nervous system. However, these control systems are relatively complex and their application to more practical situations is difficult. In this paper, we build upon a novel paradigm for the application of spiking neurons to robot control[9], whose simplicity makes its implementation on most control systems a lot easier. It consists of the addition of a single layer of independent spiking neurons to a conventional controller. While spiking neuron implementations of controllers such as a PI controller exist[10], we deliberately constraint our use of spiking neurons as a ”filter layer” to focus on the intrinsic properties of the generated spikes: In a simplified setting, a stabilization task with periodically varying random perturbations, we show that the use of a Spiking Neuron Ensemble (SNE) seems to allow previously inadequate parameters to be used successfully for stabilization. Next, we demonstrate in a more practical setting, involving the control of a robot arm in 3-D space, that the use of the SNE enhances performance on a stabilization task in a dynamically changing environment.

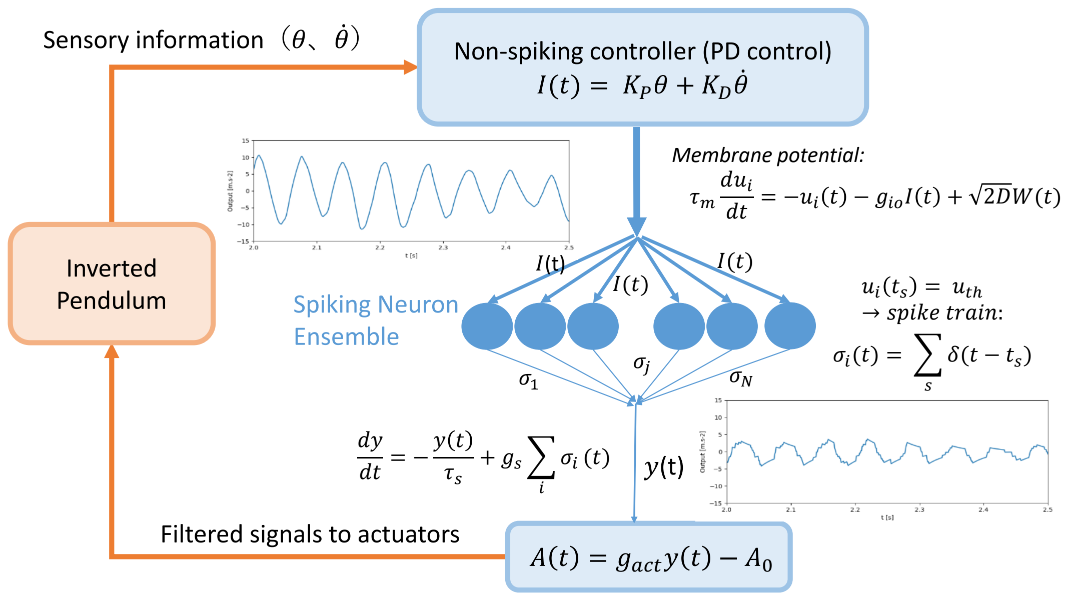
II. CONTROL MODEL, SPIKING NEURONS ENSEMBLE AS A FILTER

A. Overview of the control model

We consider a hybrid control system composed of a controller and a layer of independent spiking neurons, where the output of the controller is fed into the layer of spiking neurons. The layer is essentially an ensemble of spiking neurons that do not receive any collateral connections from other neurons and hence are solely influenced by internal noise and the common input. This architecture was first proposed in [9] and was inspired by the finite-size mean field approximation of random spiking neural networks[11], and the sparse and parallel structure of lower motor neurons innervating muscle fibers[12].

This paper is based partly on results obtained from a project commissioned by the New Energy and Industrial Technology Development Organization (NEDO), and partly on results obtained from research activity in Chair for Frontier AI Education, School of Information Science and Technology, The University of Tokyo. Furthermore, this research is partially supported by JSPS KAKENHI Grant-in-Aid for Early-Career Scientists, Grand Numbers JP18K18128.

K. Favier is with the Department of Mechano-Informatics, Faculty of Engineering, The University of Tokyo, Japan. S. Yonekura and Y. Kuniyoshi are with the Department of Mechano-Informatics, Graduate School of Information Science and Technology, The University of Tokyo, Japan {favier,yonekura,kuniyosh}@isi.imi.i.u-tokyo.ac.jp



Sensory information（θ、$\dot{\theta}$）
Non-spiking controller (PD control)
$I(t) = K_P\theta + K_D\dot{\theta}$
Membrane potential:
$\tau_m \frac{du_i}{dt} = -u_i(t) - g_{io}I(t) + \sqrt{2D}W(t)$
Inverted Pendulum
Spiking Neuron Ensemble
I(t)
$u_i(t_s) = u_{th}$
→ spike train:
$\sigma_i(t) = \sum_s \delta(t - t_s)$
σ_1
σ_j
σ_N
$\frac{dy}{dt} = -\frac{y(t)}{\tau_s} + g_s \sum_i \sigma_i(t)$
y(t)
Filtered signals to actuators
$A(t) = g_{act}y(t) - A_0$

Fig. 1: Control System. The output of a controller is fed to a filter consisting of an ensemble of independent spiking neurons

In our case, we consider a PD controller, whose output signal I is fed to an ensemble of Leaky Integrate-and-Fire (LIF) neurons. An LIF neuron is defined by the following dynamics:

$$\tau_m \dot{u} = -u + g_{io}I(t) + \sqrt{2D}W(t) \tag{1}$$

$$u = u_{th} \Rightarrow \begin{cases} t = t_s & \text{(record spike timing)} \\ u = u_r & \text{until} \quad t = t_s + \tau_{ref} \end{cases}$$

where u is the membrane potential, τ_m the membrane time constant, u_{th} the firing threshold of the neuron, t_s the time of firing, u_r the resting membrane potential, τ_{ref} the refractory period, g_{io} the input gain, D the noise intensity, and W the Gaussian input noise of unit intensity.

The ”fire and forget” dynamics of a LIF neuron (i.e. information is lost due to the reset dynamics) could also be understood as a sigmoid-like kernel with additional noise. This can be a fairly accurate model of LIF neuron dynamics in the case of constant input [13] but rapidly becomes irrelevant for time-varying inputs.

The action potentials resulting from the firing of neurons in turn act as excitatory signals σ_i to a synapse from which we obtain an activation signal A:

$$\begin{cases} \text{spike train: } \sigma_i(t) = \sum_s \delta(t - t_s) \\ \tau_s \dot{y} = -y(t) + \frac{g_s}{N} \sum_{i=1}^{N} \sigma_i(t) \\ A(t) = g_{act}y(t) - A_0 \end{cases} \tag{2}$$

Where g_s, g_{act} are gain parameters, y is the post-synaptic membrane potential, τ_s is the synaptic time constant, and N is the total number of neurons in the ensemble.

Therefore, the SNE can be seen as a signal filter rather than a controller (Fig. 1) and could be used to complement virtually any controller. Qualitatively, the filter would transform a typically smooth control signal into a stochastic and discontinuous one. Such filtering does not encode positive and negative-value inputs in the same way. To remediate this problem, similar to the presence of antagonistic muscle pairs, we use two SNEs, each encoding activation in opposite directions:

$$A_{pos}(t) = g_{act}y_{pos}(I, t) - A_0$$
$$A_{neg}(t) = g_{act}y_{neg}(-I, t) - A_0$$
$$A(t) = A_{pos}(t) - A_{neg}(t) \tag{3}$$

This controller-filter duality allows us to isolate the effects of the dynamics of LIF neurons from the controller, and the bidirectional SNEs allow a spike-based encoding of the activations. Indeed, a firing rate-based encoding in which positive and negative activations are encoded relative to a threshold firing rate is also possible (e.g. firing rates above and below 5Hz encode positive and negative outputs respectively). However, this method requires the recording of action potentials during an arbitrary time window and is maladaptive to the observation that individual spikes, as opposed to firing rates, provide valuable information[9].

B. Interpretation of the control system

The SNE can, therefore, be seen as a temporal filter of the control signal generated by the PD controller. The number of neurons N in each SNE and the synaptic time constant τ_s modulate the stochasticity of the output signal[13]. A higher number of neurons in the ensemble or a larger synaptic time constant reduce the stochasticity of the output signal. Theoretically, an infinitely large N and an adequate τ_s should completely remove the influence of internal noise on the output, making it a deterministic system. An exception is when all neurons fire synchronously, in which case the SNE's firing pattern would be closer to that of a single stochastic spiking neuron.

III. Expansion of the space of adequate control parameters

Before the inverted pendulum stabilization using a robot arm, we investigated the effect of the SNE in a simplified environment. To remove the nonlinearity and high-dimensionality associated with the use of a robot arm in 3-D space, we consider the stabilization of an inverted pendulum in 2-D space, subject to periodic perturbations.

A. Overview, inverted pendulum stabilization in 2-D

Let the inverted pendulum be a rod of length 1.5 meters, placed vertically on a flat, horizontal surface and subjected to a vertical gravitational force and a periodically changing horizontal external force at its center of mass, as well as a horizontal force at its base which is specified by the controller (Fig. 2).

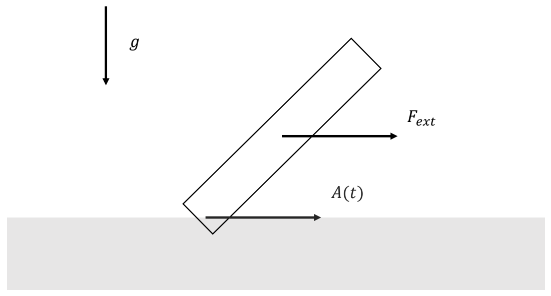


Fig. 2: Inverted pendulum stabilization in 2-D space

The system can be viewed as a one-dimensional system with an external perturbation force F_{ext} to the pendulum as in (4). The PD controller specifies the horizontal acceleration of the base $A(t) = \ddot{X}(t)$ using the angular displacement θ of the pendulum measured from a vertical position, and the angular velocity $\dot{\theta}$. Performance is defined as "the simulated time (in seconds) until the pendulum falls (vertical angle θ of the pendulum reaches $\pm\pi/2$)". The simulation time step is $\Delta t = 1.0$ms. The pendulum is initialized at varying angles from the vertical, sampled from the uniform distribution $U(-\pi/2, \pi/2)$, and with an initial angular velocity $\dot{\theta}(t = 0) = 0$.

The external perturbation F_{ext} is a constant force whose direction and amplitude were determined by sampling from the uniform distribution $U(-extF, extF)$. From now on, "Gain of external force" will refer to $extF$ which specifies the sampling space of the external perturbation force F_{ext}. Various values of $extF$ were examined. This sampling of F_{ext} was done once, every one second of simulated time.

To summarize, the whole system can be fully described by the following equation:

$$\ddot{\theta} - \frac{g}{l}\sin\theta = F\cos\theta - \frac{\ddot{X}}{l}\cos\theta \tag{4}$$

N	τ_s	g_{io}	τ_m	τ_{ref}	u_{th}	u_r
40	2.0ms	1.0	15.0ms	2.0ms	20.0mV	15.5-17.0mV

TABLE I: List of parameters set for the Spiking Neuron Ensemble

where θ is the angle of the pendulum, $\ddot{X} = A(t)$ is the acceleration at the base of the pendulum, $l = 1.5$m the length of the rod, $g = 9.81\text{m/s}^2$ the standard gravity.

Therefore, when no SNE is involved, the PD controller solely determines the acceleration of the base of the pendulum:

$$\ddot{X} = A(t) = K_p\theta(t) + K_d\dot{\theta}(t) \tag{5}$$

In the case of a SNE-driven control, the above signal $K_p\theta(t) + K_d\dot{\theta}(t)$ is given as input to the SNEs and is filtered according to (1) and (2), to produce the activation $A(t) = \ddot{X}(t)$.

B. The SNE can adapt to inadequate PD controllers

For a successful stabilization, the parameters of the PD controller need to have high enough gains but also an adequate K_p/K_d ratio. If this ratio is inadequate, even extremely high gains might not lead to stabilization. 10 seconds of stabilization time was deemed long enough to label the stabilization of the pendulum a success. We investigated how the addition of the SNEs change performance, for various ratios K_p/K_d as well as various gains of external force $extF$. We examined $extF$ ranging from 0N up to extremely high forces of 50kN (Fig. 3). These unrealistically high forces remain relevant only because of the simplicity of the system considered in this toy-problem, and no such perturbations were considered in the more realistic case using a robot arm.

All the cases considered reflected the same trend, which we describe below. All combinations of K_p and K_d taken from $\{1, 10, 10^2, 10^3, 10^4, 10^5, 3\times10^5, 5\times10^5, 7\times10^5, 10^6\}$ were examined (total of 10×10=100 cases). For the PD+SNE controller, each PD controller was tested with an output gain g_{act} from $\{1, 10^2, 10^4, 10^5, 3\times10^5, 5\times10^5, 10^6\}$ for a total of 700 cases. Among the other parameters of the SNE, N and τ_s were set at values that often worked well in our experience, and the rest was set at biologically plausible values, which can be found in Table I.

In Fig. 3, we show the results of the best PD controller found for that each of 4 K_p/K_d ratios, along with performance when a SNE layer is added that same PD controller (for various SNE output gains g_{act}). The overall best PD controller was determined manually as in most cases the best PD controller for a specific K_p/K_d showed the best results for virtually all $extF$.

Both in the case of the PD controller and the PD+SNE controller, we can observe a monotonic decrease in performance as $extF$ increases (Fig. 3). However, while for the PD controller, performance suddenly drops above a threshold $extF$, this decrease is notably smoother in the case of the PD+SNE controller, a testament to the ability of the PD+SNE controller to adapt to a wider range of external perturbations.

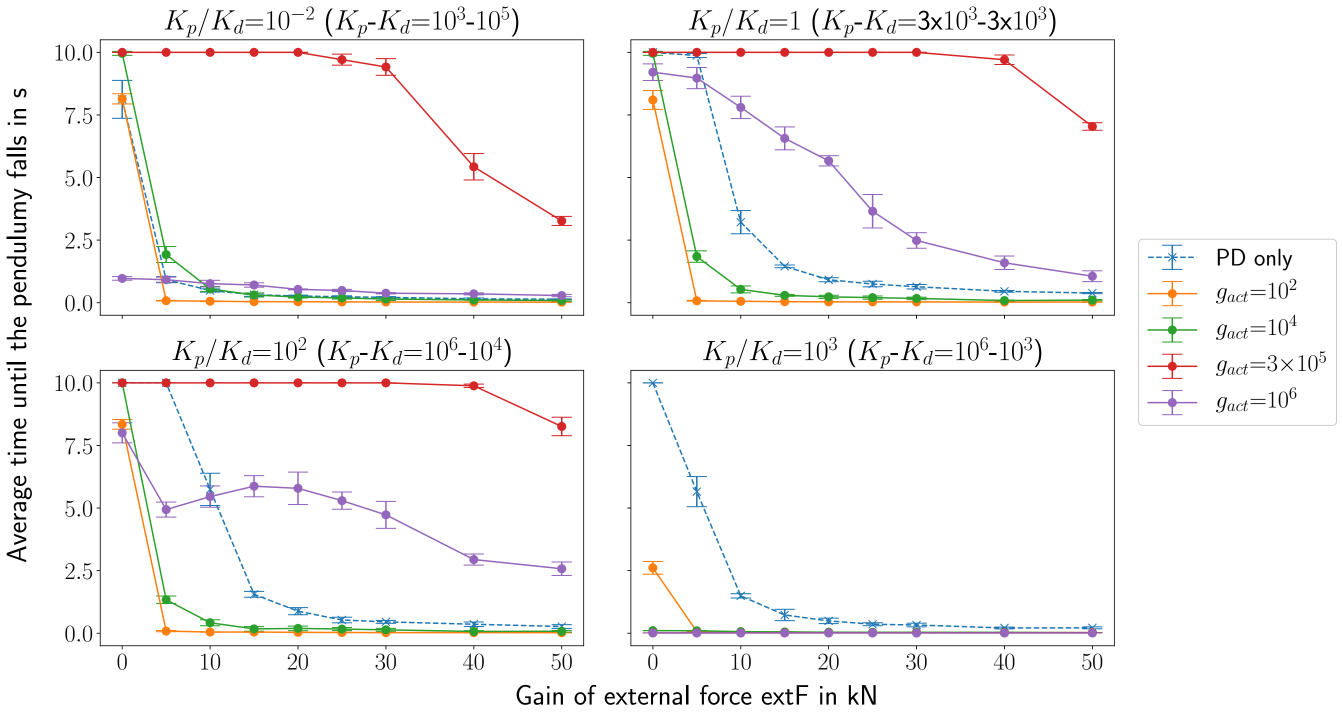


Fig. 3: Performance according to gain of external force $extF$ (in kN) for 4 K_p/K_d ratios, with and without SNE. Error bars represent standard deviations

In 3 out of the 4 cases, the additional SNEs dramatically enhances performance (K_p/K_d=10^{-2},1,10^2). Overall, an adequate K_p/K_d ratio for the PD controller seems to lie in between K_p/K_d = 10 and K_p/K_d = 100, outside of which the stabilization is only successful when no or little perturbation is involved. For the PD+SNE controller, we observed substantial improvements (similar to that seen in Fig. 3) compared to the PD controller for all ratios K_p/K_d = 10 and below. For ratios above K_p/K_d = 100, where the controller is essentially a P controller, both PD and PD+SNE controllers show poor results. For ratios between K_p/K_d = 10 and K_p/K_d = 100, both PD and PD+SNE controllers can adapt to extremely high $extF$ (above 20kN).

Therefore, the SNE seems to expand the space of adequate parameters for the PD controller, especially for ratios below K_p/K_d=10. In other words, it allows optimal control while being less demanding on the specific values of the parameters of the controller. Similar results, in a task different from the inverted pendulum, were reported in [9]. The synapse acting as the error-integration term in a PID controller is a possible explanation of the results. However, the extremely short integration time window that it would imply (1-10ms), and the "fire-and-forget" dynamics of LIF neurons make this explanation unlikely. Moreover, analytical derivations of the output of such architecture do not include any integration terms[13].

IV. Inverted Pendulum stabilization using a robot arm

In the case of a robot arm moving in 3-D space, we can expect different optimal PD control gains depending on the direction of the movement relative to the arm and depending on the initial state of the arm, as the dynamics vary significantly between a fully extended arm and one with the hand close to the body. A single PD controller with constant parameters, therefore, might not be optimal for the various states of the arm. We can expect that the lenience concerning the control parameters, granted by the use of an SNE, would work favorably and lead to overall better performance.

A. Settings

We consider an inverted pendulum, placed on the hand of a robot arm with 4 Degrees of Freedom (Fig. 4). The shoulder joint is a universal joint, the elbow and wrist joints are hinge joints with parallel rotation axes. The measurements of the arm are detailed in Table **??**. The pendulum is connected at its base to the hand with a ball joint. The pendulum is of length 0.5 meters and the dimensions of the robot are noted in Fig. 4. The inverted pendulum is initialized without angular velocity, at various angles. The mass density of each part of the arm in described in Fig. 4, with ρ_0 set at $\rho_0 = 5000 \text{kg/m}^3$.

This set-up aims at increasing the complexity of the previous task, by adding the nonlinearity and high-dimensionality associated with the use of the robot arm and the extension to 3-D space. This set-up should not be reducible to two separate 2-Dimensional problems as the dynamics of the robot arm varies significantly between the x and y axis. It also inherits the previous control architecture which specifies the acceleration of the base of the pendulum and not the torque at each joint motors. The SNEs process the signals specifying the acceleration of the base of the pendulum because filtering the signal for each joint motor would make the arm oscillate vertically. It is a well-known fact that rapid vertical oscillations can induce an inverted pendulum to stabilize[14]. An eventual performance improvement could thereby be attributed to these rapid vertical oscillations and not specifically to the activity of the spiking neurons. Therefore, the SNEs were applied to the horizontal acceleration of the hand of the robot, to circumvent the effects of vertical oscillations.

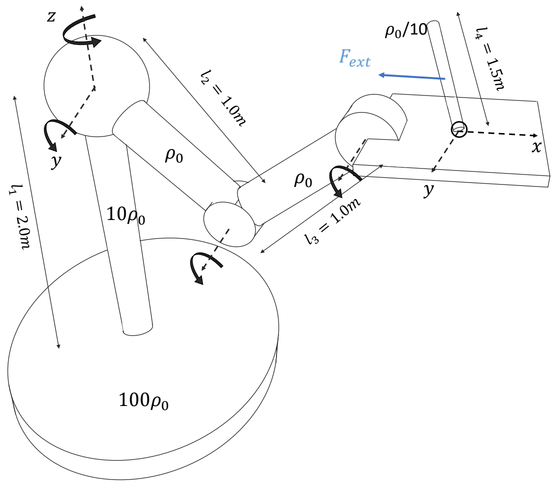


Fig. 4: An illustration of the robot arm used in the simulation and its dimensions

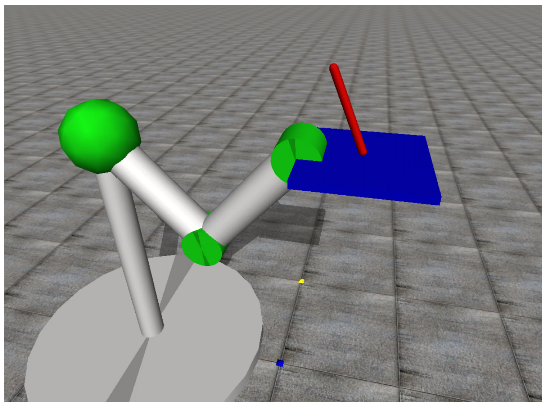


Fig. 5: A snapshot of the simulated inverted pendulum stabilization using a robot arm

An external force modeling the wind is applied, parallel to the ground, at the center of mass of the pendulum. The force is of constant amplitude $F = 0.3N$ and its direction is changed every second. Additional small variations (of 1% of the amplitude F) are applied along each axis, at every simulation time step $\Delta t = 1.0$ms. Numerical simulations of the arm, joint, and body contact dynamics were done with ODE(Open Dynamics Engine)[15].

In addition to the controller that signals the planar acceleration at the base of the pendulum (the "hand controller"), another PD controller was used to move the base of the robot (the trunk as opposed to the hand) and avoid extreme contractions or extensions of the arm during stabilization. This PD controller, which we name "trunk controller", is not subject to signal filtering by an SNE and its gain parameters are optimized in conjunction with the main controller. We assumed that in a biological system, the stochasticity of spiking neurons would have very little influences on the lateral movements of the heaviest parts of the body (the trunk), and thereby decided not to subject the "trunk controller" to SNE filtering.

Moreover, it allowed us to investigate the effects of an indirect limit on the output of the hand controller. This was done by clamping the output of the "trunk controller" to maintain it within prescribed values. This clamping limits the capabilities of the robot to readjust its base relative to the hand, and thereby it's regulation of the extension or contraction of the arm is impaired. This imposes an indirect limit to the movement of the hand as a sudden extension of the arm could lead to its full extension and therefore its inability to further adjust to the falling pendulum. Unless otherwise stated, the trunk controller is not clamped.

We also compared our results with the performance of a sigmoid kernel with additive white noise:

$$A_{pos}(t) = \frac{K_s}{1 + e^{-a_s I(t) + b_s}} + \sqrt{2D_s} W_s(t)$$

If our hypothesis made during the previous task is correct, the PD+SNE controller should demonstrate overall better performances than a regular PD controller. This is why we optimized both the PD controller and the PD+SNE controller independently, to see if the PD+SNE controller can attain performances that are not possible with the use of a PD controller alone. Both controllers were optimized using a genetic algorithm. The genetic algorithm employed, consisted of a binary gene representation of the parameters, a dynamic population size reduction over generations, a crossover rate of 80%, and a mutation rate of 1%. Parameters were tuned according to the reward function: "Simulated time until the pendulum falls (becomes parallel to the ground)". The simulation time-step was set at $\Delta t = 1.0$ms. The parameters to be optimized are: K_p, K_d (of both controllers), N, τ_s, g_{io}, and g_{act}.

This optimization was done twice, with and without clamping of the output of the trunk controller.

B. Results

After optimization of the parameters, the performances for each controller, with/without clamping of the output of the trunk controller, were as follows (Table II). These results are the mean and standard error (over 5 samples) of the mean performance, which was calculated as the mean time until the pendulum falls (over 2000 trials). At each trial, the arm was reset at the same initial position, and the polar and azimuthal angles of the pendulum were sampled from the uniform distributions $U(-\pi, \pi)$ and $U(0, \pi/2)$ respectively.

	PD	PD+SNE	PD+Sigmoid+Noise
Default	5.779 ± 0.136	7.028 ± 0.067	3.095 ± 0.062
Clamp	5.523 ± 0.080	6.977 ± 0.108	

TABLE II: Performance of the optimized PD controller and PD+SNE controller, in seconds

We found that the PD+SNE system showed an 18-26% better performance compared to the PD controller. When an indirect limitation is put on the output of the controller through clamping of the trunk controller (as discussed in the previous section), we observed a 23-30% increase in performance compared to the PD controller. Therefore, we can confirm that the use of SNEs substantially increased performance compared to the PD controller alone. Moreover, the noticeably lower performance of the application of a sigmoid kernel and white noise indicates that the effects of the SNE layer do not simply come from a non-linear gain manipulation and the addition of noise. These results are in accordance with the idea that individual spikes provide valuable information.

Drawing the robot arm in action (please refer to the video in the supplementary materials) reveals that the use of the SNE filter led to the adoption of a stabilization mode that differs from the one arising from the use of a PD controller alone. Instead of attempting to draw the pendulum closer to its vertical state, as would be expected from the PD controller, the PD+SNE controller stabilizes the pendulum by actively swinging the pendulum and keeping the swings within certain angles.

We would like to emphasize again that the SNE only acts as a filter of the control signal generated by the PD controller. These results show that the utilization of spiking neurons and the temporal filtering that it offers, add adaptivity to the agent (in the context of an inherently random and dynamically changing environment).

V. Conclusions

An ensemble of independent spiking neurons consists of using an ensemble of independent spiking neurons to filter the activation signal of the controller, before giving it to the actuators. This seemingly simple architecture applied to a PD controller for the stabilization of an inverted pendulum subjected to external forces led to a substantial expansion of the control parameters that can be used to successfully stabilize the pendulum. The mechanisms of this phenomenon need to be elucidated by further research.

In a more complex and practical setting, involving the control of a robot arm in 3-D space, this additional layer of SNE allowed noticeably higher performance (an 18-30% increase) compared to the PD controller alone. The expansion of usable control parameters which was observed in the first experiment can also be interpreted as the SNEs granting adaptability to the control system. A transient change in the environment shifts the optimal control parameters and the space of "adequate" parameters. In a conventional controller, this shift can render the current control parameters not optimal and even inadequate. However, using the SNEs can allow us to potentially avoid this problem. We hypothesize that in the case of the robot arm, the change in the arm position and the external perturbation forces transiently shift the optimal parameters for the PD controller, and as a result, the adaptability granted by the SNEs would lead to an improvement of performance.

This simple and practical addition can potentially be coupled with any controller. While analyzing this architecture involved optimization using a genetic algorithm, we often found that SNE shows the most benefits in the N=1-50 neurons and τ=1-20ms time constant range. This leads us to think that the aforementioned properties of the SNE could potentially be leveraged without or very limited optimization. Furthermore, we expect this architecture to lead to performance improvements in other tasks involving some form of stabilization such as reaching a moving target, or posture control.

References

[1] M. Davies *et al.*, "Loihi: A neuromorphic manycore processor with on-chip learning," *IEEE Micro*, vol. 38, no. 1, pp. 82–99, 2018.

[2] J. Von Neumann, "First draft of a report on the edvac," *IEEE Annals of the History of Computing*, vol. 15, no. 4, pp. 27–75, 1993.

[3] P. Blouw *et al.*, "Benchmarking keyword spotting efficiency on neuromorphic hardware," in *Proceedings of the 7th Annual Neuro-inspired Computational Elements Workshop*, 2019, pp. 1–8.

[4] T. Hwu *et al.*, "A self-driving robot using deep convolutional neural networks on neuromorphic hardware," in *2017 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2017, pp. 635–641.

[5] G. Tang *et al.*, "Spiking neural network on neuromorphic hardware for energy-efficient unidimensional slam," in *International Conference on Intelligent Robots and Systems (IROS)*. IEEE/RSJ, 2019.

[6] Joshi, Prashant and Maass, Wolfgang, "Movement generation with circuits of spiking neurons," *Neural Computation*, vol. 17, no. 8, pp. 1715–1738, 2005.

[7] M. N. Zennir *et al.*, "Spike-time dependant plasticity in a spiking neural network for robot path planning." in *AIAI Workshops*, 2015, pp. 2–13.

[8] E. Rueckert *et al.*, "Recurrent spiking networks solve planning tasks," *Scientific reports*, vol. 6, no. 1, pp. 1–10, 2016.

[9] S. Yonekura *et al.*, "Spike-induced ordering: Stochastic neural spikes provide immediate adaptability to the sensorimotor system," *Proceedings of the National Academy of Sciences*, vol. 117, no. 22, pp. 12 486–12 496, 2020.

[10] S. Glatz *et al.*, "Adaptive motor control and learning in a spiking neural network realised on a mixed-signal neuromorphic processor," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 9631–9637.

[11] T. Schwalger *et al.*, "Towards a theory of cortical columns: From spiking neurons to interacting neural populations of finite size," *PLoS computational biology*, vol. 13, no. 4, p. e1005507, 2017.

[12] Baker, SN and Lemon, RN, "Computer simulation of post-spike facilitation in spike-triggered averages of rectified emg," *Journal of Neurophysiology*, vol. 80, no. 3, pp. 1391–1406, 1998.

[13] B. Lindner, "Coherence and stochastic resonance in nonlinear dynamical systems," Ph.D. dissertation, 2002.

[14] A. Stephenson, "Xx. on induced stability," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 15, no. 86, pp. 233–236, 1908.

[15] R. Smith, "Open dynamics engine," May 2001, www.ode.org.