This paper was accepted to be presented at the 2017 International Joint Conference on Neural Networks (IJCNN 2017) will be held at the William A. Egan Civic and Convention Center in Anchorage, Alaska, USA, May 14–19, 2017.

Neurorobotic Simulations on the Degradation of Multiple Column Liquid State Machines

R. de Azambuja*†, D. H. García*, M. F. Stoelen*, A. Cangelosi*

*School of Computing, Electronics and Mathematics
University of Plymouth
Plymouth, Devon, United Kingdom

†CAPES Foundation
Ministry of Education of Brazil
Brasilia, DF 70040-020, Brazil

Email: {ricardo.deazambuja, daniel.hernandez, martin.stoelen, a.cangelosi}@plymouth.ac.uk

Abstract

Two different configurations of Liquid State Machine (LSM), a special type of Reservoir Computing with internal nodes modelled as spiking neurons, implementing multiple columns (Modular and Monolithic approaches) are tested against the decimation of neurons, connections and entire columns in order to verify which one can better withstand the damage. Based on the neurorobotics outlook, this work is part of a bigger project that aims to apply artificial neural networks to the control of humanoid robots. Therefore, as a benchmark, we made use of a robotic task where an LSM is trained to generate the joint angles needed to command a simulated version of the collaborative robot BAXTER to draw a square on top of a table. The final drawn shape is analysed through Dynamical Time Warping to generate a cost value based on how close the produced drawing is to the original shape. Our results show both approaches, Modular and Monolithic, had a similar behaviour, however the Modular was better at withstanding the decimation of neurons when it was concentrated in a single column.

Index Terms

Liquid State Machine, Reservoir Computing, Robotics, Graceful Degradation, Neurorobotics.

I. Introduction

One of the problems we face when dealing with humanoid robots is the complexity of those systems. The humanoid robot iCub [1] has more than fifty Degrees Of Freedom (DOF) and even the much simpler collaborative humanoid robot BAXTER (Rethink Robotics Inc.) still has seven DOF on each arm. Both robots can be considered redundant structures because there is an infinitude of possible ways to accomplish a certain task.

Combined efforts from neuroscience, robotics and artificial intelligence to answer this kind of questions helped to develop a new field called Neurorobotics [2], [3]. Kaplan [2] defines neurorobotics as "*the science and technology of embodied autonomous neural systems*". A neurorobotic system interacting with a real-world scenario, in order to mimic what happens with a human being, needs to integrate a large set of input signals from the visual system, vestibular system (balance), actuators (proprioceptive feedback), short term memory, long term memory

and information from different parts of the neural system. Our neurorobotic solution makes use of a special type of Spiking Neural Network (SNN), Liquid State Machines (LSM) [4], improved with an external feedback loop.

The work presented here started as part of the Bioinspired Architecture for Brain Embodied Language (BABEL) project[1], but with a focus on humanoid robot control using SNN. We aim to develop a system totally based on SNN in order to implement it on the neuromorphic system SpiNNaker [5]. The humanoid collaborative robot BAXTER will be controlled in such a way that the SNN actually can "*feel*" the world through the proprioceptive feedback received from the robot.

In a previous work [6], two different types of multiple columns LSM, Modular and Monolithic (see Section II for a better definition), enhanced with external feedback connections (see Maass et al. [7] for more details) were tested against noise injected directly into its artificial neurons[2]. The results confirmed both systems had a gracious degradation, i.e. the performance decreased proportionally to the noise amplitude instead of a catastrophic failure, but nothing was said about their behaviour in relation to damage to the internal nodes.

Here, those same LSM systems [6] were exposed to different types of decimation. They had their internal connections (Section II-B), neurons (Section II-C and II-D) and whole columns (Section II-E) decimated whilst they were benchmarked in a robotic task based on the collaborative robot BAXTER (Section II-F1) using Dynamic Time Warping (Section II-F2).

A. Background

A Liquid State Machine is an Artificial Neural Network (ANN) composed of Leaky Integrate and Fire (LIF) neurons connected in a distance-dependent probabilistic way. Another way of see it is as a recurrently connected Spiking Neural Network (SNN). The LSM is considered part of a bigger class of ANN called Reservoir Computing (RC). A group of neurons recurrently connected is what the literature calls *liquid* or *reservoir*. When first introduced by Maass et al. [4], the LSM was already known to be computationally more powerful when multiple columns were used instead of a bigger one, but it did not have the external feedback connection between the output and the input [7]. When such a feedback connection is introduced, an LSM becomes capable of emulating arbitrary Turing machines in an ideal situation.

Traditionally, an LSM experiment generates its final results averaging several trials. However, multiple trials usually have different time delays and simply averaging them do not generate the best results. A solution for this problem was presented by Azambuja et al. [8] where a parallel implementation yields better results than averaging multiple trials. Furthermore, such a system was already tested against the effects of varying the noise injected directly into the neuron model [6], showing a graceful degradation instead of a catastrophic failure.

B. The Problem

The Modular and Monolithic LSM approaches have been already tested in relation to their robustness when a noisy current is injected directly into the neuron model [6], [8], still no experiments were done to verify their

[1] www.babel-project.org

[2] Preprint version of the paper and all the source code are available at github.com/ricardodeazambuja/ICONIP2016 [6]

behaviour to internal damage. Also, it was not possible to find in the literature any work where an LSM using multiple columns and an external feedback connection [7] was tested against internal damage. Schürmann et al. [9] only mention their liquids shown robustness against faults introduced after the readout was trained, but no figures or data was presented. Hazan and Manevitz [10] presented experiments with an LSM tested against damage and noise, but their system was mono-column, it did not have the important external feedback connection (between the readout output and the input) and the task was not a robotic one.

Since in our previous work [8] we only addressed the possibility of Single-Event Upsets (SEU) or *soft-errors*, it was important to verify how the implemented LSM would react to more destructive events such as a Single-Event Latchup (SEL), Single-Event Gate Rupture (SEGR), or Single-Event Burnout (SEB).

For this work, we simulated SEL, SEGR and SEB in a very simplified way by deactivating neurons or connections. Four different possible scenarios were individually considered: decimation of internal connections (Section II-B), decimation of individual neurons (Section II-C), decimation of individual neurons in a single column (Section II-D) and decimation of entire columns (Section II-E). The verification was done using a robotic task (Section II-F1) together with Dynamic Time Warping (Section II-F2) as a benchmark. The results were analysed with the help of Welch's t-test (Section II-F3).

II. Materials and Methods

A. Modular and Monolithic Multiple Columns LSM

The Modular and Monolithic systems can be seen in the Figures 1 and 2. They have the same specifications presented in Azambuja et al. [6]. They are composed of five columns (or liquids or reservoirs), each one including six hundred artificial neurons (three layers, or a structure with 20x5x6 neurons), totalling three thousand neurons. The neuron model is the LIF with exponential synapses. In addition, the artificial neurons have white Gaussian noise ($\mu = 0$ and $\sigma = 1nA$) directly injected as a current (this happens every simulation time step) and the membrane voltage values are drawn from an uniform distribution ($13.5mV$ to $14.9mV$) every time a new simulation starts. Consequently, all trials are, by design, unique.

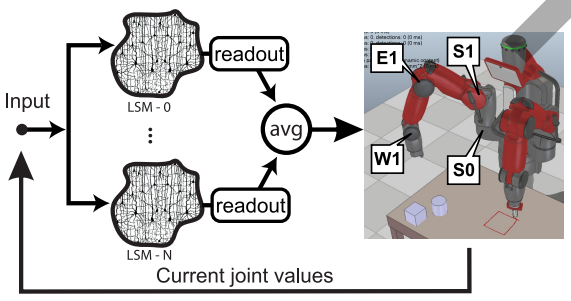

Figure 1. A Modular system is formed by multiple distinct complete LSM in parallel. They receive the same inputs (proprioceptive feedback) and their readout outputs are averaged and only then sent to the robot.

A Modular system can be implemented and put together using different hardware, because it is a complete system by itself and the training occurs individually. Even if one of the parallel pieces needed replacement, it would only

require the connection of a spare one. In this work, it was implemented using a simulator running on a PC (see Section IV for the link).

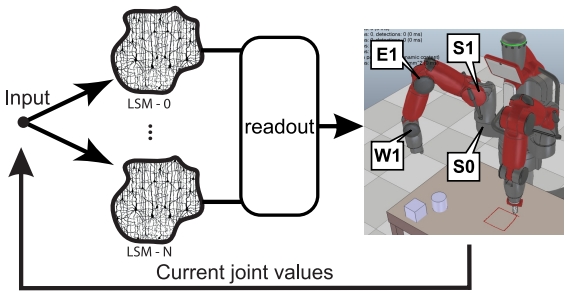

Figure 2. A Monolithic system is similar to the Modular one, since it has multiple columns (liquids). However, each column receives only part of the inputs and there is only one readout output.

The Monolithic approach has only its columns implemented separately since the inputs each column receives are not the same. For the experiments presented here, we employed the same columns from the Modular system (this was guaranteed by the use of the same random seeds) and everything was simulated in a PC using the same software mentioned earlier.

B. Decimation of Internal Connections

The experiments testing the effects of damages to the column's internal connections were simulated using nine different levels of decimation: 1%, 2.5%, 5%, 7.5%, 10%, 25%, 50%, 75% and 100% of connections. The same number of connections was randomly decimated from all columns, therefore they are evenly distributed. Modular and Monolithic systems were tested with one hundred trials each level (1800 trials in total). Both excitatory and inhibitory connections were randomly pruned during each trial. The pruning was achieved by changing the connection weight value to zero. Input and output connections were not changed.

C. Decimation of Neurons

Following pilot experiments, not presented here, the number of decimated neurons per column was defined as: 6, 12, 18, 24, 30, 36, 42, 48, 54 and 60 neurons. Again, one hundred trials were executed for each configuration (2000 trials in total). During each trial, the same number of random selected neurons was deactivated in each column and, therefore, those neurons did not generate spikes during that trial.

D. Decimation of Neurons in a Single Column

In these experiments, instead of deactivating the same number of neurons in all columns (Section II-C) the decimation was concentrated in only one column. The number of neurons decimated were: 30, 60, 90, 120, 150, 180, 210, 240, 270 and 300 neurons. For each experiment, one hundred trials were executed (2000 trials in total).

E. Decimation of Columns

After testing individual columns to partial damage (Section II-D), since the Modular approach has the possibility of implementing hot swap for each parallel LSM, individual columns were totally disconnected. Because both, Modular and Monolithic configurations, had a total of five parallel columns, we tested for the disconnection of 1, 2, 3 and 4 columns. One more time, one hundred trials were executed for each situation (totalling of 800 trials).

For the Monolithic system, all the neurons from the decimated columns stopped producing spikes and, consequently, the readout units associated to those neurons output a zero value. The Modular configuration had entire modules shut down and, therefore, those modules were not used for the generation of the final averaged output value.

F. Benchmark task

1) Simulated Baxter Robot: Our benchmark task was based on the simulated version of the humanoid collaborative robot BAXTER drawing a square on top of a table. The controlled joints employed here can be seen in Figure 3. In previous works [6], [8], all the simulations were executed using V-REP [11]. However, in this work,

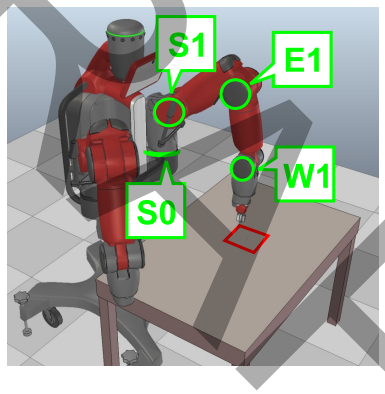

Figure 3. Simulated version of the humanoid collaborative robot BAXTER inside VREP. The four revolute joints controlled in this work (S0, S1, E1 and W1) are highlighted (green).

the testing phase was carried out with only a simplified model of the Inverse Kinematics (IK). Using this simplified IK instead of the full V-REP simulator, more trials were generated in the same amount of time. The details and source code for the IK can be found on the Github repository from the link provided in Section IV.

2) Dynamic Time Warping: The generated output from our robotic benchmark task (Section II-F1) can be seen in two different ways: a time series formed by all points $(X,\ Y,\ Z)$ or the final 2D drawing. Still, if one just analyses the time series, e.g. a Mean Square Error comparing to the original one, time delays or changes on the velocity profile could have a huge influence whilst the final drawing would look perfect. Using the cost generated by the Dynamic Time Warping (DTW) method [12] it's easier to compare the final generated 2D shape, still taking into account the time dimension. This same idea, using DTW as a benchmark for a robotic task, has already been discussed in Azambuja et al. [8]. All the DTW cost calculations were based on the algorithms available at github.com/ricardodeazambuja/DTW.

3) Welch's t-test: The comparison between the DTW results from the Modular and Monolithic approaches were made using the Welch's t-test. This test is a variant of the famous Student's t-test. The Welch's t-test was created to be applied when the variances of the two samples are unequal (heteroscedastic) [13]. We applied this test to verify if two experimental results have equal means (null hypothesis). To calculate the Welch's t-test, we have opted for the Scipy package method *ttest_ind*, with the option *equal_var=False* and both the *t-value* and the *p-value* were reported.

III. Results and Discussion

The results presented in this section are the outcome of four experiments that generated in total six thousand and six hundred unique trials. The data were analysed separately according to the type of decimation tested.

DTW results are presented in Figures 4 to 8. Each point represents the average of one hundred trials and the bars the standard error.

Statistics comparing both approaches, Modular and Monolithic, are summarised in Tables I to IV. Results that are statistically significant are shown in bold.

Finally, the drawings generated during the experiments, where the two systems were tested for the decimation of neurons in a single column (Section II-D and III-A3), are presented in Figure 7. These results are interesting because they give an idea about what range of costs the DTW method generates since the values can be verified in Figure 6.

A. Simulation Results

1) Decimation of Internal Connections: When an internal connection is decimated, even though the presynaptic neuron is still active the postsynaptic one does not receive spikes from that connection reducing its computational power. The readout is trained to generate values based on the column's dynamics and if it changes too much the readout loses its ability to generate the correct next joint values.

In the Figure 4, it's possible to visualise, when the percentage of randomly decimated connections was below 5%, the Modular configuration was less affected by decimation - what can be also verified through Table I. Clearly, from 5% both systems start converging to the same curve signalling there's a threshold value where the Modular approach can't withstand this type of aggression.

2) Decimation of Neurons: The results from the experiments where neurons were randomly deactivated to simulate a destructive event are presented in Figure 5 and Table II. The two configurations, Modular and Monolithic, had approximately the same behaviour after 18 neurons (or 3% of the neurons) were randomly decimated inside all available columns. Although, up to that point the Modular one suffered fewer effects from the damages.

When a neuron in our simulations is deactivated and stops producing spikes, all its postsynaptic connections become inactive too. The main difference in relation to the results presented in Section III-A1 (Figure 4 and Table I) is that even when some of the connections are broken the postsynaptic neuron still can produce spikes, whilst here it stops supplying information to the low-pass membrane filter and, consequently, the readout. This could explain why the decimation of connections generated a stronger reaction only when it reached 5% and the decimation of neurons had a stronger destructive effect already at 3%.

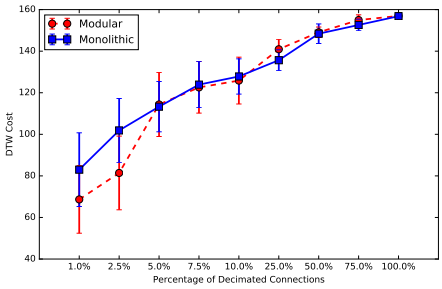

Figure 4. Decimated internal connections - Results from the experiments where a certain percentage of the internal connections between the neurons inside each column, randomly selected, were decimated. The bars represent the standard error (total of 100 trials). Statistics are presented in the Table I.

Table I

WELCH'S T-TEST RESULTS BETWEEN MODULAR AND MONOLITHIC APPROACHES - DECIMATED INTERNAL CONNECTIONS (100 TRIALS)

Percentage of decimated internal connections	*t-value*	*p-value*
1.0%	**1.9750**	**0.0497**
2.5%	**2.8941**	**0.0042**
5.0%	-0.1803	0.8571
7.5%	0.2695	0.7878
10.0%	0.4656	0.6421
25.0%	**-2.5206**	**0.0125**
50.0%	-0.4167	0.6775
75.0%	**-2.0999**	**0.0370**
100.0%	**-7.5188**	**0.0000**

3) Decimation of Neurons in a Single Column: Sometimes, when a electronic system is affected by a Single-Event Latchup, Single-Event Gate Rupture, or Single-Event Burnout, those events could generate a chain reaction affecting components closely connected. Since the systems tested here are capable to be implemented in parallel, a possible outcome would be a situation where only one column is affected. The results from Figure 6 and Table III show exactly this situation and some of the trials are presented in the Figure 7.

To clarify, in Section III-A2 the same number of neurons, randomly chosen, would affect all five columns. Here, the neurons were also randomly chosen, but only one random column was affected for each trial.

4) Decimation of Columns: The idea, when we initially presented our Modular system [8], was to be able to distribute it among several different pieces of hardware in order to have more inspiration from nature where diversity could help increasing the reliability. This would work similarly to a robotic system implemented using ROS [14] or YARP [15] - since they enable modularity, and one node would be responsible to aggregate the outputs from all

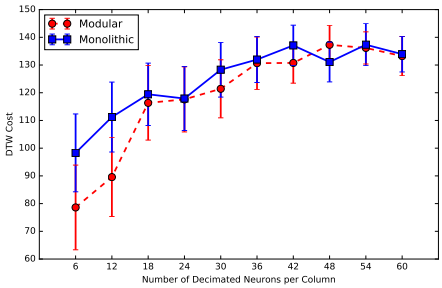

Figure 5. Decimated neurons - Results from the experiments where a certain number of neurons were randomly decimated from each of the five columns. The bars represent the standard error (total of 100 trials). Statistics are presented in the Table II. The lower the DTW Cost, the better are the results.

Table II
WELCH'S T-TEST RESULTS BETWEEN MODULAR AND MONOLITHIC APPROACHES - DECIMATED NEURONS (100 TRIALS)

Number of decimated neurons per column	*t-value*	*p-value*
6	**2.9848**	**0.0032**
12	**3.5899**	**0.0004**
18	0.5569	0.5782
24	0.0574	0.9543
30	1.5051	0.1339
36	0.3256	0.7451
42	1.9258	0.0556
48	-1.9451	0.0532
54	0.3843	0.7012
60	0.2321	0.8167

columns (the *avg* node from Figure 1). In this situation, this aggregator node would be able to ignore a node that started behaving erratically. The results in Figure 8 and Table IV show that case scenario.

A system following the Monolithic approach has its readout trained to receive inputs from all columns; therefore it fails even when only one column is disconnected. The Modular configuration is composed of stand-alone modules, consequently it withstands much better the disconnection of nodes and beats the Monolithic case in all tested scenarios (Figure 8 and Table IV).

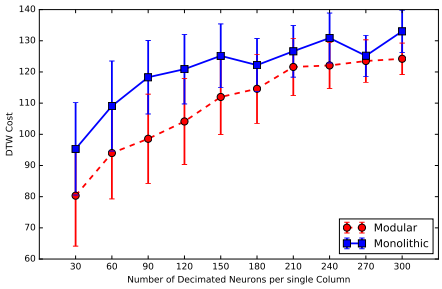

Figure 6. Decimated neurons, single column - Results from the experiments where a certain number of neurons were randomly decimated from only one (randomly chosen) of the five columns. The bars represent the standard error (total of 100 trials). Statistics are presented in the Table III. The lower the DTW Cost, the better are the results.

Table III

WELCH'S T-TEST RESULTS BETWEEN MODULAR AND MONOLITHIC APPROACHES - DECIMATED NEURONS IN A SINGLE COLUMN (100 TRIALS)

Number of decimated neurons in a single column	*t-value*	*p-value*
30	**2.1544**	**0.0324**
60	**2.3023**	**0.0224**
90	**3.3502**	**0.0010**
120	**2.9712**	**0.0034**
150	**2.6311**	**0.0092**
180	1.7074	0.0894
210	1.2765	0.2033
240	**2.5349**	**0.0120**
270	0.5581	0.5774
300	**3.2891**	**0.0012**

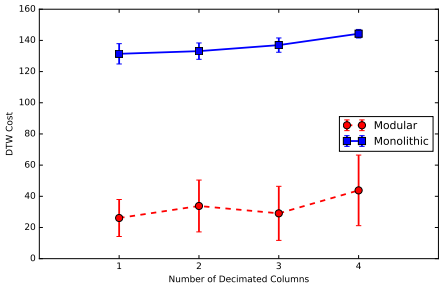

Figure 8. Decimated column - Results from the experiments where whole columns, randomly selected, were decimated. The bars represent the standard error (total of 100 trials). Statistics are presented in the Table IV. The lower the DTW Cost, the better are the results. The lower the DTW Cost, the better are the results.

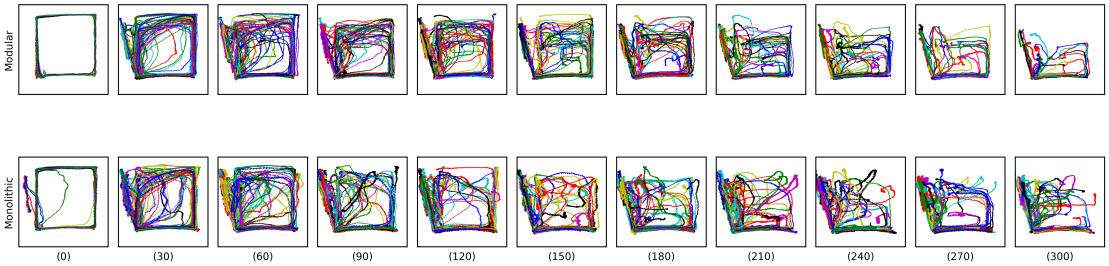

Figure 7. The final shapes that generated the DTW values in Figure 6 and Table III are presented here. It was also included one extra case were the columns were intact, therefore the number of decimated neurons was zero. The abscissa shows the Number of Decimated Neurons per single Column. All one hundred trials were plotted together.

Table IV
WELCH'S T-TEST RESULTS BETWEEN MODULAR AND MONOLITHIC APPROACHES - DECIMATED COLUMNS (100 TRIALS)

Number of decimated columns	*t-value*	*p-value*
1	**38.7050**	**0.0000**
2	**28.3430**	**0.0000**
3	**29.9274**	**0.0000**
4	**21.9195**	**0.0000**

IV. CONCLUSIONS AND FUTURE WORKS

Neurorobotic systems get inspiration from nature and also interact with the external world. To accomplish this, it is at least necessary to withstand the inherent real-world uncertainty.

In this work, we presented a framework that not only is biologically inspired, but also does not catastrophic fails when exposed to damage into its inner parts. Besides, we tested two different configurations in order to compare and contrast them in relation to the decimation of neurons and connections as an extension of what was already presented by Azambuja et al. [6].

Both approaches had quite similar behaviours when exposed to decimation in randomly chosen connections or neurons. However, the Modular one was able to better withstand decimation concentrated on a single column, or when an entire column was switched off, whilst the Monolithic catastrophic failed after only one column was completely removed.

Considering what has been already presented in previous works [6], [8], there are some ideas we expect to explore for future developments. First, because our system was only tested with tasks that kept the movement confined into the XY plan (the Z-axis was always kept constant), the next step is to try to teach the system to move and interact in the 3D space. Also, since BAXTER is a collaborative robot, it would be important to explore that with

an experiment where the robot could interact with a person - like a industrial set-up. Finally, the implementation of our neural networks directly using SpiNNaker instead of a normal PC is the oblivious next step to verify if it is possible to increase the speed of the system enabling the use of more sophisticated scenarios.

All implementation details and source code necessary to reproduce what was presented here can be found on github.com/ricardodeazambuja/IJCNN2017-2.

ACKNOWLEDGEMENT

This work was in part supported by the CAPES Foundation, Ministry of Education of Brazil (scholarship BEX 1084/13-5) and UK EPSRC project BABEL (EP/J004561/1 and EP/J00457X/1).

REFERENCES

[1] G. Sandini, G. Metta, and D. Vernon, "The icub cognitive humanoid robot: An open-system research platform for enactive cognition," in *50 Years of Artificial Intelligence*. Springer, 2007, pp. 358–369.

[2] F. Kaplan, "Neurorobotics: An experimental science of embodiment," *Frontiers in Neuroscience*, p. 23, 2008.

[3] T. Mergner and K. Tahboub, "Neurorobotics approaches to human and humanoid sensorimotor control," *Journal of Physiology-Paris*, vol. 103, no. 3-5, pp. 115–118, May 2009.

[4] W. Maass, T. Natschläger, and H. Markram, "Real-Time Computing without Stable States: A New Framework for Neural Computation Based on Perturbations," *Neural computation*, vol. 14, no. 11, pp. 2531–2560, Nov. 2002.

[5] S. B. Furber, D. R. Lester, L. A. Plana, J. D. Garside, E. Painkras, S. Temple, and A. D. Brown, "Overview of the SpiNNaker System Architecture," *IEEE Transactions on Computers*, vol. 62, no. 12, pp. 2454–2467, Dec. 2013.

[6] R. de Azambuja, F. B. Klein, M. F. Stoelen, S. V. Adams, and A. Cangelosi, "Graceful Degradation Under Noise on Brain Inspired Robot Controllers," in *Neural Information Processing*, ser. Lecture Notes in Computer Science, A. Hirose, S. Ozawa, K. Doya, K. Ikeda, M. Lee, and D. Liu, Eds. Springer International Publishing, Oct. 2016, no. 9947, pp. 195–204.

[7] W. Maass, P. Joshi, and E. D. Sontag, "Computational Aspects of Feedback in Neural Circuits," *PLoS Comput Biol*, vol. 3, no. 1, p. e165, Jan. 2007.

[8] R. de Azambuja, A. Cangelosi, and S. Adams, "Diverse, Noisy and Parallel: A New Spiking Neural Network Approach for Humanoid Robot Control," in *2016 International Joint Conference on Neural Networks (IJCNN)*, Vancouver, July 24-29 2016, pp. 1134–1142.

[9] F. Schürmann, K. Meier, and J. Schemmel, "Edge of Chaos Computation in Mixed-Mode Vlsi-a Hard Liquid," in *Advances in Neural Information Processing Systems*, 2004, pp. 1201–1208.

[10] H. Hazan and L. M. Manevitz, "Topological constraints and robustness in liquid state machines," *Expert Systems with Applications*, vol. 39, no. 2, pp. 1597–1606, Feb. 2012.

[11] E. Rohmer, S. P. Singh, and M. Freese, "V-REP: A versatile and scalable robot simulation framework," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. IEEE, 2013, pp. 1321–1326.

[12] H. Sakoe and S. Chiba, "Dynamic Programming Algorithm Optimization for Spoken Word Recognition," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 26, no. 1, pp. 43–49, Feb. 1978.

[13] G. D. Ruxton, "The Unequal Variance T-Test Is an Underused Alternative to Student's T-Test and the Mann–Whitney U Test," *Behavioral Ecology*, vol. 17, no. 4, pp. 688–690, Jan. 2006.

[14] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: An open-source Robot Operating System," in *ICRA Workshop on Open Source Software*, vol. 3, 2009, p. 5.

[15] G. Metta, P. Fitzpatrick, and L. Natale, "YARP: Yet another robot platform," *International Journal on Advanced Robotics Systems*, vol. 3, no. 1, pp. 43–48, 2006.