
Proceedings Paper:
Gan, Tian, Stepney, Susan and Trefzer, Martin A (2023) Combining Multiple Inputs to a Delay-line Reservoir Computer:Control of a Forced Van der Pol Oscillator System. In: 2023 International Joint Conference on Neural Networks (IJCNN). International Joint Conference on Neural Networks, 18 Jun 2023 Proceedings of the International Joint Conference on Neural Networks. IEEE, AUS.

https://doi.org/10.1109/IJCNN54540.2023.10191630

Combining Multiple Inputs to a Delay-line Reservoir Computer: Control of a Forced Van der Pol Oscillator System

Tian Gan
School of Physics, Engineering, and Technology
University of York
York, United Kingdom
tian.gan@york.ac.uk

Susan Stepney
Department of Computer Science
University of York
York, United Kingdom
susan.stepney@york.ac.uk

Martin A. Trefzer
School of Physics, Engineering, and Technology
University of York
York, United Kingdom
martin.trefzer@york.ac.uk

***Abstract*—The Reservoir Computing (RC) paradigm is a supervised machine learning approach that makes use of the inherent processing capacity of dynamical systems. Using the system's transient response to an external input, delayed chaotic systems offer rich dynamics for information processing, and have therefore been recognised as ideal systems for reservoir computing. A distinctive feature of delay-line reservoirs is their single-input/single-output structure, which makes them efficient for physical implementation. However, this also presents a significant limitation to multi-input tasks, as the sequence of information in the time-multiplexed input stream is not obvious. Here, we propose enhancing the input masking process used in delay-feedback RCs to mix multiple inputs in the time domain. We investigate two approaches: 'interleaved' and 'sequential', of injecting multi-input signals into a delay-line reservoir without modifying its topology. Further, we propose a novel task for RC, which inherently requires multiple inputs, to evaluate our approach: the control of a forced Van der Pol oscillator system. We use the trained reservoir as a controller to regulate the nonlinear dynamics of the Van der Pol system by constraining its trajectory to a circle. We find that, with careful choice of model parameters and offset masking scheme, the 'sequential' method outperforms the 'interleaved' method on this task.**

***Index Terms*—Unconventional computing, Delay-line Reservoir, Van der Pol, Mackey-Glass**

I. Introduction

Neuro-biological principles serve as a source of motivation for reservoir computing (RC) designs. Neural systems process information through the generation of transient activity patterns generated by sensory input signals [1]. Consequently, RC has been regarded as an effective machine learning paradigm for temporal-sequential processing [2]. As an inherent property of physical systems, deterministic chaos has garnered a great deal of attention in the study of nonlinear dynamical systems, including the neuro-biological and machine learning fields [3]–[6].

Any dynamical system with particular qualities—high dimensional state, non-linearity, and fading memory (i.e. short term memory)—might in theory serve as a reservoir computing substrate. Numerous devices having such properties have been successfully proven, including water bucket systems, memristors, and FPGAs [7], [8].

Appeltant *et al* [9] propose the computational framework for delay-feedback reservoirs. This is a useful method for reducing the complexity of hardware implementation, since the system comprises of only a single nonlinear element and a delay line. Instead of a huge network of discrete units, a number of 'virtual nodes' are created using time-multiplexing, masking, and a delay line to form a high-dimensional phase in time in the reservoir layer. In order to perform time-multiplexing, each input signal is converted into a sequence of masked signals for processing. The sample frequency usually rises as the number of virtual nodes grows, in order to ensure sufficient resolution in the masking steps. Several investigations have proven the viability of both optoelectronic and optical delay-feedback reservoir computing [10]–[12].

II. Concepts

A. Reservoir Computing

Reservoir computing is a bio-inspired machine learning paradigm that aims to use the inherent computational capacity of dynamical systems. In 1943, McCulloch and Pitts [13] presented the very first neural network model by using electrical circuits to simulate brain neurology. With several successful demonstrations over the last decade, reservoir computing has been recognised as a cutting-edge network for sequential data processing. Even for computationally difficult tasks like voice recognition and chaotic time series prediction, good results have been achieved [14], [15].

The computational model (Fig. 1) for reservoirs is based on a recurrent neural network (RNN). It is composed of three layers: the input layer, the reservoir layer, and the output layer, with only the reservoir layer exhibiting dynamic behaviour. Signals are sent from the input layer to the reservoir layer through fixed randomly weighted connections. Only the weights between the reservoir layer and the output layer are taught, and these may be trained using simple techniques, such as linear regression [2]. Consequently, the reservoir layer may

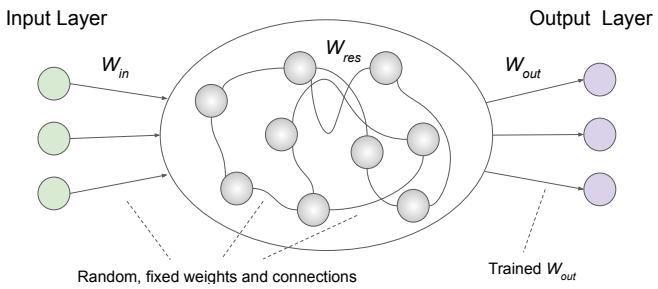


Fig. 1. Topology of Reservoir Computing.

be seen as a 'black box', enabling reservoir computing to be implemented in a broad range of physical systems with the necessary dynamics.

An RC substrate may be a physical device or material, a simulated network, or a set of equations. The characteristics of a reservoir computer rely on the underlying dynamics of the substrate a reservoir system is created with [16], [17]. The essential characteristics that physical reservoirs must possess in order to correctly perform various functions are:

1) *High-dimensionality and non-linearity.* This is depending on the quantity of distinct signals retrieved from the reservoir. If the reservoir includes a significant number of nonlinear nodes, the projection of the input data onto the reservoir is functionally equal to a mapping into a high-dimensional space. Hence, the nonlinear mapping transforms non-separable inputs into separable ones.
2) *Fading memory*. This attribute is critical for processing sequential data, since the state of a reservoir is reliant on the recent past signal but independent of the distant past, i.e. its response is dependent on relevant input, but does not become noise over time [9].

B. Delay-feedback Reservoir Computing

Appeltant *et al* [9] demonstrates that even basic nonlinear systems with delayed feedback may effectively process information. They propose a design for a nonlinear circuit with a digital delay line.

Typically, delay-differential equations are utilised to model such systems (DDEs). As the phenomenon of time delay occurs naturally in a variety of physical systems and the hardware implementation of delay-based reservoir computing requires only a single nonlinear node and a delayed feedback loop, this has resulted in numerous implementations in electronics, optoelectronics, and optics. Previous research indicates that delay differential equations are helpful for effective reservoir computation, and that masking plays a crucial role in delay-feedback reservoir computing since it defines the virtual nodes along the delay line [9], [10], [12], [18], [19].

Kuriki *et al* [20] investigates the influence of different input masking strategies on photonic reservoir computing using semiconductor lasers, utilising the Santa Fe time-series prediction problem as a benchmark. The effectiveness of masking may vary between different tasks.

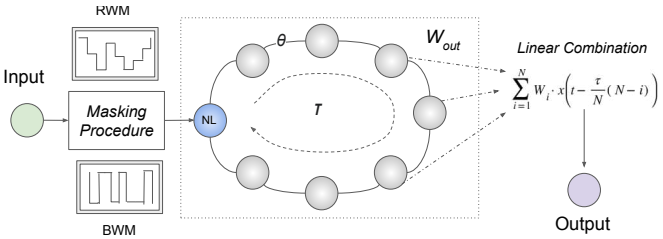


Fig. 2. Topology of Delay-feedback Reservoir Computing.

Fig. 2 shows a schematic delayed feedback reservoir. The input signal is masked, and then injected into the nonlinear node; it travels along the delay line for time interval tau, and is then re-injected into the loop, so forming a delay-feedback reservoir.

The mask is used to designate N effective virtual nodes along the delay line. The time delay between each of the virtual nodes is $\theta = \tau/N$. Masking mixes the input signal with several sets of scaling factors to elicit a dynamically rich response from the reservoir. Typically, the masking method utilizes either a *Real Weight Mask* or a *Binary Weight Mask*. We have previously investigated the effectiveness of these two input mask types under different scenarios for the NARMA-10 benchmark task [21], and we are using *Binary Weight Mask* here.

III. Models and Benchmarks

A. Mackey-Glass Model

The Mackey-Glass model [22] was established in the context of respiratory and haematological disorders, where time delay plays a crucial role [23]–[25]. The model is a first-order nonlinear delay differential equation

$$\dot{P} = \frac{\beta\theta^n P_\tau}{\theta^n + P_\tau^n} - \gamma P_t, \quad P_\tau \equiv P(t-\tau) \tag{1}$$

where the conditional variable P_t is the homogenous density of mature blood cells in a population at time t; τ is the period between the beginning of blood cell synthesis and the release of mature blood cells; β, θ, and n are related to the production rate; and γ defines the decay rate of the cells [22].

The equation exhibits a variety of aperiodic and chaotic behaviours, dependent on parameter values. Due to its robust dynamics and ability to be implemented in hardware, the model is appropriate for delay-line reservoir computing [23], [26], [27].

In the context of delay-line reservoir computation, Appeltant *et al* [9] explore the time-normalized equation with state variable x_t, and add an external input I_t to the delayed feedback value, $x_\tau \rightarrow x_\tau + \delta I_t$, where δ is an input scaling parameter:

$$\dot{x} = \frac{\beta(x_\tau + \delta I_t)}{1 + (x_\tau + \delta I_t)^n} - \gamma x_t, \quad x_\tau \equiv x(t-\tau) \tag{2}$$

Here, x_t represents the normalised voltage at physical time t; τ is the physical time delay in the feedback loop; β, n and γ are the same as previously.

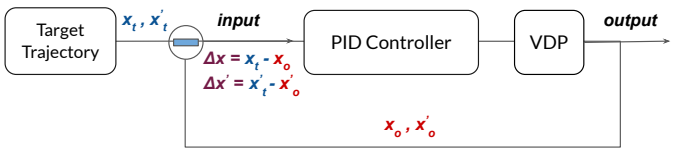


Fig. 3. PID Control topology.

B. Van der Pol Oscillator Benchmark Task

Consider the following non-linear differential equation:

$$\frac{d^2x}{dt^2} - \mu(1 - x^2)\frac{dx}{dt} + x = F(t) \tag{3}$$

When the forcing term $F(t) = 0$, this is the Van der Pol oscillator. The Van der Pol oscillator is a non-conservative dynamical system that exhibits limit cycle behavior, that is, it oscillates between two or more states in a repeating pattern.

It was first described by the Dutch physicist Balthazar van der Pol in the 1920s [28]. It is particularly useful for modeling systems that exhibit non-linear behavior, such as relaxation oscillations and self-sustained oscillations. The equation demonstrates oscillatory activity, but its amplitude is not constant; it represents an invariant set known as a 'limit cycle'. Regardless of the initial conditions, all system paths converge to this invariant.

We seek to control the VDP system so that it produces a fixed-amplitude oscillation, i.e. a circle as shown in Fig. 4, through defining a suitable control function for the external force $F(t)$ in Eq. 3. Cooper *et al* reveal the difficulties in controlling the VDP Oscillator's nonlinear dynamics with a PID controller [29].

Our approach starts with the explanation of the intrinsic dynamics and the effects of various initial conditions. Eq. 3 with $F(t) = 0$ is simulated in SIMULINK Tool, in which the nonlinear Van der Pol equation is propagated forward in time with different initial conditions, and as shown in Fig. 4a.

To achieve a controlled circular trajectory, we use a PID controller designed for optimal performance under a commanded trajectory of amplitude of 2. To compare the forced trajectory with this target trajectory, the phase plots are shown in Fig. 4.

The task is to train the reservoir to imitate the output control signal of the PID controller as closely as possible using the same input, and ultimately to replace the PID and drive the forced Van der Pol system with the trained reservoir as the controller.

When training the reservoir to imitate the PID controller, we use the normalised root mean square error (NRMSE) to assess and compare the performance of different experimental techniques:

$$\text{NRMSE} = \sqrt{\frac{1}{m}\frac{\sum_{k=1}^{m}(\hat{y_k} - y_k)^2}{\sigma^2(y_k)}} \tag{4}$$

where y is the target function, $\hat{y}$ is the reservoir output, m is the number of data samples in the run, and σ is the standard deviation. An NRMSE of 0 indicates that the system output and target output are in perfect agreement; an NRMSE of 1 indicates that the system output is equal to the mean target output.

IV. Experimental Methods

A. System Implementation

1) Masking.: Masking plays a crucial role in delay-feedback reservoir computing since it defines the virtual nodes along the delay line. In this paper, we use *Binary Weight Mask* (a random sequence of -1 and $+1$ values) with different configurations:

a) Mask-multiplexing with offset by u:

$$I(t) = I_0(t)(M(t) + 1) \tag{5}$$

b) Mask-multiplexing with no offset:

$$I(t) = I_0(t)M(t) \tag{6}$$

2) Simulation environment: We define the reservoir using Simulink. This model, shown in Fig. 5, is an intermediate step between the mathematical DDE model and a physical circuit.

In accordance with the variables in Eq. 2, appropriate function blocks are chosen for implementation and linked accordingly to implement the different terms. MATLAB is used to produce the input signal, which is then fed into the system. A subsystem transmitting information to MATLAB collects the state matrix of virtual nodes for reservoir training and assessment.

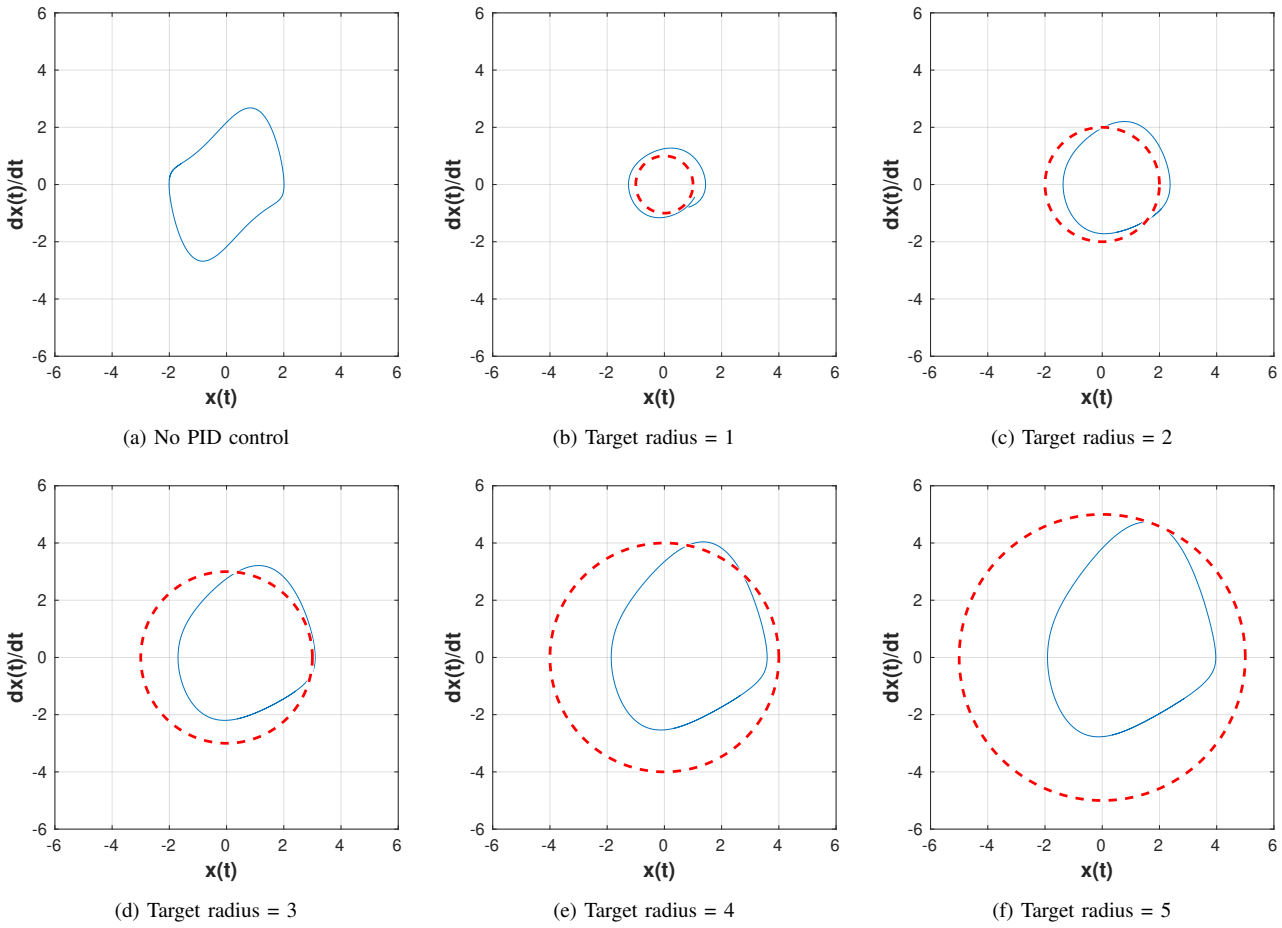
B. Multiple Inputs

In order to circumvent the limitation of single-input delay-feedback reservoirs, in this paper, we introduce two techniques of injecting multiple inputs into the system:

1) *Interleave*. The interleave technique schematic, shown in Fig. 6a, is a method that places masked dual- or multi-inputs in an alternating manner. For example, when two input streams undergo time-multiplexing procedure, they are completely mixed before injecting into the system, i.e. 'x_1,$\dot{x_1}$,x_2,$\dot{x_2}$,...' shown in the illustration with inputs being 'x and $\dot{x}$'.
2) *Sequential*. In contrast, the sequential approach leaves the inputs in sequence and concatenates them before undergoing the masking procedure(shown in Fig. 6b), i.e. 'x_1,x_2,x_3,..., $\dot{x_1}$,$\dot{x_2}$,$\dot{x_3}$,...'. The length of each input value's concatenation depends on the number of virtual nodes along the delay line. In this instance, each input value is concatenated to length of 15, as 30 virtual nodes are defined in the reservoir.

C. Parameter Setting

The experimental parameter settings are shown in Table 7. State values are always positive in this model (they represent blood cell concentrations). Here, state values are voltages that are modified by inputs; state value plus input term $x_\tau + \delta I_t$



(a) No PID control (b) Target radius = 1 (c) Target radius = 2

(d) Target radius = 3 (e) Target radius = 4 (f) Target radius = 5

Fig. 4. Forced VDP oscillator trajectories (blue solid line) and forced target trajectories (red dashed line) are shown. (a) Initial baseline trajectory without PID controller (unforced Van der Pol trajectory). (b–f) Comparisons between desired forced trajectory and actual trajectory with amplitudes of 1–5 respectively. The single PID controller is optimized for an amplitude of 2.

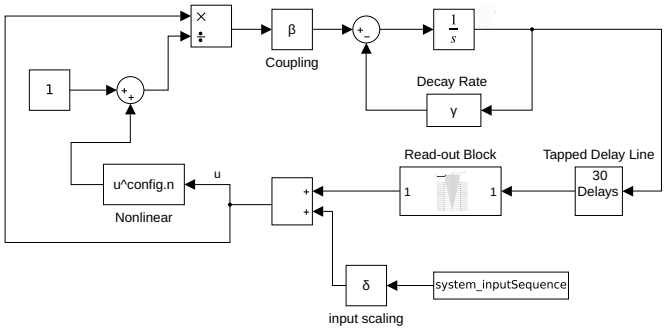


Fig. 5. Simulink circuit schematic of the Mackey-Glass delay-feedback system.

may turn negative with certain masking processes. According to our findings in previous work [21], we select the hyper parameters in Fig. 7 and limit n to integer values to prevent difficulties with raising negative values to fractional powers.

The Van der Pol oscillator is run to produce a sequence length of $L = 5000$, which is subsequently split into *training* and *testing* datasets, $L_{\text{train}} = 2280$, $L_{\text{test}} = 1140$, $L_{\text{unseen}} = 380$, and a washout of $L_{\text{washout}} = 1200$ with the first L_{washout} samples discarded.

D. Training

A training algorithm assigns an output weight w_i to each virtual node in order to compute the overall system output:

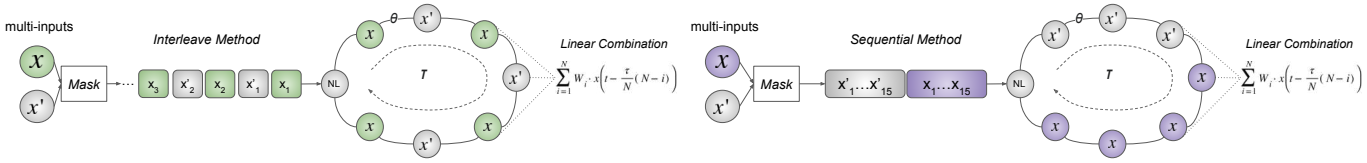
$$\hat{y}_k = \sum_{i=1}^{N} w_i \cdot x\left(k\tau - \frac{\tau}{N}(N-i)\right) \tag{7}$$

The target output y is used to produce an $[M \times L]$ dimensional target matrix, where $M = 1$ is the dimension of the target output y. We refer this $[M \times k]$ dimensional matrix as Y ($k = L_{\text{train}}$ indicates the length of the training sequence). In order to determine the optimum output weights W, it is necessary to minimise the mean square error $\|WS - Y\|^2$.

Ridge regression is applied to avoid problems with multi-collinearity in matrices through the following formula:

$$W_{opt} = YS^T(SS^T - \lambda I)^{-1} \tag{8}$$

where T is matrix transpose, λ is the regression parameter, and I is the identity matrix of dimensions $N \times N$. (This solution may also be reached by using the Moore–Penrose



(a) (b)

Fig. 6. Interleave (a) and Sequential (b) masking methods are shown.

parameter	value	description
β	2	coupling factor
γ	1	decay rate
τ	1.8 (sec)	delay in feedback loop
n	$1, 2, \ldots, 9$	nonlinearity
δ	0.25	input weighting
N	30	number of virtual nodes

Fig. 7. Parameter values for the Mackey–Glass delay-line reservoir experiment. The Mackey–Glass system parameter values are from [30]; n is required to be an integer (see Section.IV-C for details).

pseudo-inverse: $W_{\text{opt}} = YS^{+}$, where '+' denotes the pseudo-inverse function [15], although there may be instability issues manifesting as large weight values.) The trained system output $\hat{y}(t)$ is provided by

$$\hat{y}(t) = W_{\text{opt}} x(t) \tag{9}$$

After training, the computational performance of the system is assessed by injecting the test set of input signals into the reservoir and calculating the NRMSE based on the overall system output using these trained output weights, and the target output.

V. Experiments and Results

A. Imitation of PID Control Behaviour

1) Experiments.: We systematically compare the computational performance of two multi-input techniques: 'Interleave' and 'Sequential', with no-offset (Eq. 6) and offset (Eq. 5) under different nonlinearity parameters (see Section III-A): n = 1 to n = 9. In this experiment, the PID controller is optimized for a radius of 2 and these settings stay the same for all target radii used here, ranging from 1 to 5. Note that, therefore, the target sequences generated by the PID controller for radii of $1, 3, 4, 5$ are not optimal control solutions. This is deliberate, so that the capability to generalise over a range of radii $\neq 2$ can be evaluated independently for PID and reservoir controller.

2) Results.: Fig. 9 shows the results of these different experimental settings.

1) *Interleave v Sequential*. The Sequential technique displays better performance than Interleave, in the no-offset and offset arrangements, with lowest NRMSE of 0.090 and 0.069 (n = 3, $targetradius$ = 4) respectively.

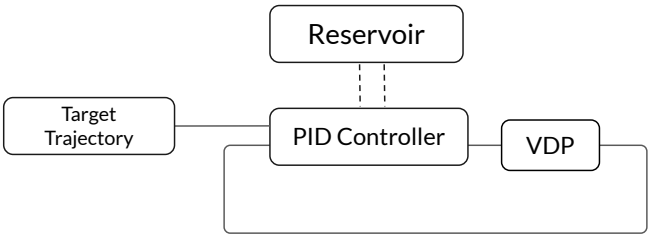


Fig. 8. Reservoir replacing the PID controller.

2) *No-offset v Offset*. In this task, offset mask gives better results. Since we are using Mackey-Glass equation (Eq. 2) as the non-linear component of the reservoir, the negative input value may cause the denominator term crushed with worse NRMSE results = 1.000 (indicated yellow in Fig. 9a and Fig. 9c).

B. Reservoir control Van der Pol Oscillator

1) Experiment: Since the ultimate goal of this benchmark task is to use a reservoir as a controller to drive the trajectory of a Van der Pol system (see Fig. 8), we first use the test data set of best trained reservoir from the previous experiment, with n=2, to drive the Van der Pol Oscillator.

2) Result: Fig. 10 shows the comparison between the target circular trajectory (red dashed line) representing the baseline, the trajectory commanded by the PID controller optimised for radius 2 (orange dotted line), and the trajectory controlled by the reservoir trained on the PID output from radius 2 (blue solid line). Note that here, the reservoir is trained using a PID controller tuned for a radius of 2 in all three cases. Since we are not seeking perfection but rather intend to demonstrate and compare the sensitivity of the PID and reservoir controller, optimised for radius 2 respectively, when applied to larger or smaller radii.

VI. Conclusion and Future Work

Delay-feedback reservoirs have input constrains which are not apparent in commonly used benchmark tasks such as, e.g., NARMA-10, Santa Fe Laser task, etc.. This is because only a single input is required. However, there is a need to consider tasks which require multiple inputs to perform the intended function successfully. Therefore, we propose a control systems benchmark task here, with the objective of employing a trained reservoir as a controller and using its output as a driving force imposing a circular trajectory on the Van der Pol oscillator

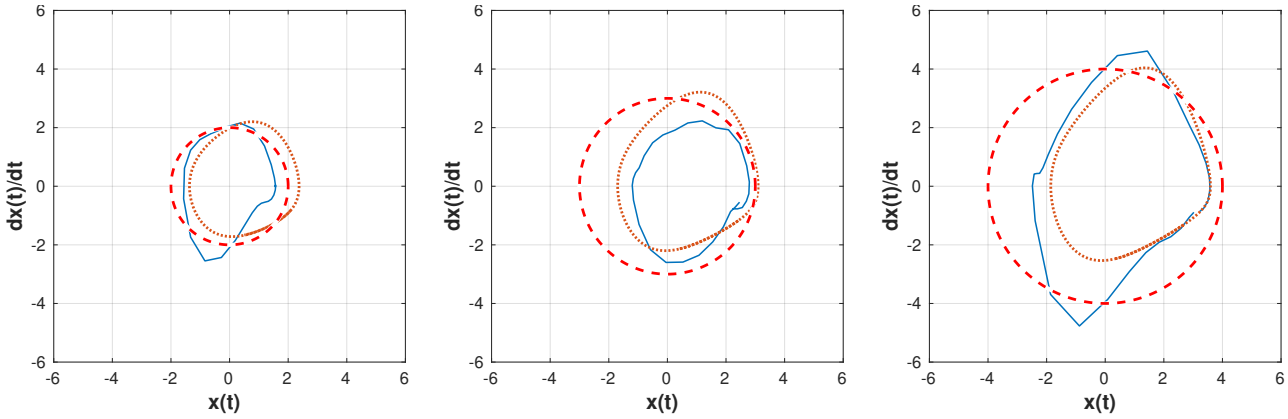
(a) Interleave with no-offset masking.

(b) Interleave with offset masking.

(c) Sequential with no-offset masking.

(d) Sequential with offset masking.

Fig. 9. Simulation results for controlling benchmark task with two multi-input methodologies: *Interleave* (fig.9a and fig.9b) and *Sequential* (fig.9c and fig.9d) based on different masking schemes (Eq. 5 and Eq. 6).



(a) Target radius = 2.

(b) Target radius = 3.

(c) Target radius = 4.

Fig. 10. Simulation results for Van der Pol nonlinear system driven by the trained reservoir. Comparisons between target circle (red dashed line), trajectory commanded by PID controller (orange dotted line), and trajectory controlled by reservoir computing (blue solid line).

system. Control tasks are relatively new to the field of RC, and we aim to begin establishing suitable benchmarks from the control problem domain by introducing this benchmark.

Our experiments have investigated the 'Interleave' and 'Sequential' techniques for injecting multiple inputs into a delay-feedback reservoir without altering its topology. Considering the effectiveness of masking and time-multiplexing functions, we compared the configurations between 'Interleave', 'Sequential' and 'No-offset', 'Offset' masks. We have trained the reservoir to imitate the output signal of the original PID controller (optimised for radius 2). Among all configurations 'Sequential' with 'Offset' features the best performance when solving the newly proposed forced Van der Pol oscillator control task. This is encouraging since these techniques might easily tackle the multi-input challenge.

Our second experiment has demonstrated using a suitably-trained reservoir to control Van der Pol Oscillator. With the same experimental implementation and parameters, we have compared the computational performance of PID controller and reservoir controller. The results show that, trained on an optised PID controller, the reservoir itself has a potential to replace the PID controller and drive the Van der Pol oscillator's initial trajectory and command forced trajectories (a desired circular trajectory in this case). This result encourages further investigation of using reservoirs as controllers, and to devise more suitable benchmarks in this domain (see Fig.8).

In future work we intend to investigate training the reservoirs on improved (further and more specifically optimised, e.g., for various radii) PID controllers to achieve a better performance for the nonlinear dynamics controlling task. For comparison purposes, it would be a sensible move to train reservoirs on different controllers, such as Predictive Functional Control (PFC) and Proportional position loop Integral and proportional Velocity loop (PIV). We will also investigate hybrid schemes, combining features of both 'Interleaved' and 'Sequential' schemes with repetitions of experiments.

ACKNOWLEDGMENT

This work is supported by the UK Engineering and Physical Sciences Research Council under the MARCH project [EP/V006029/1].

REFERENCES

[1] W. Maass, T. Natschlager, 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, 2002.

[2] M. Lukosevicius and H. Jaeger, "Survey: Reservoir computing approaches to recurrent neural network training," *Computer Science Review*, vol. 3, no. 3, pp. 127–149, 2009.

[3] W. J. Freeman, "Tutorial on neurobiology: from single neurons to brain chaos," *International Journal of Bifurcation and Chaos*, vol. 2, no. 03, pp. 451–482, 1992.

[4] K. Aihara, "Chaos in neural response and dynamical neural network models: toward a new generation of analog computing," in *Towards the Harnessing of Chaos*, M. Yamaguchi, Ed. Elsevier, 1994, pp. 83–98.

[5] W. Singer, "Striving for coherence," *Nature*, vol. 397, no. 6718, pp. 391–393, 1999.

[6] T. Weng, H. Yang, C. Gu, J. Zhang, and M. Small, "Synchronization of chaotic systems and their machine-learning models," *Physical Review E*, vol. 99, no. 4, p. 042203, 2019.

[7] C. Fernando and S. Sojakka, "Pattern recognition in a bucket," in *2003 European Conference on Artificial Life*. Springer, 2003, pp. 588–597.

[8] C. Du, F. Cai, M. A. Zidan, W. Ma, S. H. Lee, and W. D. Lu, "Reservoir computing using dynamic memristors for temporal information processing," *Nature Communications*, vol. 8, no. 1, pp. 1–10, 2017.

[9] L. Appeltant, M. C. Soriano, G. V. Der Sande, J. Danckaert, S. Massar, J. Dambre, B. Schrauwen, C. R. Mirasso, and I. Fischer, "Information processing using a single dynamical node as complex system," *Nature Communications*, vol. 2, no. 1, pp. 468–468, 2011.

[10] D. Brunner, M. C. Soriano, C. R. Mirasso, and I. Fischer, "Parallel photonic information processing at gigabyte per second data rates using transient states," *Nature Communications*, vol. 4, no. 1, pp. 1364–1364, 2013.

[11] Q. Vinckier, F. Duport, A. Smerieri, K. Vandoorne, P. Bienstman, M. Haelterman, and S. Massar, "High-performance photonic reservoir computer based on a coherently driven passive cavity," *Optica*, vol. 2, no. 5, pp. 438–446, 2015.

[12] Y. Paquot, F. Duport, A. Smerieri, J. Dambre, B. Schrauwen, M. Haelterman, and S. Massar, "Optoelectronic reservoir computing," *Scientific Reports*, vol. 2, no. 1, pp. 287–287, 2012.

[13] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas imminent in nervous activity," *The Bulletin of Mathematical Biophysics*, vol. 5, pp. 115–133, 1943.

[14] D. Verstraeten, B. Schrauwen, M. d'Haene, and D. Stroobandt, "An experimental unification of reservoir computing methods," *Neural Networks*, vol. 20, no. 3, pp. 391–403, 2007.

[15] H. Jaeger and H. Haas, "Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication," *Science*, vol. 304, no. 5667, pp. 78–80, 2004.

[16] M. Dale, J. F. Miller, S. Stepney, and M. A. Trefzer, "Evolving carbon nanotube reservoir computers," in *International Conference on Unconventional Computation and Natural Computation*. Springer, 2016, pp. 49–61.

[17] Z. Konkoli, S. Nichele, M. Dale, and S. Stepney, "Reservoir computing with computational matter," in *Computational Matter*, S. Stepney, S. Rasmussen, and M. Amos, Eds. Springer, 2018, pp. 269–293.

[18] L. Larger, M. C. Soriano, D. Brunner, L. Appeltant, J. M. Gutierrez, L. Pesquera, C. R. Mirasso, and I. Fischer, "Photonic information processing beyond turing: an optoelectronic implementation of reservoir computing," *Optics Express*, vol. 20, no. 3, pp. 3241–3249, 2012.

[19] P. Antonik, F. Duport, M. Hermans, A. Smerieri, M. Haelterman, and S. Massar, "Online training of an opto-electronic reservoir computer applied to real-time channel equalization," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 11, pp. 2686–2698, 2016.

[20] Y. Kuriki, J. Nakayama, K. Takano, and A. Uchida, "Impact of input mask signals on delay-based photonic reservoir computing with semiconductor lasers," *Optics Express*, vol. 26, no. 5, pp. 5777–5788, 2018.

[21] T. Gan, S. Stepney, and M. A. Trefzer, "Tradeoffs with physical delay feedback reservoir computing," in *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 2021, pp. 1–8.

[22] M. C. Mackey and L. Glass, "Oscillation and chaos in physiological control systems," *Science*, vol. 197, no. 4300, pp. 287–289, 1977.

[23] L. Junges and J. A. Gallas, "Intricate routes to chaos in the mackey–glass delayed feedback system," *Physics Letters A*, vol. 376, no. 30-31, pp. 2109–2116, 2012.

[24] E. Shahverdiev, R. Nuriev, R. Hashimov, and K. Shore, "Chaos synchronization between the mackey–glass systems with multiple time delays," *Chaos, Solitons & Fractals*, vol. 29, no. 4, pp. 854–861, 2006.

[25] S. Sano, A. Uchida, S. Yoshimori, and R. Roy, "Dual synchronization of chaos in mackey-glass electronic circuits with time-delayed feedback," *Physical Review E*, vol. 75, no. 1, p. 016207, 2007.

[26] L. Berezansky and E. Braverman, "Mackey–Glass equation with variable coefficients," *Computers & Mathematics with Applications*, vol. 51, no. 1, pp. 1–16, 2006.

[27] M. Dale, J. F. Miller, S. Stepney, and M. A. Trefzer, "A substrate-independent framework to characterize reservoir computers," *Proceedings of the Royal Society A*, vol. 475, no. 2226, p. 20180723, 2019.

[28] B. Van der Pol, "Over relaxatietrillingen," *Physica*, vol. 6, no. 1926, pp. 154–157, 1926.

[29] M. Cooper, P. Heidlauf, and T. Sands, "Controlling chaos—forced Van der Pol equation," *Mathematics*, vol. 5, no. 4, p. 70, 2017.

[30] L. Glass and M. Mackey, "Mackey-Glass equation," *Scholarpedia*, vol. 5, no. 3, p. 6908, 2010, revision #186443; doi:10.4249/scholarpedia.6908.