ESANN 2012 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2012, i6doc.com publ., ISBN 978-2-87419-049-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100967420.

A Discrete/Rhythmic Pattern Generating RNN

Tim Waegeman, Francis Wyffels and Benjamin Schrauwen

Department of Electronics and Information Systems, Ghent University
Sint Pietersnieuwstraat 41, B9000 Ghent, Belgium

Abstract. Biological research supports the concept that advanced motion emerges from modular building blocks, which generate both rhythmical and discrete patterns. Inspired by these ideas, roboticists try to implement such building blocks using different techniques. In this paper, we show how to build such module by using a recurrent neural network (RNN) to encapsulate both discrete and rhythmical motion patterns into a single network. We evaluate the proposed system on a planar robotic manipulator. For training, we record several handwriting motions by back driving the robot manipulator. Finally, we demonstrate the ability to learn multiple motions (even discrete and rhythmic) and evaluate the pattern generation robustness in the presence of perturbations.

1 Introduction

Animals and humans have the ability to interact with their environment in a skillful and graceful, sometimes referred to as natural, manner. For several millennia, mankind tried to replicate such natural behavior in the design of robots to reduce the need for human labor, typically repetitive tasks that requires precision. However, as human-robot interaction became increasingly important, the need of more advanced motor skills emerged. More recently, researcher are using biologically inspired techniques to enhance the motion skills of robots. One of these approaches [4] is based on the study of the locomotion of a lamprey [3]. Researchers found groups of neurons, the so called "Central Pattern Generator" (CPG) located in the spinal cord, which is mainly responsible for the rhythmic locomotion of the lamprey. Other biological research on humans and animals has shown that a wide variety of continuous and discrete motions can be represented by a limited number of discrete motion primitives. For instance, in [1] a factorization algorithm was employed to extract a small amount of invariant spatio temporal relationships, called "synergies", among muscle activations of animals. Such research indicates that a limited repertoire of discrete primitives is responsible for the advanced motion skills of humans and animals. These findings support the concept of a modularly organized motor control [2], inspiring the implementation of such motion primitives for robotic applications. In [5] discrete or rhythmic motions were learned by shaping the attractor landscape of a nonlinear differential equation with statistical learning methods. Another approach uses a learning method to estimate the parameters of a Gaussian Mixture Model to model discrete motions [8]. However, a representation of both rhythmic and discrete motions in a single trained system is not realized. In this work we propose a technique to embed both rhythmic and discrete motion patterns

ESANN 2012 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2012, i6doc.com publ., ISBN 978-2-87419-049-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100967420.

into a single attractor landscape formed by a RNN. Similar as in [11], we will use a fast training technique for large RNNs unified as "Reservoir Computing" (RC) [10] together with similar training approaches.

2 Discrete and Rhythmic Pattern Generator

The key idea behind the proposed system is to use the dynamics of a RNN to embed the desired attractors which is a limit cycle attractor for rhythmic motions (e.g.: walking) and/or a fixed point attractor for discrete motions (e.g.: reaching). In this work, Reservoir Computing (RC) [10] is used to efficiently train such a RNN which follows the Echo State Network (ESN) approach [6]. An ESN is composed of an input, a discrete-time RNN (i.e., the reservoir) and a linear readout output layer which maps the reservoir states to the desired output. Additionally, in this work, the system output is fed back to the reservoir. The system's dynamics are represented by the following equations:

$$\mathbf{x}[k+1] = (1-\gamma)\mathbf{x}[k] + \gamma \tanh\left(\mathbf{W}_{\mathrm{r}}^{\mathrm{r}}\mathbf{x}[k] + \mathbf{W}_{\mathrm{o}}^{\mathrm{r}}\mathbf{y}[k] + \mathbf{W}_{\mathrm{i}}^{\mathrm{r}}\mathbf{u}[k]\right) \quad (1)$$

$$\mathbf{y}[k+1] = \mathbf{W}_{\mathrm{r}}^{\mathrm{o}}\mathbf{x}[k+1], \quad (2)$$

where $\mathbf{x}[k]$ represents the reservoir states with $\mathbf{x}[0] = \mathbf{0}$. $\mathbf{u}[k]$ and $\mathbf{y}[k]$ represent respectively the input and output. The term γ is called the leak rate which can effectively tune the system's dynamics [7]. The weight matrices $\mathbf{W}_{*}^{\Delta}$ represent the connections from $*$ to Δ between the nodes of the network (where r, i and o denote *reservoir*, *input* and *output*, respectively). All weight matrices $\mathbf{W}_{*}^{\mathrm{r}}$ to the reservoir are initialized randomly (drawn from $\mathcal{N}(0,1)$), while all connections to the output $\mathbf{W}_{*}^{\mathrm{o}}$ are trained using standard linear regression techniques. After initialization, $\mathbf{W}_{\mathrm{r}}^{\mathrm{r}}$ is normalized by dividing it with its largest absolute eigenvalue (spectral radius). The spectral radius is typically tuned close to 1, such that the network operates at the edge of chaos where its computational power is greatest [9]. The RC-network in this work has 2 outputs because we are using 2 dimensional motions. The output is fed back to the reservoir. To allow switching between rhythmic and discrete motion patterns, or multiple patterns in general, the input $\mathbf{u}[k]$ is set to an arbitrarily chosen value, different for each motion.

During training, the output $\mathbf{y}[k]$ in equation (1) is set to the desired output $\mathbf{y}_d[k]$. By calculating and collecting subsequently each state according to equation (1) with a defined input ($\mathbf{u}[k]$), the output weights $\mathbf{W}_{\mathrm{r}}^{\mathrm{o}}$ can be determined in one shot by applying linear regression [6]:

$$\mathbf{W}_{\mathrm{r}}^{\mathrm{o}} = (\mathbf{X}^T\mathbf{X} + \rho\mathbf{I})^{-1}\mathbf{X}^T\mathbf{T}, \quad (3)$$

where $\mathbf{X}$ consists of all reservoir states $\mathbf{x}[k]$. Matrix $\mathbf{T}$ contains all desired outputs $\mathbf{y}_d[k]$. To improve generalization capabilities and to prevent overfitting of the data, ridge regression is used [12], introducing the regularization parameter ρ in equation (3). This parameter is optimized using 4-fold cross-validation.

Rhythmic pattern generation in the context of a CPG was investigated in [11]. In our approach the training of rhythmic patterns is achieved in a similar manner.

ESANN 2012 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2012, i6doc.com publ., ISBN 978-2-87419-049-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100967420.

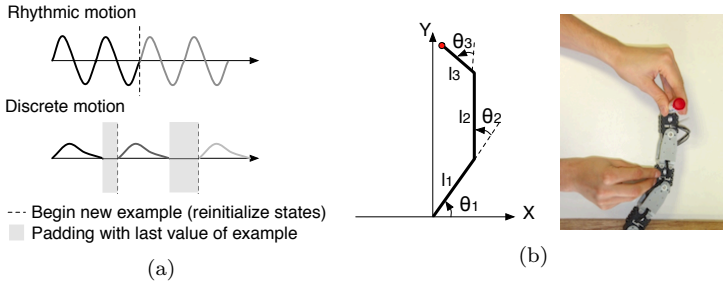

Fig. 1: (a) Illustration of how the training data is composed. Different examples of the same motion are shown for both rhythmic and discrete patterns. The dashed line indicates the moment at which the reservoir states are reinitialized. (b) Schematic representation of the planar manipulator together with the associated angles θ_i and link lengths $l_1 = l_2 = 9.15$ cm and $l_3 = 3.5$ cm.

However, to allow the training of different examples of a rhythmic motion we reinitialize the reservoir states after each example (Containing multiple periods) ($\mathbf{x}[k_e] = \mathbf{0}$ with k_e the first time step of a new example).

When training a discrete motion, after each example a random sized padding of its last example value is added. By applying this padding, the output weights of the RC-network are also trained on the transients of the writing motions to their fixed target positions. Similar as with rhythmic motions, the reservoir states are reinitialized at the beginning of an example after the padding of the previous example. This process is illustrated at the bottom of Fig. 1(a).

If the generation of different writing motions is needed, the used training data of one motion should not overlap in the 2D-plane with the training data of the others. Which motion pattern is generated is then determined by the initial position in the 2D plane where the network is initialized with. However, when including a different input $\mathbf{u}[k]$ for a different pattern during training, the pattern which is generated is determined by this input. We will demonstrate the latter in the following section.

3 Robotic Application

In order to evaluate the proposed pattern generator, we train it to reproduce letter writing motions with a planar manipulator (Fig. 1(b)). Here, the manipulator with 3 rotational joints is not just blindly commanded to follow a trajectory but is included in the control loop. Several examples of letter writing motions are recorded by moving the servos manually and reading the corresponding encoder values. After training, by initializing the feedback with a start position, it will generate the next trajectory position. Subsequently, this position is transformed to an angular position by applying inverse kinematics, which is used to control the manipulator. In turn, the encoder values are read and used as feedback to

ESANN 2012 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2012, i6doc.com publ., ISBN 978-2-87419-049-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100967420.

the RC-network after applying forward kinematics. During training, each letter was shown 6 times with different starting points but with the same target point.

In this paper, we show examples of the letter R, which contains intersecting points, to demonstrate the need of memory in the generation of some discrete motions. At such an intersecting point, knowledge about the previous positions is necessary to allow successful completion of the motion. Generating such motions with a simple memoryless mapping from one position to the other would be impossible. To present the ability to learn rhythmic motions we used 6 "examples" of a figure-8, not by back driving the robot but by using the following equation: $x = 13 + 3\cos(\psi), y = -5 + 3\sin(2\psi)$ with $\psi \in [\frac{\pi}{2}, 10\pi]$. Because each "example" is the same, the need for state reinitialization can be omitted when only rhythmic pattern generation is required. A rhythmic motion like the figure-8 also requires a certain amount of memory because of its intersection in the 2D-plane and because of the non monotonicity in each dimension.

For all experiments, we chose a reservoir size of 400 neurons. The leak rate γ was hand tuned to 0.3. We used a spectral radius of 0.99 (described in Section 2). The connection weights of the input and output to the reservoir, respectively $\mathbf{W}_{\text{i}}^{\text{r}}$ and $\mathbf{W}_{\text{o}}^{\text{r}}$, are scaled by a hand tuned scaling factor of 0.1. These parameters could be optimized by applying grid search. However, the hand tuned parameters were sufficient to demonstrate the capabilities of each RC-network.

We demonstrate[1] the training of multiple motions by generating both a rhythmic and a discrete motion pattern. This is shown in Fig. 2(a), where the input is set to -1 (arbitrarily chosen, trained to generate a discrete motion) and the feedback to the reservoir is initialized with a chosen position $(0, 20)$. After generating the desired discrete pattern, the fixed point attractor is reached. The network stays in this attractor as long as the network input stays unchanged. However, when the network input (indicated by the dashed line in the bottom plot) is switched from -1 to 1, the network reaches its limit cycle attractor after some transient behavior, and stays on its limit cycle attractor during an unchanged input. When the input is changed from 1 to -1 the fixed point attractor will be reached again by generating a discrete pattern. Interestingly, due to symmetry in the network, the generated discrete pattern is a rotated version of the trained one. By adding bias to the reservoir this symmetry is removed. The velocity of the motions depends on the velocity of the training examples. However, the transients between these patterns are unpredictable.

To investigate the robustness of a trained RC-network we perturb the output feedback during 10 time steps by holding the x-coordinate fixed and pulling the y-coordinate to another position. This is shown in Fig. 2(b).

For the discrete motion, two perturbations where introduced for respectively two generation attempts (one illustrated by a gray line, the other by a black one). A perturbation was introduced at time step 30 and 80 for the gray and black pattern, respectively. These plots demonstrate that the motion generation is robust and that after some transient behavior, the target position is reached after all. Afterwards, we notice that the system is staying in its fixed point

[1] http://www.youtube.com/watch?v=WnmHdS38Ae0

ESANN 2012 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2012, i6doc.com publ., ISBN 978-2-87419-049-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100967420.

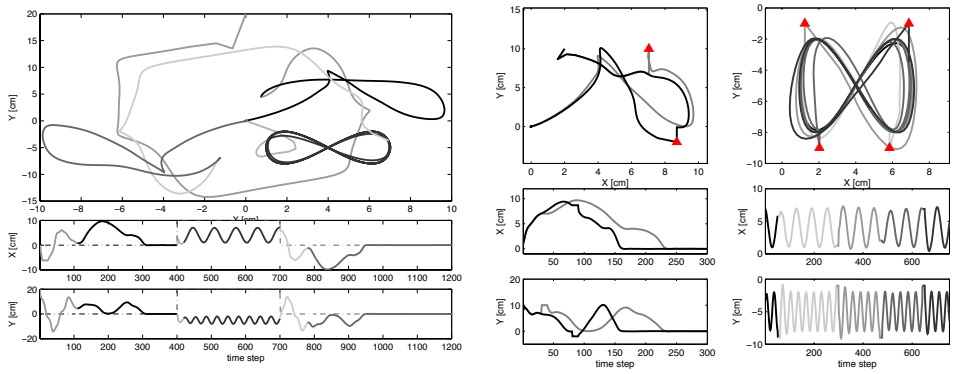

Fig. 2: (a) Plots demonstrating the switching capability. The dashed line in the bottom two plots illustrates the switching moment, set by changing the network input. The transients between these patterns are shown in a lighter gray color while the patterns are shown in darker gray. (b) Plots illustrate the robustness against perturbations (indicated by the triangles). For the discrete pattern, 2 generation attempts of the letter R are shown (in black and gray) with a different perturbation. For the rhythmic motion, 4 perturbations were applied. After each perturbation the color is changed to a different intensity of gray.

attractor for as long as the experiments takes (300 time steps).

For the evaluation of rhythmic motion generation, we perturb the motion at different time steps (time steps 100, 338, 516 and 690). The perturbation frequency is small enough to allow convergence to its limit cycle attractor between two perturbations. To illustrate the transient motions, a different color is used between two perturbations. We notice that, after some transient behavior, the system converges back to its limit cycle attractor. Additionally, because of this limit cycle attractor, the rhythmic motion will continue until the experiment stops (600 time steps).

4 Conclusion

Biological research indicates that advanced motor skills emerge from the combination of a limited amount of motion/motor primitives, which can be discrete or rhythmic patterns. Inspired by these findings, roboticists try to implement a similar modular representation to enhance the motor skills of robots. Such modules need to be capable of generating both discrete and rhythmic patterns. Discrete patterns have typically a fixed point attractor while rhythmic patterns are represented by a limit cycle attractor.

In this work we embedded such attractors into a recurrent neural network which is trained by a Reservoir Computing (RC) approach. We demonstrated the ability to learn both a discrete and rhythmic pattern in a single recurrent

ESANN 2012 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2012, i6doc.com publ., ISBN 978-2-87419-049-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100967420.

neural network. The proposed system was trained to generate letter writing motions and a rhythmic figure-8 which were evaluated and demonstrated on a planar robotic manipulator with 3 rotational joints. Furthermore, we showed the possibility to switch between both pattern types by changing the input to the network. Additionally, the robustness of the pattern generation was investigated by perturbing the feedback to the network for several time steps.

The velocity of the generated patterns is similar to the velocity of the shown training examples. The velocity of the transient behavior between discrete and rhythmic patterns is unpredictable which is undesirable in some robotic applications. Therefore we will investigate the velocity modulation of this transient behavior in future work. Furthermore, we will investigate how human motion patterns can be mapped on patterns for robots and how to sequence such patterns to finally generate advanced motor skills.

5 Acknowledgments

This work was partially funded by a Ph.D. grant of the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen) and the FP7 funded AMARSi EU project under grant agreement FP7-248311.

References

[1] A. d'Avella, A. Portone, L. Fernandez, and F. Lacquaniti. Control of fast-reaching movements by muscle synergy combinations. *The Journal of neuroscience*, 26(30):7791, 2006.

[2] T. Flash and B. Hochner. Motor primitives in vertebrates and invertebrates. *Current Opinion in Neurobiology*, 15(6):660–666, 2005.

[3] S. Grillner and P. Wallen. Cellular bases of a vertebrate locomotor systemá-ásteering, intersegmental and segmental co-ordination and sensory control. *Brain research reviews*, 40(1-3):92–106, 2002.

[4] A.J. Ijspeert, A. Crespi, D. Ryczko, and J.M. Cabelguen. From swimming to walking with a salamander robot driven by a spinal cord model. *Science*, 315(5817):1416, 2007.

[5] A.J. Ijspeert, J. Nakanishi, and S. Schaal. Learning Attractor Landscapes for Learning Motor Primitives. In *NIPS*, pages 1523–1530, 2002.

[6] Herbert Jaeger and Harald Haas. Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless telecommunication. *Science*, 308:78–80, April 2 2004.

[7] Herbert Jaeger, Mantas Lukosevicius, and Dan Popovici. Optimization and applications of echo state networks with leaky integrator neurons. *Neural Networks*, 20:335–352, 2007.

[8] S.M. Khansari-Zadeh and A. Billard. Imitation learning of globally stable non-linear point-to-point robot motions using nonlinear programming. In *Int. Conf. on Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ*, pages 2676–2683. IEEE, 2010.

[9] R. Legenstein and W. Maass. Edge of chaos and prediction of computational performance for neural circuit models. *Neural Networks*, 20(3):323–334, 2007.

[10] D. Verstraeten, B. Schrauwen, M. D'Haene, and D. Stroobandt. An experimental unification of reservoir computing methods. *Neural Networks*, 20:391–403, 2007.

[11] F. Wyffels and B. Schrauwen. Design of a central pattern generator using reservoir computing for learning human motion. *Advanced Technologies for Enhanced Quality of Life*, pages 118–122, 2009.

[12] F. Wyffels, B. Schrauwen, and D. Stroobandt. Stable output feedback in reservoir computing using ridge regression. In *Int. Conf. on Analog Neural Networks (ICANN)*, 2008.