A Dynamical System for Online Learning of Periodic Movements of Unknown Waveform and Frequency

Andrej Gams, Ludovic Righetti, Auke Jan Ijspeert and Jadran Lenarčič

***Abstract*—The paper presents a two-layered system for learning and encoding a periodic signal onto a limit cycle without any knowledge on the waveform and the frequency of the signal, and without any signal processing. The first dynamical system is responsible for extracting the main frequency of the input signal. It is based on adaptive frequency phase oscillators in a feedback structure, enabling us to extract separate frequency components without any signal processing, as all of the processing is embedded in the dynamics of the system itself. The second dynamical system is responsible for learning of the waveform. It has a built-in learning algorithm based on locally weighted regression, which adjusts the weights according to the amplitude of the input signal. By combining the output of the first system with the input of the second system we can rapidly teach new trajectories to robots. The systems works online for any periodic signal and can be applied in parallel to multiple dimensions. Furthermore, it can adapt to changes in frequency and shape, e.g. to non-stationary signals, and is computationally inexpensive. Results using simulated and hand-generated input signals, along with applying the algorithm to a HOAP-2 humanoid robot are presented.**

I. Introduction

THIS paper presents a new system for online learning and encoding periodic signals, and its application to the learning and repeating of the trajectories on a HOAP-2 humanoid robot.

Various methods of performing periodic or rhythmic movements with robots exist, ranging from simple memorizing of the whole trajectory [1], on-line generation or modification of trajectories depending on the task and the state of the actuated device [2], or for example generation of periodic trajectories using singular values decomposition and vector fields [3].

For generating trajectories for device specific tasks and in combination with feedback, Williamson [4] proposed a system based on the Matsuoka nonlinear oscillator, with which he could effectively control rhythmic motion of the arms of a humanoid robot.

Another approach to generating trajectories is to imitate recorded motion, for example by learning it using techniques that include gradient-descent learning algorithms for recurrent neural networks [5]. A different approach is the learning of attractor landscapes of point attractors to form control policies [6-9]. The essence of this approach is to use a canonical simple dynamical system, to anchor Gaussian basis functions in its phase, and to learn a weight vector that multiplies the basis functions thus forming arbitrary smooth new attractor landscapes [6,7]. Nonlinear regression techniques were employed to learn the weight vector. The concept was expanded from discrete to periodic movements using the phase of the nonlinear oscillators to anchor the basis functions [9].

The limitation of such an approach for periodic movement is that the period has to be known [9] – either specified beforehand, or extracted from the signal using signal processing, e.g. FFT. In case of frequency changes over time, an alternative solution to signal processing is to use one of the feedback quantities of the actuated system to anchor the weight vector, for example the measured torque for performing a periodic task [10]. This, on the other hand, imposes great restrictions to the generality of the approach.

This paper presents a system that surpasses these shortcomings and further expands the approach of anchoring the weight vector in phase space by employing adaptive frequency oscillators for the dynamical canonical system.

Adaptive frequency oscillators have the property to learn the frequency of any periodic input signal, thus enabling us on-line adaptation to the frequency of the input signal [11, 12]. As the whole process of frequency extraction and adaptation is totally embedded into the dynamics of the adaptive frequency oscillator [11], we do not need to specify any free parameters, such as a time window, which is often the case in signal processing algorithms, nor do we need to perform any transformations as for example FFT [12].

The key idea is the splitting of the two dynamical systems – the canonical dynamical system for the extraction of the frequency, and the output dynamical system for the learning of the waveform. This makes the system simpler than similar systems based on neural networks [5] and computationally inexpensive. Combining the two dynamical systems provides us with a system that rapidly encodes trajectories of previously undetermined frequencies, works online, and can cope with non-stationary signals as well as the changing of the waveform.

In the next sections we first present the structure of the proposed system (section IIA), followed by the presentation of the building-blocks of the system – the adaptive frequency oscillators in a feedback structure as the canonical dynamical system (section IIB), and the output dynamical

Andrej Gams is with the "Jozef Stefan" Institue (IJS), Jamova cesta 39, 1000 Ljubljana, Slovenia (e-mail: andrej.gams@ijs.si).

Ludovic Righetti and Auke Jan Ijspeert are with the School of Computer and Communication Science, École Polytechnique Fédérale de Lausanne (EPFL), Station 14, 1015 Lausanne, Switzerland.

Jadran Lenarčič is with the "Jozef Stefan" Institue (IJS), Jamova cesta 39, 1000 Ljubljana, Slovenia .

system for the learning of the waveform (section IIC). In section III we present the experimental results for both simulated (A) and hand-generated signals (B) along with some general practicality issues and the final system structure. Applying the algorithm to a 25 degree of freedom (DOF) humanoid robot HOAP-2 is presented as well (C). The paper finishes with a short discussion and possible future improvements (section IV).

II. Adaptive Nonlinear Oscillators as Control Policies

In this section we first present the structure of the proposed system, followed by the detailed explanation of the two building blocks: 1: the Canonical dynamical system for the frequency adaptation and 2: the Output dynamical system for the learning of the waveform.

A. Structure of the system

Fig. 1 shows the structure of the proposed system for the learning of a periodic signal without any information on the frequency. The algorithm can work in parallel for several dimensions.

As shown in Fig. 1, an arbitrary periodic signal is provided as input into the canonical dynamical system – adaptive frequency oscillator feedback structure of one or several adaptive oscillators – for each of the Q-dimensions. The common frequency Ω and the phase Φ of the oscillator at this frequency, are fed to the output dynamical system. The output dynamical system adjusts the weight vector, and is marked by ILWR for Incremental Locally Weighted Regression. Both frequency adaptation and learning of the weights work in parallel. The output of the algorithm can be for example joint coordinates of the robot or position in task space, and the weight vector w_i, which we can use to replay the learned trajectories.

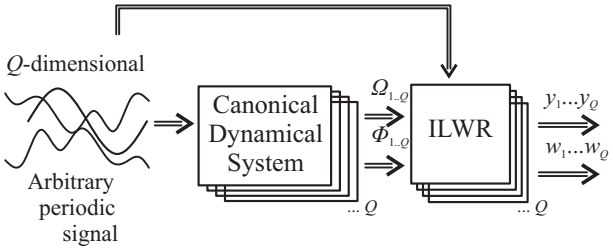

Fig. 1. Proposed structure of the system for learning arbitrary periodic signals with no knowledge of the frequency or the waveform. The two-layer composition of the system is clearly seen: the Canonical dynamical system as the first layer and the Output dynamical system (ILWR block) for the learning, as the second layer.

B. Canonical dynamical system

Instead of using a simple nonlinear oscillator as a canonical system (as originally done in [9]) to provide a phase signal to the output dynamical system, we use an adaptive nonlinear oscillator. We use the phase oscillator, see for example [13], to which we apply the adaptive frequency learning rule as introduced in [11], combining it with the feedback structure [12] shown in Fig. 2. The equation of the adaptive phase oscillator in a feedback structure is governed by the following equations:

$$\dot{\phi}_i = \omega_i - Ke(t)\sin(\phi_i) \quad (1)$$

$$\dot{\omega}_i = -Ke(t)\sin(\phi_i) \quad (2)$$

$$e(t) = y_{demo}(t) - \hat{y}(t) \quad (3)$$

$$\hat{y} = \sum_{i=1}^{M} \alpha_i \cos(x_i) \quad (4)$$

$$\dot{\alpha}_i = \eta \cos(\phi_i) e(t)\,, \quad (5)$$

where K is the coupling strength or constant, ϕ is the phase, $e(t)$ is the input into the oscillators, $y_{demo}(t)$ is the input signal into the system, $\hat{y}(t)$ is the weighted sum of the outputs of the oscillators, M is the number of oscillators, α_i is the amplitude associated to the i-th oscillator, and η is a learning constant. Throughout the paper we use K=20 and η=1. The oscillatory behavior of i-th oscillator can be explained by $\mathrm{mod}(\phi_i, 2\pi)$.

Each of the oscillators of the structure receives the same input signal, which is the difference between the signal to be learned and the signal already learned as in Eq. (3). As a negative feedback loop is used, this difference approaches zero as the weighted sum of separate frequency components, Eq. (4), approaches the learned signal, and the frequency stabilizes. Such a feedback structure can learn several frequency components of the input signal [12], but most importantly for now, resulting from Eq. (5), it enables the frequency of the oscillator to stabilize at ω_t as $t \to \infty$.

Frequency adaptation results for different target frequencies are presented in Fig. 3. The top plot shows the result of adaptation to a stationary signal and as we can see, the output frequency stabilizes very quickly at the target frequency. In general the speed of convergence depends on the coupling strength K, see also [12]. The middle plot demonstrates the property of the used structure to continuously adapt to a non-stationary signal, such as a chirp signal. The bottom plot shows the ability to adapt to step change in frequency of the input signal. As we can see, the used adaptive frequency phase oscillator can cope with the change in frequency of the input signal.

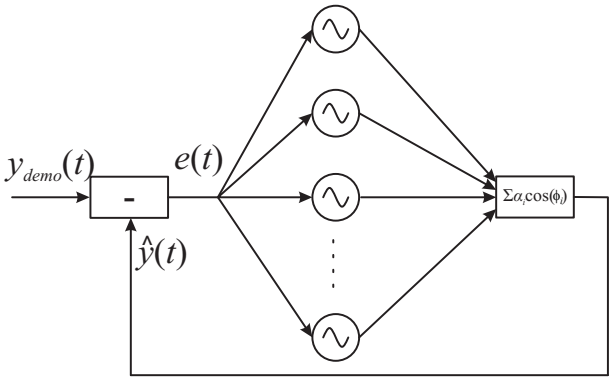

Fig. 2. Feedback structure of a network of adaptive phase oscillators. All oscillator receive the same input and have to be at different starting frequencies to converge to different final frequencies. Refer also to text and Eqs. (1-5).

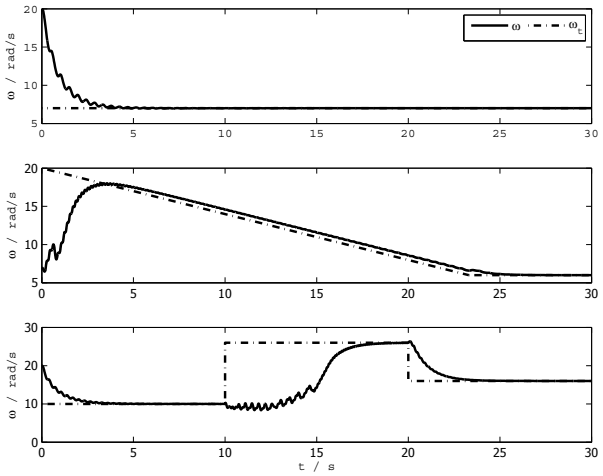

Fig. 3. Frequency adaptation of the adaptive phase oscillator in a feedback structure to periodic input signal y_{demo}=sin($\omega_t t$). The target frequencies (ω_t) are marked with dash-dot lines and the results with solid lines. Top: Adaptation to a stationary signal. The frequency adapts from the initial frequency ω_0=20rad/s to the target frequency ω_t=7rad/s. Middle: Adaptation to a chirp signal, ω_0=7rad/s, ω_t=(20-0.6t)rad/s for t<23.3s and ω_t=6rad/s for t>23.3s. Bottom: Adaptation to step change with ω_0=20rad/s, ω_t=10, 26 and 16rad/s, step changes occurring at t=10s and 20s.

This proves especially useful when adapting to the frequency of a hand-generated signal, which is never stationary, but always slightly changing. Additionally, we can exploit it to change the frequency of the input signal on-line, allowing us to modify the learning trajectory.

Fig. 4 shows the results of the canonical dynamical system adapting to an input signal with three frequency components. As we can see, the three used oscillators adapt to the three components of the signal. In general oscillators tend to adapt to the components they are closest to, as they fall into their basins of attraction [12]. The feedback structure can also identify the offset (as a component with null frequency).

Of the frequencies the oscillators adapt to we choose the lowest non-zero frequency. The chosen frequency, marked by Ω and the phase of an oscillator at that frequency Φ are sent to the output dynamical system.

C. Output dynamical system

The output dynamical system is used to learn the waveform of the input signal. For the sake of simplicity let us assume we only want to learn a one-dimensional signal. The following dynamics specify the attractor landscape of a traj-

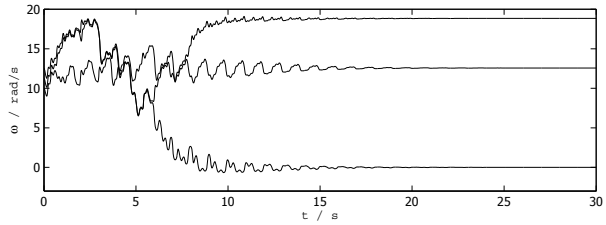

Fig. 4. Frequency adaptation of a feedback structure with three adaptive phase oscillators to an input signal $y_{demo}(t)$=0.5+0.8cos(4πt)+2sin(6πt). The initial conditions of the adaptive frequency oscillators are ω_0=[3.2, 3.4, 4] rad/s. The canonical system successfully adapts to all three frequency components.

ectory y towards the goal g with our canonical dynamical system providing us with the phase signal Φ to the function Ψ_i of the control policy:

$$\frac{1}{\Omega}\dot{z} = \alpha_z(\beta_z(g-y)-z) + \frac{\sum_{i=1}^{N}\Psi_i w_i r}{\sum_{i=1}^{N}\Psi_i} \tag{6}$$

$$\dot{y} = \Omega z \tag{7}$$

$$\Psi_i = \exp(h(\cos(\Phi - c_i)-1)). \tag{8}$$

g is the anchor for the oscillatory trajectory, in our case set to g=0, Ω is the output of the canonical dynamical system, and N is the number of Gaussian-like kernel functions given by Eq. (8). α_z and β_z are positive constants, for all the results set to α_z=8 and β_z=2, with the ratio 4:1 ensuring critical damping. r is the amplitude control parameter. Such a system without the nonlinear term presents a globally stable second-order linear system as a unique point attractor [9]. As the input into the learning algorithm we use triplets of position, velocity and acceleration ($y_{demo}(t)$, $\dot{y}_{demo}(t)$ and $\ddot{y}_{demo}(t)$). Equation (6) can be rewritten as:

$$\frac{1}{\Omega}\dot{z} - \alpha_z(\beta_z(g-y)-z) = \frac{\sum_{i=1}^{N}\Psi_i w_i r}{\sum_{i=1}^{N}\Psi_i}, \tag{9}$$

and formulated as a supervised learning problem with the target function

$$f_{target} = \frac{1}{\Omega^2}\ddot{y}_{demo} - \alpha_z(\beta_z(g-y_{demo}) - \frac{1}{\Omega}\dot{y}_{demo}),$$

which is obtained by matching y to y_{demo}, z to $\dot{y}_{demo}/\Omega$ and $\dot{z}$ to $\ddot{y}_{demo}/\Omega$. The learning of the weights is based on the well known incremental locally weighted regression [14], and is done incrementally for every time sample using equations (10-12):

$$w_i^{t+1} = w_i^t + \Psi_i P_i^{t+1} r(t) e_r(t) \tag{10}$$

$$P_i^{t+1} = \frac{1}{\lambda}\left(P_i^t - \frac{P_i^{t2} r_i^{t2}}{\frac{\lambda}{\Psi_i} + P_i^t r_i^{t2}}\right) \tag{11}$$

$$e_r(t) = f_{target}(t) - w_i^t r(t) \tag{12}$$

where Ψ_i are Gaussian-like kernel functions given in Eq. (8) with c_i equally spaced between 0 and 2π, and h determining the width of separate kernel function, set to h=2.5N throughout the paper. w_i are the weights and λ=0.95 is the forgetting factor. The recursion is started with w_i=0 and P_i=1.

Fig. 5 shows the time evolution of the system with no frequency adaptation, and the weight parameters w_i adjusted to fit the trajectory $y_{demo}(t) = \sin(\omega t) + \cos(2\omega t)$ with ω=2π rad/s. As we can see in the top-left plot, the input signal and the reconstructed signal match very closely, but not completely. This can be in part corrected by increasing the number of Gaussian-like functions.

Fig. 6 shows the results while the frequency is still adapting to the input. This has a double effect on the genera-

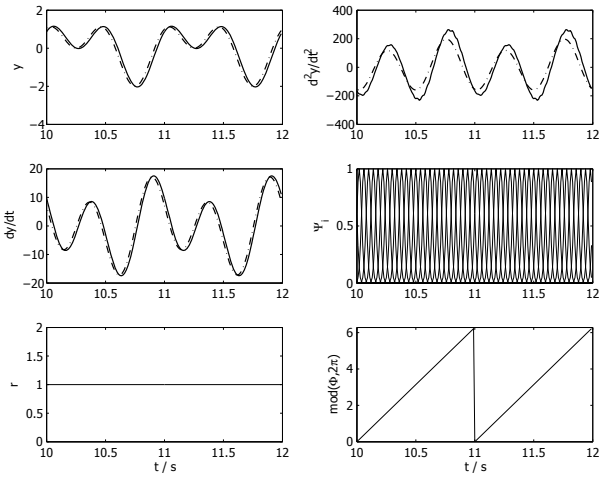

Fig. 5. The result of continuous learning and repeating of the signal with no frequency adaptation and $\Omega=2\pi$ rad/s. In all the plots the input signal is the dash-dot line while the repeated signal is the solid line. In the middle-right plot we can see the evolution of the kernel functions and in the bottom right plot the phase of the oscillator. Since we are using the phase oscillator, the amplitude is in this case r=1, as can be seen in the bottom-left plot.

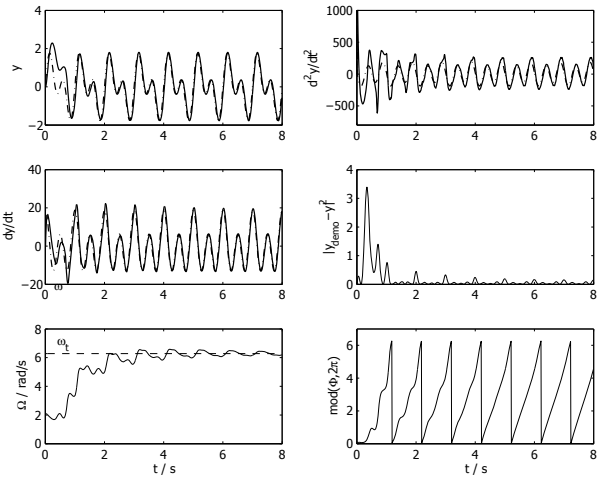

Fig. 6. The result of learning while the adaptive frequency oscillator is still adapting to the frequency of the input. Again the input signal is the dash-dot line. The middle right plot shows the square of the difference of the input and the learned signal. The bottom left plot shows the frequency of the adaptive phase oscillator, for clarity reasons only the common frequency is shown. 2 oscillators were used in the canonical dynamical system; the higher frequency stabilizes at 4π rad/s.

ted output, yet poses no problem at all, as we only have to wait it out. Firstly, (Fig. 6, $t < 4$s) the frequency is incorrect and the target trajectory is calculated wrong, and secondly, the phase changes differently in consecutive periods which leads to incorrect mapping of the input signal to the weight vector of the Gaussian-like functions. This can be clearly seen in Fig. 6, where we show the learning and frequency adaptation of signal $y_{demo}(t)=\sin(2\pi t)+\sin(4\pi t)$. As we can see towards the end of the plots ($t > 4$s), once the frequency stabilizes (bottom-left), so does the phase (bottom-right) and the matching of the input and the learned signal (top-left).

III. Experimental evaluation

We tested the proposed rhythmic control policy by imitating 2D patterns with a robot. In the following, first the results of generated data are presented, which are followed by results of imitating the motion of the tip of the human arm, implemented also on a humanoid robot.

A. Simulated input signals

We simulated different movements by inputting into our system a two dimensional signal which represents the movement in the *xy* plane. We used a canonical dynamical system for each of the dimensions, each with 3 adaptive phase oscillators in a feedback structure as a design choice. After eliminating the offset we can choose between 2 other frequencies and we choose the lower one, because if $\omega_{high}=k\omega_{low}$, where $k \in \mathbb{N}$, this is the common frequency.

Fig. 7 shows the results for a 2D input signal $x_{demo}(t)=\sin(2\pi t), y_{demo}(t)=\sin(4\pi t)+\sin(8\pi t)$. As we can see, the frequency adapts quite fast and we cannot see the difference in the plots of the separate components within 2 seconds. The *y* dimension of the input signal has two frequency components: ωt and $2\omega t$. Should there be more frequency components than adaptive frequency oscillators in the feedback structure, the adaptation of the frequency would not be as good since the feedback structure would not "eliminate" some of the components and $e(t)$ in eq. (3) would never go to zero. Consequently the resulting frequency would oscillate slightly. This can pose a problem, but with only slight oscillations, the error in learning of the input signal is not substantial. This is clearly seen in the results of learning a hand-generated signal, presented in Fig. 8, as will be discussed next.

B. Hand-motion results

We implemented a Matlab/Simulink environment interface which enables us to record the motion of the computer mouse at 100Hz. Fig. 8 shows the results of continuous learning of the shape of the hand motion while mimicking a figure-8. As we can see in the plots, the learned signal is very close to the input signal. Since we do not input the same signal all the time, but slightly different for every period due to the inaccuracy of the hand, the system continuously adapts to the input signal with a forgetting factor λ. That is why the system has some delay in adapting to the shape.

When replaying the signal using a learned set of weights and frequencies, we have to take into consideration that we are trying to recreate two separate signals which belong to the same system. In other words, the frequencies for separate dimensions are in a roughly constant ratio. If this is not the case, the signals drift and the result is not a closed loop. While we do online learning, this is constantly corrected by the learning and the frequency adaptation. When replaying, on the other hand, a common 2D loop has to be considered and the ratio of the frequencies corrected. The correction can also be implemented during the online process and in such case the control scheme is as presented in Fig. 9.

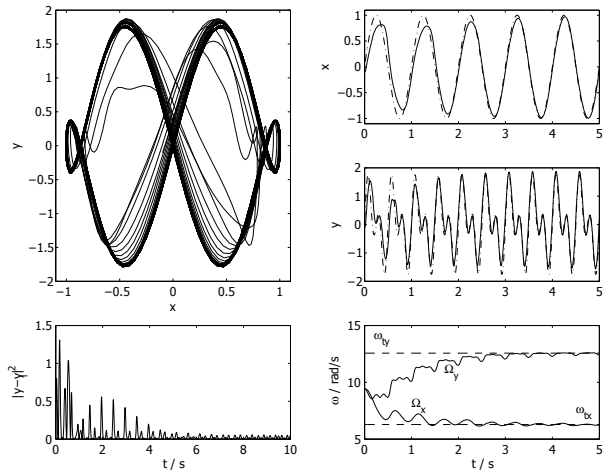

Fig. 7. Results for a simulated two-dimensional input $x_{demo}(t) = \sin(2\pi t)$, $y_{demo}(t) = \sin(4\pi t) + \sin(8\pi t)$. The left-top plot shows the *xy* plane with the dash-dot line as the input and the solid line as the learned signal. The two top-right plots show first 6 seconds of the process for each of the dimensions with the dash-dot line as the input signal while the bottom-right plot shows the adapted frequencies for both dimensions. The bottom-left plot shows the square of absolute error for the *y* dimension.

As we can see, we input a *Q*-dimensional signal into *Q* parallel adaptive oscillator feedback structures. The outputs, $\Omega_{1...Q}$ and $\Phi_{1...Q}$, are fed into a logic block, which determines if the ratio is within the defined interval of an integer number, and outputs a common frequency. For the common phase, we have two options, as we can either output one common phase for all the *Q* dimensions or we can generate separate phases for separate dimensions. Each has its own advantages. Using a common phase preserves more information, such as slightly different amplitude every other peak, but is for one of the dimensions stretched over an interval of *k* periods, where *k* is Ω_2/Ω_1 and $\Omega_2 > \Omega_1$. It also needs more kernel functions for accurate reconstruction. Using separate phases for separate learning algorithms, on the other hand, requires the generation of the phases in the determined frequency ratio.

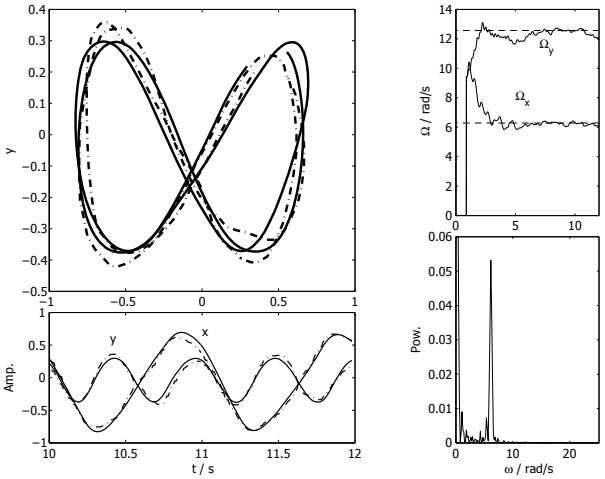

Fig. 8. Results for hand-generated signal. The top left-plot shows two seconds of the input (dash-dot line) and the learned output signal (solid). The bottom-left plot shows the same signals on a common axis. The top-right plot shows the frequency outputs of the two feedback structures used. The bottom right plot shows the power spectrum of the *x* dimension of the input signal.

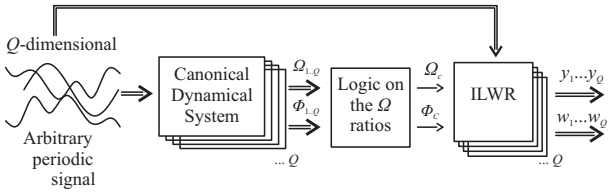

Fig. 9. Modified structure of the system for learning arbitrary periodic signals with no knowledge of the frequency, with an introduced logical block to determine frequency ratios.

By using the "Logic on the ω ratios" block as in Fig. 9 we can also learn signals with frequency components in a ratio $\Omega_1 = k\Omega_2$ with $k \notin \mathbb{N}$, such as 3Hz and 4Hz, with their common frequency at 1Hz. While the adaptive frequency oscillator will only converge to either the 3Hz or the 4Hz component, as the power spectrum of the signal has no others, we use their highest common divider as the common frequency, or if their ratio is close to an integer number, we round it. The same problem of frequencies could also pose problems when combining different dimensions of a signal.

Alternatives to using a logical block include, for example, summing the signals of the two dimensions and using only one canonical dynamical system for frequency extraction, or coupling the signals of separate dimensions. The draw-back of summing the signals is in the complex signal that serves for frequency extraction, requiring the expansion of the canonical dynamical system. Coupling, on the other hand, makes more sense if the signals are not of one trajectory, e.g. when performing two-handed drumming.

Fig. 10 shows the results of using one common frequency and one common phase to repeat a learned trajectory of a square.

C. Implementation on HOAP-2 robot

We used the 25 degree of freedom HOAP-2 humanoid robot made by Fujitsu for a demonstration of the learning algorithm and the ease of its use.

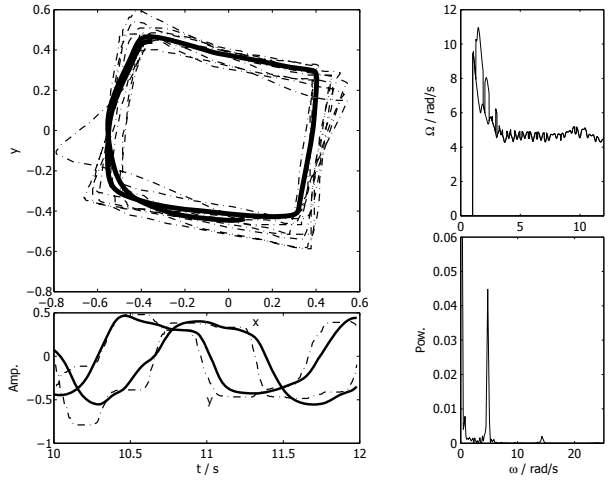

Fig. 10. Results of learning with determined ratios and using only one phase signal for both the dimensions. In the top-right plot the switching between the two frequencies can be seen from the start of the experiment. The bottom-right plot shows the power spectrum of the *x* dimension. Refer also to Fig. 8 for plot descriptions.

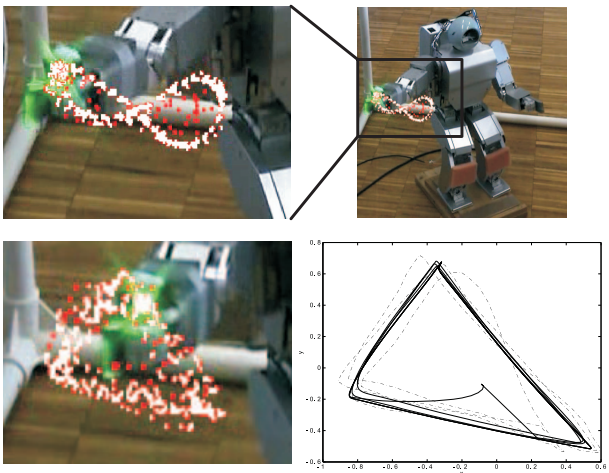

Fig. 11. Humanoid robot HOAP-2 robot repeating the learned figure-8 trajectory by mapping the learned signal to the shoulder joint of the right hand of the robot (top), and a triangle (bottom). Input and learned trajectory for triangle are shown bottom-right. To track the position of the tip of the hand we used a green LED and a color based image processing filter.

For the demonstration we mapped the x and y dimensions of the input signal from the mouse to two joints of the shoulder. With the arm fully extended and the range of the motion not too big, we can approximate the movement of the tip of the robot as moving on a vertical plane. There are different ways of controlling the robot [15]. We separated the tasks of controlling the robot on one side, and the learning algorithm including the interface on the other, to two computers. The first computer running Matlab/Simulink does the acquisition of the input signal and the learning algorithm with both frequency adaptation and adjusting the weights, and transmits the learned signals via the network to the second computer controlling the motions of the robot in a RT-Linux environment. This computer receives the signals in user-space and copies them into the real-time module, which moves the joints of the robot using position control. Fig. 11 shows the HOAP-2 robot and the learned trajectory which we tracked using position tracking software. Besides the figure-8 we successfully repeated other arbitrary trajectories, such as a square, shown in Fig. 10, or a triangle.

IV. Conclusion

The described algorithm presents several interesting properties for encoding and replaying trajectories. These are: the ease of representing and learning, compactness of the presentation, ease of use for replaying. The system also works as a filter, which makes it suitable for signals with considerable noise, such as signals generated by hand. It works completely online, adapting to stationary and non-stationary signals, and is easy to use. As the output of the system is also the weight vector, we can categorize different trajectories and could in the future use it as means of measuring their similarities.

The system also works for complex signals, needing only the common frequency. This is one of the advantages over other systems, for example as in [12]. Another function of the algorithm is also the ability to compress the information of a Q-dimensional periodic signal into a vector of size ($N \times Q$), where N is the number of Gaussian-like kernel functions. Though not discussed in this paper, the learned signals can be modulated on-line by changing the frequency, amplitude and anchor parameters as a direst result of using dynamical systems.

In the future we would like to expand the system to perform real-world tasks, such as for example drumming. Another direction to follow is to try and eliminate the logical block in our structure.

Acknowledgment

The work described in this paper was partially funded by the European Commission's Cognition Unit, project no. IST-2004-004370: RobotCub, and the EU Cognitive Systems project PACO-PLUS (FP6-2004-IST-4-027657).

References

[1] C.H. An, C.G. Atkeson, and J.M. Hollerbach, "Model-based control of a robot manipulator," MIT Press, 1988.

[2] L. Žlajpah, "Robotic yo-yo: modelling and control strategies," Robotica, 24(2): 211-220, 2006.

[3] M. Okada, K. Tatani, Y. Nakamura, "Polynomial design of the nonlinear dynamics for the brain-like information processing of whole body motion," In IEEE Int. Conf. on Robotics and Automation (ICRA 2002), pp. 1410-1415.

[4] M. M. Williamson, "Neural control of rhythmic arm movements," Neural Networks 11: 1379-1394, 1998.

[5] B. A. Pearlmutter, "Gradient calculations for dynamic recurrent neural networks: a survey," IEEE Transactions on Neural networks, 6(5): 1212-1228, 1995.

[6] A. J. Ijspeert, J. Nakanishi, T. Shibata, S. Schaal, "Nonlinear Dynamical Systems for Imitation with Humanoid Robots," Proc. of the IEEE-RAS International Conference on Humanoid Robots, 2001

[7] A. J. Ijspeert, J. Nakanishi, S. Schaal, "Learning Attractor Landscapes for Learning Motor Primitives," In S. Becker, S. Thrun, and K. Obermayer, editors, Advances in Neural Information Processing Systems 15 (NIPS2002), pages 1547-1554, 2002.

[8] A.J. Ijspeert, J. Nakanishi, S. Schaal, "Movement imitation with nonlinear dynamical systems in humanoid robots," Proc. of the IEEE International Conference on Robotics and Automation (ICRA2002), 1398-1403, 2002.

[9] A.J. Ijspeert, J. Nakanishi, S. Schaal, "Learning rhythmic movements by demonstration using nonlinear oscillators," Proc. of the IEEE/RSJ Int. Conference on Intelligent Robots and Systems (IROS2002), 958-963, 2002.

[10] A. Gams, L. Žlajpah, J. Lenarčič, "Imitating human acceleration of a gyroscopic device," Robotica, vol. 25(4), pp. 501-509, 2007.

[11] L. Righetti, J. Buchli, A.J. Ijspeert, "Dynamic hebbian learning in adaptive frequency oscillators," Physica D, 216(2), 269-281, 2006.

[12] L. Righetti, A.J. Ijspeert, "Programmable central pattern generators: an application to biped locomotion control," Proc. of the 2006 IEEE International Conference on Robotics and Automation, 2006.

[13] J. Buchli, L. Righetti, and A.J. Ijspeert, "Engineering entrainment and adaptation in limit cycle systems - from biological inspiration to applications in robotics," Biological Cybernetics, 95(6):645-664, 2006.

[14] S. Schaal and C.G. Atkeson, "Constructive incremental learning from only local information," Neural Computation, 10(8):2047-2084, 1998.

[15] A. Kos, "Izdelava programskega vmesnika za vodenje humanoidnega robota HOAP-3," ZAJC, Baldomir (ed.), TROST, Andrej (ed.). Zbornik šestnajste mednarodne Elektrotehniške in računalniške konference ERK 2007, 24. - 26. september 2007, Portorož, Slovenija, IEEE Region 8, Slovenska sekcija IEEE, 2007, zv. B, pp. 141-144.