The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

An Imitation Model based on Central Pattern Generator with application in Robotic Marionette Behavior Learning

M. Ajallooeian, M. Nili Ahmadabadi, B. N. Araabi, H. Moradi

***Abstract*— Most of the Central Pattern Generator (CPG) models are based on defining explicit dynamical systems and finding the appropriate parameters. In this paper, we propose a novel CPG model that is based on altering a nonlinear oscillator to obtain desired limit cycle behavior. This CPG model benefits from an explicit basin of attraction and also fast convergence behavior. The presented CPG model is used in an imitation model that tries to learn the proper periodical behavior by looking at a mentor. First, a mentor performs the desired periodical behavior. Then, a hand-eye coordination process, inspired from infant babbling, is initiated to extract proper motor actions from what is observed. The extracted motor actions are finally embedded into the CPG model for smooth reproduction. This imitation model is implemented on a robotic marionette behavior learning task. The outcome of the final performance of the robotic marionette is behaviorally understandable smooth actions.**

I. Introduction

Central pattern generators were first found as neural circuits located in the spine of vertebrates. Their task is to coordinate the muscles during periodic movements [1], [2]. Inspired from biology, several models of CPGs were implemented to encode periodical trajectories [3]-[6]. All of these models are based on defining explicit dynamical equations and finding appropriate parameters of such equations. In [3] an applicable model of programmable CPG is presented where a Fourier series representation of coupled adaptive oscillators is used to learn arbitrary signals. Also, in [4] a nonlinear oscillator model that modulates a canonical simple limit cycle system with statistical learning methods is presented and the ability to learn arbitrary trajectories is proved.

In this paper, we present a novel model of CPG that can learn arbitrary periodical trajectories. The main idea of our CPG model is to construct a nonlinear window that is able to change the behavior of another oscillator. One advantage of our CPG model is that different types of oscillators can be used to be altered. Also our model includes an explicit and definable basin of attraction which helps fast convergence to the desired limit cycle.

Proposed CPG model is used in an imitation procedure. The imitation procedure tries to reproduce mentors periodical actions. A similar work is done in [4] where a motion capture device is used to transfer desired trajectory from a mentor to the robot that imitates the behavior. But imitation (or *emulation* as discussed in [7]) is to see the desired behavior and find the appropriate action while this appropriate action is unknown. Our imitation model introduces a hand-eye coordination process inspired from infant babbling that extracts proper action from what is seen from mentor. The extracted action signal is then fed to the CPG model so the desired behavior can be smoothly reproduced.

To summarize, a mentor performs the desired behavior first. Then hand-eye coordination process extracts the proper action from what is observed. Finally, extracted action signal is embedded into the CPG model for smooth reproduction. Proposed model is used for a robotic marionette to learn the desired behavior.

The rest of this paper is organized as follows: section II gives an outline of the imitation model. Had-eye coordination process is introduced in section III. Section IV belongs to description of the new CPG model. Experimental results are discussed in section V.

II. Outline of The Imitation Model

Our proposed model learns the desired behavior by looking at mentor's action. The whole learning scenario could be described as shown in Fig. 1. First a mentor performs the desired behavior (as in the marionette playing, human puppeteer freely manipulates the marionette through strings). Since the mentor's body, or the manipulation mechanism, is different from the imitator, the performed motor action by the mentor is not the trajectory that is meant to be imitated by the imitator. The outcome of the mentor's action is seen through a camera and the motion path is extracted. Then a hand-eye coordination process begins and tries to find the relation between the motion of the end-effector of the imitator and the underlying motor system actions. This hand-eye coordination process is later

Manuscript received March 9, 2009.

Mostafa Ajallooeian is with Control and Intelligent Processing Center of Excellence, ECE Dept., University of Tehran, Iran; phone: +98-912-219-29-11; e-mail: ajal@ut.ac.ir (corresponding author)

Majid Nili Ahmadabadi is with Control and Intelligent Processing Center of Excellence, ECE Dept., University of Tehran, Iran, and School of Cognitive Science, IPM; e-mail: mnili@ut.ac.ir

Babak N. Araabi is with Control and Intelligent Processing Center of Excellence, ECE Dept., University of Tehran, Iran, and School of Cognitive Science, IPM; e-mail: araabi@ut.ac.ir

Hadi Moradi is with Control and Intelligent Processing Center of Excellence, ECE Dept., University of Tehran, Iran; e-mail: moradih@ut.ac.ir

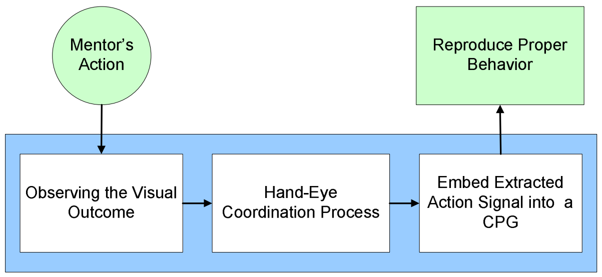
Mentor's Action
Reproduce Proper Behavior
Observing the Visual Outcome
Hand-Eye Coordination Process
Embed Extracted Action Signal into a CPG

Fig. 1. The whole learning scenario. A human puppeteer moves the marionette freely. Outcome is seen and a hand-eye coordination process tries to find the relation between what is seen and what is done. Finally, the proper action is embedded into a CPG model for later smooth preproduction.

described in section III. After completion of the hand-eye coordination process, desired visual outcome is mapped to the causing motor action. Since these motions are assumed to be periodic, a CPG model would be a good choice to embed action signals onto. The structure of the CPG model used for this goal is described in section IV. As a result, trained CPG is able to reproduce desired motions in a smooth and robust manner.

III. HAND-EYE COORDINATION

There is evidence that shows a self-learning process in infants. This process is called babbling. The concept of babbling is generally in the scope of language acquisition studies where an infant tries to experiment with uttering sounds of language but not yet reproducing any meaningful words [8]. But babbling could be seen as both motor and language skills [9]. In motor babbling, it seems that an infant tries to shake his/her limbs to learn about his/her body, in early stages of development. For instance, in the first year, infants produce rhythmic repeated movements of limbs and body [10]. Recent works imply that these early movements may continually emerge to skilled behaviors like reaching for a toy [11].

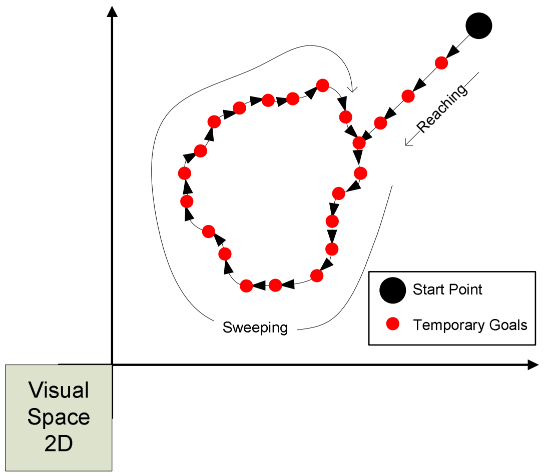

Fig. 2. Hand-eye coordination process. Starting from an initial point, the joints are randomly shaken to reach temporary goals. Temporary goals are set to reach and sweep the desired path.

TABLE I
HAND-EYE COORDINATION ALGORITHM

```
1.  start from a random configuration;
2.  set a temporary goal near to the current location of the end-
    effector of the imitator;
3.  generate small random movements in the robot joints;
4.  if the new end-effector position is closer to the temporary goal
    goto 5, else goto 3;
5.  if the new end-effector position is close enough to the
    temporary goal, update the temporary goal toward reaching
    and sweeping the desired track;
6.  terminate when last the temporary goal is met.
```

Our hand-eye coordination model takes advantage from infant babbling. The proposed algorithm is based on shaking the imitator's joints and updating the state of the imitator whenever the result is interesting. A pseudocode for the proposed algorithm is presented in Table 1. First, imitator's joints are randomly set to an arbitrary position. While this initial position in visual space is not on the desired path, the algorithm tries to make the end-effector of the imitator closer to the path and after reaching the path, the path is followed. Since the relation between the imitator's end-effector position and what is seen is unknown, following the desired path is not straightforward. The solution is to make temporary goals, like when we move the toy for an infant and let him/her follow it. When the temporary goal is set, imitator first tries random bounded joint movements in order to find an approximately close point to the temporary goal. This process is repeated until the distance between imitator's end-effector and the temporary goal is small enough. After that the temporary goal is updated. This whole process is repeated until the desired path is swept completely (Fig. 2).

Reaching data consisting end-effector positions and correspondent joint angles, is collected while the above process is executed. Then a Multi Layer Perceptron (MLP) is trained to estimate the relation between the end-effector position (from the point of view, i.e. camera), and the angles of the imitator joints using the collected data. The estimated relation could be somehow similar, but in an opposite way, to the work done in [12] where a *forward* relation between what is done and what is seen is estimated using BBN (Bayesian Belief Network).

It may be of question why the relation between end-effector position and joint angles are only calculated on a desired trajectory and not on the whole working space of the imitator. There are three reasons to this. First, learning in all of the working space is a time consuming process; compare data gathering in a strip of space compared to the whole space. Second, specific actions are usually the target of imitation systems; it is not required to learn all possible actions. Finally, there are different sets of joint

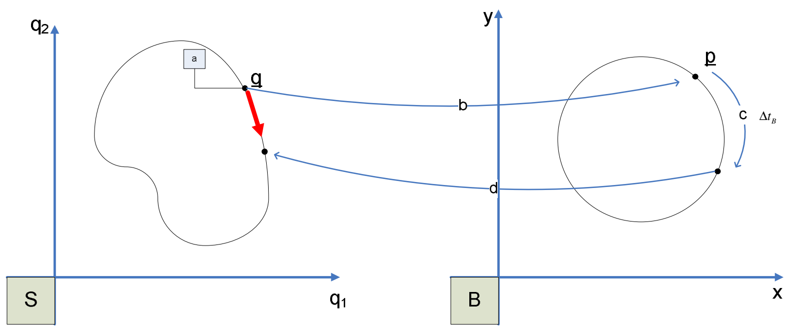

Fig. 3. Generating desired limit cycle. **a.** an initial point $\underline{q}$ is selected in S ; **b.** $\underline{q}$ is mapped to $\underline{p}$ using forward map; **c.** $\underline{p}$ is moved in B using a small time step Δt_B ; **d.** updated $\underline{p}$ is mapped to the updated $\underline{q}$ using backward map. The red bold arrow is for the corrected ***'d'*** step discussed in section IV.B.

configurations that lead to the same end-effector position in visual space. This causes the relation between end-effector position and joint positions not to be a function (single output for any input). In this case, a disambiguation pre-process, or a training trick is needed.

IV. Central Pattern Generator Model

A. *The Model*

The main idea of the proposed CPG model is to alter a nonlinear oscillator and gain the desired limit cycle behavior. To put it simple, the limit cycle of a nonlinear oscillator is seen through a nonlinear window that changes the shape of this limit cycle. So, a nonlinear map is defined as follows:

$$f : B \mapsto S \tag{1}$$

where B is the base nonlinear oscillator space, S is the original signal space (i.e. generally the space of the joint angles of the robot), and f is the nonlinear map that maps B to S. If the f is designed properly, the effect of movement in B could be seen in S. So, to generate the desired limit cycle in S, the limit cycle of the base nonlinear oscillator is tracked in B and the effect is seen through f. But this is only an open loop tracking schema and no error feedback is used. To be able to feedback the error, f is needed to be invertible.

Designing an invertible nonlinear mapping is a challenging task. Almost none of the general function approximation tools are able to design general invertible nonlinear maps. To overcome this difficulty, f is redefined as two forward and backward maps respectively:

$$\begin{aligned} f_1 &: S \mapsto B \\ f_2 &: B \mapsto S \end{aligned} \tag{2}$$

Forward and backward maps are used to build a CPG model. The CPG model takes the following steps to create the desired limit cycle in S (Fig. 3):

a. start from an initial point $\underline{q}$ in S ;

$$\underline{q} \in S \tag{4}$$

b. map $\underline{q}$ to a point $\underline{p}$ in B using f_1;

$$\underline{p} = f_1(\underline{q}),\ \ \underline{p} \in B \tag{5}$$

c. move with a small time step Δt_B in B with respect to the base nonlinear oscillator differential equations (D_B);

$$\underline{p} = \underline{p} + \frac{d}{dt} D_B(\underline{p}).\Delta t_B \tag{6}$$

d. map $\underline{p}$ to a point $\underline{q}$ in S using f_2 and update $\underline{q}$;

$$\underline{q} = f_2(\underline{p}) \tag{7}$$

e. goto b.

The above process produces the desired limit cycle in S, but it completely depends on the forward and backward maps. Following subsection describes the design process of f_1 and f_2.

B. *The Design Process*

The proposed CPG model alters a base nonlinear oscillator in order to obtain a desired limit cycle behavior in S. The characteristics of the base nonlinear oscillator used can affect the overall limit cycle behavior in S. So a good choice for the base nonlinear oscillator is the group of oscillators that have only one stable limit cycle, such as Hopf and Van-der-pol oscillators.

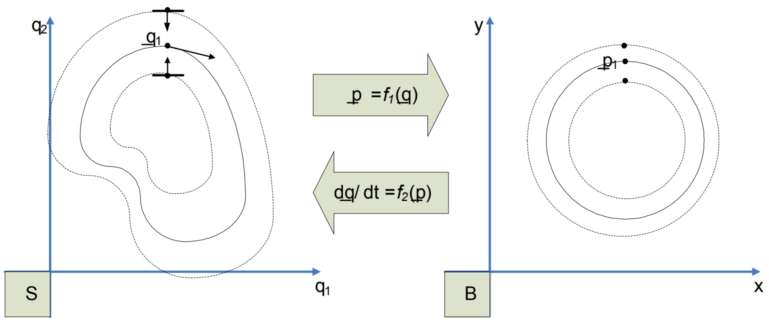

Fig. 4. Complete CPG model with Poincaré-Bendixson theorem deployed. In the forward mapping, the margins of the desired limit cycle in S are mapped to the margins of the limit cycle of the base nonlinear oscillator in B . In backward mapping, the margins in B are mapped to the derivative vectors that are orthogonal to the tangent of margins in S .

After defining the base nonlinear oscillator, forward and backward maps are to be designed. Forward map is a function that is intended to map the desired limit cycle in S to the limit cycle of the base nonlinear oscillator in B . The forward map could be implemented by a nonlinear function approximation tool with a supervised training procedure. The input data to the training procedure is the desired limit cycle sampled data in S , and the target data in training procedure is the sampled limit cycle of the base nonlinear oscillator. Backward map is designed to be the inverse of the forward map. So it is implemented like the forward map, but the input and the target data are swapped.

Backward mapping is modeled as a point-to-point mapping. It means that input and target data of training are positions in B and S . Our experiments show that this will lead to noisy tracking of the desired limit cycle. So instead of point-to-point modeling of the backward mapping, target data is replaced by derivative vectors that take the initial points in S to the updated points after a step (Fig. 3). So step *'d'* in the CPG signal generation steps in the previous subsection becomes:

$$\frac{d\underline{q}}{dt} = f_2(\underline{p}) \\ \underline{q} = \underline{q} + \frac{d\underline{q}}{dt}.\Delta t_S \tag{8}$$

where Δt_S is a small time step.

Implemented model has some drawbacks. Even if the base oscillator is selected to have a stable limit cycle, the numerical error in the training phase could corrupt the limit cycle behavior. For example, consider that in the training of the backward model, derivate vectors are estimated with a cumulative clockwise or counter clockwise error (in the plane). This will lead to converging or diverging spirals instead of a limit cycle. To solve this problem, we deployed the Poincaré-Bendixson theorem [13] in the learning process.

C. Deploying Poincaré-Bendixson Theorem

The aforementioned CPG model lacks the ability to ensure that the desired limit cycle exists. In other terms, no special mechanism to ensure that a limit cycle exists in a desired margin in S is used. Poincaré-Bendixson theorem gives the idea to solve this problem.

THEOREM (Poincaré-Bendixson). *Given a differential equation d/dt x = F(x) in the plane. Assume x(t) is an solution curve which stays in a bounded region. Then either x(t) converges for t → ∞ to an equilibrium point where F (x) = 0, or it converges to a single periodic cycle.*

To clarify, if a region in the phase plane is bounded so no derivate vectors takes a path started inside of the region to the outside of it, then an attractor in that bounded region certainly exists. This idea is deployed in the training procedure.

Two inside and outside margins are defined for the desired limit cycle in S , and respective margins are defined for the limit cycle of the base nonlinear oscillator in B . In the forward mapping, the inside and outside margins' data in S are added to the input training data, and respective margins in B are added to the target training data. In the backward mapping, margins in B are added to the input training data. Targets for inside and outside margins of B are derivative vectors in S that make the region between inside and outside margins in S bounded. For this reason, it is good to define the corresponding derivative vectors so the margins in S are transverse curves to the vector field. To put it simple, derivate vectors on the margins of S are defined orthogonal to the tangent of the margins, pointing inward (Fig. 4).

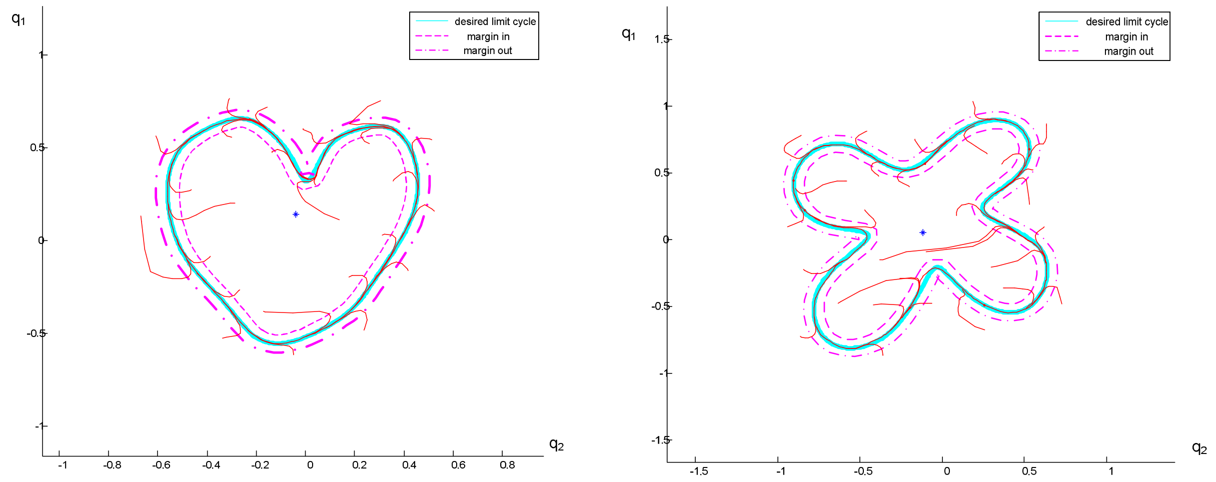

Fig. 5. Two sample CPG models. The presented shapes are in a 2D joint space. The tracks that are started inside the basins of attraction are converged to the limit cycles. Also the behavior of the models outside of the basins of attraction is satisfactory.

D. Implementation of the CPG Model

The CPG model is implemented using Hopf oscillator as the base oscillator. Hopf oscillator is defined as follows (with x and y as state variables):

$$\begin{aligned}\dot{x} &= -y + x(1-(x^2+y^2)) \\ \dot{y} &= x + y(1-(x^2+y^2))\end{aligned} \tag{9}$$

Forward and backward maps are designed using Multilayer Perceptrons (MLP) with two hidden layers. Arbitrary periodic signals are used to test the proposed CPG model (Fig. 5). As it could be seen, stable limit cycles with free shapes are successfully embedded into the CPG model. The basin of attraction for the generated limit cycles is explicitly defined and strongly contented. Any perturbations that keep the tracked signal inside this basin of attraction will fast and smoothly damp and vanish. Moreover, as it could be seen in Fig. 5, the behavior of the CPG model outside of the margins is also satisfactory. However, for some initial points far outside of the margins in S, the behavior of the CPG model may not converge to the desired limit cycle.

It may be of question that why the behavior inside the margins in S converges to a limit cycle and not to a point attractor. This is due to the structure of the function approximation tool used (MLP). MLP approximates a function smoothly, if the number of hidden neurons is selected appropriately. To have a point attractor in a region, superposition of the derivative vectors in that region should be equal to zero. So opposing derivative vectors have to be present in a small region. But the training data fed to the MLP is a directed data that determines the desired behavior. Since this behavior is not defined to shape a point attractor, as long as the MLP is not over-parameterized, the proper limit cycle behavior will occur.

E. Advantages

The proposed CPG model has a number of outstanding advantages:

1) Fast convergence: perturbations are fast and smoothly damped. Usually perturbations that keep the tracked signal inside the basin of attraction will damp in fewer than 10% of the period time of the learnt signal.
2) Guided convergence: for any initial point inside the basin of attraction the path of convergence to the limit cycle is in a straight manner. In other words, a very short and smooth path from the initial point to the limit cycle is usually followed.
3) Explicit margins: The basin of attraction is defined explicitly. This will ensure the designer to achieve an appropriate behavior inside a good margin.

The mentioned advantages are very useful in applications like biped locomotion where the stability issues are involved. In such applications, fast and guided convergence will lead to a stable gait that remains in a bounded manner.

V. Implementation on Robotic Marionette

A. Previous Works

Robotic marionettes are under-actuated string robots that are meant to perform actions via strings connected to a puppeteer platform. Systematic control of marionette robots is still a hard problem to solve and many research groups are trying to introduce a solution to this problem [14]. Some try to model the dynamics of marionette robots in order to make robot motion planning possible [14], [15]. Although these approaches provide mathematical formulation of robot dynamics, deriving these formulas is quite difficult. In addition, these resulting formulas have to be revised for different marionettes. Another approach to solve marionette robot control problem is to use motion capture data from a mentor's body. In [16] a method to transfer and adapt motion capture data from human mentor to marionette robot is introduced.

One of the main difficulties of marionette robot control is the swing of the body links. Because strings are used to move the body links, perturbations are inevitable. Control approaches that try to formulate the dynamics of motion suffer from this fact and the formulas become overly complex. This complexity is not necessary since the aim of marionette playing is to produce an understandable behavior and not necessarily a precise action.

It is not necessary to identify the dynamical system of a marionette in order to generate desired behavior. It could be beneficial to learn desired actions in an interactive process and embed the control strategy into the marionette robot. Use of an appropriate learning model will eliminate the need for deriving complex formulae in order to describe the dynamic system of the marionette.

Most of a marionette's actions are behaviors that result from periodic body movements, e.g. walking, hopping, handshaking etc. So, a model that is capable of learning periodic signals will be advantageous. For this reason, it is suitable to use a CPG as the basic component for marionette control. So, we used our imitation model that takes advantage of CPG model to learn proper motions in a robotic marionette.

B. *Implementation*

The learning model was implemented on the robotic marionette constructed in Robolab at Univesity of Tehran. The marionette is named Hootan (Fig. 6). Hootan is a 10-DoF marionette robot that is controlled via 8 servo motors; two for each arm and two for each leg. For now, back of Hootan is fixed to the puppeteer platform and will be replaced with a string in near future.

Left arm of Hootan is used to imitate the action of a human puppeteer. Since our goal is not a complex object tracking problem, we put a marker on Hootan's palm for simplicity. For detection of this marker a dataset of 50 sample values of the marker in different lighting conditions is captured and the median of the sample's values is used as marker prototype. In runtime, for each image frame, image pixels' color content are compared to the marker prototype and the median of positions of top 100 candidate pixels is indicated as palm position.

Hootan learns to imitate a puppeteer's action by looking at it. A human puppeteer moves the strings controlling Hootan's left arm freely. Hootan's palm movement's outcome is seen through a VGA camera (640×480) at 20FPS. Since the human puppeteer's hand movements are not seen, the only information transferred to the learning model is the trajectory of Hootan's palm movements in the viewpoint of the camera. Even if we had the means to capture human puppeteer's motions, there would be no straightforward use of that data regarding fundamental differences between puppeteer's body and Hootan's.

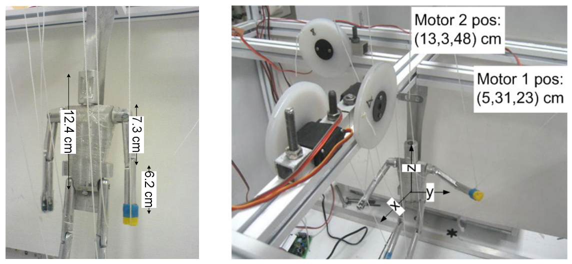

Fig. 6. Robotic marionette Hootan. Hootan is a 10-DoF marionette robot that is controlled via 8 servo motors; two for each arm and two for each leg. The coordinates for the motors that manipulate Hootan's left arm are presented.

After observing what is done by the human puppeteer, hand-eye coordination process begins. Initially Hootan's servos are set to random positions. Then Hootan's motors are continually set to new positions with respect to aforementioned hand-eye coordination algorithm. The position of Hootan's palm is extracted in each step and acquired data, set of motors and palm positions, is logged. This is done until the entire desired trajectory in visual space is swept. After that, interesting parts of the logged data, sets that were approximately close to temporary goals, is used to train a MLP with two hidden layers. A compact picture of hand-eye coordination process is shown in Fig. 7. It is important to mention that since the learning path is continues and the trial steps are small and close to each other, swing behavior is negligible and do not affect the learning process.

After learning the relation between palm position and motor positions, the appropriate motor trajectory is calculated from desired visual trajectory. Resulted motor trajectory is embedded into the CPG model. Prepared CPG model could be used to reproduce smooth trajectory that

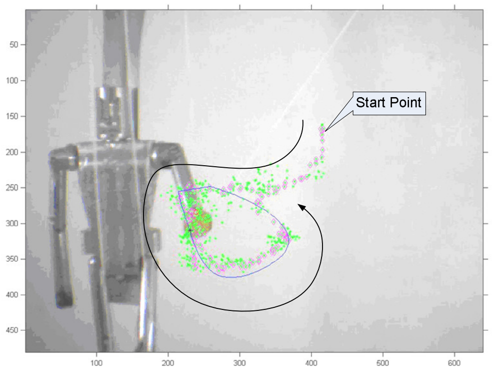

Fig. 7. Hand-eye coordination process. Learning trail starts from a random point outside of the desired trajectory and after a number of tries it reaches the desired trajectory and sweeps it.

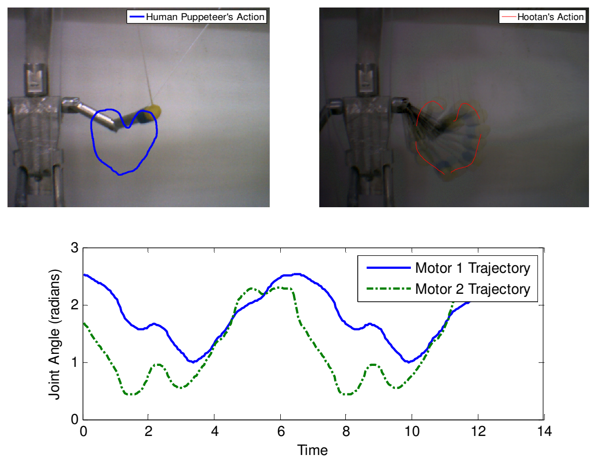

Fig. 8. *up left)* Human puppeteer's action. *up right)* Hootan's action. Sample shots are taken any 0.4 seconds. Performances are not exactly the same, but they are behaviorally recognized as same actions. *down)* the motor action that is embedded into the CPG model to generate the heart-shaped outcome.

leads to the desired visual outcome. It is possible to initiate the movement from anywhere within the defined basin of attraction and the model will converge to the desired limit cycle behavior. A sample of the outcome of actions performed by Hootan and human puppeteer is compared in Fig. 8. Although the outcomes are not exactly the same, they could be behaviorally recognized as similar actions, and that is what they are meant to be.

VI. Conclusion & Future Works

In this paper we presented a novel model of CPG that is able to learn arbitrary periodic signals. The proposed CPG model has the ability to define an explicit basin of attraction with fast and guided convergence behavior. This is a beneficial point in complex periodic tasks like humanoid walking or robotic marionette motion control. The CPG model is used in an imitation model that takes benefit from infant babbling concepts. A hand-eye coordination process inspired from infant babbling is presented. This hand-eye coordination process extracts the proper motor action from the observation of mentor. The extracted motor action is then embedded into the CPG model for smooth reproduction.

The imitation model is implemented on a robotic marionette. A human puppeteer freely manipulates the marionette's limbs through strings. The outcome of this action is seen and fed to the imitation model. After that, a hand-eye coordination process extracts the proper action for motors that pull the marionette strings. Extracted motor action is then embedded into the CPG model. The reproduced actions are behaviorally recognizable and this is the goal of a marionette playing platform.

Our future research will be directed toward implementing this imitation model on the locomotion problem. We will try to make a walking robot learn to walk by looking at a mentor. However, since the proposed CPG model is only tested for 2D motor trajectories, the model has to be extended for more dimensions. The proposed CPG model could be used as the building block of such multidimensional model.

Acknowledgment

We would like to acknowledge Ali Sadeghi for his friendly assistance and Farzad Sangi and Mohammad Mahdi Ajallooeian for their helpful cooperation.

References

[1] S. Grillner, "Neurobiological bases of rhythmic motor acts in vertebrates," *Science (New York, N.Y.)*, vol. 228, Apr. 1985, pp. 143-9.

[2] G. Orlovsky, T.G. Deliagina, and S. Grillner, *Neuronal Control of Locomotion: From Mollusc to Man*, OUP Oxford, 1999.

[3] L. Righetti and A.J. Ijspeert, "Programmable Central Pattern Generators: an application to biped locomotion control," *In Proceedings of the 2006 IEEE international conference on robotics and automation*, vol. 2006, 2006, pp. 1585--1590.

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

[5] N. Kopell and G.B. Ermentrout, "Coupled oscillators and the design of central pattern generators," *Math. Biosci*, vol. 90, 1988, pp. 87-109.

[6] L. Righetti and A.J. Ijspeert, "Design methodologies for central pattern generators: application to crawling humanoids," *Proceedings RSS 06*, 2006.

[7] J. Call and M. Carpenter, "Three sources of information in social learning," *Imitation in animals and artifacts*, MIT Press, 2002, pp. 211-228.

[8] N.S. Baron, *Growing Up With Language: How Children Learn to Talk*, Perseus Books, 1992.

[9] J.M. Iverson, Am, A.J. Hall, L. Nickel, and R.H. Wozniak, "The relationship between reduplicated babble onset and laterality biases in infant rhythmic arm movements," *Brain and Language*, vol. 101, Jun. 2007, pp. 198-207.

[10] E. Thelen, "Rhythmical stereotypies in normal human infants," *Animal Behaviour*, vol. 27, Aug. 1979, pp. 699-715.

[11] A.N. Bhat and J.C. Galloway, "Toy-oriented changes during early arm movements: hand kinematics," *Infant Behavior & Development*, vol. 29, Jul. 2006, pp. 358-72.

[12] A. Dearden and Y. Demiris, "Learning forward models for robots," *in Proceedings of IJCAI*, vol. 2005, 2005, pp. 1440--1445.

[13] I. Bendixson, "Sur les courbes définies par des équations différentielles," *Acta Mathematica*, vol. 24, 1901, pp. 1-88.

[14] K. Nguyen, K. Lim, W. Dong, Y. Goh, I. Chen, S. Yeo, B. Henry, K. Li, and C. Su, "Toward a dynamic model of robotic marionettes," *2008 IEEE International Conference on Robotics, Automation and Mechatronics, RAM 2008*, 2008, pp. 488-493.

[15] E. Johnson and T. Murphey, "Dynamic modeling and motion planning for marionettes: Rigid bodies articulated by massless strings," *Proceedings - IEEE International Conference on Robotics and Automation*, 2007, pp. 330-335.

[16] K. Yamane, J. Hodgins, and H. Brown, "Controlling a marionette with human motion capture data," *Proceedings - IEEE International Conference on Robotics and Automation*, 2003, pp. 3834-3841.