BraveCorpses 3D Soccer Team Description Paper

Hamed Shahbazi, Mohammad Masood Masaeli , Reza Ghanami , Reza Bashiri , Mehrdad Pedram , Meghdad Paknezhad

University of Sheykhbahaee , Isfahan , Iran

email : robocup@eng.shbu.ac.ir

Abstract. This paper describes the BraveCorpses team for the Robocup2011 competitions. In this paper the result of our efforts in intelligence and efficiency of robots in the simulation world will be explained. It contains running and optimizing of the motions such as walking and standing up by optimizing algorithms, and implementing a learning method named as CPG biped robots. Soon by improving the implementation of these techniques and adding better online decision making abilities you'll see the improvements of our robots' abilities.

1 Introduction

In the last few years, competitions called Robocop have been held which are founded to solve the problems and spread the knowledge of artificial intelligence and in order to absorb the cooperation of several groups, it has used the worlds' favorite sport, "soccer" and holds competitions in different levels.

We are also working in 3D soccer simulation branch to improve this knowledge and cooperate in making better ways of using machines in man's life. This team, named BraveCorpses is founded by some students and instructors from Sheykhbahaee university of Isfahan-Iran.

The problem of robot locomotion is where neuroscience and robotics converge. This common part is pattern generators in the spinal cord of vertebrate animals called "Central Pattern Generators" (CPGs). Central pattern generators are neural circuits located in the end parts of the brain and first parts of the spinal cord of a large number of animals and is responsible for generating rhythmic and periodic patterns of locomotion in different parts of the their bodies. Although these pattern generators use very simple sensory inputs imported from the sensory systems, they can produce high dimensional and complex patterns for walking, swimming, jumping, turning and other types of locomotion. The origin of many movements in animals is central pattern generators which were discovered by Brone in the early decades of the 20th century. He discovered that the movement in many animals is an outcome of central neuronal activities in some parts of their neural system, and simple sensory inputs change these activations and make them capable of responding to the extraneous perturbations [1]. The idea that CPGs are neural networks generating complex locomotion patterns with only simple inputs are a provocative one which is intended to model in this paper.

The other parts of this document contains robot model which is about the architecture and the way of matching the internal layers of the agent, then we have the discussion part that in brief represents some of discussions we made on different tasks and finally we'll see the conclusion and references.

2 Robot Model

We developed an architecture for implementing the robot to interact with Simpspark. This architecture contains three layers. These levels are Planning layer, High level layer and Low level layer and their relations showed in figure 1.

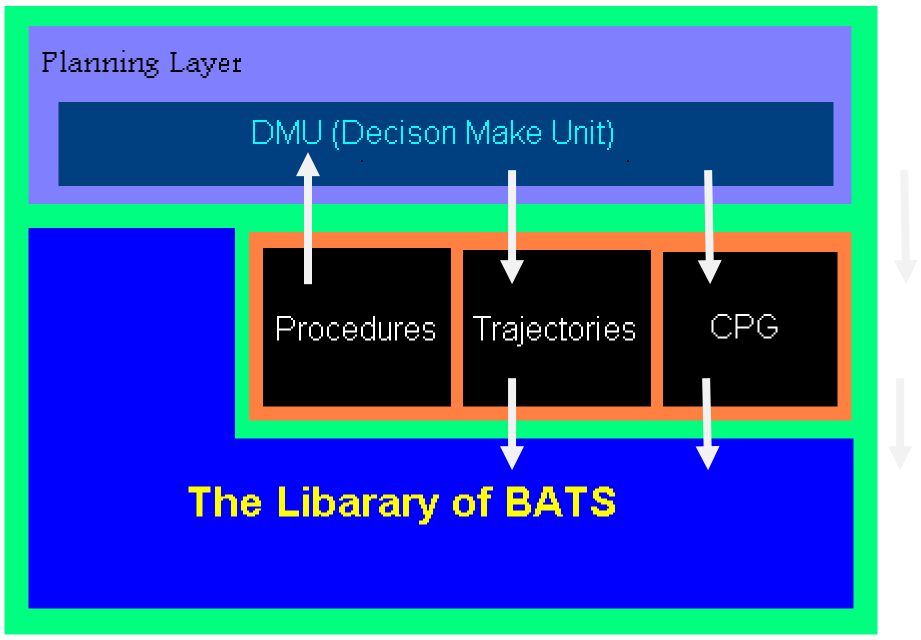

Fig. 1. The Agents' Architecture

2.1 Planning layer

2.1.1 DMU

In this unit the agent makes decisions according to its place, situation and time. The agent gets the required information for these decisions from the Localizer, World Model, and Agent Model parts included in the library of bats and sends instructions to the high level skills layer, then this layer executes these instructions in the simulation environment by the means of the control layer which is a part of the library of bats as well.

2.2. High Level Layer

2.2.1. Trajectories.

In this technique, trajectories are already stored in a text file and the agent reads the values which in fact are called joints angels and sends related motion data to the control layer to execute. More explanations is in the following:

In the Simspark simulation environment the NAO robot has servo motors to which we can send angular velocity values to move. In this environment a time period is 20ms so each angular velocity value is the variety of a joint per 20 ms.

Trajectories are series of joint angles (in radians) stored in files (named motion files) and Every line of a motion file contains the situations of some joints in a time cycle in radians. Figure 2 is the format we are talking about:

```
#WEBOTS_MOTION,V1.0,LHipYawPitch,LHipRoll,LHipPitch,LKneePitch,LAnklePitch,LAnkleRoll
00:00:280,Pose8,0,-0.001,-0.525,1.05,-0.525,0.001,0,-0.001,-0.525,1.05,-0.525,0.001
00:00:320,Pose9,0,-0.003,-0.524,1.05,-0.525,0.003,0,-0.003,-0.524,1.05,-0.525,0.003
```

Figure 2. Webots Motion Format

Therefore we can convert these values to the angular velocity by the below relation.

Angular Velocity of a Joint = Stored Value in the Trajectory file - Current Angle

The most important problem is creating these files. An idea is to start with random values, but because of too many parameters, it's clear that the idea is to be failed! The other idea is to use the tools for making motions. One of the tools is Webots program that provides the ability to make any movement by the hand and stores it in the format that we talked about before. Webots is also a robot simulator software which supports the NAO robot which its simulation is close to the Simspark software.

2.2.2. Motion learning using Programmable CPG

In this section we will explain a motion learning method which we have used to train the curvilinear bipedal walk on the Simulated Nao Soccer Robot. We will explain what the curvilinear bipedal walk is and how we will use Generic CPGs to train this walk to our Nao robot. We will discuss the nonlinear dynamical oscillators and the fundamental prosperities of these oscillators. Then the coupling scheme and its roll in walking will be discussed.

The special type of locomotion we have focused on in this paper is curvilinear walking. The curvilinear walk is a smooth traveling from a point in the soccer field to a new point elongated a curve shape path. A special case of curve shape path is a part of circle. In Figure2 this kind of walk is shown. During a curvilinear walk the robot should synchronically rotate and travel forward and it must keep its distance from what it should be.

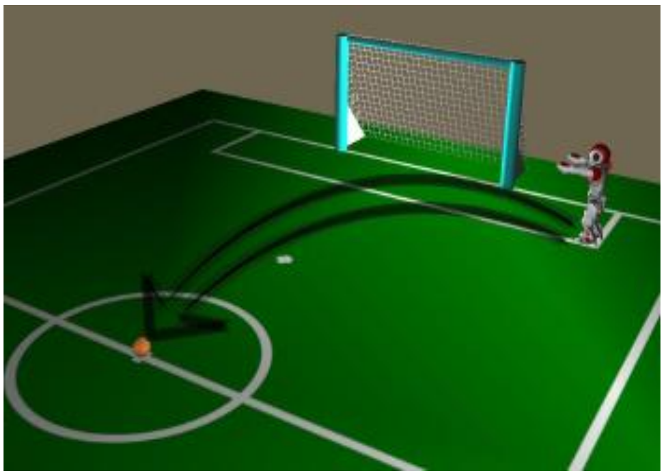

Fig. 2. Curvilinear walk in a Nao R4obot

The fundamental building block of the generic CPG is the adaptive frequency Hopf oscillator, which is proposed in the [6,7]. These oscillators have the property of learning the frequency of a periodic input signal without any external optimization. Usually, the frequency of an oscillator can be controlled by some parameter. In this model all the parameters is changed to a state variable which can be trained using some general evolution rule. It can be proven that when perturbed by a periodic input signal, these state variables will converge to one of the frequency elements of the input signal. The adaptation is an intrinsic characteristic of these oscillators. In Addition, there is no need for supervisor or external processing and this is the state of the art of this systems.

After convergence of the system, if the input signal disappears, the learned frequency would be remained encoded in the system. The equations of this oscillator are as follow:

$$\dot{x} = \gamma\,(\mu - r^2)x - \omega y + \varepsilon F(t) \qquad (1)$$

$$\dot{y} = \gamma\,(\mu - r^2)y + \omega x \qquad (2)$$

$$r = \sqrt{x^2 + y^2} \qquad (3)$$

$$\dot{\omega} = -\varepsilon\, F(t).\frac{y}{r} \qquad (4)$$

μ controls the amplitude of the oscillations, γ controls the speed of recovery after perturbation, ω controls the frequency of the oscillations, F(t) is a periodic input to which the oscillator will adapt its frequency and $\varepsilon > 0$ is a coupling constant. Its frequency will adapt to one of the frequency component of the input F(t). The frequency component adapted will depend on the initial conditions for ω.

Constructing a Programmable Central Pattern Generator requires connecting and coupling of these Hops Oscillators. In figure 3 this connection has been shown. The Pteach(t) signal is the desired trajectory which we need to train into the CPG. Qlearned(t) is what system has learned till time t. the difference of these signals is used as a perturbation signal for the Hopf oscillators. The output amplitude of each oscillator is multiplied to an alpha coefficient and then all of these outputs will be added together. The total equations are presented below:

$$\dot{x}_i = \gamma\,(\mu - r^2)x_i - \omega_i y_i + \varepsilon F(t) + \tau \sin(\theta_i - \phi_i) \quad (5)$$

$$\dot{y}_i = \gamma\,(\mu - r_i{}^2)y_i + \omega_i x_i \quad (6)$$

$$\dot{\omega}_i = -\varepsilon\, F(t).\frac{y_i}{r_i} \quad (7)$$

$$\dot{\alpha}_i = \eta x_i\, F(t) \quad (8)$$

$$\dot{\phi}_i = \sin\left(\frac{w_i}{w_0}\theta_0 - \theta_i - \phi_i\right) \quad (9)$$

$$\theta_i = \mathrm{sgn}(x_i)\cos^{-1}\left(-\frac{y_i}{r_i}\right) \quad (10)$$

$$F(t) = P_{teach}(t) - Q_{learned}(t) \quad (11)$$

$$Q_{learned}(t) = \sum_{i=0}^{N} \alpha_i\, x_i \quad (12)$$

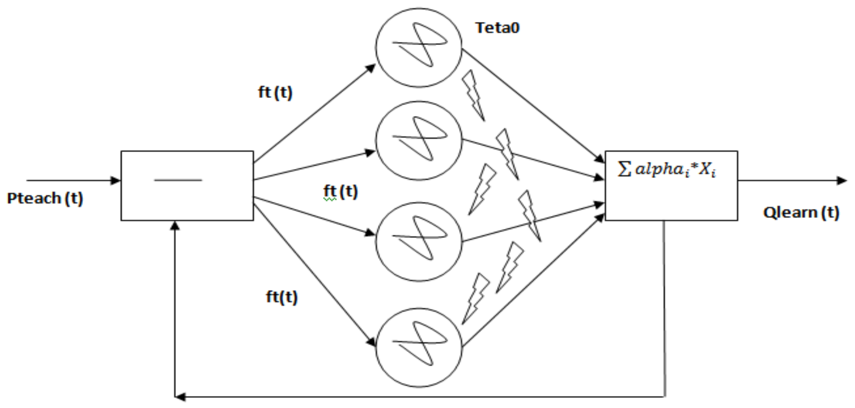

Fig. 3. The Connection of the Hops Oscillators

We use one generic CPG for each joint of the Nao in each Degree Of Freedom(DOF) of each foot. Each CPG is made of exactly 4 oscillators, as shown in Fig. 3. To coordinate these several joints, for each leg and the opposite arm, a chain coupling from the hip to the ankle of the first oscillator of each CPG has been used. This coupling is partially similar to what is used in Ref. [10] for the Hoap2 robot illustrated in Fig. 4. In this Fig, joint number 1 and 7 are (Hip yaw-pitch joints of the left and right feet) are synchronized together. Joints 2, 3, 4, 5, and 6 in one set and 8, 9, 10, 11, and 12 in the other set will be synchronized sequentially. Joint 13 (the right shoulder pitch joint) is synchronized with joints 2 and 3, and joint 14 is synchronized with joints 8 and 9 (the left-shoulder pitch joint). Also add some extra terms are added to keep correct phase differences between the DOFs. To train different trajectories to these Generic CPGs we need to enter the training trajectories to the system and after a long period of training, the parameters of CPGs are adapted to the desired input. Then the last state of the system will be transferred to the robot controller and in each step of walk, calculation of an integral should be done to find the new state of the system.

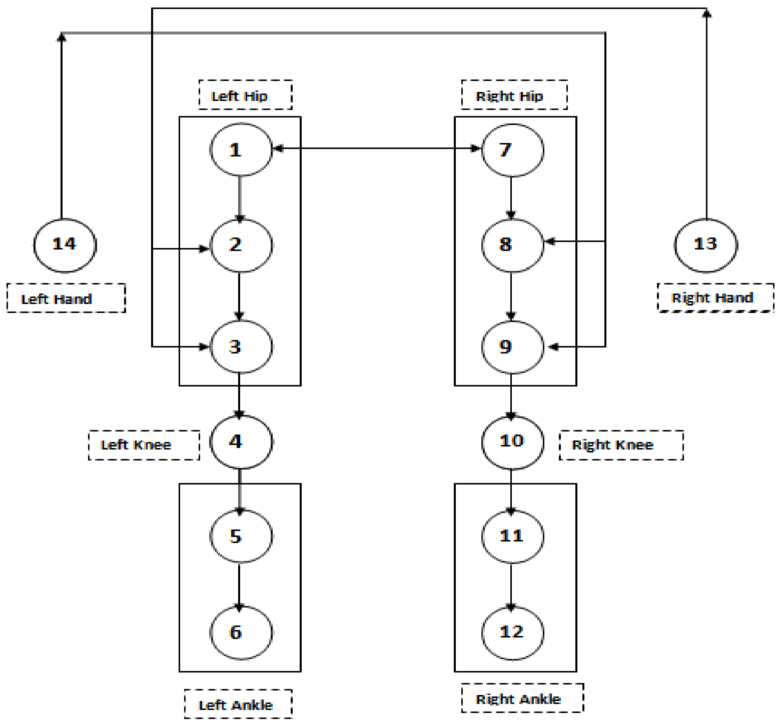

Fig4. The coupling scheme of the different generic CPGs

The Model of generic CPGs presented, generates online trajectories for each joint of Nao. We use these trajectories as the desired angles (set points) for the PID controllers controlling each joint. The major benefit of this model is that we can encode arbitrary periodic signals as limit cycles in a network of coupled oscillators. When we get all the properties of such systems, we can modulate the frequency and the amplitude in a smooth way, we have stability to perturbations and we can integrate feedback pathways to adaptively regenerate trajectories and optimize different skills in soccer player robot.

We will add a new state variable to these equations to make a curvilinear control in the generic CPG model. In fact curvilinear walk is a result of smooth changes in the Yaw/Pitch servo of the Hip joint in the Nao model. So we add Eqs 13 and 14 to the system to be used for Hip Yaw/Pitch joints:

$$\dot{x}_i = \zeta(\gamma\,(\mu - r^2)x_i + \omega y_i) + \mathrm{Ro}(t).\ \frac{x_i}{r_i} \qquad (13)$$

$$\dot{y}_i = \zeta(\gamma\,(\mu - r^2)y_i - \omega x_i) + \mathrm{Ro}(t).\ \frac{y_i}{r_i} \qquad (14)$$

Here ζ is a small coordinating coefficient which is used to make Yaw/Pitch joint behavior similar to the other joints. Ro(t) is the expected rotation of the robot in the time *t* which can be computed with this equation:

$$Ro(t) = \frac{\mathrm{AccX(t)}}{\mathrm{R}} \qquad (15)$$

In this equation R is the radius of a circular curve and AccX(t) is the speed increment of the robot in the direction of the robot's body. In fact Ro at the time t is the projection of rotation in CPGs equations and will be computed from radius of the circular curve and the amount of acceleration. R is inversely proportional with Ro(t). The larger the radius of the curve, the smaller the projection of rotation in each time, and thus robot would more slowly deviate from his straight path.

Some experiments carried out by researchers have found that the arms' movements actually provide important indirect benefits for human walking. Their study was based on the movements of some volunteers. These researches have shown that swinging the arms in opposition to the legs significantly increases the efficiency of walking. They also discovered that normal arm swinging made walking much easier and smoother. Statistics demonstrate that holding the arms at one's sides increases the effort of walking by 12 percent, which leads to quite a lot of energy consumption. These researches also showed that movements of the arm play a very important role in walking and running. To stabilize and increase the performance of our curvilinear walk we should include the role of arms in robot walking. The most important arm joint which mostly moves the arm is the shoulder pitch joint. This joint is responsible for moving the arm in the direction of the body and can be used to keep the center of the momentum in a safe and stable area. So only these pitch joints of the shoulders have been considered and the roll joints pertaining to the four elbows and the two shoulders have been locked. There are two different arm-control rules for pure rotation and pure rectilinear walks and a linear combination of these is used for a curvilinear walk. In the rectilinear case the angular momentum generated by the arms should repeal unwanted momenta around the Y axis which may be generated by the foot swinging. On the other hand, in a pure rotation case, the angular momentum generated by the arms must intensify the momentum around Z which is used for rotation.

For the arms we do not use direct programming of generic CPGs in the shoulder pitch joints. In other words we compute a shoulder pitch joint indirectly with a linear combination of the other CPG values. Since we intend to compensate the effects of swing phase the left arm should be synchronized with the right foot and the right arm should be synchronized with the left foot. It means that whenever the left foot is in its swing phase and it goes forward, the right arm should simultaneously go forward and the left arm should go backward. We will use equations (16-17) to control arm movements:

$$Pitch_{RSholder}\,(t) = R.Roll_{LHip}\,(t) + \Big((\beta - R).Pitch_{LHip}\,(t)\Big) + b \quad (16)$$

$$Pitch_{LSholder}\,(t) = (\beta - R).Roll_{RHip}\,(t) + \Big(R.Pitch_{RHip}\,(t)\Big) + b \quad (17)$$

In the above equations the left shoulder pitch-joint position at the time t is computed from a linear combination of the right hip-roll joint position and the right hip pitch-joint position. R is the radius of curvilinear rotation discussed earlier in curvilinear equations. β is the complement of R which determines how much momentum the opposite arm should generate. b is a bias value which shifts the range of the movements of the arms.

In Fig. 5 the Generic global CPG system in Simulink is illustrated which can be programmed by Nao training trajectories. The present model has been trained and its last state values have been extracted in Simulink. Then generic CPGs have been transferred to the online controller code of the Nao which was written in MATLAB.

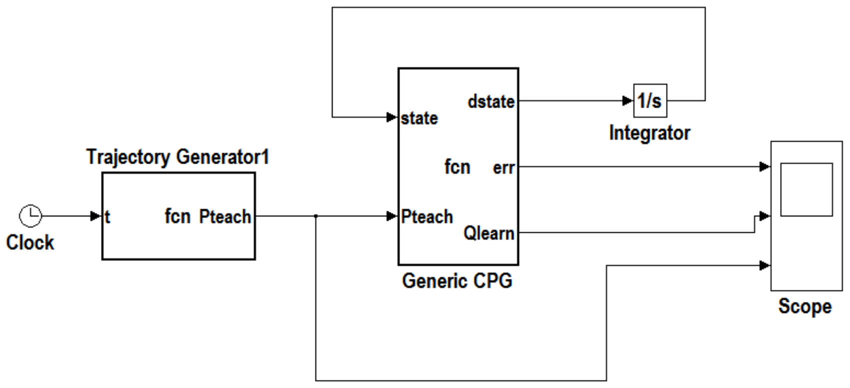

Fig5. Generic CPG which is programmed by trajectories

When the robot decides to walk on the perimeter of the circular curve with radius R, the online trained CPGs start to generate patterns based on the specific R value. Fig. 6 shows an example of such a circular curve on which the robot has traversed. In Fig. 7, 12 snapshots from the robot’s curvilinear walking are shown. In the first part of the Fig, the robot is in the middle of the Soccer field. Then it slowly walks and rotates up to the left of the soccer field. The notion that the robot never stops rotating is a key point in this walk. One can see that it is a curvilinear walk from the starting point in the middle to the corner point. Fig. 7 also illustrates how the arms of the robot move during this curvilinear walk.

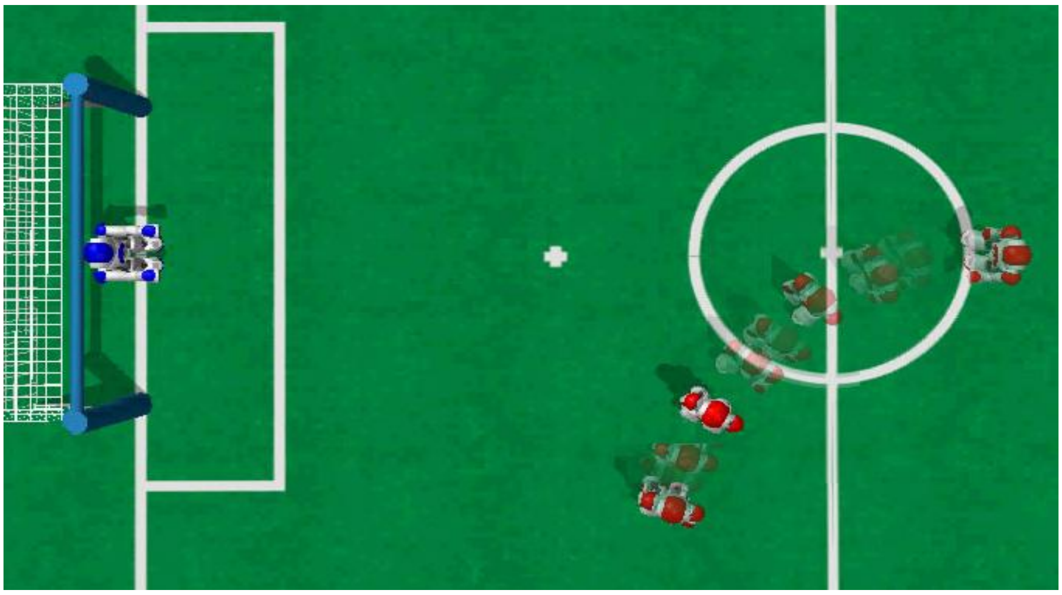

Fig6. Snapshots of our curvilinear walking

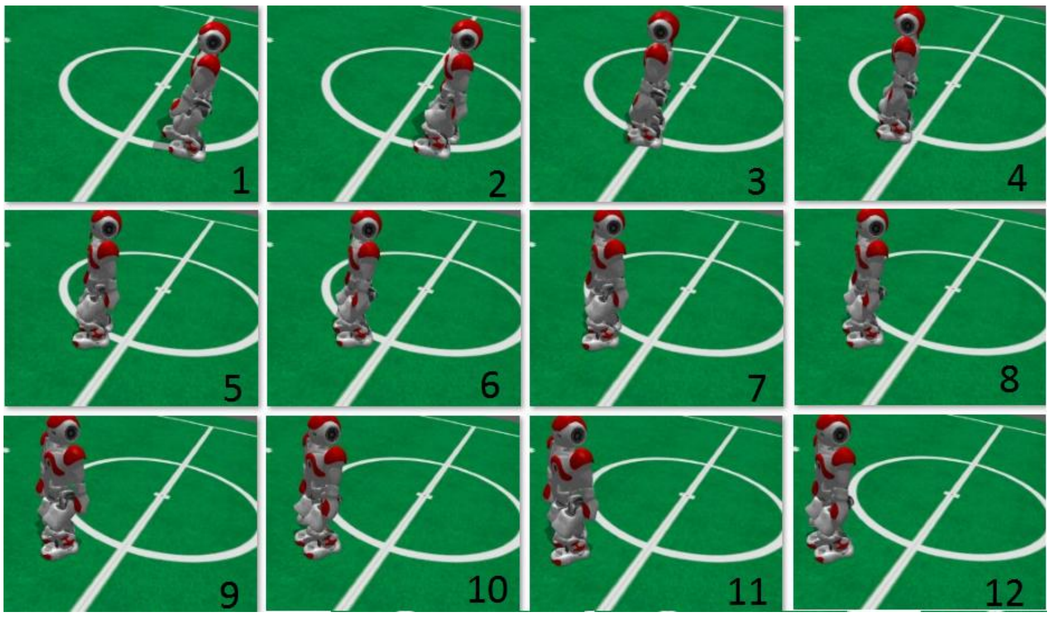

Fig7. Snapshots of our curvilinear walking

One of the most beneficial aspects of CPG based approaches in bipedal walking is modulation of generating trajectories. Modulation of these trajectories helps the robot to change its speed and style of walking. It can increase or decrease its speed by modulating basic frequencies of the trained trajectories. Extracting the Omega values of walking trajectories, we can obtain basic frequencies of all the joints. A basic frequency is the greatest comment divisor of almost all the frequencies (omega values) of the trained trajectories. It is possible to find a number which can generate the entire omega values. Increasing or decreasing this basic frequency would cause to modulation of trajectories and lead to increment or decrement of the walking speed.

2.2.3. Procedures

Sometimes the movements are so simple and only happen in specific situations. For instance when the robot is born in the Simspark, in order to initialize, it should put its hands in a specific form. We can do this via procedures which send some specified data to the control layer.

2.3. Low Level Layer

In this layer we just use the bats library which models the world and omits the noises by the Kalman algorithm and its cerebelum module is really reliable to send things to the Simspark server.

3 Discussion

3.1.Optimizing Motions

As mentioned before, doing some movements are done via trajectories which is produced by Webots. But there are some differences between the modeling of NAO robot in Simspark and Webots and on the other hand as trajectories are made by hand, they have not a good ability to balance the robot and they include extra unusable movements. So we should make these trajectories optimized.

In this regard we started with Webots walking motion and modeled this skill, then optimized the model via Matlab tools.

3.1.1. Curve Fitting

According to the groups' discussions and research, the robots' joints movements in walking are periodic, so for a accurate modeling we used sum of some sinus functions for every joint as showed in the follow:

$$\mathbf{a_1\sin(b_1x+c_1)+ a_2\sin(b_2x+c_2)+ a_3\sin(b_3x+c_3)+ a_4\sin(b_4x+c_4)+ a_5\sin(b_5x+c_5)+ a_6\sin(b_6x+c_6)+ a_7\sin(b_7x+c_7)+ a_8\sin(b_8x+c_8)}$$

In these functions the "x" variable stands for the time," a" is the amplitude, "b" is the frequency and "c" is the initial phase. Figure 8 is the result of modeling for the left knee joint in walking.

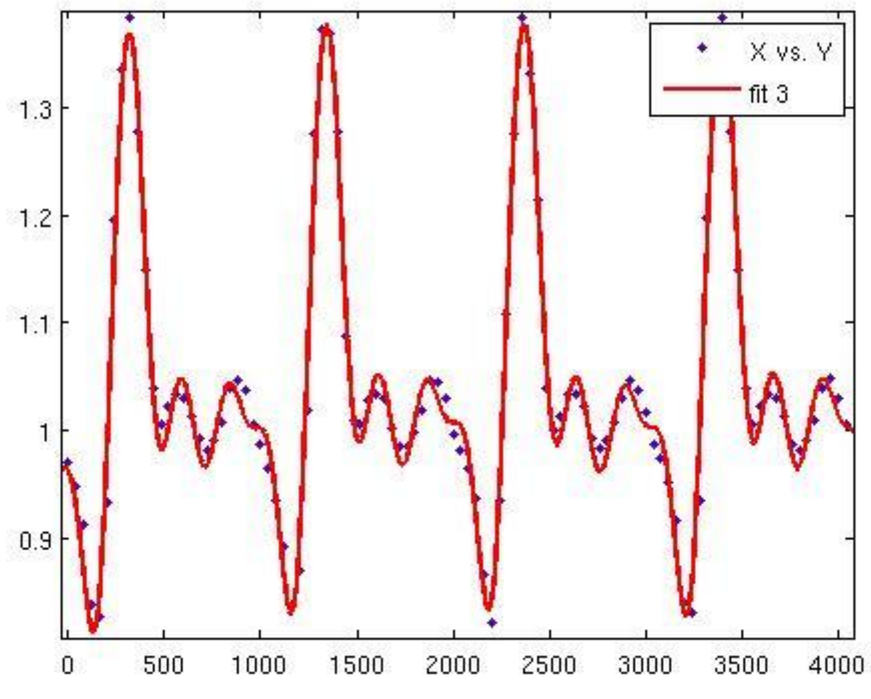

Fig 8: Curve fitting on modeling the changes of left knee

This model is implemented via the curve fitting tool-part of the Matlab software- from the trajectories which is made by Webots.

The curve fitting tool provides a function for a joint, based on the trajectory files.

Now we use the result of curve fitting to optimize the trajectories.

3.1.2. Genetic Algorithm

After modeling a trajectory by curve fitting, it's time to optimize. So we used the Genetic Algorithm (GA) to do it. This algorithm has an input which is called the initial population and regarding to the fitness function, after a while it generates better populations and individuals which are useable for better movements.

To use this algorithm, we set the initial population from the result of curve fitting and defined a fitness function to communicate the parameters via TCP/IP to our agent so every new individual could be tested on the Simspark and the agent sends back the results to Matlab.

Figure 9 shows the relations between Simspark, Our Agent and Matlab:

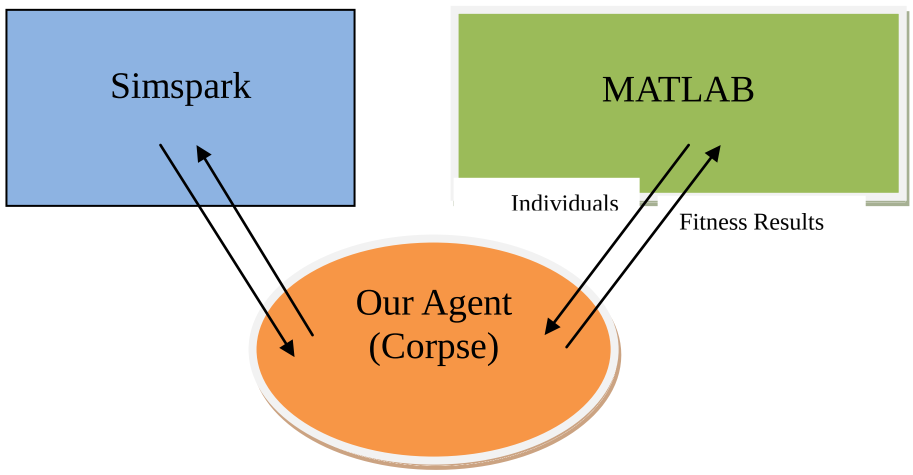

Fig 9. the relations between Simspark, Our Agent and Matlab

4 Conclusion

We introduced a new method for generating curvilinear bipedal walking pattern using a Programmable Central Pattern Generator which is trained by NAO basic walking trajectories. This model can be used for other movements in Soccer matches between NAO robots in Robocup and because of this, it would be a very useful and standard method for these competitions. There is a large number of benefits in CPGs that can be applied in this method. This is what we are going to do in our future researches. We plan to implement our method in real NAO and test its efficiency.

5 References

[1]. A.J. Ijspeert, Central pattern generators for locomotion control in animals and robots: a review, Journal of Neural Networks,Elsivier, Vol 21/4, pp. 642-653, 2008.

[2]. J.J. Kuffner, S. Kagami, Dynamically-stable Motion Planning for Humanoid Robots, Journal of Autonomous Robots vol. 12, No. 1, pp. 105-118., 2002.

[3]. A.J. Beng Tay, Walking Nao Omnidirectional Bipedal Locomotion , Thesis in University of New South Wales, 2009.

[4]. A. Cherubini, F. Giannone, L. Iocchi, M. Lombardo, G. Oriolo, Policy gradient learning for a humanoid soccer robot, Journal of Robotics and Autonomous Systems 57, pp. 808-818, 2009.

[5].M. Vukobratovic , D. Juricic, Contribution to the Synthesis of Biped Gait, IEEE Trans. On Bio-Medical Engineering, vol. BME-16, no. 1, pp.1-6,1969.

[6].L. Righetti , A.J. Ijspeert, Programmable Central Pattern Generators:an application to biped locomotion control , Proceedings of the 2006 IEEE International Conference on Robotics and Automation Orlando, Florida - May 2006.

[7].L. Righetti, J. Buchli, A.J. Ijspeert, Dynamic hebbian learning in adaptive frequency scillators, Journal of Physica D, pp. 105-116, 2005.

[8]. C. Lathion, Biped locomotion on the Hoap2 robot, Master's thesis, EPFL, 2006.

[9]. Joseph Giarratono. Gary Riley. Expert Systems Principles and Programming. Third Edition.