Design and implementation of a bio-inspired locomotion controller for a differential wheeled robot

Urziceanu Ionut*, Pablo Varona, Francisco de Borja Rodriguez**, Juan Gonzalez-Gomez*** **Fernando Hererro Carron**, Mircea Nitulescu******

** Robotics and Mechatronics Department, University of Craiova, Craiova, Romania (e-mail: urziceanuionut@yahoo.com)*
*** Autonoma University, Madrid, Spain(e-mail: pvarona@uam.es)*
**** Universidad Carlos III, Madrid, Spain (e-mail: juan@iearobotics.com)*
***** Robotics and Mechatronics Department,University of Craiova, Craiova, Romania (e-mail: mirceanitulescu@robotics.ucv.ro)*

Abstract: Bio-inspired neural circuits that fall in the category of central pattern generators have been used mainly in research concerning autonomous locomotion for legged and serpentine robots. This paper presents a novel locomotion controller based on bursting neuron models and time evolving synapses that is used to control a differential wheeled robot. We show how the proposed central pattern generator model is able to generate a flexible yet robust rhythm which is used to sequentially drive the wheels in a situation where the joints can only partially rotate.

Keywords: Central pattern generator, bio-inspired, wheeled robot, autonomous locomotion controller.

1. INTRODUCTION

Central pattern generators (CPG) are neural circuits known in neuroscience for being able to generate rhythmic activity without any external input. This type of neural network has been extensively studied in animals and experiments have shown that CPGs are responsible for activities such as locomotion, respiration, mastication, etc. For locomotion, a CPG can generate a specific sequence of motor commands in order to achieve a certain gait. Even if CPGs can generate motion patterns by themselves, sensory feedback can modulate the activity of the network that allows it to adapt to external conditions. This is essential in the real world where animals have to adapt their locomotion gaits to a variety of environments and must do this by using minimal resources.

Evolution led to central pattern generators which, in combination with the central nervous system (CNS), form a distributed architecture: high level commands are sent from the CNS to the CPG to modulate its behaviour and obtain a certain type of locomotion. Moreover, direct sensory inputs are known to exist in CPGs that allow rapid responses to the changes in the environment; this approach reduces the number of control pathways from the CNS to the muscles and minimizes the response time to a stimulus (Rabinovich, 2006).

Currently, in the field of robotics, artificial CPGs are taken into consideration as an alternative to classical locomotion control. Several attributes play in their favour:

- Robustness: they can robustly encode locomotion information;
- Adaptability and flexibility: they can adapt to external stimulus in order to change the locomotion pattern;
- Resistance to noise: due to the limit-cycle behaviour of the individual neurons, the network can autonomously recover from perturbations and regain normal operation;
- Ease of control: by changing a relatively small number of parameters, the CPG can exhibit many behaviours.

2. DIFFERENTIAL WHEELED ROBOT

2.1 Robot overview

A differential wheeled robot is a mobile robot whose locomotion is based on two wheels being controlled independently. The relationship between the rotations of each wheel determines the type of locomotion. In our case we use the SkyBot, which is a differential drive platform built for educational purposes. The customizable Skybot (figure 1) is a differential drive robot composed of printable wheels and two hobby servos.

The robot is made of polycarbonate parts and uses two electronic control boards called SkyPic and Sky293. The SkyPic is a minimal design with only the necessary components for controlling the robot. It includes an 8-bit PIC16F876A microcontroller, headers for connecting the servos, an I2C bus for additional communication, serial connection to the PC, a test LED and a switch for resetting the circuit. The Sky293 board is an extension board that was used to accommodate the sensors.

Fig. 1. The SkyBot mobile robot

2.2 Locomotion principle

In nature only limited joints exist. Thus, all animals move by means of repeated sequences of oscillatory limb movements: human gait is an example of repeated, simpler movements and each of them can be decomposed in flexion and extension. Those movements are the result of an anti-phase activity of skeletal muscles which is coordinated by CPGs and due to this sequential characteristic, most of their applications in mobile robotics target articulated snake-like and legged robots.

In this paper, the case of a wheeled robot is considered. All experiments involved the SkyBot, the differential drive platform built for educational purposes. It has been considered the case in which the wheels of the robot cannot perform complete rotations: this limitation can be attributed to the actuators that are used (linear motors, shape memory alloys, broken gearboxes, etc), to the environment (the wheel doesn't have enough space to rotate) or to the actual shape of the wheel (the wheel may be broken). Their movements are thus confined in an angular interval, let's say [-φ, φ]. In this case, a new locomotion principle can be applied such that the robot can still navigate, but with reduced mobility.

To analyze the movement, we consider the kinematic model of a differential drive robot with a castor wheel:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \theta \end{bmatrix} = \begin{bmatrix} \frac{R}{2}(\dot{\varphi}_1 + \dot{\varphi}_2)\cos(\frac{R}{L}(\varphi_1 - \varphi_2)) \\ \frac{R}{2}(\dot{\varphi}_1 + \dot{\varphi}_2)\sin(\frac{R}{L}(\varphi_1 - \varphi_2)) \\ \frac{R}{L}(\varphi_1 - \varphi_2) \end{bmatrix} \quad (1)$$

The new locomotion principle states that each wheel can perform only an oscillatory movement between the above given angle values. Coordinating the oscillatory movements of the wheels, three possible movements can be found:

- If the wheels oscillate in phase (phase difference is 0°), i.e. in the same direction and with the same frequency, the robot will move forward and backward repeatedly.
- If the wheels oscillate in anti-phase (phase difference is 180°), i.e. in opposite directions and with the same frequency, the robot will pivot left and right around a vertical axis (Instantaneous Centre of Rotation).
- If the wheels oscillate with a phase difference that's between 0 and 180°, the movement will be a combination of four movements: turning to one side, go forward, turn to the other side, go backward. We call "step" the length in a straight line of the robot's trajectory during a sequence of the four movements described above. It has been shown that the largest step value is obtained for a phase difference of 90°.

Other several terms that describe the motion are:

Initial phase (θ) determines the initial robot orientation relative to the path. It has no effect on the locomotion when the robot is travelling along its path.

Amplitude (A) is the rotation angle interval for each wheel; it has an effect on the step size.

Offset (O) is a positive or negative angular value that's added to the initial wheel angle such that the wheel's resting position is not φ, but φ+O. The maximum rotation angle will than be $\varphi_{max} - O$.

Direction (Γ) is the angle measured between the initial robot orientation and a new orientation, after changing the offset values. This depends on the physical robot parameters (wheel radius, base width). Let R and W be the wheel's radius and robot's base width, respectively. From the equations of the robot's kinematic model, the orientation angle can be found:

$$\Gamma = \frac{R}{W} \cdot O \quad (2)$$

Given the characteristics for this type of locomotion, the motion can be controlled by means of three parameters: amplitude - to modify the step, offset - to modify the direction of movement and frequency - to modify the speed. Because the locomotion is composed of coupled oscillatory movements, it is possible to design a CPG that can generate the appropriate motor commands to drive the robot.

3. CENTRAL PATTERN GENERATOR DESIGN

The first step in designing a central pattern generator is to select the building blocks and the topology that are to be used for the neural circuit. Taking into consideration the locomotion characteristics imposed by the robot platform, one must specify the means by which the information is encoded in the activity of the CPG. Three elements are needed in order to build a complete topology: neurons, synapses and some kind of modified neurons that are able

to read neural activity and translate it to motor commands capable to drive the actuators. These neurons are called motorneurons. The following sections briefly describe each component.

3.1 Rulkov neuron model

We use a neuron model developed by Rulkov et al. (Rulkov et al., 2005) that mimics the activity of living bursting neurons. The model is described as a two-dimensional map and is computationally efficient. Three stable regimes may be selected by combination of its parameters:

- Silent, in which the potential of the neuron remains in a constant resting state;
- Tonic spiking, in which the neuron produces spikes at a constant rate;
- Tonic bursting, in which bursts of spikes are produced at a constant rate, with a silent interval in between.

Furthermore, in the boundaries of the parametric regions of those regimes, chaotic behaviour may be found (Rulkov, 2002). The possible set of behaviours can be controlled depending on the selection of a few parameters. For this study we will set the parameters of the neurons to work in tonic bursting regime. Refer to figure 2 for an overall idea of the model working in all three regimes.

The mathematical description of Rulkov's model as used in this work is as follows:

$$\begin{cases} x_{n+1} = f(x_n, y_n + \beta_e \cdot I_n) \\ y_{n+1} = y_n - \mu \cdot (x_n + 1) + \mu \cdot \sigma + \mu \cdot \sigma_e \cdot I_n \end{cases} \quad (3)$$

$$f(x_n, y_n) = \begin{cases} \dfrac{\alpha}{1 - x_n} + y_n, & x_n < 0 \\ \alpha + y_n, & 0 \le x_n < \alpha + y_n \\ -1, & otherwise \end{cases} \quad (4)$$

This is a bi-dimensional model, where variable xn represents a neuron's membrane voltage and yn is a slow dynamic variable with no direct biological resemblance, but with similar meaning as gating variables in biological models that represent the fraction of open ion-channels in the cell. While xn oscillates on a fast time scale, representing individual spikes of the neuron, yn keeps track of the bursting cycle, a sort of context memory. Units are dimensionless, and can be rescaled to match the requirements of the robot. The combination of σ and α selects the working regime of the model: silent, tonic spiking or tonic bursting.

The bursting regime of the model presents a slow wave (slow time scale) with fast spikes of activity sitting on top of it (fast time scale). We use the slow time scale (yn in (3)) to encode movement duration, i.e., the temporal length of the burst defines the temporal length of the movement, and the fast time scale (xn in (3)) to define angular velocity of the servo (higher frequency of the spikes corresponds to faster servo movements). A value of μ = 0.001 was used in all experiments.

In figure 2, plot A represents the silent regime, B represents the tonic spiking regime and C displays the tonic bursting regime. Two parameters of the Rulkov model were modified to obtain different regimes; for the silent regime (A), α = 4, σ = 1; tonic spiking regime (B) is characterised by α = 4, σ = 0.01 while the tonic bursting behaviour (C) used α = 6, σ = 0.2.

Finally, external input is modelled through In. Depending on this value a neuron will modify its behaviour. For instance, an external stimulus, e.g. a sensory signal, may be input using this parameter. This property is essential for autonomous organization: processing units in the CPG must be able to negotiate the rhythm among them. Also, entrainment between the CPG and the physical robot can be achieved through In by adding an error term as external input to a neuron (Herrero-Carron, 2011 a). The total effect of this parameter will depend upon past history of events, the exact value of In and the phase within the burst cycle at which the neuron finds itself.

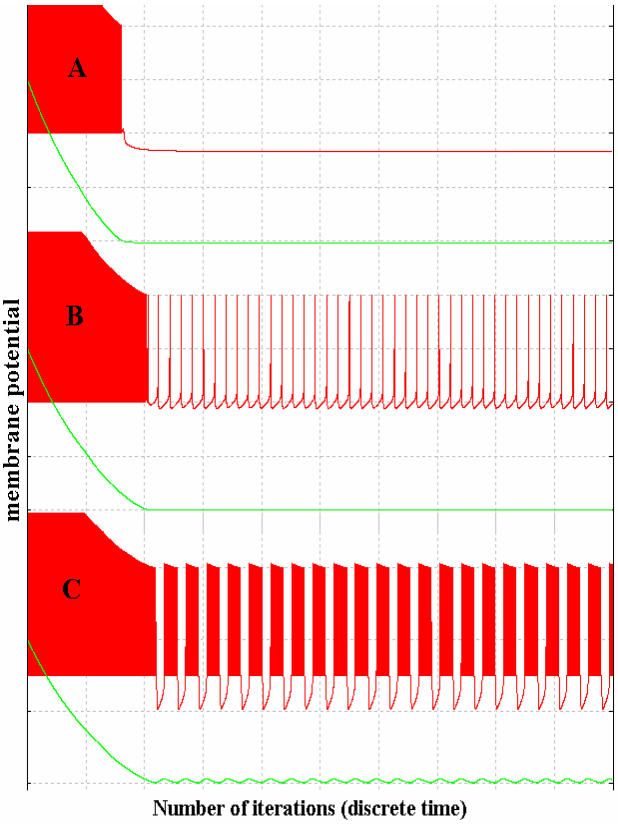

Fig. 2: Rulkov neuron regimes. Membrane potential (fast subsystem) is plotted in red while the slow subsystem is plotted in green.

In our work, In is the current flowing from one neuron to another: a periodic sampling of the continuous function described below in (6).

3.2 Kinetic synapse model

A key property of CPGs is that they are autonomous and the different units in the circuit talk to each other to negotiate the overall function. Here we present the model we have chosen to implement synapses, the communication channel of neurons. In this work we use a chemical synapse model (Destexhe, 1994). The Destexhe synapse model used in this paper keeps track of the ionic channels, neurotransmitter concentration and the binding and unbinding processes that occur between neurotransmitters and receptors located in the postsynaptic region.

Chemical synapses are unidirectional (see figure 3). When a potential spike arrives from the presynaptic neuron, the synapse releases a certain amount of neurotransmitter molecules that bind to the postsynaptic neuron's receptors. With time, neurotransmitter molecules begin to unbind. If a succession of spikes arrives within a short time, the synaptic response to each of them may overlap. Therefore the state of the synapse is dependent upon past events, a mechanism of context memory. The additional time-scale provided by kinetic synapses in a CPG enriches synchronization between bursting neurons. For instance, we may choose to synchronize two bursting neurons upon the spike (fast) time scale or the burst (slow) time scale.

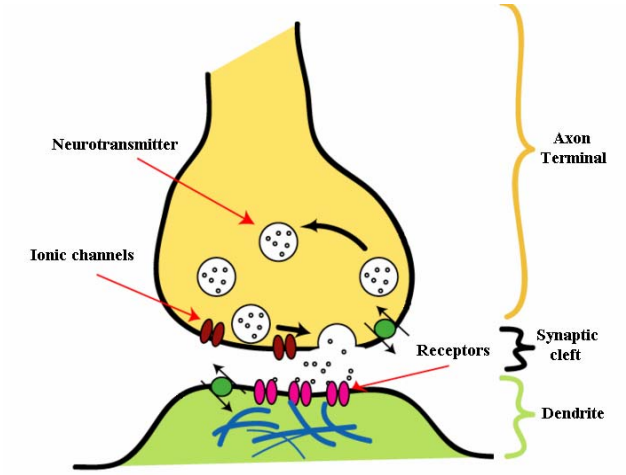

Fig. 3. Simplified model of a chemical synapse.

We have selected the kinetics of the binding and unbinding processes such that synapses act as filters of the fast time scale and synchronization occurs at the slow time scale. That is, the basic unit of synchronization will be the burst as a whole, not every individual spike. Beyond this, synapses may introduce delays for finer control of phase difference between neurons. The mathematical description of the model follows:

$$\dot{r} = \begin{cases} \lambda[T]\cdot(1-r) - \beta\cdot r, & t_f < t < t_f + t_r \\ -\beta\cdot r, & otherwise \end{cases} \quad (5)$$

This equation defines the ratio of bound chemical receptors in the postsynaptic neuron, where r is the fraction of bound receptors, and are the forward and backward rate constants for transmitter binding and [T] is neurotransmitter concentration. The equation is defined piecewise, depending on the specific times when the presynaptic neuron fires (tf): during tr units of time, the synapse is considered to be releasing neurotransmitters that bind to the postsynaptic neuron. After the release period, no more neurotransmitter is released and the only active process is that of unbinding, as described by the second part of the equation. Times tf are determined as the times when the presynaptic neuron's membrane potential crosses a given threshold. Synaptic current is then calculated as follows:

$$I(t) = g_{syn}\cdot r(t)\cdot(x_{post}(t) - E_{syn}), \quad (6)$$

where I(t) is postsynaptic current at time t, gsyn is synaptic conductance, r(t) is the fraction of bound receptors at time t, xpost(t) is the postsynaptic neuron's membrane potential and Esyn its reversal potential, the potential at which the net ionic flow through the membrane is zero. When coupling two Rulkov map neurons we will need to use a discrete synaptic function. We will build a sequence, let us call it In, by simulating I(t) as a continuous function and then taking samples every 0.001 time units. We say that a synapse is excitatory when the probability of the postsynaptic neuron firing a spike increases after the presynaptic neuron has fired. If the probability decreases, the synapse is inhibitory. If the postsynaptic neuron rhythmically emits spikes, an excitatory synapse will generally increase its frequency while an inhibitory one will generally decrease it.

3.3 Motorneuron model

Movement information is robustly encoded in the neurons' bursting episodes. A neuron called motorneuron is then responsible of decoding this information and translating it into the signal that will finally be sent to the servo controller. This signal tells the angle at which the servo should be positioned, in degrees.

With a slightly modified version of the motorneuron model provided in (Hererro-Carron, 2011 a), we have tried to mimic the real transformation occurring between living motorneurons and muscles. Motorneurons read the activity of other neurons using a simple threshold function that equals 1 if the membrane potentials of their corresponding neurons exceed the threshold values; otherwise the function equals 0. The role of this function is to detect individual spikes of neurons. By setting the threshold to, for example, υ = -1.5 a.u., this function applied to the potential trace of one neuron will have value 1 during individual spikes and 0 otherwise. In this way, communication between neurons is event-based. That is, the actual shape of neural activity is not so important, only their timing is. This can be a mechanism that the nervous system employs to lower the impact of noise. The role of motorneurons is now to integrate the individual events emitted by each one of the neurons. If a neuron emits a spike, the motorneuron will move the servo a little bit in a positive or a negative angle,

depending on the promotor or remotor effect that is intended (correspondently). If it emits a second spike close enough to the first one, the servo will be positioned a little bit further. If the neurons that are connected to the motorneuron are silent, the motorneuron will slowly drive the servo to a resting position of angle 0. This is accomplished through the following equation governing motorneurons in our CPG:

$$\dot{\varphi}_{1,2} = \dot{m}_{1,2} = C(t) - m_{1,2}(t) + O\,, \tag{7}$$

where m1,2(t) represents the output signal of the motoneurons. O is an offset value and C(t) is a function described by:

$$C(t) = \gamma \cdot (a_1 \cdot s(t,\nu) + a_2 \cdot s(t,\nu) + \ldots + a_n \cdot s(t,\nu)) \tag{8}$$

$$s(t,\nu) = \begin{cases} 1, & \text{if } x > \nu \\ 0, & \text{otherwise} \end{cases}, \tag{9}$$

where γ is the amplitude of the signal, s(t,υ) is a threshold function in which x is the membrane potential of the neuron, ν is a sort of threshold value. Terms a1,a2,…an describe the effect of the neurons over the motorneuron: if ai = 1, neuron i has a promotor effect on the corresponding motorneuron, if ai = -1, neuron i will have a remotor effect on the corresponding motorneuron.

3.4 Central pattern generator model

Given the characteristics for this type of locomotion, the motion can be controlled by means of three parameters: amplitude - to modify the step, offset - to modify the direction of movement and frequency - to modify the speed. Because the locomotion is composed of coupled oscillatory movements, it is possible to design a CPG that can generate the appropriate motor commands to drive the robot. The motion is split in four sequences:

- The wheels start by rotating in opposite directions with the same speed, offset and amplitude;
- Both wheels rotate forward;
- Both wheels rotate in opposite directions, but contrary to sequence 1;
- Both wheels rotate backwards.

The next design step is to identify what information needs to be encoded in the activity of the CPG. After that, a dynamical invariant must be defined, i.e. a stable, reproducible, repeatable neural pattern that is based on an activation sequence. As mentioned above, offsets, amplitude and speed of movement must be encoded. Also, a constant phase difference between 0 and 180° must be maintained in order to achieve a steady locomotion.

The model (figure 4) that was developed is comprised of a group of four neurons. To drive the wheels we need two motorneurons.

The main idea of the model is that all four neurons will trigger in a specific non-overlapping sequence and each of them will have a promotor or remotor effect on the motorneurons in such way that the motions sequences are obtained. The Rulkov neurons and synapse models are standard; the motorneurons were modified such that they are connected to all neurons of the model. The dynamical properties of neurons and synapses together with the topology of the circuit produce a coordinated alternating rhythm. Finally, the CPG will self-organize so that at any moment, each motorneuron will receive a signal from only one neuron to drive the wheels.

The angle values for the wheels will be encoded in the spiking activity of each neuron. Each spike represents a small increment in the wheel angle. The motorneurons then integrate the spikes and output the actual angle value, which is represented by m(t) in (3). Therefore, amplitude can be controlled by changing the number of spikes per burst period; the larger the number, the greater the wheel angle.

Amplitude can also be tweaked by changing the γ parameter. The speed of the movement is controlled by altering the burst periods. A longer burst will translate into a slower evolution of the motorneuron signal. In practice, because the angle increments from one iteration to another are very small and communication and servo lag is significant, a number of CPG iterations are done before sending new angle values to the servos.

Offsets are also encoded in the motorneurons, using the "O" parameter, such that the starting angle values are not zero, but O. By adjusting the offset values, one can change the trajectory of the robot. It is worth saying that in all cases, the robot will move in a straight line, regardless of the values of the offsets and the amplitudes. Changing the offset values, we change only the robot orientation.

Finally, the most important parameter encoded in the CPG is the phase difference that establishes the coordination between the wheels. In a more intuitive approach, phase difference is related with the time delay between the moment at which one wheel starts moving and the moment at which the second wheel starts moving. By measuring the angle difference between the wheels in this time interval, we can obtain the phase difference.

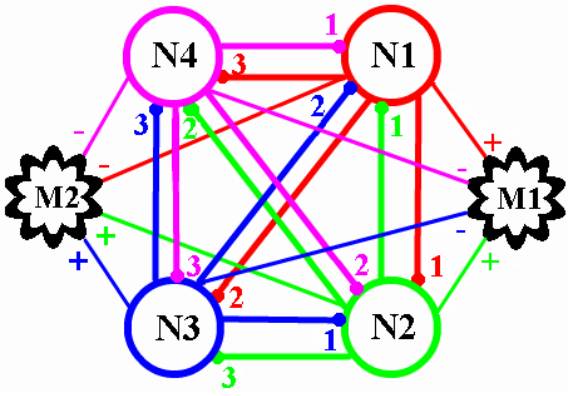

Fig. 4. CPG model. N1, N2, N3 and N4 are Rulkov models, M1 and M2 are motorneurons which output servo values.

In this CPG model, phase difference in achieved through the activation sequence of the neurons which, in term, is obtained using inhibition and asymmetric coupling. For a clearer example of how phase difference is obtained, let's consider this activation sequence: N1→N2→N3→N4. In figure 4, the network is completely connected, i.e. each neuron is connected to all others.

Also, each neuron is connected to both motorneurons and can simultaneously drive them. If all synapses have equal coupling strengths, the neurons will fire depending on which of them gets the chance to fire first. What we can do is encourage some neurons to fire before others in order to obtain the desired sequence and we do this by adjusting coupling strengths.

For the above chain of activation, synapses from the forward path (1, 3, 5, 7 in figure 4) must have a weaker inhibitory effect than all other synapses. Like so, neuron N2 will be favoured to burst before others (N1, N4), then N3 and so on. Therefore, we will end up with synapses that have a powerful inhibitory effect and synapses that have a weaker inhibitory effect. If all neurons share the same parameters (except for initial starting conditions), all strong synapses share a certain coupling strength and all weak ones share another coupling strength, the period of the whole CPG will be composed of four equal regions in which each one is a burst period of a corresponding neuron.

In figure 4, there are 12 "chemical" synapses and 8 "electrical" synapses. The signs from the electrical synapses indicate the effects (remotor or promotor) that neurons have on motorneurons. Each link between a neuron and the corresponding motorneuron has a positive or negative sign. This marks the promotor or remotor effect of the neuron on the respective motorneuron. As it can be seen in figure 4, during the activation period of N1, the motorneuron will be driven in order to increase the output signal (wheel angle), whereas an activation of N3 will decrease the motorneuron's output signal. Considering promotor and remotor neuron effects on motorneurons and given the fact that a complete wheel oscillation takes one CPG period and the delay between wheel movements are one burst region wide, a phase difference of 90° between the wheels is obtained. More intuitively, when one wheel has reached 1/4 of its oscillation, the second wheel will begin to rotate. With this model the phase difference cannot be changed due to the non-overlapping activation sequence. It is fixed to 90 degrees. A plot with the simulation of the CPG is presented in figure 5.

Note the non-linearity of the output angles. This is due to the neuron behaviour; the frequency of the individual spikes slightly varies across a bursting cycle so when spikes are integrated and we take into account the negative term in (6), near-linear responses are obtained.

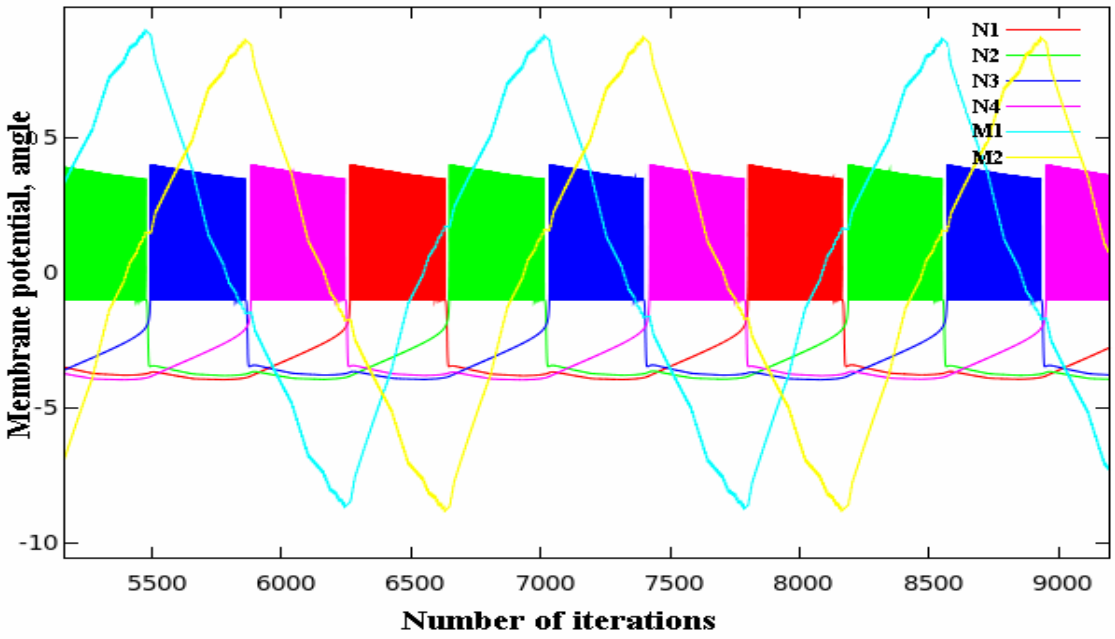

Fig. 5. A stable activation sequence is shown; the output values for the motorneurons were set between 9 and -9 ($\gamma = 9$).

To change the direction of movement between moving forward and moving backward, the activation sequence is changed by changing coupling strengths. For moving forward, the sequence is N1→N2→N3→N4. For moving backward, the sequence is N1→N4→N3→N2. To steer the robot, the wheel offsets and the amplitude have to be changed, so that $O + A \leq |\varphi_{max}|$, where φ_{max} is the maximum rotation angle (clockwise or anti-clockwise) that the joint can achieve. Figures 6 and 7 show plots from forward and backward activation sequences which were obtained solely by altering synaptic conductance. Next, we investigate what happens if we change the activation sequence, i.e. movement direction while the network is still running (figure 8).

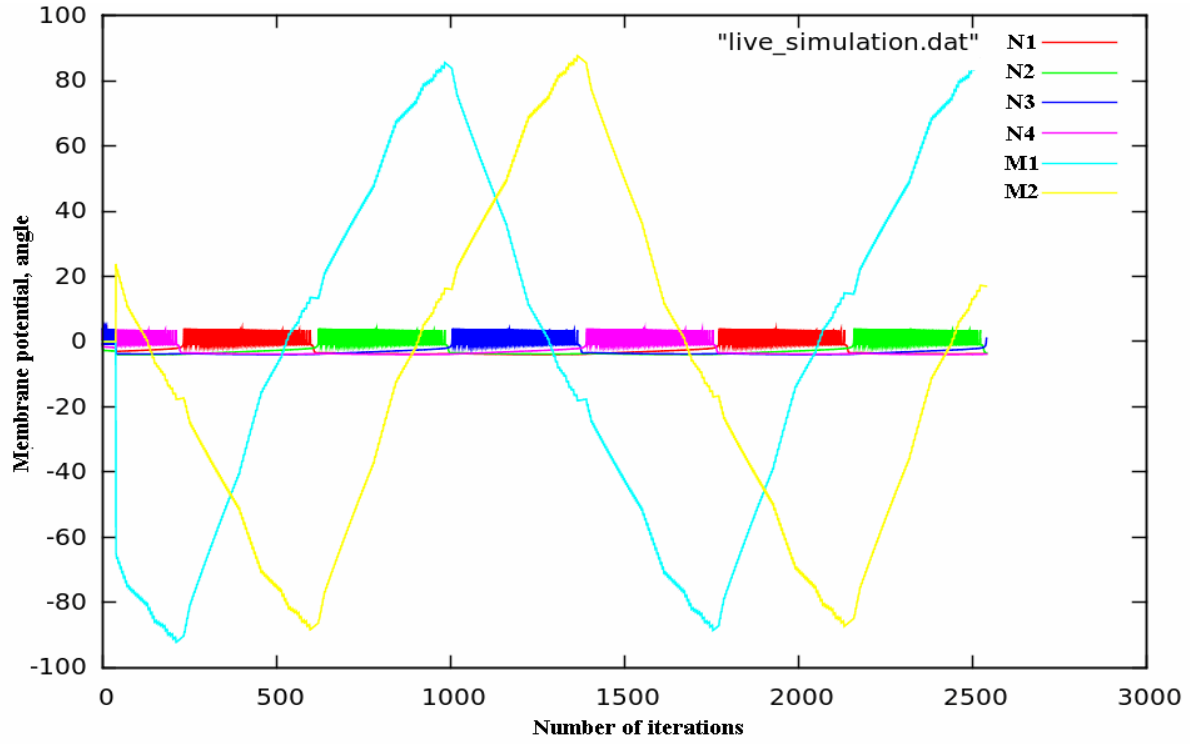

Fig. 6. Neuron activity (red, magenta, blue, green corresponding to N1, N2, N3, N4) and motorneuron signals (cyan, yellow). Plot extracted from a forward trajectory. The simulation was done by simulating one step at a time; the period, in simulation steps is about 1540. The burst periods of the neurons are approximately equal, with 364, 362, 360, 363 steps for N1, N2, N3, N4 respectively. The number of spikes per burst for each neuron was about 77.

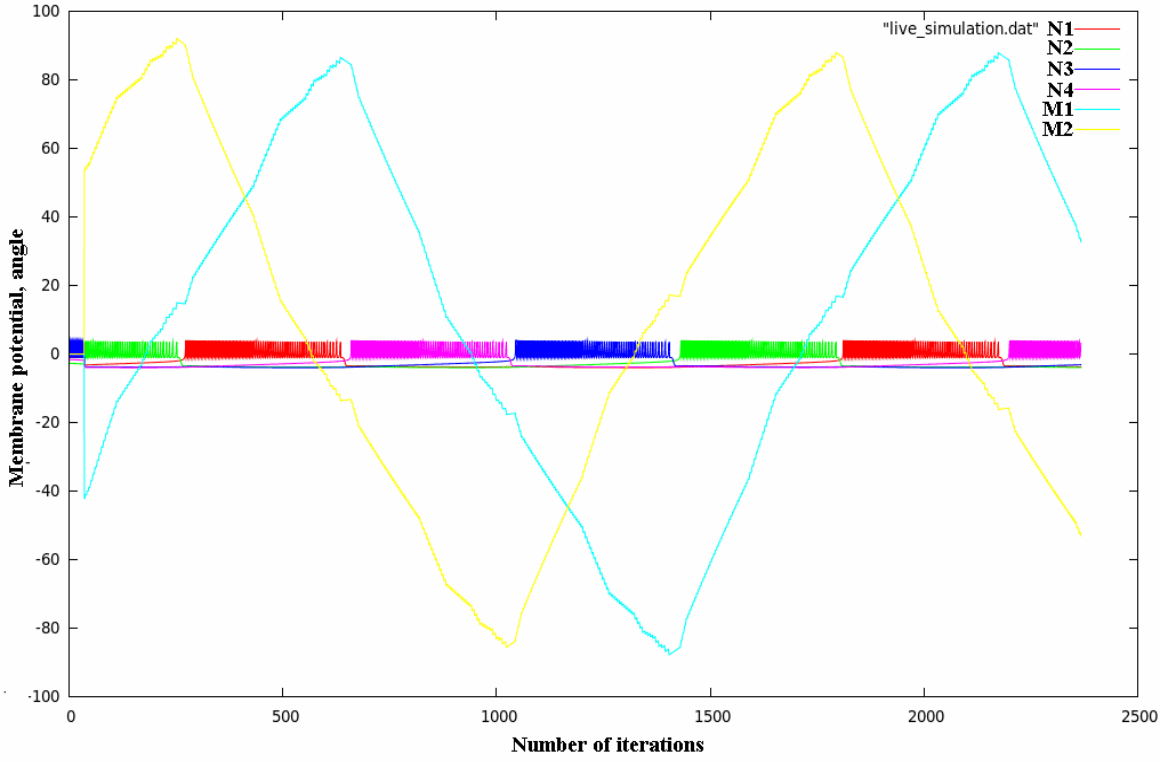

Fig. 7. Neuron activity (red, green, blue, magenta, corresponding to N1, N2, N3, N4) and motorneuron signals (cyan, yellow). Plot extracted from a backward trajectory. The simulation was done by simulating one step at a time; the period, in simulation steps is about 1540. The burst periods of the neurons are approximately equal, with 362, 362, 359, 361 steps for N1, N2, N3, N4 respectively. The number of spikes per burst for each neuron was about 77.The CPG is stable for both forward and backward trajectories, the oscillation periods are nearly identical.

In figure 8, note the transient period at the beginning of the simulation. After the synapses have been set, the neurons negotiate an activation sequence and after a certain number of iterations which depends on coupling strength, they achieve a stable rhythm (figure 8.A). We run the simulation for 10000 iterations and then change coupling strengths for a backward activation. The transient period lasts several hundred cycles in which a new stable sequence is achieved.

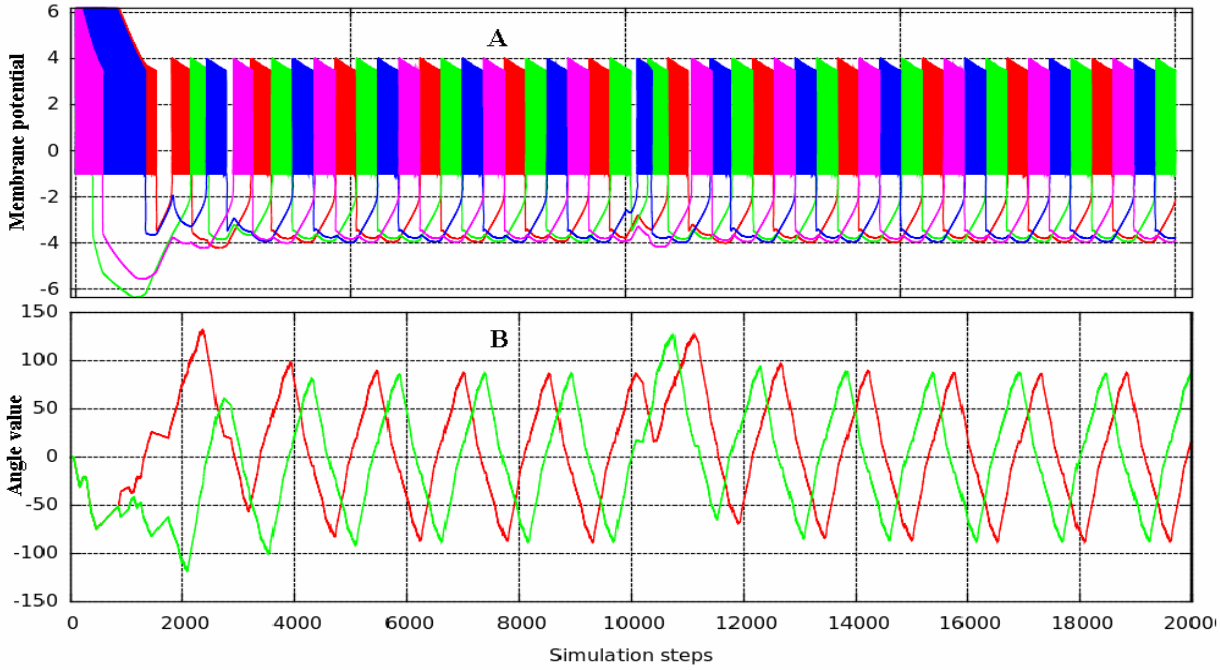

Fig. 8. Plot A: transient period including a change in movement direction. Plot B: motorneuron angles.

4. IMPLEMENTATION

4.1 Model and simulation

A first approximation of the robot trajectory, using the above described locomotion principle has been done by implementing an idealized kinematic model of a differential drive robot. By numerically integrating (1), we can obtain the robot positions and orientation relative to a global reference frame. The idealised model has been implemented using the Matlab/Simulink computing environment.

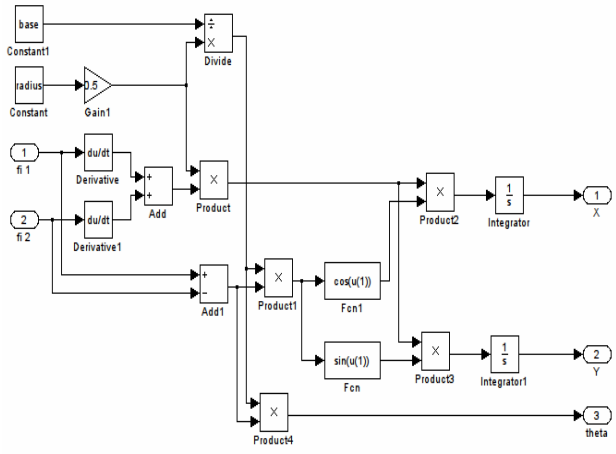

Fig. 9. The kinematic model of the robot modelled in Simulink.

We used this model to predict the robot's trajectory and observe how parameters like offset, amplitude, wheel size and base length affect the trajectory. Figure 10 illustrates the trajectory of the robot for forward locomotion.

The simulated trajectory of the differential drive using motoneuron-alike motor signals (sine signals) is phase-shifted by 90°. The simulation has been carried in Matlab/Simulink, with a signal amplitude of 90°. Wheel radius is 55 mm, base width is 103 mm.

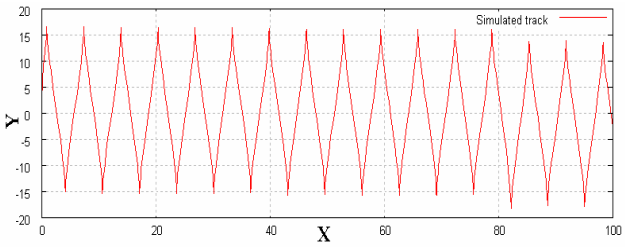

Figure 10: Simulation of a forward track using Matlab.

The CPG model has been implemented to drive the robot remotely. Because the PIC16F876A has a small data memory, the CPG model cannot be directly implemented on the microcontroller. Instead, it has been implemented on a personal computer and the results of the simulation are transmitted in real time to the robot.

Finally, we will present the schematic diagrams of the programs used to implement the CPG. Two programs were used to drive the robot.

One program runs on the PC and simulates the CPG (figure 11) and the other one (the slave) runs on the microcontroller and receives the commands sent by the first program (host program). The first program uses threads. In order to control the CPG, we must have the ability to alter its parameters while the simulation is running. For this reason we use threading.

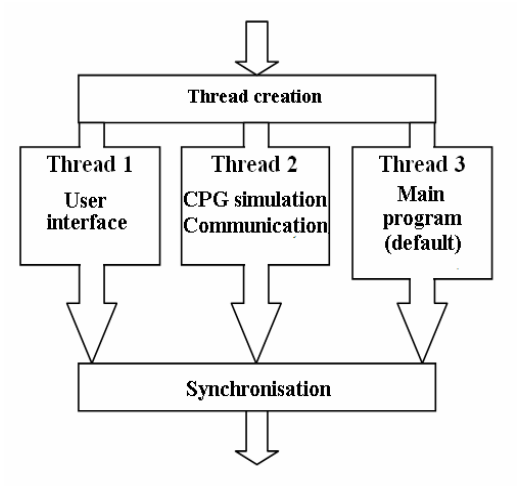

Fig. 11. Schematic diagram of the main control program.

The main thread iterates the CPG and outputs the positions for the wheels while a second thread listens for user input. According to the input, the CPG parameters are changed; figure 12 shows the flow chart on which the main thread is based. Because the angle increments between two successive iterations are relatively small (0.5 to 1.5 degrees), multiple iterations are computed before sending new wheel angles to the robot. No entrainment is implemented between the CPG and the actual servos.

The PC can compute the iterations very fast, faster than the servos can respond. To make sure that both servos have time to position before new angles are computed, a blocking communication protocol has been implemented between host and slave. The master does not compute new iterations unless it receives a “Ready for data” signal from the robot.

In order to operate correctly, mutual exclusion operations (mutex) had to be implemented. These operations are needed because in some situations different threads may try to access the same variable, leading to read-after-write, write-after-read or similar data dependencies. By using a mutex, variables that are used by a thread are locked by the same thread, not allowing others to read or modify that variable while it is in use.

The second program (figure 13) runs on the microcontroller. In the experiments, two bumpers were used as external sensors to detect collisions. The idea here is to detect collisions and generate a command that will modify the trajectory in order to avoid obstacles.

The robot will monitor the bumpers while receiving motor commands. After it received a set of angle values, the robot sends a new signal with a time delay that is long enough for the servos to position. This delay depends on the number of steps with which the host program is run. The larger this number, the bigger the angle differences between two motor commands and therefore the positioning time for the servo is bigger. Also, iterations between two motor commands must be small enough to correctly approximate the motor signals. The larger the number, the smaller is the sampling rate; that can lead to data loss in wheel positions.

Therefore, the sampling rate must be at least double the frequency of the motor signals (Nyquist).

Communications were implemented using a serial RS232 interface which is prone to errors when using simple hardware. In some cases, communications may be interrupted due to noise or other problems. For this reason, error detection and handling has been added to the protocol which enables the system to recover.

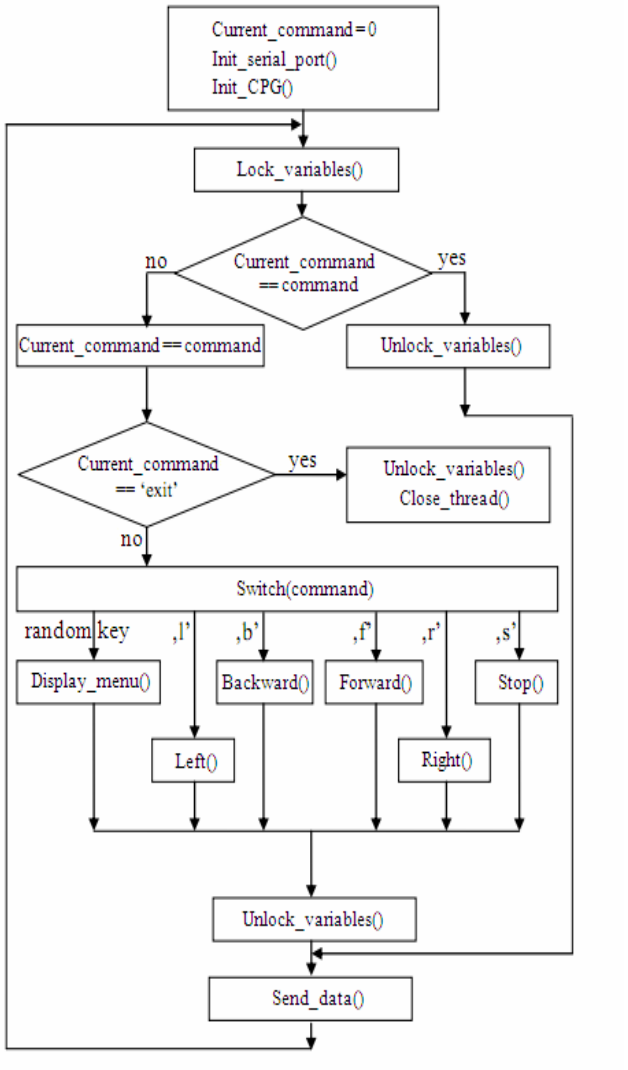

Fig. 12. Flow chart for the host CPG program.

According to the user input, functions that alter the parameters of the CPG are called. The actual CPG computation is done in the Send_data() block, where the

desired number of iterations are computed; the angle values are then sent to the robot via serial port.

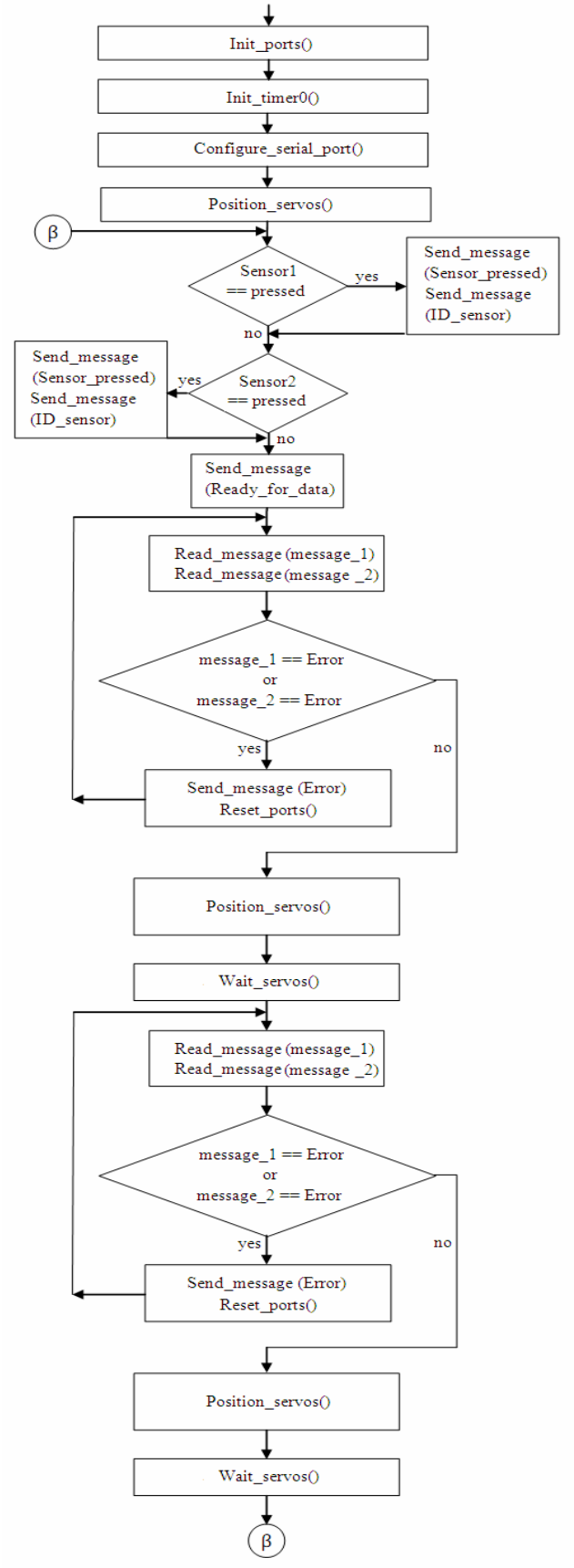

Fig. 13. Flow chart representing the SkyBot program.

Figure 13 illustrates the actual trajectory of the robot which was filmed while moving on a forward trajectory. The robot was recorded using a high resolution video camera and the trajectory has been extracted using motion capture software. Consecutive positions of the robot's center of mass were stored as coordinates which were plotted as a single XY graph.

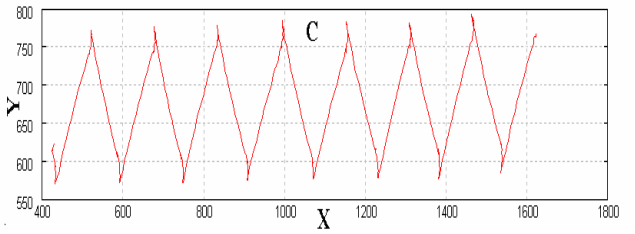

Fig. 13. Real robot trajectory extracted from video tracking of a forward movement (arbitrary pixel coordinates are used for Y axis, and number of frames is used for the X axis). Parameters used: neurons: $\alpha = 9$, $\sigma = 0.5$, $\sigma e = 1$; synapses: $\alpha = 0.5$, $\beta =$, Esyn = 9, gsyn1,2 = 25, T = 1, release_time = 0.01; motoneurons: $\gamma = 900$, $\upsilon = -1.5$, O = 0.

6. CONCLUSIONS

In this paper we show how a bio-inspired central pattern generator can be implemented so it can drive a differential wheeled robot where joint limit constraints have been added. The elements that made the CPG were considered, adding extra-functionality and explanations were considered necessary. The behaviour of the central pattern generator was analysed. Furthermore, the implementation of the control and communication protocols was described. The wide range of rhythm negotiation properties of the proposed model leads to a rich variety of self-organized locomotion. We consider that a bio-inspired approach on locomotion control will lead to increased autonomous robot behaviour. This includes better adaptation capabilities to external factors and immunity to noise.

ACKNOWLEDGMENT

This work was supported by an Erasmus grant under the Lifelong Learning Programme (2007-2013), with support from Escuela Politecnica Superior, Madrid, Spain, Universidad Carlos 3, Madrid, Spain and University of Craiova, Romania.

REFERENCES

Destexhe A., Z. F. Mainen, and T. J. Sejnowski, "An Efficient Method for Computing Synaptic Conductances Based on a Kinetic Model of Receptor Binding". *Neural Computation*, 6(1), 14-18, 1994, MIT Press. doi: 10.1162/neco.1994.6.1.14.

J. Ijspeert J. A. Central pattern generators for locomotion control in animals and robots: a review. *Neural Networks*, 21(4), pp. 642-653. Elsevier, 2008.

Gonzalez-Gomez J., J. G. Victores, A. Valero-Gomez, M. Abderrahim, “Motion Control of Differential Wheeled Robots with Joint Limit Constraints", in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011

Gonzalez-Gomez J., A. Valero-Gomez, A. Prieto-Moreno, M. Abderrahim, “A New Open Source 3D-printable Mobile Robotic Platform for Education”. *Proc. of the 6th Int. Symposium on Auto-nomous Minirobots for Research and Edutainment*, 2011, May, 23-25. Bielefeld. Germany.

Herrero-Carrón F., F. B. Rodríguez, and P. Varona (2011). Bio-inspired design strategies for central pattern generator control in modular robotics. *Bioinspir Biomim*, 6(1), 16006. doi: 10.1088/1748-3182/6/1/016006.(a)

Herrero-Carrón F, F.B. Rodriguez, P.Varona. Flexible entrainment in a bio-inspired modular oscillator for modular robot locomotion. Lect Notes *Comput. Sc* 6692: pp. 532-539.(b)

Herrero-Carrón F, F.B. Rodriguez, P. Varona. Studying robustness against noise in oscillators for robot control. *Proceedings of the ARCS Conference*, pp. 58-64, Orlando, 2011.(c)

Nițulescu M., Theoretical aspects in wheeled mobile robot control, *Proc. of International Conference on Automation, Quality and Testing, Robotics –AQTR* 2008, Cluj-Napoca, Romania, ISBN 978-1-4244-2576-1.

Rabinovich M., P. Varona, A. Selverston, and H. Abarbanel, “Dynamical principles in neuroscience”. *Reviews of Modern Physics*, 78(4), 1213-1265, 2006. APS. doi: 10.1103/RevModPhys.78.1213.

Rulkov N. F., “Modeling of spiking-bursting neural behavior using two-dimensional map”, *Physical Review*, vol. 65, 2002, pp. 1-9.

Shilnikov A. L. and N. F. Rulkov, “Origin of chaos in a two-dimensional map modeling spike-bursting neural activity”, Int. J. *Bif. and Chaos*, 13:3325–3340, 2003.