Development of a Low-cost Humanoid Robot with Neuromorphic Control System

Michele Folgheraiter, Amina Keldibek, Bauyrzhan Aubakir, Shyngys Salakchimov 1, Giuseppina Gini, Alessio Mauro Franchi, Matteo Bana 2

Abstract—A low-cost middle size humanoid biped robot built using 3D printing techniques and equipped with a neuromorphic control system is presented. The mechanical structure is made of polylactide (PLA), Acrylonitrile Butadiene Styrene (ABS) and aluminum. This allows to keep the overall weight under 3 Kg and still reach a sufficient structural rigidity. The joints are actuated by brush-less DC motors with integrated electronics and gearbox capable to generate up to 8N.m. Angular positions are measured by potentiometers and contact-less magnetic encoders. The control architecture consists of different modules capable to learn and autonomously reproduce complex periodic trajectories. Each module is represented by a recurrent neural network with a core of dynamic neurons randomly and sparsely connected with fixed synapses. A set of read-out units with adaptable connection realize a linear combination of the neurons output in order to reproduce the target signals.

I. INTRODUCTION

In order to exploit robotic technologies in household and public environments where machines are demanded to coexist with human beings a different design approach is required. If autonomy is a must for robots that need to move and act in unstructured environments, safety is of high concern to avoid that the robot comes to harm humans. In a classical industrial setup the manipulator is usually surrounded by mechanical and electronic barriers that do not allow humans to enter in its workspace. This, is not possible for humanoid robots that are required to move in environments shared with humans beings.

With the introduction of the industry 4.0 many companies, e.g. FESTO, BMW, etc., are currently upgrading their production plans by introducing new kind of manipulators that are specifically designed to interact with humans. This is achieved by increasing the footprint of the links using soft materials that reduce the damage of eventual collisions, or introducing sensory systems that implement virtual barriers that stop or deviate the trajectory of the robot if required. The first solution can not be fully implemented on a humanoid robot due the fact that it may limit its workspace. The second one presents some challenges due to the fact that the workspace of a humanoid robot is continuously changing.

A valid alternative is to reduce instead the weight and the inertia of the links by employing light-weight materials and high power-to-weight ratio actuation systems. This has the effect to reduce the potential and kinetic energy of the robot limbs while operating and therefore increases the overall safety when interacting with humans.

Autonomy and the capability to adapt to new environments are very important features for a humanoid robot. The learning and execution phases occur at the same time in biological organisms as they are forced to exist and survive in a changing environment while simultaneously learning and improving their sensory-motor strategies. Therefore, a challenging goal for humanoid robotics is to develop control systems that can be executed and adapted while the robot is operating. By integrating Artificial Neural Networks (ANN) in the control architecture the robot can elaborate input signals in real time [1] and respond to stimuli through a set of coordinated motor actions [2], [3], [4]. RNN (Recurrent Neural Networks) based on leak-integrate-and-fire neurons were proven to reproduce the behavior of microcircuits located in cortex of the human brain. In particular, they are capable to predict sensory inputs [5], compute tasks in real-time such as speech recognition and analysis of visual information [6] [7] [8] and generate reaching trajectories [9]. In [10] it was demonstrated that the presence in the circuit of feedback signals from read-out units with adaptable synapses allow the RNN to learn complex periodic signals without the excitation of additional inputs. The synapses can be adapted on the base of an error signal calculated as the difference between the target signal and the output of the neural circuit or by a reward-modulated Hebbian learning rule [11]. This, represents a more biological plausible mechanism that can be used in all the situations where the current performance of the robot depends on motor actions performed in the past [12]. In order to avoid local minima of the error function during the learning phase and a vanishing learning signal when the error is propagated “back in time” [13], an exploratory Hebbian (EH) learning mechanism [14] or p-delta rule [15] can be used to adapt the synapses of the read-out units.

The rest of the document is organized as follow: next section introduces the kinematic architecture of the robot and its main components, section III describes how the control system is organized and details the model of the RNN, section IV reports the preliminary results on a single control module capable to learn and generate periodic trajectories

1 School of Science and Technology, Robotics and Mechatronics Department, Nazarbayev University, Kazakhstan. michele.folgheraiter@nu.edu.kz, akeldibek@nu.edu.kz, b.aubakir@nu.edu.kz, shyngys.salakchimov@nu.edu.kz
2 DEIB, Politecnico di Milano, piazza L. da Vinci 32, Milano, Italy giuseppina.gini@polimi.it, alessio.mauro.franchi@polimi.it, matteo.bana@mail.polimi.it
*This work was supported by the Ministry of Education and Science of the Republic of Kazakhstan under the grant and target funding scheme agreement #220/073-2015.
generated with the robot simulator, finally last section draws the conclusions and indicates possible future developments.

II. KINEMATIC ARCHITECTURE

The system we designed presents, without considering the hands, a total of 28 Degrees of Freedom (DOFs). In particular, as depicted in Fig. 1, there are 6 DOFs in each leg, 6 DOFs in each arm, 2 DOF in the pelvis and 2 DOF in the neck. This will allow the robot to perform 3D dynamic walking and to reach different objects in its workspace by performing human like motions.

![Fig. 1. The Overall Humanoid Robot Kinematics Architecture counts 28 DOFs.](image)

Due to the fact that each joint will be subjected to a different load (torque) we considered installing different motors according to the joint location. Mainly they differ for the gearbox that allows setting a proper speed and torque maintaining in common the main motor unit. In order to reduce the complexity of the overall system we decided to set up only three different categories of motor-gearbox from which we can choose. We identified these as T1, T2 and T3. As it is possible to see from Table I each category presents a different weight (sum of the motor and gearbox weights), maximum torque, and angular speed. In general, the proximal joints (located near to the center of mass of the body) generate higher torque and are required to move more slowly. On the contrary, the actuators that move the distal joints are lighter and generate less torque. This is very important considering the fact that the distal joints need to be moved by the proximal joints. In other words, keeping the distal joints lighter will reduce the amount of torque needed by the proximal one and will reduce the weight of the overall system. To increase the system reliability we decided to employ three phases DC brushless motors with integrated electronics from Maxon company. In particular, we chose the EC32 and the EC20 flat motors combined with three different spur-gear boxes that allow reaching specific performances in terms of angular speed and torque.

![Table I: Joint Actuators Categories](image)

<table>
<thead>
<tr>
<th>Category</th>
<th>T1</th>
<th>T2</th>
<th>T3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overall Weight [g]</td>
<td>317</td>
<td>280</td>
<td>118</td>
</tr>
<tr>
<td>Torque [Nm]</td>
<td>8.5</td>
<td>3.5</td>
<td>2</td>
</tr>
<tr>
<td>Angular Speed [deg/s]</td>
<td>79</td>
<td>133</td>
<td>190</td>
</tr>
<tr>
<td>Nominal Power [W]</td>
<td>15</td>
<td>15</td>
<td>5</td>
</tr>
<tr>
<td>Torque-Angular Speed Weight [W/Kg]</td>
<td>2.11</td>
<td>1.66</td>
<td>3.2</td>
</tr>
</tbody>
</table>

By considering an average power consumption for each actuator of 40% of the nominal value during motion and 10% during a static behavior we estimate an overall power consumption of 126W and 32W respectively. To have a comparison, the human body during a normal activity consumes in average 120W.

In terms of autonomy, with a Lithium-Ion battery of 24V and 12Ah that can deliver a total of 288Wh, the system will have about two hours of working time before a recharge is needed. By considering Table 1 and the kinematics structure in Figure 1 we can also estimate the overall robot weight of about 13Kg.

\[ \text{Robot}_w = \text{Actuators}_w + \text{Structure}_w + \text{Battery}_w = 6.5Kg + 2.5Kg + 4Kg \]

(1)

Figure 2 reports a first prototype of the lower limbs build at Nazarbayev University in the Robotics and Mechatronics Department. It consists of a total of 10 DOFs, three in the hip, one in the knee and one in the ankle. The first degree of freedom in the hip mounts an absolute magnetic encoder, the AS5030 from ams AG company. It has a resolution of 1.4 deg and provides an analog signal in the range 0 – 5V. The others joints of the leg mount a potentiometer connected to the links by detachable parts. Each link including the motor has a weight of about 0.4Kg for a total of 3Kg.

III. CONTROL ARCHITECTURE

The control architecture of the robot is organized in a hierarchical manner (see Fig. 3). At the bottom level all the neural modules, that regulate the motion of each separated limb, are located. These modules have all the same structure. In particular, the core of each module consists of a RNN with fixed connections. The inputs and outputs of the module are routed toward an input and output layer of read-in and read-out units which synapses are tunable. While the structure of the low level modules is similar, the number of read-in and read-out units can be adapted during the operation of the robot. This brings the advantage to introduce new inputs and outputs when is required to perform a new task. Furthermore, if the read-in or read-out unit is wired to a specific piece of hardware (e.g. a joint motor or sensor) the architecture allows creating a new instance of the unit, adapting the correspondent synapses and passing smoothly the control of the hardware from the old to the new unit.
At an higher level the coordination module has the goal to govern the different neural modules located at the lowest level according to the specific task that needs to be performed. The outputs of this module are conveying different kind of information, e.g. the low level modules that control the leg need to receive information about the gait frequency, the step size, and the step height, while the modules that control the arms require the information about the target position, the speed and eventually the instantaneous impedance to be applied. The coordination module is receiving command and transmitting feedback to the planning module that in turns is responsible to plan and actuate the task. This, will not consist in a neural network, but will be based on logic sub modules.

A. Joints Trajectory Generation Module

Each module included in the control architecture of Fig. 3 is capable to learn and generate different elementary motor paths that represent the building blocks to implement more complex locomotion and manipulation primitives.

The RNN architecture we implemented (Fig. 4) consists of a core of neurons which are randomly and sparsely connected through excitatory and inhibitory synapses of different strengths which are set randomly as well [9]. In particular, a single-layer architecture is used; i.e. it was demonstrated that addition of hidden layers does not provide any significant improvements [16]. Feedback loops that send the output of linear read-out units back to the network should be included to make learning possible without any external input. Results of experiments conducted by Sussillo and Abbott [17] showed that an initially chaotic network is faster to train and generates more accurate and robust output signals; therefore, the initial activity of our model is set to chaotic. In this RNN architecture only read-out unit weights are modified whereas the other connections are not altered since their random initialization.

Fig. 2. Robot prototype, each leg counts 5 DOFs which axes of rotation are indicated with a dashed line.

Fig. 3. Overall Control Architecture.

Fig. 4. The single control module consists of a RNN where the outputs \( Z_i \) represent the read-out units.
Each neuron present in the circuit is modeled by a differential equation, Eq. 2, where $x_i$ is the membrane potential, $\tau_i$ a time constant, $C$ a parameter that modulates the chaoticity level of the RNN, $r_j$ and $u_j$, the output and input of the $j^{th}$ neuron respectively, and $z_j$ the signal generated by the read-out units. A nonlinearity (Eq 3) is added by a sigmoidal activation function that has also the function to limit the output of the neurons between [-1,1].

$$\tau_i \dot{x}_i(t) = -x_i(t) + C \sum_{j=1}^{N} w_{ij}^c r_j(t) + \sum_{j=1}^{L} w_{ij}^h z_j(t) + \sum_{j=1}^{M} w_{ij}^u u_j(t)$$

(2)

$$\Phi(x) = \frac{1 - e^{-kx}}{1 + e^{-kx}}$$

(3)

To explore new dynamic regimes [18] a Gaussian noise with zero average and variance $var = 0.005$ is added to the neurons output and the read-out units (Eq. 4 and Eq. 5 respectively).

$$r_j(t) = \Phi(x_j(t)) + GNoise_j(t)$$

(4)

The probability of a connection between two neurons Neuron $- I$ and Neuron $- J$ is $P = 0.2$ and its strength $W^{C}(I, J)$ is randomly initialized in the range $[-1,1]$. The RNN presents a total of $50\%$ of inhibitory synapses and $50\%$ of excitatory one.

Although the circuit is able to generate periodic trajectories without additional signals, external inputs $u(t)$ can be added to influence the neuron potentials according to the weights vector $W^{in} \in \mathbb{R}^N$. This can reduce the presence of phase shift in the generated trajectories during time.

Furthermore, an additional signal scaled by the $L$-by-$N$ matrix $W^{fb}$ forms a feedback from the outputs of read-out neurons.

The RNN's output $z(t)$ is computed by a linear combination (Eq.5) of the outputs of the read-out units $r(t)$ by mean of a $N$-by-$L$ matrix $W^Ad$.

$$z(t) = W^{Ad} r(t) + GNoise(t)$$

(5)

The adaption of the matrix $W^{Ad}$ is performed by a simple learning rule based on the error calculated as the difference between the target vector and the filtered version $\bar{z}(t)$ of the read-out unit vector $z(t)$ at the instant $t$ as in Eq. 6.

$$\text{Err}(t) = \bar{z}(t) - F(z(t))$$

(6)

$$W^{Ad}(t + 1) = W^{Ad}(t) - \eta(t) F(r(t)) \text{Err}(t)$$

(7)

The learning constant decays according to the rule in Eq. 8

$$\eta(t + 1) = \frac{\eta(t)}{1 + t/dl}.$$  

IV. SIMULATION RESULTS

As target trajectories we used the signal acquired from a simulator of the robot performing a static walking. In particular, we used the open source VREP simulator [19] that is free of charge for educational purposes. The software allows simulating different robots kinematics and dynamics, implements among others functionalities: collision detection, virtual sensors and actuators, and particles dynamics. We acquired the position of the the right leg joints during the execution of a static walk with a sampling time of $10ms$ for 220s and a total of 22000 samples.

The RNN we implemented is relatively small [14] and can be implemented on power-efficient computational units. It consists of 100 neurons and 5 read-out units receiving as inputs all the neurons outputs. The main parameters are reported in the Table II.

<table>
<thead>
<tr>
<th>Quantity</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Neurons $N$</td>
<td>100</td>
</tr>
<tr>
<td>Number of External Inputs $I$</td>
<td>0</td>
</tr>
<tr>
<td>Number of Read-Out Units $L$</td>
<td>5</td>
</tr>
<tr>
<td>Computational Time</td>
<td>0.0001s</td>
</tr>
<tr>
<td>Neuron Time Constants $\tau$</td>
<td>0.013</td>
</tr>
<tr>
<td>Learning Constant $\eta$</td>
<td>0.002</td>
</tr>
<tr>
<td>Decay Learning Constant $dl$</td>
<td>200</td>
</tr>
<tr>
<td>Chaos-Modulation Constant $C$</td>
<td>1.2</td>
</tr>
</tbody>
</table>

Particular critical are the initialization of the matrix $W^{C}(I, J)$, the Chaos-Modulation constant, the neurons time constant, and the learning constant. The first three parameters dictate the dynamics of the RNN that needs to pass from an initial chaotic regime to a stable behavior where the neuron outputs are periodic signals of constant amplitude (see Fig. 5.

Fig. 5. ReadOut-1 synapses adaptation.
7). This allows to reproduce the target signals with a specific linear combination of the neurons output performed by the read-out units. The learning constant is also very important, it should be higher at the beginning to force the RNN to reach a limit cycle (see Fig. 8), while its value should decrease during the learning phase.

To find an optimal value of the matrix $W^C(I,J)$ we performed different experiments where the main parameters were initialized and kept constant while the matrix $W^C(I,J)$ was randomly initialized for each model instance. In each experiment the RNN was trained and the overall RMS error calculated as the sum of the RMS errors of each read-out unit. As it is possible to see from Fig. 9 the average RMS error is 1.63 while its variance is 0.74.

In Fig. 10 and Fig. 11 the outputs of the first two read-out unit are reported. As a black dashed line the target signal is represent while the continuous red line represent the RNN outputs. As it is possible to notice, when the learning phase is completed the RNN is able to reproduce the periodic signals autonomously.
Fig. 11. ReadOut-2, reference signal in a dashed line and RNN signal in a red line.

start to experience a phase shift. This can however be eliminated introducing an additional RNN’s input with a specific frequency that serves as a “clock” reference for the system.

V. CONCLUSIONS

In this paper we presented the development of a humanoid robot equipped with a neuromorphic control architecture. The lower limbs present a total of 10 DOF actuated with compact brush-less DC motors with integrated electronics. The joints trajectories, necessary to perform a static walking, are learned and reproduced by different modules organized in a hierarchical fashion. Each module is represented by a RNN consisting of 100 neurons modeled by a first order differential equation. The neurons are coupled with fixed connections randomly initialized while special read-out units with adaptive synapses realize a linear combination of the neurons outputs in order to reproduce complex periodic signals. To optimize the weights matrix 10 experiments were performed where the training and validation phases were accomplished and the RMS error calculated. Preliminary simulation results show that despite the small dimension of the network the module is able to reproduce the required trajectories. As a future work we intend to implement the algorithm on a small computational unit e.g. a Raspberry Pi 3 or a BeagleBone and test the trajectories on the real prototype. It will be interesting also to integrate in the control architecture additional input signals from the sensory system of the robot. This, will allow to implement more complex motor behaviors, e.g. alter the step high when the robot encounters an obstacle in its path.

ACKNOWLEDGMENT

This work was supported by the Ministry of Education and Science of the Republic of Kazakhstan under the grant and target funding scheme agreement #220/073-2015.

REFERENCES


