Journal of the Brazilian Society of Mechanical Sciences and Engineering (2024) 46:561
https://doi.org/10.1007/s40430-024-05134-z

TECHNICAL PAPER

Design and programming of a robotic puppetry robot based on natural learner unit pattern generators neural networks

Hamed Shahbazi[1] · Behnam Khodabandeh[1] · Masoud Amirkhani[1] · Amir Hasan Monadjemi[2]

Received: 8 January 2024 / Accepted: 5 August 2024 / Published online: 19 August 2024

Abstract
The purpose of this study is to design and construct a novel interactive game. This game is a robotic learning and imitation task. It is based on visual interaction of the player. The cornerstone technique used in this game is natural learner unit pattern generator neural networks (NLUPGNN), which is able to generate required motion trajectories based on imitation learning. The systematic design of these neural networks is the main problem solved in this paper. The unit pattern generators can be divided into two subsystems, a rhythmic system and a discrete system. A special learning algorithm is designed to use these unit pattern generators. The unit pattern generators are connected and coupled to each other to form a network, and their unknown parameters are found by a natural policy gradient learning algorithm. The motion sequences train some nonlinear oscillators, then they reproduce motions for a humanoid robot. As a result, the joints of the humanoid body imitate the movements of the teacher in real time. The main contribution of this work is the development of this learning algorithm, which is able to search the weights and topology of the network simultaneously. The algorithm synchronizes the learning steps by coupling the neurons in the last step.

Keywords Puppet show · Humanoid robot · Imitation learning · Unit pattern generator · Microsoft kinect

1 Introduction

Puppetry, an ancient form of storytelling and entertainment, has roots tracing back to 421 A.D. Historically, the art required the undivided attention of the puppeteer, a limitation that modern robotics seeks to overcome. Today's technological advancements have birthed systems capable of learning and mimicking the intricate behaviors of puppeteers. One of the most promising areas in this domain is imitation learning, a branch of supervised learning, which empowers humanoid robots to assimilate and replicate complex human movements. This method leverages neural networks to absorb extensive knowledge from human demonstrations, a technique that has seen significant developments in recent years [1].

Imitation learning is a process by which a teacher's behavior adapts to a set of similar movement primitives. In other words, a set of these primitives has been defined in the human body, and when the teacher performs an action that is adapted to these primitives, imitation learning occurs. Motion primitives have three main advantages. First, they reduce the workspace of the system (robot) to the control parameters of the motion primitives and make the problem easier to solve. Second, they provide the system with fast and local feedback loops that can generate online trajectories. Finally, different degrees of freedom can be coupled to resist disturbances and external forces [2].

In the field of robotics, learning is conceptualized as the process of mapping environmental states to corresponding actions, a construct known as a policy [2]. This study employs imitation learning, a subset of supervised learning, to impart complex behaviors to humanoid robots. The methodology involves capturing experimental data from a

Technical Editor: Rogério Sales Gonçalves.

✉ Hamed Shahbazi
shahbazi@eng.ui.ac.ir

Behnam Khodabandeh
be.khodabandeh@eng.ui.ac.ir

Masoud Amirkhani
m.amirkhani@eng.ui.ac.ir

Amir Hasan Monadjemi
monadjemi@eng.ui.ac.ir

[1] Department of Mechanical Engineering, Faculty of Engineering, University of Isfahan, Isfahan, Iran

[2] Department of Artificial Intelligence, Faculty of Engineering, University of Isfahan, Isfahan, Iran

teacher's movements, which is then used to derive a policy that enables the robot to act in accordance with its current state.

Three distinct methods for policy elicitation are explored. The first, direct control policy imitation, is straightforward but lacks stability for the imitated movements and adaptability to new targets [3]. The second, model-based policy learning, utilizes motion dynamics to forecast a model, typically within the reinforcement learning framework [4]. The third, policy learning from control trajectories, guarantees movement stability and facilitates hierarchical motion management, albeit requiring advanced skills to define critical features and segments [5].

To address the challenges of time-dependent strategy extraction and trajectory encoding, inherent dynamics are employed. The hidden Markov model is utilized for learning complex motions [6, 7], while nonlinear dynamical systems are applied to teach discrete movements to a humanoid robot [8]. The movement primitive method, based on nonlinear dynamic systems, offers another avenue for motion demonstration [9]. This research diverges from traditional methods that rely on frequency detection, such as Fourier analysis. Instead, it focuses on adaptive frequency oscillators to discern and adjust to the frequencies inherent in dynamic movements, eliminating the need for explicit frequency extraction to replicate the teacher's movements.

Locomotion, a critical faculty of organisms, is fundamental to their evolutionary success and survival. In the realm of vertebrates, and notably in humans, the capacity to perform a diverse array of complex movements is indicative of sophisticated cognitive functions, which are intricately integrated within the neural and musculoskeletal frameworks [10]. This ability is reflective of the profound intelligence that characterizes vertebrates, an intelligence that is seamlessly interwoven with their neurological and physiological systems.

In the field of engineering science, there has been a concerted effort to design and construct robotic systems that draw inspiration from the biomechanical prowess of living creatures. These systems aim to replicate the locomotive abilities observed in the animal kingdom, presenting a significant challenge in the synthesis of motion within robotic constructs [11]. The generation of motion involves the creation of control trajectories that are transmitted to a robot's joints over time intervals. Imitation learning has emerged as a viable and efficient method for teaching complex behaviors to humanoid robots, offering a solution to many of the challenges inherent in motion generation [12].

The orchestration of control trajectories in robotics, guided by intelligent scheduling, represents a quintessential challenge in the domain of artificial intelligence. Pioneering research in robotics utilizing imitation learning has delved into the amalgamation of observational learning with actionable outcomes [13]. Extensive studies have been dedicated to mastering the control of human movements, capturing joint coordination through video analysis [14]. The nuanced control of industrial robot arms, aimed at replicating specific movements through learning algorithms, plays a pivotal role in advancing robotics.

Imitation learning in robotics is underpinned by a tripartite process: sensory perception, cognitive interpretation, and execution. The core challenges in this domain revolve around these elements, particularly the application of demonstrated information to initiate actions. Schaal's seminal work on motion primitives offers a compelling solution to this challenge, defining a repertoire of actions that encapsulate desired behaviors [15]. The concept of emulating complex human-like behaviors is rooted in the study of biological movement primitives. Robots achieve this through a dualistic approach, employing both discrete dynamical systems and rhythmic mechanisms, the latter known as central pattern generators (CPG). The culmination of CPG activity results in the generation of motion primitives [16].

The exploration of natural mechanisms and their synthetic counterparts has been extensive. Atkeson and Schaal's contribution to this field includes a control paradigm where robots assimilate actions of reward and punishment through demonstration, subsequently replicating them via a recurrent model. This methodology leverages specialized visual sensors to transpose a teacher's head movements onto a robotic counterpart [17].

In the traditional paradigm, robots are operated via specific interfaces, such as control panels or computers. The emerging field of social robotics, however, envisions a paradigm where robots interact with humans in a seamless and intuitive manner. Gesture recognition becomes a critical component in this context, as nonverbal cues like pointing and signaling are integral to human communication [18]. The Microsoft Kinect sensor has been utilized effectively as a body gesture recognizer to capture the instructor's movements.

The work of Góngora Alonso et al. [19] explores the application of social robots to support the elderly, including those with dementia, underscoring the benefits to their autonomy and quality of life. Di Palo and Johns [20] have introduced an innovative method for robots to learn multistage tasks from a single demonstration through self-replay, enabling autonomous data collection and problem solving without prior knowledge of objects. Correia and Alexandre [21] offer an exhaustive survey on station learning, where agents acquire skills by mimicking expert demonstrations, encompassing dataset creation, learning methodologies, optimization strategies, benchmarks, and practical applications.

Chen et al. [22] have developed a robotic system capable of drawing portraits using advanced deep learning and path planning techniques, with evaluations based on image

fidelity and operational efficiency. Wei et al. [23] discuss a vision-based navigation strategy for humanoid robots in complex environments, presenting a general algorithm for path correction validated through empirical testing. Do et al. [24] propose a framework for social robots to conduct clinical screenings in geriatric care, enhancing their utility in cognitive assessments and risk evaluations.

Section 2 of the manuscript presents the core application of this study, focusing on an interactive game that integrates puppetry elements. Subsequently, Sect. 2.2 investigates the domain of imitation learning, with a particular emphasis on the utilization of additive frequency oscillators. This subsection also provides a thorough review of previous project-related endeavors and discusses the challenges identified within these methods. Section 2.3 details the motion recognition process, which is achieved through the central pattern generator (CPG) architecture. Section 3 introduces the innovative two-layered learning system developed in this research for instructing humanoid robots in complex behaviors. The initial layer employs a Kinect sensor to capture and document the instructor's movements, subsequently extracting joint patterns via a geometric approach. The subsequent layer is responsible for training these patterns to generate comprehensive movement sets, exemplified by activities such as drumming or playing tennis. A detailed evaluation and comparative analysis of the proposed methodologies are provided in Sect. 5. The final section of the paper synthesizes the key findings and discusses potential directions for future research endeavors.

2 Literature review

2.1 Puppetry game

Theatrical robots, especially those emulating puppets, have become a significant facet of robotic entertainment (refer to Fig. 1). The enduring constraint of traditional puppetry has been the requisite for constant puppeteer engagement. Modern advancements in robotics aim to surmount this challenge by developing systems adept at assimilating and duplicating the intricate maneuvers of puppeteers. Emulating such complex movements, which are indicative of cognitive prowess in both vertebrates and humans, poses an ongoing challenge within the field of robotics. This challenge calls for pioneering solutions that are informed by the biomechanical principles observed in natural organisms.

The game system is engineered to replicate puppet movements by interpreting the player's audiovisual speech and actions. It comprises various components, each with distinct functions yet with some task overlap, and each independently produced and tested. The primary module includes a servo motor, plastic casing, control circuitry, power sources, and sensors for audiovisual transmission, showcased to the audience. A smartphone, substituting for the robot's head, captures and wirelessly transmits sounds and images to a computer. An RGB-D KINECT sensor records the player's movements, facilitating gesture recognition and creating a visual interface for human–robot interaction. Control signals derived from these gestures prompt the robot to execute tasks, thereby augmenting human–robot engagement. Control policy derivation and execution are shown in Fig. 2.

Fig. 1 Puppet show in Persia (Iran)

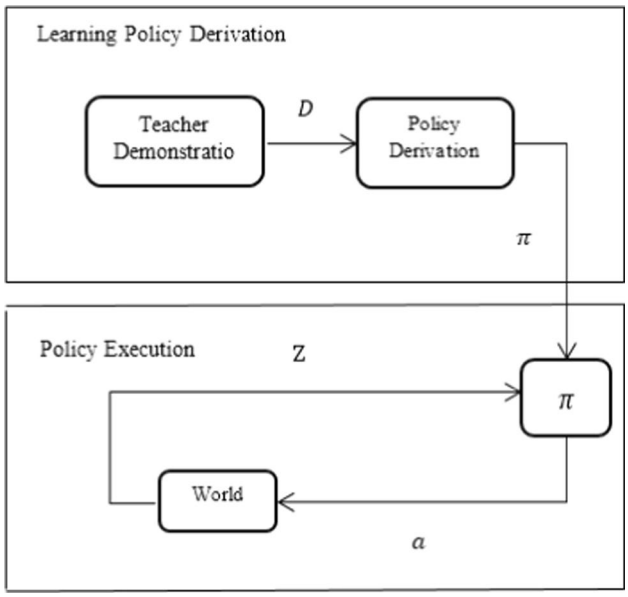

Fig. 2 Control policy derivation and execution

The third part of the system is responsible for learning and reproducing gestures and sounds. This section is able to communicate with human players and receive general commands to operate the puppetry. In the case of system malfunction that led to a crisis situation, the third part of the robot is automatically isolated from the human, and the human player controls the system. The imitation learning mathematical model of the robot was described in our publications [12, 25–28].

2.2 Example-based imitation learning

In the process of example collection for robotic tasks, two pivotal decisions are made: selecting the presenter and the display technique. The presenter, who demonstrates the task, can be either a robot or a human, depending on the robot's capabilities and the task complexity. For instance, in a box-moving task, the presenter might be a robot that physically moves the box, thereby controlling and executing the demonstration. Alternatively, a human may control the robot's movements to perform the task, or a human may directly demonstrate the task, controlling both display and execution. The presenter's choice significantly influences the learning process's efficiency, as a closer match between the teacher's and learner's state spaces simplifies the learning algorithm. Moreover, the display technique is crucial for generating quality data, with various methods available for recording sample trajectories. This strategic selection streamlines the robot's learning phase, enhancing its performance in executing the demonstrated tasks.

1. Recording movements using optical and magnetic trackers [29].
2. Physical approach [30].
3. Computer vision techniques [31].

Recording human movements for robot learning presents various methodologies, each with its unique challenges and advantages. Traditional optical tracking methods require the attachment of sensors to the human body, demanding precise calibration and posing difficulties in public settings [32]. These methods necessitate special equipment and familiarity with the robot's physical constraints, which may not be user-friendly for non-experts.

Conversely, direct motion learning approaches involve recording movements with the robot's own sensors, eliminating the need for complex system mappings [33]. This method allows for natural movement demonstrations by the teacher, directly influencing the robot's joint movements within its operational constraints.

Vision-based techniques aim to capture human movements without the need for markers, relying on camera images to extract, transform, and map actions into the robot's understanding [34]. The TP-GMM framework, for instance, encodes and reproduces human demonstrations marker-free, leveraging deep neural networks for task learning, such as table cleaning [35]. Similarly, Kinect-based motion identification methods utilize hidden Markov models to process large-scale video data, enabling natural user interfaces and user-adapted gesture recognition [36].

These motion learning methods simplify the coaching process, allowing for more natural and intuitive teaching of robots. The Kinect camera plays a pivotal role in capturing the trajectory of each joint's degree of freedom, as detailed in Sect. 3. Ultimately, the chosen technique significantly impacts the ease of teaching and the robot's ability to perform tasks accurately, highlighting the importance of selecting an appropriate method for motion capture and learning.

From this perspective, the techniques used to extract policy can be divided into three main categories:

1. Direct policy imitation: Despite its simplicity, this control method does not guarantee the stability of the imitated behavior; that is, it cannot perform the desired behavior from any starting point. Direct imitation methods usually produce policies that cannot be reused for a behavioral goal. If a specific goal is taught and the goal position changes, the commands issued are incorrect [37]. After collecting a set of key points of motion, the motion is estimated by interpolation. Direct policy imitation methods are usually called "aimless imitation" or "pure imitation" because they can only repeat the observed pattern and do not know changing this pattern in a new behavioral environment.

2. Model-based policy imitation: In this method, a predictive model of system dynamics is estimated from the displayed behavior, which is improved by trial and error [38]. The designer first obtains the robot model and then creates a control policy for the robot. This method is suitable for expensive robots that do not fall to the ground. The main problem with this method is its deep dependence on the accurate model of the robot. However, for each robot in particular, all the details of these methods need to be designed and implemented from scratch, and this dependence on the accuracy of the model affects the final performance.
3. Imitation learning from displayed trajectories: This method ensures that the imitated behavior is stable and modulatable. Various methods are included in the subset of this technique for encoding trajectories [39].

2.3 Central pattern generator

Central pattern generators are biological neural circuits that generate rhythmic patterns without sensory feedback. These generators can help us perform complex movements with simple inputs [40]. The main practical advantage of using them in robots is that they do not require mathematical modeling of the robot. In addition, due to their biological nature, these methods are not sensitive to perturbations and can quickly return to the original state after a perturbation [41]. In modeling CPGs, we focused on the set of neuronal activities and how rhythmic and oscillatory actions can be produced by the neuronal network. One of the common CPG models is the nonlinear oscillator model. The basic design of central pattern generators is based on adaptive nonlinear oscillators. In our research, we used Hopf oscillators, which can adapt their inherent frequencies to complicated signals. The main feature of these oscillators is their independence in learning information from the input signals. This means that all learning processes are embedded in the dynamic equations [27]. These adaptive oscillators are connected by a few weighted links to form the pattern generator. Input trajectories are trained by a training input signal (P_{teach}) to produce the desired output (Q_{learn}). In this structure, each oscillator receives the training signal, which is the difference between the desired and the learned pattern ($P_{\text{teach}} - \sum a_i x_i$). Then it couples to the first oscillator to eliminate the noise [42]. The connection of the Hopf oscillators is shown in Fig. 3.

In terms of robot control, the concept behind CPGs is that the motion produced by pattern generators has parameters in the form of control signals. Motion primitives are the outputs of these generators. This method reduces the problem dimensions and determines the key parameters instead of determining all paths. Generated paths produced by pattern generators can be adjusted by sensory feedback to adapt and cope with uncertain environments [43]. Dynamical systems

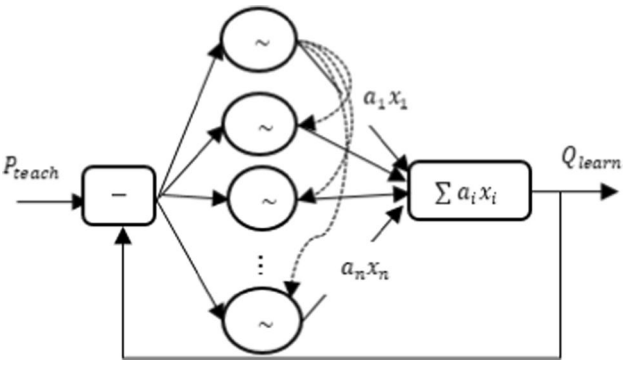

Fig. 3 Connection of the Hopf oscillator [42]

with absorbed properties were used to model these primitives. The main advantage of motion primitives over traditional methods is threefold. First, they simplify the problem by reducing the robot's workspace. Second, they generate a local feedback loop to generate online trajectories. Finally, they combine different degrees of freedom to ensure synchronized movements.

Advantages of using oscillators in central pattern generator modeling include the following:

1. Existence of stable limit cycles: A stable limit cycle is a distinct cycle that attracts all neighboring trajectories. The stable limit cycles are important because they model oscillatory generator systems. In other words, these systems oscillate even in the absence of external periodic forces. There are numerous examples of such systems, including heartbeats and the triggering of the pacemaker device.
2. Limit cycle oscillator: Limit cycle oscillators are a good model for describing the central pattern generator.
3. Synchronization property: This allows easy coordination between different degrees of freedom of the robot. Because only simple parameters such as frequency, amplitude, and phase difference between oscillators are needed to generate high-dimensional control policies, controllers based on these coupled oscillators reduce the complexity of the control problem.

Discrete and rhythmic motions are considered separately in motor control theory. In our research, all movements are modeled by a combination of discrete and rhythmic motion primitives. This dynamic system is known as a unit pattern generator.

2.4 Unit pattern generator

All trajectories are obtained from a set of unique differential equations, which are called unit pattern generators. Each unit pattern generator can be divided into two subsystems: rhythmic

and discrete. The first subsystem is responsible for producing short and goal-directed movements, and the other is responsible for producing rhythmic movements [44].

To design a special system that can produce rhythmic and discrete movements, the dynamics of both systems were combined with each other to obtain a limit cycle that can be moved in the x-direction. This was obtained by considering the discrete movement output y_i, as an offset of the rhythmic output x_i:

$$\dot{h_i} = 1 - h_i \tag{1}$$

$$\dot{y_i} = v_i \tag{2}$$

$$\dot{v_i} = \frac{1}{4}B^2{h_i}^2(y_i - \gamma_i) - Bh_iv_i \tag{3}$$

$$\dot{m_i} = C(\mu_i - m_i) \tag{4}$$

$$\dot{x_i} = \frac{A}{|\mu_i|}(m_i - {r_i}^2)(x_i - y_i) - \omega_i z_i + \epsilon \tag{5}$$

$$\dot{z_i} = \frac{A}{|\mu_i|}(m_i - {r_i}^2)z_i - \omega_i(x_i - y_i) + \epsilon \tag{6}$$

where x_i is the output of the system, z_i and m_i are auxiliary variables, and $r_i = \sqrt{({x_i}^2 - {y_i}^2) + {z_i}^2}$. A, B, and C are constants that control the system convergence. When $\mu_i > 0$, Equations 5 and 6 show a Hopf oscillator. Equations 1–6 represent a unit pattern generator that minimum number of equations are needed for control of one degree of freedom.

3 Proposed method

The learning system proposed in this research has two layers to teach complex behaviors to humanoid robots. The first layer detects and records the teacher's movements using a Kinect sensor and obtains the joint patterns through a geometric procedure. These trajectories are trained in the second layer to produce a complete set of movements, such as drumming or playing tennis.

3.1 First layer modeling

The goal of this layer is to obtain joint trajectories to teach the humanoid robot the desired hand movements. These trajectories were obtained from the Kinect sensor. Kinect cannot only provide a natural interaction interface and generate a 3D model of objects but also track the skeleton in 3D space. Simultaneously, it can also separate the specific character from complex circumstances. We can obtain the $\mathbf{x} - \mathbf{y} - \mathbf{z}$ coordinates of the point of interest through the Kinect sensor. They are used to define gesture modules. From the OpenNI tracker, body coordinates can be extracted with 20 points corresponding to 20 joints of the human body with respect to the world coordinate system.

We considered the arm and forearm as two different vectors. θ, the desired angle between arm and forearm, was calculated as

$$\overrightarrow{v_1}.\overrightarrow{v_2} = |v_1||v_2|\cos(\theta) \tag{7}$$

$$\theta = \cos^{-1}\left(\frac{\overrightarrow{v_1}.\overrightarrow{v_2}}{|v_1||v_2|}\right) \tag{8}$$

Regarding our humanoid robot (Bioloid), one degree of freedom in the elbow and two degrees of freedom in the shoulder were considered. Each degree of freedom was considered in 2D space. Algorithm 1 is used to calculate the elbow joint angle.

Algorithm 1 Computing elbow joint angle

Input : Three spatial sequence point
Output : Joint Angle
for $j \in [1, FrameNumber]$ **do**
  P = joint coordinate $(n-1:j)$ - joint coordinate $(n:j)$
  Q = joint coordinate $(n+2:j)$ - joint coordinate $(n:j)$
  $|P| = \sqrt{{P_x}^2 + {P_y}^2}$
  $|Q| = \sqrt{{Q_y}^2 + {Q_z}^2}$
  $\theta = \frac{PQ'}{|P|.|Q|+\epsilon}$
end for
$\theta = cos^{-1}(\theta)$

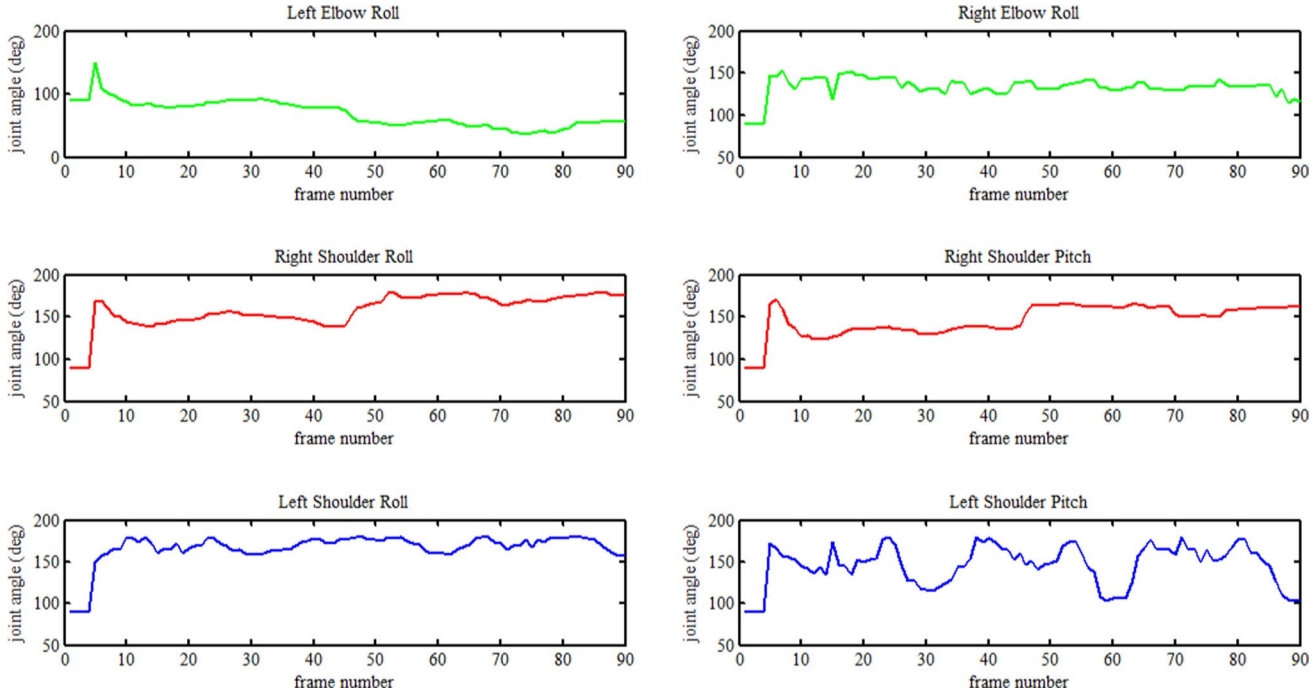

Fig. 4 Kinect output, elbow, and shoulder joint angles

The elbow and shoulder angle trajectories are shown in Fig. 4.

3.2 Second layer modeling

The purpose of this layer is to train the data obtained from the first layer. Unit pattern generators were used to train each motion trajectory. Then, the trained signals were optimized by an evolutionary algorithm to reduce differences between experimental and trained signals. The information produced by the first layer was sent to the second layer. All oscillators received the same input signal, and each oscillator adapted its signal to one of the harmonics of the input signal. With this method, the system does not learn properly for any initial values that were selected as oscillator parameters. Oscillator parameters can control the phase, frequency, and amplitude of the desired signals. Therefore, we used evolutionary optimization algorithms to select the best oscillator parameters.

The idea of collective intelligence combined with nonlinear dynamical systems is a way to overcome current challenges. There are many algorithms in the field of collective intelligence. In this study, the particle swarm optimization algorithm (PSO), imperialistic competitive algorithm (ICA), and artificial bee colony algorithm (ABC) were used to solve this problem. We used these three evolutionary algorithms to fine-tune the best oscillator parameters.

The process of obtaining the desired signals is shown in Algorithm 2. As shown, the data obtained from Kinect were sent to MATLAB, and the joint angles were calculated by the geometric method. Then, special patterns were generated by Hopf oscillators. These patterns were trained by unit pattern generators to converge with the experimental signals. For this purpose, the oscillator parameters were estimated using an evolutionary algorithm.

Algorithm 2 Learning algorithm for natural learner unit pattern generator neural networks

Input: Kinect patterns which are computed by geometric methods.
Output: optimized patterns of unit pattern generators
Initialize the Neural Network to Generate rhythmic patterns by UPGs
Apply the condition $\mu > 0$ in UPG
Detract the 1th UPG output from Kinect output
Extract the bias: $bias \leftarrow mean(y_{teach})$
$y_{teach} \leftarrow y_{teach} - bias$
Train a UPG using Natural Policy Gradient Algorithm
for $i \in [2, neuronnumber]$ **do**
    Detract the ith UPG output from i-1th UPG output
end for
Sum all UPGs.
Test UPG network.
Compute the coupling between all the UPGs:
end;

3.3 Natural policy gradient computation

In this section, we describe the natural gradient calculation used in the Algorithm 2. This method is described in detail in [45–47]. The algorithm starts from an initial point determined by θ. (The θ is also determined by UPG parameters in the Algorithm 2 in each iteration.) In the first step, the algorithm sets the initial values of the neuron θ to θ_0 and samples a state-action-reward sequence of length H. Here states are the input time of the neuron and actions are the generated output of the UPG in each time (y_{out}). The reward function is the difference between the desired output and the actual output. In the next step, the algorithm computes some statistical matrices to obtain the

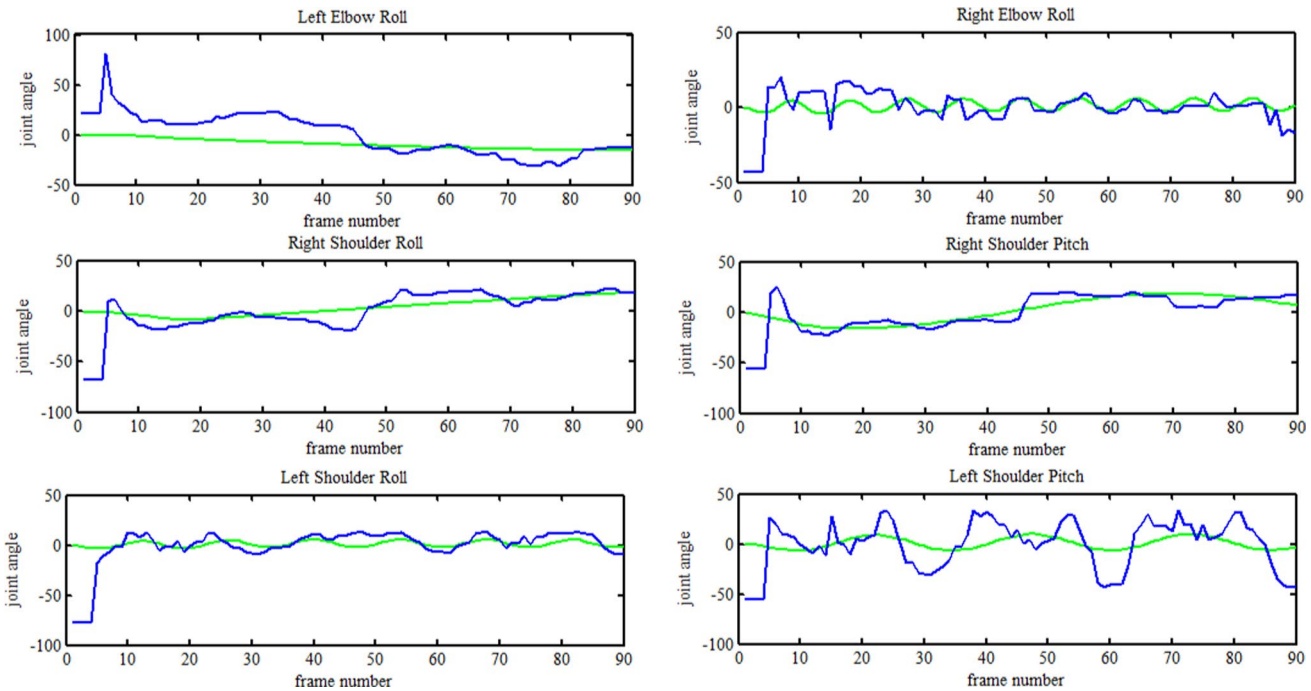

Fig. 5 Optimized learned patterns by NLUPGNN, experimental patterns (blue) (color figure online)

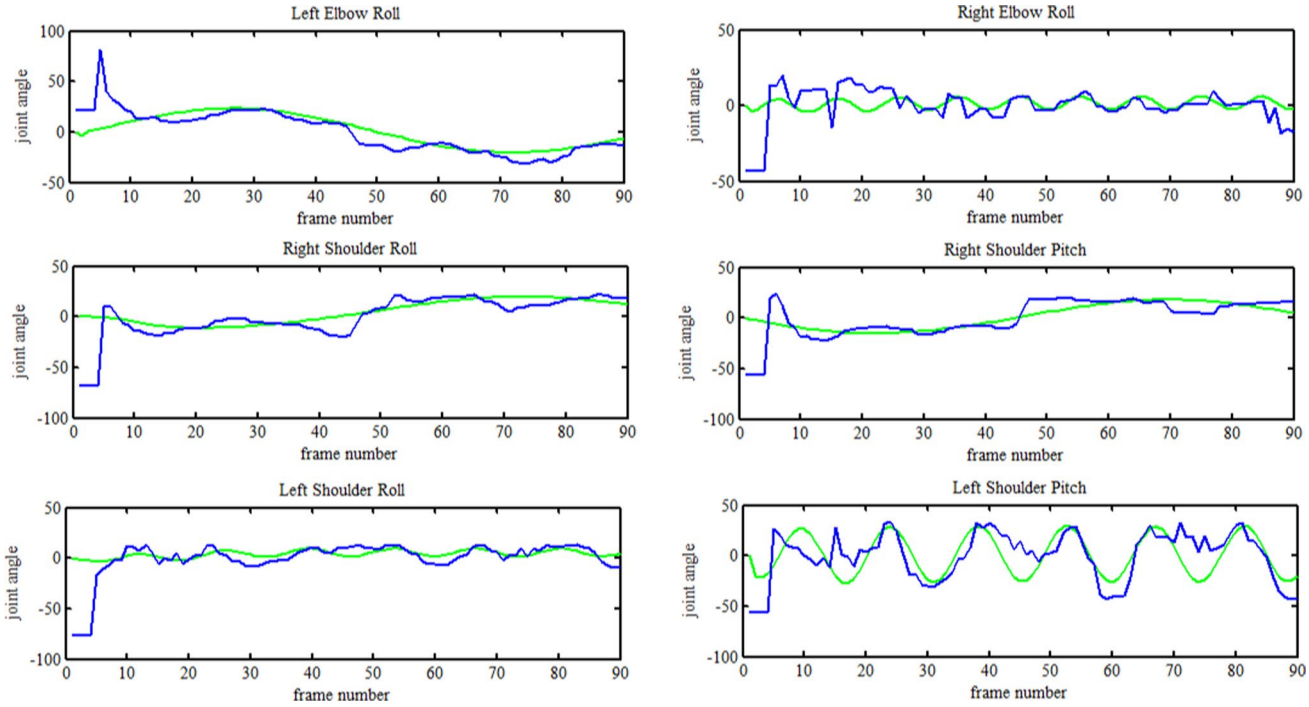

Fig. 6 Optimized learned patterns by ICA algorithm (green), experimental patterns (blue) (color figure online)

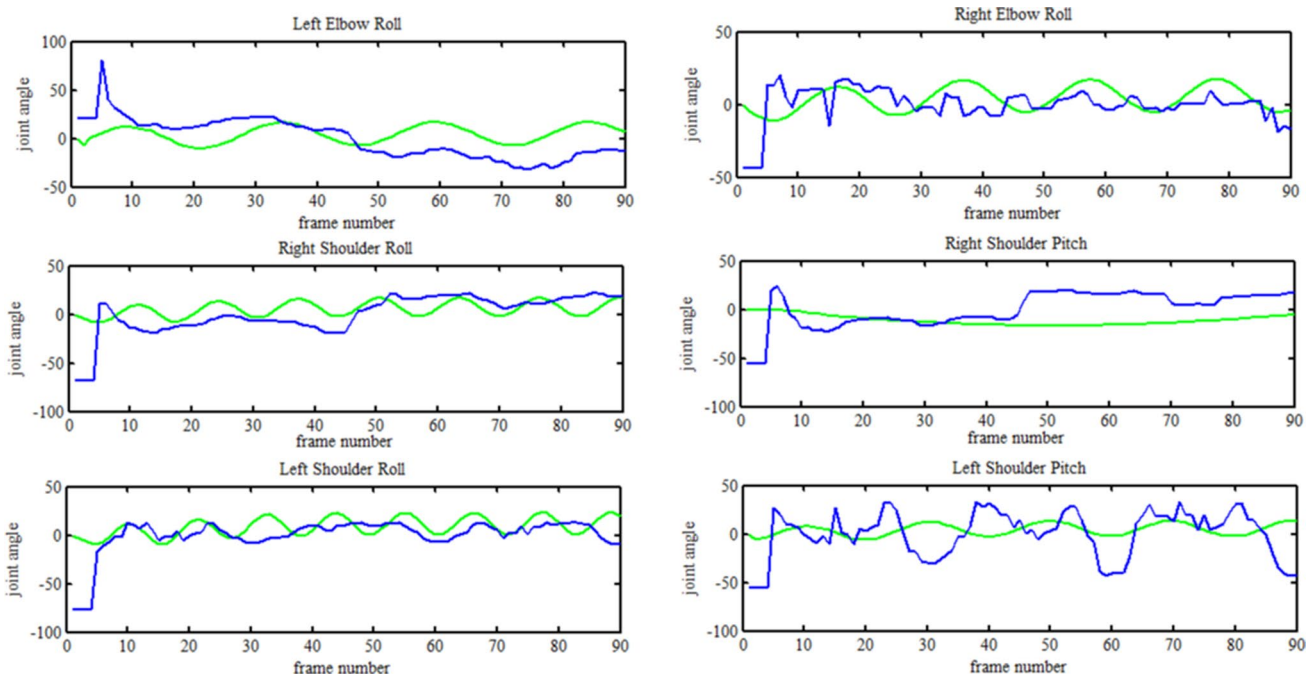

Fig. 7 Optimized learned patterns by ABC algorithm (green), experimental patterns (blue) (color figure online)

natural gradient. These are the policy derivative matrix, the eligibility matrix, the average reward matrix, the Fisher matrix, and the vanilla gradient matrix. In the last step, the algorithm computes the baseline b and the natural gradient g_{NG} using these statistical matrices.

Algorithm 3 Natural policy gradient computation

repeat
  sample $t^{0:H}, y_{out}^{0:H}, dy_{out}^{0:H}$
  Set $x \leftarrow t, u \leftarrow y_{out}, r \leftarrow (y_{teach} - y_{out})^2$.
  Obtain the sufficient statistics:
    Policy derivatives: $\psi_k \leftarrow \nabla_\theta log(\pi_\theta(x_k, u_k))$
    Eligibility: $\phi \leftarrow E\left\{(\sum_{k=0}^{H} \psi_k)\right\}$
    Average reward: $\overline{r} \leftarrow (\sum_{k=0}^{H} r_l)^T$
  Fisher Matrix: $F_\theta \leftarrow E\left\{(\sum_{k=0}^{H} \psi_k)(\sum_{l=0}^{H} \psi_l)^T\right\}$
  Vanilla Gradient: $g \leftarrow E\left\{(\sum_{k=0}^{H} \psi_k)(\sum_{l=0}^{H} r_l)^T\right\}$
  Obtain Natural Gradient by computing:
    $Q \leftarrow (1 + \phi^T(F_\theta - \phi\phi^T)^{-1}\phi)$
    Baseline $b \leftarrow Q(\overline{r} - \phi^T F_\theta^{-1} g)$
    Natural Gradient $g_{NG} \leftarrow F_\theta^{-1}(g - \phi b)$
until gradient estimate g_{NG} converged
Output g_{NG}
end;

Optimized learned patterns are shown in Figs. 5, 6, and 7.

4 Comparative evaluation of machine learning methods

In this study, we conducted a detailed comparative analysis of three advanced machine learning methods: natural learner unit pattern generator neural networks (NLUPGNN), unit pattern generator neural networks with imperial competition algorithm (UPGNN+ICA), and unit pattern generator neural networks with honey bee optimization (UPGNN+HBO). The evaluation criteria included the average rate of convergence, the average percentage of test errors, and the training time required for each method.

Table 1 presents a comprehensive comparison of the three distinct methodologies based on several critical performance metrics: the average rate of convergence, the average percentage of test errors, and the overall training duration. The average rate of convergence is quantified as the proportion of motion patterns that were effectively assimilated by the respective method. This metric serves as an indicator of the method's efficiency in adapting to new data. The average percentage of test errors is calculated as the mean squared deviation between the instructor's demonstrated movements and the movements replicated by the algorithm. This measure reflects the precision with which the method can mimic the desired motion patterns. Lastly, the training time denotes the temporal span necessary to fine-tune the system's parameters to an optimal state. For each evaluated method, the established boundaries for these metrics are set at an upper limit of 30 and a lower threshold of 0.3.

Table 1 Performance metrics of machine learning methods

Method	Convergence rate (%)	Average test error (%)	Training time (min)
NLUPGNN	93	4.4	13.3
UPGNN+ICA	89	10.31	5
UPGNN+HBO	78	32.2	45

The NLUPGNN method demonstrated a commendable average convergence rate of 93%, signifying a high success rate in training motion patterns. The average test error was recorded at a low 4.4%, indicating a strong alignment with the teacher's movements. The training time was relatively efficient, taking only 13.3 min to optimize the system parameters. Conversely, the UPGNN+ICA method, while achieving a slightly lower convergence rate of 89%, exhibited a higher average test error of 10.31%. However,

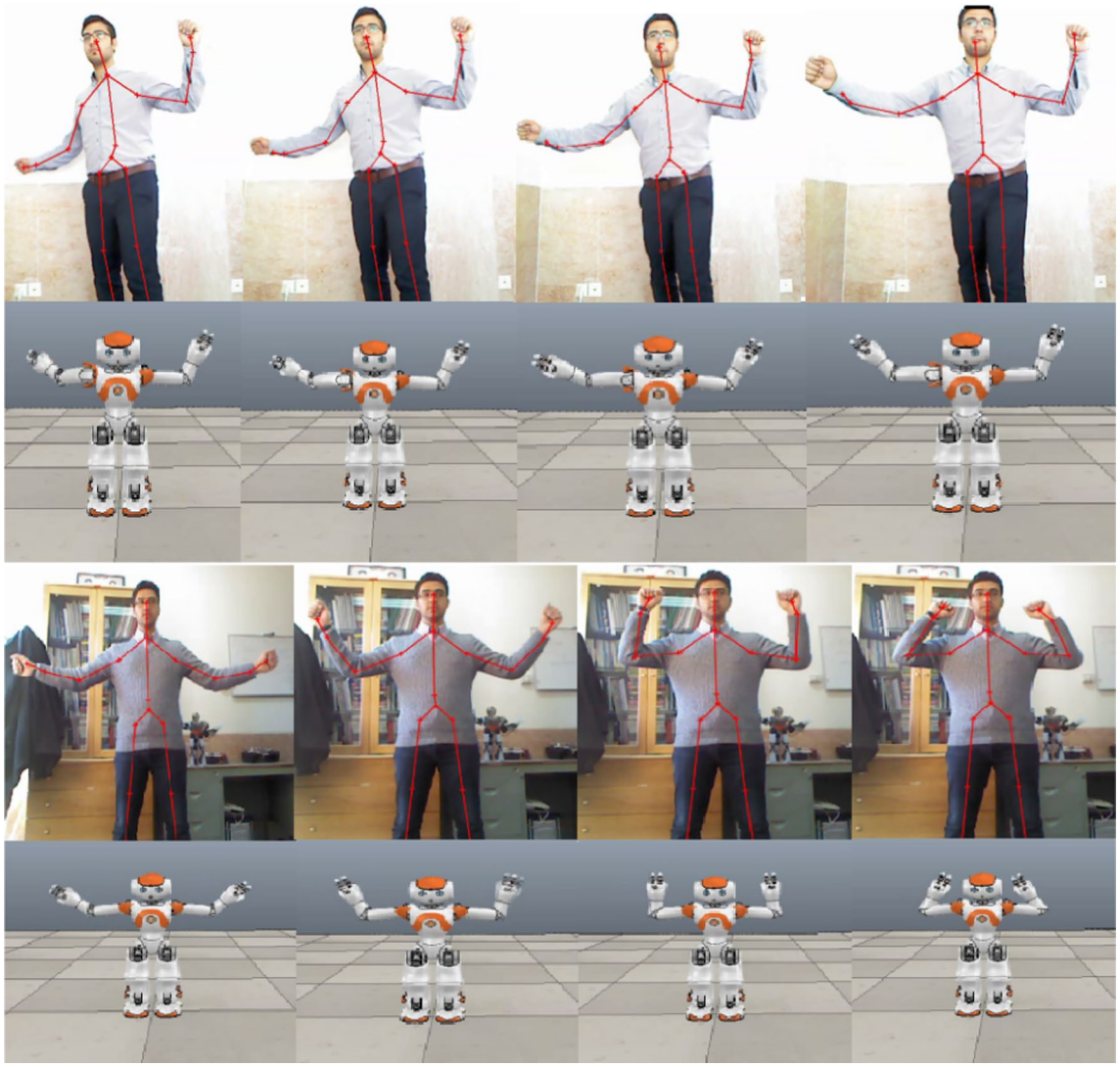

Fig. 8 System simulation in V-rep, Nao robot imitates human body movement

it excelled in training efficiency, requiring a mere 5 min to reach convergence.

The UPGNN+HBO method lagged behind with a convergence rate of 78% and the highest average test error of 32.2%. Additionally, it required a considerably longer training time of 45 min, suggesting a need for further refinement. The following table encapsulates the comparative results of the three methods:

This study underscores the importance of balancing accuracy and efficiency in machine learning algorithms, particularly for applications requiring real-time processing. Future research will aim to enhance the convergence rates and reduce test errors while minimizing training times.

4.1 System properties

The suggested system mentioned above has some features that are described in this section.

4.2 Resistance against perturbation

All trajectories generated by UPGs converge to the limit cycle for all initial conditions. Thus, the system will return to the limit cycle in the presence of short-term perturbations. This feature can be used to modulate the dynamics of the system according to feedback information.

4.3 Dynamics modulation

The dynamic parameters of the system can be controlled by its control parameters such as amplitude and frequency. The amplitude of the oscillation is directly controlled by the parameter μ and the frequency can be modulated by μ and the frequency can be modulated by ω. In some special applications, it is desirable to change the dynamic features at different times. This feature can be used to make the movement smoother and more complex.

5 Hardware and software

This study used a variety of software and hardware to train a humanoid robot to perform specific movements. Microsoft Kinect was used to detect body gestures and joint trajectories. These trajectories were analyzed using MATLAB, and the desired joint angles were obtained. The RGB-D Kinect sensor's integration into human–robot interaction systems is pivotal due to its ability to capture both color (RGB) and depth (D) data, providing a comprehensive understanding of human gestures in a three-dimensional space. The Kinect sensor's capabilities include:

- *Depth Capture:* It measures the distance between objects and the sensor, offering a 3D representation of the environment.
- *Human Skeleton Tracking:* The sensor identifies and tracks multiple points on the human body, capturing the posture and movements accurately.
- *Natural User Interface (NUI):* Kinect allows for intuitive interactions with robots, interpreting human gestures without the need for additional devices.

This technology enhances human–robot interaction by translating human gestures into digital signals that the robot's system can process, enabling the robot to mimic human movements seamlessly.

5.1 V-REP simulation

The virtual robot experimentation platform (V-REP) is an open source robotics software that can be used for robot simulation and testing of production lines. The V-REP robot simulator is based on a distributed control architecture. V-REP approaches make it very versatile and ideal for multi-robot applications [48]. In our study, MATLAB was connected to V-REP through an API interface. Remote API interface in V-REP interacts with V-REP or other simulation software. It comprises remote API server services and remote API clients. The client side can be embedded as footprint code (e.g., C/C + +, Python, Java, MATLAB, Urbi) in virtually any hardware, including factory lines, people, furniture, and real mobile or non-mobile robots. Then, it allows remote function calling [49]. The remote API interface provides a situation in which a user can control the robot in the simulator through other software. In our research, the desired signals were sent to the Nao robot using MATLAB. For this purpose, after inserting the humanoid robot, the inverse kinematics group should be disabled and the joints should be set to the torque/force mode. Figure 8 shows an example simulation of the robot for an arbitrary hand movement.

5.2 Robot implementation

We have implemented the trained patterns on a Bioloid humanoid robot. Bioloid is one of the products of the ROBOTIS company. This robot was powered by 12 servo motors known as Dynamixel. Dynamixel is a special servo motor constructed from a DC motor, gearbox, controller, driver, and some sensors. This motor can be connected by a TTL or RS485 network connection. In this project, we used an Arduino instead of the motor controller to send data to the real robot. These servo motors have some difficulties in connecting to Arduino. Here it is explained how the servo motor, plastic shell, circuit controller, battery systems, and sensors contribute to the overall operation of the puppetry system:

- Servo Motor: The servo motor is the powerhouse of the puppetry system, providing precise control of angular or linear position, speed, and torque. It operates based on a feedback loop system, which ensures accurate positioning and movement according to the commands received[1]. The servo motor's functionality is crucial for the nuanced

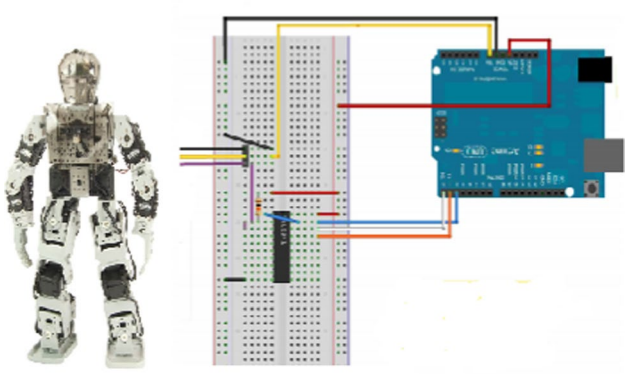

Fig. 9 Schematic of the intended interface that used for connecting Arduino to the humanoid robot

Fig. 10 Implementation of hand motion on a Bioloid robot

movements of the puppet, allowing for lifelike gestures and expressions.

- Plastic Shell: The plastic shell serves as the protective casing for the internal components of the puppet. It not only safeguards the electronics from external elements but also provides a structure to which the servo motors and other parts are attached. This shell can be designed to resemble various characters or objects, contributing to the visual appeal of the puppet.
- Circuit Controller: The circuit controller acts as the brain of the system. It interprets the input signals and translates them into actions by sending the appropriate commands to the servo motors. This component is essential for coordinating the movements of the puppet, ensuring that the performance is synchronized and fluid.
- Battery Systems: Battery systems supply the necessary power to all electronic components of the puppetry system. They are designed to provide a stable and reliable source of energy, enabling the puppet to perform for

extended periods without the need for constant recharging or power supply connections.
- Sensors: Sensors in the puppetry system are akin to the senses in a living organism. They detect external stimuli, such as sound, light, or touch, and send this information to the circuit controller. The controller then adjusts the puppet's movements in real time, allowing for interactive performances that can respond to the audience or environment.

Together, these components work in harmony to create a puppetry system that is capable of delivering complex and engaging performances. The servo motors provide the movement, the plastic shell offers form and protection, the circuit controller ensures intelligent operation, the battery systems supply power, and the sensors bring interactivity and responsiveness to the puppet's actions.

Because we have multiple servos to connect and the Dynamixel protocol is a serial one, the half-duplex communication to 1 Mbps requires additional circuitry. Therefore, the tristate buffer has to be connected to the serial port between the Arduino and the dynamite. We used a generic 12 V adapter as the power source. According to the documentation available from ROBOTIS, the most appropriate voltage for adjusting motors is between 9 and 12 V. Therefore, we placed a diode on the breadboard. Figure 9 shows a schematic of the circuit used to connect Arduino to the Bioloid humanoid robot. Figure 10 shows examples of the implementation of trained patterns on a real Bioloid robot.

6 Conclusion

In the rapidly advancing field of robotics and artificial intelligence, the development of interactive games that incorporate robotic learning and imitation tasks represents a significant leap forward. The state-of-the-art approach detailed in this study involves the creation of a novel game based on the natural learner unit pattern generator neural networks (NLUPGNN). This innovative technique stands out by generating motion trajectories through imitation learning, a method that allows robots to learn and replicate complex actions by observing human players.

The paper addresses the intricate design of these neural networks, which is a critical challenge in the field. By dividing the unit pattern generators into two subsystems—a rhythmic system and a discrete system—the study presents a structured approach to motion generation. The introduction of a specialized learning algorithm tailored for these unit pattern generators further enhances the system's capability.

A key aspect of this research is the interconnection and coupling of the unit pattern generators to form a cohesive network. The unknown parameters within this network are adeptly determined using a natural policy gradient learning algorithm. This process not only trains nonlinear oscillators but also enables the precise reproduction of motion sequences for a humanoid robot. Consequently, the robot's joints can mimic the movements of a human teacher in real time, showcasing a remarkable level of interaction and learning fidelity.

The main contribution of this work lies in the novel learning algorithm that concurrently optimizes the weights and topology of the neural network. This algorithm ensures synchronized learning steps by effectively coupling the neurons, thereby facilitating a harmonious and efficient learning process. Such advancements not only push the boundaries of robotic capabilities but also open new avenues for human–robot interaction, making this research a pivotal reference point for future innovations in the field.

The results of this study can be used to train certain behaviors in children with autism spectrum disorder, to control robots in hazardous environments, and to use the imitation robot as a toy or advertisement in shopping centers.

References

1. Zheng B et al (2022) Imitation learning: progress, taxonomies and challenges. IEEE Transactions on Neural Networks and Learning Systems
2. Argall BD, Chernova S, Veloso M, Browning B (2009) A survey of robot learning from demonstration. Robot Auton Syst 57(5):469–483
3. INOUE TIMIH, Inamura M, Inaba H (1999) Acquisition of probabilistic behavior decision model based on the interactive teaching method. In: Proceedings of the ninth international conference on advanced robotics ICAR99
4. Clouse JA (1996) On integrating apprentice learning and reinforcement learning. University of Massachusetts Amherst
5. Rao RP, Shon AP, Meltzoff AN (2004) A Bayesian model of imitation in infants and robots. Imitation and social learning in robots, humans, and animals, pp 217–247
6. Asfour T, Azad P, Gyarfas F, Dillmann R (2008) Imitation learning of dual-arm manipulation tasks in humanoid robots. Int J Humanoid Rob 5(02):183–202
7. Herzog D, Ude A, KrUger V (2008) Motion imitation and recognition using parametric hidden markov models. In: Humanoids 2008-8th IEEE-RAS international conference on humanoid robots. IEEE, pp 339–346
8. Ijspeert AJ, Nakanishi J, Schaal S (2002) Movement imitation with nonlinear dynamical systems in humanoid robots. In: IEEE international conference on robotics and automation, 2002. Proceedings. ICRA'02. vol 2. IEEE, pp 1398–1403
9. Schaal S, Peters J, Nakanishi J, Ijspeert A (2005) Learning movement primitives. In: Robotics research. The eleventh international symposium. Springer, pp 561–572
10. Ijspeert AJ (2008) Central pattern generators for locomotion control in animals and robots: a review. Neural Netw 21(4):642–653
11. Kress-Gazit H, Lahijanian M, Raman V (2018) Synthesis for robots: guarantees and feedback for robot behavior. Ann Rev Control Robot Auton Syst 1:211–236

12. Shahbazi H, Jamshidi K, Monadjemi AH, Eslami H (2014) Biologically inspired layered learning in humanoid robots. Knowl-Based Syst 57:8–27
13. Asada H, Izumi H (1989) Automatic program generation from teaching data for the hybrid control of robots. IEEE Trans Robot Autom 5(2):166–173
14. Das N, Prakash R, Behera L (2016) Learning object manipulation from demonstration through vision for the 7-dof barrett wam. In: 2016 IEEE first international conference on control, measurement and instrumentation (CMI). IEEE, pp 391–396
15. Schaal S (1999) Is imitation learning the route to humanoid robots? Trends Cognit Sci 3(6):233–242
16. Berthouze L, Bakker P, Kuniyoshi Y (1996) Learning of oculomotor control: a prelude to robotic imitation. In: International conference on intelligent robots and systems' 96, IROS 96, proceedings of the 1996 IEEE/RSJ, vol 1. IEEE, pp 376–381
17. Atkeson CG, Schaal S (1997) Learning tasks from a single demonstration. In: 1997 IEEE international conference on robotics and automation, 1997. Proceedings., vol 2.IEEE, pp 1706–1712
18. Albrektsen SM (2011) Using the Kinect Sensor for Social Robotics. MS thesis. Institutt for teknisk kybernetikk
19. Góngora Alonso S, Hamrioui S, de la Torre Díez I, Motta Cruz E, López-Coronado M, Franco M (2019) Social robots for people with aging and dementia: a systematic review of literature. Telemedicine and e-Health, vol 25, no 7, pp 533–540
20. Di Palo N, Johns E (2022) Learning multi-stage tasks with one demonstration via self-replay. In: Conference on robot learning. PMLR, pp 1180–1189
21. Correia A, Alexandre LA (2023) A survey of demonstration learning. arXiv:2303.11191
22. Chen G et al (2022) Humanoid Robot Portrait Drawing Based on Deep Learning Techniques and Efficient Path Planning. Arab J Sci Eng 47(8):9459-9470
23. Wei L, Wang Z, Jia W, Yuan J, Ma S, Li L (2019) Practical vision-based walking navigation for the humanoid robot nao in the maze-like environment. In: 2019 IEEE international conference on robotics and biomimetics (ROBIO). IEEE, pp 2743–2748
24. Do HM, Sheng W, Harrington EE, Bishop AJ (2020) Clinical screening interview using a social robot for geriatric care. IEEE Trans Autom Sci Eng 18(3):1229–1242
25. Shahbazi H, Jamshidi K, Hasan Monadjemi A (2012) Modeling of mesencephalic locomotor region for nao humanoid robot. Ind Robot Int J 39(2):136–145
26. Shahbazi H, Jamshidi K, Monadjemi AH, (2012) Curvilinear bipedal walk learning in nao humanoid robot using a cpg based policy gradient method. In: Applied mechanics and materials, vol 110. Trans Tech Publ, pp 5161–5166
27. Shahbazi H, Jamshidi K, Monadjemi AH (2013) Sensor-based programming of central pattern generators in humanoid robots. Int J Adv Robot Syst 10:192
28. Shahbazi H, Jamshidi K, Monadjemi AH, Manoochehri HE (2015) Training oscillatory neural networks using natural gradient particle swarm optimization. Robotica 33(07):1551–1567
29. Ruchanurucks M, Nakaoka S, Kudoh S, Ikeuchi K (2006) Humanoid robot motion generation with sequential physical constraints. In: Proceedings 2006 IEEE international conference on robotics and automation, ICRA 2006. IEEE, pp 2649–2654
30. Rozo L (2013) Robot learning from demonstration of force-based manipulation tasks
31. Moeslund TB, Hilton A, Krüger V (2006) A survey of advances in vision-based human motion capture and analysis. Comput Vis Image Underst 104(2–3):90–126
32. Akgun B, Cakmak M, Yoo JW, Thomaz AL (2012) Trajectories and keyframes for kinesthetic teaching: a human-robot interaction perspective. In: Proceedings of the seventh annual ACM/IEEE international conference on human-robot interaction, pp 391–398
33. Racinskis P, Arents J, Greitans M (2022) A motion capture and imitation learning based approach to robot control. Appl Sci 12(14):7186
34. Siciliano B, Khatib O, Kröger T (2008) Springer handbook of robotics, vol 200. Springer, Berlin
35. Kim J, Cauli N, Vicente P, Damas B, Cavallo F, Santos-Victor J (2018) icub, clean the table! a robot learning from demonstration approach using deep neural networks. In: 2018 IEEE international conference on autonomous robot systems and competitions (ICARSC). IEEE, pp 3–9
36. Ding I-J, Chang C-W (2016) An adaptive hidden Markov model-based gesture recognition approach using kinect to simplify large-scale video data processing for humanoid robot imitation. Multimed Tools Appl 75:15537–15551
37. Bica I, Jarrett D, van der Schaar M (2021) Invariant causal imitation learning for generalizable policies. Adv Neural Inf Process Syst 34:3952–3964
38. Kidambi R, Chang J, Sun W (2021) Mobile: model-based imitation learning from observation alone. Adv Neural Inf Process Syst 34:28598–28611
39. Wang Y, Beltran-Hernandez CC, Wan W, Harada K (2021) Robotic imitation of human assembly skills using hybrid trajectory and force learning. In: 2021 IEEE international conference on robotics and automation (ICRA). IEEE, pp 11 278–11 284
40. Barlow SM, Lund JP, Estep M, Kolta A (2010) Central pattern generators for orofacial movements and speech. Handb Behav Neurosci 19:351–369
41. Iosa M, Gizzi L, Tamburella F, Dominici N (2015) Editorial: neuro-motor control and feed-forward models of locomotion in humans. Front Hum Neurosci 9:306
42. Righetti L, Ijspeert AJ (2006) Programmable central pattern generators: an application to biped locomotion control. In: Proceedings 2006 IEEE international conference on robotics and automation. ICRA 2006. IEEE, pp 1585–1590
43. Degallier S, Righetti L, Gay S, Ijspeert A (2011) Toward simple control for complex, autonomous robotic applications: combining discrete and rhythmic motor primitives. Auton Robot 31(2–3):155–181
44. Morse G, Risi S, Snyder CR, Stanley KO (2013) Single-unit pattern generators for quadruped locomotion. In: Proceedings of the 15th annual conference on Genetic and evolutionary computation, pp 719–726
45. Peters J, Schaal S (2008) Natural actor-critic. Neurocomputing 71(7):1180–1190
46. Peters J, Schaal S (2006) Policy gradient methods for robotics. In: 2006 IEEE/RSJ international conference on intelligent robots and systems. IEEE, pp 2219–2225
47. Bhatnagar S, Sutton R, Ghavamzadeh M, Lee M (2009) Natural actor-critic algorithms. Automatica 45(11):2471–2482
48. Ma H, Wang H, Fu M, Yang C (2015) One new human-robot cooperation method based on kinect sensor and visual-servoing. In: International conference on intelligent robotics and applications. Springer, pp 523–534
49. Rohmer E, Singh SP, Freese M (2013) V-rep: a versatile and scalable robot simulation framework. In: 2013 IEEE/RSJ international conference on intelligent robots and systems. IEEE, pp 1321–1326