Heuristic and deep reinforcement learning-based PID control of trajectory tracking in a ball-and-plate system

Emmanuel Okafor, Daniel Udekwe, Yusuf Ibrahim, Muhammed Bashir Mu’azu and Ekene Gabriel Okafor

Abstract
The manual tuning of controller parameters, for example, tuning proportional integral derivative (PID) gains often relies on tedious human engineering. To curb the aforementioned problem, we propose an artificial intelligence-based deep reinforcement learning (RL) PID controller (three variants) compared with genetic algorithm-based PID (GA-PID) and classical PID; a total of five controllers were simulated for controlling and trajectory tracking of the ball dynamics in a linearized ball-and-plate (B&P) system. For the experiments, we trained novel variants of deep RL-PID built from a customized deep deterministic policy gradient (DDPG) agent (by modifying the neural network architecture), resulting in two new RL agents (DDPG-FC-350-R-PID & DDPG-FC-350-E-PID). Each of the agents interacts with the environment through a policy and a learning algorithm to produce a set of actions (optimal PID gains). Additionally, we evaluated the five controllers to assess which method provides the best performance metrics in the context of the minimum index in predictive errors, steady-state-error, peak overshoot, and time-responses. The results show that our proposed architecture (DDPG-FC-350-E-PID) yielded the best performance and surpasses all other approaches on most of the evaluation metric indices. Furthermore, an appropriate training of an artificial intelligence-based controller can aid to obtain the best path tracking.

1. Introduction
The Ball-and-Plate (B&P) system can be described as an enhanced version of a Ball-and-Beam system, whereby the ball positioning is controlled in dual directions (Mohajerin et al., 2010). The B&P system finds practical application in several dynamic systems, such as robotics, rocket systems, and unmanned aerial vehicles. The enumerated systems are often expected to follow a time parameterized reference path. Most of the laboratory-based B&P (Dong et al., 2011) systems are inherently non-linear and unstable (Kassem et al., 2015) due to irregular swings of the plate and the ball positioning. Hence results in a B&P system trajectory tracking
problem, which is a classical control challenge that put forward a premise that allows the testing of different control algorithms or techniques. It is important to design a control mechanism that has the prowess to guarantee stability and able to track a predefined trajectory path of the ball dynamics. The broad focus of this paper is to analyze two controllers (artificial intelligence and classical based PIDs) for controlling the trajectory or reference tracking of the ball positioning in a \( B&P \) system.

Some research has investigated the use of classical control techniques to deal with the trajectory tracking and point stabilization problems of the \( B&P \) system. The research paper by Galvan-Colmenares et al. (2014) demonstrated the use of Proportional Derivative (PD) controller which factored non-linear compensation, that is capable of controlling the ball position in dual axes. The works of (Hussien, Yousif et al., 2017; Kasula et al., 2018) investigated the PID controller analyzed on an Arduino implementation of a \( B&P \) system while providing a solution to the trajectory tracking and point stabilization problems. The study reported by Oravec and Jadlovská (2015) designed a model predictive control (MPC) controller for validating and verifying the reference trajectory tracking problems in a \( B&P \) laboratory model. The study by Umar et al. (2018) investigated the applicability of the use of H-

infinity control method for controlling and tracking ball positioning in a \( B&P \) system. Moreover, an interesting study by Beckerleg and Hogg (2016) investigated the implementation of a motion-based controller which considered an evolutionary inspired genetic algorithm (GA) when creating lookup tables that co-adapt to fault tolerance on a ball and plate system. There exist other forms of heuristic approaches integrated to classical controllers based on an expectation to either yield optimal stability or track the ball dynamics of a \( B&P \) system. Previous works of (Hussein, Muhammed et al., 2017) demonstrated that a weighted Artificial Fish Swarm Algorithm PID controller has the potential to tune desirable control parameters while providing trajectory tracking of the \( B&P \) system in a double feedback loop structure. Furthermore, Roy et al. (2015) constructed a Particle Swarm Optimization for tuning the PD controller gain parameters and hence was used to control trajectory tracking of the \( B&P \) system. The paper written by Dong et al. (2011) employed GA for optimizing a Fuzzy Logic Control (FLC) parameter and the resulting optimal model was used for trajectory tracking of a \( B&P \) system.

Other research works have extensively explored different variants of FLC. For example, the work of Lin et al. (2014) implemented a fuzzy logic control mechanism that controls and stabilizes the ball and plate system based on the position obtained using the charge-coupled device (CCD) mounted on \( B&P \) system. Other forms of research improvement on \( B&P \) system using expert controllers include fuzzy sliding mode controller (Negash & Singh, 2015), comparison between sliding mode control and FLC methods (Kasula et al., 2018). The researchers reported the use of dual-level FLCs (Rastin et al., 2013) to actualize obstacle avoidance and trajectory tracking.

Artificial intelligence-based controllers have also been used to control the dynamics of a \( B&P \) system. The research investigation by Zhao and Ge (2014) developed a neural network-based fuzzy multi-variable learning strategy for controlling a \( B&P \) system. Their study employed an objective cost function (optimal control index function) and then the fuzzy neural network controller parameters were computed following an offline gradient learning scheme. Mohammadi and Ryu (2020) proposed a neural network-based PID compensator for the control of a nonlinear model of the \( B&P \) system. This system consists of dual controllers which work in parallel: a base linear controller and a multi-layer perceptron based PID
The concept of deep reinforcement learning (Deep-RL) was earlier proposed by (Mnih et al.): this machine learning technique was extensively reviewed by these authors (Arulkumaran et al., 2017; François-Lavet et al., 2018) and has found practical application in gaming (Ansó et al., 2019; Mnih et al.). Based on the success attributed to Deep-RL, the current study seeks to examine the feasibility of applying this promising AI learning paradigm to control system problems.

1.1. Contribution

The motivation of this paper stems from the need to shift from classical and heuristic computational control schemes to adaptive control methods (deep reinforcement learning for controlling a classical control system). To the best of our knowledge, this is the first time a deep reinforcement learning would be used as a controller to control the trajectory tracking of ball positioning in a B&P system. To actualize the stated goal, we modeled a non-linear scenario of a B&P system, then we linearized the stated system about some operating conditions, to obtain a linear system. Then, we compared five controllers: classical PID, GA-PID, and three variants of deep reinforcement learning-based PID (DDPG-PID, DDPG-FC-350-R-PID & DDPG-FC-350-E-PID). Our proposed controllers involve a modification from an existing deep deterministic policy gradient, (DDPG) agent in the context of the network nodal structure and uniformity in activation function. This results in a proposal of two novel agents whose central task is to create optimal PID gains. Overall, a total of five controllers were used for controlling and tracking the ball position in a B&P system. We demonstrated that the training of artificial intelligence-based method (our proposed DDPG-FC-350-E-PID) yielded a performance that surpasses both heuristic and classical PIDs on most of the evaluation metrics while providing minimized index scores. This method obtained the best and effective tracking of the ball dynamic paths in a B&P system when compared with the other approaches. Each of the investigated controllers provides a closed loop stability.

1.2. Paper outline

The remaining components of the paper are outlined as follows: Section 2 comprehensively discussed the modeling and stability of the B&P system for the linear and non-linear scenario. Section 3 discusses the different PID controllers (classical PID, Deep-RL-PID, GA-PID). The result findings were discussed in Section 4. The conclusion and recommendation for future works are highlighted in Section 5.

2. System modelling

This section entails the modeling of both non-linear and linear derivation of the B&P system. Additionally, the stability of the linear B&P system was also examined.

2.1. Non-linear derivation of the B&P system

The dynamic modeling of a 2 degree of freedom (2-DOF) B&P system as shown in Figure 1 and can be modeled using the principle of Euler Lagrange (Kassem et al., 2015; Spaček, 2016).

\[
Q_k = \frac{d}{dt} \left[ \frac{\partial T}{\partial \dot{q}_k} \right] - \frac{\partial T}{\partial q_k} + \frac{\partial V}{\partial q_k}
\]  

(1)
The effective force $Q_k$ computes a partial derivative of the total kinetic energy $T$ and potential energy $V$ with respect to a generalized direction coordinate $q_k$. The described coordinates consist of 2 ball position coordinates $\{x_b, y_b\}$ and 2 tilted plate angles $\{\alpha, \beta\}$. The total kinetic energy $T$ is the summation of the kinetic energy of the ball $T_b$ and the kinetic energy of the plate $T_p$. $T$ is mathematically defined as:

$$T = T_b + T_p$$

(2)

$T_b$ computes the summation of rotational and translational energy components, which can be defined as:

$$T_b = \frac{1}{2} \left( m_b + \frac{l_b}{r_b^2} \right) (\dot{x}_b^2 + \dot{y}_b^2)$$

(3)

$T_p$ factors all the moment of inertia (for the plate $I_p$, and the ball $I_b$) as well as the plates rotational velocity $\{\dot{\alpha}, \dot{\beta}\}$ (Nokhbeh et al., 2011), given $m_b$ as the ball mass.

$$T_p = \frac{1}{2} (I_p + I_b) (\dot{\alpha}^2 + \dot{\beta}^2) + \frac{1}{2} m_b (\dot{\alpha}^2 x_b^2 + 2 \dot{\alpha} \dot{\beta} x_b y_b + \dot{\beta}^2 y_b^2)$$

(4)

$$T = \frac{1}{2} \left( m_b + \frac{l_b}{r_b^2} \right) (\dot{x}_b^2 + \dot{y}_b^2) + \frac{1}{2} (I_p + I_b) (\dot{\alpha}^2 + \dot{\beta}^2) + \frac{1}{2} m_b (\dot{x}_b^2 \dot{\alpha}^2 + 2 x_b \dot{\alpha} \dot{\beta} x_b \dot{\alpha} + y_b^2 \dot{\beta}^2)$$

(5)
We compute the potential energy of the ball $V$ based on the horizontal center inclination with the plate can be expressed as (Nokhbeh et al., 2011):

$$V = m_b g (x_b \sin \alpha + x_b \sin \beta) \quad (6)$$

After solving the Euler Lagrange from equation 1, the condensed nonlinear differential equation for the ball and plate system according to Spaček (2016) and Nokhbeh et al. (2011) can be defined as:

$$\mathbf{x}: \left( m_b + \frac{l_b}{r_b^2} \right) \ddot{x}_b - m_b \left( x_b \dot{\alpha}^2 + y_b \dot{\alpha} \dot{\beta} \right) + m_b g \sin \alpha = 0 \quad (7)$$

$$\mathbf{y}: \left( m_b + \frac{l_b}{r_b^2} \right) \ddot{y}_b - m_b \left( y_b \dot{\beta}^2 + y_b \dot{\alpha} \dot{\beta} \right) + m_b g \sin \alpha = 0 \quad (8)$$

$$\dot{\alpha}; u_\alpha = \left( I_p + l_b + m_b x_b^2 \right) \ddot{\alpha} + 2m_b x_b \dot{x}_b \dot{\alpha} + m_b x_b y_b \dot{\beta} + m_b \dot{x}_b \dot{y}_b \dot{\beta} + m_b g x_b \cos \alpha \quad (9)$$

$$\dot{\beta}; u_\beta = \left( I_p + l_b + m_b y_b^2 \right) \ddot{\beta} + 2m_b y_b \dot{y}_b \dot{\beta} + m_b y_b x_b \dot{\alpha} + m_b y_b \dot{x}_b \dot{\alpha} + m_b x_b y_b \dot{\alpha} + m_b g y_b \cos \beta \quad (10)$$

The variables $u_\alpha$ and $u_\beta$ denote the motor torques in the $\alpha$ and $\beta$ direction, respectively.

### 2.2. Linearized derivation of the ball-and-Plate system

Since the used motor does not lose any step or experience performance variations, $\beta$ and $\alpha$ angles can be treated as direct system input; the described reason explains why Equations (9) and (10) were omitted. Let us assume the ball moment of inertia is modeled as $I_b = \frac{2}{5} m_b r_b^2$, by substituting $I_b$ into Equations (7) and (8). A simplified non-linear representation of the system can be defined as:

$$m_b \left[ \frac{7}{5} \ddot{x}_b - \left( x_b \dot{\alpha}^2 + y_b \dot{\alpha} \dot{\beta} \right) + g \sin \alpha \right] = 0 \quad (11)$$

$$m_b \left[ \frac{7}{5} \ddot{y}_b - \left( y_b \dot{\beta}^2 + x_b \dot{\alpha} \dot{\beta} \right) + g \sin \beta \right] = 0 \quad (12)$$

To actualize linearization of the $B&P$ system, we made the following assumption via some operating points. We assumed the angular velocities $\dot{\alpha}$ and $\dot{\beta}$ are very small, and exhibits negligible effects when squared or multiplied together ($\dot{\alpha} \dot{\beta} \approx 0$, $\dot{\alpha}^2 \approx 0$, $\dot{\beta}^2 \approx 0$). Note that if $\alpha$ and $\beta$ are very small, it is assumed that $\alpha = \sin(\alpha)$ and $\beta = \sin(\beta)$. The simplified $B&P$ linear system can be defined as:

$$\frac{7}{5} \ddot{x}_b + g \alpha = 0 \quad (13)$$

$$\frac{7}{5} \ddot{y}_b + g \beta = 0 \quad (14)$$

We operated on Equations (13) and (14) using a Laplace Transform operator; the system
represent the frequency domain representation of the B&P system.

\[ \mathcal{L}\left( \frac{7}{5} \ddot{x}_b + g \alpha \right) = \frac{7}{5} s^2 x_b(s) + g \alpha(s) = 0 \] (15)

\[ \mathcal{L}\left( \frac{7}{5} \ddot{y}_b + g \beta \right) = \frac{7}{5} s^2 y_b(s) + g \beta(s) = 0 \] (16)

We assumed that \( \alpha(s) \) and \( \beta(s) \) are treated as inputs to the B&P system, the ratio of the output and the input results in the following linearized transfer functions.

\[ \frac{x_b(s)}{\alpha(s)} = -\frac{5g}{7s^2} \] (17)

\[ \frac{y_b(s)}{\beta(s)} = -\frac{5g}{7s^2} \] (18)

Based on the symmetrical nature of the system, we considered one ball coordinate and a plate angle (Spaček, 2016). We treated our motor as a first-order system \( G_m(s) \). The gain compensator \( K_m = -0.6854 \) and time constant \( T_m = 0.187 \).

\[ G_m(s) = \frac{K_m}{T_m s + 1} \] (19)

We cascaded the motor system and the B&P system to produce the plant transfer function similar to the function reported by Spaček (2016):

\[ G_p(s) = \frac{K_m}{T_m s + 1} \times \frac{-5g}{7s^2} = \frac{4.803}{0.187s^3 + s^2} \] (20)

It is important to note that during the linearization of non-linear B&P system to a linear system, we experienced a negligible information loss due to the assumption and approximation we made about some operating conditions which may not arise in a non-linear system.

### 2.3. Stability and control solution test

We employed the root-locus technique to describe the open-loop system stability as shown in Figure 2; a system stability can possibly exist as asymptotic stable, unstable, or marginal stability. Since the open-loop system poles are \( \leq 0 \) with repetitive 0; this depicts that the system is marginally stable (neither stable nor unstable). An analysis was carried out to determine the control solution; we report that the rank operated on the controllability and observability matrices were equal to \( N=3 \) states. This test indicates that the system is controllable and observable (there exists a control solution).

### 3. Controller design

PID controller is a universal controller, that finds application in various industrial control systems. The basic importance of this controller is associated with its operational and functional simplicity. In many control analyses, the PID controller could be referred to
as a compensator due to its ability to correct errors in a given control system (Paz, 2001). A typical controller architecture is shown in Figure 3.

PID controllers can be represented mathematically as

\[
u(t) = K_p e(t) + K_i \int_0^t e(t) dt + K_d \frac{de(t)}{dt}\]

PID controllers have three parameters: proportional gain \((K_p)\), integral gain \((K_i)\) and derivative gain \((K_d)\). \(K_p\) is dependent on the present error, \(K_i\) on the accumulation of past errors, and \(K_d\) on the prediction of future errors (Paz, 2001). The \(u(t)\) accounts for the control efforts or commands. Due to the preliminary observation in the plant marginal stability, this necessitated an exploration on several variants of a PID controllers to guarantee the plant stability as well as the system to yield a desired trajectory path. Hence in the next section, we discuss three forms of PID controllers.

Figure 3. A block diagram of a closed-loop control system consisting of a PID controller and the system model containing the plant \(G_p(s)\).
3.1. Classical PID

The classical or traditional PID was implemented using the Ziegler-Nichols (ZN) tuning strategy. The ZN heuristic tunes the PID gain parameters to steady state oscillations by progressively increasing the proportional gain $K_p$ starting from zero, derivative gain $K_d$ and integral gains $K_i$ are set to zero. This ultimate gain $K_u$ often considered as the largest gain at the instance when the control loops attain stable state, regular oscillations, and a period $T_u$ were used to determine the values of $K_p$, $K_i$ and $K_d$ (Meshram & Kanojiya, 2012). An often generalized PID gains inspired by ZN can be expressed as: $K_p = 0.6K_u$, $K_i = 1.2K_u/T_u$ and $K_d = 3K_uT_u/40$. The classical Ziegler Nichols parameters $K_u$ and $T_u$ were tuned in the bound: $K_u = \{0.1, 1.0\}$, $T_u = \{20, 30\}$. The best found classical PID gain parameters were summarized in Table 1.

3.2. Genetic algorithm based PID

Genetic algorithm (GA) is a heuristic search and optimization technique guided by the concepts of genetics and natural selection (Krishnakumar & Goldberg, 1992; Meena & Devanshu, 2017). GA finds practical application in the following domains: multi-vehicle task assignment in a drift field (Bai et al., 2018), path planning (Nazarahari et al., 2019), energy management of hybrid electric vehicle (Lü et al., 2020). In this study, the objective function is the PID command function which is dependent on the predictive error and three gain parameters $\{K_p, K_i, K_d\}$. To determine the optimal PID control gains, we employed the following genetic algorithm steps:

- Initialization of population: this step involves the collection of individual sets, also known as population; each individual denotes a solution to a problem. An individual is described by a set of parameters known as genes. Genes merge strings to form a chromosome (solution). The parameters to be optimized are defined within upper and lower bounds, $\{-5, 5\}$. In the experiment, the initial population was set to 15.
- Fitness function: the second step aids to determine how fit can an individual compete with other individuals, based on some prediction criteria. The fitness score is often used when training a GA model on train data. In the experiment, we used the integral time absolute error (ITAE) function for predicting fitness individuals.
- Selection: the third step entails selecting the fittest individuals and allows the transfer of genes from one generation to the next. Parents (fit individuals) with high fitness scores are selected for mating. We employed a stochastic uniform approach for selecting fit individuals.

Table 1. The optimal PID gains for each of the controllers.

| Controller             | $K_p$   | $K_i$    | $K_d$   |
|------------------------|---------|----------|---------|
| Classical-PID (ZN)     | 0.1360  | 0.0150   | 0.5430  |
| GA-PID                 | 3.1194  | 0.3728   | 4.9965  |
| DDPG-PID               | -0.7241 | 0.3327   | -0.8864 |
| DDPG-FC-350-R-PID     | 0.7546  | -0.0029  | 1.1991  |
| DDPG-FC-350-E-PID     | 4.2025  | 1.2529   | 5.1323  |
• Crossover: the fourth step requires parents for mating, a crossover-point is chosen at random from within the genes of the parents. This process yields offsprings. In the experiment, we considered an intermediate crossover function.
• Mutation: the last step involves a scenario where certain genes in an offspring undergo mutation based on low random probability. The essence of mutation is to guarantee diversity in the population pool and prevent poor convergence.

The five steps were repeated iteratively for 30 generations until convergence is actualized on the minimal fitness score to obtain the optimal PID gains.

3.3. Deep RL-based PID

Deep reinforcement learning (Arulkumaran et al., 2017; François-Lavet et al., 2018; Mnih et al.) is a machine learning problem that involves the interaction between an agent and the environment, by creating an optimal policy with long-term rewards. To create an intelligent controller (RL-based PID), we built several agents that act as controllers for generating optimal PID gains. In the next section, we explain the different deep RL-based PID controllers based on the original DDPG agent and our proposed agents.

3.3.1. DDPG-PID

The deep deterministic policy gradient (DDPG) agent as reported in the link¹ as explained in the water-tank example was used for computing the optimal action (gain parameters). We refer to the PID inspired by DDPG (DDPG-PID). Hence the steps required for the creation of this Deep-RL-PID controller are enumerated below:

(1) Formulation of the Problem: we defined the task for the agent to learn to determine optimal PID gains by creating a scenario that allows the agent to interacts with the environment to maximize predefined reward conditions. The resulting gains from the agent and the continuous predictive errors were used to produce a PID command that controls the trajectory tracking of the plant system.

(2) Environment: it can be described as all components in an RL network without the inclusion of the agent. Examples of the environment components include: dynamic plant model (system containing B&P and the motor) that returns a controlled trajectory signals through a feedback mechanism, the observation blocks, reward generating block, and a stopping conditions. Note that the observation block receives an a continuous error and process it into a discrete error vector: $S = (e(t) \int e(t)dt (de(t)/dt))$, which corresponds to the error, integral error, and derivative error. The $e$ is the predictive error, $e(t) = y_r - y_i$, $y_r$ is the reference point signal and $y_i$ is the controlled signal (value function of the target value). The aforementioned errors can be described as the observation states. The interaction of the agent and the environment generates a cumulative reward which is described as:

$$y_i = R_i + \gamma Q^{t+1}(S_{i}^{t+1}, \mu^{t+1}(S_{i}^{t+1}/\theta_\mu)/\theta_Q)$$  \hspace{1cm} (22)$$

where $y_i$ is the effective sum of experience reward and discounted reward for future observations $S_{i}^{t+1}$ with an actor parameter $\mu^{t+1}(S_{i}^{t+1}/\theta_\mu)/\theta_Q^{t+1})$. The $R_i$ variable
denotes the experience reward, $\gamma$ represents the discount factor, $\theta_{Q^t+1}$ accounts for the weighted or randomized parameters.

(3) Reward: the agent uses the reward signal to evaluate the performance measures compared to a reference goal. The reward is a signal often computed from the environment. Our novel experience reward function of the used RL algorithm is a quadratic cost function defined as:

$$R_i = 10 - (e(t)^2 + u(t)^2 + 10(OR(y_i \geq 1.6, y_i \leq -1)))$$

(23)

If the OR logic is satisfied, then it returns a value of 1, otherwise, the OR logic operator returns a 0.

(4) Create Agent: most RL agents rely on two main parts: the first part defines a policy representation and the second part configures the agent learning algorithm that iteratively updates itself. We trained a deep deterministic policy gradient (DDPG) agent in the following fashions. The DDPG agent estimates both the policy and the value function, the examined agent considers 4 function approximators:

- The actor $\mu(S)$ receives a set of observations, operates on it, and yields an action vector $A = \{K_p, K_i, K_d\}$ that maximizes the long-term reward. The actor contains three network layers: one hidden fully-connected layer (which contains three nodes operated by a hyperbolic tangent (tanh) activation function) and the remaining layers: either the input or output layer contains an equal number of nodes as there are in the observations. Here fully-connected layer can be defined as:

$$H^l = \sum_{i=1}^k w_i^l x_{i,l-1} + b_j^l$$

(24)

The output from the hidden layer $H^l$ is the sum of the weighted input ($w_i^l x_{i,l-1}$ from a previous layer $l-1$) and the hidden layer bias $b_j^l$.

- The target actor $\mu^{t+1}(S)$ often improve optimization stability, the agent periodically learns via an episodic updates the $\mu^{t+1}(S)$ often depends on past actor $\mu^t(S)$ parameter values.

- The critic $Q(S, A)$: the critic receives inputs ($S, A$) and outputs a long-term reward expectation. The critic component of the agent consists of a state path, action path, and common path, whose hidden networks contain varying node values that lie between $\{1, 50\}$. Moreover, the activation function used within the critic is ReLU. It is a rectifying unit which can be used to operate on $H^l$, to produce informative features $f_o = \max(0, H^l)$. $f_o$ represents activation output at the hidden layer.

- The target critic $Q^{t+1}(S, A)$: is often used to improve optimization stability, the agent involves a periodic update of the $Q^{t+1}(S, A)$ based on the previous critic parameter $Q^t(S, A)$ values. The update in the critic parameters can be actualized by minimizing a loss function $L$ across m number of sample experiences.

$$L = \frac{1}{m} \sum_{i=1}^m (y_i - Q^t(S_i, A_i/\theta^t_o))^2$$

(25)

We further updated the actor parameters of the agent by sampling the policy gradient to
maximize an expected discount reward; this is done using the expression below:

\[
\nabla \theta_\mu = \frac{1}{m} \sum_{i=1}^{m} \nabla A Q(S^i, A/\theta_\mu) \cdot \nabla \theta_\mu(S^i/\theta_\mu)
\]

(26)

(5) We trained the agent policy representation using the described environment, reward, and agent learning algorithm. The experimental settings employed during the training of the agent are described herein. We employed the MATLAB Simulink computing environment for training the DDPG agent.

### 3.3.2. Proposed methods

We proposed some modifications in the original DDPG agent by using a uniform nodal distribution within the hidden layers of both critic and actor components. For this, we specified 350 network nodes, we considered two forms of activation functions: ReLU and exponential linear unit (ELU). Each of these activation functions is used throughout a given architecture. The summary of the PID controller inspired by the proposed DDPG agents are described below:

- **DDPG-FC-350-R-PID**: we refer to DDPG that considers a ReLU (denotes R) with 350 neural network nodes in each of the hidden layers in either actor or critic as DDPG-FC-350.
- **DDPG-FC-350-E-PID**: we refer to DDPG that considers an ELU (denotes E) with 350 neural network nodes in each of the hidden layers in either actor or critic as DDPG-FC-350. An ELU (Clevert et al., 2015) can be defined mathematically as:

\[
f_\alpha(H) = \begin{cases} H & \text{if } H > 0 \\ \alpha(\exp(H) - 1) & \text{if } H \leq 0 \end{cases}
\]

(27)

Note that the actor components in the new agents contain two hidden layers as against the original DDPG agent that contains only one hidden layer.

The ELU parameter \( \alpha > 0 \) controls and penalizes the hidden layer output unit when it yields a negative output. An illustration of the Deep-RL-PID connected to the plant system is shown in Figure 4.

**Experimental Parameters**: The experiments were conducted on Hp Laptop with RAM size of 6GB. We trained each of the agents for a maximum episode of 114, both the maximum and the average number of steps was set 40. We report that the training time of the agent is \( t \leq 420.8 \) s. The learning rate for the actor and critic is \( 1 \times 10^{-5} \) and \( 1 \times 10^{-4} \) respectively. The window length for averaging is 5. The choice of the stated hyper-parameters were motivated from the original DDPG agent. However, we set other values in the training configuration settings (for example: maximum number of the episodic steps). The learning curve during the training of one of the agents is shown in Figure 5. The summary of the optimal PID control gains for each of the described methods is presented in Table 1.
4. Result analysis and discussion

This section explains the trajectory tracking and experimental results analysis of the described controllers.

4.1. Result analysis

In the preliminary experiments, we compared the proposed controllers with the original DDPG agents. The results of our findings were reported in Table 2. The definitions of the performance metrics are described herein. The Table presents simulations of the Deep-RL controllers on the B&P system while taking a reference input (a unit step signal); the original DDPG-PID yielded very high error metrics, high settling time and low peak overshoot. The poor performance could be attributed due to the usage of a limited amount of nodal neurons within the hidden layers of the network architecture. However, our proposed architecture showed significantly lower predictive error metrics and time responses. Based on the latter observation necessitated the comparison with both GA-PID and Classical-PID. In Figure 6, we show a comparative analysis of four controllers. The results show that our proposed Deep-RL agents (DDPG-FC-350-R-PID & DDPG-FC-350-E-PID) exhibited a shorter settling time than GA-PID and Classical-PID. The summary of the controller performances was reported in Table 3.

From the table, we observe that the DDPG-FC-350-E-PID and DDPG-FC-350-R-PID outperform GA-PID in 6/7 and 5/7 evaluation metrics, respectively. While the least mean steady-state error (MSSE) was obtained by DDPG-FC-350-E-PID, this performance depicts a high degree of correlation to the reference step response. Overall, our proposed reinforcement learning agents based on PID and heuristic method (GA-PID) significantly outperform the classical PID on all the evaluation metrics. We draw an inference from the table that, it is important to train computational or artificial intelligent algorithms to obtain minimum index time response and minimal predictive errors.

Figure 4. A close-loop control system consists of a deep reinforcement learning PID controller and a system model containing the plant $G_p(s)$. 
4.1.1. Trajectory tracking

This involves constraining an object to follow a predefined path. It represents how a control law can trail the desired path in the presence of unaccounted forces that stand to derail the system. We investigated how the equations described below can be used to generate the reference trajectory for the trajectory tracking problem. The ball is made to follow a circular trajectory path (with a radius 0.4 m) at an angular speed of 0.52 rad/s. Note that the reference input in both x and y directions are passed as input to the PID cascaded with the earlier described plant. All the results reported in this section were based on simulations carried out using Matlab Simulink. An illustration of the controller to follow the predefined path is shown in Figure 7.

\[ x = 0.4 \cos (2 - \omega t), \quad y = 0.4 \sin (\omega t) \]  

From the plots, our proposed method (DDPG-FC-350-E-PID) shows a high-level of path following of the ball dynamics than heuristic or classical PIDs. Moreover, that of the GA-PID demonstrated a medium-level of path trailing, since there are few instances of outlier trajectory follow. The worst technique is the classical PID showed a poor overlap and more concentric

Table 2. Performance metrics comparison between three Deep-RL PID controllers analyzed on a B&P system.

| RL Method            | \( t_r \) (s) | \( t_s \) (s) | \( M_p \)     | MAE            | IAE            | ISE            | MSSE            |
|----------------------|---------------|---------------|--------------|----------------|----------------|----------------|----------------|
| DDPG-PID             | **0.7279**    | **59.9932**   | **0.0000**   | \( 2.0345 \times 10^{-76} \) | \( 8.6469 \times 10^{-77} \) | \( 1.1591 \times 10^{-156} \) | \( 2.0345 \times 10^{-76} \) |
| DDPG-FC-350-R-PID    | 3.0440        | **5.6865**    | 0.1170       | 0.0683         | 1.9845         | 0.8941         | 0.0683         |
| DDPG-FC-350-E-PID    | 1.5393        | 10.3473       | 17.7960      | **0.0581**     | **1.7701**     | **0.6373**     | **0.0256**     |
outliers. Figure 8 shows the root-locus plot used for verifying system stability. We report that the investigated controllers when cascaded with the plant, the closed-loop transfer function returns poles which are on the left-hand part (LHP) of the S-plane, which indicate that the controllers aided in establishing the ball and plate system stability.

4.2. Discussion

The performance of a control system depends on the ability of the system to achieve the desired dynamic response, which is dependent on the tuning of the controller. The classical Ziegler Nichols PID tuning strategy relies on intuitive guess of the ideal control parameters with a drawback of a tedious trial and error process which often results in non-optimal result performances. However, the heuristic inspired (genetic algorithm) and proposed deep reinforcement learning PID controllers provide optimal performances compared with the classical PID controller; the success could have arisen due to the selection of the desired parameters from the bound of the predefined range of values, through an iterative search space. The superiority of our proposed deep reinforcement learning controllers can be ascribed due to the extensive and exhaustive search for the optimal PID gain parameters, using neural network inspired function approximations and an updated learning policy based on a reward experience scheme actualized during agent interaction with the environment.

Figure 6. Closed loop response of the PID controllers on Ball-and-Plate system for a unit step signal.

Table 3. Performance metrics comparison between the three main kinds of PID controllers analyzed on a B&P system.

| Controllers         | $t_r$ (s) | $t_s$ (s) | $M_p$   | MAE    | IAE    | ISE    | MSSE   |
|---------------------|-----------|-----------|---------|--------|--------|--------|--------|
| DDPG-FC-350-E-PID   | 1.5393    | 10.3473   | 17.7960 | 0.0581 | 1.7701 | 0.6373 | 0.0256 |
| DDPG-FC-350-R-PID   | 3.0440    | 5.6865    | 0.1170  | 0.0683 | 1.9845 | 0.8941 | 0.0683 |
| GA-PID              | 2.4006    | 20.0818   | 11.4046 | 0.1050 | 2.4731 | 0.8250 | 0.0463 |
| Classical PID       | 4.4451    | 30.31642  | 21.4427 | 0.1968 | 5.6635 | 2.2882 | 0.0665 |
5. Conclusion

We have demonstrated comparative analysis on five forms of PID controllers; Classical PID, Genetic Algorithm PID, and three variants of deep RL-PID (original DDPG agent and two reinforcement learning agents were proposed by us). Most of the methods were used to verify system stability, control, and trajectory tracking of the ball dynamics in a B&P system while computing the time responses and predictive errors. The study showed that the use of our proposed reinforcement learning agent (DDPG-FC-350-E-PID) obtained the best optimal gains, and yielded the best performance in the trajectory tracking, error metrics, and some instances of the time responses when compared with other approaches. However, we report that the original DDPG agent yielded poor performances since it was not originally designed for a B&P system, rather on a process control problem (water tank). Hence original DDPG agent architecture was not suitable for the B&P system.

The novelty from this study is that we have investigated the possibility of creating two intelligent adaptive controllers inspired by the reinforcement learning paradigm. Our proposed controllers involve a modification from an existing deep deterministic policy gradient (DDPG) agent in the context of the network nodal structure and uniformity in the activation function. The comparative analysis suggests that artificial intelligence adaptive
and heuristic controllers significantly outperform the classical PID controller in the context of the error metrics and time response performance. The success of the proposed method may be attributed to the appropriate selection of deep RL architecture.

Future research can be directed to the possibility of hybridizing heuristic and artificial intelligent-based agents that can be analyzed on an industrial control problem.

Notes
1. https://www.mathworks.com/help/reinforcement-learning/ug/train-reinforcement-learning-agents.html.
2. https://www.mathworks.com/help/reinforcement-learning/ug/train-reinforcement-learning-agents.html.
3. Tables 2 and 3 present the performance evaluation of the proposed and other methods in the context of the following performance metrics: $t_r$ defines the time taken for the closed loop response to rise from 0% to 100% of the final value. $t_s$ represents the time taken for the closed loop response of the system to stay within 2%—5% of final value. $M_p$ represents the normalized difference between the peak value of the response and the steady state output. MAE, IAE, ISE and MSSE denote the mean absolute error, the integral absolute error, the integral square error, and mean steady state error, respectively.

Acknowledgments
This work acknowledges the Department of Computer Engineering, at the Ahmadu Bello University, Zaria, Nigeria, for their support on this research.

Disclosure statement
No potential conflict of interest was reported by the author(s).

Notes on contributors
Emmanuel Okafor earned a Ph.D. degree in Artificial Intelligence from the University of Groningen, the Netherlands, in 2019. Dr. Okafor is a lecturer in the Department of Computer Engineering, Ahmadu Bello University (ABU), Zaria, Nigeria. His main research interests include computer vision, deep learning, control systems, reinforcement learning, robotics, and optimization.

Daniel Udekwe is an M.Sc. candidate of Control Engineering in the Ahmadu Bello University, Nigeria. Mr. Udekwe is a Graduate Assistant in the Department of Aerospace Engineering, Air Force Institute of Technology, Nigeria. His main research interests include control systems, artificial intelligence, and robotics.

Yusuf Ibrahim obtained his bachelor’s degree in Electrical Engineering from Ahmadu Bello University, Zaria, Nigeria in 2011. After a compulsory one-year national youth service, he joined the services of Ahmadu Bello University, Zaria in the year 2013 as an assistant lecturer in the Department of Electrical and Computer Engineering. He then further received his Master’s and doctorate degrees in Computer Engineering in the years 2017 and 2019, respectively, from the same institution. His research interests
include Digital Image Processing, Machine Learning, and Computer Vision. Yusuf is a registered member of the Council for the Regulation of Engineering in Nigeria (COREN).

Muhammed Bashir Mu’azu earned both a Ph.D. degree in Electrical Engineering and other degrees from ABU, Zaria, Nigeria. He is a full professor in the field of computational intelligence in the Department of Computer Engineering and a former director of IAIICT, from ABU, Zaria. His research interests include control system, optimization, telecommunication, artificial intelligence, and robotics.

Ekene Gabriel Okafor obtained a Ph.D. degree in Vehicle Operation Engineering from the Nanjing University of Aeronautics and Astronautics, Nanjing, China in 2012. Dr Okafor is an Associate Professor at the Air Force Institute of Technology. Dr Okafor has won several national and international research grants. His main research interests include system safety analysis, reliability analysis, and design, maintenance planning and modeling, aviation management, risk assessment, and optimization.

ORCID

Emmanuel Okafor http://orcid.org/0000-0001-6929-6880

References

Ansó N. S., Wiehe A. O., Drugan M. M., & Wiering M. A. (2019). Deep reinforcement learning for pellet eating in agar. IO. In ICAART (2) (pp. 123–133).

Arulkumaran K., Deisenroth M. P., Brundage M., & Bharath A. A. (2017). Deep reinforcement learning: A brief survey. IEEE Signal Processing Magazine, 34(6), 26–38. https://doi.org/10.1109/MSP.2017.2743240

Bai X., Yan W., S. S. Ge, & Cao M. (2018). An integrated multi-population genetic algorithm for multi-vehicle task assignment in a drift field. Information Sciences, 453, 227–238. https://doi.org/10.1016/j.ins.2018.04.044

Beckerleg M., & Hogg R. (2016). Evolving a lookup table based motion controller for a ball-plate system with fault tolerant capabilities. In 2016 IEEE 14th International Workshop on Advanced Motion Control (AMC) (pp. 257–262). IEEE.

Clevert D.-A., Unterthiner T., & Hochreiter S. (2015). Fast and accurate deep network learning by exponential linear units (elus). arXiv preprint arXiv:1511.07289 (pp. 1–15).

Dong X., Zhao Y., Xu Y., Zhang Z., & Shi P. (2011). Design of pso fuzzy neural network control for ball and plate system. International Journal of Innovative Computing, Information and Control, 7(12), 7091–7103. http://www.ijicic.org/ijicic-10-11107.pdf

François-Lavet V., Henderson P., Islam R., Bellemare M. G., & Pineau J. (2018). An introduction to deep reinforcement learning. arXiv preprint arXiv:1811.12560 (pp. 24–102).

Galvan-Colmenares S., Moreno-Armendáriz M. A., Rubio J. d. J., Ortíz-Rodríguez F., Yu W., & Aguilar-Ibáñez C. F. (2014). Dual pd control regulation with nonlinear compensation for a ball and plate system. Mathematical Problems in Engineering, 2014, 1–10. https://doi.org/10.1155/2014/894209

Hussien M. B. A., Yousif M. G. E. M., Altoum A. O. H. A., & Abdullaheem M. F. A. (2017). Design and implementation of ball and plate control system using pid controller [PhD thesis]. Sudan University of Science and Technology.

Hussein S., Muhammed B., Sikiru T., Umoh I., & Salawudeen A. (2017). Trajectory tracking control of ball on plate system using weighted artificial fish swarm algorithm based pid. In 2017 IEEE 3rd international conference on electro-technology for national development (NIGERCON) (pp. 554–561). IEEE.
Kassem A., Haddad H., & Albitar C. (2015). Comparison between different methods of control of ball and plate system with 6dof Stewart platform. IFAC-PapersOnLine, 48(11), 47–52. https://doi.org/10.1016/j.ifacol.2015.09.158

Kasula A., Thakur P., & Menon M. K. (2018). GUI based control scheme for ball-on-plate system using computer vision. In 2018 IEEE Western New York image and signal processing workshop (WNYISPW) (pp. 1–5). IEEE.

Krishnakumar K., & Goldberg D. E. (1992). Control system optimization using genetic algorithms. Journal of Guidance, Control, and Dynamics, 15(3), 735–740. https://doi.org/10.2514/3.20898

Lin C. E., Liou M.-C., & Lee C.-M. (2014). Image fuzzy control on magnetic suspension ball and plate system. International Journal of Automation and Control Engineering, (IJACE), 3(2), 35–47. https://doi.org/10.14355/ijace.2014.0302.02

Lü X., Wu Y., Lian J., Zhang Y., Chen C., Wang P., & Meng L. (2020). Energy management of hybrid electric vehicles: A review of energy optimization of fuel cell hybrid power system based on genetic algorithm. Energy Conversion and Management, 205, 112474. https://doi.org/10.1016/j.enconman.2020.112474

Meena D., & Devanshu A. (2017). Genetic algorithm tuned pid controller for process control. In 2017 International Conference on Inventive Systems and Control (ICISC) (pp. 1–6). IEEE.

Meshram P., & Kanoojia R. G. (2012). Tuning of pid controller using ziegler-nichols method for speed control of dc motor. In IEEE-international conference on advances in engineering, science and management (ICAESM-2012) (pp. 117–122). IEEE.

Mnih V., Kavukcuoglu K., Silver D., Graves A., Antonoglou I., Wierstra D., & Riedmiller M.. Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602 (pp. 1–9).

Mohajerin N., Menhaj M. B., & Doustmohammadi A. (2010). A reinforcement learning fuzzy controller for the ball and plate system. In International conference on fuzzy systems (pp. 1–8). IEEE.

Mohammadi A., & Ryu J.-C. (2020). Neural network-based pid compensation for nonlinear systems: ball-on-plate example. International Journal of Dynamics and Control, 8(1), 178–188. https://doi.org/10.1007/s40435-018-0480-5

Nazarahari M., Khanmirza E., & Doostie S. (2019). Multi-objective multi-robot path planning in continuous environment using an enhanced genetic algorithm. Expert Systems with Applications, 115, 106–120. https://doi.org/10.1016/j.eswa.2018.08.008

Negash A., & Singh N. P. (2015). Position control and tracking of ball and plate system using fuzzy sliding mode controller. In Afro-European conference for industrial advancement (pp. 123–132). Springer.

Nokhbeh M., Khashabi D., & Talebi H. (2011). Modelling and control of ball-plate system (1–22). Amirkabir University of Technology.

Oravec M., & Jadlovska A. (2015). Model predictive control of a ball and plate laboratory model. In 2015 IEEE 13th international symposium on applied machine intelligence and informatics (SAMII) (pp. 165–170). IEEE.

Paz R. A. (2001). The design of the pid controller. Klipsch School of Electrical and Computer Engineering, 8, 1–23. http://www.geocities.ws/dushang2000/Microscopy/Zeiss%20Refractometer%20Repair%20Part%202/Paz%20R.The%20design%20of%20the%20PID%20controller.pdf

Rastin M. A., Talebzadeh E., Moosavian S. A. A., & Alaeddin M. (2013). Trajectory tracking and obstacle avoidance of a ball and plate system using fuzzy theory. In 2013 13th Iranian conference on fuzzy systems (IFSC) (pp. 1–5). IEEE.

Roy P., Kar B., & Hussain I. (2015). Trajectory control of a ball in a ball and plate system using cascaded pd controllers tuned by pso. In Proceedings of fourth international conference on soft computing for problem solving (pp. 53–65). Springer.

Spaček B. L. (2016). Digital control of ce 151 ball & plate model (pp. 1–71). Zlin, República Checa.

Umar A., Muazu M. B., Aliyu U. D., Musa U., Haruna Z., & Oyibo P. O. (2018). Position and trajectory tracking control for the ball and plate system using mixed sensitivity problem. Covenant Journal of Informatics and Communication Technology, 6(1), 1–15. http://107.180.69.44/index.php/cjict/article/view/948

Zhao Y., & Ge Y. (2014). The controller of ball and plate system designed based on FNN (1–6). Luoyang Institute of Science and Technology.