IAES International Journal of Artificial Intelligence (IJ-AI)
Vol. 13, No. 3, September 2024, pp. 2999~3009
ISSN: 2252-8938, DOI: 10.11591/ijai.v13.i3.pp2999-3009

Inverse kinematic solution and singularity avoidance using a deep deterministic policy gradient approach

Atikah Surriani, Oyas Wahyunggoro, Adha Imam Cahyadi
Department of Electrical and Information Engineering, Faculty of Engineering, Gadjah Mada University, Indonesia

Article Info

Article history:

Received Jan 3, 2024
Revised Feb 13, 2024
Accepted Mar 14, 2024

Keywords:

Arm robot manipulator
Deep deterministic policy gradient
Deep reinforcement learning
Inverse kinematic
Singularity

ABSTRACT

The robotic arm emerges as a subject of paramount significance within the industrial landscape, particularly in addressing the complexities of its kinematics. A significant research challenge lies in resolving the inverse kinematics of multiple degree of freedom (M-DOF) robotic arms. The inverse kinematics of M-DOF robotic arms pose a challenging problem to resolve, thus it involves consideration of singularities which affect the arm robot movement. This study aims a novel approach utilizing deep reinforcement learning (DRL) to tackle the inverse kinematic problem of the 6-DOF PUMA manipulator as a representative case within the M-DOF manipulator. The research employs Jacobian matrix for the kinematics system that can solve the singularity, and deep deterministic policy gradient (DDPG) as the kinematics solver. This chosen technique offers enhancing speed and ensuring stability. The results of inverse kinematic solution using DDPG were experimentally validated on a 6-DOF PUMA arm robot. The DDPG successfully solves inverse kinematic solution and avoids the singularity with 1,000 episodes and yielding a commendable total reward of 1,018.

Corresponding Author:

Oyas Wahyunggoro
Department of Electrical and Information Engineering, Faculty of Engineering, Gadjah Mada University
Grafika St., UGM Campus, Yogyakarta, Indonesia
Email: oyas@ugm.ac.id

1. INTRODUCTION

Arm robot manipulators hold a paramount position in the industrial landscape, and their applications are pervasive and diverse. The adaptability of these robotic arms, particularly their end-effectors, allows for customization to suit various tasks, such as gripping, magnetizing, and grinding [1]. Their extensive utility spans crucial industries, including automobile and aircraft manufacturing, medical equipment, and beyond. As a poignant example, the da Vinci surgical robot exemplifies the deployment of robotic arms for precise medical procedures [2]–[4]. Within the realm of robotic arm control, an urgent research area revolves around kinematics, encompassing forward kinematics and inverse kinematics. Forward kinematics entails determining the coordinate values of the end-effector, typically in a 3D cartesian space, based on the joint angles of the robotic arm [5], [6]. Conversely, inverse kinematics seeks to ascertain the exact joint angles necessary to achieve a desired end-effector coordinate [7]. These kinematic aspects are of utmost importance, as they underpin the decision-making process of robotic arms in reaching desired end-effector coordinates efficiently [8].

In general, inverse kinematics solutions are divided into two types: analytical solutions and numerical solutions. Analytical solutions represent a quick and efficient calculation of the inverse kinematics of a manipulator robot from joint angles to produce the desired end-effector configuration. Most industrial

manipulator robots have analytical solutions obtained using geometric and algebraic identities, solving a set of nonlinear algebraic equations related to defining the inverse kinematics problem [9]. The research conducted by [5] addressed the inverse kinematics problem of a tetrix manipulator using a closed-form solution. Additionally, research by Artemiadis [10] solved the inverse kinematics of a redundant manipulator robot. Shabeeb [11] employed a closed-form approach to solve the inverse kinematics problem of a 5 degree of freedom (DOF)-based manipulator robot. One of the primary advantages of closed-form solutions is their ability to compute rapidly and efficiently for each joint angle contributing to the end-effector position [8], [12]. However, the limitation of using closed-form solutions lies in their dependence on the geometric configuration of the robot manipulator, making no method universally applicable to solve this kinematic problem [13].

Numerical solutions rely on interactive procedures to solve the inverse kinematics of manipulator robots [14]. In numerical solutions, computations are carried out by considering stability and efficiency to obtain appropriate approximations of values [9], [12]. Generally, one of the widely used numerical methods in the field of robotics is artificial intelligence (AI). One of the primary goals in AI is to create fully autonomous agents that interact with their environment to learn different behaviours over time through trial and error. Numerical methods based on AI frameworks built upon mathematical frameworks for autonomous learning driven by experience include deep reinforcement learning (DRL) [15]. In the field of robotics, reinforcement learning has been applied to various types of robots, including mobile robots [16], [17], aerial robots [18], bipedal robots [19]–[21], humanoid robots [22], arm manipulator robots [23], [24]. Many papers employed DRL for arm robot manipulator, such as Kim *et al.* [25] employed robot arm markov decision process (RAMDP) with both 2-DOF and 3-DOF robot manipulators using twin delayed deep deterministic policy gradient (DDPG) (TD3) and [26] utilized soft-actor-critic (SAC) to control the movement of two 7 DOF robotic arms.

This study proposes the DDPG algorithm for the 6-DoF PUMA 560 robotic arm, a widely acclaimed solution for inverse kinematic problems within robotic arms [26], [27]. The DDPG algorithm employs a deterministic policy, contributing to enhanced stability in the learning process. The utilization of a deterministic policy ensures that the generated actions consistently fall within a specific range, leading to faster convergence. Effectively handles actions in continuous domains, allowing the agent to make precise choices within a continuous action spectrum. DDPG proficiently addresses high-dimensional data, enabling the agent to navigate a vast or high-dimensional action space, particularly in scenarios involving numerous variables or parameters, with high precision [28], [29]. Considering these strengths, the DDPG algorithm will serve as the agent in resolving the inverse kinematics of the 6-DOF PUMA 560 robotic arm. The success of these implementations will be rigorously evaluated.

2. MATERIAL AND METHOD

2.1. Deep deterministic policy gradient for inverse kinematic

DDPG is subset of the reinforcement learning algorithm, which operates through iterative exploration of the problem space, gathering insights, and selecting appropriate actions. In this context, it is employed to address the inverse kinematics problem by assessing the current state of the robotic system and determining joint configurations to achieve the precise end-effector pose. Reinforcement learning serves as a practical tool for solving sequential decision problems that can be modeled as Markov decision problems (MDPs) [30], [31]. MDPs are simply formulated in Markovian Processes. The Markov property implies that the next occurring process depends solely on the current observation outcomes. This results in the agent not referencing previous observation outcomes. A Markov decision process consists of 5 tuples $(S, A, \hat{P}, R, \gamma)$ [32]. The notation of S represents states, which is a finite set of conditions achievable by the environment. The variable A represents actions, which is a finite set of possible actions that the agent may choose. $\hat{P}$ is the state transition function, which is the probability function of state transitions. R is the reward function and γ is the discount factor.

DDPG employs an actor-critic architecture. The actor-critic architecture combines a value function with an explicit representation of the policy. The "actor" (policy) learns by receiving feedback from the "critic" (value function). In doing so, these methods trade-off between reducing the variance of the policy gradient and introducing bias from the value function methods [33]. The actor network $\mu(s; \theta^{\mu})$ serves as the policy and generates actions to be taken. The critic network $Q(s, a; \theta^{Q})$ acts as the Q-function and takes actions and states as input. The output of the critic provides an estimate of the reward for a particular action in that state. The target Q-network or target critic network, has parameter $\theta^{Q'}$.The target policy network or target actor network, with has parameter $\theta^{\mu'}$.These target networks are updated at each step through "soft updates," gradually tracking the learned networks. The updates are performed as follows:

$$\theta^{Q'} \leftarrow \tau\theta^{Q} + (1-\tau)\theta^{Q'} \quad (1)$$

$$\theta^{\mu'} \leftarrow \tau\theta^{\mu} + (1-\tau)\theta^{\mu'} \quad (2)$$

Here, τ represents the smoothing factor.

In addition to the target network, DDPG utilizes the replay buffer. The replay buffer is employed to sample experiences for updating the neural network weights. It temporarily stores the experience data that occurs during the agent's interaction with the environment. This replay buffer technique supports DDPG in learning offline, enabling it to learn action policies without directly interacting with the environment. This is beneficial for avoiding high computational costs during interaction with the environment. The DDPG algorithm uses mini-batch, which are small subsets of experiences randomly selected from the replay buffer. By using mini-batch, neural network updates can be performed efficiently. Mini-batch are used to update both the actor and critic networks. During network training, a random mini-batch is taken from the buffer, and then the target Q-value is computed using (3):

$$y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}; \theta^{\mu'}); \theta^{Q'}) \quad (3)$$

Subsequently, both the critic and actor networks are updated. On one hand, the critic is updated by minimizing the mean squared loss between the updated Q-values and the original Q-values using the liss as (4):

$$L = \frac{1}{N}\sum_i (y_i - Q(s_i, a_i; \theta^{Q}))^2 \quad (4)$$

On the other hand, the actor policy is updated using the sampled policy gradient as (5):

$$\nabla_{\theta^{\mu}} J \approx \frac{1}{N}\sum_i \nabla_a Q(s, a; \theta^{Q})\big|_{s=s_i, a=\mu(s_i)} \nabla_{\theta^{\mu}} \mu(s|\theta^{\mu})|_{s=s_t} \quad (5)$$

2.2. The 6-DoF PUMA 560 manipulator

This research endeavors to construct a model or environment for the 6-DOF PUMA 560. The system environment to be developed will entail an analytical solution or closed-form inverse kinematics for the M-DOF robotic arm, focusing on the homogeneous-based transformation. The mechanism governing the motion of the wrist joint is commonly referred to as a spherical wrist. Each joint possesses unique rotation limits and ranges. The model and configuration of the robot PUMA 560 is constructed based on a consensus of several PUMA robot parameters [34], [35]. the transformation matrix equation between the end-effector and the main coordinate frame ${}^{0}_{6}T$, is defined as:

$${}^{0}_{6}T = {}^{0}_{1}A\, A_{23}\, {}^{3}_{4}A\, {}^{4}_{5}A\, {}^{5}_{6}A = \begin{bmatrix} {}^{0}_{6}R & {}^{0}_{6}o \\ 0 & 1 \end{bmatrix} \quad (6)$$

$${}^{0}_{6}T = \begin{bmatrix} {}^{0}_{6}R & {}^{0}_{6}p \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (7)$$

Based on (6), the inverse kinematics equation for the PUMA 560 robot defines the following parameters:

$$\begin{array}{l} r_{12} = o_{\hat{x},}\ r_{13} = a_{\hat{x}},\ r_{13} = a_{\hat{x},} \quad p_x = p_{\hat{x}} \\ r_{22} = o_{\hat{y}},\ r_{23} = a_{\hat{y}},\ r_{23} = a_{\hat{y}}, \quad p_y = p_{\hat{y}} \\ r_{32} = o_{\hat{z}},\ r_{33} = a_{\hat{z}},\ r_{33} = a_{\hat{z}}, p_z - d_1 = p_{\hat{z}} \end{array} \quad (8)$$

The inverse kinematics solution for the joint angle q_1, begins by defining:

$$r_{q1} = +\sqrt{p_{\hat{x}}^2 + p_{\hat{y}}^2} \quad (9)$$

$$\vartheta_{q1} = \tan^{-1}\frac{p_{\hat{y}}}{p_{\hat{x}}} \quad (10)$$

The inverse kinematics solution for q_1, if the shoulder configuration of the PUMA 560 robot is right-handed, is as (11):

$$q_1 = \tan^{-1}\frac{p_{\hat{y}}}{p_{\hat{x}}} + \pi - \sin^{-1}\frac{d_3}{r_{q1}} \quad (11)$$

Another inverse kinematics solution for q_1, if the shoulder configuration of the PUMA 560 robot is left-handed, is as (12):

$$q_1 = \tan^{-1}\frac{p_{\hat{y}}}{p_{\hat{x}}} + \sin^{-1}\frac{d_3}{r_{q1}} \quad (12)$$

The inverse kinematics solution for the PUMA 560 robot's joint q_2, is related to the elbow configuration of the PUMA 560 robot, namely, elbow-up and elbow-down. The derivation of the inverse kinematics solution for joint q_2 begins by defining the following notations:

$$V_{114} = -s_{23}d_4 + c_{23}a_3 + a_2c_2 \quad (13)$$

$$r_{q2}\sin\vartheta_{q2} = p_{\hat{z}} \quad (14)$$

$$r_{q2}\cos\vartheta_{q2} = V_{114}, r_{q2} > 0 \quad (15)$$

Thus:

$$r_{q2} = +\sqrt{V_{114}^2 + p_{\hat{z}}^2} \quad (16)$$

$$\vartheta_{q2} = \tan^{-1}\frac{p_{\hat{z}}}{V_{114}} \quad (17)$$

$$\varphi = \cos^{-1}\frac{a_2^2 - d_4^2 - a_3^2 + V_{114}^2 + p_{\hat{z}}^2}{2a_2r_{q2}} \quad (18)$$

The inverse kinematics solution for q_2, is as (19):

$$q_2 = \tan^{-1}\frac{p_{\hat{z}}}{V_{114}} + \varphi \quad (19)$$

The first inverse kinematics solution in (19) is implemented for the elbow-up condition. The elbow-down configuration has a reverse value; therefore, the other inverse kinematics solution is defined as (20):

$$q_2 = \tan^{-1}\frac{p_{\hat{z}}}{V_{114}} - \varphi \quad (20)$$

The inverse kinematics solutions q_2 for the PUMA 560 robot in (19) and (20) can experience singularity when the value of φ is zero. The solution to the inverse kinematics problem for q_3 is as (21):

$$q_3 = \tan^{-1}\frac{a_3}{d_4} - \tan^{-1}\frac{c_2V_{114} + s_2p_{\hat{z}} - a_2}{c_2p_{\hat{z}} - s_2V_{114}} \quad (21)$$

Then, the end-effector of the PUMA 560 robot comprises 3 spherical wrist joints. Therefore, the solutions for q_4, q_5, and q_6 are interrelated. Before defining the wrist joint solutions, it is necessary to define some notations based on (7), as follows:

$$V_{323} = c_1a_{\hat{y}} - s_1a_{\hat{x}} \quad (22)$$

$$V_{113} = c_1a_{\hat{x}} + s_1a_{\hat{y}} \quad (23)$$

$$V_{313} = c_{23}V_{113} + s_{23}a_{\hat{z}} \quad (24)$$

From (22), (23), (24), we define the following:

$$s_4 = -\frac{1}{s_5}V_{323} \quad (25)$$

$$c_4 = -\frac{1}{s_5} V_{313} \tag{26}$$

The inverse kinematics solution for the fourth joint, q_4, is as (27):

$$q_4 = \begin{cases} \tan^{-1}\frac{-V_{323}}{-V_{313}}, & if\ s_5 > 0 \\ \tan^{-1}\frac{V_{323}}{V_{313}}, & if\ s_5 < 0 \cdot \\ undefined, & if\ s_5 = 0 \end{cases} \tag{27}$$

There are two inverse kinematics solutions for PUMA 560 q_4 in (27), and they are closely related to the value of s_5. The inverse kinematics value of PUMA 560 q_4 is undefined when $s_5 = 0$. This is due to singularity occurring at the joint. The value of s_5 can be defined as follows:

$$s_5 = -c_4 V_{313} - s_4 V_{323} \tag{28}$$

$$c_5 = -s_{23} V_{113} + c_{23} a_{\hat{z}} \tag{29}$$

As a result, the joint value q_5 is obtained as follows:

$$q_5 = \tan^{-1}\frac{s_5}{c_5} \tag{30}$$

The inverse kinematics solution for PUMA 560 joint q_6 is solved by first defining the following notations:

$$V_{412} = c_4 V_{312} - s_4 V_{132} \tag{31}$$

$$V_{422} = V_{332} \tag{32}$$

$$V_{432} = s_4 V_{312} + c_4 V_{132}, \tag{33}$$

$$V_{312} = c_{23} V_{112} + s_{23} p_{\hat{z}} \tag{34}$$

$$V_{332} = -s_{23} V_{112} + c_{23} o_{\hat{z}} \tag{35}$$

$$V_{132} = s_1 p_{\hat{x}} - c_1 p_{\hat{y}} \tag{36}$$

$$V_{112} = c_1 + s_1 p_{\hat{y}} \tag{37}$$

Next, s_6 and c_6 can be defined as:

$$s_6 = -c_5 V_{412} - s_5 V_{422} \tag{38}$$

$$c_6 = -V_{432} \tag{39}$$

Thus, q_6 can be obtained as (40):

$$q_6 = \tan^{-1}\frac{s_6}{c_6} \tag{40}$$

2.3. Singularity avoidance of 6-DoF PUMA 560 manipulator

The robotic arm system is characterized as a resolved-rate closed-loop system [34]. Resolved-rate closed-loop motion pertains to a control methodology that manipulates joint velocities on the robot to achieve a desired motion at a specific resolution level. This approach is primarily employed in robotics to address potential error accumulations during robot movements. Specifically, resolved-rate motion involves calculating the joint velocities necessary to reach a particular pose or position in the task space. These velocities are computed by considering the disparity between the desired pose and the actual pose of the robot. By comprehending this disparity, resolved-rate motion allows the robot to make precise adjustments to joint velocities, ensuring the attainment of the desired motion objectives with improved positional accuracy. The resolved-rate closed-loop system for the robotic arm is precisely defined as such (41):

$$\dot{q}^*(k) = J\big(q(k)\big)^{-1}\Big(p^*(k) \ominus \mathcal{K}\big(q(k)\big)\Big). \tag{41}$$

In (41) defines the desired joint velocity $\dot{q}^*(k)$ as the inverse of the Jacobian matrix at time k, multiplied by the difference between the desired pose $p^*(k)$ and the actual pose $\mathcal{K}\big(q(k)\big)$. The symbols used are as follows: $\dot{q}^*(k)$ represents the desired joint velocity at time k, $J\big(q(k)\big)^{-1}$ is the inverse of the Jacobian matrix at time k, $p^*(k)$is the desired pose or p_{target}, $\mathcal{K}\big(q(k)\big)$ is the actual pose or p_{ee}, and $((p^*(k) \ominus \mathcal{K}\big(q(k)\big))$ signifies the difference between the desired pose $p^*(k)$ and the actual pose $\mathcal{K}\big(q(k)\big)$, where the symbol $\ominus$ denotes the difference operation. The system is further defined by (42):

$$\dot{q}^*(k+1) = q(k) + Kp\delta t\dot{q}^*(k) \tag{42}$$

In this (42), $\dot{q}^*(k+1)$) represents the desired joint velocity at time k+1, $q(k)$ is the actual joint position at time k, Kp is the proportional gain controlling the system's response to the difference between the desired and actual poses, and δt is the time interval between two iterations. Utilizing (41) and (42), the change in joint velocity is calculated based on the difference between the desired and actual poses. A closed-loop control loop is employed to mitigate error accumulation, ensuring more accurate movements, and addressing undesired motion issues. Figure 1 shows the resolved-rate closed-loop system that remains inverse kinematic arm robot system with singularity avoidance.

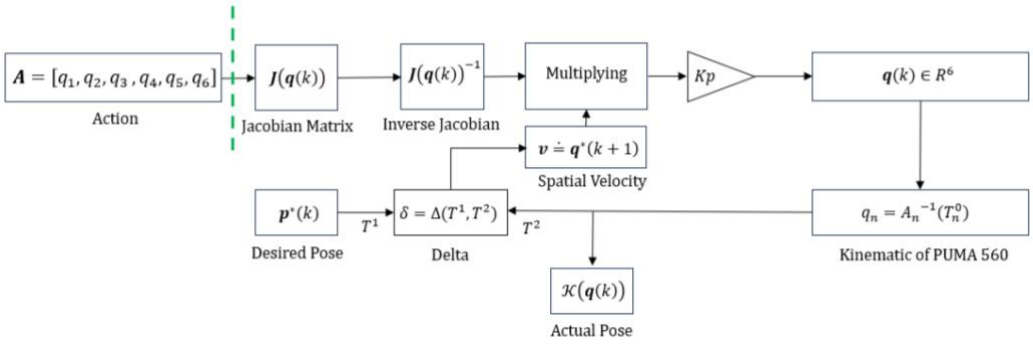

Figure 1. The resolved-rate closed-loop system of the PUMA 560

The resolved-rate closed-loop system of this robotic arm is indicated to be capable of overcoming singularity. The presence of the Jacobian matrix, $J\big(q(k)\big)^{-1}$, in the system is crucial for avoiding singularity. Singularity occurs when the Jacobian matrix loses or does not have an inverse, potentially causing issues when executing (41). Initially, this study considers a scenario where the Jacobian matrix does not become singular to preempt singularity conditions, ensuring that singularity in the inverse kinematics of the 6-DOF PUMA 560, can be avoided. The study then defines a singularity avoidance formulation for the inverse kinematics of the M-DOF manipulator robot, specifically the 6-DOF PUMA 560, as follows:

Singularity avoidance in inverse kinematics of M-DOF 6-DOF PUMA 560
Theorem 1. If $J(q)^{-1}$ is a singular matrix for all scalar $\forall p$, such that there exists $\hat{J}(q) = (J(q) + pI)^{-1}$ that is nonsingular for $\exists p$.

The statement asserts that if the inverse of the Jacobian matrix $J(q)^{-1}$ becomes singular for a certain value of p, then there may exist another value of p such that the matrix $\hat{J}(q) = (J(q) + pI)^{-1}$ does not become singular. To ensure that the matrix $\hat{J}(q) = (J(q) + pI)^{-1}$ is nonsingular, it is necessary to ensure that the matrix has a nonzero determinant. If the determinant of the matrix is nonzero, then the matrix has an inverse, and the matrix is nonsingular.

2.4. Implementation

The second phase commences with the design and subsequent implementation of the DDPG agent on 6-DOF PUMA 560 robotic arm. the PUMA 560 robotic arm. Therefore, actions in this system can be defined as follows:

$$A = [q_1, q_2, q_3, q_4, q_5, q_6]. \tag{43}$$

Where, q_i, represents the joint values. After DDPG issues a specific action to the environment, it receives feedback in the form of observations and rewards. The rewards are defined as two types:

– Encouraging reward, defined as (44):

$$R^1 = \begin{cases} 0, & if \ \|p_{ee} - p_{target}\| < 0.5 \\ -1, & if \ \|p_{ee} - p_{target}\| > 0.6 \end{cases} \tag{44}$$

– Terminating reward, defined as (45):

$$R^2 = \|p_{ee} - p_{target}\| \leq 0.5 \tag{45}$$

where, p_{ee} represents the actual coordinates of the PUMA 560 end-effector, and p_{target} represents the target coordinates. The system diagram is depicted in Figure 2. Within the artificial neural network segment of DDPG (DDPG network), there exist two distinct networks, such as the actor network and the critic network. The hyperparameter of DDPG agent is set as in Table 1.

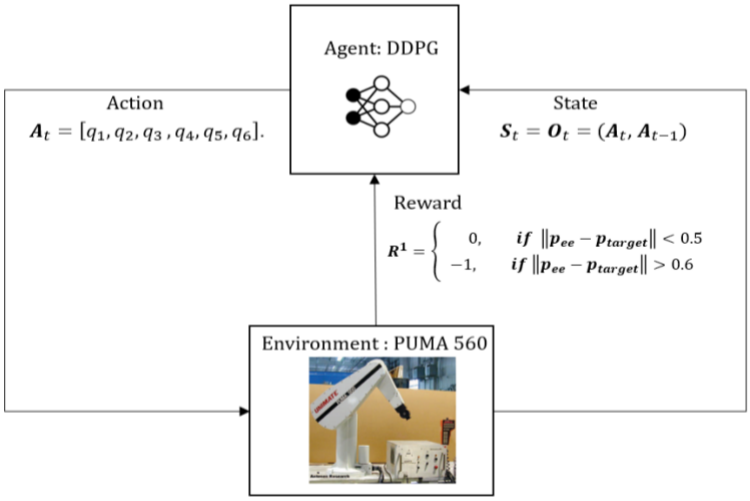

Figure 2. DDPG for inverse kinematic system

Table 1. Hyperparameter DDPG

Parameter	Value
Discount factor	0.99
Mini-batch	250
Reply buffer (R)	10^{-6}
Target smooth factor	10^{-3}

The learning process undertaken by the PUMA 560 robot with the DDPG controller is structured over k = 1,000 episodes. The evaluation of the control outcomes of the PUMA 560 robot is conducted through accuracy. The accuracy of the robot's movements will be assessed based on the actual pose of the end-effector and the desired pose. Algorithmic evaluation will be carried out considering the average reward and episode reward generated during the learning process. The average reward represents the mean reward obtained by the robot across the total episodes during the learning process. Episode reward is the reward generated by the agent in each episode throughout the learning process. The pseudocode for DDPG in solving the inverse kinematics of the 6-DOF PUMA 560 robotic arm, is outlined in Algorithm 1.

Algorithm 1. Pseudo-code DDPG for inverse kinematic solution

```
Initialisation desired pose p_target
Initialisation joint A = [q1, q2, q3, q4, q5, q6]
Initialisation actor network μ(s|θ^μ)
```

Initialisation parameter target, weight of actor network:
$\theta^{\mu} \in R, w^{\mu} = \{w^{\mu} | -0{,}5 \leq w^{\mu} \leq 0{,}5,\ w^{\mu} \in R\}, \mathrm{N_a}.$
Initialisation critic network $Q(s, a|\theta^{Q})$
Initialisation parameter target, weight of critic network:
$\theta^{Q} \in R, \quad w^{Q} = \{w^{Q} | -0{,}5 \leq w^{Q} \leq 0{,}5,\ w^{Q} \in R\}, \mathrm{N}_c$
Update: $\theta^{Q'} \leftarrow \theta^{Q}$, $\theta^{\mu'} \leftarrow \theta^{\mu}$
Initialisation parameter of DDPG: $\gamma, \tau \ll 1,\ R,\ Mini - batch.$
For episode =1, do
  For t = 1, do
 select *action* $A = [q_1, q_2, q_3, q_4, q_5, q_6]$, and
$$\mathrm{A} = \mu(s_t|\theta^{\mu}) + N_t$$
receive $S, and\ R$:
$$S = [A_t, A_{t+1}] \in R^{12},$$
$$R^1 = \begin{cases} 0, & if\ \|p_{ee} - p_{target}\| < 0.5 \\ -1, & if\ \|p_{ee} - p_{target}\| > 0.6 \end{cases}.$$
Store experience in R.
Random sample from *mini-batch* and set
$$y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'})$$
Estimate Critic network loss, L
Estimate actor network with policy gradient, $\nabla_{\theta^{\mu}} J$
update target parameter critic network, $\theta^{Q'}$, and actor network $\theta^{\mu'}$
'check if done' with terminating reward: $R^2 = \|p_{ee} - p_{target}\| \leq 0.05.$
  End for
End for

2.5. Evaluation for deep deterministic policy gradient

The parameters used to evaluate the performance of the DDPG and improved-DDPG algorithm includes episode reward (E_R), average reward (A_R), and total reward (T_R).

- The episode reward (E_R), represents the value of R^1 generated by the system in each episode, k.
- The average reward (A_R), is the average reward generated over a specific episode, given in (46):

$$A_R = \frac{\sum_{k=1}^{k'} E_R}{k'}. \tag{46}$$

where, k' is the current or actual episode.

- The total reward (T_R) is the cumulative reward generated by the system from the beginning to the end of the episode:

$$TR = \sum_{k=1}^{K=1000} R^1. \tag{47}$$

where, K is the final episode.

3. RESULTS AND DISCUSSION

3.1. Evaluation of deep deterministic policy gradient for 6-DOF PUMA 560

Evaluation of the design of DDPG agents and its implementation in the inverse kinematics environment of the 6-DOF PUMA 560 robotic arm. The results of the inverse kinematics learning of 6-DOF PUMA 560 robotic arm with DDPG is conducted for a total of 1000 episodes. Figures 3 and 4 show the reward and average reward results of the DDPG agent for the inverse kinematics solution of 6-DOF PUMA 560. Based on Figures 3 and 4, it is evident that the inverse kinematics solution process for the 6-DOF PUMA 560 robotic arm, results in an increase in reward below the 500th episode. The total reward generated from the DDPG implementation is 1,118.

3.2. Proof of singularity avoidance for 6-DOF PUMA 560

The proof provided demonstrates the singularity avoidance in the inverse kinematics of a multi-degree-of-freedom robotic arm, specifically in the context of a 6-degree-of-freedom PUMA 560 robotic arm. The proof of singularity avoidanve the 6-DOF PUMA 560 will be examined based on the determinant value and rank, that are derived from the Jacobian matrix. This proof is formulated in proof of singularity avoidance in inverse kinematics of M-DOF 6-DOF PUMA 560.

Proof of singularity avoidance in inverse kinematics of M-DOF 6-DOF PUMA 560
Proof.
If there be a Jacobian matrix *Jacobian* $J(q) \in R^{6x6}$, as follow:

$$J(q) = \begin{bmatrix} 0.1500 & -0.8636 & -0.4318 & 0.0000 & 0.0000 & 0.0000 \\ 0.0203 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0203 & 0.0203 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & -1.0000 & -1.0000 & 0.0000 & -1.0000 & 0.0000 \\ 1.0000 & 0.0000 & 0.0000 & 1.0000 & 0.0000 & 1.0000 \end{bmatrix},$$

We obtain, that $\det(J(q)) = 0$, and rank $\left(J(q)\right) = 5$, hence $J(q)$ is a singular matrix.
If the scalar value p, where $p \ll 0$, and an identity matrix I, to satisfy pI in the formulation $J(q) + pI$, so, we obtain:

$$pI = \begin{bmatrix} 0.0873 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0873 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 0.0873 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 0.0873 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0873 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0873 \end{bmatrix}.$$

Then, we obtain $J(q) + pI$ as follows:

$$J(q) + pI = \begin{bmatrix} 0.2373 & -0.8636 & -0.4318 & 0.0000 & 0.0000 & 0.0000 \\ 0.0203 & 0.0873 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0203 & 0.1076 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 0.0873 & 0.0000 & 0.0000 \\ 0.0000 & -1.0000 & -1.0000 & 0.0000 & -0.9127 & 0.0000 \\ 1.0000 & 0.0000 & 0.0000 & 1.0000 & 0.0000 & 1.0873 \end{bmatrix}.$$

$\mathrm{Det}(J(q) + pI) = -3.4116e - 04$, rank $(J(q) + pI) = 6$, hence $J(q) + pI$ is a non-singular matrix. Based on the above results, it is found that $J(q)^{-1}$ is a singular matrix, whereas $\hat{J}(q) = (J(q) + pI)^{-1}$ is a nonsingular matrix. Therefore, Theorem 1 is proven for a $p \ll 0$ in $(J(q) + pI)^{-1}$

The proof begins by examining a Jacobian matrix $J(q) \in R^{6x6}$ representing the robot arm's kinematics. The determinant and rank of $J(q)$ are calculated, indicating that it is a singular matrix. Then, a scalar value p is introduced, where $p \ll 0$, along with an identity matrix I, to form pI in the equation $(J(q) + pI)$. The resulting matrix pI is computed accordingly. Subsequently, pI is added to $J(q)$ to obtain $(J(q) + pI)$ and its determinant and rank are computed. The results show that $\hat{J}(q) = (J(q) + pI)^{-1}$ is non-singular, despite $J(q)$ being singular. Based on these findings, it is concluded that $J(q)^{-1}$ is a singular matrix, $(J(q) + pI)^{-1}$ becomes non-singular for very small values of p. This proof supports the claim made in Theorem 1 regarding singularity avoidance in the robotic arm's inverse kinematics.

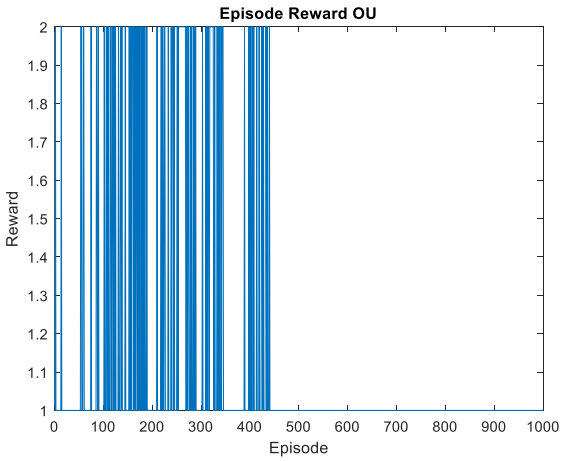

Figure 3. Reward for inverse kinematics solution of the 6-DOF PUMA 560 using DDPG

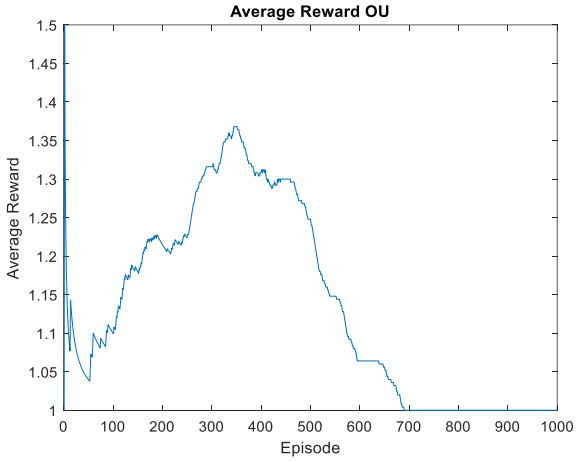

Figure 4. Average reward for inverse kinematics solution of the 6-DOF PUMA 560 using DDPG

4. CONCLUSION

This research demonstrates the performance of the DDPG algorithm in solving the inverse kinematics solution of a multi-degree-of-freedom robotic arm, with a case study involving the 6-DOF PUMA 560 robotic arm using the DDPG algorithm. Based on the conducted tests, the following conclusion is drawn that the implementation of the DDPG algorithm to solve the inverse kinematics solution of the multi-degree-of-freedom robotic arm, with a case study involving the 6-DOF PUMA 560 robotic arm, is successfully executed. The system's implementation has effectively learned to solve the inverse kinematics solution of the multi-degree-of-freedom robotic arm, with a case study involving the 6-DOF PUMA 560 robotic arm. It produces a total reward of 1,118. The DDPG algorithm has successfully completed the learning process and avoided singularity within 1,000 episodes of the learning process.

ACKNOWLEDGEMENTS

This research was granted by the *Final Project Recognition Grant* Universitas Gadjah Mada Number 5075/UN1.P.II/DitLit/PT.01.01/2023.

REFERENCES

[1] O. Kroemer, S. Niekum, and G. Konidaris, "A review of robot learning for manipulation: challenges, representations, and algorithms," *arXiv*, vol. 22, pp. 1–82, 2019.
[2] W.-J. Lee, "Ten-year experience of the da vinci robotic surgery at Severance Yonsei University Hospital in Korea," *Hanyang Medical Reviews*, vol. 36, no. 4, 2016, doi: 10.7599/hmr.2016.36.4.215.
[3] J. Bodner *et al.*, "The da Vinci robotic system for general surgical applications: a critical interim appraisal," *Swiss Medical Weekly*, vol. 135, no. 45–46, pp. 674–678, 2005.
[4] B. Liang, T. Li, Z. Chen, Y. Wang, and Y. Liao, "Robot arm dynamics control based on deep learning and simulation," in *2018 37th Chinese Control Conference (CCC)*, Technical Committee on Control Theory, Chinese Association of Automation, 2018, pp. 2921–2925.
[5] A. Novitarini, Y. Aniroh, D. Y. Anshori, and S. Budiprayitno, "A closed-form solution of inverse kinematic for 4 DOF tetrix manipulator robot," *Proceeding - ICAMIMIA 2017: International Conference on Advanced Mechatronics, Intelligent Manufacture, and Industrial Automation*, pp. 25–29, 2018, doi: 10.1109/ICAMIMIA.2017.8387551.
[6] V. O. S. Olunloyo, "On the mentor arm position placement problem : a forward kinematics analysis," *4th Robotics and Mechatronics Conference of South Africa (ROBMECH 2011)*, vol. 23, pp. 23–25, 2011.
[7] F. Hu, X. Wu, I. Senior, D. Luo, and I. Member, "Learning basic unit movements for humanoid arm motion control," *2016 IEEE 14th International Workshop on Advanced Motion Control (AMC)*, pp. 327–333, 2016, doi: 10.1109/AMC.2016.7496371.
[8] L. Zhang, J. Zuo, X. Zhang, X. Yao, and L. Shuai, "A new approach to inverse kinematic solution for a partially decoupled robot," *2015 International Conference on Control, Automation and Robotics, ICCAR 2015*, pp. 55–59, 2015, doi: 10.1109/ICCAR.2015.7166001.
[9] R. M. Murray, L. Zexiang, and S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation*. Boca Raton: CRC Press, 1994.
[10] P. Artemiadis, "Closed-form inverse kinematic solution for anthropomorphic motion in redundant robot arms," *Advances in Robotics & Automation*, vol. 02, no. 03, 2013, doi: 10.4172/2168-9695.1000110.
[11] A. H. Shabeeb, "Inverse kinematics analysis using close form solution method for 5 DOF robot manipulate," *Engineering an Technologi Journal*, vol. 33, no. 9, 2015.
[12] M. Pfurner, "Closed form inverse kinematics solution for a redundant anthropomorphic robot arm," *Computer Aided Geometric Design*, vol. 47, pp. 163–171, 2016, doi: 10.1016/j.cagd.2016.05.008.
[13] W. Shanda, L. Xiao, L. Qingsheng, and H. Baoling, "Existence conditions and general solutions of closed-form inverse kinematics for revolute serial robots," *Applied Sciences*, vol. 9, no. 20, pp. 1–36, 2019, doi: 10.3390/app9204365.
[14] H. Ren and P. Ben-tzvi, "Learning inverse kinematics and dynamics of a robotic manipulator using generative adversarial networks," *Robotics and Autonomous Systems*, vol. 124, 2020, doi: 10.1016/j.robot.2019.103386.
[15] A. Ataka, A. Sandiwan, H. Tnunay, D. R. Utomo, and A. I. Cahyadi, "Inverted Pendulum control: a comparative study from conventional control to reinforcement learning," *Jurnal Nasional Teknik Elektro dan Teknologi Informasi*, vol. 12, no. 3, pp. 197–204, 2023, doi: 10.22146/jnteti.v12i3.7065.
[16] X. Ruan, D. Ren, X. Zhu, and J. Huang, "Mobile robot navigation based on deep reinforcement learning," in *Chinese Control and Decision Conference (2019 CCDC)*, China: IEEE, 2019, pp. 6174–6178.
[17] H. Quan, Y. Li, and Y. Zhang, "A novel mobile robot navigation method based on deep reinforcement learning," *International Journal of Advanced Robotic Systems*, no. June, pp. 1–11, 2020, doi: 10.1177/1729881420921672.
[18] S. Jung, W. J. Yun, J. Kim, and J. -H. Kim, "Infrastructure-assisted cooperative multi-UAV deep reinforcement energy trading learning for big-data processing," in *2021 International Conference on Information Networking (ICOIN)*, 2021, pp. 159–162. doi: DOI: 10.1109/ICOIN50884.2021.9333895.
[19] S. Yamada, A. Watanabe, and M. Nakashirna, "Hybrid reinforcement learning and its application to biped robot control," *Advances in Neural Information Processing Systems*, pp. 1071–1077, 1998.
[20] A. Surriani, O. Wahyunggoro, and A. I. Cahyadi, "A trajectory control for bipedal walking robot using stochastic-based continuous deep reinforcement learning," *Evergreen*, vol. 10, no. 3, pp. 1538–1548, 2023, doi: 10.5109/7151701.
[21] P. B. Khoi, N. T. Giang, and H. V. Tan, "Control and simulation of a 6-DOF biped robot based on twin delayed deep deterministic policy gradient algorithm," *Indian Journal of Science and Technology*, vol. 14, no. 31, pp. 2460–2471, 2021, doi: 10.17485/ijst/v14i30.1030.
[22] Q. Shi, W. Ying, L. Lv, and J. Xie, "Deep reinforcement learning-based attitude motion control for humanoid robots with stability constraints," *Industrial Robot: the international journal ofrobotics research and application*, vol. 3, pp. 335–347, 2020, doi: 10.1108/IR-11-2019-0240.
[23] Y. Wu, Z. Yu, C. Li, M. He, B. Hua, and Z. Chen, "Reinforcement learning in dual-arm trajectory planning for a free-floating

space robot," *Aerospace Science and Technology*, vol. 98, 2020, doi: 10.1016/j.ast.2019.105657.
[24] A. Malik, Y. Lischuk, T. Henderson, and R. Prazenica, "A deep reinforcement-learning approach for inverse kinematics solution of a high degree of freedom robotic manipulator," *Robotics*, vol. 11, no. 2, 2022, doi: 10.3390/robotics11020044.
[25] M. Kim, D. Han, and J. Park, "Motion planning of robot manipulators for a smoother path using a twin delayed deep deterministic policy gradient with hindsight experience replay," *Applied Sciences*, vol. 10, no. 2, 2020, doi: 10.3390/app10020575.
[26] C. Wong, S. Chien, H. Feng, and H. Aoyama, "Motion Planning for dual-arm robot based on soft actor-critic," *IEEE Access*, vol. 9, pp. 26871–26885, 2021, doi: 10.1109/ACCESS.2021.3056903.
[27] A. S. Polydoros and L. Nalpantidis, "Survey of model-based reinforcement learning: applications on robotics," *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 86, no. 2, pp. 153–173, 2017, doi: 10.1007/s10846-017-0468-y.
[28] D. Han, B. Mulyana, V. Stankovic, and S. Cheng, "A survey on deep reinforcement learning algorithms for robotic manipulation," *Sensors*, vol. 23, no. 7, pp. 1–35, 2023, doi: 10.3390/s23073762.
[29] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," in *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*, London UK, pp. 1-14, 2016.
[30] R. S. Sutton and A. G. Barto, *Reinforcement learning: an introduction*. Cambridge, Massachusetts: MIT Press, 2018.
[31] S. Gu, E. Holly, T. Lillicrap, and S. Levine, "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates," *IEEE International Conference on Robotics and Automation*, pp. 3389–3396, 2017, doi: 10.1109/ICRA.2017.7989385.
[32] A. Surriani, O. Wahyunggoro, and A. I. Cahyadi, "Noise parameterization of continuous deep reinforcement learning for a class of non-linear system," *ICITEE 2022 - Proceedings of the 14th International Conference on Information Technology and Electrical Engineering*, pp. 24–29, 2022, doi: 10.1109/ICITEE56407.2022.9954121.
[33] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning," *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 26-38, Nov. 2017, doi: 10.1109/MSP.2017.2743240.
[34] P. Corke, *Robotics Vision and Cotrol, Fundamental Algorithms in MATLAB ®*, Berlin, Heidelberg: Springer Tracts in Advance Robotics, 2011.
[35] P. I. Corke and B. Armstrong-Helouvry, "Search for consensus among model parameters reported for the PUMA 560 robot," *IEEE International Conference on Robotics and Automation*, no. 2, pp. 1608–1613, 1994, doi: 10.1109/robot.1994.351360.

BIOGRAPHIES OF AUTHORS

Atikah Surriani received the undergraduate degree (S.T.) in electrical engineering from the Universitas Lampung (Unila), Indonesia, in 2011, and the master degree (M.Eng.) in electrical engineering from Universitas Gadjah Mada (UGM), Indonesia, in 2015. She is currently pursuing the Doctoral degree (Dr.Eng.) with Universitas Gadjah Mada (UGM), Indonesia. Currently, she is a lecturer with Universitas Gadjah Mada (UGM), Indonesia. Her research concerns control system and robotics especially, UAV control system, and arm robot. She can be contacted at email: atikah.surriani.sie13@ugm.ac.id.

Oyas Wahyunggoro was born in Yogyakarta, Indonesia. He received the undergraduate degree (Ir.) in electrical engineering from Universitas Gadjah Mada (UGM), Indonesia, Feb 1993, the master degree (M.T.) in electrical engineering from Universitas Gadjah Mada (UGM), Yogyakarta, May 2001, and Ph.D. degree in automation and control system from Universiti Teknologi PETRONAS, Malaysia, Oct 2011. Currently, he is an Associate Professor with the Department of Electrical and Information Engineering, Engineering Faculty, Universitas Gadjah Mada, Indonesia. His research focuses are BMS, applying intelligent system on automation and control system, and biomedical signal processing. He can be contacted at email: oyas@ugm.ac.id.

Adha Imam Cahyadi obtained his B.Eng. in Electrical Engineering in 2002, M.Eng. in Control Engineering in 2005, and Dr. Engineering in Robotics in 2008 respectively from Universitas Gadjah Mada (UGM) Indonesia, KMITL Thailand and Tokai University Japan. He has been with Departement of Electrical Engineering and Information Technology UGM since 2011. Since 2016, he has been serving the Department as Chairman for EE Program. His research area is mostly related to control applications and robotics such as teleoperation, control for industrial plants, and unmanned systems. He can be contacted at email: adha.imam@ugm.ac.id.