drones

MDPI

*Article*

# A Fuzzy Pure Pursuit for Autonomous UGVs Based on Model Predictive Control and Whole-Body Motion Control

**Yaoyu Sui [1], Zhong Yang [1,*], Haoze Zhuo [2], Yulong You [1], Wenqiang Que [1] and Naifeng He [1]**

[1] College of Automation Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China; suiyaoyu@nuaa.edu.cn (Y.S.); youyulong@nuaa.edu.cn (Y.Y.); quewenqiang@nuaa.edu.cn (W.Q.); nfhe@nuaa.edu.cn (N.H.)
[2] Guangxi Electric Power Research Institute, Guangxi Power Grid Co., Ltd., Nanning 530004, China; zhuohaoze@nuaa.edu.cn
* Correspondence: yangzhong@nuaa.edu.cn

**Abstract:** In this paper, we propose an adaptive fuzzy pure pursuit trajectory tracking algorithm for autonomous unmanned ground vehicles (UGVs), addressing the challenges of accurate and stable navigation in complex environments. Traditional pure pursuit methods with fixed look-ahead distances struggle to maintain precision in dynamic and uneven terrains. Our approach uniquely integrates a fuzzy control algorithm that allows for real-time adjustments of the look-ahead distance based on environmental feedback, thereby enhancing tracking accuracy and smoothness. Additionally, we combine this with model predictive control (MPC) and whole-body motion control (WBC), where MPC forecasts future states and optimally adjusts control actions, while WBC ensures coordinated motion of the UGV, maintaining balance and stability, especially in rough terrains. This integration not only improves responsiveness to changing conditions but also enables dynamic balance adjustments during movement. The proposed algorithm was validated through simulations in Gazebo and real-world experiments on physical platforms. In real-world tests, our algorithm reduced the average trajectory tracking error by 45% and the standard deviation by nearly 50%, significantly improving stability and accuracy compared to traditional methods.

**Keywords:** UGVs; trajectory tracking; pure pursuit; model predictive control; whole-body control

check for updates

**Citation:** Sui, Y.; Yang, Z.; Zhuo, H.; You, Y.; Que, W.; He, N. A Fuzzy Pure Pursuit for Autonomous UGVs Based on Model Predictive Control and Whole-Body Motion Control. *Drones* **2024**, *8*, 554. https://doi.org/10.3390/drones8100554

Academic Editor: Haibin Duan

Received: 11 September 2024
Revised: 2 October 2024
Accepted: 4 October 2024
Published: 6 October 2024

## 1. Introduction

In recent years, unmanned ground vehicles (UGVs) have become a key focus in the field of robotics, particularly for their ability to navigate autonomously in diverse environments. UGVs, designed for autonomous operation in both structured and unstructured terrains, are increasingly used in applications ranging from military reconnaissance to industrial logistics. Among various types of UGVs, legged robots, especially quadruped robots, stand out due to their unique locomotion capabilities. Unlike traditional wheeled or tracked UGVs, quadruped robots [1–6] can traverse complex and uneven terrains more effectively, making them highly versatile for outdoor environments. This makes legged robots an important branch in the field of UGVs that attracts much attention [7–9]. This versatility positions quadruped robots as promising candidates for applications such as detection in unstructured environments, where terrain instability presents significant challenges for navigation and trajectory tracking [10]. In such scenarios, quadruped robots can provide crucial data and insights due to their ability to traverse rugged surfaces and adapt to varying conditions. Their unique locomotion capabilities enable them to maintain stability and maneuver effectively in environments that would hinder traditional wheeled or tracked vehicles. Studies [11] have highlighted the effectiveness of quadruped robots in navigating unstructured terrains, emphasizing the importance of developing advanced control algorithms to enhance their performance in these demanding situations. With the development and maturity of bionic technology, people seek inspiration from animals and start to study bionic quadruped robots, hoping to obtain the powerful locomotion abilities of these animals through mechanical bodies [12–16]. Figure 1 shows our quadruped robot performing the challenging task of tracking in complex terrain.

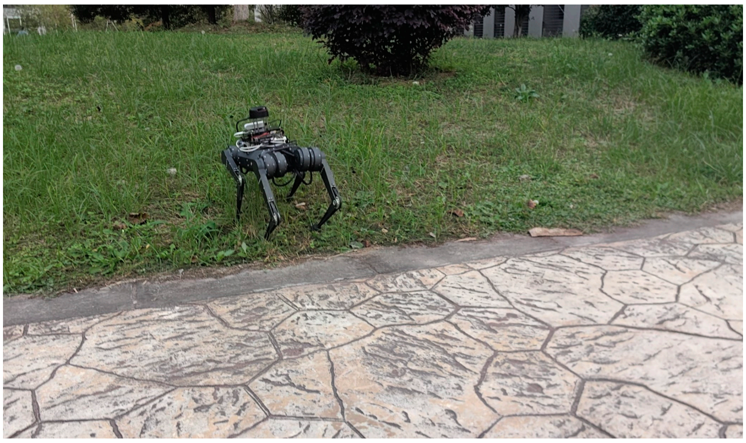


**Figure 1.** The quadruped robot used in the physical experiments is about to take on the challenge of tracking an S-shaped curve in complex outdoor terrain.

Path tracking is very important for quadruped robots, as it can enable them to move and navigate autonomously along a planned path in complex environments, avoiding collisions and traps and accomplishing various tasks [17–20]. Path tracking refers to planning a reasonable trajectory according to the target position and environmental information, making the quadruped robot walk along this trajectory, and adjusting and optimizing the trajectory according to real-time feedback information. However, there is still no more reasonable solution to the trajectory tracking ability of existing quadruped robots in complex environments.

Based on this, this paper firstly adopts the pure pursuit tracking algorithm, which is widely used in the path-tracking task of quadruped robots, to complete the path-tracking task of quadruped robots for the specified trajectory [21,22]. The basic idea of the pure pursuit tracking algorithm is that the robot chooses a look-ahead point within a certain range of itself as the target point at each moment, then calculates the turning angle between the robot and the look-ahead point and controls the robot's motion according to the turning angle. In this way, the robot can gradually approach and follow the path, achieving path tracking. Pure pursuit tracking can adapt to paths with large curvature changes and has the advantage of easy implementation.

However, it relies heavily on a fixed look-ahead distance, which can lead to significant errors, particularly in dynamic and uneven terrains. This fixed look-ahead distance also adversely affects the turning capabilities of quadruped robots in complex terrains, resulting in difficulties in accurately tracking the desired trajectory. Additionally, traditional pure pursuit methods often struggle with maintaining high precision and stability, especially when faced with unexpected obstacles or rapid changes in the environment.

Despite advancements, these methods do not adequately address the unique challenges presented by complex terrains. For instance, while some researchers have proposed improvements to the pure pursuit algorithm, such as using optimization techniques for look-ahead distance, they often require extensive parameter tuning, which is not feasible in real-time applications. Furthermore, the need for predefined gait patterns in certain adaptive gait methods limits their flexibility and responsiveness in rapidly changing environments. This highlights the necessity for a more robust and adaptive approach that can dynamically adjust to varying conditions without extensive manual intervention.

To solve the above problems, this paper uses a fuzzy controller to adjust the look-ahead distance so that it can be adjusted online according to the real-time error of the robot [23–25]. A fuzzy controller is a control method based on fuzzy logic, which can deal with uncertainty and nonlinearity and provide an intuitive and flexible way to design control rules. By applying the fuzzy controller to the pure pursuit tracking algorithm, this paper can improve the accuracy and robustness of path tracking while adapting to different speeds and environments.

Moreover, due to the limitations of the control algorithm, the quadruped robot still exhibits poor trajectory tracking performance when navigating uneven terrain and other complex environments. To tackle these limitations, this paper introduces a combined control approach that integrates model predictive control (MPC) and whole-body motion control (WBC) with the fuzzy pure pursuit algorithm. The combination of MPC and WBC provides a comprehensive solution that not only optimizes the trajectory tracking process but also enhances the stability and adaptability of the quadruped robot. MPC can generate the optimal control input according to the robot's kinematic or dynamic model and some constraints [26–28]. WBC is a control method based on task space, which can allocate joint torques according to the robot's desired posture and contact state, and some objective functions [29–31].

MPC plays a critical role by predicting future states based on the robot's kinematic and dynamic models, allowing for real-time adjustments to control actions. This predictive capability is essential for navigating uneven terrains effectively. Meanwhile, WBC allocates joint torques based on the robot's desired posture and contact states, ensuring that the robot maintains balance and stability during movement.

By integrating these methodologies, our approach addresses the shortcomings of existing algorithms and enables the quadruped robot to adapt its motion dynamically in response to environmental changes, thereby significantly improving its path-tracking performance in complex scenarios.

Gao Haibo [32,33] proposed a method for quadruped robot environment cognition learning and autonomous navigation based on deep reinforcement learning. He applied this method to outdoor legged robots, demonstrating adaptability to various complex terrains. Although his method uses deep reinforcement learning to achieve end-to-end learning, it requires a large amount of data and computational resources and may have issues such as instability, non-convergence, and overfitting. This paper uses fuzzy pure pursuit to dynamically select target points based on environmental information and robot error and adjusts the position and distance of the target points according to fuzzy logic, enabling the robot to flexibly follow the path in different environments.

Koco [34] and others used an adaptive gait method that adjusts the robot's gait parameters in real time based on the robot's speed, direction, and terrain information. However, this method relies on several predefined gait patterns, and the adjustment of parameters may be limited and delayed in complex and changing environments. Our method uses fuzzy pure pursuit, which adjusts the look-ahead distance adaptively based on the robot's current state and desired trajectory, ensuring that the robot moves along the desired trajectory. This method does not require predefined gait patterns and has strong robustness and adaptability.

Li Zhongyu [35] proposed a guide dog method based on visual and tactile feedback. He applied this method to Mini-Cheetah, enabling it to sense the environment, track the owner's position, and tighten or loosen the belt to guide the owner. Although Li Zhongyu's method uses visual and tactile feedback to perceive the environment and guide the owner, it depends on the performance of visual sensors and algorithms and may be affected by factors such as lighting, occlusion, and dynamic changes. This paper uses MPC to generate the robot's desired posture and contact state and optimizes the control quantity according to some constraints and objective functions, enabling the robot to maintain balance and stability on different terrains.

Rui Wang [36] proposed a new algorithm that improves the tracking accuracy of the pure pursuit algorithm, referred to as OLDPPA. This algorithm uses the Salmon Swarm Algorithm (SSA) to find the optimal look-ahead distance and introduces Brownian motion and a weighting mechanism to enhance SSA's search capability and convergence speed. However, this algorithm has not been applied to quadruped robots.

Yaofu Huang [37] proposed an optimization method based on the PID controller. First, the look-ahead distance is dynamically calculated based on the speed and path information of the unmanned vehicle; then, the PID controller is used to optimize the target steering angle calculated by the pure pursuit algorithm, outputting a smooth steering angle. However, a fixed look-ahead distance can lead to large lateral errors and heading angle errors during tracking. This paper uses a fuzzy algorithm to adjust the look-ahead distance online, effectively improving the stability and accuracy of path tracking.

Steve Macenski [38] proposed an improved pure pursuit algorithm. This algorithm adds two heuristic functions to the existing adaptive pure pursuit algorithm, reducing the robot's linear speed based on path curvature and obstacle distance. However, the pure pursuit algorithm and its variants may short-circuit or overshoot during high-curvature turns, leading to path deviation. Although the improved algorithm reduces this phenomenon by lowering the speed, it does not completely eliminate it. The fuzzy pure pursuit algorithm proposed in this paper adjusts the look-ahead distance online according to the tracking error, deviating from the original path with large errors, making the tracking process more accurate.

Traditional pure trajectory tracking controllers struggle to track the desired trajectory with high precision, speed, and stability in unknown environments. In the past, the parameter gains of pure trajectory tracking controllers often required expert-level prior knowledge and numerous experiments for repeated adjustments to achieve optimal control. Therefore, to avoid the tedious process of adjusting controller parameter gains for different tasks, this paper will study a fuzzy adaptive trajectory tracking control method and use a combination of MPC and WBC to solve the automatic tuning problem of trajectory tracking controller parameters under different environmental requirements. This section will detail the theory of pure trajectory tracking control and the fuzzy control algorithm. These basic concepts are the foundation of the adaptive trajectory tracking control for quadruped robots proposed in this paper.

In Section 2 of this paper, we model the kinematics and dynamics of the quadruped robot. In Section 3, we describe the pure trajectory tracking algorithm based on the kinematic model of the quadruped robot and use a fuzzy algorithm to adjust its look-ahead distance online. In Section 4, based on the dynamic model of the quadruped robot, we combine the MPC and WBC control algorithms to further improve the quadruped robot's ability to track the desired trajectory in complex terrains. In Section 5, we verify the accuracy and effectiveness of the algorithm proposed in this paper through multiple sets of simulations and physical experiments.

## 2. Simplified Robot Kinematics and Dynamics

### *2.1. Steering Kinematics for Quadruped Robot*

When a quadruped robot is turning, it needs to plan the end-effector and body poses based on its own and motion direction parameters. The body coordinate system, denoted as $\{\mathcal{B}\}$, is defined with its origin fixed at the center of mass of the robot. It moves together with the robot during forward motion, with the positive $x$-axis aligned with the body's forward direction, the positive $y$-axis pointing towards the rotation center, and the positive $z$-axis oriented vertically upward with respect to the horizontal ground. All coordinate systems are defined in Figure 2.

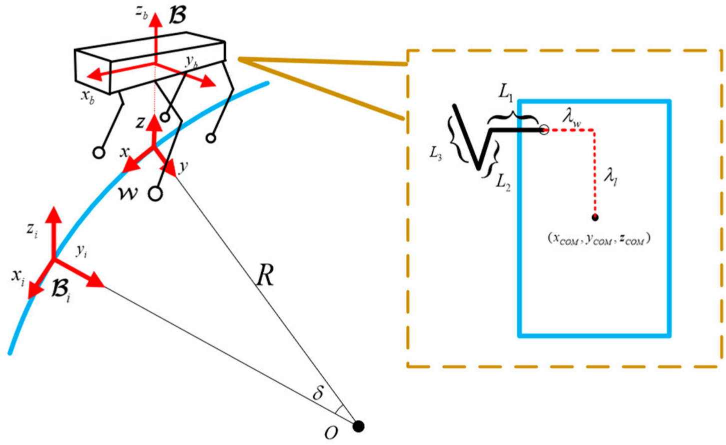


**Figure 2.** Simplified model of quadruped robot.

The coordinates of the end-effector in the body coordinate system $\{\mathcal{B}\}$ are calculated using the Denavit–Hartenberg (DH) method:

$$P^{\mathcal{B}} = \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} = \begin{bmatrix} L_2 s_2 + L_3 s_{23} + \lambda_l \\ s_1(L_1 + L_2 c_2 + L_3 c_{23}) + \lambda_W \\ -c_1(L_1 + L_2 c_2 + L_3 c_{23}) - \lambda_c \end{bmatrix} \tag{1}$$

where in the forward kinematics equation of the quadruped robot, $P_x$, $P_y$, and $P_z$ represent the $x$, $y$, and $z$ coordinates of the robot's left front foot in the body coordinate system. The lengths of the robot's three links are denoted by $L_1$, $L_2$, and $L_3$. The joint angles are represented as $\theta_1$, $\theta_2$, and $\theta_3$, which correspond to the hip, knee, and ankle joints, respectively. To describe the orientation, $s_i = \sin(\theta_i)$, $s_{ij} = \sin(\theta_i + \theta_j)$, $c_i = \cos(\theta_i)$, $c_{ij} = \cos(\theta_i + \theta_j)$. $\lambda_l$, $\lambda_w$, and $\lambda_c$ are used to specify the distances from the center of the robot's body coordinate system to the hip joint in the $x$-, $y$-, and $z$-axis directions, respectively. Since the hip joint and the robot's body share the same $z$-coordinate, the value of $\lambda_c$ is zero. The same approach can be applied to calculate the positions for the other three legs.

Define the world coordinate system $\{W\}$ fixed at the projection of the coordinate system on the ground at the beginning of the turning movement. Define the local reference coordinate system $\{\mathcal{B}_i\}$ to be fixed at the projection of coordinate system $\{\mathcal{B}\}$ on the ground at the beginning of each turning gait cycle. $\delta$ is the angle that the robot turns in each gait cycle, and $R$ is the turn radius.

At the beginning of the $i$-th period, the robot is turned by $(i-1)\delta$; then, the parameter transformation matrix of the local reference coordinate system $\{\mathcal{B}_i\}$ with respect to the world coordinate system $\{W\}$ is as follows:

$$T^{W}_{\mathcal{B}_i} = \begin{bmatrix} \cos((i-1)\delta) & -\sin((i-1)\delta) & 0 & R\sin(i-1)\delta \\ \sin((i-1))\delta & \cos((i-1)\delta) & 0 & 2R\sin^2(\frac{(i-1)\delta}{2}) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2}$$

The transformation matrix between the coordinate system $\{\mathcal{B}\}$ and $\{\mathcal{B}_i\}$ can be expressed as follows:

$$T^{\mathcal{B}}_{\mathcal{B}_i} = \begin{bmatrix} 1 & 0 & 0 & x_{COM} \\ 0 & 1 & 0 & y_{COM} \\ 0 & 0 & 1 & z_{COM} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3}$$

The coordinate $P^W$ of the foot end in $\{W\}$ can be represented as follows:

$$P^W = T^W_{\mathcal{B}_i} T^W_{\mathcal{B}_i} P^{\mathcal{B}} \tag{4}$$

### *2.2. Dynamics of Quadruped Robot*

The state variables are defined as the robot's pose angles, center of mass position, body angular velocity, and body center of mass angular acceleration.

The dynamic model is established as shown in the following equation:

$$\Omega = \begin{bmatrix} \Theta \\ P \\ \omega \\ V \end{bmatrix} \tag{5}$$

where $\Theta = [\phi, \theta, \varphi]^\top$ represents the robot's orientation, which are the roll, pitch, and yaw angles of the robot, respectively. $P \in \mathrm{R}^3$ is position of the robot. $\omega \in \mathrm{R}^3$ is the angular velocity of the robot. $V \in \mathrm{R}^3$ represents the velocity of the robot.

The relationship between body force and acceleration can be derived from Newton's equations:

$$\dot{V} = \frac{\sum_{i=1}^{n} f_i}{m} - g \tag{6}$$

where $f_i \in \mathrm{R}^3$ represents the external forces acting on foot end of the robot, $i = 0, 1, 2, 3$ represents robot's $i$-th foot, $m$ is the total mass of the robot, and $g \in \mathrm{R}^3$ is the gravitational acceleration.

The rigid body dynamic model in the world coordinate system is described by the following equations:

$$\begin{aligned} \dot{T} &= [\omega]_\times T \\ T &= T_z(\varphi) T_y(\theta) T_x(\phi) \end{aligned} \tag{7}$$

$T \in \mathrm{R}^3$ represents the rotation matrix for the transformation from the body coordinate system to the world. $[\omega]_\times \in \mathrm{R}^3$ is defined as the skew-symmetric matrix. $T_n(\alpha)$ means a positive rotation of $\alpha$ about the $n$-axis.

During the motion, the pitch and roll angles of the robot can be approximated as zero. Therefore, the angular velocity dynamics in the world coordinate system can be simplified as follows:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\varphi} \end{bmatrix} \approx \begin{bmatrix} cos(\varphi) & sin(\varphi) & 0 \\ -sin(\varphi) & cos(\varphi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \omega \approx T_z(\varphi)\omega \tag{8}$$

where

$$\omega = \begin{bmatrix} \cos(\theta)\cos(\varphi) & -\sin(\varphi) & 0 \\ \cos(\theta)\sin(\varphi) & \cos(\varphi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\varphi} \end{bmatrix} \tag{9}$$

The rotation torque of the robot can be obtained as the following:

$$\frac{d}{dt}(I\omega) = \sum_{i=1}^{n} r_i \times f_i \tag{10}$$

where $I \in \mathrm{R}^3$ is the inertia tensor matrix of the robot:

$$I = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \tag{11}$$

The above equation can be approximated as follows:

$$\frac{d}{dt}(I\omega) = I\dot{\omega} + \omega \times (I\omega) \approx I\dot{\omega} \tag{12}$$

In summary, an approximate inertia tensor $\widetilde{I}$ can be expressed as follows:

$$\widetilde{I} = T_z(\varphi) I T_z(\varphi)^{\top} \tag{13}$$

From the analysis above, if $[0 \quad 0 \quad 0 \quad g]^T$ is rewritten as a state quantity $\mathbf{g}$ of the system, the quadruped robot dynamics can be expressed as follows:

$$\frac{d}{dt}\begin{bmatrix} \Theta \\ P \\ \omega \\ V \\ \mathbf{g} \end{bmatrix} = \begin{bmatrix} O_3 & O_3 & T_z(\varphi) & O_3 & 0 \\ O_3 & O_3 & O_3 & 1_3 & 0 \\ O_3 & O_3 & O_3 & O_3 & 0 \\ O_3 & O_3 & O_3 & O_3 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} \Theta \\ P \\ \omega \\ V \\ \mathbf{g} \end{bmatrix} + \begin{bmatrix} O_3 & O_3 & O_3 & O_3 \\ O_3 & O_3 & O_3 & O_3 \\ \widetilde{I}^{-1}[r_1]_\times & \widetilde{I}^{-1}[r_2]_\times & \widetilde{I}^{-1}[r_3]_\times & \widetilde{I}^{-1}[r_4]_\times \\ 1_3/m & 1_3/m & 1_3/m & 1_3/m \\ 0 & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix} \tag{14}$$

## 3. Fuzzy Pure Pursuit

### *3.1. Pure Pursuit for Quadruped Robots*

Pure pursuit (PP) is a geometry-based path-tracking algorithm. Its control method is intuitive and can be well adapted to mobile robots. In the steering kinematics above, the relationship between the steering radius, the steering angle and the coordinates of the robot foot end has been revealed. Based on this, in the research of the pure trajectory tracking algorithm, it is necessary to link the desired path with both.

The main idea of pure trajectory tracking is to decompose the desired path into a few path points. In the process of tracking the trajectory, the robot sequentially passes through the path points on the trajectory until it reaches the endpoint. Therefore, the overall process of trajectory tracking can be divided into multiple stages, delineated by the waypoints along the trajectory. The robot's task is to advance towards the nearest target point, adjusting its speed and steering angle during this process to ensure effective tracking of the desired path.

As shown in Figure 3, this illustrates the process of a quadruped robot tracking one of the waypoints. In the diagram, $(g_x, g_y)$ represents the next waypoint to be tracked, located on the previously planned global path. The current task involves controlling the center of the quadruped robot to ensure it passes through this waypoint.

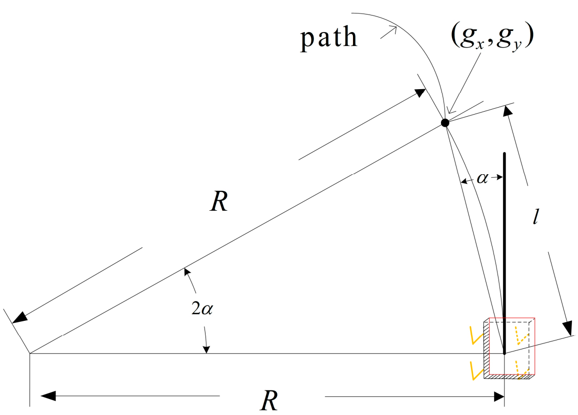


**Figure 3.** Pure pursuit for quadruped robots.

$l$ denotes the distance from the center of the robot to the target path point, and $\alpha$ represents the current vehicle posture's angle relative to the target path point. In accordance with the Law of Sines, we have the following:

$$\frac{l}{\sin(2\alpha)} = \frac{R}{\sin(\frac{\pi}{2} - \alpha)} \tag{15}$$

After derivation, the expression can be obtained as follows:

$$\begin{gathered} \frac{l}{2\sin\alpha\cos\alpha} = \frac{R}{\cos\alpha} \\ R = \frac{l}{2\sin\alpha} \end{gathered} \tag{16}$$

### 3.2. Fuzzification of Parameters

However, during the locomotion of quadruped robots, it is susceptible to road conditions and perturbations in their inherent model parameters.

In the process of trajectory tracking, if the parameters of pure pursuit are fixed, the robustness and disturbance resistance of the controller are limited, leading to a decrease in control effectiveness.

To address this issue, this paper proposes a design of an adaptive fuzzy pure pursuit (PP) controller by employing fuzzy control for online parameter adjustment.

The fuzzy control algorithm mimics human thinking by dynamically adjusting the parameters of the robot during path tracking. For example, just like a driver continuously adjusts the steering wheel based on road conditions and the vehicle's state, fuzzy control adjusts the 'look-ahead distance' of the robot based on real-time feedback. This ensures that the robot can move smoothly even in complex terrains.

The adaptive fuzzy pure pursuit (PP) controller has the capability to dynamically adjust the look-ahead distance in PP tracking based on the error between the actual and desired positions as well as the rate of change of the error. This adaptive adjustment enhances the robot's ability to track the desired trajectory. The adaptive fuzzy controller comprises fuzzification, fuzzy rule inference, and defuzzification, corresponding to input membership functions, the formulation of control rules, and logical reasoning, respectively.

This fuzzy system primarily encompasses three aspects: the fuzzification of variables, the formulation of fuzzy rules, and the defuzzification process.

In the context of pure pursuit (PP) control, the look-ahead distance $l$ can be expressed as follows [39]:

$$l = l_0 + \lambda_v k_v v^2 + \lambda_w k_w v \tag{17}$$

where $l_0$ represents the preset value of the look-ahead distance. $v$ is velocity of the robot. $v^2$ represents the squared term of the robot's forward velocity. $\lambda_v$ and $\lambda_w$ are the scope factors for these two items, respectively. $k_{v0}$, $k_{w0}$ are the original two coefficients in the look-ahead distance. $\Delta k_v$, $\Delta k_w$ are the two parameters output by the fuzzy controller. $k_v$, $k_w$ are the final parameters obtained from the adaptive pure pursuit control. The parameters for the adaptive fuzzy PP controller are determined as follows:

$$\begin{cases} k_v = k_{v0} + \Delta k_v \\ k_w = k_{w0} + \Delta k_w \end{cases} \tag{18}$$

The difference between the actual and expected trajectories $e$, along with its derivative $ec$, is obtained and then fuzzified.

Figure 4 shows the fuzzy logic diagram. Further, tuning parameters $\Delta k_v$ and $\Delta k_w$ are obtained through knowledge base and logical reasoning. Finally, $k_v$ and $k_w$ are computed using the aforementioned formulas.

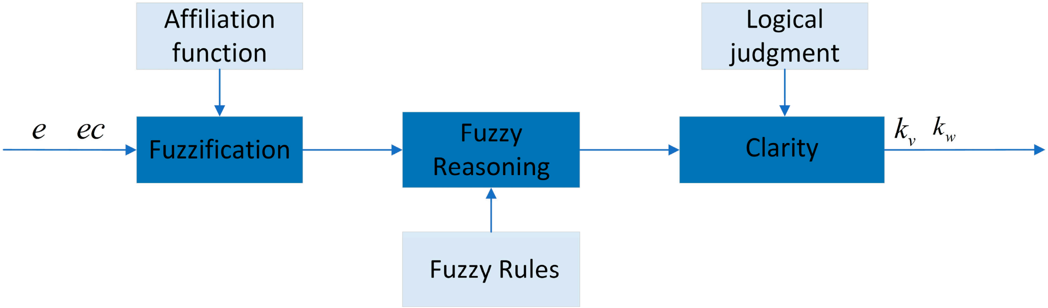


**Figure 4.** Fuzzy logic diagram.

Based on the analysis above, we establish a Mamdani-type two-dimensional fuzzy dual-input dual-output controller. The two inputs are the control variable deviation $e$ and its derivative $ec$. The input for the adaptive pure trajectory tracking controller are the tuning parameter $\Delta k_v$ and $\Delta k_w$.

The input and output universes of discourse for the adaptive PP control are as follows:

$$e, ec, \Delta k_v, \Delta k_w = \{-3, -2, -1, 0, 1, 2, 3\} \tag{19}$$

The description of the fuzzy subsets for the input variable is as follows:

$$e, ec, \Delta k_v, \Delta k_w = \{\mathrm{NB}, \mathrm{NM}, \mathrm{NS}, \mathrm{Z0}, \mathrm{PS}, \mathrm{PM}, \mathrm{PB}\} \tag{20}$$

Each is characterized by its membership function. These subsets represent different linguistic terms in the context of fuzzy control, namely negative large, negative medium, negative small, zero, positive small, positive medium, and positive large.

We choose triangular membership functions with uniform distribution, as illustrated in Figure 5.

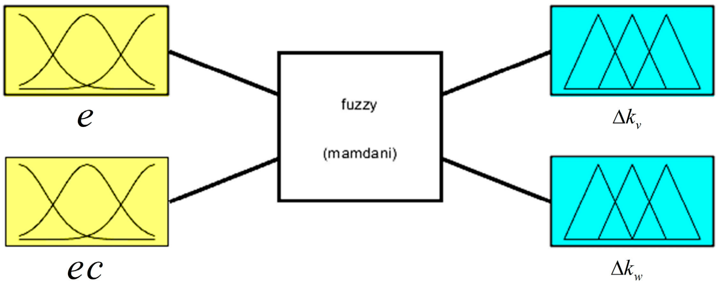


**Figure 5.** Fuzzy inputs and outputs.

The look-ahead distance not only determines the path target point and yaw but also influences the stability of the path-tracking process.

The adaptive PP controller is capable of adjusting the look-ahead distance through appropriate fuzzy rules.

The look-ahead distance of the controller is kept within the optimal range, whether on curved or straight road surfaces, leading to a significant enhancement in the tracking performance of the target trajectory by the PP algorithm.

As outlined in Equation (17), the coefficients $k_v$ and $k_w$ correspond to the squared velocity term and the linear velocity term, respectively. These coefficients play a critical role in dynamically adjusting the look-ahead distance, which constitutes the core of the fuzzy control algorithm.

Specifically, when the robot's velocity increases, the influence of the $k_v$ term becomes more pronounced, enabling more flexible adjustments to the look-ahead distance. This ensures that at higher speeds, the robot can better adapt to complex and varying terrains, thereby enhancing its path-tracking precision. Conversely, at lower speeds, the $k_w$ term ensures that the robot retains significant control over the look-ahead distance, enabling dynamic adjustments that maintain appropriate steering sensitivity, especially when approaching the target. At the same time, based on this strategy, we have also combined a lot of debugging experience and carried out many experimental adjustments, and fuzzy rules for two parameters are established as shown in the following: Tables 1 and 2.

**Table 1.** Fuzzy rules for $\Delta k_v$.

| $\boldsymbol{\Delta k_v}$ | | | | ***ec*** | | | |
|---|---|---|---|---|---|---|---|
| ***e*** | **NB** | **NM** | **NS** | **Z0** | **PS** | **PM** | **PB** |
| **NB** | PB | PB | PB | PB | PM | PS | Z0 |
| **NM** | PB | PB | PB | PB | PM | Z0 | Z0 |
| **NS** | PM | PM | PM | PM | Z0 | PS | NS |
| **Z0** | PM | PM | PS | Z0 | NS | NS | NM |
| **PS** | PS | PS | Z0 | NS | NM | NM | NM |
| **PM** | PS | Z0 | NS | NM | NM | NM | NB |
| **PB** | Z0 | Z0 | NM | NM | NM | NB | NB |

**Table 2.** Fuzzy rules for $\Delta k_w$.

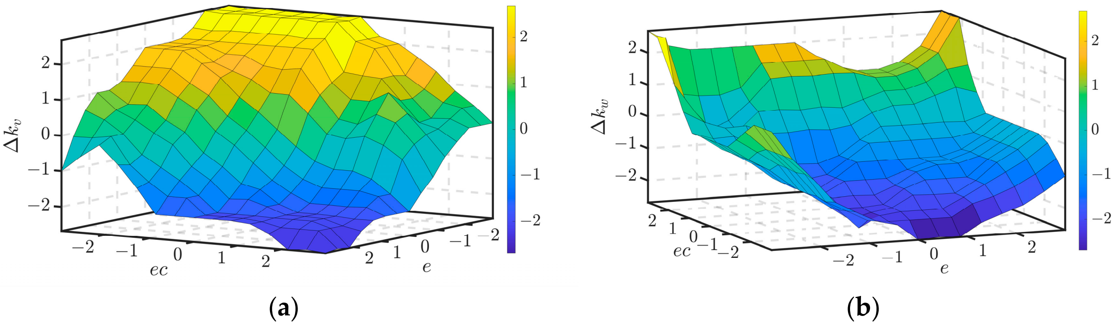
| $\boldsymbol{\Delta k_w}$ | | | | ***ec*** | | | |
|---|---|---|---|---|---|---|---|
| ***e*** | **NB** | **NM** | **NS** | **Z0** | **PS** | **PM** | **PB** |
| **NB** | PS | PS | Z0 | Z0 | Z0 | PB | PB |
| **NM** | NS | NS | NS | NS | Z0 | NS | PM |
| **NS** | NB | NB | NM | NM | NS | PS | PM |
| **Z0** | NB | NM | NM | NS | NS | NS | PM |
| **PS** | NB | NM | NS | NS | Z0 | PS | PS |
| **PM** | NM | NS | NS | NS | Z0 | PS | PS |
| **PB** | NS | Z0 | Z0 | Z0 | Z0 | PB | PB |

Utilizing the Mamdani fuzzy inference method and integrating expertise from various domains, we employ the centroid method for defuzzification, as indicated by the following formula:

$$z_0 = \frac{\sum\limits_{i=0}^{n} u_c(z_i) \times z_i}{\sum\limits_{i=0}^{n} u_c(z_i)} \tag{21}$$

where $z_0$ represents the output value after the defuzzification of the adaptive PP control, $z_i$ is the value in the domain of the fuzzy control variable, and $u_c(z_i)$ is the membership value of $z_i$.

Fuzzy control yields the control quantity surfaces for $\Delta k_v$ and $\Delta k_w$, as depicted in the accompanying figure, Figure 6:



(**a**) (**b**)

**Figure 6.** The control quantity surfaces for $\Delta k_v$ and $\Delta k_w$ are listed as (**a**) the control quantity surfaces for $\Delta k_v$ and (**b**) the control quantity surfaces for $\Delta k_w$.

Whenever the robot receives new sensor data, the fuzzy controller updates the control parameters based on predefined rules. These rules are similar to heuristic principles, adjusting parameter values according to the size of the error. Larger errors result in more significant adjustments, while smaller errors lead to more precise control.

## 4. Control Algorithm Combined MPC and WBC

In this section, we primarily delineate the motion control algorithm for the quadruped robot.

In the proposed algorithm, model predictive control (MPC) plays a crucial role in determining the optimal control inputs for the robot by predicting its future behavior based on the robot's kinematic and dynamic models. MPC calculates the desired posture and contact states of the robot, ensuring that the system follows the desired trajectory while adhering to certain constraints, such as avoiding collisions or maintaining balance.

Whole-body control (WBC), on the other hand, translates the desired posture and contact states generated by MPC into specific commands for the robot's joints. By allocating joint torques and ensuring proper coordination between the robot's limbs, WBC adjusts the robot's movements in real time, allowing it to react to external disturbances and maintain stability. The combined approach of MPC and WBC enables the robot to handle complex terrains by continuously refining its posture and movements.

Figure 7 illustrates the collaborative interaction between MPC and WBC in a quadruped robot.

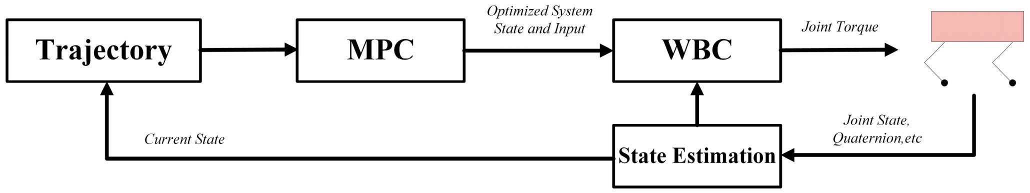


**Figure 7.** MPC–WBC framework diagram.

Following (14), the discrete dynamics of the system can be expressed as follows:

$$x(k+1) = \mathrm{A}x(k) + \mathrm{B}u(k) + \mathrm{g} \tag{22}$$

where

$$\begin{aligned}
x &= \begin{bmatrix} \Theta^{\mathrm{T}} & P^{\mathrm{T}} & \omega^{\mathrm{T}} & V^{\mathrm{T}} \end{bmatrix} \\
u &= \begin{bmatrix} f_1 & f_2 & f_3 & f_4 \end{bmatrix}^{\top} \\
\mathrm{A} &= \begin{bmatrix} 1_3 & O_3 & T_z(\varphi)\Delta t & O_3 \\ O_3 & 1_3 & O_3 & 1_3\Delta t \\ O_3 & O_3 & 1_3 & O_3 \\ O_3 & O_3 & O_3 & 1_3 \end{bmatrix} \\
\mathrm{B} &= \begin{bmatrix} O_3 & O_3 & O_3 & O_3 \\ O_3 & O_3 & O_3 & O_3 \\ \tilde{I}^{-1}[r_1]_{\times}\Delta t & \tilde{I}^{-1}[r_2]_{\times}\Delta t & \tilde{I}^{-1}[r_3]_{\times}\Delta t & \tilde{I}^{-1}[r_4]_{\times}\Delta t \\ 1_3\Delta t/m & 1_3\Delta t/m & 1_3\Delta t/m & 1_3\Delta t/m \end{bmatrix}
\end{aligned} \tag{23}$$

Assuming the system has $t_n$ discrete steps starting from $t_0$, it can be iteratively obtained based on the above discrete equation:

$$\begin{aligned} x(k+t_n) = {} & \mathrm{A}^{t_n-1}\mathrm{B}u(k+1) + \mathrm{A}^{t_n-2}\mathrm{B}u(k+1) + \cdots + \\ & \mathrm{A}\mathrm{B}u(k+1) + \mathrm{B}u(k+1) + \mathrm{A}^{t_n}x(k) \end{aligned} \tag{24}$$

The relevant state variables and control variables can be combined into the form of a state space equation and the following state equation obtained:

$$\begin{bmatrix} x(k+1) \\ x(k+2) \\ \vdots \\ x(k+t_n) \end{bmatrix} = \begin{bmatrix} \mathrm{A} \\ \mathrm{A}^2 \\ \vdots \\ \mathrm{A}^{t_n} \end{bmatrix} x(k) + \begin{bmatrix} \mathrm{B} & 0 & \cdots & 0 \\ \mathrm{AB} & \mathrm{B} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{A}^{t_n-1}\mathrm{B} & \mathrm{A}^{t_n-2}\mathrm{B} & \cdots & \mathrm{B} \end{bmatrix} \begin{bmatrix} u(k) \\ u(k+1) \\ \vdots \\ u(k+t_n-1) \end{bmatrix} \tag{25}$$

The state equation for the future h steps can be written as follows:

$$\mathrm{X} = Ax_k + BU \tag{26}$$

where

$$\begin{gathered} \mathrm{X} = \begin{bmatrix} x_{k+1} & x_{k+2} & \cdots & x_{k+h} \end{bmatrix}^T \\ \mathbf{A} = \begin{bmatrix} \mathrm{A} & \mathrm{A}^2 & \cdots & \mathrm{A}^h \end{bmatrix}^T \\ \mathbf{B} = \begin{bmatrix} \mathrm{B} & 0 & \cdots & 0 \\ \mathrm{AB} & \mathrm{B} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{A}^{h-1}\mathrm{B} & \mathrm{A}^{h-2}\mathrm{B} & \cdots & \mathrm{B} \end{bmatrix} \\ U = \begin{bmatrix} u(k) \\ u(k+1) \\ \vdots \\ u(k+t_n-1) \end{bmatrix} \end{gathered} \tag{27}$$

Assume that the matrix of differences between the state variables of the touchdown time and the state variables of the reference schedule is as follows:

$$E = \begin{bmatrix} x(k+1) - x_{ref}(k+1) \\ x(k+2) - x_{ref}(k+2) \\ x(k+3) - x_{ref}(k+3) \\ \vdots \\ x(k+N) - x_{ref}(k+t_n) \end{bmatrix} \tag{28}$$

A cost function reflecting the deviation of the target expectation from the predicted trajectory is established as follows:

$$\mathbf{J} = E^T QE + U^T WU \tag{29}$$

where $Q$ is the diagonal matrix representing the state deviation weights and $W$ is the diagonal weight matrix representing the forces.

We employ the following formula to formulate a QP problem for minimizing joint forces:

$$\min_{x,f} \sum_{i=0}^{k-1} Q^T (x_{i+1} - x_{i+1,ref})Q + W^T u(i) W \tag{30}$$

where $Q$ denotes the diagonal matrix of state bias weights and $W$ is the diagonal weight matrix representing the force.

Additionally, subject to dynamics and initial condition constraints, the friction cone constraint is defined as follows:

$$|f_x| \le \mu f_z, \left|f_y\right| \le \mu f_z, f_z > 0 \tag{31}$$

These constraints limit the minimum and maximum z-forces as well as the pyramid approximation of the friction cone.

After transformation into a QP problem, it is solved using a QP solver to obtain a set of inputs in the control time domain:

$$U = \begin{bmatrix} U_k & U_{k+1} & \cdots & U_{k+h-1} \end{bmatrix} \tag{32}$$

According to the basic principles of the model predictive controller, the first control input $f_c$ in the control time domain is applied to the system.

Utilizing the reaction forces determined by MPC, WBC computes commands for joint position, velocity, and torque.

For the computation of joint position, velocity, and acceleration, a strictly prioritized task-based inverse kinematics algorithm is employed. In the calculation of torque commands, a quadratic programming approach is utilized to determine the reaction forces, simultaneously minimizing tracking errors in acceleration and reaction force commands while satisfying the inequality constraints on the resultant reaction forces.

The desired trajectory output from the upper-level planner is received by WBC. Different priorities for various tasks and physical constraints are established, and through feedback calculations, the desired trajectory in joint space is computed by the lower-level controller. This enables the completion of diverse tracking tasks while ensuring the robot's stability.

If the number of control tasks is $i$, the workspace position of the $i$-th task is represented by $x_i$. Its Jacobian matrix and null space matrix are denoted as $J_i$ and $N_i$, respectively.

Priority increases as the value of $i$ decreases. The smaller $i$ is, the higher the priority the $i$-th task has. $\dot{\mathbf{q}}_i$ represents the joint space velocity of the first $i$-th tasks. When considering the $i$-th task, the first $i-1$ tasks are combined into $A_{i-1}$:

$$\underbrace{\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \vdots \\ \dot{x}_{i-1} \end{bmatrix}}_{\dot{x}^A_{i-1}} = \underbrace{\begin{bmatrix} J_1 \\ J_2 \\ \vdots \\ J_{i-1} \end{bmatrix}}_{J^A_{i-1}} \dot{\mathbf{q}}_{i-1} \tag{33}$$

The corresponding zero-space matrix is expressed as follows:

$$N^A_{i-1} = I - (J^A_{i-1})^+ \cdot J^A_{i-1} \tag{34}$$

where $J^+$ is defined as the right inverse matrix of $J$.

In addition, the mapping of the robot workspace velocity to the joint space velocity can be expressed as follows:

$$\dot{x}_i = J_i \dot{\mathbf{q}}_i \tag{35}$$

The equation can be transformed into the following:

$$\dot{\mathbf{q}}_i = J_i^+ \dot{x}_i + N_i \dot{\mathbf{q}}_\delta \tag{36}$$

where changing $\dot{\mathbf{q}}_\delta$ to satisfy the $i-1$ task can be introduced:

$$\dot{\mathbf{q}}_\delta = (J_i N^A_{i-1})^+ \cdot (\dot{\mathbf{q}}_i - J_i \dot{\mathbf{q}}_{i-1}) \tag{37}$$

The final expression for $\dot{\mathbf{q}}_i$ can be derived:

$$\dot{\mathbf{q}}_i = \dot{\mathbf{q}}_{i-1} + (J_i N^A_{i-1})^+ \cdot (\dot{x}_i - J_i \dot{\mathbf{q}}_{i-1}) \tag{38}$$

Similarly, position deviation and acceleration can be derived:

$$\begin{aligned} \Delta\mathbf{q}_i &= \Delta\mathbf{q}_{i-1} + (J_i N_{i-1}^A)^+ \cdot (e - J_i \Delta\mathbf{q}_{i-1}) \\ \ddot{\mathbf{q}}_i &= \ddot{\mathbf{q}}_{i-1} + (J_i N_{i-1}^A)^+ \cdot (\ddot{x}_i - \dot{J}_i \dot{\mathbf{q}} - J_i \ddot{\mathbf{q}}_{i-1}) \end{aligned} \tag{39}$$

where $e$ represents workspace position tracking error and $\Delta\mathbf{q}$ represents joint space position tracking error. By inputting initial values, the position, velocity, and acceleration of the robot in joint space can be obtained.

In the context of whole-body control for the quadruped robot described in this paper, the overall control task is divided into four sub-tasks, prioritized from high to low as Table 3:

**Table 3.** Priority of four sub-tasks.

| Priority | Task |
|---|---|
| 0 | No motion at the contact points |
| 1 | Body rotation control |
| 2 | Body lateral motion control |
| 3 | Swing leg foot trajectory tracking |

Stable and reliable contact between the supporting leg and the ground is a prerequisite for utilizing other control algorithms, and the supporting leg trajectory tracking task holds the highest priority.

It is an important control objective for the body to remain smooth during walking. Therefore, body rotation control and body lateral motion control are listed in priority 2 and 3. The robot's interaction with the external environment is primarily dependent on the contact leg. The trajectory following the control of the swing leg is placed at the lowest priority.

The pseudocode for the recursive whole-body control of the quadruped robot is presented as follows:

$$\begin{aligned} &\Delta\mathbf{q}_1^{cmd} = 0, \dot{\mathbf{q}}_1^{cmd} = 0 \\ &\ddot{\mathbf{q}}_1^{cmd} = J_1^+(-\dot{J}_1\dot{\mathbf{q}}) \\ &for\ i = 2\ to\ 4\ do \\ &\quad \ddot{x}_i^c = \ddot{x}_i^d + K_p^i(x_i^d - x_i) + K_d^i(\dot{x}_i^d - \dot{x}_i) \\ &\quad \Delta\mathbf{q}_i^{cmd} = \Delta\mathbf{q}_{i-1}^{cmd} + (J_i N_{i-1}^A)^+ \cdot (e_i - J_i \Delta\mathbf{q}_{i-1}) \\ &\quad \dot{\mathbf{q}}_i^{cmd} = \dot{\mathbf{q}}_{i-1}^{cmd} + (J_i N_{i-1}^A)^+ \cdot (\dot{x}_i^d - J_i \dot{\mathbf{q}}_{i-1}^{cmd}) \\ &\quad \ddot{\mathbf{q}}_i = \ddot{\mathbf{q}}_{i-1}^{cmd} + (J_i N_{i-1}^A)^+ \cdot (\ddot{x}_i^{cmd} - \dot{J}_i \dot{\mathbf{q}} - J_i \ddot{\mathbf{q}}_{i-1}^{cmd}) \\ &end \end{aligned} \tag{40}$$

where $K_p^i$ and $K_d^i$ are the position and velocity feedback gains for the corresponding control tasks, and all the control tasks' workspaces are described in the world frame. In the above equation, the expected joint space position $\Delta\mathbf{q}^{cmd}$, velocity $\dot{\mathbf{q}}^{cmd}$, and acceleration $\ddot{\mathbf{q}}^{cmd}$ can be represented as follows:

$$\begin{aligned} \mathbf{q}^{cmd} &= \mathbf{q} + \Delta\mathbf{q}_4^{cmd} \\ \dot{\mathbf{q}}^{cmd} &= \dot{\mathbf{q}}_4^{cmd} \\ \ddot{\mathbf{q}}^{cmd} &= \ddot{\mathbf{q}}_4^{cmd} \end{aligned} \tag{41}$$

The dynamic equation of the floating base robot in the joint space usually has the following standard form:

$$M(q)\begin{bmatrix} \ddot{q}_f \\ \ddot{q}_j \end{bmatrix} + C(q, \dot{q}) + G(q) = \begin{bmatrix} 0_6 \\ \tau_j \end{bmatrix} + J_c^T f_c \tag{42}$$

where $\ddot{q}_f \in \mathrm{R}^6$ is the acceleration of the floating base, $\ddot{q}_j \in \mathrm{R}^{12}$ is the vector of joint accelerations, $M(q)$ is the mass matrix, $C(q, \dot{q})$ is the Coriolis and centrifugal matrix, $G(q)$ is the gravity vector, $\tau_j$ is the vector of actuated joint torques, $J_c$ is the Jacobian matrix, and $f_c$ is the vector of external forces, which is the matrix solved in Equation (32) above.

$$\begin{bmatrix} 0_{6\times 1} \\ \tau_j \end{bmatrix} = M(q) \begin{bmatrix} \ddot{q}_f \\ \ddot{q}_j \end{bmatrix} + C(q, \dot{q}) + G(q) - J_c^T f_c \tag{43}$$

The final output joint torque $\tau_f$ can be determined by the following expression:

$$\tau_f = \tau_j + k_p(q_j^{cmd} - q_j) + k_d(\dot{q}_j^{cmd} - \dot{q}_j) \tag{44}$$

where $k_p$ and $k_d$ are parameters for the joint PD controller.

## 5. Results

This section aims to validate the robustness, real-time performance, and disturbance rejection capabilities of the fuzzy adaptive trajectory tracking control algorithm. First, we verify its performance in ROS Gazebo. Then, we conduct external experiments, comparing the proposed algorithm with a pure trajectory tracking algorithm without fuzzy. Finally, we analyze the experimental results, demonstrating the feasibility and effectiveness of the proposed algorithm.

### *5.1. Simulation Experiments*

The simulation experiments are conducted on ROS Gazebo.

To test the feasibility and effectiveness of the algorithm, we conducted simulation experiments with a quadruped robot in Gazebo, as shown in Figure 8. The simulations were performed on two different types of terrains to simulate the complex surfaces the robot would encounter in real-world conditions. Both terrains were randomly generated undulating grounds, which are representative of moderately uneven terrains that test the robot's walking stability and control precision.

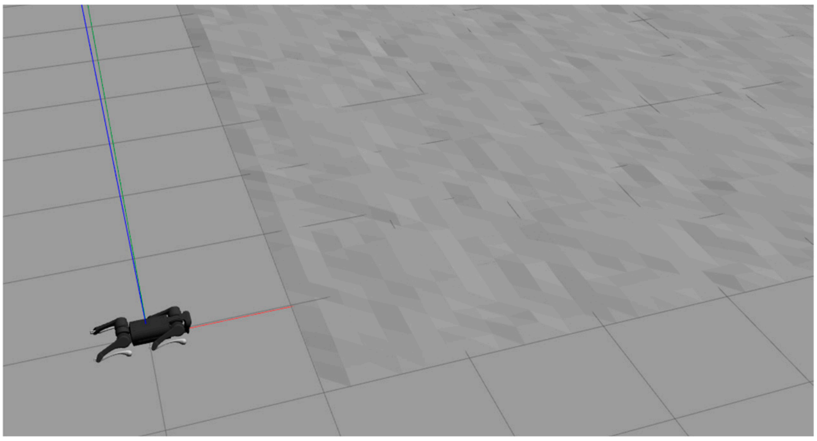


**Figure 8.** Simulation in Gazebo.

For all experiments, the look-ahead distance was initially set to 2 and dynamically adjusted using a fuzzy control algorithm. The look-ahead distance was adapted in real time based on the robot's tracking error and the rate of change in the error. This allowed the algorithm to optimize path tracking, ensuring that the robot smoothly followed the desired trajectory while reacting to variations in the terrain.

We designed three key simulation experiments:

In the first experiment, the robot walked in a straight line from point (0,0) to point (10,10) on a terrain with an average undulation of 3 cm. The purpose of this experiment was to assess the robot's tracking accuracy and control stability on relatively flat but uneven ground.

In the second experiment, the robot again walked in a straight line from (0,0) to (10,10), but this time, the terrain had an average undulation of 6 cm, representing a more challenging environment. This experiment tested the algorithm's ability to handle more significant terrain variations while maintaining stable walking.

In the third experiment, the robot followed a circular trajectory with a radius of 5 m, starting from (0,0) on a terrain with a 6 cm undulation. This experiment was designed to evaluate the robot's capability to perform continuous turns while navigating over rough ground, a more complex task than straight-line walking.

In all experiments, the robot's tracking speed was set to 0.6 m/s, which was chosen to ensure that the robot moved at a controlled pace, allowing the algorithm to adjust the look-ahead distance and correct any deviations in real time. The terrain undulations were selected to simulate realistic surface variations the robot may encounter in real-world scenarios, such as outdoor environments with uneven ground.

The chosen simulation parameters and their configurations ensure that the robot's control algorithm is tested thoroughly for different types of movements (straight-line and circular trajectories) and varying terrain complexities. This setup provides a comprehensive evaluation of the algorithm's robustness and adaptability. The parameters of the quadruped robot are shown in the following Table 4:

**Table 4.** Parameters of the quadruped robot.

| Variable/Parameters | Symbols | Value |
|---|---|---|
| Body mass | $m$ | 12.786 kg |
| Body length | $B_L$ | 0.267 m |
| Body width | $B_W$ | 0.194 m |
| Body height | $B_H$ | 0.114 m |
| Ab/Ad joint length | $L_1$ | 0.075 m |
| Hip joint length | $L_2$ | 0.22 m |
| Knee joint length | $L_3$ | 0.22 m |

The robot is shown as Figure 9 below:

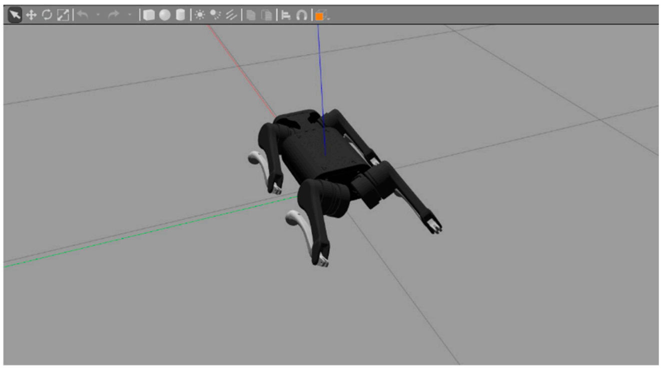


**Figure 9.** Quadruped robot used for simulation.

### 5.1.1. Ground Undulation 3 cm Test

As shown in Figure 10, this experiment tested the fuzzy pure trajectory tracking algorithm on a quadruped robot on an uneven road surface, with the purpose of verifying the algorithm's tracking quality and steadiness. In the experiment, the robot's desired trajectory was a linear path going from (0,0) to (10,10); the gait was a trot, and the desired walking velocity was 0.6 m/s.

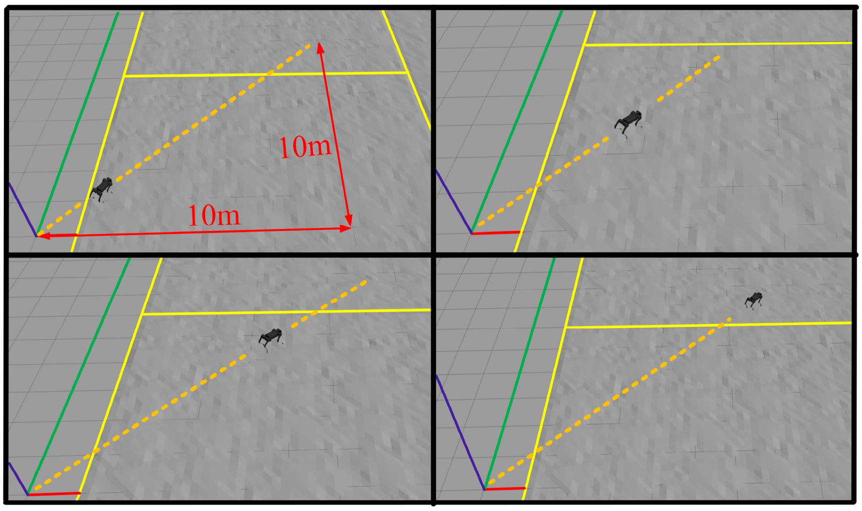


**Figure 10.** Ground undulation 3 cm test (walk in line).

This experiment recorded the robot's trotting ability on a low-undulating terrain. To demonstrate the quadruped robot's walking ability in complex environments, this paper collected and analyzed the experimental data from the following aspects: actual and desired trajectory tracking performance; position and yaw angle errors in $x$, $y$ directions; three joint torques of one leg; fuzzy parameters $k_v$ and $k_w$; and a look-ahead distance $l$ of pure trajectory tracking. The first two aspects reflect the accuracy of the quadruped robot's trajectory tracking, the third aspect reflects the robot's own control ability, and the last two aspects illustrate the effectiveness of the fuzzy control algorithm.

As shown in Figure 11, the actual trajectory of the robot is very close to the desired trajectory, indicating that the robot can track the desired trajectory well. This demonstrates that the algorithm has high tracking accuracy.

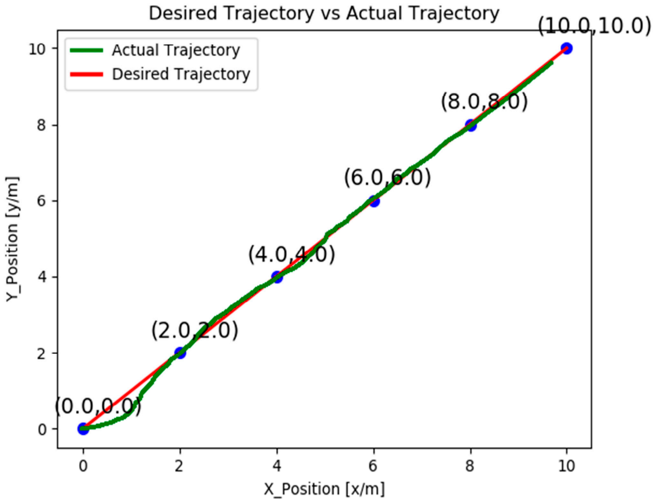


**Figure 11.** Robot desired trajectory vs. actual trajectory (3 cm, line).

Figure 12 reveals that the position errors and yaw angle errors of the robot in the $x$ and $y$ directions are very small, and they do not diverge over time, indicating that the robot can effectively suppress external disturbances.

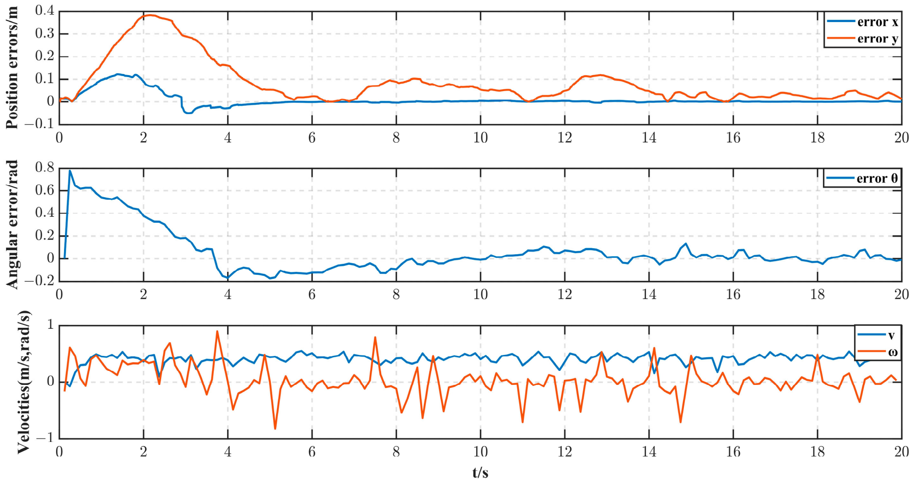


**Figure 12.** Position errors, angular error, and velocities during robot walking (3 cm, line).

In terms of velocity and angular velocity, the quadruped robot is able to keep walking forward at the desired speed. The angular velocity fluctuated slightly due to the uneven ground, but it was still within the acceptable range for the experiment. The effectiveness of the algorithm can be well illustrated.

As shown in Figure 13, the three joint torques of one leg of the robot are within a reasonable range, and there are no sudden or excessive phenomena, indicating that the robot's motion is smooth.

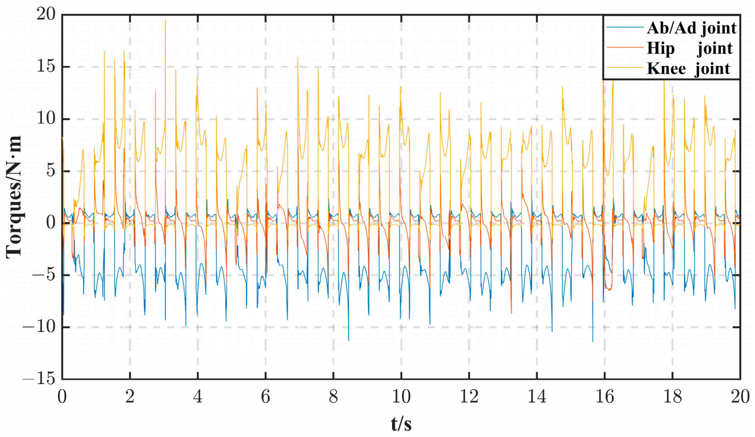


**Figure 13.** Three joint torques of one leg (3 cm, line).

By observing Figure 14a, we can observe that the two fuzzy parameters $k_v$ and $k_w$ vary dynamically with the robot's motion, and they can adaptively adjust according to different motion states.

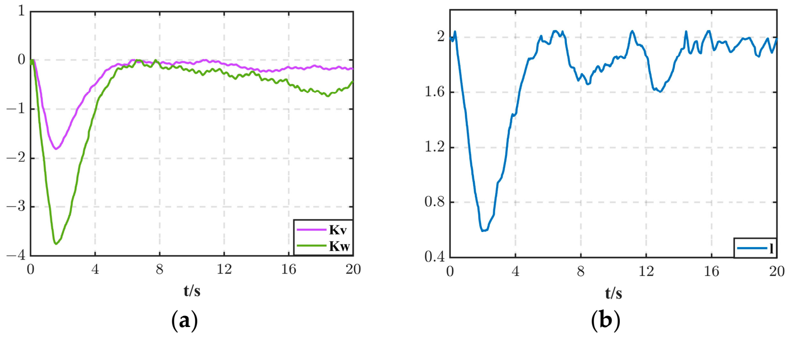


**Figure 14.** (**a**) Value of $k_v$ and $k_w$; (**b**) value of $l$.

As shown in Figure 14b, the look-ahead distance $l$ of pure trajectory tracking also varies dynamically with the robot's motion, and it can adaptively adjust according to different road conditions.

In conclusion, this experiment verified the feasibility and effectiveness of the fuzzy pure trajectory tracking algorithm on the quadruped robot, showing that the algorithm can enable the robot to have a high tracking accuracy on complex terrain. It also boosted the robustness and dynamics of the quadruped robot, endowing its trajectory tracking ability with more environmental adaptability.

### 5.1.2. Ground Undulation 6 cm Test

We raised the terrain undulation of the quadruped robot's movement from 3 cm to 6 cm based on the previous experiment, with the purpose of further testing the quality and steadiness of the fuzzy pure trajectory tracking algorithm on more complex terrains.

In the second experiment, as shown in Figure 15, the robot's expected trajectory, gait, and walking speed were the same as in the previous experiment. The walking desired curve was set as a straight line extending from (0,0) to (10,10), the desired velocity was 0.6 m/s, and the gait was in trot gait. The results are as follows:

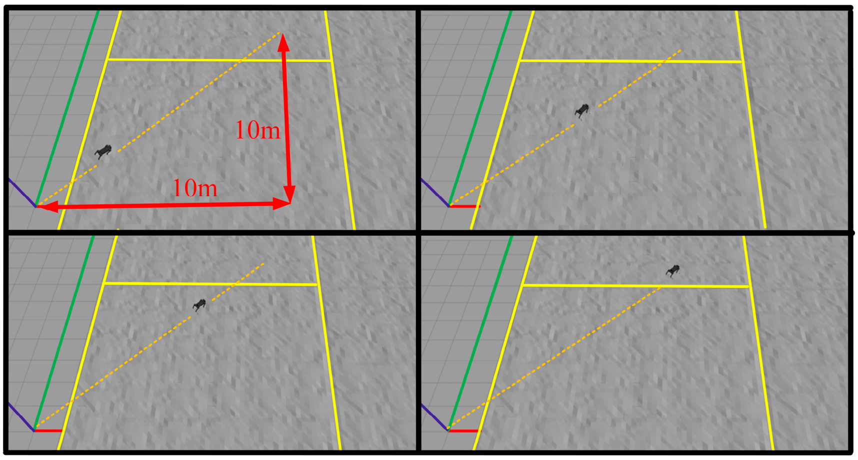


**Figure 15.** Ground undulation 6 cm test (work in line).

As shown in Figure 16, the actual trajectory of the robot is still very close to the desired trajectory. Considering that the experiment uses a bottom surface with 6 cm of undulation, it can be shown that the algorithm has good tracking performance for the desired trajectory even on bumpier roads.

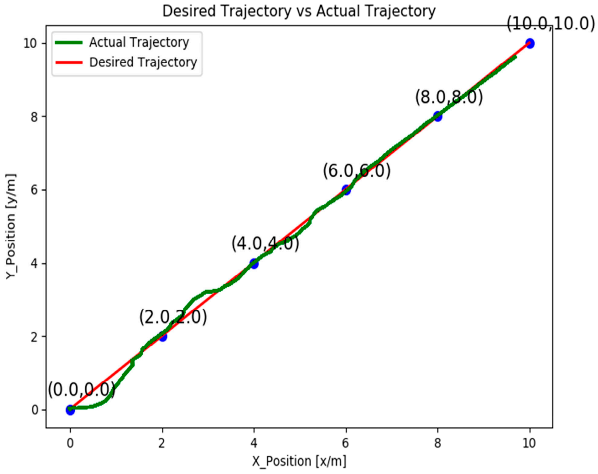


**Figure 16.** Robot desired trajectory vs. actual trajectory (6 cm, line).

As can be seen in Figure 17, the position and yaw angle errors increase slightly under the influence of 6 cm bumpy terrain compared to 3 cm of road surface. The robot is subjected to more random perturbations during the walking process but still manages to maintain good stability at the macro level. In terms of velocity and angular velocity, the fluctuations of both increased slightly from previous experiments; nevertheless, they were able to stay within the desired range.

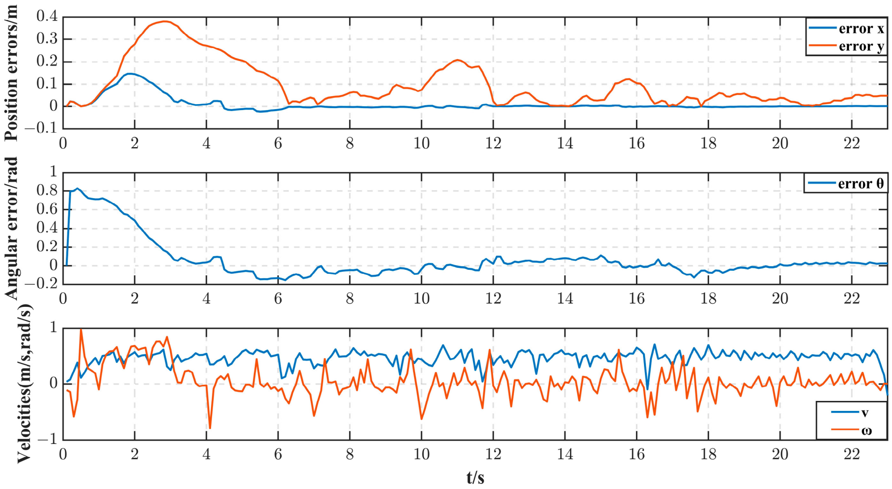


**Figure 17.** Position errors, angular error, and velocities during robot walking (6 cm, line).

Figure 18 shows that the three joint torques of one leg are still within a reasonable range, and there is no sudden or excessive phenomenon, indicating that the robot's motion is stable.

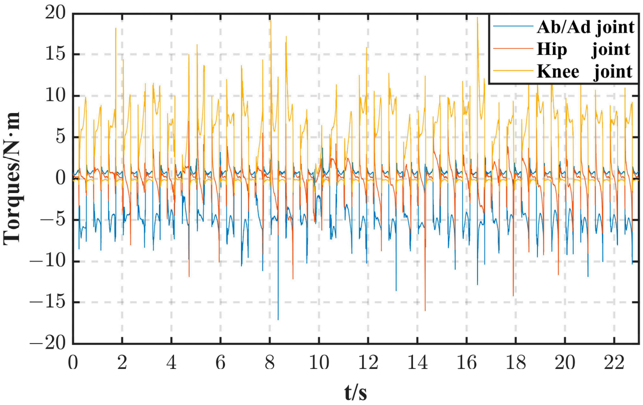


**Figure 18.** Three joint torques of one leg (6 cm, line).

The experiment shows the two fuzzy parameters $k_v$, $k_w$ and the look-ahead distance $l$ in Figure 19. The slightly larger range of variation of these parameters compared to previous experiments means that they can be adjusted in real time as the robot walks in the face of bumpier terrain, enhancing the real-time nature of the control system.

This test confirms the feasibility and effectiveness of the fuzzy pure pursuit tracking algorithm on quadruped robots. The algorithm allows the robot to track the trajectory with high robustness and adaptability on complex terrains. The experiment also demonstrates the resilience and flexibility to the variations of the road surface compared with the previous experiment.

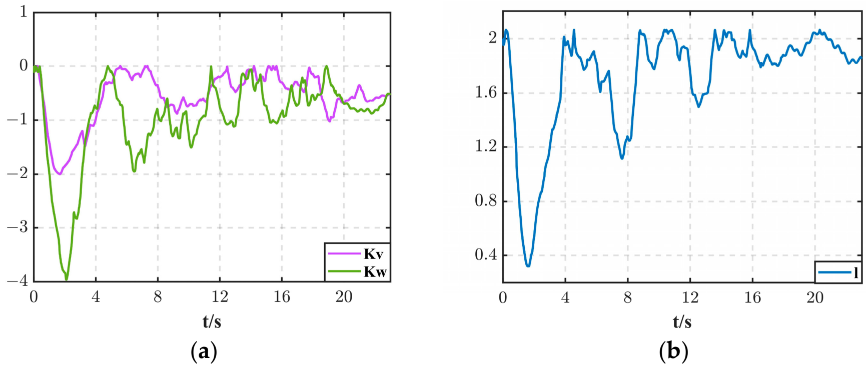


**Figure 19.** (**a**) Value of $k_v$ and $k_w$; (**b**) value of $l$.

### 5.1.3. Robot Circle Walking

To verify the feasibility and stability of the algorithm on more complex trajectories, we set the desired trajectory of the robot's walking to a circle with a radius of 5 m, as shown in Figure 20. The robot's gait and speed were kept the same as in the last two experiments, and the road surface undulation was set to 6 cm.

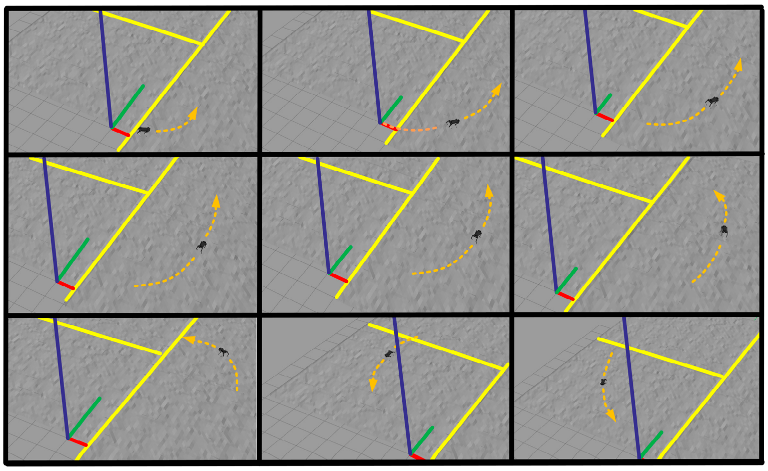


**Figure 20.** Ground undulation 6 cm test (walk in circle).

Figure 21 shows the expected trajectory versus the actual trajectory of the quadrupedal robot when traversing the circular path. During the walking process, the quadrupedal robot was challenged by the fact that the circular trajectory required the robot to continuously steer during the walking process, and the terrain undulation was set to 6 cm. As can be seen from the figure, although the robot understeered or oversteered several times during the steering process, it was always able to move around the desired trajectory until the task was completed.

The quadrupedal robot took 62 s to complete the circular trajectory, and as can be seen in Figure 22, the robot kept correcting its own trajectory during the movement. The positional and angular errors fluctuated within a certain range due to the undulations of the terrain, but the robot was still thus able to follow the intended trajectory at the desired velocity.

Figures 23 and 24 show that during the turning process of the quadruped robot, the torque is kept within a suitable range, the fuzzy parameters are changed in real time according to the error, and the look-ahead distance is adjusted to complete the steering.

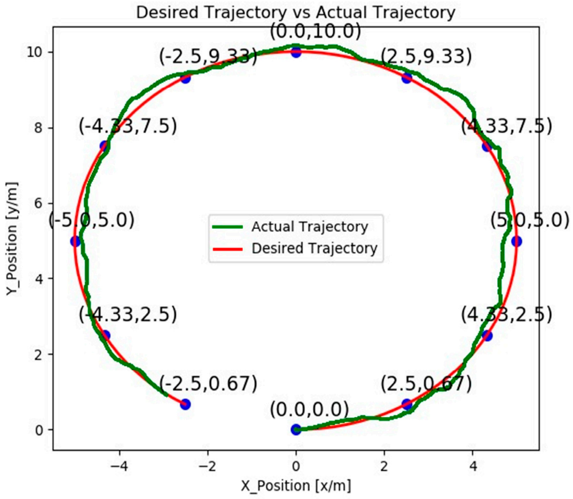


**Figure 21.** Robot desired trajectory vs. actual trajectory (6 cm, circle).

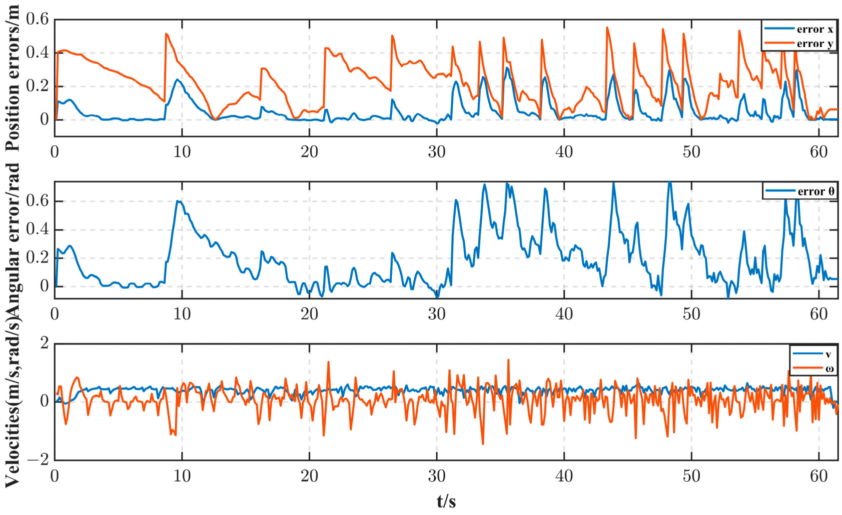


**Figure 22.** Position errors, angular error, and velocities during robot walking (6 cm, circle).

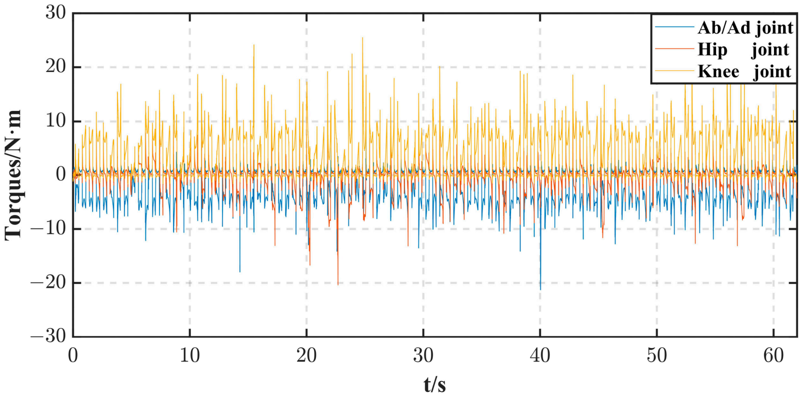
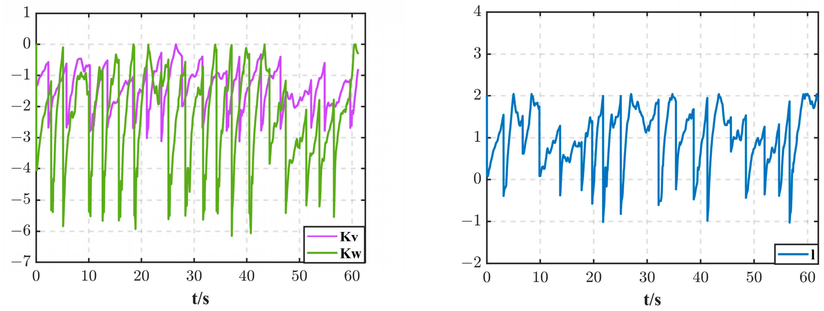


**Figure 23.** Three joint torques of one leg (6 cm, circle).

Compared to straight trajectories, quadruped robots face a greater difficulty in following circular trajectories, which are longer and require constant turning, and the robot inevitably faces curvature-varying turning when on uneven ground. In this condition, if the look-ahead distance is fixed, it will greatly affect the steering performance of the quadruped robot. The look-ahead distance under fuzzy adaptive trajectory changes in real time during steering so that the quadruped robot can complete the steering in a smoother way in order to better complete the tracking of the circular trajectory.



(**a**) (**b**)

**Figure 24.** (**a**) Value of $k_v$ and $k_w$; (**b**) value of $l$.

All three simulation experiments have been completed. In the simulation experiments, we tested the walking ability of the quadrupedal robot in two types of terrain, 3 cm and 6 cm, and two types of trajectories, straight line and circle. The data show that the quadruped robot can track the preset trajectory precisely and complete the walking experiments without any deviation or excessive torque, which verifies the effectiveness and stability of the algorithm.

After completing the simulation experiment, we will further verify it in the following physical experiment.

### 5.2. USLGO1 Walking

We constructed a medium-sized quadruped robot named USLGO1, as shown in Figure 25. The robot has a Jetson as the control core and a motor drive board as the drive unit. The robot is also equipped with an IMU, lidar, and depth camera to monitor the robot's performance. In addition, a dedicated power board is responsible for powering each module.

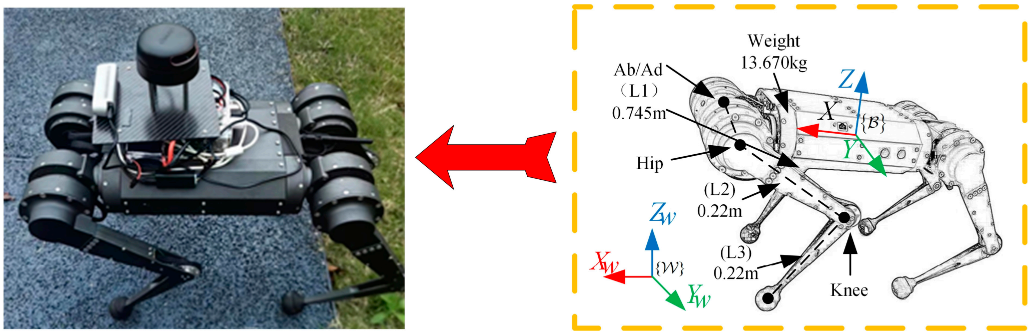


**Figure 25.** USLGO1.

We conducted a test of the fuzzy pure pursuit tracking algorithm on an actual quadruped robot in an outdoor environment. Corresponding to the simulation, the USLGO1 quadrupedal robot was set to walk on a desired path, during which the walking data were collected by built-in sensors and the desired speed of the robot was set to 0.5 m/s.

For the terrain, we chose an open grassy field and arranged a complex S-curve for the robot as the desired trajectory. As shown in Figure 26, we divided this trajectory into four parts, where section (a) is 15° uphill, sections (b,c) are dirt roads with a maximum undulation of 4.5 cm, and section (d) is 20° downhill. This trajectory has a long distance, both straight and steering sections, and complex terrain, which can accurately reflect the comprehensive walking ability of the quadruped robot.

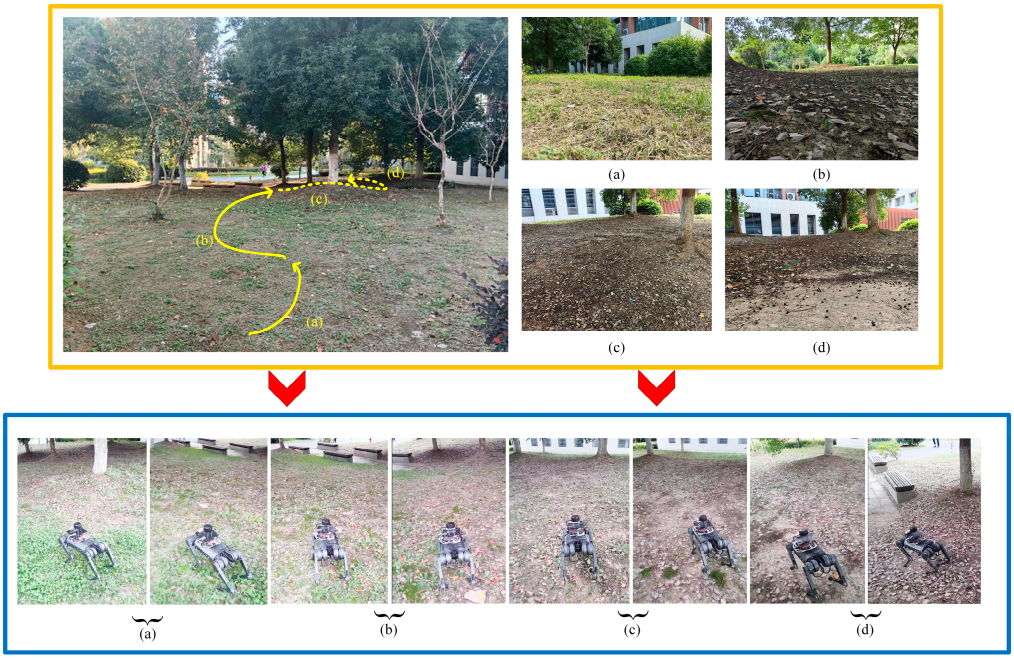


**Figure 26.** Demonstration of experimental robot walking routes in complex outdoor environments. (**a**–**d**) represent the 4 parts of the S-curve respectively.

We conducted three sets of comparative trials on the same desired trajectory and terrain. The first test utilized a fuzzy pure trajectory tracking algorithm that merged MPC and WBC. The second test employed the identical algorithm but without the fuzzy adaptation of the parameters. The third test exclusively used MPC without WBC for the optimization of motion control. The video of the robot walking was posted at https://www.youtube.com/watch?v=hZIiUEAhQFc (accessed on 14 August 2024). The data shown in Figure 27 present the results of the comparative experiments:

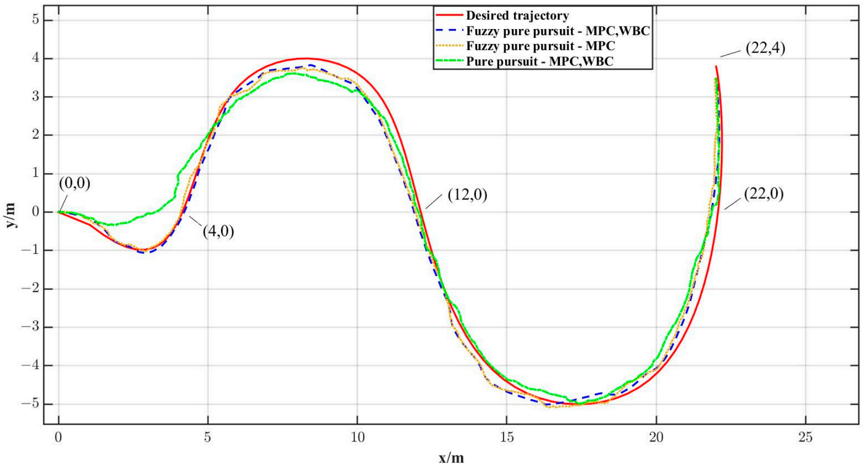


**Figure 27.** Comparison of real-world robot trajectories with different algorithms.

The tracking error is calculated based on the target position error in the tracking framework. In the tracking algorithm, the target position is predicted based on the tracking error framework, and the distance between the predicted position and the actual target position is taken as the tracking error. In specific implementations, the tracking error is usually defined as the Euclidean distance between the predicted position of the tracking error frame and the actual target position:

$$error_i = \sqrt{(x_{pre,i} - x_i)^2 + (y_{pre,i} - y_i)^2} \tag{45}$$

The tracking error can be evaluated by calculating the average error within the number of frames tracked by the target to assess the accuracy of the tracking algorithm.

$$\eta = \frac{1}{N_e}\sum_{i=1}^{N}|error_i| \tag{46}$$

In addition to the mean error, the stability of the tracking algorithm can be evaluated using the standard deviation of the error calculated based on the number of frames tracked. The standard deviation indicates the extent to which a set of data deviates from the mean and reflects the accuracy of the tracking. The standard deviation of the tracking error is calculated using the following formula:

$$\sigma = \sqrt{\frac{\sum\limits_{i=1}^{N} error_i^2 - \eta^2 N_e}{N_e - 1}} \tag{47}$$

Based on the above three sets of experiments comparing the error, mean error, standard deviation, and completion time of the trajectories in question, the relevant data are shown in the Table 5 below (where fuzzy pure pursuit -MPC, WBC is simplified as FpMW; fuzzy pure pursuit -MPC is simplified as FpM; pure pursuit -MPC, WBC simplifies to pMW):

**Table 5.** Comparison of three algorithms in mean error, standard deviation, and completion time.

| | **FpMW** | **FpM** | **pMW** |
|---|---|---|---|
| $\eta$ | 0.2801 | 0.2933 | 0.5133 |
| $\sigma$ | 0.1957 | 0.1981 | 0.3822 |
| Completion time/s | 64 | 68 | 75 |

As can be seen in Table 5, of the three proposed trajectory tracking metrics, FpMW used 64 s to complete the experiment, outperforming the other two algorithms in terms of time. FpMW also exhibits the lowest values under mean error and standard deviation, proving the effectiveness of the algorithm.

As shown in Figure 28, it can be seen that the quadruped robot has more obvious error fluctuations in the uphill and downhill and steering phases, but it is also able to track the desired trajectory, and the tracking effect is the best when it enters the straight line part. In terms of angular tracking, the robot has a large angular error at the beginning of the uphill stage, which is related to the undulation of the terrain during the experiment. After adjustments, the robot regains the stability of angular tracking. In terms of walking velocities, the robot has been able to maintain the desired speed for stable walking. Figure 29a,b shows the variation of leg triple-joint moments and look-ahead distance during the walking process of the quadruped robot.

In summary, in the physical experiments, comparing the other two algorithms, in terms of trajectory tracking accuracy and the time used to complete the trajectory, the algorithm in this paper, after combining the use of the MPC algorithm and WBC algorithm as well as the fuzzy algorithm to optimize the parameters online, shows the best trajectory tracking accuracy in three experiments and also the algorithm that takes the least time to complete the trajectory, which proves that the algorithm is able to track the complex trajectory very well, proving the effectiveness and stability of the algorithm.

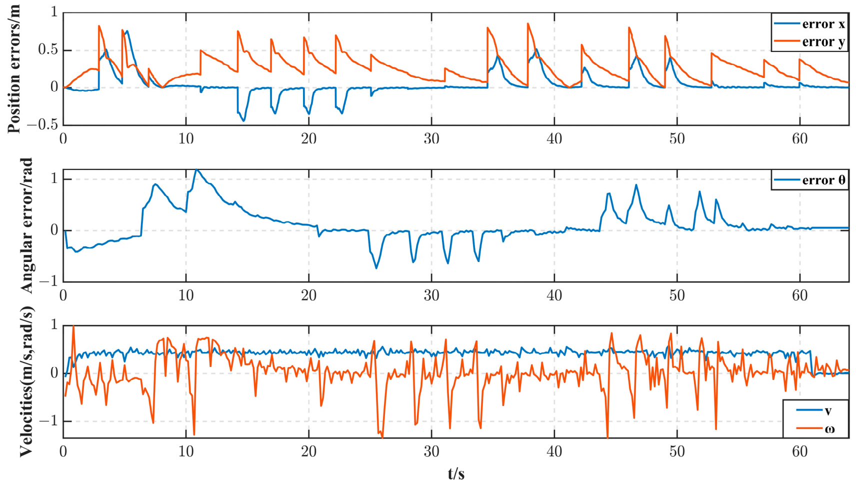
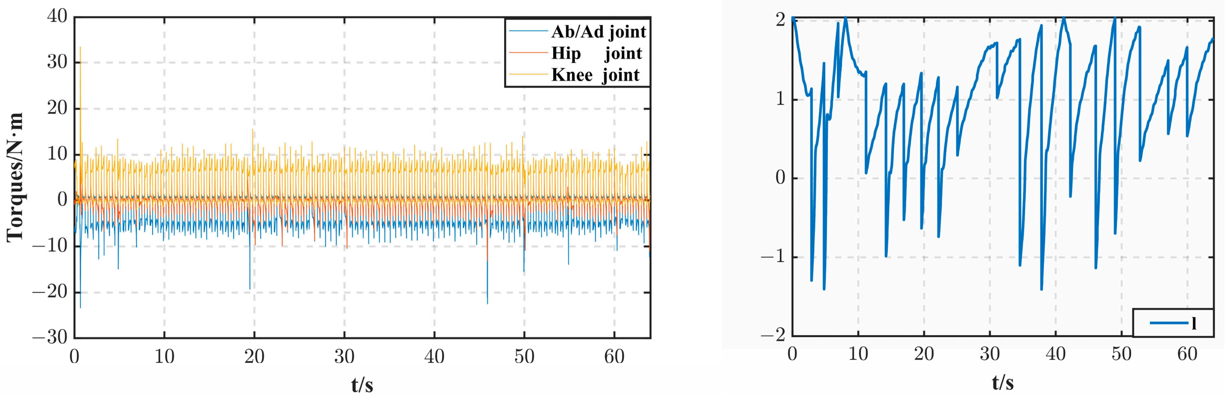


**Figure 28.** Position errors, angular error, and velocities during robot walking (in real situation).



(**a**) (**b**)

**Figure 29.** (**a**) Three joint torques of one leg (in real situation); (**b**) value of *l*.

## 6. Conclusions

This study focuses on the path-tracking problem for unmanned ground vehicles (UGVs), particularly legged robots, emphasizing the autonomous movement and navigation of quadrupedal robots in complex environments. By adopting a pure path-tracking algorithm, we successfully achieved trajectory tracking for specified paths. We acknowledge that the pure path-tracking algorithm is highly sensitive to the choice of look-ahead distance. To address this, we introduced a fuzzy controller to dynamically adjust the look-ahead distance, enhancing the accuracy and robustness of path tracking.

Despite these advancements, quadrupedal UGVs still face challenges in path tracking when traversing uneven terrains. To overcome this limitation, we explored a hybrid algorithm that combines model predictive control (MPC) with whole-body control (WBC). MPC optimally computes control inputs based on the robot's kinematic or dynamic model and constraints, while WBC allocates joint torques in task space. Integrating MPC and WBC enables quadrupedal UGVs to walk on complex surfaces, improving stability and disturbance rejection.

Our research findings were validated through both simulation experiments and real-world outdoor tests. In the simulation environment, we successfully tracked straight lines and circular trajectories, achieving the desired path-tracking performance. Furthermore, in practical outdoor scenarios, our algorithm demonstrated robustness and stability even on uneven ground, effectively following expected trajectories.

The algorithm presented in this paper, while effective in following predetermined trajectories, lacks the capacity for real-time adaptive obstacle avoidance when confronted with dynamic environments. In scenarios where obstacles are encountered during movement, the incorporation of more advanced path-planning algorithms will be essential to achieve flexible and real-time obstacle avoidance. Moreover, although the algorithm's feasibility has been demonstrated in controlled, non-extreme environments, real-world applications often involve extreme conditions such as snow-covered or icy surfaces, which introduce significant challenges to the robot's control mechanisms. Addressing these challenges requires further research. The novel bioinspired motor neuron dynamical unit introduced in [40], where neural dynamics are embedded into the actuator dynamics, replacing the recovery variable in traditional neuron equations, provides a new approach for our research.

Looking forward, the proposed path-tracking algorithm is not confined to quadrupedal UGVs; it can also be extended to other robotic platforms, including wheeled UGVs, drones, and autonomous vehicles. This broader applicability provides a robust foundation for advancing autonomous navigation and obstacle avoidance across diverse robotic platforms. Future research will continue to explore these avenues, aiming to refine and optimize the algorithm through comprehensive experimentation in increasingly complex environments.

In summary, our study provides an effective and stable solution to the path-tracking problem in autonomous UGVs and other robotic platforms, offering valuable insights for the development and application of intelligent robots in the future.

**Author Contributions:** Conceptualization, Y.S.; methodology, Y.S.; software, Y.Y.; validation, H.Z.; formal analysis, Y.S.; data curation, Y.Y. and W.Q.; writing—original draft, Y.S.; writing—review & editing, Z.Y.; project administration, N.H. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Data Availability Statement:** Data are contained within the article.

**Conflicts of Interest:** Author Haoze Zhuo was employed by the company Electric Power Research Institute of Guangxi Power Grid Co., Ltd., The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

## References

1. Raibert, M.; Blankespoor, K.; Nelson, G.; Playter, R. Bigdog, the rough-terrain quadruped robot. *IFAC Proc. Vol.* **2008**, *41*, 10822–10825. [CrossRef]
2. Meng, X.; Wang, S.; Cao, Z.; Zhang, L. A review of quadruped robots and environment perception. In Proceedings of the 2016 35th Chinese Control Conference (CCC), Chengdu, China, 27–29 July 2016; pp. 6350–6356.
3. Katz, B.; Di Carlo, J.; Kim, S. Mini cheetah: A platform for pushing the limits of dynamic quadruped control. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 6295–6301.
4. Ji, Y.; Margolis, G.B.; Agrawal, P. Dribblebot: Dynamic legged manipulation in the wild. In Proceedings of the 2023 IEEE International Conference on Robotics and Automation (ICRA), London, UK, 29 May–2 June 2023; pp. 5155–5162.
5. Hwangbo, J.; Lee, J.; Dosovitskiy, A.; Bellicoso, D.; Tsounis, V.; Koltun, V.; Hutter, M. Learning agile and dynamic motor skills for legged robots. *Sci. Robot.* **2019**, *4*, eaau5872. [CrossRef] [PubMed]
6. Park, H.-W.; Wensing, P.M.; Kim, S. Jumping over obstacles with MIT Cheetah 2. *Robot. Auton. Syst.* **2021**, *136*, 103703. [CrossRef]
7. Caluwaerts, K.; Iscen, A.; Kew, J.C.; Yu, W.; Zhang, T.; Freeman, D.; Lee, K.-H.; Lee, L.; Saliceti, S.; Zhuang, V. Barkour: Benchmarking animal-level agility with quadruped robots. *arXiv* **2023**, arXiv:2305.14654.
8. Yang, Y.; Norby, J.; Yim, J.K.; Johnson, A.M. Proprioception and tail control enable extreme terrain traversal by quadruped robots. In Proceedings of the 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Detroit, MI, USA, 1–5 October 2023; pp. 735–742.
9. Choi, S.; Ji, G.; Park, J.; Kim, H.; Mun, J.; Lee, J.H.; Hwangbo, J. Learning quadrupedal locomotion on deformable terrain. *Sci. Robot.* **2023**, *8*, eade2256. [CrossRef] [PubMed]
10. Patané, L. Bio-Inspired Robotic Solutions for Landslide Monitoring. *Energies* **2019**, *12*, 1256. [CrossRef]
11. Arena, P.; Patané, L.; Taffara, S. Learning risk-mediated traversability maps in unstructured terrains navigation through robot-oriented models. *Inf. Sci.* **2021**, *576*, 1–23. [CrossRef]
12. Ji, G.; Mun, J.; Kim, H.; Hwangbo, J. Concurrent training of a control policy and a state estimator for dynamic and robust legged locomotion. *IEEE Robot. Autom. Lett.* **2022**, *7*, 4630–4637. [CrossRef]
13. Rudin, N.; Kolvenbach, H.; Tsounis, V.; Hutter, M. Cat-like jumping and landing of legged robots in low gravity using deep reinforcement learning. *IEEE Trans. Robot.* **2021**, *38*, 317–328. [CrossRef]

14. Park, H.-W.; Wensing, P.M.; Kim, S. High-speed bounding with the MIT Cheetah 2: Control design and experiments. *Int. J. Robot. Res.* **2017**, *36*, 167–192. [CrossRef]
15. Hubicki, C.M.; Aguilar, J.J.; Goldman, D.I.; Ames, A.D. Tractable terrain-aware motion planning on granular media: An impulsive jumping study. In Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon, Republic of Korea, 9–14 October 2016; pp. 3887–3892.
16. Nguyen, Q.; Powell, M.J.; Katz, B.; Di Carlo, J.; Kim, S. Optimized jumping on the mit cheetah 3 robot. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 7448–7454.
17. Kim, D.; Carballo, D.; Di Carlo, J.; Katz, B.; Bledt, G.; Lim, B.; Kim, S. Vision aided dynamic exploration of unstructured terrain with a small-scale quadruped robot. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020; pp. 2464–2470.
18. Loquercio, A.; Kumar, A.; Malik, J. Learning visual locomotion with cross-modal supervision. In Proceedings of the 2023 IEEE International Conference on Robotics and Automation (ICRA), London, UK, 29 May–2 June 2023; pp. 7295–7302.
19. Rudin, N.; Hoeller, D.; Bjelonic, M.; Hutter, M. Advanced skills by learning locomotion and local navigation end-to-end. In Proceedings of the 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Kyoto, Japan, 23–27 October 2022; pp. 2497–2503.
20. Wellhausen, L.; Hutter, M. Rough terrain navigation for legged robots using reachability planning and template learning. In Proceedings of the 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Prague, Czech Republic, 27 September–1 October 2021; pp. 6914–6921.
21. Peng, M.; Gong, Z.; Sun, C.; Chen, L.; Cao, D. Imitative reinforcement learning fusing vision and pure pursuit for self-driving. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020; pp. 3298–3304.
22. Sun, Q.; Wang, Z.; Li, M.; Liu, B.; Cheng, J.; Tai, J. Path tracking control of wheeled mobile robot based on improved pure pursuit algorithm. In Proceedings of the 2019 Chinese Automation Congress (CAC), Hangzhou, China, 22–24 November 2019; pp. 4239–4244.
23. Lin, J.; Ni, T. Research on optimized motion control of soccer robot based on fuzzy-PID control. In Proceedings of the 2018 5th International Conference on Information Science and Control Engineering (ICISCE), Zhengzhou, China, 20–22 July 2018; pp. 952–956.
24. Wildani, F.; Mardiati, R.; Mulyana, E.; Setiawan, A.E.; Nurmalasari, R.R.; Sartika, N. Fuzzy logic control for semi-autonomous navigation robot using integrated remote control. In Proceedings of the 2022 8th International Conference on Wireless and Telematics (ICWT), Yogyakarta, Indonesia, 21–22 July 2022; pp. 1–5.
25. Moran, A.; Nagai, M. Autonomous driving of truck-trailer mobile robots with linear-fuzzy control for trajectory following. In Proceedings of the 2020 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), Glasgow, UK, 19–24 July 2020; pp. 1–8.
26. Di Carlo, J.; Wensing, P.M.; Katz, B.; Bledt, G.; Kim, S. Dynamic locomotion in the mit cheetah 3 through convex model-predictive control. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 1–9.
27. Bledt, G.; Powell, M.J.; Katz, B.; Di Carlo, J.; Wensing, P.M.; Kim, S. Mit cheetah 3: Design and control of a robust, dynamic quadruped robot. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 2245–2252.
28. Ding, Y.; Pandala, A.; Li, C.; Shin, Y.-H.; Park, H.-W. Representation-free model predictive control for dynamic motions in quadrupeds. *IEEE Trans. Robot.* **2021**, *37*, 1154–1171. [CrossRef]
29. Kim, D.; Di Carlo, J.; Katz, B.; Bledt, G.; Kim, S. Highly dynamic quadruped locomotion via whole-body impulse control and model predictive control. *arXiv* **2019**, arXiv:1909.06586.
30. Bellicoso, C.D.; Jenelten, F.; Fankhauser, P.; Gehring, C.; Hwangbo, J.; Hutter, M. Dynamic locomotion and whole-body control for quadrupedal robots. In Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, 24–28 September 2017; pp. 3359–3365.
31. Fahmi, S.; Mastalli, C.; Focchi, M.; Semini, C. Passive whole-body control for quadruped robots: Experimental validation over challenging terrain. *IEEE Robot. Autom. Lett.* **2019**, *4*, 2553–2560. [CrossRef]
32. Li, S.; An, X.; Ding, L.; Wang, Q.; Gao, H.; Hou, Y.; Deng, Z. Adaptive neural network-based finite-time tracking control for nonstrict nonaffined MIMO nonlinear systems. *IEEE Trans. Syst. Man Cybern. Syst.* **2019**, *51*, 4814–4824. [CrossRef]
33. Xu, P.; Ding, L.; Li, Z.; Yang, H.; Wang, Z.; Gao, H.; Zhou, R.; Su, Y.; Deng, Z.; Huang, Y. Learning physical characteristics like animals for legged robots. *Natl. Sci. Rev.* **2023**, *10*, nwad045. [CrossRef] [PubMed]
34. Koco, E.; Mirkovic, D.; Kovačić, Z. Hybrid compliance control for locomotion of electrically actuated quadruped robot. *J. Intell. Robot. Syst.* **2019**, *94*, 537–563. [CrossRef]
35. Xiao, A.; Tong, W.; Yang, L.; Zeng, J.; Li, Z.; Sreenath, K. Robotic guide dog: Leading a human with leash-guided hybrid physical interaction. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA), Xian, China, 30 May–5 June 2021; pp. 11470–11476.
36. Wang, R.; Li, Y.; Fan, J.; Wang, T.; Chen, X. A novel pure pursuit algorithm for autonomous vehicles based on salp swarm algorithm and velocity controller. *IEEE Access* **2020**, *8*, 166525–166540. [CrossRef]

37. Huang, Y.; Tian, Z.; Jiang, Q.; Xu, J. Path tracking based on improved pure pursuit model and pid. In Proceedings of the 2020 IEEE 2nd International Conference on Civil Aviation Safety and Information Technology (ICCASIT), Weihai, China, 14–16 October 2020; pp. 359–364.
38. Macenski, S.; Singh, S.; Martín, F.; Ginés, J. Regulated pure pursuit for robot path tracking. *Auton. Robot.* **2023**, *47*, 685–694. [CrossRef]
39. Liu, K. Study of Architecture System and Navigation Technology of Unmanned Grand Vehicle. Master's Thesis, Beijing Institute of Technology, Beijing, China, 2010.
40. Arena, P.; Patanè, L.; Spinosa, A.G. A New Embodied Motor-Neuron Architecture. *IEEE Trans. Control Syst. Technol.* **2021**, *30*, 2212–2219. [CrossRef]

**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.