agriculture

MDPI

Article

Design and Development of an Intelligent Robotic Feeding Control System for Sheep

Haina Jiang, Haijun Li * and Guoxing Cai

College of Electromechanical Engineering, Inner Mongolia Agricultural University, Hohhot 010018, China; haina@emails.imau.edu.cn (H.J.); 15124792538@163.com (G.C.)
* Correspondence: lhj70927@imau.edu.cn

Abstract

With the widespread adoption of intelligent technologies in animal husbandry, traditional manual feeding methods can no longer meet the demands for precision and efficiency in modern sheep farming. To address this gap, we present an intelligent robotic feeding system designed to enhance feeding efficiency, reduce labor intensity, and enable precise delivery of feed. This system, developed on the ROS platform, integrates LiDAR-based SLAM with point cloud rendering and an Octomap 3D grid map. It combines an improved bidirectional RRT* algorithm with Dynamic Window Approach (DWA) for efficient path planning and uses 3D LiDAR data along with the RANSAC algorithm for slope detection and navigation information extraction. The YOLOv8s model is utilized for precise sheep pen marker identification, while integration with weighing sensors and a farm management system ensures accurate feed distribution control. The main research contribution lies in the development of a comprehensive, multi-sensor fusion system capable of achieving autonomous feeding in dynamic and complex environments. Experimental results show that the system achieves centimeter-level accuracy in localization and attitude control, with FAST-LIO2 maintaining precision within 1° of attitude angle errors. Compared to baseline performance, the system reduces node count by 17.67%, shortens path length by 0.58 cm, and cuts computation time by 42.97%. At a speed of 0.8 m/s, the robot achieves a maximum longitudinal deviation of 7.5 cm and a maximum heading error of 5.6°, while straight-line deviation remains within ±2.2 cm. In a 30 kg feeding task, the system demonstrates zero feed wastage, highlighting its potential for intelligent feeding in modern sheep farming.

Keywords: smart feeding; sheep robot; vision SLAM; path planning; target recognition

check for updates

Academic Editor: José Lima

Received: 29 July 2025
Revised: 6 September 2025
Accepted: 8 September 2025
Published: 9 September 2025

Citation: Jiang, H.; Li, H.; Cai, G. Design and Development of an Intelligent Robotic Feeding Control System for Sheep. *Agriculture* **2025**, *15*, 1912. https://doi.org/10.3390/agriculture15181912

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

As the livestock industry undergoes structural transformation, the sheep industry—recognized as a grain-saving livestock sector—has made large-scale production a key benchmark for modernizing animal husbandry. Sheep are raised using various methods, including mixed, pasture-based, and stall feeding, with stall feeding offering advantages in improving production scale and standardization. However, current stall-feeding practices heavily depend on manual labor, leading to high workloads and increased risk of zoonotic disease transmission. In response to emerging policies, intelligent unmanned feeding robots are becoming a promising development direction. Although existing automated feeding systems—such as track-mounted or bicycle-type feeders—help reduce labor intensity, they face limitations including high infrastructure requirements, high costs, limited adaptability to individualized feeding, and low levels of automation. They still require on-site human

intervention for parameter adjustment and system monitoring. To address these challenges, there is an urgent need to develop advanced intelligent feeding robots equipped with environmental sensing, autonomous decision-making, precise localization, and target recognition capabilities. These robots should autonomously navigate into and out of sheep sheds, identify pen markers, integrate with weighing sensors for accurate and individualized feeding, and incorporate obstacle avoidance to ensure safe and efficient operations. Such systems would significantly enhance feeding precision, reduce labor demands, and accelerate the intelligent transformation of sheep farming.

Accurate environmental perception inside and outside the sheep pens, along with robust navigation capabilities in livestock farm environments, are critical technologies for intelligent feeding robots. These serve as foundational guarantees for achieving precise, efficient, and intelligent feeding. Key tasks include slope detection and real-time obstacle avoidance.

The Lely Vector intelligent feeding system developed by Dutch company Lely (Maassluis, The Netherlands) [1] can perform autonomous navigation tasks in livestock areas. The device is equipped with a feed trough height monitoring system, which can automatically trigger precise feeding programs based on pre-set thresholds. D. Piwcznski et al. [2] proposed an automated milking system that combines milking robots with feeding stations to effectively increase milk production and milking efficiency. Jangho Bae et al. [3] developed a total mixed ration (TMR) feeding robot capable of carrying up to 2 tons of feed. The robot uses lidar and cameras to recognize its environment, relies on RFID for positioning, and allows staff to obtain the robot's current location via RFID. S. M. Mikhailichenko et al. [4] used a graph theory-based simulation method to optimize the volume of feed spreaders while considering feed distribution time.

With the development of computer technology, simultaneous localization and mapping (SLAM) technology is being applied more and more widely in robot localization and mapping.

By enhancing robustness and real-time performance [5–10], multi-sensor fusion [11,12] improves the stability and accuracy of SLAM systems in complex environments, while error modeling is also employed to enhance positioning accuracy and robustness. Shen et al. [13] proposed a laser radar inertial SLAM system based on the LIO-SAM framework, further improving matching accuracy. Cao et al. [14] contributed to lightweight design with their BEV-LSLAM algorithm. Lin et al. subsequently proposed the R2LIVE algorithm [15] and R3LIVE algorithm [16] to improve localization accuracy, while Wang et al. [17] proposed the F-LOAM algorithm, which uses scan-to-map optimization of pose on the basis of the Ceres Solver [18] to enhance system operational efficiency. By integrating multiple strategies to improve planning efficiency and path quality [19–23], incorporating machine learning and intelligent strategies to enhance adaptability [24–26], combining door recognition visual perception methods [27] to enhance practicality and accuracy, improving robustness and practicality [28–30], and enhancing search efficiency to reduce invalid nodes for obstacle avoidance and optimal path construction [31–34].

However, despite these advancements, several key limitations persist in the existing research: Limited Adaptability to Complex Farm Environments: Many of the existing robotic feeding systems, such as the Lely Vector system [1] and Jangho Bae's TMR feeding robot [3], perform well in controlled environments but are less effective when deployed in dynamic, complex farm conditions where obstacles and terrain vary significantly. These systems still require significant human intervention for parameter adjustment and system monitoring. Lack of Integration of Multi-Sensor Fusion for Robust Navigation: Although advancements in SLAM and multi-sensor fusion have been made (e.g., LIO-SAM, BEV-LSLAM, and R2LIVE algorithms), current systems still struggle with real-time decision-making and robust environmental perception, especially in unpredictable environments. For instance, despite the advancements made by Shen et al. [13] and Cao et al. [14], robust

integration of LiDAR, cameras, and environmental sensors for dynamic obstacle avoidance and precise path planning remains a challenge.

This research aims to address these limitations by developing an intelligent feeding robot that integrates advanced multi-sensor fusion, robust SLAM navigation, and autonomous decision-making capabilities to enable individualized feeding in complex, dynamic farm environments. Unlike current systems, our approach focuses on the seamless integration of various sensors for real-time obstacle avoidance, dynamic path planning, and precise feeding control, with the goal of reducing labor intensity and improving operational efficiency in modern sheep farming.

2. Materials and Methods

This feeding robot utilizes advanced technologies to automate the feeding of sheep, as shown in Figure 1. It adopts a modular design architecture, with the control unit centered around a YCT-IPC-065 industrial computer (Shenzhen Yanchengtuo Technology Co., Ltd., Shenzhen, China). Equipped with a laser radar (Livox, Shenzhen, China), the robot can calculate the target's position in 3D space and perform 3D modeling. The sensor system is a core component of the robot, consisting of a front camera Astra Pro Camera (ORBBEC, Shenzhen, China), a side camera Drive Free Camera (Yahboom, Shenzhen, China), a temperature and humidity sensor, a weighing sensor, and a rail-mounted weighing module. The front camera integrates the high-precision distance measurements from the laser radar with the color information from the camera, generating a point cloud that contains both spatial location and color features. The side camera is used to identify pen numbers within the sheep shed. The temperature and humidity sensor monitors the environmental conditions in real time. The weighing sensor and the rail-mounted weighing module monitor the feed weight in the silo, enabling precise control of feed dispensing. The central control unit coordinates the operation of all components and processes data to make decisions, while the power supply system ensures continuous and stable operation of the robot. Through the coordinated function of these hardware components, the feeding robot can efficiently and autonomously complete sheep feeding tasks.

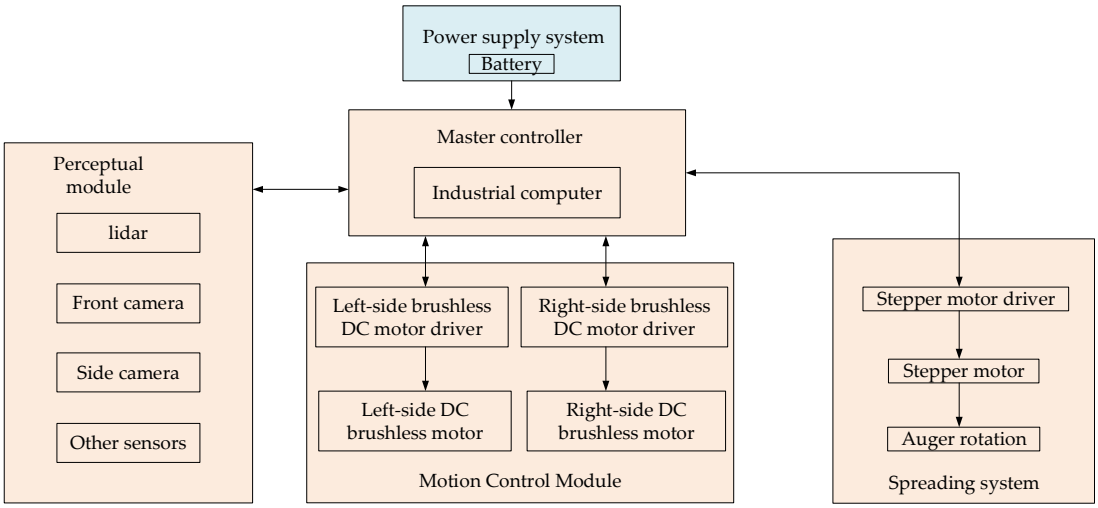


Figure 1. Hardware System Framework for Sheep Feeding Robot.

The software framework of the sheep feeding robot is divided into five functional nodes: environmental perception, map building, path planning, motion control, and target recognition. The structure is illustrated in Figure 2.

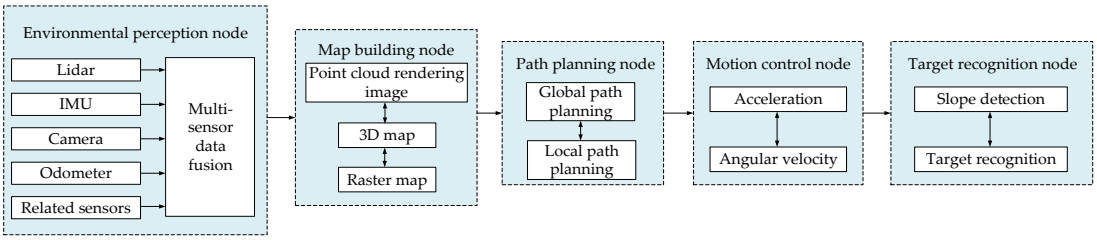


Figure 2. Software Systems Framework.

(1) Environment sensing node: collect environmental information and robot position data using LiDAR, IMU, wheel odometer and camera.

(2) Map construction node: the acquired environmental data are used to construct a 3D map of the surrounding environment and a point cloud rendering map after the fusion of LIDAR and camera data, and the sheep feeding robot localizes itself based on this map model and projects its position. By using the Octomap function package v1.9.8., the 3D point cloud map is converted into a 3D raster map, while retaining the key spatial information in the 3D model of the environment.

(3) Path planning node: Visualize the raster map in Rviz, set the start point and end point, the path planning node receives the map data sent by the map module, and the global path planning algorithm ensures that the robot efficiently selects the best path from the start point to the end point. When the robot advances along this path, it acquires point cloud data in real time, and the local path planning algorithm avoids dynamic obstacles by adjusting linear and angular velocities.

(4) Motion control node: the industrial computer sends control commands to transfer acceleration, angular velocity and other information to the motor drive node, and the drive node controls the motor accordingly to realize the robot motion.

(5) Target recognition node: the entrance of the sheep house is uphill and the exit is downhill. The industrial control machine analyzes the slope detection of the ground data acquired by the LiDAR to determine whether the robot reaches the entrance or exit of the sheep house. The ICP acquires the picture of the sheep pen sign inside the sheep house through the side camera and uses the YOLOv8s model to complete the recognition of the sheep pen sign to realize accurate material spreading.

Camera lenses and sensors may introduce radial aberrations and tangential aberrations leading to straight-line curvature or inaccurate shape of the object, so it is necessary to accurately obtain the internal parameters of the camera, and to correct the aberrations through the internal parameters. Firstly, a black and white checkerboard grid with 11 $\times$8 number of internal corner points and 0.02 m length of small square sides is prepared.

The internal parameters of the camera were solved using the ros software package, and the calibration program appeared after inputting the number of internal corner points and the size of the checkerboard grid. By tilting the black and white checkerboard grid back and forth, left and right, up and down, and tilting the black and white checkerboard grid, the results of “X”, “Y”, ‘Size’, and “Skew” are all green. Click “CALIBRATE” to obtain the internal reference of the camera after the result of “X”, “Y”, ‘Size’ and “Skew” is green. Click “SAVE” to write the calibration results of the front and side cameras into the “ost.yaml” file.

Use Livox’s official open-source livox_lidar_camera_calibration for joint calibration of the front camera and lidar. The calibration process is as follows: (1) Prepare an 80 $\times$ 80 cm black-and-white checkerboard with 9 internal corners and small squares of 0.08 m side length. Install the source code in the ROS workspace (ROS1) and compile it. Write the obtained front camera internal parameters to a file. Place the checkerboard at different

positions, record 10 s of lidar point cloud data and camera images at each position, and name the point cloud data and camera images identically at the same position. Run the program, click on the four corners of the checkerboard in order, and record the coordinates of the four corners in each photo. Finally, write the coordinates of all corners in the photos into a file. Run the program to convert the lidar point cloud data packages in the folder into visualizable files. Open the PCD file, obtain the x, y, and z coordinates of each corner point according to the order in which the corner points were obtained, and finally write the coordinates of all corner points in the PCD files into a file. Run the command in the workspace to iteratively calculate the data in the file and solve for the external parameters of the camera and lidar.

The coordinate system of the sheep feeding robot is expressed in Cartesian coordinate system, and the $X_W O_W Y_W$ global coordinate system and the $X_T O_T Y_T$ robot coordinate system are usually used to describe the position of the robot. The robot coordinate system Y_T points to the front of the robot, and the orientation angle θ of the robot represents the rotation angle of the subject coordinate system relative to the global coordinate system, and the mapping of the subject coordinate system to the global coordinate system can be represented by an orthogonal rotation matrix as shown in Equation (1).

$$R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

The robot is driven by differential rear wheels, according to the differential kinematic model as shown in Figure 3.

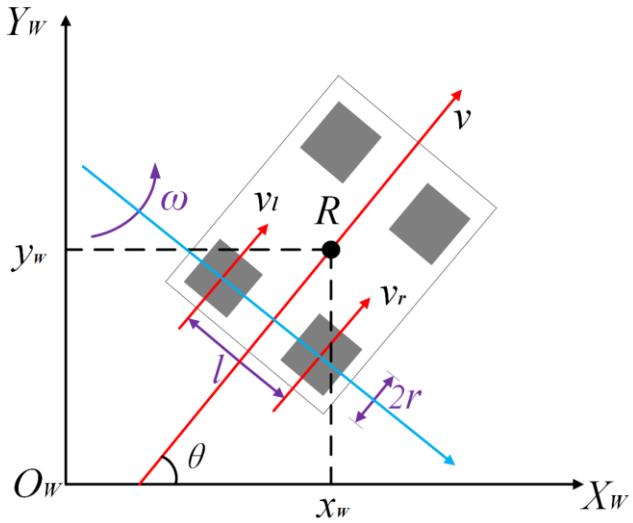


Figure 3. Differential Kinematics Modeling.

The bit position of the sheep feeding robot at a given moment can be expressed as Equation (2):

$$R = (x_W, y_W, \theta)^T \tag{2}$$

According to the principles of statics, the forward kinematic equation for the differential motion of the sheep feeding robot can be obtained as shown in Equations (3) and (4):

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{l} & -\frac{1}{l} \end{bmatrix} \begin{bmatrix} v_r \\ v_l \end{bmatrix} \tag{3}$$

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \theta \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{4}$$

In the global coordinate system, the coordinates and motion state of the point R approximately represent the robot's position. Therefore, the robot's position can be obtained by integrating and solving Equation (4) as follows:

$$\begin{cases} x = x_0 + \frac{1}{2}\int_0^t (v_r + v_l)\cos\theta \, dt \\ y = y_0 + \frac{1}{2}\int_0^t (v_r + v_l)\sin\theta \, dt \\ \theta = \theta_0 + \frac{1}{l}\int_0^t (v_r - v_l) dt \end{cases} \tag{5}$$

The trajectory of the robot in a short time interval is approximated as a straight line and restricts the robot to only forward or rotational motion, as shown in Figure 4.

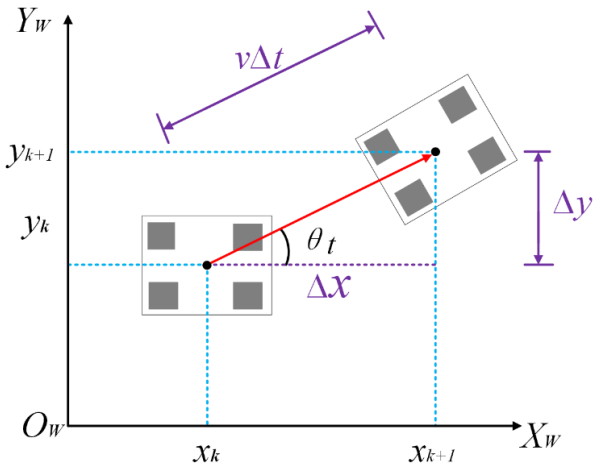


Figure 4. Trajectory Projection Model.

Within the time interval Δt, the robot's pose transformation can be regarded as a combination of translating a distance $v\Delta t$ and rotating an angle θ_t. In the global coordinate system, the coordinate change in the robot's center of gravity from time k to time $k+1$ in Equation (6) is as follows:

$$\begin{cases} \Delta x = v\Delta t\cos(\theta_t) \\ \Delta y = v\Delta t\sin(\theta_t) \end{cases} \tag{6}$$

The pose of the robot at time k + 1 can be expressed as Equation (7):

$$\begin{cases} x_{k+1} = x_t + v\Delta t\cos(\theta_t) \\ y_{k+1} = y_t + v\Delta t\sin(\theta_t) \\ \theta_{k+1} = \theta_t + \omega\Delta t \end{cases} \tag{7}$$

According to Equation (2), the linear velocity and angular velocity at the robot's center of gravity are (8) and (9), respectively:

$$v = \frac{v_r + v_l}{2} \tag{8}$$

$$\omega = \frac{v_r - v_l}{2l} \tag{9}$$

In the case of $v_r = v_l$, the robot carries out linear motion and the trajectory is a straight-line segment; in the case of $v_r \neq v_l$, the trajectory of the robot is a curve. According to

Equations (2), (3), (8) and (9), the inverse kinematics of the sheep feeding robot can be expressed as:

$$\begin{cases} v_r = v + \omega l \\ v_l = v - \omega l \end{cases} \tag{10}$$

in other words:

$$\begin{bmatrix} v_r \\ v_l \end{bmatrix} = \begin{bmatrix} 1 & l \\ 1 & -l \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{11}$$

In the motion control of the sheep feeding robot, the initial linear and angular velocities at the center of mass of the robot are set according to the actual situation, and the actual linear and angular velocities of the left and right driving wheels are calculated according to Equations (10) and (11). The speed of the drive wheels needs to be adjusted by the motor speed, and Equation (12) is the formula for calculating the rotational speed and the speed of the drive wheels, where n_l and n_r are the rotational speeds of the left and right drive wheels, respectively.

$$\begin{cases} n_l = \frac{v_l}{60 \times 2\pi r} \\ n_r = \frac{v_r}{60 \times 2\pi r} \end{cases} \tag{12}$$

The calibration relationship between the LIDAR and the frontal camera correlates the laser data with the image data and renders the point cloud data of the 3D LIDAR to obtain the color laser, so as to obtain the real ground environment map. Acquire the point cloud data of LiDAR and the image information of the frontal camera; store the x, y, z coordinates and intensity value of each point cloud data through a loop; convert the point cloud data from the coordinate system of LiDAR to the coordinate system of the camera through matrix multiplication according to the internal and external parameters obtained from the joint calibration of LiDAR and the frontal camera. Calculate the position of each point cloud data in the coordinate system of the camera, keep the pixel points that are located in the image range and have an x-value greater than 0 in the coordinates of the point cloud, obtain the color information from the image through the coordinate system of the camera, and assign the RGB color to the valid point cloud data. The point cloud data assigned with RGB color is converted from camera coordinate system to world coordinate system, and then the point cloud data is converted to ROS message format, and the point cloud message is published through ROS.

3D maps constructed by 3D LIDAR with FAST-LIO2 algorithm cannot be directly used for navigation. The 3D point cloud data is transformed into raster map in 2D environment by Octomap_server, which preserves the height information in the environment model. The specific scheme is shown in Figure 5.

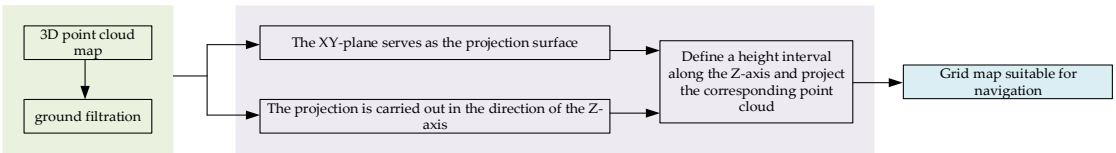


Figure 5. 2D Raster Map Conversion Program.

In the ROS navigation system, the raster map consists of several small grid cells, each of which can store different values. Typically, white areas represent passable areas, corresponding to a raster value of 0; black areas represent obstacles, with a raster value of 100; and gray areas represent unknowns, where passability is still unclear, corresponding to a raster value of -1.

Converting the 2D raster map requires configuring the Octomap_server startup file according to the point cloud topic and the Z-axis height. The point cloud topic for the

FAST-LIO2 algorithm is "camera_init", and the overall height of the sheep-feeding robot is 1.54 m. Therefore, adjusting the Z-axis projection range is $-0.5\text{m} \sim 1.7\text{m}$.

Sheep feeding robots require good path planning to achieve intelligence. The robot loads at the loading point and then travels to the designated sheep shed. There are various known and unknown obstacles in the environment where the robot reaches the designated sheep shed from the loading point. An improved two-way RRT* algorithm is proposed as a global path planning method for mobile robots. The improved Dynamic Window Approach (DWA) is used as a local path planning strategy for the mobile robot.

The artificial potential field method is introduced, which is mainly used for obstacle avoidance and target guidance. The core of this method is to construct a gravitational potential field at the target point and a repulsive potential field around the obstacles according to the distribution of the environment map and obstacles, and the robot moves along the direction of the combined force of the attractive force and the repulsive force.

The gravitational potential energy is positively correlated with the Euclidean distance between the robot and the target point, and the gravitational potential energy $U_{att}(q)$ is expressed by Equation (13), where k_{att} is the gain coefficient of the gravitational field, and $\rho\left(q, q_{goal}\right)$ is the Euclidean distance between the robot's location and the target point q_{goal}, and the direction is pointing to the target point from the robot's location. The attraction $F_{att}(q)$ can be expressed by Equation (14).

$$Uattq = 12katt\rho 2q,qgoal \tag{13}$$

$$Fattq = -\nabla Uatt = katt\rho q,ggoal \tag{14}$$

The obstacle generates repulsive potential energy on the mobile robot, but in order to prevent the potential energy generated by the obstacle from affecting each other, a range of influence is set for it, and the repulsive force is applied when the mobile robot enters into the range of influence. If it does not enter the influence range, the repulsive force is zero. And the size of the repulsive potential energy is negatively correlated with the Euclidean distance between the mobile robot and the obstacle. The repulsive potential energy function $U_{rep}(q)$ is expressed by Equation (15). k_{rep} is the repulsive potential field gain coefficient, which is used to adjust the size and influence range of the repulsive potential field generated by the obstacle. $\rho(q, q_{obs})$ denotes the nearest distance from the robot's position q to the obstacle. ρ_0 is the defining distance, and the repulsive potential energy is zero when $\rho(q, q_{obs})$ is greater than the defining distance ρ_0, in other words., the repulsive force is generated only for obstacles within the defined distance.

$$U_{rep}(q) = \begin{cases} \frac{1}{2}k_{rep}\left(\frac{1}{\rho(q,q_{obs})} - \frac{1}{\rho_0}\right)^2, 0 \le \rho(q, q_{obs}) \le \rho_0 \\ 0 \qquad\qquad\qquad , \rho(q, q_{obs}) \ge \rho_0 \end{cases} \tag{15}$$

The repulsive force is the negative gradient of the repulsive potential energy, which can be expressed by Equation (16).

$$F_{rep}(q) = \begin{cases} k_{rep}\left(\frac{1}{\rho(q,q_{obz})} - \frac{1}{\rho_0}\right)\frac{1}{\rho^2(q,q_{obz})}\nabla\rho(q, q_{obz}), \rho(q, q_{obz}) \le \rho_0 \\ 0 \qquad\qquad\qquad\qquad\qquad\qquad , \rho(q, q_{obz}) \ge \rho_0 \end{cases} \tag{16}$$

The robot is subject to the repulsive force of multiple obstacles within its sphere of influence as its distance from the obstacle changes continuously during its movement. At a certain point in time during the movement of the robot, there are n obstacles within its

sphere of influence, and the combined potential energy of the force on the robot is given by Equation (17):

$$U(q) = U_{att}(q) + \sum_{i=0}^{n} U_{rep}(q) \tag{17}$$

According to the potential energy of the resultant force, the resultant force acting on the robot is given by Equation (18):

$$F(q) = F_{att}(q) + \sum_{i=0}^{n} F_{rep}(q) \tag{18}$$

The bias probability is dynamically adjusted according to the density of obstacles in the current extended region, reducing the bias probability in regions with dense obstacles and increasing it in regions with fewer obstacles. This method effectively reduces the number of collision detection failures and improves the exploration speed. In addition, the bias probability is adjusted according to the distance from the target point, increasing the bias probability when it is farther away from the target point and decreasing the bias probability when it is closer to the target point in order to avoid being blocked by the obstacles near the target point. Each time the sampling point q_{rand} is generated, q_{rand} is biased towards q_{goal} according to different bias probabilities.

Open region, a shorter step size then leads to a shorter extension distance, which significantly slows down the exploration speed [35]. This study dynamically adjusts the extension step length according to the local obstacle density at q_{new}. The step length decreases in regions with high localized obstacle density, while the step length increases in regions with low localized obstacle density. The dynamic step size S is calculated as:

$$S = S_{min} + (S_{max} - S_{min}) \times e^{-D_0} \tag{19}$$

In Equation (19): S_{max} is the maximum step size and S_{min} is the minimum step size.

(1) Introduction of Rectangular Obstacles

The morphology of obstacles is usually not completely circular or simple geometric shapes. Rectangle is one of the most common obstacle morphology; the introduction of rectangular obstacle processing makes it possible to consider the shape of obstacles more reasonably when selecting the motion trajectory, and more accurate collision detection can be carried out to avoid collision between the robot and these obstacles.

Assuming that the width of the rectangular obstacle is w, the height is h, the coordinates of the rectangular center are (x, y), and the coordinates of the current sampling point q_{rand} are (x_1, y_2), then the distance ρ between the current sampling point q_{rand} and the rectangular obstacle is Equation (20):

$$\rho = \sqrt{\left(|x_1 - x| - \frac{w}{2}\right)^2 + \left(|y_1 - y| - \frac{h}{2}\right)^2} \tag{20}$$

Determine whether the sampling point is located inside the rectangular obstacle according to Equation (21):

$$inside = \left((|x_1 - x|) \leq \frac{w}{2}\right) \cap \left((|y_1 - y|) \leq \frac{h}{2}\right) \tag{21}$$

When the distance between the robot and the rectangular obstacle is less than the range of influence ρ_0, the repulsive force formula for the rectangular obstacle is given by Equation (22):

$$F_{rep} = \frac{1}{2} k_{rep} \left(\frac{1}{\rho} - \frac{1}{\rho_0} \right)^2 \tag{22}$$

(2) Dynamically adjusting the coefficient of attraction

Dynamically adjusting the coefficient of attraction can change the strategy according to the relative position of the robot and the target point and the complexity of the environment, so as to improve the efficiency of the movement and guarantee the safety. When there are fewer obstacles, the coefficient of attraction is raised to twice the original, and the robot can approach the target quickly. When there are more obstacles, the robot needs to avoid the obstacles more cautiously, and the attractiveness coefficient is appropriately reduced to 0.7 times of the original attractiveness coefficient to enhance the weight of repulsion.

In the navigation function package, the tf transformation is first introduced to realize the data transformation between the central coordinate system of LiDAR and the central coordinate system of the robot's bottom [36]. Then the delay function is introduced and the navigation module is later than the map building module. This allows the navigation module to make an initial path planning based on the environment information. The navigation framework is shown in Figure 6.

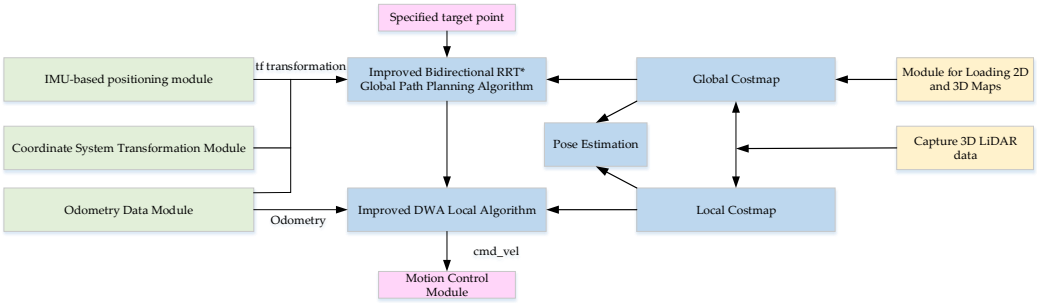


Figure 6. Move_base navigation framework.

3. Results

3.1. Slope Detection

3.1.1. Point Cloud Filtering

In sheep shed slope analysis, to accurately extract road elevation data, the original point cloud needs to be filtered to remove random noise and retain effective point cloud data relevant to the analysis.

This study employs the least squares method for data filtering. First, the point cloud data is divided into regular grids, and the maximum and minimum values along the X and Y axes are checked to determine the lowest and highest points of the grid. Next, the point cloud data is divided into $p \times q$ regular grids along the X and Y axes, with Equation (23) representing the grid spacing.

$$\begin{cases} l_x = (x_{max} - x_{min})/p \\ l_y = (y_{max} - y_{min})/q \end{cases} \tag{23}$$

Based on the defined grid boundaries and intervals, determine the boundaries of each small grid to identify the point cloud data within each grid, ensuring that the point cloud data is fully contained within the grid. Next, locate the lowest Z-value within the point cloud data of each grid and determine whether this value is the minimum Z-value within

that grid, thereby completing the filtering process. The model obtained from quadratic surface fitting can be expressed using Equation (24).

$$Z_k = a_0 + a_1 x_k + a_2 y_k + a_3 x_k^2 + a_4 x_k y_k + a_5 y_k^2 \tag{24}$$

In Equation (24): a_0, a_1, a_2, a_3, a_4, and a_5 are the coefficients of the quadratic surface.

3.1.2. Slope Detection Based on the RANSAC Algorithm

The principle of the RANSAC algorithm for fitting a plane is to first randomly select a set of samples from the data. Since at least three points are required to determine a plane, the number of points in the randomly selected sample set is set to 3. Next, the plane is fitted using the least squares method, and the model A, B, C, and corresponding deviations are calculated. Subsequently, the deviations of the point cloud sample data sets are calculated sequentially. If the calculated deviation value exceeds the set threshold range, the point is considered an outlier; if the deviation value is below the threshold, the point is considered an inlier. All outliers are excluded from the statistics, and the number of inliers is counted. The process is iterated repeatedly, with appropriate iteration conditions set to obtain optimal model parameters and ensure the maximum number of inlier point clouds.

The RANSAC algorithm repeatedly calculates the point cloud data within the range to finally fit the gray absolute horizontal plane and purple slope shown in Figure 7. Assuming that the normal vector of the robot on the absolute horizontal plane is n_0 and the normal vector on the slope is n_1, the slope θ can be obtained using Equation (25).

$$\theta = \cos^{-1}\left(\frac{n_1 \cdot n_0}{\|n_1\|^2 \cdot \|n_0\|^2}\right) \tag{25}$$

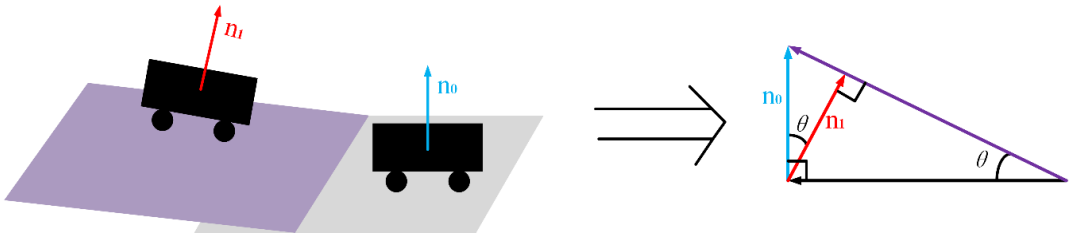


Figure 7. Slope solution.

After point cloud filtering, we extracted the ground surface and used the RANSAC algorithm to fit the slope plane to the uphill data package. The fitting results are shown in Figure 8, which displays the ground surface information and slope features extracted using the RANSAC algorithm. Figure 9 shows the robot climbing up the slope.



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

Figure 8. (**a**) Slope Fitting; (**b**) Actual Slope.



Figure 9. Robot Uphill.

3.2. External Path Planning for Sheep Farms

In a real-world environment, robots encounter dynamic obstacles during operation. Therefore, in this local path planning simulation, code has been inserted to generate ten randomly moving circular obstacles on a grid map. After the code is inserted, it checks whether any of the dynamic obstacles are too close to the starting point. If a dynamic obstacle is too close to, or overlaps with, the starting point, it can cause an operational error.

Figure 10 illustrates the initial state of the local path planning simulation on a 100 × 100 grid map. The green starting point is set at the bottom-left corner with coordinates (10,10), and the red goal point is at the top-right corner with coordinates (90,90). The robot is represented by a red circle, with a black line in front indicating its intended direction of movement. The gray patterns represent static obstacles, while the black patterns indicate dynamically moving obstacles. The blue lines represent the global path generated by the algorithm based on the obstacles. Black circles on the blue path represent waypoints that guide the robot along the global path. The blue dashed circles around each waypoint define the collision detection range, with a radius of 5 m, indicating the robot's proximity to obstacles.

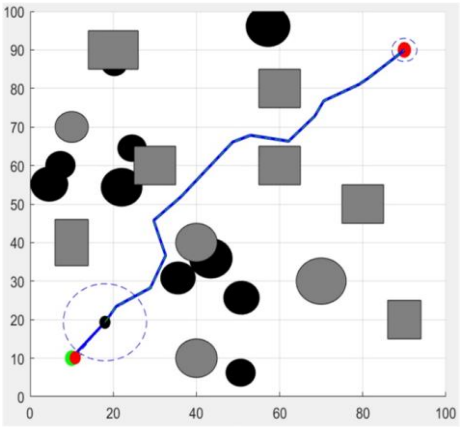


Figure 10. Initial State Diagram in the Simulation.

Figure 11a shows the robot's first dynamic obstacle avoidance. Due to the presence of a stationary obstacle on the right side of the robot, along with two dynamic obstacles that

are moving leftward, the robot moves towards the upper-left direction, following the path indicated by the black line, to avoid the dynamic obstacles. Figure 11b depicts the robot's second dynamic obstacle avoidance. As the circular dynamic obstacle on the left side is moving toward the robot, the robot follows the direction indicated by the black line to perform dynamic obstacle avoidance. Figure 11c shows the robot's third dynamic obstacle avoidance. The circular dynamic obstacle at the lower-right corner is moving toward the robot, so the robot moves along the direction indicated by the black line to avoid the obstacle. Figure 11d illustrates the robot approaching the goal, nearing the target location.

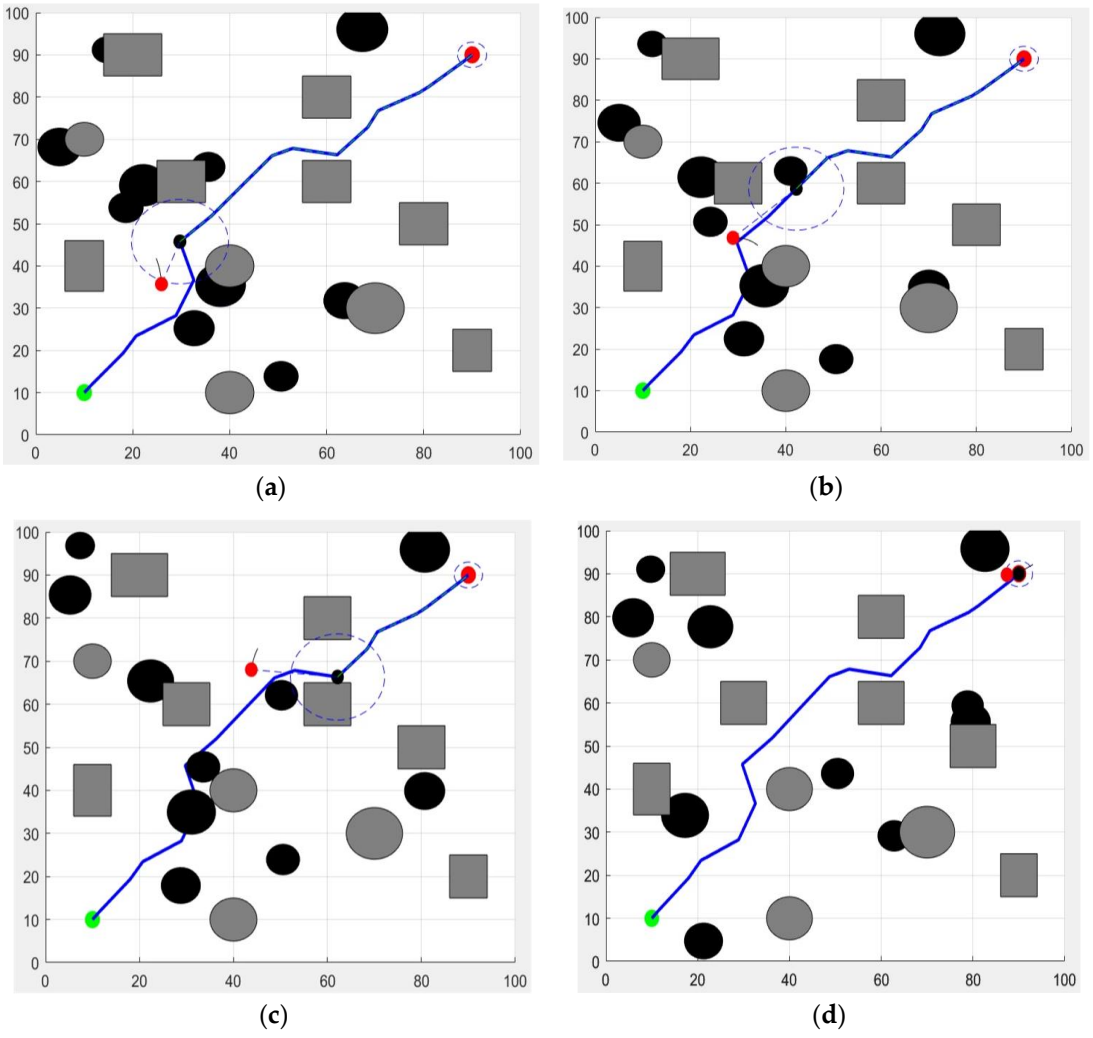


Figure 11. (**a**) First Dynamic Obstacle Avoidance; (**b**) Second Dynamic Obstacle Avoidance; (**c**) Third Dynamic Obstacle Avoidance; (**d**) The Robot has Reached the Destination.

3.3. Path Planning Inside the Sheep Shed

When the sheep feeding robot is distributing feed inside the sheep shed, it should first ensure that it can move in a straight line along the sheep pen fence, accurately dispensing feed into the feed trough. If a dynamic obstacle avoidance scheme is adopted upon encountering obstacles, the feed may be scattered outside the feed trough, which constitutes significant waste for the sheep farm. Therefore, when the robot encounters an obstacle inside the sheep shed, it should stop moving and cease feeding until the obstacle is removed, after which it can resume operations. This ensures that the sheep feeding robot can accurately and efficiently complete its feeding tasks. Figure 12 shows an actual view of the interior of the sheep shed.



Figure 12. Interior view of the sheep house. Note: 1 Roof; 2 Passageway; 3 Feed trough; 4 Fence; 5 Sheep.

3.4. Object Detection Algorithm Based on YOLOv8s

The feed dispensed by the sheep feeding robot in each pen should vary according to the size and age of the sheep in that pen. Generally, sheep farms have specific feeding plans for each pen. Therefore, the side camera of the sheep feeding robot needs to perform object detection on the information on the pen signs. After accurately obtaining the sign information, the robot will dispense feed according to the feeding plan specified by the sheep farm for that pen. The stepper motor drives the auger to rotate, beginning to dispense feed into the feed trough. When the weight sensor detects that the feeding task for the pen is complete, the stepper motor stops operating.

The YOLOv8s model was trained using annotated pen signs, with the training results shown in Figure 13. Experimental results indicate that the proposed object detection model demonstrated excellent convergence and detection performance during training. The object localization loss rapidly decreases and eventually converges, indicating that the gap between the model's predicted bounding boxes and the actual bounding boxes is small, enabling accurate localization of the object's position and shape. The classification loss also rapidly decreases and eventually converges. A low classification loss indicates that the model has strong classification capabilities and can correctly identify object categories. In terms of detection accuracy evaluation, precision measures the proportion of correctly classified objects among those predicted as positive classes. As training progresses, the precision curve rapidly increases and eventually converges, indicating that the model's detection and recognition capabilities are gradually improving. Recall measures the proportion of true positive objects that the model can detect; a higher recall indicates a lower false negative rate. In this experiment, the recall rate approaches 1 at the end of training, indicating that the model possesses strong object detection capabilities and can effectively reduce false negatives. Average precision (mAP_0.5) reflects the average precision at an intersection-over-union (IoU) threshold of 0.5. A higher value indicates better detection performance across all categories, and further validates the model's detection capabilities, generalization capabilities, and robustness.

Overall, the model's bounding box loss, object localization loss, and classification loss all gradually decrease and eventually converge during training, indicating that the model is effectively learning and continuously improving its performance. The simultaneous rapid improvement and eventual convergence of precision and recall rates indicate that the model has achieved significant results in reducing false positives and false negatives, validating its superior performance in object detection tasks.

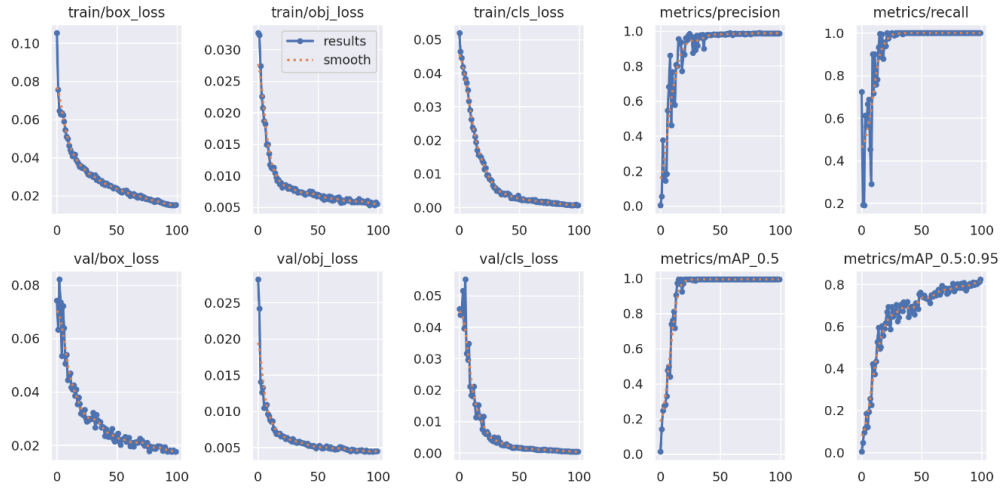


Figure 13. Sheep pen tagging training results.

Figure 14 shows the F1 curve of the model. The F1 curve is typically used to assess the stability of the model. Observing the trend of the F1 curve, it is evident that the F1 curve for most categories increases as the confidence threshold increases. When the threshold reaches 0.882, the F1 curve reaches its peak value of 1, indicating that the model achieves perfect performance in identifying all categories at this confidence threshold.

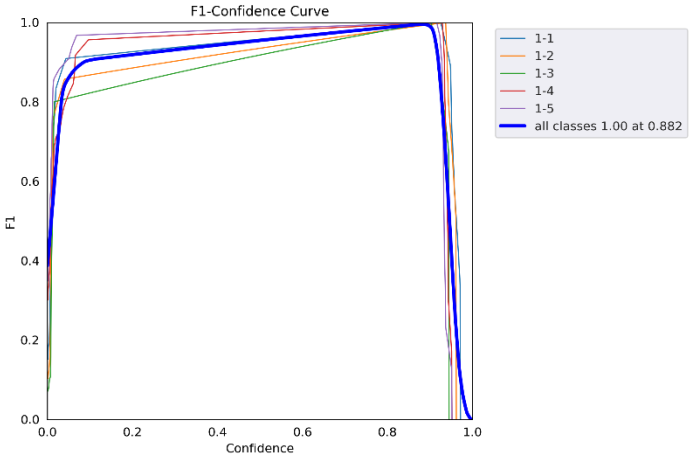


Figure 14. F1 curve of the model.

3.5. On-Site Experiment at the Sheep Farm

On-site mapping, navigation accuracy, straight-line walking stability, and feed distribution experiments were conducted at the sheep farm to further demonstrate the practicality of the sheep feeding robot.

3.5.1. Navigation Accuracy Experiment

A path from the loading point to Sheep Shed No. 8 was selected as the navigation accuracy test route. Navigation accuracy was assessed by comparing the deviation between the planned path and the actual path. The robot's orientation toward the target direction was defined as the positive direction of the X-axis, and the vertical direction as the positive direction of the Y-axis. During navigation, the robot's target position was (x_0, y_0), and when the robot reached this position, it was recorded as (x_1, y_1). Longitudinal deviation refers to the absolute value of the difference between the final position and the target position along the Y-axis. Heading deviation reflects the angular relationship between the Y-axis. To test the robot's navigation accuracy at different speeds, navigation tests were

conducted at speeds of 0.2 m/s, 0.5 m/s, and 0.8 m/s, and the maximum, average, and standard deviation values of longitudinal deviation and heading deviation were recorded. Table 1 presents the experimental results.

Table 1. Vertical and heading deviation statistics.

Speed (m/s)	Longitudinal Deviation			Heading Deviation		
	Maximum Value (cm)	Average Value (cm)	Standard Deviation (cm)	Maximum Value (°)	Average Value (°)	Standard Deviation (°)
0.2	3.6	2.5	1.5	2.9	1.2	1.2
0.5	5.9	4.3	2.9	4.1	2.6	1.9
0.8	7.5	4.9	3.5	5.6	3.2	3.1

As shown in Table 1, as the speed of the sheep feeding robot increases, the average values and standard deviations of both the longitudinal deviation and heading deviation also increase, reaching their maximum values at 0.8 m/s. Despite the increase in navigation errors caused by higher speeds, the robot is still able to successfully complete navigation tasks, with overall accuracy remaining within an acceptable range.

3.5.2. Straight-Line Walking Stability Test Inside the Sheep Shed

To verify the driving stability of the sheep feeding robot inside the sheep shed, this experiment used an industrial control computer to control the robot's straight-line driving commands. The robot uses a laser radar for environmental perception and monitors the accuracy of its driving path using real-time point cloud data to ensure it can travel along the predefined straight path while maintaining a 585 cm safety distance from the sheep shed walls. In this experiment, the robot traveled a distance of 60 m, with laser radar data measurements taken every 1 m to obtain the real-time distance values between the robot and the sheep shed walls.

Throughout the experiment, LiDAR data was analyzed to monitor and quantify the variations in the distance between the robot and the wall during its motion. The results showed that throughout the 60 m journey, the robot maintained a distance of approximately 585 cm from the wall, with a maximum deviation of 2.2 cm. This deviation was within the set safety range, indicating that the robot was able to travel steadily along the predetermined path and effectively avoid the risk of collision with the fence.

Through this experiment, the precision and stability of the robot's straight-line driving control system within the sheep shed were validated, proving that the system can maintain high driving accuracy in practical applications, thereby ensuring the robot's safety and reliability within the sheep shed.

3.5.3. Feed Distribution Test

Based on the general conditions of the sheep farm, a feed distribution test was conducted on five sheep pens, with 30 kg of feed distributed to each pen. The results of the feed distribution are shown in Figure 15. As can be seen from Figure 12, due to the stability of the sheep feeding robot's straight-line movement, all the feed was precisely distributed into the feed troughs.

At the same time, in order to test the effectiveness of the sheep feeding robot under special conditions, namely when feeding 60 kg of feed to each pen, another sheep shed that had not undergone feeding was selected to conduct a feeding trial on five pens. The results of the trial are shown in Figure 16.



Figure 15. Effectiveness of feeding 30 kg of feed per sheep pen.



Figure 16. Effectiveness of feeding 60 kg of feed per sheep pen.

As shown in Figure 16, some feed will fall outside the feed trough. Therefore, collect the feed particles that have fallen outside the feed trough every 1 m and weigh them using an electronic scale, as shown in Figure 17.



Figure 17. Electronic Scale Weighing.

The five sheep pens are 100 m long, so the results of 100 weighings were statistically analyzed. Within a 1 m interval, the maximum amount of feed scattered outside the feed

trough was 28.8 g. A total of 300 kg of feed was administered, with 1.52 kg scattered outside the feed trough. Therefore, the feed waste rate was 0.51%, within the acceptable error range.

The feed distribution experiment validated the feeding accuracy of the sheep feeding robot used in this study during the feeding process. Precise feed distribution is a critical component in improving feeding efficiency and reducing resource waste. Especially in sheep farm management, precise feeding technology not only enhances the growth performance of sheep but also effectively reduces feed waste and lowers production costs.

4. Conclusions

With the widespread adoption of intelligent technologies in animal husbandry, traditional manual feeding methods can no longer meet the demands for precision and efficiency in modern sheep farming. To address this gap, this study proposes an intelligent robotic feeding system designed to enhance feeding efficiency, reduce labor intensity, and enable precise feed delivery. This system, developed on the ROS platform, integrates LiDAR-based SLAM, YOLOv8s sign recognition, and the bidirectional RRT* algorithm to achieve autonomous feeding in dynamic environments.

Experimental results show that the proposed system achieves centimeter-level accuracy in localization and attitude control, with FAST-LIO2 maintaining an attitude angle error within 1°. Compared to the baseline system, the proposed system reduces node count by 17.67%, shortens path length by 0.58 cm, and reduces computation time by 42.97%. In a 30 kg feeding task, the system demonstrates zero feed wastage, further validating its potential for precise feeding.

However, despite the significant progress achieved by the sheep feeding robot system in indoor operations within the sheep pen, some limitations remain. First, the system relies on LiDAR and cameras for environmental perception, which may encounter stability issues under complex lighting conditions and in extreme environments. Future research could explore the integration of additional sensor types, such as infrared or ultrasonic sensors, to enhance the system's robustness. Second, while the improved bidirectional RRT* algorithm has enhanced path planning, its efficiency remains low in areas with high-density obstacles. Further optimization of the algorithm is necessary to improve computational efficiency and the responsiveness of path planning. Regarding precise feeding, the current sign recognition model shows reduced accuracy when signs are damaged or obstructed. Future work could consider incorporating image enhancement techniques to improve recognition performance under such conditions. Finally, although a multi-sensor fusion control system was implemented, the system's real-time performance and stability still need further validation. Future research will focus on improving real-time processing capabilities and ensuring stability in dynamic environments.

Programming a sheep-feeding robot presents greater challenges compared to traditional industrial robots. While industrial robots typically operate in controlled environments, following fixed tasks, sheep-feeding robots must navigate dynamic and unpredictable environments. These include uneven terrain, changing lighting conditions, and moving obstacles, requiring real-time processing of data from multiple sensors such as LiDAR, cameras, and temperature sensors. Furthermore, sheep-feeding robots must handle complex tasks such as dealing with damaged or obstructed signs and performing dynamic obstacle avoidance, placing higher demands on programming. In contrast to industrial robots, sheep-feeding robots need stronger adaptability and more precise control to ensure stable operation in real-time environments.

Overall, programming a sheep-feeding robot is far more complex than programming a traditional industrial robot, involving multi-sensor fusion, real-time decision-making, and efficient operation in uncertain environments. As technology continues to evolve,

we believe these challenges will be gradually overcome, promoting the development and application of intelligent farming systems, particularly in the management of large-scale sheep farms, offering significant practical value.

Author Contributions: Conceptualization, H.J. and G.C.; Methodology, H.J.; Software, H.J.; Validation, H.J. and G.C.; Formal Analysis, H.J.; Investigation, H.J.; Resources, H.J.; Data Curation, H.J.; Writing—Original Draft Preparation, H.J.; Writing—Review and Editing, H.J.; Visualization, H.J.; Supervision, H.L. All authors have read and agreed to the published version of the manuscript.

Funding: No funding was received for this study.

Data Availability Statement: The data supporting the conclusions of this study are not publicly available due to size limitations, but they can be obtained from the corresponding author upon reasonable request.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Trioliet. Triomatic HP Suspended Feeding Robot [EB/OL]. Available online: https://www.trioliet.com/products/automatic-feeding-systems/feeding-robot/suspended-feeding-robot (accessed on 10 March 2022).
2. Piwcznski, D.; Siatka, K.; Sitkowska, B.; Kolenda, M.; Özkaya, S.; Gondek, J. Comparison of selected parameters of automated milking in dairy cattle barns equipped with a concentrate feeding system. *Animal* **2023**, *17*, 1751–7311. [CrossRef]
3. Bae, J.; Park, S.; Jeon, K.; Choi, J.Y. Autonomous System of TMR (Total Mixed Ration) Feed Feeding Robot for Smart Cattle Farm. *Int. J. Precis. Eng. Manuf.* **2023**, *24*, 423–433. [CrossRef]
4. Mikhailichenko, S.M.; Kupreenko, A.I.; Ivanov, Y.G.; Nikitin, E.A. Optimization of Volume for an Automatic Feed Wagon by Graph Theory Based Modeling. *Agric. Mach. Technol.* **2023**, *17*, 35–41. [CrossRef]
5. Shan, T.; Englot, B.; Ratti, C.; Rus, D. Lvi-sam: Tightly-coupled lidar-visual-inertial odometry via smoothing and mapping. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Xi'an, China, 30 May–5 June 2021; pp. 5692–5698.
6. Tagliabue, A.; Tordesillas, J.; Cai, X.; Santamaria-Navarro, A.; How, J.P.; Carlone, L.; Agha-Mohammadi, A.A. Lion: Lidar-inertial observability-aware navigator for vision-denied environments. In Proceedings of the Experimental Robotics: The 17th International Symposium, La Valletta, Malta, 11–15 November 2021; pp. 380–390.
7. Trybala, P.; Szrek, J.; Dębogórski, B.; Ziętek, B.; Blachowski, J.; Wodecki, J.; Zimroz, R. Analysis of Lidar Actuator System Influence on the Quality of Dense 3D Point Cloud Obtained with SLAM. *Sensors* **2023**, *23*, 721. [CrossRef]
8. Le, J.; Komatsu, R.; Shinozaki, M.; Kitajima, T.; Asama, H.; An, Q.; Yamashita, A. Switch-SLAM: Switching-Based LiDAR-Inertial-Visual SLAM for Degenerate Environments. *IEEE Robot. Autom. Lett.* **2024**, *9*, 7270–7277. [CrossRef]
9. Gkillas, A.; Aris, S.; Lalos, E.; Markakis, K.; Politis, I. A Federated Deep Unrolling Method for Lidar Super-Resolution: Benefits in SLAM. *IEEE Trans. Intell. Veh.* **2024**, *9*, 199–215. [CrossRef]
10. Zhang, D.; Tan, W.; Zelek, J.; Ma, L.; Li, J. SLAM-TSM: Enhanced Indoor LiDAR SLAM With Total Station Measurements for Accurate Trajectory Estimation. *IEEE Trans. Intell. Transp. Syst.* **2025**, *26*, 1743–1753. [CrossRef]
11. Wang, W.; Wang, C.; Liu, J.; Su, X.; Luo, B.; Zhang, C. HVL-SLAM: Hybrid Vision and LiDAR Fusion for SLAM. *IEEE Trans. Geosci. Remote Sens.* **2024**, *62*, 1–14. [CrossRef]
12. Wu, W.; Chen, C.; Yang, B.; Zou, X.; Liang, F.; Xu, Y.; He, X. DALI-SLAM: Degeneracy-aware LiDAR-inertial SLAM with novel distortion correction and accurate multi-constraint pose graph optimization. *ISPRS J. Photogramm. Remote Sens.* **2025**, *221*, 92–108. [CrossRef]
13. Shen, B.; Xie, W.; Peng, X.; Qiao, X.; Guo, Z. LIO-SAM++: A Lidar-Inertial Semantic SLAM with Association Optimization and Keyframe Selection. *Sensors* **2024**, *24*, 7546. [CrossRef] [PubMed]
14. Cao, F.; Wang, S.; Chen, X.; Wang, T.; Liu, L. BEV-LSLAM: A Novel and Compact BEV LiDAR SLAM for Outdoor Environment. *IEEE Robot. Autom. Lett.* **2025**, *10*, 2462–2469. [CrossRef]
15. Lin, J.; Zheng, C.; Xu, W.; Zhang, F. R2LIVE: A Robust, Real-Time, LiDAR-Inertial-Visual Tightly Coupled State Estimator and Mapping. *IEEE Robot. Autom. Lett.* **2021**, *6*, 7469–7476. [CrossRef]
16. Lin, J.; Zhang, F. R3LIVE: A Robust, Real-Time, RGB-Colored, LiDAR-Inertial-Visual Tightly Coupled State Estimation and Mapping Package. In Proceedings of the International Conference on Robotics and Automation (ICRA), Philadelphia, PA, USA, 23–27 May 2022; pp. 10672–10678.

17. Wang, H.; Wang, C.; Chen, C.L.; Xie, L.H. F-LOAM: Fast LiDAR Odometry and Mapping. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Prague, Czech Republic, 27 September–1 October 2021; pp. 4390–4396.
18. Agarwal, S.; Mierle, K. Ceres solver: Tutorial & reference. *Google Inc* **2012**, *2*, 8.
19. Ganesan, S.; Natarajan, S.K. A novel directional sampling-based path planning algorithm for ambient intelligence navigation scheme in autonomous mobile robots. *J. Ambient. Intell. Smart Environ.* **2023**, *15*, 269–284. [CrossRef]
20. Li, K.; Gong, X.; Muhammad, T.; Wang, T.; Rajesh, K. Towards Path Planning Algorithm Combining with A-Star Algorithm and Dynamic Window Approach Algorithm. *Int. J. Adv. Comput. Sci. Appl.* **2023**, *14*, 511–519. [CrossRef]
21. Akay, R.; Yildirim, M.Y. SBA*: An efficient method for 3D path planning of unmanned vehicles. *Math. Comput. Simul.* **2025**, *231*, 294–317. [CrossRef]
22. Jiang, S.; Sun, S.; Li, C. Path Planning for Outdoor Mobile Robots Based on IDDQN. *IEEE Access* **2024**, *12*, 51012–51025. [CrossRef]
23. Hu, W.; Zhang, Q.; Ye, S. An enhanced dung beetle optimizer with multiple strategies for robot path planning. *Sci. Rep.* **2025**, *15*, 4655. [CrossRef] [PubMed]
24. Andrea, F.; Maria-Cristina, R.; Elizabeth, M. Robots in Partially Known and Unknown Environments: A Simulated Annealing Approach for Re-Planning. *Appl. Sci.* **2024**, *14*, 10644. [CrossRef]
25. Wijegunawardana, I.D.; Samarakoon, S.B.P.; Muthugala, M.V.J.; Elara, M.R. Risk-Aware Complete Coverage Path Planning Using Reinforcement Learning. *IEEE Trans. Syst. Man Cybern. Syst.* **2025**, *55*, 2476–2488. [CrossRef]
26. De Carvalho, K.B.; de OB Batista, H.; Fagundes-Junior, L.A.; de Oliveira, I.R.L.; Brandão, A.S. Q-learning global path planning for UAV navigation with pondered priorities. *Intell. Syst. Appl.* **2025**, *25*, 200485. [CrossRef]
27. Vikram, C.; Vikram, C.; Mirihagalla, M.D.M.; Yeo, M.S.; Zeng, Z.; Borusu, C.S.C.S.; Muthugala, M.V.J.; Elara, M.R. Door-Density-Aware Path Planning. *IEEE Access* **2024**, *12*, 136880–136888. [CrossRef]
28. Ammar, A. ERA*: Enhanced Relaxed A* algorithm for solving the shortest path problem in regular grid maps. *Inf. Sci.* **2024**, *657*, 120000. [CrossRef]
29. Saati, T.; Albitar, C.; Jafar, A. An Improved Path Planning Algorithm for Indoor Mobile Robots in Partially-Known Environments. *Autom. Control. Comput. Sci.* **2023**, *57*, 1–13. [CrossRef]
30. Zhang, R.; Xu, Q.; Su, Y.; Chen, R.; Sun, K.; Li, F.; Zhang, G. CPP: A path planning method taking into account obstacle shadow hiding. *Complex Intell. Syst.* **2025**, *11*, 129. [CrossRef]
31. Dai, J.; Li, D.; Zhao, J.; Li, Y. Autonomous navigation of robots based on the improved informed-RRT algorithm and DWA. *J. Robot.* **2022**, *2022*, 3477265. [CrossRef]
32. Ding, J.; Zhou, Y.; Huang, X.; Song, K.; Lu, S.; Wang, L. An Improved RRT* Algorithm for Robot Path Planning Based on Path Expansion Heuristic Sampling. *J. Comput. Sci.* **2022**, *67*, 101937. [CrossRef]
33. He, P.F.; Fan, P.F.; Wu, S.E.; Zhang, Y. Research on Path Planning Based on Bidirectional A* Algorithm. *IEEE Access* **2024**, *12*, 109625–109633. [CrossRef]
34. Dong, L. Improved A* Algorithm for Intelligent Navigation Path Planning. *Informatica* **2024**, *48*, 181–194. [CrossRef]
35. Yang, F.; Fan, J. Based on the heuristic bias method of efficient algorithm of RRT—Connect. In Proceedings of the 2023 2nd International Conference on Artificial Intelligence and Computer Information Technology, AICIT, Yichang, China, 15–17 September 2023; pp. 1–4.
36. Ogiwara, Y.; Yorozu, A.; Ohya, A.; Kawashima, H. Making ros tf transactional. In Proceedings of the 2022 ACM/IEEE 13th International Conference on Cyber-Physical Systems (ICCPS), Milan, Italy, 4–6 May 2022; pp. 318–319.

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.