



# Design and Implementation of a Wheeled Robot with Obstacle Avoidance and Navigation Capabilities for Drug Delivery

Zhenghao Yu

Faculty of Engineering, The University of Hong Kong, 999077 Pok fu lam, Hong Kong, China  
zhenghao@connect.hku.hk

**Abstract.** In medical settings, wheeled robots with obstacle avoidance and navigation capabilities have become a key enabler for automated precision delivery, effectively addressing the need for efficient medical logistics and reducing the workload of nursing staff. However, significant technical challenges persist in this field like obstacle misdetection rates reach 18% under complex lighting conditions, hindering safe navigation, resolving path conflicts among multiple robots takes an average of 4.2 seconds, impacting collaborative efficiency and insufficient elderly-friendly interaction design results in a 21% error rate among elderly users. These limitations collectively restrict the widespread adoption of robotic solutions in dynamic medical environments. To address these challenges, this study proposes a novel wheel-leg hybrid mechanism that combines the mobility of wheeled movement with the terrain adaptability of legged systems. It integrates multi-modal sensor fusion technology, including lidar, ultrasonic sensors, and visual cameras, to achieve high-precision environmental perception. For path planning, it employs a strategy combining the optimized A\* algorithm with the dynamic window method, supporting real-time global path optimization and local obstacle avoidance. The integrated system aims to demonstrate significant performance improvements including enhancing obstacle recognition accuracy, controlling path re-planning latency, decreasing the proportion of nurses' non-clinical work time and increasing delivery efficiency, expanding application scenarios to multi-story hospital environments and outdoor communities. The paper will mainly address key limitations in the field of medical robotics through a systematic approach, providing a robust technical framework to drive the development of automated medical delivery systems toward higher reliability and usability.

**Keywords:** Wheeled-legged robot, Obstacle avoidance, Medical delivery, Multi-sensor fusion, Path planning.

## 1 Introduction

Currently, medication delivery robots on the market are intelligent medical devices with autonomous navigation at their core. They integrate multimodal sensing components such as laser radar (LiDAR), ultrasonic sensors, and visual cameras, combined with

© The Author(s) 2026

S. Zhang (ed.), *Proceedings of the 2025 International Conference on Electronics, Electrical and Grid Technology (ICEEGT 2025)*, Advances in Engineering Research 292,

[https://doi.org/10.2991/978-94-6463-986-5\\_36](https://doi.org/10.2991/978-94-6463-986-5_36)

rapid exploration random tree (RRT) algorithms, to achieve automated and accurate delivery of medications from storage points to patients [1].

Its core features are manifested in three aspects. Firstly, the wheel-foot composite motion mechanism provides dynamic environmental adaptability. By adjusting the chassis height (e.g., the standard structure of 397 mm can overcome a 5 cm step), it overcomes the traditional limitations of wheeled robots in navigating obstacles such as thresholds and cables, making it particularly suitable for narrow environments such as hospital corridors (1.8–2.4 m wide) and elevators [1]. Secondly, real-time path planning based on the RRT\* algorithm and multi-sensor data fusion (LiDAR point cloud and visual images) achieve obstacle recognition accuracy of 95.8% in dynamic crowds, with path re-planning latency  $\leq 0.3$  seconds, ensuring safe obstacle avoidance in narrow spaces [1,2]. Finally, a task-closed-loop management system is established through a human-machine interface (HMI) or voice commands, achieving full automation from medication loading, path execution to patient identity verification, with a human intervention error rate  $< 5\%$  [2].

The standardized workflow of medication delivery robots begins with order acquisition from the hospital information system (HIS), followed by the generation of the optimal global path through a grid map (average planning time  $\leq 2$  minutes). During movement, the Robots in Real Time (RRT) algorithm is used to avoid dynamic obstacles such as wheelchairs and IV stands in near real time which the response delay is no more than 0.3 seconds. Finally, visual recognition of wristbands completes medication delivery and data synchronization [1]. Its application value lies in reducing nurses' non-clinical work time from 30% to 12% through 24/7 continuous operation, while improving delivery efficiency by 60% [2]. In isolation ward scenarios, non-contact delivery and hydrogen peroxide vapor disinfection modules achieve a surface bacterial colony removal rate of 92.3%, while sub-meter-level positioning technology (error  $< 5$  cm) reduces medication delivery errors by 98% [3,4]. Additionally, the terrain adaptability of the wheel-and-leg composite structure (e.g., navigating a  $15^\circ$  slope and 8 cm steps) expands its application from single-story hospitals to multi-story buildings and outdoor communities [1,5].

From the current state of research, European and American teams focus on multi-robot collaboration and the integration of infection control technologies. For example, the German PeTRA project uses a Behavior Tree architecture to achieve multi-task scheduling. Its wheeled robots have completed over 100 hours of testing in three hospitals, with a response time of  $\leq 1.2$  seconds for patient fall detection in emergency situations [2]. Meanwhile, the Italian HOSBOT project achieves surface disinfection through low-temperature hydrogen peroxide vaporization, but its disinfection rate for hard-to-reach areas such as drawer crevices is only 68%, requiring mechanical wiping to improve effectiveness [3]. Domestic research, however, focuses on institutional innovation and scenario implementation. For example, the four-legged robot developed by Yangzhou Customs and Yushu Technology uses AI posture control algorithms to autonomously navigate around obstacles at a speed of 0.5 m/s on wet decks, achieving a 91% success rate in obstacle avoidance [5]. Domestically manufactured isolation ward robots utilize UWB positioning and voice interaction, improving nurse satisfaction with operations to 85%. However, the error rate for elderly patients following voice

commands remains at 21%, highlighting the necessity of optimizing human-machine interface design [4]. In summarizing, current technical challenges include obstacle misdetection rates of 18% in complex lighting condition, average path conflict resolution times of 4.2 seconds for multi-robot coordination, and insufficient aging-friendly interaction design [1,2,4].

With the rapid development of artificial intelligence and robotics, service robots equipped with self-balancing and autonomous navigation and obstacle avoidance capabilities have demonstrated broad application prospects in complex dynamic environments such as healthcare and logistics. Especially in scenarios with high requirements for stability and flexibility, such as hospitals, robots not only need to have good self-balancing performance but also need to achieve efficient path planning and real-time obstacle avoidance capabilities to complete tasks such as material transportation and patient guidance. However, most current research focuses on the realization of single functions, lacking integration and optimization of the overall system architecture. Additionally, some solutions rely on high-cost sensors or simplified environmental models, making it difficult to balance performance and practicality. Therefore, this paper aims to design a low-cost, high-performance, and functionally integrated mobile robot system. Through reasonable hardware selection, control algorithm optimization, and system integration, the system's adaptability and reliability in complex environments will be enhanced.

## 2 System design

### 2.1 Hardware design and modelling

**Overall system design.** The system adopts a “hardware modularization + software layered control” architecture design, achieving terrain adaptability and motion flexibility through a wheel-foot composite motion institution. Combined with multi-modal sensor fusion and inertial measurement units, it constructs an environment perception and attitude monitoring system. At the control level, the system integrates global path planning (optimized A algorithm) and local dynamic obstacle avoidance (dynamic window method) to achieve real-time path re-planning. The software framework is developed based on ROS 2, supporting sensor data fusion, self-balancing control, and task closed-loop management. The overall design balances cost and performance, enhancing portability through lightweight materials and a modular structure, making it suitable for narrow environments such as hospital corridors and elevators, as well as multi-story building environments.

**Drive system.** The drive system consists of a wheel-legged drive module and a base drive motor, which together enable the robot to move flexibly on different terrains. The wheel-legged drive module uses an MG995 servo motor with a planetary gearbox (torque 15 kg · cm, speed 60 rpm) to control the movement of the wheel-legged joints,

supporting both wheeled high-speed translation and legged obstacle climbing modes. Drawing inspiration from the drive structure design of the four-legged robot ANYmal, closed-loop motor control is employed to achieve smooth transitions between gaits [6]. In wheel mode, the maximum speed reaches 3 m/s, while in leg mode, the robot can overcome obstacles as high as 8 cm, significantly enhancing its adaptability to complex terrains.

Meanwhile, the chassis drive motor uses a JGB37-520 DC reduction motor (power 30 W, reduction ratio 1:50), which is combined with differential control to achieve flexible steering and stable movement. It has a maximum climbing ability of  $15^\circ$  and fully meets the requirements of typical application scenarios such as hospital corridors with slopes  $\leq 5^\circ$  and common outdoor gentle slopes.

**Perception and navigation components.** It employs a multi-modal sensor fusion strategy to achieve high-precision perception of the surrounding environment. The LiDAR system uses the RP-LiDAR A3, which features  $360^\circ$  scanning capability and  $\pm 2$  cm ranging accuracy, to construct a 2D occupancy grid map. Combined with SLAM technology, it enables centimeter-level localization, providing a reliable map foundation for path planning. Additionally, the system is equipped with four HC-SR04 ultrasonic sensors, with a ranging distance of approximately 2–400 cm, for close-range obstacle detection. It also integrates two Sharp GP2Y0 infrared cameras to achieve stereo vision ranging, enhancing the system's ability to identify dynamic obstacles and improving overall robustness [6].

To ensure stable operation of the robot in complex environments, the system integrates an MPU6050 inertial measurement unit to collect body acceleration and angular velocity information in real time and accurately monitor attitude changes. Through PID control algorithm feedback, the wheel-foot joint angle is adjusted to achieve self-balancing control, keeping the body tilt angle error within  $\pm 2^\circ$ , effectively improving the movement stability of the robot in both wheel and foot modes. This module provides key support for the flexible movement and safe navigation of the robot.

**Power supply and master control.** Four 18650 lithium-ion battery packs (total capacity is 5200 mAh) paired with a BMS battery management system provide the robot with over 4 hours of runtime and support fast charging technology for a full charge in two hours, meeting the demands of prolonged continuous operation. The main control platform adopts the StackForce motherboard, based on the high-performance STM32H750XB microcontroller with a clock speed of 480 MHz, and integrates the ROS 2 framework. It is responsible for coordinating core tasks such as sensor data collection, path planning, and motion control. The system also supports remote monitoring and firmware upgrades via WiFi, enhancing operational flexibility and maintenance convenience.

**Structure and modular design.** The robot's main body is designed using lightweight materials. The chassis frame is made of 3D-printed nylon material with a density of  $1.15 \text{ g/cm}^3$  and a tensile strength of 60 MPa. Critical load-bearing components such as

wheel axles and joint connectors are constructed from aerospace-grade aluminium alloy 6061-T6 with density of  $2.7 \text{ g/cm}^3$  and yield strength of 276 MPa. As a result, the total weight of the robot does not exceed 15 kg, while its load capacity is greater than or equal to 10 kg, effectively balancing strength and portability. The system features a modular design, including a magnetic-latch cargo compartment (dimensions:  $200 \text{ mm} \times 200 \text{ mm} \times 80 \text{ mm}$ ) with built-in pressure sensors to monitor load status. Sensor mounts are strategically positioned, with a LiDAR sensor mounted on the top and ultrasonic and infrared sensors distributed around the chassis, enabling  $360^\circ$  obstacle-free environmental awareness.

As shown in Figures 1 and 2, the wheel-foot composite leg structure consists of a single leg composed of a hip joint, a knee joint, and a wheel drive joint, forming a series four-bar linkage mechanism. Through motor-driven control, it achieves flexible switching between “wheel rolling” and “foot walking” modes [6]. The wheel diameter is 120 mm, and the tread surface is made of high-friction rubber material (static friction coefficient  $\mu \geq 0.6$ ), enhancing ground adaptability. Additionally, the chassis employs a diamond-shaped link adjustment mechanism, enabling dynamic height adjustment between 150 mm and 200 mm (raising the chassis to clear obstacles and lowering it afterward).

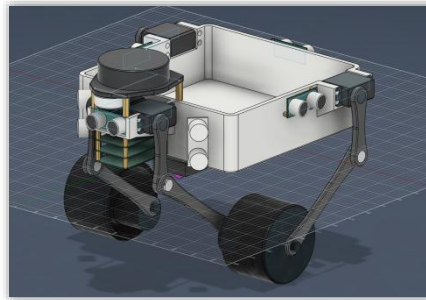


Fig. 1. 3D model top view.

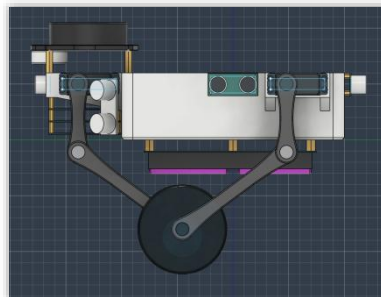


Fig. 2. Left view of 3D model.

## 2.2 Software system design

**Keeping balance in statistics and dynamics.** Balance control is the foundation for the stable operation of wheeled robots in complex environments. Its core lies in real-time perception of the body posture and dynamic adjustment of the wheel drive torque to ensure that the projection of the center of gravity always remains within the support polygon. This study establishes a mathematical modelling based on proportional control principles. By utilizing real-time sensor data to obtain the deviation between the pitch angle and the target speed, the required restoring torque is calculated to achieve decoupling between attitude stability and speed control. Specifically, a hierarchical control strategy is employed to fuse multi-sensor data (such as inertial measurement units and lidar) with a dynamic torque distribution algorithm to compensate for center of gravity shifts caused by wheel-legged oscillations, thereby maintaining the body's lateral attitude [7].

*Attitude Sensing and Proportional Control Models.* The robot uses an MPU6050 inertial measurement unit to continuously monitor the pitch angle and roll angle of the robot body in real time. By integrating data from wheel encoders, it calculates the average speed (speedAvg) and establishes a feedback control system. To further enhance attitude estimation accuracy, the system employs an Extended Kalman Filter (EKF) algorithm to fuse LiDAR point cloud data, enabling real-time estimation of the yaw angle ( $\beta$ ) and yaw rate ( $\omega_z$ ). This method effectively improves attitude estimation accuracy under complex motion conditions. Experimental results show that the lateral slip angle error is controlled within  $\pm 0.2^\circ$ , and the maximum error of the yaw rate does not exceed 0.0075 rad/s [8].

Using the finely tuned proportional coefficient  $K_p$ , an accurate linear relationship between the robot's pitch angle and restoring torque is established. However, TargetAngle is not a fixed value, but is flexibly derived based on the target speed. For example, when the robot needs to brake urgently and the target speed drops to 0, a reverse pitch angle braking strategy can be used, whereby the target angle (targetAngle) is dynamically adjusted according to the desired speed.

In the steering control, a differential torque (turnTorque) mechanism is introduced, which uses the yaw rate (GyroZ) measured by the gyroscope to adjust the output torque of the two wheels axles.

$$\text{turnTorque} = K_p \times (\text{robotMotion.turn} - \text{robotPose.GyroZ}) \quad (1)$$

The final torque outputs of the left and right wheels are:

$$\text{Torque1} = K_p \times (\text{targetAngle} - \text{pitch}) + \text{turnTorque} \quad (2)$$

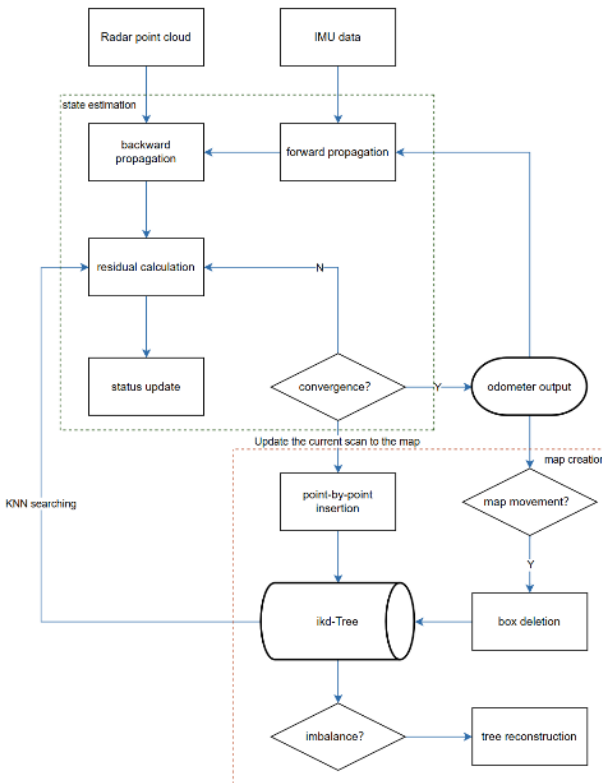
$$\text{Torque2} = K_p \times (\text{targetAngle} - \text{pitch}) - \text{turnTorque} \quad (3)$$

*Multi-mode motion control strategy.* The robot supports both wheeled and legged motion modes to adapt to different ground conditions. On flat surfaces, such as hospital corridors, it uses a two-wheel drive mode with differential control for flexible steering,

achieving an average speed of 2.5 m/s and keeping path tracking error within 10 cm. The system employs a PID controller to dynamically regulate the total drive torque ( $T_e$ ), ensuring the robot can precisely follow the planned path with a speed tracking error below 1.2%, thereby significantly enhancing driving stability and positioning accuracy [8].

When encountering complex terrain such as steps less than 8 cm high or slopes with a gradient of less than  $15^\circ$ , the robot can automatically switch to a bipedal support mode. It uses inverse kinematics algorithms to calculate the target angles of each joint in real time, enabling stable gait control. The maximum single-leg lift height reaches 10 cm, ensuring smooth obstacle crossing. In this mode, the robot coordinates the cooperation of leg and wheel joints to maintain a smooth transition of the center of gravity during transitions, enhancing its mobility and adaptability in unstructured environments.

**Indoor Navigation.** In terms of the mapping module, this system mainly relies on radar point cloud, IMU acceleration, and angular velocity data. The main functions of this system are real-time robot position and attitude estimation and map management. The entire mapping process is shown in Figure 3.



**Fig. 3.** Flowchart of the graph building algorithm

It primarily constructs two key models. One is a state transition model, which utilises an IMU measurement model, a motion model, and an error model to construct state transition equations. By integrating the IMU measurement model, the kinematic model, and the error model, a discrete-time state transition equation is established:

$$\mathbf{x}_k = f(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}) + \omega_{k-1} \quad (4)$$

Where  $\mathbf{x}_k$  is the state vector at time  $k$  (containing position, velocity, and attitude),  $\mathbf{u}_k$  is the control input (such as motor speed), and  $\omega_k$  the process noise following a normal distribution [9]. This model captures the instantaneous changes in the robot's motion through high-frequency sampling of IMU data, providing a dynamic foundation for pose prediction.

Subsequently, an observation model is constructed based on IMU measurements to compensate for motion distortion. Assuming that laser points are uniformly distributed on local plane blocks, implicit observation equations are established.

$$r_{ij} = z_{ij} - h(\mathbf{x}_k, p_j) \quad (5)$$

where  $z_{ij}$  are laser point measurements,  $p_j$  are map points, and  $r_{ij}$  are point-to-surface distance residuals [7]. By minimising the sum of squared residuals, such as the Huber loss function, geometric alignment between the laser point cloud and the map is achieved.

During the optimisation process, the state is estimated mainly through iterative Kalman filtering, which is divided into two stages. The first stage is forward propagation. When each IMU measurement value arrives, the forward propagation prediction of the current state and covariance is performed according to the state transition model, and the mean value and covariance of the state are updated [10]:

$$\hat{\mathbf{x}}_{k|k-1} = f(\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_{k-1}) \quad (6)$$

$$\mathbf{P}_{k|k-1} = \mathbf{F}_{k-1} \mathbf{P}_{k-1|k-1} \mathbf{F}_{k-1}^T + \mathbf{Q}_{k-1} \quad (7)$$

The Jacobian matrix  $\mathbf{F}_{k-1}$  represents state transfer matrix, and the process noise covariance matrix is  $\mathbf{Q}_{k-1}$ . The second stage is the backward propagation. When the LiDAR scan frame arrives, the point cloud timestamps are aligned to the end of the scan using IMU pre-integration to compensate for motion distortion. The pose is iteratively optimised using the Gauss-Newton method to solve the incremental transformation matrix  $\Delta T$ . Through the projection function, the current frame and the map point cloud ( $p_i$  and  $q_i$ ) are obtained [11]:

$$\Delta T = \arg \min_{\Delta T} \sum_{i=1}^n \|\pi(\Delta T \cdot p_i) - q_i\|^2 \quad (8)$$

During the pose optimization process, ikd-Tree will be used for map management. The insertion strategy is to insert new point clouds point by point through voxel hashing only for non-overlapping areas after the state is updated, avoiding duplicate storage and reducing computational redundancy. Similarly, the deletion strategy is to remove historical map points that are more than 2 meters away from the current position through the box deletion method when the odometer accumulates movement exceeding a certain threshold, reducing the interference of invalid data on positioning. Furthermore, the

reconstruction strategy involves real-time monitoring of the tree height and balance. When the imbalance exceeds the threshold (e.g., the number of nodes in the left subtree minus the number of nodes in the right subtree  $> 1$ ), reconstruction is automatically triggered to reduce the time required for operations such as queries, insertions, and deletions, thereby optimizing query efficiency [12,13].

After establishing a map based on point cloud information provided by sensors installed on the robot, the next step is to determine how to locate (re-locate) the robot in the established map using sensor data collected in real time. The solution is to divide the re-location problem into two steps: first, perform rough scene recognition, and then perform fine pose regression. Scene recognition is firstly used to determine the initial position and orientation of the robot. Typically, the initial position and orientation of the vehicle are unknown. Without the assistance of external devices or when indoor GPS signals are inaccurate, it is necessary to use sensors to collect environmental information and autonomously determine the position and orientation. The current solution is based on particle filtering for scene recognition, which is achieved through the following steps [12]:

- (1) Particle initialisation: uniformly distribute the particle set  $\{x_k^{(i)}\}$  in the state space.
- (2) Prediction update: Generate prediction particles based on the motion model  $x_k^{(i)} = f(x_{k-1}^{(i)}, u_{k-1})$ .
- (3) Weighting: The NDT matching score of the laser point cloud to the map is used to assign weights to the particles.

$$w_k^{(i)} = \exp\left(-\frac{1}{2\sigma^2} \sum_{p \in P} \|p - T(x_k^{(i)})q\|^2\right) \tag{9}$$

where  $T(x_k^{(i)})$  is the bit-pose transformation matrix, and P and Q are the source and target point clouds, respectively;

- (4) Resampling: a roulette wheel method is used to retain high-weighted particles and maintain particle diversity.

Using particle filtering for localisation is more accurate than simply relying on measurements. However, this method also has disadvantages, namely that it cannot recover position information from robot kidnapping or global localisation failure, and when the number of particles is small, there is a possibility that all particles near the correct position will be discarded.

Therefore, after obtaining the initial coarse positioning information of the wheeled robot, it is necessary to optimize this pose using NDT-based point cloud registration. There are two points clouds, the source point cloud P and the target point cloud Q:

- (1) Divide the space where the source point cloud P is located into unit grids (i.e., the projection of three-dimensional space onto a two-dimensional plane)
- (2) Based on the distribution of points within the divided unit grids, calculate the normal distribution PDF parameters of the unit grids.
- (3) Based on the transformation matrix, the points in the target point cloud Q are transformed.

- (4) The number of points from the target point cloud within the grid of the source point cloud  $P$  is counted, and the corresponding probability distribution function is determined based on the distribution of the points.
- (5) The optimal solution for all points is solved, i.e., the rigid transformation between the target cloud and the source point cloud is determined.
- (6) The score is obtained by squaring the Euclidean distance between the corresponding nearest points of the output cloud and the target cloud. A lower score indicates better matching performance.

**Obstacle avoidance.** Robot path planning and navigation technology is primarily divided into two stages which are global path planning and local path planning. Global path planning involves planning a global path to reach a predetermined destination based on known current location information and obstacle information in the environment map. This stage primarily uses the Dijkstra algorithm and the A\* algorithm. Local path planning is applicable in uncertain environments, where the robot uses information from its own sensors to determine the distribution of obstacles in its local environment and generate a local safe trajectory for obstacle avoidance operations. Moreover, the common local path planning algorithms include the Dynamic Window Algorithm (DWA), Model Predictive Control (MPC), and Time-Elastic Band (TEB) methods. The path planning component of this project employs an improved dynamic window algorithm based on global path planning.

*Optimization of the A\* algorithm.* Given the current location information and a grid map of obstacles in the environment, the A\* graph search algorithm is the simplest and most efficient method for finding the shortest path to a target point. As a classic heuristic search algorithm, the A\* algorithm optimises paths by evaluating the function  $f(n) = g(n) + h(n)$ , where  $g(n)$  is the actual cost from the starting point to the current node, and  $h(n)$  is the heuristic estimated cost from the current node to the target point, such as Manhattan distance or Euclidean distance. This algorithm guarantees path optimality in grid maps but suffers from low computational efficiency. For example, in a map with 60,000 cells, the traditional A\* algorithm takes over 1 hour per computation, and the paths contain numerous turns, resulting in poor smoothness, making it unsuitable for real-time navigation scenarios [14].

To optimise A\*, two optimisation methods, Jump Point Search (JPS) and Basic Theta\* algorithm, are introduced. JPS can prune invalid nodes and reduce the search space. For instance, in a 120-cell scene, JPS evaluates only 8 nodes, with a computation time of only 21 ms, which is more than 10 times faster than traditional A\*, but the path length may increase by 2–5%. The Basic Theta\* algorithm uses ‘visibility’ detection to directly connect non-adjacent traversable nodes, reducing path turning points. This shortens the path length by approximately 5% compared to traditional A\*, but the computation time increases by 3 times. In contrast, JPS is significantly faster and more suitable for highly symmetrical environments, such as hospital corridors. However, in low-symmetry environments, Basic Theta\* is 2.28% faster than JPS but requires more time for computation [14].

With the aim of achieving more accurate and faster route planning, it uses layered cost maps to divide the environment into a static layer (pre-built map), an obstacle layer (real-time sensor data), and an inflation layer (safety buffer zone), and achieves dynamic obstacle avoidance through weighting and superposition. In a hospital scenario, the Inflation Layer is set to 3.1 m to ensure the robot maintains a safe distance from obstacles. Meanwhile, the Obstacle Layer is updated in real-time via Bayesian filtering, and IMU data is used to compensate for LiDAR motion distortion, thereby enhancing map accuracy [9,12].

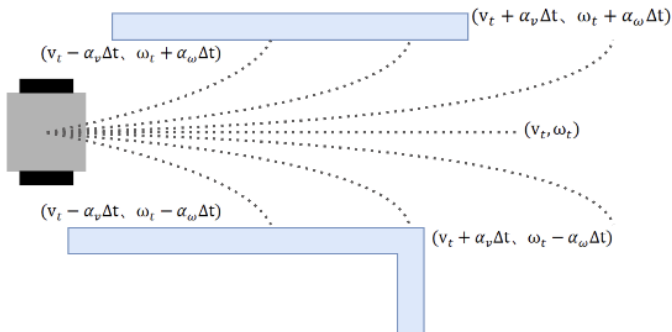
*Trajectory optimisation in dynamic environments.* The dynamic window Algorithm (DWA), due to its advantages in stability and obstacle avoidance, has been widely applied in the field of robot autonomous navigation. The basic idea is to consider factors such as the motion model of the robot, its initial velocity, its hardware conditions, and the environment, calculate the velocity sampling space of the robot under the current conditions, evaluate the sampled velocities using a trajectory evaluation function, select the optimal path, and execute it. Simultaneously, the state of the robot changes, the environmental information is updated, the velocity sampling window is updated, and the next round of planning begins. The global path can serve as a reference value or heuristic for local planning, fully leveraging the superiority of the global path to enable the robot to move toward the target point while avoiding obstacles and generally following the global path.

The dynamic window method primarily consists of velocity sampling and trajectory evaluation. Velocity sampling involves sampling velocities within a constrained velocity window. Since a robot's velocity and angular velocity are not infinite and are subject to obstacle constraints, the velocity sampling space is limited in range.

Speed sampling Mainly considers the movement limit speed constraint, limit acceleration constraint, and obstacle avoidance speed constraint:  $v_m = v \in [v_{\min}, v_{\max}]$ ,  $\omega \in [\omega_{\min}, \omega_{\max}]$ . Among them,  $v_{\min}$  and  $v_{\max}$  represent the minimum and maximum linear speeds of the robot.  $\omega_{\min}$  and  $\omega_{\max}$  represent the minimum and maximum angular speeds of the robot, respectively.

Since the torque of the motor is limited, the acceleration and braking performance of mobile robots is affected by the motor performance. Therefore, the maximum acceleration limits speed sampling to a dynamic window related to the current speed:  $V_d = (v, \omega)_{\substack{v \in [v_c - \dot{v}_b \Delta t, v_c + \dot{v}_a \Delta t] \\ \omega \in [\omega_c - \dot{\omega}_b \Delta t, \omega_c + \dot{\omega}_a \Delta t]}}$ , where  $v_c$  and  $\omega_c$  denote the robot's current linear velocity and angular velocity;  $\dot{v}_b$  and  $\dot{\omega}_b$  denote the robot's maximum deceleration and maximum angular deceleration; and  $\dot{v}_a$  and  $\dot{\omega}_a$  denote the robot's maximum acceleration and maximum angular acceleration.

As for effective obstacle avoidance during movement, the robot must maintain a certain safe distance from obstacles. The obstacle avoidance speed constraint is based on  $V_a = \{(v, \omega) | \sqrt{2 \cdot dist(v, \omega) \cdot \dot{\omega}_b} \Lambda \omega \leq \sqrt{2 \cdot dist(v, \omega) \cdot \dot{\omega}_b}$  where  $dist(v, \omega)$  denotes the minimum distance from the obstacle at the trajectory corresponding to the sampling velocity  $(v, \omega)$ . This condition is selected by considering the distance between the robot and the obstacle and the robot's maximum deceleration, thereby ensuring that the robot can stop before colliding with the obstacle, just like showing in Figure 4.



**Fig. 4.** Velocity Sampling Schematic

The evaluation function can filter out the optimal speed in the speed group that not only enables the starting robot to avoid obstacles in the environment but also moves towards the target point at a relatively fast speed. Trajectory evaluation involves evaluating each trajectory in the sampled speed group using the evaluation function and selecting the optimal speed command. The evaluation function for DWA is expressed as  $(v, \omega) = \sigma(\alpha \cdot heading(v, \omega) + \beta \cdot dist(v, \omega) + \gamma \cdot velocity(v, \omega))$ . Among these,  $heading(v, \omega)$  represents the angle between the direction of the current end point of the trajectory being evaluated and the direction from the current position to the target point, primarily to encourage the robot to continuously align its heading angle toward the target point during movement;  $dist(v, \omega)$  represents the distance between the end point of the trajectory and the global path, indicating the robot's 'adherence' to the global optimal path;  $velocity(v, \omega)$  represents the evaluation function set to achieve rapid arrival at the target, typically denoting the absolute value of the linear velocity of current trajectory.

### 3 Simulation on 2D environment

#### 3.1 Experimental environment construction

Using Python and Matplotlib, a  $50 \times 50$  two-dimensional grid simulation environment that replicates typical hospital features and achieves a similar spatial layout is created. The outer boundary (marked as -1) simulates walls, and the interior contains functional areas such as wards, corridors, and nurses' stations, which are connected by doorways (2-3 grid widths) to form a complex interconnected space.

In obstacle designing, static and dynamic obstacles are added separately. Static obstacles are randomly distributed around medical equipment, hospital beds, and other objects (marked as 1) at a density of approximately 10% to 15% of the grid to simulate common obstructions in hospital corridors. Dynamic obstacles are generated at set intervals by a script to create moving 'pedestrians' (temporary obstacles that move along

preset trajectories) to test the robot's response to dynamic obstacles. Additionally, the simulated robot is capable of diagonal movement adaptation. Its extended environment model supports 8-direction movement (4 orthogonal directions and 4 diagonal directions), with diagonal movement requiring no obstacles between adjacent grid cells (i.e., adjacent diagonal grid cells must be free space). Path loss for different movement directions is distinguished via color-coded paths (orthogonal paths are blue, diagonal paths are cyan).

### 3.2 Parameters and Slant Shift Optimization

This simulation uses Manhattan distance as the heuristic function to calculate the estimated cost  $g(n) = |x_{current} - x_{goal}| + |y_{current} - y_{goal}|$ , with a time complexity of  $O(1)$ , which is computationally efficient and naturally suited to grid environments. For diagonal movement scenarios, path optimization is achieved by separating orthogonal and diagonal movement costs with the orthogonal movement cost  $g_{ortho}$  equals to 1, and the diagonal movement cost  $g_{ortho}$  equals to approximately 1.414. Such design simulates the additional energy consumption caused by turning in actual motion and the algorithm balances path length and energy consumption through weight adjustment, automatically selecting the optimal path combination of short distance priority and low energy consumption priority, thereby ensuring navigation efficiency while reducing robot operating costs.

In an accessible open area, an eight-direction movement strategy reduces path length by approximately 29% compared to traditional four-direction paths. For example, moving from coordinates (0,0) to (4,4), the four-direction path requires 8 units of length, while the diagonal movement path only requires about 5.66 units. When facing an L-shaped obstacle, the traditional four-direction path requires 5 turns to bypass it, while the eight-direction path reduces the number of turns to 3 by moving diagonally, effectively reducing path complexity and the number of times the robot needs to turn.

To avoid collisions between the robot and obstacles during diagonal movements, a dynamic safety buffer zone mechanism is introduced. When the robot expands a 1-grid safety distance (marked as -2) around an obstacle, a prohibited area is clearly defined where the robot is not allowed to enter. The system also synchronously detects the 8-neighborhood range between the current grid and the target grid during diagonal movement to ensure that the minimum distance between the movement path and the obstacle is no less than 2 grids (approximately 0.4 meters, corresponding to the physical size of the robot in real-world scenarios). However, although this mechanism significantly reduces the collision probability during diagonal movements and effectively improves navigation safety, it increases the average path length, revealing the trade-off between safety and efficiency. In practical applications, parameters such as the buffer zone range and obstacle detection threshold can be adjusted to dynamically balance the priorities of the two in different scenarios, thereby adapting to the navigation requirements of complex environments such as hospital corridors and patient rooms.

### 3.3 Simulation verification and result analysis

The simulation results are shown in Figure 5, with six typical output cases selected. Its green point is the start point, red point is the end point, blue line is the path, black point is the obstacle as well as blue '+' is the boundary.

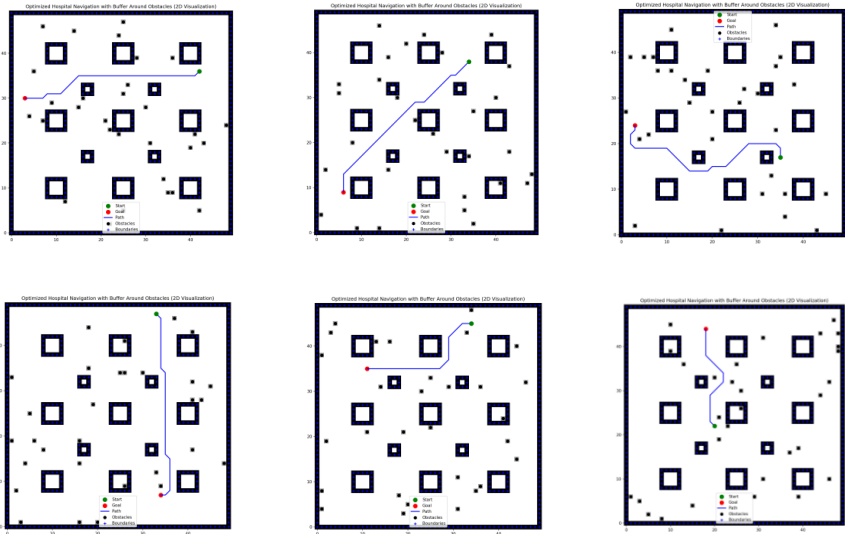


Fig. 5. Velocity Sampling Schematic

From the perspective of scene universality, the algorithm demonstrates two core characteristics: adaptability to complex environments and compatibility with functional requirements. In terms of complex environment adaptation, the algorithm employs spatial semantic modelling and dynamic path evolution mechanisms to actively accommodate the nested building layouts and random obstacle interferences in medical scenarios. It converts wall boundaries, room contours, and other elements into impermeable environmental constraints, reserving safe buffer zones for dynamic obstacles, and constructs a multi-level environmental semantic model. Simultaneously, based on A\* algorithm-inspired heuristic search and priority queuing, it achieves a balanced dynamic evolution of paths from distance cost to safety cost. When encountering nested room obstacles, it can navigate around corridors in unoccupied areas, and when facing random obstacles, it can quickly switch to the next optimal feasible path. At the functional requirement compatibility level, flexible protection design and closed-loop verification synergistically balance the triangle constraints of efficiency, safety, and execution. Through safety distance maintenance and multi-constraint verification mechanisms, the obstacle avoidance function is endowed with an “elastic safety boundary,” which avoids rigid collision risks while accommodating the physical limitations of robots.

Additionally, through path backtracking and physical reachability verification, a complete logical closed-loop of ‘environment modelling - intelligent decision-

making - execution verification' fully evolve the logical closed-loop system. This design breaks through the traditional passive obstacle avoidance model by integrating scene semantics and execution constraints into the core logic of path planning, providing a comprehensive reference from theoretical framework to practical implementation for medical and medical-like scenarios, driving obstacle avoidance strategies from addressing single obstacles to adapting to complex scenario ecosystems, and providing critical support for the deep application of mobile robots in the medical field.

## 4 Conclusion

This study focuses on the demand for automated delivery in medical settings and proposes a drug delivery robot system architecture based on a wheel-leg composite motion mechanism. Through multimodal sensor data fusion technology and optimized path planning algorithms, the system achieves autonomous navigation and dynamic obstacle avoidance in complex environments. The results show that the system has significant advantages in improving the robot's terrain adaptability and task execution efficiency, providing an innovative solution for automated material delivery in smart healthcare. However, there are still several limitations. On the one hand, the 2D simulation environment does not fully consider height changes in three-dimensional space (such as thresholds and slopes), resulting in the applicability of the oblique movement strategy in actual 3D scenarios needing to be further verified in combination with the wheel-leg mechanism kinematic model. On the other hand, the angular range limitations of the wheel-leg joints in the mechanical structure may cause mismatches between the theoretical planning path and the actual execution capabilities.

Future research will introduce the ROS-Gazebo platform to construct a 3D simulation environment containing vertical obstacles to explore the dynamic characteristics of wheel-leg cooperative motion in depth. Secondly, based on inverse kinematics theory, it is needed to optimize the pre-planning algorithm for feasible motion space, improve the feasibility of control strategies by dynamically adjusting path planning weight factors, and integrate AI vision and 5G communication technologies to develop a multimodal navigation system suitable for multi-story buildings and outdoor community scenarios. Additionally, efforts will be made to optimize human-machine interaction interfaces for the elderly, integrate efficient disinfection modules, and study multi-robot collaborative delivery mechanisms. These advancements will drive the development of medical robotics toward intelligent and unmanned systems, providing a more robust technological foundation for the construction of a comprehensive smart healthcare service system.

## References

1. Stoev, P., Fichero, R., & Georgiev, M. Development of a Mobile Robot for Distribution of Medicine in Hospitals [J]. IFAC PapersOnLine, 2024, 58 (3): 221 - 225.

2. Kardas, P., Bielec, F., Brauncajs, M, et al. Evaluation of Disinfection Methods for Autonomous Mobile Robots Used in Hospital Logistics in Emergency Departments [J]. *Journal of Hospital Infection*, 2025, S0195-6701 (25) 00151-3.
3. Zachariae, A., Plahl, F., Tang, Y, et al. Human-robot interactions in autonomous hospital transports [J]. *Robotics and Autonomous Systems*, 2024, 179: 104755.
4. Yoo, H. J., Kim, E. H., & Lee, H. Mobile robots for isolation-room hospital settings: A scenario-based preliminary study [J]. *Computational and Structural Biotechnology Journal*, 2024, 24: 237 - 246.
5. Chand, R., Sharma, B., & Kumar, S. A. Systematic review of mobile robot applications in smart cities with future directions [J]. *Journal of Industrial Information Integration*, 2025, 45: 100821.
6. Zhu, Q., Guan, X., Yu, B., et al. (2024). Overview of structure and drive for wheel-legged robots. *Robotics and Autonomous Systems*, 181, 104777.
7. Xiao, F., Fang, J., Guo, X., Zhang, Y., & Huang, R. (2025). Enhanced dynamic visual SLAM system for hospital logistics robots: Nonlinear optimal filtering, deep learning, and real-time positioning. *Robotics and Autonomous Systems*, 193, 105081. <https://doi.org/10.1016/j.robot.2025.105081>
8. Chen Pinglu, Liu Shengyang, Xu Jing, et al. Stability control of a wheel-legged mobile platform used in hilly orchards[J]. *Biosystems Engineering*, 2025, 256: 104195. DOI: 10.1016/j.biosystemseng.2025.104195.
9. Liu, L., Wang, X., Yang, X., Liu, H., Li, J., & Wang, P. (2023). Path planning techniques for mobile robots: Review and prospect. *Expert Systems With Applications*, 227, 120254. <https://doi.org/10.1016/j.eswa.2023.120254>
10. Zhang, L., Cai, K., Sun, Z., Bing, Z., Wang, C., Figueredo, L., ... Knoll, A. (2025). Motion planning for robotics: A review for sampling-based planners. *Biomimetic Intelligence and Robotics*, 5, 100207. <https://doi.org/10.1016/j.birob.2024.100207>
11. Yao, X., Zhang, B., Wang, X., Su, Y., Cao, G., & Bian, Y. (2025). Adaptive navigation for robots in unstructured agricultural environments using stable feature localization and multi-sensor obstacle detection. *Computers and Electronics in Agriculture*, 234, 110302. <https://doi.org/10.1016/j.compag.2025.110302>
12. Ospina, R., & Itakura, K. (2025). Obstacle detection and avoidance system based on layered costmaps for robot tractors. *Smart Agricultural Technology*, 11, 100973
13. Si, G., Zhang, R., & Jin, X. (2025). Path planning of factory handling robot integrating fuzzy logic-PID control technology. *Systems and Soft Computing*, 7, 200188. <https://doi.org/10.1016/j.sasc.2025.200188>
14. Duchoň, F., Babineca, A., Kajana, M., Florek, M., Fico, T., & Jurišica, L. (2014). Path planning with modified A star algorithm for a mobile robot. *Procedia Engineering*, 96, 59–69. <https://doi.org/10.1016/j.proeng.2014.12.098>

**Open Access** This chapter is licensed under the terms of the Creative Commons Attribution-NonCommercial 4.0 International License (<http://creativecommons.org/licenses/by-nc/4.0/>), which permits any noncommercial use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

