PAPER • **OPEN ACCESS**

Design, simulation, and optimized path planning of a smart mecanum wheelchair using RRT* algorithm

To cite this article: Sadaf Zeeshan *et al* 2025 *Eng. Res. Express* **7** 0452e8

Engineering Research Express

OPEN ACCESS

RECEIVED
1 April 2025

REVISED
11 November 2025

ACCEPTED FOR PUBLICATION
1 December 2025

PUBLISHED
17 December 2025

PAPER

Design, simulation, and optimized path planning of a smart mecanum wheelchair using RRT* algorithm

Sadaf Zeeshan[1], **Muhammad Ali Ijaz Malik**[2,*] **and Shahzaib Aslam**[1]

[1] Department of Mechanical Engineering, University of Central Punjab, Lahore, Pakistan
[2] School of Civil and Environmental Engineering, Faculty of Engineering and Information Technology, University of Technology Sydney, NSW 2007, Australia
* Author to whom any correspondence should be addressed.

E-mail: muhammadaliijaz.malik@student.uts.edu.au

Keywords: intelligent navigation, mecanum wheel, omnidirectional mobility, smart trajectories

Abstract

The present study reports the design and assessment of a Mecanum wheel-based intelligent wheelchair that can navigate independently in dynamic interior environments. The system combines encoder-driven odometry, sensor-based perception, and mechanical design into a single control architecture. Static analysis in ANSYS is used to verify the wheelchair's structural integrity after it has been modelled in SolidWorks to support a 120 kg payload. Omnidirectional mobility is made possible by four Mecanum wheels, which improve manoeuvrability in tight areas. For path design, the quickly exploring Random Tree Star technique is used. It is optimized through node rewiring and B-spline smoothing to produce workable, smooth paths in real time. While quadrature encoders guarantee precise pose estimation with Mecanum-specific inverse kinematics, RGB-D cameras offer dynamic obstacle recognition and ambient mapping. By combining onboard odometry with real-time visual feedback within a closed-loop path planning framework, the current study fills the gap left by previous autonomous wheelchair systems' poor adaptability and controller dependency. Four more challenging navigation situations are used to evaluate the performance. The system obtained success rates of 100%, 98.4%, 97.6%, and 96.7%, respectively, in cases 1–4, displaying reliable obstacle avoidance and constant convergence. With increased node sampling and collision checks, the computational cost rose with complexity (219 ms to 270 ms). For variable surroundings, the wheelchair runtime differed from 72 min to 48 min, while the path time changed from 32 seconds to 48 seconds. The anticipated path is more closely aligned with the actual trajectory when the Root Mean Square Error value is lower (0.015–0.03). These findings demonstrate the system's capacity for precise, flexible, and efficient navigation while highlighting the need for more optimization to lower computational load and improve energy economy for real-time assistive mobility.

Nomenclature

A*	A-Star path planning algorithm
PSO	Particle Swarm Optimization
AGV	Automated Guided Vehicle
PWM	Pulse Width Modulation
ANSYS	Analysis System Software
R	Radius of the Mecanum wheel
BCO	Bee Colony Optimization
RMSE	Root Mean Square Error
BTS	Bipolar Transistor Switch

ω	Angular velocity of the wheelchair
DC	Direct Current
Vx	Longitudinal velocity of the wheelchair
ESP32	Espressif microcontroller
Vy	Lateral velocity of the wheelchair
IQR	Interquartile Range
LM2596s	Linear Monolithic 2596 Switching Regulator
L	Longitudinal distance from the wheelchair's center to wheel
W	Lateral distance from the wheelchair's center to the left and right wheels
RRT*	Rapidly-exploring Random Tree Star algorithm

1. Introduction

Most individuals with mobility impairments face significant difficulties carrying out daily tasks, particularly navigating complex situations [1]. Traditional wheelchairs can show limits in terms of agility and spatial sense, despite their numerous advantages [2]. These limitations are most noticeable in places where precise movement is required, including crowded spaces, narrow passageways, or locations with obstacles [3, 4]. People using conventional wheelchairs, for example, may struggle to maneuver around furniture or other objects in a room, through narrow doors, or around tight corners. It elevates user annoyance and diminishes both the user's independence and the wheelchair's overall usefulness [5]. In this regard, the Mecanum wheelchair represents a turning point in agility technology, giving disabled people a great deal of mobility and independence [6]. Unlike traditional wheelchairs, which are limited to forward, backward, and turning motions only, the Mecanum wheelchair design allows mobility in all directions, including diagonal, lateral, and on-the-spot rotation [7]. The useful concept of the Mecanum wheelchair has the potential to be of great use, especially in busy and cluttered areas where traditional wheelchairs may have trouble navigating [8]. The precise omnidirectional movement and intelligent control of Mecanum wheelchairs make it easier for users to perform daily tasks and improve their quality of life. By addressing the problems caused by infrastructure that is still inaccessible to wheelchair users, particularly in poor nations, the Mecanum wheelchair technology has the potential to completely transform accessibility on a bigger scale. The possibility of omnidirectional movement through Mecanum wheels was examined by Thongpance and Chotikunnan [9]. With rollers positioned at a 45-degree angle to the wheel's axis of rotation, the innovative wheel design allows for movement in any direction without need for difficult turning maneuvers. The 360-degree rotation gives users improved control and navigation in small areas [10]. However, autonomous operation requires more than simply mechanical prowess.

Effective navigation also requires intelligent path-planning algorithms that can dynamically alter the wheelchair's trajectory to avoid obstacles and maximize progress [11]. Therefore, it is essential to integrate Mecanum wheel technology with sophisticated path-planning algorithms not only increasing the effectiveness and autonomy of assisted wheelchairs, but enabling them to handle challenging environments with the utmost safety and efficacy [12]. The enhanced RRT* algorithm is incorporated into a smart Mecanum-wheelchair system in the current study to enhance real-time navigational responsiveness and flexibility. The RRT* algorithm acts as an intelligent helper, enabling the wheelchair to autonomously avoid obstacles and ensure route optimization, even if it is largely controlled by a joystick. Even in confined interior locations, Mecanum wheels offer exceptional omnidirectional motion. However, their broad use in assistive mobility applications has been constrained by worries about their mechanical complexity and long-term durability on uneven terrain [13]. However, in predictable and organized environments like homes, hospitals, and rehabilitation centers, their ability to perform lateral and rotational motions with a tiny turning radius offers a clear practical advantage [14]. Mecanum wheels are ideal for wheelchair research because they work well in indoor environments with smooth, level ground. In order to evaluate design applicability, a static structural analysis is carried out to verify load-bearing capacity and traction performance under certain operating conditions. The system performance of this wheel design in combination with a closed-loop control system and an RRT*-based path planning algorithm was also assessed through simulations. Path planning is crucial for the independent operation of intelligent devices, particularly in terms of effective agility, such as smart wheelchairs.

Smooth wheelchair repositioning from the starting point to the destination is ensured by effective path design, which avoids collisions and reduces travel time. The traditional path planning algorithms, such as Dijkstra's and A*, have been widely used for navigation in organized domains. Shafiq *et al* [15] employed A* algorithm for path planning of the Mecanum wheel to prevent fixed hurdles in a room environment. Likewise, Dai

et al [16] also steered simulations through an optimized A* algorithm for effective path planning. Xu *et al* [17] reduced the path time by 13.53% through A* algorithm. Despite the successful applications of these algorithms, they still underperform in dynamic environments where obstacles may emerge or shift during operations [18]. Yadav *et al* [19] employed bee colony optimization (BCO) for path planning of Mecanum wheelchair, resulting to 5.9% reduction in length. Tsai *et al* [20] used simulated Particle Swarm Optimization (PSO),and achieve favorable results. Bhargave *et al* [21] employed both A* and PSO to achieve the Mecanum wheel AGV path optimization in path length by 3%. RRT* algorithm proved to be more advanced and a better choice for wheelchairs' path planning in nonlinear and multidimensional spaces than previous algorithms that limit the wheelchair's performance [22]. The probabilistic algorithm called RRT* gradually constructs a tree of potential routes from the starting point to the final destination. It investigates the surroundings by expanding the tree in the direction of randomly selected spots. RRT* algorithm continuously upgrades the path as it progresses by optimizing the trajectory to decrease the cost, such as travel distance or time. The wheelchair can follow a safe and efficient route rather than a sharp one with obstructions thanks to this real-time adjustment. Such dynamic replanning increases robustness and guarantees system flexibility. The smart Mecanum wheelchair's incorporation of the RRT* algorithm maximizes navigation accuracy and permits real-time responsiveness. It enables Mecanum wheelchairs to operate autonomously and effectively in challenging and changing conditions [23].

There are noticeable variations in performance, efficiency, and suitability for complicated situations when various RRT* variants are compared. According to Islam *et al* [24], RE-RRT* accomplished effective pathfinding at a modest cost with a path distance of 250 meters and a computing time of 3.0 seconds. In such situations, the suggested method provided a more practical and economical alternative. In complex indoor settings, Wang *et al* [25] further reported that the enhanced RRT*, which incorporates target-biased sampling and adaptive bias expansion based on a generalized Voronoi diagram and heuristic guidance, reduced the number of samples and search time by 71.34% and 79.60%, respectively, confirming its superior efficiency over traditional RRT*. Due to their probabilistic completeness and asymptotic optimality, the RRT* algorithm and its variations have been widely utilized for path planning; nonetheless, a number of drawbacks still exist. Notably, in order to guarantee smoothness and viability for real-time robotic applications, the RRT* algorithm frequently produces pathways with needless detours and abrupt twists that call for post-processing. The problem of producing intrinsically short and smooth paths during the planning stage is still not sufficiently addressed by current research, which concentrates on enhancing path optimality. To promote convergence toward the ideal path more effectively, the current study contributes a unique form of RRT* that constructs new nodes selectively within a favourable path radius.

Additionally, the suggested approach guarantees that the final path is both shorter and smoother by incorporating spline interpolation during the node connection phase. As a result, there is less need for external path-smoothing modules. Our method guides the tree expansion toward viable, smooth trajectories by using node sequencing algorithms. The gap in producing execution-ready pathways in dynamic contexts is immediately addressed by this combination of node selection, radius-based biasing, and real-time smoothing. Although earlier research has shown advancements in Mecanum wheel-based navigation with A*, PSO, and conventional RRT algorithms, these algorithms frequently fail in dynamic interior environments. Path abnormalities, a lack of real-time adaptability, and a dependence on external controllers for trajectory execution are the causes. Usually regarded as separate modules, path execution and path planning require additional post-processing or control methods to ensure viability. Nonetheless, a major obstacle in the planning stage is always ensuring a smooth and execution-ready road. However, one of the main drawbacks of the assistive mobility systems available today is the discrepancy between accurate path planning and its real-time viability.

The current study addresses the following gaps:

- RRT* paths often showcase sharp turns and detours, resulting in need for post-processing to achieve smoothness. The current study introduces a method that generates inherently smooth and short paths using advanced node selection and spline-based real-time smoothing.
- Many systems rely on PID, SMC, or CTM for tracking, which increases system complexity. This research eliminates external controllers by integrating encoder-based and RGB-D odometry within a closed-loop RRT*-based planner.
- Current approaches frequently don't respond swiftly enough to changes in the environment. The suggested approach improves adaptability by enabling real-time trajectory correction through merged visual and encoder-based localization.

Several studies incorporated external controllers such as Sliding Mode Control (SMC), PID, or Computed Torque Methods (CTM) to enhance path tracking and stability in mobile robot navigation. For example, Yadav *et al* [26] employed a kinematic model with SMC for path planning and demonstrated that the proposed controller outperformed CTM under conditions of uncertainty. Similarly, PID was utilized by Balambica *et al* [27] to carry out and stabilize motion commands. Nevertheless, using external controllers frequently makes the system more complex and necessitates careful parameter tweaking. The current work, on the other hand, does not require external controllers because it uses the RRT* algorithm for path planning in conjunction with encoder-based odometry and RGB-D visual odometry for localization. Using real-time sensor data, a closed-loop feedback mechanism was put in place to continuously adjust the robot's position. This setup proved adequate for a smart Mecanum wheelchair's precise and adaptive navigation, guaranteeing dependable performance without the need for extra control layers.

Therefore, for adaptive Mecanum-wheel mobility, a novel controller-independent navigation pipeline is suggested that combines an improved RRT planner with fused RGB-D and encoder-based closed-loop localization. In dynamic interior situations, the system improves path accuracy, obstacle reactivity, and real-time performance while lowering control complexity and producing naturally smooth, execution-ready trajectories. The current study is organized so that the introduction highlights the importance of the study and summarizes relevant literature, such as studies on path planning algorithms, wheelchair design, and Mecanum wheels. The steps and methods of the research are described in the methodology. Sections about the smart Mecanum wheelchair's design, control, and path planning come next. While the results and discussion section present the simulation, evaluates the results, and considers the implications of the findings, the experimental setup describes the prototype arrangement. Lastly, the conclusion section provides the key findings of the current study.

2. Methodology

To attain accurate and adaptive navigation, the Mecanum wheelchair development technique combines encoder-based odometry, simulation-based performance sensor integration, and mechanical design. With a focus on the positioning and arrangement of the four Mecanum wheels, which allow for omnidirectional mobility, the wheelchair is structurally constructed using SolidWorks to hold a maximum payload of 120 kg. Static analysis is conducted using ANSYS with an emphasis on wheel performance under anticipated loading circumstances to guarantee mechanical reliability. The RRT* algorithm, which works well in dynamic and partially known situations, was used in the path planning method for autonomous navigation. Beginning from the wheelchair's start position as the root node, the RRT* algorithm gradually samples the configuration space, joining additional nodes while considering kinematic limitations. Using RGB-D cameras for real-time collision detection, obstacle avoidance is controlled while simultaneously improving localization and updating the environment map. RRT* rewires the tree structure repeatedly for smoother and more effective navigation and assesses neighbouring nodes to minimize traversal cost to further optimize the planned path. The ESP32-based system at the heart of the wheelchair's control architecture interfaces with four DC motors through separate motor drivers. A programmable control box controls signal integration and power distribution, while a joystick is used to issue user commands.

A lead-acid battery that is controlled by a 3A adjustable DC-DC converter (LM2596s) provides power. Furthermore, tick-based feedback for real-time odometry is provided via quadrature encoders installed on each wheel, resulting in precise determination of the robot's pose (x, y, θ). Using Mecanum-specific inverse kinematics, the encoder data is used to determine linear and angular velocities. This information along with visual feedback improves motion estimation. Path accuracy, trajectory smoothness, length, and obstacle avoidance success rate are among the metrics used to evaluate the entire system. For both organized and unstructured situations, accurate, safe, and energy-efficient navigation is ensured by the integration of sensor-based feedback, encoder odometry, and adaptive path planning under a strong control system. Path accuracy, path length, and obstacle avoidance success rate during navigation are the evaluation metrics used to evaluate the system. The entire process of designing, analyzing, and simulating an optimal path using the RRT* algorithm is depicted in figure 1.

3. Mecanum wheelchair design

The smart Mecanum wheelchair is designed and simulated using SolidWorks. Figure 2 shows the design and the dimensions of the designed wheelchair. Figure 2(a) presents the front view, figure 2(b) portrays the side view, while figure 2(c) provides the bottom view. Additionally, figure 2(d) showcases the three-dimensional representation of the designed wheelchair.

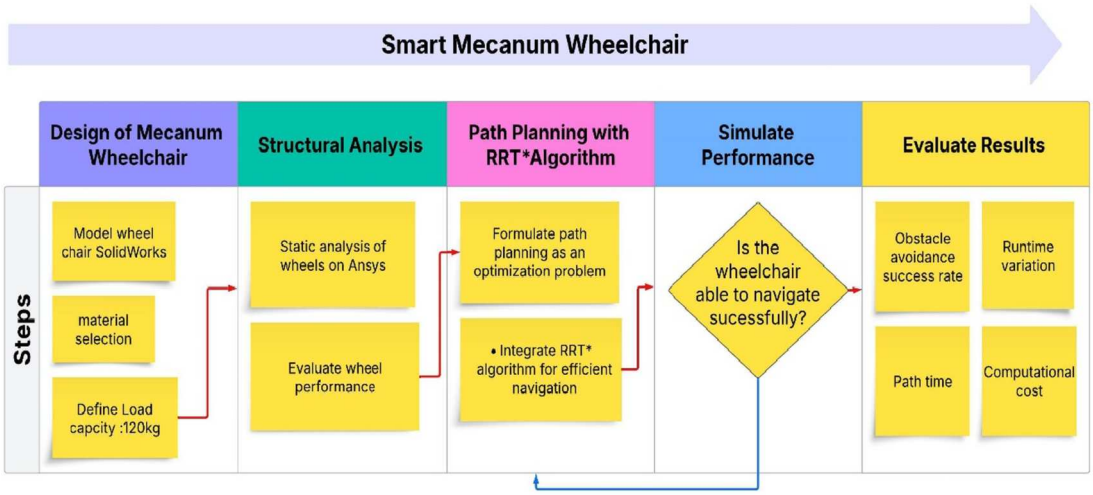

Figure 1. Process for design and path planning optimization of a smart Mecanum wheelchair.

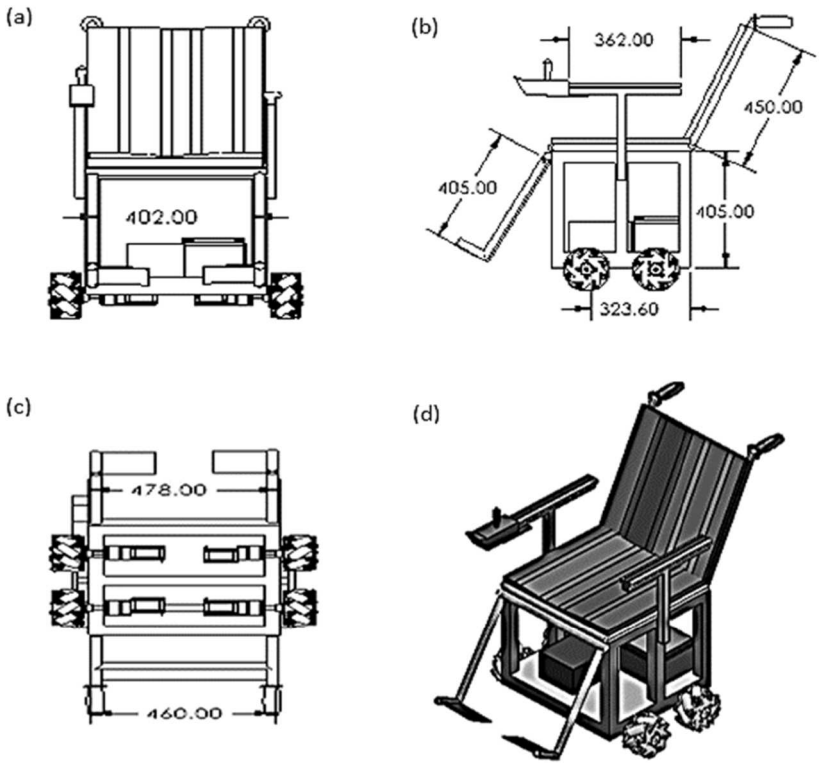

Figure 2. Design of the smart Mecanum wheelchair on SolidWorks: (a) front view, (b) side view, (c) bottom view, and (d) 3D view.

Figure 3 presents an exploded view of the smart Mecanum wheelchair, illustrating all labelled components integrated into the design. Each part is displayed and labelled to indicate its position and placement within the assembly.

The Mecanum wheelchair is designed for a maximum load of 120 kg. According to a study by Fan *et al*, the average mass of a wheelchair user is approximately 81.6 kg [28]. Most lightweight wheelchairs are designed to support a maximum weight capacity of 100 kg. For example, Gao *et al* [8] and Badejoko *et al* [29] developed smart wheelchairs with a 100 kg weight limit. Similarly, Zhewen *et al* [30] designed a Mecanum-wheel wheelchair with the same load capacity. Based on these previous studies and a market survey of lightweight wheelchair designs, the maximum load capacity for the smart wheelchair in our study has been set at 120 kg to ensure greater versatility and user safety. The static structural study of a Mecanum wheel using ANSYS 2022 R1 is shown in figure 4, which also shows the model's equivalent (von Mises) stress distribution. When the wheel is

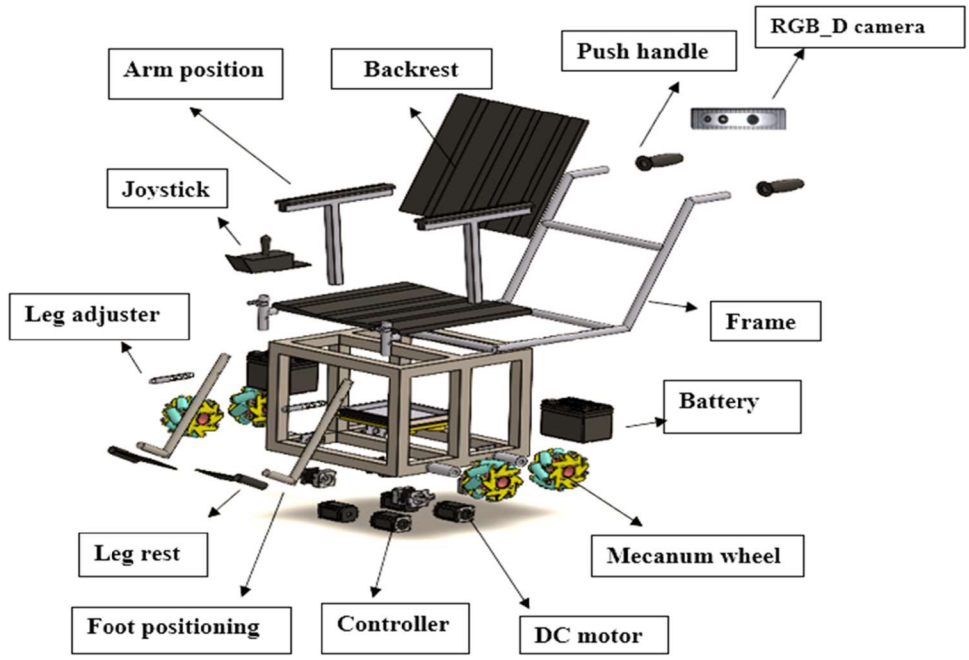

Figure 3. Exploded view of the designed smart Mecanum wheelchair.

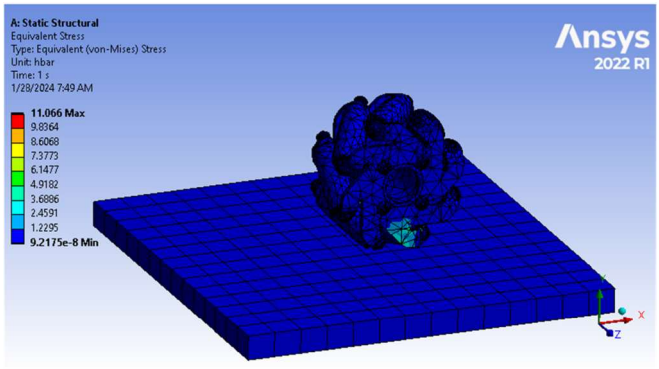

Figure 4. Static analysis of Mecanum wheel on Ansys software.

subjected to external loading on a rigid surface, the analysis assesses the stress response. By displaying greater stress areas at the wheel-surface contact locations, the mesh structure guarantees precise stress calculations. The analysis aids in determining the Mecanum wheel design's material strength, structural integrity, and possible failure sites.

Using ANSYS 2022 R1, a static structural analysis is carried out to evaluate the structural integrity of the planned Mecanum wheel under operational loading. To guarantee accurate interpretation of stress and deformation behaviour, the simulation setup includes realistic loading, boundary conditions, and precise meshing. With a consistent element size of 2 mm, a structured hexahedral mesh is created, featuring local refinements at high-stress areas such contact zones and roller-hub interfaces. The solution stayed stable within a 5% fluctuation for both von Mises stress and total deformation, according to mesh convergence analysis. With roughly 56,000 nodes and 43,000 elements in the final mesh, the resolution was adequate for precise stress capture while preserving computing efficiency. All translational and rotational degrees of freedom are completely constrained by the fixed support condition supplied to the support platform. Rubber interacting with smooth interior flooring is an example of frictional contact, which is described as the wheel and ground surface having a coefficient of friction of 0.6. To simulate one-fourth of the total cargo (120 kg) being evenly distributed among the four wheels, a vertical load of 294 N (corresponding to 30 kg) is imposed centrally on the hub surface in the negative Y-direction. Young's modulus of 2.1 GPa, Poisson's ratio of 0.35, and material density of 1040 kg m^{-13} are used to simulate the wheel material. The entire deformation study of a Mecanum wheel using ANSYS 2022 R1 is shown in figure 5. With deformation expressed in millimeters, the simulation assesses the wheel's

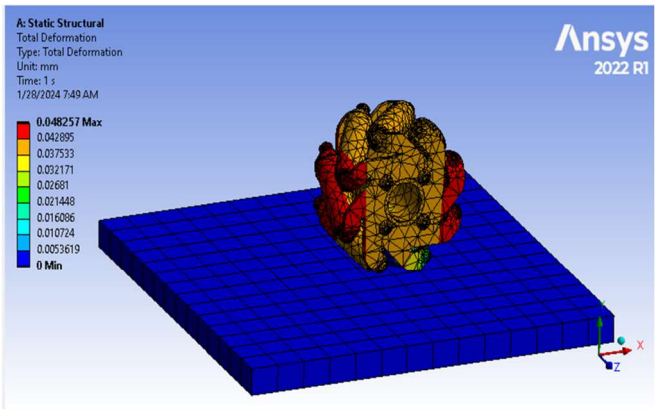

Figure 5. Deformation analysis of the Mecanum wheel on Ansys software.

structural response under imposed loading conditions. While the base platform stays firm, the outer rollers of the wheel exhibit the most deformation. The purpose of this investigation is to evaluate the Mecanum wheel's performance under operational pressures as well as its material flexibility and structural integrity.

The aluminium alloy 6061 frame of the clever wheelchair provides robustness without being overly heavy. With a yield strength of 275 MPa and an ultimate tensile strength of 310 MPa, the chosen material guarantees that the frame will retain its structural integrity in practical situations. The design was verified by ANSYS simulations. It verifies that the applied stress stayed securely within bounds at 112 MPa under a 120 kg load, with only 1.2 mm of deformation. This validates our design's stability and safety. Mecanum wheels' polyurethane-coated metal hubs enhance their functionality even more. Each wheel has a maximum stress of 11.07 MPa, a compressive strain of 0.03, and the capacity to sustain up to 30 kg. It permits smooth omnidirectional movement and has remarkable structural integrity. These results guarantee that the wheelchair can withstand common user demands such as long-term dependability, safety, and seamless maneuverability in dynamic paths. Polyurethane, a thermoplastic elastomer recognized for its exceptional fatigue resistance, abrasion resistance, and suppleness, is used to make the Mecanum wheel rollers. Tensile strengths of 20 to 50 MPa and elongation at break of 300% to 600% are examples of the repeated cyclic deformation that polyurethane can withstand without failing. Because of its intrinsic elasticity, it can effectively absorb dynamic loads, vibrations, and shocks. This ensures an increase in the fatigue life of the rollers, which can withstand millions of load cycles under typical working conditions. The rollers offer a useful compromise between flexibility and wear resistance in abrasive settings, with hardness values typically falling between Shore A 80 and Shore D 60. The aluminium alloy 6061 used to make the supporting frame has a modulus of elasticity of roughly 69 GPa and tensile strengths ranging from 290 MPa (yield) to 570 MPa (ultimate). With an endurance limit of roughly 96 MPa for 10^7 load cycles, the alloy exhibits good fatigue performance. These elements guarantee a design that is well-suited to withstand the torsional and dynamic bending forces experienced during manoeuvring. For the dependable operation of Mecanum wheel-based intelligent mobility systems, a reasonable solution in terms of durability, fatigue resistance, and load-bearing capacity is made possible by the combination of polyurethane rollers and aluminium alloy frames. Mecanum wheels allow for 360° mobility, making a Mecanum wheelchair incredibly adaptable. It can translate laterally without altering orientation, rotate 360° in position with a zero turning radius, and move forward and backward like a conventional wheelchair [6]. It also follows arc-based trajectories for smooth rotations and provides diagonal motion by modifying the speeds of individual wheels [31]. Its capacity to do intricate path-following, including a variety of curved manoeuvres, improves agility in dynamic or restricted spaces. The different ways that a smart Mecanum wheelchair can move are displayed in table 1.

The geometry of the Mecanum wheels is crucial for ensuring optimal movement and manoeuvrability. The wheel radius R is 51.5 mm, φ is the angular spacing between rollers, and the roller axis angle α is 45°. Using these parameters, we determine the relevant wheel dimensions, including the roller length Lr and wheel width Iw, given by:

$$L_R = 2.R.\sin\left(\frac{\varnothing}{2}\right)\Big/\sin\alpha$$

$$I_w = L_R.\cos\alpha$$

The system needs to support a wheelchair with a total load of 120 kg, requiring the motors to generate sufficient torque to overcome friction and any external forces encountered. The total normal force (N) acting

Table 1. Overview of movement modes for the smart mecanum wheelchair.

Movement by a Mecanum wheelchair	Description of movement
Forward/Backward	Moves straight ahead or backward like a traditional wheelchair.
In-Place Rotation	Rotates 360° around its center with a zero turning radius.
Lateral Motion	Moves directly left or right without changing orientation.
Diagonal Motion	Moves diagonally at an angle by varying wheel speed.
Arc-Based Turning	Can follow a curved trajectory instead of sharp turns.
Complex Path Following	Moves in custom patterns, such as smooth curves or S-shaped paths.

on each wheel is as follows:

$$N = W = m.g$$
$$F_r = \mu.N$$
$$\tau = F_r.r$$

τ represents the torque required per wheel to overcome friction, F_r is the frictional force, and μ is the coefficient of friction for the material (polyurethane).

After taking a safety factor into account, the necessary torque per wheel is determined to be 27.9 Nm. To provide smooth motion and effective operations, the chosen motors must retain sufficient torque. The chosen 24V DC gear motors have an ideal indoor speed of 4.8 to 6 km/h, a maximum speed of 350 RPM, and a torque capability of 30 to 35 Nm. The IBT-2 driver manages motor control and provides consistent operation with a maximum current of 20A continuous and 43A peak. Furthermore, the chosen motor's PWM control capabilities guarantee seamless direction and speed regulation. The design is geared for a maximum weight of 120 kg and smooth, level interior terrain. Due to motor torque and traction restrictions, performance may suffer under larger loads or on steep inclines.

4. Kinematic modelling and control system

This system made use of the Kinect v2 RGB-D camera, used for recording both colour (RGB) and depth data. It offers a field of vision of 70° horizontally and 60° vertically, a depth range of 0.5 to 4.5 meters, and an RGB resolution of 1080 p at 30 frames per second. Visual odometry and encoder-based odometry are used to produce localization. Visual odometry uses the RGB-D camera to track characteristics over successive frames to estimate the motion of the robot. However, collecting errors from poor feature matching, environmental changes, or occlusions can cause visual odometry to drift. Encoder data minimises this by continuously providing feedback on the wheelchair's motion, which increases localization accuracy and corrects drift. Therefore, the drift can be reduced by integrating both sensors into a closed-loop localization system. Hence, accuracy is increased as drift is reduced by combining encoder-based and visual odometry in a closed-loop system. The wheelchair can manoeuvre more consistently because of its sensor integration, even in busy and dynamic interior spaces. The wheelchair's actual motion is continuously tracked by the system, which then compares it to the intended trajectory produced by the RRT* algorithm. Real-time closed-loop result enables corrective control operations that minimize any error.

The RRT* algorithm computes an optimized trajectory

$$X_d(t) = [x_d(t) y_d(t) \theta_d(t)]^T$$

where $x_d(t)$ and $y_d(t)$ are the desired position coordinates and $\theta_d(t)$ is the desired orientation.

The actual pose of the wheelchair is obtained by fusing encoder data for wheel displacement and rotation, and RGB-D camera for visual odometry and obstacle detection. It is given as:

$$X(t) = [x(t),\ y(t),\ \theta(t)]^T$$

Wheel encoders measure tick counts Ni (for each wheel i = 1,2,3,4), which are converted to wheel angular velocities ω:

$$w_i = \frac{2\pi N_i}{N_{PPR}.\Delta t}$$

Where N_{PPR} is the pulses per revolution and Δt is the time interval.

The forward kinematics of a Mecanum-wheeled wheelchair determine the linear and angular velocities based on the four-wheel speeds. The wheelchair's velocity components, longitudinal (Vx), lateral (Vy), and angular (ω), are derived from the wheel velocities (v_1,v_2,v_3,v_4) using the following transformation matrix:

$$\begin{bmatrix} Vx \\ Vy \\ \omega \end{bmatrix} = R/4 \begin{bmatrix} 1 & 1 & 1 & 1 \\ -1 & 1 & 1 & -1 \\ -1/(L+W) & 1/(L+W) & -1/(L+W) & 1/(L+W) \end{bmatrix} \begin{bmatrix} v1 \\ v2 \\ v3 \\ v4 \end{bmatrix}$$

where,

- L is the distance from the wheelchair's center to the front and rear wheels (longitudinal distance).
- W is the distance from the center to the left and right wheels (lateral distance).
- R is the wheel radius

Hence, forward motion (v_x) is influenced by the sum of all wheel speeds, while lateral motion (v_y) depends on the combination of diagonal wheel speeds. The rotational velocity (ω) is governed by the difference in opposite wheel speeds, normalized by the wheelchair dimensions ($L+W$). Conversely, inverse kinematics solve for the required wheel speeds to achieve a desired wheelchair motion. The transformation matrix for inverse kinematics is given by:

$$\begin{bmatrix} v1 \\ v2 \\ v3 \\ v4 \end{bmatrix} = 1/R \begin{bmatrix} 1 & -1 & -(L+W) \\ 1 & 1 & (L+W) \\ 1 & 1 & -(L+W) \\ 1 & -1 & (L+W) \end{bmatrix} \begin{bmatrix} vx \\ vy \\ \omega \end{bmatrix}$$

These equations ensure precise computation of wheel velocities for controlled movement in any direction, considering the parameters L, W, and R.

To explicitly incorporate friction and force efficiency, the kinematic model was extended by including dynamic forces acting on the Mecanum wheels. The wheelchair dynamics are governed by

$$Mx^{\cdot\cdot} + C(x^{\cdot})x^{\cdot} + F_{fric} = \tau(\mathrm{t})$$

where F_{fric} accounts for rolling resistance, slip, and lateral friction at each wheel and $\tau(t)$ denotes the motor torque. Rolling resistance and slip friction forces are modeled based on standard friction coefficients obtained from polyurethane rollers and ground interaction. The unique roller orientation induces complex lateral friction components that reduce force efficiency, which is included by decomposing the wheel-ground contact forces into longitudinal and transverse components.

The tracking error is computed as:

$$e(t) = x_d(t) - x(t)$$

where,

$$e(y) = [e_x(t),\ e_y(t),\ e_\theta(t)]^T$$

It must be noted that the robotic wheelchair navigation utilizing Mecanum wheels, energy consumption is closely linked to path planning efficiency. The total energy expenditure E during wheelchair motion can be expressed as the integral of instantaneous mechanical power over the travel time T, given by

$$E = \int_0^T \tau(t).\omega(t)\,dt$$

where $\tau(t)$ denotes the motor torque and $\omega(t)$ the angular velocity at time t. Under typical operating conditions, assuming quasi-constant torque and velocity profiles, this relationship can be approximated as

$$E \approx \tau.\omega.\frac{L}{v}$$

where L is the total path length and v is the linear velocity. This formulation indicates a linear dependency of energy consumption on path length.

An ESP32 microprocessor powers the Mecanum wheel-based wheelchair, combining manual joystick input with autonomous path optimization using the RRT* algorithm. System initialization is the first step in the procedure, which involves turning on the ESP32, checking the battery and sensors, such as the RGB-D camera and encoders. The ESP32 microcontroller was chosen because of its low power consumption, real-time control capabilities, and integrated Wi-Fi/Bluetooth interfaces that enable firmware updates and remote diagnostics. When the system is ready, the user can send analog signals to the ESP32 by using the joystick as input. After that, the system decides if path optimization is necessary. When enabled, the ESP32 uses the RRT method to calculate an optimal trajectory. This method combines real-time data from the RGB-D camera with obstacle recognition and environment mapping to determine the location and orientation of the wheelchair using visual odometry. This camera-based localization guarantees dynamic obstacle avoidance and improves the

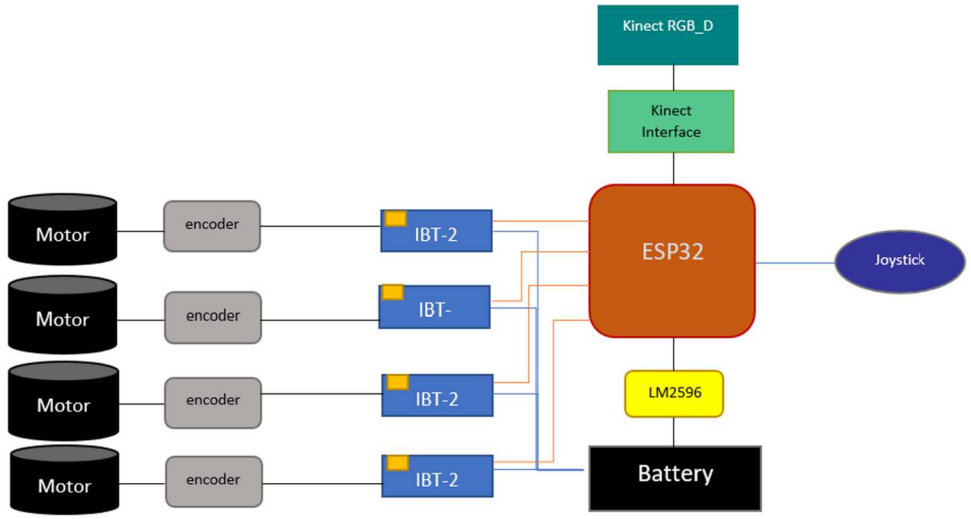

Figure 6. Control circuit of a smart Mecanum wheelchair.

configuration space's correctness when paired with encoder feedback. The joystick input is processed directly by the system if path optimization is not enabled. Following input processing, the ESP32 sends relevant PWM signals to the motor drivers after determining the necessary speed and direction for each of the four DC motors. By controlling the power supply to the motors, these drivers enable the Mecanum wheels to travel in all directions: forward, backward, lateral, and diagonal.

Visual odometry refreshes the environment map while the wheelchair is operating by continuously estimating its pose. The precise control of wheel speed is supported by the encoder data. Smooth navigation and real-time trajectory correction are made possible by this dual-feedback technique. If the path planning work is underway or the joystick is being used, the control loop is active. The flowchart also has error-handling procedures to deal with possible problems like low battery or sensor faults to guarantee safe functioning. Both manual and autonomous navigation features may be effectively controlled and robustly fault managed thanks to this organized architecture. The ESP32 microcontroller, which serves as the main processing and control unit, is at the center of the control circuit for the smart Mecanum wheelchair, as seen in figure 6. Mecanum wheels' omnidirectional mobility is made possible by IBT-2 motor driver modules that individually run four DC motors, each of which has an encoder. To guarantee consistent local power regulation, each IBT-2 module has an integrated LM2596 voltage regulator. The ESP32 receives real-time feedback from the motor encoders, enabling closed-loop control for precise speed and direction changes. The ESP32 is interfaced with a joystick to allow for manual user input. A Kinect RGB-D sensor is used for autonomous navigation and obstacle detection, and a Raspberry Pi is used as the dedicated Kinect Interface to interpret sensor data and interact with the ESP32. A central battery pack powers the entire system: the motor drivers directly draw from the battery's unregulated high-current output to match the motors' significant power demands, while the LM2596 provides a regulated low-voltage supply for the ESP32 and peripherals. Intelligent and responsive mobility assistance is made possible by this integrated design, which allows both manual and semi-autonomous operation. A Mecanum wheelchair's operational workflow is depicted in the flowchart in figure 7.

5. Path planning using RRT* algorithm

An innovative sampling-based motion planning technique called the RRT* algorithm is used to find the best routes in challenging situations. By adding a rewiring technique that improves path quality over time, it expands upon the fundamental RRT algorithm [32]. The RRT* is based on the following features: asymptotic optimality, which permits convergence to an optimal path as the number of samples increases; optimization by rewiring the tree to minimize cost metrics like path length or energy consumption; and environment exploration through random sampling of points [33]. Path planning in Mecanum-based wheelchair systems is especially well-suited to the capacity to explore dynamic and crowded spaces. As a result, it guarantees avoiding obstacles while taking advantage of the wheelchair's special motion characteristics [34]. By reducing cost functions like energy consumption and trip length, it further optimizes path planning. Additionally, it guarantees efficient and seamless trajectories for user comfort and flexibility in changing surroundings [35]. It guarantees the Mecanum wheels' omnidirectional mobility with the fewest needless rotations and intricate

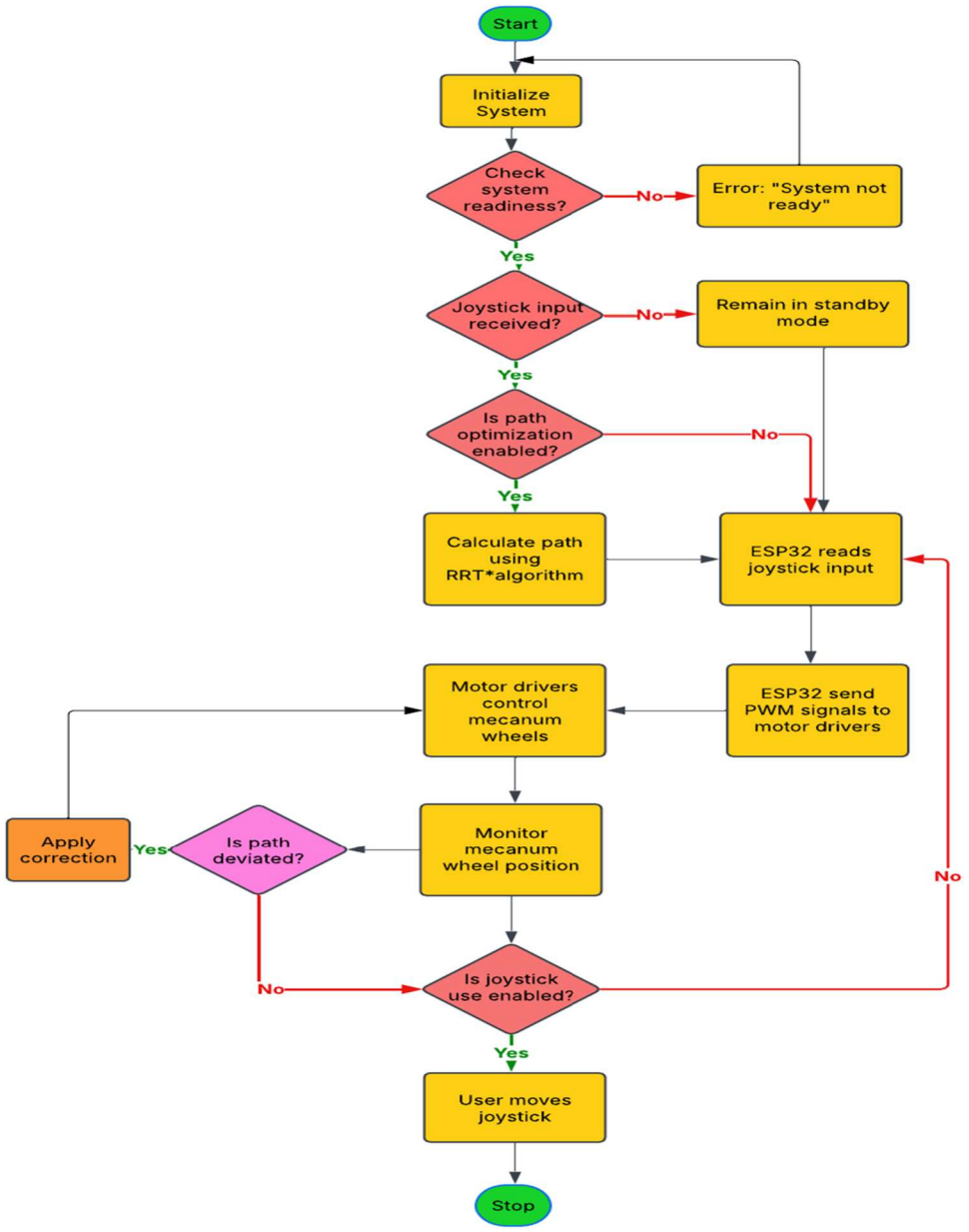

Figure 7. Control workflow of smart Mecanum wheelchair.

maneuvers, thus enhancing navigation efficiency [36]. The wheelchair may react to impediments by altering course thanks to the inclusion of RGB-D cameras (real-time sensor inputs), which guarantees constant updates of the planned path [37]. The RRT* algorithm is fine-tuned through several simulations to achieve smooth and computationally efficient path planning for a smart Mecanum wheelchair. Five thousand is the maximum number of iterations. This value made it possible to strike a compromise between real-time viability and sufficient configuration space exploration. A 0.2-meter step size is chosen. This made it possible to precisely maneuver in situations with tight turns or small passages by striking a balance between path smoothness and planning speed. The goal bias is set at 20%, which enables effective goal convergence without requiring a lot of processing. Because frequent rewiring allows for local path optimization, the rewiring radius is set at 1.2 meters. To ensure dependable obstacle avoidance, especially in congested situations, a collision-checking resolution of 0.015 meters is selected. B-spline smoothing is employed on the raw trajectory to refine the navigation path from sharp turns and improve smooth motion. The smooth B-spline trajectory is produced by computing the spline representation of discrete waypoints through splprep with a smoothing factor. Moreover, the spline is calculated at fine parameter intervals using 'splev' to ascertain smooth interpolated path points. Table 2 exhibits the fine-tuned parameters for RRT* path planning algorithm.

Table 2. Tuned RRT* Parameters for Smooth Path Planning.

Parameters	Values
Maximum Number of Iterations	5000
Step Size (m)	0.2
Sampling Rate	0.2
Rewiring Radius (m)	1.2
Collision check resolution (m)	0.015

6. Simulation environment

RRT* algorithm is developed in Python to develop a reliable motion planning system for the smart Mecanum wheelchair. The simulation environment for the Mecanum wheelchair utilized an RGB-D camera (e.g., Kinect) to acquire real-time data, including synchronized RGB images and depth frames. The depth data is utilized for constructing a 3D point cloud, which was then mapped onto a 2D occupancy grid. The grid consisted of occupied cells representing obstacles, free cells indicating traversable areas, and unknown cells for surveyed regions. Accuracy and computational effort were matched by the grid resolution of 0.05 m per cell. As the wheelchair moved, new depth frames were recorded, and this grid was updated dynamically.

By extracting and matching feature points from successive RGB frames, visual odometry is utilized for localization. The accompanying depth data is then used to infer 3D motion. The wheelchair's location and orientation on the map were calculated by this frame-to-frame motion tracking. A target posture, usually in the form of coordinates and orientation [x, y, θ], was used to specify the wheelchair's goal. The system then planned a route to get there. To ensure that the path avoided collisions with obstructions, the configuration space (C-space) model integrated the wheelchair's dimensions with kinematic limitations, such as maximum speed and turning radius. Within this configuration space, the RRT* planner constructed an optimal tree toward the goal pose by iteratively sampling feasible points and verifying them against the occupancy grid. After a suitable path has been identified, it is smoothed using spline interpolation to create a continuous trajectory that the Mecanum drive system can execute. The RGB-D sensor provided closed-loop feedback to the wheelchair as it followed this path, enabling both dynamic obstacle identification and localization correction. The algorithm updates the map and replans in real time if unforeseen challenges arise.

Safety margins were applied to maintain a buffer from surrounding obstacles. Finally, the wheelchair is continuously checked for its position concerning the goal. The task is marked complete when it reaches the target pose. The algorithm for RRT* for the optimized path is given as follows:

Initialize the tree with the starting node
Initialize the set of nodes in the tree
Add the starting node to the tree
While the number of nodes in the tree is less than the maximum allowed

- Sample a random point in the configuration space, named 'x_rand'
- Find the nearest node in the tree to 'x_rand', named 'x_nearest'
- Steer from 'x_nearest' towards 'x_rand' to generate a new point, named 'x_new'
- If the path from 'x_nearest' to 'x_new' is collision-free
- Find all nodes in the tree within a certain radius of 'x_new', named 'X_near'
- Choose the best parent for 'x_new' from 'X_near' to minimize the cost-to-come
- Add 'x_new' to the tree with the chosen parent
- Rewire the tree: For each node in 'X_near', check if connecting it through 'x_new' results in a lower cost-to-come
- If yes, update the parent of that node to 'x_new'

End If
End While loop

This study's enhanced RRT* algorithm has a number of benefits over the conventional RRT* method, especially when it comes to real-time indoor navigation with a Mecanum-wheel wheelchair. The uniform random

Table 3. Specifications of the smart mecanum wheelchair.

Specifications	Values
Load Capacity (kg)	120
Battery Consumption (W)	350
Maximum Speed (km/h)	6
Maximum Slope Climbing Angle (degrees)	12°
Chassis material	Aluminum

sampling used in traditional RRT* frequently results in ineffective exploration and non-smooth trajectories that require further post-processing. The enhanced RRT*, on the other hand, prioritises practical and high-value areas of the configuration space through guided and collision-aware sampling techniques. As a result, the pathways are naturally shorter and smoother. The improved version of RRT* examines all nodes within a specified neighbourhood radius and chooses the parent that minimizes the cost-to-come, enhancing path optimality, in contrast to classic RRT*, which chooses the closest node solely based on distance. Additionally, the revised algorithm's cost-driven rewiring process allows neighbouring nodes to be interconnected via newly inserted nodes. This results in a more effective tree structure by lowering the total path cost. Additionally, because collision-aware steering lowers invalid expansions and enhances planner performance in congested situations, collision management is likewise more efficient in the improved RRT*. The enhanced RRT* version incorporates real-time updates enabled by fused RGB-D and encoder-based localization, enabling adaptive trajectory correction, while the traditional RRT* suffers from fast environmental changes and may need total replanning.

7. Experimental setup

ANSYS static analysis is used to test the structural integrity of a prototype Mecanum wheelchair designed to handle a 120 kg load. To assess the wheelchair's navigational abilities, a controlled environment with predetermined obstacles was part of the experimental setting. There are four tested scenarios. The wheelchair has a camera installed to record data in real time for localization and obstacle recognition. The wheelchair has a programmable control system that is kept in a separate control box. Four BTS7960 H-Bridge DC motor drives, a piezo buzzer, and a 3A adjustable DC-DC step-down power module (LM2596s) were included in the system. The wheelchair is powered by a lead-acid battery and is controlled by motor drivers and an ESP32 microcontroller joystick. The smart Mecanum wheelchair's features are displayed in table 3.

The smart Mecanum wheelchair that was conceived and built is depicted in figure 8. The goal of the current study is to assess the RRT* method in a Python-based environment by generating and optimizing possible paths inside a specified workspace using numerical computing and visualization packages. A systematic experimental strategy is necessary to assess the precision of the path planning and movement capabilities of the smart Mecanum wheelchair. By comparing the wheelchair's actual path with the intended trajectory produced by the RRT* algorithm, path accuracy is evaluated. A motion tracking system and a camera-based system are used to do it. This improved the precision of the navigation evaluation path. The precision of omnidirectional motion is revealed by metrics like positional accuracy, which is the absolute difference between expected and actual positions, directional accuracy, which measures angular deviations, and lateral drift, which evaluates unintentional sideways movement. A combined path and movement accuracy test under dynamic settings was carried out for a thorough evaluation. In order to do this, obstacles had to be placed in the test area, and the wheelchair's ability to follow an ideal path while avoiding collisions had to be assessed. measures such replanning time and obstacle avoidance success rate, which is determined as the proportion of successful navigations without collisions. This is fundamental for assessing applicability in the real world. To determine whether the smart Mecanum wheel combined with the RRT* algorithm can successfully locate the shortest way and optimize the path, the path length is also calculated.

8. Results and discussion

The current section is based on path simulations and algorithm characterization based on performance metrics, as discussed below:

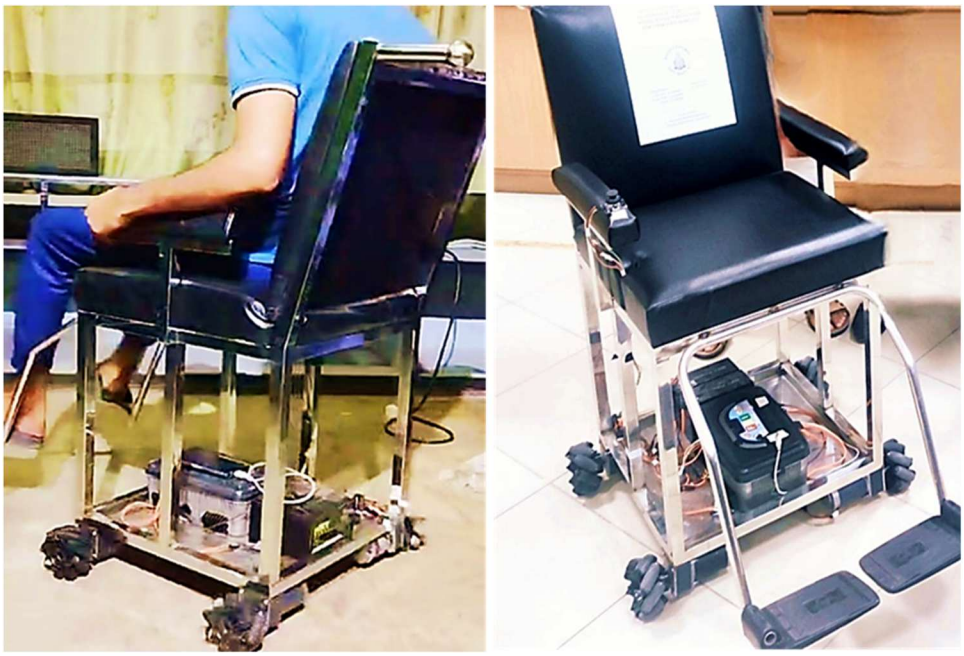

Figure 8. Fabricated smart Mecanum wheelchair.

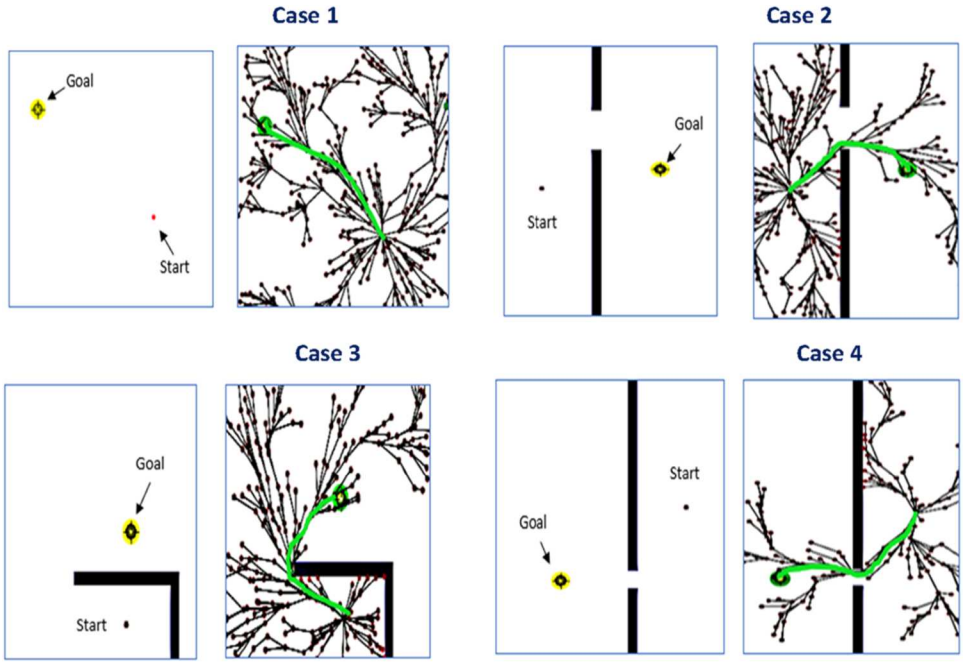

Figure 9. Path simulations for the RRT* algorithm: Case 1 characterizes a path with no obstruction, Case 2 shows a path with a vertical obstacle confined within a wide gap, Case 3 is a path with an L-shaped obstacle, and Case 4 is a path with a vertical obstacle but with a narrow gap.

8.1. Path simulations

The RRT* algorithm's path simulation in various settings is shown in figure 9. There are two subplots in each example. In every scenario, the problem setting is represented by the left subplot (part a), which shows the start location (represented by a dot) and the goal position (represented by a yellow marker) within the environment, which may contain obstacles. In every instance, the tree expansion process is shown in the right subplot (part b), which shows how the RRT* algorithm builds a tree and finds the best route to the objective, which is indicated in green. Since there were no obstacles in the way, it was possible to move freely between the start and goal positions in Case 1. The technique was able to efficiently build a straight path because the tree expansion

process was unrestricted. Because there were no impediments, this instance produced the shortest path length (35 seconds) and the fastest runtime of all. Two vertical impediments in Case 2 limited mobility to some extent. In comparison to Case 1, the tree expansion required more iterations to locate and navigate the opening, resulting in a moderate path length of 60 seconds. Additionally, the runtime is longer than in free space, suggesting that a little more processing power is needed to find the path and redefine it. In Case 3, path complexity is introduced by the L-shaped impediment. Because the method was designed to find different ways around the impediment, the tree included more nodes. Due to extensive manoeuvring, the path length is extended by exactly 95 seconds. Because path optimization and obstacle avoidance require a higher computing level, the extra repetition also results in the highest runtime. Like example 2, case 4 has stricter limitations, such as abrupt turns and the need for effective navigation to optimize the course. Because the algorithm must investigate intricate paths before arriving at the best solution, the tree expansion mechanism in case 4 is more complicated than in case 2.

It results in a path length of 85 s, which is shorter than case 3 but significantly longer than case 1 and case 2. The repeated rewiring and constrained movements are the main root causes for the elevated runtime and higher computational complexity. Generally, the RRT* path simulations portray significant correlation between path length, environmental complexity and runtime. The shortest path length of 35 s is observed in case 1 because of no obstacles. The longest path length of 95 s is observed in case 3 due to the L-shaped obstruction. The path lengths in case 2 and case 4 are observed to be 60 s and 85 s, respectively, which are between the two extreme cases (1 and 3). The extra turning constraints in case 4 result in a higher path length than in case 2. It can be concluded that the runtime and path length are higher in complex environments.

To quantitatively evaluate the accuracy of the trajectory generated by the RRT* algorithm, the Root Mean Square Error (RMSE) metric is employed. RMSE provides a statistical measure of the average deviation between the planned path and the ground truth or reference trajectory. After the RRT* algorithm generates a sequence of waypoints, each point is compared with its corresponding reference location as obtained from simulation or sensor-based ground truth. The Euclidean distance is used to calculate the positional error at each point, defined as:

$$d = \sqrt{(x_i^p - x_i^g)^2 - (y_i^p - y_i^g)^2}$$

where x_i^p andy_i^p are the coordinates of the planned path and are x_i^g *and*y_i^g the ground truth coordinates for the ith point. The RMSE is then computed by:

$$RMSE = \sqrt{\left(\frac{1}{N}\right)\sum_{i=1}^{N} d^2}$$

This value serves as an aggregate measure of path accuracy, with lower RMSE values indicating closer adherence of the planned path to the true trajectory. In this study, RMSE was calculated across multiple test scenarios to assess the localization precision and trajectory fidelity of the smart Mecanum wheelchair navigation system. Figure 10 shows the path accuracy results for all 4 cases.

The RMSE values for planned versus ground truth pathways in four RRT* simulation scenarios for a Mecanum wheelchair are shown in figure 11. High positional accuracy was demonstrated by cases 1 and 3, which had the lowest RMSE of roughly 0.015 meters. Case 4 had the largest inaccuracy at 0.030 meters, probably because of more complicated surroundings, while Case 2 had a somewhat higher RMSE of 0.022 meters. These findings highlight the significance of scenario-specific tweaking in RRT* implementations and show how path accuracy varies depending on obstacle arrangement. To evaluate the model's accuracy, figure 11 displays the RMSE for the planned versus the ground truth path for RRT* simulation.

8.2. Performance metrics

A box plot of the path length distribution for four RRT* simulation scenarios is shown in figure 12, with colours denoting various settings. The path length distribution is shown by the y-axis, which shows statistical characteristics like the median, interquartile range, and outliers, while the x-axis shows the cases. Case 1, shown by the blue colour, produces the shortest and most dependable path length and illustrates efficient navigation in an obstacle-free area. As shown in scenario 2, which is highlighted in red, the constraints imposed by a limited environment lead to moderate variability and a modest median path length. The yellow colour in example 3 represents an additional increase in path length and unpredictability due to the L-shaped obstacle. Due to extra path turns, case 4 exhibits the largest path length and unpredictability, as demonstrated by the purple colour. The growing trend in path length and variability across all four examples indicates that RRT* performance has been influenced by environmental complexity. The technique necessitates a thorough examination of spatial constraints and obstructions, which raises computing overhead and results in minor changes in path length estimation. The primary causes of deviations are the probabilistic nature of sampling and the need for iterative reconfiguration while avoiding obstacles. The box plot, which shows the distribution of path lengths over

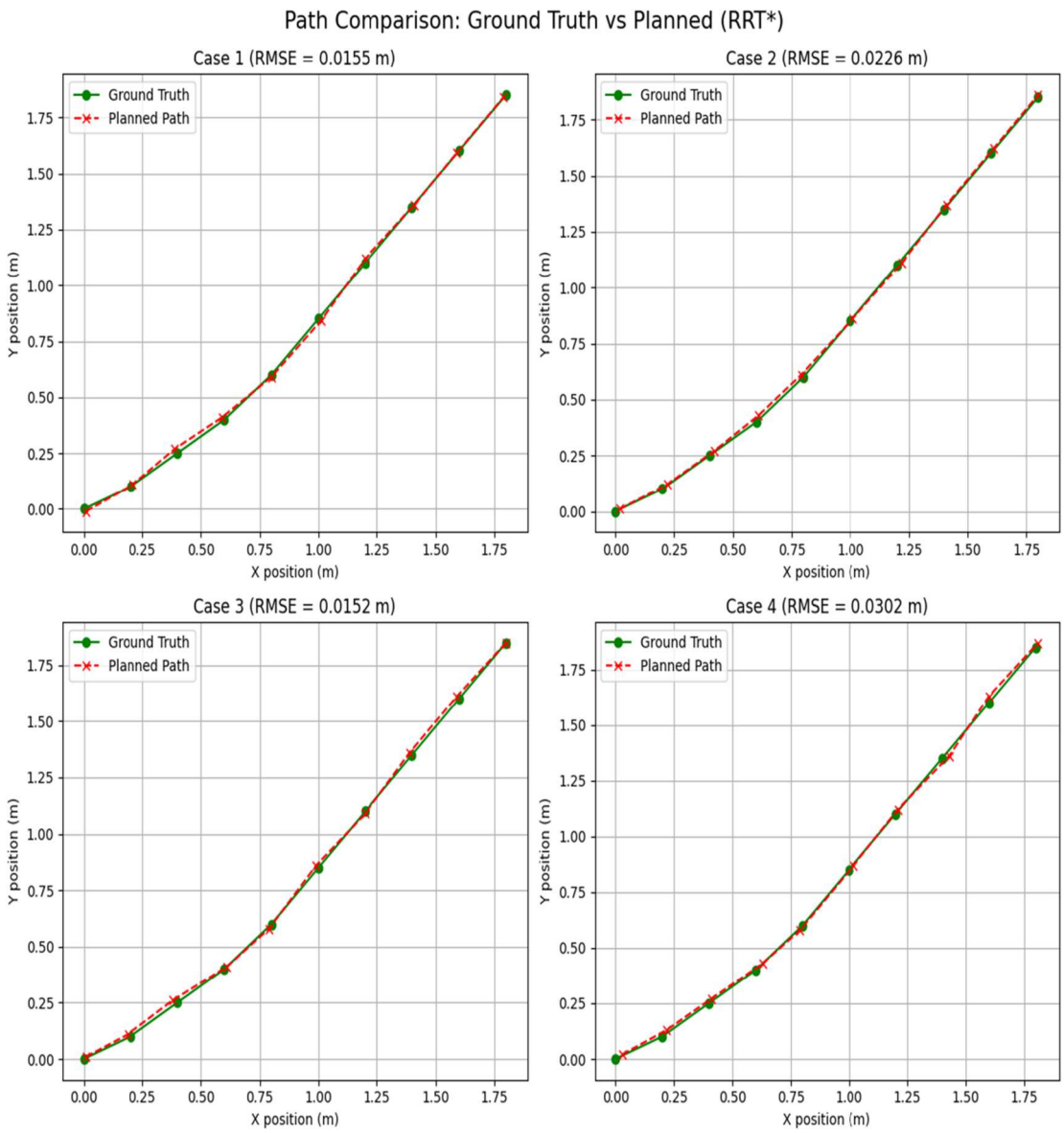

Figure 10. Path comparison of planned versus true path for the RRT star algorithm.

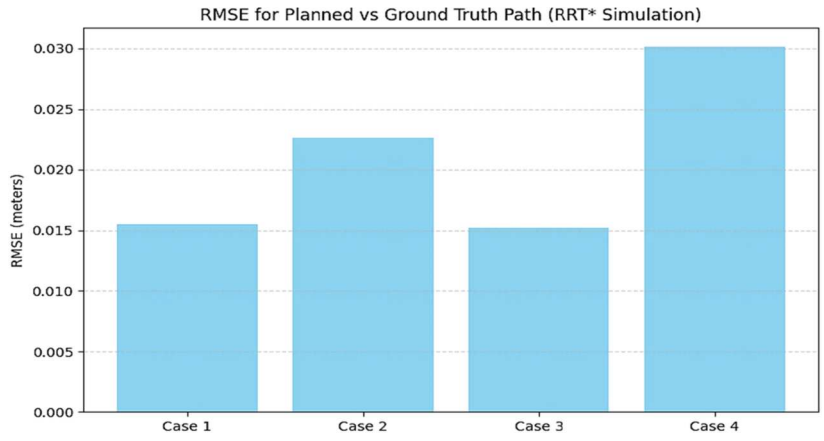

Figure 11. RMSE for Planned versus Ground Truth Path for RRT* simulation.

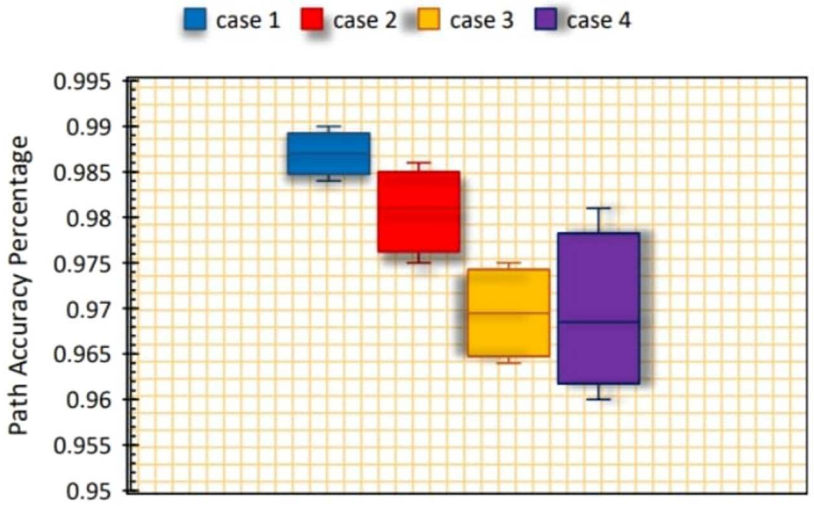

Figure 12. Box plot of path lengths for four RRT* simulation cases.

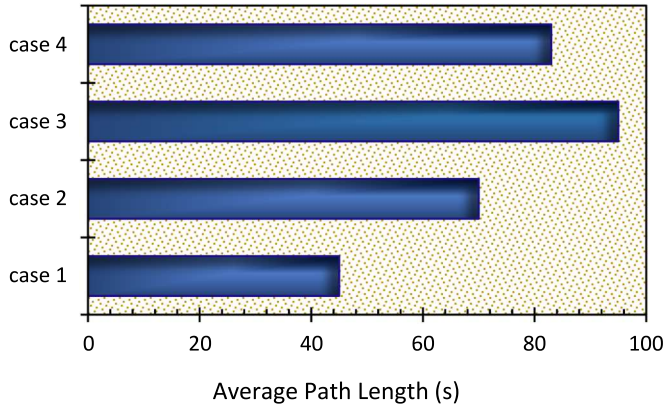

Figure 13. Average path length for four Cases of RRT* path planning.

several trials, reinforces these findings. The RRT* algorithm demonstrated strong path-planning performance, leading to dependable and efficient navigation in a variety of settings. To guarantee a strong statistically significant analysis, over twenty separate simulation runs are performed for every scenario. Outliers are identified and removed using the 1.5 × IQR rule, which eliminates data values that fall outside of 1.5 times the interquartile range from the first to third quartile. The current method validated that the deviations do not modify the overall interpretation of the data and greatly increased clarity. An Intel Core i7-10510U CPU running at 1.80 GHz, 8 GB of RAM, and a 64-bit Windows 10 environment are used to run the current simulations.

Figure 13 shows how traversal time is affected by navigation complexity by displaying the average path lengths (in seconds) for each of the four test situations. Shorter roads represent simpler environments, but longer paths suggest more complexity due to obstacles or manoeuvring requirements. In this inquiry, Case 1 recorded the shortest journey (35 s) and profited from free movement. In comparison, Case 3 took the longest path (95 s) because of its complex manoeuvring. Case 2 (60 s) and Case 4 (85 s) were in the center, with Case 4 being somewhat faster as a result of additional turning restrictions. The findings demonstrate a direct proportional relationship between runtime, journey length, and environmental complexity. The outcomes confirm that the RRT* algorithm can effectively estimate path lengths while taking environmental constraints into account. Turning limitations amplify the computational cost as the algorithm must train itself and assess several alternative paths before execution. In restricted places and surroundings with more obstacles, non-linear path changes and more collision avoidance maneuverability are necessary. As a result, RRT* has to carry out more iterations, which results in longer pathways and longer execution times.

The obstacle avoidance success rate for four RRT* path simulation scenarios is shown in figure 14. The success rate is calculated using 20 randomized trials for each scenario, for a total of 80 trials. When the RRT*

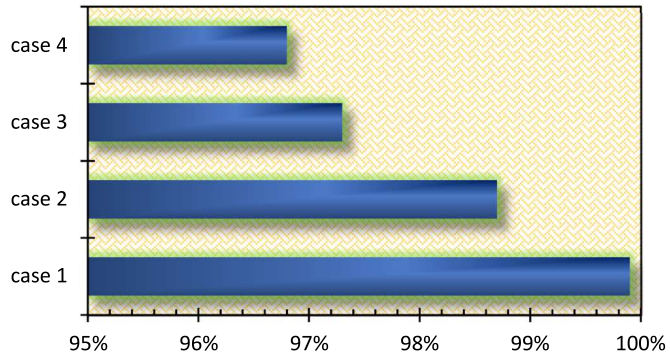

Figure 14. Obstacle avoidance success rate for RRT* path planning.

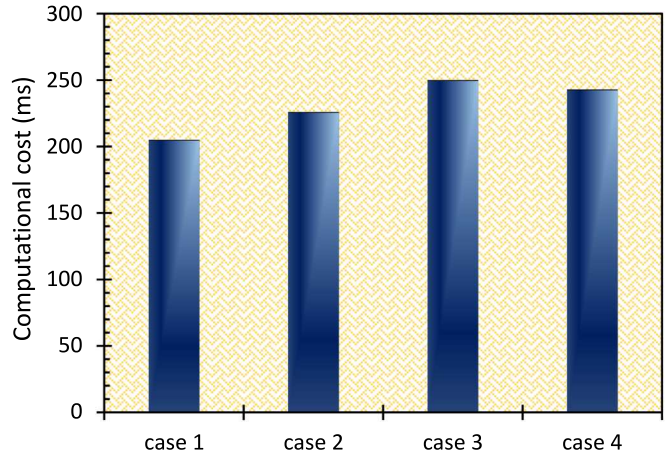

Figure 15. Computational cost for RRT* path simulation cases.

algorithm produced a full, collision-free path from start to goal inside the specified configuration space, the experiment was deemed successful. Because there were no barriers, Case 1 had the highest success rate—nearly 99.9%. It guaranteed a seamless navigation and made it possible for the algorithm to produce a nearly ideal direct trajectory with little investigation. Case 2 had a success rate of about 99%, which was marginally lower. A lengthy but effective navigation route resulted from the need to maneuver via a tiny waterway. The success percentage was further lowered to about 97% in Case 3, which involved an L-shaped obstacle. This indicated that establishing a suitable path became more challenging because more substantial directional alterations and node expansions were required to guarantee feasible traversal. Due to extra turning limits that affect the algorithm's navigation efficiency, Case 4 had the lowest success rate, 96.7%. The observed trend points to a clear relationship between obstacle avoidance success rates and environmental complexity. The findings show that although the RRT* method works well in high-dimensional exploration, the complexity of feasible path construction affects its effectiveness, especially in severely confined situations.

The computational cost (in milliseconds) for four RRT* path simulation scenarios is shown in figure 15. Because there are no barriers and the path can be produced immediately, Case 1 had the lowest computing cost in our study—roughly 200 ms. Because the algorithm had to maneuver through a tiny path, Case 2 took a little longer—roughly 220 ms. Due to the increased difficulty of navigating around an L-shaped obstruction, Case 3 had the highest cost—nearly 250 ms. Because of the turning limits that increased processing time, Case 4 of this investigation displayed a cost that was comparable to Case 3, at slightly less than 250 ms. Overall, the findings show a strong correlation between obstacle density and algorithm efficiency, with the computing cost rising with environmental complexity. The method needs more iterations, node expansions, and collision checks to provide a feasible path when obstacles and confined areas grow. Rewiring is more challenging because more nodes need to be changed in order to optimize the path. RRT* is still effective for exploring high-dimensional regions, but its efficiency is significantly affected by the extent of environmental constraints. Tighter

Table 4. Performance metrics of RRT* path simulation cases.

Performance metrics	Case 1	Case 2	Case 3	Case 4
Computational Cost (ms)	219	250	231	270
Path Time (s)	32	45	37	48
Run Time of wheelchair (min)	72	60	48	65

maneuverability and increased obstacle density eventually lead to lower real-time performance, highlighting the computational cost of path planning in complex scenarios.

The four RRT* simulation cases' wheelchair runtime, path time, computing cost, and critical performance parameters are reviewed in table 4. From 219 ms in Case 1 to 270 ms in Case 4, the computational cost consistently increased with environmental complexity. It demonstrates that tasks like node extension, rewiring, and collision checking in limited spaces require more processing for simulations. Due to an unrestricted path, Case 1 had the smallest path time of 32 s and the longest operating time of 72 min, whereas Case 4 had the longest path time of 48 s because of more turning limits and further manoeuvring. Because Case 3 repeatedly maneuvers around the L-shaped impediment, it has the shortest runtime of 48 min. Cases 2 and 4 resulted in intermediate runtimes of 60 and 65 min, respectively, demonstrating that turning limits and obstacle density have a direct impact on path execution efficiency and energy consumption. The findings thus demonstrate a clear connection between algorithm performance and environmental complexity.

In addition to increasing computing complexity, more complex settings lengthen traversal durations and lower wheelchair runtime overall. The computational expense of path planning in dense and structured settings is shown by the fact that, although RRT* was able to maintain collision-free navigation in every scenario, its efficiency decreased as limitations increased. Constrained pathways also frequently lead to more abrupt stops, more corrections, and eventually shorter runtimes. Our study's results demonstrate the trade-offs in sampling-based motion planning, where more environmental restrictions necessitate higher processing power and longer navigation times. Our findings highlight the necessity of optimizing RRT* using effective node selection, adaptive sampling techniques, and heuristic improvements. This is necessary to provide seamless and efficient navigation in challenging surroundings by striking a balance between computational efficiency and real-time viability. Based on average power consumption, run time is calculated analytically. Data specifications and empirical observations of the motors, motor drivers (IBT-2 with LM2596 regulators), sensors (Kinect RGB-D), and the ESP32 control unit were used to determine average current draw values. The nominal battery capacity was divided by the calculated average system current to determine the theoretical run time:

$$Estimated\ Runtime\,(hours) = \frac{Battery\ Capacity\,(Ah)}{Average\ Current\ Draw\,(A)}$$

The runtime estimation assumes moderate motor load, stable sensor and controller power consumption, constant voltage regulation efficiency from LM2596 modules, and excludes peripheral power usage.

The superior performance of RRT* in terms of path optimality and navigation efficiency, especially under complex and dynamic settings, is constantly highlighted by a thorough comparison of popular path planning algorithms, namely A*, D*, PRM, RRT, and RRT*, across numerous studies. Zeeshan *et al* [38] showed that of A*, PRM, and RRT*, the latter consistently generated the shortest and most optimal pathways, with a 21%–26% improvement in path planning time. RRT* outperformed A* and PRM in both path quality and success rates, despite having higher computational costs than regular RRT. A* showed the longest planning periods, even though it occasionally produced the quickest routes. Although PRM and RRT were relatively quicker, they lacked RRT's accuracy and dependability. A*'s strong convergence capabilities in static situations were further highlighted by Massoud *et al* [39] , who ranked it just below RRT* in reliability with an overall ranking of convergence of 81.67%, exceeding PRM (80%) and RRT* (71.67%) in dependability. While graph-based methods like A* and D* provide good accuracy in limited search spaces, Reda *et al* [40] pointed out that their scalability is restricted in large or dynamic situations.

Sampling-based techniques, such as RRT and RRT*, on the other hand, provide quicker execution and flexibility, often at the price of path smoothness. Tan *et al* [41] recognized RRT* for its capacity to overcome the suboptimality of normal RRT by incrementally refining pathways by local rewiring, whereas D* was recognized for its incremental re-planning capability, which is useful in dynamic contexts. Enhancements to RRT*, as shown by Wu *et al* [42] and Lei *et al* [43], further improved its convergence speed and path quality, solidifying its role as a robust solution for complex navigation tasks. Moreover, Fu *et al* [44] showed that RRT* is useful in real-world robotic operations by attaining 83% coverage and a path length of 53 units in inspection planning scenarios.

The RRT* algorithm produced collision-free pathways in simple, unobstructed settings with a 100% success rate in the current experimental environment. The success percentage slightly decreased to 96.7% in more complicated situations with higher barrier densities. When it came to path feasibility and collision avoidance, RRT* consistently performed better than other algorithms. Rarely, the algorithm's nodes might extend in non-optimal ways, necessitating several iterations before a workable path could be found. These findings are consistent with earlier research. The effectiveness of the A* algorithm in a simulated environment with different obstacle densities was examined by Liu *et al* [45]. According to their findings, the success rate dropped from 100% to 93% as the number of challenges rose. This demonstrated how sensitive the algorithm was to the intricacy of the surrounding environment. Wang *et al* [46] compared RRT, RRT*, and an improved variant and reported respective success rates of 73.48%, 84.98%, and 96.62%. This highlighted the advantages of RRT* and its enhanced implementations. Li *et al* [47] applied a modified Probabilistic Roadmap (PRM) approach to smart vehicle navigation. They achieved success rates between 90.9% and 100% based on scenario complexity. Zhao *et al* [48] employed a multi-agent reinforcement learning strategy for UAV path planning in simulation. A success rate of 80% was achieved, further validating the potential of learning-based methods in complex environments.

A wheelchair navigation system that uses the RRT* algorithm in crowded indoor situations was the subject of a targeted benchmark investigation. With success rates ranging from 96.7% to 100%, the algorithm maintained excellent accuracy in obstacle avoidance, but as environmental complexity rose, its computational efficiency decreased. As barrier density grew, path execution time climbed from 32 to 48 seconds, and computation time increased from 219 to 270 ms. Furthermore, differences in wheelchair runtime across various path complexities imply that RRT*'s processing requirements and energy usage are directly related. These findings support the findings of earlier studies. It explains that although RRT*'s durability and adaptability make it a viable contender for optimal path planning, its computational complexity presents a problem in real-time applications. Existing wheelchair solutions exhibit a trade-off between cost and functionality, according to the feasibility study. Despite being reasonably priced at about $100 on average [49], manual wheelchairs lack automation and sophisticated support functions.

This reduces upper body strength or movement, necessitates powered support, and restricts their appropriateness for older people or those with severe disabilities. Moula *et al* developed an inexpensive, multipurpose wheelchair at $168 price, but it lacks of automated features of a smart wheelchair [50]. Safy *et al* [51] developed a low-cost smart wheelchair for $317; however, it employed only two motors, offering limited mobility and flexibility. In contrast, high-mobility smart wheelchairs such as those designed by Jacob *et al* [52] cost over $1000, and the Mecanum-wheel-based smart wheelchair by Thongpance *et al* [53] was priced at $1200. In this regard, the suggested design of a smart wheelchair with Mecanum wheels that costs only $532 offers a notable improvement. Using an aluminium frame and an economical structural design significantly lowers manufacturing costs without sacrificing quality, mobility, or intelligent features. Although the current work uses a comprehensive simulation to show that the suggested approach is feasible and effective, there are still some drawbacks. The next stage is to conduct real-world dynamic testing to evaluate the system in real-world scenarios with unpredictable obstacle behaviour and changing external conditions. Standard assumptions like idealized sensor feedback and uniform terrain are also included in the simulation setting, which may not match actual deployment conditions. Finally, even though the system demonstrated encouraging computational performance, a thorough examination of control loop latency is outside the current purview and will be addressed in subsequent real-time implementations.

9. Conclusions

- The current work describes the creation of a sophisticated, low-cost intelligent wheelchair based on Mecanum wheels, envisioned to enhance autonomous navigation and path planning in dynamic environments. Without depending on external controllers, the RRT* algorithm combined with onboard control and sensors, display improved movement accuracy, obstacle avoidance, and adaptability.
- The performance of navigation stayed high. Success rates for avoiding obstacles varied from 96.7% to 100%, with a minor decrease noted as environmental complexity increased. The computational cost increased from 219 ms to 270 ms as a result. More extensive node enlargement, path rewiring, and collision checking are the key factors. The wheelchair's operational lifespan varied between 72 and 48 min for different difficulties, while the path completion time ranged from 32 to 48 seconds. These results confirm the viability and efficiency of the suggested navigation system under practical circumstances.
- High path-tracking accuracy is confirmed by the low RMSE values (0.015–0.03), although higher obstacle density increases computation and traversal time. These findings emphasize the need of adaptive, energy-efficient path-planning techniques for guaranteeing responsiveness and dependability in challenging real-world settings.

- Future work includes testing the system on the smart Mecanum wheelchair in real indoor settings, validating navigation under dynamic obstacles, refining calibration and localization, adding adaptive replanning, and exploring deep learning and reinforcement learning to enhance autonomy and enable real-time execution on embedded controllers.

Conflicts of interest

The authors declare no conflict of interest.

Data availability statement

Data can be made available upon request. The data that support the findings of this study are available upon reasonable request from the authors. Data will be available from 2 June 2026.

Author contributions

Sadaf Zeeshan 0000-0003-2776-9031
Data curation (equal), Formal analysis (equal), Investigation (equal), Writing – original draft (equal)

Muhammad Ali Ijaz Malik 0000-0002-0227-4289
Investigation (equal), Methodology (equal), Validation (equal), Writing – original draft (equal), Writing – review & editing (equal)

Shahzaib Aslam
Investigation (equal), Methodology (equal), Validation (equal), Writing – original draft (equal), Writing – review & editing (equal)

References

[1] Karin M, Christin E, Claudia L, Rainer S and Gerhard W 2022 Traveling more independently: a study on the diverse needs and challenges of people with visual or mobility impairments in unfamiliar indoor environments *ACM Transactions on Accessible Computing* **15** 1–44
[2] Sushil Kumar S and Bibhuti Bhusan C 2024 Autonomous navigation and obstacle avoidance in smart robotic wheelchairs *Journal of Decision Analytics and Intelligent Computing* **4** 47–66
[3] Mohd Azri A M and Norsinnira Z A 2020 Prototype development of mecanum wheels mobile robot: a review *Applied Research and Smart Technology* **1** 71–82
[4] Sara C, Alessandro D F, Carlos V, Luca M and Francesco F 2020 ARI: the social assistive robot and companion presented at the *29th IEEE Int. Conf. on Robot and Human Interactive Communication (Naples, Italy)*
[5] Ezgi Ö, Zehra Güçhan T and Hüseyin A 2021 Determinants of travel participation and experiences of wheelchair users traveling to the bodrum region: a qualitative study *Int. J. Environ. Res. Public Health* **18** 2218
[6] Zhang Z, Yu H, Wu C, Huang P and Wu J 2024 A comprehensive study on Mecanum wheel-based mobility and suspension solutions for intelligent nursing wheelchairs *Sci. Rep.* **14** 20644
[7] Luigi T and Giuseppe Q 2023 On the design of MoviWE.Q: an omnidirectional electric-powered wheelchair for indoor mobility presented at the *Advances in Mechanism and Machine Science* (IFToMM WC) 2023
[8] Han G, Yuanze T, Zhiwen W and Qing W 2021 Design of a multifuntional nursing wheelchair *J. Phys. Conf. Ser.* **1885** 052044
[9] Thongpance N and Chotikunnan P 2023 Design and construction of electric wheelchair with mecanum wheel *Journal of Robotics and Control* **4** 71–82
[10] Tiberiu G, Ghita B, Ioan V and Cristina P 2022 Mecanum wheeled platforms for special applications presented at the *Int. Conf. Knowledge-Based Organization*
[11] Chunfang L, Bin L and Yan S 2025 An integrated solution to improve the environment awareness and path planning efficiency of smart wheelchairs *Intelligent Robotics* **2355** 286–300 presented at the
[12] Mostafa Mo. M, Massoud A., Abdellatif and Mostafa R A A 2024 Selecting dynamic path planning algorithm based-upon ranking approach for omni-wheeled mobile robot *Journal of Advanced Research in Applied Sciences and Engineering Technology* **41** 125–38
[13] Tiberiu G, Ghiţă B, Ioan V and Cristina P 2022 Mecanum wheeled platforms for special applications *Int. Conf. Knowledge-based Organization* 28 (Sciendo)
[14] Antonio P D, Ming-Hsun I and Chih-Hung G L 2022 Optimization of mecanum wheels for mitigation of AGV vibration *Int. J. Adv. Manuf. Technol.* **121** 633–45
[15] Muhammad Umair S, Abid I, Sajjad M, Afraz Hussain M, Bilal A, Ilyas K and Abdullah M 2024 Real-time navigation of mecanum wheel-based mobile robot in a dynamic environment *Heliyon* **10** e26829
[16] Jianhong D, Huazhang W, Yuchao H, Yixuan W and Tao Z 2023 Intelligent multifunctional wheelchair controlled by MTPA and improved a-star algorithm *Third Int. Conf. on Mechanical, Electronics, and Electrical and Automation Control (Hangzhou, China)* (https://doi.org/10.1117/12.2679942)
[17] Huimin X, Gaohong Y, Yimiao W, Xiong Z, Yijin C and Jiangang L 2023 Path Planning of mecanum wheel chassis based on improved A* algorithm *Electronics* **2** 1754

[18] Jinyang W, Yuhua L, Ruixuan L, Hao C and Kejing C 2022 Trajectory planning for UAV navigation in dynamic environments with matrix alignment Dijkstra *Soft Computing* **26** 12599–610
[19] Pankaj Singh Y, Mohanta J C, Vandana A and Faiyaz A M 2024 Modeling and performance evaluation of mecanum wheel platform with reduced vibration and a robust path planning *Journal of the Brazilian Society of Mechanical Sciences and Engineering* **46** 292
[20] Ching-Chih T, Ching-Zu K, Chun-Chieh C and Xiao-Ci W 2013 Global path planning and navigation of an omnidirectional Mecanum mobile robot presented at the *CACS Int. Automatic Control Conf. (Nantou, Taiwan)*
[21] Ankur B, Mohammad S and Ajay K S S 2024 An omnidirectional mecanum wheel automated guided vehicle control using hybrid modified A* algorithm *Robotica* **43** 1–50
[22] Mohammad R, Mohammed E A, Javaid B, Holger V and Alireza R 2024 Mobile robotics and 3D printing: addressing challenges in path planning and scalability *Virtual and Physical Prototyping* **19** e2433588
[23] Youssef M and Youssef L 2024 Autonomous navigation: integrating stability concepts with RRT* in complex environments *Int. Conf. on Computing, Internet of Things and Microwave Systems (Gatineau, QC, Canada)* (IEEE) (https://doi.org/10.1109/ICCIMS61672.2024.10690752)
[24] Naeem U I, Kaynat G, Faiz F, Syed S U and Ikram S 2024 Trajectory optimization and obstacle avoidance of autonomous robot using robust and efficient rapidly exploring random tree *PLoS One* **19** e0311179
[25] Jiqiang W and Enhui Z 2024 Path planning of a mobile robot based on the improved rapidly exploring random trees star algorithm *Electronics* **13** 2340
[26] Pankaj Singh Y, Vandana A, Mohanta J C and Faiyaz A M D 2022 A robust sliding mode control of mecanum wheel-chair for trajectory tracking *Mater. Today Proc.* **56** 623–30
[27] Balambica V, Anto A, Achudhan M, Deepak V, Juzer M and Tamil S 2021 PID sensor controlled automatic wheelchair for physically disabled people *Turkish Journal of Computer and Mathematics Education* **12** 5848–57
[28] Fan M 2023 A low cost power assist for active wheelchair users *PhD Diss.* Massachusetts Institute of Technology
[29] Badejoko-Okunade A, Adefemi A, Oluwole A and Samuel A 2020 Development of a wheelchair automator *FUOYE Journal of Engineering and Technology* **5** 109–13
[30] Zhewen Z, Yu H, Wu C, Huang P and Wu J 2024 A comprehensive study on Mecanum wheel-based mobility and suspension solutions for intelligent nursing wheelchairs *Sci. Rep.* **14** 20644
[31] Selven R, Raghavan S, Gokul N, Kiran Raaj K V and Damoodaran S 2025 Finite element analysis and functional performance of mecanum wheel, *Recent Innovations in Sciences and Humanities* (CRC Press)
[32] Haojian Z, Yunkuan W, Jun Z and Junzhi Y 2018 Path planning of industrial robot based on improved RRT algorithm in complex environments *IEEE Access* **6** 53296–306
[33] Jin-Gu K, Dong-Woo L, Yong-Sik C, Woo-Jin J and Jin-Woo J 2021 Improved RRT-connect algorithm based on triangular inequality for robot path planning *Sensors* **21** 333
[34] Hussein M, Lotfi R and Mohammad A J 2021 RRT*N: an efficient approach to path planning in 3D for static and dynamic environments *Adv. Robot.* **35** 168–80
[35] Jun S, Hao X, Jianfeng L, Wei S, Zheng C and Jason G 2021 RRT-GoalBias and path smoothing based motion planning of mobile manipulators with obstacle avoidance presented at the *IEEE Int. Conf. on Real-time Computing and Robotics (RCAR) (Xining, China)*
[36] Wei L, Xiang J, Tianlin W and Han Z 2021 Improved RRT algorithms to solve path planning of multi-glider in time-varying ocean currents *IEEE Access* **9** 158098–115
[37] Zifu L, Rizky Mulya S, Rasika D A R M, Victor Massaki N and Tofael A 2024 Development of a collision-free path planning method for a 6-DoF orchard harvesting manipulator using RGB-D camera and Bi-RRT algorithm *Sensors* **24** 8113
[38] Sadaf Z and Aized T 2023 Performance analysis of path planning algorithms for fruit harvesting robot *Journal of Biosystems Engineering* **48** 178–97
[39] Mostafa Mo M, Abdellatif A and Atia M R A 2022 Different path planning techniques for an indoor omni-wheeled mobile robot: experimental implementation, comparison and optimization *Applied Sciences* **12** 12951
[40] Reda M, Ahmed O, Amira Y H and Ali G 2024 Path planning algorithms in the autonomous driving system: a comprehensive review *Rob. Autom. Syst.* **174** 104630
[41] Tan C S, Rosmiwati M-M and Rizal A M 2021 A comprehensive review of coverage path planning in robotics using classical and heuristic algorithms, *IEEE Access* **9** 119310–42
[42] Wu D, Lisheng W, Guanling W, Li T and Guangzhen D 2022 APF-IRRT*: an improved informed rapidly-exploring random trees-star algorithm by introducing artificial potential field method for mobile robot path planning, *Applied Sciences* **12** 10905
[43] Lei S, Li T, Gao X, Xue P and Song G 2025 Research on improved RRT path planning algorithm based on multi-strategy fusion *Sci. Rep.* **15** 13312
[44] Fu M, Alan K, Oren S and Ron A 2019 Toward asymptotically-optimal inspection planning via efficient near-optimal graph search *Robotics Science and Systems* **1** 1–25
[45] Liu Z, Liu H, Lu Z and Zeng Q 2021 A dynamic fusion pathfinding algorithm using delaunay triangulation and improved a-star for mobile robots *IEEE Access* **9** 20602–21
[46] Haotian W, Xiaolong Z, Jianyong L, Zhilun Y and Linlin C 2024 Improved RRT* algorithm for disinfecting robot path planning *Sensors* **24** 1520
[47] Qiongqiong L, Yiqi X, Shengqiang B and Jiafu Y 2022 Smart vehicle path planning based on modified PRM algorithm *Sensors* **22** 6581
[48] Xiaoru Z, Rennong Y, Liangsheng Z and Zhiwei H 2024 Multi-UAV path planning and following based on multi-agent reinforcement learning *Drones* **8** 18
[49] Ernesto S-T, Md Abdul B S, Micheal V C, Owen C, Christopher D L and Masudul H I 2021 Design of a low-cost, lightweight smart wheelchair *IEEE Microelectronics Design & Test Symp. (MDTS) (Albany, NY, USA)* (IEEE) (https://doi.org/10.1109/MDTS52103.2021.9476093)
[50] Saraf Fariha M, Sumaiya Tabassum S, Nelima Azad C and Md Doulotuzzaman X 2023 Design and development of a multifunctional convertible wheelchair in a low-income country context *Indonesian Journal of Computing, Engineering, and Design* **5** 8–17
[51] Mohamed S, Mayada M, Nardeen E T, Ahmed I N, ELa G A E, Bayoumy A and Ashraf M 2022 Low-cost smart wheelchair to support paraplegic patients *International Journal of Industry and Sustainable Development* **3** 1–9
[52] Jacob M and Stephen S 2022 Propulsion cost changes of ultra-lightweight manual wheelchairs after one year of simulated use *ASME Open J. Engineering* **1** 011047
[53] Nuntachai T and Phichitphon C 2023 Design and construction of electric wheelchair with mecanum wheel *Journal of Robotics and Control* **4** 71–82