Preprints.org

Article

Geometrical Optimal Navigation and Path Planning – Bridging Theory, Algorithms, and Applications

Hedieh Jafarpourdavatgar , Samaneh Alsadat Saeedinia , Mahsa Mohaghegh [*]

Article

Geometrical Optimal Navigation and Path Planning—Bridging Theory, Algorithms, and Applications

Hedieh Jafarpourdavatgar [1], Samaneh Alsadat Saeedinia [2] and Mahsa Mohaghegh [3,*]

[1] Control Engineering Department, Amirkabir University, Tehran, Iran
[2] Control Electrical Engineering Department,Iran University of Science of Technology, Tehran, Iran
[3] Faculty of Design and Creative Technologies, Auckland University of Technology, Auckland, New Zealand
* Correspondence: mahsa.mohaghegh@aut.ac.nz

Abstract

Autonomous systems, such as self-driving cars, surgical robots, and space rovers, require efficient and collision-free navigation in dynamic environments. Geometric optimal navigation and path planning have become critical research areas, combining geometry, optimization, and machine learning to address these challenges. This paper systematically reviews state-of-the-art methodologies in geometric navigation and path planning, focusing on the integration of advanced geometric principles, optimization techniques, and machine learning algorithms. It examines recent advancements in continuous optimization, real-time adaptability, and learning-based strategies, which enable robots to navigate dynamic environments, avoid moving obstacles, and optimize trajectories under complex constraints. The study identifies several unresolved challenges in the field, including scalability in high-dimensional spaces, real-time computation for dynamic environments, and the integration of perception systems for accurate environment modeling. Additionally, ethical and safety concerns in human-robot interactions are highlighted as critical issues for real-world deployment. The paper provides a comprehensive framework for addressing these challenges, bridging the gap between classical algorithms and modern techniques. By emphasizing recent advancements and unresolved challenges, this work contributes to the broader understanding of geometric optimal navigation and path planning. The insights presented here aim to inspire future research and foster the development of more robust, efficient, and intelligent navigation systems. This survey not only highlights the novelty of integrating geometry, optimization, and machine learning but also provides a roadmap for addressing critical issues in the field, paving the way for the next generation of autonomous systems.

Keywords: geometric navigation; path planning; optimization algorithms; autonomous systems; collision-free navigation; dynamic environments

---

1. Introduction

The automated industry is rapidly evolving, with robots being increasingly deployed in diverse applications such as manufacturing, healthcare, transportation, and smart homes [1,2]. To operate safely and efficiently in these environments, robots must navigate autonomously while avoiding obstacles, making path planning and motion planning critical components of robotics. Path planning involves generating a collision-free trajectory that guides a robot from an initial state to a desired goal. This requires accurate environment modeling to understand the spatial structure and relationships between locations. Three primary approaches are used to link the environment and navigation strategy: *Geometric*, *Topological*, and *Semantic*.

Geometric methods focus on guiding the robot from a start point to a goal based on map information, often leveraging mathematical representations of the environment [3]. These methods are widely used due to their simplicity and efficiency in static environments. *Topological methods* represent the

environment as a graph, enabling decision-making that mimics human-like navigation [4]. *Semantic approaches* use logical representations of the environment, incorporating human cognitive processes to infer navigation strategies [5]. While topological and semantic methods are gaining attention for their adaptability, geometric approaches remain highly relevant, especially when combined with optimization techniques [6].

This paper provides a systematic overview of geometric optimal navigation and path planning, focusing on the integration of classical geometric methods with modern optimization and machine learning techniques. We explore recent advancements in continuous optimization, real-time adaptability, and learning-based strategies, which enable robots to navigate complex environments, avoid moving obstacles, and optimize trajectories under constraints. Additionally, we highlight unresolved challenges, including scalability in high-dimensional spaces, real-time computation for dynamic environments, and the integration of perception systems for accurate environment modeling. Safety concerns in human-robot interactions are also discussed as critical issues for real-world deployment [7].

2. Navigation Methods

Path planning algorithms are essential for enabling autonomous robots to navigate complex environments efficiently. These algorithms can be broadly categorized into three primary approaches: **classical**, **reactive**, and **hybrid**. Each category addresses distinct challenges in path planning, offering unique advantages and limitations depending on the application and environment.

2.1. Classical Path Planning Methods

Classical path planning methods are rooted in geometric and mathematical approaches, relying on map reconstruction and environmental geometry to construct a navigable graph. These methods typically focus on finding the shortest or safest path between a start point and a target destination. Key techniques within this category include:

- **Roadmap Methods**: These methods create a graph representation of the environment, connecting obstacles and the target point. Examples include:
  - **Visibility Graph**: Generates shorter paths by connecting visible vertices of obstacles, optimizing for path length [8].
  - **Voronoi Diagram**: Prioritizes safety by creating paths that maximize the distance from obstacles, suitable for environments where collision avoidance is critical [9].
- **Cell Decomposition**: This approach divides the workspace into smaller cells, enabling the robot to navigate through collision-free regions. Notable examples include:
  - **Lozano-Perez's C-Space Decomposition (1983)**: Treats the robot as a C-shaped object and subdivides the environment into cells to identify feasible paths [10].
  - **Grid-Based Methods**: Popularized by Hachour (2008), this technique discretizes the environment into a grid, simplifying path planning through cell-by-cell exploration [11].
- **Potential Field Method**: Models the robot as a particle influenced by artificial potential fields. Attractive potentials guide the robot toward the goal, while repulsive potentials push it away from obstacles. This method ensures smooth navigation but can suffer from local minima issues [12].
- **Mathematical Programming**: Formulates path planning as an optimization problem, treating obstacle avoidance as a set of inequalities. The goal is to minimize a scalar quantity (e.g., path length or energy consumption) while finding a feasible curve between the start and target points [8].

Classical methods excel in static environments with well-defined maps but often struggle in dynamic settings due to their reliance on precomputed maps and lack of real-time adaptability.

2.2. Reactive Path Planning Methods

Reactive methods prioritize real-time decision-making by relying on sensory data rather than preconstructed maps. These approaches are well-suited for dynamic environments where obstacles and conditions change unpredictably. Key reactive techniques include:

- **Subsumption Architecture**: Organizes robot behaviors into hierarchical layers, with higher-level actions overriding lower-level ones. This structure enables quick responses to environmental changes but may lack global optimization [13].
- **Motor Schemas**: Generates output vectors for distinct behaviors (e.g., obstacle avoidance, goal seeking), which are combined through vector summation to determine the robot's overall response. This approach allows for flexible and adaptive navigation [14].

While reactive methods excel in dynamic environments and ensure collision avoidance, they do not guarantee optimal paths and may result in inefficient or oscillatory motion.

2.3. Hybrid Path Planning Methods

Traditional (classical) methods, often categorized as global navigation techniques, rely heavily on map reconstruction. While they ensure optimal or near-optimal paths, their computational burden increases significantly with larger or more complex environments [15]. Additionally, they may fail to handle dynamic obstacles effectively. Reactive methods, on the other hand, excel in dynamic settings but lack global optimization, potentially leading to suboptimal paths [16]. Hybrid methods combine the strengths of classical and reactive approaches, addressing the limitations of each. By integrating global planning with real-time adaptability, hybrid methods offer robust solutions for both static and dynamic environments. These methods can be categorized into:

- **Managerial Approaches**: Use a high-level planner to generate global paths while employing reactive strategies for local obstacle avoidance [17].
- **State Hierarchies**: Organize navigation tasks into hierarchical states, enabling seamless transitions between global and local planning [18].
- **Model-Oriented Styles**: Incorporate environmental models to enhance decision-making, balancing long-term planning with real-time adjustments [19].

Hybrid methods are particularly effective in complex environments, as they leverage the precision of classical methods and the adaptability of reactive strategies. However, they require careful design to balance computational efficiency and real-time performance.
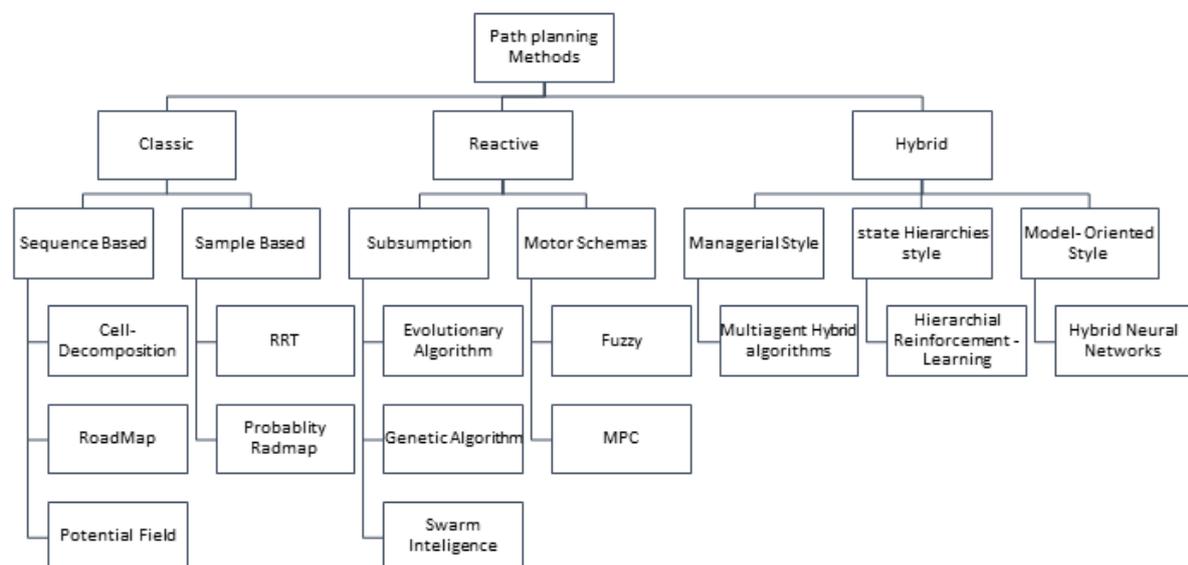

Figure 1. Integration of classical and reactive structures in hybrid path planning methods.

3. Optimization Criteria in Geometric Trajectory Planning

In geometric trajectory planning, the primary and most observable parameter is the determination of the shortest possible trajectory. However, while trajectory length is a critical factor, it should not be the sole criterion for optimization. A comprehensive approach must consider additional factors such as trajectory smoothness, time efficiency, energy consumption, and potential hazards, including collisions or environmental disturbances like wind resistance. Furthermore, the robot's specific task and objectives often necessitate the inclusion of other pertinent models and metrics to ensure optimal performance.

3.1. Key Optimization Criteria

The following criteria are essential for evaluating and optimizing trajectories in geometric path planning:

- **Trajectory Length**: Minimizing the path length is often the primary objective, as it directly impacts the robot's efficiency and resource utilization. Shorter trajectories reduce travel time and energy consumption, making them ideal for many applications [8].
- **Trajectory Smoothness**: Smooth trajectories are crucial for ensuring stable and efficient robot motion. Abrupt changes in direction or velocity can lead to mechanical stress, increased energy consumption, and reduced accuracy. Smoothness is often quantified using curvature and jerk metrics [20].
- **Time Efficiency**: Time-optimal trajectories are critical in applications where speed is a priority, such as in industrial automation or search-and-rescue operations. Time efficiency is closely tied to the robot's velocity profile and acceleration limits [21].
- **Energy Consumption**: Energy-efficient trajectories are vital for battery-powered robots or systems operating in energy-constrained environments. Optimizing energy usage involves minimizing unnecessary acceleration, deceleration, and idling [22].
- **Collision Avoidance**: Ensuring collision-free trajectories is a fundamental requirement in any navigation task. This involves not only avoiding static obstacles but also dynamically adapting to moving obstacles in real-time [12].
- **Environmental Factors**: External conditions such as wind resistance, terrain roughness, or fluid dynamics (in underwater or aerial robots) can significantly impact trajectory planning. These factors must be modeled and accounted for to ensure robust performance [23].

3.2. Path Length as an Optimization Criterion

Path length is one of the most critical parameters in the optimization of mobile robot path planning. It is frequently employed as a primary criterion for evaluating and optimizing trajectories. The length of a path is typically calculated by summing the distances traveled by the robot at each time step as it moves from its initial position to the target destination. Mathematically, path length can be determined using algebraic norms, with the Euclidean norm (or L_2 norm) being one of the most widely used distance metrics. The general definition of the L_p norm, denoted as $\| \cdot \|_p$, between two points in an n-dimensional space is given by:

$$\|\mathbf{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p}, \tag{1}$$

where $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ is a vector representing the coordinates of a point, and p is the order of the norm. For path planning in three-dimensional (3D) space, $n = 3$, and the coordinates x, y, and z correspond to the Cartesian axes.

The Euclidean norm (L_2 norm) is a special case of the L_p norm where $p = 2$. It is defined as:

$$\|\mathbf{x}\|_2 = \sqrt{x^2 + y^2 + z^2}. \tag{2}$$

Path Length Calculation

To calculate the total length of a path, the Euclidean norm is applied to each segment of the path. For a path consisting of m points, the total path length L_k for the k-th path is given by:

$$L_k = \sum_{l=1}^{m-1} \|\mathbf{p}_{l+1} - \mathbf{p}_l\|_2, \tag{3}$$

where $\mathbf{p}_l = (x_l, y_l, z_l)$ represents the coordinates of the l-th point on the path, and m is the total number of points, including the starting and target points. To illustrate this concept, consider Figure 2, which depicts two possible paths from an initial state to a target destination. Path 1 consists of 8 intermediate points ($m = 9$), while Path 2 consists of 6 intermediate points ($m = 7$). Using Equation (3), the total path lengths L_1 and L_2 for Path 1 and Path 2, respectively, can be calculated. Typically, the path with the shorter length is preferred, as it minimizes travel distance and, consequently, energy consumption and time expenditure [8].
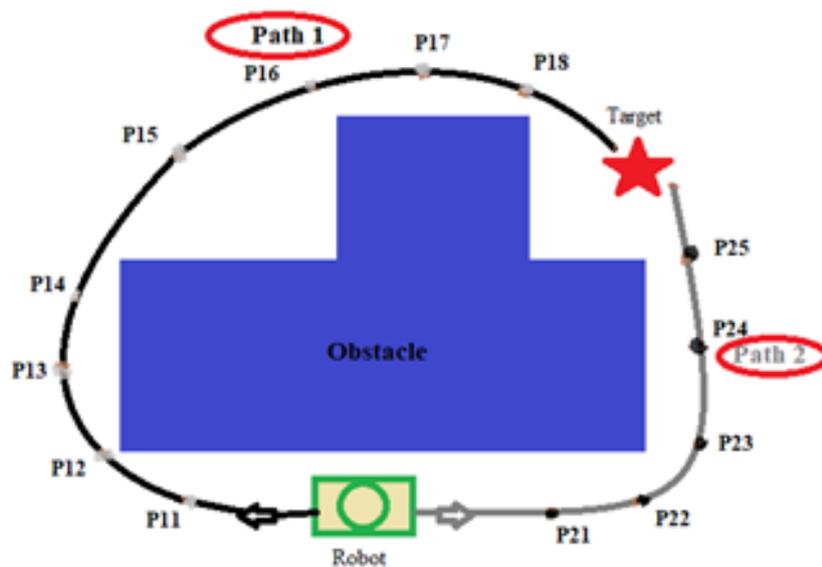

Figure 2. Two possible paths from an initial state to a target destination. Path 1 (blue) consists of 8 intermediate points, while Path 2 (red) consists of 6 intermediate points.

While minimizing path length is a common objective in path planning, it is essential to consider other factors such as trajectory smoothness, collision avoidance, and energy efficiency. For instance, a shorter path may require sharper turns or higher acceleration, which can increase energy consumption and mechanical stress on the robot [21]. Therefore, path length should be optimized in conjunction with other criteria to achieve a balanced and efficient trajectory.

3.3. Path Smoothness in Robotic Navigation

The notion of path smoothness in navigation arises from the limitations imposed by the robot's constrained angular velocity or the disparity in movement angles between its current intended state and the subsequent state (Figure 3) [24]. However, it is essential to acknowledge that the smoothness of a planned path is a complex outcome influenced by various factors, encompassing the robot's dynamics and kinematics, as well as the dynamics associated with surrounding obstacles. Parameters such as angle, velocity, acceleration, and environmental dynamics collectively contribute to determining the displacement and regularity of a path. Neglecting elements such as obstacle avoidance, non-holonomic features, and speed limitations can result in a significant disparity between the intended course and the actual path followed by the robot [24].

The pursuit of a smooth and uninterrupted path is motivated by its capability to enable the robot to navigate without sudden and acute turns. To accomplish path smoothness, three primary approaches are commonly employed: interpolation [24,25], special curves [26,27], and optimization [28, 29]. Interpolation algorithms, while aiming to generate regular paths, can encounter challenges such as high computational costs and the non-convergence issue known as Runge's phenomenon [30]. Transition curves, connecting straight segments of a path with a curve, offer advantages like zero start curvature, tangential joining of circular curves, and a uniform rate of curvature change. However, they necessitate tuning of control points and parameters and may incur high computational costs, proving less effective at higher speeds [31].

Optimization-Based Path Smoothing

One notably effective technique for achieving path smoothness is optimization-based path smoothing. This approach involves identifying the best path that satisfies various criteria, encompassing path length, safety, and energy consumption. Formulating the path design problem as an optimization issue allows the development of algorithms that seek paths meeting these criteria. Recent research has extensively explored diverse optimization techniques for path design and trajectory optimization in both ground and aerial vehicles [32–34].

Figure 3 provides a visual comparison between non-smooth and smooth paths, clearly illustrating that the smooth path not only covers a shorter distance but also follows a feasible route in terms of the robot's dynamics. Accordingly, Figure 3 illustrates that the angular difference is reduced in the smooth path when compared to the non-smooth path.
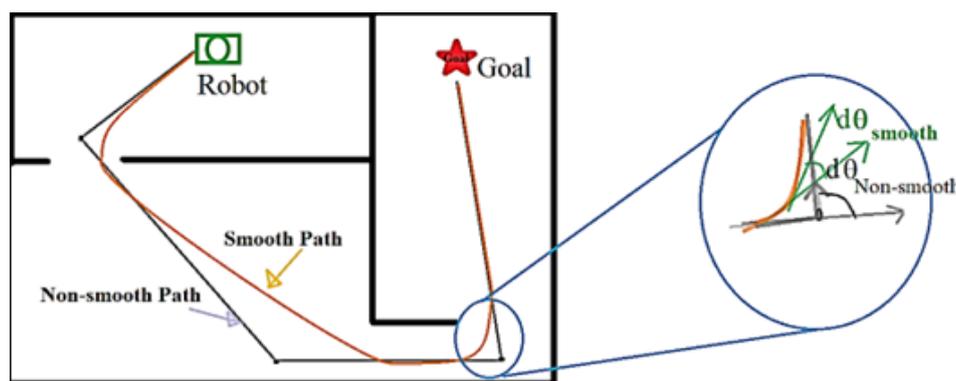

Figure 3. Movement angle difference in the smooth path and non-smooth path.

Three-Dimensional Navigation

Optimal strategies for navigating in three-dimensional environments have witnessed notable advancements, as a variety of approaches have been developed that incorporate transition curves to abide by kinematic and dynamic motion constraints. Transition curves play a crucial role in achieving a seamless integration of smooth paths within these navigation frameworks. The key lies in ensuring that these curves meet the intricate requirements imposed by the motion dynamics of the robot, thus guaranteeing a harmonious trajectory. The notion of path smoothing, which is essential for enhancing the efficiency of robotic systems in terms of navigation, is intricately intertwined with a set of motion constraints. This becomes particularly vital in the context of three-dimensional environments, where the robot must traverse complex spatial configurations. The utilization of transition curves serves as an advanced solution, contributing to the overall optimization of the robot's trajectory. Recent studies have introduced advanced smoothness criteria to address the limitations of traditional methods. These include:

- **Curvature Continuity**: Ensuring that the curvature of the path is continuous, which is critical for high-speed navigation and dynamic environments [35].
- **Jerk Minimization**: Minimizing the rate of change of acceleration (jerk) to ensure smoother motion and reduce wear on the robot's actuators [36].

- **Energy-Efficient Smoothing**: Optimizing paths to minimize energy consumption, which is particularly important for battery-operated robots [37].
- **Adaptive Smoothing**: Dynamically adjusting the smoothness of the path based on environmental changes and obstacle movements [38].

Path Smoothing Methods

Table 1 provides a summary of some of the well-established path-smoothing methods, including recent advancements. These methods leverage transition curves and encompass a diverse array of techniques aimed at addressing motion constraints. The table also includes detailed descriptions of the parameters and variables used in the mathematical formulations.

Table 1. Well-Known Path Smoothing Methods

Method	Mathematical Definition	Parameters and Description
Bézier Curve [24,27]	$B(t) = \sum_{i=0}^{n} \binom{n}{i}(1-t)^{n-i} t^i P_i$	**Parameters:** $t \in [0,1]$, P_i (control points), n (degree). **Description:** Smooth curve defined by control points, parameterized by t.
Elastic Stretching [39]	$\min \int \|\kappa(s)\|^2 ds$	**Parameters:** $\kappa(s)$ (curvature), s (arc length). **Description:** Minimizes total squared curvature for smooth paths.
Minimum Angle Difference [24]	$\min \|\theta_{i+1} - \theta_i\|$	**Parameters:** θ_i (angle at waypoint i). **Description:** Minimizes angular difference for smooth turns.
Curvature Continuity [35]	$\min \int \|\kappa'(s)\|^2 ds$	**Parameters:** $\kappa'(s)$ (curvature derivative). **Description:** Ensures continuous curvature along the path.
Jerk Minimization [36]	$\min \int \|j(t)\|^2 dt$	**Parameters:** $j(t)$ (jerk at time t). **Description:** Minimizes jerk for smoother motion.
Energy-Efficient Smoothing [37]	$\min \int \|F(v(t))\|^2 dt$	**Parameters:** $F(v(t))$ (force/energy as a function of velocity). **Description:** Optimizes energy consumption.
Adaptive Smoothing [38]	$\min \int \|\kappa(s,t)\|^2 ds$	**Parameters:** $\kappa(s,t)$ (curvature at arc length s and time t). **Description:** Dynamically adjusts smoothness based on environment.

This section has presented a holistic perspective on the integration of transition curves in optimal navigation methods, emphasizing the crucial consideration of both kinematic and dynamic constraints. By doing so, the goal is to contribute to the development of a comprehensive and efficient robotic navigation experience. In this context, optimization-based methods emerge as a key player, adept at striking a delicate balance between competing criteria.

Optimization-based methods, showcased as a robust framework, demonstrate their prowess in generating paths that not only optimize efficiency but also ensure safety in robotic navigation. Leveraging mathematical optimization algorithms, these methods engage in an iterative refinement of the trajectory. This iterative process takes into account a spectrum of factors. The ultimate result of these optimization-based approaches is the creation of a path that optimally navigates the conflicting objectives inherent in robotic navigation systems. This involves a delicate equilibrium between minimizing path length, ensuring safety through obstacle avoidance, and conserving energy. The

synthesis of these considerations culminates in an enhanced overall performance of robotic navigation systems, marking a significant stride toward achieving seamless and effective autonomous navigation.

3.4. Time Cost in Robotic Navigation

Time cost in navigation encapsulates the duration it takes for a robot to execute a predetermined path within its operational constraints. The imperative to minimize this time cost emerges as a pivotal optimization criterion, intricately shaped by the dynamic interplay of the robot's functionality and the environmental dynamics and kinematics. The overarching objective is to bolster navigation efficiency by curtailing the time required for precise path tracking.

Time-Optimal Trajectory Planning (TOTP)

At the forefront of time optimization strategies is **Time-Optimal Trajectory Planning (TOTP)**, a process finely tuned to craft a robot's path for expeditious tracking of the predetermined trajectory. The pivotal juncture when the robot seamlessly adheres to the path while meeting operational requisites is termed the **execution time**. TOTP methodologies are strategically crafted to streamline path tracking, minimizing the time cost, which unfolds as a multifaceted function intricately woven with the robot's dynamics and the environmental intricacies.

However, the resolution constraints inherent in both the robot and its environment pose formidable challenges to achieving TOTP, rendering modeling a formidable task. One pivotal approach involves maximizing velocity while judiciously considering constraints, mitigating the risk of undesirable jerking in the robot's motion. The delineation of two primary Time-Optimal criteria, namely **continuous time** and **discrete time** problem definitions, further enriches the spectrum of time optimization strategies. Two types of time-optimal criteria can be defined for the general TOTP problem: continuous time and discrete time problem definitions. The **continuous time-optimal optimization problem** is defined as:

$$\min_{u(t)} \int_0^T 1 \, dt \quad \text{subject to} \quad \dot{x}(t) = f(x(t), u(t)), \tag{4}$$

where:

- $f(x, u)$ is the state space of the robot dynamics,
- x is the state vector, consisting of position, velocity, and possibly acceleration,
- u is the control input that may depend on voltage, torque, or other functions of control manipulators,
- T is the total execution time.

The **discrete time-optimal optimization problem** is defined as:

$$\min_{u_k} \sum_{k=0}^N 1 \quad \text{subject to} \quad x_{k+1} = g(x_k, u_k), \tag{5}$$

where:

- $g(.)$ is the robot motion acceleration, which is a function of the control input u_k at the k-th time sample,
- N is the final time step when the robot reaches the goal,
- x_k is the k-th collision-free waypoint.

The methods for calculating the time cost of robotic navigation incorporate dynamic constraints, environmental factors, and optimization techniques to minimize the total execution time. Below are some key formulas used in recent research:

Time Cost with Dynamic Constraints

The time cost J for a trajectory can be calculated as:

$$J = \int_0^T \left(\|\dot{x}(t)\|^2 + \lambda \|u(t)\|^2 \right) dt, \tag{6}$$

where:

- $\dot{x}(t)$ is the velocity of the robot,
- $u(t)$ is the control input,
- λ is a weighting factor that balances the trade-off between velocity and control effort.

This formulation ensures that the robot minimizes both the time and energy consumption during navigation [40].

Time Cost with Environmental Constraints

In dynamic environments, the time cost must account for obstacles and environmental changes. A common approach is to use a penalty function:

$$J = \int_0^T \left(\|\dot{x}(t)\|^2 + \lambda \|u(t)\|^2 + \sum_i \phi_i(x(t)) \right) dt, \tag{7}$$

where:

- $\phi_i(x(t))$ is a penalty function that increases the cost when the robot approaches obstacles or violates environmental constraints,
- λ is a weighting factor for the control effort.

This method ensures collision-free navigation while minimizing time cost [41].

Time Cost in Multi-Robot Systems

For multi-robot systems, the time cost can be extended to include coordination constraints:

$$J = \sum_{j=1}^M \int_0^T \left(\|\dot{x}_j(t)\|^2 + \lambda \|u_j(t)\|^2 + \sum_{k \neq j} \psi_{jk}(x_j(t), x_k(t)) \right) dt, \tag{8}$$

where:

- M is the number of robots,
- $\psi_{jk}(x_j(t), x_k(t))$ is a penalty function that ensures collision avoidance between robots j and k.

This formulation is particularly useful in swarm robotics and collaborative tasks [42]. Recent research has introduced advanced methods to address the challenges of TOTP, including:

Deep Reinforcement Learning (DRL)

DRL-based approaches have been employed to learn time-optimal trajectories in complex environments [?]. These methods leverage neural networks to approximate the optimal policy, minimizing the time cost while satisfying dynamic constraints. The calculation criteria for DRL include:

- **Reward Function**: Designed to penalize time consumption and deviations from the desired trajectory.
- **State-Action Space**: Encodes the robot's dynamics and environmental constraints.
- **Training Efficiency**: Measured by the convergence rate and computational resources required.

Model Predictive Control (MPC)

MPC frameworks have been extended to incorporate time-optimal constraints, enabling real-time trajectory optimization [43]. The calculation criteria for MPC include:

- **Horizon Length**: Determines the number of future steps considered in the optimization.
- **Constraint Handling**: Ensures feasibility of the trajectory under dynamic and environmental constraints.
- **Computational Complexity**: Measured by the time required to solve the optimization problem at each time step.

Multi-Objective Optimization

Recent studies have combined time-optimality with other objectives, such as energy efficiency and safety, using Pareto optimization techniques [44]. The calculation criteria for multi-objective optimization include:

- **Pareto Front**: Represents the trade-off between competing objectives.
- **Weighting Factors**: Used to prioritize time-optimality over other objectives.
- **Scalability**: Evaluated based on the ability to handle high-dimensional state spaces.

Adaptive TOTP

Adaptive methods dynamically adjust the trajectory based on environmental changes, ensuring time-optimality in dynamic settings [45]. The calculation criteria for adaptive TOTP include:

- **Replanning Frequency**: Determines how often the trajectory is updated.
- **Convergence Speed**: Measures the time required to adapt to new environmental conditions.
- **Robustness**: Evaluated based on the ability to handle uncertainties and disturbances.

Applications and Challenges

Time-optimal trajectory planning finds applications in various domains, including:

- Autonomous vehicles,
- Industrial robotics,
- Aerial drones.

However, challenges such as computational complexity, real-time adaptability, and the trade-off between time-optimality and other constraints (e.g., energy consumption) remain active areas of research.

In a nutshell, the aim of TOTP involves the consideration of crucial factors, including acceleration, velocity, jerk, and the dynamic characteristics of the robot. In defining the TOTP problem, two primary criteria are typically emphasized: continuous time and discrete time optimization. These criteria are essential for optimizing the performance of robotic systems, ensuring the shortest time for trajectory execution. Recent advancements, such as DRL, MPC, and adaptive methods, have further enhanced the capabilities of TOTP, paving the way for more efficient and robust robotic navigation systems.

3.5. Energy Cost in Robotic Navigation

The optimization of energy consumption in mobile robots is a multifaceted endeavor intricately tied to the dynamic and kinematic model of the robot and environmental characteristics. Formulating optimal control strategies necessitates a nuanced consideration of the interplay between these factors. A pivotal criterion for effective energy management involves the reduction of the generated path length. Additionally, environmental features play a crucial role in shaping an optimum governing strategy for energy consumption in mobile robots. Research indicates that minimizing unnecessary movements and deploying predictive control algorithms are key strategies for significant energy reduction. This holistic perspective encompasses the summation of various energy costs, including:

- **Kinetic Energy (Ek)**: Energy associated with the robot's motion.
- **Traction Resistance Energy (Ef)**: Energy dissipated in overcoming traction resistances.
- **Motor Heating Energy (Ee)**: Energy lost as heat in the motors.
- **Mechanical Friction Energy (Em)**: Energy dissipated in overcoming friction torque.
- **Idle Energy (Eidle)**: Energy consumed by idling motors and onboard electric devices.

The total energy cost E_{total} of a mobile robot can be comprehensively defined as:

$$E_{\text{total}} = \int_0^T P_{\text{total}}(t)\,dt, \tag{9}$$

where:

- $P_{\text{total}}(t)$ is the total power consumption and loss at time t,
- T is the total execution time.

This formulation provides a comprehensive metric for evaluating and optimizing the energy efficiency of mobile robots [46]. Recent research has introduced advanced methods to optimize energy consumption in robotic navigation, including:

Predictive Energy Management

Predictive control algorithms leverage environmental data to minimize energy consumption. The energy cost can be expressed as:

$$E_{\text{predictive}} = \int_0^T \left(P_{\text{motion}}(t) + P_{\text{idle}}(t) \right) dt, \tag{10}$$

where:

- $P_{\text{motion}}(t)$ is the power consumed during motion,
- $P_{\text{idle}}(t)$ is the power consumed during idle states.

This approach has been shown to reduce energy consumption by up to 20% in dynamic environments [47].

Regenerative Energy Recovery

Regenerative braking systems recover kinetic energy during deceleration. The recovered energy E_{regen} can be calculated as:

$$E_{\text{regen}} = \int_0^T \eta \cdot P_{\text{braking}}(t) \, dt, \tag{11}$$

where:

- η is the efficiency of the regenerative braking system,
- $P_{\text{braking}}(t)$ is the power generated during braking.

This method is particularly effective in electric and hybrid robotic systems [48].

3.6. Risk Cost in Robotic Navigation

Ensuring safety is a pivotal consideration in path planning algorithms, and it is essential to create a risk map that accurately assesses the level of danger associated with different routes. The development of such a risk map is indispensable for evaluating the potential hazards when traversing specific positions, taking into account the presence of obstacles. To construct a risk map, a grid of probabilities is established using two-dimensional coordinates. Within this framework, a probability of zero signifies a negligible collision risk, while a probability of one indicates a heightened likelihood of collision. The computation of risk costs is contingent upon the probability of unforeseen events, encompassing:

- **Collision Risk**: Probability of collisions with environmental elements or individuals.
- **Robot Malfunction**: Probability of robot failure or abrupt movements.
- **Environmental Hazards**: Probability of natural events such as rain or wind increasing the risk of slipping or crashing.

The risk level $R(t)$ at a specific location $r(t)$ can be calculated as:

$$R(t) = P_{\text{event}}(t) \cdot C_{\text{event}}, \tag{12}$$

where:

- $P_{\text{event}}(t)$ is the probability of a risk event at time t,
- C_{event} is the cost associated with the event.

The total risk cost R_{total} of a generated path from r_0 to r_{goal} can be expressed as:

$$R_{\text{total}} = \int_0^T R(t) \, dt. \tag{13}$$

Recent studies have presented innovative techniques for evaluating risks in robotic navigation, which encompass:

Neural Network-Based Risk Prediction

Neural networks are used to predict collision probabilities based on historical data and real-time sensor inputs. The risk cost can be expressed as:

$$R_{\text{NN}}(t) = f_{\text{NN}}(r(t), \theta), \tag{14}$$

where:

- f_{NN} is the neural network function,
- θ represents the network parameters.

This approach has been shown to improve risk prediction accuracy by up to 30% [49].

Fuzzy Logic for Dynamic Risk Mapping

Fuzzy logic systems are employed to handle uncertainties in dynamic environments. The risk cost can be calculated as:

$$R_{\text{fuzzy}}(t) = \sum_i \mu_i(r(t)) \cdot C_i, \tag{15}$$

where:

- $\mu_i(r(t))$ is the membership function for the i-th risk factor,
- C_i is the cost associated with the i-th risk factor.

This method is particularly effective in unstructured environments [50].

Figure 4 illustrates a concrete case example of a risk map within the context of path planning. This visual representation showcases the practical application of risk assessment and mapping techniques in determining optimal paths, providing valuable insights for navigating through dynamic environments safely and efficiently.
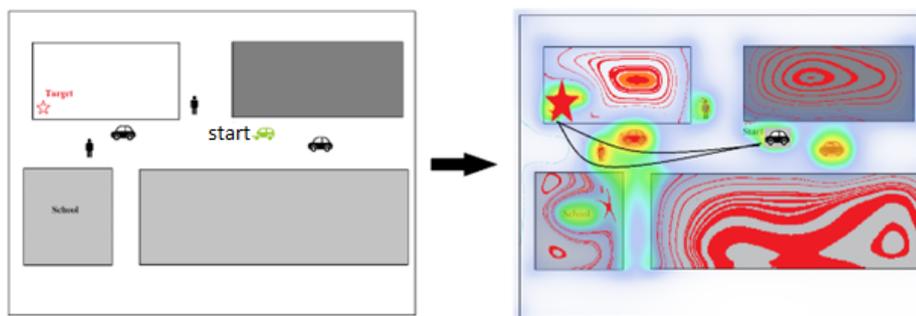

Figure 4. Risk Map Example and Possible Planned Path

3.7. Integration of Optimization Criteria

The integration of these criteria into a unified optimization framework is a challenging yet essential task. Multi-objective optimization techniques are often employed to balance competing criteria, such as minimizing trajectory length while ensuring smoothness and collision avoidance [51]. Additionally, task-specific requirements, such as payload constraints or mission deadlines, may further influence the optimization process.

4. Approaches to Solving Optimal Navigation Problems

To address optimal navigation challenges, researchers use various methodologies, including numerical and analytical approaches. Analytical methods aim to minimize costs directly to obtain optimal solutions, but they can be challenging to apply in complex and constrained navigation scenarios. Therefore, researchers tend to use numerical solutions, which include sampling and sequence-based techniques such as Greedy algorithms, Dynamic Programming, Evolutionary Algorithms, and Sampling-based algorithms. These techniques have been applied in various studies, including multi-robot navigation, off-road navigation, and navigation under localization uncertainty [52].

4.1. Greedy Algorithms

Greedy algorithms are a type of optimization algorithm that selects the best immediate solution at each step without considering the entire problem. These algorithms are top-down approaches that select the local best optimal solution at each step without considering the previous step. They are simple to implement and understand but do not guarantee the best overall solution and can be slow. Some classic sequence-based approaches of Greedy algorithms include Depth First Search (DFS) [53], Dijkstra's Algorithm [54], A* Algorithm [55], D* Algorithm [56], and Phi* Algorithm [57]. Greedy algorithms are commonly used for various applications, such as robotic navigation and problem-solving. While they can be ideal for problems with an optimal substructure, they may not always yield the best solution. The comparison provided in Figure 5 offers insights into the fundamental differences between the greedy algorithm and optimal approaches in navigation scenarios.
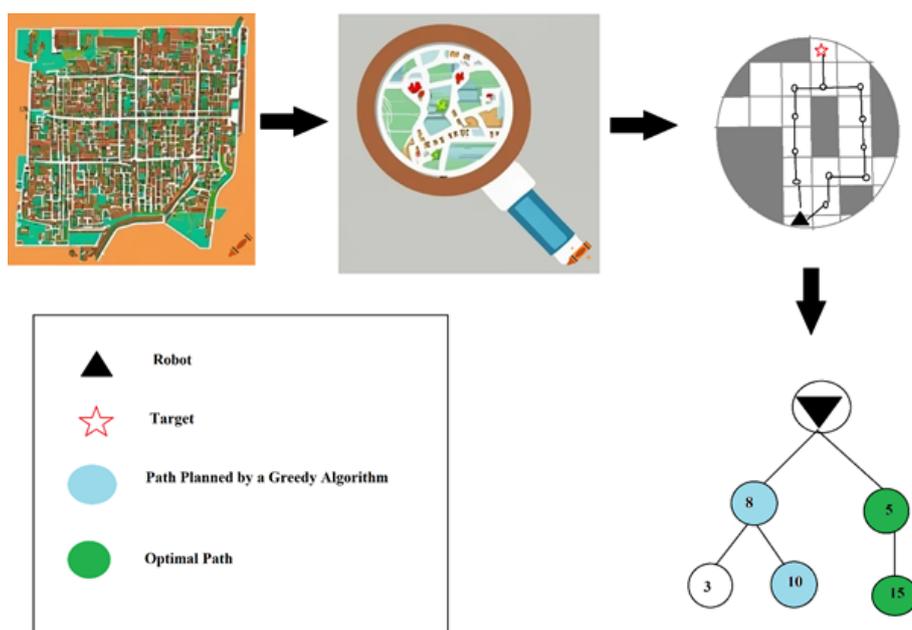

Figure 5. Greedy Algorithm application in path-planning and comparison with the optimal path.

4.2. Dynamic Programming (DP)

Dynamic Programming (DP) strategically addresses optimal problems by breaking them down into more manageable suboptimal problems, progressing towards the optimal solution step by step [52]. This process enhances efficiency and computational effectiveness. In the context of graph theory, DP formulations play a pivotal role in determining optimal sequences for visiting nodes within an adjacency graph associated with a decomposed environment. Figure 6 provides a general overview of dynamic programming in the context of navigation.
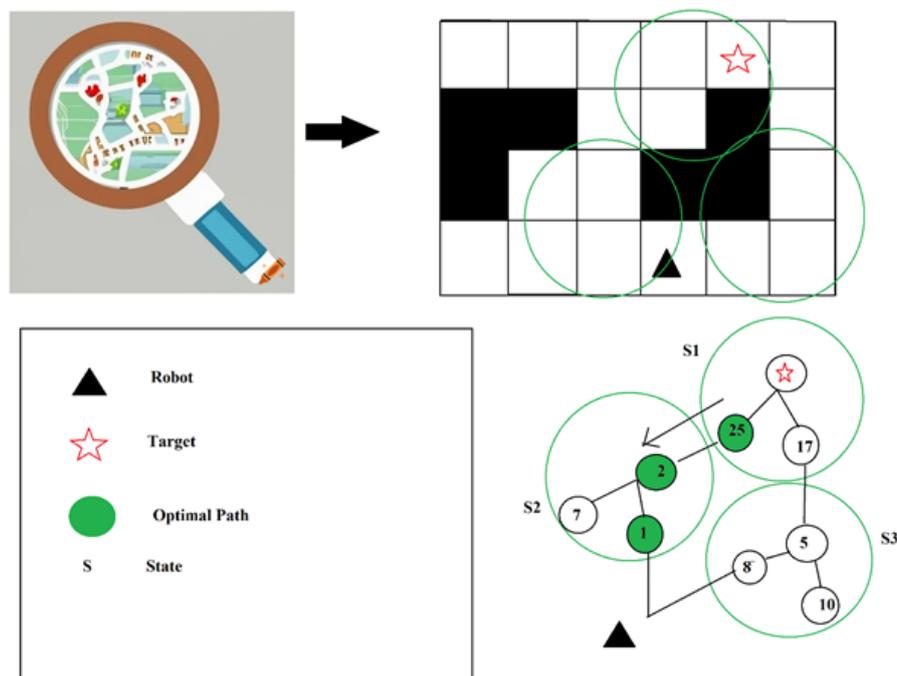
Figure 6. Optimal path generated by the Dynamic Programming approach.

4.3. Evolutionary Algorithms (EAs)

Evolutionary Algorithms (EAs) randomly select a candidate set of solutions and apply the quality function as an abstract fitness measure. EAs utilize a fitness function to find the optimal solution by converging from the initial state to the global optimal. These algorithms, including Particle Swarm Optimization (PSO) [58], Ant Colony Optimization (ACO) [59], and Genetic Algorithm (GA) [60], fall under the umbrella of EAs. Particularly effective in addressing multi-objective problems, they offer viable solutions for optimizing path planning scenarios with diverse criteria. The optimization process involves iteratively converging from an initial state towards the global optimum using a fitness function. However, it's essential to note that the inclusion of constraints can introduce computational challenges for these algorithms. Constraints are factors that restrict the feasible solution space, potentially increasing the computational burden. Despite this, EAs remain powerful tools for solving complex optimization problems, providing valuable solutions in various domains. Recent state-of-the-art (SOTA) approaches combine EAs with hybrid strategies, multi-objective optimization, and adaptive mechanisms to improve path quality, efficiency, and robustness.

Table 2. SOTA Approaches in Evolutionary Algorithms for Navigation and Path Planning.

Approach/Algorithm	Application Area	Main Advantages/Findings	Citation
Twin-Reinforced Chimp Optimization + Evolutionary Programming	Robot path planning	Outperforms other meta-heuristics in path length, consistency, time complexity, and success rate	(Zhang & Zhang, 2024)
Improved PSO with Evolutionary Operators (IPSO-EOPs)	Multi-robot navigation	Superior to DE and standard PSO in arrival time, safety, and energy use	(Das & Jena, 2020)
Many-Objective EAs (HypE, GrEA, KnEA, NSGA-III)	Agricultural robot route planning	HypE delivers best performance for minimizing navigation cost and turning angle	(Zhang et al., 2022)
Decomposition-based Multi-Objective EA (M2M-DW)	UAV path planning	Effectively handles constraints and infeasible solutions, reliable in complex scenarios	(Peng & Qiu, 2022; Jiang et al., 2024)
Multi-Objective Evolutionary PSO (MOEPSO)	Mobile robot path planning	Finds shortest, smoothest, and safest paths in static and dynamic environments	(Thammachantuek & Ketcham, 2022)
Bi-level Co-evolutionary Genetic Algorithm (IGA-CPP)	Coverage path planning	Efficient for irregular regions, fast convergence, optimized path length	(Chen et al., 2025)
NSGA-II and Multi-Objective EAs	Mobile robot navigation	NSGA-II excels in balancing path time and smoothness across diverse environments	(Jiménez-Domínguez et al., 2024)
Distributed Multi-Population EA	Maritime navigation	Multi-population approach improves solution quality over single-population EAs	(Smierzchalski et al., 2013)

4.4. Sampling-Based Algorithms

Sampling-based algorithms compromise between greedy algorithms and exploring data in unknown and large areas, making them suitable for high-dimensional and complex problems with low computational costs. Probabilistic Roadmap (PRM) [61], Rapidly-exploring Random Tree (RRT) [62], and Rapidly-exploring Random Tree Star (RRT*) [63] are some of the popular sampling-based algorithms. However, these algorithms may fail in dynamic environments. Recent advancements, such as Adaptive RRT* [64] and Deep Sampling-Based Planning [65], have improved their performance in dynamic and uncertain environments.

Table 3. Recent Sampling-Based Navigation Algorithms.

Method	Formula	Parameter Definitions
Adaptive RRT*	$x_{\text{new}} = x_{\text{near}} + \eta \frac{x_{\text{rand}} - x_{\text{near}}}{\|x_{\text{rand}} - x_{\text{near}}\|}$ $\eta_{k+1} = \alpha \eta_k + (1 - \alpha) f(\text{local density})$	x_{new}: new node x_{near}: nearest node x_{rand}: random sample η: adaptive step size α: smoothing factor $f(\cdot)$: environment feature function
Deep Sampling-Based	$x_{\text{sample}} \sim P_{\text{NN}}(x|\text{map}, \text{goal})$	P_{NN}: learned sampling distribution x_{sample}: sampled state
Reward-Adaptive Sampling	$P(x_j) = \frac{w_j}{\sum_k w_k}, \quad w_j = \frac{1}{d_j^{p_j}} \cdot r_j$	d_j: distance metric p_j: adaptive exponent r_j: reward factor
NAMR-RRT	Neural network predicts guiding regions and risk-aware expansion	Uses neural heuristics for sampling bias Incorporates dynamic obstacle risk metrics

Recent research has introduced advanced methods to address the challenges of optimal navigation, including:

- **Deep Reinforcement Learning (DRL)**: DRL-based approaches have been employed to learn optimal navigation policies in complex environments [66].
- **Model Predictive Control (MPC)**: MPC frameworks have been extended to incorporate dynamic constraints, enabling real-time navigation optimization [67].
- **Multi-Objective Optimization**: Recent studies have combined navigation objectives, such as time-optimality and energy efficiency, using Pareto optimization techniques [68].
- **Adaptive Navigation**: Adaptive methods dynamically adjust navigation strategies based on environmental changes, ensuring robustness in dynamic settings [69].

The selection of path planning strategies and styles should consider the characteristics of the robot and environments, as well as priorities and tasks. A general comparison of different path planning architectures is provided by [70]. Recent advancements in greedy algorithms, dynamic programming, evolutionary algorithms, and sampling-based algorithms have significantly improved the efficiency and robustness of optimal navigation systems. These approaches, combined with modern techniques such as DRL and MPC, pave the way for more effective and adaptive navigation solutions in complex and dynamic environments.

5. Overview of Collision-Free Path Planning Strategy

To craft robust path-planning algorithms, a meticulous understanding of both priorities and constraints specific to the problem is essential. Beyond prioritization, the path planning algorithm design must address various constraints inherent in the system. The process involves a comprehensive evaluation of factors influencing the navigation system's performance. Figure 7 provides valuable insight into the overarching block diagram of a navigation system, serving as a guide to identify additional elements crucial for an inclusive design approach. This holistic perspective ensures that the developed algorithms align with the intricate requirements of the navigation environment, fostering efficiency and adaptability.
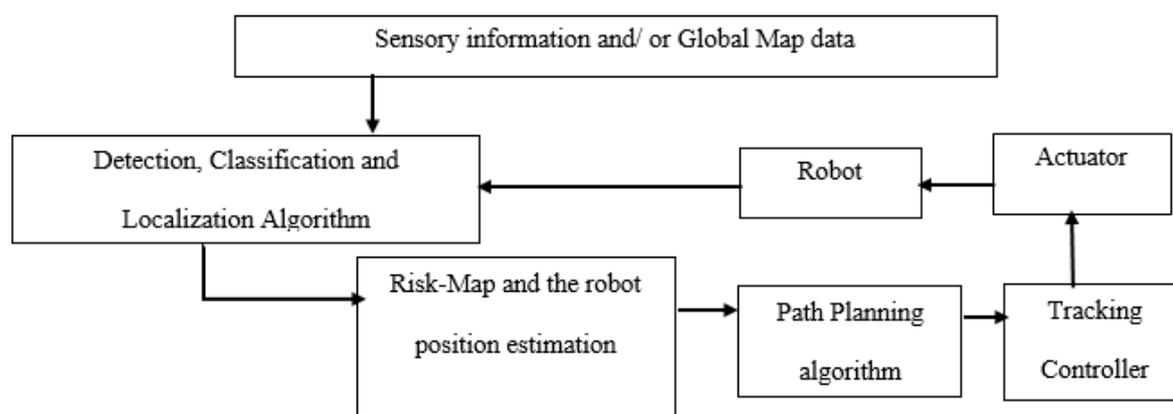

Figure 7. Overview of a collision-free path planning system diagram.

Assuming the robot is equipped with a tracking controller and relies on monitored motion, global map data, real-time sensory information, and obstacle positions, Figure 7 emphasizes the generation of a risk-map matrix and the design of a path-planning algorithm. While optimizing path length is a priority, the limitations of global approaches in ensuring safety within dynamic environments are acknowledged. To address this, a hybrid path planning approach integrates reactive and classic methods. This combination offers a robust solution to navigate challenges in dynamic environments, where real-time adaptation is crucial for ensuring both efficiency and safety. This approach can benefit from the advantages of both classic and reactive methods while reducing their deficiencies.

5.1. Hybrid Path Planning

Hybrid path planning combines the strengths of global and local planning methods to achieve robust navigation in dynamic environments. A typical hybrid approach can be formulated as:

$$P_{\text{hybrid}} = \alpha \cdot P_{\text{global}} + (1 - \alpha) \cdot P_{\text{local}}, \tag{16}$$

where:

- P_{hybrid} is the hybrid path,
- P_{global} is the globally optimal path,
- P_{local} is the locally adjusted path,
- α is a weighting factor that balances global and local planning.

Recent studies have explored adaptive weighting strategies to dynamically adjust α based on environmental changes [71].

5.2. Real-Time Adaptation

Real-time adaptation is crucial for ensuring collision-free navigation in dynamic environments. One approach involves the use of Model Predictive Control (MPC) to continuously update the robot's trajectory based on real-time sensory data. The MPC formulation can be expressed as:

$$\min_{u(t)} \sum_{t=0}^{T} \|x(t) - x_{\text{ref}}(t)\|^2 + \|u(t)\|^2, \tag{17}$$

where:

- $x(t)$ is the robot's state at time t,
- $x_{\text{ref}}(t)$ is the reference trajectory,
- $u(t)$ is the control input,
- T is the prediction horizon.

Recent advancements in MPC include the integration of deep learning models to improve prediction accuracy [72].

5.3. Recent Advancements in Collision-Free Path Planning

Recent research has introduced advanced methods to address the challenges of collision-free path planning, including:

- **Deep Reinforcement Learning (DRL)**: DRL-based approaches have been employed to learn collision-free navigation policies in complex environments [73].
- **Multi-Agent Path Planning**: Techniques for coordinating multiple robots to avoid collisions while achieving individual goals [74].
- **Uncertainty-Aware Planning**: Methods that account for uncertainties in sensor data and environmental dynamics [75].
- **Energy-Efficient Path Planning**: Energy-efficient path planning combines optimization of energy consumption with collision avoidance, employing various algorithmic strategies tailored to different robotic systems and environments. These approaches address challenges like terrain roughness, multi-agent coordination, and dynamic obstacles while minimizing motion costs [76].

The design of collision-free path planning algorithms requires a holistic approach that considers both global and local constraints. Recent advancements in risk-map generation, hybrid path planning, and real-time adaptation have significantly improved the robustness and efficiency of navigation systems. These approaches, combined with modern techniques such as DRL and MPC, pave the way for more effective and adaptive collision-free navigation solutions in complex and dynamic environments.

6. Challenges in Geometric Optimal Navigation

Geometric optimal navigation involves finding the shortest or most efficient path for a robot while avoiding obstacles and adhering to dynamic constraints. However, several challenges arise in this domain, particularly in high-dimensional spaces, real-time computation, and the integration of perception systems. This section discusses these challenges, recent advancements, and their implications for robotic navigation.

6.1. Scalability in High-Dimensional Spaces

High-dimensional spaces pose significant computational challenges for path planning. As the dimensionality of the environment increases, the computational complexity grows exponentially, making it difficult to find optimal paths efficiently. Recent studies have explored dimensionality reduction techniques and machine learning to address this issue [77].

6.1.1. Dimensionality Reduction Techniques

Dimensionality reduction techniques, such as Principal Component Analysis (PCA) and t-Distributed Stochastic Neighbor Embedding (t-SNE), have been applied to reduce the complexity of high-dimensional spaces. Mathematically, PCA can be expressed as:

$$\mathbf{Y} = \mathbf{X} \cdot \mathbf{W}, \tag{18}$$

where:

- $\mathbf{X}$ is the high-dimensional data matrix,
- $\mathbf{W}$ is the transformation matrix,
- $\mathbf{Y}$ is the low-dimensional representation.

These techniques help reduce the computational burden while preserving the essential structure of the environment [78].

6.1.2. Machine Learning Approaches

Machine learning approaches, such as deep autoencoders, have been used to learn compact representations of high-dimensional spaces. The autoencoder loss function is given by:

$$\mathcal{L} = \|\mathbf{X} - \hat{\mathbf{X}}\|^2, \tag{19}$$

where $\hat{\mathbf{X}}$ is the reconstructed data. These methods enable efficient path planning in high-dimensional environments by learning low-dimensional embeddings [79].

6.2. Real-Time Computation for Dynamic Environments

Real-time navigation in dynamic environments involves solving optimal path planning problems under stringent timing requirements while accounting for moving obstacles, robot dynamics, and environmental uncertainty. As problem complexity scales with environment size, obstacle density, and number of agents, scalable algorithmic techniques become vital [80]. The *optimal navigation problem* in a dynamic environment can be formulated as:

$$\min_{\mathbf{x}(t),\mathbf{u}(t)} \quad J = \int_0^T c\big(\mathbf{x}(t),\mathbf{u}(t),t\big)\,dt \tag{20}$$

$$\text{s.t.} \quad \begin{cases} \dot{\mathbf{x}}(t) = f(\mathbf{x}(t),\mathbf{u}(t),t) & \text{(dynamics)} \\ \mathbf{x}(t) \notin \mathcal{O}(t) & \text{(collision avoidance)} \\ \mathbf{x}(0) = \mathbf{x}_{\text{start}}, \quad \mathbf{x}(T) = \mathbf{x}_{\text{goal}} & \text{(boundary conditions)} \\ \mathbf{u}(t) \in \mathcal{U} & \text{(control limits)} \end{cases}$$

Where $\mathbf{x}(t)$ is the robot state trajectory, $\mathbf{u}(t)$ are controls, $c(\cdot)$ is a cost integrating path length, smoothness, energy, or safety margins, and $\mathcal{O}(t)$ models time-varying obstacles.

Scalability challenges arise from the curse of dimensionality, dynamic constraints, and frequent replanning necessity triggered by environmental changes.

Overall scalability strategies include:

- Parallelizing computationally intensive steps (sampling, collision checking, graph search)
- Adaptive spatial and temporal resolution to reduce unnecessary computation
- Decomposing the problem into hierarchical or modular subproblems
- Leveraging learned models to prune search space or estimate costs rapidly

6.3. Parallel Optimization for Real-Time Trajectory Planning

Optimization-based planners seek continuous trajectories by minimizing a cost function under dynamic and environmental constraints. While effective in theory, traditional solvers often converge too slowly for real-time navigation, limiting their use in dynamic environments.

Parallel gradient-based optimization addresses these challenges by leveraging multi-core CPUs or GPUs to accelerate cost and gradient evaluations. These methods enable planners to meet strict real-time requirements while maintaining trajectory quality.

Algorithm 1 Real-Time Parallel Trajectory Optimization

1: **Input:** Previous trajectory $\mathbf{x}(t)$, control inputs $\mathbf{u}(t)$ (warm-start)
2: **while** within time budget and convergence not met **do**
3:   **Parallel:** For each knot along the trajectory:
4:     Compute gradients for smoothness, obstacle cost, and dynamics
5:     Compute constraint gradients (e.g., collision or control limits)
6:   Aggregate gradients
7:   Update $\mathbf{x}(t)$ and $\mathbf{u}(t)$ (e.g., via projected gradient descent)
8:   Project updated trajectory to enforce hard constraints
9: **end while**
10: **Output:** Refined trajectory $\mathbf{x}^*$, controls $\mathbf{u}^*$

6.4. Strategies for Real-Time Scalability

Effective real-time trajectory optimization relies on a combination of algorithmic innovations and hardware acceleration. The following strategies form the backbone of scalable real-time planners:

- **GPU-Accelerated Evaluation:** Modern GPUs allow hundreds to thousands of trajectory knot gradients to be evaluated simultaneously, enabling speed-ups by orders of magnitude compared to serial CPUs. This parallelism dramatically reduces iteration times for gradient computation and constraint checking, empowering planners to meet real-time deadlines even in high-dimensional state spaces [81–83].
  Recent work by Rastgar [82] proposes novel GPU-parallel optimization algorithms that adapt constraint formulations to fully leverage GPU architectures, markedly enhancing scalability and robustness in dynamic scenarios. Similarly, Yu et al. [83] introduce TOP, a trajectory optimization via parallel consensus ADMM that achieves near-constant time complexity per iteration by decomposing long trajectories into parallelizable segments, enabling large-scale real-time path planning on GPUs.
- **Warm Starting:** Utilizing the previously computed optimal trajectory as an initial guess accelerates convergence of iterative solvers. This approach is especially effective in dynamic environments where consecutive plans differ only slightly [82,84].
  By reusing prior solutions, planners reduce redundant computation while improving continuity and smoothness of resulting trajectories. This practical technique aligns with state-of-the-art GPU-accelerated approaches that integrate warm-starting for real-time feasibility.

- **Receding Horizon Execution:** Instead of optimizing over the entire trajectory horizon, planners focus on a shorter, fixed-duration segment. Only the initial portion of the plan is executed before replanning occurs, maintaining continual responsiveness to environmental changes and dynamic obstacles [82,85].

  This limited horizon approach bounds computational demands, facilitates faster replanning, and supports adaptive trajectory refinement, key for scalable navigation in cluttered or rapidly changing environments.

- **Adaptive Termination and Step Sizing:** Optimization algorithms dynamically adjust their step sizes and employ early stopping criteria once trajectories reach a suitable quality level within the operational time budget. This balances the trade-off between latency and solution optimality.

  For example, algorithms may terminate as soon as collision-free smooth paths are found, even if not perfectly optimal, ensuring timely availability of actionable plans without compromising safety [83].

- **Integrated Recent Advances:**

  Incorporating the above core principles, recent methods extend scalability and robustness in real-time planning:

  - **Semantic-Aware Optimization:** He et al. [86] develop a spatio-temporal semantic graph optimizer tailored for urban autonomous driving. Their approach handles dynamic obstacle semantics through sparse graph formulations, enabling real-time feasible trajectories that intelligently incorporate semantic understanding of traffic participants and road elements.
  - **Hybrid Sampling and Rewiring:** Silveira et al. [87] propose RT-FMT, a hybrid of Fast Marching Tree and RT-RRT*, that combines incremental rewiring and local-to-global tree reuse for faster execution and improved path quality in dynamic environments. By efficiently reusing prior tree structures and limiting expansion scope, RT-FMT exemplifies algorithmic decomposition enabling scalability.
  - **Piecewise Parallel Optimization:** Yu et al. [83] introduce the TOP framework which decomposes long trajectories into smaller segments solved in parallel while ensuring high-order continuity through consensus constraints. Deploying this method on GPUs supports extremely large-scale and long-horizon real-time trajectory optimization, pushing the frontier of computational performance in robotics.
  - **Constraint Reformulation for Parallelism:** Rastgar's thesis [82] innovates by remodeling kinematic and collision constraints to be more amenable to parallel GPU computation, thereby enhancing planner scalability and resulting in more reliable real-time performance.

Together, these innovations illustrate a robust framework for trajectory optimization under real-time constraints, combining modern parallel hardware, algorithmic acceleration, and semantic-aware planning to maintain low latency and high fidelity in dynamic, obstacle-rich environments.

6.5. Integration of Perception Systems

Accurate environment modeling relies on the seamless integration of perception systems, such as LiDAR, cameras, and IMUs. Recent advancements in multi-sensor fusion and deep learning have improved perception accuracy [88].

6.5.1. Deep Learning for Perception

Deep learning models, such as convolutional neural networks (CNNs), have been used to process sensory data and extract meaningful features. The loss function for training a CNN is given by:

$$\mathcal{L} = \sum_{i=1}^{N} \|y_i - \hat{y}_i\|^2, \tag{21}$$

where y_i and $\hat{y}_i$ are the ground truth and predicted values, respectively. These models improve perception accuracy in complex environments [89].

Geometric optimal navigation is a critical component of robotic systems, enabling efficient and safe movement in complex environments. However, future research should focus on developing more adaptive and energy-efficient navigation strategies, as well as improving the robustness of perception systems in uncertain and dynamic environments. By addressing these challenges, we can pave the way for more effective and reliable robotic navigation systems.

6.6. Ethical and Safety Concerns in Human-Robot Interactions

Ethical and safety concerns are critical for the real-world deployment of autonomous systems. Recent research has focused on developing ethical frameworks and safety protocols for human-robot interactionsEthical frameworks guide robots to act in ways that respect societal norms and human rights. Core principles include:

- **Transparency**: Robots should clearly communicate intentions and behaviors to enhance human trust.
- **Accountability**: Developers and operators must be accountable for system behavior, particularly in high-stakes contexts.
- **Fairness**: Systems should actively mitigate bias and promote equitable treatment across user demographics.

Ethically-aligned systems must reconcile trade-offs between efficiency, safety, and moral responsibility [90,91]. Recent research emphasizes formalizing these principles in algorithms for real-time navigation and decision-making [87].

Beyond ethical reasoning, physical safety remains a critical design constraint. Key safety mechanisms include:

- **Collision Avoidance**: Sensor fusion and predictive models allow robots to anticipate and avoid human contact.
- **Emergency Stop Mechanisms**: Systems must be capable of halting immediately under risk conditions.
- **Human-in-the-Loop Control**: Dynamic shared autonomy enables humans to intervene in uncertain or dangerous scenarios.

Recent developments use machine learning to dynamically assess and adapt to human proximity and behavior, improving safety in unpredictable settings [92–94].

6.7. Toward Human-Centric Design and Learning

Human-centric design integrates physical, cognitive, and emotional dimensions of safety:

- **Safe Reinforcement Learning (Safe RL)**: Safety-aware RL ensures that robots learn policies without violating safety constraints—even in exploration phases [92,95].
- **Context-Aware Adaptation**: Integrating human feedback and context improves trust and user comfort in collaborative robotics [94,96].

Future systems must adopt unified frameworks that combine ethical reasoning, real-time risk assessment, and safe learning under uncertainty [86,87].

While progress is promising, significant challenges remain:

- **Integration Complexity**: Embedding ethical and safety modules into fast, real-time planners on resource-constrained platforms remains difficult.
- **Standardization Gaps**: Harmonization of international safety and ethics standards (e.g., ISO 12100, ISO/TS 15066) with learning-based models is ongoing [93].
- **Human Perception**: Models often neglect psychological dimensions such as perceived safety and emotional response.

Looking forward, future research should aim to integrate ethical decision-making algorithms, robust safe reinforcement learning methods, and human-centric design principles into cohesive

frameworks. These frameworks must balance operational efficiency with respect for human values and dynamic risk management, paving the way for autonomous robotic systems that are reliable, trustworthy, and ethically aligned [86,87,91].

By addressing these multifaceted challenges, the robotics community can accelerate the deployment of effective, reliable, and ethically sound robotic navigation and collaboration systems, fostering safer and more harmonious human-robot coexistence.

7. Conclusion

This survey presents a comprehensive synthesis of geometric optimal navigation and path planning strategies, with an emphasis on scalability, real-time performance, and adaptability to dynamic environments. By unifying classical geometric methods with contemporary optimization techniques and learning-based models, including GPU-accelerated frameworks and neural approximators, this paper underscores the significant strides made toward real-time, scalable autonomous navigation.

The reviewed literature reveals a clear trend toward hybrid approaches—combining hierarchical decomposition, parallel computation, and learning-augmented modules—to tackle increasing computational demands and environmental complexity. Recent advancements such as attention-based neural networks, particularly Transformer architectures, offer new pathways to represent spatial-temporal dependencies and plan in high-dimensional, uncertain spaces more efficiently.

This work also highlights the importance of ethical and safety considerations in deploying optimization-based navigation systems. The real-world deployment of autonomous agents in safety-critical domains (e.g., healthcare robotics, autonomous vehicles, drones) necessitates guarantees on bounded suboptimality, resilience to adversarial conditions, and alignment with human-centric values.

Key Takeaways for Future Research

- **Transformer-Based Planning:** Future systems should explore integrating spatial-temporal attention mechanisms into planning pipelines to improve generalization and context-aware navigation, especially in multi-agent and partially observable environments.
- **Real-Time and Embedded Efficiency:** Further innovation is needed to support GPU and neuromorphic execution on power-constrained platforms, ensuring autonomy is feasible for small-scale robots and edge devices.
- **Scalable Multi-Agent Coordination:** Scalability remains a bottleneck. Approaches that combine decentralized optimization, learning-based approximations, and adaptive communication protocols are promising.
- **Safe and Ethical Optimization:** New planning frameworks should incorporate constraints and verification layers that explicitly account for safety, fairness, and human preferences, particularly when operating alongside humans.
- **Standardization and Benchmarking:** To assess progress meaningfully, standardized evaluation frameworks and real-world benchmarks—particularly those involving uncertainty, real-time constraints, and ethical dilemmas—must be developed.

In summary, the convergence of parallel computing, geometric control theory, and deep learning promises robust, real-time path planning in dynamic environments. However, ensuring ethical behavior, safety guarantees, and hardware efficiency will remain central challenges in realizing truly autonomous, intelligent agents.

References

1. Licardo, J.T.; Domjan, M.; Orehovački, T. Intelligent robotics—A systematic review of emerging technologies and trends. *Electronics* **2024**, *13*, 542.
2. Kamuni, N.; Dodda, S.; Vuppalapati, V.S.M.; Arlagadda, J.S.; Vemasani, P. Advancements in Reinforcement Learning Techniques for Robotics. *Journal of Basic Science and Engineering*, *19*, 101–111.
3. Zhang, Y.; Zhao, W.; Wang, J.; Yuan, Y. Recent progress, challenges and future prospects of applied deep reinforcement learning: A practical perspective in path planning. *Neurocomputing* **2024**, *608*, 128423.
4. Kanakis, G.S.; Rovithakis, G.A. Motion Planning in Topologically Complex Environments Via Hybrid Feedback. *IEEE Transactions on Automatic Control* **2024**.
5. Alqobali, R.; Alshmrani, M.; Alnasser, R.; Rashidi, A.; Alhmiedat, T.; Alia, O.M. A survey on robot semantic navigation systems for indoor environments. *Applied Sciences* **2023**, *14*, 89.
6. Zhao, Z.; Cheng, S.; Ding, Y.; Zhou, Z.; Zhang, S.; Xu, D.; Zhao, Y. A survey of optimization-based task and motion planning: From classical to learning approaches. *IEEE/ASME Transactions on Mechatronics* **2024**.
7. Sutikno, T. The future of artificial intelligence-driven robotics: Applications and implications. *IAES Int. J. Robot. Autom* **2024**, *13*, 361–372.
8. LaValle, S.M. *Planning Algorithms*; Cambridge University Press, 2006.
9. Choset, H.; Lynch, K.M.; Hutchinson, S.; Kantor, G.; Burgard, W.; Kavraki, L.E.; Thrun, S. *Principles of Robot Motion: Theory, Algorithms, and Implementations*; MIT Press, 2005.
10. Lozano-Perez, T. Spatial Planning: A Configuration Space Approach. *IEEE Transactions on Computers* **1983**, *32*, 108–120.
11. Sánchez-Ibáñez, J.R.; Pérez-del Pulgar, C.J.; García-Cerezo, A. Path planning for autonomous mobile robots: A review. *Sensors* **2021**, *21*, 7898.
12. Khatib, O. Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. *The International Journal of Robotics Research* **1986**, *5*, 90–98.
13. Brooks, R.A. A Robust Layered Control System for a Mobile Robot. *IEEE Journal on Robotics and Automation* **1986**, *2*, 14–23.
14. Burgess-Limerick, B.; Haviland, J.; Lehnert, C.; Corke, P. Reactive base control for on-the-move mobile manipulation in dynamic environments. *IEEE Robotics and Automation Letters* **2024**, *9*, 2048–2055.
15. Zhang, Y.; Zhao, W.; Wang, J.; Yuan, Y. Recent progress, challenges and future prospects of applied deep reinforcement learning: A practical perspective in path planning. *Neurocomputing* **2024**, *608*, 128423.
16. Mohanan, M.; Salgoankar, A. A survey of robotic motion planning in dynamic environments. *Robotics and Autonomous Systems* **2018**, *100*, 171–185.
17. Amar, L.B.; Jasim, W.M. Hybrid metaheuristic approach for robot path planning in dynamic environment. *Bulletin of Electrical Engineering and Informatics* **2021**, *10*, 2152–2162.
18. He, H.; Shi, P.; Zhao, Y. Adaptive connected hierarchical optimization algorithm for minimum energy spacecraft attitude maneuver path planning. *Astrodynamics* **2023**, *7*, 197–209.
19. Wang, N.; Jin, Z.; Wang, T.; Xiao, J.; Zhang, Z.; Wang, H.; Zhang, M.; Li, H. Hybrid path planning methods for complete coverage in harvesting operation scenarios. *Computers and Electronics in Agriculture* **2025**, *231*, 109946.
20. Ben-Messaoud, W.; Basset, M.; Lauffenburger, J.P.; Orjuela, R. Smooth obstacle avoidance path planning for autonomous vehicles **2018**. pp. 1–6.
21. Betts, J.T. *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*; SIAM, 2010.
22. Zhang, H.; Zhang, Y.; Yang, T. A survey of energy-efficient motion planning for wheeled mobile robots. *Industrial Robot: the international journal of robotics research and application* **2020**, *47*, 607–621.
23. Liu, J.; Yan, Y.; Yang, Y.; Li, J. An improved artificial potential field UAV path planning algorithm guided by RRT under environment-aware modeling: Theory and simulation. *IEEE Access* **2024**, *12*, 12080–12097.
24. Karur, K.; Sharma, N.; Dharmatti, C.; Siegel, J.E. A survey of path planning algorithms for mobile robots. *Vehicles* **2021**, *3*, 448–468.
25. Sun, S.; Yu, D.; Wang, C.; Xie, C. A smooth tool path generation and real-time interpolation algorithm based on B-spline curves. *Advances in Mechanical Engineering* **2018**, *10*, 1687814017750281.
26. Sun, S.; Yu, D.; Wang, C.; Xie, C. A smooth tool path generation and real-time interpolation algorithm based on B-spline curves. *Advances in Mechanical Engineering* **2018**, *10*, 1687814017750281.
27. Simba, K.R.; Uchiyama, N.; Sano, S. Real-time smooth trajectory generation for nonholonomic mobile robots using Bézier curves. *Robotics and Computer-Integrated Manufacturing* **2016**, *41*, 31–42.

28. Zafar, M.N.; Mohanta, J. Methodology for path planning and optimization of mobile robots: A review. *Procedia computer science* **2018**, *133*, 141–152.

29. Niu, H.; Lu, Y.; Savvaris, A.; Tsourdos, A. An energy-efficient path planning algorithm for unmanned surface vehicles. *Ocean Engineering* **2018**, *161*, 308–321.

30. Belanger, N. External fake constraints interpolation: The end of Runge phenomenon with high degree polynomials relying on equispaced nodes Application to aerial robotics motion planning. *indicator* **2017**, *2*, 1–9.

31. Villagra, J.; Milanés, V.; Rastelli, J.P.; Godoy, J.; Onieva, E. Path and speed planning for smooth autonomous navigation **2012**.

32. Wang, H.; Ren, G.; Chen, J.; Ding, G.; Yang, Y. Unmanned aerial vehicle-aided communications: Joint transmit power and trajectory optimization. *IEEE Wireless Communications Letters* **2018**, *7*, 522–525.

33. Liu, J.; Anavatti, S.; Garratt, M.; Abbass, H.A. Modified continuous ant colony optimisation for multiple unmanned ground vehicle path planning. *Expert Systems with Applications* **2022**, *196*, 116605.

34. Ma, C.; Zhao, M.; Liu, Y. Vehicle navigation path optimization based on complex networks. *Physica A: Statistical Mechanics and its Applications* **2025**, *665*, 130509.

35. Wang, X.; Jiang, P.; Li, D.; Sun, T. Curvature continuous and bounded path planning for fixed-wing UAVs. *Sensors* **2017**, *17*, 2155.

36. Dai, C.; Lefebvre, S.; Yu, K.M.; Geraedts, J.M.; Wang, C.C. Planning jerk-optimized trajectory with discrete time constraints for redundant robots. *IEEE Transactions on Automation Science and Engineering* **2020**, *17*, 1711–1724.

37. Datouo, R.; Motto, F.B.; Zobo, B.E.; Melingui, A.; Bensekrane, I.; Merzouki, R. Optimal motion planning for minimizing energy consumption of wheeled mobile robots **2017**. pp. 2179–2184.

38. Liu, X.; Li, Y.; Zhang, J.; Zheng, J.; Yang, C. Self-adaptive dynamic obstacle avoidance and path planning for USV under complex maritime environment. *Ieee Access* **2019**, *7*, 114945–114954.

39.

40. Shen, P.; Zhang, X.; Fang, Y. Complete and time-optimal path-constrained trajectory planning with torque and velocity constraints: Theory and applications. *IEEE/ASME Transactions on Mechatronics* **2018**, *23*, 735–746.

41. Xu, Z.; Han, X.; Shen, H.; Jin, H.; Shimada, K. Navrl: Learning safe flight in dynamic environments. *IEEE Robotics and Automation Letters* **2025**.

42. Choudhury, S.; Gupta, J.K.; Kochenderfer, M.J.; Sadigh, D.; Bohg, J. Dynamic multi-robot task allocation under uncertainty and temporal constraints. *Autonomous Robots* **2022**, *46*, 231–247.

43. Saeedinia, S.A.; Tale Masouleh, M. The synergy of the multi-modal MPC and Q-learning approach for the navigation of a three-wheeled omnidirectional robot based on the dynamic model with obstacle collision avoidance purposes. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* **2022**, *236*, 9716–9729.

44. Mohaghegh, M.; Saeedinia, S.A.; Roozbehi, Z. Optimal predictive neuro-navigator design for mobile robot navigation with moving obstacles. *Frontiers in Robotics and AI* **2023**, *10*, 1226028.

45. Zhang, L.; Wang, Y.; Zhao, X.; Zhao, P.; He, L. Time-optimal trajectory planning of serial manipulator based on adaptive cuckoo search algorithm. *Journal of Mechanical Science and Technology* **2021**, *35*, 3171–3181.

46. Zhang, H.; Zhang, Y.; Yang, T. A survey of energy-efficient motion planning for wheeled mobile robots. *Industrial Robot: the international journal of robotics research and application* **2020**, *47*, 607–621.

47. Parasuraman, R.; Kershaw, K.; Pagala, P.; Ferre, M. Model based on-line energy prediction system for semi-autonomous mobile robots **2014**. pp. 411–416.

48. Palomba, I.; Wehrle, E.; Carabin, G.; Vidoni, R. Minimization of the energy consumption in industrial robots through regenerative drives and optimally designed compliant elements. *Applied Sciences* **2020**, *10*, 7475.

49. Shamsfakhr, F.; Bigham, B.S. A neural network approach to navigation of a mobile robot and obstacle avoidance in dynamic and unknown environments. *Turkish Journal of Electrical Engineering and Computer Sciences* **2017**, *25*, 1629–1642.

50. Jacob, I.J. Fuzzy Logic Decision Model for Robust Risk Management in ubiquitous environment-A Review. *Journal of Ubiquitous Computing and Communication Technologies* **2024**, *5*, 399–411.

51. Zhang, X.; Shi, G. Multi-objective optimal trajectory planning for manipulators in the presence of obstacles. *Robotica* **2022**, *40*, 888–906.

52. Mellouk, A.; Benmachiche, A. A survey on navigation systems in dynamic environments **2020**. pp. 1–7.

53. Xiao, B.; Zhang, Z.; Wang, Q.; Zhang, B.; Zheng, S. TSP-based depth-first search algorithms for enhanced path planning in laser-based directed energy. *Precision Engineering* **2025**, *93*, 224–236.

54. Sundarraj, S.; Reddy, R.V.K.; Basam, M.B.; Lokesh, G.H.; Flammini, F.; Natarajan, R. Route planning for an autonomous robotic vehicle employing a weight-controlled particle swarm-optimized Dijkstra algorithm. *IEEE Access* **2023**, *11*, 92433–92442.

55. Korkmaz, M.; Durdu, A. Comparison of optimal path planning algorithms **2018**. pp. 255–258.

56. Halder, U.; Das, S.; Maity, D. A cluster-based differential evolution algorithm with external archive for optimization in dynamic environments. *IEEE transactions on cybernetics* **2013**, *43*, 881–897.

57. Saeedi, S.; Bodin, B.; Wagstaff, H.; Nisbet, A.; Nardi, L.; Mawer, J.; Melot, N.; Palomar, O.; Vespa, E.; Spink, T.; et al. Navigating the landscape for real-time localization and mapping for robotics and virtual and augmented reality. *Proceedings of the IEEE* **2018**, *106*, 2020–2039.

58. Mohaghegh, M.; Jafarpourdavatgar, H.; Saeedinia, S.A. New design of smooth PSO-IPF navigator with kinematic constraints. *IEEE Access* **2024**.

59. Miao, C.; Chen, G.; Yan, C.; Wu, Y. Path planning optimization of indoor mobile robot based on adaptive ant colony algorithm. *Computers & Industrial Engineering* **2021**, *156*, 107230.

60. Bao, C.; Gao, D.; Gu, W.; Xu, L.; Goodman, E.D. A new adaptive decomposition-based evolutionary algorithm for multi-and many-objective optimization. *Expert Systems with Applications* **2023**, *213*, 119080.

61. Hu, Y.; Yang, X.; Zhou, K.; Liu, Q.; Ding, K.; Gao, H.; Zhu, P.; Liu, C. SwarmPRM: Probabilistic Roadmap Motion Planning for Large-Scale Swarm Robotic Systems **2024**. pp. 10222–10228.

62. Zhang, Z.; Qiao, B.; Zhao, W.; Chen, X. A predictive path planning algorithm for mobile robot in dynamic environments based on rapidly exploring random tree. *Arabian Journal for Science and Engineering* **2021**, *46*, 8223–8232.

63. Wang, J.; Chi, W.; Li, C.; Wang, C.; Meng, M.Q.H. Neural RRT*: Learning-based optimal path planning. *IEEE Transactions on Automation Science and Engineering* **2020**, *17*, 1748–1758.

64. Sun, Z.; Xia, B.; Xie, P.; Li, X.; Wang, J. NAMR-RRT: Neural Adaptive Motion Planning for Mobile Robots in Dynamic Environments. *IEEE Transactions on Automation Science and Engineering* **2025**.

65. Wang, J.; Jia, X.; Zhang, T.; Ma, N.; Meng, M.Q.H. Deep neural network enhanced sampling-based path planning in 3D space. *IEEE Transactions on Automation Science and Engineering* **2021**, *19*, 3434–3443.

66. Zhu, K.; Zhang, T. Deep reinforcement learning based mobile robot navigation: A review. *Tsinghua Science and Technology* **2021**, *26*, 674–691.

67. Han, J.R.; Thomas, H.; Zhang, J.; Rhinehart, N.; Barfoot, T.D. DR-MPC: Deep Residual Model Predictive Control for Real-World Social Navigation. *IEEE Robotics and Automation Letters* **2025**.

68. Sellers, T.; Lei, T.; Jan, G.E.; Wang, Y.; Luo, C. Multi-objective optimization robot navigation through a graph-driven PSO mechanism **2022**. pp. 66–77.

69. Zeng, J.; Qin, L.; Hu, Y.; Yin, Q.; Hu, C. Integrating a path planner and an adaptive motion controller for navigation in dynamic environments. *Applied Sciences* **2019**, *9*, 1384.

70. Hsueh, H.Y.; Toma, A.I.; Ali Jaafar, H.; Stow, E.; Murai, R.; Kelly, P.H.; Saeedi, S. Systematic comparison of path planning algorithms using PathBench. *Advanced Robotics* **2022**, *36*, 566–581.

71. Zhong, X.; Tian, J.; Hu, H.; Peng, X. Hybrid path planning based on safe A* algorithm and adaptive window approach for mobile robot in large-scale dynamic environment. *Journal of Intelligent & Robotic Systems* **2020**, *99*, 65–77.

72. Wang, J.; Swartz, C.L.; Huang, K. Deep learning-based model predictive control for real-time supply chain optimization. *Journal of Process Control* **2023**, *129*, 103049.

73. Wang, H.; Qiu, Y.; Hou, Y.; Shi, Q.; Huang, H.W.; Huang, Q.; Fukuda, T. Deep Reinforcement Learning-Based Collision-Free Navigation for Magnetic Helical Microrobots in Dynamic Environments. *IEEE Transactions on Automation Science and Engineering* **2024**.

74. Murano, A.; Perelli, G.; Rubin, S. Multi-agent path planning in known dynamic environments **2015**. pp. 218–231.

75. Nardi, L.; Stachniss, C. Uncertainty-aware path planning for navigation on road networks using augmented MDPs **2019**. pp. 5780–5786.

76. Gouda, K.C.; Thakur, R. Energy-efficient clustering and path planning for UAV-assisted D2D cellular networks. *Ad Hoc Networks* **2025**, *170*, 103757.

77. Yu, H.; Lu, W.; Han, Y.; Liu, D.; Zhang, M. Heterogeneous dimensionality reduction for efficient motion planning in high-dimensional spaces. *IEEE Access* **2020**, *8*, 42619–42632.

78. Khalilullah, K.I.; Jindai, M.; Ota, S.; Yasuda, T. Fast road detection methods on a large scale dataset for assisting robot navigation using kernel principal component analysis and deep learning **2018**. pp. 798–803.

79. Hu, K.; O'Connor, P. Learning a representation map for robot navigation using deep variational autoencoder. *arXiv preprint arXiv:1807.02401* **2018**.

80. Williams, S.; Indelman, V.; Kaess, M.; Roberts, R.; Leonard, J.J.; Dellaert, F. Concurrent filtering and smoothing: A parallel architecture for real-time navigation and full smoothing. *The International Journal of Robotics Research* **2014**, *33*, 1544–1568.

81.

82. Rastgar, F. Towards Reliable Real-Time Trajectory Optimization. PhD thesis, University of Tartu, 2024.

83. Yu, J.; Chen, N.; Liu, G.; Xu, C.; Gao, F.; Cao, Y. TOP: Trajectory Optimization via Parallel Optimization towards Constant Time Complexity **2025**. Preprint at https://arxiv.org/abs/2507.10290.

84. Dastider, A.; Ray, A. Retro-trajectory optimization: A memory-efficient warm-start approach. *IEEE Transactions on Robotics* **2023**, *39*, 1105–1117.

85. Williams, G.; et al. Model predictive path integral control: From theory to parallel computation. In Proceedings of the IEEE ICRA, 2017.

86. He, S.; Ma, Y.; Song, T.; Jiang, Y.; Wu, X. A Real-Time Spatio-Temporal Trajectory Planner for Autonomous Vehicles With Semantic Graph Optimization. *IEEE Robotics and Automation Letters* **2024**.

87. Silveira, J.; Cabral, K.; Givigi, S.; Marshall, J. Real-Time Fast Marching Tree for Mobile Robot Motion Planning in Dynamic Environments. *arXiv preprint arXiv:2502.09556* **2025**.

88. Guo, X.; Jiang, F.; Chen, Q.; Wang, Y.; Sha, K.; Chen, J. Deep learning-enhanced environment perception for autonomous driving: MDNet with CSP-DarkNet53. *Pattern Recognition* **2025**, *160*, 111174.

89. Papachristos, C.; Mascarich, F.; Khattak, S.; Dang, T.; Alexis, K. Localization uncertainty-aware autonomous exploration and mapping with aerial robots using receding horizon path-planning. *Autonomous Robots* **2019**, *43*, 2131–2161.

90. Parasuraman, R.; Cosenzo, K.A.; De Visser, E.J. Integrating automation, human factors and ethics for effective human–robot interaction in military settings. *Military Psychology* **2010**, *22*, 1–10.

91. Eder, K.; Harper, R.; Leonards, U. Towards ethical robots: A survey of existing paradigms. In Proceedings of the IEEE International Conference on Advanced Robotics, 2014, pp. 1–6.

92. Martinetti, A.; van Wingerden, J.W.; et al. Redefining safety in human-robot interaction: From physical protection to social understanding. *Frontiers in Robotics and AI* **2021**, *8*.

93. Valori, A.; Mendes, A. Validating robot safety protocols with international ISO standards: Challenges and best practices. In Proceedings of the IEEE Int. Conf. on Human-Robot Interaction (HRI), 2021.

94. Shuming, L.; Cho, Y. Safety-Aware Learning for Human-Robot Teams in Real-Time Interaction. *IEEE Transactions on Robotics* **2024**.

95. Achiam, J.; Held, D.; Tian, A.; Abbeel, P. Constrained policy optimization. In Proceedings of the International Conference on Machine Learning, 2017, pp. 22–31.

96. Ramviyas, A.; Mataric, M. Integrating physical and psychological safety in human-robot collaboration: A unified model. *Robotics and Autonomous Systems* **2010**.