A Dynamic Mission Planning Framework Combining Improved A* and Neural-OFMPC

Gözde Körpe*†, Mehmet Önder Efe*
*Department of Computer Engineering, Hacettepe University, Ankara, Türkiye
†Aselsan Inc., Ankara, Türkiye
korpegozde@gmail.com, onderefe@gmail.com

***Abstract*—This paper presents a hierarchical mission planning framework in dynamic environments. The framework combines an improved A* global path planner with a Neural Optimization-Free Model Predictive Control (Neural-OFMPC) local controller. The Improved A* algorithm generates safer and smoother reference paths by incorporating safety distance weighting and key-turning node extraction, reducing unnecessary heading changes and simplifying local tracking. At the control level, Neural-OFMPC augments an optimization-free MPC with a lightweight neural residual policy that adaptively refines conservative control actions. The neural controller enables proactive avoidance maneuvers, such as steering and acceleration, while preserving safety through predictive collision checking and a minimal safety shield. Since no online optimization is required, the proposed approach maintains low and consistent computational cost.**

Simulation results demonstrate that the proposed framework improves success rate, execution time, and navigation efficiency compared to conventional OFMPC-based approaches, particularly in complex scenarios with dynamic obstacles. These results indicate that the proposed method is well-suited for real-time mission planning on resource-constrained robotic platforms.

***Index Terms*—Hierarchical mission planning, Path Planning, Motion Planning, Improved A*, MPC, Neural Model Predictive Control, Dynamic Environments**

I. Introduction

Autonomous navigation in dynamic environments remains a fundamental challenge in robotics, requiring effective coordination between global planning and local control. Global planners such as A* efficiently compute collision-free paths for static environments, but typically produce piecewise trajectories with sharp turns and limited safety awareness, and cannot account for dynamic changes. These limitations complicate local tracking and reduce robustness when dynamic obstacles are present. On the other hand, Model Predictive Control (MPC) provides strong local adaptability and constraint handling but incurs significant computational overhead, limiting its applicability in real-time systems.

To address these challenges, hierarchical planning frameworks and learning-based control strategies have been explored. However, many learning-based MPC approaches either approximate full optimization or still require solving computationally expensive problems online, which can undermine real-time performance and reliability.

This paper proposes a mission planning framework that integrates an Improved A* global planner with a Neural Optimization-Free Model Predictive Control (Neural-OFMPC) local controller. Here, the term *mission* refers specifically to safe start-to-goal navigation in dynamic environments, where successful mission completion is defined by reaching the goal without collision under real-time constraints.

The Improved A* planner improves path quality through safety-aware cost shaping and key-turning node extraction, resulting in smoother and safer reference trajectories. At the control level, an optimization-free MPC predicts future collisions using short-horizon forward simulation and closed-form control laws, while a lightweight neural residual policy refines conservative control actions to enable proactive avoidance maneuvers under a minimal safety verification mechanism. Importantly, the proposed approach does not aim to approximate or replace MPC optimization using learning. Instead, a deterministic and safety-aware optimization-free predictive controller is deliberately retained as the core, while learning is used only to relax conservative behaviors when beneficial. This design prioritizes predictability, safety, and real-time performance over end-to-end optimality.

The main contributions of this work are summarized as follows:

- A Neural-OFMPC controller that combines predictive collision checking with learned residual control, without requiring online optimization.
- A hierarchical mission planning framework that integrates an improved A* global planner with a Neural Optimization-Free MPC local controller for navigation in dynamic environments.
- A comprehensive simulation-based evaluation demonstrating improved efficiency, robustness, and real-time suitability in dynamic environments.

The remainder of the paper presents the proposed methodology and evaluates its performance through simulation.

II. Related Work

Autonomous mission planning generally relies on a two-layer architecture, with most approaches adopting a hierarchical structure that separates planning from motion control. This section reviews related studies on path planning, motion control methods, learning-augmented control, and hierarchical integration frameworks, and positions the proposed approach within this literature.

A. Global Path Planning Algorithms

Global path planning methods aim to generate collision-free reference paths while balancing optimality and safety. Classical and sampling-based planners, including A*, Dijkstra, and Rapidly-exploring Random Tree (RRT) variants have been extensively studied. A comprehensive overview of these approaches is provided in the systematic review by Prasad *et al.* [1]. Among these, A* provides a balance between completeness and computational efficiency by combining cost-to-come and heuristic estimates.

Significant effort has been devoted to improving A* through safety-aware cost shaping, heuristic adaptation, and hybrid integrations. A comprehensive review of improved A* variants and their hybrid extensions is presented by Doan *et al.* [2], demonstrating that safety distance modeling and path refinement techniques can substantially improve path smoothness and clearance. Among these, Zhai *et al.* [3] proposed a real-time A* variant based on a Safety Distance Matrix and adaptive weighting, generating smoother and safer paths. This formulation serves as the basis for the improved A* planner employed in this work.

B. MPC-Based Local Motion Control and Learning-Based MPC

Following global path generation, a local controller is responsible for trajectory tracking while respecting system dynamics and constraints. Model Predictive Control (MPC) has become a dominant approach for local motion planning due to its ability to handle nonlinear dynamics, constraints, and prediction over a finite horizon. A detailed overview of recent developments and future directions in MPC is provided by Mayne [4], who highlights both the strengths of MPC and the computational challenges associated with online optimization.

To address these limitations, learning-based approaches have been explored to enhance MPC performance or reduce computational burden, as reviewed in [5]. Soloperto *et al.* [6] demonstrated learning-based MPC for iterative tasks, showing how data-driven components can enhance control performance.

More recently, neural network approximations of MPC have been explored for agile and high-speed robotic systems. Salzmann *et al.* [7] proposed a real-time neural MPC framework for quadrotors, where deep learning is used to approximate the MPC policy directly. While these approaches demonstrate impressive reactivity, they typically focus on local control and rely on learned approximations of optimization-based MPC formulations.

Unlike these approaches, the proposed Neural-OFMPC does not approximate the MPC optimizer or value function, but instead learns a residual policy that refines an optimization-free predictive controller while preserving deterministic safety checks.

C. Hierarchical Planning and Framework Integration

Hierarchical frameworks combining global planning and MPC-based local control are widely adopted for autonomous navigation. Katrakazas *et al.* [8] provide an overview, emphasizing the importance of combining global planning with reactive local control for robust operation in dynamic environments. A recent example is the Hybrid A*–MPC framework proposed by Lu *et al.* [9], which integrates planning and optimization-based control for structured environments, where the primary control objective is accurate reference trajectory tracking.

In contrast, while the present work also utilizes a reference path for guidance, it prioritizes safe and efficient navigation in dynamic environments, allowing deliberate deviations from the path when required for obstacle avoidance and computational efficiency.

III. Improved A* Algorithm

The traditional A* algorithm plans an optimal path by minimizing the cost function $F(n) = G(n) + H(n)$, where $G(n)$ is the actual cost from the start node to the current node, and $H(n)$ is the heuristic estimate from the current node to the goal. Although effective for global path planning, the standard A* algorithm often produces redundant nodes, sharp turns, and paths that are too close to obstacles, which can degrade safety and tracking performance.

To address these issues, *Zhai et al. (2024)* proposed an Improved A* algorithm that integrates two main strategies:
(1) Safety Distance Matrix (SDM) to enhance the safety of the generated path by biasing the search away from obstacles and,
(2) Key Turning Node Extraction Strategy to eliminate redundant nodes and ensure smooth, dynamically feasible motion. The method is summarized here for completeness and integrated into the proposed framework for subsequent control and learning-based evaluation.

*1) **Safety Distance Matrix**:* To improve safety, a Safety Distance Matrix (SDM) K_n is defined for each candidate node, representing a local rectangular region centered on the node. Unlike obstacle expansion methods that modify the global map, the SDM influences path selection by directly augmenting the cost function:

$$F'(n) = G(n) + H(n) + aS(n), \tag{1}$$

where $S(n)$ is the safety cost from the SDM and a is its weighting coefficient. If obstacles are detected within the "outer" positions of K_n, $S(n)$ is set to 1, increasing the node cost and encouraging paths with greater clearance. The SDM dimension is given by

$$d = [2(L \times b) + 1], \tag{2}$$

where L is the safety distance, and b is the map resolution. This formulation implicitly enforces safety margins without altering the global map structure.

*2) **Key Turning Node Extraction Strategy**:* The key turning node extraction strategy removes redundant intermediate nodes while preserving essential turning points. Given the path $U = \{P_i \mid 1 \leq i \leq n\}$ from the original A* search, the area method is applied to three consecutive nodes (P_{i-1}, P_i, P_{i+1}).

If the area of the triangle they form is zero, P_i is redundant; otherwise, it is a turning node. For each turning node P_k, the algorithm checks if the straight line between P_{k-1} and P_{k+1} intersects any obstacle. If it does, P_k is labeled as a *key turning node*. This method reduces unnecessary stops and turns, resulting in a smoother path that better satisfies nonholonomic motion constraints.

3) Performance Comparison: As reported by [3] and shown in Fig.1, the improved A* algorithm achieves smoother paths with fewer sharp turns and increased obstacle clearance compared to the standard version. This qualitative improvement simplifies path tracking for Neural-OFMPC.

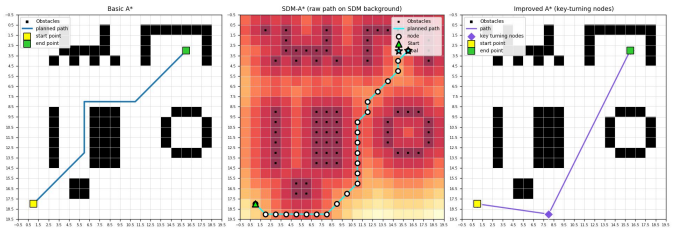

Fig. 1: Comparison of paths (a) Basic A* (b) SDM-A* (c) Improved A*.

IV. Optimization-Free Model Predictive Control (OFMPC)

Classical Model Predictive Control (MPC) computes control inputs by repeatedly solving a finite-horizon optimal control problem in a receding-horizon manner, where only the first control input is applied. While effective, this formulation requires online numerical optimization, which limits real-time applicability in dynamic environments.

A. OFMPC Formulation

In this work, we propose an Optimization-Free Model Predictive Control (OFMPC) formulation that preserves the predictive, receding-horizon structure of classical MPC while eliminating online optimization. OFMPC forward-simulates the robot and surrounding obstacles over a finite horizon and applies analytical control laws based on predicted safety and feasibility. Although no optimization problem is solved at runtime, it retains key characteristics of Model Predictive Control, including explicit system modeling, receding-horizon prediction, constraint awareness, and online feedback. For this reason, it is intentionally framed within the MPC family, despite its optimization-free implementation.

The robot is modeled using a unicycle kinematic model. Dynamic obstacles are assumed to follow constant-velocity motion. A pure-pursuit path-following strategy is used to compute nominal linear and angular velocities based on heading error to a look-ahead point on the reference path.

B. Predictive Collision Assessment

At each control step, the robot and surrounding dynamic obstacles are forward-simulated over a short horizon $a \in \{1, 2, 3\}$, for rapid collision detection, while a longer 10-step horizon is used in OFMPC for trajectory evaluation. This horizon is sufficient given obstacle speeds of 0.2–0.5 m/s and the robot's stopping distance at maximum velocity. The minimum predicted separation distance is

$$d_{\min} = \min_{a} \left\| \mathbf{p}_{k+a}^{(r)} - \mathbf{p}_{k+a}^{(o)} \right\|. \tag{3}$$

A collision risk is predicted if

$$d_{\min} \leq d_{\text{safe}} + \delta, \tag{4}$$

where δ is a safe-zone margin and

$$d_{\text{safe}} = r_r + r_o + v_k t_r + \frac{v_k^2}{2a_{\text{emg}}}, \tag{5}$$

where r_r and r_o denote the robot and obstacle radii, t_r is a reaction-time constant, and a_{emg} is an emergency deceleration bound.

a) Time-to-Closest-Approach (TCA): In addition to distance-based evaluation, OFMPC estimates the time-to-closest-approach (TCA) between the robot and each dynamic obstacle:

$$t_{\text{ca}} = -\frac{(\mathbf{p}_o - \mathbf{p}_r)^\top (\mathbf{v}_o - \mathbf{v}_r)}{\|\mathbf{v}_o - \mathbf{v}_r\|^2}, \tag{6}$$

where $\mathbf{p}_r, \mathbf{v}_r$ and $\mathbf{p}_o, \mathbf{v}_o$ denote the positions and velocities of the robot and obstacle, respectively. The TCA provides a time-aware measure of collision imminence and complements the distance-based safety assessment.

C. Optimization-Free Velocity Modulation

OFMPC applies a closed-form, clearance-aware velocity modulation policy. Given the nominal forward velocity v_{nom}, the commanded velocity is

$$v_k = \alpha(d_{\min}) \cdot v_{\text{nom}}, \tag{7}$$

where $\alpha(\cdot)$ is a bounded scaling function that reduces speed as predicted clearance decreases.

In practice, α is implemented as a piecewise, time-aware control law based on the predicted minimum distance and time-to-closest-approach, enabling early collision avoidance without online optimization. Dynamic feasibility is enforced through acceleration limits:

$$|v_k - v_{k-1}| \leq a_{\max} \Delta t. \tag{8}$$

Additional safety overrides enforce full stops in cases of imminent collision or blocked corridors.

D. A*-OFMPC vs. Improved A*-OFMPC

The next stage integrates the A* and Improved A* planners with OFMPC to evaluate tracking performance and control efficiency before introducing the Neural-OFMPC extension.

Table I shows that the Improved A*–OFMPC significantly outperforms the baseline A*–OFMPC. The success rate increases from 77.0% to 93.0%, while all failures in both methods are due to timeouts, i.e., reaching the maximum allowed number of control steps (12,000). Higher average speed and substantially lower tracking error further indicate that high-quality global paths simplify and accelerate the local control.

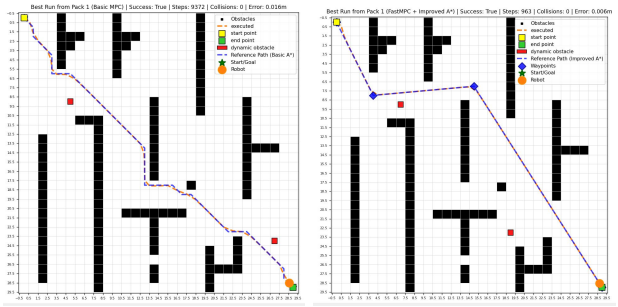

Fig. 2: The simulation results of scenario 1. (a) A*-OFMPC and (b) Improved A*-OFMPC.

TABLE I: Comparison of A*-OFMPC and Improved A*-OFMPC in Fig 2 over 100 simulation runs.

Metric	A*-OFMPC	Improved A*-OFMPC
Success rate (%)	77.0	**93.0**
Avg. steps	7224.8	**7112.7**
Avg. planning time (s)	0.0075	0.0587
Avg. OFMPC time (s)	5.6688	**3.0207**
Avg. speed (m/s)	0.381	0.442
Avg. tracking error (m)	0.065	**0.019**

V. Neural-OFMPC Controller

A. Overview

The proposed *Neural-OFMPC* augments the Optimization-Free MPC (OFMPC) by introducing a learned residual control policy that refines conservative control actions, while preserving the safety guarantees of the base controller. Instead of replacing OFMPC with an end-to-end neural policy, the neural controller augments its output, enabling more aggressive yet safe behaviors when it is beneficial, such as accelerating to avoid dynamic obstacles or executing smooth maneuvers.

At each control step, OFMPC computes a nominal control command based on predictive collision assessment and closed-form control laws. A neural network then generates residual corrections, which are filtered by a deterministic safety shield to ensure feasibility and collision avoidance.

B. Control Formulation

The Neural-OFMPC follows a residual control formulation:

$$\mathbf{u}_k = \mathbf{u}_k^{\text{of}} + \Delta\mathbf{u}_k, \tag{9}$$

where $\mathbf{u}_k = [v_k, \omega_k]^\top$, $\mathbf{u}_k^{\text{of}} = [v_{\text{of}}, \omega_{\text{of}}]^\top$ denotes the OFMPC output, and $\Delta\mathbf{u}_k = [\Delta v_k, \Delta\omega_k]^\top$ is the neural residual. To preserve interpretability and boundedness, the residual is parameterized as

$$v_k = \beta_v \cdot v_{\text{of}}, \qquad \omega_k = \omega_{\text{of}} + \Delta\omega, \tag{10}$$

where β_v is a learned speed scaling factor and $\Delta\omega$ is a learned steering correction.

This formulation allows the neural policy to refine conservative OFMPC decisions without overriding the controller, enabling more responsive speed and steering adjustments while avoiding destabilizing actions.

C. Neural Policy Architecture

The neural policy receives a compact 15-dimensional feature vector that encodes normalized predictive safety metrics (collision flags, time-to-collision, clearance, relative obstacle motion), tracking errors (heading and lateral deviation), motion states (current linear and angular velocity), task progress (goal distance and proximity), obstacle bearing, and a binary episode outcome indicator. These features are selected to capture predictive safety, tracking error, and dynamic interaction information necessary for effective residual control. All features are derived from quantities already computed within the OFMPC pipeline.

The residual policy is implemented as a multilayer perceptron (MLP):

$$(\beta_v, \Delta\omega) = f_\theta(\mathbf{s}), \tag{11}$$

where $\mathbf{s} \in \mathbb{R}^{15}$ denotes the extracted feature vector. The network consists of three fully connected hidden layers with ReLU activations and a Tanh output layer,

$$\mathbb{R}^{15} \xrightarrow{\text{Linear}} 64 \xrightarrow{\text{ReLU}} 64 \xrightarrow{\text{ReLU}} 32 \xrightarrow{\text{ReLU}} 2 \xrightarrow{\text{Tanh}} (\beta_v, \Delta\omega) \tag{12}$$

D. Safety Shielding

All neural outputs are filtered through a deterministic safety shield by the admissible control set:

$$\mathcal{U}_{\text{safe}}(\mathbf{x}_k) = \left\{ \mathbf{u} \;\middle|\; \begin{array}{l} d_{\min}(\mathbf{x}_k, \mathbf{u}) \geq d_{\text{safe}}, \\ v_{\min} \leq v \leq v_{\max}, \\ |\omega| \leq \omega_{\max} \end{array} \right\}, \tag{13}$$

where $d_{\min}$ denotes the predicted minimum distance to obstacles over a short forward simulation horizon.

The final command is obtained via:

$$\mathbf{u}_k^{\text{safe}} = \Pi_{\mathcal{U}_{\text{safe}}}\left(\mathbf{u}_k^{\text{of}} + \Delta\mathbf{u}_k\right), \tag{14}$$

where $\Pi_{\mathcal{U}_{\text{safe}}}(\cdot)$ denotes a projection or replacement operator that enforces feasibility. If the neural correction violates safety constraints, the control command rejected in favor of the nominal OFMPC action. The safety shield operates under standard assumptions on state estimation,obstacle motion, and perception reliability. Under these modeled conditions, the safety shield provides robust collision avoidance in practice; however, it does not constitute a formal global safety guarantee.

VI. Training & Data Collection for Neural-OFMPC

A. Hybrid Expert Data Generation

The neural policy is trained via supervised learning from a hybrid expert that combines the conservative OFMPC controller with a search-based pseudo-expert. This hybrid expert exposes the neural policy to actions(e.g. avoidance maneuvers) that are safe but more aggressive than those executed by OFMPC alone. The training dataset consists of multiple simulated navigation scenarios with varying initial conditions,

obstacle configurations, and reference paths generated by the Improved A* planner.

At each control step, OFMPC provides a baseline command $\mathbf{u}_{\text{base}} = (v_{\text{base}}, \omega_{\text{base}})$. A finite set of candidate actions is generated by locally perturbing this baseline command:

$$\mathcal{U}_{\text{cand}} = \{(v_{\text{base}}\alpha_i,\ \omega_{\text{base}} + \Delta\omega_j)\}\,, \tag{15}$$

where α_i and $\Delta\omega_j$ are sampled from predefined discrete sets.

Each candidate action is evaluated through short-horizon forward simulation using the unicycle kinematic model over a short horizon(2 steps) and unsafe candidates are discarded using multi-step collision checking, ensuring that only collision-free trajectories are considered. Among the remaining safe candidates, the action with the lowest accumulated cost accounting for tracking error, obstacle clearance, smoothness, and forward progress is selected as the expert label.

B. Learning Objective and Loss Formulation

Training samples consist of state–target pairs $(\mathbf{s}_k, \mathbf{y}_k)$, where $\mathbf{s}_k$ denotes the extracted feature vector and

$$\mathbf{y}_k = [\beta_v^*,\ \Delta\omega^*]\,, \tag{16}$$

represents the desired velocity scaling and angular correction. The neural policy $f_\theta(\cdot)$ is trained using a weighted Huber loss [10]:

$$\mathcal{L} = \mathbb{E}[w_k \cdot \text{Huber}(f_\theta(\mathbf{s}_k) - \mathbf{y}_k)]\,, \tag{17}$$

where w_k denotes a per-sample importance weight that emphasize collision-critical states and successful avoidance maneuvers. The Huber loss is chosen for its robustness to outliers and suitability for multi-modal avoidance behavior. It prevents over-penalization of alternative but safe control actions. The network was trained for 50 episodes using the Adam optimizer (lr = 0.001) with batch size 64, and 20 epochs, using an 80/20 train/validation split.

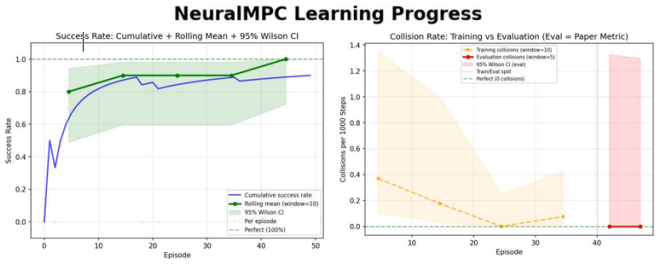

Fig. 3: (a) Cumulative success rate and (b) collision rate of the learning process.

As shown in Fig. 3, the cumulative success rate increases rapidly while the collision rate decreases to zero, indicating effective learning of safe avoidance behaviors. Also, training and validation losses decreased consistently and remained closely aligned, indicating stable learning without overfitting.

VII. Proposed Mission Planning Framework

The proposed mission planning framework integrates an improved A* global planner with a Neural-OFMPC local controller in a hierarchical architecture. The global planner computes a safe and efficient reference path, while the local controller executes this path in real time under dynamic environmental conditions. At initialization, the Improved A* algorithm generates a global path from the start to the goal using SDM and KTN extraction, producing a smoothed waypoint sequence that serves as the reference trajectory. During execution, Neural-OFMPC tracks the reference path while proactively avoiding static and dynamic obstacles using predictive collision assessment and learned residual corrections. A deterministic safety shield vetoes unsafe neural commands in rare cases of imminent collision, preserving both safety and reactivity.

VIII. Experimental Setup

All simulations and training experiments were performed on an Intel Core i9 system with 16 GB RAM.

Experiments were conducted in a 2D planar grid-based environment with 0.2, m resolution, containing static obstacles and randomly sampled start–goal pairs. Perfect state information of obstacle positions and velocities is assumed.

Dynamic obstacles follow linear or smooth random walks with speeds in 0.2–0.5, m/s. The robot follows a unicycle model with linear and angular velocity limits of 1.0, m/s and $20°/$s. Safety parameters include a robot radius of 0.12 m ,an early-warning guard zone of 0.5 m and the safe-zone margin δ = 0.2 m.

IX. Results and Discussion

In order to verify and highlight the improvements in control efficiency and real-time performance of the proposed **Improved A*–Neural-OFMPC** framework, simulation comparison experiments are conducted for the **Improved A*–OFMPC** framework.

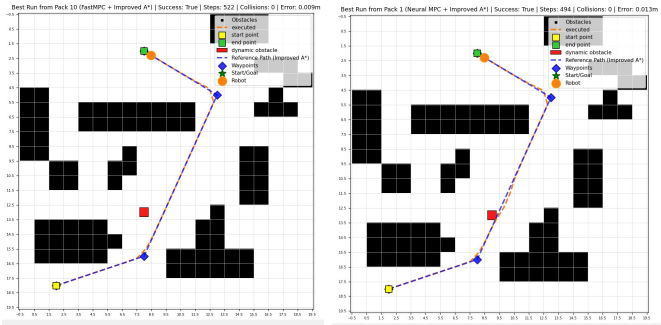

Fig. 4: The simulation results of scenario 2. (a)Improved A*–OFMPC and (b) Proposed Method.

From the executed path of the robot, it is observed that while improved A*-OFMPC reacts conservatively to dynamic obstacles by decelerating, the proposed improved A*-Neural-OFMPC makes avoidance maneuvers to avoid collisions instead of slowing down and waiting for the obstacle to pass.

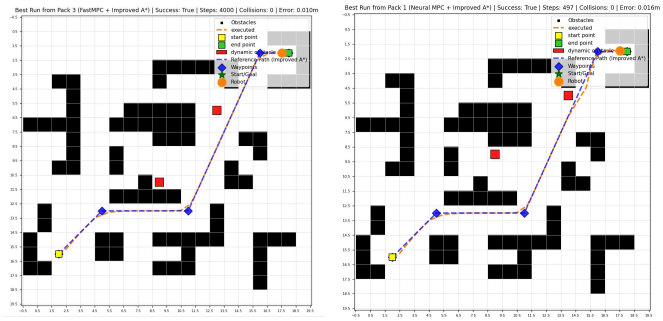

Fig. 5: The simulation results of scenario 3. (a) Improved A*–OFMPC and (b) Proposed Method.

This behaviour indicates that learned policy captures effective avoidance patterns beyond conservative strategies.

TABLE II: Performance comparison over two scenarios (100 runs each).

Scn.	Metric	Imp. A*–OFMPC	Proposed
S2	Success rate (%)	99.0	**100.0**
	Avg. steps	1716.4	**526.6**
	Avg. controller time (s)	0.7851	**0.4819**
	Avg. speed (m/s)	0.315	**0.462**
	Avg. tracking error (m)	0.012	**0.05**
S3	Success rate (%)	86.0	**97.0**
	Avg. steps	6680.9	**846.9**
	Avg. controller time (s)	4.6056	**1.3658**
	Avg. speed (m/s)	0.363	**0.488**
	Avg. tracking error (m)	0.019	**0.043**

Table II compares the Improved A*–OFMPC and the proposed Improved A*–Neural-OFMPC across two scenarios of increasing difficulty.

In Scenario S2, both methods achieve near-perfect success; however, the proposed method reduces the average number of control steps by approximately 69% and decreases the average controller execution time by about 39%. The average robot speed increases by roughly 47%, indicating less conservative motion execution. A moderate increase in tracking error is observed , which is a deliberate trade-off, results from intentional deviations from the reference path during dynamic obstacle avoidance and enables faster task completion without compromising safety.

In the more challenging Scenario S3, the benefits of the proposed method become more pronounced. The success rate improves from 86.0% to 97.0%, corresponding to an absolute increase of 11 percentage points. The average number of control steps is reduced by approximately 87%, while the average controller execution time decreases by about 70%. The average robot speed increases by approximately 34%, reflecting the ability of the proposed method to avoid unnecessary deceleration in dense dynamic environments.

Overall, the performance gap between the two methods is modest in simpler scenarios, where conservative control strategies are often sufficient. In contrast, in complex environments with dense dynamic obstacles, the proposed Neural-OFMPC framework provides substantial gains in success rate, computational efficiency, and task completion speed. These results highlight the effectiveness of learning-based residual control in improving decision-making under challenging navigation conditions.

X. Conclusion

This paper proposed a hierarchical mission planning framework that combines an improved A* global planner with a Neural-OFMPC local controller for navigation in dynamic environments. The Improved A* planner generates safer and smoother reference paths using safety-aware cost shaping and key-turning node extraction, reducing the burden on the local controller. The Neural-OFMPC augments an optimization-free MPC with a lightweight neural residual policy that refines control actions while preserving safety through a minimal safety shield.

Simulation results show that the proposed method achieves higher success rates, fewer execution steps, and faster mission completion compared to OFMPC-based baselines, particularly in complex scenarios with dynamic obstacles. Since the controller avoids online optimization and relies on simple forward prediction and neural inference, the computational load remains low and consistent. This makes the proposed framework well-suited for real-time deployment on resource-constrained robotic platforms.

Future work will consider real-world experiments and extensions to more complex environments and multi-agent scenarios.

References

[1] A. Prasad, K. Chaudhary, B. Sharma, and A. S. Ali, "Robot path planning and motion control: A systematic review," *Engineering Science Publisher*, 2023. Just Accepted.

[2] D. T. Xuan, N. T. Hung, and V. T. Thang, "A comprehensive review of improved a* path planning algorithms and their hybrid integrations," *Robotics*, vol. 6, no. 4, 2024.

[3] X. Zhai, J. Tian, and J. Li, "A real-time path planning algorithm for mobile robots based on safety distance matrix and adaptive weight adjustment strategy," *International Journal of Control, Automation, and Systems*, vol. 22, no. 4, pp. 1385–1399, 2024.

[4] D. Q. Mayne, "Model predictive control: Recent developments and future promise," *Automatica*, vol. 50, no. 12, pp. 2967–2986, 2014.

[5] L. Hewing, K. Wabersich, and M. Zeilinger, "Learning-based model predictive control: Toward safe learning in control," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, pp. 269–296, 2020.

[6] R. Soloperto, J. Köhler, and M. Zanon, "Learning model predictive control for iterative tasks: A data-driven control framework," in *European Control Conference (ECC)*, pp. 2470–2476, 2019.

[7] T. Salzmann, E. Kaufmann, J. Arrizabalaga, M. Pavone, D. Scaramuzza, and M. Ryll, "Real-time neural mpc: Deep learning model predictive control for quadrotors and agile robotic platforms," *IEEE Robotics and Automation Letters*, 2023. Preprint available at arXiv:2203.07747.

[8] C. Katrakazas, M. Quddus, W. Chen, and L. Deka, "Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions," *Transportation Research Part C*, vol. 60, pp. 416–442, 2015.

[9] M. Lu, H. Gao, H. Dai, Q. Lei, and C. Liu, "Path-tracking hybrid a* and hierarchical mpc framework for autonomous agricultural vehicles," *arXiv preprint*, 2024.

[10] P. J. Huber, "Robust estimation of a location parameter," *The Annals of Mathematical Statistics*, vol. 35, no. 1, pp. 73–101, 1964.