Path Planning Transformer Supervised by Improved RRT* with Reduced Random Map Size for Mobile Robots

Aphilak Lonklang and János Botzheim

***Abstract*—The Improved Rapidly-exploring Random Tree with Reduced Random Map Size (IRRT*-RRMS) algorithm was previ- ously developed to find collision-free paths for mobile robot path planning. Given the excellent performance of Transformer Neural Networks with sequential data, we propose an encoder-decoder transformer model combining a Vision Transformer (ViT) as the encoder and a time-series forecasting module as the decoder to learn the path planning algorithm. The novelty of this paper lies in developing a model supervised by a dataset generated from the IRRT*-RRMS algorithm and using this trained model for the path planning task. The trained model efficiently predicts intermediate points between the desired starting and goal points. The performance was validated on a real robot, demonstrating that the trained model required less computation time compared to the IRRT*-RRMS algorithm.**

***Index Terms*—Path Planning Transformers, Mobile Robots, RRT*.**

I. INTRODUCTION

Material distribution is a key operation in modern manufacturing and logistics, where autonomous mobile robots (AMRs) and automated guided vehicles (AGVs) are deployed to improve efficiency, reduce human effort, and optimize resource use [1]. Path-planning algorithms enable these robots to navigate safely through complex environments involving static and dynamic obstacles. Classical methods such as A* and Dijkstra [2], [3] generate collision-free paths but rely on grid-based maps, which can be computationally and memory-intensive in large-scale environments [4], [5].

Sampling-based algorithms like Rapidly-exploring Random Trees (RRT) and RRT* [6], [7] address scalability by incrementally building space-filling trees. However, they often suffer from slow convergence and require numerous random samples. Informed RRT* [8] improves efficiency by focusing sampling in promising regions. Despite improvements, these methods remain costly in dynamic or large environments.

RRT, introduced by LaValle et al. [9], is widely used due to its simplicity and effectiveness in high-dimensional spaces [10]. It explores the environment via random sampling and builds a tree from the start to the goal. RRT* enhances this by optimizing path length [11], [12], but it still suffers from redundant sampling and inconsistent path quality [13].

Department of Artificial Intelligence, Faculty of Informatics, ELTE Eötvös Loránd University, Budapest, Hungary (E-mail: {aphilak, botzheim}@inf.elte.hu)

To improve efficiency, the Improved RRT* with Reduced Random Map Size (IRRT*-RRMS) [14] was proposed. It removes nodes in obstacle regions from the sampling space and excludes reused nodes in future iterations, preventing redundancy. This modification accelerates tree exploration and convergence. IRRT*-RRMS is especially suitable for dynamic environments with unknown static obstacles [15].

As environments become complex, traditional planners encounter limitations in speed and generalization [16], [17]. To overcome this, machine learning approaches—particularly neural networks—have been proposed [18]–[20]. CNN-based planners can encode spatial data but are often constrained by fixed-size maps. Recently, transformer-based models like the Motion Planning Transformer (MPT) [4] have shown promising results by learning to predict feasible paths in complex, high-dimensional spaces.

Transformers [21], originally designed for NLP, leverage attention mechanisms to model long-range dependencies. They have been successfully applied to tasks like image classification [22], visual inspection [23], and motion planning [24]. Variants have also shown success in time-series forecasting [25], [26] across diverse domains including traffic [27] and agriculture [28].

Although transformer-based planners like MPT offer enhanced flexibility and planning efficiency, they still face challenges such as generalization and dependency on large training datasets. To address this, we propose the Path Planning Transformer (PPT), which leverages IRRT*-RRMS as a data generator. This allows for high-quality, structured datasets with fewer noisy waypoints. Furthermore, post-processing techniques like Bacterial Mutation and Node Deletion are integrated to refine the predicted paths.

Classical sampling-based planners such as RRT* and the proposed IRRT*-RRMS algorithm are deterministic methods that explicitly explore the configuration space to guarantee collision-free solutions when feasible paths exist. However, such algorithms often require extensive sampling and iterative computation, which may lead to high planning times in complex environments. In contrast, learning-based approaches such as transformer neural networks operate as stochastic predictive models that approximate feasible paths based on patterns learned from training data. While these models can generate solutions significantly faster during inference, they may introduce uncertainty and occasionally produce infea-

DOI: 10.36244/ICJ.2026.1.11

sible paths. To leverage the advantages of both paradigms, the proposed framework uses the deterministic IRRT*-RRMS algorithm to generate high-quality training data, while the Path Planning Transformer learns to predict intermediate waypoints efficiently. Post-processing techniques such as Bacterial Mutation and Node Deletion are then applied to improve path feasibility and robustness.

In summary, our approach combines the strengths of classical and learning-based methods to deliver a robust, real-time path-planning framework for mobile robots, improving accuracy, generalization, and computational efficiency.

II. Problem Definition

In mobile robot navigation systems, a global path planner is typically used to compute an initial collision-free path between the starting location and the goal position using a global occupancy map. Classical sampling-based algorithms such as RRT* provide reliable solutions but often require significant computational effort due to repeated random sampling and iterative optimization. The objective of this work is to investigate whether a transformer-based neural network can learn the path generation behavior of the IRRT*-RRMS algorithm and generate feasible global paths more efficiently. The proposed Path Planning Transformer therefore focuses on accelerating the global path planning stage by predicting intermediate waypoints from map representations and start–goal configurations.

III. Improved RRT* with Reduced Random Map Size

A. Environment and Mapping

The environment is modeled as an $M \times N$ matrix, where each cell contains a unique index and its 2D Cartesian coordinates. As shown in Fig. 1, white cells indicate feasible regions, while red cells denote obstacles.

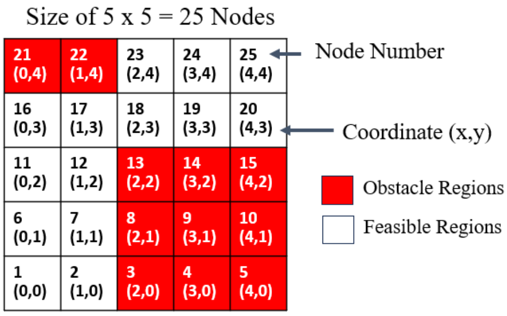


Fig. 1: Environment Mapping

B. Traditional RRT* and Limitations

Traditional RRT* incrementally builds a tree by sampling random nodes (q_{rand}), steering toward them from the nearest node (q_{near}), and optimizing via nearby nodes (q_{min}) within radius R [9], [11]. However, it retains unusable nodes (in red) [13], increasing computational cost in complex environments (Fig. 2).

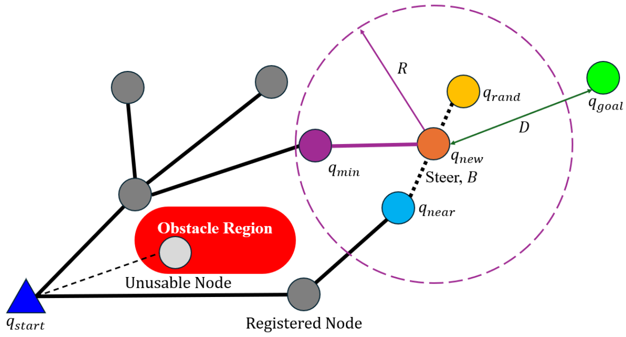


Fig. 2: Traditional RRT* Illustration

C. Feasible Region Mapping

To overcome this, the global map is flattened into a 1D vector ($randMap$), and obstacle nodes are removed [14]. Fig. 3 illustrates this pruning, which retains only feasible nodes for sampling.

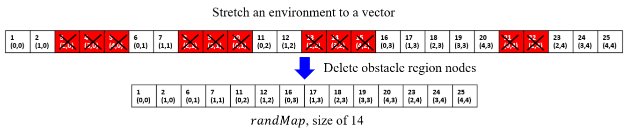
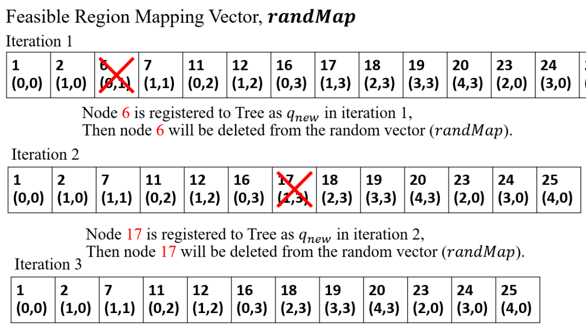


Fig. 3: Feasible Region Mapping

D. IRRT* with Reduced Random Map Size

IRRT*-RRMS [15], [29] further improves sampling by removing each used node from $randMap$ after registration, ensuring no repetitions and more efficient exploration. This is shown in Fig. 4, where node "6" is removed after iteration.



"Fig. 4: Reduced Random Map Size Technique"

The pseudocode in Alg. 1 summarizes the full algorithm, including pre-processing and tree construction steps.

E. Post-processing: Bacterial Mutation and Node Deletion

To refine paths, we apply two post-processing steps:

- **Bacterial Mutation:** introduces small variations to improve smoothness and collision avoidance.

Algorithm 1 Improved RRT* with Reduced Random Map Size Algorithm

Map = **ReadMap** from file (.bmp)
$randMap$ = **StretchMap** from matrix to row vector
for $i <$ Length($randMap$) **do**
  if $randMap(i)$ is an obstacle region **then**
    Delete $randMap(i)$ from $randMap$ vector
  end if
end for
Initialize q_{start} and q_{goal}
for $i < MaxIteration$ **do**
  $q_{rand} \leftarrow$ random node $randMap$
  $q_{near} \leftarrow$ find nearest node from Tree
  if obstacle free between q_{near} and q_{new} **then**
    $q_{new} \leftarrow$ steer from q_{near}
    Find minimum cost from q_{min} and q_{new} in radius of R
    Add q_{new} to Tree
    Remove q_{new} from $randMap$
  end if
  if distance between q_{new} and $q_{goal} \leq D$ **then**
    Stop iteration
  end if
end for
Return Tree and Define Path as a Bacterium
for $i <$ size of bacterium **do**
  Bacterial Mutation
end for
Return Fine-tuned Bacterium
for $i <$ size of bacterium **do**
  Node Deletion
end for
Return Final Path
End

- **Node Deletion:** removes redundant nodes to reduce path length and complexity.

The cost function (Eq. (1)) evaluates path fitness using length L, Penalty for collisions, Turns, and smoothness factor S.

$$Cost(path) = L(path) + Penalty \cdot Coll(path) + S \cdot Turn(path) \quad (1)$$

Parameter settings are summarized in Table I. These were tuned for a 480×480 map with max iterations of 600, steering distance $B = 45$, and optimized radius $R = 80$.

F. Application and Integration

We validated IRRT*-RRMS using a real robot in unknown static obstacle scenarios via ROS and MATLAB. Fig. 5 shows the result: the magenta path is the raw IRRT*-RRMS output, and the red path is the final result after post-processing. The path is smooth, collision-free, and deployable in real-time scenarios.

TABLE I
PARAMETER SETTINGS

Algorithm	Parameter	Value
RRT*	Max Iteration	600
	Map Size	480×480
	Steering Distance (B)	45
	Radius (R)	80
Bacterial Mutation	N_{clone}	20
	Generations	Size of Bacterium
	$Penalty$	10,000
	S	0.1
Node Deletion	Iteration	Size of Bacterium

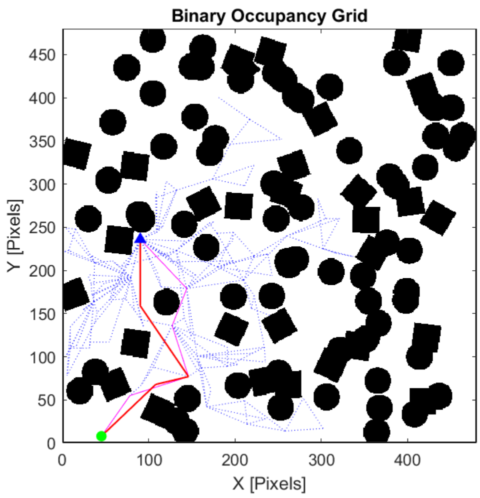


Fig. 5: Final path result after IRRT*-RRMS and post-processing from the start point (blue triangle) to the goal point (green circle)

In summary, IRRT*-RRMS reduces computational complexity by limiting the sampling space while maintaining high-quality path outputs. These paths are used to train the Path Planning Transformer, enabling learning from structured, optimized examples.

IV. TRANSFORMER NEURAL NETWORKS FOR MOBILE ROBOT PATH PLANNING

A. Transformer Architectures and Variants

The Transformer Neural Network, introduced by Vaswani et al. [21], revolutionized sequence modeling through its self-attention mechanism, replacing RNNs and CNNs in many NLP tasks. Its encoder-decoder structure allows efficient parallelization and long-range dependency modeling, forming the backbone of models like BERT and GPT.

Due to its scalability and modular design, the Transformer has been widely adopted beyond NLP, including vision and time-series domains. The Vision Transformer (ViT) [30] treats images as sequences of patches, applying attention mechanisms to capture spatial features, achieving competitive results in classification and segmentation. Similarly, Time Series Transformer variants extend this architecture to model temporal dependencies for forecasting applications, although a canonical form has not yet been standardized.

B. Motion Planning Transformer (MPT)

Transformers have emerged as a powerful tool in robotics, enabling efficient sensor data processing and spatial understanding for tasks such as navigation and obstacle avoidance. The Motion Planning Transformer (MPT) [24] introduces a data-driven approach to reduce the path planning search space by predicting feasible regions from environmental data. This reduces planning time and improves generalization across diverse environments.

MPT combines Transformer-based predictions with sampling-based motion planning (SMP), enhancing planning speed and accuracy. As shown in Fig. 6, the model outputs "Region Proposals" (green areas) for planning between start (blue) and goal (green) points. The resulting path (red) is computed based on these feasible regions.

In contrast, our approach uses IRRT*-RRMS to generate training data and feasible paths. The Transformer model can generate solutions directly, with optional refinement through post-processing algorithms.

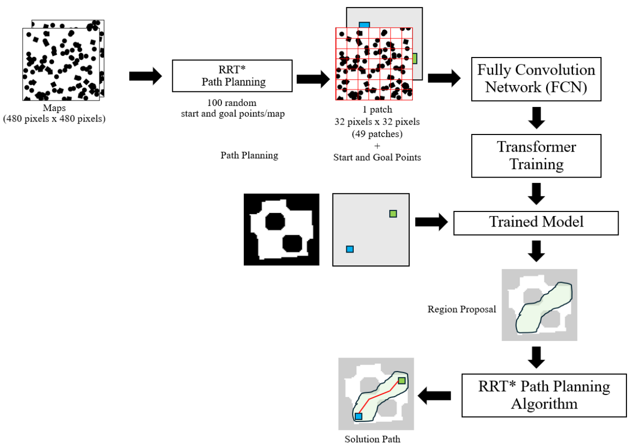


Fig. 6: Motion Planning Transformer (MPT)

V. Proposed Algorithm: Path Planning Transformers

This work proposes Path Planning Transformers, a transformer-based model for predicting collision-free paths in mobile robot navigation (Fig. 7). The pipeline begins with dataset generation using IRRT*-RRMS. Two datasets were created: a point robot dataset with 100 maps and 10,000 paths, and a real-size robot dataset with 30 maps and 3,000 paths. Each map is 480×480 pixels, and paths are stored as CSV files with start, goal, and intermediate coordinates.

A. Dataset Generation

We used random forest maps from the MPT paper. The IRRT*-RRMS algorithm, with post-processing, was used to generate paths. For real-robot datasets, obstacle areas were expanded by half the robot size. Maps were resized to 224×224 pixels for ViT compatibility, and 100 paths were generated

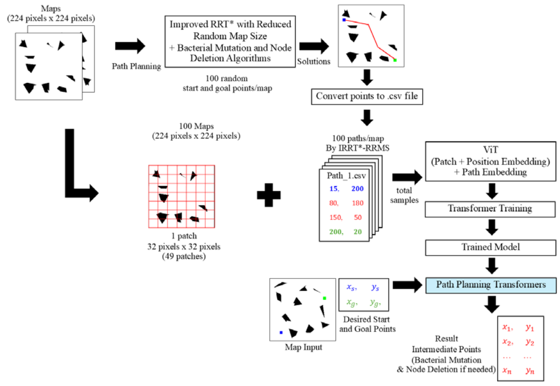


Fig. 7: Path Planning Transformers (PPT)

per map. Fig. 8 shows dataset examples and CSV files with start (x_s, y_s), goal (x_g, y_g), and intermediate points (x_i, y_i).

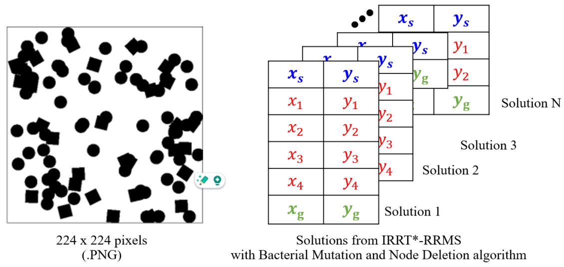
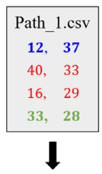


Fig. 8: Dataset Illustration

B. Dataset Encoding

We followed the ViT approach [30] to tokenize map patches (32×32) into 49 tokens, flattened to 1D embeddings. Each path is limited to 32 coordinates, including start (type-0), goal (type-1), intermediate (type-2), stop (type-4), and padding (type-3). Stop is marked as (0,0); padding as (-1,-1). Fig. 9 shows reordering for transformer input: $(0, 1, 2, 4, 3, \dots)$.



coordinate	12,37	33, 28	40,33	16,29	0,0	-1,-1	...	-1,-1
type	0	1	2	2	4	3	3	3
Attention mask	1	1	1	1	1	0	0	0

A total of 32 coordinates

Fig. 9: Path Embedding and Reordering

C. Decoder and Loss Function

The decoder is based on the Time Series Transformer [31]. We used 3 decoder layers, a feature size of 768, and predicted 32 waypoints. The loss function (Eq. (2)) combines coordinate MSE and type classification cross-entropy.

$$L = \frac{1}{S}\sum_{j=1}^{S}\left(\frac{1}{n}\sum_{i=1}^{n}(x_{ij}-\hat{x}_{ij})^2 + (y_{ij}-\hat{y}_{ij})^2 - t_{ij}\log p_{ij}\right) \quad (2)$$

where

n = Number of intermediate points in each sample
S = Total number of samples
x = Prediction x-coordinate of each point
y = Prediction y-coordinate of each point
$\hat{x}$ = Ground truth x-coordinate of each point
$\hat{y}$ = Ground truth y-coordinate of each point
t = One hot encoded of the truth value of coordinate type
p = Soft-Max probability for the type of each point

D. Training

Model training was performed using PyTorch on an Intel i9 laptop with an RTX 3080 GPU. Learning started at $5e-5$ and decayed to $1e-5$ as validation loss converged. Each dataset was trained with the Adam optimizer, batch size 100.

Point Robot Model: Trained on 10,000 samples for 156 epochs. The best model was saved at epoch 105 with loss 1,571 (Fig. 10).

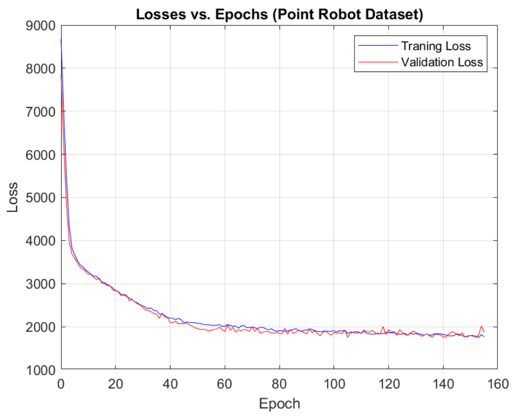


Fig. 10: Validation Loss (Point Robot Dataset)

Real-size Robot Model: Trained on 3,000 samples for 450 epochs. Optimal validation loss: 1,371 at epoch 353 (Fig. 11).

Figures 10 and 11 illustrate the training and validation loss curves for the transformer model trained on the point robot dataset and the real-size robot dataset, respectively. In both cases, the loss decreases rapidly during the initial training stage and gradually stabilizes as the training progresses. The validation loss closely follows the training loss throughout the training process, indicating stable convergence and suggesting that the model does not suffer from significant overfitting.

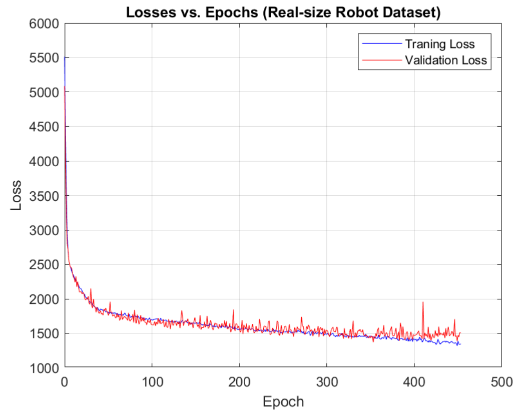


Fig. 11: Validation Loss (Real-size Robot Dataset)

These results demonstrate that the transformer model successfully learns the relationship between the map representation and the corresponding waypoint sequences generated by the IRRT*-RRMS algorithm.

E. Prediction Flow

The prediction processes are depicted in Fig. 12. The iterations start by determining the starting and goal points. The model predicts the result and concatenates the first coordinate to the original series. The iterations continue until the predicted coordinate is the padding or stop point; then, the iterations stop, and the solution is exported as a series of coordinates. The path results are checked for collision with the environment. If the path is collision-free, the solution can be used. On the other hand, the path results are fine-tuned with post-processing algorithms, which were mentioned in subsection post-processing algorithms. The solution path with the post-processing algorithms is called a Path Planning Transformer with Bacterial Mutation and Node Deletion algorithms (PPT-BM-ND).

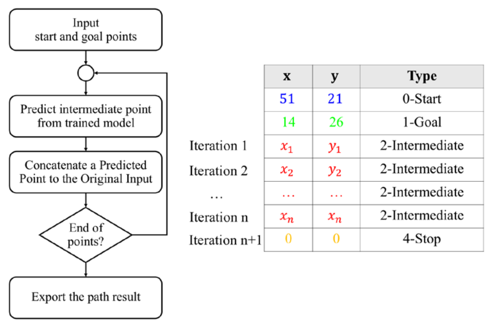


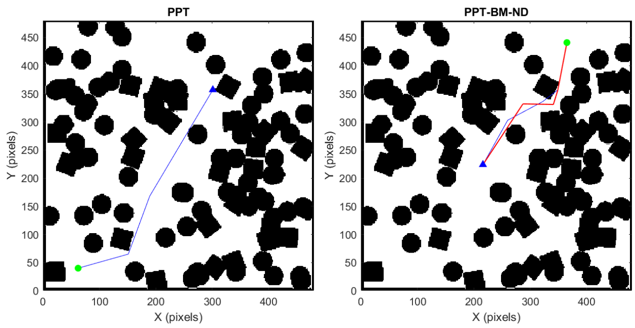
Fig. 12: Prediction Process of PPT

VI. RESULTS

In the following result figures, a blue triangle marker denotes the starting point, while a green circle marker represents the goal point. The blue line indicates the path generated by the Proposed Path Transformer model. For post-processing enhancement methods, the red line illustrates the final trajectory produced by the PPT-BM-ND approach, which incorporates bacterial mutation and node deletion algorithms.

A. Point Robot Case

Fig. 13 compares path planning results using PPT and PPT-BM-ND. PPT alone (blue) occasionally results in collisions, while PPT-BM-ND (red) corrects these. As seen in Table II, PPT-BM-ND achieves 100% collision-free paths with minimal increase in path length and substantially lower computational time than IRRT*-RRMS. The average computation time for 100 samples is 0.614 s. Although maps are from training, start and goal points were randomly chosen.



(a) PPT Only (blue) (b) PPT-BM-ND (red) vs. PPT (blue)

Fig. 13: Point Robot Path Planning Results from Start (blue triangle) to Goal (green circle) Points

TABLE II
NUMERICAL RESULTS: POINT ROBOT (Fig. 13b)

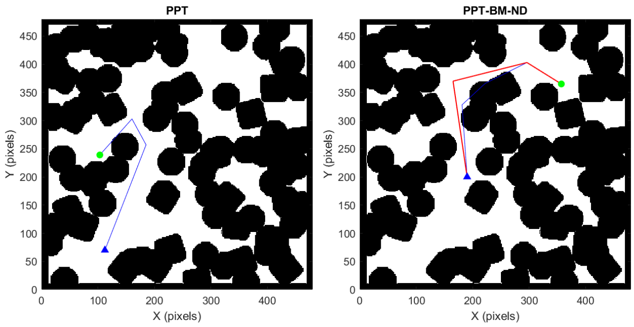
Method	Path Length (px)	Time (s)	Collision-Free?
IRRT*-RRMS	295	26.45	Yes
PPT	280	0.09	No
PPT-BM-ND	296	2.09	Yes

B. Real-size Robot Case

Fig. 14 shows results for real-size robots. PPT-BM-ND achieves 100% success, improving upon PPT's 75% and maintaining a shorter computation time than IRRT*-RRMS. The average inference time was 0.520 s (Table III).

TABLE III
NUMERICAL RESULTS: REAL-SIZE ROBOT (Fig. 14b)

Method	Path Length (px)	Time (s)	Collision-Free?
IRRT*-RRMS	367	17.25	Yes
PPT	338	0.88	No
PPT-BM-ND	379	2.70	Yes

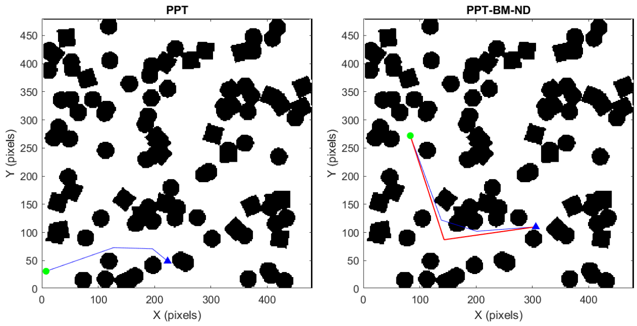


(a) PPT Only (blue) (b) PPT-BM-ND (red) vs. PPT (blue)

Fig. 14: Real-size Robot Results from Start (blue triangle) to Goal (green circle) Points

C. Unseen Map (Point Robot)

We tested generalization using unseen maps. As Fig. 15 shows, PPT produced valid paths in 40% of cases, while PPT-BM-ND achieved 100%. Table IV shows comparable path lengths but faster computation than IRRT*-RRMS. Average inference time: 1.22 s.



(a) PPT Only (blue) (b) PPT-BM-ND (red) vs. PPT (blue)

Fig. 15: Unseen Map Test: Point Robot from Start (blue triangle) to Goal (green circle) Points

TABLE IV
UNSEEN MAP RESULTS (Fig. 15b)

Method	Path Length (px)	Time (s)	Collision-Free?
IRRT*-RRMS	321	13.48	Yes
PPT	332	0.05	No
PPT-BM-ND	331	2.06	Yes

D. Real-World Implementation

A real environment was built to evaluate PPT on unseen data using the “Stephen” robot (Fig. 16). The test area (5.2 m. × 4.8 m.) was mapped using ROS and LiDAR (Fig. 17). The robot first explored the environment to construct an occupancy grid map representing obstacle regions and free space. This map was then used as the global map input for the path planning algorithm. The real-world experiment primarily serves as a proof-of-concept validation demonstrating that the proposed transformer-based planning approach can be deployed on a physical robot platform. Obstacle regions were added to ensure safety. Fig. 18 shows that PPT only produced a non-usable path (blue), while PPT-BM-ND (red) generated a collision-free solution suitable for execution. A demonstration video is available: (Video Link).

To further evaluate the robustness of the proposed approach, an additional real-world navigation experiment was conducted. In this experiment, different start and goal positions were selected within the environment to test the ability of the Path Planning Transformer to generate feasible paths under varying navigation conditions.

As shown in Fig. 19, the transformer model predicts intermediate waypoints between the start and goal locations. The predicted path is then refined using the Bacterial Mutation and Node Deletion post-processing algorithms to ensure collision-free navigation.

The results demonstrate that the proposed framework successfully generates feasible paths in different navigation scenarios within the same real-world environment. The refined path avoids obstacle regions and produces smooth trajectories that can be executed by the robot.

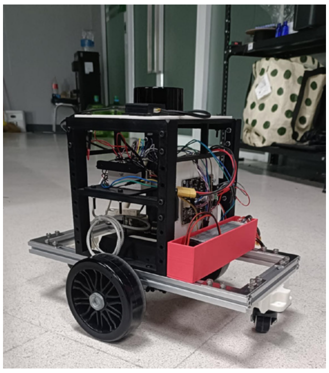


Fig. 16: Stephen Robot

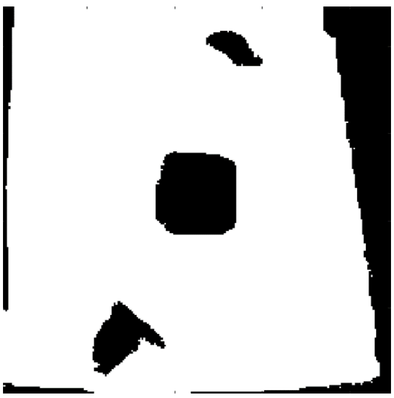


Fig. 17: Scanned Environment (SLAM)

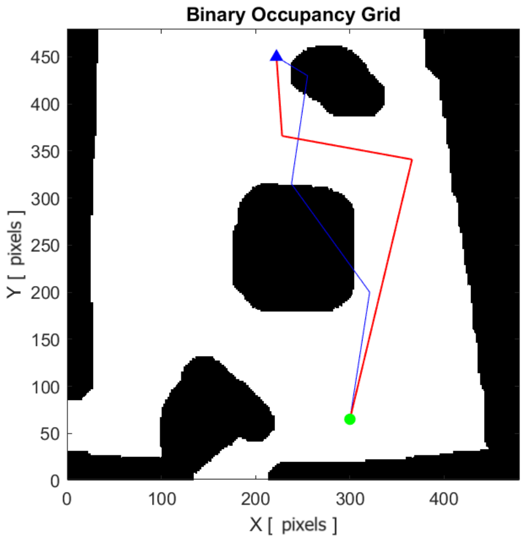


Fig. 18: Case I: Path Planning in Real Environment, The blue line shows the path predicted by the Path Planning Transformer, while the red line represents the refined trajectory obtained after applying the Bacterial Mutation and Node Deletion post-processing algorithms.

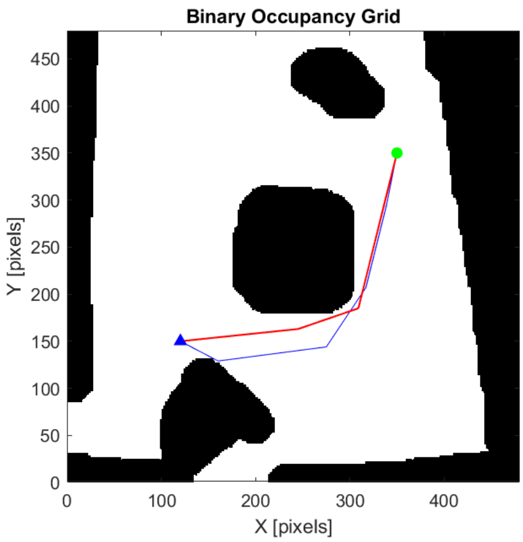


Fig. 19: Case II: Path Planning in Real Environment.

VII. DISCUSSION

The PPT-BM-ND approach consistently generated accurate, collision-free paths with fewer vertices and shorter inference times than IRRT*-RRMS and MPT. Table V summarizes the performance across algorithms. PPT-BM-ND achieved 100% collision-free path success rate in all tested cases, including unseen maps and real-world deployment. Although randomly generated maps were used during dataset generation to create diverse navigation scenarios, future work will investigate structured environments such as factory layouts with corridors and workstations.

TABLE V
PERFORMANCE COMPARISON: MPT vs. PPT VARIANTS

Method	Collision-free Path Success Rate (%)	Vertices Count
IRRT*-RRMS	100	0–10
MPT	100	60
PPT (Point Robot)	70	0–10
PPT (Real-size Robot)	75	0–10
PPT (Unseen Map)	40	0–10
PPT-BM-ND	100	0–10

A. Comparison of Deterministic and Stochastic Approaches

The IRRT*-RRMS algorithm operates as a deterministic sampling-based planner that guarantees convergence to an optimal solution given sufficient sampling. However, its iterative nature can be computationally intensive in complex or dynamic environments. In contrast, the proposed Path Planning Transformer is a stochastic model that learns the underlying patterns of collision-free paths from the IRRT*-RRMS dataset.

The primary benefits of this hybrid approach include:

- Computational Efficiency: Once trained, the PPT predicts paths in a single forward pass, significantly reducing the computation time compared to the iterative sampling required by IRRT*-RRMS.
- Knowledge Distillation: The model leverages high-quality, structured datasets generated by the improved algorithm to achieve better generalization than models trained on raw RRT data.

The primary drawback is the inherent uncertainty (stochasticity) of neural network predictions, which may occasionally result in sub-optimal or infeasible paths. To address this, we integrated Bacterial Mutation and Node Deletion as post-processing steps to refine these stochastic outputs into reliable, collision-free trajectories.

Although the proposed framework demonstrates promising performance in the evaluated scenarios, additional robustness tests such as noisy map inputs, partial observability, and dynamic obstacle environments could further strengthen the evaluation. These conditions are common in real-world robotic systems where sensor measurements may be uncertain or incomplete. Investigating the performance of transformer-based path planning models under such conditions is an important direction for future research.

VIII. CONCLUSION

In this paper, a Path Planning Transformer framework supervised by the Improved Rapidly-exploring Random Tree with Reduced Random Map Size algorithm was proposed for mobile robot navigation. The deterministic IRRT*-RRMS algorithm was first used to generate high-quality training datasets, which enabled the transformer model to learn structured path planning patterns from optimized solutions. The trained model was then applied to predict intermediate waypoints between the starting and goal positions in unknown environments.

Experimental results demonstrate that the proposed transformer-based model significantly reduces the computation time required for path planning compared with the original IRRT*-RRMS algorithm. While the PPT model alone occasionally produces infeasible paths due to the stochastic nature of neural network predictions, the integration of Bacterial Mutation and Node Deletion post-processing algorithms (PPT-BM-ND) effectively improves the reliability of the predicted trajectories. The results show that the proposed framework achieves a 100% collision-free success rate while maintaining substantially lower computation time than traditional sampling-based planning methods.

The proposed approach highlights the potential of combining deterministic planning algorithms with learning-based models, where classical planners provide reliable training data and transformer architectures enable fast inference during deployment. The experimental validation on both simulated environments and a real robot platform demonstrates the practical applicability of the method.

Future work will focus on extending the proposed framework to more complex navigation scenarios, including dynamic environments, multi-robot coordination, and larger-scale environments, as well as investigating improved learning strategies to further enhance the generalization capability of transformer-based path planning models.

REFERENCES

[1] M. Balogh and A. Vidács, "Optimizing camera stream transport in cloud-based industrial robotic systems," *Infocommunications Journal*, vol. XIV, no. 1, pp. 36–42, Mar. 2022. DOI: 10.36244/ICJ.2022.1.5. [Online]. Available: https://doi.org/10.36244/ICJ.2022.1.5.

[2] L. Liu, X. Wang, X. Yang, H. Liu, J. Li, and P. Wang, "Path planning techniques for mobile robots: Review and prospect," *Expert Systems with Applications*, vol. 227, p. 120 254, 2023, ISSN: 0957-4174. DOI: 10.1016/j.eswa.2023.120254. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S095741742300756X.

[3] S. Venu and M. Gurusamy, "A comprehensive review of path planning algorithms for autonomous navigation," *Results in Engineering*, vol. 28, p. 107 750, 2025, ISSN: 2590-1230. DOI: 10.1016/j.rineng.2025.107750. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2590123025038034.

[4] J. J. Johnson, U. S. Kalra, A. Bhatia, L. Li, A. H. Qureshi, and M. C. Yip, Motion planning transformers: A motion planning framework for mobile robots, 2022. *arXiv: 2106.02791* [cs.RO].

[5] J. Tang, Z. Mao, and H. Ma, "Large-scale multi-robot coverage path planning on grids with path deconfliction," *IEEE Transactions on Robotics*, vol. 41, pp. 3348–3367, 2025, ISSN: 1941-0468. DOI: 10.1109/tro.2025.3567476. [Online]. Available: http://dx.doi.org/10.1109/TRO.2025.3567476.

[6] F. Tao, Z. Ding, Z. Fu, M. Li, and B. Ji, "Efficient path planning for autonomous vehicles based on RRT* with variable probability strategy and artificial potential field approach," *Scientific Reports*, vol. 14, no. 1, p. 24 698, 2024, ISSN: 2045-2322. DOI: 10.1038/s41598-024-76299-9. [Online]. Available: https://doi.org/10.1038/s41598-024-76299-9.

[7] C. Cui, Z. Wang, J. Sui, Y. Zhang, and C. Guo, "An improved RRT behavioral planning method for robots based on PTM algorithm," *Scientific Reports*, vol. 14, no. 1, p. 21 776, 2024, ISSN: 2045-2322. DOI: 10.1038/s41598-024-72616-4. [Online]. Available: https://doi.org/10.1038/s41598-024-72616-4.

[8] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot, “Informed RRT*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic,” in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE*, Sep. 2014, pp. 2997–3004. DOI: 10.1109/iros.2014.6942976. [Online]. Available: http://dx.doi.org/10.1109/IROS.2014.6942976.

[9] S. M. LaValle, “Rapidly-exploring random trees: A new tool for path planning,” *The annual research report*, 1998. [Online]. Available: https://api.semanticscholar.org/CorpusID:14744621.

[10] J. Wang, J. Li, Y. Song, Y. Tuo, and C. Liu, “FC-RRT*: A modified RRT* with rapid convergence in complex environments,” *Journal of Computational Science*, vol. 77, p. 102 239, 2024, ISSN: 1877-7503. DOI: 10.1016/j.jocs.2024.102239. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877750324000322.

[11] S. Karaman and E. Frazzoli, “Incremental sampling-based algorithms for optimal motion planning,” *CoRR*, vol. abs/1005.0416, 2010. *arXiv: 1005.0416*. [Online]. Available: http://arxiv.org/abs/1005.0416.

[12] O. Arslan, K. Berntorp, and P. Tsiotras, “Sampling-based algorithms for optimal motion planning using closed-loop prediction,” in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 4991–4996. DOI: 10.1109/ICRA.2017.7989581.

[13] J. Ding, Y. Zhou, X. Huang, K. Song, S. Lu, and L. Wang, “An improved RRT* algorithm for robot path planning based on path expansion heuristic sampling,” *Journal of Computational Science*, vol. 67, p. 101 937, 2023, ISSN: 1877-7503. DOI: 10.1016/j.jocs.2022.101937. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877750322002964.

[14] A. Lonklang and J. Botzheim, “Improved rapidly exploring random tree with bacterial mutation and node deletion for offline path planning of mobile robot,” *Electronics*, vol. 11, no. 9, 2022, ISSN: 2079-9292. DOI: 10.3390/electronics11091459. [Online]. Available: https://www.mdpi.com/2079-9292/11/9/1459.

[15] A. Lonklang and J. Botzheim, “Mobile robot path planning for unknown static obstacle avoidance by improved RRT* algorithm,” in 2024 10th International Conference on Automation, *Robotics and Applications (ICARA)*, 2024, pp. 155–159. DOI: 10.1109/ICARA60736.2024.10553042.

[16] Y. Pan, Y. Tao, W. Lu, G. Li, and J. Cong, “Dynamic path planning of vehicles based on the adaptive potential field and hierarchical replacement immune algorithm,” *Arabian Journal for Science and Engineering*, 2024, ISSN: 2191-4281. DOI: 10.1007/s13369-023-08541-x. [Online]. Available: https://doi.org/10.1007/s13369-023-08541-x.

[17] F. Hamad, H. N. Fakhouri, F. Alzghoul, and J. Zraqou, “Development and design of object avoider robot and object, path follower robot based on artificial intelligence,” *Arabian Journal for Science and Engineering*, 2024, ISSN: 2191-4281. DOI: 10.1007/s13369-024-09365-z. [Online]. Available: https://doi.org/10.1007/s13369-024-09365-z.

[18] J. Yu, Y. Su, and Y. Liao, “The path planning of mobile robot by neural networks and hierarchical reinforcement learning,” *Frontiers in Neurorobotics*, vol. 14, 2020, ISSN: 1662-5218. DOI: 10.3389/fnbot.2020.00063. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fnbot.2020.00063.

[19] H. Bharadwaj and V. K. E, “Comparative study of neural networks in path planning for catering robots,” *Procedia Computer Science*, vol. 133, pp. 417–423, 2018, International Conference on Robotics and Smart Manufacturing (RoSMa2018), ISSN: 1877-0509. DOI: 10.1016/j.procs.2018.07.051. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050918309967.

[20] R. Yonetani, T. Taniai, M. Barekatain, M. Nishimura, and A. Kanezaki, “Path planning using neural A* search,” *CoRR*, vol. abs/2009.07476, 2020. *arXiv:2009.07476*. [Online]. Available: https://arxiv.org/abs/2009.07476.

[21] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, “Attention is all you need,” *CoRR*, vol. abs/1706.03762, 2017. *arXiv: 1706.03762*. [Online]. Available: http://arxiv.org/abs/1706.03762.

[22] Y. Bazi, L. Bashmal, M. M. A. Rahhal, R. A. Dayil, and N. A. Ajlan, “Vision transformers for remote sensing image classification,” *Remote Sensing*, vol. 13, no. 3, 2021, ISSN: 2072-4292. DOI: 10.3390/rs13030516. [Online]. Available: https://www.mdpi.com/2072-4292/13/3/516.

[23] N. Hütten, R. Meyes, and T. Meisen, “Vision trans-former in industrial visual inspection,” *Applied Sciences*, vol. 12, no. 23, 2022, ISSN: 2076-3417. DOI: 10.3390/app122311981. [Online]. Available: https://www.mdpi.com/2076-3417/12/23/11981.

[24] J. J. Johnson, L. Li, A. H. Qureshi, and M. C. Yip, “Motion planning transformers: One model to plan them all,” *CoRR*, vol. abs/2106.02791, 2021. *arXiv: 2106.02791*. [Online]. Available: https://arxiv.org/abs/2106.02791.

[25] S. Jung, K. Kim, H. Kwak, and Y. Park, “A worrying analysis of probabilistic time-series models for sales forecasting,” *CoRR*, vol. abs/2011.10715, 2020. *arXiv: 2011.10715*. [Online]. Available: https://arxiv.org/abs/2011.10715.

[26] Q. Wen, T. Zhou, C. Zhang, *et al.*, Transformers in time series: A survey, 2023. *arXiv: 2202.07125* [cs.LG].

[27] V. Flunkert, D. Salinas, and J. Gasthaus, “DeepAR: Probabilistic forecasting with autoregressive recurrent networks,” *CoRR*, vol. abs/1704.04110, 2017. *arXiv: 1704.04110*. [Online]. Available: http://arxiv.org/abs/1704.04110.

[28] L. Bi, O. Wally, G. Hu, A. U. Tenuta, Y. R. Kandel, and D. S. Mueller, “A transformer-based approach for early prediction of soybean yield using time-series images,” *Front Plant Sci*, vol. 14, p. 1 173 036, 2023, ISSN: 1664-462X (Print) 1664-462x. DOI: 10.3389/fpls.2023.1173036.

[29] A. Lonklang and J. Botzheim, “A rapidly-exploring random tree algorithm with reduced random map size,” in *2023 9th International Conference on Automation, Robotics and Applications (ICARA)*, 2023, pp. 356–361. DOI: 10.1109/ICARA56516.2023.10125934.

[30] A. Dosovitskiy, L. Beyer, A. Kolesnikov, *et al.*, “An image is worth 16x16 words: Transformers for image recognition at scale,” *CoRR*, vol. abs/2010.11929, 2020. *arXiv: 2010.11929*. [Online]. Available: https://arxiv.org/abs/2010.11929.

[31] K. Rasul, A. Ashok, A. R. Williams, et al., Lag-Llama: Towards foundation models for probabilistic time series forecasting, 2024. *arXiv: 2310.08278* [cs.LG]. [On-line]. Available: https://arxiv.org/abs/2310.08278.

Aphilak Lonklang’s degrees earned: Bachelor’s degree in Aeronautical Engineering (First Class Honor B.Eng) and Master’s degree in Mechanical Engineering (M.Eng) from the Suranaree University of Technology (SUT), Nakhon Ratchasima, Thailand. Experiences: Research Assistant in Industrial Automation Research Unit for three years in SUT (2013-2016). Lecturer at the School of Mechanical Engineering, Institute of Engineering, SUT (August 2016 - present). Award in Teaching Experience: Associate Fellow of the Higher Education Association from UKPSF (December 1st 2021). Research Interests: Industrial Automation, Industrial Robotics, Computational Intelligence.

János Botzheim received the M.Sc. and Ph.D. degrees in computer engineering from the Budapest University of Technology and Economics, in 2001 and 2008, respectively. He is an associate professor and the Head of Department of Artificial Intelligence at Eötvös Loránd University (ELTE), Faculty of Informatics, Budapest, Hungary. He had several international visiting fellowships. For six years he was an associate professor at Tokyo Metropolitan University. He is a member of several scientific societies such as IEEE, Hungarian Academy of Engineering, John von Neumann Computer Science Society, and Hungarian Fuzzy Association. His research interests are computational intelligence and cognitive robotics.