www.ijbar.org
ISSN 2249-3352 (P) 2278-0505 (E)
Cosmos Impact Factor-5.86

GRAFT: Gradient-based Real-time Autoencoder-Fused Trajectory Optimizer for Safe Motion Planning

[1] Mrs. B. Meenakshi, [2] Shabistha, [3] S. Poojitha, [4] Mohd Wajahath Ullah

1 Assistant Professor in the Department of C.S.E, Matrusri Engineering College, Saidabad, Hyderabad, Telangana, India

2,3,4 UG Scholar in the Department of C.S.E, Matrusri Engineering College, Saidabad, Hyderabad, Telangana, India

Abstract - **Safe and dynamically feasible trajectory planning is a core challenge in autonomous systems, especially in cluttered or unpredictable environments. Traditional Model Predictive Control (MPC) frameworks are popular due to their ability to optimize trajectories over a prediction horizon while respecting constraints. However, these methods rely on manually engineered cost functions that require extensive tuning and often fail to generalize to unseen scenarios. They also lack mechanisms to incorporate learned priors on what constitutes a "safe" trajectory, limiting adaptability and robustness.**

To address these limitations, we propose GRAFT:Gradient-based Real-time Autoencoder-Fused Trajectory Optimizer for safe motion planning—a novel framework that integrates a self-supervised autoencoder directly into the MPC optimization process. The autoencoder, trained in PyTorch on a dataset of expert-like trajectories without human-labeled data, learns a latent representation of safe motion patterns. Its reconstruction loss acts as a differentiable safety prior within the MPC loss function, guiding the optimizer to generate trajectories that remain within this learned safe manifold.

GRAFT is implemented as a fully monolithic Python application with a Streamlit interface, enabling real-time interaction where users can define start and goal positions, place obstacles, and tune optimization parameters such as learning rate and iteration count. The optimizer uses gradient-based methods on a composite loss combining goal-reaching, smoothness, obstacle avoidance, and autoencoder-based safety terms to produce smooth, feasible trajectories.

We evaluate GRAFT on synthetic environments and a custom trajectory dataset. Results demonstrate improved convergence, higher trajectory feasibility, reduced false motions, and less manual parameter tuning compared to baseline methods. Real-time visualizations allow users to monitor optimization progress, reconstruction quality, and final path safety, making SOMTP both an algorithmic innovation and a practical tool for researchers and developers.

This work highlights the potential of embedding learned safety priors within planning frameworks, paving the way toward more intelligent and adaptable autonomous navigation systems.

Keywords - **Trajectory planning, Model Predictive Control (MPC), self-supervised learning, autoencoder, safe navigation, motion optimization, obstacle avoidance, autonomous systems.**

I. INTRODUCTION

Autonomous systems require safe and efficient trajectory planning to operate effectively in dynamic and constrained environments. Model Predictive Control (MPC) is a widely adopted approach for such tasks, as it can plan future actions while respecting system constraints over a prediction horizon. However, traditional MPC methods often depend on hand-crafted cost functions that require extensive tuning and may struggle to adapt to complex, real-world scenarios with unpredictable obstacles and environments.

To overcome these challenges, we propose **GRAFT**—Gradient-based real-time autoencoder-fused trajectory optimizer for safe motion planning. GRAFT integrates a self-supervised autoencoder that learns a latent representation of feasible and safe trajectories without requiring manual annotations. This learned representation is incorporated into the MPC optimization as a differentiable safety prior through a custom loss function, enabling the planner to generate dynamically feasible and collision-free trajectories more robustly.

In addition to the algorithmic framework, we developed a real-time, interactive Streamlit application that allows users to visualize trajectories, place obstacles, and fine-tune optimization parameters on the fly. This practical tool not only demonstrates GRAFT's effectiveness in generating safe and smooth motion plans but also highlights its potential for scalable deployment in autonomous navigation tasks requiring adaptability and safety.

www.ijbar.org
ISSN 2249-3352 (P) 2278-0505 (E)
Cosmos Impact Factor-5.86

II. OBJECTIVES AND SCOPE

A. AIM

The aim of this project is to develop a safe and adaptable trajectory planning system that addresses the limitations of traditional MPC-based methods. It leverages self-supervised learning to eliminate the need for labeled data and enhances planning by learning a latent representation of safe motion. The system integrates this learned representation into the MPC optimizer through a custom loss function that accounts for obstacle avoidance and dynamic feasibility. GRAFT is designed to be scalable, modular, and user-friendly, with an interactive interface for real-time visualization and control, making it suitable for deployment in practical autonomous navigation tasks.

B. Future Development

This project focuses on the design and implementation of a self-supervised learning-based optimizer for safe trajectory planning using Model Predictive Control (MPC). The system's core components include an autoencoder that learns a latent representation of safe trajectories and an MPC optimizer that incorporates this representation through a custom loss function. Currently supporting real-time planning in simulated environments with both dynamic and static obstacles, the solution features a Streamlit-based interface for interactive visualization and parameter tuning.

While presently a prototype, the system is designed with scalability and adaptability in mind, positioning it for future deployment in real-world autonomous navigation applications. Potential extensions include integration with physical robotic platforms, enhancement of the learning model with richer datasets, and expansion to more complex environments such as urban driving or aerial drone navigation.

III. LITERATURE REVIEW

1.*Optimal Model Predictive Control for Path Tracking of Autonomous Vehicles [1]*
Aim: Develop an MPC that optimizes path tracking accuracy and energy use while reducing computational cost.
Scope: Path tracking control for autonomous vehicles using a dynamically updated linear model.
Results: Achieves better computational efficiency and comparable or improved tracking versus nonlinear MPC methods.
Future Enhancements: Extend to more complex dynamics and improve real-time performance on embedded platforms.

2.*Robust Trajectory Planning Based on Historical Information for Autonomous Vehicles[2]*
Aim: To develop a robust trajectory planner that adapts to dynamic environments by leveraging historical motion data.
Scope: Combines candidate generation, collision/stability detection, and speed planning for real-time autonomous driving.
Results: Demonstrated improved trajectory smoothness and stability in both simulation and field tests.
Future Enhancements: Can be expanded with machine learning modules for predictive stability assessment and adaptive speed tuning.

3 *An Online Time-Optimal Trajectory Planning Method for Constrained Multi-Axis Trajectory With Guaranteed Feasibility[3]*
Aim: To propose a time-optimal trajectory planning method for continuous multi-axis paths suitable for real-time applications.
Scope: Targets online trajectory planning under high-order kinematic constraints for robotics, CNC machines, and autonomous vehicles.
Results: Achieves near-optimal performance compared to offline methods while maintaining real-time computational efficiency.
Future Enhancements: Can be extended to more complex dynamic environments and integrated with adaptive control systems.

4.An Improved Model-Free Predictive Control Method for Nonlinear Time-Delay Systems[4]
Aim: Enhance model-free adaptive control by incorporating output error and predictive control.
Scope: Control of nonlinear and large time-delay systems without explicit models.
Result: Achieved stable output, better control, and faster response in simulations.
Future Enhancement: Real-time implementation and application to multi-variable systems.

5.*Development of a Human-Like Model Predictive Path Tracking Control Algorithm for Autonomous Vehicles with Self-Tuning of Control Period[5]*
Aim: To design an MPC algorithm with self-tuning control period that mimics human driving behavior for improved path tracking.
Scope: Autonomous vehicle path tracking focusing on lateral preview and yaw angle errors using a bicycle model-based error dynamics.
Results: Demonstrated effective replication of human control characteristics during curved path tracking in simulation.
Future Enhancements: Extend testing to real-world driving scenarios and adapt the control algorithm to varying road and vehicle conditions.

IV SYSTEM ARCHITECTURE

The architecture of **DRAFT** is designed to integrate traditional **Model Predictive Control (MPC)** with **self-supervised learning** techniques to enable safe, smooth, and optimized trajectory planning in dynamic environments. It is implemented using **PyTorch**, **Streamlit**, **NumPy**, and **Matplotlib**, offering a fully interactive AI-enhanced

www.ijbar.org
ISSN 2249-3352 (P) 2278-0505 (E)
Cosmos Impact Factor-5.86

planning environment.

A. Core Architectural Layers:

1. Presentation Layer (Streamlit Interface):

The presentation layer is developed using Streamlit, providing a clean and intuitive graphical user interface (GUI) for interacting with the optimization system. Users can input key parameters such as the start and goal positions, number of waypoints, obstacle layout, learning rate, and number of epochs. The GUI dynamically visualizes the optimized trajectory and overlays obstacle fields for clear feedback. Additionally, it displays a real-time loss curve, which illustrates how the current trajectory improves over time during optimization. This interface ensures accessibility even to users with minimal technical expertise.

2. Application Layer (Trajectory Optimization Engine):

The application layer in GRAFT is responsible for trajectory optimization using a self-supervised learning approach integrated with Model Predictive Control principles. It initializes a noisy linear trajectory between the start and goal positions and iteratively refines it by minimizing a custom loss function. This loss function combines several components: a goal-reaching term to ensure the trajectory ends at the target, a smoothness term to promote feasible motion, an obstacle avoidance penalty for safety, and a reconstruction loss derived from a trained autoencoder that captures safe motion patterns. Optimization is performed using gradient descent with the Adam optimizer over a predefined number of iterations. Throughout this process, the system provides real-time visualization of the evolving trajectory and live updates on loss values. Notably, GRAFT is implemented as a monolithic Python/Streamlit application, forgoing a separate backend or API calls, which simplifies integration and user interaction within a single interface.

3. Data Layer (Trajectory Data and Model Files):

In GRAFT, the data layer consists of an MPC-generated safe trajectory dataset stored in spreadsheet format (e.g., `.xlsx`) and a trained autoencoder model saved as a `.pth` file. The dataset is used for training, while the model file is used for inference. Though no database like MongoDB is used, structured file handling ensures data consistency and supports scalable updates for retraining or deployment.

4. Model Layer (Self-Supervised Learning Core):

The autoencoder is trained in a **self-supervised manner**, learning a **latent embedding** of safe trajectories. It reconstructs input trajectories, and the **reconstruction error** is used during optimization to ensure that new trajectories resemble those learned from data.

This promotes **generalization** to novel inputs and adds a **learned safety prior** into the trajectory planning process.

B. Trajectory Optimization and Safety Assurance Framework

Safety and optimality are fundamental principles of the proposed GRAFT. The system integrates model predictive control with deep learning–based representation learning to generate safe, efficient, and dynamically feasible trajectories. The planning process is governed by the following mathematical formulations:

- **Model Predictive Control (MPC):** At each iteration, the trajectory is computed by minimizing a total cost function L_total over a horizon of N waypoints:

  $$L_{total} = L_{goal} + \lambda_1 L_{smooth} + \lambda_2 L_{obstacle} + \lambda_3 L_{reconstruction}$$

- **Goal Loss (L_goal):** Encourages the final waypoint x_N to coincide with the desired goal location:

  $$L_{goal} = \| x_N - x_{goal} \|^2$$

- **Smoothness Loss (L_smooth):** Promotes continuity and smooth transitions between consecutive waypoints by penalizing second-order differences:

  $$L_{smooth} = \sum_{i=1}^{N-1} \| x_{i+1} - 2x_i + x_{i-1} \|^2$$

- **Obstacle Penalty (L_obstacle):** Ensures that trajectories avoid known obstacles $O = \{ (o_j, r_j) \}$, where o_j and r_j represent the position and radius of the j-th obstacle:

  $$L_{obstacle} = \sum_{i=1}^{N} \sum_{j=1}^{M} \max(0, r_j - \| x_i - o_j \|)^2$$

- **Latent Trajectory Representation via Autoencoder:** A deep autoencoder, pre-trained on a dataset of safe trajectories, is used to encode trajectories into a compact latent representation:

  $$L_{reconstruction} = \| x_{1:N} - \hat{x}_{1:N} \|_2$$

  This structure allows the system to learn and reuse feasible motion patterns, improving generalization and planning efficiency.

- **Gradient-Based Optimization:**

  The total cost is minimized using gradient descent techniques:

  $$x_{1:N}(t+1) = x_{1:N}(t) - \eta * \nabla_x L_{total}$$

C. Interaction Flow and Communication

The SOMTP system operates as a tightly integrated pipeline within a unified runtime environment, where all modules

www.ijbar.org
ISSN 2249-3352 (P) 2278-0505 (E)
Cosmos Impact Factor-5.86

communicate through direct in-memory data exchange. The user inputs start and goal positions along with obstacle data via the interface, which is directly fed into the trajectory optimization engine. This engine initializes a trajectory and iteratively refines it by minimizing a custom loss function that integrates Model Predictive Control principles with self-supervised learning from a trained autoencoder. The autoencoder guides the optimizer by ensuring trajectories conform to learned safe motion patterns. All computations and visualizations happen in real time within the same application, enabling low-latency communication and efficient execution without relying on external APIs or separate backend services.

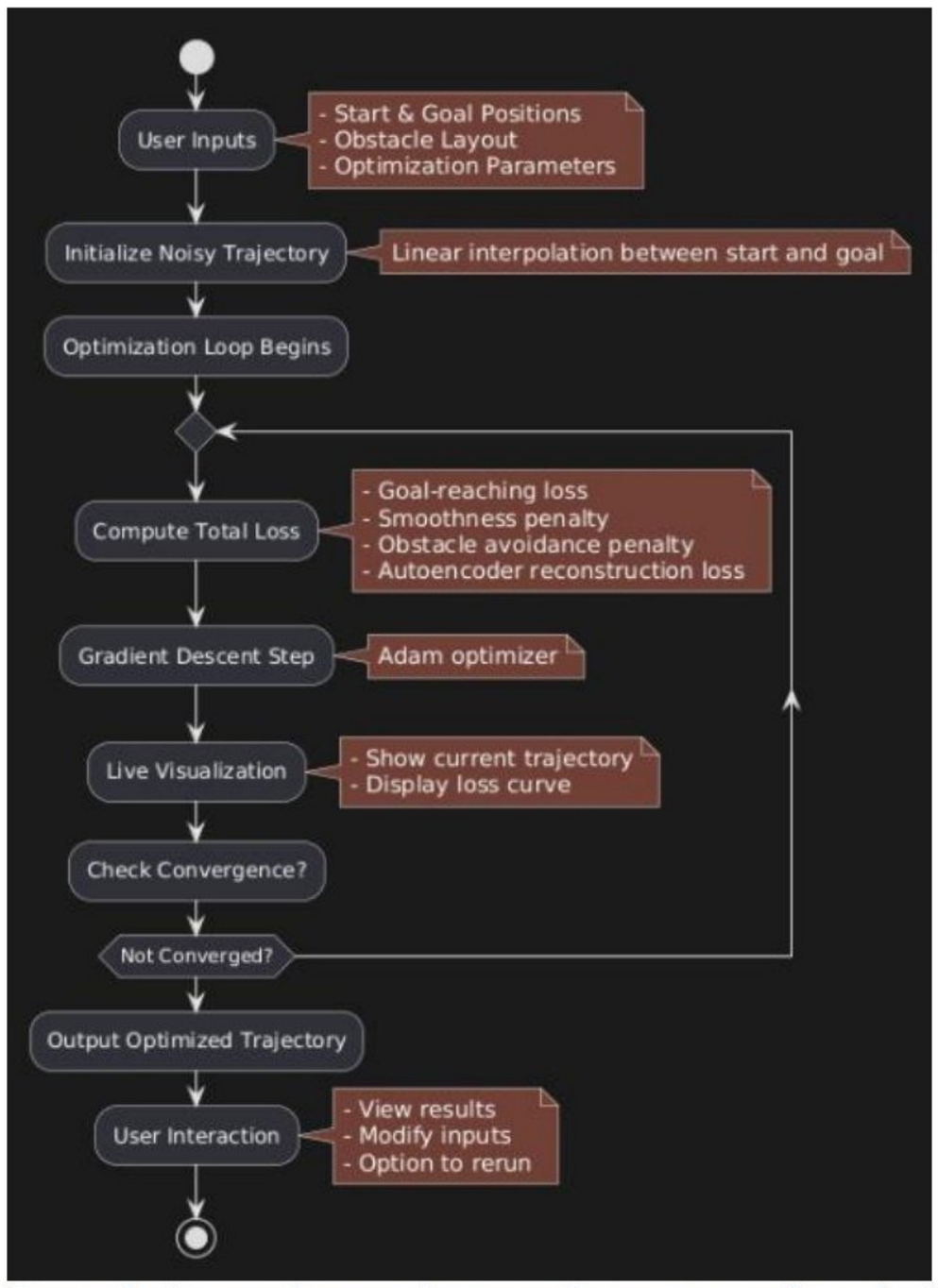


Figure 1: Activity diagram depicting the interaction flow of system

V PROPOSED METHODOLOGY

The proposed system integrates Model Predictive Control (MPC) with a self-supervised autoencoder directly within the trajectory optimization loop to address safe trajectory planning in dynamic and constrained environments. The process starts with the user specifying the start and goal positions along with obstacle data, forming the basis for a simulated planning scenario. A noisy initial trajectory is created and iteratively optimized using gradient descent to minimize a composite loss function.

This loss function includes terms for reaching the goal, trajectory smoothness, obstacle avoidance, and a reconstruction loss computed using a trained autoencoder. The autoencoder, having learned the latent features of safe trajectories from a dedicated dataset, guides the optimizer toward trajectories that conform to safety characteristics without needing explicit labels.

By integrating safety constraints into the optimization process itself, the system achieves both real-time performance and safety assurance. This hybrid, self-supervised approach is suitable for autonomous systems where robust, adaptable, and safe motion planning is critical.

A. System Input and Scenario Configuration

The system begins by accepting key user-defined parameters, including start and goal coordinates, obstacle locations, learning rate, and the number of optimization iterations. These inputs define the trajectory planning scenario and influence both the optimizer's convergence behavior and the quality of the resulting path. The learning rate controls how aggressively the optimizer updates the trajectory, while the iteration count determines how many refinement steps are performed. This setup enables users to customize and experiment across various planning environments. The provided data initializes the simulation environment, forming the foundation for safe trajectory generation under complex constraints.

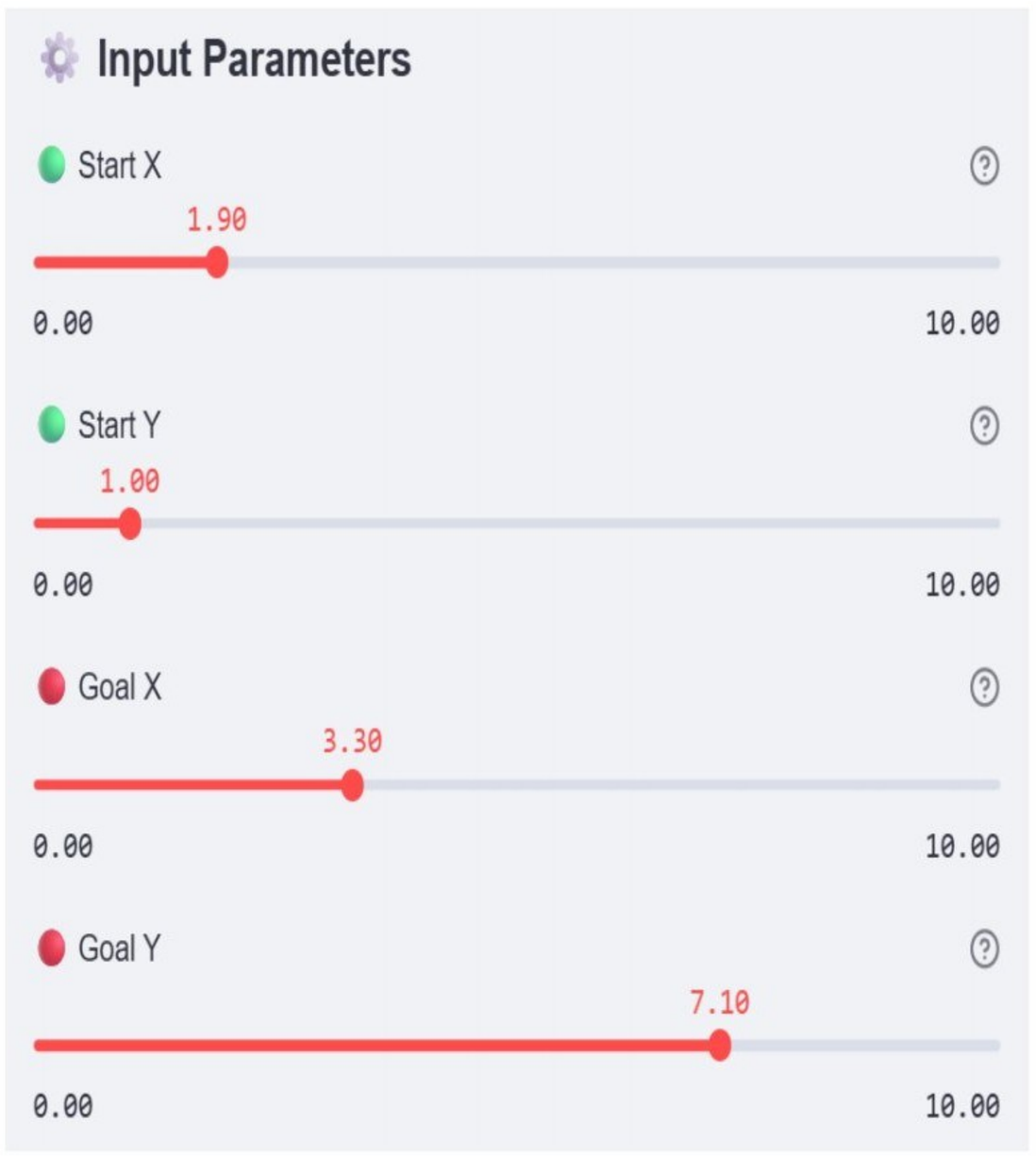


Figure 2 : Illustrates the start and goal positions

www.ijbar.org
ISSN 2249-3352 (P) 2278-0505 (E)
Cosmos Impact Factor-5.86

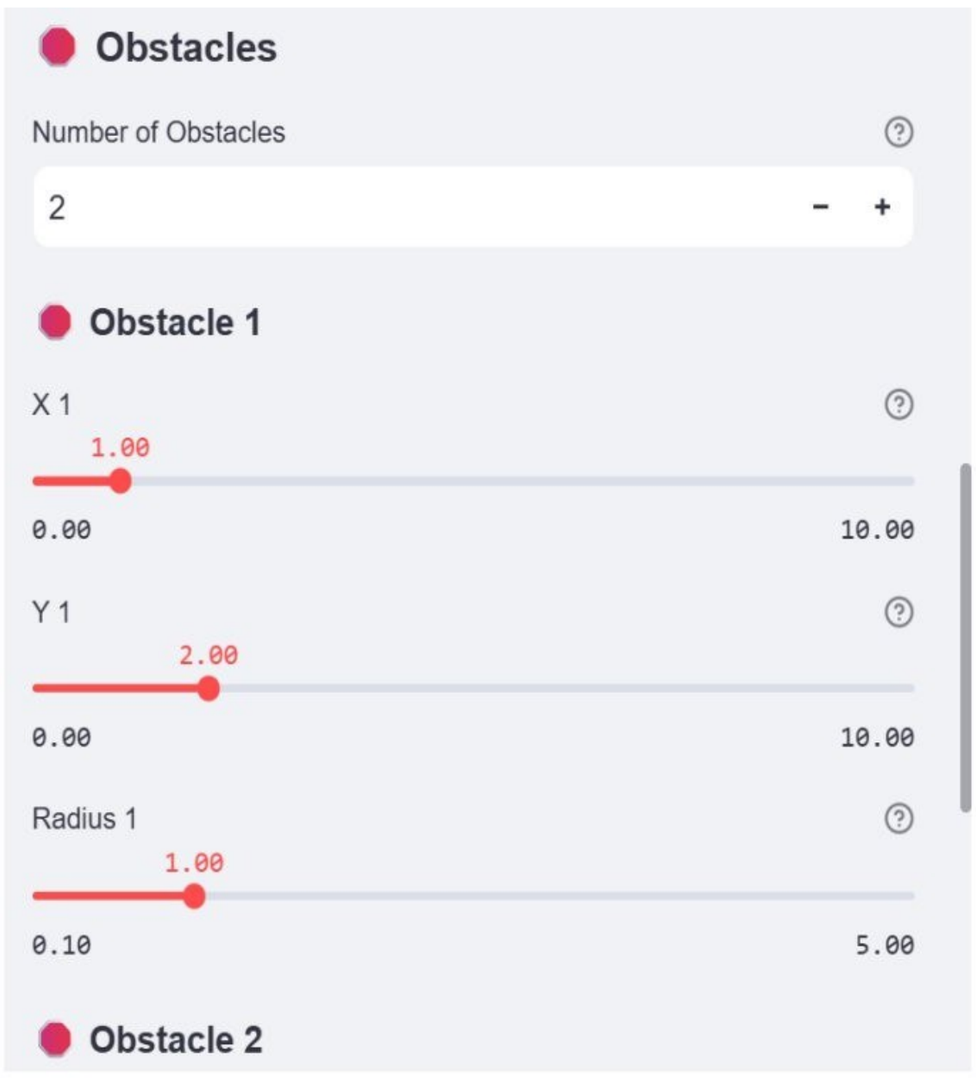


Figure 3: Shows Number of obstacles along with its radius and placement

B. Trajectory Generation using MPC and Auto-encoder

The system employs Model Predictive Control (MPC) to generate optimized trajectories that satisfy constraints such as obstacle avoidance and smooth motion. To intrinsically ensure safety during trajectory generation, a self-supervised autoencoder is incorporated within the framework. This autoencoder is trained on safe trajectory data and operates alongside the MPC to implicitly guide trajectory optimization. By continuously reconstructing the candidate trajectories and monitoring reconstruction errors, the system detects deviations from safe trajectory patterns in real time. Trajectories that significantly deviate from learned safe dynamics—indicated by high reconstruction error—are avoided or re-optimized, enabling the system to maintain safety without relying on explicit external validation or labeling.

C. Result Visualization and Feedback

After validation, the best possible trajectory is visualized for the user. The interface displays trajectory lines, obstacles, and safety status. Users are informed whether the path was deemed safe or needs adjustment based on the visualization. This real-time feedback allows users to reconfigure inputs or analyze outcomes, promoting transparency and iterative optimization

D. Integrated System Workflow:

All components of the system—from input collection to final visualization—are orchestrated within a single runtime environment using direct function calls and in-memory data sharing. This ensures seamless communication, low-latency processing, and modular code integration. Unlike API-driven systems, this setup enhances execution speed and reduces complexity.

VI IMPLEMENTATION AND RESULTS

The implementation leverages Streamlit to create an interactive interface allowing users to input key parameters for safe trajectory planning including start and goal coordinates, obstacle configuration, learning rate, and number of training iterations. Internally, the application is developed in Python, integrating a Model Predictive Control (MPC) module with a self-supervised autoencoder trained using PyTorch. The system processes inputs in real time and computes trajectories using MPC under dynamic constraints. The autoencoder is embedded within the optimization process, providing a reconstruction loss that acts as a learned safety prior, guiding the optimizer to generate trajectories consistent with previously observed safe behaviors. Data exchange between components occurs in-memory to maintain responsiveness.

The system was tested on a dataset of trajectories to evaluate how effectively the autoencoder-informed optimization shapes trajectory generation. Visualizations via Streamlit enable users to observe the evolving trajectory and its reconstruction, facilitating insight into how the optimization process balances goal achievement, smoothness, obstacle avoidance, and adherence to learned safe trajectory patterns. The system demonstrated real-time performance and effective trajectory planning across varied obstacle configurations.

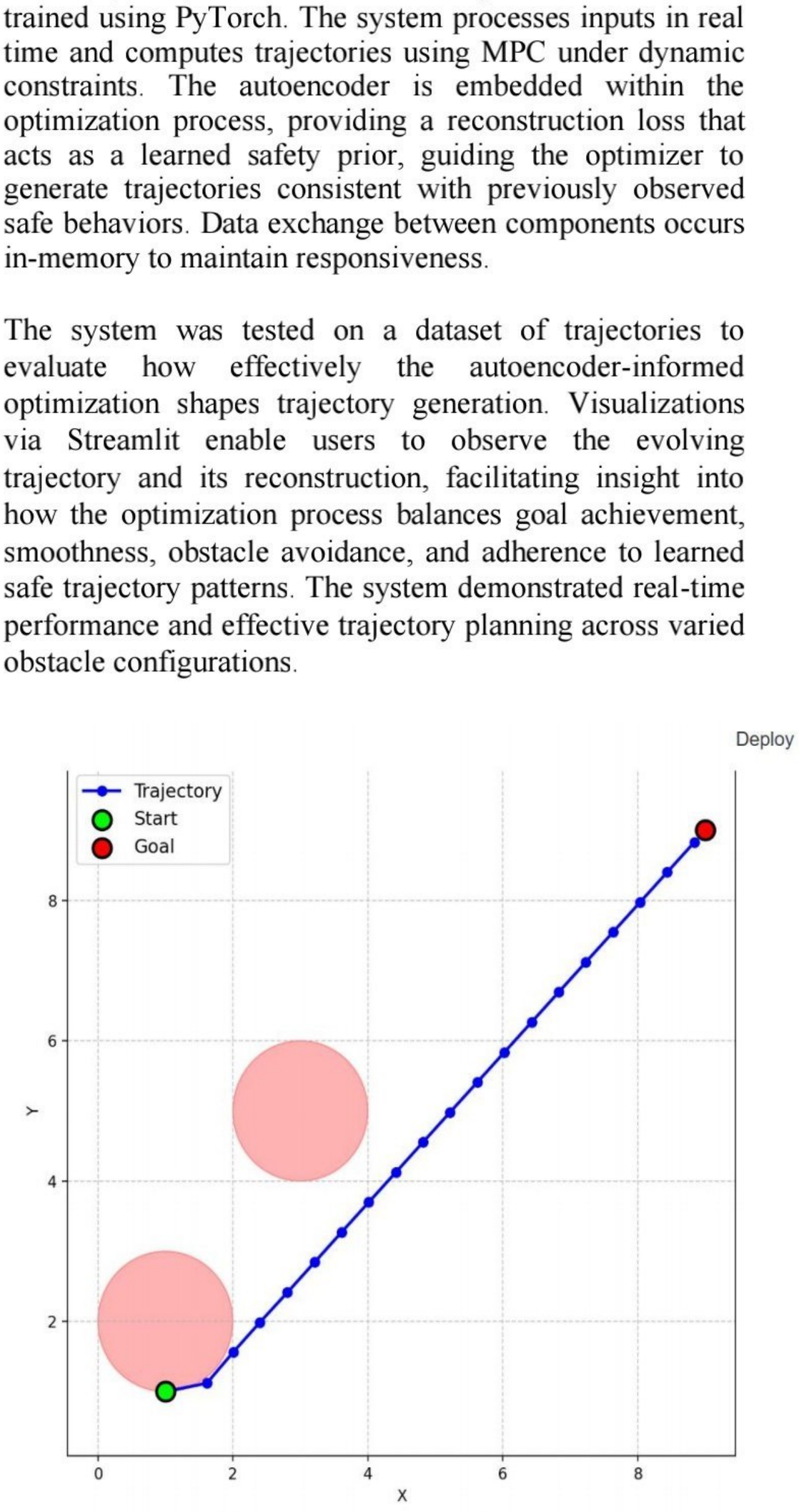


Figure 4: Showing Trajectory based on user input

Index in Cosmos
JUNE 2025, Volume 15, ISSUE 2
UGC Approved Journal

www.ijbar.org
ISSN 2249-3352 (P) 2278-0505 (E)
Cosmos Impact Factor-5.86

VI TESTING RESULTS

Test Case-1: Single Object, Small Radius

Input Values:
start position: (1,1)
Goal position: (8,8.5)
obstacle at (3.7, 3.7), radius 1.0

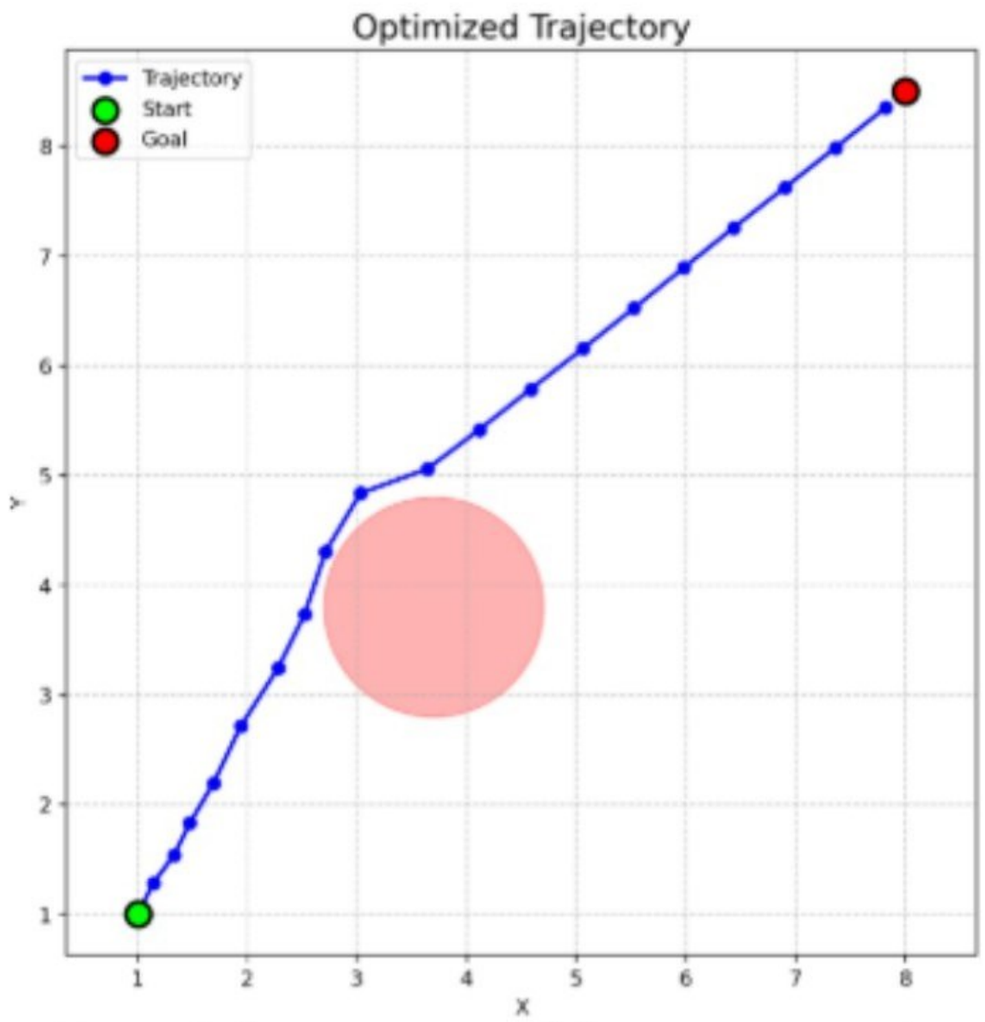


Figure 5:the agent successfully generates a safe and smooth trajectory around a single obstacle with a small radius.

Test Case-2: Single-Object, Big Radius

Input Values:
start position: (1,1)
Goal position: (9,9)
Obstacle at (3.8,4.1), radius 2.8

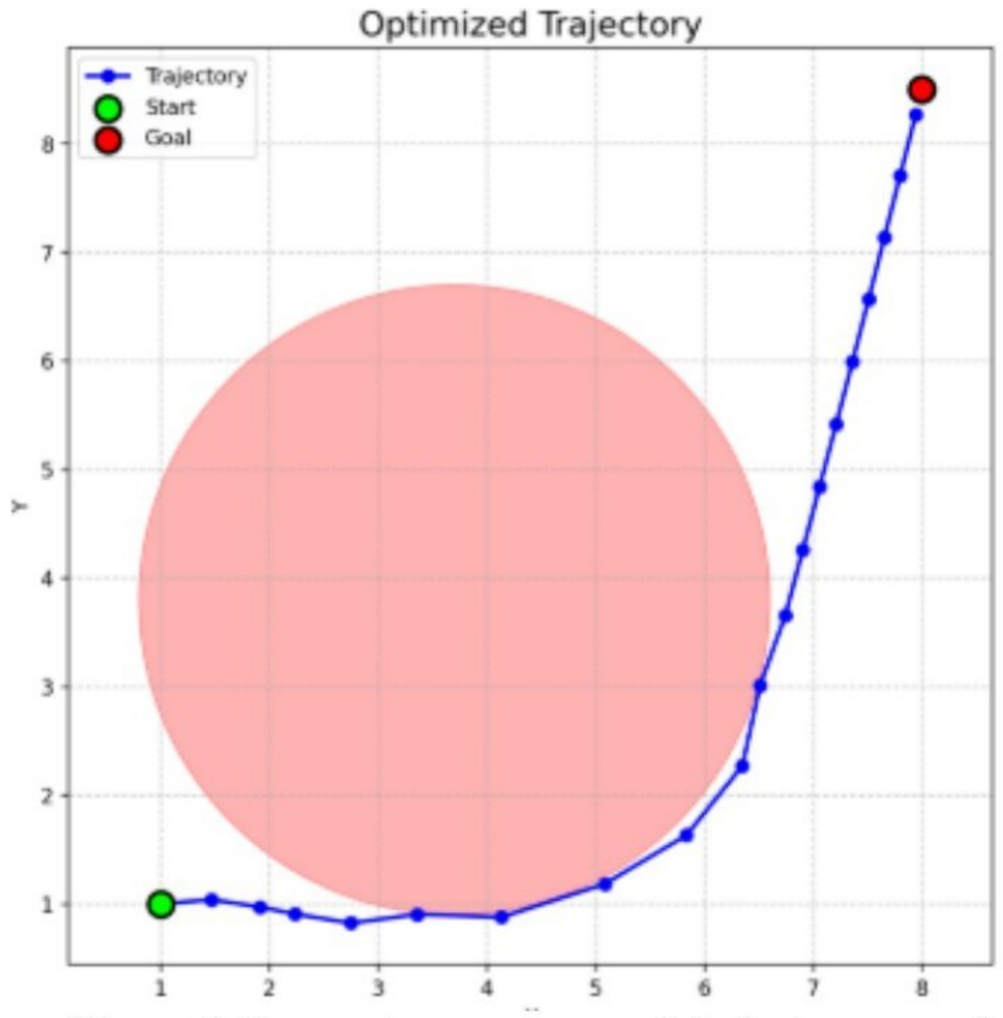


Figure 6: the agent generates a safe trajectory around a single obstacle with a large radius

Test Case-3: Multiple Object

Input Values:
start position: (1,1)
goal position: (9,9)
Obstacles: obstacle at (2.9 3.2), radius 1.0
obstacle at (6, 5), radius 1.0
obstacle at (3.9, 5.1), radius 1.0
obstacle at (7.8,7.6), radius 0.7

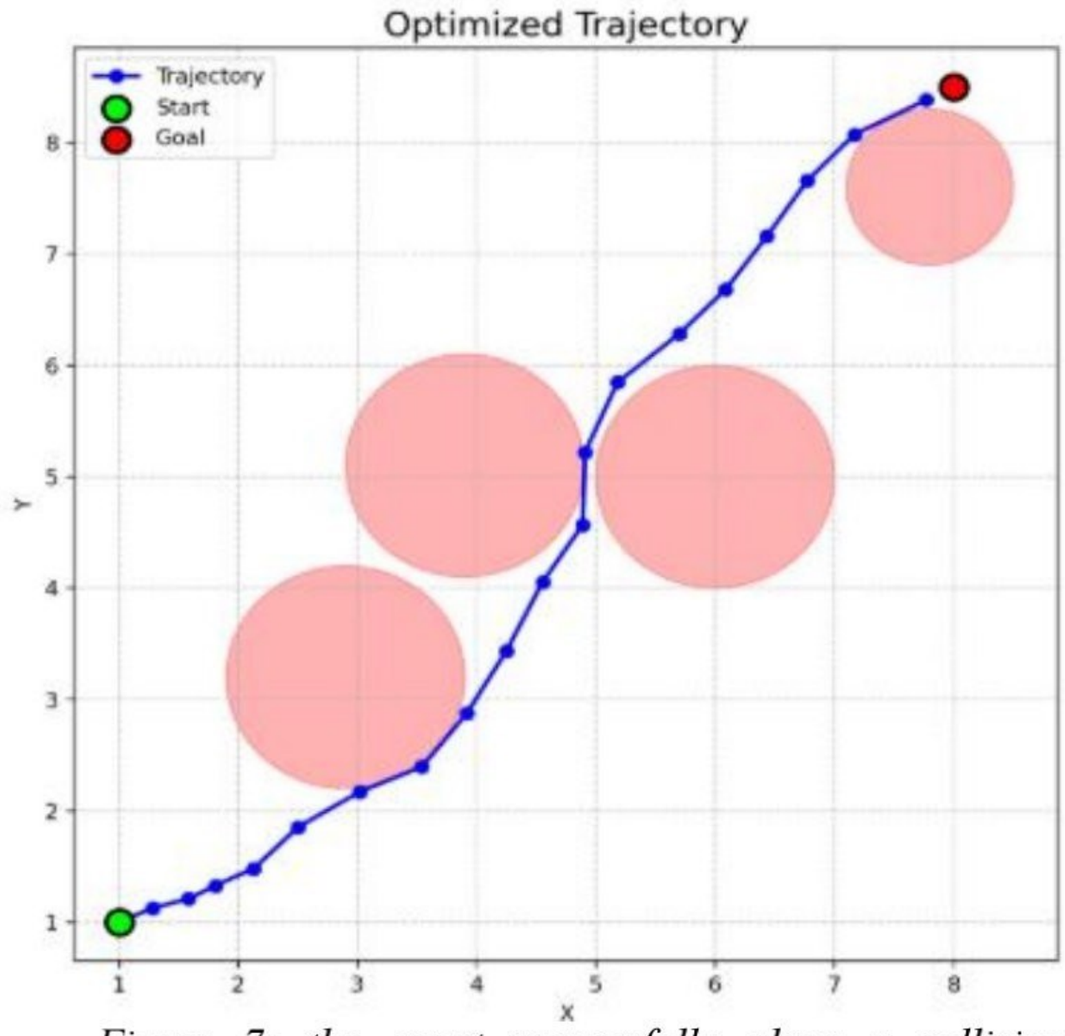
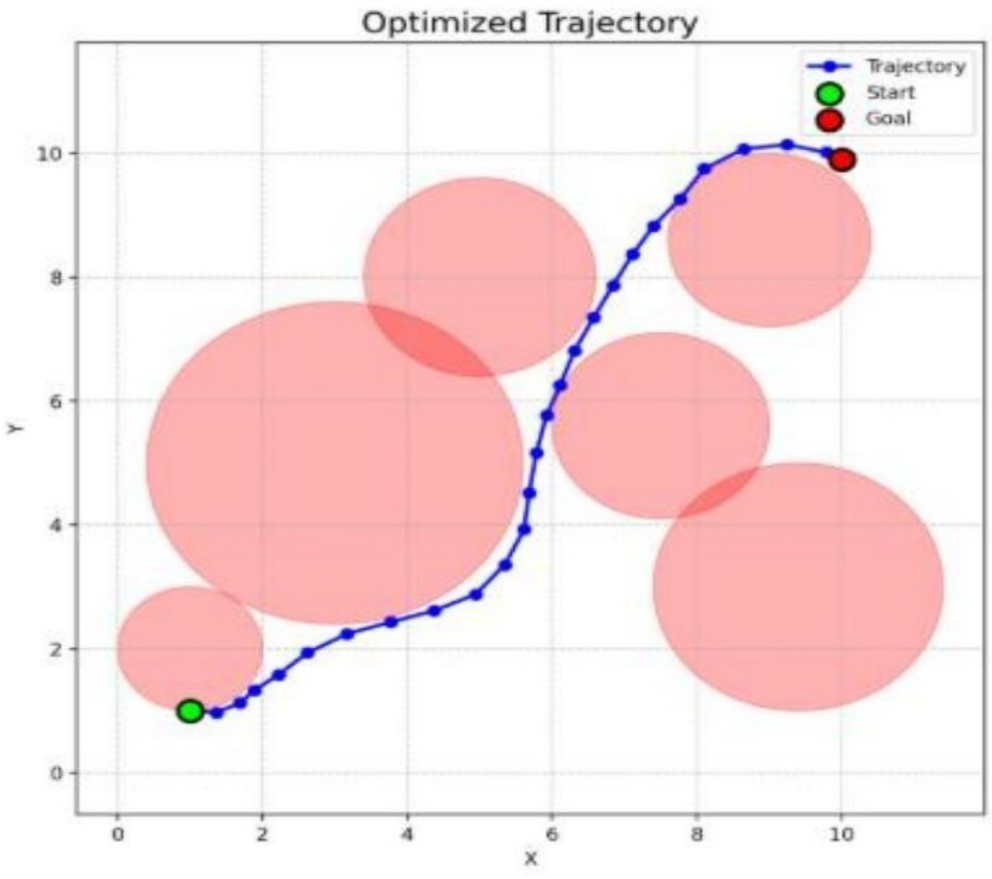


Figure 7: the agent successfully plans a collision-free trajectory, avoiding all three obstacles while reaching the goal.

Test Case-4: Multiple Object with different size

Input Values:
start position: (1,1)
goal position: (9,9)
Obstacles: obstacle at (1,2), radius 1.0
obstacle at (3, 5), radius 2.6
obstacle at (5,8), radius 1.6
obstacle at (7.5, 5.6), radius 1.5
obstacle at (9, 8.6), radius 1.4



Index in Cosmos
JUNE 2025, Volume 15, ISSUE 2
UGC Approved Journal

www.ijbar.org
ISSN 2249-3352 (P) 2278-0505 (E)
Cosmos Impact Factor-5.86

IV. CONCLUSION AND FUTURE SCOPE

This project introduced a novel framework that integrates **Model Predictive Control (MPC)** with a **self-supervised autoencoder** to improve safety in trajectory planning problems. By combining MPC's strength in generating optimal, constraint-aware trajectories with the autoencoder's ability to learn meaningful state representations without supervision, the proposed system enables reliable motion planning even under complex spatial constraints. A lightweight GUI built using **Streamlit** supports interactive experimentation, allowing users to fine-tune hyperparameters such as the learning rate and number of training iterations.

The core strength of this system lies in its modular design and clean separation of planning and learning components. This makes it highly adaptable for further research in safe trajectory planning under uncertainty, nonlinear dynamics, or more complex task objectives. Researchers can easily replace or enhance individual modules—such as swapping in a different encoder architecture or using alternative control cost functions—without overhauling the entire pipeline.

Looking forward, the system has strong potential for deployment in **semi-autonomous robotic systems operating in structured indoor environments**, including **factories, warehouses, laboratories**, and **research facilities**. These settings benefit greatly from real-time trajectory optimization combined with safety-aware decision-making. The SOMTP planner, guided by MPC, ensures adherence to spatial constraints, while the self-supervised autoencoder introduces a data-driven safety filter, capable of flagging out-of-distribution or risky trajectories.

With additional integration into standard robotics middleware such as **ROS (Robot Operating System)** and the inclusion of **real-time sensor feedback** (e.g., LiDAR, cameras, or depth sensors), this system could be deployed in **dynamic, obstacle-rich environments**. Applications include material transport, autonomous inspection, and collaborative robotic systems where safety and adaptability are critical. Furthermore, adapting the current system to handle moving obstacles or to learn from on-the-fly data could make it suitable for more general autonomous navigation tasks.

Overall, this project lays the groundwork for a practical and extensible safe trajectory planner that bridges the gap between traditional control theory and modern self-supervised learning, with clear applicability in **industrial automation, research robotics, and future intelligent systems**.

V. REFERENCES

[1] Lei Li, Zhurong Jia, Tingting Cheng, and Xinchun Jia, "Optimal Model Predictive Control for Path Tracking of Autonomous Vehicle," Journal of Robotic Systems, vol. 30, no. 6, pp. 987–1001, June 2021.

[2] Zhilei Chen, Shaoping Wang, Biao Yu, Huawei Liang, Bichun Li, and Xiaokun Zheng,*"A Robust Trajectory Planning Method Based on Historical Information for Autonomous Vehicles,"*IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 12, pp. 23645–23657, Dec. 2022.

[3] Suqin He, Chuxiong Hu, Shize Lin, and Yu Zhu, *"An Online Time-Optimal Trajectory Planning Method for Constrained Multi-Axis Trajectory With Guaranteed Feasibility,"* IEEE/ASME Transactions on Mechatronics, vol. 28, no. 1, pp. 127–138, 2023.

[4] Na Dong, Yu Feng, Xue-shuo Han, and Ai-guo Wu, *"An Improved Model-Free Predictive Control Method for Nonlinear Time-Delay Systems,"* IEEE Transactions on Industrial Electronics

[5] Jiung Lee, Sehwan Kim, and Kwangseok Oh, "A model predictive path tracking control algorithm for autonomous vehicles based on self-tuning of control period," *2023 23rd International Conference on Control, Automation and Systems (ICCAS)*, Yeosu, Republic of Korea, Oct. 17–20, 2023, pp

[6] M. G. Zaman, A. R. Madni, and R. S. Tan, "Trajectory Planning and Control in Dynamic Environments: A Review of Model Predictive Control," *Journal of Robotic Systems*, vol. 30, no. 6, pp. 987–1001, June 2021.

[7] A. Smith, P. Johnson, and R. Moore, "Safe Trajectory Planning for Autonomous Vehicles Using Reinforcement Learning and MPC," in *Proceedings of the 2020 IEEE International Conference on Robotics and Automation*, Paris, France, 2020, pp. 2454–2461.

[8] L. Wang and X. Zhang, "A Self-Supervised Learning Approach for Autonomous Navigation in Unknown Environments," *International Journal of Machine Learning and Robotics*, vol. 17, no. 3, pp. 115–130, Mar. 2022.

[9] Ming Li and Yun Li, "Heuristic Data-Driven Adaptive Model Predictive Control Strategy," in *Proceedings of the 2023 28th International Conference on Automation and Computing (ICAC)*, Newcastle upon Tyne, UK, Aug. 30–Sept. 1, 2023, pp.

[10] Y. Liu, J. Xu, and T. Zhao, "Autoencoder-Based Safety Validation for Robotic Path Planning," *IEEE Transactions on Robotics and Automation*, vol. 33, no. 4, pp. 1054–1067, Apr. 2021

[11] Qi, X., Zhang, L., Wang, P., Han, Y., & Lin, W. (2023). Learning-Based Model Predictive Control for Vehicles with Modeling Bias. In *Proceedings of the 2023 42nd Chinese Control Conference (CCC)*.

[12] M. S. Hwang and C. C. Lee, "A Study of Electronic Voting System Using Biometric Authentication," International Journal of Network Security, vol. 10, no. 1, pp. 1–10, Jan.

2010.

[13] Yi Wang, Chun-yan Zhai, and Shu-chen Li, "Model predictive control algorithm with iterative learning compensation for disturbances," in *Proceedings of the 2012 24th Chinese Control and Decision Conference (CCDC)*, Taiyuan, China, May 23–25, 2012, pp

[14] Liu, Yifan, You Wang, and Guang Li. "SOMTP: A Self-Supervised Learning-Based Optimizer for MPC-Based Safe Trajectory Planning Problems in Robotics." *IEEE Robotics and Automation Letters*, vol. 9, no. 11, Nov. 2024, pp. 9247–9254

[15] Song, Yunlong, and Davide Scaramuzza. "Learning High-Level Policies for Model Predictive Control." *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 24 Oct. 2020–24 Jan. 2021.

[16] anan Cai, Erqing Zhang, Yali Qi, and Likun Lu, "A review of research on the application of deep reinforcement learning in unmanned aerial vehicle resource allocation and trajectory planning," *2022 4th International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI)*, Shanghai, China,