Methodology Section: A Step-by-Step Plan

This section outlines the comprehensive methodology to be employed in this research, detailing the systematic approach to achieve the stated objectives. The proposed framework integrates a Deep Neural Network (DNN)-guided Probabilistic Roadmap (PRM) for global path planning with the Dynamic Window Approach (DWA) for local reactive control. This hybrid architecture aims to address the limitations of traditional navigation algorithms in dynamic and partially known environments, enhancing efficiency, robustness, and safety in autonomous robot navigation.

1. Overall Research Approach: Hybrid Planning Architecture

The core of this research revolves around developing and evaluating a novel hybrid planning architecture. This architecture synergistically combines the global foresight and path optimality potential of a PRM-based planner with the real-time reactivity and kinematic feasibility of a DWA-based local controller. The key innovation lies in leveraging a Deep Neural Network to intelligently guide the sampling process of the PRM, thereby improving its computational efficiency, particularly in complex and constrained environments. The DWA then utilizes the globally planned path as a reference, ensuring smooth, collision-free, and kinematically feasible execution in dynamic scenarios.

This approach is chosen to capitalize on the strengths of both deliberative (global) and reactive (local) planning paradigms while mitigating their individual weaknesses. The DNN-guided PRM is expected to overcome the computational bottlenecks of traditional PRM by focusing sampling efforts on high-utility regions, while DWA provides the necessary agility to handle unforeseen obstacles and dynamic changes in the immediate environment. The entire system will be developed and rigorously evaluated within a comprehensive simulation framework, allowing for controlled experimentation and quantitative performance assessment against established baselines.

2. Detailed Methodological Steps per Specific Objective

Each specific objective of this thesis necessitates a distinct set of methodological steps, carefully designed to contribute to the overall research goal. These steps are outlined below, providing a clear roadmap for the implementation and evaluation phases of the project.

2.1. Objective 1: Develop and Train a DNN for Sampling Guidance

This objective focuses on the creation and optimization of the intelligent component that enhances the PRM's efficiency. The methodology for this objective involves three critical sub-steps:

2.1.1. Data Generation for DNN Training

To effectively train a Deep Neural Network to guide PRM sampling, a diverse and representative dataset is paramount. This dataset will consist of 2D environment representations (occupancy grids) paired with corresponding

high-utility sampling regions. The process will involve:

- **Environment Generation:** A variety of 2D environments will be programmatically generated, encompassing different complexities such as open spaces, cluttered areas, narrow passages, and environments with static obstacles. These environments will simulate real-world scenarios relevant to autonomous robot navigation, such as warehouses, urban landscapes, or indoor settings. The generation process will ensure sufficient diversity to prevent overfitting and promote generalization of the DNN model.
- **Ground Truth Generation (High-Utility Sampling Regions):** For each generated environment, the 'ground truth' for high-utility sampling regions will be identified. This can be achieved through several approaches:
    - **Expert Knowledge/Heuristics:** Initial identification of critical areas (e.g., bottlenecks, intersections, areas near obstacles) based on expert understanding of PRM behavior and connectivity requirements. This can involve manual annotation or rule-based algorithms.
    - **Exhaustive PRM Runs:** Running a traditional PRM with a very high number of samples in each environment and analyzing the distribution of samples that contribute to successful pathfinding. Regions with a higher density of successful samples would be designated as high-utility. This approach provides a data-driven method for identifying crucial sampling locations.
    - **Graph Connectivity Analysis:** Analyzing the connectivity of the roadmap generated by traditional PRM. Regions that are critical for connecting different components of the free space, or those that bridge disconnected regions, would be considered high-utility. Techniques like graph centrality measures could be employed.

- **Data Representation:** The generated environments will be represented as occupancy grids, where each cell indicates whether it is occupied by an obstacle or free space. The high-utility sampling regions will be represented as probability maps or heatmaps, indicating the likelihood of a sample being useful in that region. This format is suitable for input and output of a convolutional neural network (CNN) architecture.

2.1.2. DNN Architecture Selection and Training

The selection of an appropriate DNN architecture is crucial for effectively mapping occupancy grids to sampling probability distributions. Given the nature of the task, architectures well-suited for image-to-image translation or semantic segmentation are strong candidates. Potential architectures include:

- **U-Net:** A convolutional network architecture originally developed for biomedical image segmentation. Its U-shaped architecture with skip connections allows it to capture both local and global context, making it suitable for generating dense prediction maps like probability distributions. [1]

- **ResNet-FPN (Feature Pyramid Network with ResNet backbone):** ResNet (Residual Network) is known for its ability to train very deep networks, while FPN enhances feature representation across different scales. This combination can be effective for capturing features at various resolutions within the occupancy grid. [2]

- **ViT (Vision Transformer):** While traditionally used for image classification, Vision Transformers have shown promising results in various computer vision tasks. Adapting a ViT for dense prediction tasks might involve using a decoder similar to those in semantic segmentation models. [3]

Training Methodology:

- **Loss Function:** A suitable loss function will be chosen to optimize the DNN. For probability distribution prediction, common choices include:
  - **Binary Cross-Entropy (BCE) Loss:** If the high-utility regions are treated as a binary classification problem (useful/not useful). [4]
  - **Kullback-Leibler (KL) Divergence:** If the output is a probability distribution, KL divergence can measure the difference between the predicted and ground truth distributions. [5]

- **Optimization:** Standard optimization algorithms such as Adam or SGD with momentum will be employed. Learning rate schedules (e.g., cosine annealing, step decay) will be used to ensure stable and efficient training. [6]

- **Regularization:** Techniques like dropout, batch normalization, and L2 regularization will be applied to prevent overfitting and improve generalization performance. [7]
- **Hardware:** Training will be conducted on GPUs (e.g., NVIDIA A100, V100) to accelerate the computational process, given the potentially large dataset and complex model architectures.

2.1.3. Optimization for Efficient Inference

For real-time robot navigation, the trained DNN model must be optimized for efficient inference on robotic platforms, which often have limited computational resources. This involves:

- **Model Quantization:** Reducing the precision of the model's weights and activations (e.g., from float32 to float16 or int8) to decrease memory footprint and accelerate computation without significant loss in accuracy. [8]
- **Model Pruning:** Removing redundant or less important connections/neurons from the network to reduce model size and computational load. [9]
- **Hardware-Aware Optimization:** Utilizing specialized libraries and frameworks (e.g., TensorRT for NVIDIA GPUs, OpenVINO for Intel CPUs) that optimize model execution for specific hardware architectures. [10]
- **Batching:** Processing multiple occupancy grids simultaneously (if applicable) to leverage parallel processing capabilities of GPUs.

References for Objective 1:

[1] Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*.

[2] Lin, T. Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature Pyramid Networks for Object Detection. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[3] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... & Houlsby, N. (2021). An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *International Conference on Learning Representations (ICLR)*.

[4] Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.

[5] Cover, T. M., & Thomas, J. A. (2006). *Elements of Information Theory* (2nd ed.). Wiley-Interscience.

[6] Kingma, D. P., & Ba, J. (2014). Adam: A Method for Stochastic Optimization. *International Conference on Learning Representations (ICLR)*.

[7] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research, 15*, 1929-1958.

[8] Jacob, B., Kligys, M., Chen, B., Zhu, M., Tang, M., Howard, A., ... & Vanhoucke, V. (2018). Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[9] Han, S., Pool, J., Tran, J., & Dally, W. J. (2015). Learning both Weights and Connections for Efficient Neural Networks. *Advances in Neural Information Processing Systems 28 (NIPS 2015)*.

[10] NVIDIA. (n.d.). *NVIDIA TensorRT*. Retrieved from

Intel. (n.d.). *OpenVINO Toolkit*. Retrieved from

2.2. Objective 2: Integrate DNN Guidance into PRM

This objective focuses on incorporating the trained DNN model into the Probabilistic Roadmap (PRM) algorithm to enhance its sampling efficiency and overall performance. The methodology for this objective involves three key sub-steps:

2.2.1. Implementing DNN-Guided PRM Sampling

The traditional PRM algorithm relies on random sampling of configurations within the free space to build a roadmap. This can be inefficient, especially in complex environments with narrow passages or cluttered regions. The DNN-guided approach aims to overcome this by directing the sampling process towards more promising areas. The implementation will involve:

- **DNN Inference Integration:** The trained DNN model will be integrated into the PRM sampling loop. Before generating a new sample, the current occupancy grid of the environment will be fed as input to the DNN. The DNN will output a probability distribution or heatmap indicating the high-utility regions for sampling.

- **Guided Sampling Strategy:** Instead of uniform random sampling, new configurations (nodes) for the PRM roadmap will be sampled probabilistically based on the output of the DNN. Regions with higher probabilities from the DNN output will have a greater chance of being sampled. This can be achieved using techniques like importance sampling or by directly sampling from the predicted probability distribution. This targeted sampling is expected to significantly reduce the number of samples required to achieve good coverage and connectivity of the free space, particularly in challenging areas.

- **Collision Checking:** For each sampled configuration, a collision check will be performed to ensure that the sampled point is in the free space and does not collide with any obstacles. This is a standard PRM component and will be implemented efficiently, potentially using spatial data structures (e.g., k-d trees) for faster nearest neighbor queries during edge connection.

2.2.2. Adaptive Sampling Strategy for Connectivity and Coverage

While DNN guidance improves sampling efficiency, an adaptive sampling strategy is crucial to ensure comprehensive coverage of the free space and robust connectivity of the roadmap, especially in areas where the DNN might be less confident or where connectivity is inherently difficult. This involves:

- **Hybrid Sampling:** A hybrid approach combining DNN-guided sampling with a small percentage of uniform random sampling. This ensures that even if the DNN misses certain critical regions, the uniform sampling component provides a fallback mechanism to explore the entire configuration space. The ratio between guided and random sampling can be dynamically adjusted based on the environment complexity or the progress of roadmap construction.

- **Connectivity-Driven Sampling:** Monitoring the connectivity of the PRM graph. If certain regions or components of the free space remain disconnected, additional samples can be strategically generated in the vicinity of these disconnected components to bridge the gaps. This can involve analyzing the connected components of the graph and focusing sampling efforts at their boundaries.

- **Coverage-Driven Sampling:** Ensuring that the entire free space is adequately covered by the roadmap. This can involve maintaining a grid over the free space and identifying cells that are not yet covered by any node in the PRM. New samples can then be preferentially generated in these uncovered regions to improve the overall coverage of the roadmap.

2.2.3. PRM Graph Construction and Path Search with Smoothing

Once a sufficient number of nodes are sampled and validated (collision-free), the PRM graph needs to be constructed, and an efficient path search algorithm applied. Finally, the generated path will be smoothed for practical robot execution. This involves:

- **Graph Construction:** For each sampled node, connections (edges) will be attempted to its k-nearest neighbors (or neighbors within a certain radius) in the free space. An edge is added if the straight line path between the two nodes is collision-free. The choice of 'k' or the radius will be determined empirically to balance graph density and computational cost.

- **Efficient Path Search (A*):** Once the PRM graph is constructed, a path from the start configuration to the goal configuration will be found using an efficient graph search algorithm. The A *algorithm is a suitable choice due to its optimality and efficiency in finding the shortest path in a graph. The cost function for A* will consider both the Euclidean distance between nodes and potentially other factors like clearance from obstacles.

- **Path Smoothing:** The path generated by PRM and A* can often be jagged or non-smooth, which is undesirable for robot motion. Path smoothing techniques will be applied to generate a more continuous and kinematically feasible trajectory. This can involve:

  - **B-spline or Cubic Spline Interpolation:** Fitting a smooth curve through the waypoints of the generated path. [11]
  - **Shortcutting:** Iteratively attempting to connect non-adjacent nodes in the path with a straight line if the segment is collision-free, thereby removing redundant intermediate waypoints. [12]
  - **Optimization-based Smoothing:** Formulating path smoothing as an optimization problem to minimize path length, curvature, and maximize clearance from obstacles, subject to kinematic constraints. [13]

References for Objective 2:

[11] Farin, G. (2002). *Curves and Surfaces for CAGD: A Practical Guide* (5th ed.). Morgan Kaufmann.

[12] Kavraki, L. E., Svestka, P., Latombe, J. C., & Overmars, M. H. (1996). Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces. *IEEE Transactions on Robotics and Automation, 12*(3), 402-41Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces. *IEEE Transactions on Robotics and Automation, 12*(3), 402-413.

[13] Ratliff, N., Zucker, M., Bagnell, J. A., & Srinivasa, S. (2009). CHOMP: Gradient Optimization Techniques for High-Dimensional Robot Navigation. *2009 IEEE International Conference on Robotics and Automation*.

2.3. Objective 3: Implement and Tune the DWA Local Planner

This objective focuses on the implementation and meticulous tuning of the Dynamic Window Approach (DWA) local planner, ensuring its effectiveness in real-time obstacle avoidance and adherence to robot kinematic constraints. The methodology for this objective involves three critical sub-steps:

2.3.1. DWA Algorithm Implementation with Kinematic Constraints

The Dynamic Window Approach (DWA) is a widely used local navigation algorithm that samples possible robot velocities (linear and angular) within a dynamic window, which is constrained by the robot's kinematic capabilities and environmental obstacles. The implementation will involve:

- **Robot Kinematic Model:** Accurately modeling the robot's kinematics (e.g., differential drive, omnidirectional) to define the feasible velocity space. This involves understanding the maximum linear and angular velocities, as well as acceleration limits. The dynamic window is then derived from these constraints, considering the time horizon for prediction. [14]

- **Trajectory Prediction:** For each sampled velocity pair (v, ω) within the dynamic window, predict the robot's trajectory over a short time horizon. This prediction should account for the robot's current state and its kinematic limitations. The predicted trajectories will be used to evaluate potential collisions and assess motion quality. [15]

- **Obstacle Detection and Representation:** Integrating sensor data (e.g., lidar, depth camera) to detect local obstacles in the robot's immediate vicinity. These obstacles will be represented in a format suitable for collision checking, such as point clouds or local occupancy grids. The DWA will continuously update its understanding of the local environment to react to dynamic obstacles.

2.3.2. Systematic Tuning of DWA Parameters and Cost Function Design

The performance of DWA is highly dependent on its parameters and the design of its cost function. Systematic tuning is essential to achieve an optimal balance between path tracking, obstacle avoidance, and motion quality. This involves:

- **Cost Function Components:** The DWA cost function typically comprises several components, each weighted to reflect its importance:
    - **Heading/Goal Alignment:** A term that penalizes trajectories that deviate significantly from the desired heading towards the global goal or reference path. This ensures the robot progresses towards its objective. [16]
    - **Obstacle Clearance:** A term that maximizes the distance to the closest obstacle, ensuring safe navigation and preventing collisions. This component is critical for reactive obstacle avoidance. [17]
    - **Velocity/Speed:** A term that encourages higher velocities to reach the goal faster, while respecting maximum speed limits. This promotes efficient navigation. [18]
    - **Path Following (for hybrid integration):** A new component will be introduced or heavily weighted to ensure the DWA closely follows the global path provided by the DNN-guided PRM. This term will guide the local planner to adhere to the global plan while still allowing for local deviations to avoid immediate obstacles.
- **Parameter Tuning:** The weights associated with each component of the cost function, as well as other DWA parameters (e.g., prediction horizon, angular and linear acceleration limits), will be systematically tuned. This can be done through:
    - **Grid Search/Random Search:** Exploring a range of parameter values in simulation and evaluating their impact on performance metrics (e.g., success rate, path length, smoothness, time to goal). [19]
    - **Optimization Algorithms:** Employing optimization techniques (e.g., Bayesian optimization, genetic algorithms) to automatically find optimal parameter sets that maximize a defined performance objective. [20]
    - **Empirical Tuning:** Iterative manual adjustment of parameters based on observed behavior in various simulated scenarios, leveraging expert intuition.

2.3.3. Integration of Mechanisms for Handling Local Dynamic Obstacles

One of the key advantages of DWA is its ability to react to dynamic obstacles. The implementation will include specific mechanisms to effectively handle such scenarios:

- **Dynamic Obstacle Prediction:** While DWA primarily reacts to the current state of obstacles, incorporating simple prediction models for dynamic obstacles (e.g., constant velocity model) can improve its anticipatory capabilities. This allows the DWA to select velocities that avoid future collisions with moving objects. [21]
- **Time-to-Collision (TTC) Consideration:** The cost function can be augmented to include a term that penalizes trajectories with a low time-to-collision, prioritizing safety when approaching dynamic obstacles. [22]
- **Prioritization of Avoidance:** Ensuring that obstacle avoidance takes precedence over path following when an imminent collision with a dynamic obstacle is detected. This might involve dynamically adjusting the weights of the cost function components or implementing specific emergency maneuvers.

References for Objective 3:

[14] Fox, D., Burgard, W., & Thrun, S. (1997). The Dynamic Window Approach to Collision Avoidance. *IEEE Robotics & Automation Magazine, 4*(1), 23-33.

[15] Arkin, R. C. (1998). *Behavior-Based Robotics*. MIT Press.

[16] Brock, O., & Khatib, O. (2002). High-Speed Navigation Using the Global Dynamic Window Approach. *Proceedings 2002 IEEE International Conference on Robotics and Automation (ICRA)*.

[17] Minguez, J., & Montano, L. (2004). Nearness Diagram Navigation for Mobile Robots. *Robotics and Autonomous Systems, 47*(1), 17-39.

[18] Seder, M., & Petrovic, I. (2012). Dynamic Window Approach with Adaptive Parameters for Mobile Robot Navigation. *2012 IEEE International Conference on Robotics and Automation (ICRA)*. Bergstra, J., & Bengio, Y. (2012). Random Search for Hyper-Parameter Optimization. *Journal of Machine Learning Research, 13*, 281-305.

[19] Snoek, J., Larochelle, H., & Adams, R. P. (2012). Practical Bayesian Optimization of Machine Learning Algorithms. *Advances in Neural Information Processing Systems 25 (NIPS 2012)*.

[20] Trautman, P., & Ma, Y. (2010). Robot Navigation in Crowds: Towards a Socially Aware and Collision-Free Approach. *2010 IEEE International Conference on Robotics and Automation (ICRA)*.

[21] Fraichard, T., & Asfour, T. (2000). A Time-to-Collision Based Approach to Obstacle Avoidance. *Proceedings 2000 IEEE International Conference on Robotics and Automation (ICRA)*.

2.4. Objective 4: Integrate the Global and Local Planners

This objective focuses on seamlessly integrating the DNN-guided PRM global planner with the DWA local planner to form a cohesive and robust hybrid navigation system. The methodology for this objective involves two crucial sub-steps:

2.4.1. Hierarchical Control Architecture Development

The integration of global and local planners will be achieved through a hierarchical control architecture. This architecture defines the roles and interactions between the two planning layers, ensuring that the local planner effectively follows the global guidance while maintaining reactive capabilities. The development will involve:

- **Global Path Reference:** The DNN-guided PRM will generate a globally optimal and collision-free path from the robot's current position to the target destination. This path will serve as the primary reference for the DWA local planner. The global path will be represented as a sequence of waypoints or a continuous trajectory.

- **Local Path Following:** The DWA local planner will operate in a reactive loop, continuously receiving the global path reference and the current robot state. Its primary task will be to generate kinematically feasible and collision-free trajectories that closely follow the global path while avoiding immediate obstacles (both static and dynamic) in its local sensing range. The cost function of the DWA will be heavily weighted towards path following, as discussed in Objective 3.

- **Coordinate Transformation and Synchronization:** Ensuring seamless communication and data exchange between the global and local planning layers. This involves proper coordinate transformations between the global map frame and the robot's local frame, as well as synchronization of planning cycles to ensure the DWA always has an up-to-date global path reference. [23]

- **State Machine for Navigation:** Implementing a state machine to manage the overall navigation process. This state machine will handle transitions between different navigation states (e.g., following global path, obstacle avoidance, replanning) and coordinate the actions of the global and local planners.

2.4.2. Robust Replanning Logic and Recovery Behaviors

Even with a robust hybrid architecture, unforeseen circumstances (e.g., significant environmental changes, persistent obstacles, planner failures) can necessitate replanning or trigger recovery behaviors. This sub-step focuses on developing mechanisms to handle such situations gracefully:

- **Trigger Conditions for Replanning:** Defining clear criteria that trigger a global replanning event. These conditions could include:
    - **Significant Deviation from Global Path:** If the robot's actual trajectory deviates beyond a predefined threshold from the global path, indicating that the local planner is struggling to follow the plan or an unexpected obstacle is blocking the way. [24]
    - **Unforeseen Obstacles:** Detection of new, persistent obstacles that cannot be locally avoided by the DWA, effectively blocking the current global path.
    - **Planner Failure:** If either the global or local planner reports an internal error or fails to find a valid solution within a certain time limit.
    - **Dynamic Environment Changes:** If the environment undergoes significant changes (e.g., a large object moves into the robot's path, a new area becomes accessible).
- **Replanning Strategy:** When a replanning event is triggered, the system will initiate a new global path planning request to the DNN-guided PRM, using the robot's current position as the new start point and the original destination as the goal. The system will prioritize finding a new valid path quickly.
- **Recovery Behaviors:** Implementing specific recovery behaviors to handle situations where immediate replanning is not feasible or sufficient. These behaviors could include:
    - **Stop and Wait:** If the robot encounters an insurmountable obstacle or a critical error, it will stop safely and wait for a new plan or human intervention.
    - **Backtracking/Local Exploration:** If stuck in a local minimum or a dead-end, the robot might attempt to backtrack a short distance or perform a limited local exploration to find an alternative route. [25]
    - **Emergency Braking:** In case of imminent collision, an emergency braking mechanism will override all other controls to prevent damage.

References for Objective 4:

[22] Siegwart, R., Nourbakhsh, I. R., & Scaramuzza, D. (2011). *Introduction to Autonomous Mobile Robots* (2nd ed.). MIT Press.

[23] Thrun, S., Burgard, W., & Fox, D. (2005). *Probabilistic Robotics*. MIT Press.

[24] LaValle, S. M. (2006). *Planning Algorithms*. Cambridge University Press.

2.5. Objective 5: Evaluate System Performance

This objective focuses on rigorously evaluating the performance of the proposed hybrid navigation framework. A comprehensive evaluation strategy is crucial to demonstrate the advantages of the DNN-guided PRM with DWA system over traditional methods. The methodology for this objective involves four key sub-steps:

2.5.1. Establishing a Comprehensive Simulation Framework

All development and evaluation will be conducted within a robust and flexible simulation environment. Simulation offers a controlled and repeatable setting for testing complex robotic systems, allowing for extensive experimentation without the risks and costs associated with physical hardware. The simulation framework will encompass:

- **Simulation Environment Selection:** A suitable robotics simulator (e.g., Gazebo, Webots, PyBullet, or a custom-built 2D simulator) will be chosen. The selection criteria will include its ability to accurately model robot kinematics and dynamics, simulate various sensor types (e.g., lidar for occupancy grid generation), and provide tools for creating diverse 2D environments. The simulator must support the creation of static, dynamic, cluttered, and narrow-passage scenarios. [26]

- **Environment Generation:** A library of diverse 2D environments will be created within the chosen simulator. These environments will represent a range of complexities and challenges, including:
  - **Static Environments:** Simple open spaces, environments with fixed obstacles (e.g., walls, furniture), and mazes.
  - **Dynamic Environments:** Environments with moving obstacles (e.g., other robots, pedestrians) with varying speeds and trajectories.
  - **Cluttered Environments:** Spaces with a high density of obstacles, requiring sophisticated collision avoidance.
  - **Narrow Passages:** Constrained areas that test the planner's ability to navigate through tight spaces efficiently.

- **Robot Model Integration:** The robot model used in the simulation will accurately reflect the kinematic and dynamic properties of the target autonomous mobile robot. This includes its dimensions, maximum velocities, and acceleration limits, which are crucial for the DWA component.
- **Sensor Simulation:** Realistic sensor models (e.g., simulated lidar or range sensors) will be integrated to provide the occupancy grid maps and local obstacle information required by the planning algorithms. The sensor models will account for noise and limitations to reflect real-world conditions.

2.5.2. Quantitative Performance Metrics

To objectively assess the performance of the proposed system, a set of quantitative metrics will be defined and measured across all test scenarios. These metrics will provide a comprehensive understanding of the system's efficiency, success, safety, and resource utilization. Key metrics include:

- **Planning Efficiency:**
  - **Path Planning Time:** The time taken by the global planner (DNN-guided PRM) to find a path from start to goal. This will be a critical metric to demonstrate the efficiency gains from DNN guidance. [27]
  - **Number of Samples (for PRM):** The total number of nodes sampled by the PRM to construct a connected roadmap. A lower number indicates higher sampling efficiency.
  - **Computational Load (CPU/GPU Usage):** Monitoring the computational resources consumed by both the global and local planners during operation.
- **Navigation Success and Quality:**
  - **Success Rate:** The percentage of trials in which the robot successfully reaches the goal without collisions or getting stuck. [28]
  - **Path Length:** The total length of the trajectory executed by the robot. Shorter paths are generally preferred.
  - **Path Smoothness/Jerk:** Metrics to quantify the smoothness of the robot's trajectory, indicating comfortable and energy-efficient motion. Lower jerk values are desirable. [29]
  - **Time to Goal:** The total time taken for the robot to reach its destination.
- **Safety and Dynamic Handling:**
  - **Number of Collisions:** The count of instances where the robot collides with static or dynamic obstacles. This should ideally be zero. [30]

   - **Minimum Distance to Obstacles:** The closest distance the robot maintains to obstacles during navigation. Higher values indicate safer navigation.
   - **Dynamic Obstacle Avoidance Success:** Specific metrics for scenarios with moving obstacles, such as the ability to successfully navigate through dynamic crowds or avoid fast-moving objects.

- **Resource Utilization:**

   - **Memory Footprint:** The amount of memory consumed by the planning algorithms.

2.5.3. Statistical Comparison Against Baseline Navigation Methods

To validate the superiority of the proposed hybrid framework, its performance will be statistically compared against several established baseline navigation methods. This comparative analysis will highlight the specific advantages of the DNN-guided PRM with DWA. Baseline methods will include:

- **Traditional PRM + DWA:** A standard implementation of PRM coupled with DWA, without the DNN guidance for sampling. This will serve as a direct comparison to quantify the impact of the DNN. [31]
- **RRT* + DWA:** The Rapidly-exploring Random Tree Star (RRT*) algorithm, known for its asymptotic optimality, coupled with DWA. This provides a comparison against another advanced sampling-based global planner. [32]
- **A* + DWA:** A grid-based global planner (A*) coupled with DWA. This represents a common and effective combination for known environments. [33]
- **TEB (Timed Elastic Band):** A local trajectory optimization approach that can handle dynamic environments and kinematic constraints. This provides a comparison against a purely local optimization method. [34]

For each metric, statistical tests (e.g., t-tests, ANOVA) will be performed to determine the statistical significance of any observed performance differences between the proposed system and the baseline methods. This will ensure that the conclusions drawn from the evaluation are robust and reliable.

2.5.4. Robustness Assessment to Sensor Noise and Parameter Variations

Real-world environments are inherently noisy and uncertain. Therefore, assessing the robustness of the proposed system to sensor noise and variations in its internal parameters is crucial. This will involve:

- **Sensor Noise Injection:** Introducing simulated noise into the sensor readings (e.g., Gaussian noise to lidar measurements, localization errors) to evaluate how the system performs under imperfect sensing conditions. [35]

- **Parameter Sensitivity Analysis:** Systematically varying key parameters of the DNN-guided PRM (e.g., DNN confidence threshold for sampling, PRM connection radius) and DWA (e.g., cost function weights, prediction horizon) to understand their impact on overall system performance. This will help identify critical parameters and their optimal ranges.
- **Adversarial Scenarios:** Designing specific challenging scenarios (e.g., sudden appearance of obstacles, highly dynamic environments) to test the system's ability to maintain performance under stress.

References for Objective 5:

[25] Koenig, N., & Howard, A. (2004). Gazebo Physics Simulator. *IEEE International Conference on Robotics and Automation (ICRA)*.

[26] Kuffner, J. J., & LaValle, S. M. (2000). RRT-Connect: An Efficient Approach to Single-Query Path Planning. *IEEE International Conference on Robotics and Automation (ICRA)*.

[27] Macfarlane, S., & Croft, E. A. (2018). Metrics for Evaluating Robot Navigation Performance: A Survey. *Robotics and Autonomous Systems, 104*, 1-15.

[28] Schulman, J., Moritz, P., Levine, S., Jordan, M. I., & Abbeel, P. (2015). High-Dimensional Continuous Control Using Generalized Advantage Estimation. *International Conference on Learning Representations (ICLR)*.

[29] Ferrer, J., & Sanfeliu, A. (2017). A Survey on Collision Avoidance for Mobile Robots. *Robotics and Autonomous Systems, 97*, 1-19.

[30] LaValle, S. M. (2006). *Planning Algorithms*. Cambridge University Press.

[31] Karaman, S., & Frazzoli, E. (2011). Sampling-based Algorithms for Optimal Motion Planning. *International Journal of Robotics Research, 30*(7), 846-894.

[32] Hart, P. E., Nilsson, N. J., & Raphael, B. (1968). A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics, 4*(2), 100-107.

[33] Rösmann, C., Feiten, W., Wösch, T., Hoffmann, F., & Bertram, T. (2017). Trajectory Planning for Mobile Robots with Kinematic and Dynamic Constraints in the Timed Elastic

Band Approach. *International Journal of Robotics Research, 36*(12), 1244-1261.

[34] Thrun, S., Burgard, W., & Fox, D. (2005). *Probabilistic Robotics*. MIT Press.