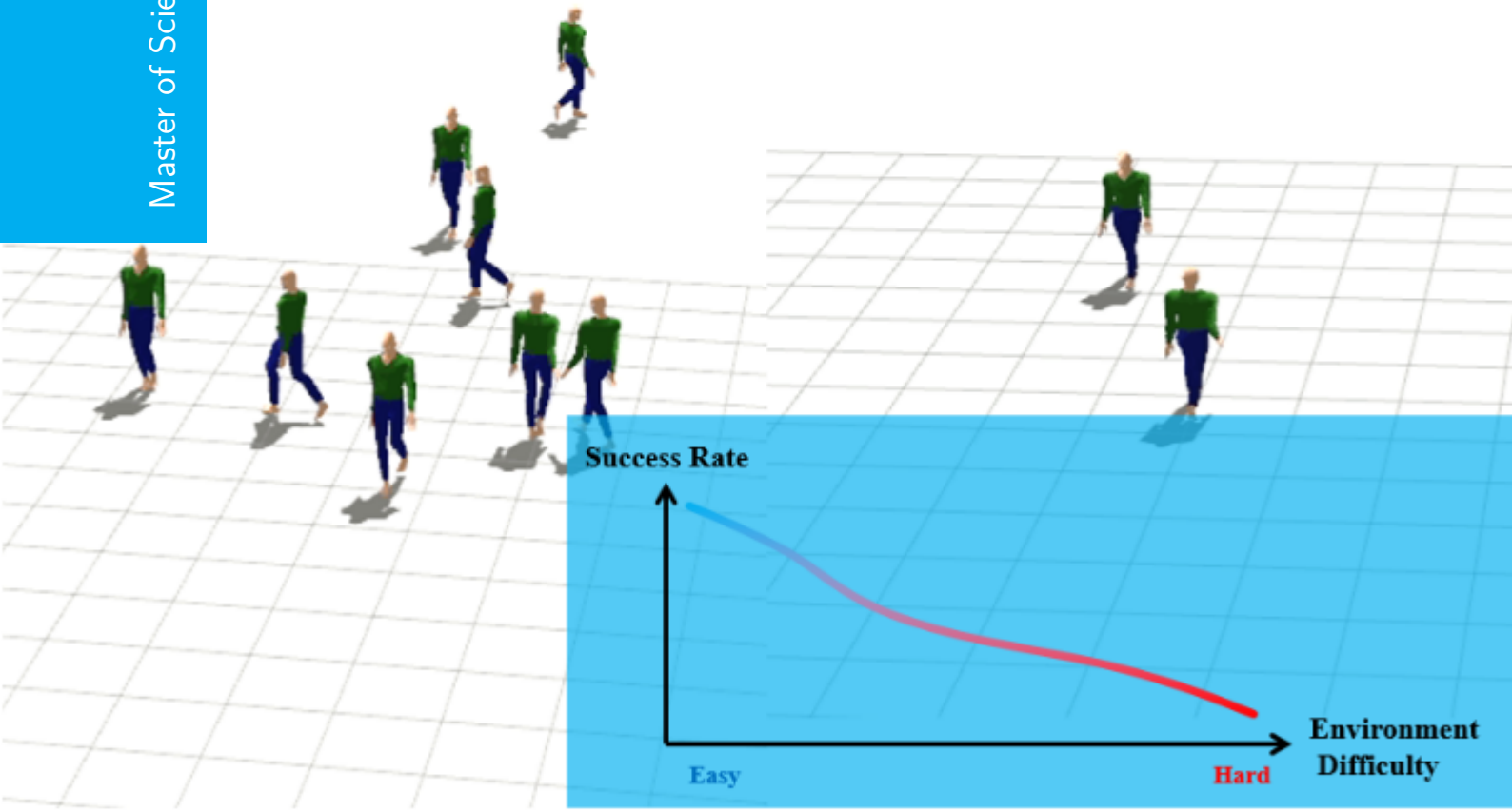
Evaluating Dynamic Environment Difficulty for Obstacle Avoidance Benchmarking

Moji Shi

Master of Science Thesis



TU Delft Delft University of Technology

Cognitive Robotics

Evaluating Dynamic Environment Difficulty for Obstacle Avoidance Benchmarking

Master of Science Thesis

For the degree of Master of Science in Robotics at Delft University of Technology

Moji Shi

August 25, 2023

Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of Technology

Cognitive
Robotics

Copyright © Cognitive Robotics (CoR)
All rights reserved.

Delft University of Technology
Department of
Cognitive Robotics (CoR)

The undersigned hereby certify that they have read and recommend to the Faculty of Mechanical, Maritime and Materials Engineering (3mE) for acceptance a thesis entitled

Evaluating Dynamic Environment Difficulty for Obstacle Avoidance Benchmarking

by

Moji Shi

in partial fulfillment of the requirements for the degree of

Master of Science.

Dated: August 25, 2023

Supervisor:

Dr. Javier Alonso-Mora, Dr. Gang Chen

Readers:

Prof.dr.ir. Martijn Wisse

Dr. Salua Hamaza

Abstract

Dynamic obstacle avoidance remains a crucial research area for autonomous systems, such as Micro Aerial Vehicles (MAVs) and service robots. Efforts to develop dynamic collision avoidance techniques in unknown environments have proliferated in recent years. While these methods exhibit impressive and reliable performance in simpler environments, their efficacy in more challenging settings remains an area ripe for enhancement. The difficulty of these environments arises from a multitude of factors, and currently, no standardized approach exists to quantify this complexity. Additionally, to fairly compare different dynamic collision avoidance strategies, it's essential to assess them in environments with a similar degree of difficulty. Therefore, devising a metric capable of accurately gauging the intricacy of dynamic environments becomes imperative.

Building on this context, this master's thesis endeavors to fill this critical gap through three contributions: 1) The establishment and validation of map difficulty metrics that represent the difficulty of dynamic environments, 2) The introduction of a robust benchmarking pipeline to critically validate the representativeness of the proposed metrics and evaluate various collision avoidance strategies, and 3) The provision of a framework for comparative analysis of different planning strategies, utilizing the introduced map difficulty metric.

The proposed survivability metric effectively captures environmental complexity. Its validity is evidenced by a notable correlation with the success rates of typical collision avoidance methods, with over 1.7 million collision avoidance trials on over six hundred maps, securing a Spearman's Rank correlation coefficient (SRCC) of over 0.9. This metric serves as an indispensable tool for facilitating fair comparisons in this dynamic research domain. More importantly, it offers valuable insights for the future refinement and improvement of dynamic collision avoidance strategies, making a contribution to the continuous advancement of autonomous systems.

Table of Contents

List of Figures

List of Tables

Acknowledgements

I deeply thank Prof. Javier Alonso Mora, my main supervisor. His overarching vision and general guidance provided the foundation for this research. His expertise laid down the path I needed to follow, and his unwavering trust in my capabilities was a constant source of motivation.

I am profoundly grateful to Dr. Gang Chen, my daily supervisor, who has been an anchor throughout this journey. His dedication to this work is unparalleled. The countless hours we spent discussing the intricacies of the research, his patient mentorship, and his meticulous attention to every detail of my writing and presentations were invaluable. His commitment to excellence was both a challenge and an inspiration, pushing me to achieve more than I thought possible.

Your feedback and collaboration have been indispensable to all the faculty members, peers, and colleagues who played a direct or indirect role in this research.

Finally, I extend my heartfelt appreciation to my family and friends. Their unwavering belief in me and constant support have been pillars of strength throughout my academic journey.

Moji Shi

Delft, University of Technology,
August 25, 2023

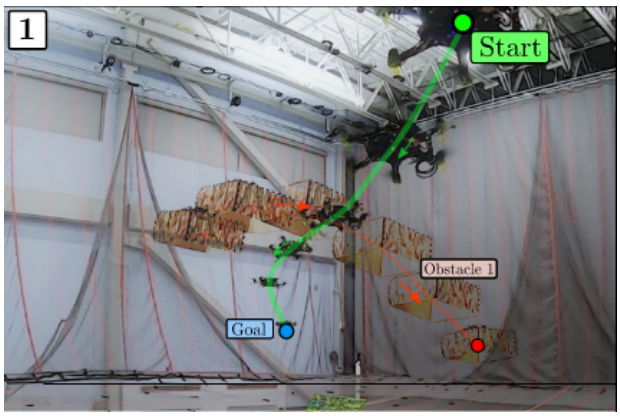
Chapter 1

Introduction

1-1 Background and Objective

Recently, societal anticipation and expectations regarding the advancement of autonomous systems have surged considerably. Various solutions are proposed to substitute human involvement in labor-intensive and dangerous tasks, including urban and suburban package delivery [1] and search-rescue operations [2]. During operational execution, these systems may encounter dynamic obstacles [3, 4], the details of which may not be pre-known to the autonomous agents. This situation highlights the critical requirement for robust dynamic collision avoidance systems, proficient in adapting to many environmental circumstances and challenges, and an intensified scholarly focus has been put on this particular domain.

(a) A ground robot trying to navigate through a crowd of pedestrians [5].



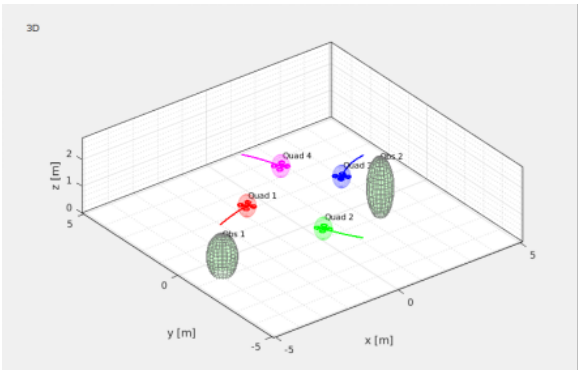
(b) A MAV trying to navigate through an environment with other dynamic obstacles [4].

Figure 1-1: Examples of dynamic collision avoidance scenarios of autonomous systems.

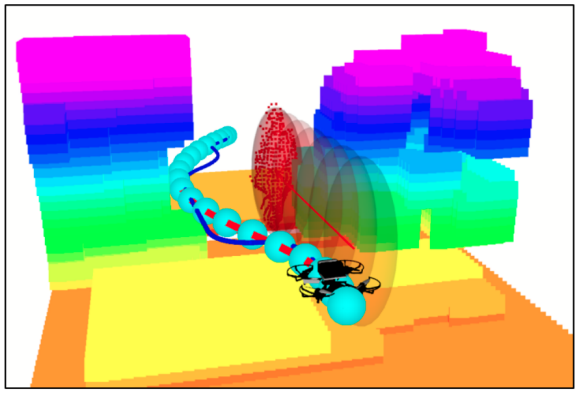
One crucial aspect of developing such collision avoidance methods involves performance evaluation. In domains such as computer vision (CV) and natural language processing (NLP), standardized datasets and metrics, such as KITTI [6], ImageNet [7], COCO [8], GLUE [9], and SQuAD [10] have been established to facilitate this process. These resources, termed benchmarks, offer standardized datasets and evaluation protocols, enabling comprehensive comparisons between varying methodologies.

However, a dearth of such standardized performance evaluation mechanisms currently exists for dynamic collision avoidance. Instead, researchers handcraft their custom testing environment for evaluation like what Figure 1-2 shows.

(a) Gazebo Simulation with pedestrians moving in straight lines [11]



(b) Matlab Simulation [12]



(c) rviz simulation environment with obstacles moving along preset trajectories [13]

Figure 1-2: Examples of three different environment setups. These various setups make comparing proposed methods difficult.

This approach is problematic without indicating how difficult the environment is for dynamic collision avoidance. An **environment difficulty metric** is urgently required in the assessment of a collision avoidance method's performance for several reasons:

1. **Fair Comparison among Different Collision Avoidance methods:** Testing different collision avoidance methods in equivalent setups is pivotal to ensure a fair comparison. Presently, numerous techniques are implemented across various simulations. Drawing fair comparisons among these methods remains challenging without a quantified indicator of environmental difficulty. Incorporating an environmental difficulty metric paves the way for a more balanced comparison of disparate method performances, thereby enhancing the overall accuracy and reliability of the comparative process.

2. **Evaluation of Collision Avoidance Method Under Different Difficulty Level:** The environmental difficulty significantly affects the method's performance. A system demonstrating flawless collision avoidance in a straightforward environment might encounter problems in a more complex one.

3. **Incremental Refinement of Collision Avoidance Methods:** An environment difficulty metric also assists in incremental method refinement. For example, researchers can test their method in a simple environment and gradually increase the difficulty level. This approach promotes more efficient benchmarking of progress and illuminates areas requiring additional development.

To design such a representative environment difficulty metric, we need to investigate the relationship between the metrics and the performance of various collision avoidance methods. Specifically, each collision avoidance method should correspond to a lower success rate in environments that score higher on the difficulty metric. Considering a typical dynamic collision avoidance pipeline depicted in Figure 1-3, the success rate of an avoidance maneuver depends not only on the external factors (the difficulty of the environment) but also on the internal factors inherent in the entire navigation pipeline. These internal factors include perception, prediction, localization, planning, and control.

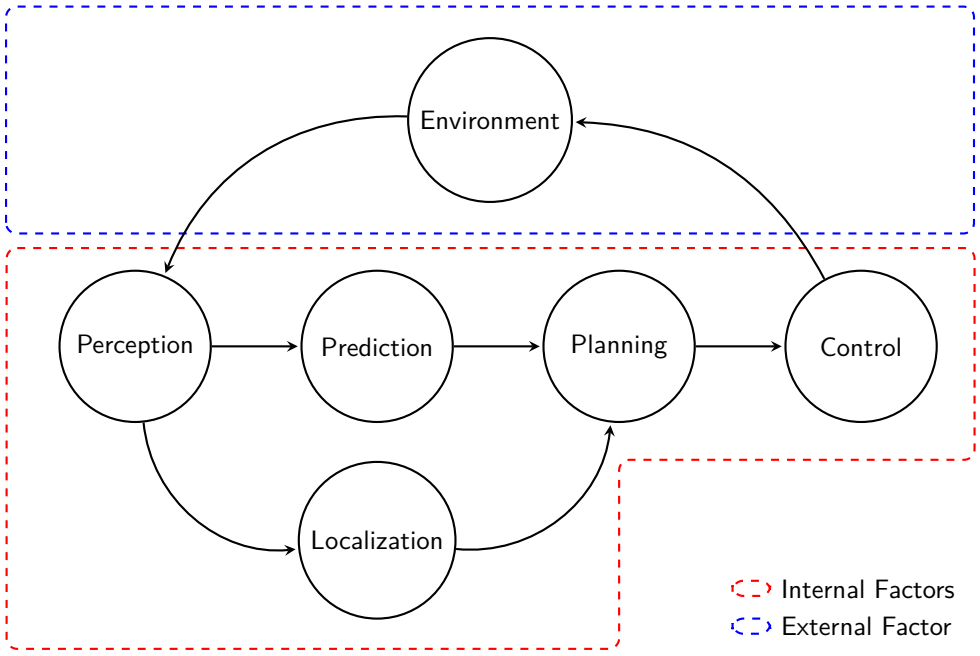


Figure 1-3: A typical dynamic collision avoidance pipeline. Dashed lines encircle the internal (Perception, Prediction, Localization, Planning, and Control) and external (Environment) factors. The legend describes the color and pattern of each dashed line box.

The internal factors can be categorized into those we assess and those we keep constant. For instance, the components of perception and control are typically held constant, as they are dependent on the hardware of the robots or algorithmic design choices that

are not under assessment. The planning part we are evaluating would be varied. In contrast, the difficulty of the environment is the external factor, and this is the difficulty that our proposed metric is intended to measure.

Therefore, our approach calls for the design of a specialized simulator, which effectively isolates the influence of the environmental difficulty metric on the success rate of various collision avoidance methods. This design should effectively minimize the impact of extraneous factors, thereby allowing for a more accurate and controlled analysis of the performance of collision avoidance methods in relation to environmental difficulty.

In this master thesis, we will propose environment difficulty metrics for benchmarking dynamic collision avoidance methods and validate whether they are representative of the difficulty by experimenting in a custom simulator with typical collision avoidance methods.

1-2 Related Works

1-2-1 Collision avoidance in unknown dynamic environments

Collision avoidance for Micro Aerial Vehicles (MAVs) operating in dynamic and unknown environments remains a persistent challenge in the field of robotics. Various methodologies have been proposed in recent years to address this problem.

Some works assume the future movement of dynamic obstacles is known to the MAV in advance and focus on the trajectory generation part. Sampling-based methods, such as those proposed by [14, 15, 16], generate multiple candidate trajectories by sampling motion primitives. Among these, the candidate with the minimum cost is selected and concatenated to form the final trajectory. Notably, [17] proposed a novel strategy in which the trajectory is generated iteratively towards a sub-goal rather than in a one-shot manner. The authors in [15] also incorporate the MAV's dynamics into their model by generating motion primitives from thrust. On the other hand, optimization-based methods, such as those in [12, 18, 19], cast the collision avoidance problem into an optimization framework. These methods define collision avoidance constraints and solve an optimization problem to generate the desired trajectories. For instance, authors in [12] model the obstacles as ellipsoids and express the collision avoidance constraints as the distance between these ellipsoids. Another noteworthy approach is by [20], where the authors formulate collision avoidance constraints using convex safety corridors. In recent years, reinforcement learning-based methods have also been applied to this problem [21, 22, 23, 5]. This represents an exciting direction in collision avoidance and opens up new possibilities for handling dynamic and unknown environments.

Several other studies have challenged the assumption regarding the anticipated movement of dynamic obstacles. When considering perception for motion planning, dynamic obstacles need to be continuously tracked to ensure accurate future predictions. Furthermore, areas previously deemed safe could potentially become hazardous if their status isn't periodically updated. This isn't an issue when the field of view (FOV) spans 360 degrees. Studies such as those by [24, 20] introduced methods for generating

collision-free trajectories within a 360-degree FOV range. However, real-world scenarios often impose FOV limitations on most MAVs. Thus, adding gaze planning, i.e., choosing the most advantageous direction for the MAV to look at, becomes necessary. Studies like [25, 11, 26] have suggested some basic policies, including looking in the current velocity direction or the target position direction. Although these methods work well in static environments, they may not guarantee optimal results in dynamic environments. In order to achieve robust gaze planning in dynamic environments, more sophisticated methods are proposed by [3, 17, 27, 4, 28]. These studies model the problem as an optimization problem, assigning handcrafted objective functions to guide the MAV look toward potentially hazardous directions. For instance, authors in [4] prioritize the direction of already observed obstacles likely to collide with the MAV, while authors in [3] emphasize areas that have remained not updated for extended periods. Authors of [17] consider different objectives in a multi-objective optimization framework, including the MAV's current velocity direction, the target position direction, and the direction of the nearest obstacle.

1-2-2 Collision avoidance performance benchmarking

While the previously mentioned research has significantly advanced dynamic collision avoidance methods, a key factor for further progress in this field involves objectively evaluating and comparing these methods. This necessitates benchmarking to assess their performance and robustness across various conditions accurately.

The most rudimentary benchmarking approach for collision avoidance methods involves comparing the success rates, as exemplified in studies like [3, 4, 11]. Here, the authors manually construct the testing environments, demonstrating that their proposed methods achieve higher success rates. Some studies, such as [29, 30, 31, 32, 33], have proposed more standardized benchmarking suites for comparing collision avoidance methods in randomly generated static environments as well as some metrics utilized to quantify the difficulty of these static environments such as obstacle density [34, 35] and traversability [36, 37, 38]. Furthermore, [39, 40, 41, 42] provide benchmarks for testing in dynamic environments, allowing researchers to compare their methods' performance in more diverse scenarios. These benchmarks provide various testing APIs (Application programming interfaces) for different robots [40] and are built upon high-fidelity simulators like Gazebo [39]. However, they lack a comprehensive set of metrics to quantify the difficulty of the dynamic environments, and relying solely on the success rate without indicating the difficulty of the environments is insufficient to identify a method's strengths and weaknesses. For instance, a method might excel in simple environments, reflected by a higher success rate, but underperform in more challenging environments. Consequently, a benchmarking suite for dynamic collision avoidance must incorporate a wide range of environments, accompanied by metrics that articulate their difficulty levels. This is the primary motivation for this thesis.

1-3 Contribution

The contribution of this thesis can be summarized in three parts:

1. We propose metrics to determine the difficulty of a dynamic environment and validate their representativeness through experiments.
2. We build a custom simulator that allows rapid, versatile tests for dynamic collision avoidance in 2D cases.
3. We compare typical dynamic collision avoidance algorithms based on our proposed metrics and provide intuitions for choosing these methods in different scenarios.

1-4 Organization

The organization of this thesis report is as follows:

- **Chapter 2** is dedicated to the introduction of the custom simulator's design, which is crucial for investigating and testing difficulty metrics. This chapter will detail the map generator and navigation modules that allow us to recreate and navigate through different environments with varied complexity levels.
- **Chapter 3** delves into the definition and motivation of several environment difficulty metrics, essential for evaluating the complexity of an environment and hence the performance of different collision avoidance methods. This chapter will also discuss how these metrics are compared, thereby providing insights into their applicability and effectiveness in different situations.
- **Chapter 4** provides a comprehensive overview of the experimental setup and the subsequent results. Initially, it presents the reasoning behind the selection of specific parameters and collision avoidance methods utilized throughout the experiments. It then proceeds to illustrate the performance of various difficulty metrics and collision avoidance methods within various environments. This exploration serves to either validate or challenge the concepts that were previously introduced in preceding chapters and then select metrics that accurately represent the difficulty of dynamic environments.
- **Chapter 5** delves deeply into the outcomes of our experimental findings. Through this discussion, we evaluate the efficacy of each difficulty metric and collision avoidance technique, emphasizing their relevance in distinct scenarios. Moreover, we outline specific use cases where the selected difficulty metrics prove to be most advantageous.
- **Chapter 6** offers a comprehensive summary derived from our comparison of metrics, highlighting the specific conditions under which each metric is most applicable. Additionally, we briefly outline potential avenues for future research aimed at refining these metrics and advancing the field.

Chapter 2

Simulator Design

This chapter outlines the design and implementation of a specialized simulator to investigate the environment difficulty metric. First, we establish the critical requirements for the simulator, followed by a detailed explanation of its practical implementation, encompassing elements such as the software platform, map generation, and navigation modules.

2-1 Requirement Analysis

As discussed in Chapter 1, the proposed simulator must fulfill two primary objectives:

1. It should facilitate fair and efficient testing and comparison of various methods.
2. It should enable the isolation of map difficulty as an independent variable influencing the performance of different methods.

For the first objective, the simulator must possess a standardized API, which enables various methods to be seamlessly integrated and evaluated within the simulator. Considering practical scenarios, most MAVs possess a limited Field of View (FOV) due to their restricted capacity to carry sensors [17]. Consequently, our simulator should be capable of simulating various methods for both trajectory planning and gaze planning.

Addressing the second objective requires the simulator's highly modular and simplified structure. This structure should allow us to discount extraneous factors such as perception errors, localization errors, control errors, and computational time. We propose a simulator composed of several vital components, as depicted in Figure 2-1: Map Generator, Environment Updater, Perception Module, Planning Module, Experiment Recorder, Map Difficulty Calculator, and Map Analyzer. Each component plays a distinct role, collectively contributing to the overall function and utility of the simulator.

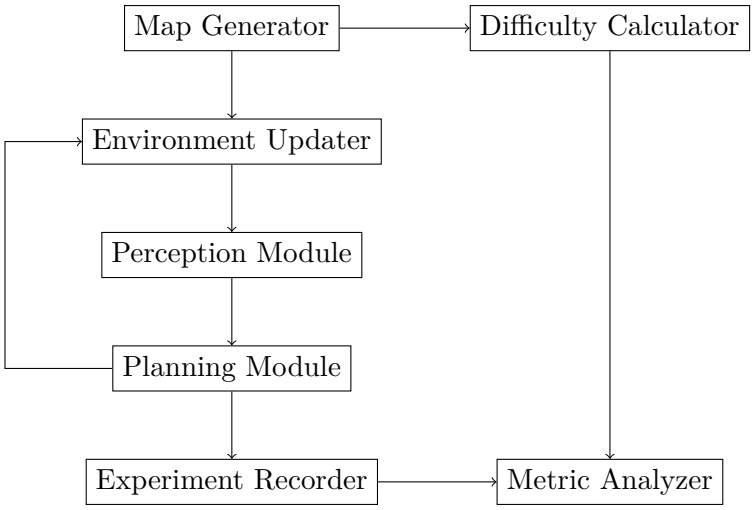


Figure 2-1: The pipeline of the metric testing gym.

2-2 Simulation Platform

The entire pipeline is based on the OpenAI gym [43] to reproduce, test, and compare different methods quickly. OpenAI gym is an environment designed for the training and implementation of reinforcement learning algorithms. The reason for choosing OpenAI gym over other high-fidelity simulators, such as AirSim [44] and Gazebo [45], can be summarized as follows:

1. OpenAI gym uses the classic "agent-environment" loop, a sequential process allowing the pipeline's modularization.
2. The sequential process also allows us to eliminate some irrelevant factors from computational time. The computational time is an irrelevant factor to the map difficulty, but it will be able to affect the performance of different methods.
3. The OpenAI gym is a lightweight simulator that runs faster than other high-fidelity simulators.

2-3 Map Generator

The Map Generator can generate different random maps according to the user's needs. In this Map generator, the obstacles are modeled as circles, and the user can define the number of obstacles, the radius of each obstacle, and the maximum speed of each obstacle like Figure 2-2.

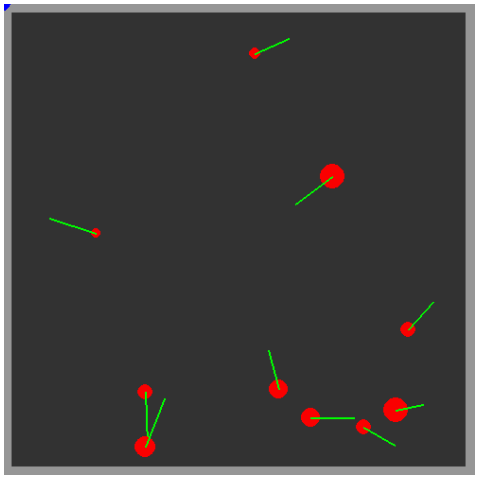
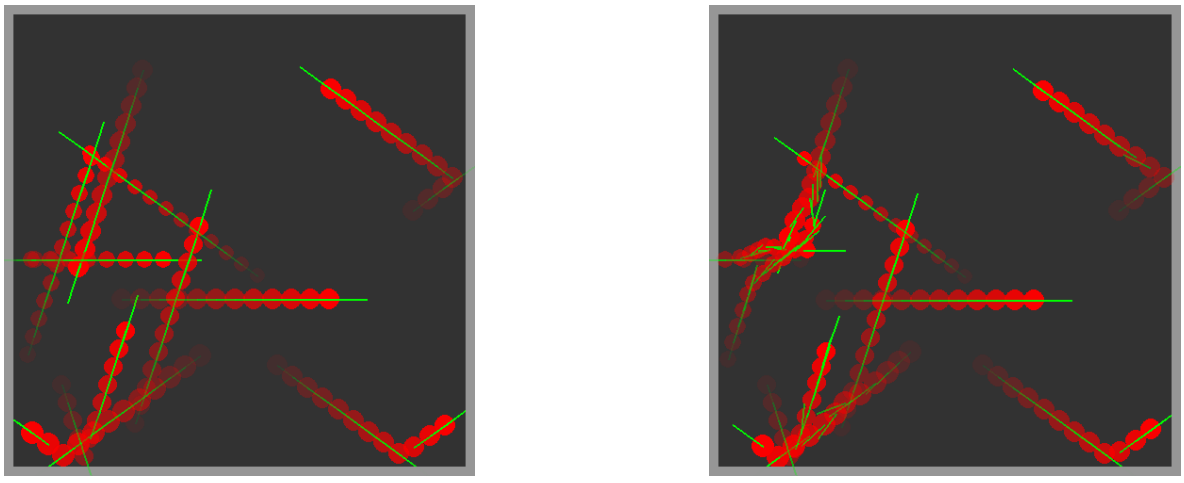


Figure 2-2: One map of obstacles with different sizes and velocities. The red circle represents the obstacles, and the green line represents the velocities of the obstacles.

The user can also define the motion profile of the obstacles, which is the policy of the obstacles' movements. The motion profile can be Constant Velocity Model(CVM) and Reciprocal Velocity Obstacle(RVO) like Figure 2-3.



(a) CVM **(b)** RVO

Figure 2-3: Two different motion profiles. (a) The CVM is a simple motion profile that the obstacles move with a constant velocity without considering the interaction among the obstacles. (b) The RVO implemented by [46] is a more complicated motion profile that the obstacles move with a constant velocity and avoid collisions with other obstacles.

We will generate two different datasets for the experiments with the map generator. The first dataset is the controlled map dataset, where we propose some assumptions and control a limited number of parameters to generate maps with a wide range of difficulties. The second dataset is the uncontrolled map dataset, where we eliminate the assumptions and test the generalization of the difficulty metrics.

Controlled Map Dataset

A wide variety of maps is essential to adequately test the environment difficulty metrics. As noted previously, numerous parameters are involved in the map generation process. To streamline this process and focus on the key aspects of our investigation, we introduce the following assumptions:

1. The dynamic obstacles are assumed to be circles with the same radius.
2. The velocities of the dynamic obstacles are assumed to be the same.
3. The obstacles are assumed to be moving with a constant velocity.

Then we generate a controlled map dataset by defining the following controlled variables in Table 2-1.

Parameter Name	**Symbol**	**Explanation**
Agent Number	n_{obs}	The agent here refers to dynamic obstacles in the environment. The user can define the number of agents as n_{obs} on the map.
Agent Radius	r_{obs}	The user can define the radius of the agents. In specialized maps, the agents are assumed to be circles with the same radius.
Agent Speed	v_{obs}	The user can define the maximum speed of the agents.

Table 2-1: Parameters of the map

Uncontrolled Map Dataset

Following the validation of the environment difficulty metrics on the controlled map dataset, we aim to evaluate the generalization capacity of these metrics by removing the imposed assumptions present in the controlled map dataset. Consequently, we generate a more diverse dataset termed the "Uncontrolled Map Dataset". The assumptions about the obstacles' velocities, the obstacles' size, and the obstacles' motion profile are removed in the uncontrolled map dataset.

2-4 Navigation Modules

2-4-1 MAV setup

Here we represent the MAV as a 2D circle with radius r_u. The MAV has a FOV of θ_{fov}, which is a sector with the angle of θ_{fov} and the radius of r_{fov}. The state of the MAV is represented as (x_u, y_u, ψ_u), where (x_u, y_u) is the position of the MAV and ψ_u is the heading angle of the MAV. The MAV has a maximum speed of v_{max}, a maximum acceleration of a_{max}, and a maximum yaw speed of $\dot{\psi}_u$.

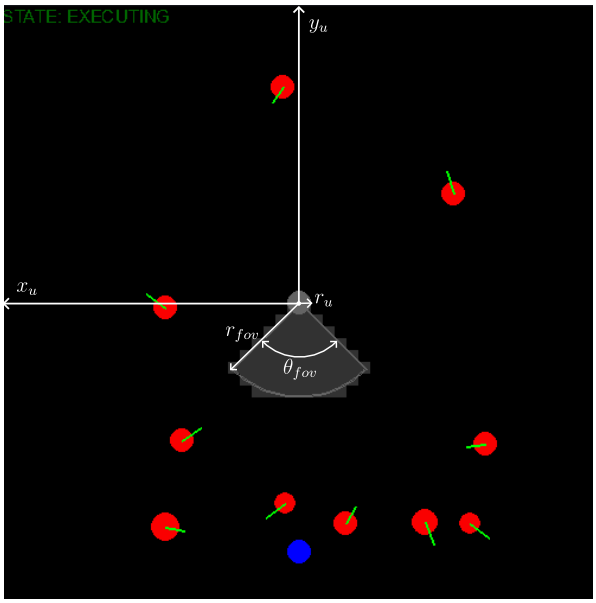


Figure 2-4: Parameters of the MAV. The MAV is represented as a 2D circle with radius r_u. The MAV has a FOV of θ_{fov}, which is a sector with the angle of θ_{fov} and the radius of r_{fov}. The state of the MAV is represented as (x_u, y_u, ψ_u), where (x_u, y_u) is the position of the MAV and ψ_u is the heading angle of the MAV.

2-4-2 Perception Module

An occupancy grid map is maintained for the MAV. Each grid m_{ij} in this grid map can be unexplored, unoccupied, or occupied, which is translated into the value of m_{ij} to be $\{-1, 0, 1\}$ respectively. The grid type at position (x, y) can be achieved through the function: $m_{ij} = getGrid(x, y)$. Initially, all of the grids are unexplored. Then similarly in [47], ray-casting is applied to update the grid in the FOV. We assume that the MAV has no perception error, which means the grids in the FOV are updated with ground truth values.

For dynamic obstacles, we further assume that the perception module can distinguish between static and dynamic obstacles and between different dynamic obstacles. Because of that, the observed dynamic obstacles are not added to the occupancy grid map, and no data association is needed for tracking dynamic obstacles. As in [48] and [12],

constant velocity models(CVM) are assumed for the dynamic obstacles. The tracking of dynamic obstacles is then achieved by Kalman Filter (KF) [49]. The estimated state of the dynamic obstacle i is defined as $\hat{x}_i = [x_i, y_i, v_{x,i}, v_{y,i}]^T$, where x_i and y_i are the estimated position of the dynamic obstacle i, and $v_{x,i}$ and $v_{y,i}$ are the estimated velocity of the dynamic obstacle i.

The observation of the dynamic obstacle i is defined as $z_i = [x_i, y_i]^T$. When rays from the ray-casting hit a dynamic obstacle, the corresponding KF tracker will be updated with the following equations:

$$\hat{x}_i^- = A\hat{x}_i + B\hat{u}_i \tag{2-1}$$
$$P_i^- = AP_iA^T + Q \tag{2-2}$$
$$K_i = P_i^- H^T (HP_i^- H^T + R)^{-1} \tag{2-3}$$
$$\hat{x}_i = \hat{x}_i^- + K_i(z_i - H\hat{x}_i^-) \tag{2-4}$$
$$P_i = (I - K_iH)P_i^- \tag{2-5}$$

where $\hat{x}_i$ is the estimated state of the dynamic obstacle i, P_i is the covariance matrix of the states, z_i is the observation of the dynamic obstacle i, A is the state transition matrix, Q is the covariance matrix of the process noise, H is the observation matrix, and R is the covariance matrix of the observation noise. The state transition matrix A and the observation matrix H are defined as follows:

$$A = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2-6}$$
$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{2-7}$$

Where Δt is the update rate of the environment. The covariance matrix Q and R are defined as follows:

$$Q = \begin{bmatrix} \sigma_x^2 & 0 & 0 & 0 \\ 0 & \sigma_y^2 & 0 & 0 \\ 0 & 0 & \sigma_{v_x}^2 & 0 \\ 0 & 0 & 0 & \sigma_{v_y}^2 \end{bmatrix} \tag{2-8}$$
$$R = \begin{bmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{bmatrix} \tag{2-9}$$

Once the dynamic obstacle is in the FOV of the drone, the corresponding KF tracker will be updated using its ground truth position. If the movement of dynamic obstacles is simulated using CVM, the KF tracker can accurately estimate the dynamic obstacle's position. However, if the movement of dynamic obstacles is simulated using RVO, the KF tracker cannot give a good estimation if the dynamic obstacle is out of the FOV.

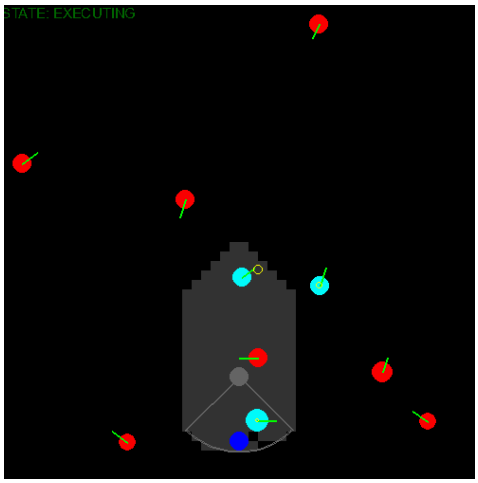


Figure 2-5: Perception of the MAV. Once the obstacles are in the FOV of the MAV, the corresponding KF tracker will be active, and the obstacles will be marked as blue circles. The estimation of the KF tracker is marked as yellow circles.

2-4-3 Planning Module

The Planning Module takes input from the perception module through the KF trackers with estimated positions and velocities of every observed obstacle. With the tracker information, the future prediction of dynamic obstacles can be achieved for collision detections.

The output of the planning module is composed of future trajectories and the yaw angle velocity for the next time step from the trajectory planner and the gaze planner, respectively.

Trajectory Planner

In the course of this research, several trajectory planning algorithms have been implemented. These implementations were realized by extending a base class, `traj_planner`, and specifically by overriding two of its methods: `plan` and `replan_check`. Utilizing this object-oriented approach enabled leveraging code reuse and maintaining a consistent interface across different planning algorithms, facilitating the ease of comparing and evaluating their performances.

At each timestep, the `replan_check` method is invoked to assess whether a new trajectory is warranted. This determination is contingent on the type of planner in operation. Local planners utilize a predefined update rate, such as initiating a replan every 5 seconds, whereas global planners evaluate the validity of the current trajectory following the incorporation of newly received perception data. If the `replan_check` method ascertains the necessity for a new trajectory (i.e., it returns True), the `plan` method is engaged to generate this new path. A successful planning execution will result in the `plan` method returning True, triggering the implementation of the newly created trajectory. In the event of planning failure, the drone engages a braking maneuver and

Algorithm 1 Trajectory Planning

```
Initialize: fail_count ← 0, state_machine ← 'PLANNING'
while not done do
    recheck_flag ← TRAJ_PLANNER.REPLAN_CHECK(drone)
    if recheck_flag is True then
        state_machine ← 'PLANNING'
        planning_success ← PLANNER.PLAN(drone, dt)
        if planning_success not True then
            DRONE.BRAKE
            state_machine ← 'PLANNING'
            fail_count ← fail_count + 1
        else
            state_machine ← 'EXECUTING'
            fail_count ← 0
        end if
    end if
end while
```

attempts to replan the trajectory in the next iteration. If the planning failure persists for a predefined number of consecutive iterations, the process is terminated with a deadlock state.

Gaze Planner

The gaze planner is designed to operate at each time step, constituting a crucial aspect of our dynamic navigation framework. This planner intakes a comprehensive snapshot of the current observations, which encapsulates information about the observed obstacles, the MAV's current position and velocity, and the planned future trajectory. Based on these inputs, the gaze planner computes the yaw angle velocity output for the next time step.

2-5 Result Recorder

Once a process is terminated, the experiment recorder will save the experiment data into a `.csv` file. The experiment data includes the following information:

Table 2-2: Sample data from experiments

Parameter	Experiment 1	Experiment 2
Gaze Planner	LookAhead	Owl
Planner	Primitive	MPC
Motion Profile	CVM	RVO
Map ID	0	1
Agent size	0.5	0.5

Table 2-2: Sample data from experiments

Parameter	Experiment 1	Experiment 2
Number of agents	10	10
Agent speed	2	2
Drone speed	4	4
Initial position	(4, 4)	(4, 4)
Target position	(4, 25)	(4, 46)
Flight time	9.8	12.9
Grid discovered	361	544
Agent tracked	1	2
Agent tracked time	4.0	3.05
Success	1	0
Static Collision	0	0
Dynamic Collision	0	1
Freezing	0	0
Dead Lock	0	0
State Machine	1	3

For each experiment, the experiment recorder will record the experiment results, including the flight time, unknown grid discovered, agent tracked, and agent tracked time. Each experiment will terminate in one of the following states: success, static collision, dynamic collision, freezing, and deadlock. Freezing is when the MAV cannot reach the target position in the given time. Deadlock is when the MAV fails to replan for a predefined number of consecutive iterations. The experiment recorder will also record the state machine state at the end of the experiment to indicate the reason for the collision.

2-6 Summary

The proposed simulator for investigating the environment difficulty metric consists of several components: the Map Generator, Environment Updater, Perception Module, Planning Module, Experiment Recorder, Map Difficulty Calculator, and Map Analyzer. It provides a modular and simplified structure for reproducing dynamic collision avoidance methods and evaluating the environment difficulty metric.

In the experiments, we will use the simulator to generate maps with different difficulty levels and evaluate the performance of the dynamic collision avoidance methods. The results will be analyzed to determine the correlation between the environment difficulty metric and the performance of the dynamic collision avoidance methods.

Chapter 3

Difficulty Metric Design

Building on the testing environment outlined in Chapter 2, we now turn our attention to the core concept of environment difficulty metrics. This chapter will introduce the proposed metrics, underpinned by an evaluation methodology that aims to provide a comprehensive understanding of each metric's functionality and effectiveness. We delve into the rationale behind these metrics, exploring their underlying assumptions and evaluating their feasibility within the context of dynamic collision avoidance. This discussion sets the foundation for a systematic approach toward understanding and quantifying map difficulty, ultimately contributing to a more reliable and robust benchmarking process.

3-1 Metric Design

We now introduce six different metrics for evaluating the difficulty of dynamic environments, including Obstacle Density, Traversability, Dynamic Traversability, VO Feasibility, Survivability, and Global Survivability.

3-1-1 Obstacle Density

Obstacle density is a widely used metric in collision avoidance in static environments [50, 51]. It is defined as the ratio between the area of obstacles and the area of the map:

$$D = \frac{\sum_{i=1}^{N} A_i}{A_{map}} \tag{3-1}$$

where A_i is the area of the i-th obstacle, N is the number of obstacles, and A_{map} is the area of the map.

3-1-2 Traversibility

In [36], the authors proposed a metric called traversability for static environments. The traversability is calculated by sampling positions and heading directions of the MAV and averaging over the furthest distance the MAV can travel without colliding with any obstacles. The traversability is defined as:

$$T = \frac{1}{N} \sum_{i=1}^{N} d_i \tag{3-2}$$

where d_i is the furthest distance the MAV can travel without colliding with any obstacles at the i-th sample, and N is the number of samples.

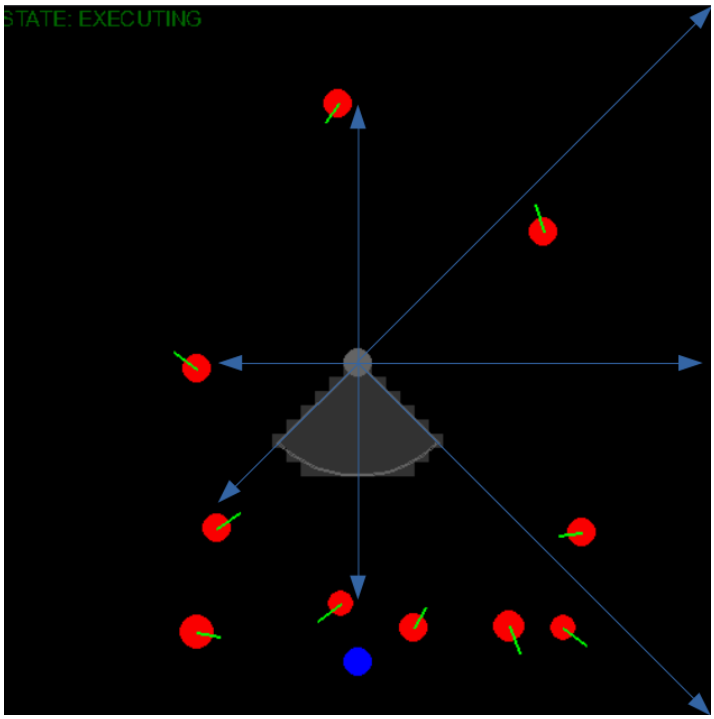


Figure 3-1: Traversability sample on a generated dynamic map. The blue arrow lines represent the furthest distance the MAV can travel without colliding with any obstacles in certain directions from the sampled position.

3-1-3 Dynamic Traversability

Since we are working on a dynamic map, the traversability at each time step might differ. We improve it by introducing the concept of dynamic traversability. The dynamic traversability is calculated by sampling time step and averaging the traversability over the sampled time steps:

$$T = \frac{1}{MN} \sum_{j=0}^{M} \sum_{i=1}^{N} d_i(t_j) \tag{3-3}$$

where $d_i(t_j)$ is the furthest distance the MAV can travel without colliding with any obstacles at the i-th sample at time step t_j, N is the number of samples, and M is the number of time steps.

3-1-4 VO Feasibility

[52] proposed the velocity obstacle (VO) concept for multi-agent collision avoidance tasks. The VO is defined as the set of velocities that will result in a collision with the other agent. Assuming the ego-robot is A and a dynamic obstacle is B, the VO for A considering collision with B is:

$$VO_B = \{\vec{v_A} | \exists t > 0 : (\vec{v_A} - \vec{v_B})t \in D(\vec{p_B} - \vec{p_A}, r_A + r_B)\} \tag{3-4}$$

where $\vec{v_A}$ and $\vec{v_B}$ are the velocities of A and B, $\vec{p_A}$ and $\vec{p_B}$ are the positions of A and B, r_A and r_B are the radius of A and B, and $D(\vec{p_B} - \vec{p_A}, r_A + r_B)$ is the disk centered at $\vec{p_B} - \vec{p_A}$ with radius $r_A + r_B$. The VO is illustrated in Figure 3-2.

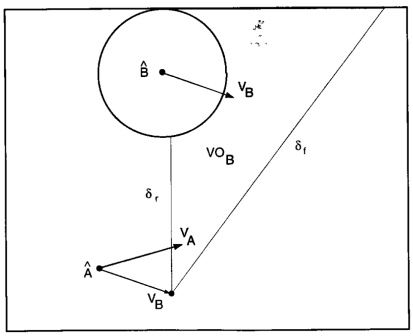


Figure 3-2: Velocity obstacle (VO) for collision avoidance [52].

The union of the VO of all the obstacles becomes the region from which the ego-robot cannot pick a velocity. Intuitively, the larger the VO area is, the more difficult the environment is for collision avoidance. Therefore, we propose a VO feasibility metric to measure the difficulty of the dynamic environment. It is calculated by:

1. Sampling positions in the dynamic environment
2. Sampling the possible velocities for each position
3. Calculating the proportion of the sampled velocities that are not in the VO of any obstacles

One example of calculating the VO feasibility at one sampled position is shown in Figure 3-3. Velocities are uniformly sampled in the $[0, v_{max}]$ range and in all directions. In this case, the proportion of feasible velocities is $\frac{n_{feasible}}{n_{feasible}+n_{infeasible}} = 0.75$. The VO feasibility is calculated by averaging over all the sampled positions.

$$F = \frac{1}{N}\sum_{i=1}^{N}\frac{n_{feasible}(i)}{n_{feasible}(i) + n_{infeasible}(i)} \tag{3-5}$$

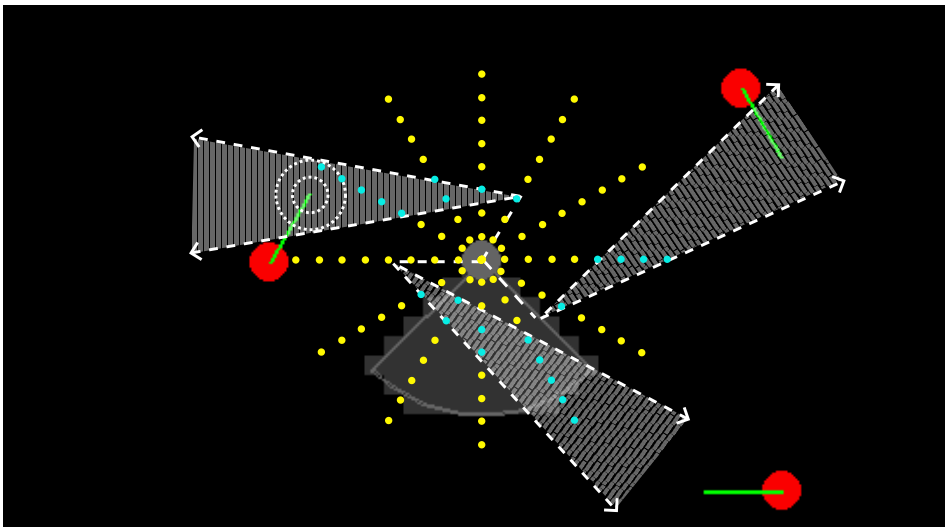


Figure 3-3: Example for calculating the VO feasibility at one sampled position. The yellow dots are feasible, and the blue dots are infeasible sampled velocities. The shadow area is the infeasible area of VO.

3-1-5 Survivability

While the traversability metric offers a viable measure of difficulty for static environments [36], its effectiveness diminishes in dynamic settings. In dynamic environments, the challenge of collision avoidance is intricately intertwined with the density of obstacles and their velocities. Unfortunately, this crucial element remains unaddressed in the current traversability metric, highlighting a critical limitation in its applicability to dynamic scenarios. Thus, we need a more comprehensive metric to accurately capture dynamic environments' complexity.

Therefore, we propose a survivability metric to measure a dynamic map's collision avoidance difficulty. In survivability calculation, we sample some positions on the map and put the static MAV at the sampled places for T_{max}. The survivability is calculated by averaging the surviving time of these static MAVs:

$$S = \frac{1}{N} \sum_{i=1}^{N} \min(t_i, T_{max}) \tag{3-6}$$

where t_i is the surviving time of the MAV at the i-th sample, and N is the number of MAV samples. Intuitively, if the obstacles move faster, they can cover more areas in the same period. Therefore, the survivability of the MAV will be lower.

3-1-6 Global Survivability

In case the obstacles are moving very slowly, the survivability metric might not be able to capture the difficulty of the entire map. Therefore, we introduce the concept of global survivability. Instead of posing one static MAV at each sample, we put N MAVs

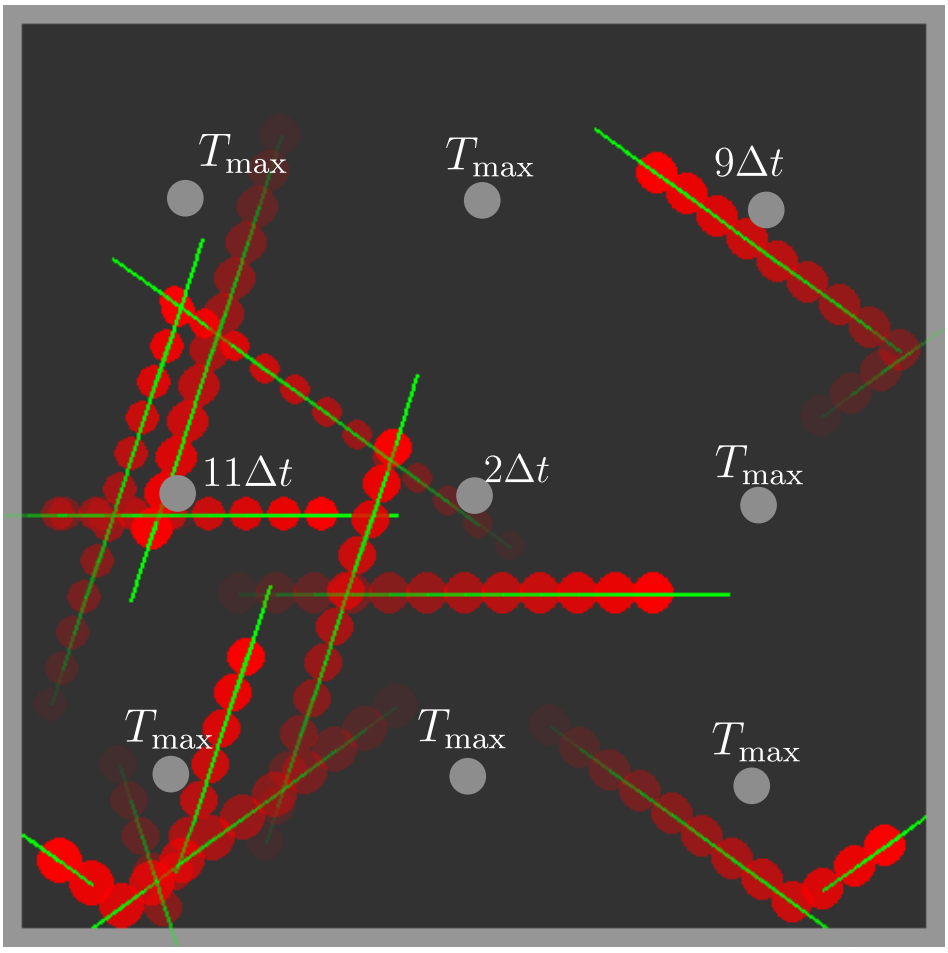


Figure 3-4: Example for calculating the survivability. The MAV is put at the sampled positions for T_{max}, and the surviving time of each sample is recorded. Δt represents the update time of the dynamic map.

around the map at K different timesteps and terminate the process when anyone of them collides with an obstacle. Global survivability is defined as:

$$S = \frac{1}{K}\sum_{j=1}^{K} \min(t_1^j, t_2^j \ldots, t_N^j, T_{max}) \tag{3-7}$$

where t_n^j is the surviving time of the n-th MAV at the j-th timestep, and K is the number of samples of timesteps.

3-2 Metric Evaluation

Assuming we have a map pool containing n different maps, each denoted by M_i. The map pool can be represented as $\mathcal{M} = \{M_1, M_2, ..., M_n\}$. m different planners are tested on these maps. The map difficulty metric is defined as a function D that maps a map

M_i to a real number $\mathrm{D}(M_i)$ and the success rate of a planner j on map M_i is defined as SR_{ij}. We can introduce two quantitative indicators for evaluating representativeness:

Spearman's Rank Correlation Coefficient(SRCC): This metric measures the monotonic relationship between two variables. Principally, a good metric $\mathrm{D}(M_i)$ should show a good monotonical relationship with the success rate SR_{ij} for each planner j. The Spearman's Rank Correlation Coefficient $Srcc_j$ for planner j is defined as:

$$Srcc_j = 1 - \frac{6\sum_{i=1}^{n}(\mathrm{R}(\mathrm{D}(M_i)) - \mathrm{R}(SR_{ij}))^2}{n(n^2-1)} \tag{3-8}$$

where the $\mathrm{R}(\mathrm{D}(M_i))$ and $\mathrm{R}(SR_{ij})$ are the rank of $\mathrm{D}(M_i)$ and SR_{ij} in all the sampled data respectively. The $Srcc_j$ is between -1 and 1. The larger the $|Srcc_j|$ is, the more monotonically related the $\mathrm{D}(M_i)$ and SR_{ij} are. Since a good difficulty metric should have this monotonical relationship with the success rate for every planner, we will use the average of $Srcc_j$ as the representativeness of the map difficulty metric $\mathrm{D}(M_i)$:

$$Srcc = \frac{1}{m}\sum_{j=1}^{m} Srcc_j \tag{3-9}$$

Coefficient of Variation(CV): This metric measures the variation of the success rate SR_{ij} under the same map difficulty level. We want to know whether the performance of one specific planner under different maps with the same difficulty level is stable. The maps are made into groups according to the map difficulty metric. For example, if the range of the function D is $[0, 10]$, we will divide the maps into ten groups:

$$\mathcal{M}_k = \{M_i | \mathrm{D}(M_i) \in [k, k+1]\} \tag{3-10}$$

For each group $\mathcal{M}_k$, we can calculate the Coefficient of Variation Cv_{jk} for planner j:

$$Cv_{jk} = \frac{\sigma_{jk}}{\mu_{jk}} \tag{3-11}$$

where σ_{jk} is the standard deviation of the success rate SR_{ij} in group $\mathcal{M}_k$ and μ_{jk} is the mean of the success rate SR_{ij} in group $\mathcal{M}_k$. The Cv_{jk} is between 0 and 1. The smaller the Cv_{jk} is, the more stable the performance of planner j is under the same map difficulty level. Since we want to have this stability for every planner, we will use the average of Cv_{jk} as the representativeness of the map difficulty metric D_i:

$$Cv = \frac{1}{m}\sum_{j=1}^{m} Cv_{jk} \tag{3-12}$$

3-3 Summary

In Chapter 3, we introduced a range of metrics designed to evaluate the difficulty of maps in the context of dynamic collision avoidance. These metrics were presented with a detailed explanation, illustrating the principles and assumptions.

Complementing the discussion of these metrics, we introduced a methodology for evaluating their effectiveness. This method treats the task of assessing map difficulty as a function that maps from a given map to a set of quantitative indicators. Through this approach, we sought to provide a comprehensive and flexible framework for understanding and quantifying the challenges presented by dynamic environments.

With the metrics and evaluation method, we are now ready to execute the experiments and analyze the results in the simulator developed in Chapter 2.

Chapter 4

Experiments

This chapter is first devoted to elaborating the experimental design tailored to assess the performance of the metrics outlined in Chapter 3. Our focus will be on reproducing a variety of trajectory and gaze planners, which will then be tested across two distinct map datasets as described in Section 2-3.

Following the testing of various planners, we will proceed to compute the associated metrics for each map. The objective is to delineate a relationship between these derived metrics and the success rate of the planners. This exploration offers a comprehensive insight into how specific difficulty metrics influence or correlate with the performance of various planner algorithms in the context of dynamic collision avoidance.

The rationale for recreating multiple planners is grounded in our pursuit of a universal and robust understanding of map difficulty. Essentially, we seek to validate the notion that the relationship between planner performance and difficulty metric is not contingent upon a specific planner. Instead, it should maintain consistency across various planners, providing a universally applicable measure of map difficulty.

4-1 Experiment Setup

In this section, general experiment setups will be introduced. We first list the general parameters that remain constant throughout all the experiments and justify our choice. Here the choice of parameters is listed in Table 4-1, including the general environment setups and the MAV properties. Here we will define the size of the entire map, the map resolution(size of one grid in the occupancy map), the update rate of the environment, the collision radius of the MAV, the maximum acceleration of the MAV, and the start and target position for the planning tasks.

When choosing the parameters for FOV, we will consider two scenarios: 1) the MAV has 360-degree FOV, and 2) the MAV has a limited FOV. In the first scenario, we

Parameter Name	Symbol	Choice
Map Size	$l \times w$	$50m \times 50m$
Map Resolution	$-$	$1m$
Update Rate	Δt	$0.1s$

(a) Parameters of the map

Parameter Name	Symbol	Choice
MAV radius	r_u	$1m$
MAV Acceleration	a_{max}	$4m/s^2$
FOV range	θ_{fov}	$90°$ or $360°$
FOV radius	r_{fov}	$8m$
Yaw angle velocity	$\dot{\psi}_u$	$1.4rad/s$
Start Position	$x_u(0)$ and $y_u(0)$	$\{4m, 25m, 46m\}$
Target Position	x_t and y_t	$\{4m, 25m, 46m\}$

(b) Parameters of the MAV

Table 4-1: Experimental parameters for the map and MAV

assume that the MAV is equipped with multiple cameras or Lidars, which can cover the entire space of a certain range. In the second scenario, we assume that the MAV is equipped with an intel real-sense d455 camera with a horizontal FOV of 90°. The range of the FOV is set to $8m$, within which the depth information is reliable. The yaw angle velocity is set to $1.4rad/s$. According to [17], if the relative tangential velocity of the obstacles to the camera v_{cam}^{obs} is larger than $6m/s$, the depth information is not reliable. We assume that one obstacle is $4m$ far away from the MAV(half of the FOV range, we want to have more accurate results within this range). To ensure the depth estimation error is less than $0.3m$, the maximum yaw angle velocity is calculated by: $\dot{\psi}_u = \frac{v_{cam}^{obs}}{4} = 1.5rad/s$. Considering the movement of the obstacle and the MAV, we set this number smaller to $1.4rad/s$.

Given our primary aim to examine the difficulty metrics through their relationship with the performance of various planners, it becomes imperative to reproduce these planners initially. As delineated in Section 2-4, this includes both trajectory and gaze planners.

4-1-1 Trajectory Planner

Table 4-2 shows a list of reproduced trajectory planners. The following part will introduce the implementation details of each planner.

Planner	Planning Scope	Methodology	Reference
Global Motion Primitives	Global Planner	Sampling-based	[14]
Local Motion Primitives	Local Planner	Sampling-based	[17]
Model Predictive Control	Local Planner	Optimization-based	[12]

Table 4-2: Reproduced trajectory planners.

Global Motion Primitives

Global Motion Primitives [14] is a sampling-based method for trajectory generation. The basic idea is to sample a set of motion primitives, use a cost function to evaluate each primitive, and then select and concatenate the primitives into an entire smooth trajectory. The detailed implementation of this planner is shown in Appendix A-1.

The replanning check of this planner is to check if the future trajectory will be in collision with the dynamic obstacles according to their predictions from the KF trackers introduced in Section 2-4-2. The planning process can be demonstrated in Figure 4-1.

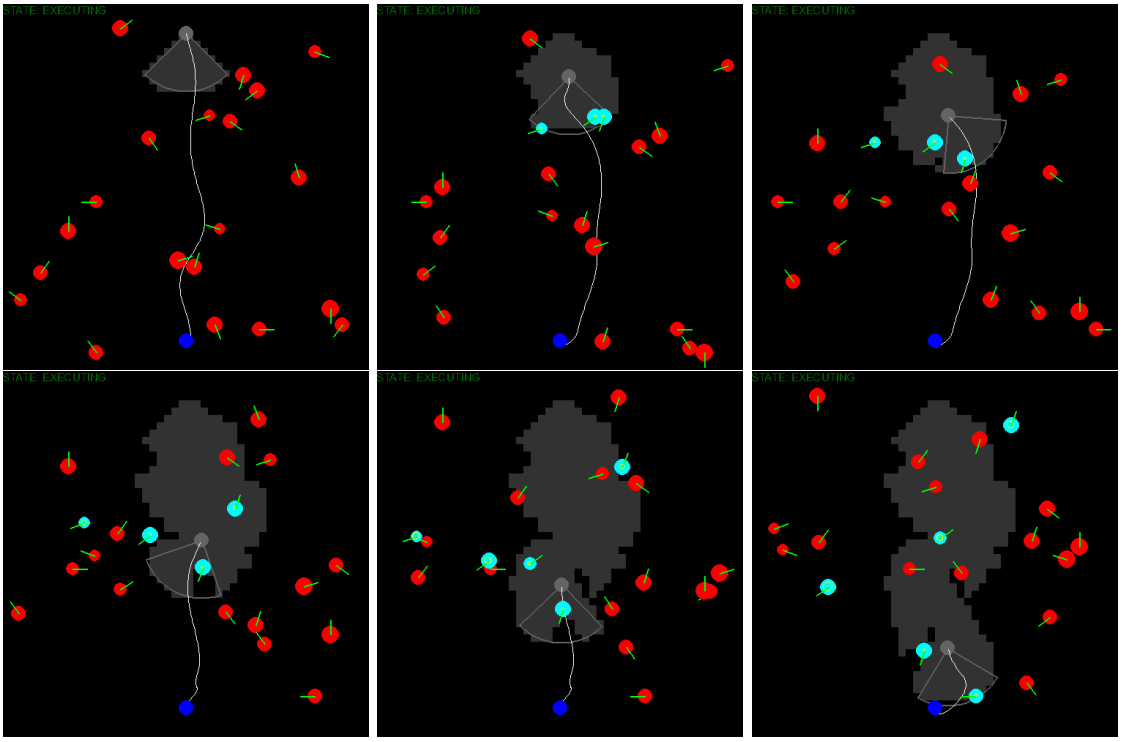


Figure 4-1: Example of Global Motion Primitives. The white line denotes the future trajectory.

Local Motion Primitives

Local Motion Primitives [17] is a local motion primitive planner. Instead of generating an entire trajectory to the target position, the planner only generates a short trajectory to the next waypoint and optimizes a heuristic cost function to select the next waypoint. The planning process can be demonstrated in Figure 4-2, and the detailed implementation is shown in Appendix A-2.

Model Predictive Control

In [12], dynamic collision avoidance is formulated into Model Predictive Control. In this planner, the dynamic obstacles are modeled with ellipsoids, and the constraints are

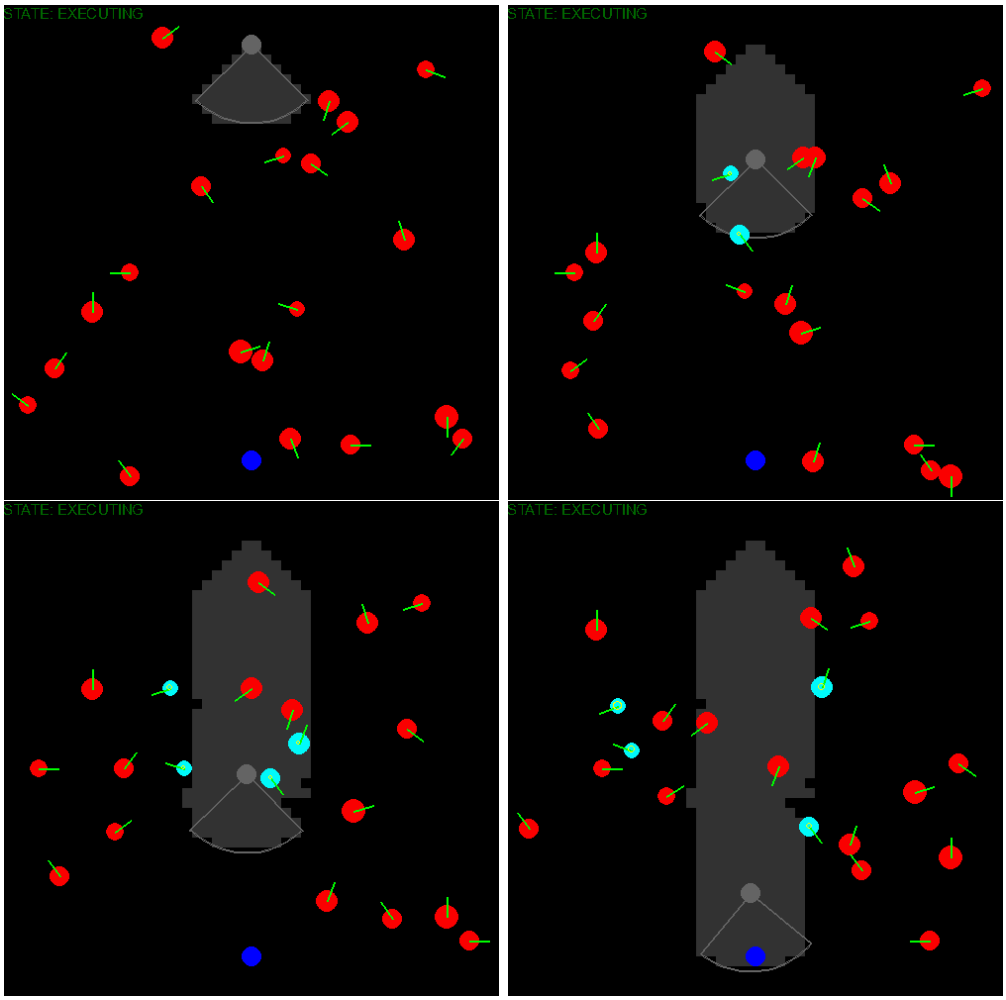


Figure 4-2: Example of Local Motion Primitives.

formulated as the distance between the MAV and the ellipsoids. The planning process can be demonstrated in Figure 4-3. The detailed implementation of this planner is shown in Appendix A-3.

4-1-2 Gaze Planner

Referring to Section 2-4-1, the gaze planner updates the MAV ψ_r yaw angle. It takes the MAV's current state and observation, then outputs the yaw angle speed $\dot{\psi}_r$ for the next time step. The yaw angle is updated by $\hat{\psi}_r = \psi_r + \dot{\psi}_r \Delta t$. The direction of the updated yaw angle is denoted as $\vec{d_r} = \begin{bmatrix} \cos(\psi_r + \dot{\psi}_r \Delta t) & \sin(\psi_r + \dot{\psi}_r \Delta t) \end{bmatrix}$.Multiple gaze planners are implemented in this work, including the following:

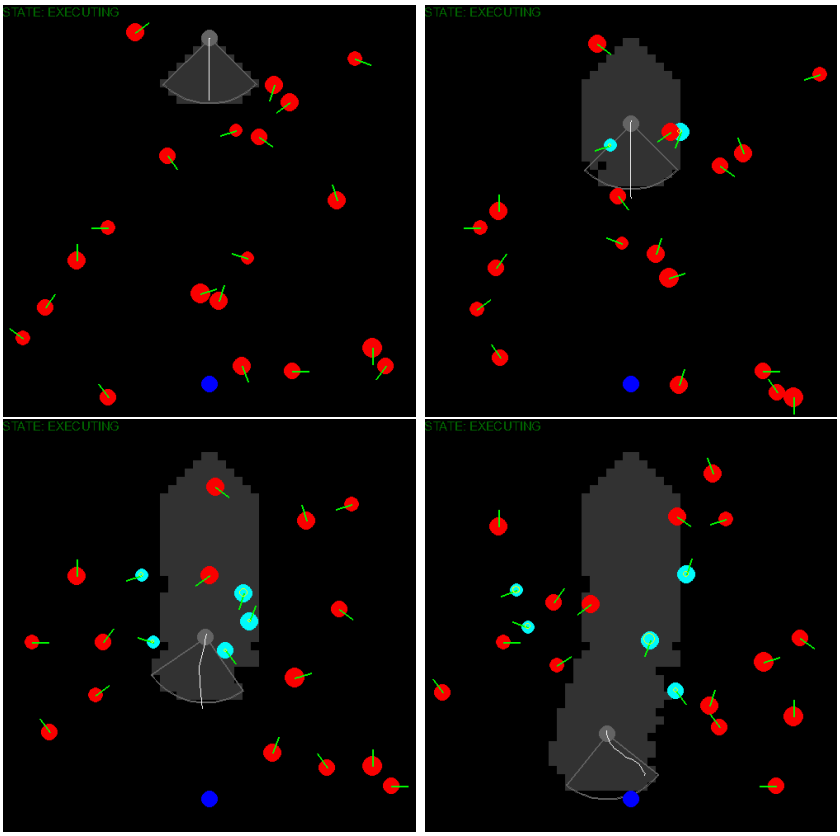


Figure 4-3: Example of Model Predictive Control.

Gaze Planner	Reference
LookAhead	[25, 27]
LookGoal	[26, 53]
Rotating	-
Finean et al.	[3]
Owl	[17]

Table 4-3: Reproduced trajectory planners.

LookAhead

LookAhead planner is a simple planner that aligns the heading of the MAV to the direction of current velocity $\vec{d_v} = \begin{bmatrix} \dot{x}_r & \dot{y}_r \end{bmatrix}$. It can be formulated as the following optimization problem:

$$\min_{\dot{\psi}_r} \quad \arccos\left(\frac{\vec{d_r} \cdot \vec{d_v}}{\left\|\vec{d_r}\right\| \left\|\vec{d_v}\right\|}\right) \tag{4-1}$$

$$\text{s.t.} \quad \dot{\psi}_r \in \left[-\dot{\psi}_{max}, \dot{\psi}_{max}\right] \tag{4-2}$$

LookGoal

LookGoal planner aligns the heading of the MAV to the direction of the target position $\vec{d_t} = \begin{bmatrix} x_t - x_r & y_t - y_r \end{bmatrix}$. It can be formulated as the following optimization problem:

$$\min_{\dot{\psi}_r} \quad \arccos\left(\frac{\vec{d_r} \cdot \vec{d_t}}{\left\|\vec{d_r}\right\| \left\|\vec{d_t}\right\|}\right) \tag{4-3}$$

$$\text{s.t.} \quad \dot{\psi}_r \in \left[-\dot{\psi}_{max}, \dot{\psi}_{max}\right] \tag{4-4}$$

Finean et al.

Finean et al. [3] is an optimization-based gaze planning method. Two voxel grid maps are maintained. One is the "Last Time Observed Map" t_i, representing when the voxel i is last observed. The other is the "Future Occupancy Map" v_i, which represents whether the MAV will occupy the voxel i in the future. Each voxel is assigned the value:

$$v_i = \begin{cases} 0 & \text{if the future trajectory does not occupy the voxel} \\ \tau & \text{if the future trajectory occupies the voxel at time step } \tau \end{cases} \tag{4-5}$$

One example of the two grid maps at one moment is shown in Figure 4-4. With the two maps, the planner calculates the reward for each voxel i as:

$$r_i = \begin{cases} c_1 & 0 < v_i \leq \tau_s \quad \text{and} \quad t_i \geq \tau_c \\ c_2 & v_i > \tau_s \quad \text{and} \quad t_i \geq \tau_c \\ \max(c_3 t_i, 1) & 0 < \text{otherwise} \end{cases} \tag{4-6}$$

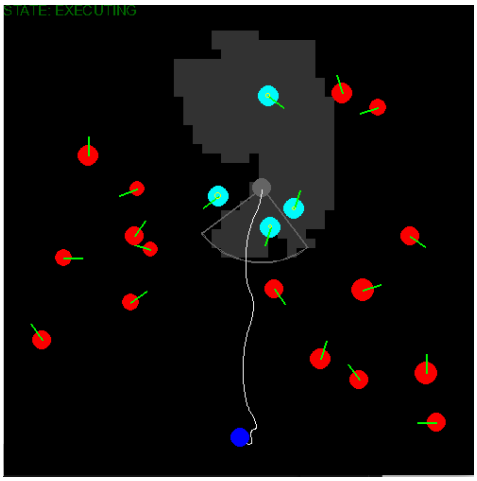
where τ_s, τ_c, c_1, c_2, and c_3 are hyperparameters. The planner then selects the yaw angle velocity $\dot{\psi}_r$ that maximizes the reward in the FOV:

$$\dot{\psi}_r = \underset{\dot{\psi}_r}{\text{argmax}} \sum_{i \in FOV(\vec{d_r})} r_i \tag{4-7}$$

Owl

Owl planner is proposed in [17]. It also considers gaze planning as an optimization problem. Compared to the Oxford planner, the Owl planner considers more factors in a dynamic environment. Four directions are prioritized in four cost functions f_1 to f_4: the direction of the target position, the direction of the current velocity, the direction of the observed dynamic obstacles, and the direction that has not been updated for a period. The last cost function f_5 is defined so that the yaw angle velocity $\dot{\psi}_r$ is

(a) Two voxel grid maps maintained in [3]. The left is the "Last Time Observed Map" t_i; Darker color means the voxel is observed more recently. The map on the right is the "Future Occupancy Map" v_i. The future trajectory is projected onto the map. The darker color means the MAV will occupy the voxel in the nearer future.



(b) The environment where the two maps above are generated.

Figure 4-4: Example of Oxford planner. One example of the two grid maps in the cost function at one moment is shown.

not too large. Finally, the gaze planning problem is formulated as a multi-objective optimization problem:

$$\min_{\dot{\psi}_r} \quad \sum_{i=1}^{5} \lambda_i f_i(\vec{d_r}) \tag{4-8}$$

$$\text{s.t.} \quad \dot{\psi}_r \in \left[-\dot{\psi}_{max}, \dot{\psi}_{max}\right] \tag{4-9}$$

4-2 Experiments in the Controlled Map Dataset

4-2-1 Parameter Setups

As introduced in Section 2-3, several parameters are used to control the map generation in Controlled Map Dataset. Here the choice of these parameters is listed:

Parameter Name	Symbol	Choice
Agent Number	n_{obs}	$\{10, 20, 30\}$
Agent Radius	r_{obs}	$\{0.5, 1, 1.5\}\, m$
Agent Speed	v_{obs}	$\{2, 4, 6\}\, m/s$
Agent Motion Profile	-	CVM

(a) Parameters of the map

Parameter Name	Symbol	Choice
MAV Speed	v_{max}	$\{2, 4, 6\}\, m/s$

(b) Parameters of the MAV

Table 4-4: Experimental parameters for the map and MAV

According to this parameter setup, we have three distinct setups for n_{obs}, r_{obs}, v_{obs} and thus have $3 \times 3 \times 3 = 27$ different map setups in total. We generate 20 different maps for each map setup by modifying random seeds. Therefore, we have $27 \times 20 = 540$ maps in the controlled map dataset. We run multiple experiments on each map with different MAV setups and starting and target positions. Each map has nine candidate positions and $9 \times 8 = 72$ combinations of distinct start and target positions. We will also test each planner with three different maximum MAV speed v_{max}. Thus, we run $3 \times 72 = 216$ experiments for each planner and map.

4-2-2 Experiment Results

In this section, we will show the experiment results of the correlation between different metrics and the success rate under all maps in the controlled map dataset. We will apply two pre-processing methods for every metric: normalization and reverse. The normalization method is to normalize the metric value to the $[0, 10]$ range. For metric m, the normalized value is calculated by:

$$m_{norm} = \frac{m - m_{min}}{m_{max} - m_{min}} \times 10 \tag{4-10}$$

The reverse method is to make the metric represent the map's difficulty according to the SRCC value. Specifically, we want to have a lower success rate for maps with higher metric values. We reverse the metric value by:

$$m_{rev} = \begin{cases} 10 - m_{norm} & \text{if } Srcc(m_{norm}, SR) > 0 \\ m_{norm} & \text{if } Srcc(m_{norm}, SR) \leq 0 \end{cases} \tag{4-11}$$

Then we will show the correlation figures between the metric and the success rate. The x-axis is the pre-processed metrics, and the y-axis is the success rate. Different colors denote different planners. The light-colored band surrounding the curve represents one standard deviation (or one sigma) of the success rate of each planner under each level of the difficulty metric.

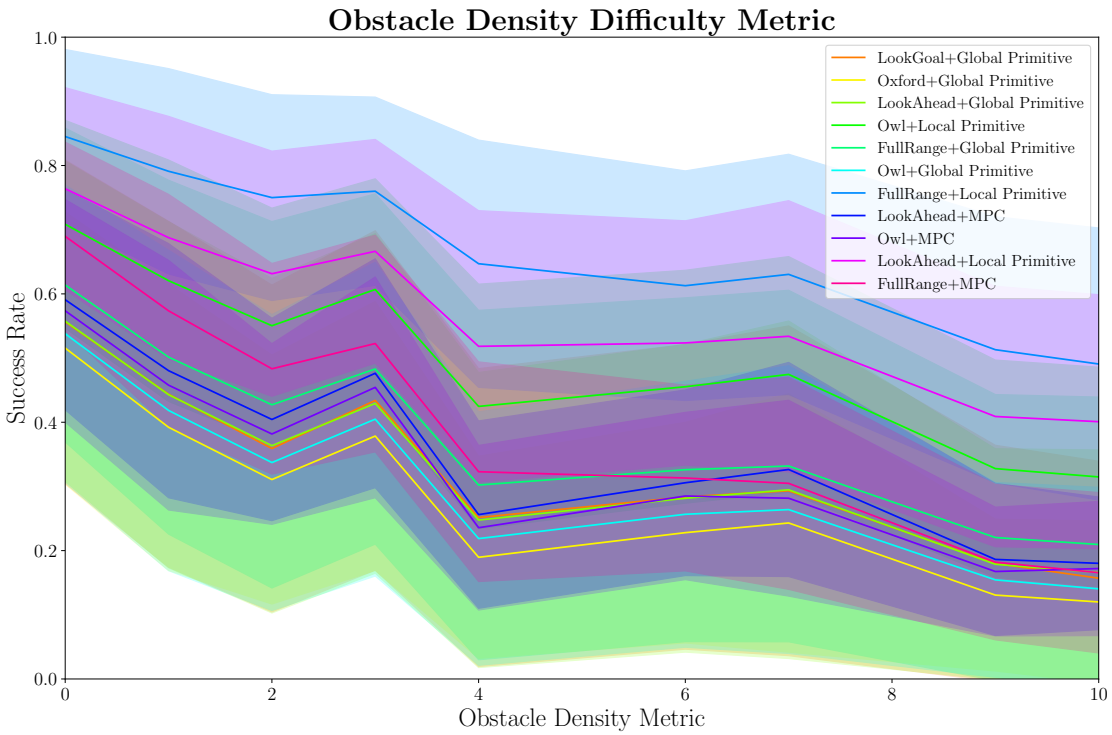


Figure 4-5: Correlation between obstacle density metric (3-1-1) and success rate

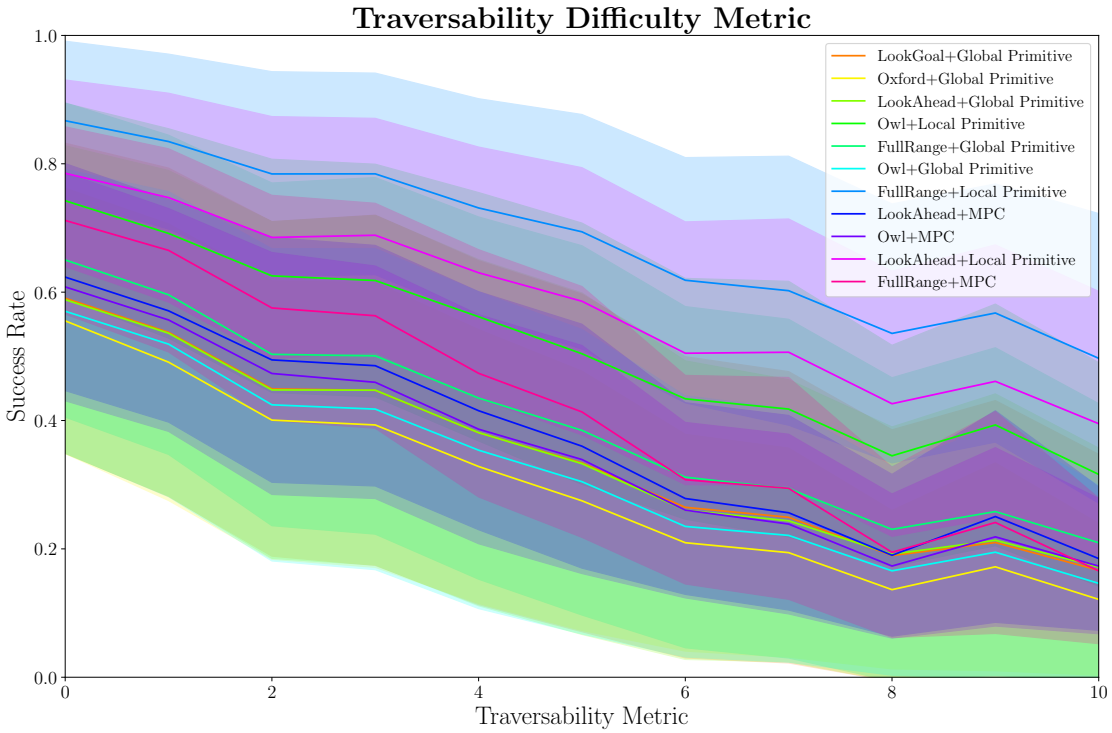


Figure 4-6: Correlation between traversability metric (3-1-2) and success rate

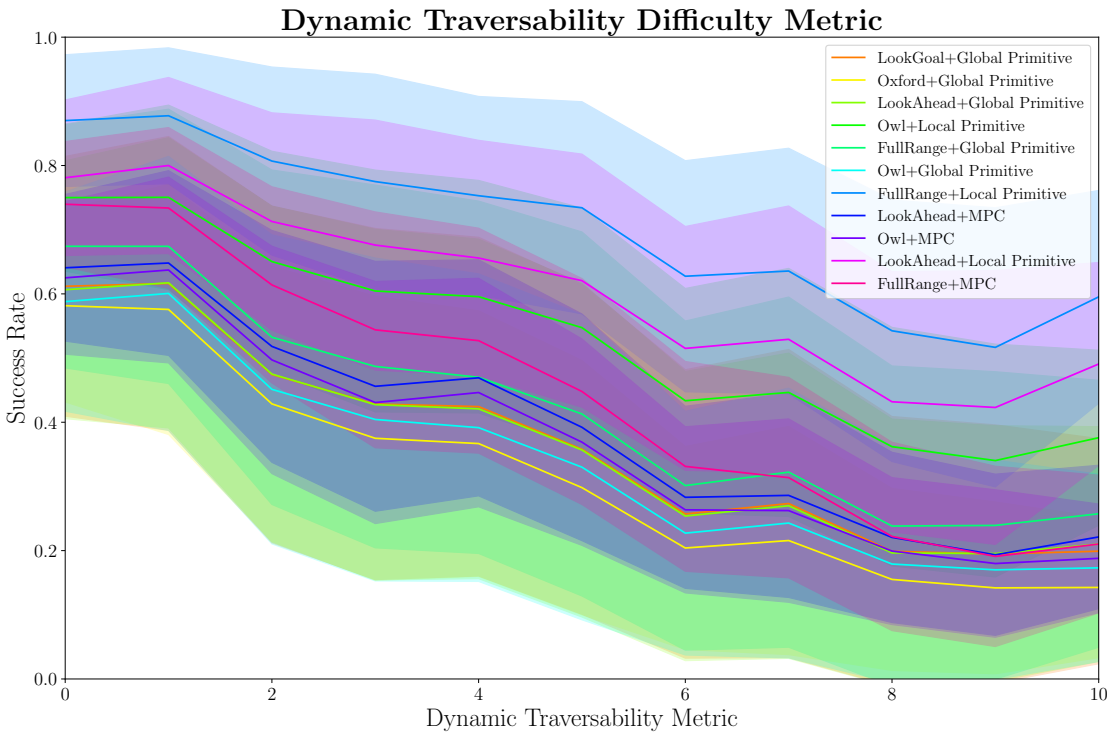


Figure 4-7: Correlation between dynamic traversability metric (3-1-3) and success rate

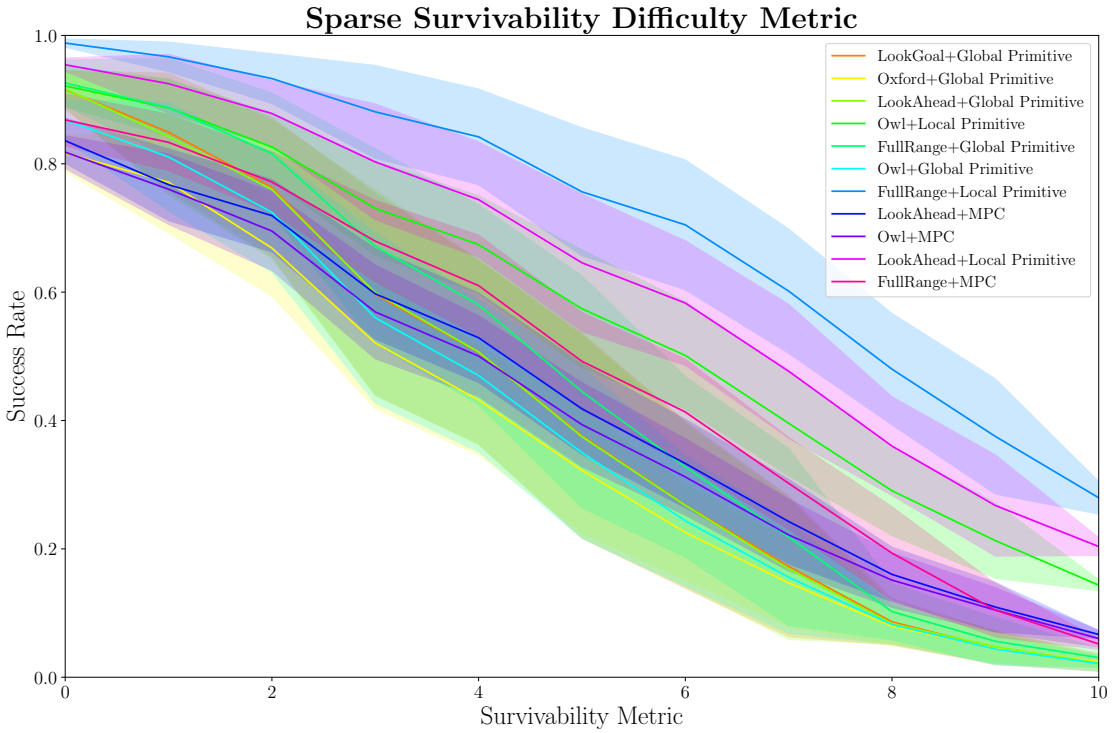


Figure 4-8: Correlation between sparse survivability metric (3-1-5) and success rate

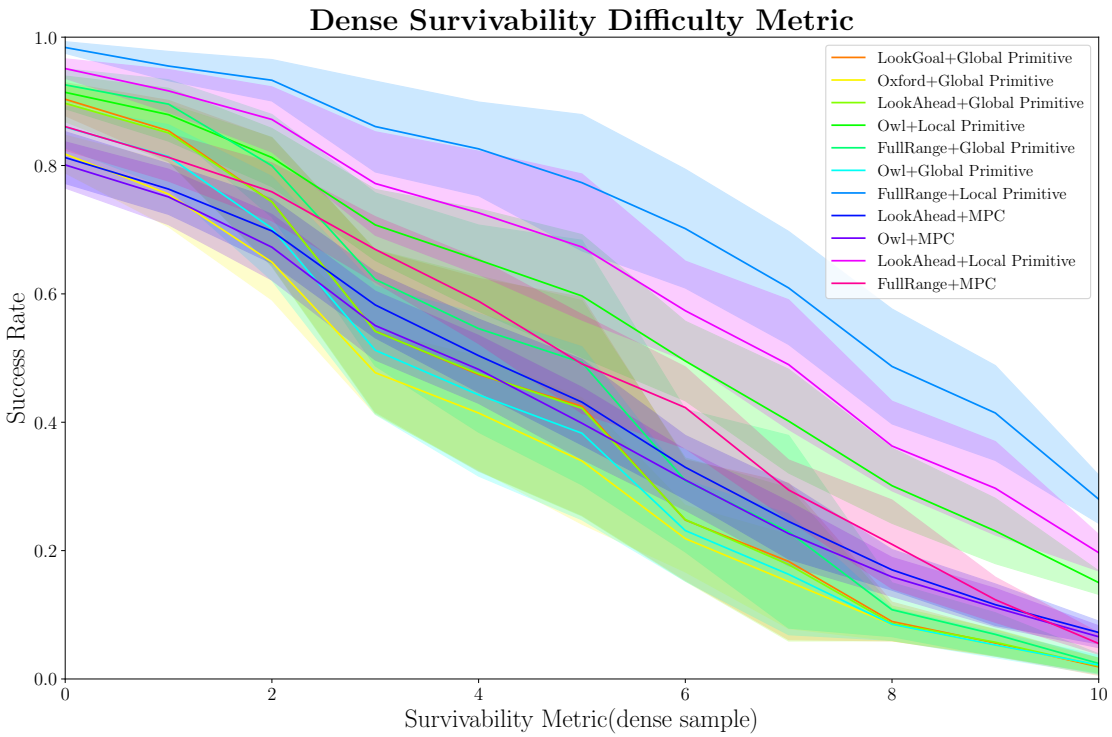


Figure 4-9: Correlation between dense survivability metric (3-1-5) and success rate

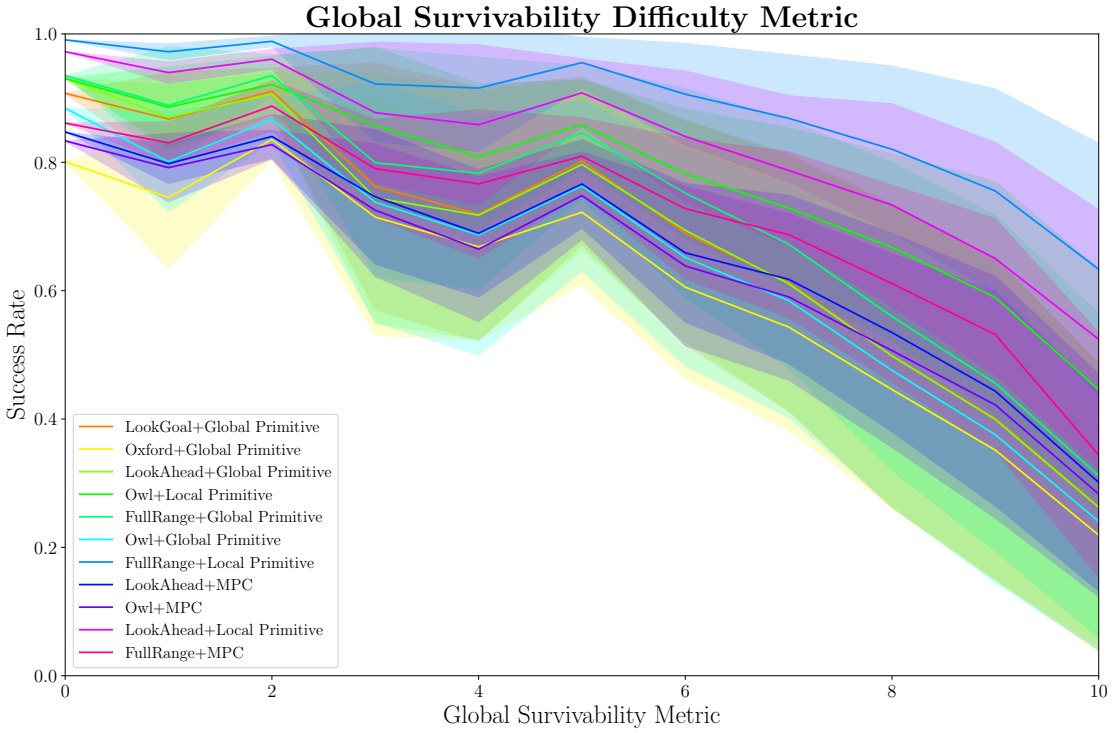


Figure 4-10: Correlation between global survivability metric (3-1-6) and success rate

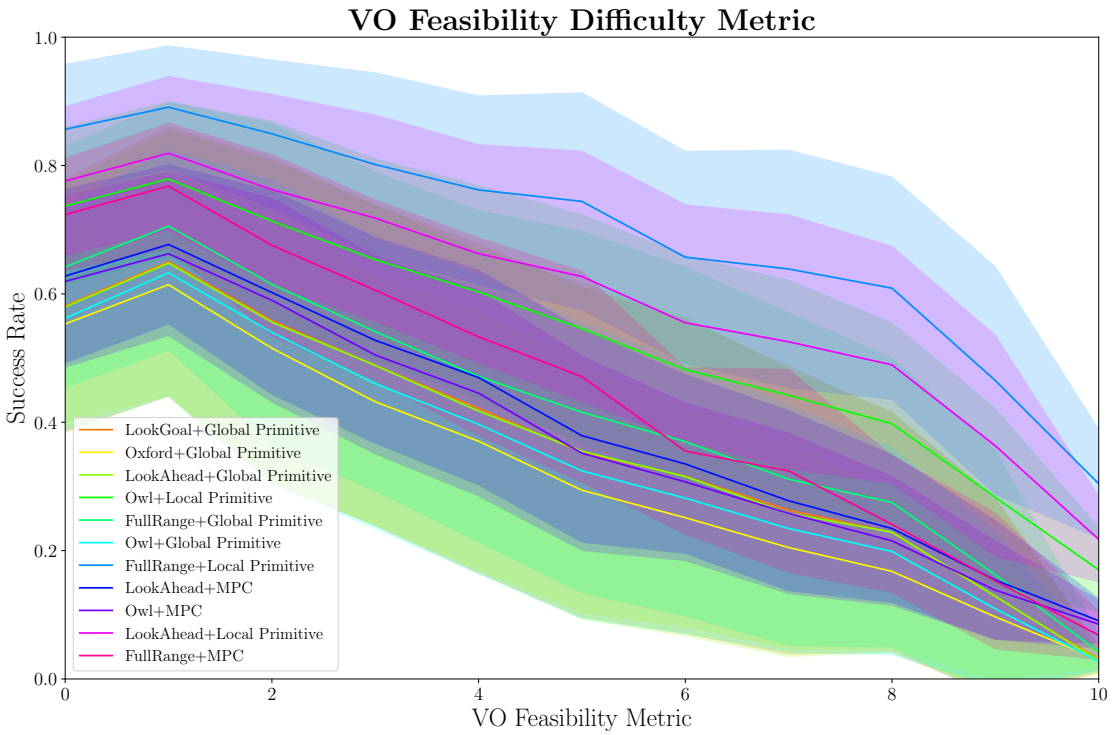


Figure 4-11: Correlation between VO feasibility metric (3-1-4) and success rate

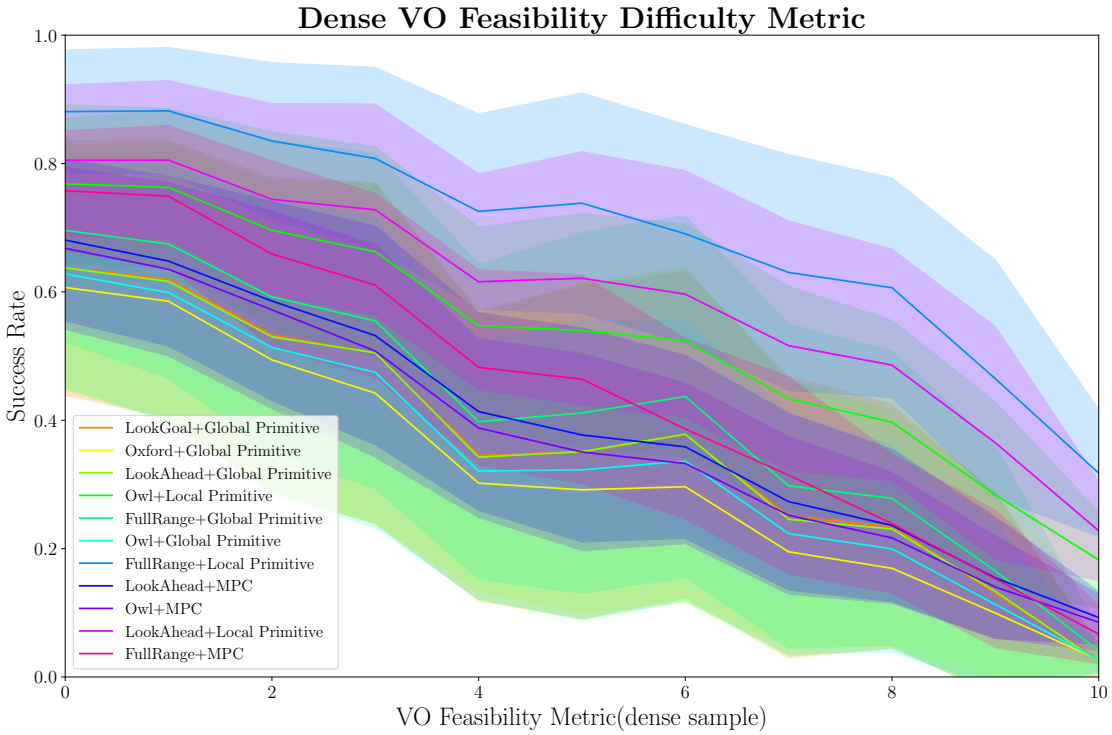


Figure 4-12: Correlation between dense VO feasibility metric (3-1-4) and success rate

Comparison between metrics

According to Section 3-2, we evaluate the metrics on the controlled map dataset using the mentioned quantitative indicators.

Difficulty Metric	**Spearman's Rank Correlation Coefficient(SRCC)**	**Coefficient of Variation (CV)**	**Computational Time**
Obstacle Density	0.501 ± 0.068	0.571 ± 0.193	**2s**
Traversability	0.511 ± 0.068	0.568 ± 0.184	$7s$
Dynamic Traversability	0.551 ± 0.073	0.516 ± 0.173	$30s$
Survivability(sparse)	0.932 ± 0.023	0.230 ± 0.095	$20s$
Survivability(dense)	$\mathbf{0.941 \pm 0.025}$	$\mathbf{0.211 \pm 0.087}$	$90s$
Global Survivability	0.607 ± 0.044	0.242 ± 0.078	$20s$
VO feasibility(sparse)	0.651 ± 0.077	0.461 ± 0.161	$7s$
VO feasibility(dense)	0.658 ± 0.077	0.473 ± 0.165	$15s$

Table 4-5: For each planner, we evaluate the metrics using the quantitative indicators mentioned in Section 3-2: Spearman's Rank Correlation Coefficient(SRCC), Coefficient of Variation (CV). These metrics can reflect the correlation between the metrics and the success rate. The computational time is the time to compute the metric for each map.

The comparison between different metrics can also be shown in Figure 4-13.

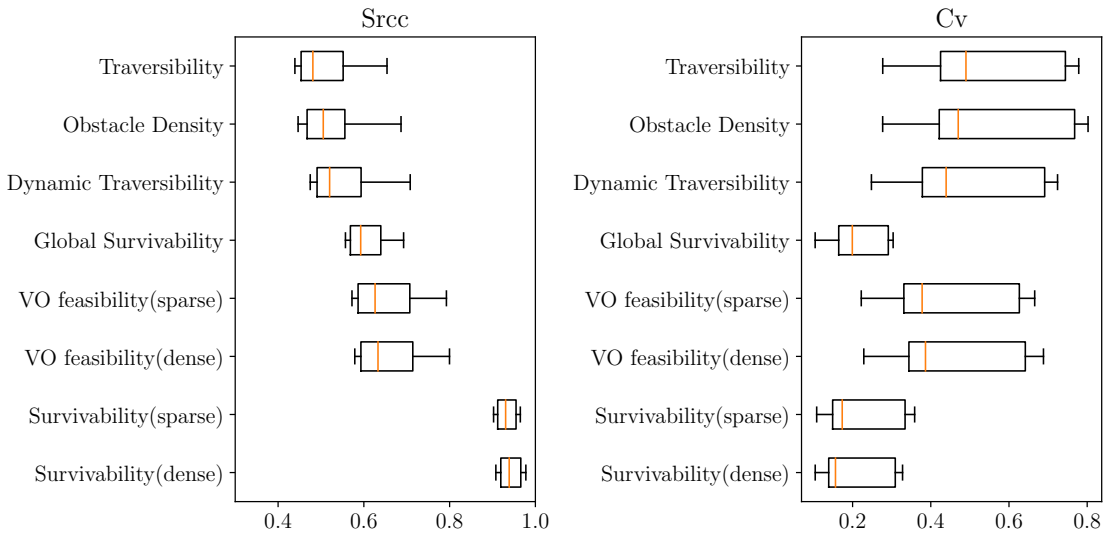


Figure 4-13: Boxplot of SRCC and CV of the metrics. If the SRCC is close to 1, the metric highly correlates with the success rate. If the CV is close to 0, the performance of planners under different maps with the same metric value is similar.

4-3 Experiments in the Uncontrolled Map Dataset

Through the evaluation results from the Controlled Map Dataset, only the survivability metric has demonstrated a high correlation with the success rate. Thus, we will further test the survivability metric on the Uncontrolled Map Dataset. Specifically, there will be three different categories of environments:

1. the environment of obstacles with various velocities
2. the environment of obstacles with various sizes
3. the environment of obstacles with a different motion profile RVO

4-3-1 Environment of obstacles with various velocities

To test the survivability metric on the environment of various obstacles of various velocities, we generate 45 maps in which the obstacle number is chosen from $\{10, 20, 30\}$, the obstacle size is chosen from $\{0.5, 1.0, 1.5\}m$, and the obstacle velocity is randomly sampled from $[2, 6]m/s$. The survivability metric is computed for each map. The scatter plot of the survivability metric and the success rate is shown in Figure 4-14.

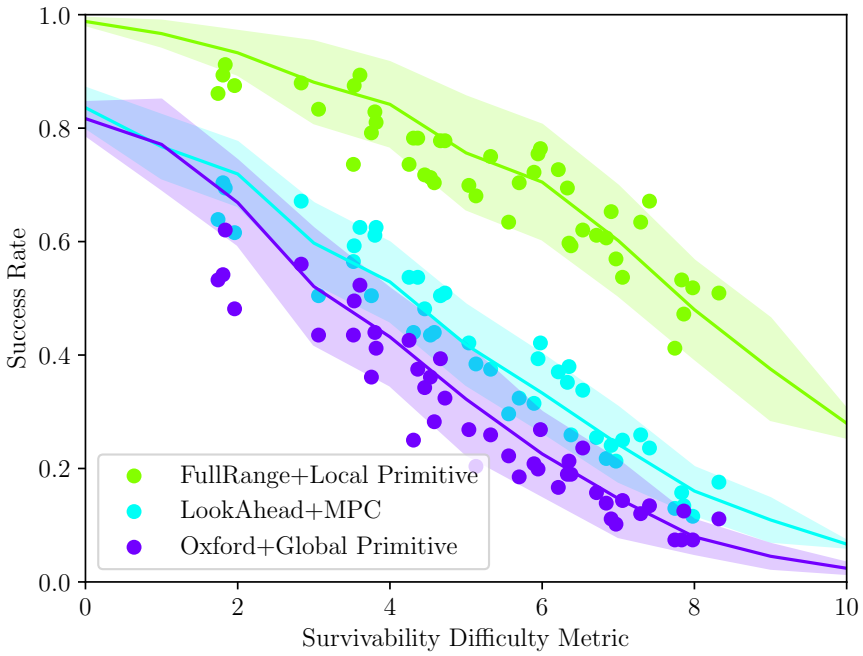


Figure 4-14: Scatter plot of the survivability metric (3-1-5) and the success rate on the environment of obstacles with various velocities. The curves are fitted curves from the controlled map dataset. The scatters are the survivability metric and the success rate on the uncontrolled map dataset.

Here three different planners are tested on the uncontrolled map dataset. The *FullRange+Local Primitive* and *Oxford+Global Primitive* are chosen for having the best

and worst performance on the controlled map dataset. The *LookAhead+MPC* is chosen for having a different trajectory and gaze planner and increasing diversity. The results of the proportion of data points within one standard deviation, two standard deviations, and three standard deviations from the Gaussian distribution fitted in the controlled map dataset are shown in Table 4-6.

Type of Map	Within 1 σ	Within 2 σ	Within 3 σ
Controlled	68%	95%	99.7%
Uncontrolled (Different Velocities)	76.3%	98.5%	100%

Table 4-6: The percentage of Uncontrolled Map Dataset data points(with various obstacle velocities) within one standard deviation, two standard deviations, and three standard deviations from the Gaussian distribution fitted in Controlled Map Dataset.

4-3-2 Environment of obstacles with various sizes

To test the survivability metric on the environment of various obstacles of various sizes, we generate 45 maps in which the obstacle number is chosen from $\{10, 20, 30\}$, the obstacle velocity is chosen from $\{2, 4, 6\} m/s$(All obstacles in one map have the same velocity), and the obstacle sizes are randomly sampled from $[0.5, 1.5] m$. The survivability metric is computed for each map. The scatter plot of the survivability metric and the success rate is shown in Figure 4-15. The testing planners are chosen the same as in the previous section.

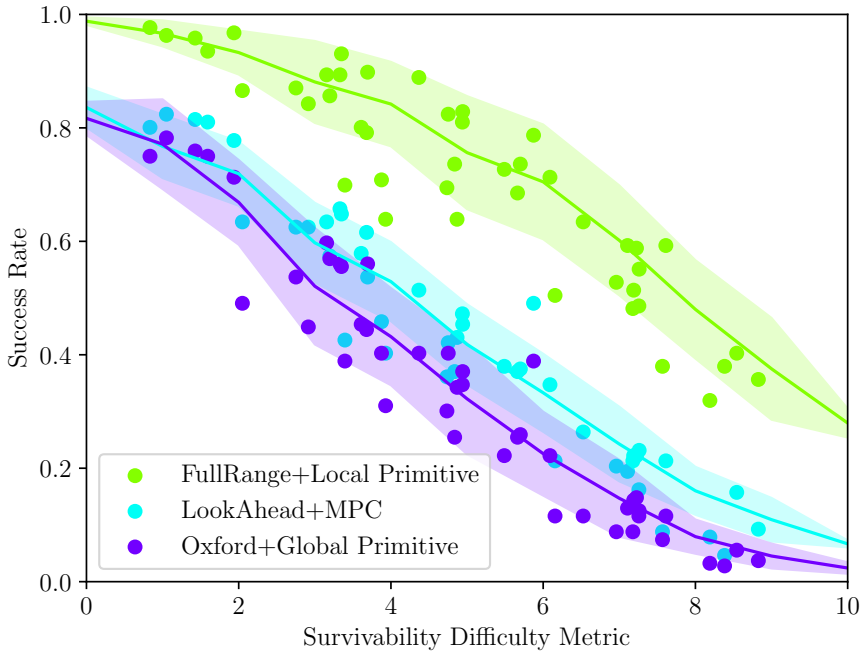


Figure 4-15: Scatter Plot of Survivability Metric (3-1-5) and Success Rate in Maps with Various Obstacle Sizes. The curve is the fitted curve from the controlled map dataset. The scatters are the data from the uncontrolled map dataset.

The results of the proportion of data points within one standard deviation, two standard deviations, and three standard deviations from the Gaussian distribution fitted in the controlled map dataset are shown in Table 4-7.

Type of Map	Within 1 σ	Within 2 σ	Within 3 σ
Controlled	68%	95%	99.7%
Uncontrolled (Different Sizes)	71.1%	94.8%	100%

Table 4-7: The percentage of Uncontrolled Map Dataset data points(with various obstacle sizes) within one standard deviation, two standard deviations, and three standard deviations from the Gaussian distribution fitted in Controlled Map Dataset.

4-3-3 Environment of obstacles with RVO motion profiles

To test the survivability metric on the environment of obstacles with RVO motion profiles, we generate 45 maps in which the obstacle number is chosen from $\{10, 20, 30\}$, the obstacle velocity is chosen from $\{2, 4, 6\}m/s$(All obstacles in one map have the same velocity), and the obstacle sizes are randomly sampled from $[0.5, 1.5]m$. The survivability metric is computed for each map. The scatter plot of the survivability metric and the success rate is shown in Figure 4-16. The testing planners are chosen the same as in the previous section.

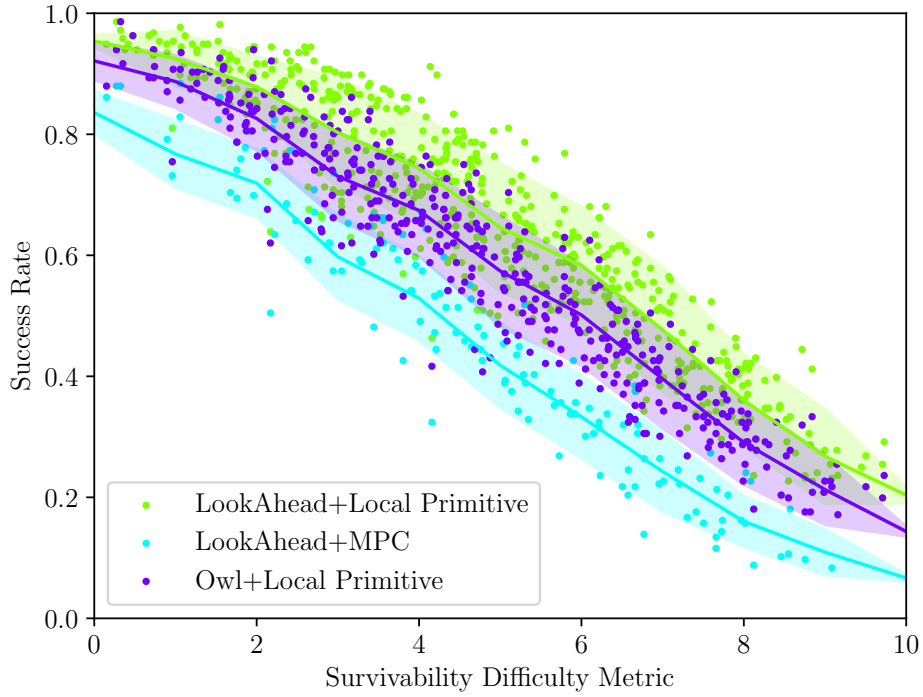
The results of the proportion of data points within one standard deviation, two standard deviations, and three standard deviations from the Gaussian distribution fitted in the controlled map dataset are shown in Table 4-8.

Type of Map	Within 1 σ	Within 2 σ	Within 3 σ
Controlled	68%	95%	99.7%
Uncontrolled (RVO)	63.7%	92.5%	98.0%

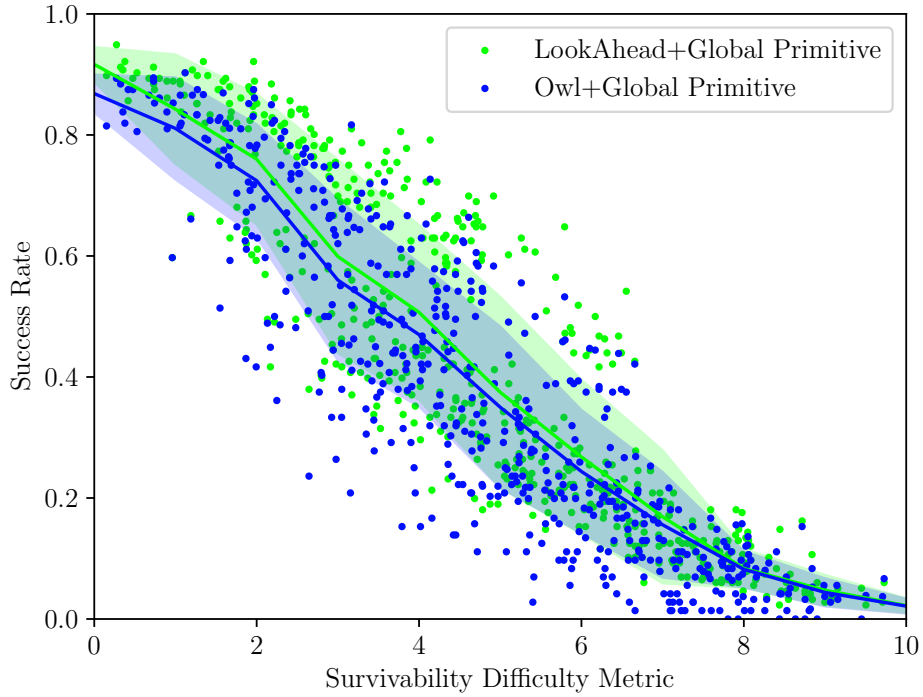
Table 4-8: The percentage of Uncontrolled Map Dataset data points(with RVO motion profile) within one standard deviation, two standard deviations, and three standard deviations from the Gaussian distribution fitted in Controlled Map Dataset.

4-4 Summary

In this chapter, we have introduced the experiments in the controlled and uncontrolled map datasets. The survivability metric has demonstrated a high correlation with the success rate in the controlled map dataset and is thus further tested in the uncontrolled map dataset. We will discuss the findings in the next chapter.



(a) Scatter Plot of Survivability Metric (3-1-5) and Success Rate of *Local Primitive* and *MPC* in Maps with RVO Motion Profiles.



(b) Scatter Plot of Survivability Metric (3-1-5) and Success Rate of *Global Primitive* in Maps with RVO Motion Profiles.

Figure 4-16: The environment of obstacles with RVO motion profiles.

Chapter 5

Discussion

In this chapter, we dive into the results of our experiments conducted in the previous chapter. Our primary focus is to assess how suitable each metric is for evaluating the complexity of environmental maps. We pinpoint the specific situations where each metric works best and explain the reasons for their performance. Also, we introduce possible real-world applications of these metrics, using examples to make them easier to understand. Our goal, through a thorough analysis, is to build a detailed understanding of how these metrics can be used and why they're essential in the field of collision avoidance research.

5-1 Scope of Metrics for Environmental Difficulty Evaluation

Certain assumptions were considered during the experimental design phase in the previous chapter. These assumptions define the limitations and the range of applications for the proposed metrics:

1. The environment under consideration is 2-dimensional.
2. The environment solely contains dynamic obstacles.
3. Perception and control errors are not taken into account.

Additionally, our experiments' findings have led to identifying specific constraints related to the scope of each metric's application. These constraints will be explored in the following sections.

5-1-1 Obstacle Density

Obstacle density is frequently employed as a metric to illustrate the difficulty of an environment. This has been demonstrated in various studies such as [50, 51], where the consensus is that a higher obstacle density in static environments makes collision avoidance more challenging. However, our experimental results show a different trend for dynamic environments in Table 5-1. The Spearman's Rank Correlation Coefficient (SRCC) between the obstacle density and the success rate is only 0.501, suggesting that in a dynamic environment, an increase in obstacle density doesn't necessarily result in a lower success rate of collision avoidance. The Coefficient of Variation (CV) for the obstacle density stands at 0.577, indicating that even under the same obstacle density, the success rate of collision avoidance can exhibit significant variance across different maps.

Difficulty Metric	SRCC	CV
Obstacle Density	0.501 ± 0.068	0.577 ± 0.204

Table 5-1: The correlation between obstacle density metric and success rate in all maps from the controlled map dataset. The obstacle velocities can be different on different maps.

The obstacle density metric primarily serves as a measure of difficulty for static environments. Hence, the relatively low correlation observed between obstacle density and success rate in dynamic environments is chiefly attributable to the fact that obstacle density does not account for obstacle velocity. In dynamic environments, obstacle velocity is vital to collision avoidance performance. To illustrate this point, we depict the performance of various planners under different obstacle velocities in Figure 5-1.

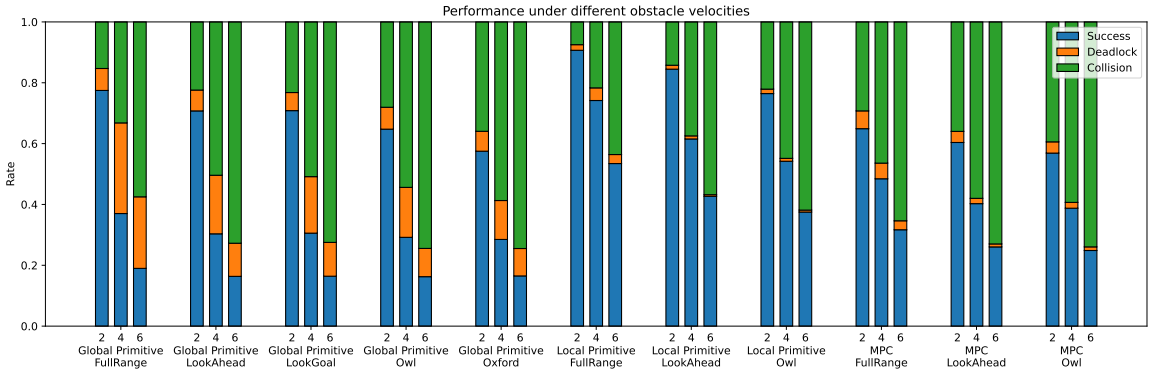


Figure 5-1: The performance of planners under different obstacle velocities. Each group represents the performance of different planners, and each bar under one group represents different obstacle velocities, specifically 2.0 m/s, 4.0 m/s, and 6.0 m/s.

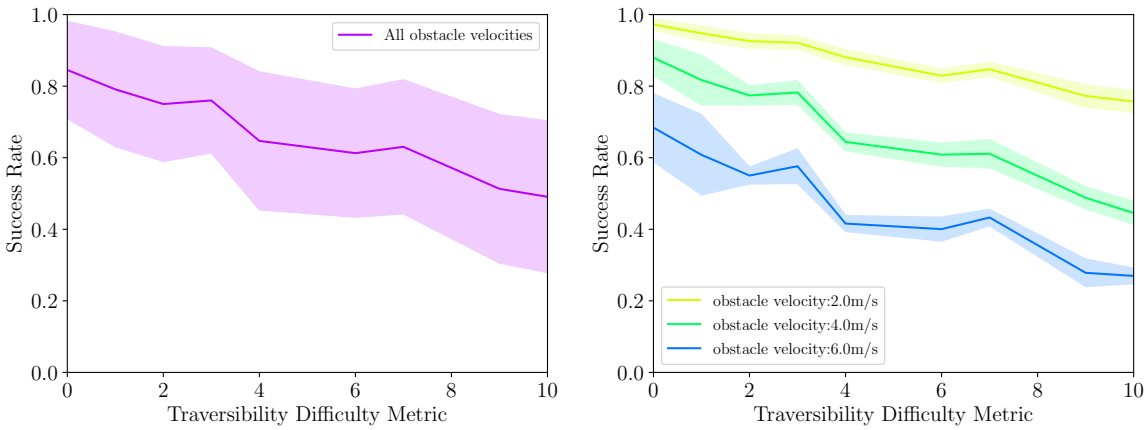
Our results show that the performance decreases significantly for each planner as the obstacle velocity increases. However, the current calculation for obstacle density does not account for obstacle velocities, leading to its low correlation with the success rate. Thus, it underscores the importance of integrating dynamic factors such as obstacle velocity when assessing environmental difficulty in dynamic scenarios.

We categorized the maps into three groups based on their obstacle velocities to further dissect the relationship between obstacle density and the success rate. We then evaluated the relevance of the obstacle density metric within each of these groups. The results are depicted in Table 5-2.

Obstacle Velocity	SRCC	CV
2.0 m/s	0.788 ± 0.045	0.075 ± 0.026
4.0 m/s	0.784 ± 0.030	0.158 ± 0.050
6.0 m/s	0.758 ± 0.029	0.272 ± 0.109

Table 5-2: After dividing the controlled map dataset into three groups according to the obstacle velocity setup, the correlation between the obstacle density metric and the success rate in each group. It has been significantly improved compared with Table 5-1

After segregating the map dataset into three subsets based on the defined obstacle velocities, we noticed a substantial improvement in the correlation between the obstacle density and success rate within each group. This enhancement in correlation, when compared with the results presented in Table 5-1, indicates the increased effectiveness of the obstacle density metric when used within a controlled obstacle velocity environment.

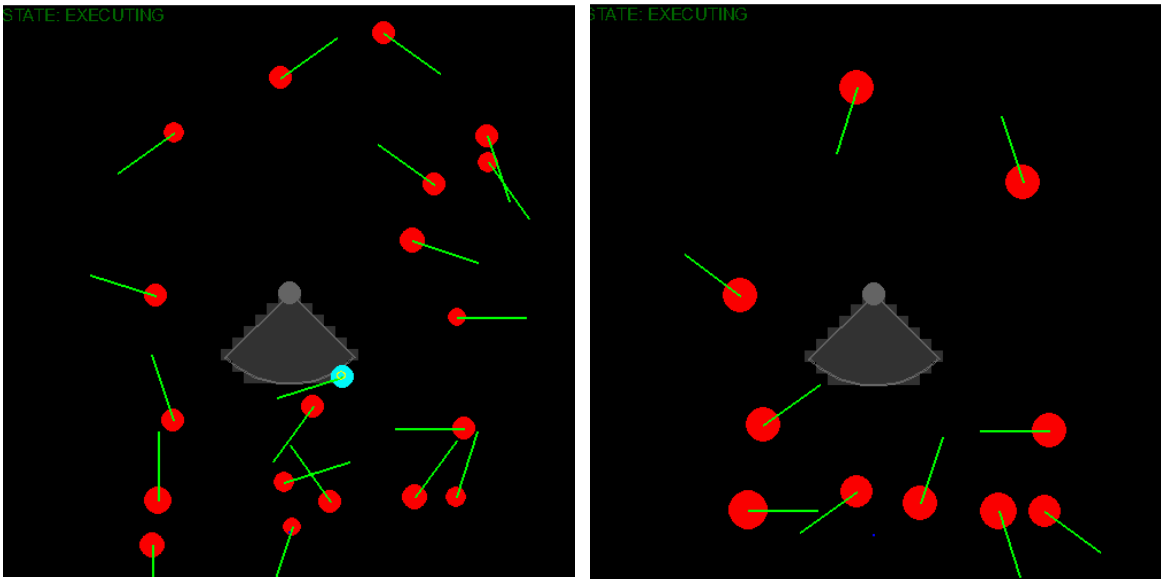


(a) The correlation between obstacle density metric and success rate in all maps from the controlled map dataset.

(b) The correlation between obstacle density metric and success rate in each group of maps. Each group includes maps with the same obstacle velocity.

Figure 5-2: The correlation changes between obstacle density metric and success rate of planner *FullRange+Local Primitive* after dividing the maps into groups based on the obstacle velocity.

However, the monotonic relationship between obstacle density and the success rate still fails to exhibit consistency across the entire obstacle density range. Notably, we see in Figure 5-2 an increase in the success rate within metric ranges of 2.0 to 3.0 and 6.0 to 7.0 as the obstacle density metric intensifies. Upon investigating these cases, we discerned that this anomaly is the heightened success rate achieved by planners in environments with fewer but larger obstacles compared to those with a larger number of smaller obstacles. This phenomenon is demonstrated in Figure 5-3.



(a) 20 obstacles with 1 m radius. **(b)** 10 obstacles with 1.5 m radius.

Figure 5-3: The performance of planners in two maps with the same obstacle density metric. The map in (a) has 20 obstacles with a 1 m radius, and the map in (b) has ten obstacles with a 1.5 m radius. The planners achieve a higher success rate in the second map with a smaller obstacle density.

5-1-2 Traversability

The concept of traversability is yet another metric utilized for assessing the complexity of maneuvering through a static environment [36, 37]. Introduced in Section 3, traversability is described as the distance that a MAV can cover starting from a randomly chosen point and moving in a randomly selected direction. In a static environment, this metric essentially equates to executing multiple tasks with different starting positions and trajectories and calculating the average distance traversed. Given the understanding that a densely populated environment poses greater navigational challenges, we would expect the traversability metric to have a similar performance to that of the obstacle density metric.

This observation clarifies why the traversability and obstacle density metrics yield similar SRCC and CV values. Analogous to the obstacle density metric, we evaluated the traversability metric across map groups with varying obstacle velocities, as depicted in Figure 5-5. The findings demonstrate that, within controlled environments, the performance of the traversability metric mirrors that of the obstacle density metric.

5-1-3 Dynamic Traversability

The dynamic traversability metric is an evolution of the static traversability metric, devised to evaluate dynamic environments by computing the traversability at a distinct sampled time step. Nevertheless, as we pointed out while discussing the static

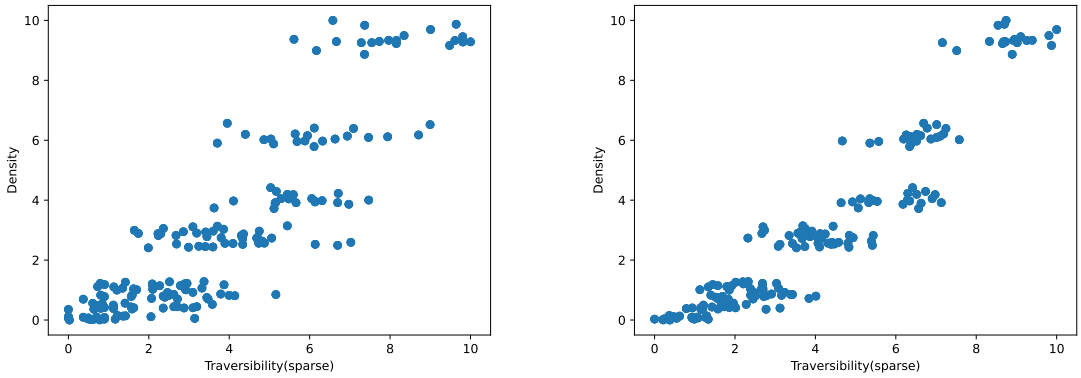
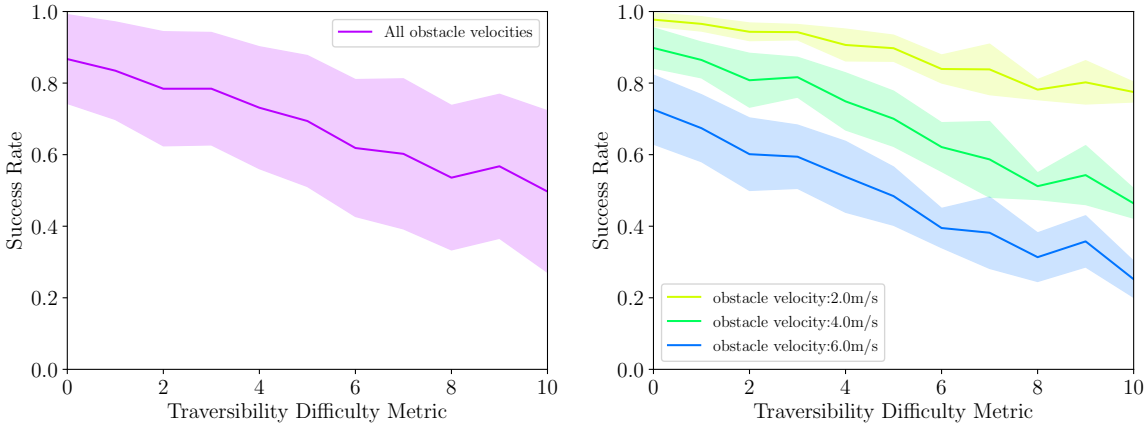


Figure 5-4: This figure demonstrates the correlation between traversability and obstacle density. A strong correlation between the traversability metric and the obstacle density can be observed with a sufficiently high sampling density.



(a) The correlation between traversability metric and success rate in all maps from the controlled map dataset.

(b) The correlation between traversability metric and success rate in each group of maps. Each group includes maps with the same obstacle velocity.

Figure 5-5: The correlation changes between traversability metric and success rate of planner *FullRange+Local Primitive* after dividing the maps into groups based on the obstacle velocity.

traversability metric, its performance closely mirrors that of the obstacle density metric. Consequently, calculating the traversability at each time step basically equates to computing the obstacle density at each time step. This approach fails to encapsulate the dynamic character of the environment. Hence, the dynamic traversability metric doesn't significantly enhance the basic traversability metric.

5-1-4 VO feasibility

The Velocity Obstacle (VO) feasibility metric, a new introduction in our study, aims to assess the difficulty of collision avoidance in dynamic settings. Since different configurations of obstacle velocities can result in substantial variations in VO feasibility

regions, there will be notable discrepancies in metric values. Consequently, the VO feasibility metric correlates more significantly with the success rate than traversability and obstacle density metrics. This makes it a more reliable indicator of the difficulty of collision avoidance in dynamic environments.

However, upon closer examination of the scatter plot of the VO feasibility metric against each map's success rate, as depicted in Figure 5-6, we observe the presence of distinct layers of data points. These layers are representative of maps that exhibit different obstacle velocities.

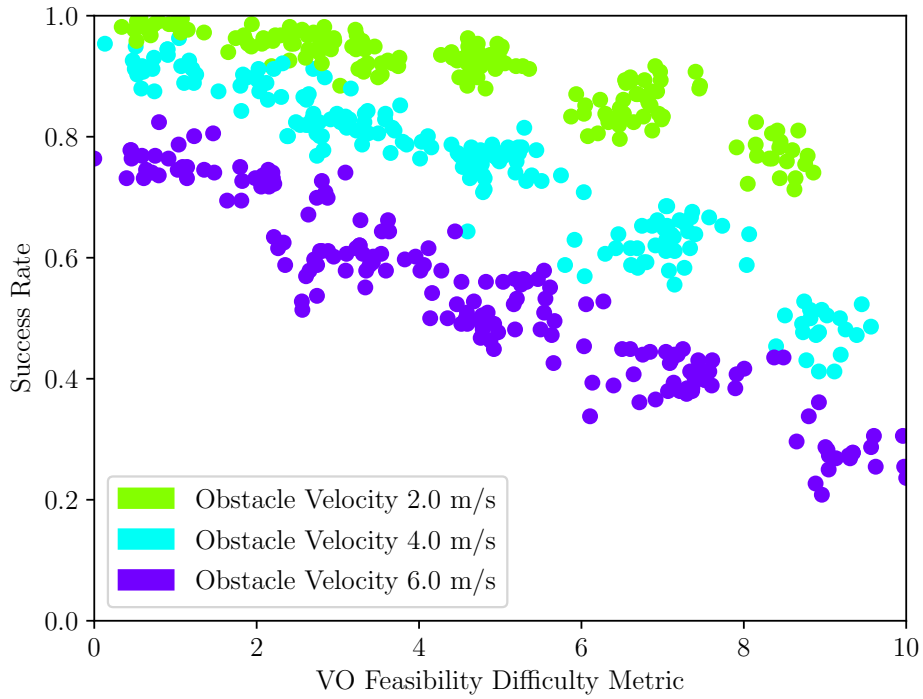


Figure 5-6: The scatter plot of the VO feasibility metric (3-1-4) and the success rate of planner *FullRange+Local Primitive* for each map. The layers of points correspond to the maps with different obstacle velocities.

This figure underscores that the VO feasibility metric does not wholly encapsulate the impact of varying obstacle velocities on collision avoidance performance. This limitation can be traced back to the generation process of VO infeasible areas. Consider a scenario where two maps feature an obstacle at an identical position with different velocities. In this case, the VO infeasible area merely shifts along the direction of the obstacle's velocity without altering its size. This behavior is depicted in Figure 5-7. The metric's inability to reflect variations in the obstacle velocity inhibits its ability to accurately gauge collision avoidance's difficulty in dynamic environments.

5-1-5 Survivability

The Survivability metric, introduced in this study, measures the complexity of executing collision avoidance within dynamic environments. It boasts the highest SRCC and

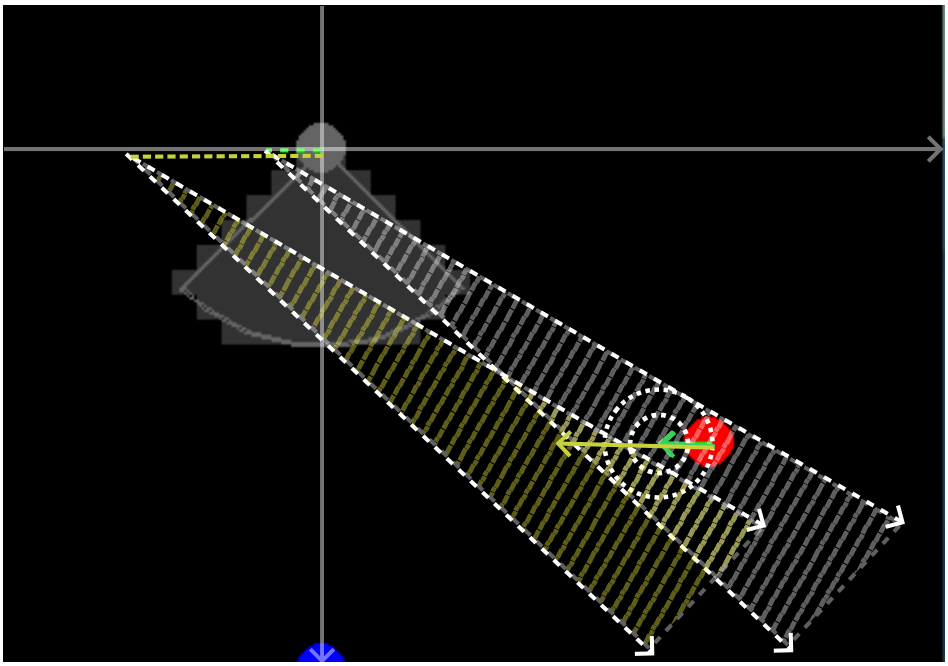


Figure 5-7: The VO infeasible areas for different obstacle velocities. The yellow shallow represents the infeasible area after increasing the obstacle velocity. The size of the infeasible area does not change compared with the original infeasible area in grey.

lowest CV values among all metrics evaluated. This finding suggests a robust relationship between the survivability metric and success rate, indicating that an increase in survivability results in a lower success rate. Moreover, the success rate remains relatively consistent when evaluated under the same survivability metric. Consequently, we can infer that the survivability metric proves to be an effective tool for gauging the difficulty of collision avoidance in our Controlled Map Dataset.

We further tested the survivability metric in three types of uncontrolled maps to relax some assumptions made in the Controlled Map Dataset. These types of maps are characterized by varying obstacle velocities, different obstacle sizes, and maps incorporating the Reciprocal Velocity Obstacles (RVO) motion profiles. As illustrated in Figures 4-14, 4-15, and 4-16, the survivability metric still displays a robust correlation with the success rate, even under these less controlled conditions. Upon analysis of our data, we also made a summary of how many data points lie within one standard deviation, two standard deviations, and three standard deviations from the trend line. The results are shown in Table 5-3.

Notably, a few outliers are evident in these figures. The following sections will delve deeper into the reasons behind these anomalies.

In examining uncontrolled maps with differing obstacle velocities, Figure 5-8 shows that the success rate falls below the trend line when looking at lower survivability metrics. As the velocities of the obstacles are derived from a distribution in this case, there is always a possibility of encountering obstacles moving at high speeds. These fast-moving obstacles present considerable hazards, profoundly impacting collision avoidance per-

Type of Map	Within 1 σ	Within 2 σ	Within 3 σ
Controlled	68%	95%	99.7%
Uncontrolled (Different Velocities)	76.3%	98.5%	100%
Uncontrolled (Different Sizes)	71.1%	94.8%	100%
Uncontrolled (RVO)	63.7%	92.5%	98.0%

Table 5-3: The percentage of Uncontrolled Map Dataset data points within one standard deviation, two standard deviations, and three standard deviations from the Gaussian distribution fitted in Controlled Map Dataset.

formance. However, the survivability metric may not adequately account for these scenarios, leading to an observed deviation from the anticipated trend.

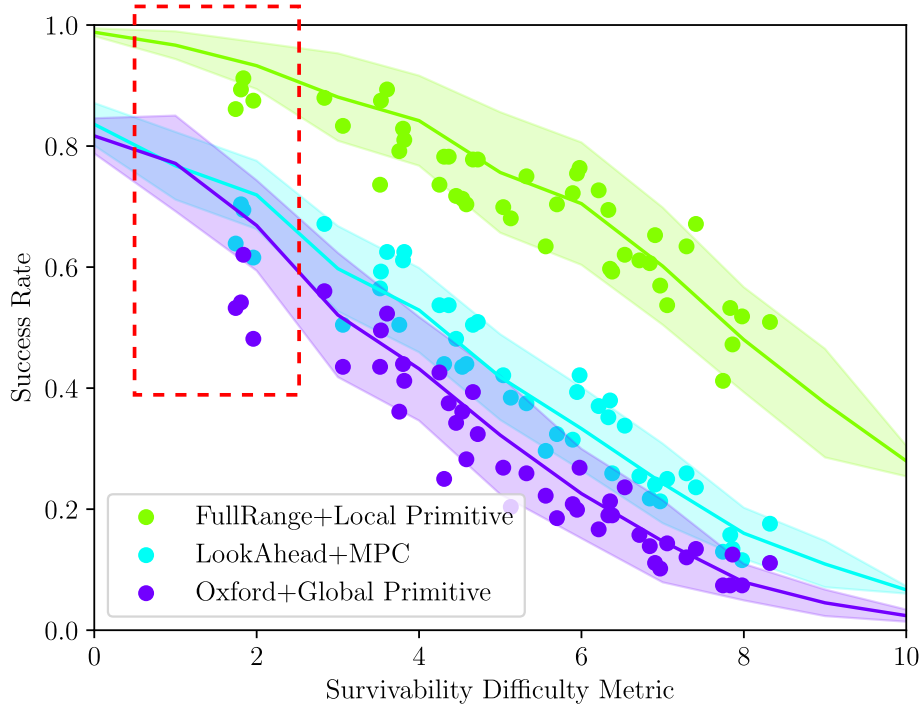


Figure 5-8: The scatter plot displays the relationship between the survivability metric (3-1-5) and the success rate of the planner, specifically *FullRange+Local Primitive*, for each map featuring different obstacle velocities. Noticeably, at lower levels of survivability metrics (within the red dashed rectangle), the success rate drops below the projected trend line.

In the Uncontrolled Maps with RVO motion profiles, we observe a considerable variance in the success rate of the *Global Primitive* planner in Figure 4-16. However, the other two trajectory planners demonstrate far less variance. In the original paper on *Global Primitive* [14], the planner was tested in dynamic maps where the future trajectories of all obstacles were known. Similarly, the obstacles move at constant velocities in our Controlled Map Dataset. Once detected, their future trajectories are known. However, the future trajectories are unknown in the Uncontrolled Maps with RVO motion profiles. Consequently, the *Global Primitive* planner requires frequent replanning, leading to significant variance in the success rate. The other two planners, in contrast, can

frequently replan without a substantial impact on their performance. Therefore, we can conclude that, despite the variances observed in the *Global Primitive* planner, the survivability metric remains suitable for evaluating the difficulty of collision avoidance in the Uncontrolled Map Dataset with RVO motion profiles.

5-1-6 Global Survivability

Global Survivability is proposed to capture the local difficulty associated with collision avoidance. However, it does not have a stronger correlation with the success rate than the original Survivability metric. The Global Survivability metric operates by sampling multiple positions within the map. If any of these sampled positions are found to be in a collision state, the entire map is deemed to be in a collision state. Consequently, most of the maps are labeled as difficult under this metric.

In Figure 4-10, we can observe such a pattern: the variance at low levels of Global Survivability is small, implying that for maps deemed to have low difficulty, the success rate is relatively consistent. However, at high levels of Global Survivability, the substantial variance indicates a significant discrepancy in the success rates among maps perceived as difficult. This demonstrates the limitation of the Global Survivability metric in accurately gauging the difficulty across various maps.

5-2 Use Case of Metrics

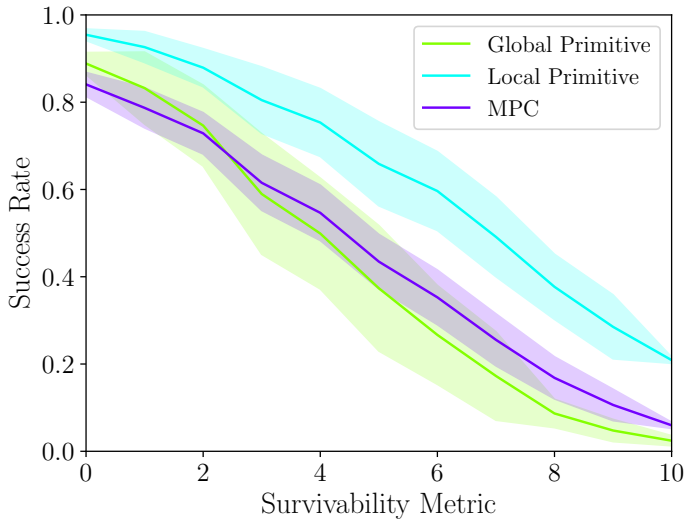
In this section, we delve into the applications of the Survivability metric, focusing specifically on its role in evaluating and comparing different collision avoidance strategies.

5-2-1 Comparison the Performance of Different Planners Using Survivability

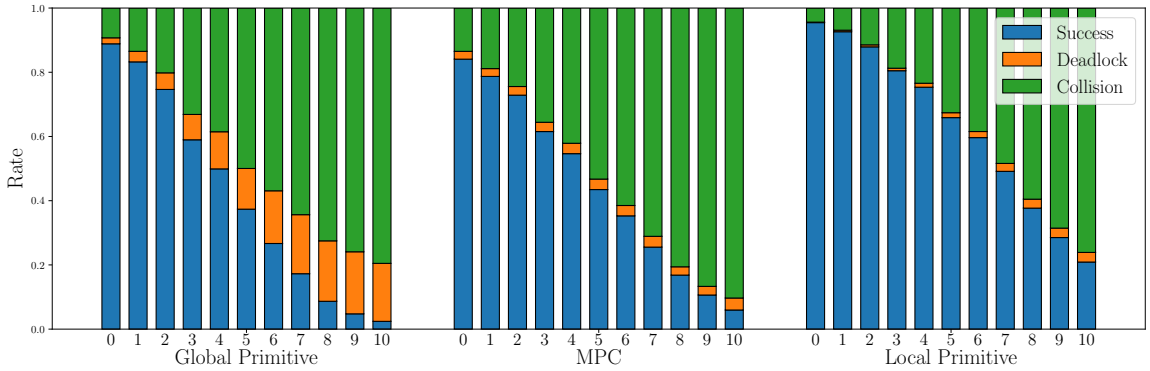
Comparison of Trajectory Planners

In the experimental portion of our work, we focused on the detailed reproduction and testing of three distinct trajectory planners: *Local Primitive*, *Global Primitive*, and *Model Predictive Control (MPC)*. Utilizing the Survivability metric as a comparative measure, we can evaluate and contrast the performance of these three planners in the context of the Controlled Map Dataset.

Figure 5-9a presents the success rates of the three trajectory planners under varying levels of the survivability metric. It is clear that the *Local Primitive* planner has the highest success rate across all levels of survivability. Comparatively, the success rate of *Global Primitive* surpasses that of the *Model Predictive Control (MPC)* when the survivability metric is low. However, as the survivability metric rises, the success rate of *MPC* exceeds that of the *Global Primitive*.



(a) The success rate of three trajectory planners under different survivability metrics.



(b) Three trajectory planners' success and deadlock rates under different survivability metrics. Each bar represents one planner's success and deadlock rates under one survivability metric.

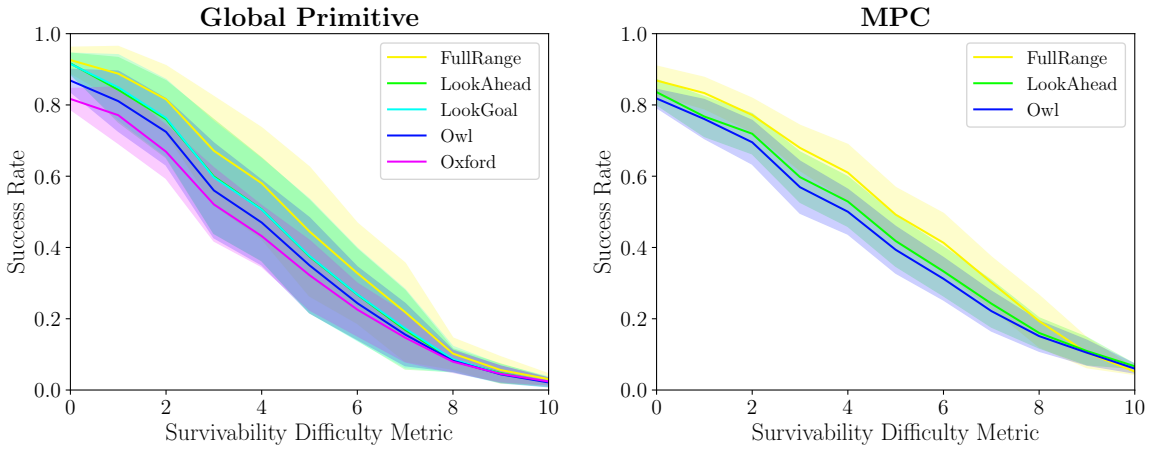
Figure 5-9: The comparison of three trajectory planners using the survivability metric.

This phenomenon is further illustrated in Figure 5-9b, which displays each planner's deadlock rate in addition to the success rate. Under high survivability metrics, the deadlock rate for *Global Primitive* sharply increases, while the deadlock rate for *MPC* remains steady, regardless of the survivability level. This suggests that *MPC* maintains a higher level of robustness than *Global Primitive* when confronted with increasingly challenging dynamic environments.

It can be concluded that under the assumptions we made in section 5-1 (2D space; Only dynamic obstacles; No perception and control error), the *Local Primitive* planner is the most robust. The *MPC* is worse than *Local Primitive* when the survivability metric is low, but it is more robust than *Global Primitive* when the survivability metric is high.

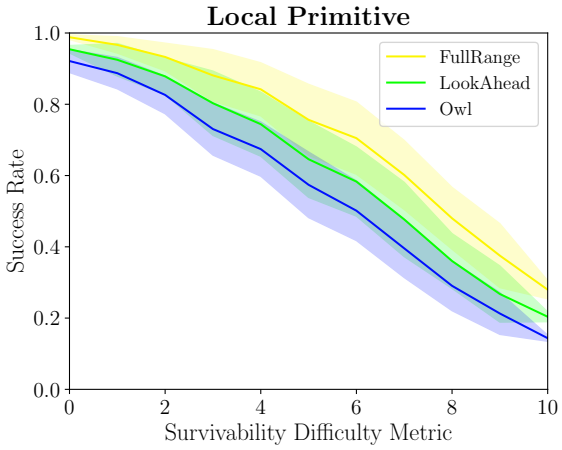
Comparison of Gaze Planners

We also reproduced and experimented with several gaze planners in the experiment part.



(a) The success rate of *Global Primitive* with different gaze planners under different survivability metrics.

(b) The success rate of *MPC* with different gaze planners under different survivability metrics.



(c) The success rate of *Local Primitive* with different gaze planners under different survivability metrics.

Figure 5-10: The comparison of gaze planners using the survivability metric.

Among the gaze planners being studied, the *FullRange* assumes the MAV has a complete 360-degree observation of its surroundings, ideally representing the upper limit of performance for all other gaze planners. In comparison, all the other gaze planners only have a limited Field Of View (FOV). Under the assumptions we made in section 5-1 (2D space; Only dynamic obstacles; No perception and control error), the *LookAhead* gaze planner performs the best across all tested scenarios. Moreover, its superiority holds consistent across varying levels of the survivability metric, reinforcing that the *LookAhead* gaze planner excels at collision avoidance in dynamic environments.

5-2-2 Generating Controlled Maps with Predefined Difficulty Levels

In the preceding section, we demonstrated how the survivability metric can be employed to compare the performance of different planners. This section will illustrate how to generate controlled maps featuring predefined levels of survivability difficulty.

In the Controlled Map Dataset, four variables are needed to generate a controlled map: the number of obstacles n_{obs}, the size of obstacles r_{obs}, the velocity of obstacles v_{obs}, and the map ID (which determines the random seed for initialization). The first three variables are the primary factors influencing the survivability metric. We can investigate the relationship between these three variables and the survivability metric in the Controlled Map Dataset. Subsequently, we can utilize this relationship to generate controlled maps with predefined levels of survivability difficulty metric. We employ a linear regression model to capture the relationship between the survivability metric and these three variables:

$$S = f(n_{obs}, r_{obs}, v_{obs}) = \beta_0 + \beta_1 n_{obs} + \beta_2 r_{obs} + \beta_3 v_{obs} \tag{5-1}$$

The prediction outcomes from the linear regression model are visualized in Figure 5-11. The model boasts an R^2 score of 0.95, indicating an excellent fit to the data. The parameters obtained from this model are listed in Table 5-4.

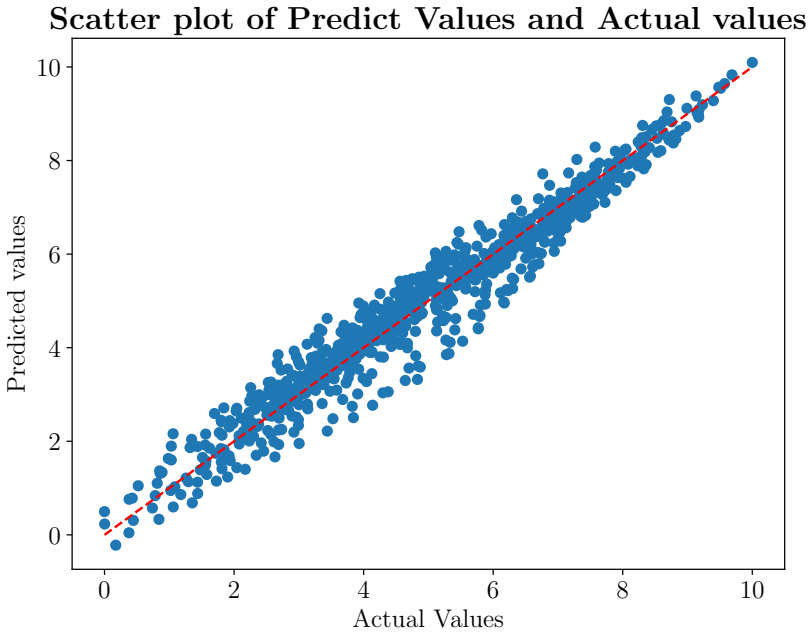


Figure 5-11: The prediction results of the linear regression model.

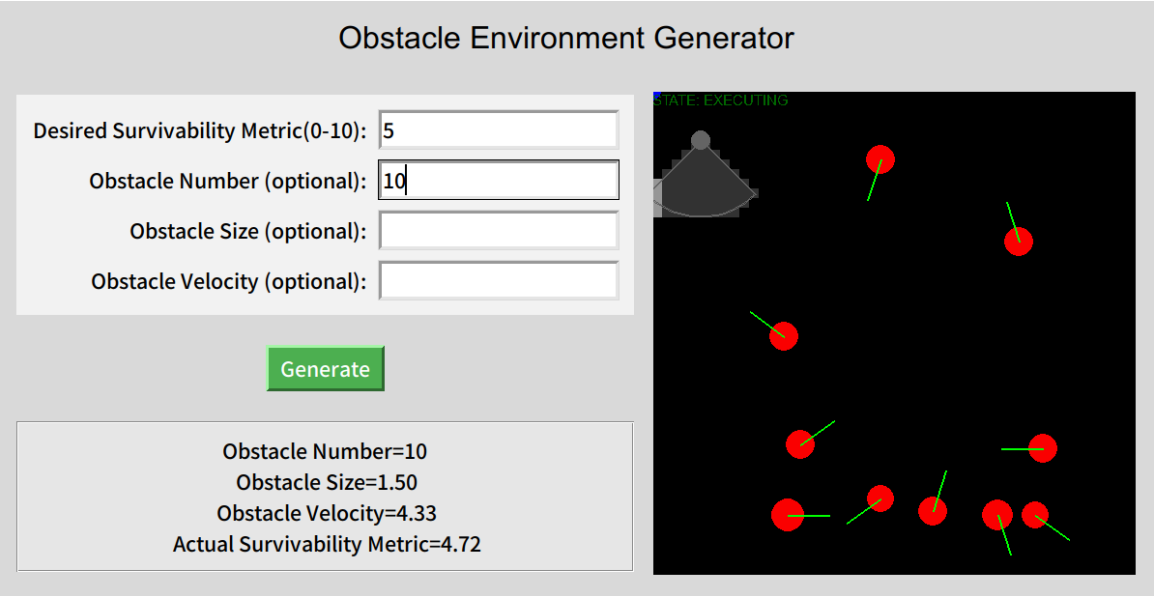
β_0	β_1	β_2	β_3
-6.014	0.226	2.646	1.104

Table 5-4: The fitting results of the parameters in the linear regression model

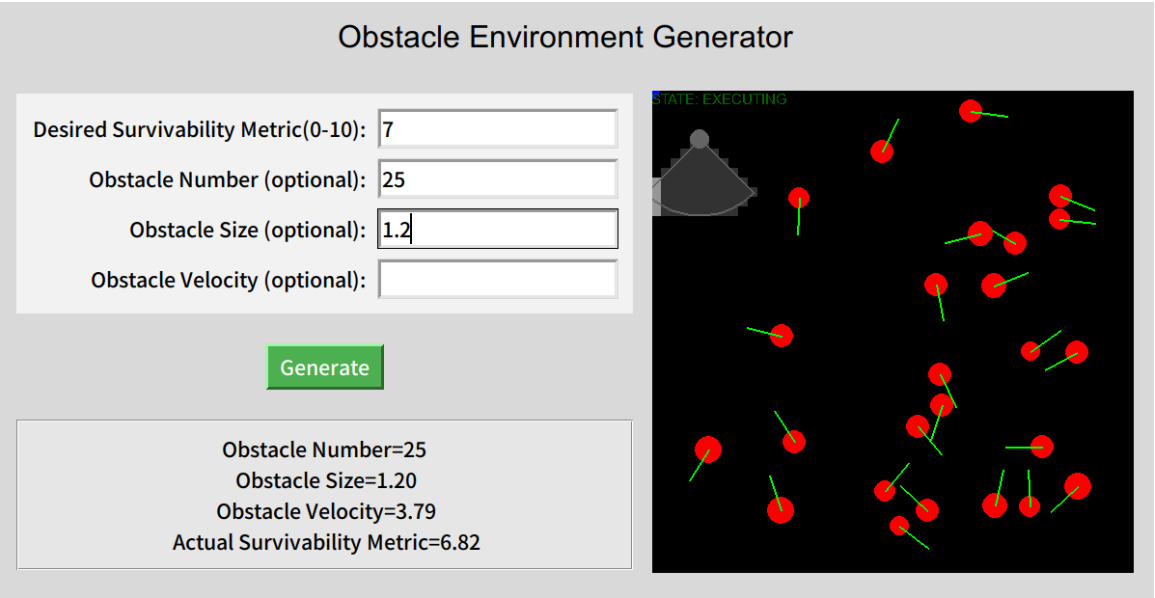
Using the fitted model f, we have designed a Graphical User Interface (GUI) that generates maps with a specified survivability metric S. The map generation process is formulated as an Integer Linear Programming (ILP) optimization problem:

$$\begin{aligned} \text{minimize} \quad & |f(n_{obs}, r_{obs}, v_{obs}) - S| && \text{(5-2)} \\ \text{s.t.} \quad & n_{obs} \in \mathbb{Z} && \text{(5-3)} \\ & 10 \leq n_{obs} \leq 30 && \text{(5-4)} \\ & 0.5 \leq r_{obs} \leq 1.5 && \text{(5-5)} \\ & 2 \leq v_{obs} \leq 6 && \text{(5-6)} \\ & && \text{(5-7)} \end{aligned}$$

The user can input the desired survivability metric S and generate a map with the closest survivability metric to S. Or the user can also specify one or two variables together with the survivability metric to generate a map with the closest survivability metric to S and the specified variable(s). Figure 5-12a and Figure 5-12b are two examples of generating a map with the survivability metric.



(a) The generated map with the survivability metric of 5. The number of obstacles is specified to be 10.



(b) The generated map with the survivability metric of 7. The number of obstacles is specified to be 25, and the radius is specified to be 1.2 m.

Figure 5-12: Two examples of generating controlled maps with predefined survivability metrics.

5-2-3 Calculating the Difficulty Level of a Map

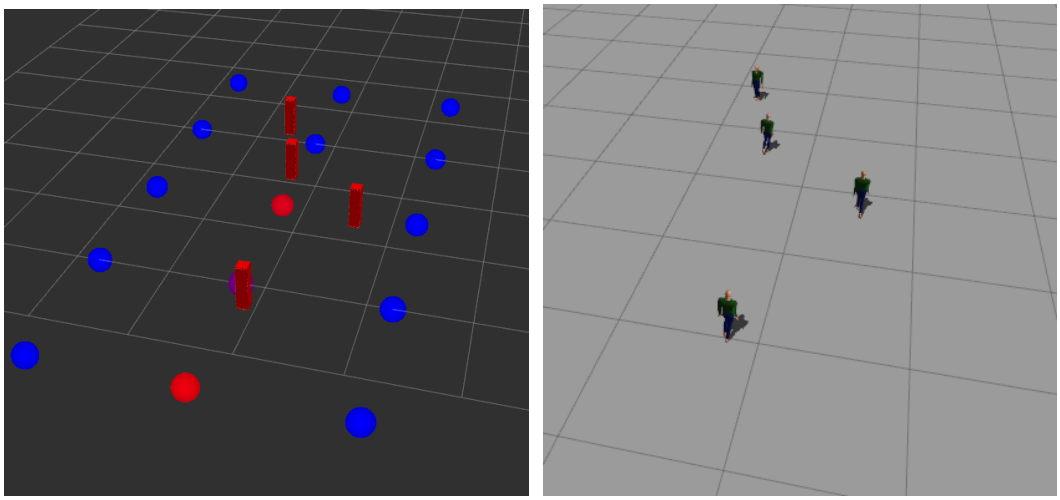
This metric can also be easily utilized in other simulators or real-world environments for comparing different collision avoidance methods as long as two conditions are met:

1. The environment is deterministic in its reproduction. Given a specific initialization or a predetermined set of parameters, the movement of dynamic obstacles

within the environment remains consistent and unchanged every time the environment is rerun.

2. It is possible to pose a robot in the environment and do the collision checking.

Figure 5-13 provides an example of calculating the survivability metric in a gazebo simulation.



(a) The visualization of the survivability metric calculation. The pillars represent the moving pedestrians. The spheres represent the sampled MAV positions, the red spheres represent the sampled MAV positions in the collision state, and the blue spheres represent the sampled MAV positions that still survive.

(b) The corresponding state in the gazebo simulation environment. The world is with four moving pedestrians.

Figure 5-13: The calculation of the survivability metric in a gazebo simulation environment. The calculated survivability metric of this map is 2.3.

The methodology for calculating survivability difficulty levels offers a robust approach to testing and analyzing various planning algorithms. It can be seamlessly integrated as long as the environment adheres to the stated conditions, ensuring consistent dynamic obstacle behavior and enabling accurate collision checking. Such a framework paves the way for consistent and reliable comparisons across different planners in varied simulations and real-world scenarios.

5-3 Problem Difficulty

While our primary focus has been on evaluating environmental difficulty, it's essential to consider the broader concept of problem difficulty. The environment does not solely determine problem difficulty; it is a composite measure influenced by a range of factors, including specific robotic setups and task configurations. For instance, properties of the robot, such as the size of the MAV and its maximum speed, as well as task configurations

like the start and target positions, play a pivotal role in the overall challenge a planner faces.

One noteworthy finding regarding problem difficulty is presented in this section about the significance of the velocity ratio between the MAV and the obstacles shown in Figure 5-14.

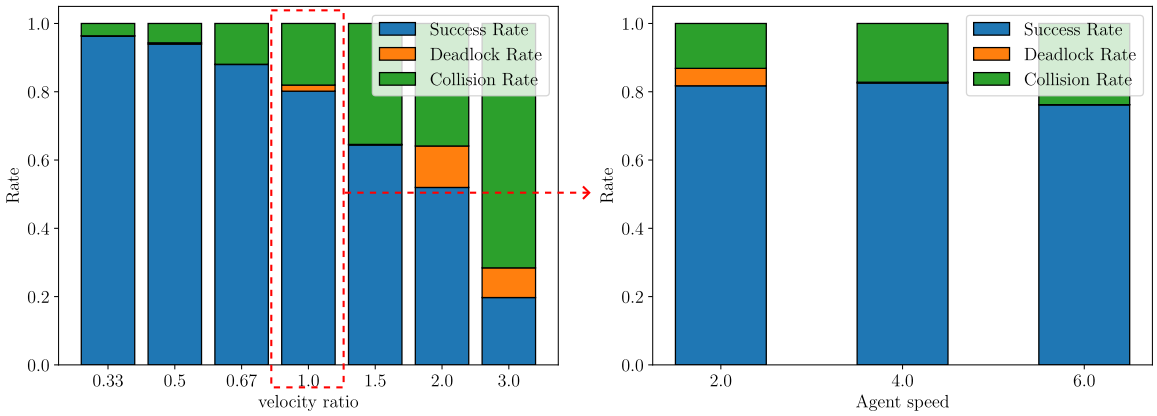


Figure 5-14: Performance of the planner *FullRange+Local Primitive* under different velocity ratios. The first figure shows that the velocity ratio between the MAV and the obstacles is a significant factor in determining the performance of the planner *FullRange+Local Primitive*. The second figure shows that given the same velocity ratio($ratio = \frac{v_{obs}}{v_u} = 1$ in this case), the planner's performance is not significantly affected by the obstacle velocity(Agent speed).

Our research delves deep into the intricacies of the velocity ratio and its profound impact on a planner's effectiveness in collision avoidance. Interestingly, while the inherent velocities of obstacles may seem like a natural determinant of the problem's complexity, our results paint a nuanced picture. Specifically, when the velocity ratio is held constant, a mere increment in obstacle velocity doesn't automatically translate to a heightened challenge in collision avoidance for a planner. This underscores the intricate dance of relative speeds and their role in shaping real-world outcomes. For instance, an obstacle moving at a high velocity may not necessarily be harder to evade if the MAV has a corresponding increase in speed, maintaining its velocity ratio. On the other hand, if the MAV's speed remains stagnant while obstacle speeds increase, the planner might face significant challenges.

Therefore, to understand a planner's efficiency and ability to navigate dynamic environments, it's essential to consider individual velocities and emphasize the velocity ratio between the MAV and the obstacles. This factor, often overlooked, could be the keystone in fine-tuning planners for optimal performance in varied scenarios.

Chapter 6

Conclusion and Future Works

In this chapter, we will summarize the contributions of this thesis and discuss possible future work for evaluating dynamic environment difficulty for collision avoidance problems.

6-1 Conclusion

In this thesis, we have proposed various metrics for evaluating the environmental difficulty of dynamic environments for collision avoidance problems. We validate their effectiveness through a large number of experiments on our custom simulator and provide a detailed analysis of the results, aiming to demonstrate insights into the limitations of these metrics and their potential applications. In diving deeper into the metrics presented in this thesis, it is crucial to understand the specific scenarios or applications where each metric excels:

- **Obstacle Density:** The essence of this metric is its suitability for static environments. It provides a quick insight into the density of obstructions and barriers, and while straightforward, its limitations become evident in dynamic environments. Our extensive testing reveals that when dynamic elements come into play, especially those with varying velocities, the metric struggles to represent the environment's difficulty accurately. Even when the velocities of obstacles are kept uniform, the metric's inadequacy persists, underscoring its best fit for purely static settings.
- **Traversability:** This metric is proposed for evaluating the difficulty of static environments. It is a more sophisticated metric than obstacle density, as it considers the agent's size and obstacles. However, it still has the same limitations as obstacle density, as it is unsuitable for dynamic environments. The metric's

inability to capture the difficulty of dynamic environments is evident in our experiments, where it fails to provide any meaningful insights into the difficulty of the environment. We further investigate that when the density of traversability samples is high, the metric can strongly correlate with Obstacle Density, further highlighting its unsuitability for dynamic environments.

- **VO Feasibility:** This metric is designed to assess the challenges posed by dynamic environments. Nonetheless, our analysis reveals that it struggles to encapsulate the complexities introduced by escalating obstacle velocities. It is most apt for situations where obstacle velocities remain consistent. In these scenarios, the metric effectively discerns the intricacy of the environment.
- **Survivability:** Designed specifically to gauge the complexity of dynamic environments, the Survivability metric stands out as the most effective tool introduced in this thesis. Its versatility and accuracy are further confirmed through rigorous tests across diverse scenarios. Notably, it remains robust and insightful across environments with varying obstacle velocities, differing obstacle sizes, and when the obstacles follow the RVO motion profile. This underscores its adaptability and precision in evaluating dynamically challenging terrains.

In essence, the applicability of each metric is intrinsically tied to the nature of the environment and its specific challenges. We suggest the Survivability metric for assessing dynamic environment difficulty, especially when collision checks can be simulated within the environment. Furthermore, if users can adapt their planner to our tailored environment, they can create additional maps with predetermined difficulty levels, leading to a more comprehensive evaluation of their planning strategies. In contexts such as real-world testing, simulating collision check becomes costly due to potential hardware damages. Furthermore, reproducing identical environmental settings, especially ensuring that dynamic obstacles maintain consistent initial positions and trajectories across resets, can be particularly challenging. In this case, the survivability metric might not be the most suitable choice.

For a swift assessment of a dynamic environment where obstacle velocities are relatively consistent, both the Obstacle Density and VO Feasibility metrics serve as suitable choices. Recognizing these subtleties allows us to select the most appropriate metric for a given situation, ensuring a more precise evaluation of collision avoidance mechanisms in dynamic settings.

6-2 Future Research

The landscape of collision avoidance in dynamic environments is ever-evolving, with numerous facets still ripe for exploration. Our study has laid the groundwork, systematically evaluating various metrics to determine environmental difficulty. However, as with any burgeoning research area, many avenues are yet to be traversed. In this section, we outline potential future directions, aspiring to both refine the metrics we've introduced and broaden the horizons of this research domain.

6-2-1 Metrics Refinement

In this thesis, we have proposed several metrics for evaluating dynamic environment difficulty for collision avoidance problems, and we finally pick the Survivability metric as the most effective one and conduct a generalization test on it. However, as introduced in Section 5-1, there are still many limitations to the testing scenarios. We can further refine the metrics by conducting more experiments on different scenarios:

- 3D environments.
- Environments with different obstacle shapes.
- Environments with both static and dynamic obstacles.

Under these scenarios, we also need to adjust the design of our metrics to make them more suitable for these scenarios.

6-2-2 Furture Analysis on Problem Difficulty

As mentioned in Section 5-3, the problem difficulty includes all the factors that could affect the performance of a collision avoidance system and are not related to the property of the collision avoidance method itself. In this thesis, we have only focused on environmental difficulty, which is only one of the factors that could affect the performance of a collision avoidance system. In the future, we can also investigate the problem's difficulty and try to find a way to evaluate it.

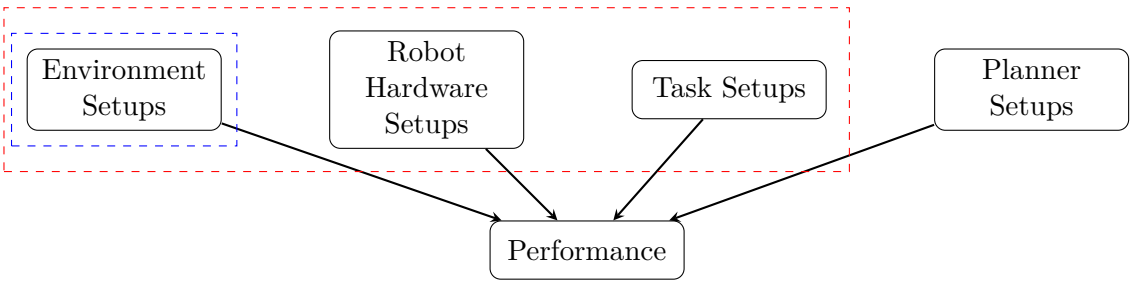


Figure 6-1: All factors that could affect the performance of a collision avoidance system. The blue dashed rectangle represents the environmental difficulty which is the focus of this thesis. The red dashed rectangle represents the problem difficulty which is the focus of future research.

Appendix A

Trajectory Planning Algorithms

This appendix contains details of the algorithms used to generate the trajectories for the UAV including the global motion primitives [14], local motion primitives [17], and MPC [12]. All these trajectory planners share the format of inputs and outputs as shown in Table A-1.

Input Parameter	**Symbol**
Target Position	$\vec{p_t} = (x_t, y_t)$
Start Position	$\vec{p_0} = (x_0, y_0)$
Start Velocity	$\vec{v_0}$
Occupancy Map	$\mathcal{M} = \{m_{ij}\}$, $m_{ij} = \begin{cases} 1 & \text{if not occupied} \\ 0 & \text{if unexplored} \\ -1 & \text{if occupied} \end{cases}$
Active Trackers	$\mathcal{T} = \left\{\vec{t_i}\right\}$, $\vec{t_i} = \{\vec{p_i}, \vec{v_i}, \Sigma_{\vec{p_i},\vec{v_i}}\}$

(a) Input parameters of the trajectory planners

Output Results	**Symbol**
Future Trajectory	$\mathcal{F} = \{\vec{p_i}, \vec{v_i}, \vec{a_i}\}$

(b) Output results of the trajectory planners

Constraint Parameters	**Symbol**
Maximum MAV Velocity	v_{max}
Maximum MAV Acceleration	a_{max}
MAV Radius	r_u

(c) Other constraint parameters of the trajectory planners

Table A-1: Inputs, outputs, and other parameters of constraints of the trajectory planners

A-1 Global Motion Primitives

In Global Motion Primitives, the trajectories are decomposed into several motion primitives, each with constant acceleration. The duration of all primitives is T. The connection between two trajectories is defined as a node with the position and velocity of the end of the first trajectory and the start of the second trajectory. The Global Motion Primitives algorithm tries to expand a graph from the starting node to the goal node:

Algorithm 2 Global Motion Primitives Algorithm

Input: Starting position $\vec{p}_0$, starting velocity $\vec{v}_0$, target position $\vec{p}_\mathrm{t}$, duration T, sampled accelerations $\mathcal{A}$
Output: Path from start to target
Initialize open list $\mathcal{O}$ with node $N(\vec{p}_0, \vec{v}_0, 0)$
Initialize closed list $\mathcal{C} = \emptyset$
while $\mathcal{O}$ is not empty **do**
    N_current = node from $\mathcal{O}$ with the lowest cost
    Move N_current from $\mathcal{O}$ to $\mathcal{C}$
    if N_current.*position* is close to $\vec{p}_\mathrm{target}$ **then**
        return reconstructed path from $N(\vec{p}_0, \vec{v}_0, 0)$ to N_current, add to $\mathcal{F}$
    end if
    for each $\vec{a}$ in $\mathcal{A}$ **do**
        Generate trajectory τ using $\vec{a}$ over duration T starting from N_current.*position* and N_current.*velocity*
        $\vec{p}_\mathrm{new}$, $\vec{v}_\mathrm{new}$ = endpoint of τ
        $\mathrm{cost}_\mathrm{new}$ = cost of N_current + cost of τ
        if τ is collision-free and $N(\vec{p}_\mathrm{new}, \vec{v}_\mathrm{new}, \mathrm{cost}_\mathrm{new})$ is not in $\mathcal{C}$ **then**
            Add $N(\vec{p}_\mathrm{new}, \vec{v}_\mathrm{new}, \mathrm{cost}_\mathrm{new})$ to $\mathcal{O}$
        end if
    end for
end while
return no path found

The heuristic cost of each node is derived from the time-optimal control problem:

$$C = \frac{12\left\|\vec{p}_t - \vec{p}\right\|^2}{T^3} - \frac{12\vec{v}\cdot(\vec{p}_t - \vec{p})}{T^2} + \frac{4\left\|\vec{v}\right\|^2}{T} + \rho T \tag{A-1}$$

where $\vec{p}$ and $\vec{v}$ denote the position and velocity of the node. The MAV will follow the trajectory after getting the future trajectory $\mathcal{F}$. At each timestep, the MAV will execute the collision check based on the updated occupancy map $\mathcal{M}$ and the active trackers $\mathcal{T}$:

$$\left\|\vec{p} - (\vec{p}_i + k\Delta t\vec{v}_i)\right\| \geq r_i, \forall t_i \in \mathcal{T} \tag{A-2}$$

$$m(\vec{p_i}) \neq -1 \tag{A-3}$$

If every point on the future trajectory is collision-free, the MAV will execute the trajectory. Otherwise, the MAV will start replanning.

A-2 Local Motion Primitives

Local Primitives do not generate full trajectories toward the target position in one shot. Instead, they generate a motion primitive from the current position to a temporary target position with minimum heuristic cost. The planning process can be explained in Algorithm 3.

Algorithm 3 Local Primitives with Optimal Jerk Control

Input: Initial position $\vec{p}_0$, velocity $\vec{v}_0$, and acceleration $\vec{a}_0$; final acceleration $\vec{a}_f$; duration T; time steps t
Output: Optimal collision-free trajectory
Generate temporary target list $\mathcal{P}_t$
for $\vec{p}_{tt} \in \mathcal{P}_t$ **do**
    Generate a trajectory τ from $\vec{p}_0$ to $\vec{p}_{tt}$ using the optimal jerk control [15]
end for
Sort temporary targets by their heuristic costs $h(\tau)$ in ascending order
for each ranked temporary target **do**
    If its trajectory is collision-free:
        Return the trajectory
end for

The list of temporary targets is generated by sampling around the current position of the MAV with a fixed radius and uniformly sampled angles:

$$\mathcal{P}_t = \left\{ \vec{p}_0 + \begin{bmatrix} r\cos\theta \\ r\sin\theta \end{bmatrix} \mid \theta \in [0, 2\pi) \right\} \tag{A-4}$$

A-3 MPC

In MPC, the state of the MAV is defined as $\vec{x}^k = \begin{bmatrix} \vec{p}^k \\ \vec{v}^k \end{bmatrix}$, where $\vec{p}^k$ and $\vec{v}^k$ are the position and velocity of MAV at time step k. The control input is defined as $\vec{u}^k = \begin{bmatrix} \vec{a}^k \end{bmatrix}$, where $\vec{a}^k$ is the acceleration of MAV at time step k. The dynamics of the MAV can be defined as:

$$\vec{x}^{k+1} = \vec{f}\left(\vec{x}^k, \vec{u}^k\right) = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \vec{x}^k + \begin{bmatrix} \frac{\Delta t^2}{2} & 0 \\ 0 & \frac{\Delta t^2}{2} \\ \Delta t & 0 \\ 0 & \Delta t \end{bmatrix} \vec{u}^k \tag{A-5}$$

where Δt is the time step. The MPC formulation is as follows:

$$\min_{\vec{x}^{1:N},\vec{u}^{0:N-1}} \sum_{k=0}^{N} J^k\left(\vec{x}^k,\vec{u}^k\right) + J^N\left(\vec{x}^N\right) \tag{A-6}$$

$$\text{s.t. } \vec{x}^0 = \vec{x}(0), \quad Vecx^k = \vec{f}\left(\vec{x}^{k-1},\vec{u}^{k-1}\right), \tag{A-7}$$

$$\vec{u}^{k-1} \in \mathcal{U}, \quad \vec{x}^k \in \mathcal{X}, \tag{A-8}$$

$$\forall k \in \{1,\ldots,N\}. \tag{A-9}$$

where J^k is the cost function at time step k, $\mathcal{U}$ and $\mathcal{X}$ are the control input and state constraints. For each tracked dynamic obstacles $t_i \in \mathcal{T}$, the collision constraint is defined as:

$$\left\|\vec{p}^k - (\vec{p}_i + k\Delta t\vec{v}_i)\right\| \geq r_i \tag{A-10}$$

Where the constant velocity model is used to estimate the future position of the obstacle, the cost function is defined as:

$$J^k\left(\vec{x}^k,\vec{u}^k\right) = \vec{x}^{k\top}Q\vec{x}^k + \vec{u}^{k\top}R\vec{u}^k \tag{A-11}$$

where Q and R are the weight matrices that can be tuned to achieve different performance.

Bibliography

[1] Wen-Chyuan Chiang, Yuyu Li, Jennifer Shang, and Timothy L Urban. "Impact of drone delivery on sustainability and cost: Realizing the UAV potential through vehicle routing optimization". In: *Applied energy* 242 (2019), pp. 1164–1175.

[2] Teodor Tomic, Korbinian Schmid, Philipp Lutz, Andreas Domel, Michael Kassecker, Elmar Mair, Iris Lynne Grixa, Felix Ruess, Michael Suppa, and Darius Burschka. "Toward a fully autonomous UAV: Research platform for indoor and outdoor urban search and rescue". In: *IEEE robotics & automation magazine* 19.3 (2012), pp. 46–56.

[3] Mark Nicholas Finean, Wolfgang Merkt, and Ioannis Havoutis. "Where Should I Look? Optimised Gaze Control for Whole-Body Collision Avoidance in Dynamic Environments". In: *IEEE Robotics and Automation Letters* 7.2 (Apr. 2022), pp. 1095–1102. ISSN: 2377-3766, 2377-3774. DOI: 10.1109/LRA.2021.3137545.. arXiv: 2109.04721.

[4] Jesus Tordesillas and Jonathan P. How. "PANTHER: Perception-Aware Trajectory Planner in Dynamic Environments". In: *IEEE Access* 10 (2022), pp. 22662–22677. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2022.3154037. arXiv: 2103.06372 [cs].

[5] Gianluca Monaci, Michel Aractingi, and Tomi Silander. "DiPCAN: Distilling Privileged Information for Crowd-Aware Navigation". In: (2022). DOI: 10/gqfrhw.

[6] Andreas Geiger, Philip Lenz, and Raquel Urtasun. "Are we ready for autonomous driving? the kitti vision benchmark suite". In: *2012 IEEE conference on computer vision and pattern recognition*. IEEE. 2012, pp. 3354–3361.

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. "ImageNet: A Large-Scale Hierarchical Image Database". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. June 2009. DOI: 10.1109/CVPR.2009.5206848.

[8] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. "Microsoft coco: Common objects in context". In: *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*. Springer. 2014, pp. 740–755.

[9] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. "GLUE: A multi-task benchmark and analysis platform for natural language understanding". In: *arXiv preprint arXiv:1804.07461* (2018).

[10] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. "Squad: 100,000+ questions for machine comprehension of text". In: *arXiv preprint arXiv:1606.05250* (2016).

[11] Gang Chen, Siyuan Wu, Moji Shi, Wei Dong, Hai Zhu, and Javier Alonso-Mora. "RAST: Risk-Aware Spatio-Temporal Safety Corridors for MAV Navigation in Dynamic Uncertain Environments". In: *IEEE Robotics and Automation Letters* 8.2 (Feb. 2023), pp. 808–815. ISSN: 2377-3766. DOI: 10.1109/LRA.2022.3231832.

[12] Hai Zhu and Javier Alonso-Mora. "Chance-Constrained Collision Avoidance for MAVs in Dynamic Environments". In: *IEEE Robotics and Automation Letters* 4.2 (Apr. 2019), pp. 776–783. ISSN: 2377-3766, 2377-3774. DOI: 10.1109/LRA.2019.2893494.

[13] Yingjian Wang, Jialin Ji, Qianhao Wang, Chao Xu, and Fei Gao. *Autonomous Flights in Dynamic Environments with Onboard Vision*. 2021. arXiv: 2103.05870 [cs.RO].

[14] Sikang Liu, Nikolay Atanasov, Kartik Mohta, and Vijay Kumar. "Search-Based Motion Planning for Quadrotors Using Linear Quadratic Minimum Time Control". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Sept. 2017, pp. 2872–2879. DOI: 10.1109/IROS.2017.8206119.

[15] Mark W. Mueller, Markus Hehn, and Raffaello D'Andrea. "A Computationally Efficient Motion Primitive for Quadrocopter Trajectory Generation". In: *IEEE Transactions on Robotics* 31.6 (Dec. 2015), pp. 1294–1310. ISSN: 1552-3098, 1941-0468. DOI: 10.1109/TRO.2015.2479878.

[16] Brett T. Lopez and Jonathan P. How. "Aggressive 3-D Collision Avoidance for High-Speed Navigation". In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. May 2017. DOI: 10.1109/ICRA.2017.7989677.

[17] Gang Chen, Wei Dong, Xinjun Sheng, Xiangyang Zhu, and Han Ding. "An Active Sense and Avoid System for Flying Robots in Dynamic Environments". In: *IEEE/ASME Transactions on Mechatronics* 26.2 (Apr. 2021), pp. 668–678. ISSN: 1083-4435, 1941-014X. DOI: 10.1109/TMECH.2021.3060511. arXiv: 2010.04977 [cs].

[18] Daniel Mellinger and Vijay Kumar. "Minimum Snap Trajectory Generation and Control for Quadrotors". In: *2011 IEEE International Conference on Robotics and Automation*. May 2011, pp. 2520–2525. DOI: 10.1109/ICRA.2011.5980409.

[19] Lars Blackmore, Masahiro Ono, and Brian C. Williams. "Chance-Constrained Optimal Path Planning With Obstacles". In: *IEEE Transactions on Robotics* (Dec. 2011). DOI: 10.1109/TRO.2011.2161160.

[20] Sikang Liu, Michael Watterson, Kartik Mohta, Ke Sun, Subhrajit Bhattacharya, Camillo J. Taylor, and Vijay Kumar. "Planning Dynamically Feasible Trajectories for Quadrotors Using Safe Flight Corridors in 3-D Complex Environments". In: *IEEE Robotics and Automation Letters* 2.3 (July 2017), pp. 1688–1695. ISSN: 2377-3766, 2377-3774. DOI: 10.1109/LRA.2017.2663526.

[21] Michael Everett, Yu Fan Chen, and Jonathan P. How. "Motion Planning Among Dynamic, Decision-Making Agents with Deep Reinforcement Learning". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Madrid, Spain, Sept. 2018. URL: https://arxiv.org/pdf/1805.01956.pdf.

[22] Michael Everett, Yu Fan Chen, and Jonathan P How. "Collision avoidance in pedestrian-rich environments with deep reinforcement learning". In: *IEEE Access* 9 (2021), pp. 10357–10377.

[23] Dawei Wang, Tingxiang Fan, Tao Han, and Jia Pan. "A Two-Stage Reinforcement Learning Approach for Multi-UAV Collision Avoidance Under Imperfect Sensing". In: *IEEE Robotics and Automation Letters* (Apr. 2020). DOI: 10.1109/LRA.2020.2974648.

[24] Jing Chen, Tianbo Liu, and Shaojie Shen. "Online Generation of Collision-Free Trajectories for Quadrotor Flight in Unknown Cluttered Environments". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. May 2016, pp. 1476–1483. DOI: 10.1109/ICRA.2016.7487283.

[25] Catrina Lim, Boyang Li, Ee Meng Ng, Xin Liu, and Kin Huat Low. "Three-Dimensional (3D) Dynamic Obstacle Perception in a Detect-and-Avoid Framework for Unmanned Aerial Vehicles". In: *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*. Atlanta, GA, USA: IEEE, June 2019, pp. 996–1004. DOI: 10.1109/ICUAS.2019.8797844.

[26] Jesus Tordesillas, Brett T. Lopez, Michael Everett, and Jonathan P. How. "FASTER: Fast and Safe Trajectory Planner for Navigation in Unknown Environments". Aug. 30, 2021.

[27] Boyu Zhou, Jie Pan, Fei Gao, and Shaojie Shen. *RAPTOR: Robust and Perception-aware Trajectory Replanning for Quadrotor Fast Flight*. July 6, 2020. DOI: 10.48550/arXiv.2007.03465. arXiv: 2007.03465 [cs].

[28] Jesus Tordesillas and Jonathan P. How. *Deep-PANTHER: Learning-Based Perception-Aware Trajectory Planner in Dynamic Environments*. Sept. 2, 2022. DOI: 10.48550/arXiv.2209.01268. arXiv: 2209.01268 [cs].

[29] Eric Heiden, Luigi Palmieri, Leonard Bruns, Kai O. Arras, Gaurav S. Sukhatme, and Sven Koenig. "Bench-MR: A Motion Planning Benchmark for Wheeled Mobile Robots". In: *IEEE Robotics and Automation Letters* (July 2021). DOI: 10.1109/LRA.2021.3068913.

[30] Hang Yu, Guido C. H. E. de Croon, and Christophe De Wagter. *AvoidBench: A High-Fidelity Vision-Based Obstacle Avoidance Benchmarking Suite for Multi-Rotors*. Jan. 18, 2023. DOI: 10.48550/arXiv.2301.07430. preprint.

[31] N. R. Sturtevant. "Benchmarks for Grid-Based Pathfinding". In: *IEEE Transactions on Computational Intelligence and AI in Games* (June 2012). DOI: 10.1109/TCIAIG.2012.2197681.

[32] Vahid Behzadan and Arslan Munir. "Adversarial reinforcement learning framework for benchmarking collision avoidance mechanisms in autonomous vehicles". In: *IEEE Intelligent Transportation Systems Magazine* 13.2 (2019), pp. 236–241.

[33] Shawn Singh, Mubbasir Kapadia, Petros Faloutsos, and Glenn Reinman. "Steerbench: a benchmark suite for evaluating steering behaviors". In: *Computer Animation and Virtual Worlds* 20.5-6 (2009), pp. 533–548.

[34] Maria Isabel Ribeiro. "Obstacle avoidance". In: *Instituto de Sistemas e Robótica, Instituto Superio Técnico* 1 (2005).

[35] Brett T Lopez and Jonathan P How. "Aggressive collision avoidance with limited field-of-view sensing". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 1358–1365.

[36] Clint Nous, Roland Meertens, Christophe De Wagter, and Guido de Croon. "Performance Evaluation in Obstacle Avoidance". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Daejeon, South Korea: IEEE, Oct. 2016, pp. 3614–3619. DOI: 10.1109/IROS.2016.7759532.

[37] Daniel Perille, Abigail Truong, Xuesu Xiao, and Peter Stone. *Benchmarking Metric Ground Navigation*. Nov. 2, 2020. preprint.

[38] Hsueh-Cheng Wang, Siao-Cing Huang, Po-Jui Huang, Kuo-Lun Wang, Yi-Chen Teng, Yu-Ting Ko, Dongsuk Jeon, and I-Chen Wu. "Curriculum Reinforcement Learning From Avoiding Collisions to Navigating Among Movable Obstacles in Diverse Environments". In: *IEEE Robotics and Automation Letters* 8.5 (2023), pp. 2740–2747.

[39] Anirudh Nair, Fulin Jiang, Kang Hou, Zifan Xu, Shuozhe Li, Xuesu Xiao, and Peter Stone. "DynaBARN: Benchmarking Metric Ground Navigation in Dynamic Environments". In: *2022 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. Nov. 2022. DOI: 10.1109/SSRR56537.2022.10018758.

[40] Linh Kästner, Teham Bhuiyan, Tuan Anh Le, Elias Treis, Johannes Cox, Boris Meinardus, Jacek Kmiecik, Reyk Carstens, Duc Pichel, Bassel Fatloun, Niloufar Khorsandi, and Jens Lambrecht. "Arena-Bench: A Benchmarking Suite for Obstacle Avoidance Approaches in Highly Dynamic Environments". In: *IEEE Robotics and Automation Letters* (Oct. 2022). DOI: 10.1109/LRA.2022.3190086.

[41] Luis Martinez-Gomez and Thierry Fraichard. "Benchmarking collision avoidance schemes for dynamic environments". In: *ICRA Workshop on Safe Navigation in Open and Dynamic Environments*. 2009.

[42] Luis Martinez-Gomez and Thierry Fraichard. "Collision avoidance in dynamic environments: an ics-based solution and its comparative evaluation". In: *2009 IEEE International Conference on Robotics and Automation*. IEEE. 2009, pp. 100–105.

[43] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. *OpenAI Gym*. 2016. eprint: arXiv:1606.01540.

[44] Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. "Airsim: High-fidelity visual and physical simulation for autonomous vehicles". In: *Field and Service Robotics: Results of the 11th International Conference*. Springer. 2018, pp. 621–635.

[45] N. Koenig and A. Howard. "Design and use paradigms for Gazebo, an open-source multi-robot simulator". In: *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*. Vol. 3. 2004, 2149–2154 vol.3. DOI: 10.1109/IROS.2004.1389727.

[46] Meng Guo and Michael M. Zavlanos. “Multirobot Data Gathering Under Buffer Constraints and Intermittent Communication”. In: *IEEE Transactions on Robotics* 34.4 (Aug. 2018), pp. 1082–1097. ISSN: 1941-0468. DOI: 10.1109/TRO.2018.2830370.

[47] Andreas Bircher, Mina Kamel, Kostas Alexis, Helen Oleynikova, and Roland Siegwart. “Receding Horizon "Next-Best-View" Planner for 3D Exploration”. In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. Stockholm, Sweden: IEEE, May 2016, pp. 1462–1468. DOI: 10.1109/ICRA.2016.7487281.

[48] Tobias Nägeli, Javier Alonso-Mora, Alexander Domahidi, Daniela Rus, and Otmar Hilliges. “Real-Time Motion Planning for Aerial Videography With Dynamic Obstacle Avoidance and Viewpoint Optimization”. In: *IEEE Robotics and Automation Letters* 2.3 (July 2017), pp. 1696–1703. ISSN: 2377-3766. DOI: 10.1109/LRA.2017.2665693.

[49] Rudolph Emil Kalman. “A new approach to linear filtering and prediction problems”. In: (1960).

[50] Afzal Ahmad, Viktor Walter, Pavel Petrá̲vcek, Mat̲vej Petrlík, Tomá̲vs Bá̲vca, David v̲Zaitlík, and Martin Saska. “Autonomous aerial swarming in gnss-denied environments with high obstacle density”. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 570–576.

[51] Johann Borenstein and Yoram Koren. “Real-time obstacle avoidance for fast mobile robots in cluttered environments”. In: *Proceedings., IEEE International Conference on Robotics and Automation*. IEEE. 1990, pp. 572–577.

[52] Paolo Fiorini and Zvi Shiller. “Motion planning in dynamic environments using velocity obstacles”. In: *The international journal of robotics research* 17.7 (1998), pp. 760–772.

[53] Akash Patel, Björn Lindqvist, Christoforos Kanellakis, and George Nikolakopoulos. “Fast Planner for MAV Navigation in Unknown Environments Based on Adaptive Search of Safe Look-Ahead Poses”. In: *2022 30th Mediterranean Conference on Control and Automation (MED)*. June 2022. DOI: 10.1109/MED54222.2022.9837293.