Evaluating Dynamic Environment Difficulty for Obstacle Avoidance Benchmarking

Moji Shi, Gang Chen, Álvaro Serra Gómez, Siyuan Wu and Javier Alonso-Mora

Abstract— Dynamic obstacle avoidance is a popular research topic for autonomous systems, such as micro aerial vehicles and service robots. Accurately evaluating the performance of dynamic obstacle avoidance methods necessitates the establishment of a metric to quantify the environment's difficulty, a crucial aspect that remains unexplored. In this paper, we propose four metrics to measure the difficulty of dynamic environments. These metrics aim to comprehensively capture the influence of obstacles' number, size, velocity, and other factors on the difficulty. We compare the proposed metrics with existing static environment difficulty metrics and validate them through over 1.5 million trials in a customized simulator. This simulator excludes the effects of perception and control errors and supports different motion and gaze planners for obstacle avoidance. The results indicate that the survivability metric outperforms and establishes a monotonic relationship between the success rate, with a Spearman's Rank Correlation Coefficient (SRCC) of over 0.9. Specifically, for every planner, lower survivability leads to a higher success rate. This metric not only facilitates fair and comprehensive benchmarking but also provides insights for refining collision avoidance methods, thereby furthering the evolution of autonomous systems in dynamic environments.

I. INTRODUCTION

Dynamic obstacle avoidance is a popular research topic in the field of robotics [1]–[6]. To evaluate the obstacle avoidance methods, most works handcraft their custom test maps in simulation or the real world to demonstrate a higher success rate. However, the difficulty of the chosen maps is not stated or evaluated. This indication of difficulty is vital. One method with a very high success rate in a simple environment may drastically fail in a more difficult environment. It is ideal to test the methods under different difficulty levels. Furthermore, when comparing with other obstacle avoidance methods, pointing out the difficulty or creating environments with similar difficulties used in the baselines can promote a fair comparison.

In static environments, the difficulty is usually evaluated by the density of the environment [7], [8]. In dynamic environments, defining a difficulty metric is much more complicated because the difficulty is influenced by many factors [9], such as obstacle size, number, velocity, and motion profile. While Fig. 1 makes it clear that Map (b) is
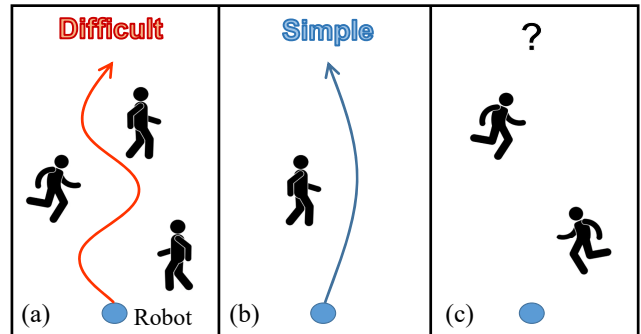
Fig. 1: Dynamic environments with different difficulties. (a) shows a map with two walking pedestrians and one running pedestrian; (b) is with one walking pedestrian; (3) is with two running pedestrians. While it is intuitively clear that (b) represents a simpler environment than (a), determining whether (c) is simpler or more difficult than (a) remains unclear. The magnitude of the difficulty is hard to determine.

less challenging than (a), discerning whether Map (c), with fewer but faster pedestrians, is more difficult than Map (a) remains a hard task without a quantitative metric. Currently, there is no existing metric to quantify the difficulty of a dynamic map.

In this paper, we design four quantitative metrics from different perspectives, such as survivability, global survivability, traversability, and velocity obstacle (VO) feasibility, to evaluate the difficulty of a dynamic environment. Recognizing the comprehensive impact of various environmental factors on difficulty, our metrics avoid constructing a formula with these factors but try to capture the influence caused by them, starting from one single foundational premise: a map with higher difficulty should consistently result in a lower obstacle avoidance success rate. To evaluate the effectiveness of these metrics, we developed an efficient custom simulator that excludes the influence of perception and control error and computational power, while retaining the influence of environmental factors. Numerous tests using different motion and gaze planners are conducted to identify the best metric that aligns with our foundational premise. We then delve into a discussion of the merits and drawbacks of the metrics and present guidelines for their utilization in both simulation and real-world scenarios.

The code of using the metric and the simulator is available at **Code:** https://github.com/smoggy-P/gym-Drone2D-ActivePerception **Homepage:** https://smoggy-

II. RELATED WORK

A. Collision avoidance in unknown dynamic environments

Robot collision avoidance in dynamic, unpredictable environments is a longstanding challenge in robotics, even under full observability conditions. A plethora of solutions have emerged to tackle this issue. Methods in [3]–[6] utilize sampling-based methods to create multiple trajectories, from which the one of lowest cost is selected. In contrast, optimization-based methods like [10]–[12] treat collision avoidance as an optimization problem, setting specific constraints to guide trajectory generation.

Gaze planning has proved to be important in dynamic obstacle avoidance. With real-time perception, dynamic obstacles require constant monitoring for future predictions. While researchers like [13], [14] discuss generating collision-free trajectories with 360 degrees of FOV, many real-world situations limit the range of FOV. This necessitates gaze planning to decide where the robot should look to gather more information for better collision avoidance. Simple strategies like those in [15]–[17] suggest looking in the velocity or target direction. More intricate methods, found in [1], [2], [6], [18], [19], treat gaze planning as an optimization problem, directing the gaze based on various objectives.

B. Benchmarking for collision avoidance

To further advance dynamic collision avoidance methods, it is essential to objectively compare them using benchmarks that evaluate performance and robustness across different conditions. The most common benchmarking approach for collision avoidance methods involves comparing success rates across manually designed test environments [1], [2], [16]. Some studies have proposed more standardized benchmarking suites for collision avoidance in static environments, such as [20]–[24]. These benchmarking suites provide randomly generated static maps and standardized evaluation pipeline to guarantee fair comparison. Furthermore, some metrics are defined to quantify the static environments difficulty like obstacle density [25], [26] and traversability [27]–[29]. These metrics provide more insights into the testing maps and thus allow a more comprehensive comparison between collision avoidance methods.

For benchmarking in dynamic collision avoidance, [9], [30]–[32] provide standard benchmarking suites where collision avoidance methods can be tested in randomly generated dynamic maps and be compared. [9] also mentions the non-quantitative difficulty of dynamic maps. However, they lack a comprehensive set of metrics to quantify the difficulty of the dynamic environments and the validation of these metrics.

III. SIMULATOR DESIGN

In this section, the designed simulator is introduced to validate the proposed difficulty metrics presented in Sec. IV. These metrics will be introduced in Sec. IV.
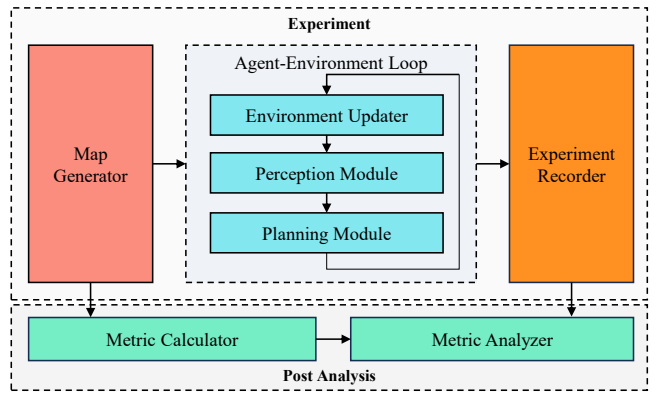

Fig. 2: The pipeline of the proposed custom simulator

A. Requirements and Assumptions

We validate the difficulty metrics by investigating their correlation with the performance of typical planners (i.e. How the success rate of planners changes when the difficulty metrics change). Thus, an ideal simulator should adhere to several criteria:

- Comprehensive and rapid assessment of different collision avoidance methods (including trajectory planners and gaze planners mentioned in Sec. V-A.3).
- Ensure the isolation of map difficulty as a variable affecting the performance of various methods, excluding the effect from factors such as perception error, control error, and computational power.

Meanwhile, several assumptions have been made to simplify the simulator: a) The environment is planar. b) Only disc-shaped dynamic obstacles are employed.

B. Simulator Pipeline

We design our simulator following the OpenAI gym standard, informed by insights from [33] due to its lightweight feature and its "agent-environment" loop design. Based on that, we realize a sequential navigation pipeline that allows us to exclude the perception noise, control noise, and computational time. As shown in Fig. 2, our simulator is composed of the following components:

a) Map Generator creates random maps with varying dynamic obstacles in terms of number, size, velocity, and motion profiles. Motion profiles determine the movement of dynamic obstacles, further detailed in the Sec. III-B. We craft two map datasets using this generator. *Dataset I* assumes uniform size and velocity for all obstacles in a map. The map is thus characterized by the number of obstacles n_{obs}, obstacles' size r_{obs}, and velocity v_{obs}. A random seed is used to initialize the obstacles' position and direction. In contrast, *Dataset II* is for more general scenarios with varying obstacle sizes, velocities, and motion profiles. One example of the generated map can be found in Fig. 3.

b) Environment Updater updates dynamic obstacles' position according to the motion profile defined by Map Generator. In *Dataset I*, the motion profile is defined as the constant velocity model (CVM). In *Dataset II*, it can be defined as
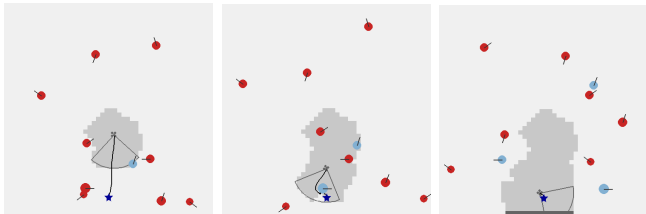
Fig. 3: Visualization of one experiment trial in the custom simulator. ⬤: Unobserved Obstacle; ⬤: Observed Obstacle; ☐: Unexplored Area; ▨: Unoccupied Area

	Metric	Expected Correlation with Difficulty	Description
1)	Obstacle Density [7]	+	Maps with large obstacle density are considered difficult.
2)	Traversability [21], [27]	−	Maps with large traversable space are considered simple.
3)	Dynamic Traversability	−	Maps with large dynamic traversable space are considered simple.
4)	VO Feasibility	−	Maps with more feasible velocities are considered simple.
5)	Survivability	−	Long survival time of sampled position means the map is simple.
6)	Global Survivability	−	Same as survivability.

TABLE I: Expected correlation between metrics, map difficulty, and success rate of planner. We reverse some metrics to make sure all metrics increase monotonously as difficulty increases. Reversed metrics include Traversability, Dynamic Traversability, VO Feasibility, Survivability, and Global Survivability. This preprocessing step will be further introduced in Sec. V-B.

the reciprocal velocity obstacle (RVO) [34] which makes obstacles to avoid each other.

c) Perception Module applies a ray-casting algorithm to simulate the FOV of the robot, producing a 2D occupancy map and dynamic trackers. Grids can be unexplored, unoccupied, or occupied. Observation of dynamic obstacles is updated using a Kalman Filter (KF) tracker, with visible grids and obstacles directly updated to their ground truth, eliminating perception noise. The perception output can also be seen in Fig. 3.

d) Planning Module processes perception outputs to generate the future trajectory and yaw angle velocity using the planners to be evaluated in Tab. III. If no trajectory is feasible, a replan is triggered for the next step and outputs a braking command. The output trajectory is directly executed, ensuring no control errors impact planner performance.

e) Experiment Recorder logs results, categorizing outcomes as Success, Collision, or Deadlock. Deadlock arises when the robot fails to find a feasible trajectory after 5 consecutive replan attempts.

After achieving the experiment results, we calculate metrics of maps and analyze the correlation with the success rates through Metric Calculator and Metric Analyzer. These two modules will be further explained in Sec. IV

IV. METRICS DESIGN

In this section, metrics are defined to quantify dynamic environment difficulty corresponding to Metric Calculator in Fig. 2 including Obstacle Density [7], Traversability [21], [27], Dynamic Traversability, VO Feasibility, Survivability, and Global Survivability. The expected correlation between these metrics and map difficulty is shown in Tab. I. Obstacle Density and Traversability are two existing difficulty metrics for static environments and are introduced as baselines. The other four metrics are designed by us to measure dynamic environment difficulty. Finally, we detail the evaluation method for the metrics.

A. Difficulty Metrics

1) Obstacle Density: Obstacle density is a widely used metric to quantify the difficulty of static environments [7], [8]. It is defined as the areas occupied by obstacles divided by the total area of the map:

$$\text{Obstacle Density} = \frac{A_{obs}}{A_{map}} \quad (1)$$

In dynamic maps, obstacle density is irrelevant with time and thus will not change at different time steps as we suppose that obstacles do not overlap with each other. Thus, we only calculate it once when initializing the dynamic environment.

2) Traversability: Traversability is proposed in [21], [27] to evaluate the difficulty of static environments. It is defined as the average traversable distance for all uniformly sampled positions and directions as shown in Fig. 4a:

$$\text{Traversability} = \frac{1}{N} \sum_{i=1}^{N} d_i \quad (2)$$

where d_i represents the traversable distance at i-th sampled position and direction, and N represents the total number of sampled positions and directions.

3) Dynamic Traversability: The traversability at different time steps might differ. We improve it for dynamic maps by designing dynamic traversability. The dynamic traversability is calculated by sampling the time step and averaging the traversability over the sampled time step.

$$\text{Dynamic Traversability} = \frac{1}{MN} \sum_{j=1}^{M} \sum_{i=1}^{N} d_i(t_j) \quad (3)$$

where the $d_i(t_j)$ is the traversable distance at i-th sampled position and direction and at the time step t_j. The time step is sampled uniformly from the beginning of the dynamic map: $t_j = (j-1) \cdot t_{\text{sample}}, \forall j \in \{1, 2 \cdots M\}$. N is the number of sampled positions and directions, and M is the number of sampled time steps.

4) VO Feasibility: The velocity obstacle (VO) is introduced in [35] for multi-agent collision avoidance. The VO for robot A regarding a collision with obstacle B is given

by:

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

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 union of all VOs determines infeasible velocities for the ego-robot. Intuitively, a larger VO area implies a more challenging environment. Hence, we propose a VO feasibility metric, where we first sample N positions around the map. At each position indexed by i, we sample n_{vel} velocities and calculate the percentage of sampled velocities outside any VO:

$$\text{VO Feasibility} = \frac{1}{N} \sum_{i=1}^{N} \frac{n_{feasible}(i)}{n_{feasible}(i) + n_{infeasible}(i)} \quad (5)$$

$n_{feasible}$ and $n_{infeasible}$ denote the number of sampled velocities that lie outside and inside the VO area as shown in Fig. 4b and $n_{vel} = n_{feasible}(i) + n_{infeasible}(i), \forall i \in \{1, 2 \cdots N\}$.

5) Survivability: The aforementioned metrics mainly assess discrete difficulty as they are calculated at certain time steps. For example, Obstacle Density, Traversability, and VO feasibility are only calculated at the initial step of the dynamic map, and dynamic traversability samples M discrete steps. These metrics neglect the continuous changes in the environment. To address this issue, the survivability metric is proposed. We assume that static robots are placed at sampled positions and calculate their average survival time. The survival time is defined as the duration from the initial time step until one obstacle moves into and collides with the static robot, as shown in Fig. 4c:

$$\text{Survivability} = \frac{1}{N} \sum_{i=1}^{N} \min(t_i, T_{max}) \quad (6)$$

where t_i is the surviving time of the robot at the i-th sample, N is the number of robot samples, and T_{max} is the upper bound of the survival time for normalization. The N positions are sampled from a uniform grid with d_{sample} as the distance between two grid points.

Note that in the real-world tests, there is no need to actually "place" static robots on the map. We can record the trajectories of all obstacles and calculate the Survivability by replaying these trajectories and recording survival time in these replays.

6) Global Survivability: Instead of putting one static robot on the map at each sample in the Survivability calculation, Global Survivability is calculated by assuming that N robots are simultaneously placed at different positions across the map. For each deployment, the duration from time step t_j until any robot collides with an obstacle is recorded. Global survivability quantifies this average survival time, considering deployments from multiple start time steps t_j:

$$\text{Global Survivability} = \frac{1}{K} \sum_{j=1}^{K} \min(t_j^1, t_j^2 \dots, t_j^N, T_{max}) \quad (7)$$

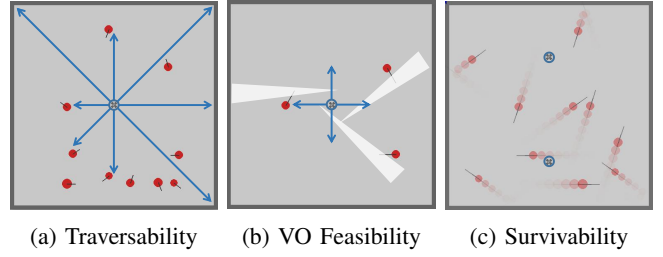(a) Traversability   (b) VO Feasibility   (c) Survivability

Fig. 4: Examples of metrics calculation. In (a), traversability is the average of traversable distances(shown by blue arrows) in 8 uniformly distributed directions at one sampled position. In (b), the white areas represent the infeasible VO regions. 4 velocity vectors(shown by blue arrows) are sampled at the sampled position. Only the velocity pointing right lies in the infeasible VO regions. So we have $n_{feasible} = 3$ and $n_{infeasible} = 1$. The resulting VO feasibility is thus $\frac{3}{4}$. In (c), a simple example of Survivability calculation is shown. The static robot is assumed to be placed in two positions. The static robot above does not intersect with any obstacle history trajectories in T_{max} while the static robot below survives 2 seconds until it collides with the obstacle heading toward the left. The survivability is thus $(T_{max}+2)/2 = (3+2)/2 = 2.5$

where t_j^n is the surviving time of the n-th robot starting from t_j (i.e. The duration from t_j until collision), and K is the number of samples of start time steps.

B. Evaluation Methodology

We want to evaluate all metrics in *Dataset I* mentioned in Sec. III-B. Assuming *Dataset I* contains n different maps, each denoted by M_i. The dataset can be denoted as $\mathcal{M} = \{M_1, M_2, ..., M_n\}$. m different planners are tested on these maps. The map difficulty is defined as a scalar function $D(M_i)$. The success rate of a planner j on map M_i is defined as SR_i^j. We introduce two quantitative indicators for evaluating whether the metrics are representative of the dynamic map difficulty:

Spearman's Rank Correlation Coefficient (SRCC): This metric evaluates the monotonic relationship between two variables. For a given metric $D(M_i)$, we evaluate its monotonical relation with the success rate SR_i^j for planner j and denote it as $SRCC_j$. The overall effectiveness of the difficulty metric is the average of $SRCC_j$.

Coefficient of Variation (CV): This metric suggests the variation of the success rate SR_i^j on maps with close difficulty metrics. The smaller *CV* value means that if two maps have very close metrics, the success rate of one planner on these two maps will not differ too much, indicating a good correlation between the metric and the success rate. Here, we round the metric to the nearest integers to construct map groups so that each group includes maps with similar metrics:

$$\mathcal{M}_k = \{M_i | D(M_i) \in [k, k+1]\} \quad k = 0, 1, 2, \dots, 9 \quad (8)$$

For each group $\mathcal{M}_k$, CV_j^k for planner j denotes performance stability:

$$CV_j^k = \frac{\sigma_j^k}{\mu_j^k} \qquad (9)$$

σ_j^k and μ_j^k are the standard deviation and the mean value of the success rate of planner j in map group $\mathcal{M}_k$. The *CV* measure for the difficulty metric is given by the average of CV_j^k.

V. EXPERIMENT

To evaluate the metrics introduced in Sec. IV, we replicate various trajectory and gaze planners, testing them within two map datasets outlined in Sec. III-B. After testing, we compute metrics for each map, aiming to explore the relationship between these metrics and planner success rates.

A. Experiment Setup

1) Map Dataset: All maps are generated in a $50m \times 50m$ square area. As mentioned in Sec. III, maps in *Dataset I* are generated by defining 3 variables and one random seed. The range of these variables is shown in Tab. II. For each variable setting, we generate 20 maps with different random seeds. We will thus have $3 \times 3 \times 3 \times 20 = 540$ maps in *Dataset I*.

Parameter	n_{obs}	r_{obs}	v_{obs}
Range	$\{10, 20, 30\}$	$\{0.5, 1, 1.5\}\, m$	$\{2, 4, 6\}\, m/s$

TABLE II: Range of variables for maps in *Dataset I*

If the metric performs well in *Dataset I*, it is further evaluated in *Dataset II*, where assumptions are relaxed and three more general map types are introduced: (a) obstacle velocities within a map differ, sampled from the distribution $[2, 6]$ m/s; (b) obstacle sizes within a map differ, sampled from $[0.5, 1.5]$ m; (c) obstacles move using the RVO [34] motion profile. For each type, we generate 40 maps. Therefore, we have 120 maps in *Dataset II*.

2) Robot Parameters: We assume that the robot is a 2D circle with a radius of 1m. The maximum acceleration of the robot is 4m/s^2. The depth of the FOV is 8m, and the range of the FOV can be 90 to simulate a single depth camera or 360 degrees to simulate multiple depth cameras and LiDAR. Since a larger yaw angle velocity will bring large errors in depth estimation, the maximum yaw angle velocity is set to 1.4 rad/s according to [6].

3) Planners: We use multiple planners for trajectory and gaze planner in Tab. III and Tab. IV to ensure that the observed relationship remains consistent rather than being specific to a particular planner.

Here the trajectory planners are chosen in terms of different planning horizons ([3] generates a global trajectory while [6], [10] generate local trajectory) and planning methods([3], [6] are sampling-based methods while [10] is optimization-based method).

Method	Description
Global Motion Primitive [3]	Samples trajectories, filters these trajectories with collision checking and conducts a graph search in these trajectories until reaching the goal.
Model Predictive Control (MPC) [10]	Defines the cost as the distance between future trajectory and target position plus control input, and defines the collision constraints as the distance between the robot and obstacles which are modeled as ellipsoids.
Local Primitive [6]	Samples local targets, generates an optimal local trajectory [4] towards the local target with the lowest cost, and iteratively updates the local target until reaching the global target.

TABLE III: Baselines of Trajectory Planner

Method	Description
FullRange	The perception range is expanded to 360 degrees. No gaze planner is needed in this case.
LookAhead [15]	Look at the current velocity direction.
LookGoal [17]	Look at the current target direction.
Rotating	Rotate in the largest rotating velocity
Finean et al. [1]	Optimize an objective function to find a trade-off between looking at the future trajectory and looking at grids that have not been updated for a long time.
Owl [6]	Optimize multiple objectives, including looking at the velocity direction, the target position, the direction that has not been updated for a long time, and the observed dynamic obstacles.

TABLE IV: Baselines of Gaze Planner

B. Experiment Results

For each planner on each map, we experiment with multiple trials to calculate a corresponding success rate. In these trials, we apply different start and target positions and different robot velocities. There are 9 different position candidates for the start and target positions, and the robot velocity is chosen from $\{2, 4, 6\}$ m/s. Therefore, we have $\binom{9}{2} \times 3 = 216$ trials for each planner on each map to calculate the success rate.

The metrics $D(M_i)$ of these maps undergo two pre-processing steps: normalization and reverse. Normalization scales the metric value so that $D(M_i) \in [0, 10]$. Reverse will let $D(M_i) = 10 - D(M_i)$ for some metrics to ensure that the high metric always indicates difficult maps as explained in Tab. I. Then we show the correlation figures between the metrics and the success rate in Fig. 5. The quantitative comparison of different metrics can also be seen in Tab. V. The survivability metric achieves the best SRCC and CV, and it can be demonstrated from the line plot in Fig. 5 (a) that the survivability shows a good monotonical relationship with the success rate of every planner.

VI. DISCUSSION

In this section, we further evaluate the difficulty metrics from Sec. IV using SRCC and CV evaluation method intro-

(a) Line plot of average success rate and environment difficulty metrics using different planners
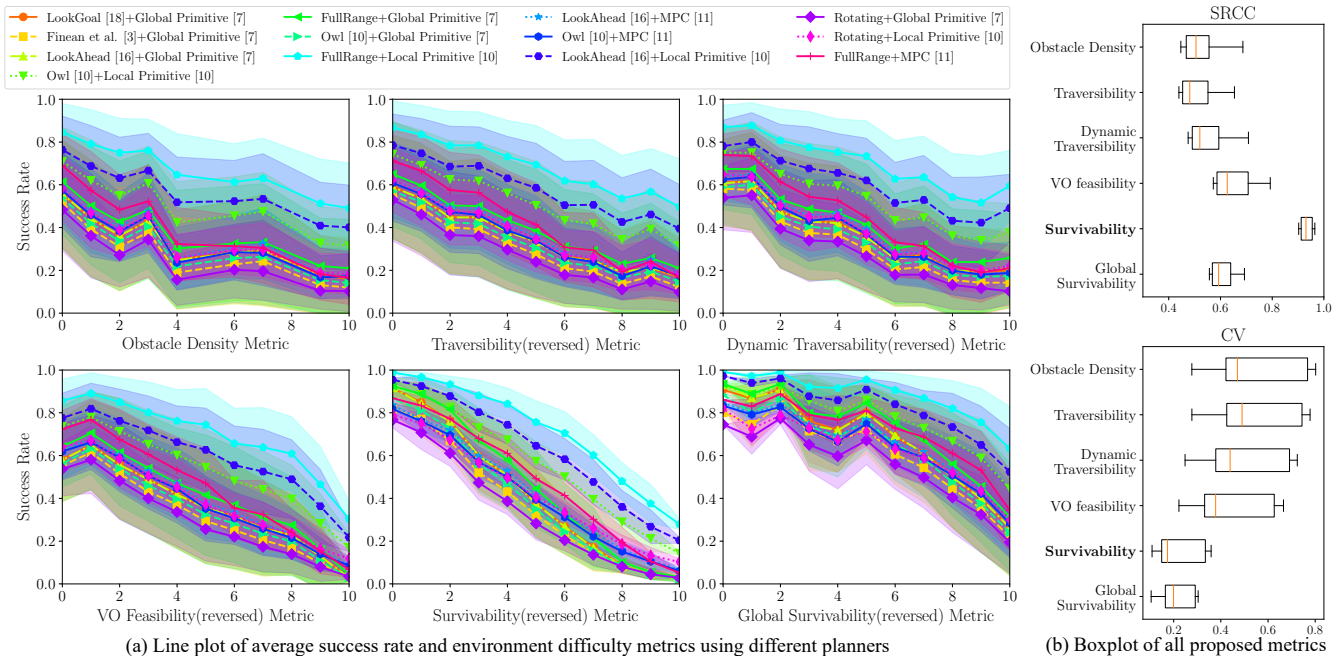
(b) Boxplot of all proposed metrics

Fig. 5: All experiment results of the proposed metrics. In (a), the plot shows the relationship between the success rate and the pre-processed metrics for different planners. Different colors denote planners. The light-colored band surrounding the curve represents one standard deviation of the success rate of each planner under each level of the difficulty metric. In (b), the boxplot shows SRCC and CV values of different metrics.

Difficulty Metric	SRCC	CV	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	$30s$
VO feasibility	0.651 ± 0.077	0.461 ± 0.161	$7s$
Survivability	$\mathbf{0.932 \pm 0.023}$	$\mathbf{0.230 \pm 0.095}$	$20s$
Global Survivability	0.607 ± 0.044	0.242 ± 0.078	$20s$

TABLE V: Evaluation of metrics in terms of their SRCC, CV, and calculation time of the metric for one map.

Metric	Obstacle Velocity	SRCC	CV
Obstacle Density	$2.0 - 6.0$m/s	0.501 ± 0.068	0.571 ± 0.193
	2.0m/s	0.788 ± 0.045	0.075 ± 0.026
	4.0m/s	0.784 ± 0.030	0.158 ± 0.050
	6.0m/s	0.758 ± 0.029	0.272 ± 0.109
VO Feasibility	$2.0 - 6.0$m/s	0.651 ± 0.077	0.461 ± 0.161
	2.0m/s	0.945 ± 0.017	0.006 ± 0.002
	4.0m/s	0.953 ± 0.005	0.124 ± 0.038
	6.0m/s	0.946 ± 0.007	0.214 ± 0.079

TABLE VI: SRCC and CV of metrics after grouping the maps according to the obstacle velocity. The reported improvements are in comparison to their original values.

duced in Sec. IV-B. Sec. VI-A analyzes the reason why each metric works or not, and Sec. VI-B presents the specific use case of the best metric Survivability.

A. Scope of Metrics for Difficulty Evaluation

1) Obstacle density: Obstacle density mainly evaluates static environment difficulty, explaining its weak correlation with success rate in dynamic settings where velocity is not considered. We further validate this by grouping maps based on obstacle velocities and investigating the correlation in each group. Tab. VI shows that the correlation in each group improves significantly, as reflected by higher SRCC and lower CV values.

2) & 3) Traversability and Dynamic Traversability: Similar to obstacle density, the poor performance of traversability and dynamic traversability can also be attributed to neglecting obstacle velocity.

4) VO Feasibility: Since different configurations of obstacle velocities can result in variations in VO feasibility regions, it is expected to be a good metric for dynamic

environments. However, it does not have a good correlation with the success rate, featured by low SRCC and high CV. As demonstrated in Tab. VI, the SRCC and CV values become significantly better if the obstacle velocities are the same. This indicates that the VO feasibility metric is similar to the metrics for static environments, which are not sensitive to the change in obstacle velocities. This limitation can be traced back to the generation process of VO infeasible areas. For two maps with obstacles at the same position but different velocities, the infeasible area only shifts directionally. The size of these infeasible regions remains unchanged. Thus, the VO feasibility metric cannot capture dynamic environment difficulty variations caused by different obstacle velocities.

5) Survivability: The Survivability metric consistently scores the highest in SRCC and lowest in CV, revealing a strong correlation between survivability and success rate,

meaning the metric is effective in assessing dynamic map difficulty in *Dataset I*. We also test survivability with higher sampling density. By reducing the distance between sample positions, d_{sample} from 12.5 m to 10 m, the number of samples increases from 9 to 16 in each map. However, the SRCC only demonstrates a small improvement from 0.932 to 0.941 at the cost of 2 times the calculation time.

We further investigate the generalization ability of survivability by fitting a Gaussian distribution to the success rate of *Dataset I* under the same survivability metric and then calculating the Mahalanobis Distance of the data points in *Dataset II* to the fitted Gaussian distribution. The average Mahalanobis Distance is 0.74, and 99.3% of the data points in *Dataset II* lie within the 3σ range of the fitted distribution. This suggests the survivability metric effectively evaluates the dynamic map difficulty when the sizes and velocities of obstacles differ in the map and when the motion profile is RVO. For example, if there are two maps with CVM and RVO motion profiles that have similar survivability, the success rate of one planner in these two maps is expected to be similar. The detailed analysis, including maps with different motion profiles, will be presented in the supplementary file.

6) Global Survivability: It does not correlate better with success rates than the original Survivability metric. Fig. 5 shows that the variance of success rates for maps with low difficulty is small, while the variance of success rates for maps with higher difficulty is large. Since we calculate the minimum survival time of all sampled positions, most maps have similarly small Global Survivability and are thus considered difficult. It is hard to tell the difference between them by Global Survivability.

B. Use Cases of Survivability Metric

There are three use cases of the survivability metric:

1) Comparison of Different Planners: Fig. 6 shows success rates for three trajectory planners with varied survivability levels. The *Local Primitive* planner consistently outperforms others. While *Global Primitive* starts stronger than *MPC*, it is surpassed by *MPC* as the difficulty increases. The conclusions here are based on the assumptions mentioned when we design the simulator in Sec. III-A.

Fig. 6 also compares five gaze planners and *FullRange* Perception against survivability levels. Here, *FullRange* sets the upper bound. *LookAhead* and *LookGoal* are top performers, with *Rotating* lagging behind.

2) Generate Maps with Predefined Survivability Metric: For benchmarking collision avoidance methods, it is also important to generate maps with pre-defined difficulty so that we can gradually test our method in harder scenarios. In *Dataset I*, the survivability of a map is determined by: n_{obs}, r_{obs}, v_{obs}. We use a linear regression model to fit this relationship:

$$S = f(n_{obs}, r_{obs}, v_{obs}) = \beta_0 + \beta_1 n_{obs} + \beta_2 r_{obs} + \beta_3 v_{obs} \quad (10)$$

The resulting coefficients are -6.014, 0.226, 2.646, and 1.104 for β_0, β_1, β_2, and β_3, respectively. Using the fitted model f, we can generate maps with a specified survivability
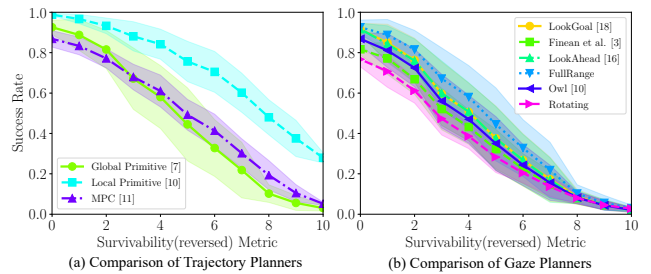

(a) Comparison of Trajectory Planners   (b) Comparison of Gaze Planners

Fig. 6: The comparison of planners using the survivability metric.

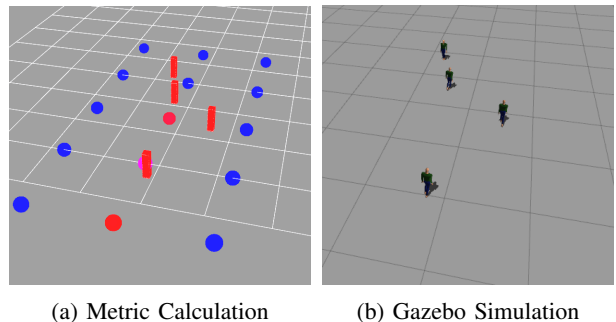(a) Metric Calculation   (b) Gazebo Simulation

Fig. 7: The calculation of the survivability metric in a gazebo simulation environment. In (a), the pillars represent the moving pedestrians. The spheres represent the sampled robot positions, where red ones have already collided with obstacles and blue ones have not.

metric *S*. The process is formulated as an Integer Linear Programming (ILP) optimization problem:

$$\text{minimize} \quad |f(n_{obs}, r_{obs}, v_{obs}) - S| \quad (11)$$
$$\text{s.t.} \quad n_{obs} \in \mathbb{Z}, 10 \leq n_{obs} \leq 30, \quad (12)$$
$$0.5 \leq r_{obs} \leq 1.5, 2 \leq v_{obs} \leq 6 \quad (13)$$

3) Calculating Survivability Metric in Other Simulator and real world: We can easily calculate survivability in other high-fidelity simulators. An example using the gazebo simulation is shown in Fig. 7. For the real-world test, survivability can also be calculated by recording the obstacles as mentioned in Sec. IV-A.5.

VII. CONCLUSIONS

In this paper, we propose four metrics to evaluate the environmental difficulty of dynamic environments for collision avoidance problems. We validate their effectiveness through extensive 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. Results show that the proposed survivability metric is suitable for assessing the dynamic environment difficulty. VO Feasibility metrics can also be used for rapid evaluations in consistent obstacle velocities. Our future work will explore the applicability of these metrics in 3D scenarios to further expand their scope of usage.

REFERENCES

[1] M. N. Finean, W. Merkt, and I. Havoutis, "Where should i look? optimized gaze control for whole-body collision avoidance in dynamic environments," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1095–1102, 2021.

[2] J. Tordesillas and J. P. How, "Panther: Perception-aware trajectory planner in dynamic environments," *IEEE Access*, vol. 10, pp. 22 662–22 677, 2022.

[3] S. Liu, N. Atanasov, K. Mohta, and V. 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)*, pp. 2872–2879.

[4] M. W. Mueller, M. Hehn, and R. D'Andrea, "A computationally efficient algorithm for state-to-state quadrocopter trajectory generation and feasibility verification," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 3480–3486.

[5] B. T. Lopez and J. P. How, "Aggressive 3-D collision avoidance for high-speed navigation," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*.

[6] G. Chen, W. Dong, X. Sheng, X. Zhu, and H. Ding, "An active sense and avoid system for flying robots in dynamic environments," *IEEE/ASME Transactions on Mechatronics*, vol. 26, no. 2, pp. 668–678, 2021.

[7] A. Ahmad, V. Walter, P. Petráček, M. Petrlík, T. Báča, D. Žaitlík, and M. Saska, "Autonomous aerial swarming in gnss-denied environments with high obstacle density," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 570–576.

[8] J. Borenstein and Y. Koren, "Real-time obstacle avoidance for fast mobile robots in cluttered environments," in *Proceedings., IEEE International Conference on Robotics and Automation*, 1990, pp. 572–577.

[9] A. Nair, F. Jiang, K. Hou, Z. Xu, S. Li, X. Xiao, and P. Stone, "DynaBARN: Benchmarking Metric Ground Navigation in Dynamic Environments," in *2022 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*.

[10] H. Zhu and J. Alonso-Mora, "Chance-constrained collision avoidance for mavs in dynamic environments," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 776–783, 2019.

[11] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *2011 IEEE International Conference on Robotics and Automation*, pp. 2520–2525.

[12] L. Blackmore, M. Ono, and B. C. Williams, "Chance-constrained optimal path planning with obstacles," *IEEE Transactions on Robotics*, vol. 27, no. 6, pp. 1080–1094, 2011.

[13] J. Chen, T. Liu, and S. Shen, "Online generation of collision-free trajectories for quadrotor flight in unknown cluttered environments," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1476–1483.

[14] S. Liu, M. Watterson, K. Mohta, K. Sun, S. Bhattacharya, C. J. Taylor, and V. Kumar, "Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-d complex environments," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1688–1695, 2017.

[15] C. Lim, B. Li, E. M. Ng, X. Liu, and K. H. 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)*, pp. 996–1004.

[16] G. Chen, S. Wu, M. Shi, W. Dong, H. Zhu, and J. Alonso-Mora, "Rast: Risk-aware spatio-temporal safety corridors for mav navigation in dynamic uncertain environments," *IEEE Robotics and Automation Letters*, vol. 8, no. 2, pp. 808–815, 2022.

[17] J. Tordesillas, B. T. Lopez, M. Everett, and J. P. How, "Faster: Fast and safe trajectory planner for navigation in unknown environments," *IEEE Transactions on Robotics*, vol. 38, no. 2, pp. 922–938, 2021.

[18] B. Zhou, J. Pan, F. Gao, and S. Shen, "Raptor: Robust and perception-aware trajectory replanning for quadrotor fast flight," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1992–2009, 2021.

[19] J. Tordesillas and J. P. How, "Deep-panther: Learning-based perception-aware trajectory planner in dynamic environments," *IEEE Robotics and Automation Letters*, vol. 8, no. 3, pp. 1399–1406, 2023.

[20] E. Heiden, L. Palmieri, L. Bruns, K. O. Arras, G. S. Sukhatme, and S. Koenig, "Bench-mr: A motion planning benchmark for wheeled mobile robots," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4536–4543, 2021.

[21] H. Yu, G. C. E. de Croon, and C. De Wagter, "Avoidbench: A high-fidelity vision-based obstacle avoidance benchmarking suite for multi-rotors," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 9183–9189.

[22] N. R. Sturtevant, "Benchmarks for grid-based pathfinding," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 4, no. 2, pp. 144–148, 2012.

[23] V. Behzadan and A. Munir, "Adversarial reinforcement learning framework for benchmarking collision avoidance mechanisms in autonomous vehicles," *IEEE Intelligent Transportation Systems Magazine*, vol. 13, no. 2, pp. 236–241, 2019.

[24] S. Singh, M. Kapadia, P. Faloutsos, and G. Reinman, "Steerbench: a benchmark suite for evaluating steering behaviors," *Computer Animation and Virtual Worlds*, vol. 20, no. 5-6, pp. 533–548, 2009.

[25] M. I. Ribeiro, "Obstacle avoidance," *Instituto de Sistemas e Robótica, Instituto Superio Técnico*, vol. 1, 2005.

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

[27] C. Nous, R. Meertens, C. De Wagter, and G. de Croon, "Performance evaluation in obstacle avoidance," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3614–3619.

[28] D. Perille, A. Truong, X. Xiao, and P. Stone, "Benchmarking metric ground navigation," in *2020 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. IEEE, 2020, pp. 116–121.

[29] H.-C. Wang, S.-C. Huang, P.-J. Huang, K.-L. Wang, Y.-C. Teng, Y.-T. Ko, D. Jeon, and I.-C. Wu, "Curriculum reinforcement learning from avoiding collisions to navigating among movable obstacles in diverse environments," *IEEE Robotics and Automation Letters*, vol. 8, no. 5, pp. 2740–2747, 2023.

[30] L. Kästner, T. Bhuiyan, T. A. Le, E. Treis, J. Cox, B. Meinardus, J. Kmiecik, R. Carstens, D. Pichel, B. Fatloun *et al.*, "Arena-bench: A benchmarking suite for obstacle avoidance approaches in highly dynamic environments," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 9477–9484, 2022.

[31] L. Martinez-Gomez and T. Fraichard, "Benchmarking collision avoidance schemes for dynamic environments," in *ICRA Workshop on Safe Navigation in Open and Dynamic Environments*, 2009.

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

[33] M. Körber, J. Lange, S. Rediske, S. Steinmann, and R. Glück, "Comparing popular simulation environments in the scope of robotics and reinforcement learning," *arXiv preprint arXiv:2103.04616*, 2021.

[34] J. van den Berg, Ming Lin, and D. Manocha, "Reciprocal Velocity Obstacles for real-time multi-agent navigation," in *2008 IEEE International Conference on Robotics and Automation*, pp. 1928–1935.

[35] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *The international journal of robotics research*, vol. 17, no. 7, pp. 760–772, 1998.