2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

DiPPeST: Diffusion-based Path Planner for Synthesizing Trajectories Applied on Quadruped Robots

Maria Stamatopoulou*, Jianwei Liu*, and Dimitrios Kanoulas

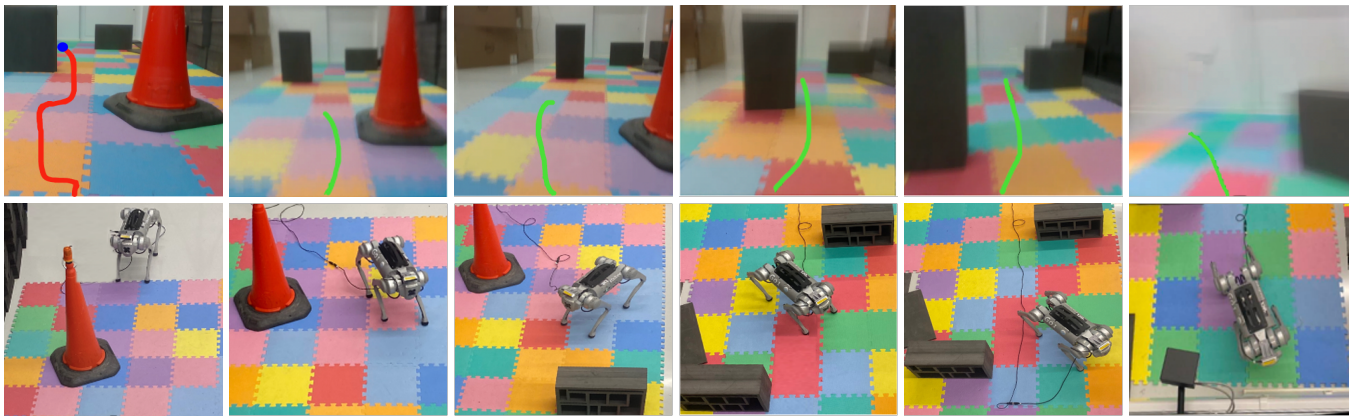


Fig. 1: DiPPeST: (top) Consecutive robot camera frames with the overlaid generated global (red) and local (green) trajectories, captured in real-time; (bottom) A quadruped robot following the planned path. Blurriness is due to robot fast motion.

***Abstract*— We present DiPPeST, a novel image and goal conditioned diffusion-based trajectory generator for quadrupedal robot path planning. DiPPeST is a zero-shot adaptation of our previously introduced diffusion-based 2D global trajectory generator (DiPPeR). The introduced system incorporates a novel strategy for local real-time path refinements, that is reactive to camera input, without requiring any further training, image processing, or environment interpretation techniques. DiPPeST achieves 92% success rate in obstacle avoidance for nominal environments and an average of 88% success rate when tested in environments that are up to 3.5 times more complex in pixel variation than DiPPeR. A visual-servoing framework is developed to allow for real-world execution, tested on the quadruped robot, achieving 80% success rate in different environments and showcasing improved behavior than complex state-of-the-art local planners, in narrow environments. Website: https://rpl-cs-ucl.github.io/DiPPeSTweb/**

I. INTRODUCTION

Mobile robots have been gaining increased popularity due to their multi-purpose applications, varying from industrial production sites to search-and-rescue tasks [1]. To successfully complete such tasks, real-time autonomous navigation in complex environments is required, rendering path planning an important open challenge [2]. Many works focus on generating global paths to provide a comprehensive path from a start to a goal point, assuming fully known and static environments [3], [4] however, these methods cannot adapt in dynamic or unknown ones. Studies on local planners aim to address this limitation by utilizing real-time sensory data, enabling robots to make immediate decisions and adapt their paths in response to dynamic changes in their surroundings. However, these approaches may sacrifice the optimality of the paths due to their myopic decision-making process. Hence, the ideal solution lies on the approach where the global planner sets an initial course path, and the local planner refines this path in real-time, ensuring both efficiency and adaptability. Learning from demonstration methods using image-conditioned Diffusion has demonstrated promising results in trajectory planning, primarily in the domain of manipulators [5], [6], [7], with few works extending these principles to mobile robot path planning [8], [3]. Diffusion policies iteratively calculate the action-score gradient based on visual observations, allowing for multi-modal action distribution representation, scalability to larger output spaces, and training stability while preserving expressive distribution capabilities [5]. However, training a diffusion model is computationally expensive and timely, requiring large datasets, which makes their adoption more challenging than other methods.

In this paper, we present a novel approach to path planning by extending our global Diffusion-based 2D path planner, DiPPeR [3], through a zero-shot adaptation that integrates an adaptive local path planner with no further training or fine-tuning. We introduce a robot path planning framework utilizing RGB data, complemented by a visual servoing pipeline for converting planned paths into actionable robot movements. We test our method on a real-world quadruped robot and compare the performance with state-of-the-art

The authors are with the Department of Computer Science, University College London, Gower Street, WC1E 6BT, London, UK. Dimitrios Kanoulas is also with Archimedes/Athena RC, Greece. {maria.stamatopoulou.21,jianwei.liu.21, d.kanoulas}@ucl.ac.uk

*equal contribution

This work was supported by the UKRI Future Leaders Fellowship [MR/V025333/1] (RoboHike) and the CDT for Foundational Artificial Intelligence [EP/S021566/1].

local planners both in static and dynamic environments. Our primary contributions include:

1) A novel image-conditioned diffusion path planner for mobile robots for real-time path refinements.
2) A zero-shot adaptation of DiPPeR to incorporate local path planning without re-training, fine-tuning, image pre-processing, or environment contextual or geometric information (e.g. semantics, obstacle recognition).
3) A visual servoing real-world deployment stack.

The remainder of the paper is structured as follows. In Sec. II, we briefly introduce literature in local path planning relevant to our proposed method. In Sec. III, we provide the necessary background knowledge, including a brief description of our previously introduced system, DiPPeR. In Sec. IV, we describe our proposed method, with our experimental results presented in Sec. V. Finally, in Sec. VI, we summarize the results and conclude with some future work.

II. RELATED WORK

Local path planning and visual servoing are extensively researched areas, as they enable mobile robots to make real-time decisions and adapt to dynamic environments efficiently. We briefly review these methods from both traditional and diffusion-based perspectives.

A. Local Path Planning and Visual Servoing

Traditional local path planning methods typically represent the environment using geometric primitives to identify traversal areas and obstacles [9], [10], [11]. However, these approaches often lack context awareness, characterized as the algorithm's inability to understand the environment beyond physical obstacles and free spaces and assume a static environment. This leads to limited environment interpretation and exponentially increasing computational demands for more complex environments [12]. One commonly used variation of geometric planners is the Dynamic Window Approach (DWA), where the robot's velocities are sampled to generate a trajectory optimized based on obstacle avoidance, reaching the target, and maintaining base velocity [13], [14], [15]. Chang et al. [16] improved on the original DWA evaluation functions by employing Q-learning to adaptively learn the parameters, enhancing global navigation performance and efficiency. However, the method relies on the balance between effectiveness, speed, training quality and the complexity of the environment, which might limit its effectiveness in unpredictable or complex scenarios. More recent works attribute contextual information to gain a better understanding of the environment [17]. Roth et al. [12] introduced a learned local path planning approach, utilizing both geometric and semantic information to identify terrain traversability and obstacles effectively. Crespo et al. [18] proposed a path-planning approach employing co-safe Linear Temporal Logic over perception-based atomic predicates, enabling the generation of uncertain semantic maps for navigation. While semantic-driven sampling-based algorithms offer promising results, they heavily rely on the accuracy of the semantic map and the quality of sensor data, and they suffer from increased computational complexity under severe perceptual uncertainty. Our method aims to address the limitations of both semantic and geometric methods by not relying on any environment representation other than visual input and avoiding scaling computational complexity, as demonstrated in our previous work [3].

Visual servoing provides a means of adaptively navigating based on visual cues [19]. Traditional visual servoing techniques have been integrated with visual cognition towards path planning. For instance, closer to our method, Rodziewicz-Bielewicz et al. [20] utilize YOLO and real-time image processing from a central camera to determine the robot positions and orientations for precise trajectory planning and execution in indoor environments. However, the method is constrained by specific environmental conditions and the generalization capabilities of YOLO in complex environments. Zhu et al. [21] deploy a visibility-based search tree with a greedy search strategy that segments results based on field-of-view (FOV) constraints, ensuring the feature points remain within the camera's view. The approach assumes consistent visibility and the need to keep feature points within the FOV at all times, which may not be feasible in highly dynamic environments or in situations with occlusions, potentially leading to servo failures. Our methodology relaxes these dependencies, facilitating global and local path planning interaction. This ensures alignment with the overarching objective while simultaneously allowing for local refinements.

B. Diffusion for Path Planning

Diffusion methods have gained popularity in the domain of path planning, with several works demonstrating promising results. Hong et al. [22] deploy diffusion maps to find local paths for reaching a goal while avoiding collisions with dynamic obstacles simultaneously, by computing transition probabilities between grid points. Liu et al. [3] introduce an image and goal-conditioned diffusion-based global path planner, trained using a CNN, showcasing great generalization and constant inference speed for various map types. Sridhar et al. [8] present a unified robotic navigation diffusion policy that handles task-oriented navigation and task-agnostic exploration in unseen environments, exhibiting significant performance improvements and computational efficiency compared to existing approaches. We aim to leverage the advancement made in this diffusion-based path planning realm to develop our trajectory generator. While all these works require training of diffusion models, which can be computationally expensive and require large training datasets, our proposed approach does not require further training or fine-tuning of a diffusion policy.

III. PRELIMINARIES

Path planning involves computing a trajectory for a robot to follow, by optimizing for properties such as finding a collision-free, shortest, and smoothest route between start and goal positions. As elaborated in Sec. II, there are several

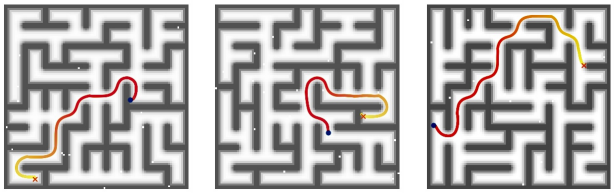


Fig. 2: Examples of DiPPeR's training dataset: 100×100 random solvable maps with examples of end-to-end trajectories, generated through A^*.

approaches to solving path planning. For relevance to our method, we expand on our previous work in probabilistic diffusion-based path planning (DiPPeR [3]), which serves as a baseline for the developed DiPPeST system.

DiPPeR is an image-based diffusion model designed to plan global paths from a starting to a goal position, given a 2D map in which space is classified as free or occupied. Trained on images of random mazes and ground truth trajectories generated by A^* path planning, DiPPeR is able to generate paths through a Denoising Diffusion Probabilistic Model (DDPM) [23] . A set of 2D path trajectories A_t^k with added Gaussian noise and the map image observation O are given as input, K denoising iterations are performed via gradient descent, and a noise-free path trajectory A_t^0 is produced:

$$A_t^{k-1} = \alpha(A_t^k - \gamma\epsilon_\theta(O, A_t^k, k) + \mathcal{N}(0, \sigma^2 I)), \quad (1)$$

where ϵ_θ is the noise prediction network (1D temporal CNN), and α, γ, σ define the scheduling learning rate.

During training, Gaussian noise ϵ^k is iteratively added to the clean trajectory A_t^0. The noise predictor ϵ_θ is trained using gradient decent of the mean squared error $\mathcal{L} = MSE(\epsilon^k, \epsilon_\theta(O, A_t^0 + \epsilon^k, k))$. DiPPeR achieves an average speedup of $23\times$ compared to state-of-the-art global path planning methods [3]. However, the method faces certain constraints. Initially, the path step number required for the denoising process must be empirically determined and manually selected for each experiment. This process is sub-optimal as it requires trial and error to determine the optimal length. Additionally, due to the inherent global planning nature of the approach, the goal position is manually chosen and remains unaltered throughout the experiment. This fixed setting proves suboptimal in scenarios requiring the dynamism of a local planner or in cases necessitating refinements.

IV. METHOD

In this work, we utilize DiPPeR for global path and end-to-end trajectory generation while addressing its existing limitations. We further modify the framework to introduce DiPPeST, a diffusion-based local path planner for adaptive real-time local refinements leveraging visual input. Additionally, we incorporate a customized visual servoing strategy to ensure efficient execution of real-world paths. DiPPeST is a zero-shot adaptation of DiPPeR, requiring no further training or fine-turning while introducing a novel local planning framework.

A. Global Planner

Firstly, our method involves the generation of a global path plan based on RGB input camera data. This path is generated via DiPPeR, utilizing the first captured frame from the robot's onboard camera which is positioned at it's front, facing towards the goal. The model generates a feasible path that avoids all obstacles while aiming to reach the set goal, by maintaining local uniformity in trajectory generation and traversing through pixels with consistent intensity levels. Unlike DiPPeR's original method, a map of the environment is not required, and planning happens directly through the RGB frame of the camera. Despite significant variations in the input images, DiPPeR demonstrates zero-shot adaptation to the robot's perspective and RGB image inputs despite being initially only trained on top-down grayscale maze images (Fig. 2). This transferability likely stems from the models' property to prioritize the generated trajectory's consistency and goal-reaching condition and its ability to learn color gradient-based features through ResNet, which are transferable from grayscale to RBG. Further exploration of the model's generalization capabilities is provided in Sec. V-A (also demonstrated in Fig. 6).

B. Local Planner

Given the generated global path, frame-to-frame local path adjustments are required to better navigate around obstacles and adapt to dynamic environments as the robot progresses toward its goal. To address this, we introduce DiPPeST, which takes an RGB image as input and outputs a 2D local goal position along with a 2D feasible collision-free trajectory to be followed, all within the camera frame. The method involves tracking the global path waypoints and the global goal position within individual local camera frames. At each frame, a strategy is devised to select the optimal waypoint, and a region of interest (ROI) is created from the robot's current position to the selected waypoint position. Processing within this ROI determines the optimal local goal position for the robot to reach in the current frame and DiPPeR generates the feasible path to the goal. An overview of DiPPeST is illustrated in Fig. 3.

1) Waypoint Tracking: We utilize the Lucas-Kanade optical flow estimation with the pyramidal approach for tracking the global path features across frames [24]. The method assumes that optical flow remains constant within a local neighborhood of pixels. Let I_x and I_y denote the gradients of the image in the x and y directions, respectively, and let I_t represent the temporal gradient between consecutive frames. The linear optical flow vector (u, v) is given by:

$$I_x \cdot u + I_y \cdot v = -I_t \quad (2)$$

This can be estimated by solving the system of linear equations for each pixel via the pyramidal approach, i.e. building an image pyramid where each level is a down-sampled version of the previous one. The optical flow is estimated at each level iteratively. The waypoints are then tracked by individual frames as seen in Fig. 3-C.

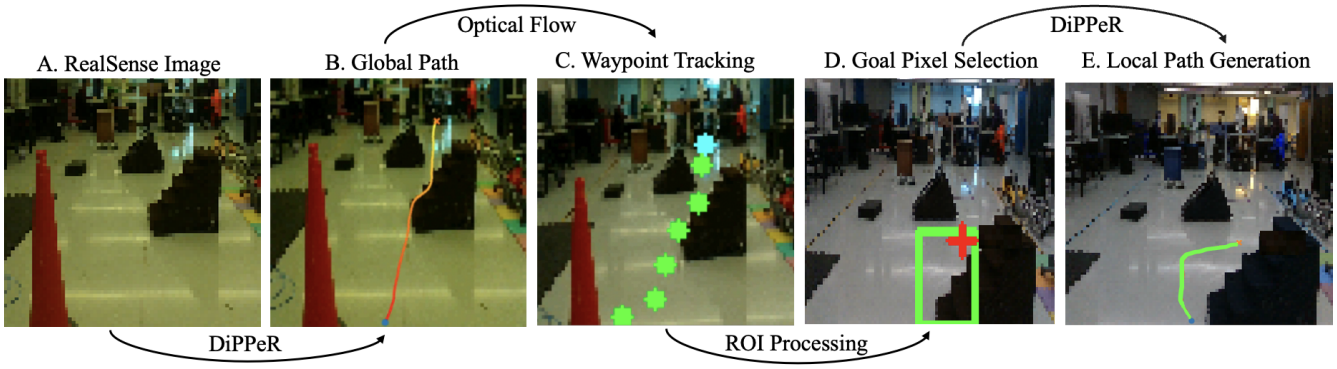


Fig. 3: DiPPeST process of generating the local path and correcting global path failure. DiPPeST takes an input camera frame (A) and utilizes DiPPeR to generate a global path (B) and the waypoints. These are tracked at each input frame (C), and the optimal waypoint is selected. At each frame a, ROI is created around the current position and the waypoint and the optimal local goal position (D) within the ROI is selected (red cross) to avoid assigning the goal close to the obstacle due to (C) waypoint suboptimal positioning. DiPPeR is then used to generate a path to the goal position (E). Low resolution images are displayed representing the exact output of the diffusion model.

2) Intermediate Waypoint Selection: At each frame, the optimal waypoint to follow is determined based on two conditions: (a) the half-point distance $dist$ from the current position to the furthest away waypoint and (b) the highest similarity sim to the direction vector **dir** of the goal position. This ensures that for each frame, the planned trajectory will aim toward the goal position while allowing for a significant look-ahead distance for refinements. For condition (a) the Euclidean distance $dist$ is calculated between the current position $\mathbf{p}_{curr}$ and all n waypoint positions $\mathbf{p}_{way}$ as:

$$dist(n) = \sqrt{\sum_{i=1}^{n} (\mathbf{p}_{curr}[i] - \mathbf{p}_{way}[i])^2}. \tag{3}$$

The median distance is then selected as the optimal waypoint position. Choosing the median distance waypoint helps avoid myopic local decisions that may be sub-optimal on a global scale. For condition (b), the dot product between the current position $\mathbf{p}_{curr}$ and all n waypoint positions $\mathbf{p}_{way}$ is calculated as follows:

$$\mathbf{dir} = \mathbf{p}_{curr} - \mathbf{p}_{way} \tag{4}$$

$$\mathbf{p}_{norm} = \frac{\mathbf{dir}}{\|\mathbf{dir}\|_2}, \quad \mathbf{g}_{norm} = \frac{\mathbf{g}}{\|\mathbf{g}\|_2} \tag{5}$$

$$sim = \mathbf{p}_{norm} \cdot \mathbf{g}_{norm}^T \tag{6}$$

where $\mathbf{p}_{norm}$ is the norm direction, $\mathbf{g}$ represents the goal position, $\mathbf{g}_{norm}$ is the norm goal, and $\cdot$ the dot product. Finally, the next waypoint i ($\mathbf{p}_{way}[i]$) with the median distance $dist$ and highest similarity sim is chosen as the next intermediate goal.

3) ROI Processing and Goal Selection: To ensure that the local goal is set optimally within a traversable region and to account for the possibility of a waypoint being incorrectly positioned within an obstacle, we further condition its selection. A ROI is defined in the camera frame, spanning between the current robot position and the selected intermediate waypoint, with the ROI size being variable based on the distance between the points. This ROI segment is utilized to assess the variation in pixel intensities. Let px_{std} be the standard deviation of pixel values in each color channel:

$$px_{std} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (ROI_i - \overline{ROI})^2} \tag{7}$$

where N denotes the number of pixels, ROI_i is the pixel value at position i and $\overline{ROI}$ is the mean pixel value in each color channel. The normalized standard deviation (px_{var}) is then obtained by dividing px_{std} by the maximum possible pixel value, typically 255 for 8-bit images. This serves as a quantitative measure of intensity variation within the ROI. The value of px_{var} is used in Sec. IV-B.4 to determine the number of path steps during the denoising process.

To determine the local optimal goal position, the most common pixel intensity px_{int} within the ROI is obtained by calculating the mode of the mean pixel intensity $\overline{px}$ within the ROI, as described in Eq. 8. To maintain consistency across frames, the previously selected pixel intensity px_{pint}^{-1} serves as a prior and is used as a tolerance value τ within which the new px_{int} should be set, to maintain local consistency for the diffusion model.

$$px_{\text{int}} = \text{mode}\left(\{px \mid px \in \text{ROI}, |\overline{px} - px_{\text{int}}^{-1}| \leq \tau\}\right) \tag{8}$$

Also, in the case that the ROI is predominantly occupied by obstacles, the use of px_{pint}^{-1} ensures that the local goal is still set in a traverisble pixel, as the new goal pixel intensity is limited to a certain tolerance. The optimal pixel px^* needs to have intensity px_{int} while also being the closest to the waypoint position, ensuring that the selected goal does not deviate significantly from the global path. The px^* is chosen as the next local goal, and its coordinates are scaled back to the entire image frame. An example of this correction is visualized by the red cross in Fig. 3-D.

4) Local Path Generation: After the new goal position is selected, DiPPeR is utilized to generate the feasible path, involving path step selection and path correction. Firstly, the path step number ps is selected through a combination of the Euclidean distance between the current position to the local goal position, px_d and the px_{var} within the ROI following:

$$ps = px_d + \alpha e^{(10 \times px_{var})} \tag{9}$$

The use of this method addresses DiPPeR's limitation of manually selecting the denoising vector length, which is now set equal to ps, and allows the length to increase exponentially as the variation of pixel intensity increases, thereby providing adaptation space scalable to obstacle density. The step increase is not significant large to affect the planning time. This step is essential for DiPPeST as local planning necessitates higher detail to avoid obstacles and can significantly differ from the global path due to variations in the input frames received, as opposed to global planning, where less detail is sufficient since the local planner can later address finer adjustments. We set α, the scaling coefficient, equal to 4, to agree with the input dimensions of the network. In cases where the local trajectory fails to avoid obstacles, a mitigating method is implemented to allow for re-planning, ensuring that the robot follows only feasible trajectories. The generated local trajectory is evaluated within the ROI to ensure that it does not traverse pixels whose intensity falls outside of the tolerance range compared to the local goal position pixel intensity.

C. Path Following

To execute the local paths generated by DiPPeST in the real world, the trajectory coordinates need to be de-projected from 2D image coordinates (x, y) obtained by the camera mounted on the robot's head into 3D world coordinates. Given the depth d from the depth reading for pixel coordinate (i, j) of the camera:

$$\text{Deproj}(i, j, d) = \left(d \cdot U_{\text{Model}} \left(\frac{(i, j) - P}{F} \right), d \right) \tag{10}$$

where $P = (p_x, p_y)$ is the principal point $F = (f_x, f_y)$ the focal length, both intrinsic parameters of the camera, U_{Model} represents the model for lens distortion, and d is the depth obtained directly from the depth image. The resulting de-projected 3D local path is then executed by a path follower module [11], which generates velocity commands from the input path. These commands utilize odometry readings to adjust the robot's movement in real time, ensuring accurate adherence to the planned trajectory. Finally, the robot driver module translates the velocity commands into motor commands, considering the robot's joint states to drive the hardware toward the desired state as defined by the 3D trajectory. A twist-command correction is implemented to ensure the robot's view does not significantly divert from the global goal position. An overview of the framework is depicted in Fig. 4.

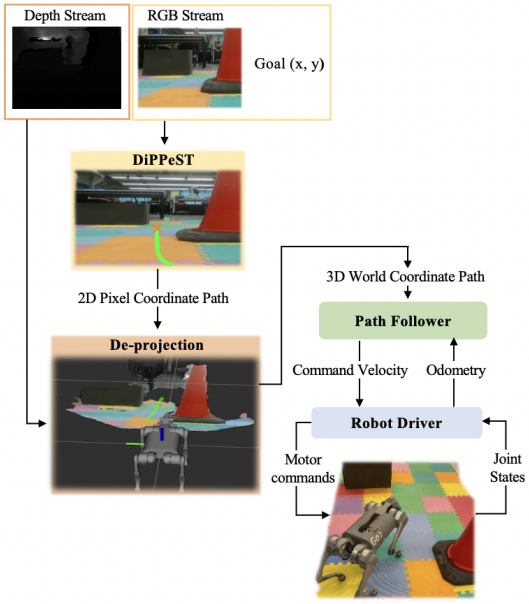


Fig. 4: Visual servoing framework for real-world robot path execution. DiPPeST 2D trajectories within RGB frame are de-projected into 3D world coordinates. The path follower translates the trajectories into velocity commands, and the robot module into motor commands to drive the robot towards the desired path.

V. RESULTS

A. Performance Evaluation

We conduct experiments to assess the robustness of DiPPeST against varying input conditions and validate its performance and generalization capabilities in out-of-distribution cases. These experiments are performed using the RGB frames obtained through a RealSense D435i camera. We focus on a) variation of the traversable region and obstacle pixel intensity, b) change of input image size, and c) variation of camera point-of-view (PoV), as seen in Fig.6. We quantify the effect as the average percentage of successful collision avoidance attempts for all obstacles in the scene, over 10 trials for each case. DiPPeST is executed on a computer with an Nvidia RTX 3090 GPU and has a constant inference time of 0.42s per frame, as it depends on DiPPeR's denoising time of 0.4s.

We examine the impact of varying the color of the traversable regions and obstacles in DiPPeST trajectory generation performance, as DiPPeR was only trained on black and white images (Fig. 2), with white representing the feasible path and black the obstacles. The color of the traversable region is varied by adding multi-color pads, introducing varying pixel intensities within the ROI. The mean variance values are converted into percentage values to standardize comparisons. DiPPeST's performance is quantified by assessing the correlation between changing the percentage ROI pixel variance and the success rate. DiPPeST achieves a mean success rate of 85% for generating successful trajectories, showing good generalization capabilities

when considering the training dataset. It successfully avoids obstacles with a 92% success rate for an ROI variance of less than 30%, representing the most common environmental scenario. Increasing pixel intensity variation results in a 19% decline for the edge case of ROI variance exceeding 80%, indicating a slight negative trend. This trend could potentially be corrected by introducing colorful maps into the training dataset.

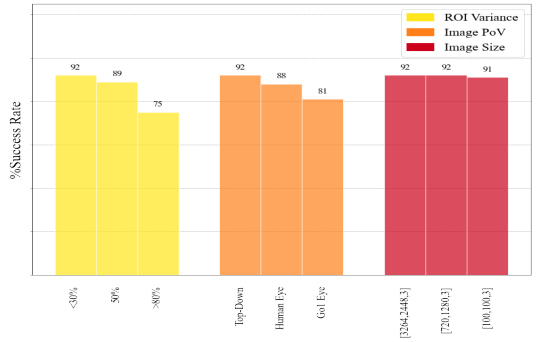


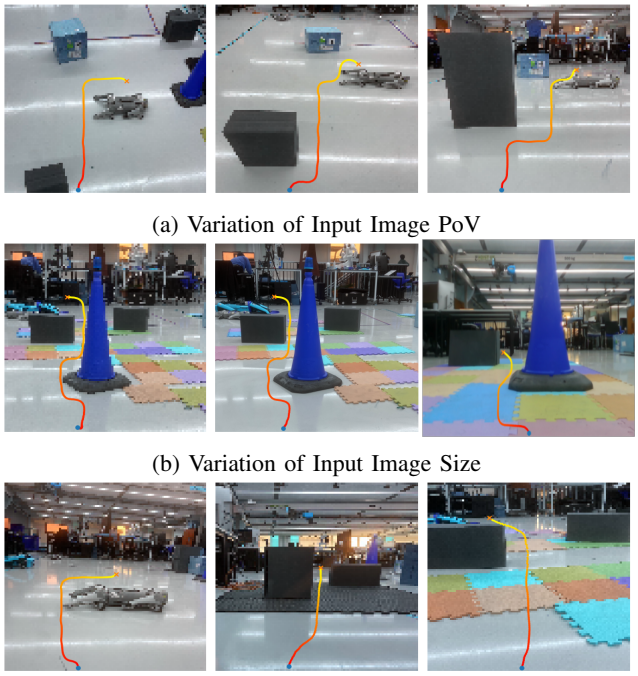
Fig. 5: The effect of variation of a) floor and obstacle color, b) input image size, and c) camera PoV, over DiPPeST % success rate.

We further evaluate the impact of image PoV on the success rate, considering that DiPPeR is trained on maps with a top-down PoV. DiPPeST should generalize to input images of varying PoV, reflecting variations in camera angle and heights in real-world scenarios. To assess this, we conduct the experiments on recorded RGB frames obtained by changing the angle and height of the position of the camera, as depicted in Fig.6a. DiPPeST generates successful trajectories with an average 87% success rate for all PoV variation experiments. There is a slight negative trend as the PoV angle changes from top-down to robot-level with an overall 13% decline in success rate, which could be corrected by refining the training dataset.

We also validate DiPPeST's performance for image inputs of variable sizes, as DiPPeR is trained on maps of the same size. DiPPeST receives camera input, which may have a variable FoV based on hardware specifications. This concern is based on the compression ratio variations occurring when the image input is passed on the encoder network. We examine images of varying sizes, including those of the same size as the training data set $[100, 100, 3]$, images taken from an iPhone 11 camera $[3264, 2448, 3]$ and a RealSense camera $[720, 1280, 3]$, as seen in Fig.6b. However, there is no significant effect on the success rate, with all scenarios displaying a 92% generalization success rate.

Overall, DiPPeST achieved an average success rate of 92% for standard environments, surpassing DiPPeR's validation performance. Additionally, it achieves an overall generalization success rate of 88%. These results are summarized in Fig. 5.

We also investigate the edge cases where the obstacles are similar in pixel intensity to the traversable region. DiPPeST's trajectory generation capabilities for these cases are shown in Fig. 6c. In the first subfigure of Fig. 6c, the obstacle and the floor have an average pixel intensity difference of 26% for all three channels, and on the second subfigure, a difference of 28%. Given that the intensity difference of DiPPeR validation maps is 97%, DiPPeST showcases a 73% superiority in generalization capabilities with an improvement factor of 3.5.



(a) Variation of Input Image PoV

(b) Variation of Input Image Size

(c) Edge Cases

Fig. 6: Illustration of DiPPeST generated trajectories for cases of variation of a) of camera PoV: top-down, human-eye, robot-eye, b) input image size: training dataset, iPhone 11, RealSense D435i, and c) edge cases. Low resolution frames correspond to the compressed images as received for the denoising process.

B. Real-World Evaluation

For real-world evaluation, the Unitree Go1 robot is used with DiPPeST, taking as input images from an Intel RealSense D435i camera mounted on the front at an angle of 10 degree depression. For all experiments, the global plan is generated from the first frame (Sec. IV-A) while the robot remains stationary. We test DiPPeST's performance for a) static environments and b) dynamic environments by measuring the rate of successful attempts in avoiding obstacles and reaching the goal position across all attempts. For static environment experiments, we test our method in lab conditions by constructing a course with stationary obstacles. For dynamic environments, walking robots are added within the scene to introduce dynamic obstacles. Each experiment is performed for 3 different instances of each environment, with increasing obstacle density and decreasing obstacle distance to showcase avoidance in narrow environments. The results are presented in Table I.

DiPPeST achieves an overall success rate of **80%** for all

Environment	sCA	sGR	dCA	dGR	%mSR
DiPPeST	**85**	79	78	**79**	**80.3**
IPlanner	76	**82**	67	78	75.6
NoMad	83	71	**83**	73	77.5

TABLE I: This table presents DiPPeST real-world Collision Avoidance (CA) and Goal Reaching (GR) mean % success rates (%mSR), compared to other SOTA methods.

environments, showing improved behavior against IPlanner, a geometric-based local planner and NoMad, a diffusion-based local planner. These SOTA works were selected as they generate local paths through RGB input, however both methods require extensive specific training for local planning as opposed to DiPPeST that is a zero-shot adapattion of DiPPeR which was trained on synthetic data. IPlanner displayed capabilities in reaching the goal, however it often stumbled upon obstacles while doing so. On the other hand, NoMaD demonstrates successful obstacle avoidance in both dynamic and static environments, but it would get stuck when reaching too close to a non-traversable area. Both SOTA methods avoid the narrow passage and redirect following a safer and less efficient path. An example of an executed DiPPeST path is depicted in Fig.1. DiPPeST's failure case in goal reaching occurred when the global waypoint disappeared from the view of many consecutive image frames, hence the local plan was continuously deviating from the global goal. Another limitation arises due to the restrictions on re-planning speed, which might lead to a collision if the speed of the dynamic obstacles is high.

VI. CONCLUSION

In this work, we introduced DiPPeST, a novel image-guided diffusion-based method for synthesizing both global and local paths for mobile robots. DiPPeST achieves a 92% success rate for simple environments and an average of 88% success rate in generalization and edge case handling for environments up to 3.5 times harder than DiPPeR. We implement visual-servoing to allow real-robot execution of the novel local planning framework, achieving 80% success rate in different environments and showcasing better behavior than two SOTA planners in a narrow passage experimental set-up. We identify that re-planning speed depends on inference time, which we aim to speed up by improving the denoising process. Additionally, goal-reaching is limited by the requirement of the global goal being visible within each frame, which we plan to address through the utilization of memory mechanism in the model or the use of Visual Odometry to better track features beyond a single frame. Our next steps focus on incorporating the kinodynamic properties of the robot during the trajectory generation process to achieve feasible real-world plans, which will enable the planner to navigate more complex environments with greater accuracy and efficiency.

REFERENCES

[1] N. Kottege and et al, "Editorial: Towards real-world deployment of legged robots," *Frontiers in Robotics and AI*, vol. 8, 2022.

[2] K. Cai, C. Wang, J. Cheng, C. W. De Silva, and M. Q.-H. Meng, "Mobile Robot Path Planning in Dynamic Environments: A Survey," *arXiv preprint arXiv:2006.14195*, 2020.

[3] J. Liu, M. Stamatopoulou, and D. Kanoulas, "Dipper: Diffusion-based 2d path planner applied on legged robots," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2024.

[4] J. Liu and et al, "ViT-A*: Legged Robot Path Planning using Vision Transformer A*," in *IEEE-RAS 22nd International Conference on Humanoid Robots (Humanoids)*, 2023, pp. 1–6.

[5] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song, "Diffusion Policy: Visuomotor Policy Learning via Action Diffusion," in *Robotics: Science and Systems (RSS)*, 2023.

[6] J. Carvalho, A. T. Le, M. Baierl, D. Koert, and J. Peters, "Motion Planning Diffusion: Learning and Planning of Robot Motions with Diffusion Models," *arXiv preprint arXiv:2308.01557*, 2023.

[7] M. Janner, Y. Du, J. Tenenbaum, and S. Levine, "Planning with Diffusion for Flexible Behavior Synthesis," in *International Conference on Machine Learning*, 2022.

[8] A. Sridhar, D. Shah, C. Glossop, and S. Levine, "Nomad: Goal masked diffusion policies for navigation and exploration," 2023.

[9] D. Kanoulas and et al, "Curved Patch Mapping and Tracking for Irregular Terrain Modeling: Application to Bipedal Robot Foot Placement," *Robotics and Autonomous Systems*, vol. 119, pp. 13–30, 2019.

[10] R. Saeed, D. R. Recupero, and P. Remagnino, "A boundary node method for path planning of mobile robots," *Robotics and Autonomous Systems*, vol. 123, p. 103320, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889018307310

[11] F. Yang, C. Wang, C. Cadena, and M. Hutter, "iplanner: Imperative path planning," 2023.

[12] P. Roth, J. Nubert, F. Yang, M. Mittal, and M. Hutter, "Viplanner: Visual semantic imperative learning for local navigation," 2023.

[13] D. H. Lee, S. S. Lee, C. K. Ahn, P. Shi, and C.-C. Lim, "Finite distribution estimation-based dynamic window approach to reliable obstacle avoidance of mobile robot," *IEEE Transactions on Industrial Electronics*, vol. 68, no. 10, pp. 9998–10 006, 2021.

[14] M. Missura and M. Bennewitz, "Predictive collision avoidance for the dynamic window approach," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 8620–8626.

[15] E. J. Molinos, Ángel Llamazares, and M. Ocaña, "Dynamic window based approaches for avoiding obstacles in moving," *Robotics and Autonomous Systems*, vol. 118, pp. 112–130, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889018309746

[16] L. Chang, L. Shan, C. Jiang, and Y. Dai, "Reinforcement based mobile robot path planning with improved dynamic window approach in unknown environment," *Autonomous Robots*, vol. 45, no. 1, pp. 51–76, 2021. [Online]. Available: https://doi.org/10.1007/s10514-020-09947-4

[17] Y. Kantaros, S. Kalluraya, Q. Jin, and G. J. Pappas, "Perception-based temporal logic planning in uncertain semantic maps," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2536–2556, 2022.

[18] J. Crespo, J. Castillo, O. Mozos, and R. Barber, "Semantic information for robot navigation: A survey," *Applied Sciences*, vol. 10, p. 497, 2020. [Online]. Available: https://doi.org/10.3390/app10020497

[19] J. K. Johnson, "Visual Servoing for Mobile Ground Navigation," in *88th IEEE Vehicular Technology Conference, VTC Fall 2018, Chicago, IL, USA, August 27-30, 2018*, 2018, pp. 1–5.

[20] J. Rodziewicz-Bielewicz and M. Korzeń, "Vision-based mobile robots control along a given trajectory," in *Artificial Intelligence and Soft Computing*, L. Rutkowski, R. Scherer, M. Korytkowski, W. Pedrycz, R. Tadeusiewicz, and J. M. Zurada, Eds. Cham: Springer Nature Switzerland, 2023, pp. 69–77.

[21] Z. Zhu, R. Wang, and X. Zhang, "Visible rrt*: Asymptotically optimal random search tree for visual servo tasks with the fov constraint," in *2023 42nd Chinese Control Conference (CCC)*, 2023, pp. 4633–4638.

[22] S. Hong, J. Lu, and D. P. Filev, "Dynamic Diffusion Maps-based Path Planning for Real-time Collision Avoidance of Mobile Robots," in *IEEE Intelligent Vehicles Symposium (IV)*, 2018, pp. 2224–2229.

[23] H. Ali, S. Murad, and Z. Shah, "Spot the Fake Lungs: Generating Synthetic Medical Images Using Neural Diffusion Models," in *Artificial Intelligence and Cognitive Science*, L. Longo and R. O'Reilly, Eds. Cham: Springer Nature Switzerland, 2023, pp. 32–39.

[24] N. Sharmin and R. Brad, "Optimal filter estimation for lucas-kanade optical flow," *Sensors (Basel)*, 2012.