ViT-A*: Legged Robot Path Planning using Vision Transformer A*

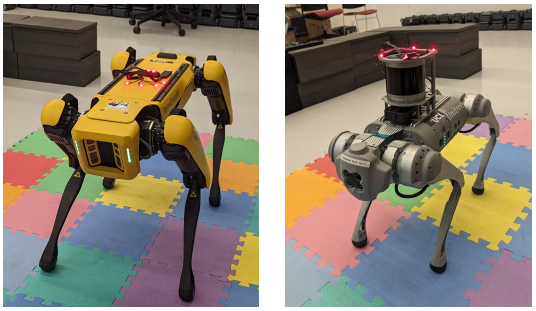
Jianwei Liu*, Shirui Lyu*, Denis Hadjivelichkov, Valerio Modugno, and Dimitrios Kanoulas

***Abstract*— Legged robots, particularly quadrupeds, offer promising navigation capabilities, especially in scenarios requiring traversal over diverse terrains and obstacle avoidance. This paper addresses the challenge of enabling legged robots to navigate complex environments effectively through the integration of data-driven path-planning methods. We propose an approach that utilizes differentiable planners, allowing the learning of end-to-end global plans via a neural network for commanding quadruped robots. The approach leverages 2D maps and obstacle specifications as inputs to generate a global path. To enhance the functionality of the developed neural network-based path planner, we use Vision Transformers (ViT) for map pre-processing, to enable the effective handling of larger maps. Experimental evaluations on two real robotic quadrupeds (Boston Dynamics Spot and Unitree Go1) demonstrate the effectiveness and versatility of the proposed approach in generating reliable path plans.**

I. INTRODUCTION

Legged robots, and especially quadrupeds, have seen tremendous progress over the past few years allowing them to carry out a wide range of tasks, ranging from package delivery [1], agricultural production [2], and search-rescue missions [3]. Path planning plays a crucial role in enabling legged robots to navigate autonomously and effectively in various complex environments. Several studies, e.g., [4], [5], were dedicated to the development of efficient path planning algorithms for quadrupedal robots, aiming to ensure their safe navigation while avoiding collisions with obstacles. Many of these works have utilized traditional methods such as Rapidly-exploring Random Trees (RRT) and A^*-based methods. Despite all the great efforts, achieving efficient and reliable path plans for mobile robots continues to present an ongoing challenge when using such traditional approaches. For example, established planning methods often struggle to effectively handle the complexities and uncertainties associated with real-time sensor inputs [6].

In contrast, the emergence of planners that integrate data-driven methods, fostered by the advancements in Deep Learning (DL), offers a promising avenue for addressing some of these challenges, e.g., empowering robots to learn and adapt from real-world data [7], [8]. This ability to gather knowledge from real-world experiences equips robots with the capacity to make more informed decisions in diverse and challenging situations.



(a) Boston Dynamics Spot (b) Unitree Go1

Fig. 1: The two robots (left: Boston Dynamics Spot, right: Unitree Go1) used for validating the proposed method.

In this paper, we present an approach that builds upon recent advancements in differentiable planners [8], enabling the learning of end-to-end mapping. Specifically, we focus on generating global paths for quadrupedal robots, by feeding 2D maps with obstacles into a deep neural network for A^*-based learning. Moreover, we enhance the functionality of our neural network-based planner by using a map pre-processing step with Visual Transformers (ViT) [9]. By introducing such an encoding, our method can leverage the strengths of transformers, particularly in capturing long-range dependencies and learning complex relationships in the input map images, while enabling the handling of larger maps efficiently. In the remainder of this paper, we refer to the proposed method as *ViT-A^** Path Planner. The contributions of this work can be summarized as follows. We introduce:

- a ViT-based Neural A^* Path Planner (ViT-A^*) that operates efficiently on maps of any dimension;
- a control stack to ensure the successful application of the proposed method on real quadruped robots in numerous application scenarios.

The subsequent sections are structured as follows. Sec. II provides an overview of the relevant literature, discussing previous works in the field, while Sec. III outlines the proposed method in detail. Sec. IV presents the experimental setup, including both simulation and real robot experiments. Finally, in Sec. V, we present the conclusions and discuss future developments of our research.

The authors are with the Department of Computer Science, University College London, Gower Street, WC1E 6BT, London, UK. {jianwei.liu.21, shirui.lyu.19, dennis.hadjivelichkov, v.modugno, 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].

II. RELATED WORK

A. Classical Path Planning

There are two main approaches to classical path planning algorithms: search-based and sampling-based methods. Search-based path planning provides mathematical guarantees of converging to a solution if it exists. A^* and its modifications have since found extensive use in robot navigation due to their simplicity in implementation and effectiveness in finding valid paths. For instance, in [10] the authors introduced an extension of A^* to drive a mobile platform to sanitize rooms. In [11], [12] A^* algorithms were used to find collision-free paths for a legged or legged-wheeled [13], [14] robots to achieve autonomous navigation. Extensions to these include footstep perception [15], [16], [17] and planning [18], or even navigation among movable obstacles [19], [20], [21]. Traditional methods heavily rely on a fixed heuristic function, such as the Euclidean distance, which lacks adaptability to varying robot configurations and environments. In our work, we propose a novel approach where we learn a heuristic based on the visual appearance of the application scenarios allowing the robot to make more informed decisions and thus reducing the overall search area and planning time.

Sampling-based planners efficiently create paths in high-dimensional space by sampling points in the state space. They can effectively work with continuous spaces. The literature in this context is vast, especially for applications in legged robotics. Some notable contributions in the field include [4], where an extension of an RRT-based algorithm is used for controlling a quadruped robot during the DARPA Challenge in 2015. More recently [5] introduced a novel sampling-based planner that shortens the computational time to find a new path in quadrupedal robots. While these approaches demonstrate satisfactory performance and probabilistic convergence, their limitations lie in the inability to incorporate image-based information directly into the planning process. As a result, their application is restricted in scenarios where planning based on visual data is not essential.

B. Data-Driven Path Planning

In contrast to the classical path-planning methods, state-of-the-art research in the field has shifted towards more practical solutions, which involve incorporating machine learning techniques. Data-driven methods have emerged as robust solutions to address these challenges by directly learning the behavior of pathfinding. These methods employ approaches such as expert demonstration [22] or imitation learning [6] to learn how to plan paths. Recent works directly address the issue of lack of semantically labeled maps in classical search-based methods by using data-driven approaches directly on raw image [23], [6], [24]. Specifically, Yonetani et al. [7] introduced Neural A^* – a differentiable variant of the canonical A^*, coupled with a neural network trained end-to-end. The method works by encoding natural image inputs into guidance maps and searching for path-planning solutions on them, resulting in significant performance improvements over previous approaches both in terms of efficiency and optimality. Our work expands upon this paper.

C. Vision Transformers

While methods such as the Neural A^* [7] have shown great promise in terms of performance improvements, they face limitations in processing larger maps due to the use of Convolutional Neural Networks (CNNs), where dealing with maps with increasing size could lead to a reduction in performance. This has posed some constraints in terms of processing larger maps. Transformers have emerged as a promising alternative, exhibiting significant performance improvements in various computer vision tasks [25], [26], [27] and robot vision tasks [28], [29], [30]. Transformers have the ability to capture long-range dependencies in images, thanks to their self-attention layers that enable them to attend to any part of the image regardless of the distance from the current location [9], [31]. This is in contrast to CNNs, which are confined to focus on local image patches. Moreover, due to their stacked layers, transformers can learn more complex relationships between different parts of the images while assuming fewer inductive biases [31]. In this work, we exploit the capability of the transformers to learn long-range dependencies to enhance the Neural A* performances with larger maps.

III. VIT-BASED NEURAL A^* PATH PLANNER

A. Neural A^* Planner

This work expands upon the Neural A^* Path Planner, introduced in [7]. Our method aims to provide global path plans as depicted in Fig. 2. In our approach, we introduce a ViT network, instead of the original CNN-based encoder-decoder structure, to process 2D maps of the environment. Unlike classification tasks using transformers that output a fixed-sized vector, our design allows a path planner to operate on variable-sized map inputs. To achieve this, we incorporate a decoder architecture that converts the embedded vectors for individual image patches back to the required guidance map. By utilizing the attention mechanism, our planner can effectively focus on key features in the planner task, such as obstacles, as well as start and goal positions, while exploiting the differentiable A^*'s ability to learn the decoding efficiently.

Neural A^* is a path planning algorithm that combines the convergence guarantees of A^* with the flexibility that characterizes a neural network to learn how to exploit visual cues in order to find near-optimal paths. In our setting, a i-th path planning problem is defined as

$$Q^i = (X^i, v_s^i, v_g^i, \overline{P}^i), \tag{1}$$

where X^i represents a 2D map of the current scenario, v_s^i and v_g^i are respectively the start and goal position, and $\overline{P}^i$ is a ground truth binary map representing the desired path. The Neural A^* path planner is composed of distinct sequential steps. Firstly, the 2D map X^i which has dimensions of $H \times W \times C$, where H and M are the dimensions of the map

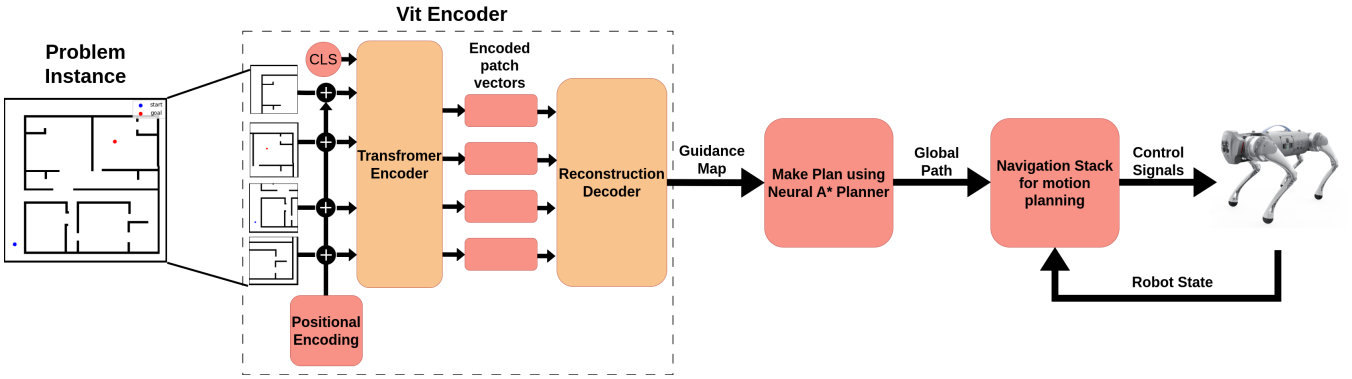


Fig. 2: Overall system, tested on the real robots. The 2D map is decomposed into patches and then fed to the ViT module. After the encoding-decoding process, the resulting Guidance Map is given to the A^* and it is used to find a global path. Finally, the global path is executed by the navigation stack, which controls the real robot to ensure small tracking errors.

and C indicates the number of color channels ($C = 3$ for RGB maps and $C = 1$ for binary occupancy maps) is fed to a CNN-based encoder. The encoder learns to map the raw image input to a guidance map defined as

$$f : \mathbb{R}^{H\times W\times C} \rightarrow \mathbb{R}^{H\times W}.$$

The guidance map represents the cost of traveling to adjacent nodes in the map, which is equivalent to the sum of the heuristic cost and the grid travel cost in regular A^* algorithms. Finally, a minimal cost path is found, following the guidance map and using the traditional A^* algorithm to explore the search space and find a valid path.

The differentiability of the path-finding process in the neural A^* path planner plays a crucial role in enabling the training of the CNN-based encoder pre-processor. This allows the neural network to learn and capture the essential features and patterns required for efficient path planning. The differentiability is achieved through a full matrix reformulation of the A^* algorithm, enabling the computation of gradients accounting for every search step during the back-propagation stage. In this paper, we focus solely on the node selection step for the original A^* for the sake of simplicity (for Neural A^* the cost terms in the node selections step are replaced with the guidance map). Therefore, given the regular A^* node selection rule:

$$v^* = argmin_{v\in\mathcal{O}} \left(g(v) + h(v)\right), \tag{2}$$

where $\mathcal{O}$ represents the list of candidate nodes (assuming a 2D map represented a graph where each pixel is a node), $g(v)$ refers to the accumulated total cost on the optimal path up to the node v, and $h(v)$ is the heuristic function that provides an estimation from the candidate node to the goal, the node selection step of the A^* path planner can be redefined in a matrix form as:

$$\mathcal{V}^* = \mathcal{I}_{max}\left(\frac{exp\left(-(G+H)/\tau\right)\odot O}{\langle exp\left(-(G+H)/\tau\right), O\rangle}\right), \tag{3}$$

where the functions G and H are the matrix formulation of the functions $g(v)$ and $h(v)$ respectively. The one-hot-encoding scheme is used in the following steps, which acts as a matrix mask such that only the selected node has the value one and zero otherwise. The one-hot-encoding for the next optimal node is denoted as $\mathcal{V}^*$. The parameter τ is determined empirically, and the symbol $A \odot B$ indicates an element-wise product between matrices A and B. During the forward pass, $\mathcal{I}_{max}$ is determined using the $argmax$ function, while during back-propagation, it is treated as an identity.

The loss function is computed as the average L_1 loss between the selected nodes from the A^* denoted as P (which represents a global path), and the ground-truth path map $\overline{P}^i$, which is given as input:

$$\mathcal{L} = \|P - \overline{P}^i\|_1/\mathcal{V} \tag{4}$$

This loss function serves to supervise the guidance map driven nodes selection by penalizing two types of errors: the absence of nodes that should have been included in P to correctly reconstruct $\overline{P}^i$, and the presence of an excessive number of nodes in P that do not belong to $\overline{P}^i$.

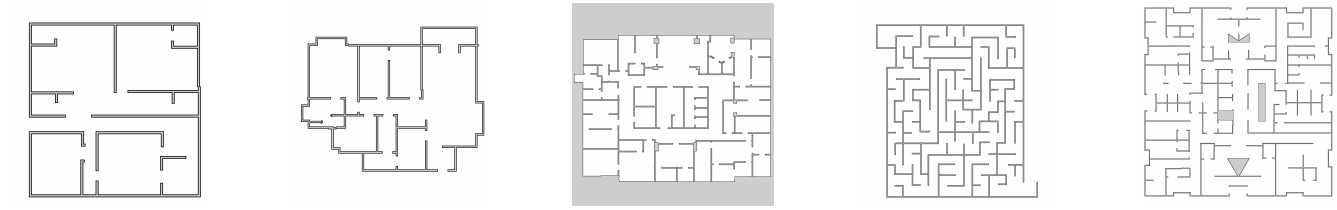
B. Vision Transformer for Guidance Map Encoding

To achieve a sophisticated and attention-based encoding for raw image inputs, we have introduced a Vision Transformer (ViT) [9] to extend the capability of the proposed method. The purpose of the model is to encode an input image into a guidance map by taking into account the visual cues. Detailed schematics of the ViT module can be found in Fig. 2.

The input map is initially represented as a tensor

$$X^i \in \mathbb{R}^{H\times W\times C},$$

where H, W, and C have already been defined in Sec III-A. Since the ViT module expects a sequential input, we reshape the map matrix into a flattened sequence of 2D patch vectors denoted as x_s, with the shape of $x_s \in R^{N\times(S^2\cdot C)}$. Here, S represents a hyper-parameter indicating the patch dimension, and $N = HW/S^2$ is the resulting number of patches from the map input.



(a) office01 (280 × 280) (b) room02 (360 × 360) (c) office02 (600 × 600) (d) maze (600 × 600) (e) mall (760 × 760)

Fig. 3: Maps used for comparing different planning methods with their sizes.

To ensure that the input map size is compatible with the required patches, we have introduced padding. This is necessary when the dimensions of the input map, H and W, are not an integer multiple of patch size S, and thus may not be segmented into the required patches.

In the sequence of patches, we incorporate a positional embedding, following the approach used in the ViT models [9]. This positional embedding serves to indicate the positional relationship between the patches, mimicking the spatial information presented in the original raw image.

To address the challenge of variable size inputs, we follow the idea proposed in the work presented in [32] where a positional upper bound representing the maximum number of possible patches is introduced. This ensures that the model can handle inputs of varying sizes without sacrificing training efficiency. Subsequently, each vector in the sequence is subjected to encoding, leading to the generation of an embedded vector projected into the hidden dimension. This encoding process effectively converts the input patches into a latent representation that captures their significant features.

Finally, the embedded vector sequence is decoded into vectors of size S^2 using the reconstruction decoder. The purpose of this decoder is to reconstruct the guidance map, which is required to have the same dimensions as the input map. The guidance map is a crucial component as it provides essential information for the planning cost. Each individual entry in the guidance map represents a corresponding guidance cost for planning, facilitating the decision-making process based on the encoded visual cues captured by the model.

IV. EXPERIMENTAL RESULTS

In this section, we test our method. Firstly, in Sec. IV-A, we run a path planning comparison between standard A^*, Neural Network-based A^* and ViT-based A^*. In Sec. IV-B, we test our method on two real robots, Boston Dynamics Spot and Unitree Go1.

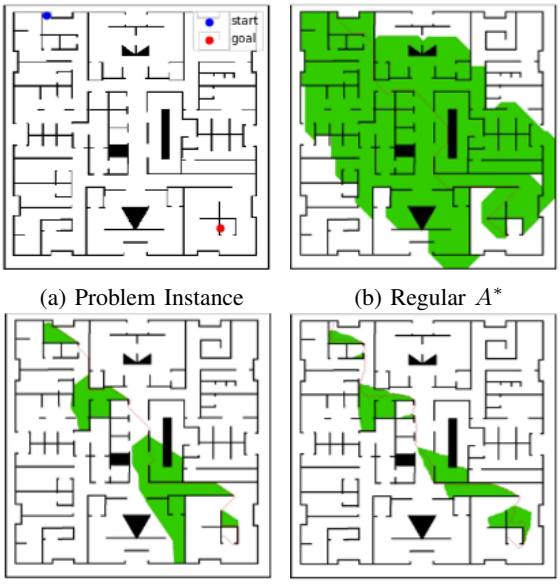
A. Benchmarking Comparison

Here, we compare our proposed ViT-based A^* path planner (ViT-A^*) as described in Sec. III-B, against two baselines: the Neural Network-based A^* (N-A^*) [7] and a classic A^* planner, as described in Sec. III-A. The evaluation is based on 2D maps coming from the MRPB benchmark dataset [33]. As we enabled our planner to work on variable map dimensions without specific training, the maps selected have different sizes that range from 280×280 to 760×760 pixels, and are all depicting realistic scenarios, such as offices or rooms (Fig. 3), which helps to bridge the reality gap when deploying the proposed method on real quadrupedal robots.

maps	ViT-A^*	N-A^*	A^*
(a)	5.68	**4.70**	6.03
(b)	17.31	**14.73**	17.51
(c)	**4.81**	5.17	15.59
(d)	**69.97**	75.20	84.84
(e)	**12.73**	16.57	36.24

TABLE I: **Planning time**: Average run-time (in sec) required to solve a single planning problem on maps from Fig. 3.

To guarantee an effective comparison, we defined a precise generation protocol for testing cases. First, to test the planners' generalization capability (especially for maps with different sizes) we generate random samples of start and goal positions. The randomly generated start and goal must not intersect with obstacles in order to define a valid test case. This constraint, together with the fact that there are no closed



(a) Problem Instance (b) Regular A^* (c) Neural-A^* (d) ViT-A^*

Fig. 4: Visualization of planning results by Regular A*, Neural A*, and ViT A*. In greem, the search area.

Fig. 5: From left to right we show a sequence of the Unitree Go1 (top) and Boston Dynamics Spot (bottom) robots navigating around obstacles using the ViT-A^* driven navigation stack.

regions in the map we used, ensures the completeness of the plan so it is ensured that a solution path exists. Furthermore, to avoid trivial planning tasks that are too short in length, we force the generation process to separate the start and goal positions by a threshold value. Hence, every planning task exceeds a certain length in the experiments.

In the experiments, we applied the loss function defined in Eq. 4 to train both models. To train the models, the RMSprop optimizer is selected and for both models, a learning rate of 0.001 is used. The CNN model and the ViT model are trained on the same dataset for 300 epochs or until converge criteria are met.

Each planner is evaluated by considering the *planning time* metric. The planning time is a key feature in evaluating the quality of a planner since it is a measure of the algorithm's efficiency.

The results shown in Table I are obtained by running each planner on the maps shown in Fig. 3. For each map, we compute the average planning time by repeating the path planning task 25 times with different start and goal positions. from Table I it is clear how our approach outperforms N-A^* and A^* especially for maps with larger size. Moreover, in Fig. 4 we show one instance of global path planning for the mall map (Fig. 3e). In this figure, it is apparent that our method is more efficient in finding a path due to the reduction of the search area depicted in green.

B. Experiments on Real Quadrupeds

To integrate the ViT-based A^* path planner with the legged robot's navigation system, the planner is incorporated into an existing 2D ROS Navigation stack[1] as a global path planner module. The overall architecture of the navigation stack is illustrated in Fig. 6.

Within the stack, the ViT-A^* module generates the globally optimal path given the occupancy map. This path is then refined via the local planner to ensure compliance with the robot's kinodynamic constraints. In this case, the Timed-Elastic-Band (TEB)[34], [35] local planner is employed. To mitigate the impact of state estimation on the quality of

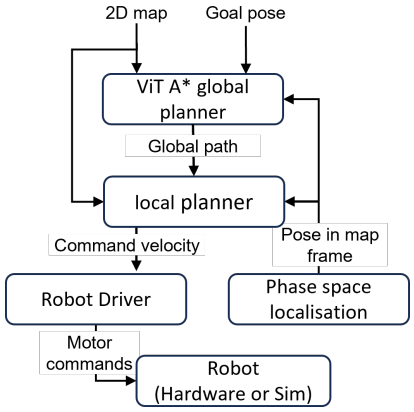


Fig. 6: The schematic structure of the navigation stack for the real robot.

evaluation of the path planning module, an external tracking system, specifically the Phasespace tracking cameras[2], is utilized. These cameras offer real-time localization for the robot at 960 Hz. Examples of the robots with active LED tracking markers can be seen in Fig. 1.

Note that in order to integrate the Neural A^* module as a global planner plugin, several modifications were necessary to interface it with the rest of the navigation stack. Firstly, the navigation stack utilizes the *OccupancyGrid* message to encode the map, which represents each cell in the map with the obstacle probability $p_o \in [0, 100]$. However, the current version of the ViT-A^* module can only take a binary occupancy map as input, where each $\text{cell}_i \in \{0, 1\}$. To convert the map, an occupancy threshold t is applied, in accordance to

$$\text{cell}_i = \begin{cases} 1, & p_o \geq t \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

Furthermore, the ViT-A^* module only produces paths as a sequence of positions, without considering the robot's orientation. Consequently, to generate the 3 Degrees of Freedom path compatible with the ROS stack, a simple

[1]http://wiki.ros.org/navigation

[2]https://www.phasespace.com/

forward-only orientation filter is implemented. This filter defines the orientation as the direction facing forward along the path, given a sequence of positions $\{v_i\}$ extracted from a global path, excluding the start and goal positions v_s and v_g (as their orientations are fixed by inputs - i.e. the current pose of the robot and desired goal pose). Hence, the orientation θ_i is defined as:

$$\theta_i = cos^{-1} \frac{v_i \cdot v_{i+1}}{|v_i||v_{i+1}|} \tag{6}$$

To validate the complete pipeline, tests were conducted on two real robots: Boston Dynamics Spot and Unitree Go1. In these tests, a predefined map is fed into the ViT-A^* module, which generates a global path. Subsequently, the robot executes the planned path via the ROS stack described in Figure 6. The effectiveness of the pipeline has been demonstrated in navigation scenarios around obstacles in our laboratory, as shown in Figure 5.

V. CONCLUSION

In this study, we present the ViT-A^* planning strategy that enables quadrupedal robots to autonomously and safely navigate in various and complex scenarios. Our proposed method builds upon recent advancements in differential planning and introduces a pre-processing model based on ViT, enabling our approach to handle maps of any size. The effectiveness of the proposed approach has been validated through successful comparison in simulation and on real quadrupedal robots across different scenarios. In future work, we intend to evaluate the performance of our method in outdoor or more complex settings (e.g., autonomous task planning [36]) and explore the benefits of planning directly on an RGB map with a ground truth path designed by humans.

REFERENCES

[1] Y. Ji, G. B. Margolis, and P. Agrawal, "Dribblebot: Dynamic legged manipulation in the wild," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 5155–5162.

[11] V. S. Raghavan *et al.*, "Agile Legged-Wheeled Reconfigurable Navigation Planner applied on the CENTAURO Robot," in *IEEE ICRA*, 2020.

[2] S. S. Garimella and S. Revzen, "Dandelion-Picking Legged Robot," *CoRR*, vol. abs/2112.05383, 2021.

[3] J. Liu, M. Tan, and X. Zhao, "Legged Robots — an Overview," *TIM*, vol. 29, no. 2, pp. 185–202, 2007.

[4] C. Lau and K. Byl, "Smooth RRT-Connect: An Extension of RRT-connect for Practical Use in Robots," in *IEEE Int. Conf. on Technologies for Practical Robot Applications (TePRA)*, 2015, pp. 1–7.

[5] Y. Zhang, H. Jiang, X. Zhong, X. Zhong, and J. Zhao, "MI-RRT-Connect Algorithm for Quadruped Robotics Navigation with Efficiently Path Planning," *JPCS*, vol. 2402, no. 1, p. 012014, 2022.

[6] S. Choudhury, M. Bhardwaj, S. Arora, A. Kapoor, G. Ranade, S. Scherer, and D. Dey, "Data-Driven Planning via Imitation Learning," *IJRR*, vol. 37, no. 13-14, pp. 1632–1672, 2018.

[7] R. Yonetani, T. Taniai, M. Barekatain, M. Nishimura, and A. Kanezaki, "Path Planning using Neural A* Search," in *International Conference on Machine Learning*, 2021, pp. 12 029–12 039.

[8] M. V. Pogančić, A. Paulus, V. Musil, G. Martius, and M. Rolinek, "Differentiation of Blackbox Combinatorial Solvers," in *International Conference on Learning Representations*, 2020.

[9] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, *et al.*, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," in *ICLR*, 2020.

[10] H. Huang, Y. Li, and Q. Bai, "An improved A* Algorithm for Wheeled Robots Path Planning with Jump Points Search and Pruning Method," *Complex Eng Syst*, vol. 2, p. 11, 2022.

[12] M. Kusuma, C. Machbub, *et al.*, "Humanoid Robot Path Planning and Rerouting Using A-Star Search Algorithm," in *IEEE International Conference on Signals and Systems (ICSigSys)*, 2019, pp. 110–115.

[13] V. S. Raghavan *et al.*, "Variable Configuration Planner for Legged-Rolling Obstacle Negotiation Locomotion: Application on the CENTAURO Robot," in *IEEE/RSJ IROS*, 2019.

[14] V. Sushrutha Raghavan *et al.*, "Reconfigurable and Agile Legged-Wheeled Robot Navigation in Cluttered Environments with Movable Obstacles," *IEEE Access*, 2021.

[15] D. Kanoulas *et al.*, "Vision-Based Foothold Contact Reasoning using Curved Surface Patches," in *IEEE Humanoids*, 2017, pp. 121–128.

[16] A. Stumpf and O. von Stryk, "A Universal Footstep Planning Methodology for Continuous Walking in Challenging Terrain Applicable to Different Types of Legged Robots," in *ICRA*, 2022, pp. 10 420–10 427.

[17] D. Kanoulas, N. G. Tsagarakis, and M. Vona, "Curved patch mapping and tracking for irregular terrain modeling: Application to bipedal robot foot placement," *Robotics and Autonomous Systems*, 2019.

[18] D. Kanoulas *et al.*, "Footstep Planning in Rough Terrain for Bipedal Robots using Curved Contact Patches," in *IEEE ICRA*, 2018.

[19] K. Ellis *et al.*, "Navigation Among Movable Obstacles with Object Localization using Photorealistic Simulation," in *IEEE IROS*, 2022.

[20] others, "Navigation Among Movable Obstacles via Multi-Object Pushing into Storage Zones," *IEEE Access*, 2023.

[21] Y. Linghong, V. Modugno, Y. Liu, D. Stoyanov, and D. Kanoulas, "Local Navigation Among Movable Obstacles with Deep Reinforcement Learning," *arXiv*, 2023.

[22] M. Pfeiffer, M. Schaeuble, J. Nieto, R. Siegwart, and C. Cadena, "From Perception to Decision: A Data-Driven Approach to End-to-End Motion Planning for Autonomous Ground Robots," in *ICRA*. IEEE, 2017, pp. 1527–1533.

[23] B. Ichter and M. Pavone, "Robot Motion Planning in Learned Latent Spaces," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2407–2414, 2019.

[24] L. Lee, E. Parisotto, D. S. Chaplot, E. Xing, and R. Salakhutdinov, "Gated Path Planning Networks," in *International Conference on Machine Learning*. PMLR, 2018, pp. 2947–2955.

[25] Z. Zong, G. Song, and Y. Liu, "DETRs with collaborative hybrid assignments training," 2022.

[26] P. Wang, S. Wang, J. Lin, S. Bai, X. Zhou, J. Zhou, X. Wang, and C. Zhou, "ONE-PEACE: Exploring one general representation model toward unlimited modalities," *arXiv preprint arXiv:2305.11172*, 2023.

[27] J. Zhu, H. Bai, and L. Wang, "Patch-Mix Transformer for Unsupervised Domain Adaptation: A Game Perspective," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 3561–3571.

[28] H. Ha and S. Song, "Semantic Abstraction: Open-World 3D Scene Understanding from 2D Vision-Language Models," in *Conference on Robot Learning*, 2022.

[29] W. Hao, C. Li, X. Li, L. Carin, and J. Gao, "Towards Learning a Generic Agent for Vision-and-Language Navigation via Pre-training," *CVPR*, 2020.

[30] D. Hadjivelichkov *et al.*, "One-shot transfer of affordance regions? AffCorrs!" in *Conference on Robot Learning (CoRL)*, ser. Proceedings of Machine Learning Research, K. Liu, D. Kulic, and J. Ichnowski, Eds., vol. 205, 14–18 Dec 2023, pp. 550–560.

[31] M. Raghu, T. Unterthiner, S. Kornblith, C. Zhang, and A. Dosovitskiy, "Do Vision Transformers See Like Convolutional Neural Networks?" in *Neural Information Processing Systems*, 2021.

[32] B. Wang, L. Chen, and B. Yang, "DM-NeRF: 3D Scene Geometry Decomposition and Manipulation from 2D Images," in *ICLR*, 2022.

[33] J. Wen, X. Zhang, Q. Bi, Z. Pan, Y. Feng, J. Yuan, and Y. Fang, "MRPB 1.0: A Unified Benchmark for the Evaluation of Mobile Robot Local Planning Approaches," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 8238–8244.

[34] C. Rösmann, W. Feiten, T. Wösch, F. Hoffmann, and T. Bertram, "Trajectory modification considering dynamic constraints of autonomous robots," in *German Conference on Robotics*. VDE, 2012, pp. 1–6.

[35] ——, "Efficient Trajectory Optimization Using a Sparse Model," in *European Conference on Mobile Robots*. IEEE, 2013, pp. 138–143.

[36] C. Zhou, C. Peers, Y. Wan, R. Richardson, and D. Kanoulas, "TeLeMan: Teleoperation for Legged Robot Loco-Manipulation using Wearable IMU-based Motion Capture," *arXiv*, 2022.