E(2)-Equivariant Graph Planning for Navigation

Linfeng Zhao, Hongyu Li, *Graduate Student Member, IEEE*, Taşkın Padır, *Senior Member, IEEE*, Huaizu Jiang, *Member, IEEE*, and Lawson L.S. Wong, *Member, IEEE*

***Abstract*—Learning for robot navigation presents a critical and challenging task. The scarcity and costliness of real-world datasets necessitate efficient learning approaches. In this letter, we exploit Euclidean symmetry in planning for 2D navigation, which originates from Euclidean transformations between reference frames and enables parameter sharing. To address the challenges of unstructured environments, we formulate the navigation problem as planning on a geometric graph and develop an equivariant message passing network to perform value iteration. Furthermore, to handle multi-camera input, we propose a learnable equivariant layer to lift features to a desired space. We conduct comprehensive evaluations across five diverse tasks encompassing structured and unstructured environments, along with maps of known and unknown, given point goals or semantic goals. Our experiments confirm the substantial benefits on training efficiency, stability, and generalization.**

***Index Terms*—Integrated planning and learning, deep learning methods, vision-based navigation.**

I. Introduction

NAVIGATION is a fundamental capability of mobile robots. Traditional navigation approaches, such as A* [1], focus on finding shortest-distance collision-free paths to a provided goal location in a pre-built occupancy map or known costmap. Recently, learning-based approaches to robot navigation have been proposed [2], [3], [4], [5], [6], which are particularly useful when the costs or goals are not explicitly provided and need to be learned from data. For example, in visual navigation, the cost to navigate between locations may depend on high-dimensional visual features, and the goal may likewise need to be visually identified (e.g., "find a mug"). As another example, in imitation learning, users may provide information about their preferred navigation policy implicitly via demonstrations, and the costs or optical actions need to be learned using features from the robot's state-action space.

Manuscript received 20 September 2023; accepted 19 January 2024. Date of publication 30 January 2024; date of current version 27 February 2024. This letter was recommended for publication by Associate Editor H. Soh and Editor H. Kurniawati upon evaluation of the reviewers' comments. This work was supported by NSF under Grant 2107256 and Grant 2142519. *(Linfeng Zhao, Huaizu Jiang, Lawson L.S. Wong, and Hongyu Li contributed equally to this work.) (Corresponding author: Hongyu Li.)*

Linfeng Zhao, Taşkın Padır, Huaizu Jiang, and Lawson L.S. Wong are with Northeastern University, Boston, MA 02115 USA (e-mail: zhao.linf@northeastern.edu; t.padir@northeastern.edu; h.jiang@northeastern.edu; l.wong@northeastern.edu).

Hongyu Li was with Northeastern University, Boston, MA 02115 USA. He is now with Brown University, Providence, RI 02912 USA (e-mail: hongyu@brown.edu).

More details can be found at the project website: https://lhy.xyz/e2-planning/.

Digital Object Identifier 10.1109/LRA.2024.3360011

While the aforementioned learning-based approaches exhibit remarkable capability in handling high-dimensional observations, they typically require a considerable amount of data and intensive training [2], [3]. Furthermore, these methods lack guarantees regarding generalization capabilities. In this work, we investigate the potential benefits of Euclidean symmetry in navigation tasks. It stems from Euclidean transformations among reference frames, enabling parameter sharing, enhancing efficiency, and improving generalizability. The utilization of symmetry in navigation within the *grid world domain* is explored in the earlier study by Zhao et al. [7] (left of Fig. 1). They introduce the equivariant version of the value iteration network (VIN) [8] under *discrete translations, rotations, and reflections*, along with a differentiable navigation planner. Their work showcases notable improvements compared to baseline approaches [8], [9]. However, they only focused on navigation in discrete 2D grids, which limits its applicability to robot navigation.

In our work, we introduce an equivariant learning-based navigation approach that operates on graphs in continuous space and considers symmetry with respect to an infinitely larger continuous group – the Euclidean group $\mathrm{E}(2)$ (right of Fig. 1). Specifically, we use *geometric graphs* (or spatial graphs) [10], where nodes in our graph correspond to states (and their features) arbitrarily located in 2D space. This eliminates the confinement to a grid, enabling the environment to remain non-discretized and permitting variable resolution. This also helps when the robot's motion deviates from grid-like patterns. Moreover, our approach accounts for continuous rotational symmetry, enhancing learning efficiency compared to discrete symmetry like Dihedral group D_4.

However, to exploit Euclidean symmetry in graph-based navigation, we need to solve two major challenges. First, previous work on 2D grids exploited the grid nature of their problem and used standard 2D symmetric convolution, which is no longer applicable in our case. Instead, we derive a new $\mathrm{E}(2)$-equivariant message-passing version of VIN and validate that it satisfies our notion of symmetry. Second, to capture symmetry in visual inputs/features, previous work relied on a very specific setup. As illustrated in Fig. 1, the agent was assumed to have four cameras, each situated 90° apart, exactly matching the D_4 symmetry being considered, such that a group transformation (rotation) can be implemented as a permutation to the four images. Extending this approach directly to $\mathrm{E}(2)$ would technically require an infinite number of cameras (or at least an infinite-resolution panoramic camera). We lift this restriction by introducing a learnable equivariant layer that can take images from a camera

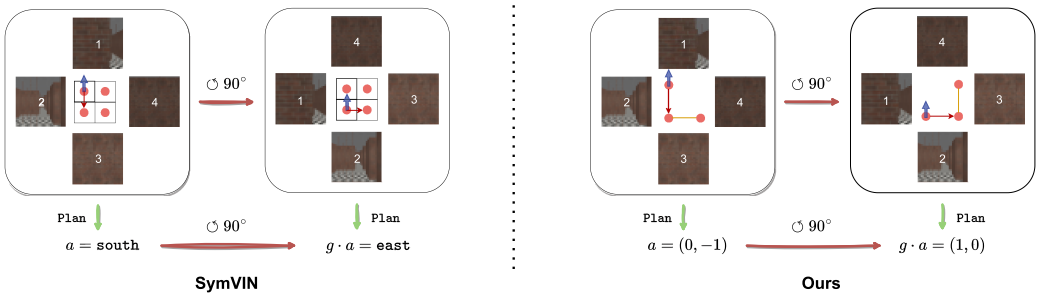

Fig. 1. Illustration of rotation equivariance. We provide a side-by-side comparison with SymVIN [7]. We use the blue arrow to show the orientation of the robot. Rotating the robot $\circlearrowleft 90°$ is equivalent to rotating the world frame $\circlearrowleft 90°$. When camera views are cyclically permuted, action output (red arrow) is transformed by a rotation matrix. The state space of SymVIN (left) is confined to the grid, and it only produces discrete actions. Our approach acts on continuous 2D space and produces $\mathbb{R}^2$ actions. .

array conforming to a subgroup of $\mathrm{E}(2)$ (such as D_8) and lift their features to become $\mathrm{E}(2)$-equivariant.

We empirically demonstrate the effectiveness of our approach on various navigation environments, including 2D grid, 2D geometric graphs, and Miniworld visual navigation [11] on both grid and graph. Moreover, in demonstrating its potential suitability for semantic goals and real-world environments, we provide a proof-of-concept experiment on semantic navigation tasks in the Habitat simulator [12]. Among these studies, we observe a consistent improvement in learning efficiency and stability. Overall, our study provides insight into the application of equivariance in navigation and the challenges. Our contributions are three-fold:

- We study the equivariance properties of 2D navigation and identify the two challenges.
- To address the challenges, we (1) derive the geometric message passing (MP) version of value iteration on geometric graphs and (2) propose using a learnable equivariant layer that converts multi-camera images to desired feature space, respectively.
- We demonstrate the empirical performance of navigation on Grid World (2D grid), Graph World (2D geometric graphs), and Miniworld visual navigation on both grid and graph. We provide proof-of-concept results on semantic navigation in Habitat simulator.

II. Related Works

Geometric deep learning: Our exploration of Euclidean symmetry utilizes tools from geometric deep learning [10], [13], [14], [15], [16]. Geometric deep learning and equivariant networks extend the study of classic 2D translation-equivariant convolution neural networks into more symmetry groups and spaces [10], [13]. Cohen and Welling [13] propose group convolution network (G-CNN), a pioneer work that studies rotation symmetry, followed by an extension to steerable convolution, Steerable CNN [17]. It has also been extended to the 3D case [18] and supported by a library in $\mathrm{E}(2)$ [16]. For graphs, equivariant message passing uses equivariant multilayer perceptrons (MLPs) to propagate geometric quantities between nodes to preserve the symmetry [14], [15]. Different from $\mathrm{E}(3)$-equivariant message passing in [15], we work on $\mathrm{E}(2)$ case. Additionally, the relationship between geometric graphs and value iteration has been discussed in [19]. In practice, equivariant networks enable sharing parameters and reduce the number of parameters.

Equivariance in reinforcement learning and planning: Our work draws upon previous research on symmetry in reinforcement learning (RL) and planning [7], [20], [21]. Symmetry and equivariance have been studied in reinforcement learning and planning before and in the era of deep learning [22], [23]. Invariance of the optimal value function and equivariance of the optimal policy of a Markov Decision Process (MDP) with symmetry have been shown in Zinkevich and Balch [24]. When using function approximation, equivariant policy networks and invariant value networks have been used to improve training efficiency in model-free RL [20], [21], and equivariance also helps in transition model and model-based RL [7], [25], [26], [27].

Learning to navigate: To achieve end-to-end navigation learning, several works investigate the differentiable planning algorithms [8], [28]. In this letter, we aim to investigate a particular class of planning algorithms that rely on Value Iteration Network (VIN) [8] and its variants [7], [9], [29], [30]. The selection of VINs is motivated by the fact that value iteration is fully differentiable and inherently encompasses an equivariant convolution [7]. Gupta et al. [31] adapt VIN to real-world applications with simultaneous mapping and planning, and Karkus et al. [32] propose DAN for end-to-end learning with structured representation. Prior to us, Zhao et al. [7] improved VIN with symmetry. However, these works operate on a structured 2D grid $\mathbb{Z}^2$. In this letter, we extend the planning to the 2D plane, enabling navigation in more realistic unstructured environments.

III. Background and Problem Formulation: Navigation as Geometric Graphs

In this section, we define the navigation problem under study and explore its symmetry aspects. Our formulation is a straightforward generalization of the global planning on occupancy grid [7], [8], with extensions including representing the navigation task through a *geometric graph* [10], [15]. Our objective is to train a planner that generates action $\boldsymbol{a}_t$ at state $\boldsymbol{s}_t$, guiding the agent to reach a target $\boldsymbol{w}$ on the graph: $\boldsymbol{a}_t = \texttt{policy}_\theta(\boldsymbol{s}_t, \boldsymbol{w})$. The target can be a spatial location (point goal) or semantic goal.

We base on the differentiable planner – VIN [8], allowing to consume input in high-dimensional features, e.g., image or even text embedding. As background, we first explain the problem definition, alongside the geometric structure and symmetry in the navigation graph. Then, we introduce the extension of equivariance in value iteration. Lastly, we delve into the incorporation of equivariance within the value iteration framework on the geometric graph.

Definition: We approach navigation as a 2D continuous path planning problem, building upon the 2D discrete grid version introduced in [7], [8], while extending it to the utilization of the *geometric graph* $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$ in 2D Euclidean space $\mathbb{R}^2$. In navigation tasks, the agent observes a state $\boldsymbol{s}_t \in \mathcal{S}$ at each step, and the action is to move on the 2D plane $\boldsymbol{a}_t \in \mathcal{A} = \mathbb{R}^2$. State $\boldsymbol{s}_t$ can be a 2D position in $\mathbb{R}^2$ or egocentric panoramic images in $\mathbb{R}^{K\times H\times W}$ (where K denotes the number of images of resolution $H \times W$).[1] To convert the task into a geometric graph, each node $v_i \in \mathcal{V}$ corresponds to a state $\boldsymbol{s} \in \mathcal{S}$ and is associated with a *node feature* $\boldsymbol{h}_i$ (such as images) and has a position $\boldsymbol{x}_i \in \mathbb{R}^2$. It is also possible to use *edge features*.[2] In this letter, we focus on addressing the global planning problem: given a navigation task (state $\boldsymbol{s}$ and target $\boldsymbol{w}$) as a feature field/map M, we output action field $\Pi = \texttt{Plan}_\theta(M)$.

Assumptions: The navigation challenge under consideration pertains to high-level global planning. In this context, we abstract the perception aspect (e.g., the method of acquisition) and the control aspect (assuming the feasibility of 2D relative movement output). Even if the execution of action does not arrive at another graph node, we may use action from the closest states or interpolate surrounding states. We assume a relatively accurate localization is provided.

Geometric Structure: The navigation problem can be defined as a MDP, and an inherent geometric structure emerges: it can be conceptualized as a geometric graph (defined above) situated within *a 2D Euclidean space*. Specifically, this graph can be transformed through 2D Euclidean isometric symmetries, without impacting the optimal solution of the MDP [7], [20]. The set of all such transformations in 2D is called *Euclidean group* $\mathrm{E}(2)$, which can be uniquely decomposed into translation part $\mathbb{R}^2$ and rotation/reflection part $\mathrm{O}(2)$, denoted as semi-direct product $\rtimes$: $\mathrm{E}(2) = \mathbb{R}^2 \rtimes \mathrm{O}(2)$ [13], [16]. We only require the node features $\boldsymbol{h}$ (and edge features) are transformable by a subgroup $G \leq \mathrm{E}(2)$. Following the notation in [16], we denote the rotation/reflection symmetry part as compact symmetry group $G \leq \mathrm{GL}(2)$, because translation group is *not compact* and many useful theorems do not hold. In our implementation, translation equivariance is achieved by using *relative position*. For any subgroups G of rotation/reflection, its equivariance needs *group convolution* [13] or *steerable convolution* [17].

Value Iteration and Symmetry: When symmetry appears in an MDP, the value and policy functions are equivariant [7], [20]. Abstractly, we can write value iteration ($\texttt{VI}$) as iteratively applying the Bellman operator $\mathcal{T} : V_t \mapsto V_{t+1}$:

$$\begin{aligned} Q_t(\boldsymbol{s}, \boldsymbol{a}) &:= R(\boldsymbol{s}, \boldsymbol{a}) + \int_{\mathbb{R}^2} d\boldsymbol{s}' P(\boldsymbol{s}' \mid \boldsymbol{s}, \boldsymbol{a}) V(\boldsymbol{s}'), \\ V_{t+1}(\boldsymbol{s}) &= \max_{\boldsymbol{a}} Q_t(\boldsymbol{s}, \boldsymbol{a}), \end{aligned} \tag{1}$$

where the input and output of the Bellman operator are both value function $V : \mathcal{S} \to \mathbb{R}$. Specifically, $\boldsymbol{s} \in \mathcal{S}, \boldsymbol{a} \in \mathcal{A}, R(\boldsymbol{s}, \boldsymbol{a}), P(\boldsymbol{s}' \mid \boldsymbol{s}, \boldsymbol{a})$ represent the states, actions, rewards, and transitions, respectively. In VIN, $\texttt{VI}(M) = \mathcal{T}_M^k[V_0]$ is executed k times, which takes an initial value V_0 and the map M (with a goal) as input

$$g \cdot \texttt{VI}(M) \equiv g \cdot \mathcal{T}_M^k[V_0] = \mathcal{T}_M^k[g \cdot V_0] \equiv \texttt{VI}(g \cdot M). \tag{2}$$

Zhao et al. [7] explore the equivariance for a 2D grid case. We extend it to geometric graph: $\mathcal{T}$ is performed on a graph, which is implemented using message passing.

Symmetry Transformations: In this paragraph, we unify the concepts presented in the preceding two paragraphs to demonstrate the implementation of equivariance constraints, which establish equivalence between transformed and original input/output [13], [17], [33]. Under the group transformation g, a (left) *regular* representation L_g transforms a feature map with c_{out}-dimensional vector (vector field) $f : X \to \mathbb{R}^{c_{\text{out}}}$ as [10], [17], [33]:

$$[L_g f](x) = \left[f \circ g^{-1}\right](x) = \rho_{\text{out}}(g) \cdot f\left(g^{-1} x\right), \tag{3}$$

where ρ_{out} is the G-representation associated with output $\mathbb{R}^{c_{\text{out}}}$. For example, for *action* feature map $\Pi : \mathbb{R}^2 \to \mathbb{R}^2$ (i.e., every position $\boldsymbol{x} \in \mathbb{R}^2$ is associated with a relative 2D movement), rotating the vector needs a 2×2 rotation matrix.

There are several useful functions in reinforcement learning (RL) and planning that can be written as graph features, e.g., node features as functions on $\mathcal{S}$ and edge features as functions on $\mathcal{S} \times \mathcal{A}$. We use $\rho_{\mathcal{S}}(g)$ to represent how the state is transformed under rotations and reflections $g \in G$, and similarly for action associated with representation $\rho_{\mathcal{A}}(g)$. Note that M and Π are *vector* maps, requiring additional transformation for their respective fibers (vectors). When $\mathcal{A}$ is continuous action, $\rho_{\mathcal{A}}$ is rotation matrices. For image-input case of $M : \mathcal{S} \to \mathbb{R}^{K\times H\times W}$, $\rho_{\text{camera}}(g)$ means cyclically permuting K cameras: $\rho_{\text{camera}}(g) \cdot M(\boldsymbol{s}_t) = M(\rho_{\mathcal{S}}(g) \cdot \boldsymbol{s}_t)$. It will be discussed in the next section. We list the equivariance conditions of the key MDP functions here.

$$\begin{aligned} R : \mathcal{S} \times \mathcal{A} \to \mathbb{R} : &\quad R(\boldsymbol{s}_t, \boldsymbol{a}_t) = R(\rho_{\mathcal{S}}(g) \cdot \boldsymbol{s}_t, \rho_{\mathcal{A}}(g) \cdot \boldsymbol{a}_t) \\ Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R} : &\quad Q(\boldsymbol{s}_t, \boldsymbol{a}_t) = Q(\rho_{\mathcal{S}}(g) \cdot \boldsymbol{s}_t, \rho_{\mathcal{A}}(g) \cdot \boldsymbol{a}_t) \\ V : \mathcal{S} \to \mathbb{R} : &\quad V(\boldsymbol{s}_t) = V(\rho_{\mathcal{S}}(g) \cdot \boldsymbol{s}_t) \\ \Pi : \mathcal{S} \to \mathcal{A} : &\quad \rho_{\mathcal{A}}(g) \cdot \Pi(\boldsymbol{s}_t) = \Pi(\rho_{\mathcal{S}}(g) \cdot \boldsymbol{s}_t) \end{aligned} \tag{4}$$

IV. Methodology: Equivariant Message Passing for Value Iteration

Following the spirit of VIN, we build a geometric message passing network and extend it to learning value iteration on geometric graphs: $\Pi = \texttt{Plan}_\theta(M)$. Given that the input feature

[1] We omit image RGB channel for notation simplicity.

[2] Similarly, each edge $e_{ij} \in \mathcal{E}$ corresponds to a state-action transition $(\boldsymbol{s}, \boldsymbol{a}) \in \mathcal{S} \times \mathcal{A}$ and has an *edge feature* $\in \mathbb{R}^{c_e}$ (such as distance or movement cost).

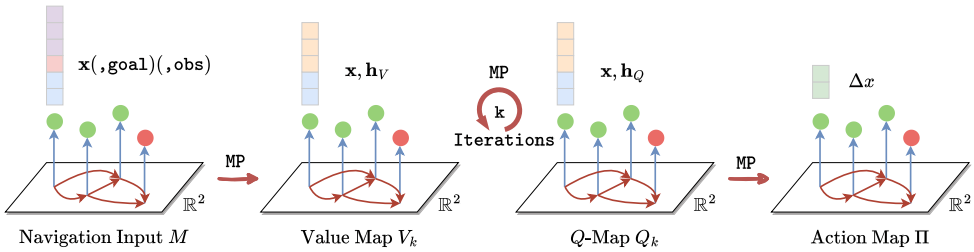

Fig. 2. Overview of the message passing planner network (MP-VIN). It takes the map M as the input, which contains the node position $\boldsymbol{x} \in \mathbb{R}^2$ and is optionally appended by the goal information (the red node is goal node) *or* observations depending on the navigation task. Then, the output is applied value iteration for k times. The state value map h_V and Q-value map h_Q are updated during value iterations. The final output is an action map Π: for each node, it is a continuous relative movement $\Delta x \in \mathbb{R}^2$.

map M and the resulting action map Π are both amenable to transformation within the same group, we enforce equivariance constraints throughout the MP network (shown in Fig. 1): $g \cdot \Pi = \texttt{Plan}_\theta(g \cdot M)$.

A. *Message Passing Value Iteration Networks (MP-VIN)*

The overview of the MP-VIN is shown in Fig. 2. For the input feature map M, each node contains a node position $\boldsymbol{x} \in \mathbb{R}^2$. Varied navigation tasks may lead to the augmentation of additional features, including goal features, observations, or a combination of both. For example, in the Grid World experiment (Section V-A), only the goal feature (a boolean value) is provided for each node. In the Miniworld experiment (Section V-C), both the goal feature and egocentric RGB observation are provided. In the semantic goal navigation experiment (Section V-E), only the RGB observation is provided, rendering the goal implicit.

Regarding the value iteration process, our MP-VIN is analogous to the original VIN formulation. However, what sets our approach apart is the improvement brought about by the inclusion of the geometric graph using an equivariant message passing layer (discussed in the next section). There are two advantages of using graph format: (1) cover the environment with irregular graphs to achieve variable resolution, and (2) output continuous actions.

B. $\mathrm{O}(2)$-*Equivariant Message Passing Layer: Equivariant Value Iteration on Graph*

Discretization to Graph: We could employ standard 2D convolution on regular grids for value iteration, as seen in VIN. However, irregular graphs render grids unsuitable. In the prior study of Niu et al. [34], an earlier iteration of graph convolution was employed. However, it exhibited equivariance only with respect to $\mathbb{R}^2$ translations, and it did not encompass considerations for rotation or reflection symmetries ($\mathrm{O}(2)$). Here, we derive from first principles using the original continuous form of value iteration.

The integral term in VI can be written as a mapping Φ

$$\boldsymbol{h}'(\boldsymbol{x}) = \Phi[h](\boldsymbol{x}) = \int_{\mathbb{R}^2} \boldsymbol{K}(\boldsymbol{x}, \boldsymbol{x}')\boldsymbol{h}(\boldsymbol{x}'), \tag{5}$$

where $\boldsymbol{K} : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}^{c_{\text{out}} \times c_{\text{in}}}$ is the kernal function.[3] $\boldsymbol{h} : \mathbb{R}^2 \to \mathbb{R}^{c_{\text{in}}}$ and $\boldsymbol{h}' : \mathbb{R}^2 \to \mathbb{R}^{c_{\text{out}}}$ are input and output feature map. If translation equivariance is desired, the kernel can be further simplified from two-argument to one-argument case, and the mapping is *convolution* [7], [35]. The continuous steerable convolution $\star$ is defined (via cross-correlation) by [16], [17], [33]:

$$\boldsymbol{h}'(\boldsymbol{x}) = [\boldsymbol{K} \star \boldsymbol{h}](\boldsymbol{x}) = \int_{\mathbb{R}^2} \boldsymbol{K}(\boldsymbol{x}' - \boldsymbol{x})\boldsymbol{h}(\boldsymbol{x}'), \tag{6}$$

where $\boldsymbol{K} : \mathbb{R}^2 \to \mathbb{R}^{c_{\text{out}} \times c_{\text{in}}}$ is a (steerable) kernel.

If we sample nodes in $\mathbb{R}^2$ and construct edges by transition $\mathcal{S} \times \mathcal{A}$, the continuous convolution on $\mathbb{R}^2$ can be discretized, which is similar to strategy of PointConv [15].[4] We use *nonlinear message passing* to replace linear convolution. We use two MLPs for computing messages ($\texttt{propagate}_\theta$) and updating node features ($\texttt{update}_\theta$), and has form

$$\begin{aligned} \boldsymbol{m}_{ij} &= \texttt{propagate}_\theta\left(\boldsymbol{h}_i, \boldsymbol{h}_j, \boldsymbol{x}_i, \boldsymbol{x}_j\right), \\ \boldsymbol{h}'_i &= \texttt{update}_\theta\left(\boldsymbol{h}_i, \sum_{j \in \mathcal{N}(i)} \boldsymbol{m}_{ij}\right). \end{aligned} \tag{7}$$

Implementation of Equivariance: We implement $\mathrm{E}(2)$-equivariant message passing on the graph that is equivariant under two parts:

Translation $\mathbb{R}^2$: In the plannar convolution on 2D grid, it is known to be equivariant to translation because it only relies on relative position between two cells as input and never takes absolute coordinates. Analogously, we use relative position between nodes $\boldsymbol{x}_i - \boldsymbol{x}_j$ as input to the message passing function [15]:

$$\boldsymbol{m}_{ij} = \texttt{propagate}_\theta\left(\boldsymbol{h}_i, \boldsymbol{h}_j, \boldsymbol{x}_i - \boldsymbol{x}_j\right). \tag{8}$$

It is a direct generalization of translation-equivariant 2D convolution that relies only on relative positions or local coordinates (shown in 6), allowing generalization to larger maps.

Rotation and Reflection $\mathrm{O}(2)$: We use steerable equivariant network to implement $\mathrm{O}(2)$-equivariance [13], [15], [16], [17],

[3]Note that the kernel $\boldsymbol{K}$ here is different from the notation K we use to represent the number of images.

[4]Brandstetter et al. [15] discuss other strategy for 3D steerable messsage passing, which expands the feature maps to spherical harmonics. Analogously, it is possible to expand the features to cyclic harmonics.

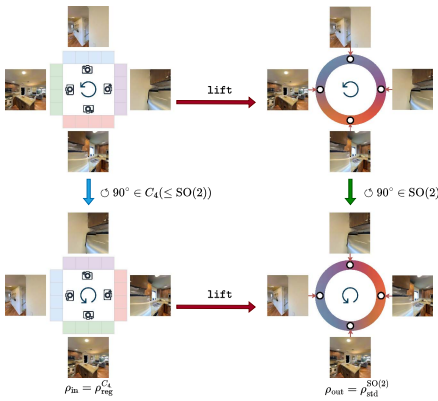

Fig. 3. Our proposed `lift` layer and its equivariance.

[35]. The $\mathrm{O}(2)$ group is compact and thus its representations are decomposable into *irreducible representations* [16], [33], thus convolutions can be performed in Fourier domain and more efficient. We use it to build equivariant MLPs of `propagate` and `update` (effectively 1×1 convolution). The kernel $\boldsymbol{K}$ of G-steerable convolution needs to satisfy constraint [16], [33], where G can be any (discrete) subgroup of $\mathrm{O}(2)$:

$$\boldsymbol{K}(gx) = \rho_{\text{out}}(g) \circ \boldsymbol{K}(x) \circ \rho_{\text{in}}(g)^{-1} \quad \forall g \in G, x \in \mathbb{R}^2, \quad (9)$$

where ρ_{in} and ρ_{out} stand for representations of the layer's input and output, respectively. This kernel constraint guarantees that the layer is G-equivariant: $\boldsymbol{K}(gx)\rho_{\text{in}}(g) = \rho_{\text{out}}(g) \circ \boldsymbol{K}(x)$. We refer the readers to Weiler and Cesa [16], Cohen and Welling [17] for more details.

C. C_K-Equivariant Lifting Layer: Processing Camera Array

In the previous section, we extend from discrete symmetry in SymVIN to continuous symmetry, such as continuous rotations $\mathrm{SO}(2)$. Injecting such equivariance into the *entire* network requires us to know how to **continuously** rotate sensory input by $g \in SO(2)$. This can be naturally achieved by two types of observations: (1) 360° point cloud input from a LiDAR (naturally continuous) or (2) 360° cylindrical camera. However, (1) may not seamlessly incorporate semantic information from RGB images, and (2) is hard to obtain and process. Thus, we need to relax this requirement of $\mathrm{SO}(2)$-transformable input modality. As a solution, we introduce a learnable layer `lift` that can map camera images from different views to a $\mathrm{SO}(2)$-transformable feature. This enhances our ability to exploit symmetry in the planning process.

We visually illustrate this in Fig. 3. For example, assume we have a robot equipped with four cameras facing north, east, west, and south (shown in the top left, as top down view). The observation from this camera array could only be cyclically permuted by $\circlearrowleft 90^\circ$ (or reflected), shown in the bottom left using the blue arrow. By using a equivariant learnable layer `lift`, it lifts the image features to become features on circle $S^1 \simeq \mathrm{SO}(2)$ shown on the right. They are transformable by $\mathrm{SO}(2)$, as shown via green arrow. We use small black circles to highlight that the feature at that point corresponds to that image.

Although the output is $\mathrm{SO}(2)$-transformable, the left side is only C_4-transformable, so the layer `lift` can only be *restricted* to be C_4-equivariant. The restriction from $G = \mathrm{SO}(2)$ to subgroup $H = C_4$ is called *restricted representation:* This layer is a special kind of equivariant induction layer [36]. It can lift features on a subgroup $H \leq G$ to a group G and is H-equivariant. Intuitively, it needs to satisfy the equivariance constraint only for $\circlearrowleft 90^\circ \in C_4$, which is a subgroup $C_4 \leq \mathrm{SO}(2)$:

$$\texttt{lift}(\circlearrowleft 90^\circ \cdot \texttt{images}) = \circlearrowleft 90^\circ \cdot \texttt{features}, \quad (10)$$

where we assume 4 images and output $\mathrm{SO}(2)$ features, while it can be any group such that C_4 is its subgroup.[5]

V. Experiments

We evaluate our proposed approach MP-VIN and baselines on four different tasks. Among these tasks, we perform point goal navigation under different environments: known structured environments (`Grid World`), known unstructured environments (`Graph World`), unknown structured environments (`Miniworld`), and unknown unstructured environments (`Miniworld-Graph`). Results are shown in Table 1.

Methods: We experiment four variants of our methods, with or without translation ($\mathbb{R}^2$) or rotation/reflection (using $G = D_8 \leq \mathrm{O}(2)$) equivariance: No-Sym, D_8, $\mathbb{R}^2$, and $\mathbb{R}^2 \rtimes D_8$. We use two grid-based methods: VIN [8] and SymVIN [7] (with D_4-equivariance). These methods are grid-based; therefore, we apply several modifications, including pre-processing and post-processing, to ensure fair comparisons. These modifications are detailed in the later sections. We also replace and compare our message passing module with the Graph Convolutional Networks [37] (GCN-VIN) and Graph Attention Networks [38] (GAT-VIN).

A. Planning on Known Maps: Grid World

Setting: In this task, we randomly generate synthetic mazes with size $m \times m$ (`Grid World`). We validate the performance on two different sizes $m \in \{15, 27\}$. Each cell on the maze map is represented as occupied (0) or unoccupied (1). There are four actions available for each cell on the map: north, east, west, and south. We randomly select a goal on the map and generate a goal map, where the cell containing the goal is marked as 1. Each cell is labeled by the ground-truth action using Dijkstra's algorithm.

To apply our planners for graphs, we transform the grid representation into connectivity graphs [34]. Each *node* of the graph is a cell in the 2D grid and is associated with a 4-D *node feature vector*, which has (1) (x, y) location, (2) whether the node is an obstacle, and (3) whether the node is the goal. Any two nodes are connected by an *edge* if they are neighbors on the grid map, i.e., obstacles are not connected.

Results: MP-VIN with $\mathbb{R}^2 \rtimes D_8$ symmetry demonstrates faster learning efficiency than its graph-based variants and VIN (the left of Fig. 4). We surprisingly find that MP-VIN with

[5]One solution for D_4 group is to use *quotient* representations, but it not generally applicable for higher-degree rotations such as D_8 or infinitesimal rotations $\mathrm{SO}(2)$.

TABLE I
AVERAGED TEST SUCCESS RATE (%) WITH STANDARD DEVIATION. THE BEST RESULT IS BOLDED

Method	Grid World		Graph World		Miniworld	
	15×15	27×27	128 nodes	256 nodes	Grid	Graph
VIN [8]	78.51±1.81	50.15±3.94	18.75±1.95	20.09±6.87	57.14±8.92	18.90±2.87
SymVIN [7]	**95.85**±5.02	**93.73**±7.33	24.40±2.11	27.53±4.73	**91.67**±2.58	27.98±4.34
MP-VIN (No Sym)	87.07±4.53	55.99±39.56	63.10±17.26	54.76±3.29	–	–
MP-VIN: D_8	87.19±2.78	38.53±16.17	52.38±5.02	32.89±3.47	–	–
MP-VIN: $\mathbb{R}^2$	90.81±1.02	72.45±31.94	<u>70.24</u>±2.69	<u>58.33</u>±5.93	79.76±24.06	<u>96.58</u>±2.46
MP-VIN: $\mathbb{R}^2 \rtimes D_8$	<u>91.50</u>±1.04	<u>84.52</u>±6.04	**72.17**±5.08	**61.90**±5.33	<u>90.89</u>±1.63	**96.96**±1.00

The best result is bolded. The second-best result is underlined.

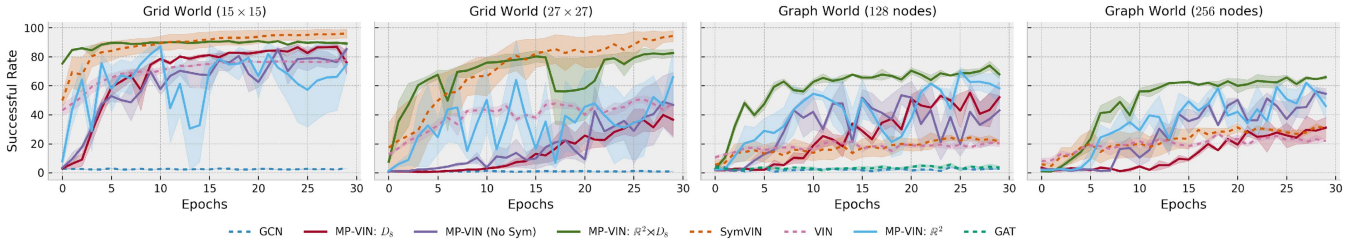

Fig. 4. Learning curves on the Grid World experiments (left two) and the Graph World experiments (right two). The shadow area shows the standard error. Dashed lines are for non-MP-VIN methods (VIN, SymVIN, GCN-VIN, and GAT-VIN).

$\mathbb{R}^2 \rtimes D_8$ has much smoother learning curves than its variants without D_8 symmetry ($\mathbb{R}^2$ and No-Sym). This indicates that injecting Euclidean symmetry may improve the loss landscape. In terms of absolute performance gain, by adding D_8 symmetry to MP-VIN with $\mathbb{R}^2$, we obtain another 0.69% and 12.07% success rate on the 15×15 and 27×27 mazes, respectively. However, it is still outperformed by SymVIN, which uses steerable 2D convolution to process the input. It is reasonable as it directly uses the regular grid structure, while our graph version can handle unstructured grpahs and is more expressive, while we apply both of them on grid maps. When the map size increases, MP-VIN with $\mathbb{R}^2 \rtimes D_8$ symmetry demonstrates the second-least performance degradation, showing better generalization to larger maps.

B. Planning on Known Graphs: Graph World

Setting: We validate the performance of our approach in unstructured graph environments (`Graph World`). We follow the setup of [34] to generate the random graphs. We randomly generate N nodes, each with coordinates between (0,0) and (m, m). These nodes are connected using a KNN graph. We randomly select some nodes as the obstacle nodes, and one node as the goal node.

To verify the performance of grid-based approaches in Graph World, we discretize the environment [34]. We round down the coordinates of each node to map it to a cell on the grid. The obstacle feature is carried over to the grid. Ultimately, we verify its performance on the graph by transforming the discrete actions into continuous actions (represented in $[x, y]$ coordinate). For example, `north` is transformed into [1,0], and `east` is transformed into [0,1].

Results: MP-VIN with $\mathbb{R}^2 \rtimes D_8$ symmetry demonstrates the strongest performance in this task in terms of learning efficiency and smoothness of learning curve (Fig. 4). This is because the graphs generally do not have regular structure, i.e. four neighbors only in four directions. Thus, all methods encounter performance degradation, while grid-based methods struggle more in such unstructure graphs.

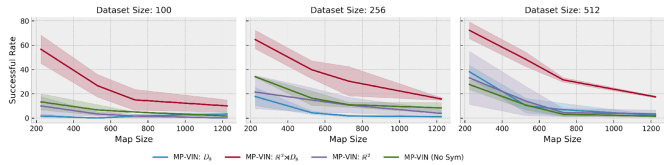

Fig. 5. Data efficiency and size generalization. We demonstrate data efficiency across 100, 256, and 512 training samples. For models trained on each dataset, we show size generalization by training them on the smallest size and directly testing them on larger ones.

Data Efficiency: As shown in Fig. 5, we evaluate the data efficiency by assessing the model tranined on varied dataset sizes (100, 256, 512 samples). Even when trained with only 100 samples, our approach, incorporating E(2) symmetry, consistently outperforms the baselines (w/o E(2) symmetry) trained with 512 samples. These results confirm the substantial gains in data efficiency achieved by leveraging Euclidean symmetry.

Size Generalization: Fig. 5 illustrates our evaluation of size generalization ability. We train our model on a small graph consisting of 225 nodes and subsequently test it on larger graphs without any further fine-tuning. Remarkably, even as the complexity of the environment increases, our approach consistently outperforms the compared methods by a large margin. This underscores the added value of incorporating E(2) symmetry in enhancing the model's generalizability to diverse environments.

C. Mapping and Planning Under Unknown Maps: Miniworld

Setting: We compare different methods in a more challenging visual environment (`Miniworld`), where the models learn

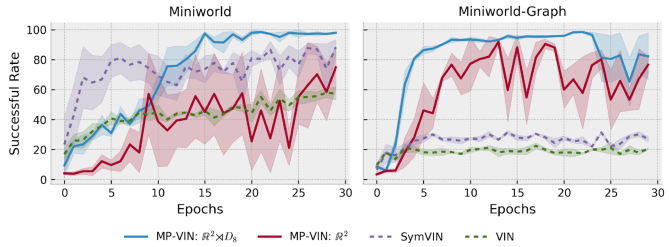

Fig. 6. Learning curves on the `Miniworld` experiment (top) and `Miniworld-Graph` experiment.

Fig. 7. Visualization of our navigation environment. On the left, we show the constructed geometric graph in an HM3DSem scene. The density of color represents the distance to the goal. On the right, we demonstrate the observation example on each node, which consists of four egocentric RGB images facing four directions.

mapping and planning simultaneously. We leverage the Miniworld simulator [11] to render the randomly generated maze into a 3D visual environment. Different from the Grid World, we are not given a map in this experiment. Instead, we use egocentric RGB observations. For each cell in the maze environment, we obtain the RGB images from the cameras facing four orientations (0°, 90°, 180°, 270°). We transform the dataset into a graph using the same approach as mentioned in Section V-A. In order to estimate the map of the environment, we encode the visual observations into occupancy features using a mapper network [7], [9]. Results are shown in Fig. 6

Results: Since this task is based on a 15×15 grid using visual observations, the experiment results are similar to the Grid World. MP-VIN with $\mathbb{R}^2 \rtimes D_8$ symmetry demonstrates higher learning efficiency than MP-VIN with only $\mathbb{R}^2$ symmetry and VIN. However, every method faces a performance drop due to the mapping uncertainty. We observe that the performance gap of MP-VIN with $\mathbb{R}^2 \rtimes D_8$ symmetry (0.61%) is lower than that of SymVIN (4.18%). Therefore, the performance gap between MP-VIN with $\mathbb{R}^2 \rtimes D_8$ symmetry and SymVIN becomes narrower.

D. Mapping and Planning Under Unknown Graphs: Miniworld-Graph

While Miniworld is a 3D-rendered visual environment, the state and action spaces are still discrete (grid-based). To show real-world feasibility, we aim at a more realistic setting in this experiment.

Setting: We sample random navigation graphs (256 nodes) in the Miniworld environment. The edges between nodes represent navigability. Like the Miniworld experiment in Section V-C, each node contains a panoramic egocentric RGB observation facing four directions. We use a similar mapper to estimate the map from the visual observations. Differently, we estimate the occupancy graph instead of the occupancy grid.

Results: Similar to the Miniworld experiment on grid representation, we observe the MP-VIN with $\mathbb{R}^2 \rtimes D_8$ symmetry has higher learning efficiency than MP-VIN with only $\mathbb{R}^2$ symmetry. Grid-based approaches suffer in this task since it is hard for CNN to process the expressive unstructured environment, also indicated in the previous `Graph World` experiment. In both Miniworld experiments, we observe that MP-VIN with $\mathbb{R}^2 \rtimes D_8$ symmetry has a much smoother learning curve and lower variance than MP-VIN with only $\mathbb{R}^2$ symmetry. This indicates that by adding $\mathbb{R}^2$ symmetry, we could further optimize the network with better stability.

TABLE II
PERFORMANCE IN SEMANTIC NAVIGATION TASK

Method	Successful Rate (%)
MP-VIN (No Sym)	69.70±1.07
MP-VIN: $\mathbb{R}^2 \rtimes C_4$	**74.27**±3.12

E. Planning With Semantic Goal

Setting: To confirm the validity of our approach in a more realistic setting, we perform a proof-of-concept semantic navigation task using the real-world collected Habitat-Matterport 3D semantics dataset (HM3DSem) [39]. Our model learns to seek an object in the environment given only RGB observations. In our experiment, we consider the most common object among all environments ("refrigerator"). We assume access to fully-observable environment information, i.e. camera observations at any location.

To achieve this, we randomly sample nodes in the navigatable areas using the Habitat simulator [12]. We construct a graph on the sampled nodes using a radius graph. The edges that lead to infeasible motion (e.g. crossing the wall) are removed. We use the provided ground-truth object location to label the ground-truth action for each node. Note that the object location is unknown during testing. We obtain four egocentric RGB images for each node, which are the only observations given to our model. Visualization can be found at Fig. 7.

For each scene, we randomly sample 20 graphs, in which each graph contains 128 nodes. Each RGB image has the size of $3 \times H \times W$, and we set $H = W = 128$ in our following experiments. We extract image features ($d = 128$) from the RGB observation using ResNet-34. During training, we freeze the entire ResNet except for the last output layer. The image features are fed into the differentiable planner to generate the optimal plan.

Results: We utilize MP-VIN without symmetry as the baseline, and compare MP-VIN with $\mathbb{R}^2 \rtimes C_4$ symmetry. The result is shown in Table II. Our findings demonstrate that incorporating C_4 symmetry into MP-VIN leads to an improvement of 4.57% in the success rate.

VI. Conclusion and Discussion

In this letter, we explored the applicability of exploiting Euclidean symmetry within the context of a navigation planner. We contributed a novel equivariant differentiable planner. The effectiveness of the proposed approach is extensively assessed across four distinct tasks involving structured and unstructured environments, with known and unknown maps. The empirical findings demonstrate a significant enhancement in learning efficiency when Euclidean symmetry is integrated into 2D navigation planning. Furthermore, the results indicate that leveraging Euclidean symmetry yields more stable optimization and yields superior overall performance. In the future, we hope to extend our work to navigation task that has higher dimension, such as semantic navigation [2], [3], [4].

Limitations: Inheriting from VIN, our message passing planner also considers only fully-observable states and takes observations of all states as input at once [7], [8], [9], which is impractical in real-world navigation. One potential direction is to consider partial observation, as done in [40]. Another potential direction is to combine with differentiable filter to counter the uncertainties [32]. To facilitate deployment on a real robot, it may be helpful to consider augmenting our state space with an additional orientation dimension [40].

In this letter, our primary emphasis lies within the Euclidean group E(2). Nevertheless, in future works, potential performance improvement may be achieved by broadening the scope of the group employed, e.g. incorporating the scaling and general linear group.

References

[1] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cybern.*, vol. 4, no. 2, pp. 100–107, Jul. 1968.

[2] D. S. Chaplot et al., "Object goal navigation using goal-oriented semantic exploration," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 4247–4258.

[3] M. Chang, A. Gupta, and S. Gupta, "Semantic visual navigation by watching YouTube videos," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 4283–4294.

[4] Y. Liang, B. Chen, and S. Song, "SSCNav: Confidence-aware semantic scene completion for visual semantic navigation," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 13194–13200.

[5] N. Ü. Akmandor, H. Li, G. Lvov, E. Dusel, and T. Padir, "Deep reinforcement learning based robot navigation in dynamic environments using occupancy values of motion primitives," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 11687–11694.

[6] A. Brohan et al., "RT-1: Robotics transformer for real-world control at scale," Dec. 2022, *arXiv:2212.06817*.

[7] L. Zhao, X. Zhu, L. Kong, R. Walters, and L. L. S. Wong, "Integrating symmetry into differentiable planning with steerable convolutions," in *Proc. 11th Int. Conf. Learn. Representations*, 2023.

[8] A. Tamar et al., "Value iteration networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 2146–2154.

[9] L. Lee et al., "Gated path planning networks," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 2947–2955.

[10] M. M. Bronstein et al., "Geometric deep learning: Grids, groups, graphs, geodesics, and gauges," Apr. 2021, *arXiv:2104.13478*.

[11] M. Chevalier-Boisvert et al., "Minigrid & miniworld: Modular & customizable reinforcement learning environments for goal-oriented tasks," in *Proc. 37th Conf. Neural Inf. Process. Syst. Datasets Benchmarks Track*, 2023.

[12] M. Savva et al., "Habitat: A platform for embodied AI research," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 9339–9347.

[13] T. Cohen and M. Welling, "Group equivariant convolutional networks," in *Proc. 33rd Int. Conf. Mach. Learn.*, 2016, pp. 2990–2999.

[14] V. G. Satorras, E. Hoogeboom, and M. Welling, "E(n) equivariant graph neural networks," in *Proc. 38th Int. Conf. Mach. Learn.*, 2021, pp. 9323–9332.

[15] J. Brandstetter et al., "Geometric and physical quantities improve E(3) equivariant message passing," in *Proc. Int. Conf. Learn. Representations*, 2021.

[16] M. Weiler and G. Cesa, "General E(2)-equivariant steerable CNNs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 14334–14345.

[17] T. S. Cohen and M. Welling, "Steerable CNNs," in *Proc. Int. Conf. Learn. Representations*, 2016.

[18] M. Weiler et al., "3D steerable CNNs: Learning rotationally equivariant features in volumetric data," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 10402–10413.

[19] A. J. Dudzik and P. Veličković, "Graph neural networks are dynamic programmers," in *Proc. Adv. Neural Inf. Process. Syst.*, 2022, pp. 20635–20647.

[20] E. van der Pol et al., "MDP homomorphic networks: Group symmetries in reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 4199–4210.

[21] D. Wang, R. Walters, and R. Platt, "SO(2)-equivariant reinforcement learning," in *Proc. Int. Conf. Learn. Representations*, 2021.

[22] M. Fox and D. Long, "Extending the exploitation of symmetries in planning," in *Proc. 6th Int. Conf. Artif. Intell. Plan. Syst.*, 2002, pp. 83–91.

[23] N. Pochter, A. Zohar, and J. S. Rosenschein, "Exploiting problem symmetries in state-based planners," in *Proc. 25th AAAI Conf. Artif. Intell.*, 2011, pp. 1004–1009.

[24] M. Zinkevich and T. R. Balch, "Symmetry in markov decision processes and its implications for single agent and multiagent learning," in *Proc. Int. Conf. Mach. Learn.*, 2001, p. 632.

[25] L. Zhao et al., "Toward compositional generalization in object-oriented world modeling," in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 26841–26864.

[26] J. Y. Park et al., "Learning symmetric embeddings for equivariant world models," in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 17372–17389.

[27] L. Zhao et al., "SE(3) frame equivariance in dynamics modeling and reinforcement learning," in *Proc. Int. Conf. Learn. Representations Workshop Phys. Mach. Learn.*, 2023.

[28] R. Yonetani et al., "Path planning using neural A* Search," in *Proc. 38th Int. Conf. Mach. Learn.*, 2021, pp. 12029–12039.

[29] L. Zhao, H. Xu, and L. L. Wong, "Scaling up and stabilizing differentiable planning with implicit differentiation," in *Proc. Int. Conf. Learn. Representations*, 2023.

[30] L. Zhao and L. L. Wong, "Model-based navigation in environments with novel layouts using abstract 2-D maps," in *Proc. Int. Conf. Neural Inf. Process. Syst. Workshop Deep Reinforcement Learn.*, 2020.

[31] S. Gupta et al., "Cognitive mapping and planning for visual navigation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.* 2017, pp. 2616–2625.

[32] P. Karkus et al., "Differentiable algorithm networks for composable robot learning," in *Proc. Robot.: Sci. Syst.*, 2019.

[33] L. Lang and M. Weiler, "A wigner-eckart theorem for group equivariant convolution kernels," in *Proc. Int. Conf. Learn. Representations*, 2020.

[34] S. Niu et al., "Generalized value iteration networks: Life beyond lattices," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 6246–6253.

[35] T. S. Cohen, M. Geiger, and M. Weiler, "A general theory of equivariant CNNs on homogeneous spaces," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 9142–9153.

[36] O. L. Howell, D. Klee, O. Biza, L. Zhao, and R. Walters, "Equivariant single view pose prediction via induced and restriction representations," in *Proc. 37th Conf. Neural Inf. Process. Syst.*, 2023.

[37] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Representations*, 2017.

[38] P. Veličković et al., "Graph attention networks," in *Proc. Int. Conf. Learn. Representations*, 2018.

[39] K. Yadav et al., "Habitat-matterport 3D semantics dataset," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 4927–4936.

[40] S. Ishida and J. F. Henriques, "Towards real-world navigation with deep differentiable planners," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit*, 2022, pp. 17306–17315.