IROS '24
ABU DHABI

Northeastern University

BROWN

E(2) Equivariant Graph Planning for Navigation

IEEE Robotics and Automation Letters (RA-L)
Presenting at IROS 2024

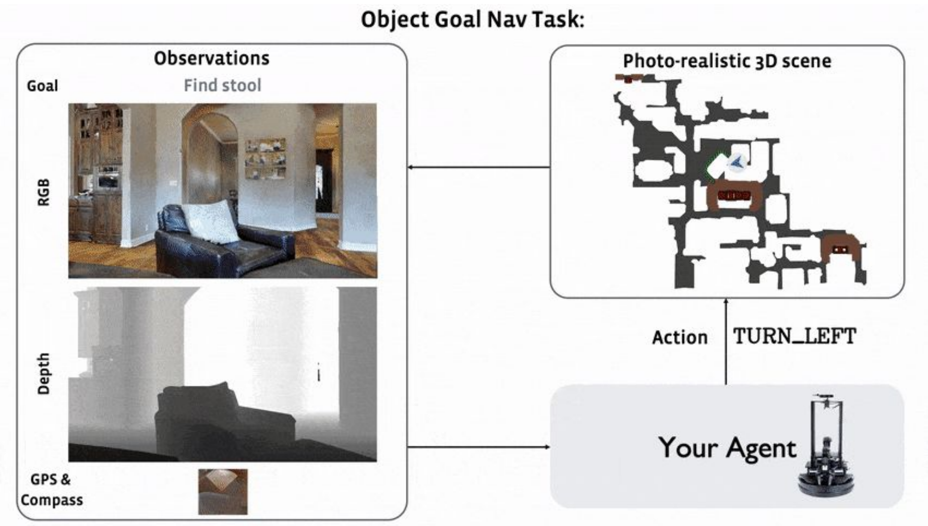
Linfeng Zhao*, Hongyu Li*
Taskin Padir, Huaizu Jiang†, Lawson L.S Wong†

Tackling Point/Object Goal Navigation

We focus on long-horizon path planning in unstructured world

The robot finds a goal specified by:

1) a point on the map ("point navigation")
2) an object name ("semantic navigation")



[Example: Habitat Challenge]

Path Planning for Robot Navigation

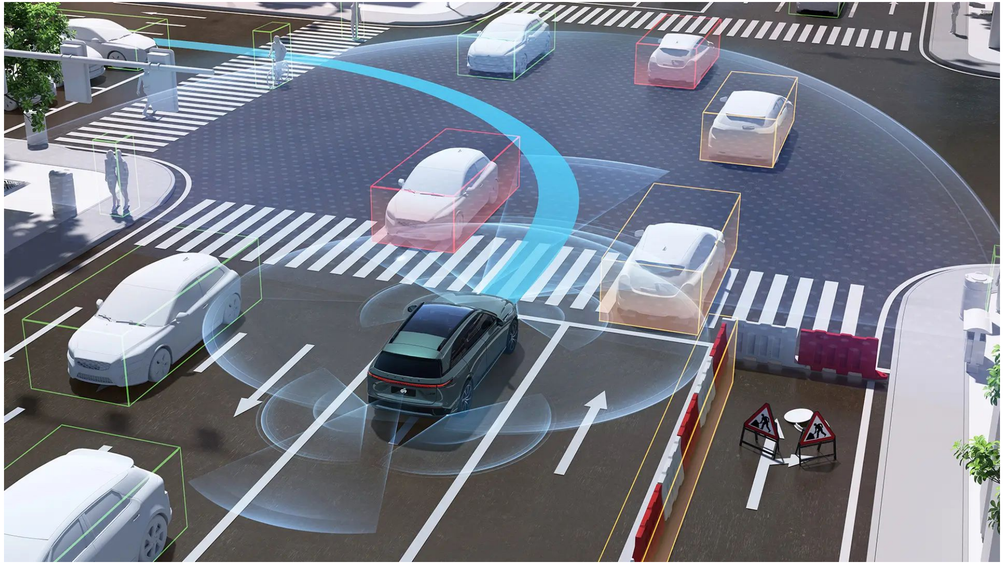
Classical path planning algorithms for robot navigation need full specification of the environment:

- The state is known and is represented as e.g., occupancy grids
- The transition dynamics is fully known
- Challenging to handle semantic features

Learning-based Planning

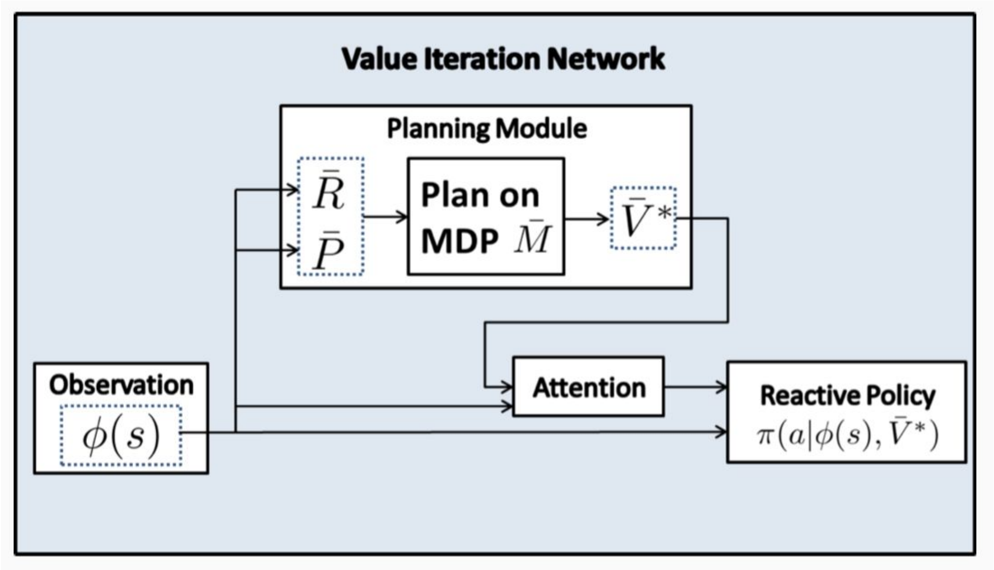
Map visual observations to actions and **learn** in an end-to-end manner.

Eliminate the need for **explicitly** constructing complete **intermediate representations.**

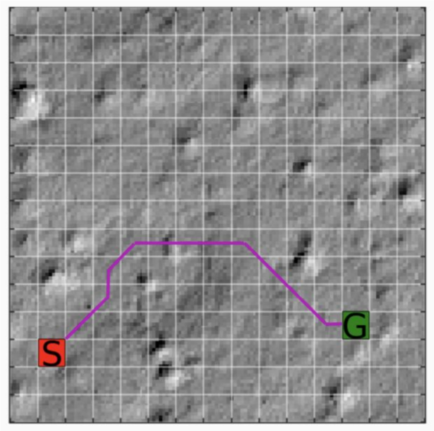


Background & Formulation

Value Iteration (Networks) for Path Planning

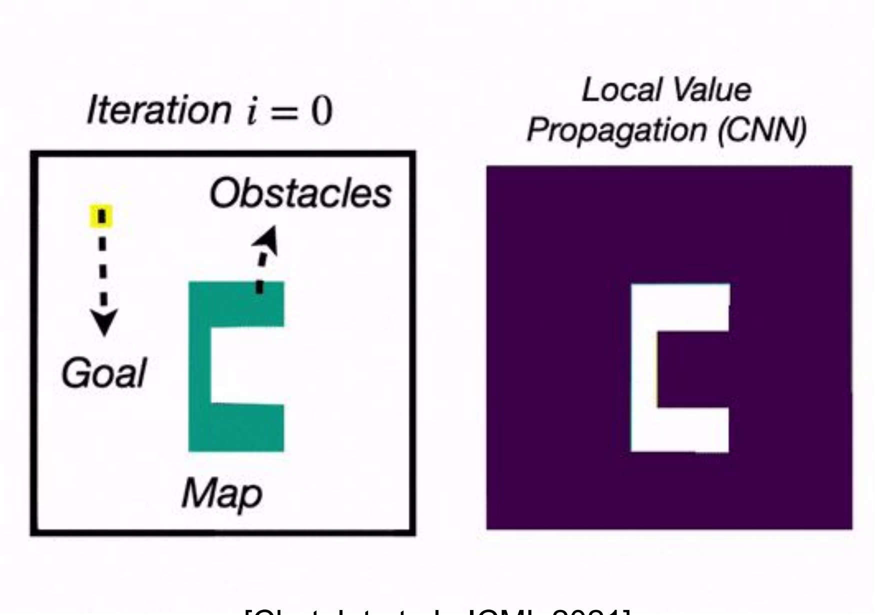


[Tamar et al. NIPS 2016]



A learning-based approach to perform value iteration on raw images.

Learned Value Iteration Network on 2D Grid



[Chatplot et al., ICML 2021]

Value Iteration

$$Q(s,a) = R(s,a) + \sum_a p(s' \mid s,a) V(s,a)$$

Trained VIN propagates values backward in time to the entire grid.

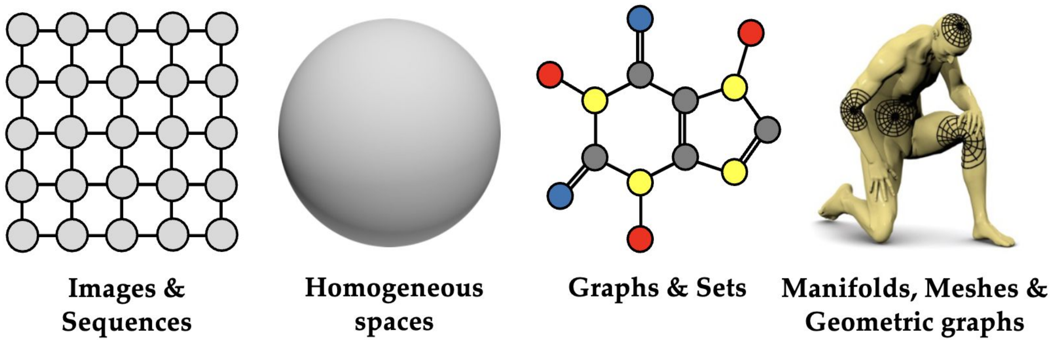
$$Q_a^{(k)} = R_a + \text{Conv2D}\left(V^{(k)}; W_a\right)$$

Leveraging Symmetry for Efficient Learning

Learning-based approaches need **intensive training data**.

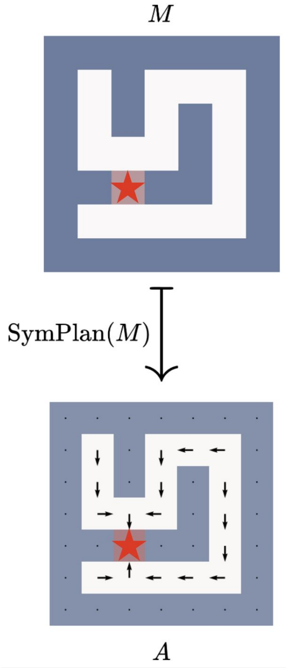
Robots navigating in physical world has **intrinsic symmetry**:

The robot moves on the 2D plane and has 2D translation/rotation equivariance, or E(2) symmetry.



[M. Bronstein et al., Geometric deep learning, arXiv]

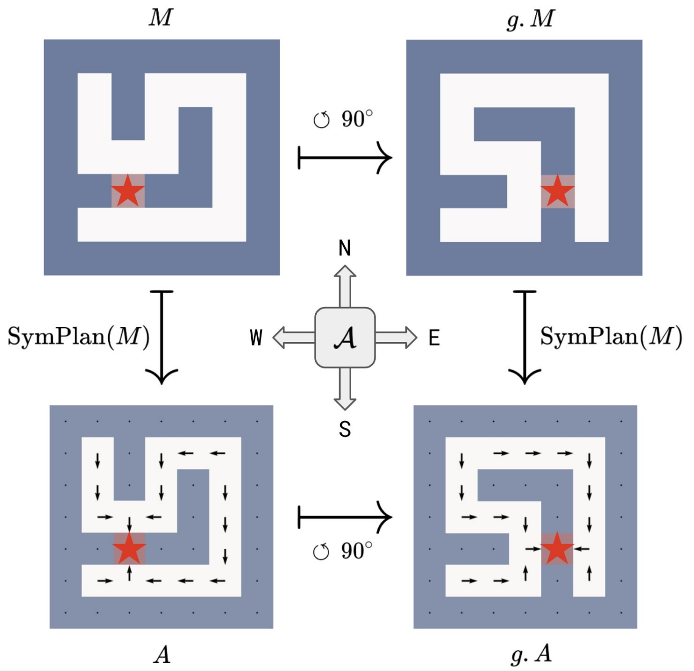
Symmetry in Path Planning



Zhao et al. (ICLR 2023) leverage symmetry in path planning on 2D grid.

[Zhao et al. ICLR 2023]

Symmetry in Path Planning



Zhao et al. (ICLR 2023) leverage symmetry in path planning on 2D grid.

$$\circlearrowleft 90^\circ \circ (\text{Plan}(M)) = \text{Plan}(\circlearrowleft 90^\circ \circ M)$$

[Zhao et al. ICLR 2023]

Extending From Planning on Grids to Graphs

Why do we move to graphs?

1. **Flexible state representation**: Nodes can represent continuous coordinates w/ features
2. **Continuous actions**: Graphs can be expanded to new 2D locations
3. **Continuous Euclidean symmetry**: Geometric graphs on 2D are SO(2)-transformable

They are less restrictive and better suited for unstructured environments.

Question: How to achieve path planning in unstructured environments, e.g., graphs with continuous Euclidean symmetry?

Approach: E(2)-Equivariant Graph Planning Network

Desiderata: A Learned Planner for Navigation

1) **Generalizable and long-horizon navigation**

We enable the model to learn to plan on *graphs* to generalize to different goals

2) **Data efficiency in learning**

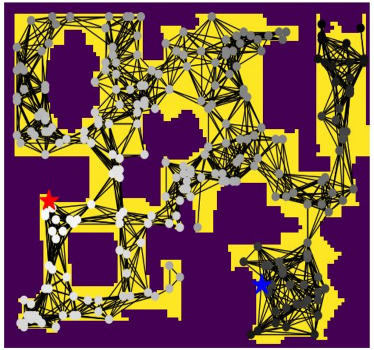
We inject symmetry into the planner model to learn from limited data

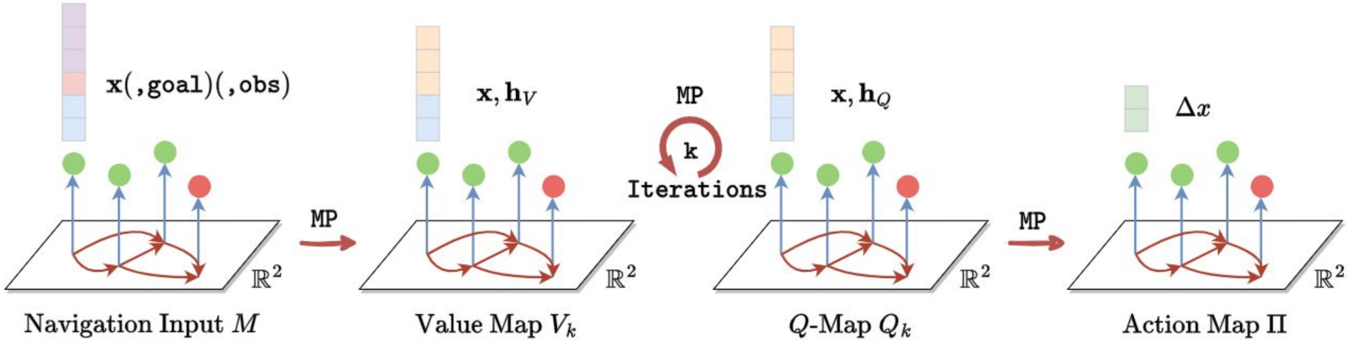
3) **Handling camera inputs**

Enable mapping perception from finite camera views by a learned lifting layer

For 1: Message Passing for Planning on Graphs

We extend Value Iteration Network on 2D grids (with CNN) to Value Iteration with message passing networks on **graphs**





For 1: Message Passing for Planning on Graphs

We develop the message passing version with following steps:

- Value iteration is written by integral over next states
- The integral is over the 2D plane, which can be written as a convolutional kernel
- We construct a graph by sampling finite points

$$Q_t(\boldsymbol{s}, \boldsymbol{a}) := R(\boldsymbol{s}, \boldsymbol{a}) + \int_{\mathbb{R}^2} d\boldsymbol{s}' P(\boldsymbol{s}' \mid \boldsymbol{s}, \boldsymbol{a}) V(\boldsymbol{s}'),$$

$$V_{t+1}(\boldsymbol{s}) = \max_{\boldsymbol{a}} Q_t(\boldsymbol{s}, \boldsymbol{a}),$$

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

For 2: Enable E(2)-Equivariance in Message Passing

Vanilla Message Passing Networks don't support equivariance

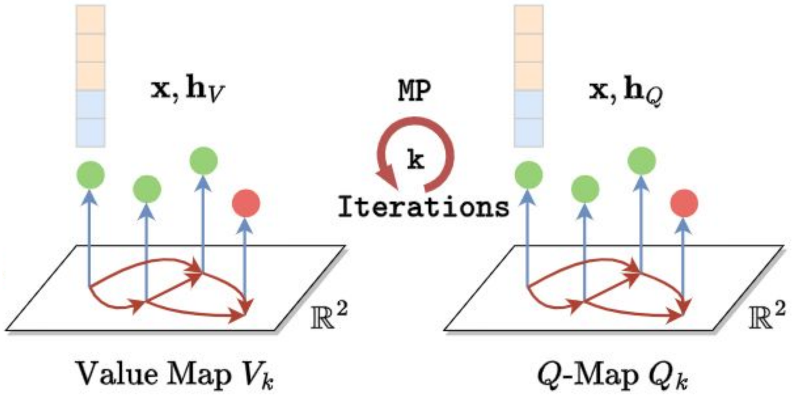
$$\boldsymbol{m}_{ij} = \texttt{propagate}_\theta \left(\boldsymbol{h}_i, \boldsymbol{h}_j, \boldsymbol{x}_i, \boldsymbol{x}_j\right),$$

$$\boldsymbol{h}_i' = \texttt{update}_\theta \left(\boldsymbol{h}_i, \sum_{j \in \mathcal{N}(i)} \boldsymbol{m}_{ij}\right).$$

The Value Iteration operation on graph can be equivariant through enforcing equivariance constraints

$$g \cdot \mathrm{MP}_\theta(V) = \mathrm{MP}_\theta(g \cdot V)$$

For 2: Enable E(2)-Equivariance in Message Passing

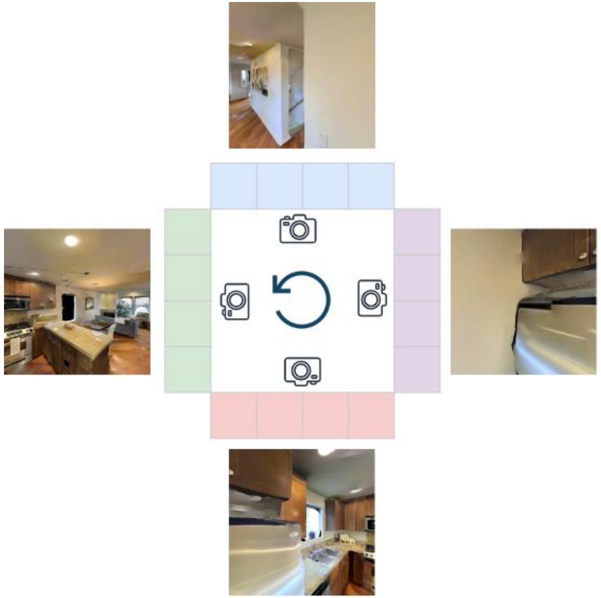


The Value Iteration operation on graph with E(2)-equivariance: *Continuous Rotations/Translations*

$$\circlearrowleft 90^\circ \circ (\mathrm{MP}(V_k)) = \circlearrowleft 90^\circ \circ Q_k$$

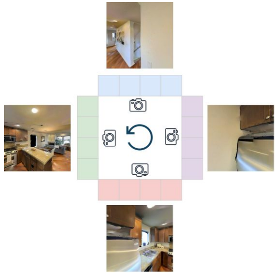
For 3: Finite camera views

For 3: Lifting From Finite Cameras to Infinite Symmetry



One challenge in learning graph planner from *finite camera views*:

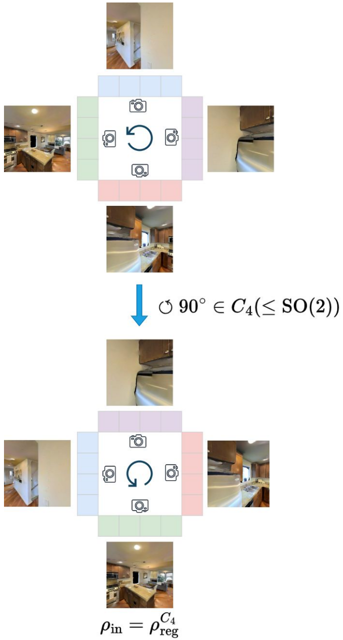
For 3: Lifting From Finite Cameras to Infinite Symmetry



One challenge in learning graph planner from *finite camera views*:

The robot's camera views can't be continuously rotated with robot

For 3: Lifting From Finite Cameras to Infinite Symmetry



One challenge in learning graph planner from *finite camera views*:

The robot's camera views can't be continuously rotated with robot

For 3: Lifting From Finite Cameras to Infinite Symmetry

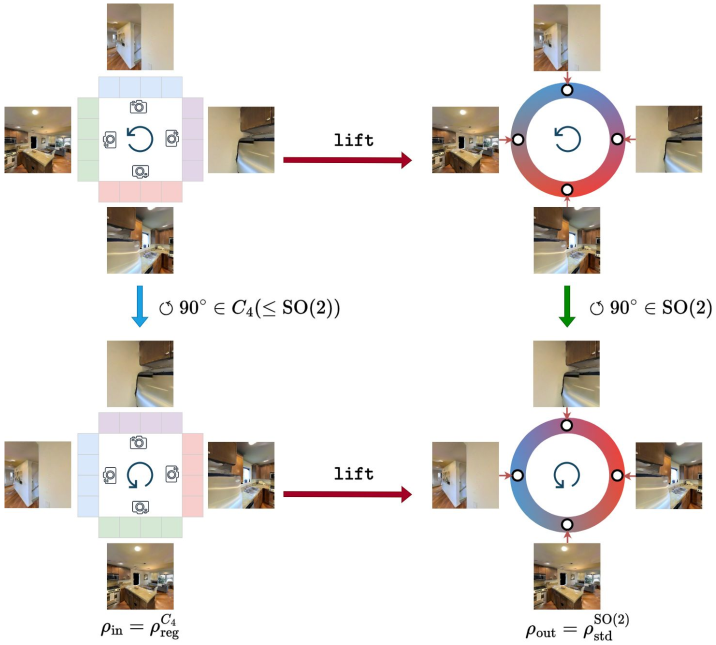
One challenge in learning graph planner from *finite camera views*:

The robot's camera views can't be continuously rotated with robot

How to obtain SO(2)-transformable feature maps?

We proposed an equivariant lifting layer

For 3: Lifting From Finite Cameras to Infinite Symmetry



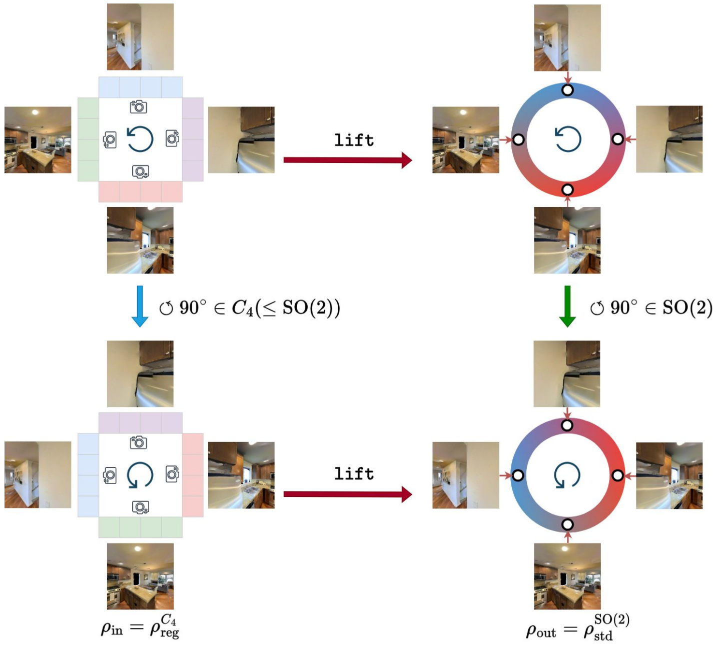
One challenge in learning graph planner from *finite camera views*:

The robot's camera views can't be continuously rotated with robot

How to obtain SO(2)-transformable feature maps?

We proposed an equivariant lifting layer

For 3: Lifting From Finite Cameras to Infinite Symmetry



One challenge in learning graph planner from *finite camera views*:

The robot's camera views can't be continuously rotated with robot

How to obtain SO(2)-transformable feature maps?

We proposed an equivariant lifting layer

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

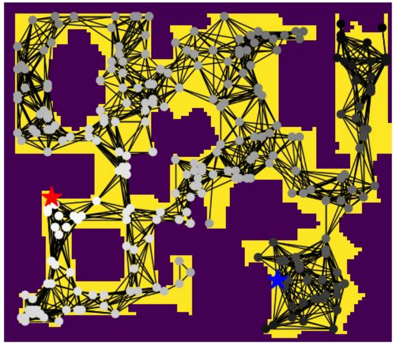
Experiments

Experiment Setup

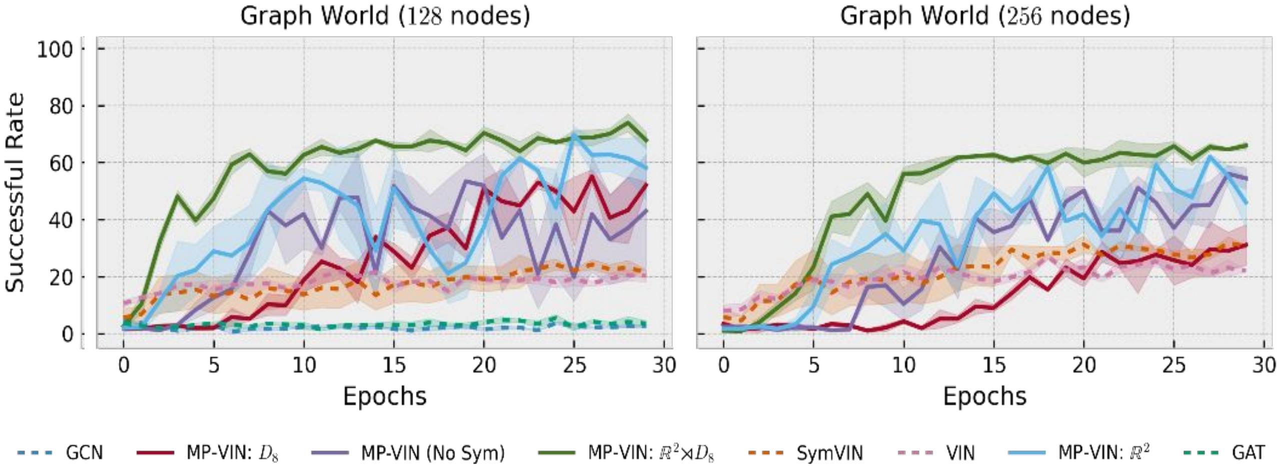
Example:
On Habitat semantic navigation
Every graph node on the 2D plane has input of
4-directional views

The node features in the graph can represent either **occupancy** or **learned semantic features** (from perception output), which are used to predict optimal actions.

Results: Planning on Graph World



(Randomly Sampled Graph)

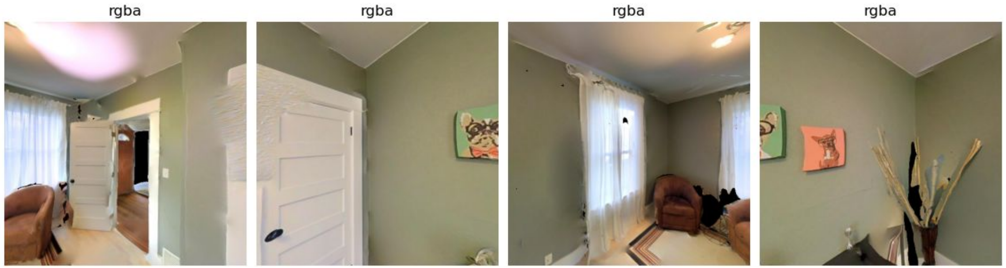


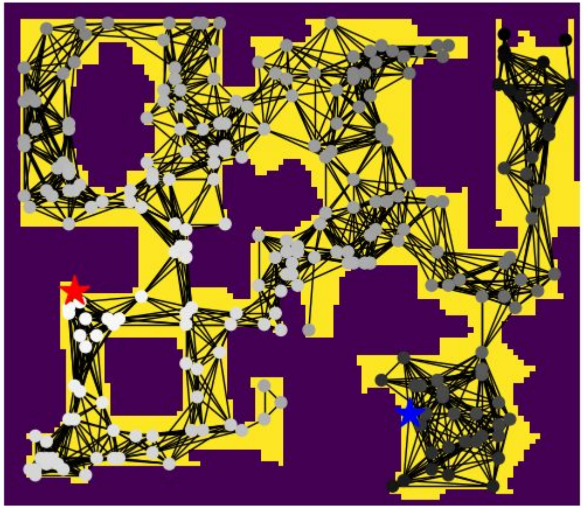
Full equivariance (**MP-VIN with *translation+rotation equivariance***) improves over **graph** planners **without symmetry**, **only rotation equivariance** and **only translation equivariance**

Setup: Semantic Navigation, Habitat

We experiment on a more realistic setting:

- Habitat with HM3D Dataset
- Randomly sample nodes and connect k-NN
- Egocentric RGB cameras facing four directions
- Find e.g., the “refrigerator” object





Results: Semantic Navigation

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

Compared with **MP-VIN** (**without symmetry**), full equivariance (**MP-VIN with *translation+rotation equivariance***) has higher successful rate.

Summary

The proposed approach (MP-VIN)

- Utilizes **symmetry** (translation + rotation) for more **efficient** robot navigation learning.
- Requires **fewer training samples** and achieves **higher performance** with **smoother** learning curves and **faster** convergence.
- Excels in real-world-like environments, demonstrating strong **generalization** to new, unstructured environments in navigation tasks.

Thank you!

Website Link and Contact:

Linfeng Zhao* (zhao.linf@northeastern.edu)

Hongyu Li (hongyu@brown.edu)

Linfeng Zhao is last-year PhD on market for postdocs and industry research jobs.

Outline v1

- Introduction + Motivation [2min]
    - Linfeng
- Background+Formulation: differentiable planning, from grids to graphs [5min]
    - Linfeng
- Method: E(2)-equivariant graph planning network [2-3min]
    - Linfeng+Hongyu
- Experiments: graph planning for navigation [2-3min]
    - Hongyu
- Conclusions / Summary [1min]
    - Hongyu+Linfeng
- Q&A [3min]

Outline v2

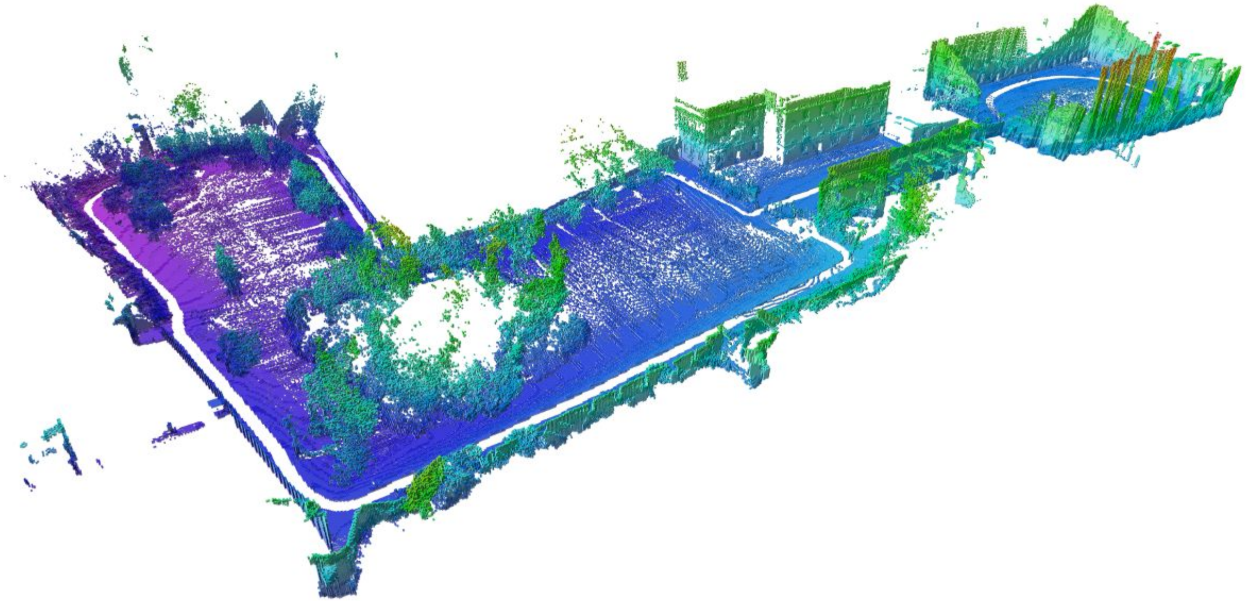
- Introduction + Motivation **[1min] – cut!**
  - Linfeng
- Background+Formulation: differentiable planning, from grids to graphs **[5-6min] expand**
  - Linfeng
- Method: E(2)-equivariant graph planning network **[2-3min]**
  - Linfeng+Hongyu
- Experiments: graph planning for navigation **[2-3min] – cut!**
  - Hongyu
- Conclusions / Summary [1min]
  - Hongyu+Linfeng
- Q&A [3min]

Motivation

Navigation in Complex Environments

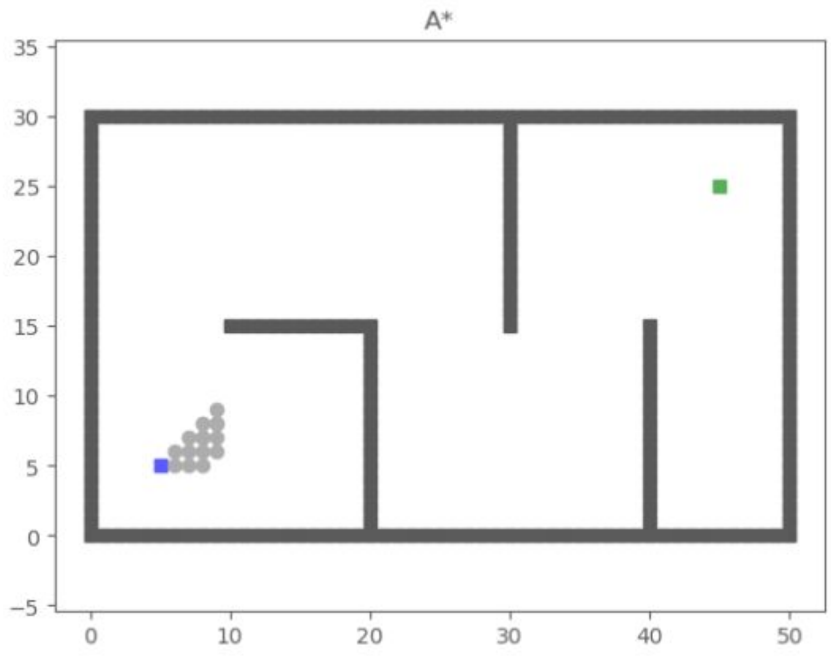
The path planning algorithms (A*, RRT, …) produce exact solutions for navigation

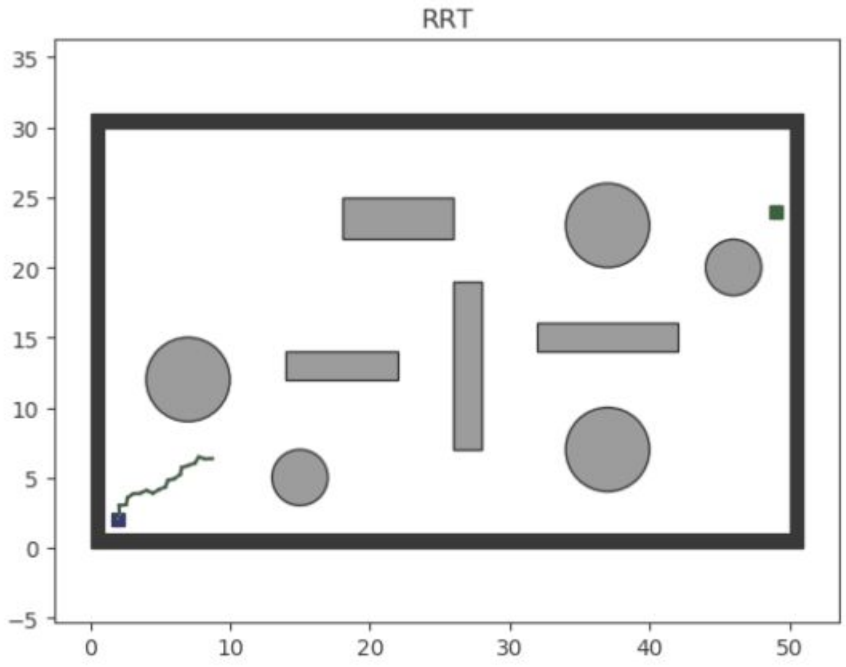
They need to first build the exact representation of the world, e.g., occupancy grids



[Octomap]

Classical Planning Algorithms are Great!

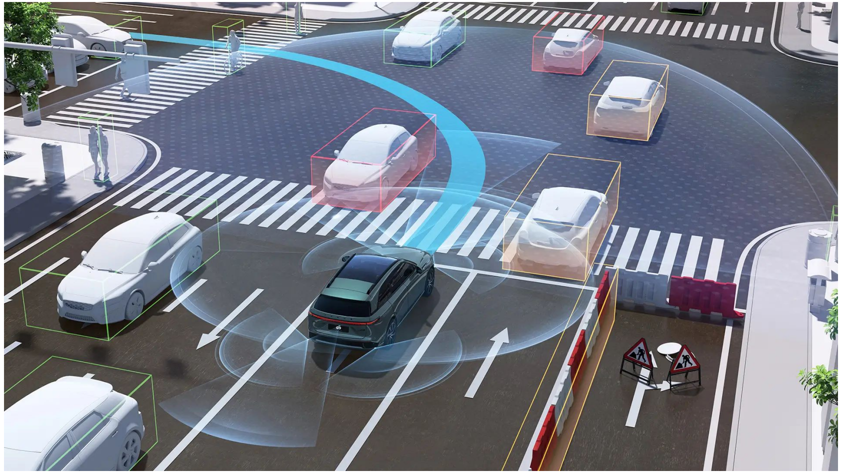




Challenges in Classical Planning

They rely on fully structured input, such as specific problem specification:

- The input is represented as known occupancy grids
- The transition dynamics is fully known



Benefits of Learning-based Planning

- Learning-based approaches **eliminate** the need for **explicit intermediate representations** by directly mapping observations (e.g., from cameras) to actions.
- This connection between representation learning and planning enables the system to produce actions directly, allowing for greater scalability in unstructured environments.
- **Advantage**: Reduces complexity by **avoiding the need to construct structured representations**, making it easier to handle dynamic, complex environments.
- **An example**: Tesla relied on the decoupled, engineered pipelines before "FSD" via fully end-to-end learning

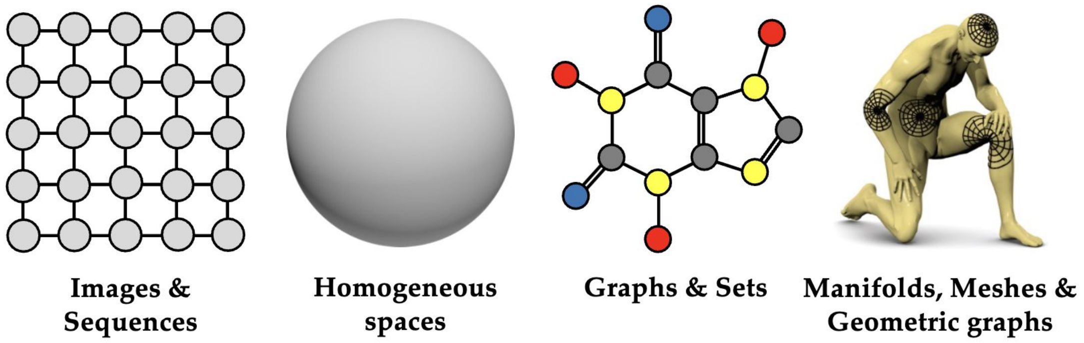
Challenges

- However: Short-horizon decision-making has been dominated by learning-based approaches
- It is still unclear that whether long-horizon tasks (such as navigation) that need long-horizon planning can be done by learning-based approaches.
- We study a specific type of planning (path planning) through learning-based solutions.

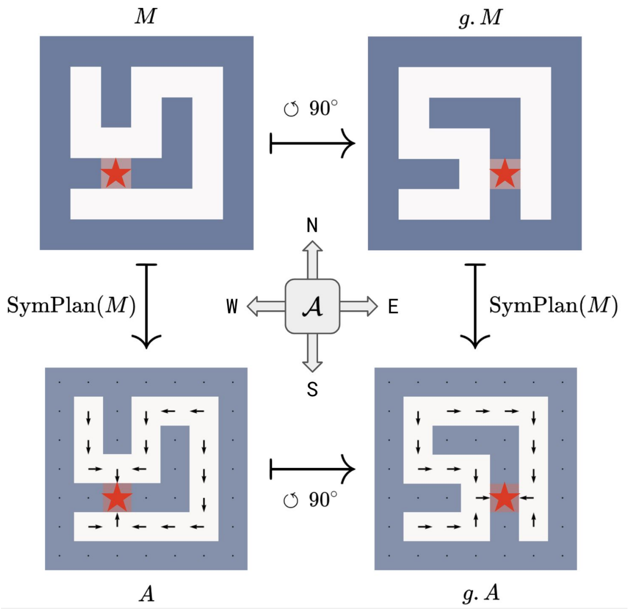
Equivariance in Learning for Planning

Path planning on 2D grids is steerable convolution and has discrete rotation symmetry

It can directly generalize to other geometric spaces



[M. Bronstein et al., Geometric deep learning, arXiv]



[Zhao et al. ICLR 2023]

$$Q_t(\boldsymbol{s}, \boldsymbol{a}) := R(\boldsymbol{s}, \boldsymbol{a}) + \int_{\mathbb{R}^2} d\boldsymbol{s}' P(\boldsymbol{s}' \mid \boldsymbol{s}, \boldsymbol{a}) V(\boldsymbol{s}'),$$

$$V_{t+1}(\boldsymbol{s}) = \max_{\boldsymbol{a}} Q_t(\boldsymbol{s}, \boldsymbol{a}),$$

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

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

Generalizing Graph Convolution to Message Passing

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

$$\boldsymbol{h}_i' = \texttt{update}_\theta\left(\boldsymbol{h}_i, \sum_{j \in \mathcal{N}(i)} \boldsymbol{m}_{ij}\right).$$

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

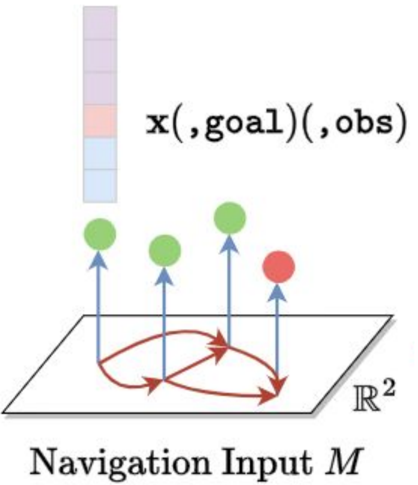
$$\boldsymbol{K}(gx) = \rho_{\text{out}}(g) \circ \boldsymbol{K}(x) \circ \rho_{\text{in}}(g)^{-1} \quad \forall g \in G, x \in \mathbb{R}^2,$$

How to Transform input graph and output action

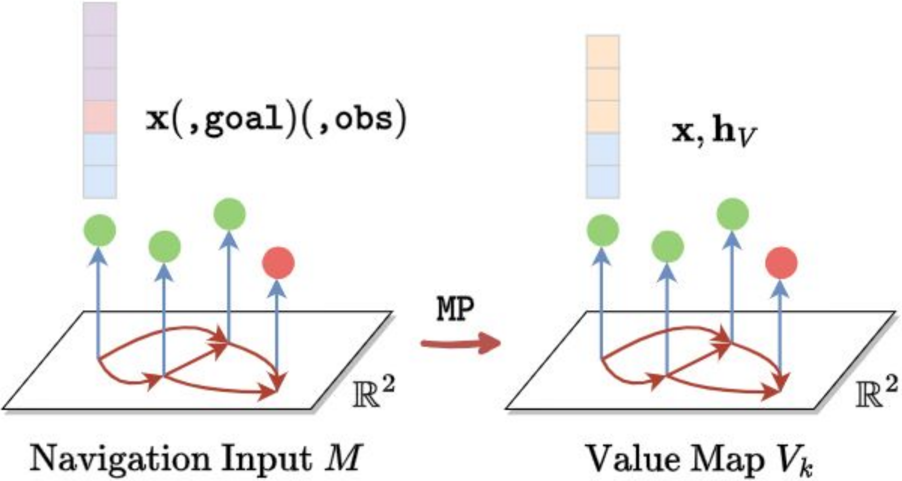
TODO

Message Passing with *Translation* Equivariance

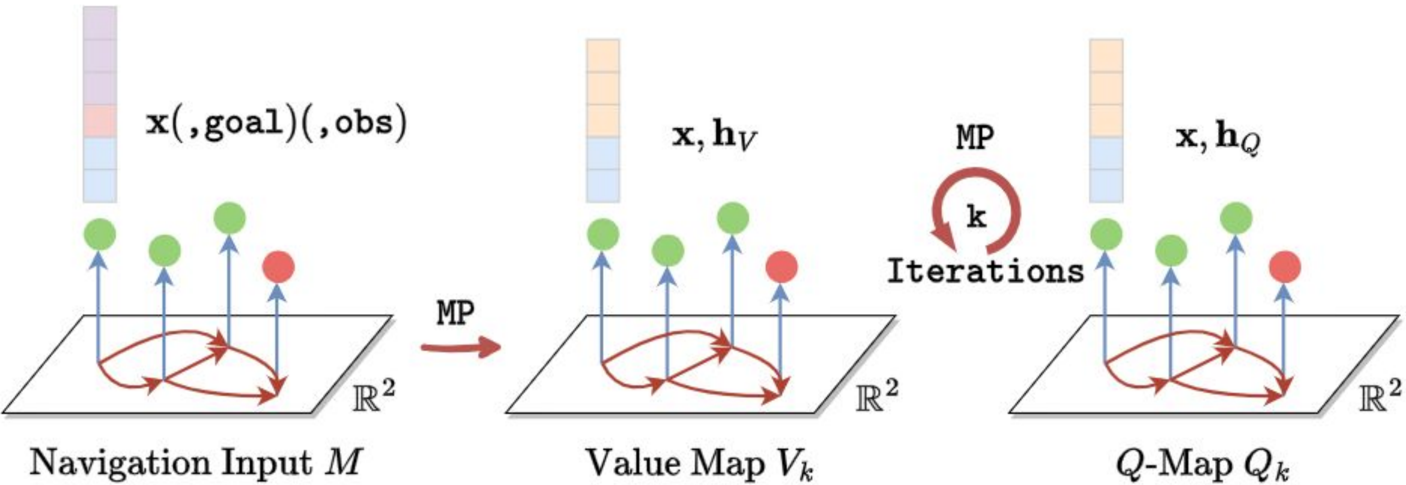
A Message Passing Network for Planning on Graphs



A Message Passing Network for Planning on Graphs



A Message Passing Network for Planning on Graphs



Convolution on Graphs with *Rotation Equivariance*

We generalize value iteration to graphs:

1. Value iteration involves integral over next states
2. The integral is over the 2D plane, which can be written as a convolutional kernel
3. The kernel can be proved to satisfy G-steerable kernel constraint
4. We construct a graph by sampling finite points

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

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

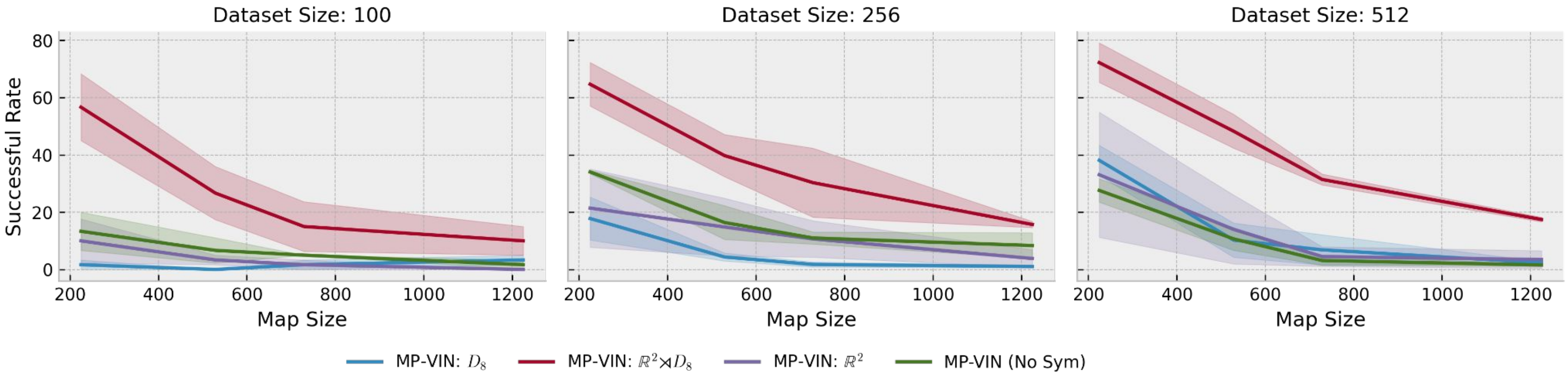
$$\boldsymbol{K}(gx) = \rho_{\text{out}}(g) \circ \boldsymbol{K}(x) \circ \rho_{\text{in}}(g)^{-1} \quad \forall g \in G, x \in \mathbb{R}^2,$$

Experiment Design

Here we aim to answer these questions:

- Q1, Does equivariance in the graph planner (Equivariant MP-VIN) improve performance?
- Q2, How is data efficiency and generalization ability of the trained planner?
- Q3, Can the proposed graph planner handle navigation in 3D visual environments that need a perception network head?

Data Efficiency & Generalization w.r.t. Map Size



A2: With full equivariance (**MP-VIN with *translation+rotation equivariance***), it generalizes better to larger maps with less training data, compared to **without symmetry**, **only rotation equivariance** and **translation equivariance**

IROS '24
ABU DHABI

Miniworld Experiment