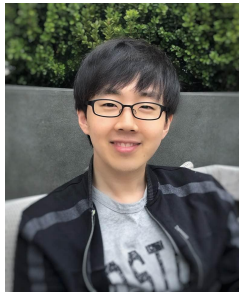
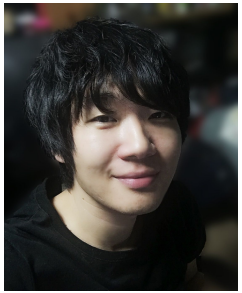
Successor Feature Landmarks for Long-Horizon Goal-Conditioned Reinforcement Learning

Christopher Hoang
University of Michigan

Sungryull Sohn
University of Michigan, LG AI Research



Jongwook Choi
University of Michigan

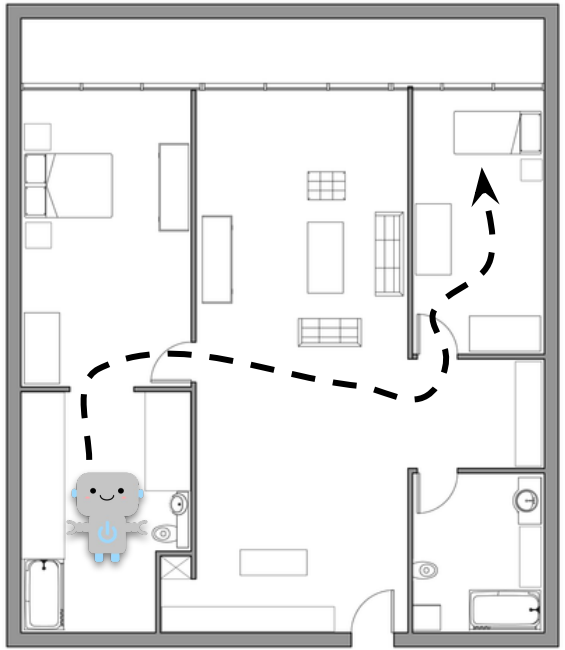


Wilka Carvalho
University of Michigan

Honglak Lee
University of Michigan, LG AI Research

Motivating scenario

Suppose we place a household robot in a *new* home



We would like the robot to be able to do the following

1. Explore the home and understand its layout
2. Reach *any* location (to perform a task) from *any* starting point

Scenario specifications

1. Task can be *long-horizon*, i.e. it can take the robot many steps to move from one room to another
2. Robot only has access to *first-person* images

Long-horizon GCRL

Goal-conditioned reinforcement learning (GCRL)

1. Markov Decision Process extended with set of goals (subset of state space)
2. Goal-conditioned reward function

Problem: find optimal goal-conditioned policy that maximizes expected cumulative *goal-conditioned* reward

Long-horizon: goals can be distant from agent's starting state, requiring policy to operate over longer temporal sequences

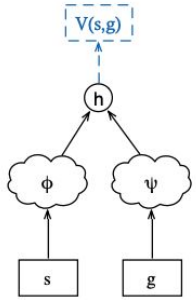
Recent work: GCRL

Universal value function approximator (UVFA; Schaul et al., 2015): value function approximator that generalizes over goal space

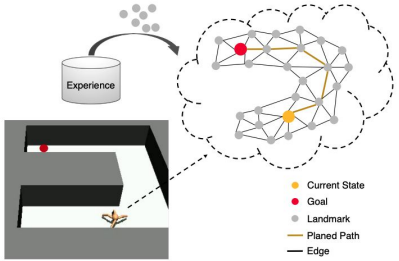
UVFAs have difficulty scaling to long-horizon tasks

Solution: augment UVFAs with planning mechanisms

1. MSS (Huang et al., 2019) and SoRB (Eysenbach et al., 2019): use UVFA as distance metric to build graph of landmarks to conduct planning over
2. LEAP (Nasiriany et al., 2019) - use UVFA to decompose long-horizon tasks as series of reachable subgoals



UVFA architecture with state *s* and goal *g* (Schaul et al., 2015)

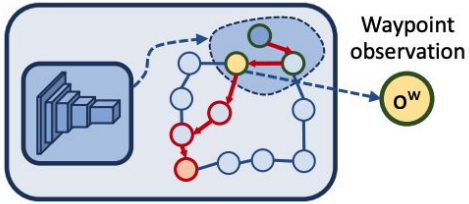


Planning on landmark graph in MSS framework (Huang et al., 2019)

Recent work: graph-based planning

Recent works in navigation have also proposed graph-based planning approaches

1. Conduct planning on high-level graph representations
2. Use low-level controller to move between nodes



Selecting next landmark node to navigate towards in SPTM (Savinov et al., 2018)

Papers

1. SPTM (Savinov et al., 2018): form graph by using reachability network as distance metric
   a. Relies on human demonstrations
2. SGM (Laskin et al., 2020): form sparse graph by merging similar observations
   a. Assumes exploration mechanism

Background: successor features (SF)

Successor representation (SR) of ***(s, a, s')***: expected discounted future occupancy of state ***s'*** starting in state ***s*** and action ***a***, acting under policy π (Dayan, 1993)

$$M_\pi(s, a, s') = \mathbb{E}^\pi \left[\sum_{t'=t}^{\infty} \gamma^{t'-t} \mathbb{I}(S_{t'} = s') \middle| \begin{matrix} S_t = s, \\ A_t = a \end{matrix} \right]$$

SF extend SR to function approximation case where ***s'*** is replaced by feature vector ϕ (Barreto et al., 2017)

SF captures the dynamics of an environment

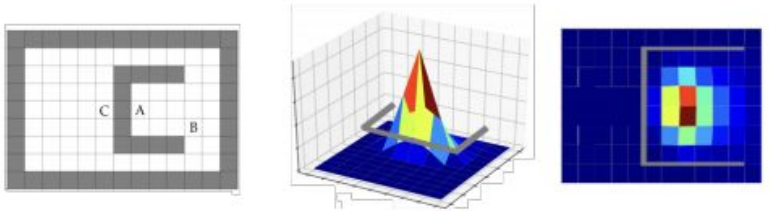


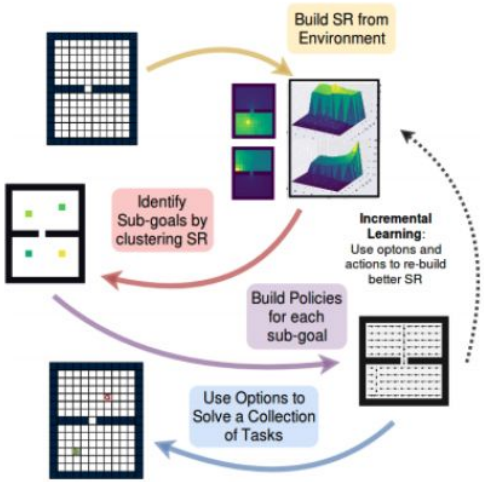
Figure 1: Successor representation, with respect to the uniform random policy, of state A (left). This example is similar to Dayan's (1993). The red color represents larger values while the blue color represents smaller values (states that are temporally further away).

Figure 1 (Machado et al., 2018)

Recent work: SF

Successor Options (Ramesh et al., 2019)

1. Use clustering in SR space to identify subgoals
2. Learn options to subgoals via SR-based option reward function
3. Only qualitative results in high-dimensional state spaces



Successor Options framework (Ramesh et al., 2019)

Idea: leverage SF for graph-based planning

SF: state representation that captures temporally-extended environment dynamics

1. Reflects long-horizon setting
2. Suitable for formulating distance metric
3. Learned via self-supervised training signal

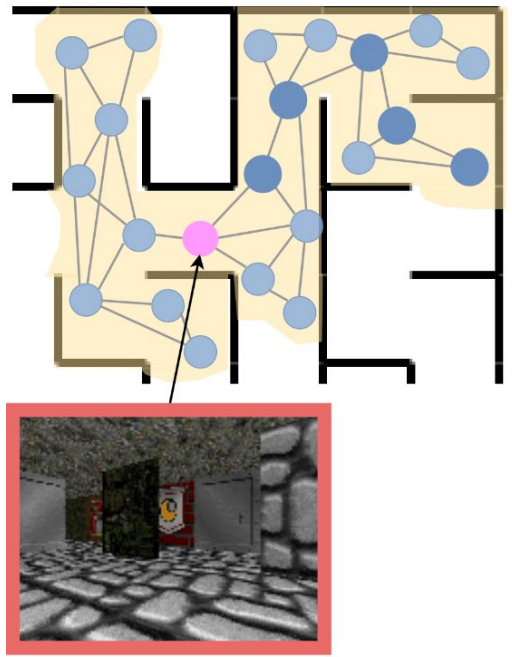
SF enables transfer between goals (Barreto et al., 2016)

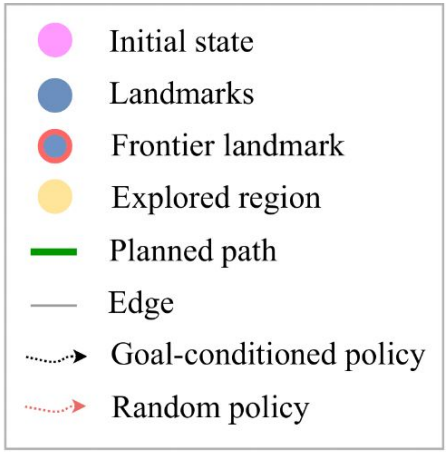
1. Decouples environment dynamics (SF) from task in Q-value formulation

$$r(s, a, s') = \boldsymbol{\phi}(s, a, s')^\top \mathbf{w}$$

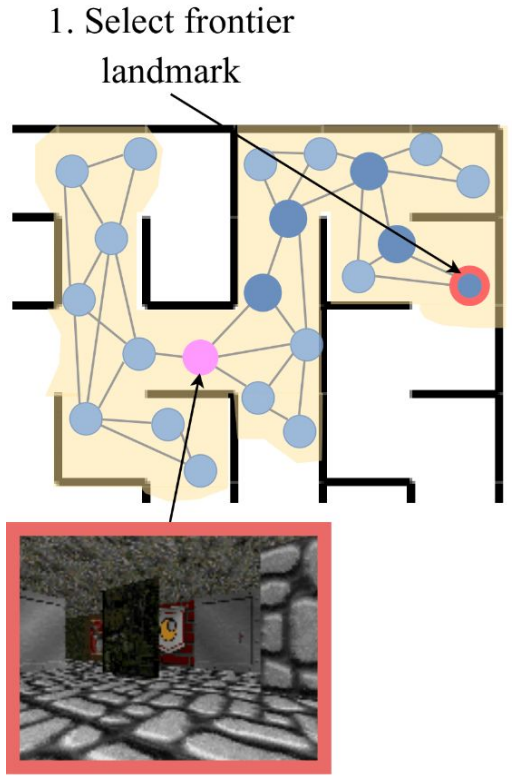
$$\begin{aligned} Q^\pi(s, a) &= \mathrm{E}^\pi \left[r_{t+1} + \gamma r_{t+2} + \ldots \,|\, S_t = s, A_t = a \right] \\ &= \mathrm{E}^\pi \left[\boldsymbol{\phi}_{t+1}^\top \mathbf{w} + \gamma \boldsymbol{\phi}_{t+2}^\top \mathbf{w} + \ldots \,|\, S_t = s, A_t = a \right] \\ &= \mathrm{E}^\pi \left[\textstyle\sum_{i=t}^{\infty} \gamma^{i-t} \boldsymbol{\phi}_{i+1} \,|\, S_t = s, A_t = a \right]^\top \mathbf{w} = \boldsymbol{\psi}^\pi(s, a)^\top \mathbf{w} \end{aligned}$$

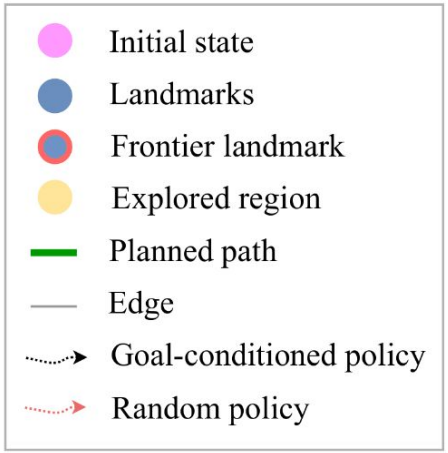
Overview of Successor Feature Landmarks (SFL)



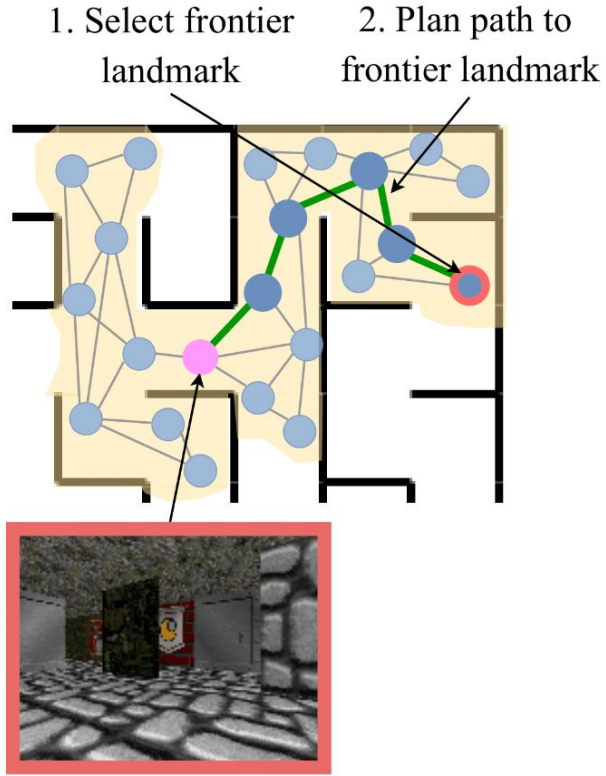


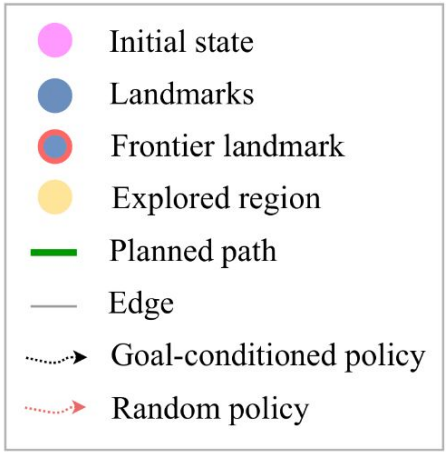
Overview of Successor Feature Landmarks (SFL)



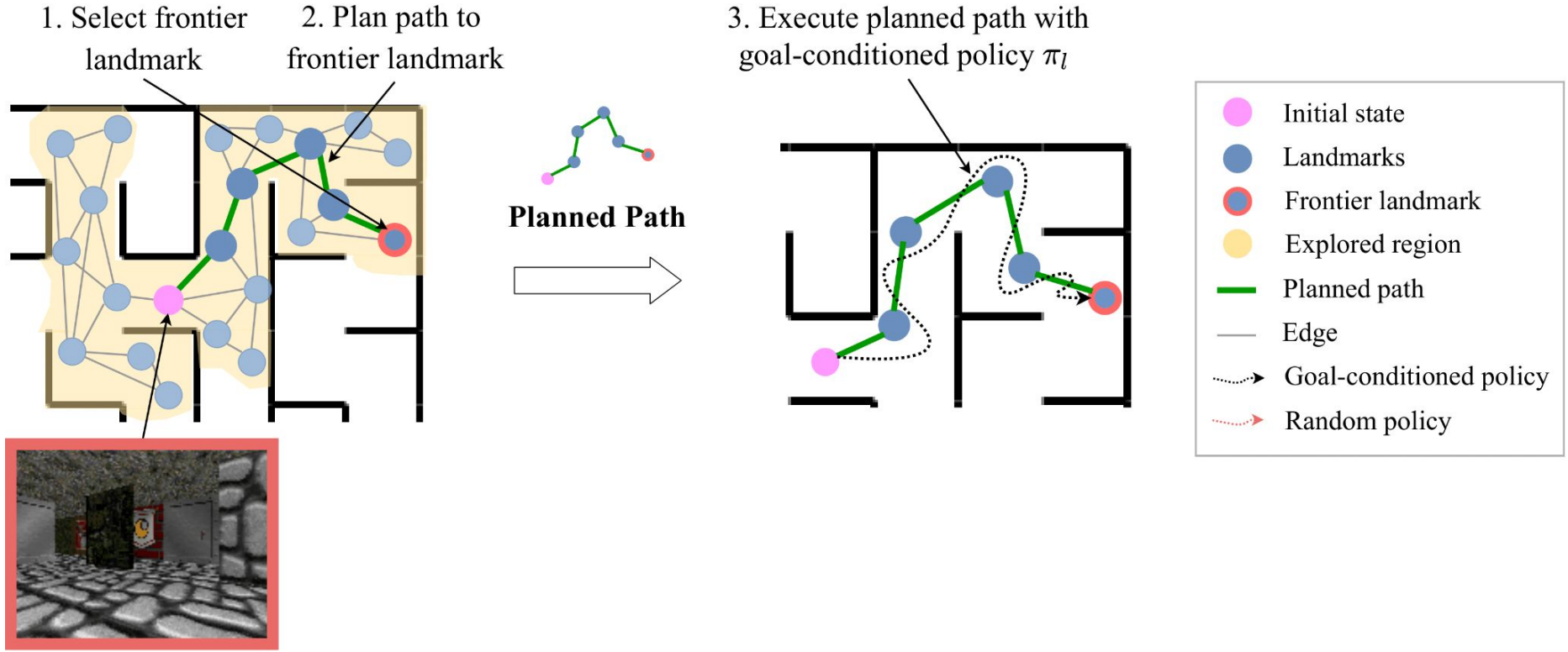


Overview of Successor Feature Landmarks (SFL)

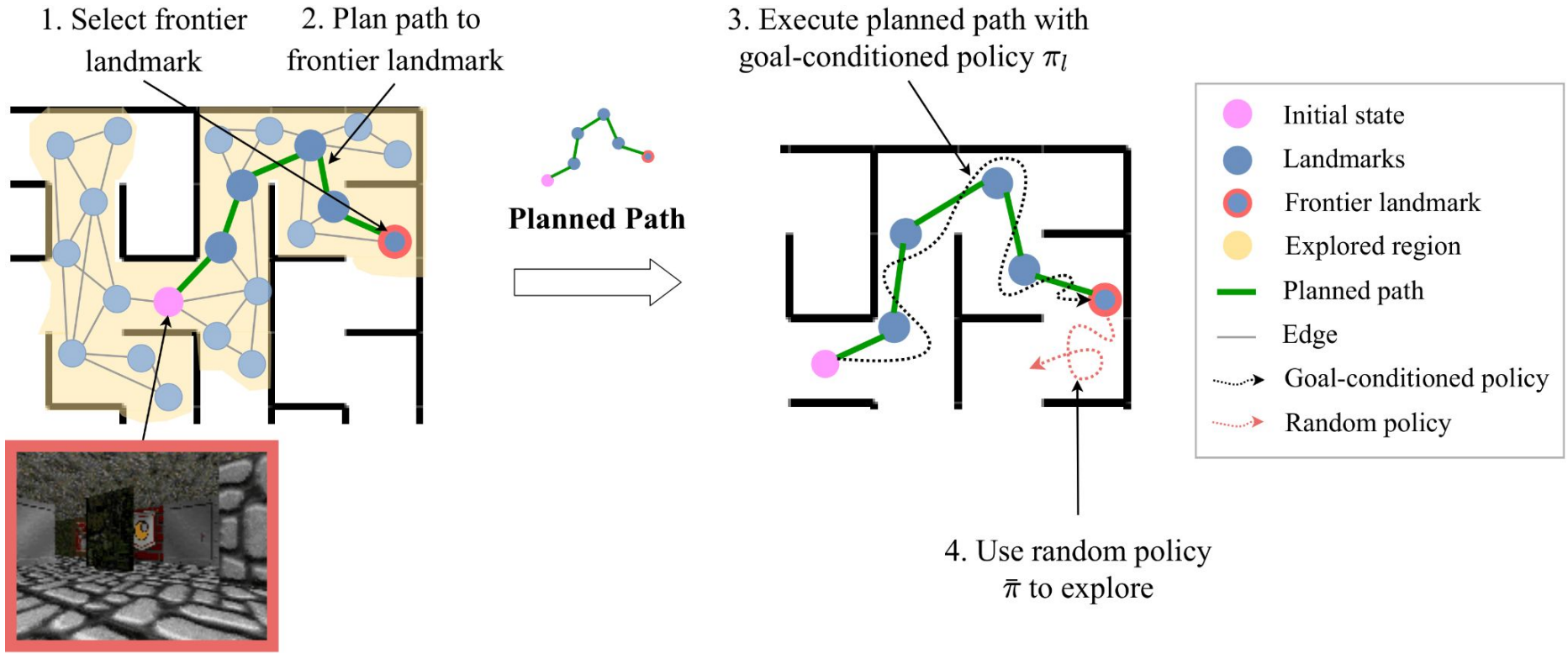




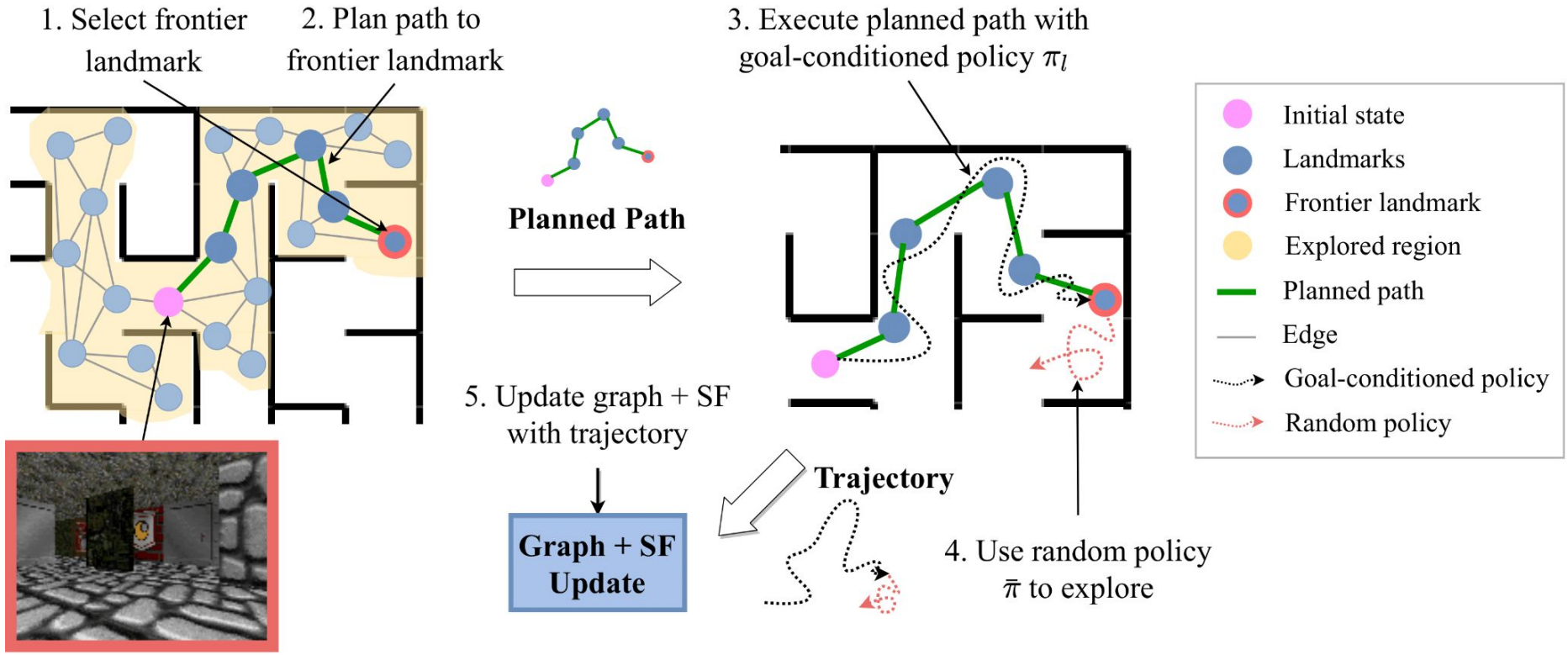
Overview of Successor Feature Landmarks (SFL)



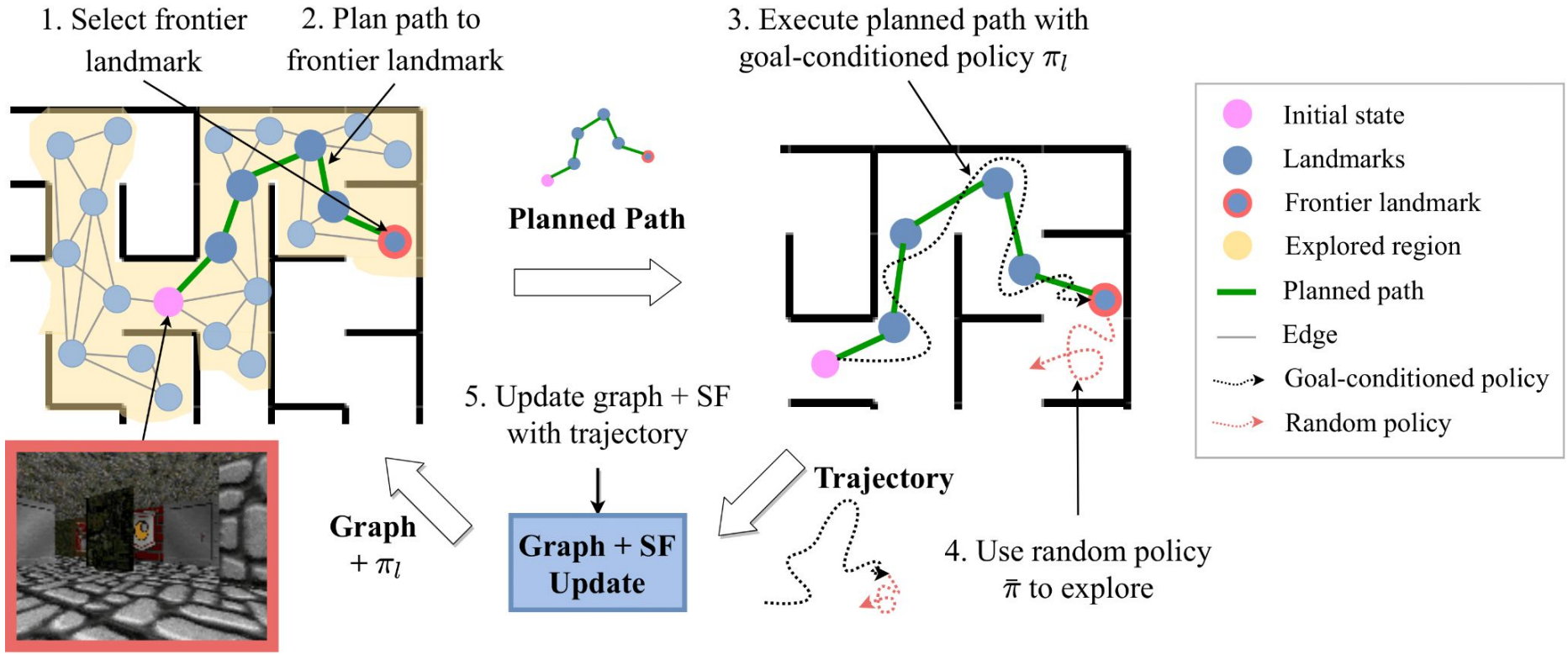
Overview of Successor Feature Landmarks (SFL)



Overview of Successor Feature Landmarks (SFL)



Overview of Successor Feature Landmarks (SFL)



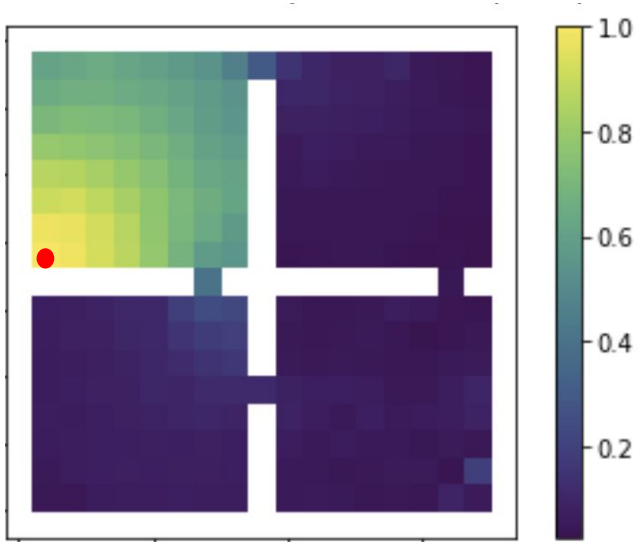
Successor Feature Similarity (SFS)

Defined as cosine similarity in SF space (since $||\psi|| = 1$)

$$\mathrm{SFS}^{\pi}((s_1, a_1), (s_2, a_2)) = \psi^{\pi}(s_1, a_1)^{\top} \psi^{\pi}(s_1, a_2)$$

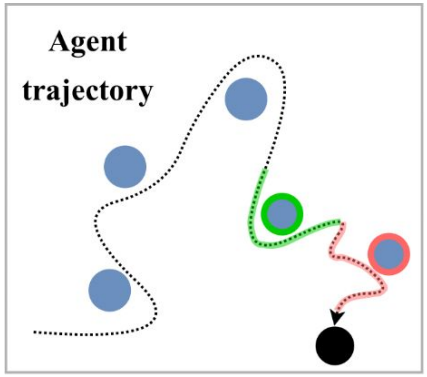
Uses

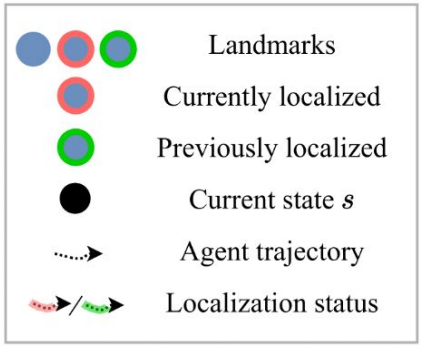
1. Localize agent to nearest landmark
2. Determine which states to add as landmarks
3. Obtain local goal-conditioned policy ***instantly***



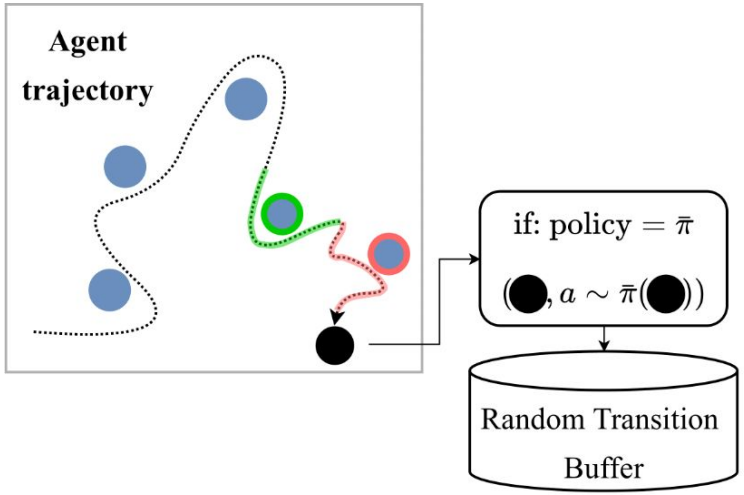
SFS relative to red dot in ***MiniGrid***

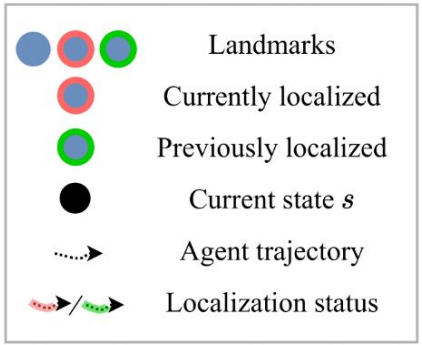
Learning SF


Agent
trajectory

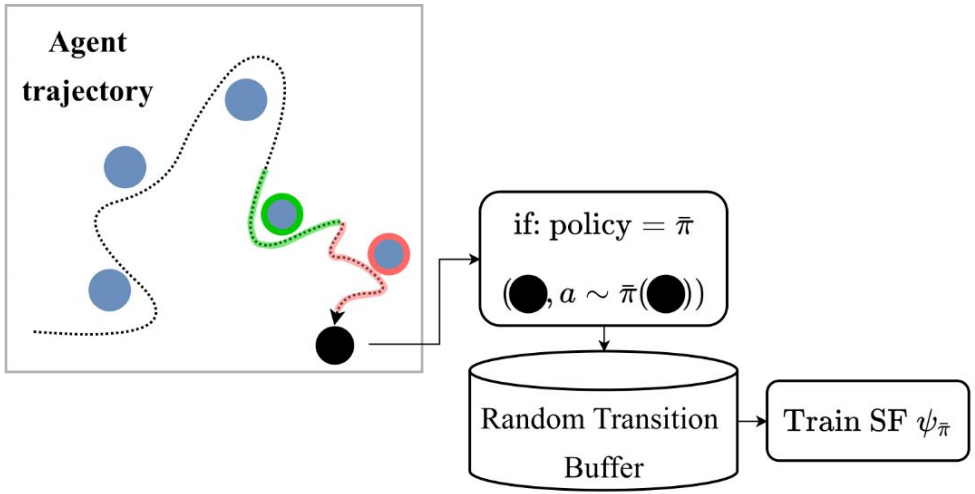

Landmarks
Currently localized
Previously localized
Current state s
Agent trajectory
Localization status

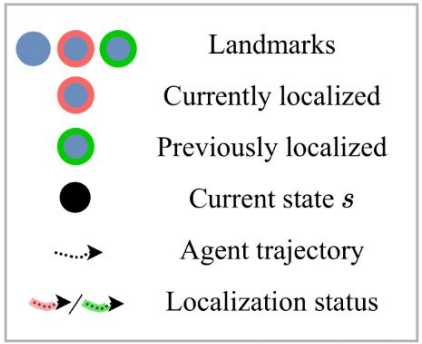
Learning SF


Agent
trajectory
if: policy $= \bar{\pi}$
$(\bullet, a \sim \bar{\pi}(\bullet))$
Random Transition
Buffer

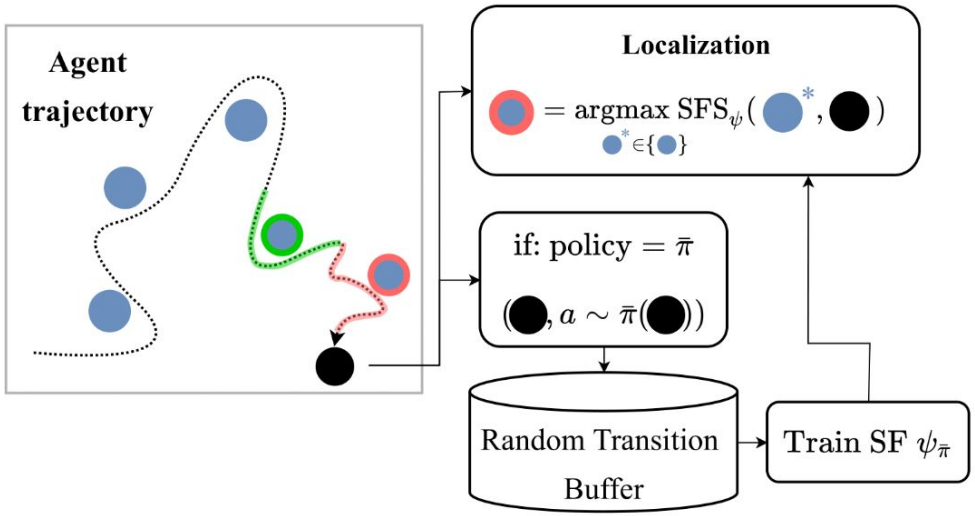

Landmarks
Currently localized
Previously localized
Current state s
Agent trajectory
Localization status

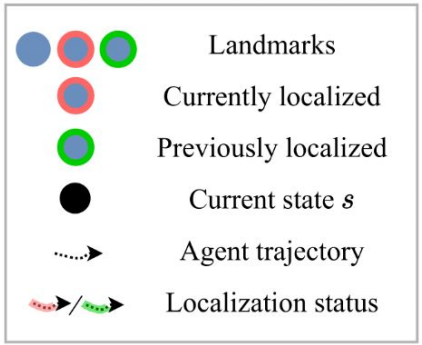
Learning SF


Agent
trajectory
if: policy $= \bar{\pi}$
$(\bullet, a \sim \bar{\pi}(\bullet))$
Random Transition
Buffer
Train SF $\psi_{\bar{\pi}}$

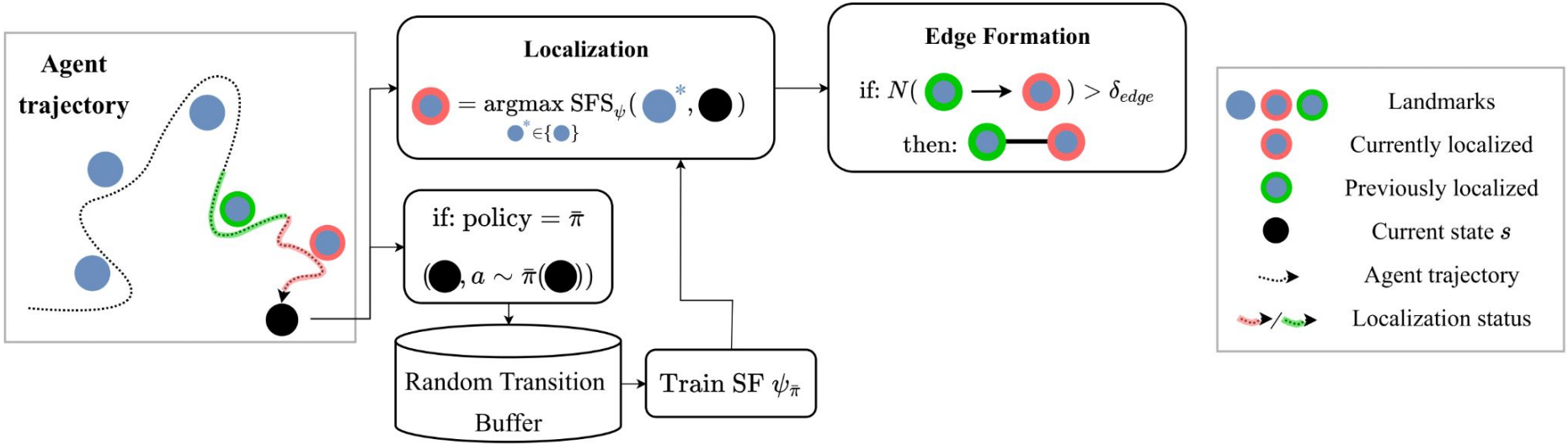

Landmarks
Currently localized
Previously localized
Current state s
Agent trajectory
Localization status

Building landmark graph

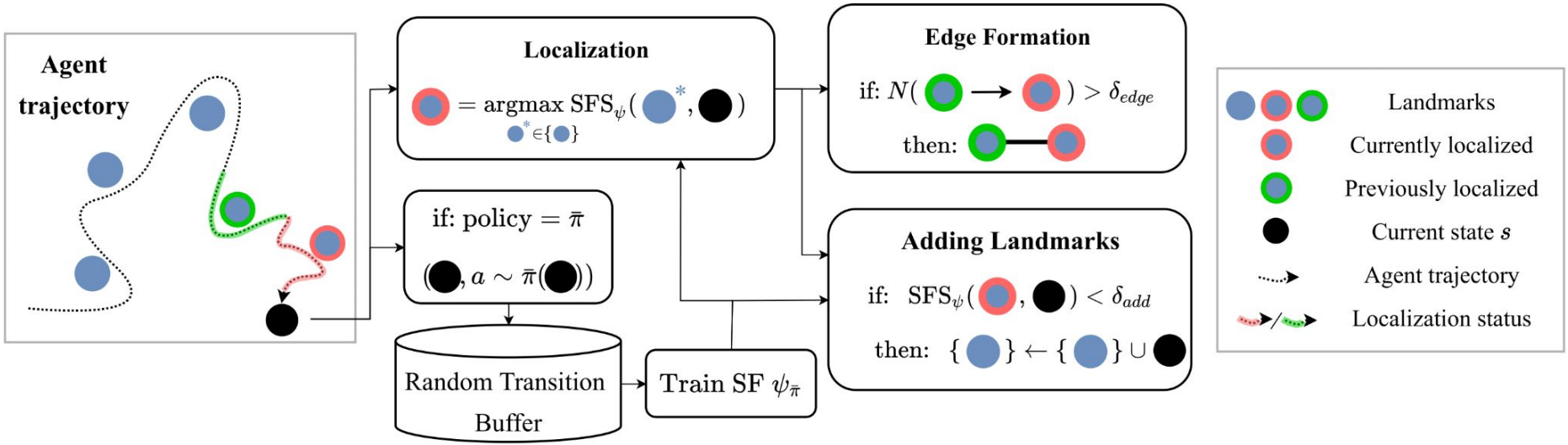




Building landmark graph



Building landmark graph



Local goal-conditioned policy

Goal-conditioned reward function based on features ϕ of current state and SF ψ of goal

$$r(s, a, g) = \phi(s, a)^\top \psi^{\bar{\pi}}(g)$$

Derived Q-value formulation *is equivalent to* SFS

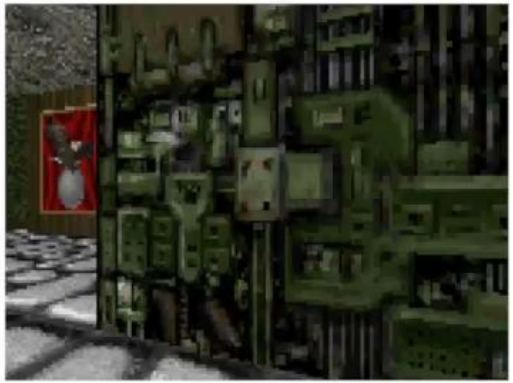
$$\begin{aligned} Q^{\bar{\pi}}(s, a) &= \psi^{\bar{\pi}}(s, a)^\top \psi^{\bar{\pi}}(g) \\ &= \mathrm{SFS}^{\bar{\pi}}(s, a, g). \end{aligned}$$

Policy given by taking action leading to highest SFS

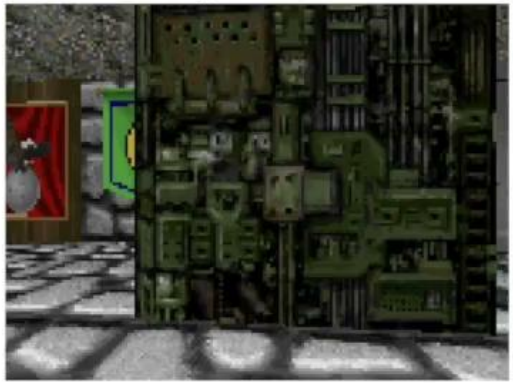
$$\pi = \operatorname*{argmax}_a \mathrm{SFS}^{\bar{\pi}}(s, a, g)$$

Planning

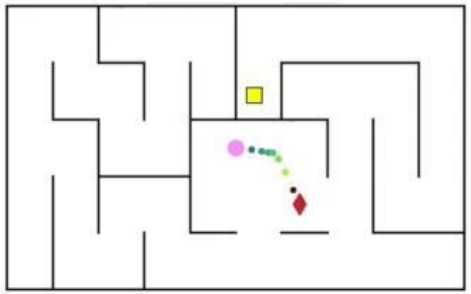
Current Observation



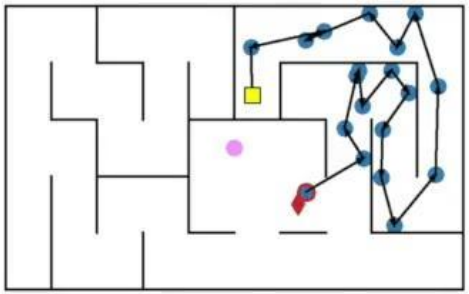
Target Landmark



Agent Trajectory



Landmark Path to Goal



Agent begins at **start** and tries to reach **goal** in **400** steps

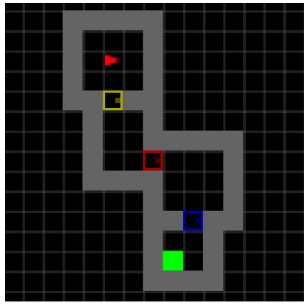
Experimental settings

Problem settings

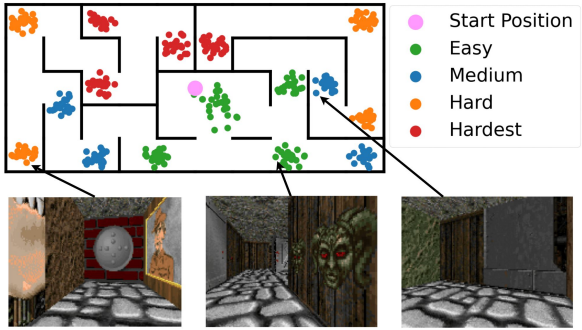
1. *Random spawn*
   a. In training, agent randomly spawned across map
   b. In evaluation, agent tested on random start-goal pairs
2. *Fixed spawn*
   a. In training, agent spawned at fixed starting point
   b. In evaluation, agent tested on random goals with same starting point as in training

Domains

1. ***MiniGrid***: 2D gridworld with top-down view observations
2. ***ViZDoom***: visual navigation environment with 3D first-person observations and large-scale mazes



Four-room MultiRoom ***MiniGrid*** with agent (red arrow) and goal (green square)



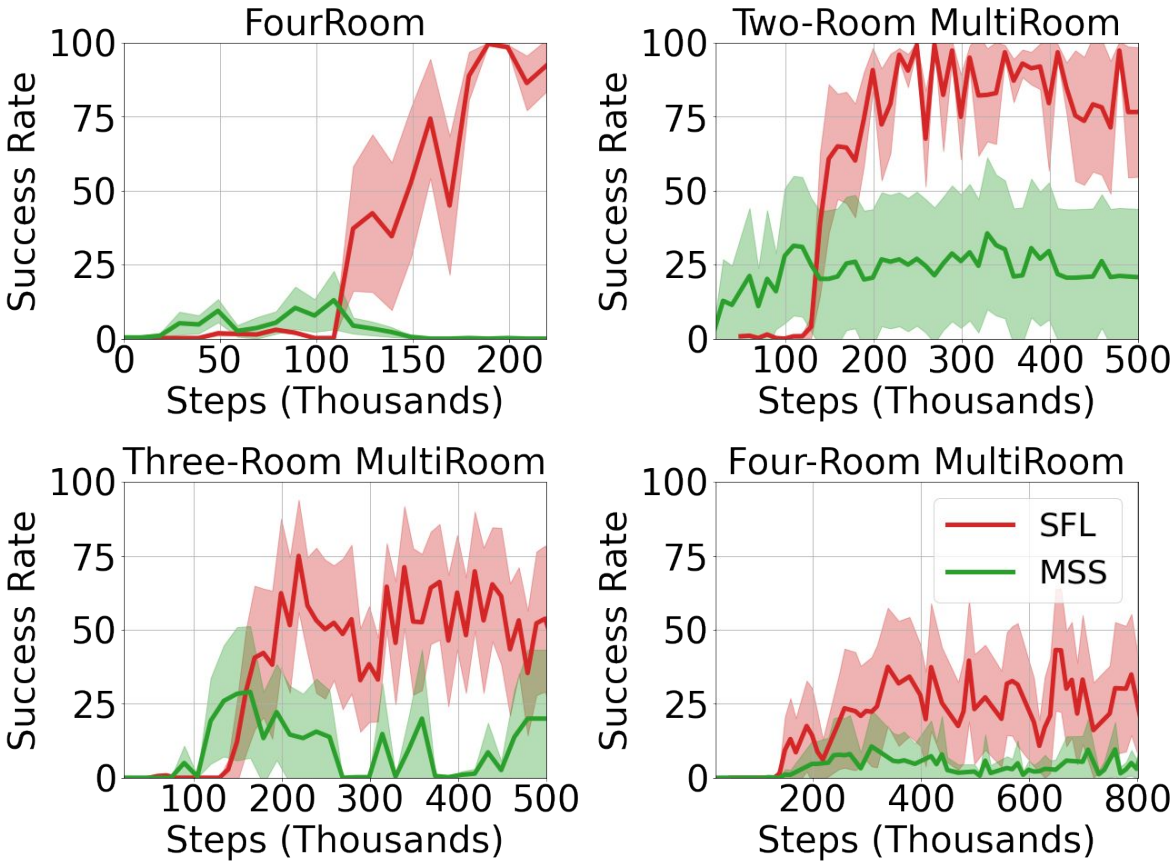
Top-down view of ***ViZDoom*** maze in *fixed spawn* with sampled goals

Baseline methods

Baseline	Local policy	Planning mechanism	Exploration mechanism	Assumptions
Visual controller	Inverse dynamics prediction	*N/A*	*N/A*	Requires random spawning
SPTM	^	Build graph with reachability network	*N/A*	Requires human demonstrations
SGM	^	^ + encourage sparsity in graph with two-way consistency objective	*N/A*	Requires random spawning
Episodic curiosity (EC)* + {SPTM, SGM}	^	^	Curiosity bonus based on reachability network	
MSS	UVFA	Build graph with UVFA distance metric	Select landmarks to be far apart from each other	

*- Savinov et al., 2018

Results on *MiniGrid*



Results on *ViZDoom*

Method	*SGM-Map*			*Test-2*			*Test-6*		
	Easy	Medium	Hard	Easy	Medium	Hard	Easy	Medium	Hard
Random Actions	58%	22%	12%	70%	39%	16%	80%	31%	18%
Visual Controller	75%	35%	19%	83%	51%	30%	89%	39%	20%
SPTM [27]	70%	34%	14%	78%	48%	18%	88%	40%	18%
SGM [15]	**92%**	64%	26%	**86%**	54%	32%	83%	43%	27%
SFL [Ours]	**92%**	**82%**	**67%**	82%	**66%**	**48%**	**92%**	**66%**	**60%**

Table 1: (*Random spawn*) The success rates of compared methods on three **ViZDoom** maps.

Method	*Test-1*				*Test-4*			
	Easy	Medium	Hard	Hardest	Easy	Medium	Hard	Hardest
MSS [10]	23%	9%	1%	1%	21%	7%	7%	7%
EC [28] + SPTM [27]	48%	16%	2%	0%	20%	10%	4%	0%
EC [28] + SGM [15]	43%	3%	0%	0%	28%	7%	4%	1%
SFL [Ours]	**85%**	**59%**	**62%**	**50%**	**66%**	**44%**	**27%**	**23%**

Table 2: (*Fixed spawn*) The success rates of compared methods on three **ViZDoom** maps.

Conclusion

Introduced Successor Feature Landmarks, a graph-based planning framework built *entirely* upon successor features

Experiments on MiniGrid and ViZDoom demonstrated that this method outperforms current graph-based approaches on long-horizon goal-reaching tasks

We hope that future work will explore more properties and uses of successor features