# Constructing an Optimal Behavior Basis for the Option Keyboard


UFRGS Universidade Federal do Rio Grande do Sul
University of Massachusetts 1863
Google DeepMind

Lucas N. Alegre, Ana L. C. Bazzan, André Barreto, Bruno C. da Silva


Open Review .net

## Setting

- Transfer in Reinforcement Learning (RL)
- **Idea:**
  1. Learn a compact set of policies (***behavior basis***)
  2. Combine known policies to rapidly solve novel tasks

**Open Problem:**
Learn a **behavior basis** whose policies can be combined to optimally solve (*zero-shot*) *any* novel task

## Multi-Task RL via Successor Features (SFs)

Tasks defined by linear rewards: $r_w(s,a,s') = \phi(s,a,s') \cdot w$

SFs: $\psi^\pi(s,a) \triangleq \mathbb{E}_\pi \left[ \sum_{i=0}^{\infty} \gamma^i \phi_{t+i} \mid S_t = s, A_t = a \right]$

Generalized Policy Evaluation (GPE): $q_w^\pi(s,a) = \psi^\pi(s,a) \cdot w$

## Generalized Policy Improvement (GPI)

Identifies a policy that improves over a ***set** of policies* $\Pi = \{\pi_i\}_{i=1}^n$

$$\pi^{\text{GPI}}(s; w) = \arg\max_{a \in \mathcal{A}} \max_{\pi \in \Pi} q_w^\pi(s,a)$$

GPI Theorem:

$$q_w^{\text{GPI}}(s,a) \geq \max_{\pi \in \Pi} q_w^\pi(s,a)$$

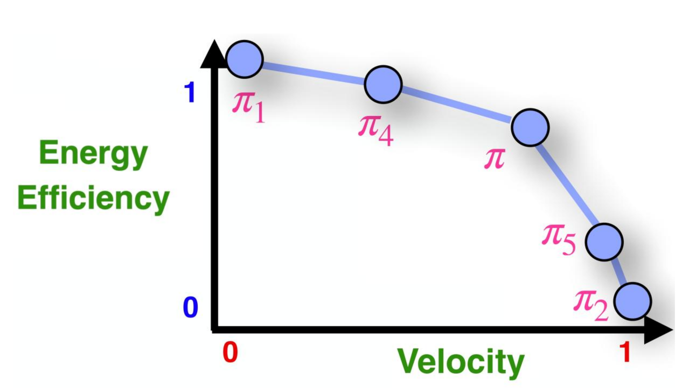
*for any* $w \in \mathcal{W}$

The resulting policy is not guaranteed to be optimal!

## Convex Coverage Set (CCS)

Methods that compute a CCS ensure optimality but are intractable

**Challenge:**
CCS grows exponentially with number of reward features!

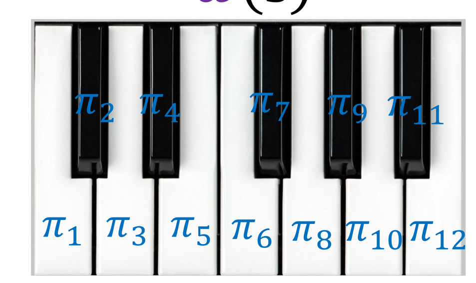



$\text{CCS} = \{\psi^\pi \mid \exists \mathbf{w} \text{ s.t. } \forall \psi^{\pi'}, \psi^\pi \cdot \mathbf{w} \geq \psi^{\pi'} \cdot \mathbf{w}\}$

## Option Keyboard (OK)

- Extends GPI
- *Learned meta-policy:* $\omega(s) \to \boldsymbol{z} \in \mathbb{R}^d$
- Increases expressivity → better performance

$\omega(s)$



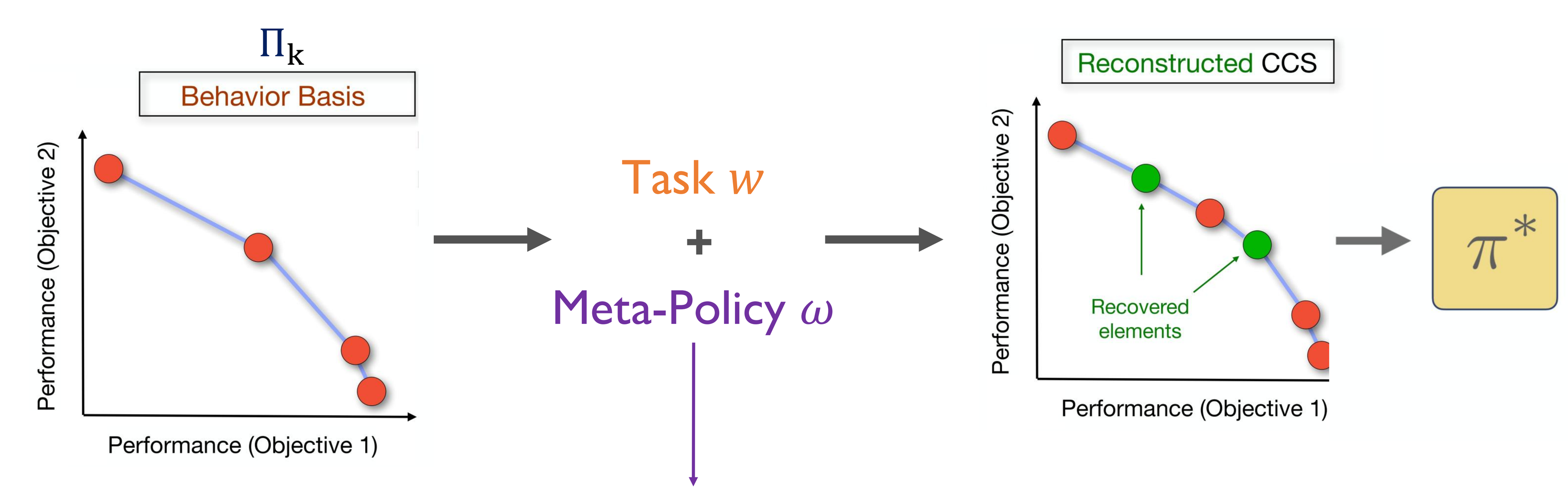
$$\pi_\omega^{\text{OK}}(s\,;\Pi) \in \arg\max_{a \in \mathcal{A}} \max_{\pi \in \Pi} \psi^\pi(s,a) \cdot \omega(s)$$

No principled techniques to identify a good behavior basis $\Pi$

## Goal

Learn a *small* set of policies (***behavior basis***) $\Pi_k$ such that:

- The behavior basis is smaller than a CCS: $|\Pi_k| \leq |\text{CCS}|$
- The Option Keyboard **guarantees optimality for any linear task**

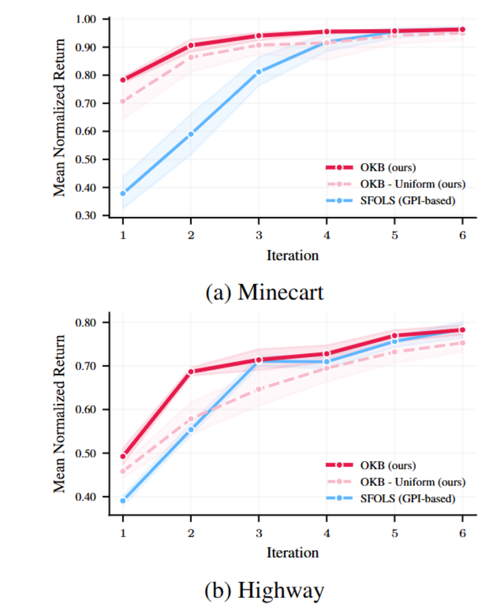


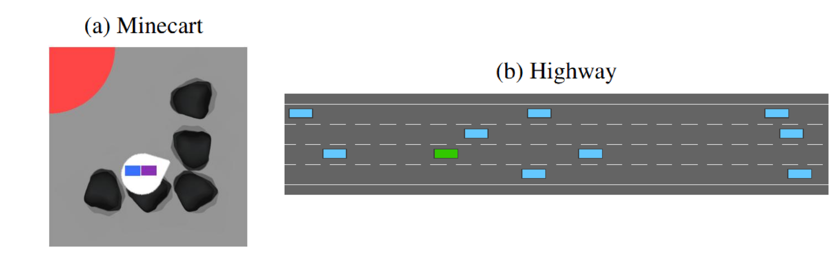

$$\pi_\omega^{\text{OK}}(s, w\,;\Pi_k) \in \arg\max_{a \in \mathcal{A}} \max_{\pi \in \Pi_k} \psi^\pi(s,a) \cdot \omega(s,w)$$

## Option Keyboard Basis (OKB)

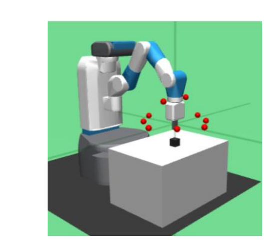
1. Novel method → identifies a small number of ***base policies / behavior basis*** ($\Pi_k$) for the Option Keyboard (OK)
2. Given novel task $w$ (weights of linear reward function):
   - Our method **combines** policies from the behavior basis
   - Combination mechanism → ***learned OK's meta-policy*** ($\omega$)
3. **Directly** identifies the ***optimal solution*** for the new task
   - No additional training needed!
   - **Zero shot** solution to **any** new linear reward function

## Experiments & Results

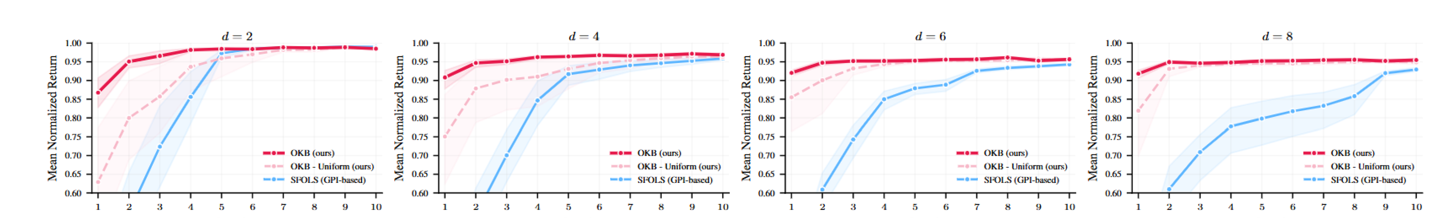


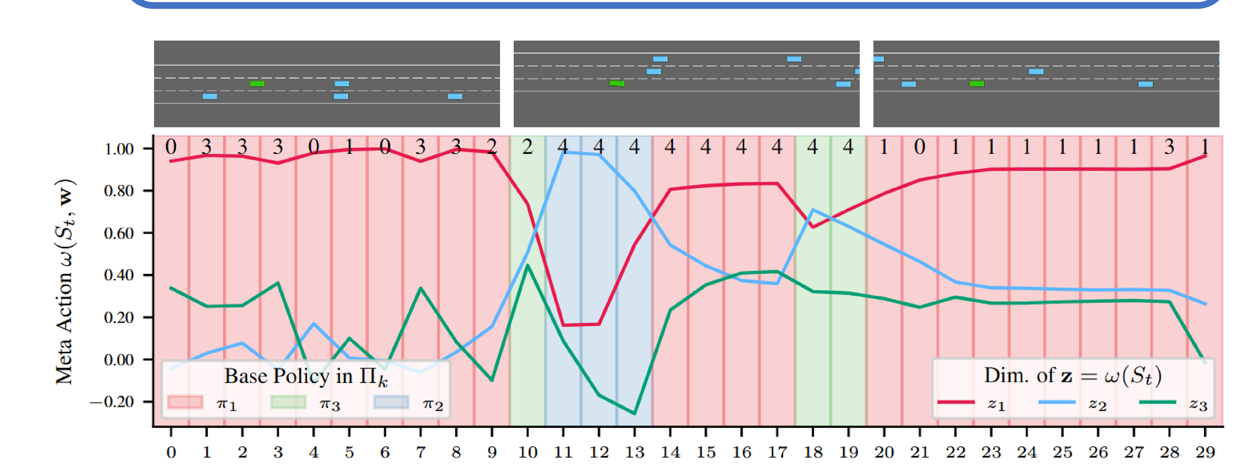

OKB approximates a CCS more effectively and using fewer base policies, when compared to SOTA methods

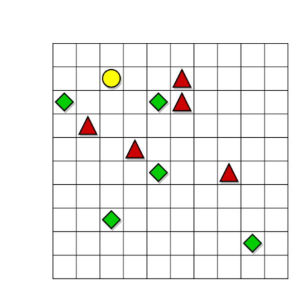


As number of reward features ($d$) increases, performance gap between OKB and SFOLS (GPI-based) increases significantly

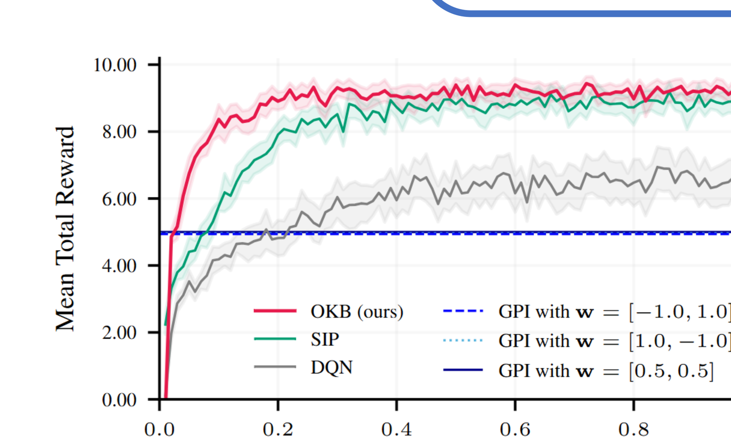



Learned base policies are temporally consistent (akin to *options* or *skills*)



After learning a behavior basis $\Pi_k$, OKB's meta-policy can *also* be trained to solve tasks with non-linear reward function




OKB can optimally solve classes of tasks with non-linear rewards (see Prop. 4.4)