Using Task Features for Zero-Shot Knowledge Transfer in Lifelong Learning

Penn Engineering


David Isele*
University of Pennsylvania


Mohammad Rostami*
University of Pennsylvania


Eric Eaton
University of Pennsylvania

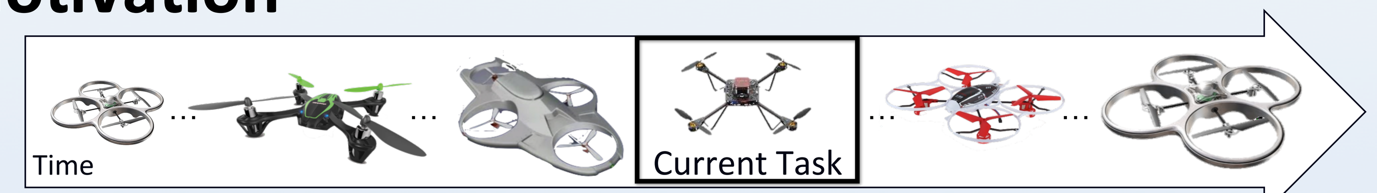
Summary

Knowledge transfer between tasks requires an accurate estimate of the inter-task relationships, which is inefficient in lifelong learning settings. We develop a **lifelong reinforcement learning** method that incorporates **high-level task descriptors** to model the inter-task relationships.

→ Improves the performance of the learned task policy

→ Accurately predicts the policy for a new task via zero-shot learning, given only the task description

Motivation



Lifelong learning accelerates training of each consecutive new task by building upon previously acquired knowledge via transfer

→ Relevant knowledge/tasks must be identified before transfer can occur

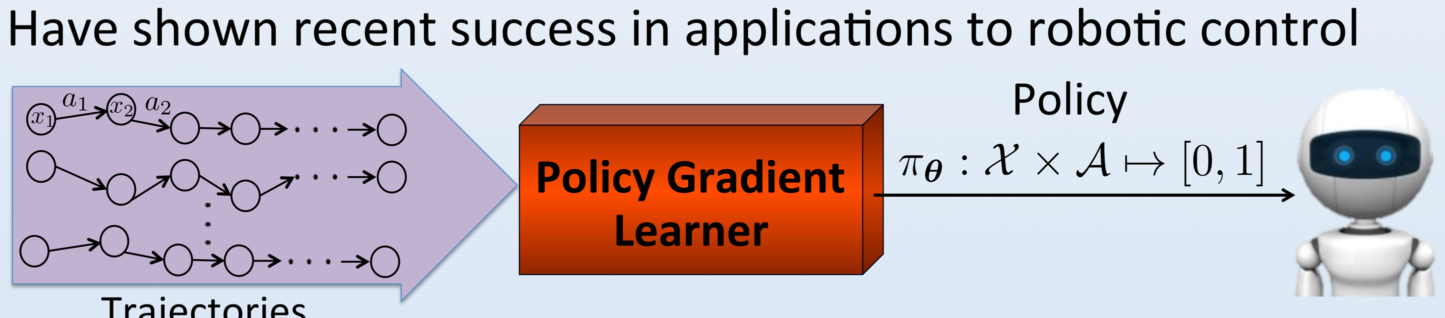
→ Requires interacting with the new task (i.e., sampling trajectories, learning, etc.) to characterize it

Alternative Idea: Can we use a high-level description of the task to identify relevant knowledge for transfer in lifelong learning?

Example task descriptor: physical specification of a quadrotor

Background: Policy Gradient (PG) Methods

- Agent interacts with environment, taking consecutive actions
- PG methods support continuous state and action spaces
  - Have shown recent success in applications to robotic control



Goal: find policy π_θ that minimizes $\mathcal{J}(\boldsymbol{\theta}) = \int_{\mathbb{T}} p_\theta(\boldsymbol{\tau})\mathcal{R}(\boldsymbol{\tau})d\boldsymbol{\tau}$

probability of trajectory

$$p_\theta(\boldsymbol{\tau}) = p_0(\mathbf{x}_0)\prod_{h=1}^{H} p(\mathbf{x}_{h+1}|\mathbf{x}_h, \mathbf{a}_h)\pi_\theta(\mathbf{a}_h|\mathbf{x}_h)$$

cost of trajectory

$$\mathcal{R}(\boldsymbol{\tau}) = \frac{1}{H}\sum_{h=0}^{H} r_{h+1}$$

Sharing Knowledge Between Multiple Tasks

- Policy for task t: $\pi_{\boldsymbol{\theta}^{(t)}} : \mathcal{X} \times \mathcal{A} \mapsto [0,1]$
- Factor the policy as $\boldsymbol{\theta}^{(t)} = \mathbf{L}\mathbf{s}^{(t)}$

Multi-Task Obj. Fn.

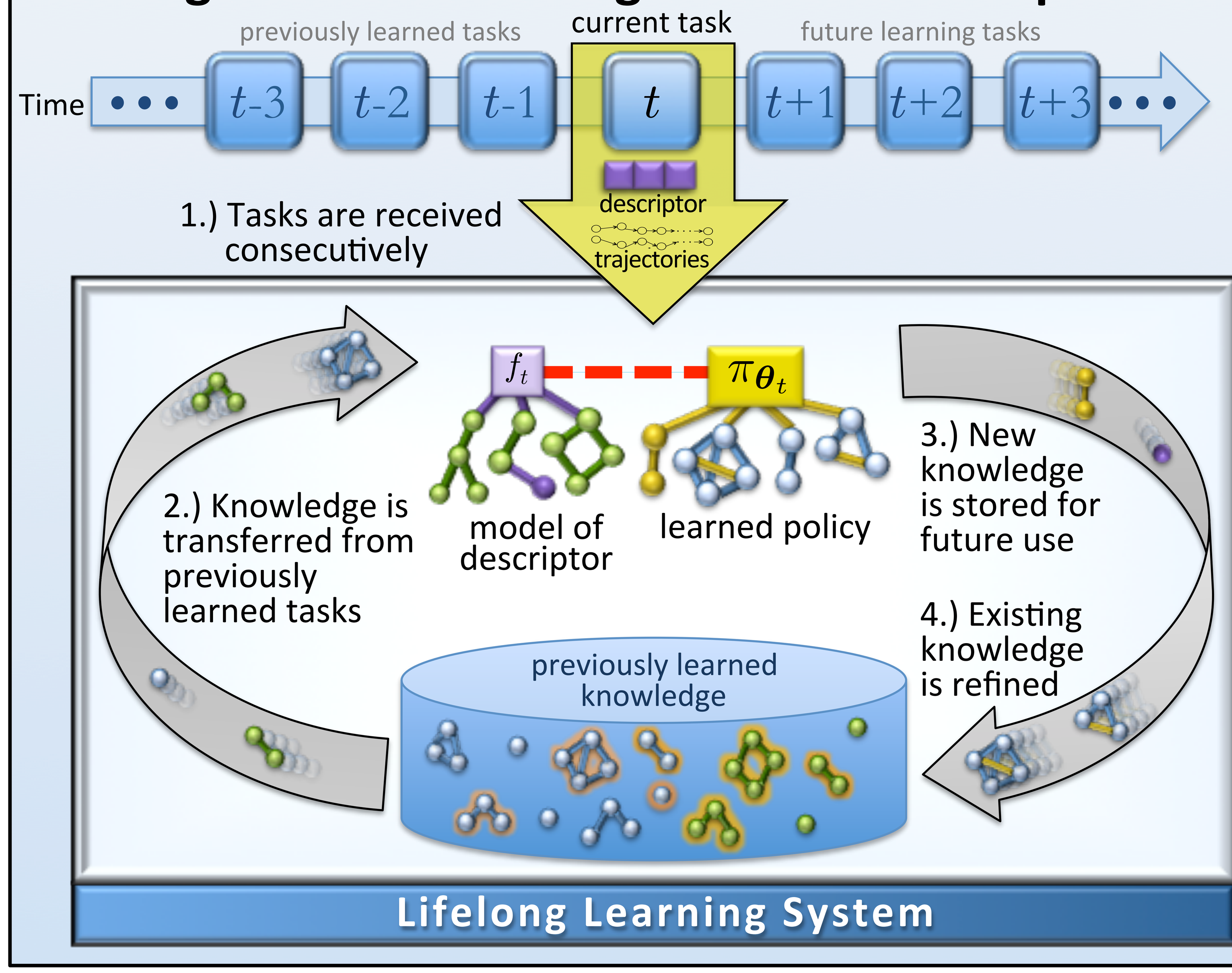
$$e_T(\boldsymbol{L}) = \frac{1}{T}\sum_{t=1}^{T}\min_{\boldsymbol{s}^{(t)}}\Big[\underbrace{-\mathcal{J}\left(\boldsymbol{\theta}^{(t)}\right)}_{\text{fit to each task}} + \underbrace{\mu\left\|\mathbf{s}^{(t)}\right\|_1}_{\text{sparsity of coefficients}}\Big] + \underbrace{\lambda\|\mathbf{L}\|_{\mathsf{F}}^2}_{\text{regularize basis complexity}}$$

Batch Optimization (all tasks are given) → **Multi-Task Learning**

Online Optimization (tasks arrive consecutively) → **Lifelong Learning (PG-ELLA)**

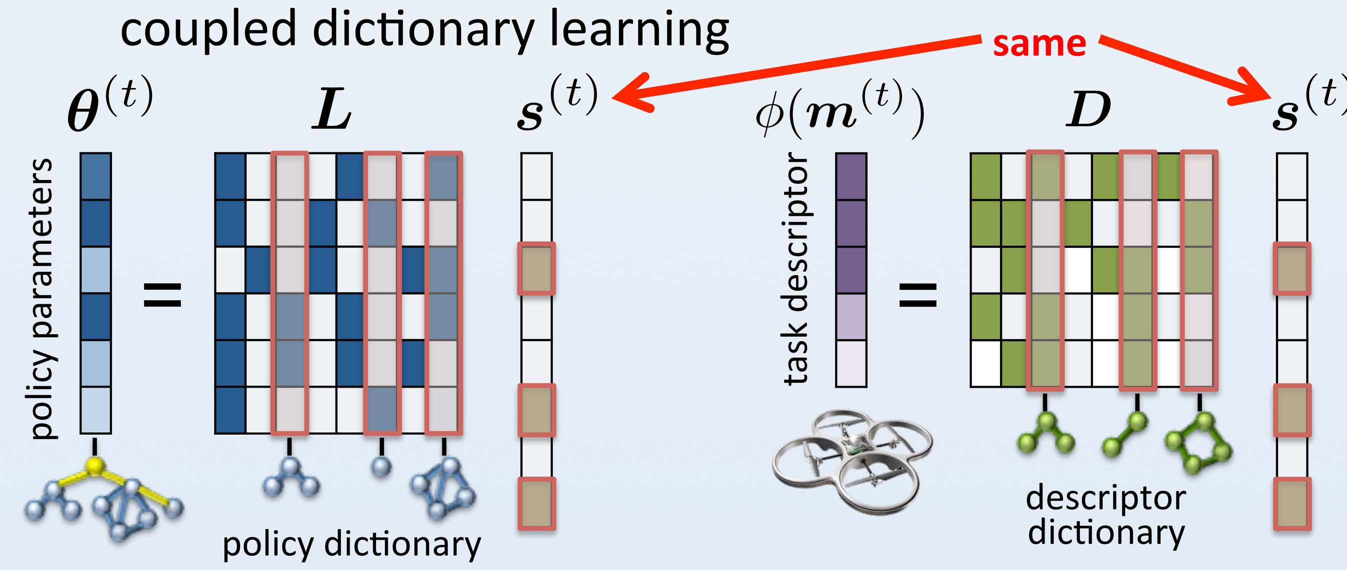
[Bou Ammar, Eaton, et al., ICML '14]

Lifelong Machine Learning with Task Descriptors



Incorporating Task Descriptors into Lifelong Learning

Key Idea: Relate policy parameters and task descriptors via coupled dictionary learning



Update the objective function to learn both dictionaries:

Obj. Fn.

$$\min_{\boldsymbol{L},\boldsymbol{D},\boldsymbol{S}} \frac{1}{T}\sum_t \Big[\underbrace{\left\|\boldsymbol{\alpha}^{(t)} - \boldsymbol{L}\boldsymbol{s}^{(t)}\right\|_{\boldsymbol{\Gamma}^{(t)}}^2}_{\text{policy fit}} + \underbrace{\rho\left\|\phi(\boldsymbol{m}^{(t)}) - \boldsymbol{D}\boldsymbol{s}^{(t)}\right\|_2^2}_{\text{descriptor fit}} + \underbrace{\mu\left\|\boldsymbol{s}^{(t)}\right\|_1}_{\text{sparsity}}\Big] + \underbrace{\lambda(\|\boldsymbol{L}\|_{\mathsf{F}}^2 + \|\boldsymbol{D}\|_{\mathsf{F}}^2)}_{\text{complexity}}$$

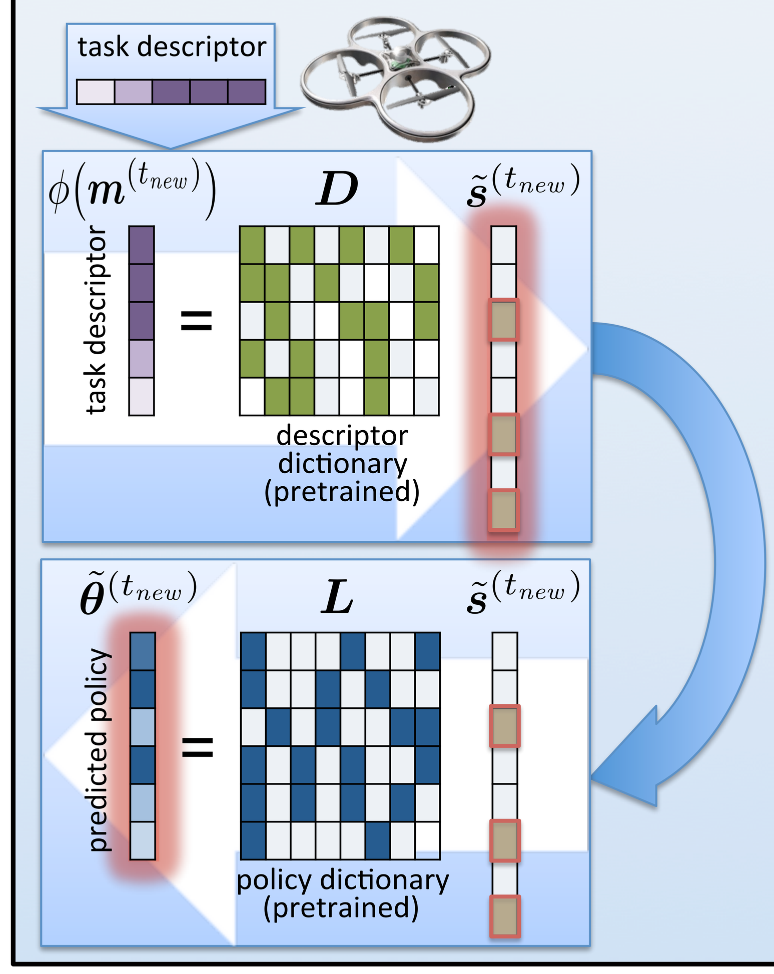
Multi-Task Learning (TaDeMTL)
- Fit via alternating optimization

Lifelong Learning (TaDeLL)
1. Merge $\boldsymbol{L}$ and $\boldsymbol{D}$ into single dictionary $\boldsymbol{K}$
2. Estimate policy $\boldsymbol{\alpha}^{(t)}$ via single-task learning
3. Sparse code estimated policy and descriptor in $\boldsymbol{K}$
4. Update $\boldsymbol{L}$ and $\boldsymbol{D}$

Algorithm 1 TaDeLL (k, λ, μ)

```
 1: T ← 0
 2: L ← RandomMatrix_{d,k},  D ← RandomMatrix_{m,k}
 3: while some task (Z^(t), φ(m^(t))) is available do
 4:     if isNewTask(Z^(t)) then
 5:         T ← T + 1
 6:         𝕋^(t) ← sampleRandomTrajectories(Z^(t))
 7:     else
 8:         𝕋^(t) ← sampleTrajectories(Z^(t), π_{α^(t)})
 9:     end if
10:     Compute α^(t) and Γ^(t) from 𝕋^(t)
11:     s^(t) ← arg min_s ||β^(t) − K s||²_{A^(t)} + μ||s||_1
12:     L ← updateL(L, s^(t), α^(t), Γ^(t), λ)
13:     D ← updateD(D, s^(t), φ(m^(t)), ρI_{d_m}, λ)
14:     for t ∈ {1, . . . , T} do: θ^(t) ← L s^(t)
15: end while
```

Zero-Shot Transfer via Task Descriptors



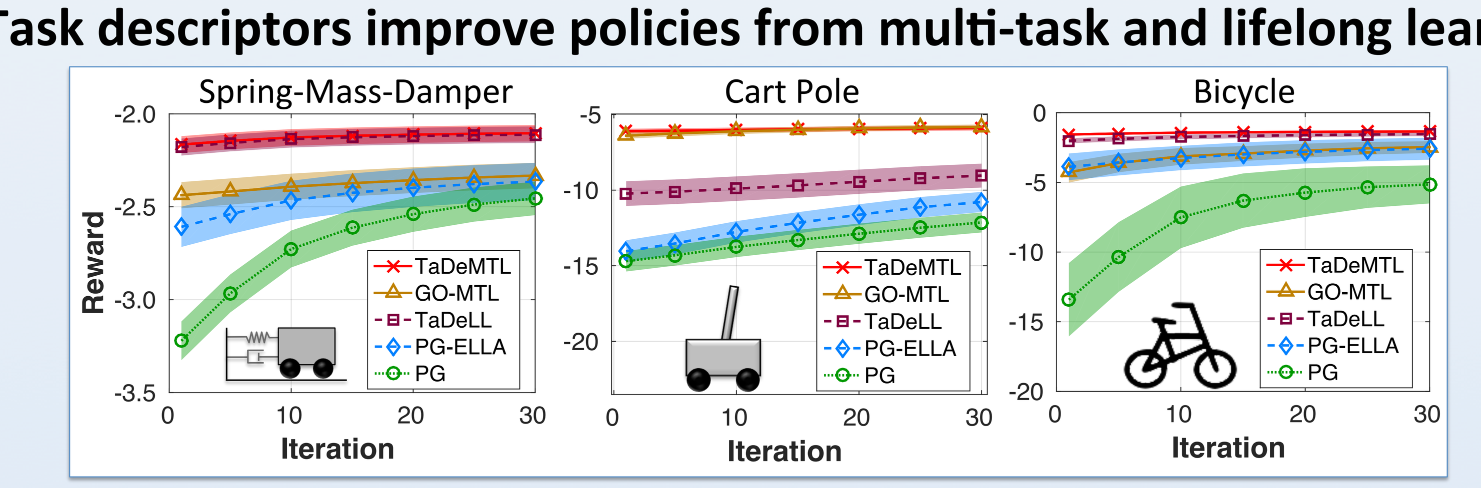
Given: descriptor for new task (e.g., quadrotor specs)

1. Use descriptor and descriptor dictionary to recover sparse coefficients via LASSO:
$$\tilde{\boldsymbol{s}}^{(t_{new})} \leftarrow \arg\min_{\boldsymbol{s}} \left\|\phi(\boldsymbol{m}^{(t)}) - \boldsymbol{D}\boldsymbol{s}\right\|_2^2 + \mu\|\boldsymbol{s}\|_1$$
2. Use recovered coefficients and policy dictionary to predict policy parameters
$$\tilde{\boldsymbol{\theta}}^{(t_{new})} = \boldsymbol{L}\tilde{\boldsymbol{s}}^{(t_{new})}$$

Experimental Results on Dynamical Systems

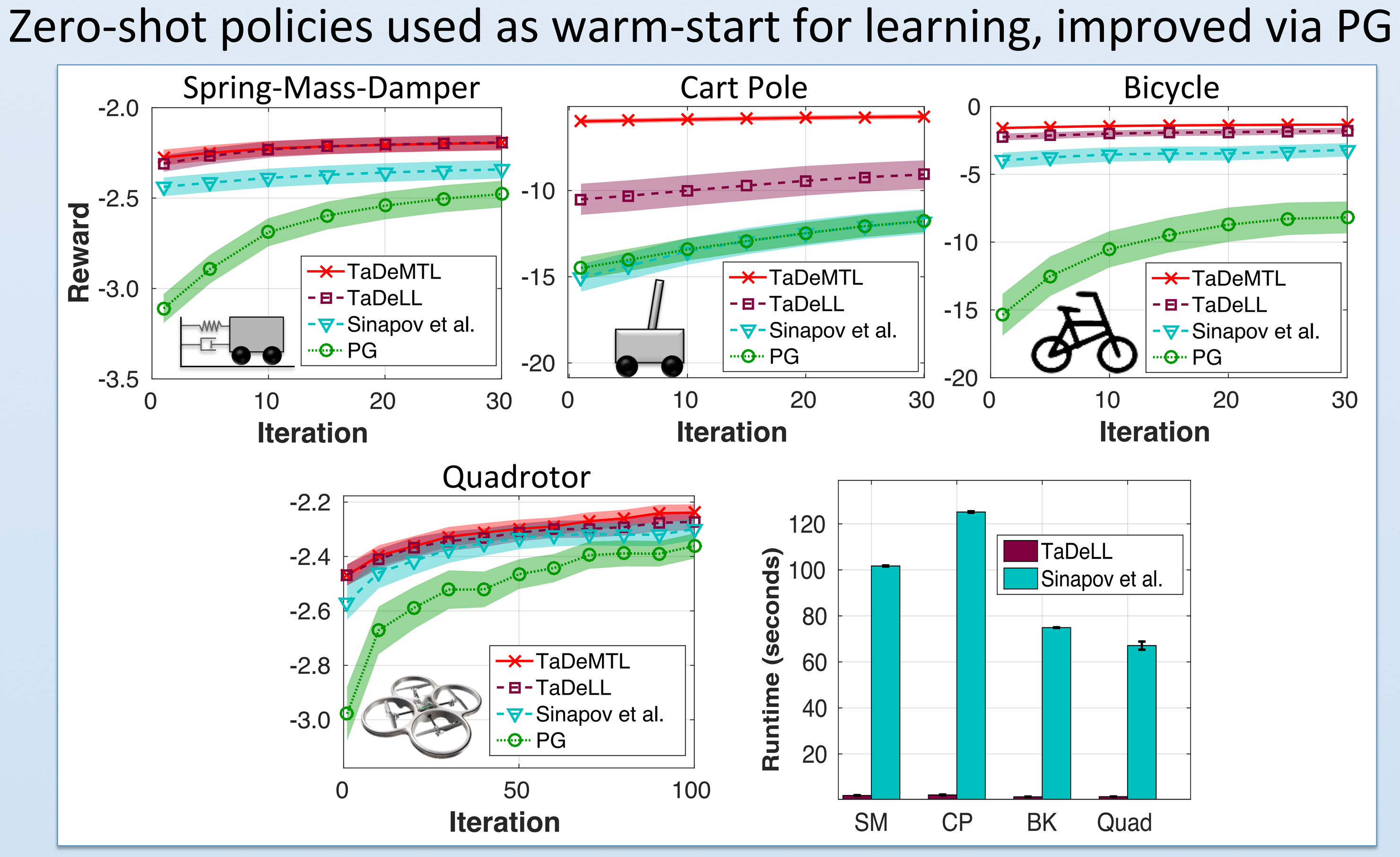
- Train on 40 different consecutive control tasks, transfer to new tasks

Task descriptors improve policies from multi-task and lifelong learning



Effective zero-shot transfer to new tasks

- Zero-shot policies used as warm-start for learning, improved via PG



Acknowledgements and Notes

This research was supported by ONR grant #N00014-11-1-0139 and AFRL grant #FA8750-14-1-0069.

* Authors contributed equally