Composing Complex Skills by Learning Transition Policies

Youngwoon Lee*, Shao-Hua Sun*, Sriram Somasundaram, Edward S. Hu, Joseph J. Lim

Presented in ICLR 2019
Compositional Task

\( \pi_{\text{walk}} \) \( \pi_{\text{jump}} \) \( \pi_{\text{crawl}} \)

\( \pi_{\text{forward}} \) \( \pi_{\text{backward}} \) \( \pi_{\text{balance}} \)

Reusable Skills

Compositional Task

\( \pi_{\text{walk}} \) \( \pi_{\text{crawl}} \) \( \pi_{\text{walk}} \) \( \pi_{\text{jump}} \) \( \pi_{\text{walk}} \)
Fail since these skills never learned to connect.
Good initial states for $\pi_{\text{walk}}$
$\pi_{\text{jump}}$

Good initial states for $\pi_{\text{walk}}$
Good initial states for $\pi_{\text{walk}}$
Good initial states for $\pi_{\text{walk}}$
Good initial states for $\pi_{\text{walk}}$
Good initial states for $\pi_{\text{walk}}$
Good initial states for $\pi_{\text{walk}}$
Good initial states for $\pi_{walk}$

- $\pi_{jump}$
- $\pi_{crawl}$
- $\pi_{walk}$

$\pi_{walk}$ fails
Good initial states for $\pi_{walk}$

Need to bring an agent from an ending state of a previous policy to a good initial state of a next policy

$\pi_{jump}$

$\pi_{crawl}$

$\pi_{walk}$

$\pi_{walk}$ fails
Good initial states for $\pi_{walk}$
Good initial states for $\pi_{walk}$
Good initial states for $\pi_{\text{walk}}$
Good initial states for $\pi_{\text{walk}}$
Good initial states for $\pi_{\text{walk}}$ succeed.
Good initial states for $\pi_{\text{walk}}$
Good initial states for $\pi_{\text{walk}}$ succeeds.
Obstacle course

Repetitive pick
Obstacle course

\[ \pi_{\text{crawl}} \]

Repetitive pick

\[ \pi_{\text{walk}} \]

\[ \pi_{\text{pick}} \]

\[ \pi_{\text{pick}} \]
Obstacle course

Repetitive pick
Obstacle course

\[ \pi_{\text{crawl}} \rightarrow \text{Transition policy} \rightarrow \pi_{\text{walk}} \]

Repetitive pick

\[ \pi_{\text{pick}} \rightarrow \text{Transition policy} \rightarrow \pi_{\text{pick}} \]
Smoothly connect skills

Obstacle course

Repetitive pick
Model

Observation ➔ Meta policy

Skill library

Jumping  Walking  Crawling
Model

Observation → Meta policy

Skill library

Jumping → Walking → Crawling

Transition policy

$\pi_{jump}$

Action

Termination

Repeat until reach a good initial state
Model

Observation → Meta policy

Skill library

Jumping  Walking  Crawling

Transition policy

\( \pi_{\text{jump}} \)

Action

Termination

Repeat until reach a good initial state
Model

Observation → Meta policy

Skill library

Jumping → Walking → Crawling

Transition policy ← $\pi_{\text{jump}}$

Action

Termination
Model

- Observation
  - Meta policy
    - Jumping
    - Walking
    - Crawling

- Skill library
- Transition policy
- $\pi_{\text{jump}}$

- Action
- Termination
Model

Observation → Meta policy

Skill library

Jumping → Walking → Crawling

Transition policy

\( \pi_{\text{jump}} \)

Action

Termination

if Termination == True
Model

Observation \rightarrow \text{Meta policy}

Skill library

- Jumping
- Walking
- Crawling

\text{Transition policy} \rightarrow \pi_{\text{jump}}

Action

Termination
How do we train a transition policy?
What is reward for learning a transition policy?
How do we train a transition policy?

What is reward for learning a transition policy?

- **Success of the following skill**
Learning Transition Policy

**Bad** initial states for $\pi_{\text{walk}}$

**Good** initial states for $\pi_{\text{walk}}$
Learning Transition Policy

**Bad** initial states for $\pi_{\text{walk}}$

**Good** initial states for $\pi_{\text{walk}}$

succeed
Learning Transition Policy

Bad initial states for $\pi_{\text{walk}}$

Good initial states for $\pi_{\text{walk}}$

succeed
Learning Transition Policy

Bad initial states for $\pi_{\text{walk}}$

Good initial states for $\pi_{\text{walk}}$
Learning Transition Policy

Successful execution of the following skill: +1

**Bad** initial states for $\pi_{\text{walk}}$

**Good** initial states for $\pi_{\text{walk}}$
Learning Transition Policy

Successful execution of the following skill: +1
Failing execution of the following skill: 0

**Bad** initial states for $\pi_{\text{walk}}$

**Good** initial states for $\pi_{\text{walk}}$
Learning Transition Policy

Successful execution of the following skill: +1
Failing execution of the following skill: 0

**Bad** initial states for $\pi_{\text{walk}}$

**Good** initial states for $\pi_{\text{walk}}$
Learning Transition Policy

Successful execution of the following skill: +1
Failing execution of the following skill: 0

Reward is *binary* and *sparse*!

We want *dense* reward for easier policy training.
Proximity Reward

Instead of binary reward

**Bad** initial states for $\pi_{\text{walk}}$

**Good** initial states for $\pi_{\text{walk}}$
Proximity Reward

Instead of binary reward, use "proximity prediction", which estimates how close to good initial states.
Proximity Reward

Instead of binary reward, use "proximity prediction", which estimates how close to good initial states.

We define proximity as: \( P(s) = \delta^{step} \)
Proximity Reward

Instead of binary reward, use “proximity prediction”, which estimates how close to good initial states.

We define proximity as: \( P(s) = \delta^{step} \)
Instead of binary reward, use “proximity prediction”, which estimates how close to good initial states

We define proximity as: \( P(s) = \delta^{\text{step}} \)
and provide proximity reward every step: \( P(s_{t+1}) - P(s_t) \)
Training Proximity Predictor

\[ \text{Success buffer} \quad (s, P(s)) \quad \text{Failure buffer} \]

\[ \text{success or failure?} \]

\[ \text{Jumping} \]
Training Proximity Predictor

Collect training data for proximity predictors
Training Proximity Predictor

Success buffer \( (s, P(s)) \)

Failure buffer

Collect training data for proximity predictors
Training Proximity Predictor

Collect training data for proximity predictors
Training Proximity Predictor

Train proximity predictors

Jumping
Transition
π
Success buffer
Failure buffer
(s, P(s))
Proximity predictor
Proximity P(s)

Train proximity predictors

Transition_{jump} π_{jump} Transition_{walk} π_{walk} Transition_{crawl} π_{crawl}

success or failure?

success
failure
Training Proximity Predictor

Train proximity predictors
Training Proximity Predictor

Transition $\text{jump} \quad \pi_{\text{jump}} \quad \text{Transition}_{\text{walk}} \quad \pi_{\text{walk}} \quad \text{Transition}_{\text{crawl}} \quad \pi_{\text{crawl}}$

Success buffer

Failure buffer

$(s, P(s))$

Proximity predictor

$s$

$P(s_{t+1}) - P(s_t)$

Provide more accurate proximity reward
Training Proximity Predictor

Getter better data with improved policy
Training Proximity Predictor

Train all transition policies simultaneously
Training Proximity Predictor

Train all transition policies simultaneously
Obstacle Course

- Crawl
- Transition
- Walk
Obstacle Course

Crawl

Transition

Walk
Obstacle Course

Crawl

Transition

Walk
Walk Forward & Backward

Walk Forward

Transition

Walk Backward
Walk Forward & Backward

Walk Forward

**Transition**

Walk Backward
Walk Forward & Backward

Walk Forward

Transition

Walk Backward
Repetitive Pick

Pick

Transition

Pick
Repetitive Pick

Pick

*Transition*

Pick
Repetitive Pick

Pick

Transition

Pick
Toss & Hit

Toss

Transition

Hit
Toss & Hit

Toss

Transition

Hit
Toss & Hit

Toss

Transition

Hit
Quantitative Results

(a) Repetitive picking up
(b) Repetitive catching
(c) Serve
(d) Patrol
(e) Hurdle
(f) Obstacle course
## Quantitative Results

### Manipulation

<table>
<thead>
<tr>
<th></th>
<th>Reward</th>
<th>Repetitive picking up</th>
<th>Repetitive catching</th>
<th>Serve</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRPO</td>
<td>dense</td>
<td>0.69 ± 0.46</td>
<td>4.54 ± 1.21</td>
<td>0.32 ± 0.47</td>
</tr>
<tr>
<td>PPO</td>
<td>dense</td>
<td>0.95 ± 0.53</td>
<td>4.26 ± 1.63</td>
<td>0.00 ± 0.00</td>
</tr>
<tr>
<td>Without TP</td>
<td>sparse</td>
<td>0.99 ± 0.08</td>
<td>1.00 ± 0.00</td>
<td>0.11 ± 0.32</td>
</tr>
<tr>
<td>TP-Task</td>
<td>sparse</td>
<td>0.99 ± 0.08</td>
<td>4.87 ± 0.58</td>
<td>0.05 ± 0.21</td>
</tr>
<tr>
<td>TP-Sparse</td>
<td>sparse</td>
<td>1.52 ± 1.12</td>
<td>4.88 ± 0.59</td>
<td>0.92 ± 0.27</td>
</tr>
<tr>
<td>TP-Dense (ours)</td>
<td>sparse</td>
<td><strong>4.84 ± 0.63</strong></td>
<td><strong>4.97 ± 0.33</strong></td>
<td><strong>0.92 ± 0.27</strong></td>
</tr>
</tbody>
</table>

### Locomotion

<table>
<thead>
<tr>
<th></th>
<th>Reward</th>
<th>Patrol</th>
<th>Hurdle</th>
<th>Obstacle course</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRPO</td>
<td>dense</td>
<td>1.37 ± 0.52</td>
<td><strong>4.13 ± 1.54</strong></td>
<td>0.98 ± 1.09</td>
</tr>
<tr>
<td>PPO</td>
<td>dense</td>
<td>1.53 ± 0.53</td>
<td>2.87 ± 1.92</td>
<td>0.85 ± 1.07</td>
</tr>
<tr>
<td>Without TP</td>
<td>sparse</td>
<td>1.02 ± 0.14</td>
<td>0.49 ± 0.75</td>
<td>0.72 ± 0.72</td>
</tr>
<tr>
<td>TP-Task</td>
<td>sparse</td>
<td>1.69 ± 0.63</td>
<td>1.73 ± 1.28</td>
<td>1.08 ± 0.78</td>
</tr>
<tr>
<td>TP-Sparse</td>
<td>sparse</td>
<td>2.51 ± 1.26</td>
<td>1.47 ± 1.53</td>
<td>1.32 ± 0.99</td>
</tr>
<tr>
<td>TP-Dense (Ours)</td>
<td>sparse</td>
<td><strong>3.33 ± 1.38</strong></td>
<td><strong>3.14 ± 1.69</strong></td>
<td><strong>1.90 ± 1.45</strong></td>
</tr>
</tbody>
</table>
Transition Trajectories
We propose to reuse skills to compose complex, long-horizon tasks.

Naive execution of skills fail since the skills never learned to connect.

**Transition policies** learn to smoothly connect skills.

**Proximity predictors** provide dense reward for efficient training of transition policies.