Deictic Image Mapping

Robert Platt
Colin Kohler
Marcus Gualtieri

Computer Science
Northeastern University
Goal: open world manipulation

Task: pick up stuff from floor and throw it out
   – novel objects
   – unstructured environments
   – novel starting configurations
Background: Grasp Pose Detection (GPD)

Our work: Gualtieri et al., ICRA 2016, ten Pas et al., IJRR 2017
Work by others: Mahler et al., RSS 2017; Levine et al., ISER 2016

ROS Package: https://github.com/atenpas/gpd
This talk: open world manipulation

Evaluation: define task with built-in variation
  – e.g. make coffee w/ single cup maker
  – e.g. put dishes in dish washer

Axes of variation:
  – variation over machines/mugs
  – variation over object pose
This talk: open world manipulation

Challenge: interact with objects w/o full models

If object pose is known:
This talk: open world manipulation

**Challenge:** interact with objects w/o full models

If object pose is unknown:
This talk: open world manipulation

Reinforcement learning?
Deep RL is not pose invariant
Deep reinforcement learning

Convolutional Layers

Fully Connected Layers

Run?

Jump?

Image Features

Value function
Deep RL for robotic manipulation
Suppose agent learns this policy...

Policy A

Convolutional Layers

Fully Connected Layers

Move left?

Rotate?
It does not generalize

Policy B
Agent must see both poses during training
Training set must span SE(2)
Training set must span camera poses
The *pose invariant policy learning* problem

Don’t want to learn a separate policy for each mug.
The *pose invariant policy learning* problem

Don’t want to learn to make coffee twice!
How do we encode things so that we generalize from this?
Question

How do we encode things so that we generalize from this to this?
Deictic Image Mapping
Assume actions are collision-free motions

Each action performs an end-to-end collision-free motion
Assume actions are collision-free motions

Each action performs an end-to-end collision-free motion – action set spans SE(2) or SE(3)
Standard Encoding of Action

Encode actions as coordinates in image
Deictic Image Encoding of Action

Encode this action as that image patch
DQN with Deictic Image Mapping

Reduced state representation → state → Convolutional Layers → action → Fully Connected Layers → $Q(s,a)$

Action represented as image patch
DQN with Deictic Image Mapping

Reduced state representation

Convolutional Layers

Fully Connected Layers

state

action

Q(s,a)

These two actions have same encoding
DQN with Deictic Image Mapping

Reduced state representation

These two actions have same encoding
Illustration: Pick/Place on Grid

**Actions**: pick or place in any cell (32 actions total)

**State**: set of grid configurations

**Transitions**: deterministic

**Reward**: two-in-a-row = 1; else 0.
Standard DQN Encoding

What’s the value of this transition:  

holding ∈ \{0, 1\}

32 actions
Deictic Action Encoding

What's the value of this transition: 💡?
Deictic Action Encoding

We want these two actions to have the same encoding:

That’s what Deictic Image Map accomplishes:
Comparison

DQN: deictic encoding

Grid Disk World

DQN: standard encoding

- 5x5 disk grid world
- avg over 10 trials

Platt, et al., Submitted 2018
Let’s think about the encoding as a mapping

Underlying State/Action

\[ S \times A \]

State

Action
Pick cell (1,3)

Abstract State/Action

\[ S' \times A' \]

holding \( \in \{0, 1\} \)

state

action

Convolutional Layers

Fully Connected Layers

Q(s, a)
Deictic Image Mapping

Underlying Space

\[ S \times A \]

Abstract

Abstract Space

\[ S' \times A' \]
Deictic Image Mapping

Underlying Space: $S \times A$

Abstract Space: $S' \times A'$

Abstract

Solve

$S' \times A'$
Use neural network to estimate: \( Q' : S' \times A' \rightarrow \mathbb{R} \)

Get underlying greedy action: \( a^* = \arg \max_{a \in A} Q'(f(s), g_s(a)) \)
Use neural network to estimate: \[ Q' : S' \times A' \rightarrow \mathbb{R} \]

Get underlying greedy action: \[ a^* = \arg\max_{a \in A} Q'(f(s), g_s(a)) \]
Deictic Image Mapping

Does this approach learn suboptimal policies? **No!**: can prove optimality using theory of MDP homomorphisms

Use neural network to estimate: $Q' : S' \times A' \rightarrow \mathbb{R}$

Get underlying greedy action: $a^* = \arg \max_{a \in A} Q'(f(s), g_s(a))$
Key challenge: large action space

e.g., this task has 13.5k pick actions + 13.5k place actions

Several ways to handle this:

- pass all actions in a single batch
- use fully convolutional network
- use hierarchical value function
- curriculum learning
Key challenge: large action space

e.g., this task has 13.5k pick actions + 13.5k place actions

Several ways to handle this:

- pass all actions in a single batch
- use fully convolutional network
- use hierarchical value function
- curriculum learning
Key challenge: large action space

Curriculum:
1. Disks, 100 actions
2. Disks, 400 actions
3. Blocks, 100 actions
4. Blocks, 200 actions
5. Blocks, 1.3k actions
6. Blocks, 7k actions
7. Blocks, 13.5k actions
8. Blocks, 26.9k actions

Total train time: ~1.5 hrs on one NVIDIA 1080
Demonstration: Novel Object Pick/Place

<table>
<thead>
<tr>
<th></th>
<th>Num Pick Attempts</th>
<th>Pick SR</th>
<th>Place SR</th>
<th>Task SR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bottle (Single)</td>
<td>112</td>
<td>99%</td>
<td>98%</td>
<td>97%</td>
</tr>
<tr>
<td>Bottle (Clutter)</td>
<td>107</td>
<td>97%</td>
<td>94%</td>
<td>92%</td>
</tr>
<tr>
<td>Mug (Single)</td>
<td>96</td>
<td>96%</td>
<td>93%</td>
<td>90%</td>
</tr>
<tr>
<td>Mug (Clutter)</td>
<td>96</td>
<td>93%</td>
<td>87%</td>
<td>80%</td>
</tr>
</tbody>
</table>

Gualtieri, et al., ICRA 2018
Demonstration: Novel Object Regrasping

<table>
<thead>
<tr>
<th></th>
<th>Regrasp</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grasp</td>
<td>0.94</td>
</tr>
<tr>
<td>FinalPlace</td>
<td>1.00</td>
</tr>
<tr>
<td>TempPlace</td>
<td>1.00</td>
</tr>
<tr>
<td>EntireTask</td>
<td>0.68</td>
</tr>
</tbody>
</table>

Gualtieri, et al., ICRA 2018
Toward a system for learning arbitrary tasks

<table>
<thead>
<tr>
<th>Task</th>
<th>Success Rate</th>
<th>Avg # Steps</th>
<th>Num Trials</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-in-row cube</td>
<td>94%</td>
<td>7.8</td>
<td>50</td>
</tr>
<tr>
<td>3 Cube Stack</td>
<td>90%</td>
<td>4.7</td>
<td>50</td>
</tr>
<tr>
<td>4 Cube Stack</td>
<td>84%</td>
<td>6.9</td>
<td>50</td>
</tr>
<tr>
<td>2 Rectangular Block Stack</td>
<td>80%</td>
<td>N/A</td>
<td>83</td>
</tr>
<tr>
<td>Cube in mug</td>
<td>94%</td>
<td>2.72</td>
<td>50</td>
</tr>
</tbody>
</table>

Gualtieri, et al., CoRL 2018, Platt et al., AAAI 2019
Summary

1. We tackle the *pose invariant policy learning* problem.

2. Assume a large space of reach actions – must be careful about how Q-function is encoded.

3. Provably correct.

4. Can work well in practice.

Marcus Gaultieri  
Colin Kohler