Do We Need 3D Representations for Robot Manipulation?

Yu Xiang
Assistant Professor
Intelligent Robotics and Vision Lab
The University of Texas at Dallas

1st Workshop on 3D Visual Representations for Robot Manipulation.
5/17/2024
Future Intelligent Robots in Human Environments

Manipulation

Senior Care

Assisting

Serving

Cooking

Cleaning

Dish washing
Some Recent Breakthroughs

Diffusion Policy, Columbia & MIT & TRI
Cheng Chi, Shuran Song, et al.
https://diffusion-policy.cs.columbia.edu/

Mobile ALOHA, Stanford
Zipeng Fu, Tony Zhao, Chelsea Finn
https://mobile-aloha.github.io/
What is the Representation in these Models?

Diffusion Policy, Columbia & MIT & TRI

Mobile ALOHA, Stanford

Do we need 3D representations?

Camera images ➔ Visual Encoder ➔ Policy Network ➔ Actions

End-to-end Learning from Images
Robot Autonomy

Task Diversity

Multiple Tasks

Single Task

Environment Diversity

Single Environment

Multiple Environments

Intelligent Robots

- Navigation
- Manipulation
- Long-horizon tasks

Current End-to-end Policy Learning

- Collect more data in many environments for many tasks for learning?
- Use simulators for learning with sim-to-real transfer?
- Enable robots to understand the 3D physical world with planning and control?

Etc. (open question)
The Perception, Planning and Control Loop

Good Old Fashioned Engineering (GOFE)

How to Represent Objects?
How to Represent Objects?

- 3D CAD models (Model-based)

- Point clouds (Model-free)
Using 3D Object Models

Perception

- 6D object pose estimation

Planning

- Grasp planning and motion planning

Control

- Manipulation trajectory following

2X
6D Object Pose Estimation

- PoseCNN, RSS’17
- DeepIM, ECCV’18
- DOPE, CoRL’18
- PoseRBPF, RSS’19, T-TO’21
- Self-supervised 6D Pose, ICRA’20
- LatentFusion, CVPR’20

FoundationPose: Unified 6D Pose Estimation and Tracking of Novel Objects
Bowen Wen, Wei Yang, Jan Kautz, Stan Birchfield
Grasp Planning: GraspIt!

GraspIt!  https://graspit-simulator.github.io/

Grasp Planning: A Physics-based Approach
MultiGripperGrasp

• A large-scale dataset for robotic grasping
  • 11 grippers, 345 objects, 30M grasps
MultiGripperGrasp

• 11 grippers (aligned with palm directions)

  • 2-finger grippers: Fetch, Franka Panda, WSG50, Sawyer, H5 Hand
  • 3-finger grippers: Barrett, Robotiq-3F, Jaco Robot
  • 4-finger grippers: Allegro
  • 5 finger grippers: Shadow, Human Hand
MultiGripperGrasp

• Generate initial grasps using GraspIt!
• Ranking grasps in Isaac Sim
MultiGripperGrasp

• Grasp Transfer in Isaac Sim

Source: Fetch

Grasp Transfer

Sawyer  WSG50  Panda  H5 Hand  Barrett  Jaco Robot  Robotiq-3F  Allegro  Shadow  Human Hand

https://irvlutd.github.io/MultiGripperGrasp/
The Open Motion Planning Library in MoveIt

https://ompl.kavrakilab.org/index.html
Using 3D Object Models

• Pros
  • Encodes appearance, 3D shape, affordance, physical properties for perception, planning and simulation

• Cons
  • We cannot build 3D models for all objects
Using 3D Point Clouds

Perception → Planning → Control

- object instance segmentation
- Grasp planning from point clouds
- Control to reach grasp

Figure Credit: Murali-Mousavian-Eppner-Paxton-Fox, ICRA’20
Segmenting Unseen Objects

Input Image

Output Label

Xie-Xiang-Mousavian-Fox, CoRL’19, T-RO’21, CoRL’21
Xiang-Xie-Mousavian-Fox, CoRL’20
Lu-Khargonkar-Xu-Averill-Palanisamy-Hang-Guo-Ruozi-Xiang, RSS’23
Lu-Chen-Ruozi-Xiang, ICRA’24
Qian-Lu-Ren-Wang-Khargonkar-Xiang-Hang, ICRA’24
Leveraging Large Models from the Vision Community

- Gounding Dino (object detection)
- SAM (object segmentation)

- Grounding DINO: Marrying DINO with Grounded Pre-Training for Open-Set Object Detection. Liu et al., 2023
- Segment Anything. Kirillov et al., 2023
Grasp Planning with Point Clouds

6D GraspNet
6-DOF GraspNet: Variational Grasp Generation for Object Manipulation. Mousavian et al., ICCV’19

Contact-GraspNet
Contact-GraspNet: Efficient 6-DoF Grasp Generation in Cluttered Scenes. Sundermeyer, et al., ICRA’21

SE(3)-DiffusionFields
SE(3)-DiffusionFields: Learning smooth cost functions for joint grasp and motion optimization through diffusion. Urain et al., 2023
Model-free Grasping Example

Demo Scene 1

Segmentation
RGB

Rviz capture
RealSense camera capture

8X speed up
Grasping Trajectory Optimization with Point Clouds

(a) Task Space
(b) Grasp Planning
(c) Grasp Trajectory Optimization

Grasping Trajectory Optimization with Point Clouds. Yu Xiang, Sai Haneesh Allu, Rohith Peddi, Tyler Summers, Vibhav Gogate
In arXiv, 2024.
Grasping Trajectory Optimization with Point Clouds

• Represent robots as point clouds (can be used for any robot)

(a) A Fetch Mobile Manipulator
(b) A Franka Panda Arm
Grasping Trajectory Optimization with Point Clouds

- Represent task spaces as point clouds (can be used for any task)
- Build signed distance fields using point clouds for collision avoidance
Grasping Trajectory Optimization with Point Clouds

• Solve a trajectory with joint positions and joint velocities

\[ Q = (q_1, \ldots, q_T) \quad \dot{Q} = (\dot{q}_1, \ldots, \dot{q}_T) \]

\[
\arg \min_{Q, \dot{Q}} \left( \min_{i=1}^{K} \left( c_{\text{goal}}(T(q_T), T_i) + c_{\text{standoff}}(T(q_{T-\delta}), T_i T_\Delta) \right) \right. \\
+ \lambda_1 \sum_{t=1}^{T} c_{\text{collision}}(q_t) + \lambda_2 \sum_{t=1}^{T} \| \dot{q}_t \|^2 \\
\left. \quad \text{s.t.,} \quad q_1 = q_0 \right) \\
\dot{q}_1 = 0, \dot{q}_T = 0 \\
q_{t+1} = q_t + \dot{q}_t dt, t = 1, \ldots, T - 1 \\
q_l \leq q_t \leq q_u, t = 1, \ldots, T \\
\dot{q}_l \leq \dot{q}_t \leq \dot{q}_u, t = 1, \ldots, T, \]
Grasping Trajectory Optimization with Point Clouds

• Simulation results

PyBullet Shelf Grasping
SceneReplica Benchmark

20 Scenes

SceneReplica, ICRA’24: https://irvlutd.github.io/SceneReplica/
Real-World Scene Setup

Reference Image

Real World Setup
## SceneReplica Benchmark

<table>
<thead>
<tr>
<th>Method #</th>
<th>Perception</th>
<th>Grasp Planning</th>
<th>Motion Planning</th>
<th>Control</th>
<th>Ordering</th>
<th>Pick-and-Place Success</th>
<th>Grasping Success</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>MSMFormer [27]</td>
<td>Contact-graspnet [29] + Top-down</td>
<td>OMPL [24]</td>
<td>MoveIt</td>
<td>Near-to-far</td>
<td>57 / 100</td>
<td>65 / 100</td>
</tr>
<tr>
<td>8</td>
<td>MSMFormer [27]</td>
<td>Top-down</td>
<td>OMPL [24]</td>
<td>MoveIt</td>
<td>Fixed</td>
<td>61 / 100</td>
<td>70 / 100</td>
</tr>
<tr>
<td>9</td>
<td>Dex-Net 2.0 [37] (Top-Down Grasping)</td>
<td>OMPL [24]</td>
<td>MoveIt</td>
<td>Algorithmic</td>
<td>43 /100</td>
<td>51 / 100</td>
<td></td>
</tr>
</tbody>
</table>
Grasping Trajectory Optimization with Point Clouds

• Real world experiments

<table>
<thead>
<tr>
<th>Method #</th>
<th>Perception</th>
<th>Grasp Planning</th>
<th>Motion Planning</th>
<th>Control</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>MSMFormer [33]</td>
<td>Contact-graspnet [29] + Top-down</td>
<td>OMPL [34]</td>
<td>MoveIt</td>
</tr>
<tr>
<td>2</td>
<td>MSMFormer [33]</td>
<td>Contact-graspnet [29] + Top-down</td>
<td>GTO (Ours)</td>
<td>MoveIt</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ordering</th>
<th>Pick-and-Place Success</th>
<th>Grasping Success</th>
</tr>
</thead>
<tbody>
<tr>
<td>Near-to-far</td>
<td>57 / 100</td>
<td>65 / 100</td>
</tr>
<tr>
<td>Near-to-far</td>
<td>65 / 100</td>
<td>71 / 100</td>
</tr>
</tbody>
</table>
Using 3D Point Clouds

• Pros
  • No need to build 3D models
  • Direct sensor input from RGB-D cameras
  • Encode appearance and 3D geometry

• Cons
  • It is difficult to capture depth for certain objects (flat, thin, transparent, metal)
  • Planning from partial observations
Do We Need 3D Representations for Robot Manipulation?

• Depends on your goals

• Goal: A repetitive task in a specific environment
  • Probably no need

• Goal: enabling any robot to do any task in any environment
  • We need to think about generalization
  • 3D Understanding might help generalization in robots, tasks and environments
  • 3D representation + policy learning (e.g., 3D diffusion policy)

Thank you!