Principles of Robot Autonomy II

Learning-based Approaches to **Grasping and Manipulation**

Jeannette Bohg
Learning Outcome for next four Lectures

Modeling and Evaluating Grasps

Modeling and Executing Manipulation

Apply Learning to Grasping and Manipulation

Use Manipulation to Perceive better
Why are friction cones triangles (2D) or cones (3D)?

\[ f = f_{\text{normal}} + f_{\text{tangent}}, \]

\[ \mathcal{F} = \{ f \mid \| f_{\text{tangent}} \| \leq \mu_s \| f_{\text{normal}} \|, \quad f_z \geq 0 \}. \]
Grasp Force Optimization

Figure adapted from *A Grasping Force Optimization Algorithm for Multiarm Robots With Multifingered Hands*. Lipiello et al. Transactions on Robotics. 2013

Fig. 3. Sequence of significant configurations of the bottle and of the forces during task execution with $n = 10$. 
Equilibrium Constraints – Force Closure

Compact notation

• Contact force vector $f \in \mathbb{R}^{3M}$
  
  \[ f = (f^{(1)}, \ldots, f^{(M)}) \]

• Contact Matrices $G_i \in \mathbb{R}^{6\times3}$
  
  \[ G_i = \frac{Q^{(i)}}{S^{(i)}Q^{(i)}}, i = 1 \ldots M \]

• Grasp matrix
  
  \[ G = [G_1, \ldots, G_M] \in \mathbb{R}^{6\times3M} \]

• External Wrench $\omega^{\text{ext}} = (f^{\text{ext}}, \tau^{\text{ext}})$

• Equilibrium conditions
  
  \[ Gf + \omega^{\text{ext}} = 0 \]

Convex Optimization Problem

• Second-order cone program because friction cones are quadratic.

• Objective function:

\[ F_{\text{max}} = \max \{ \| f^{(1)} \|, \ldots, \| f^{(M)} \| \} \]
\[ = \max_{i=1,\ldots,M} \sqrt{f_x^{(i)2} + f_y^{(i)2} + f_z^{(i)2}} \]

• Optimization problem:
  • minimize \( F_{\text{max}} \)
  • subject to \( f^{(i)} \in K_i, i = 1 \ldots M \)
  • \( Gf + \omega^{\text{ext}} = 0 \)

Today’s itinerary

• Modeling Push/Non-Prehensile Manipulation
• Learning-based Approaches to
  • Grasping
  • Planar Pushing
  • Manipulation (Guest Lecture Feb 21 by Quan Vuong from Google DeepMind)
For a Deeper Dive into Grasping and Manipulation

• CS326 – Topics in Advanced Robotic Manipulation – Fall 2024
Case Study – Planar Pushing

Reorient parts
- Mason 1986

Transport *large objects*
- Meričli 2015

*Push-grasp* under clutter
- Dogar 2010

Track object *pose*
- Koval 2015

Stable Pushes to manoeuvre an object around obstacles. Adopted from Chapter 37, Fig 37.11 in Springer Handbook of Robotics.

\[
\arg \min_{u(t)} h(x(T)) + \int_0^T g(x(t), u(t)) \, dt
\]
Modeling Planar Pushing

**Friction limit surface**: describes friction forces occurring when part slides over support.

When pushed with a wrench within the limit surface: no motion.

For **quasi-static pushing**: wrench on the limit surface; object twist normal to limit surface where **twist** = linear and angular velocity: $t_i = (v_x^i, v_y^i, \omega_z^i)$

If **object translates without rotation** the friction force magnitude $\mu mg$ where $\mu$ = friction coefficient, $m$ = object mass, $g$ = gravitational acceleration

Relation between wrench cone, limit surface and unit twist sphere. Adopted from Chapter 37, Fig 37.10 in Springer Handbook of Robotics.
Modeling Planar Pushing – Voting theorem

How will the object rotate? Adopted from Chapter 37, Fig 37.12 in Springer Handbook of Robotics.

Validating Models for Planar Pushing

IROS 2016, "More than a Million Ways to Be Pushed: A High-Fidelity Experimental Dataset of Planar Pushing" by Peter Yu, Maria Bauza et al.
Validating Models for Planar Pushing

More than a Million Ways to Be Pushed.
A High-Fidelity Experimental Dataset of Planar Pushing

Kuan-Ting Yu, Maria Bauza, Nima Fazeli, and Alberto Rodriguez
Computer Science and Artificial Intelligence Lab &
Mechanical Engineering Department, MIT

IROS 2016, "More than a Million Ways to Be Pushed: A High-Fidelity Experimental Dataset of Planar Pushing" by Peter Yu, Maria Bauza et al.
Validating Models for Planar Pushing

IROS 2016, "More than a Million Ways to Be Pushed: A High-Fidelity Experimental Dataset of Planar Pushing" by Peter Yu, Maria Bauza et al.
Validating Models for Planar Pushing

More than a Million Ways to Be Pushed.
A High-Fidelity Experimental Dataset of Planar Pushing

Kuan-Ting Yu, Maria Bauza, Nima Fazeli, and Alberto Rodriguez
Computer Science and Artificial Intelligence Lab &
Mechanical Engineering Department, MIT

IROS 2016, "More than a Million Ways to Be Pushed: A High-Fidelity Experimental Dataset of Planar Pushing" by Peter Yu, Maria Bauza et al.
Validating Models for Planar Pushing

\[ \arg \min_{u(t)} h(x(T)) + \int_0^T g(x(t), u(t)) \, dt \]
Suggested Reading

• More than a Million Ways to Be Pushed: A High-Fidelity Experimental Dataset of Planar Pushing by Peter Yu, Maria Bauza et al. IROS 2016.

• Maria Bauza and Alberto Rodriguez. A probabilistic data-driven model for planar pushing. ICRA 2017
What are common assumptions?
How do we generate a grasp?

**Offline**
- Grasp Evaluation
- Offline database with grasps linked to 3D objects

**Online**
- Perception
- Motion Planning
How do we execute a grasp?

Top-Down & Open-Loop

Acquiring a grasp + Closed Loop

Grasp Force Optimization
Data-Driven Approaches to Grasping

Covered up till now

This lecture

Detecting 2D Grasping Points

Grasp Point Detection as a Classification Problem

\[ P(y = 1| x) \]

Supervised Learning

Feature

Grasp Success?

From 2D Grasping Points to 6D Grasp Pose

\[ P(y = 1 | x) \]

Grasping previously unseen objects using only 2D images without 3D meshes
Supervised learning pipeline

**TRAINING**

Input: Labeled 2D images

- Raw data & target
  - Feature Engineering
  - Training Set
  - Validation Set
  - Test Set
  - Machine Learning
  - Hyperparameters tuning
  - Model selection
  - Model training

**PREDICTING**

- New data
  - Feature Engineering
  - Predict
  - Target
Data collection

We could collect real images…

…but labeling them is cumbersome / prone to errors.
Data collection

Solution? Use synthetic data!

Realistic rendering using ray tracing.
Enables automatic labeling: random lighting, color, orientation, size...

2500 images
5 object classes
Supervised learning pipeline

TRAINING

Preprocessing

Raw data & target → Feature Engineering → Training Set → model training

Validation Set → hyperparameters tuning → model selection

Test Set → evaluation → Model

PREDICTING

New data → Feature Engineering → Predict → Target
Image preprocessing

**RGB** -> **Y** (luma) **Cb** (chroma) **Cr** (chroma)

**Y**: intensity; **Cb**: **B** - **Y**; **Cr**: **R** - **Y**
Image preprocessing

Edge filters (Y):

Texture filters (Y):

Average filter (Cb/Cr):

6 (edge) + 9 (texture) + 1 (average) * 2 = 17 features per patch
Image preprocessing

Apply filters on:
- 3 different scales for the patch centered at the pixel of interest
- 1 scale for the 24 surrounding patches in a 5x5 window

17 (# features/patch) * (3 + 24) = 459 features per patch of interest
Supervised learning pipeline
Binary classification task

Is a given pixel $(u,v)$ on the image a grasping point (1) or not (0)?
Binary classification task

Prediction time:

\[ P(z(u, v) = 1 | C) = \sigma(\hat{\theta}^T x) \]

How likely is pixel \((u, v)\) on image \(C\) a grasping point?

Learned parameter

Features of the patch centered on \((u, v)\)
Supervised learning pipeline
2D -> 3D

Link 2D to 3D intuitively:

\[ z_i(u, v) = 0 \iff y_{r_1}(u, v) = 0 \land \ldots \land y_{r_k}(u, v) = 0 \]

Pixel is not a grasping point

No grid cells along the ray passing through the pixel contain a grasping point
Supervised learning pipeline
Hardware setup

5 dof arm

Random object location on uncluttered table top
Evaluation results

1. Synthetic data:
   Classification accuracy on unseen images is 94.2% (2D).
   Accuracy on unseen images after triangulation is higher (3D), mean error 0.84 cm.

2. Real data:
   Mean error after triangulation (3D) 1.84 cm.
   Picked up novel objects 87.8% of the time.
Application task: unloading dishwasher

Added real images + depth measurements

<table>
<thead>
<tr>
<th>Tested on</th>
<th>Grasp success rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plates</td>
<td>100%</td>
</tr>
<tr>
<td>Bowls</td>
<td>80%</td>
</tr>
<tr>
<td>Mugs</td>
<td>60%</td>
</tr>
<tr>
<td>Wine glass</td>
<td>80%</td>
</tr>
<tr>
<td>Overall</td>
<td>80%</td>
</tr>
</tbody>
</table>
Conclusion

- Learning-based method
- Only input is 2D images, no 3D mesh model needed
- **Generalizes** to previously unseen objects
- Cool applications!
Using more sensing modalities and data to learn features and grasp policies

- DexNet 1.0 – 4.0 – Berkeley – AutoLab
- Google Arm Farm

"Learning Hand-Eye Coordination for Robotic Grasping with Deep Learning and Large-Scale Data Collection" by Levine et al. IRR 2017.

https://berkeleyautomation.github.io/dex-net
Dex-Net 2.0: Deep Learning to Plan Robust Grasps with Synthetic Point Clouds and Analytic Grasp Metrics

Dataset Generation

At test time: \( \pi_\theta(y) = \arg\max_{u \in \mathcal{C}} Q_\theta(u, y) \) where \( y = \text{pointcloud} \), \( u = \text{grasp parameters} \)

Video

Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection

Sergey Levine, Peter Pastor, Alex Krizhevsky, Julian Ibarz, Deirdre Quillen
Problem Statement

**End-to-end learn** to grasp a **wide variety** of household objects in **clutter** using **real hardware**.
Assumptions

- 3D Model of Object
  - Depth Sensing
  - Wrist Mounted Camera
- Specific Representation of Geometry
- Contact Model
- Simulated Data
- Hand Annotations
- Hand Designed Path Planner

RGB Camera
Mounted Over-the-Shoulder
- Camera-to-Base Calibration
So what do we have?

- Monocular RGB camera
- 7 DoF robotic manipulator
- 2-finger gripper
- Object bin

Underactuated to conform to object geometry
So what do we have?

+ Time
Goal

“Examine to what degree a grasping method based entirely on learning from raw autonomously collected data can scale to complex and diverse grasp scenarios”
Uncertainty

- Using real hardware leads to a ton of uncertainty
  - Object
    - Geometry & Pose
    - Material Properties
      - weight, frictional properties, deformability
  - Robot
    - End-Effector Pose
    - Wear and Tear
- Accentuated by lack of explicit hand-eye-coordination
Dataset

**Data Point Format**
(Image, Motor Command, Label)

\[(p_T - p_t)\]

Success
or
Failure
Dataset

Two Rounds of Self-Supervised Data Collection

800k grasp attempts
2 months

900k grasp attempts
4 months

1.7M Grasp Attempts
Self-Supervised Data Collection: Phase 1

- **t = 0**: “Random” Motor Command
- **t = 1**: “Random” Motor Command + Close Gripper
- **t = 2 = T**: Check if successfully grasped an object

Dataset

\[(I_0, p_0)\] \[\rightarrow\] \[(I_1, p_1)\] \[\rightarrow\] \[(I_T, p_T)\]

\[\rightarrow\] \[(I_0, p_T - p_0, 1)\] \[\rightarrow\] \[(I_1, p_T - p_1, 1)\]

Stanford University
Self-Supervised Data Collection: Phase 2

```
6HO16XSHUYLVHG'DWD&RROOHFWLRQ3KDVH
0RWRU&RPPDQG
&KHFNLI
VXFFHVVIXOO\JUDVSHGDQREMHFW
W
W7
ORWRU&RPPDQG
&ORVH*ULSSHU
'DWDVHW
7'DWD3RLQWV*UDVSLQJ$OJRULWKP
AA 274B | Lecture 9

\( I_0, p_0 \) \hspace{1cm} \( I_1, p_1 \) \hspace{1cm} \cdots \hspace{1cm} \( I_T, p_T \)
```

Stanford University
Grasping Algorithm

Grasp Prediction Network

\[ 0 \leq p \leq 1 \]

Continuous Servoing

\[ \mathcal{U}_t \]

\[ \mathcal{U}_t^* \]
Grasp Prediction Network

Forward Pass

Backward Pass

Cross-Entropy Loss

\[-(y \log(p) + (1 - y) \log(1 - p))\]
Continuous Servoing: Cross-Entropy Method

Current Image + Sample N Motor Commands → Grasp Prediction Network

Probability of Success → Sort

M<N Best Motor Commands → $u^*_t$
Continuous Servoing

\[ I_t \]

**Cross Entropy Method**

\[ \nu_t^* \]

\[ \nu_t = 0 \]

\[ p = g(I_t, 0)/g(I_t, \nu_t^*) \]

- Don’t move, close gripper
- Raise gripper up
- Execute \( \nu_t^* \)

\[ p > 0.9 \]

\[ p < 0.5 \]

else
Video
Overall Performance: Failure Rate Results

Table 1. Failure rates of each method for each evaluation condition. When evaluating without replacement, we report the failure rate on the first 10, 20, and 30 grasp attempts, averaged over 4 repetitions of the experiment. $N$ indicates the number of grasps used to compute each value. The experiments without replacement were repeated four times.

<table>
<thead>
<tr>
<th></th>
<th>First 10 $(N = 40)$</th>
<th>First 20 $(N = 80)$</th>
<th>First 30 $(N = 120)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>67.5%</td>
<td>70.0%</td>
<td>72.5%</td>
</tr>
<tr>
<td>Hand-designed</td>
<td>32.5%</td>
<td>35.0%</td>
<td>50.8%</td>
</tr>
<tr>
<td>Open loop</td>
<td>27.5%</td>
<td>38.7%</td>
<td>33.7%</td>
</tr>
<tr>
<td><strong>Our method</strong></td>
<td><strong>10.0%</strong></td>
<td><strong>17.5%</strong></td>
<td><strong>17.5%</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Failure rate $(N = 100)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>69%</td>
</tr>
<tr>
<td>Hand-designed</td>
<td>35%</td>
</tr>
<tr>
<td>Open loop</td>
<td>43%</td>
</tr>
<tr>
<td><strong>Our method</strong></td>
<td><strong>20%</strong></td>
</tr>
</tbody>
</table>
Discussion

- **End-to-end learning** can achieve good results with **few assumptions**
- It requires **a lot of data** to achieve good performance
  - More tolerable the more **generalizable**
  - Variation in hardware was **small-scale**
## Conclusion: Two Approaches

<table>
<thead>
<tr>
<th></th>
<th>Dex-Net</th>
<th>Arm Farm</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Setup</strong></td>
<td>Single object in simulation</td>
<td>Bin of objects in real world</td>
</tr>
<tr>
<td><strong>Number Data Points</strong></td>
<td>13,000 objects, 2.5M grasps</td>
<td>1,100 objects, 1.7M grasps</td>
</tr>
<tr>
<td><strong>Data Point</strong></td>
<td>(object, grasp, label = probability of success)</td>
<td>(Image, motor command, label = ground truth success)</td>
</tr>
<tr>
<td><strong>Diversity of Objects</strong></td>
<td>Rigid, Opaque</td>
<td>Rigid &amp; deformable, opaque &amp; translucent</td>
</tr>
<tr>
<td><strong>Object Representation</strong></td>
<td>3D Mesh Model</td>
<td>None</td>
</tr>
<tr>
<td><strong>Data Collection Method</strong></td>
<td>Generated in simulation</td>
<td>Self-supervised on real hardware</td>
</tr>
<tr>
<td><strong>Type of Learning</strong></td>
<td>Deep learning, reinforcement learning</td>
<td>End-to-end deep learning</td>
</tr>
</tbody>
</table>
Still Missing

Contact-GraspNet: Efficient 6-DoF Grasp Generation in Cluttered Scenes. Sundermeyer et al. ICRA 2021
Suggested Reading

• **Data-Driven Grasp Synthesis – A survey** by Bohg et al. TRO 2014


• **Dex-Net 2.0: Deep Learning to Plan Robust Grasps with Synthetic Point Clouds and Analytic Grasp Metrics** by Mahler et al.. RSS 2017. [https://berkeleyautomation.github.io/dex-net](https://berkeleyautomation.github.io/dex-net)

• **Learning Hand-Eye Coordination for Robotic Grasping with Deep Learning and Large-Scale Data Collection** by Levine et al. IJRR 2017.

"Dex-Net 2.0: Deep Learning to Plan Robust Grasps with Synthetic Point Clouds and Analytic Grasp Metrics" by Mahler et al.. RSS 2017. [https://berkeleyautomation.github.io/dex-net](https://berkeleyautomation.github.io/dex-net)

Next time

• Interactive Perception