Principles of Robot Autonomy II

Learning-based Approaches to **Manipulation & Interactive Perception**

Jeannette Bohg
Learning Outcome for next four Lectures

Modeling and Evaluating Grasps

Apply Learning to Grasping and Manipulation

Modeling and Executing Manipulation

Use Manipulation to Perceive better
Today’s itinerary

• Recap
  • Learning-based Grasping

• Learning-based Approaches to
  • Grasping
  • Planar Pushing
  • 6D Manipulation Tasks

• Interactive Perception
  • Perception is not passive but active
  • Not seeing, but looking
For a Deeper Dive into Grasping and Manipulation

• CS326 – Topics in Advanced Robotic Manipulation – Fall 2020
How do we generate a grasp?

Grasp Evaluation

Offline database with grasps linked to 3D objects

Offline

Perception

Online

Motion Planning
How do we execute a grasp?

Top-Down & Open-Loop

- monocular RGB camera
- 7 DoF robotic manipulator
- 2-finger gripper
- object bin

Acquiring a grasp + Closed Loop

Grasp Force Optimization
Data-Driven Approaches to Grasping


Grasping previously unseen objects using only 2D images \textit{without} 3D meshes.
Supervised learning pipeline

**TRAINING**

Input:
Labeled 2D images

- Raw data & target
- Feature Engineering
- Training Set
- Validation Set
- Model

- Model training
- Hyperparameters tuning
- Model selection
- Evaluation

**PREDICTING**

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

Solution? Use synthetic data!

2500 images
5 object classes

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

TRAINING

Preprocessing

Raw data & target → Feature Engineering → Training Set → model training → Machine Learning → hyperparameters tuning → model selection → Validation Set → Test Set → evaluation → Model

PREDICTING

New data → Feature Engineering → Predict → Target
Image preprocessing

Edge filters (Y):

Texture filters (Y):

Average filter (Cb/Cr):

6 (edge) + 9 (texture) + 1 (average) * 2 = 17 features per patch
Supervised learning pipeline

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

PREDICTING
- New data
- Feature Engineering
- Predict
- Target
Binary classification task

Is a given pixel \((u, v)\) on the image a grasping point (1) or not (0)?

- \(C_i, i \in [1, N]\): image \(i\);
- \(z_i(u, v) = 1\{u, v \text{ is the projection of a grasping point}\}, i \in [1, N]\).

\[
P(z_i(u, v) = 1|C_i) = P(z_i(u, v) = 1|C_i; \theta)
\]
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!

• But
  • Not very robust in clutter unless trained on this specific scenario
  • Hand-designed features in 2D
  • Simulated sensory data
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.

Dex-Net 2.0: Deep Learning to Plan Robust Grasps with Synthetic Point Clouds and Analytic Grasp Metrics

At test time: \( \pi_\theta (y) = \text{argmax}_{u \in \mathcal{C}} Q_\theta (u, y) \) 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
Monted 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

1.7M Grasp Attempts

800k grasp attempts
2 months

900k grasp attempts
4 months
Grasping Algorithm

\[ 0 \leq p \leq 1 \]

\[ V_t \rightarrow \text{Grasp Prediction Network} \]

\[ \text{Continuous Servoing} \rightarrow V_t^* \]
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

\[ \nu_t^* \]

\[ \nu_t = 0 \]

GPN

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

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>Without replacement</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>With replacement</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>

- Struggled with clutter
- Unable to react to objects moving
- Performs better and requires fewer assumptions
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>
Today’s itinerary

• Recap
  • Learning-based Grasping

• Learning-based Approaches to
  • Grasping
  • Planar Pushing
  • 6D Manipulation Tasks

• Interactive Perception
  • Perception is not passive but active
  • Not seeing, but looking
Controlling through contact
Controlling through contact

Better Models
Better Feedback
Predictive Model

Model

Predictive Effect

Sensory Observations

Action

Model—Predictive Control
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.

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

IROS 2016. "More than a Million Ways to Be Pushed: A High-Fidelity Experimental Dataset of Planar Pushing" by Peter Yu and Maria Bauza.
Predicting physical effect
Keep Predicting
Bias-Variance Tradeoff

- Bias
- Variance
- Min Error

Model Complexity

Error
Model-based
Bias-Variance Tradeoff

Error

Bias

Model-based

Model Complexity

Variance

Min Error
Data-Driven
Bias-Variance Tradeoff

Error

Bias

Learning

Model-based

Min Error

Model Complexity

Variance

Structure

Data-Driven
Hypothesis

Physics Models + Learning = Generalization

Example:

Real motion

Simulated motion (Lynch 1992)

BIG difference!
Hybrid Model

Sensory Observations → Learned Model → Parameters → Physics-based Model → Predicted Effect

End-to-End Loss on Effect
Testing Hypothesis on a Case Study

Analytic Model

Relation between wrench cone, limit surface and unit twist sphere. Adopted from Chapter 37, Fig 37.10 in Springer Handbook of Robotics.
Compared Architectures

Raw Sensory Observations

**Neural** Network only

![Neural Network Diagram]

- Sensory Observations
- Action
- Learned Model
- Predicted Effect

**Hybrid** Model

![Hybrid Model Diagram]

- Sensory Observations
- Action
- Learned Model
- Parameters
- Physics-based Model
- Predicted Effect

**Training**: End-to-End

**Loss**: Error between Predicted and Ground Truth Effect
Advantages

Data Efficiency  Extrapolation
Testing Data Efficiency

---

trans [%] vs. thousand training examples

---

physics
Testing Data Efficiency

The graph shows the performance of different methods over varying numbers of training examples. The y-axis represents the transformation rate in percentage, and the x-axis shows the number of training examples in thousands. Two lines are plotted: one for 'hybrid' methods and another for 'physics' methods. The hybrid methods show a gradual decline in transformation rate as the number of training examples increases, while the physics methods maintain a constant rate.
Testing Data Efficiency

Training = Test Distribution

Graph showing the relationship between training data efficiency and the number of thousand training examples, with curves for neural, hybrid, and physics models.
Testing Generalization

New Pushing Angles & Contact Points

New Push Velocities

New Object Shapes

Interpolation

Extrapolation

Generalization to new push velocities
Generalization to new push velocities

Extrapolation

- **Training**
- **Testing**
Generalization to new push velocities

Training = Test Distribution
Don’t throw away structure
Learn to extract given state representation from raw data
A Concrete Suggestion

Sensory Observations

Learned Model

Parameters

Physics-based Model

Predicted Effect

End-to-End Loss on Effect
Interpretability
Today’s itinerary

• Recap
  • Learning-based Grasping

• Learning-based Approaches to
  • Grasping
  • Planar Pushing
  • 6D Manipulation Tasks

• Interactive Perception
  • Perception is not passive but active
  • Not seeing, but looking
Models will never be perfect
Human multimodal sensor–motor coordination

Sensory Inputs

Example of Task–Relevant Information

pose  geo  force  friction
Human multimodal sensor–motor coordination

- Sensory Inputs
  - pose
  - geo
  - force
  - friction

- Example of Task–Relevant Information

- Action
  - \( \pi \)
Human multimodal sensor–motor coordination

Sensory Inputs

Example of Task–Relevant Information

pose  geo  force  friction

Action

π
Generalizable representations for multimodal inputs
Generalizable representations for multimodal inputs

\[ \pi(f(o_1, o_2, o_3 \ldots o_n)) = a \]

- multimodal sensory inputs
- Learn representation
  - pose
  - geo
  - force
  - friction
- learn policy for new task instances
Experimental setup

Multimodal sensory inputs

RGB  force/torque  robot states
Experimental setup

Peg geometry

Training

Testing
Learning generalizable representation

Inputs
- RGB image
- Force data
- Robot state

Encoder

Decoder

Representation
Learning generalizable representation

Inputs

RGB image

Force data

Robot state

Fusion

Representation

Decoder
Learning generalizable representation

Inputs
- RGB image
- Force data
- Robot state

Fusion

Representation

Decoder
- pose
- geo
- friction

Task-relevant labels

EXPENSIVE
Collect labeled data with self-supervision
Dynamics prediction from self-supervision

Inputs
- RGB image
- Force data
- Robot state

Representations
- Encoder
- Decoders

Outputs
- Action-conditional optical flow
Dynamics prediction from self-supervision

**Inputs**
- RGB image
- Force data
- Robot state

**Encoder**
- robot action

**Representation**

**Decoders**
- Action-conditional optical flow

**0 / 1 contact in the next step?**
Concurrency prediction from self-supervision

Inputs
- RGB image
- Force data
- Robot state

Encoder
- Robot action

Representation
- 0 / 1 contact in the next step?
- 0 / 1 time-aligned?

Decoders
- Action-conditional optical flow

Representation
- 0 / 1 contact in the next step?
- 0 / 1 time-aligned?
Concurrency prediction from self-supervision

Inputs
- RGB image
- Force data
- Robot state

Encoders
- robot action

Representation

Decoders
- Action-conditional optical flow
  - 0 / 1 contact in the next step?
- 0 / 1 time-aligned?
Concurrency prediction from self-supervision

Inputs
- RGB image
- Force data
- Robot state

Encoder
- robot action

Representation
- 0 / 1 contact in the next step?

Decoders
- Action-conditional optical flow

Output
- yes, time aligned
Concurrency prediction from self-supervision

Unpaired Inputs
- RGB image
- Force data
- Robot state

Encoder
- Robot action

Representation

Decoders
- Action-conditional optical flow
- 0 / 1 contact in the next step?
- 0 / 1 time-aligned?
Concurrency prediction from self-supervision

Unpaired Inputs
- RGB image
- Force data
- Robot state

Decoders
- Action-conditional optical flow
- 0 / 1 contact in the next step?
  - no, not time aligned!

Encoder

Representation

robot action
Learning sample efficient policies

Inputs

RGB image

Force data

Robot state

Encoder

Representation
How to efficiently learn a policy?

Inputs

- RGB image
- Force data
- Robot state

Encoder

Freeze 500k parameters

Representation

RL Policy
How to efficiently learn a policy?

Inputs
- RGB image
- Force data
- Robot state

Encoder
- Freeze 500k parameters
- Learn 15k parameters

Representation

RL Policy
We evaluate our representation with policy learning

Episode 0
0% success rate

Episode 100
21% success rate
We efficiently learn policies in 5 hours

Episode 300
73% success rate

Episode 300
71% success rate

Episode 300
92% success rate
Our multimodal policy is robust against sensor noise

1. Force Perturbation
2. Camera Occlusion
Camera Occlusion

Agent View

x1
How is each modality used?

Simulation Results
(Randomized box location)

<table>
<thead>
<tr>
<th>Modality</th>
<th>Success Rate (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Force Only</td>
<td>1</td>
</tr>
<tr>
<td>Image Only</td>
<td>49</td>
</tr>
<tr>
<td>Force &amp; Image</td>
<td>77</td>
</tr>
</tbody>
</table>

**Force Only**: Can’t find box

**Image Only**: Struggles with peg alignment

**Force & Image**: Can learn full task completion
Does our representation generalize to new geometries?
Does our representation generalize to new geometries?

92% Success Rate

Tested on

Representation

Policy
Does our representation generalize to new policies?

92% Success Rate

Tested on

Representation

Policy

62% Success Rate

Tested on

Representation

Policy

Policy does not transfer
Does our representation generalize?

92% Success Rate

Policy does not transfer

Tested on

Representation

Policy

62% Success Rate

Tested on

Representation

Policy

92% Success Rate

Representation transfers

Tested on

Representation

Policy
Overview of method

Self-supervised data collection

\(0_{\text{RGB}}, 0_{\text{force}}, 0_{\text{robot}}\)

100k data points
90 minutes

Representation learning

\(f(0_{\text{RGB}}, 0_{\text{force}}, 0_{\text{robot}})\)

20 epochs on GPU
24 hours

Policy learning

\(\pi(f(\cdot)) = a\)

Deep RL
5 hours

Lessons Learned

1. **Self-supervision** gives us rich learning objectives

2. Representation that captures **concurrency** and **dynamics** can **generalize** across task instances

3. Our experiments show that multimodal representation leads to **learning efficiency** and **policy robustness**
State Representation - Physically Meaningful or Learned?

Explicit Representation

- Sensory Observations
- Action
- Learned Model
  - Parameters
  - Physics-based Model
  - Predicted Effect

Learned Representation

- Predicted Effect
- Model
- Action
- Sensory Observations
Interactive Perception

\[ S \times A \times t \]

Sensory Data \quad Actions \quad Time

Exploiting Multi-Modality

J. J. Gibson (1966) – The Senses considered as a Perceptual System.
Concurrency of Motion and Sensing

Held and Hein (1963). Movement-Produced Stimulation in the Development of Visually-Guided Behaviour
Accumulation over Time

Thanks to Octavia Camps at Northeastern University, Boston
Interactive Perception

Selfsupervised Learning of a Multimodal Representation

Concurrency prediction from self-supervision

Inputs
- RGB image
- Force data
- Robot state

Robot action

Encoder

Representation

Decoders
- Action-conditional optical flow
- 0 / 1 contact in the next step?
- 0 / 1 time-aligned?

Interactive Perception

Self-supervised Learning of a Multimodal Representation

Exploiting RGB, Depth and Motion for Instance Segmentation
Instance segmentation from a robotics viewpoint.
Unknown number of rigid objects

Time

Data Set of Frame Pairs

Qualitative Results - Synthetic

<table>
<thead>
<tr>
<th>RGB Inputs</th>
<th>Scene Flow</th>
<th>Segmentation</th>
<th>Ground Truth</th>
</tr>
</thead>
<tbody>
<tr>
<td>frame $I^{t-1}$</td>
<td>PD-flow</td>
<td>OurC+vL+Rig</td>
<td>Ground Truth</td>
</tr>
<tr>
<td>frame $I^t$</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Qualitative Results - Real

<table>
<thead>
<tr>
<th>RGB Inputs</th>
<th>Scene Flow</th>
<th>Segmentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>frame $I^{t-1}$</td>
<td>PD-flow</td>
<td>HOMC</td>
</tr>
<tr>
<td>frame $I^t$</td>
<td>OurC+vL+Rig</td>
<td>OurC+vL+Rig</td>
</tr>
</tbody>
</table>

Interactive Perception

Exploiting RGB, Depth and Motion for Instance Segmentation
Interactive Perception

Homework 3 - Problem 3 Learning Intuitive Physics
Conclusions

• Interaction generates a rich sensory signal that eases perception.
• Action-conditional, multi-modal representation helps contact-rich manipulation and generalizes.
Suggested Reading

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


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


• “Combining learned and analytical models for predicting action effects”, Kloss et al, IJRR 2020.

Next time

Guest Lecture by Jeremy Ma

Teaching Robots in the Home

In this talk, we present the latest research work from the Toyota Research Institute on robotic assistance in the home. Motivated by the growing problem of an ageing society, a mobile manipulator robot is demonstrated that can be easily taught by human demonstration in virtual reality to achieve complex tasks. We show our latest results of real tasks executed in various homes around the Bay Area and discuss the future of where Toyota is headed with this research.