Robot Learning

Sim-to-real transfer

USC University of Southern California

LiraLab

Remember…

$$\underset{\pi}{\text{maximize}} \quad \mathbb{E}_{\boldsymbol{w}}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right]$$

$$\text{subject to } s_t = f(s_t, a_t, w_t)$$
$$a_t = \pi(s_t)$$

Where does this come from?

Inverse reinforcement learning (IRL)

- Kalman, 1964: Inverse optimal control for 1D problems
- Boyd et al., 1994: Linear matrix inequality (LMI) for LQ setting
- Ng, Russel, 2000: First MDP formulation and reward ambiguity
- Abbeel, Ng, 2004: Apprenticeship learning (feature matching)
- Ratliff et al., 2006: Max margin planning (MMP)
- Ziebart et al., 2008: Max-Ent IRL

Remember…

$$\underset{\pi}{\text{maximize}} \quad \mathbb{E}_{\boldsymbol{w}}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right]$$

$$\text{subject to } s_t = f(s_t, a_t, w_t)$$
$$a_t = \pi(s_t)$$

Where does this come from?

Some methods require full knowledge of f whereas some require only ability to execute/simulate it.

Why do we need simulations?

- Robots are expensive.

- Robots break and degrade all the time.
  … and they will likely break more if you try to train things on them.

- Robots are slow.

- Labeling real world is difficult.

The premise of robot learning

Designing controllers for robots is difficult and does not scale well. Instead, we will collect a lot of experience and let the algorithm handle the rest.

Collecting a lot of experience

OpenAI GPT-3

IMAGENET

45 TB of text

(Brown et al. 2020)

Language model that produces human-like texts

14,000,000 images

(Deng et al. 2009)

Image recognition models at human-level proficiency

44,000,000 chess games

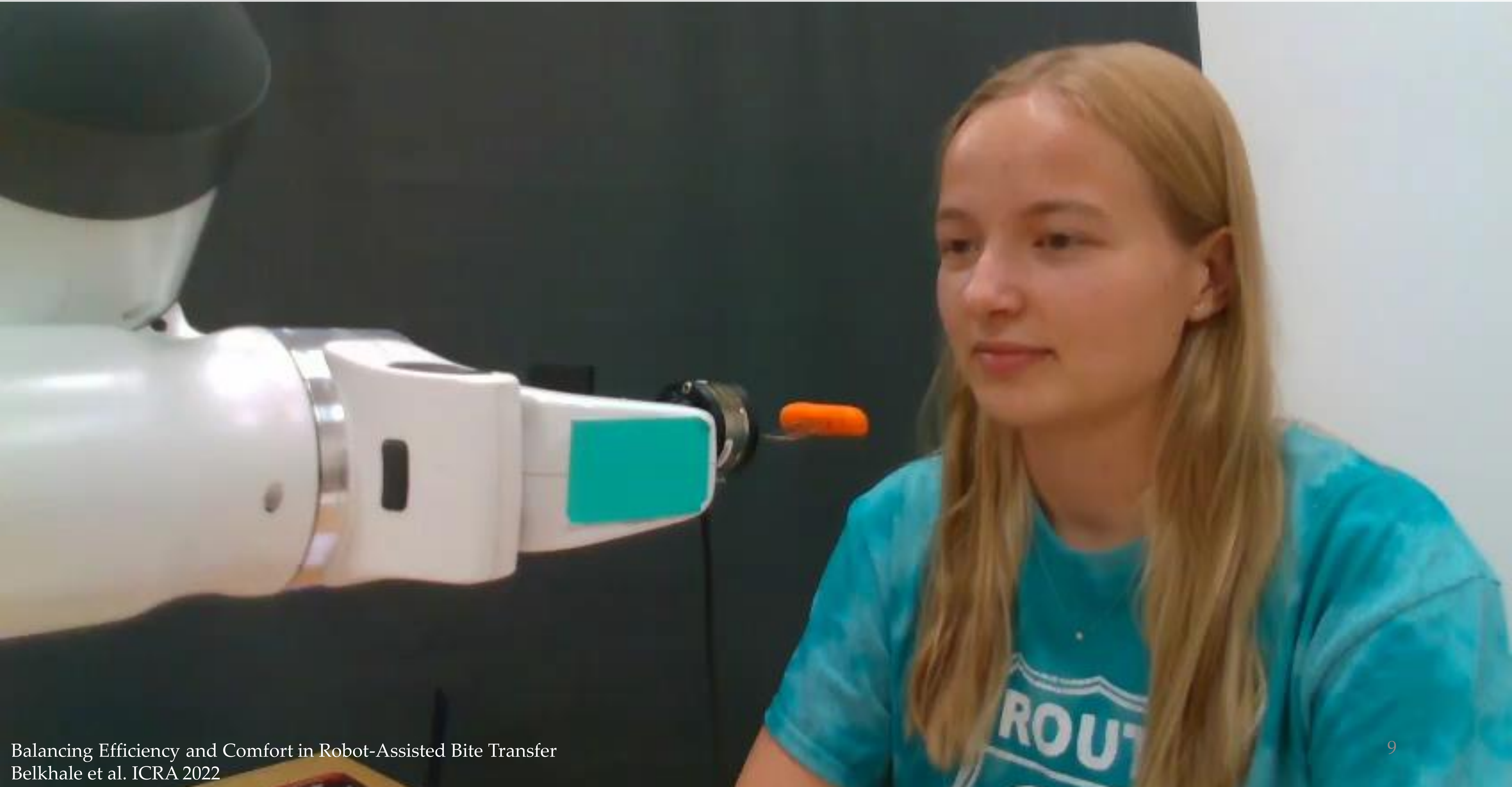
(Silver et al. 2017)

Super-human chess engines

We do not have large datasets in robotics

Learning Hand-Eye Coordination for Robotic Grasping with Deep Learning and Large-Scale Data Collection
Levine et al., IJRR 2018

Data collection is more expensive and safety-critical when humans are involved



Balancing Efficiency and Comfort in Robot-Assisted Bite Transfer
Belkhale et al. ICRA 2022

This lecture is based on:

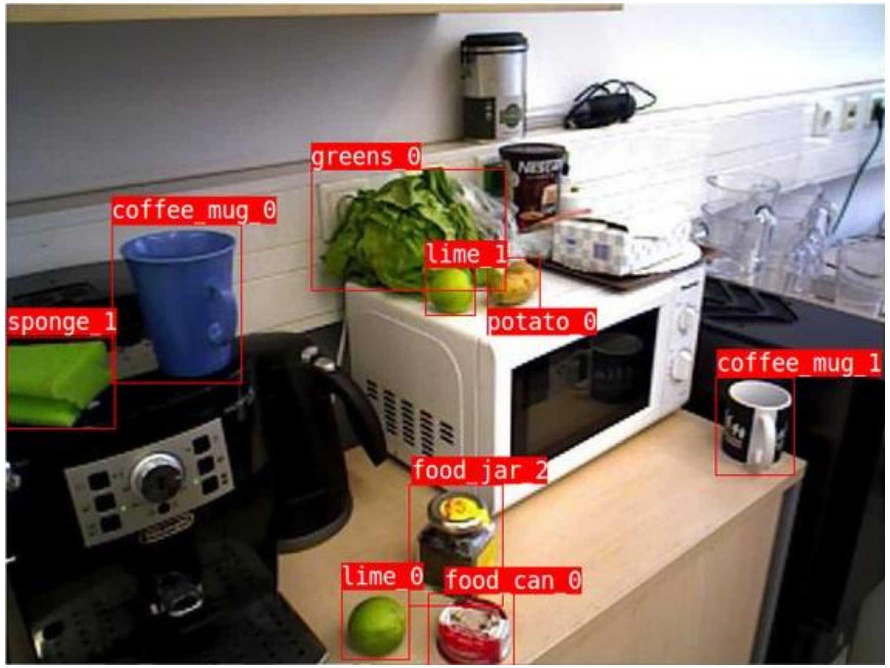
"Randomization and the reality gap: how to transfer robotic policies from sim to real" by Josh Tobin:

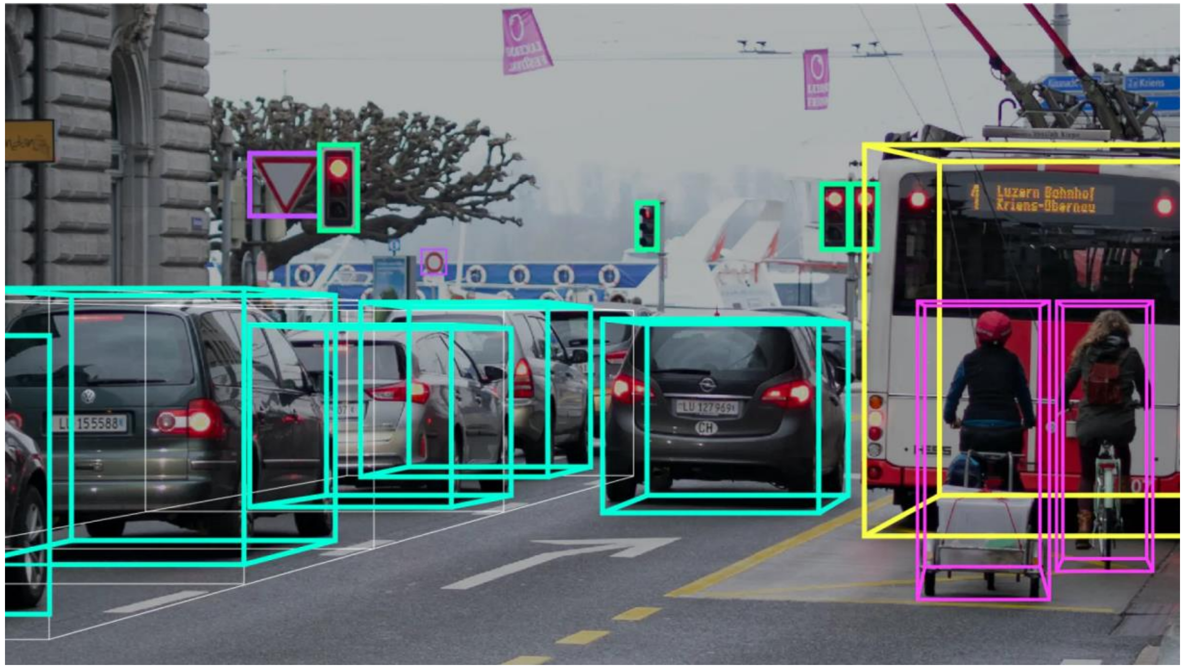
https://youtu.be/ac_W9IgKX2c

Simulated data

- Cheap
- Fast
- Scalable
- Safe
- Labeled
- Not beholden to real-world probability distributions

Labels (and rewards) are free





Left: Recognizing Objects In-the-wild: Where Do We Stand?
Loghmani et al., ICRA 2017
Right: From Caroline Lasorsa (Superb AI)

Not beholden to real-world distributions

From: 7NEWS Adelaide


Swindon's Magic Roundabout from the air
Mark Winter, 2016

Meskel Square, Addis Abeba
Rift Valley Expeditions, 2012

Not beholden to real-world distributions

Scalable End-to-End Autonomous Vehicle Testing via Rare-event Simulation
O'Kelly et al., NeurIPS 2018

Sim-to-real problem

There is a real danger (in fact, a near certainty) that programs which work well on simulated robots will completely fail on real robots because of the differences in real world sensing and actuation – it is very hard to simulate the actual dynamics of the real world.

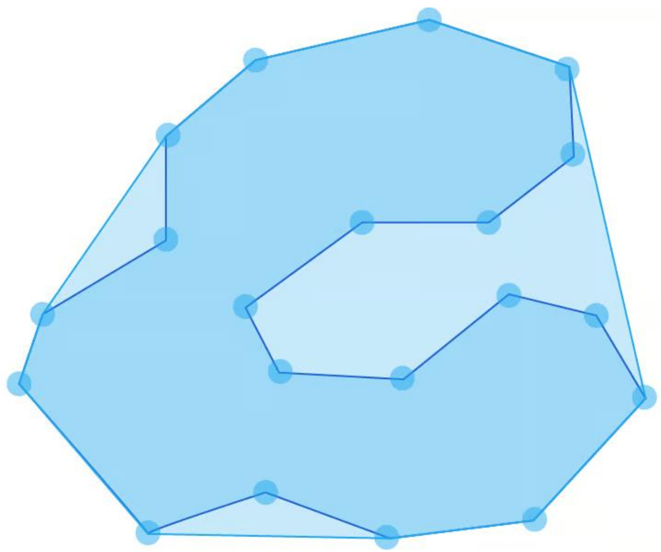
Artificial Life and Real Robots
Rodney Brooks, 1992

Today

- Difficulty of using simulated data
- Using simulation data without solving sim-to-real
- Building simulations
- Domain adaptation
- Domain randomization

Difficulty of using simulated data

Physics simulators make big assumptions to run faster

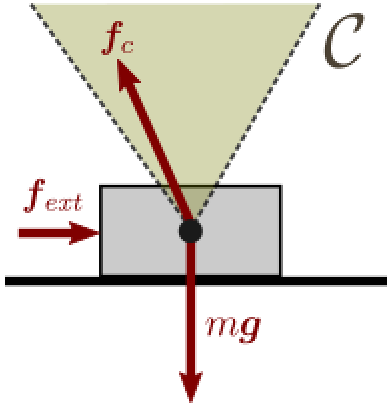


Convex objects

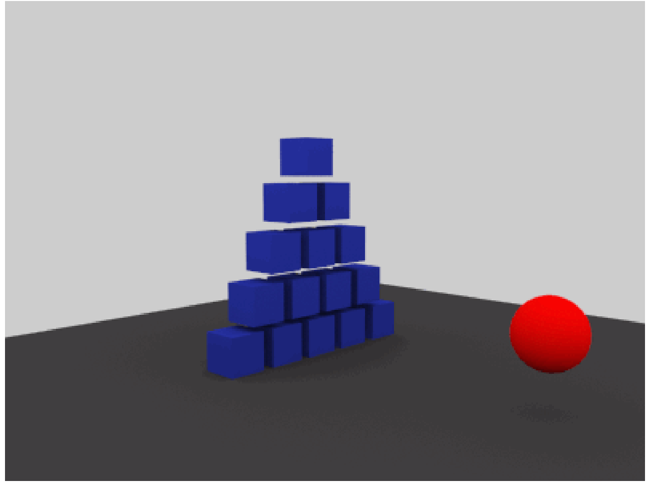
Discrete time

Difficulty of using simulated data

Physics simulators make big assumptions to run faster



Coulomb friction

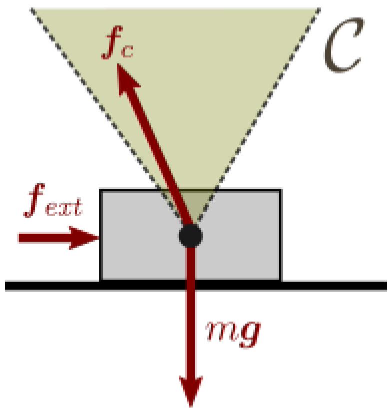


Rigid bodies

Left: Stephane Caron (Inria Paris)
Right: Cheetah3D

Difficulty of using simulated data

Physics simulators make big assumptions to run faster



What is the friction coefficient?

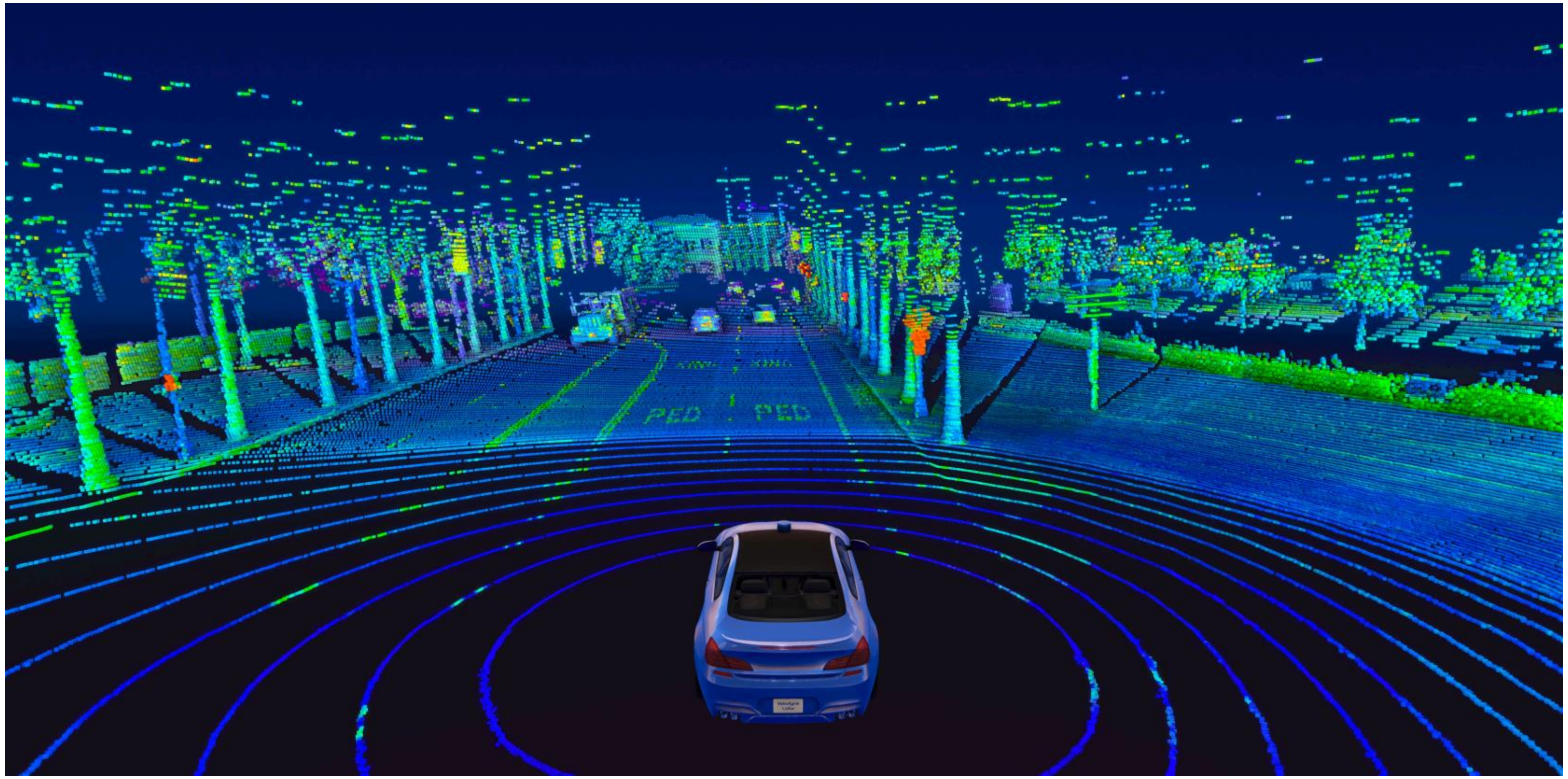
How about other parameters?
Inertia? Damping? Spring constants?

More accurate model ⇒ More parameters to learn ⇒ More data needed

Left: Stephane Caron (Inria Paris)

Difficulty of using simulated data

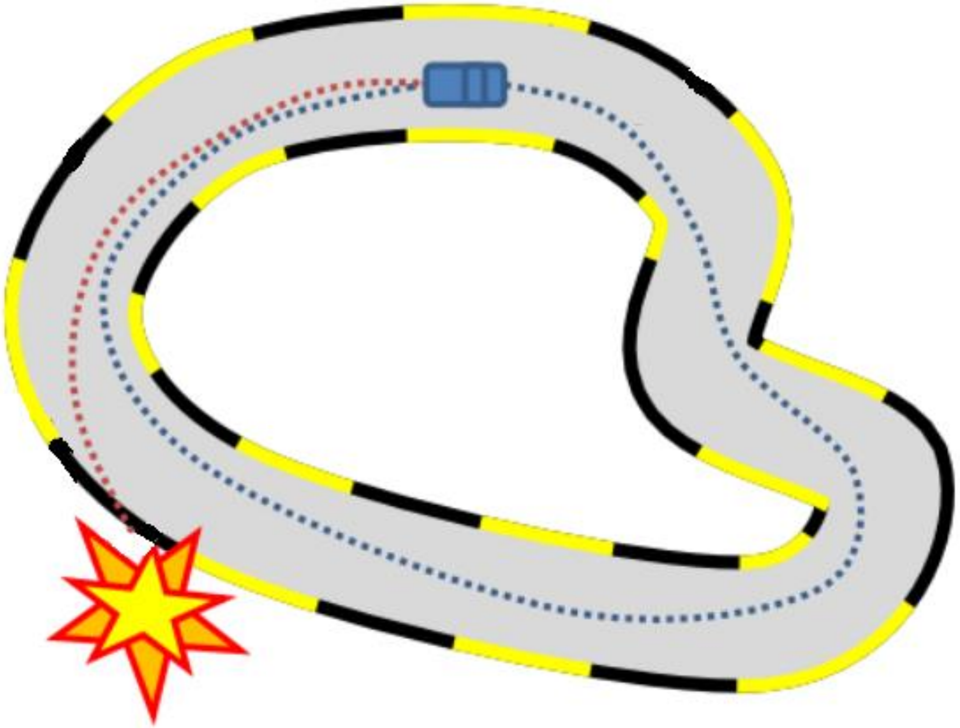
Photorealistic simulation is expensive.



A Guide to Lidar Wavelengths for Autonomous Vehicles and Driver Assistance
Velodyne, 2018

Difficulty of using simulated data

Remember…



Similar problem in sim-to-real:

Small modeling errors cause large control errors.

Difficulty of using simulated data

Neural nets will exploit/overfit to differences in data distributions

Multi-object tracking accuracy:

Sim: 63.7%

Real: 78.1%

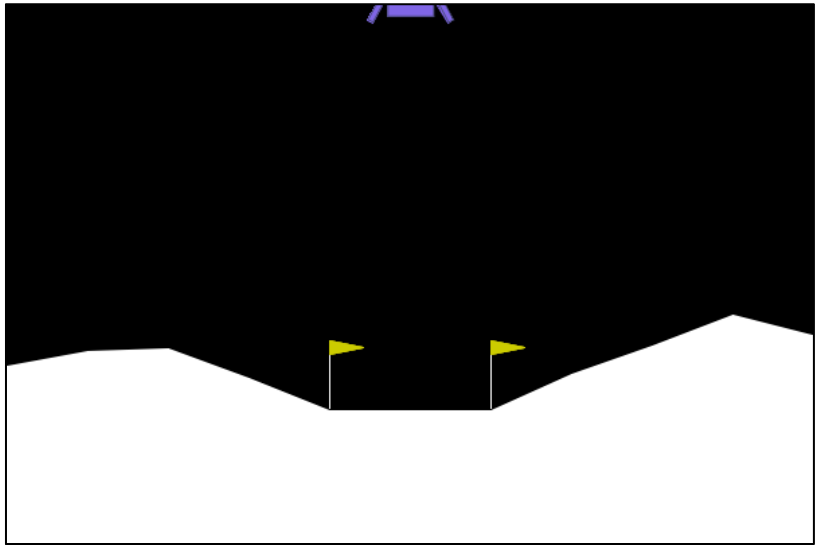
Virtual Worlds as Proxy for Multi-Object Tracking Analysis
Gaidon et al., CVPR 2016

Today

- Difficulty of using simulated data
- Using simulation data without solving sim-to-real
- Building simulations
- Domain adaptation
- Domain randomization

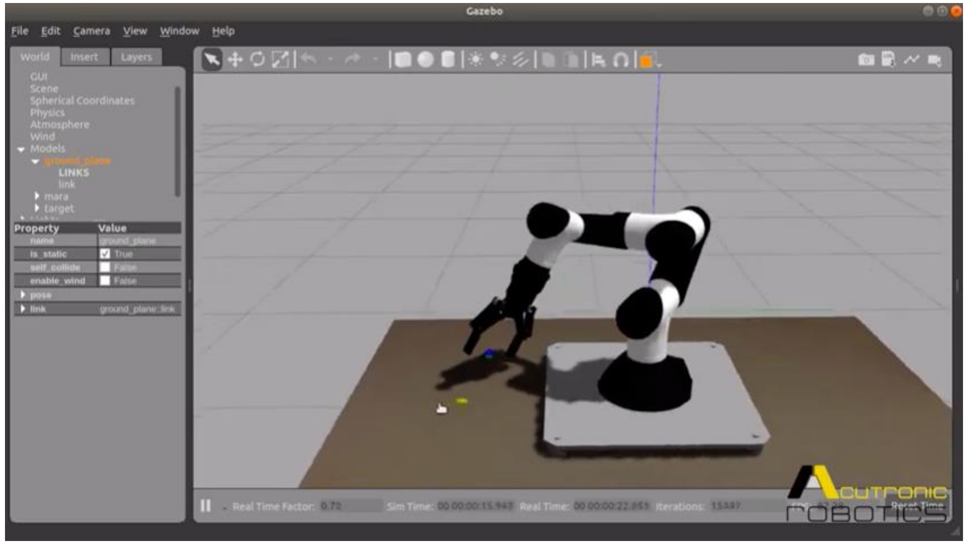
Sim data without solving sim-to-real

Prototyping Algorithms



Verify/compare algorithms in simulation

Debugging

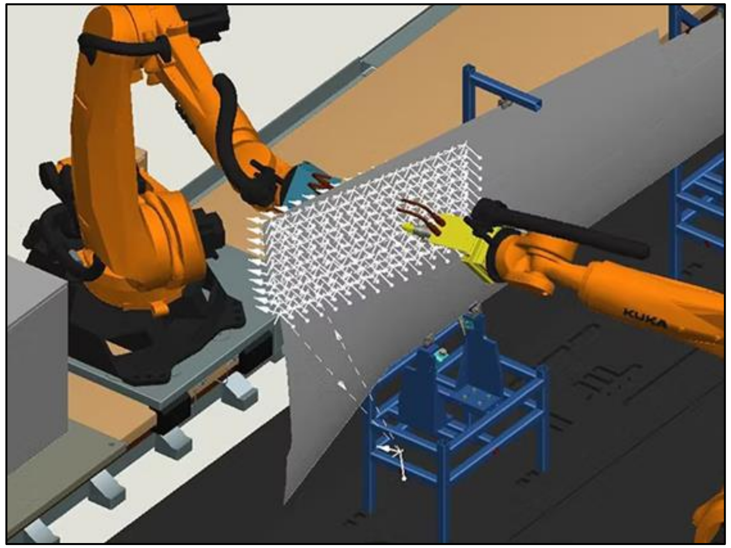


Replace the robot with a simulator to debug the full stack

Left: OpenAI Gym
Brockman et al., 2016
Right: Robotic Arm Simulation with ROS and Gazebo
Dineshkumar (Skyfi Labs)

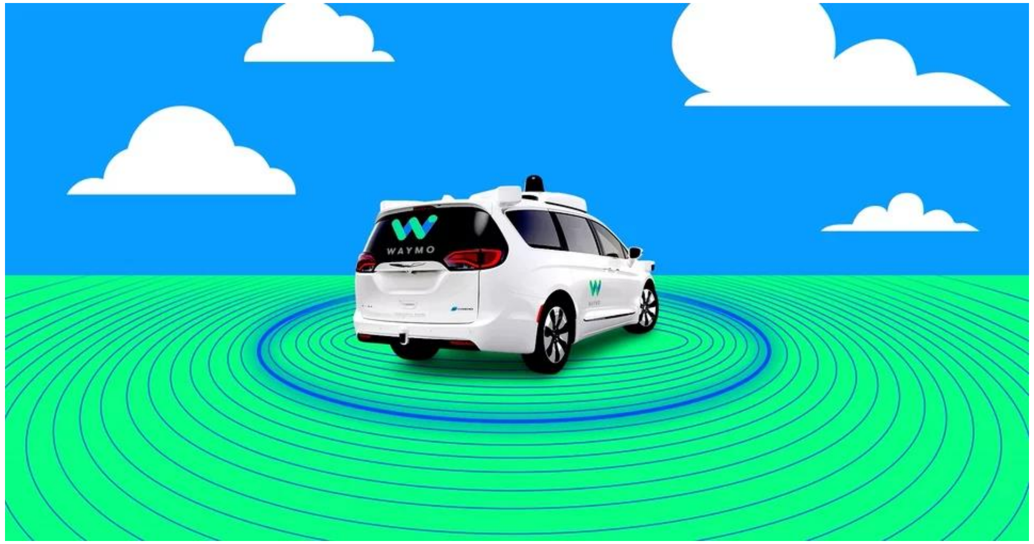
Sim data without solving sim-to-real

Prototyping Systems



Choose the robot / verify its ability

Testing



Test the performance in edge cases, etc.
Waymo: 1000x testing in simulation than real-world (2017)

Left: Siemens Tecnomatix
Right: Inside Waymo's Secret World for Training Self-Driving Cars
Madrigal, The Atlantic (2017)

Today

- Difficulty of using simulated data
- Using simulation data without solving sim-to-real
- Building simulations
- Domain adaptation
- Domain randomization

Building simulations

1. Design simulation model
   - This is where we implement physics.
   - In practice, we pick an existing model, e.g., MuJoCo, PyBullet, Gazebo.

2. Create scenarios
   - We create 3D models, or get them: ShapeNet, YCB, Dex-Net, Unity, …
   - We then create a scenario (e.g., decide where to place the objects)

3. Collect data and potentially improve simulation

This is "System ID"

System ID

$$\min_{\eta} \sum_{a} D\big(\tau_\eta(a), \tau_r(a)\big)$$

Simulation parameters

Action sequence

Trajectory from following a in simulation

Trajectory from following a in the real world

Building simulations

1. Design simulation model
   - This is where we implement physics.
   - In practice, we pick an existing model, e.g., MuJoCo, PyBullet, Gazebo.

2. Create scenarios
   - We create 3D models, or get them: ShapeNet, YCB, Dex-Net, Unity, …
   - We then create a scenario (e.g., decide where to place the objects)

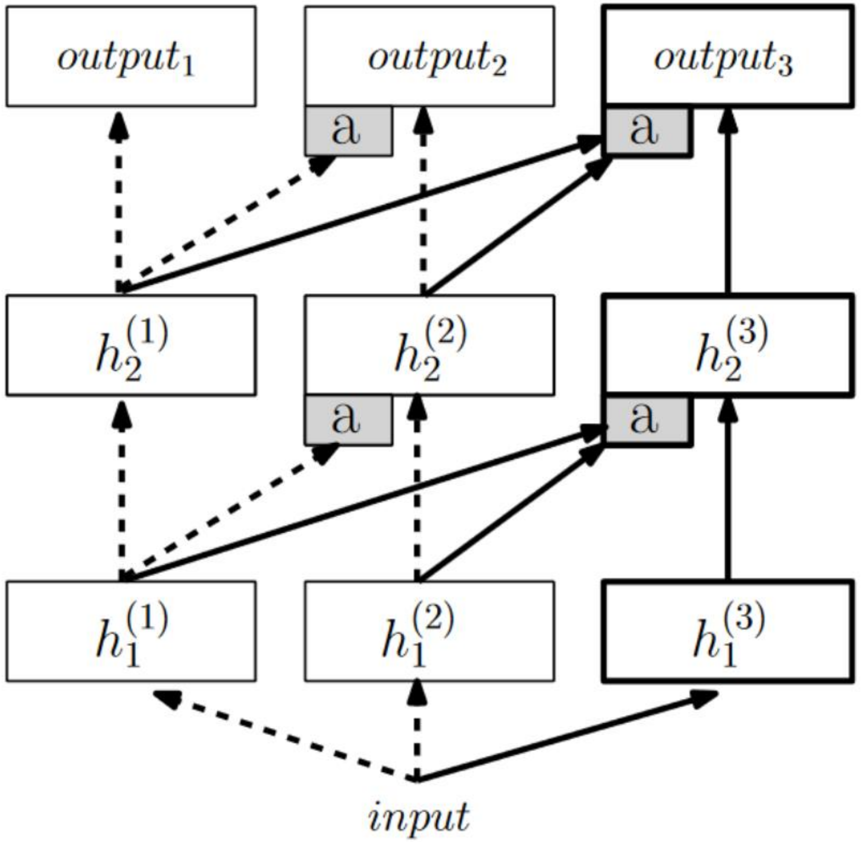
3. Collect data and potentially improve simulation

Today

- Difficulty of using simulated data
- Using simulation data without solving sim-to-real
- Building simulations
- Domain adaptation
- Domain randomization

Supervised domain adaptation

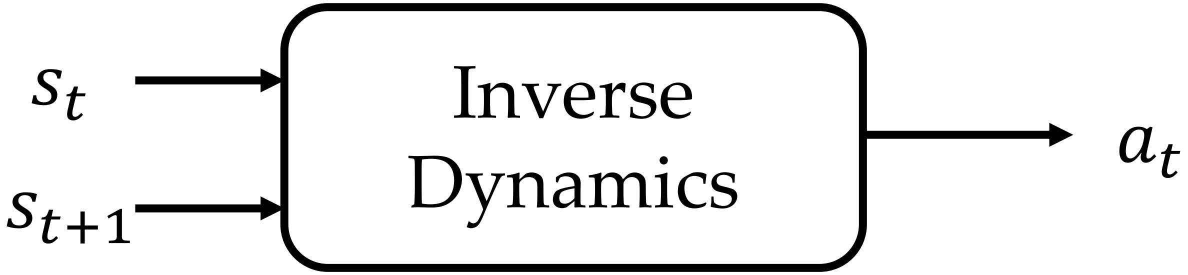
Think of each simulation as a new task and train a progressive neural network.



Progressive neural networks
Rusu et al., 2016

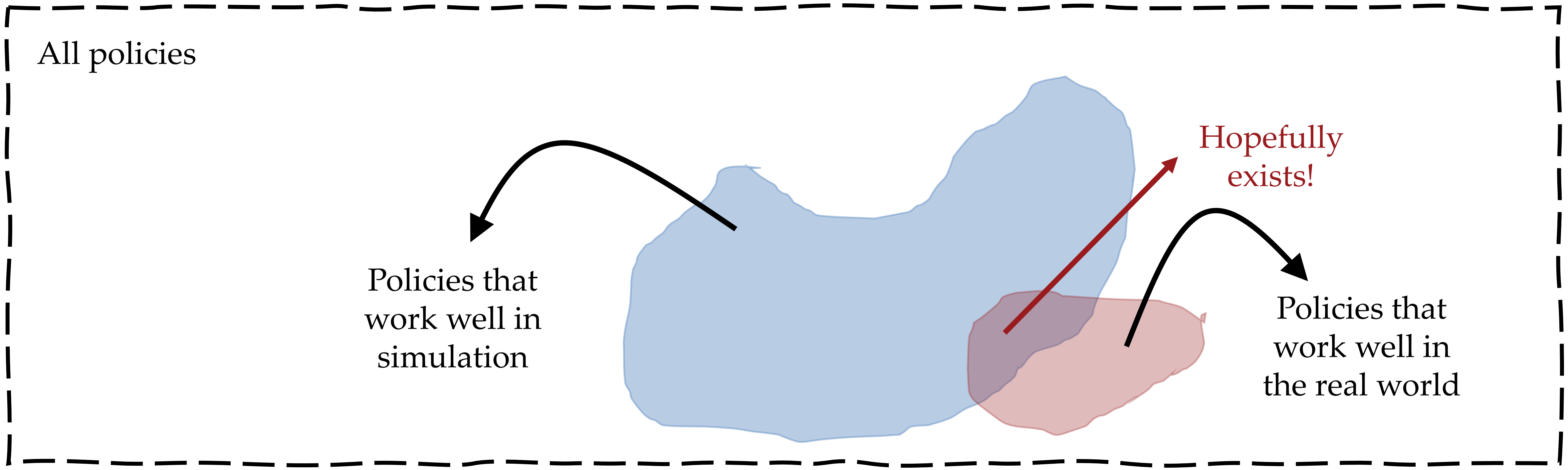
Supervised domain adaptation

Learn inverse dynamics over a set of simulations.



Supervised domain adaptation

Train in simulation to find a submanifold of the policy space or learn a Bayesian prior that may perform well in the real world.



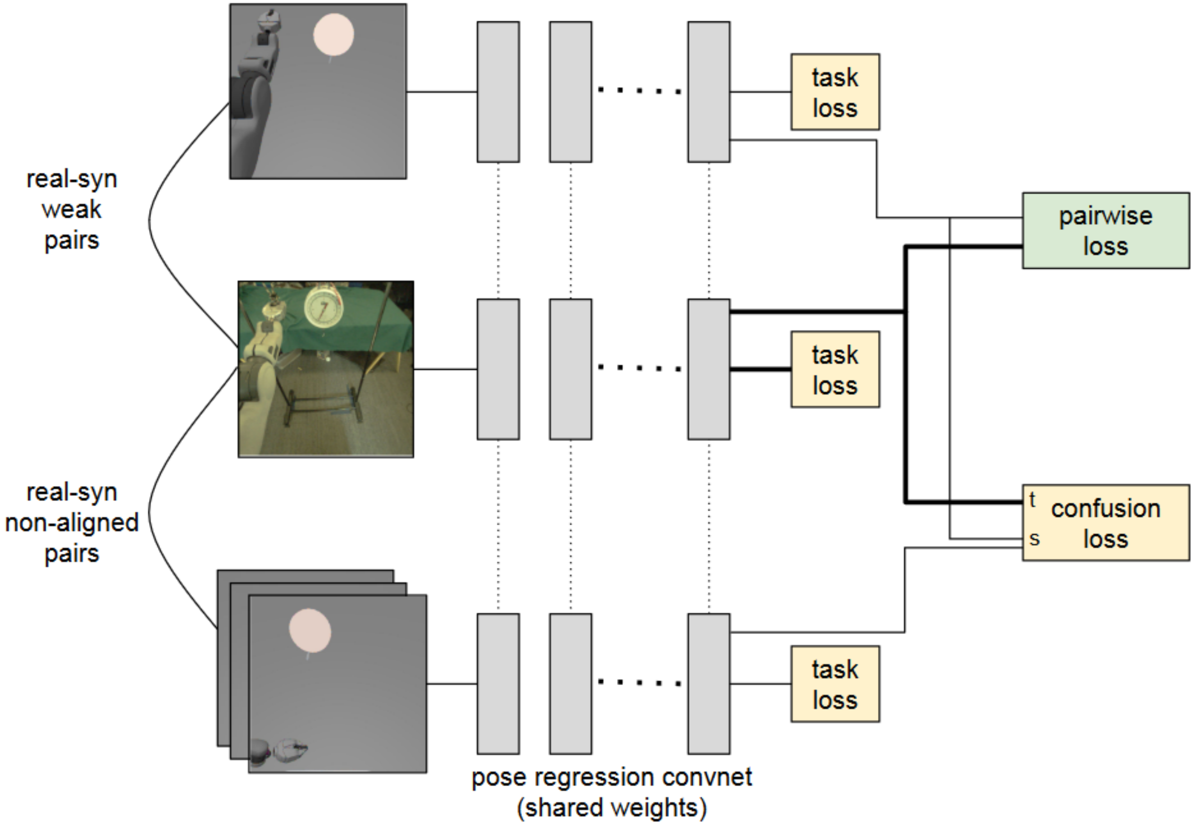
Weakly supervised domain adaptation

Use weak supervision to learn policies that are robust to distribution shift.

Task loss: our actual objective

Confusion loss: objective for classifying sim vs. real

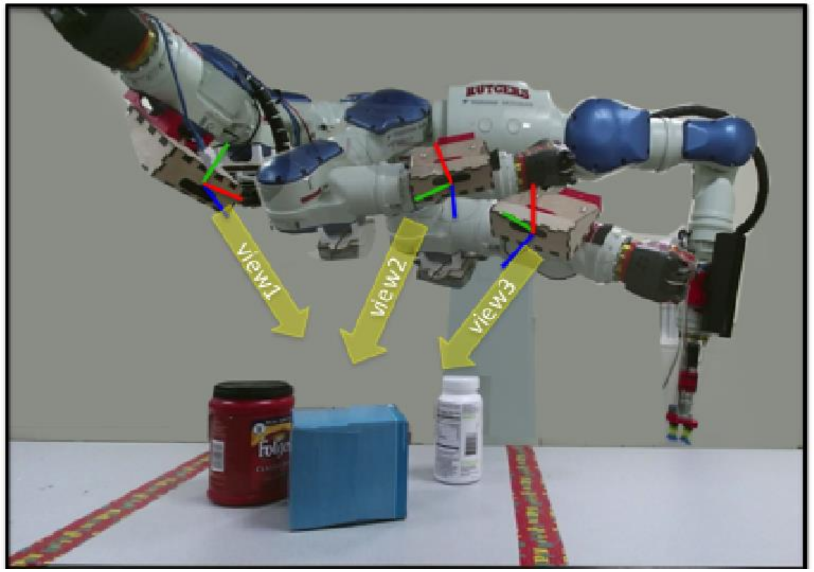
Pairwise loss: objective for aligning states/frames

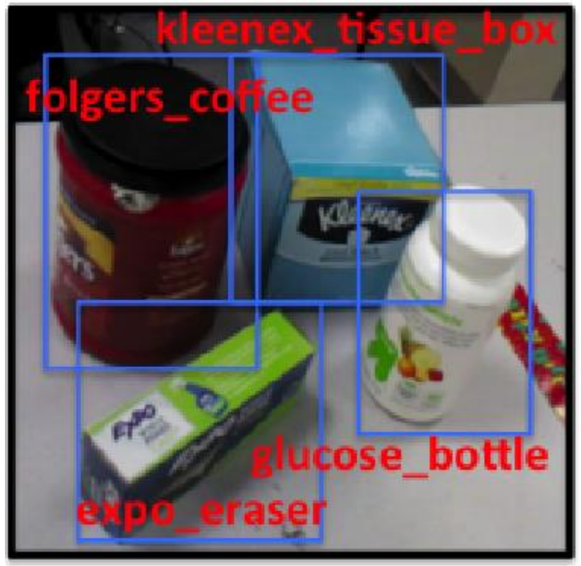


Adapting Deep Visuomotor Representations with Weak Pairwise Constraints
Tzeng et al., WAFR 2020

Self-supervised domain adaptation

Use a model trained on simulation to label real world data.
Bootstrapping with such self-supervised labels helps adaptation.

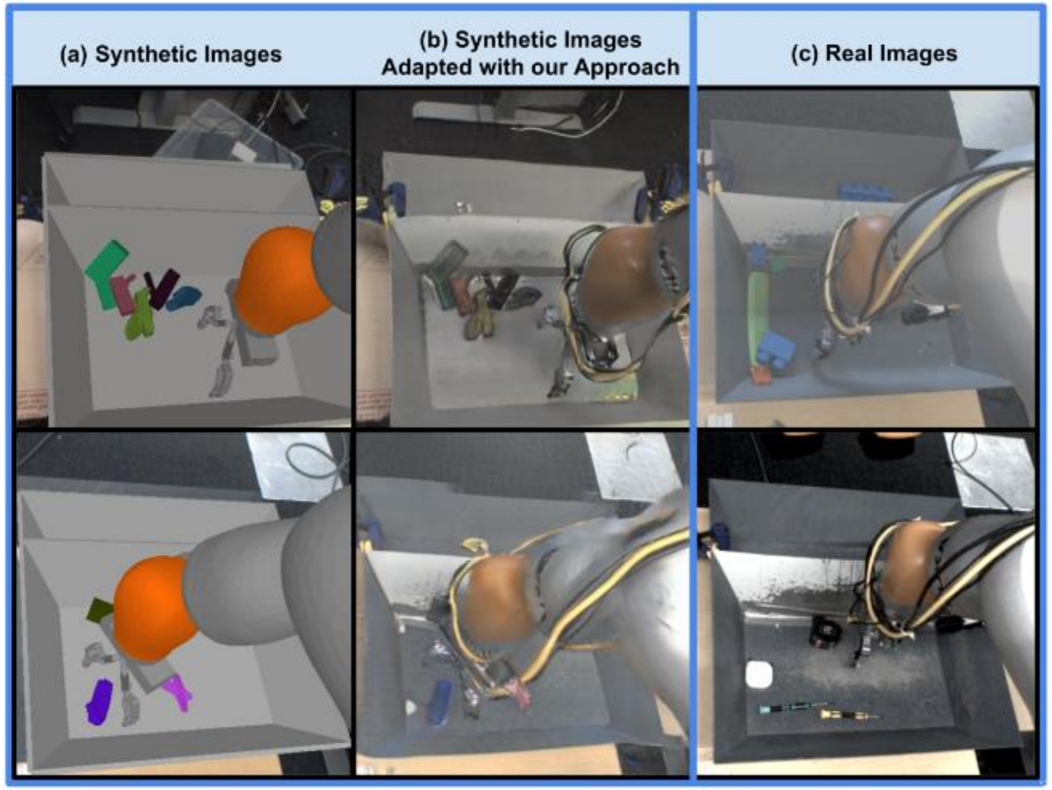


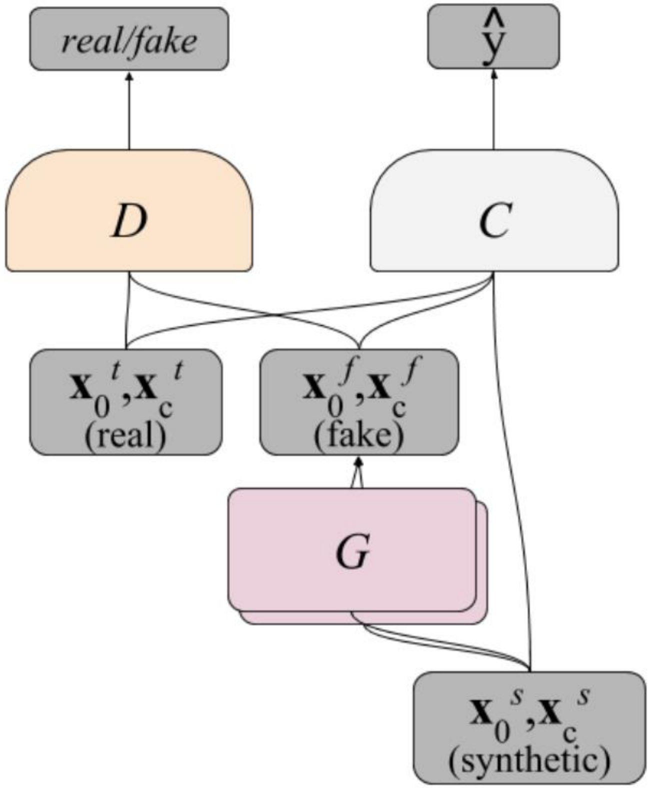


A Self-supervised Learning System for Object Detection using Physics Simulation and Multi-view Pose Estimation
Mitash et al., IROS 2017

Unsupervised domain adaptation

Train a GAN to convert labeled simulation data into realistic data.





Using simulation and domain adaptation to improve efficiency of deep robotic grasping
Bousmalis et al., ICRA 2018

Today

- Difficulty of using simulated data
- Using simulation data without solving sim-to-real
- Building simulations
- Domain adaptation
- Domain randomization

Domain randomization

Idea: Increase the diversity in simulation domains so that the real world may look like another simulator.

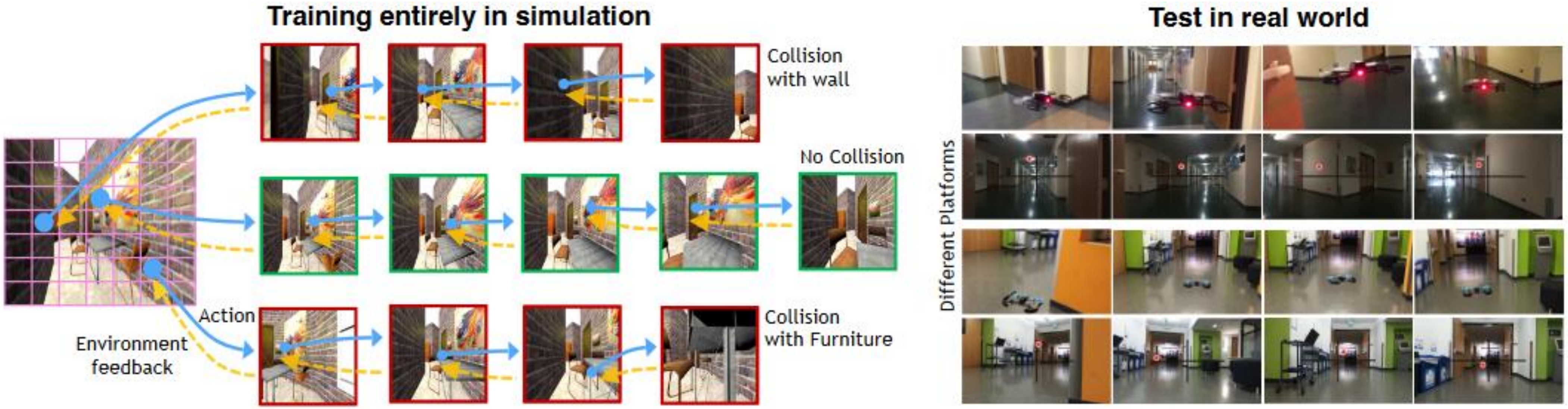
This idea goes back to 1997:

- randomize the important aspects a bit for robustness
- randomize the other aspects so that the controller will ignore them

Evolutionary Robotics and the Radical Envelope-of-Noise Hypothesis
Jakobi, Adaptive Behavior 1997

Domain randomization

CAD²RL for quadcopter collision avoidance



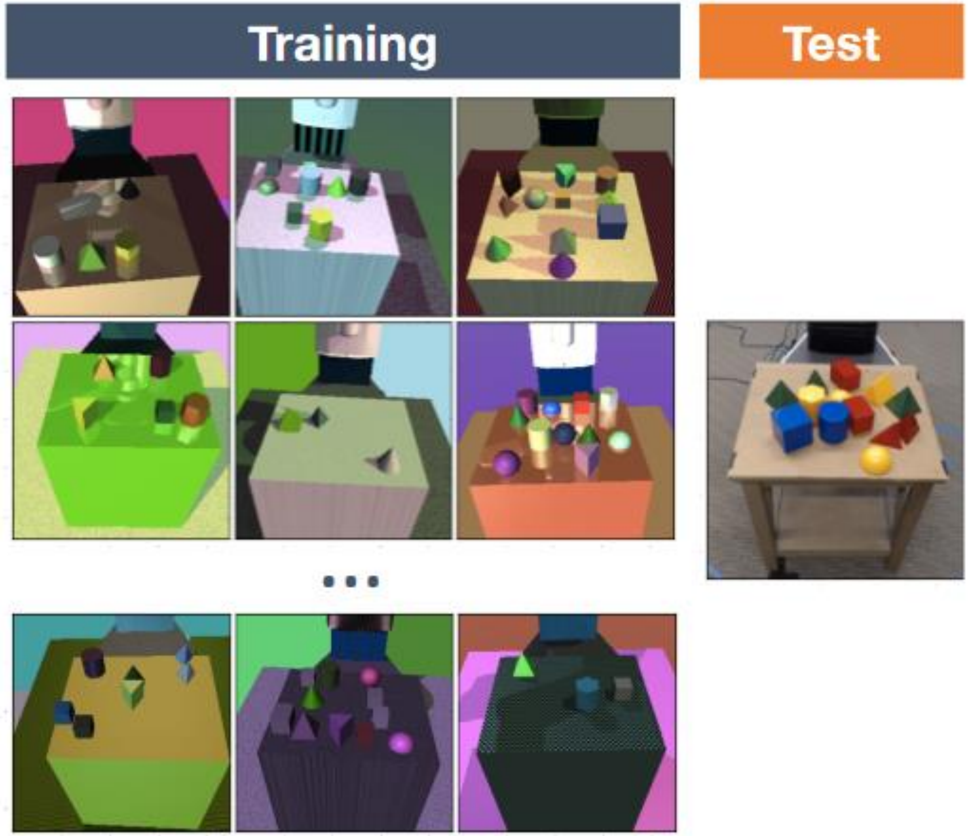
~500 semirealistic textures, 12 floorplans

CAD²RL: Real Single-Image Flight Without a Single Real Image
Sadeghi and Levine, RSS 2017

Domain randomization

Simulators do not even need to be realistic.

We randomize not only the scene but also the objects.

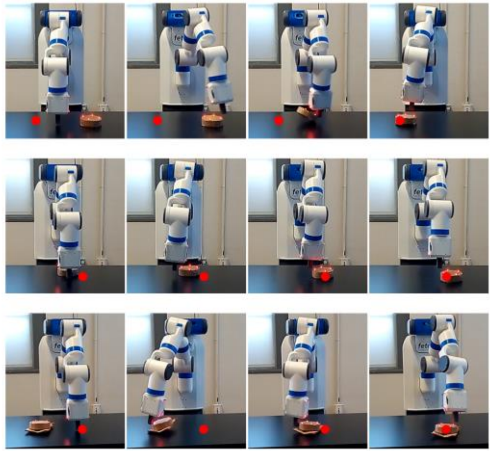


Applications of domain randomization

- Pose estimation
- Object detection
- Localization and tracking
- Visuomotor control
- Manipulation

Domain randomization for dynamics

What if the mismatch between the simulation and the real world is due to dynamics?

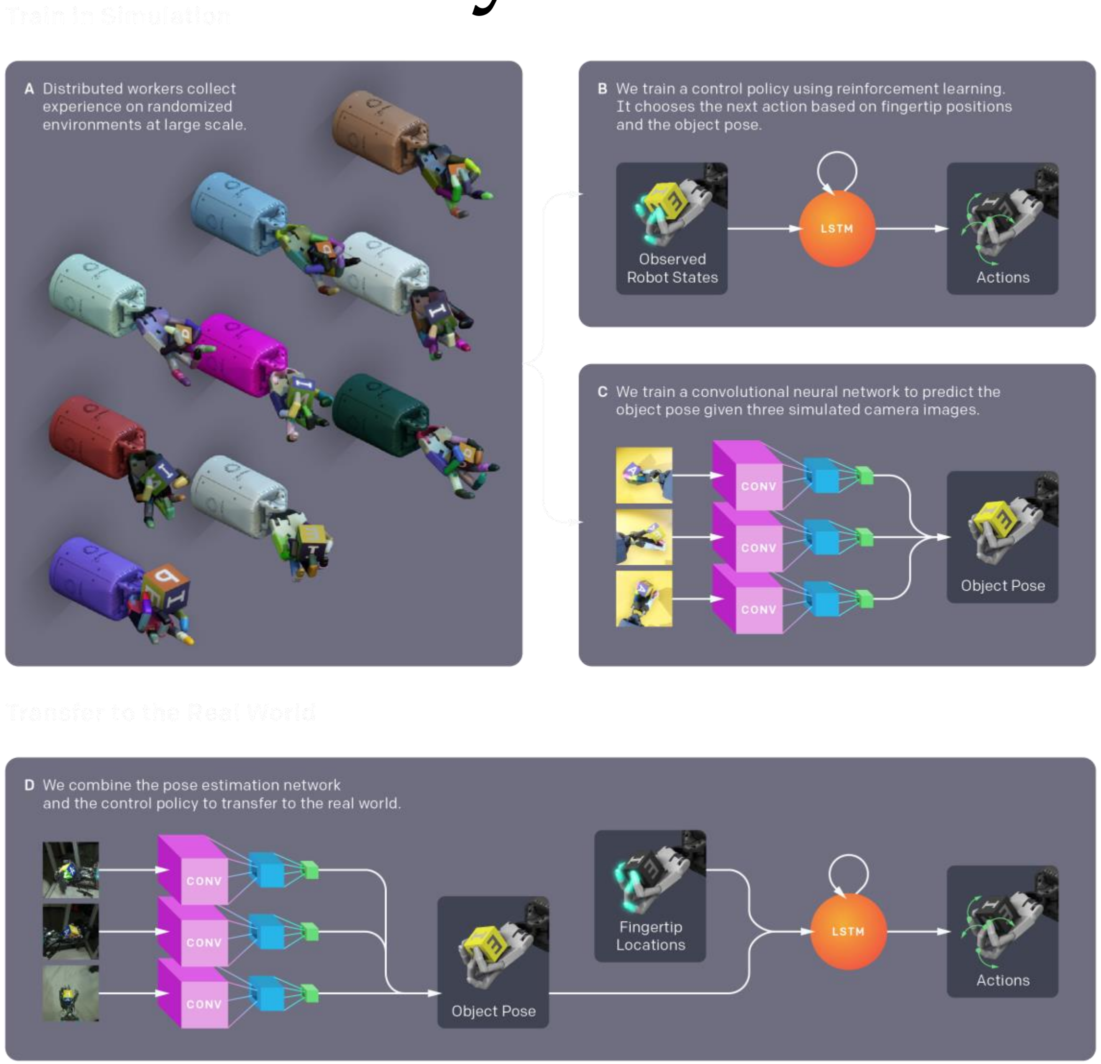


Parameter	Range
Link Mass	$[0.25, 4] \times$ default mass of each link
Joint Damping	$[0.2, 20] \times$ default damping of each joint
Puck Mass	$[0.1, 0.4] kg$
Puck Friction	$[0.1, 5]$
Puck Damping	$[0.01, 0.2] Ns/m$
Table Height	$[0.73, 0.77] m$
Controller Gains	$[0.5, 2] \times$ default gains
Action Timestep λ	$[125, 1000] s^{-1}$

Will a feedforward neural network policy work?

Sim-to-real transfer of robotic control with dynamics randomization
Peng et al., ICRA 2018

Dexterity with domain randomization



Learning dexterous in-hand manipulation
OpenAI, IJRR 2020

Why does domain randomization work?

- Training is done over a distribution of domains that contain the real world.

- Domain randomization helps the model identify what to ignore.

- Domain randomization is meta learning.

Domain randomization recipe

1. Build a simulator
2. Calibrate it to the environment
3. Design randomizations
4. Train a model
5. Evaluate the model in real-world
6. Examine failures
7. If unhappy, go to step 3

Next time…

Week 10
Fri, Oct 27

Lecture Meta & Multi-task learning
Presentation Meta & Multi-task learning

Due **Homework #3**

- Chan et al., Human Irrationality: Both Bad and Good for Reward Inference (2021).
- Julian et al., Never Stop Learning: The Effectiveness of Fine-Tuning in Robotic Reinforcement Learning (2020).
- Kim et al., Bayesian Model-Agnostic Meta-Learning (2018).
- Sodhani et al., Multi-Task Reinforcement Learning with Context-based Representations (2021).
- Shridhar et al., Perceiver-Actor: A Multi-Task Transformer for Robotic Manipulation (2022).