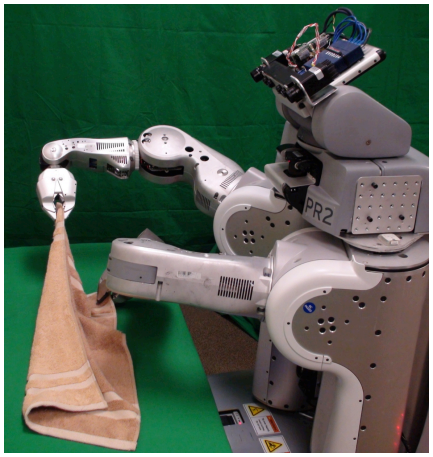
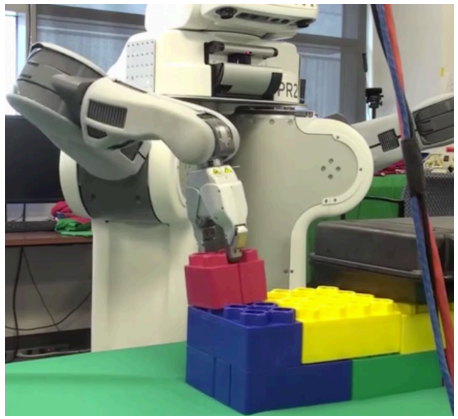
Making Robots Learn

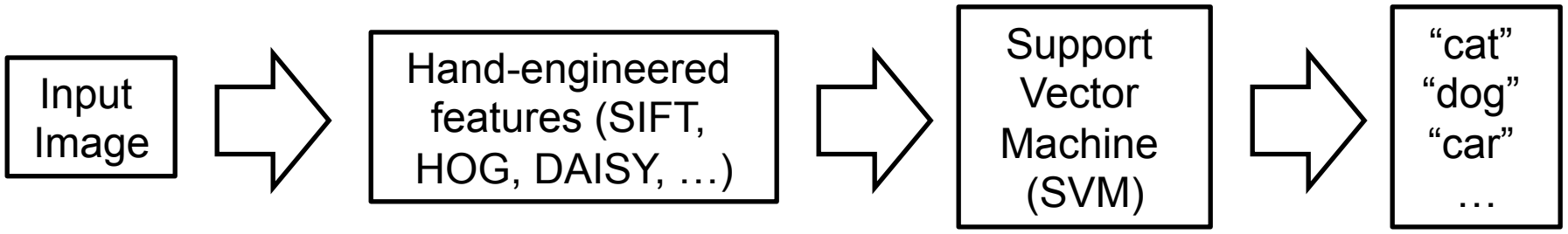
Pieter Abbeel -- UC Berkeley EECS



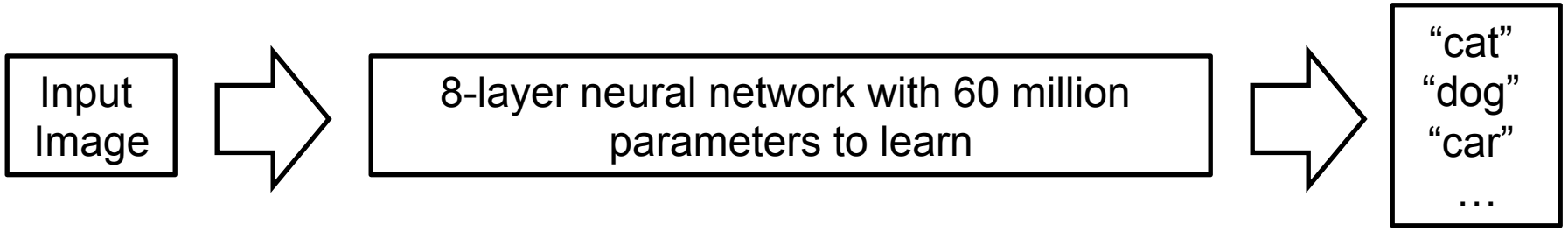


Object Detection in Computer Vision

- State-of-the-art object detection until 2012:

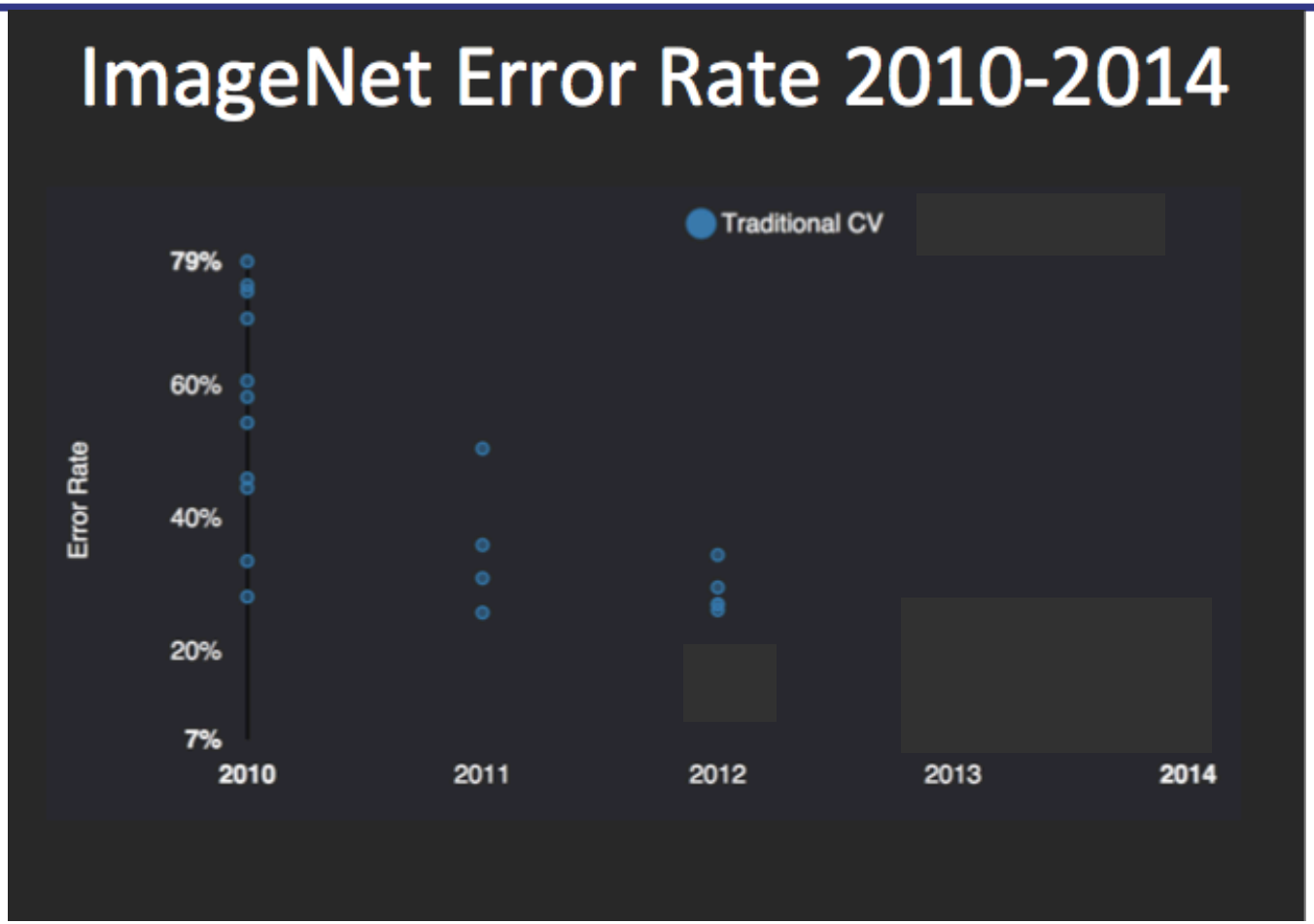


- Deep Supervised Learning (Krizhevsky, Sutskever, Hinton 2012; also LeCun, Bengio, Ng, Darrell, …):



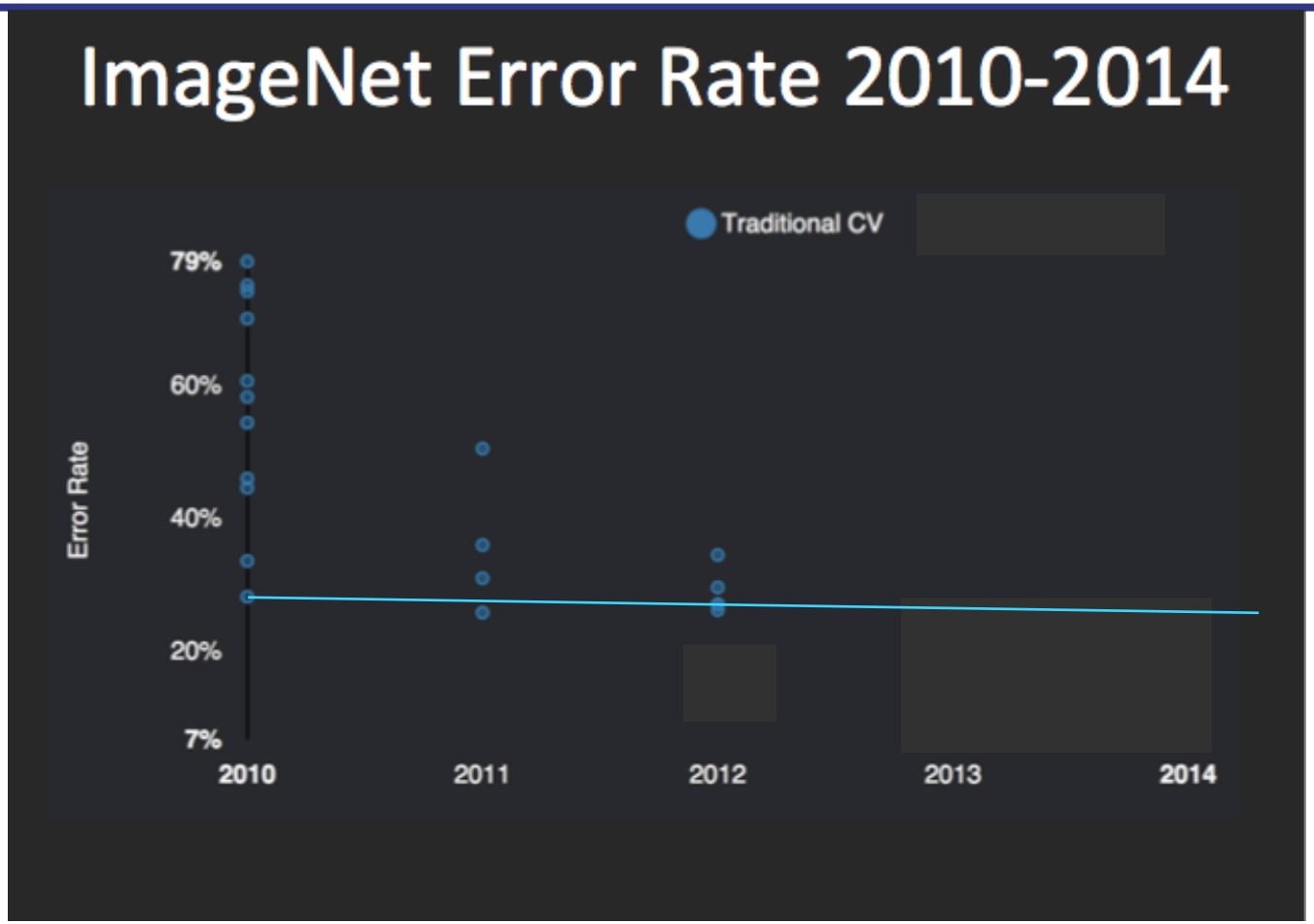
  - ~1.2 million training images from ImageNet [Deng, Dong, Socher, Li, Li, Fei-Fei, 2009]

Performance



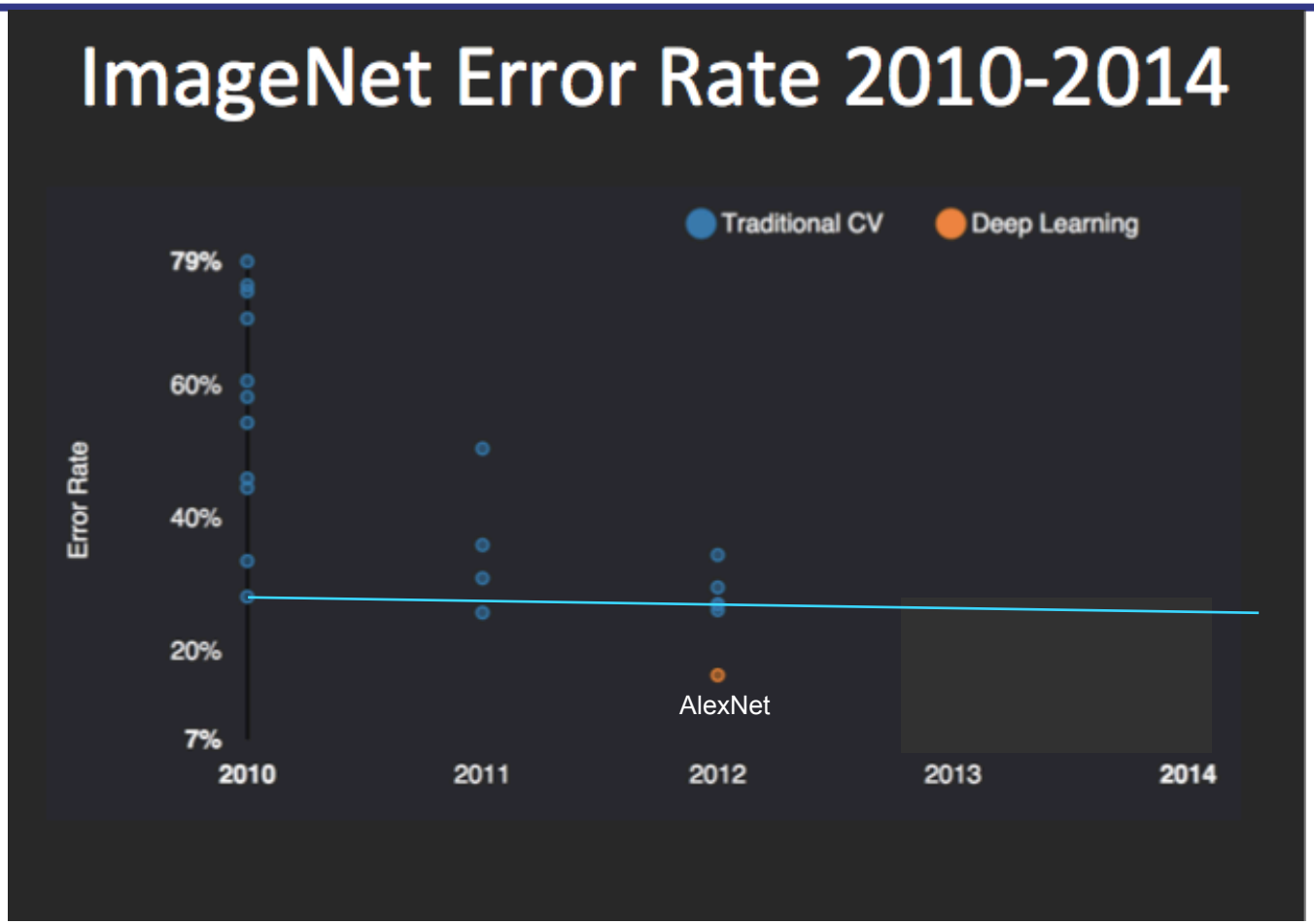
graph credit Matt Zeiler, Clarifai

Performance



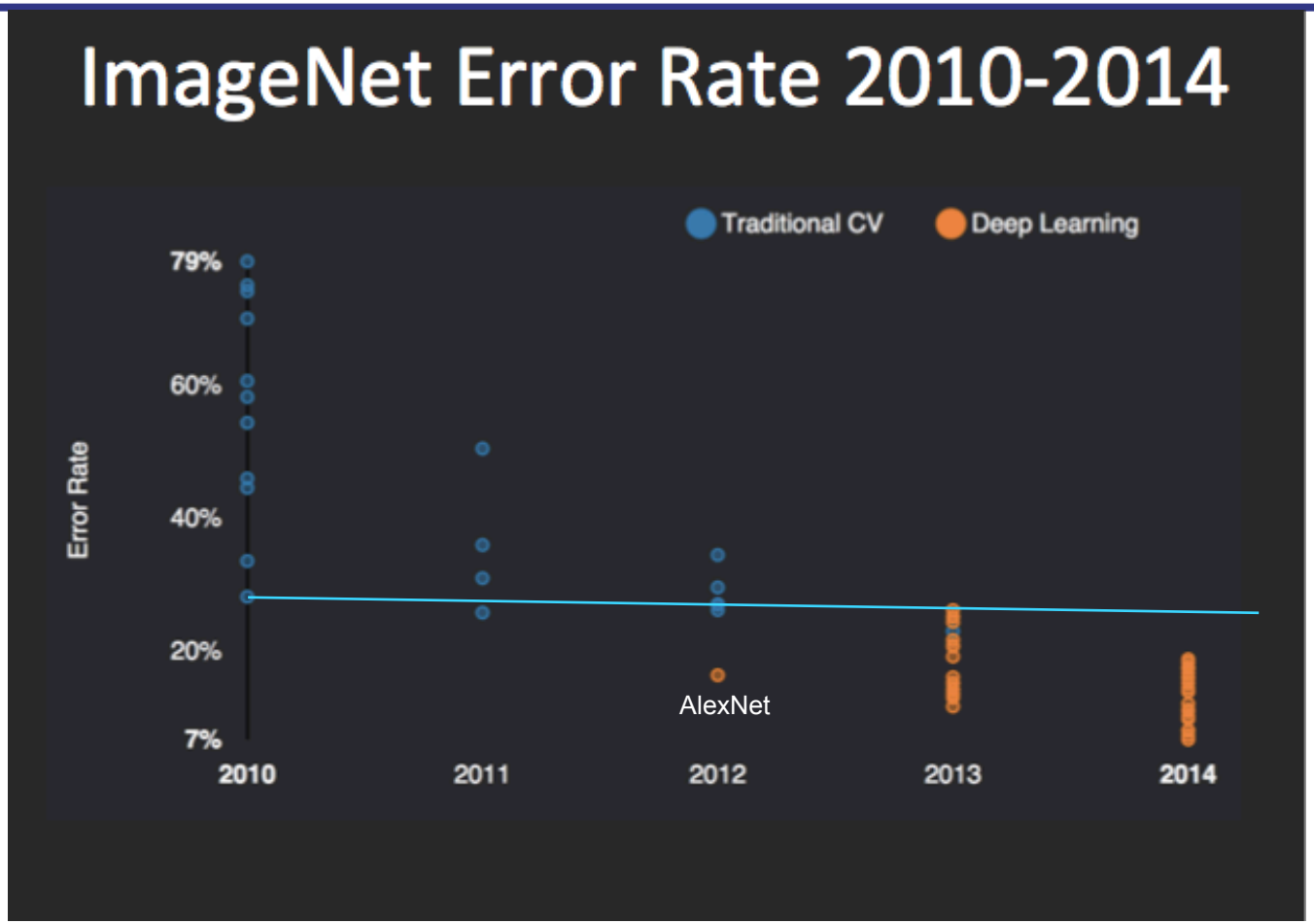
graph credit Matt Zeiler, Clarifai

Performance



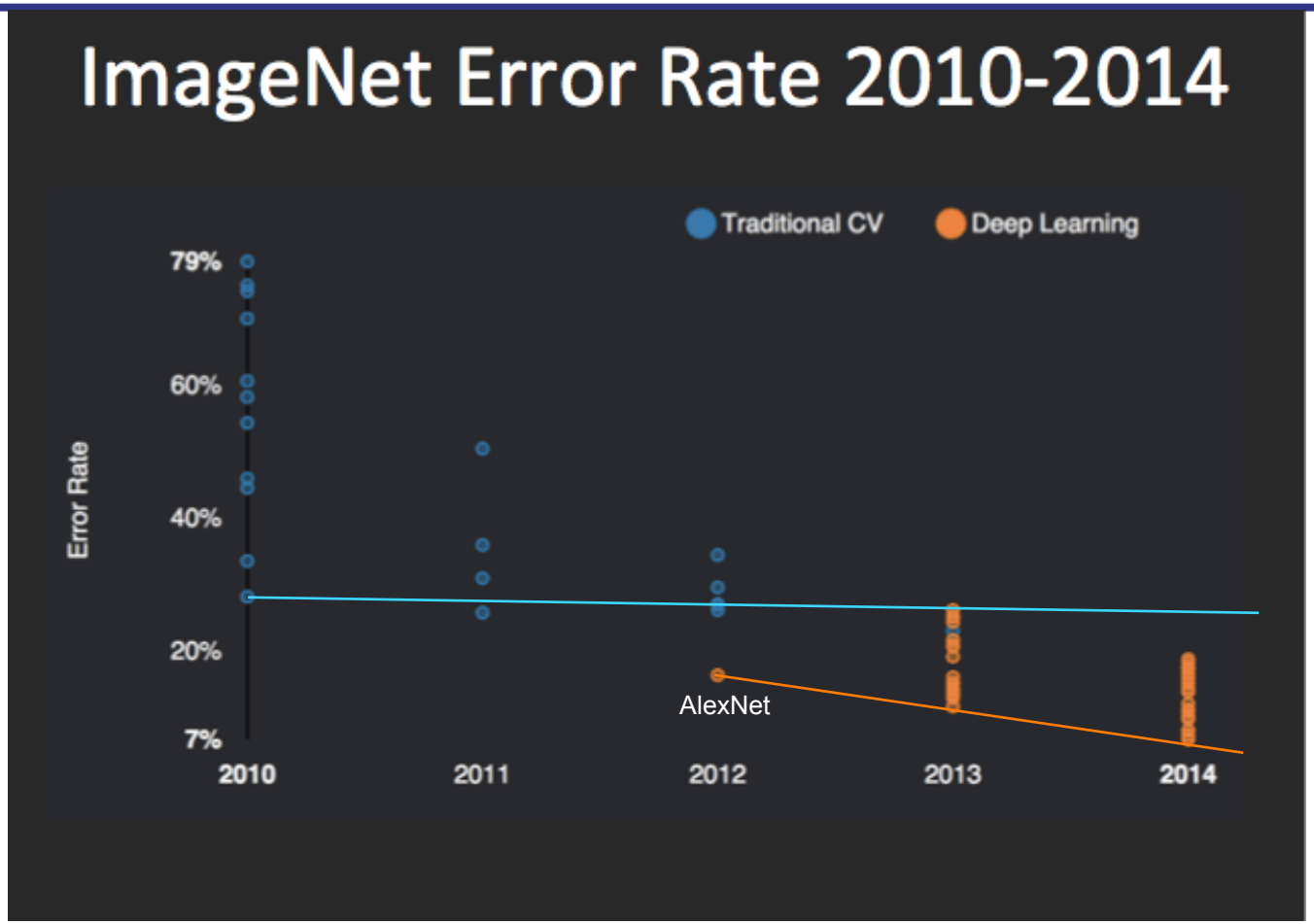
graph credit Matt Zeiler, Clarifai

Performance



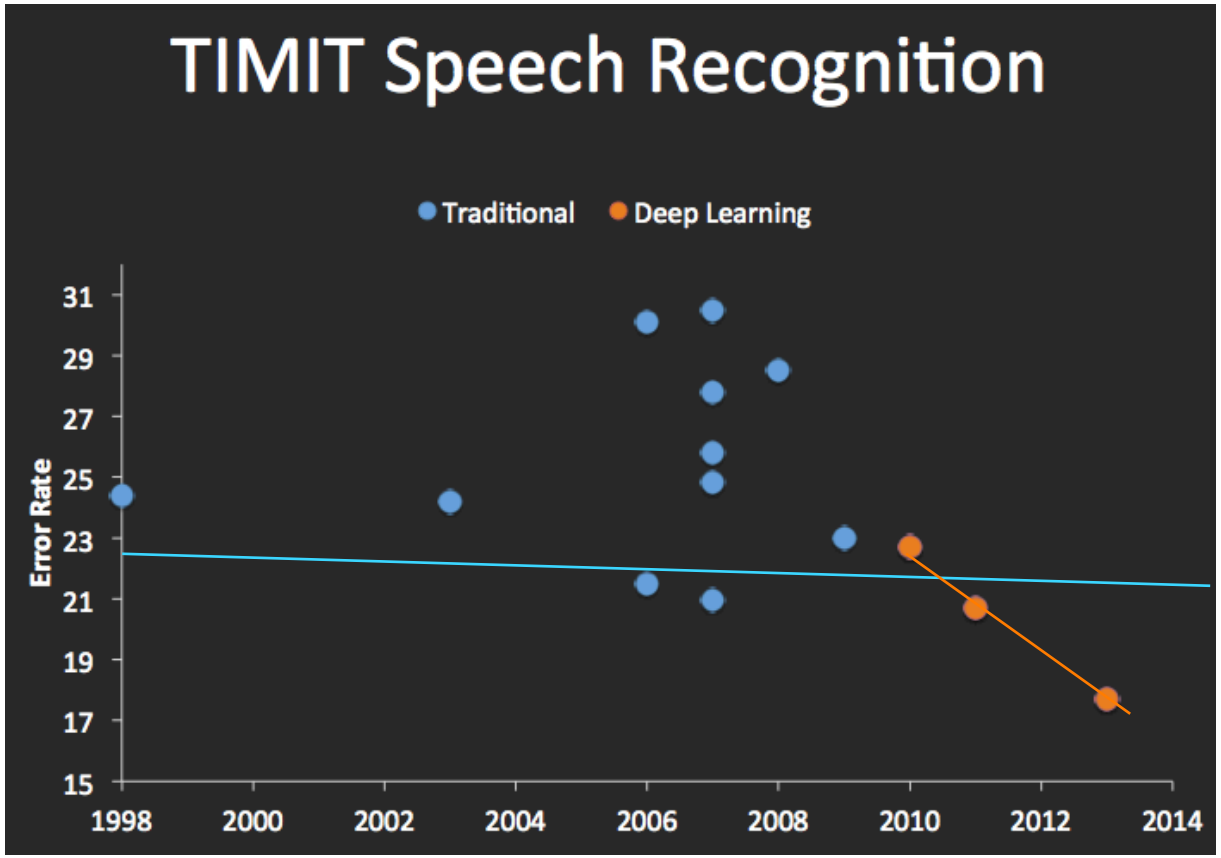
graph credit Matt Zeiler, Clarifai

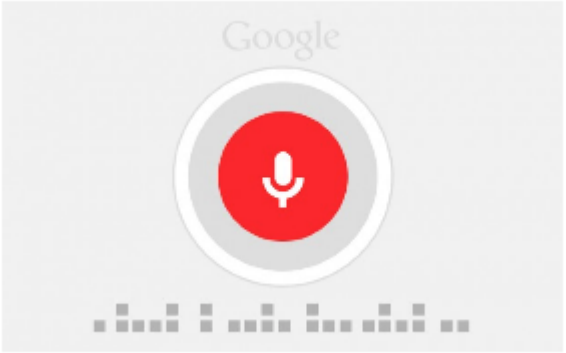
Performance



graph credit Matt Zeiler, Clarifai

Speech Recognition





graph credit Matt Zeiler, Clarifai

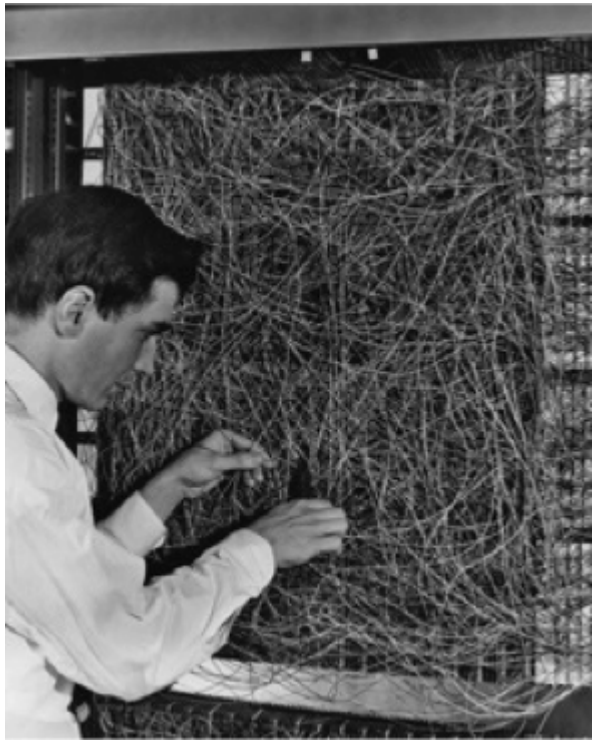
History

1958 Rosenblatt proposed perceptrons
1980 Neocognitron (Fukushima, 1980)
1982 Hopfield network, SOM (Kohonen, 1982), Neural PCA (Oja, 1982)
1985 Boltzmann machines (Ackley et al., 1985)
1986 Multilayer perceptrons and backpropagation (Rumelhart et al., 1986)
1988 RBF networks (Broomhead&Lowe, 1988)
1989 Autoencoders (Baldi&Hornik, 1989), Convolutional network (LeCun, 1989)
1992 Sigmoid belief network (Neal, 1992)
1993 Sparse coding (Field, 1993) (Olshausen, 1996)

2000s Sparse, Probabilistic, and Energy models (Hinton, Bengio, LeCun, Ng)

Is deep learning 3, 30, or 60 years old?

based on history by K. Cho

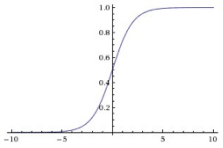


Rosenblatt's Perceptron

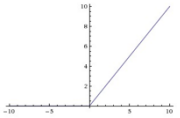
What's Changed

- Data
  - 1.2M training examples
  - 2048 (different crops)
  - 90 (PCA re-colorings)
- Compute power
  - Two NVIDIA GTX 580 GPUs
  - 5-6 days of training time
- Nonlinearity

  Sigmoid

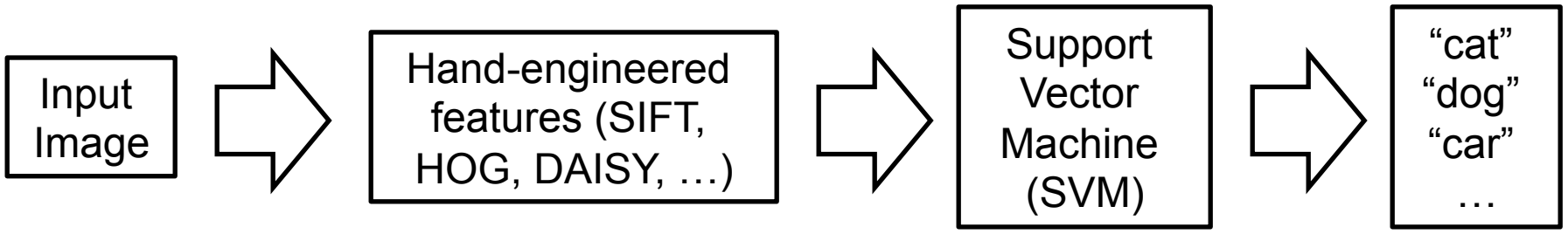
  

  → ReLU

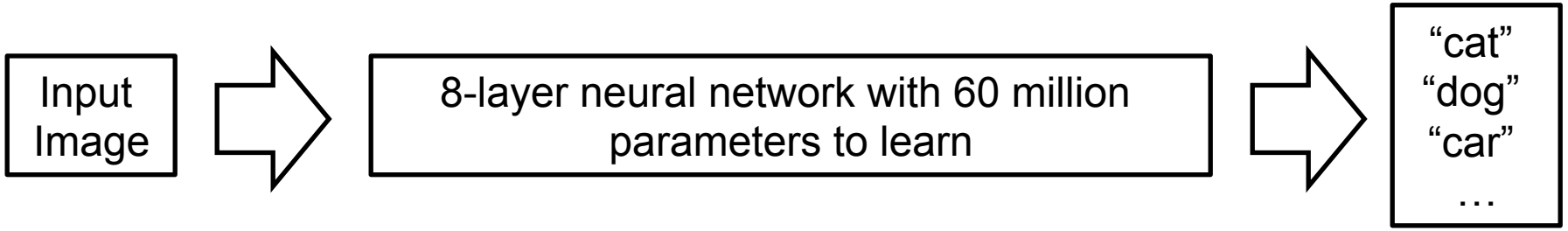
  
- Regularization
  - Drop-out
- Exploration of model structure
- Optimization know-how

Object Detection in Computer Vision

- State-of-the-art object detection until 2012:



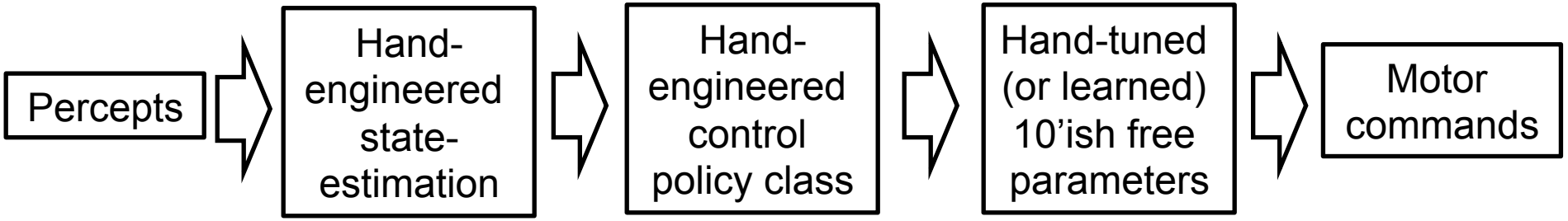
- Deep Supervised Learning (Krizhevsky, Sutskever, Hinton 2012; also LeCun, Bengio, Ng, Darrell, …):



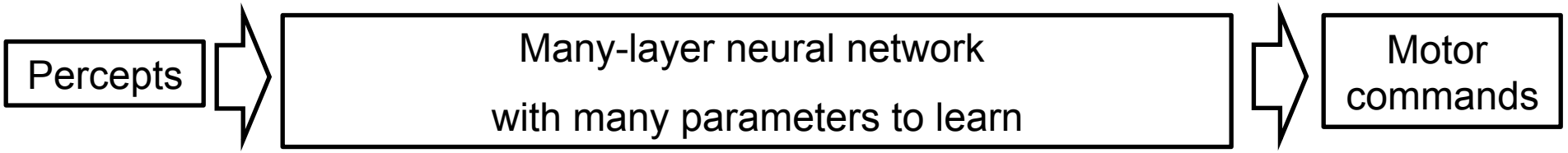
  - ~1.2 million training images from ImageNet [Deng, Dong, Socher, Li, Li, Fei-Fei, 2009]

Robotics

- **Current state-of-the-art robotics**

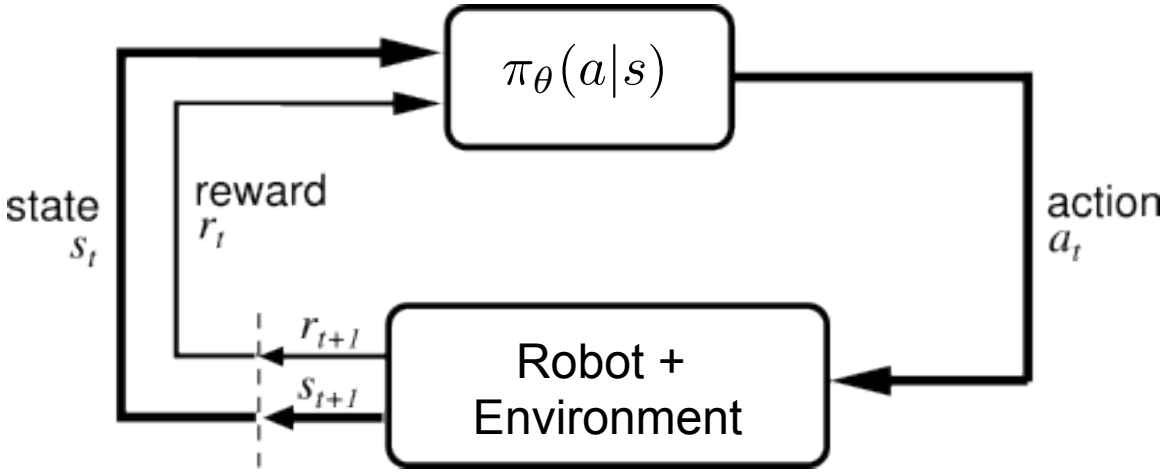


- **Deep reinforcement learning**



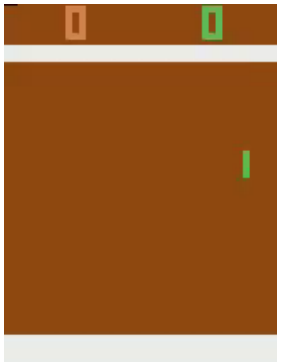
Reinforcement Learning

probability of taking action a in state s

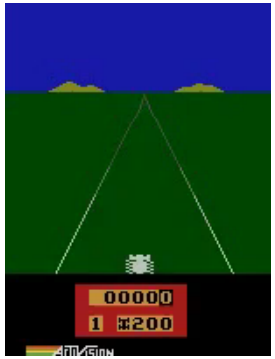


- Goal: $\max_\theta \ \mathrm{E}[\sum_{t=0}^{H} R(s_t)|\pi_\theta]$

From Pixels to Actions?

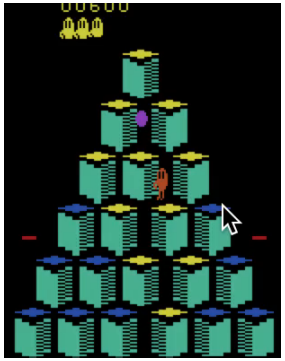


Pong



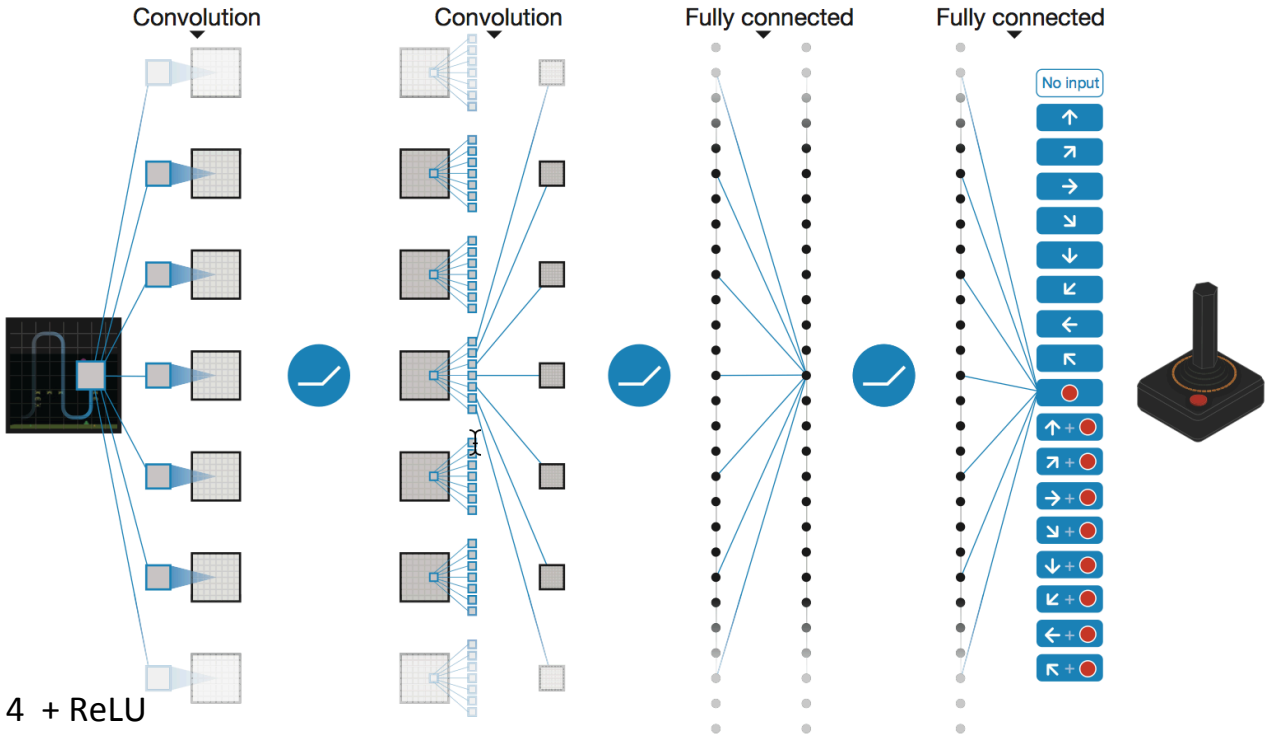
Enduro

Beamrider



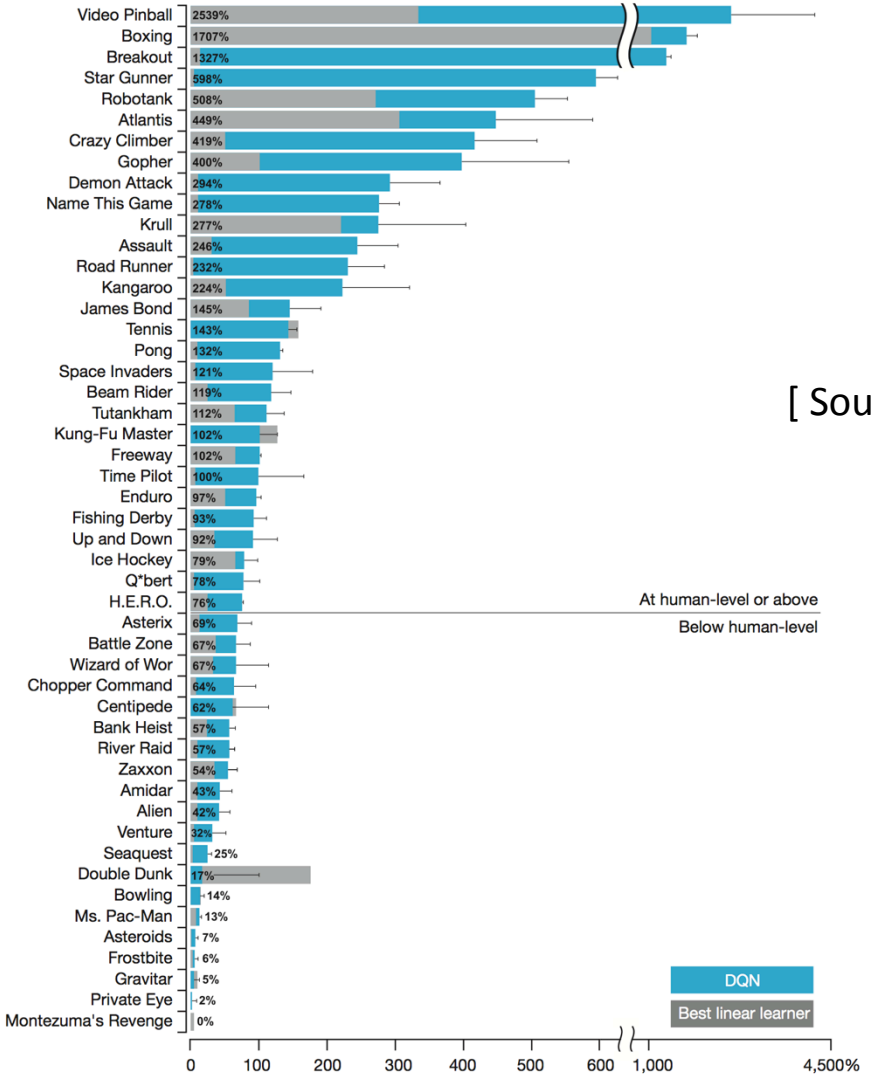
Q*bert

Deep Q-Network (DQN): From Pixels to Joystick Commands



32 8x8 filters with stride 4 + ReLU
64 4x4 filters with stride 2 + ReLU
64 3x3 filters with stride 1 + ReLU
fully connected 512 units + ReLU
fully connected output units, one per action

[Source: Mnih et al., Nature 2015 (DeepMind)]



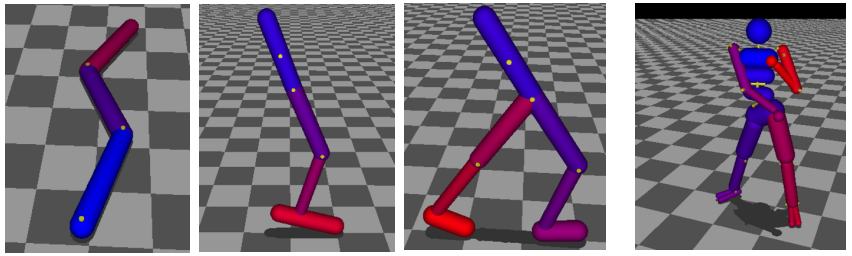
[Source: Mnih et al., Nature 2015 (DeepMind)]

Deep Q-Network (DQN)

- Approach:
  - Q-learning with e-greedy and deep network as function approximator
- Key idea 1: stabilizing Q-learning
  - Mini-batches of size 32 (vs. single sample updates)
  - Q-values used to compute temporal difference only updated every 10,000 updates
- Key idea 2: lots of data / compute
  - trained for a total of 50 million frames (=***38 days of game experience***) and use a replay memory of one million most recent frames

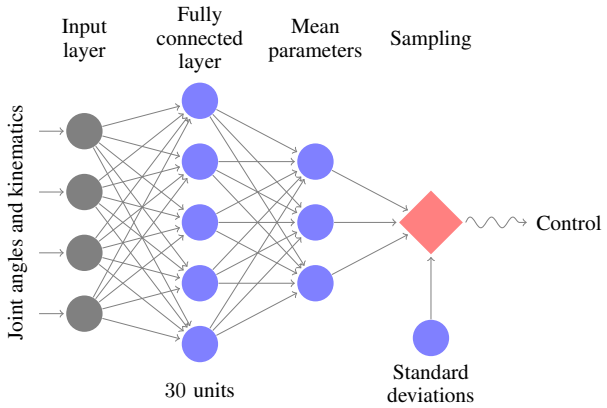
How About Continuous Control, e.g., Locomotion?

Robot models in physics simulator (MuJoCo, from Emo Todorov)



Input: joint angles and velocities
Output: joint torques

Neural network architecture:



Challenges with Q-Learning

- How to score every possible action?
- How to ensure monotonic progress?

Policy Optimization

$$\max_{\theta} \quad \mathrm{E}[\sum_{t=0}^{H} R(s_t)|\pi_\theta]$$

- Often simpler to represent good policies than good value functions
- True objective of expected cost is optimized (vs. a surrogate like Bellman error)

- Existing work: (natural) policy gradients
  - Challenges: good, large step directions

Trust Region Policy Optimization

$$\max_{\theta} \quad \mathrm{E}[\sum_{t=0}^{H} R(s_t)|\pi_\theta]$$

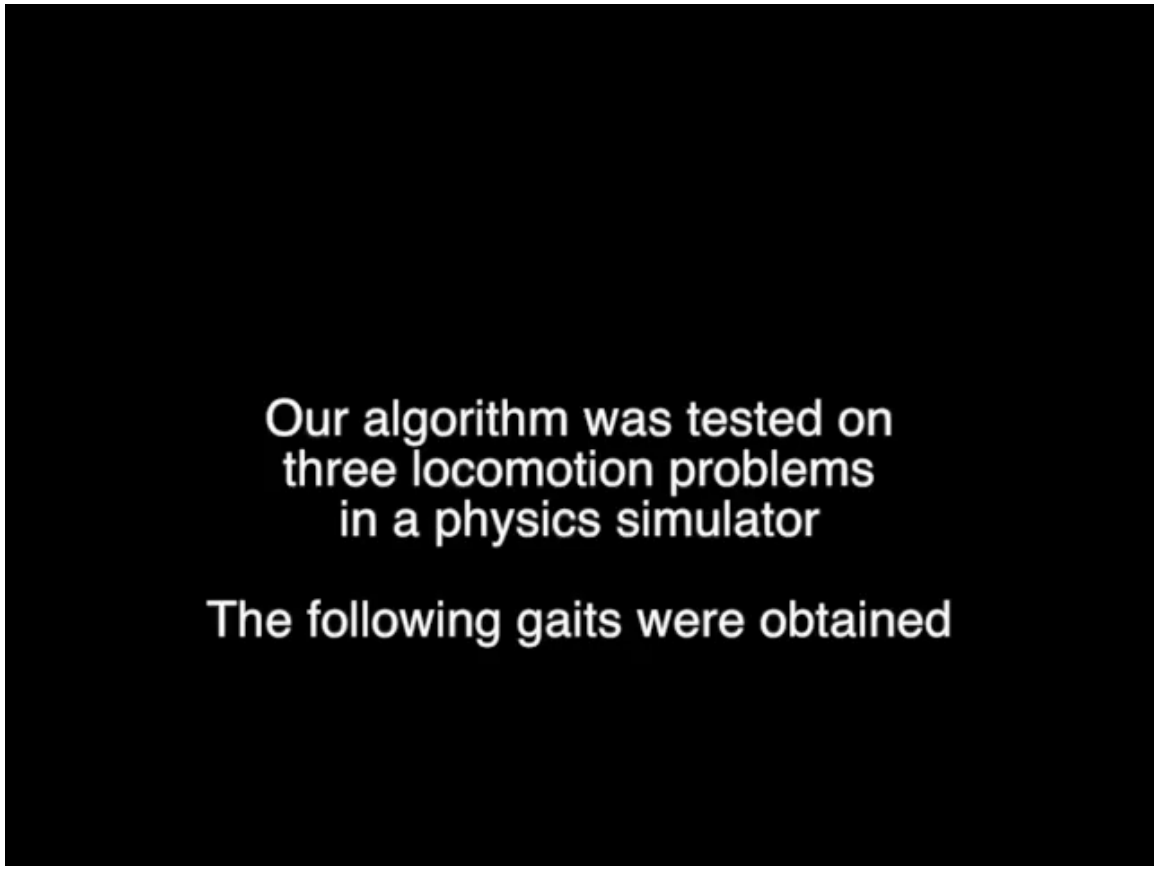
$$\max_{\delta\theta} \quad \hat{g}^\top \delta\theta$$

$$\text{s.t.} \quad KL(P(\tau;\theta)||P(\tau;\theta+\delta\theta)) \leq \varepsilon$$

- Trust Region:
  - Sampled evaluation of gradient
  - Gradient only locally a good approximation
  - Change in policy changes state-action visitation frequencies

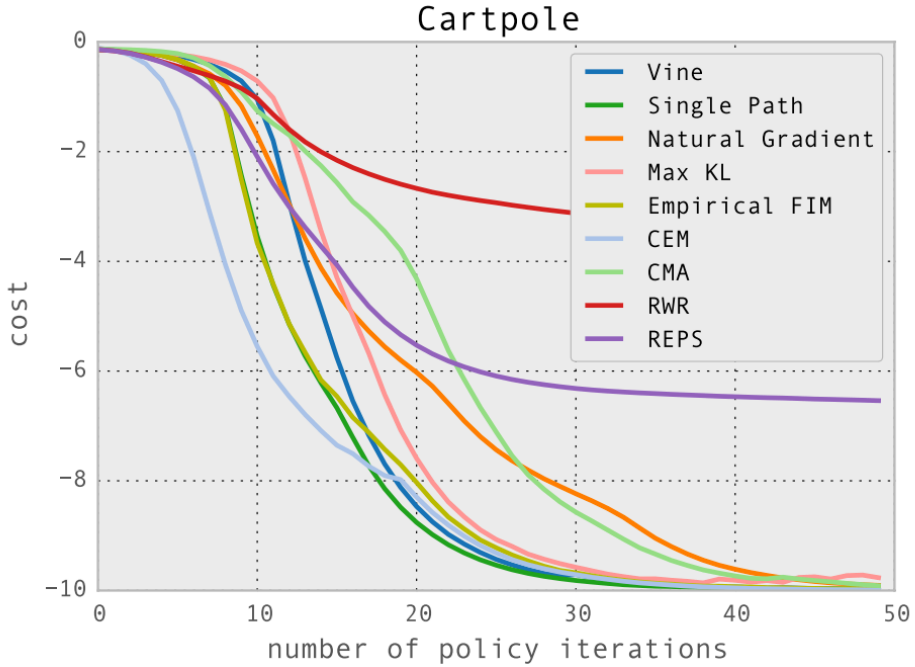
[Schulman, Levine, Moritz, Jordan, Abbeel, 2015]

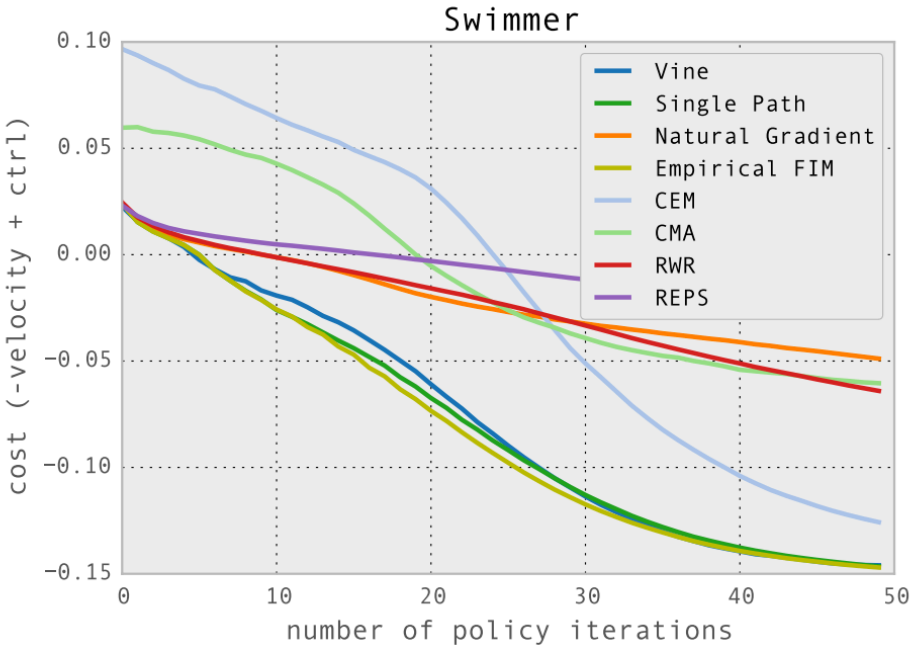
Experiments in Locomotion



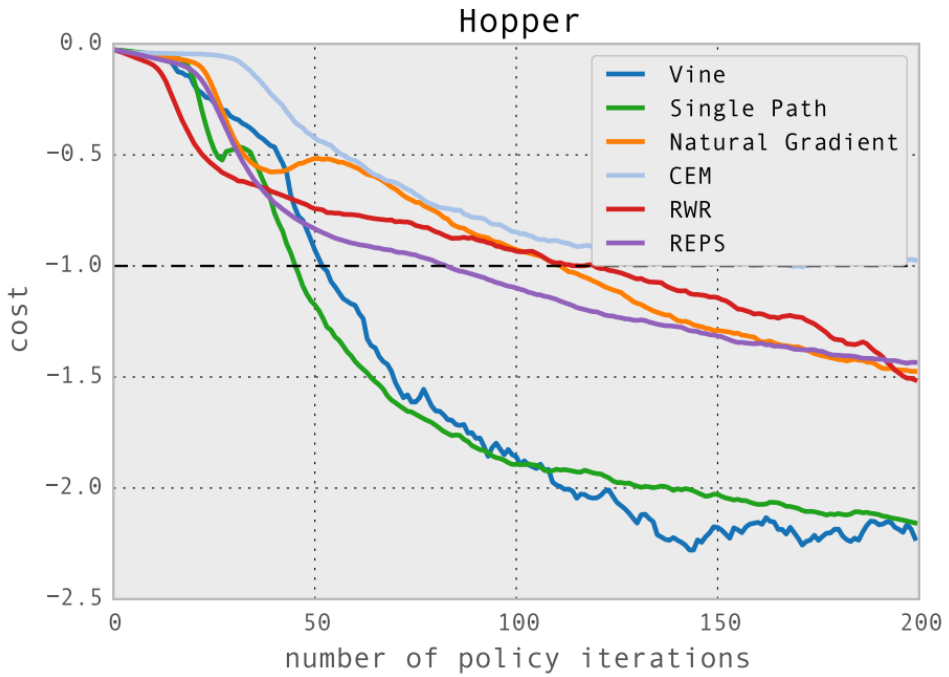
[Schulman, Levine, A.]

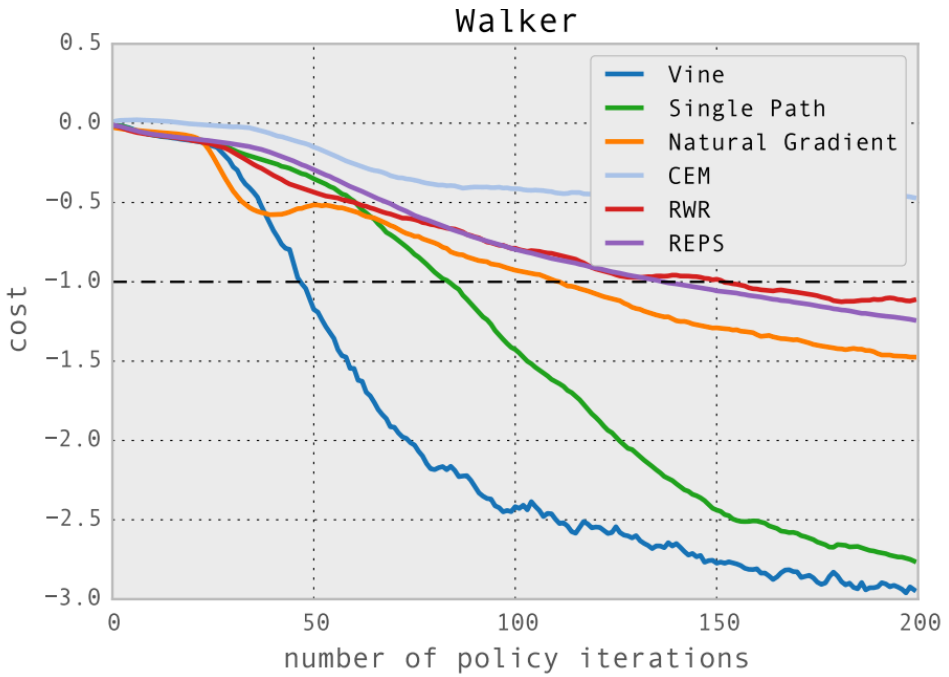
Learning Curves -- Comparison





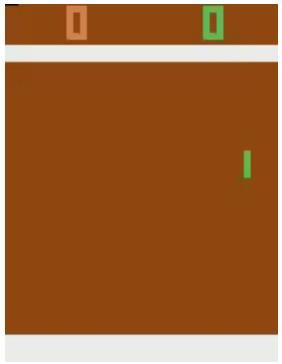
Learning Curves -- Comparison



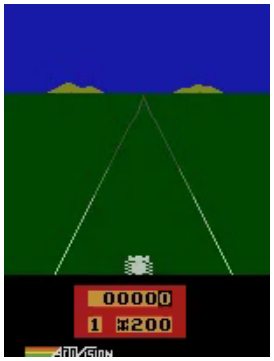


Atari Games

- Deep Q-Network (DQN) [Mnih et al, 2013/2015]
- Dagger with Monte Carlo Tree Search [Xiao-Xiao et al, 2014]
- Trust Region Policy Optimization [Schulman, Levine, Moritz, Jordan, Abbeel, 2015]

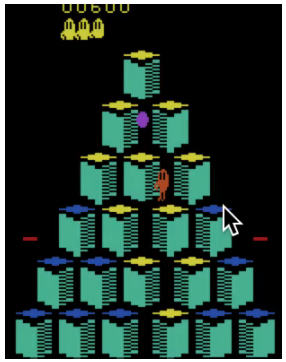


Pong



Enduro

Beamrider



Q*bert

Generalized Advantage Estimation (GAE)

Objective: $\max_\theta \quad \mathrm{E}[\sum_{t=0}^{H} R(s_t)|\pi_\theta]$

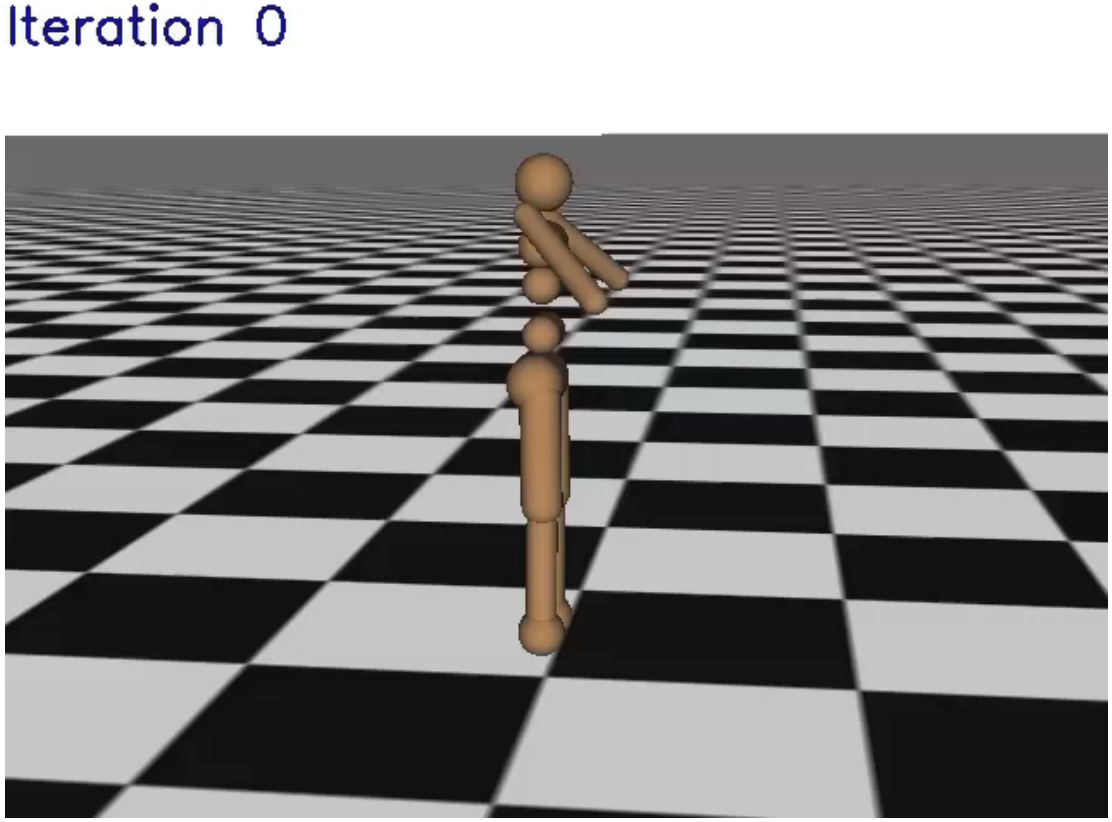
Gradient: $\mathrm{E}[\sum_{t=0}^{H} \nabla_\theta \log \pi_\theta(a_t|s_t) \left(\sum_{k=t}^{H} R(s_k) - V(s_t) \right)]$

single sample estimate of advantage

- Generalized Advantage Estimation
  - Exponential interpolation between actor-critic and Monte Carlo estimates
  - Trust region approach to (high-dimensional) value function estimation

[Schulman, Moritz, Levine, Jordan, Abbeel, 2015]

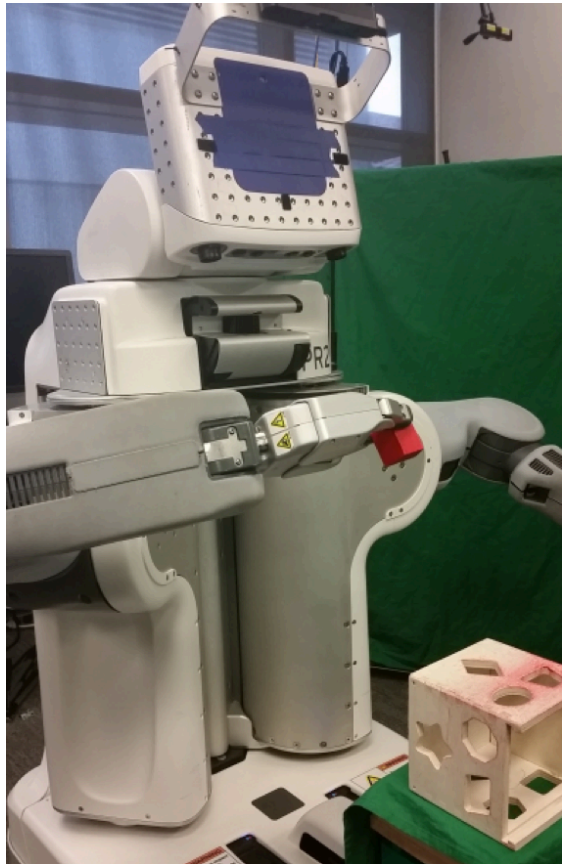
Learning Locomotion



[Schulman, Moritz, Levine, Jordan, Abbeel, 2015]

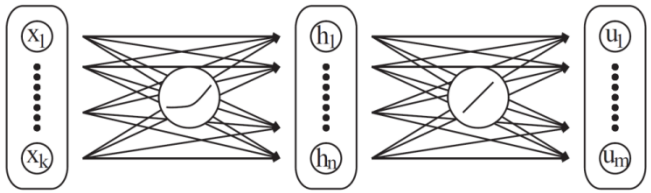
In Contrast: Darpa Robotics Challenge

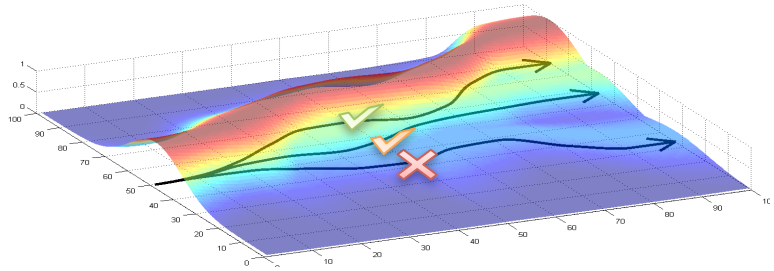
How About Real Robotic Visuo-Motor Skills?



Guided Policy Search

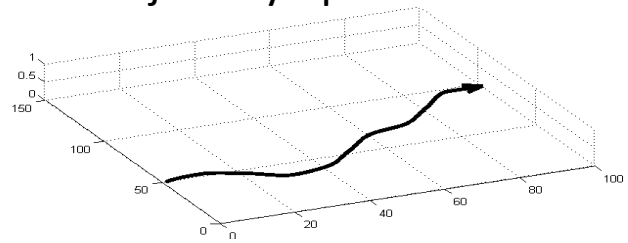
general-purpose neural network controller



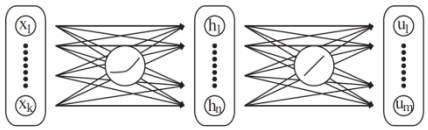


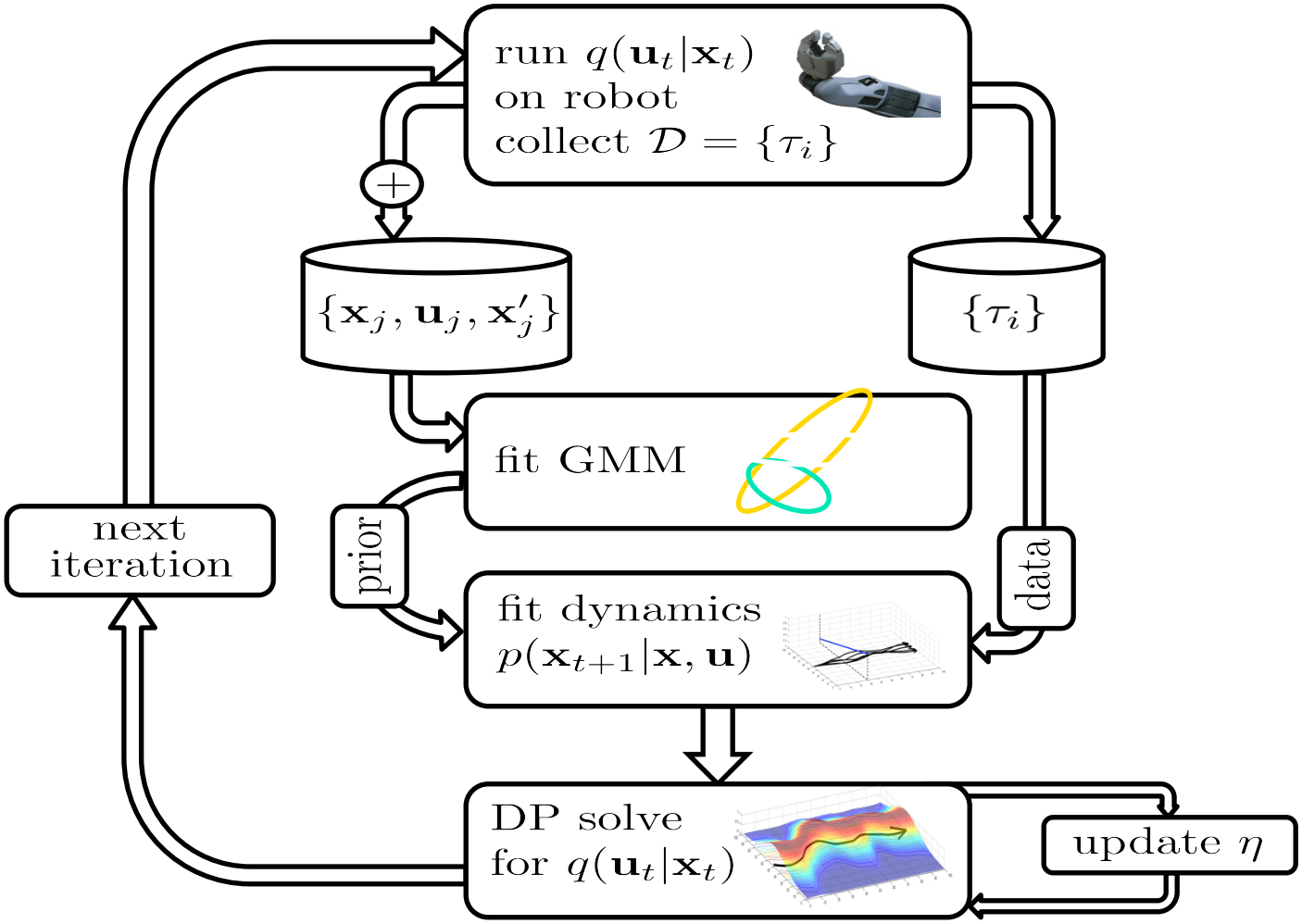
policy search (RL)	complex dynamics	complex policy	HARD
supervised learning	~~complex dynamics~~	complex policy	EASY
trajectory optimization	complex dynamics	~~complex policy~~	EASY

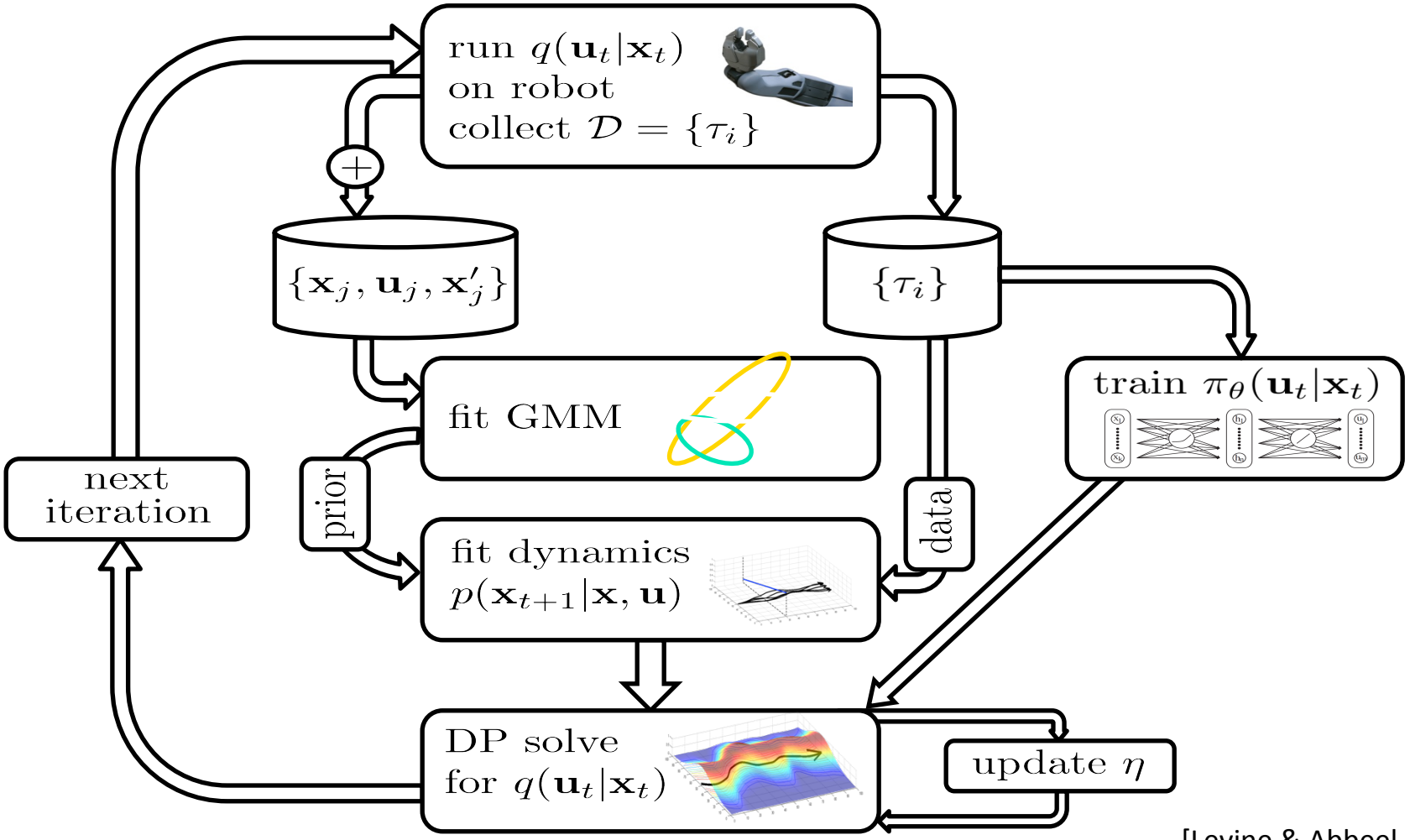
trajectory optimization



Supervised learning




run $q(\mathbf{u}_t|\mathbf{x}_t)$
on robot
collect $\mathcal{D} = \{\tau_i\}$
$\{\mathbf{x}_j, \mathbf{u}_j, \mathbf{x}'_j\}$
$\{\tau_i\}$
fit GMM
prior
data
next
iteration
fit dynamics
$p(\mathbf{x}_{t+1}|\mathbf{x}, \mathbf{u})$
DP solve
for $q(\mathbf{u}_t|\mathbf{x}_t)$
update η



[Levine & Abbeel, NIPS 2014]

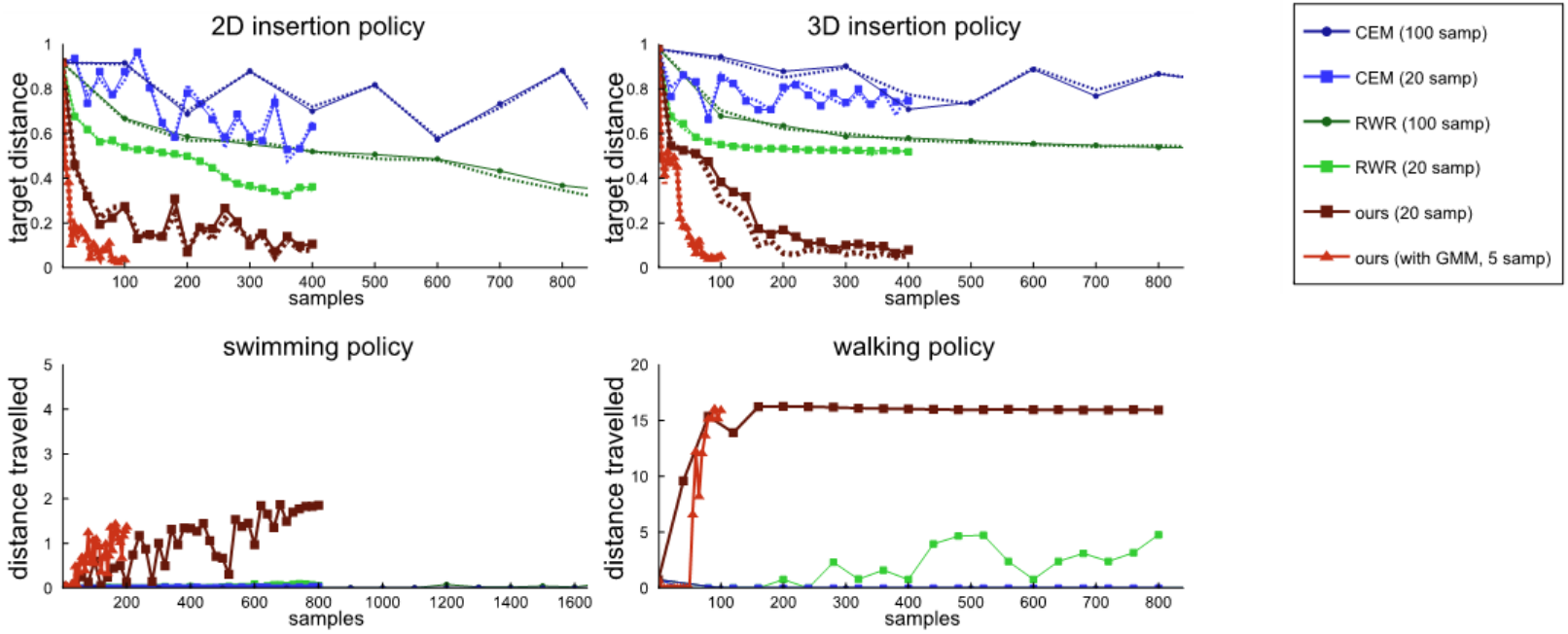
Guided Policy Search

Swimming

learned policy

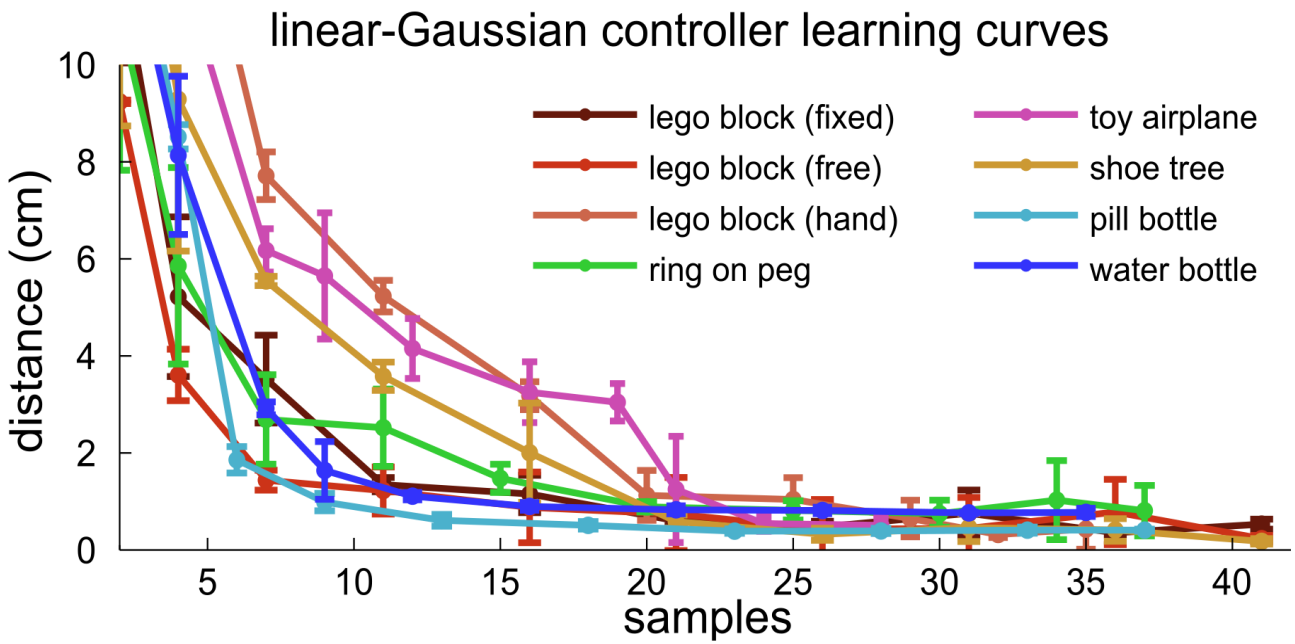
[neural network]

Comparison


2D insertion policy
target distance
samples
3D insertion policy
target distance
samples
swimming policy
distance travelled
samples
walking policy
distance travelled
samples
CEM (100 samp)
CEM (20 samp)
RWR (100 samp)
RWR (20 samp)
ours (20 samp)
ours (with GMM, 5 samp)

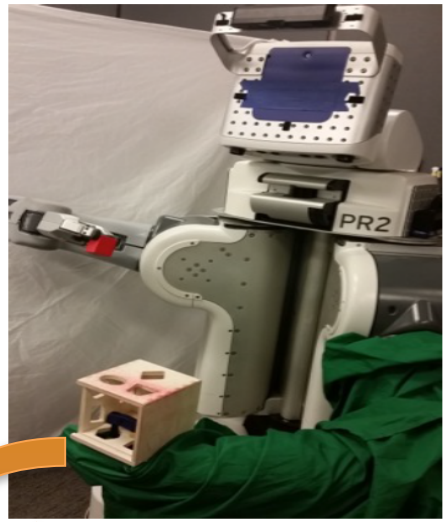
Block Stacking – Learning the Controller for a Single Instance

Linear-Gaussian Controller Learning Curves

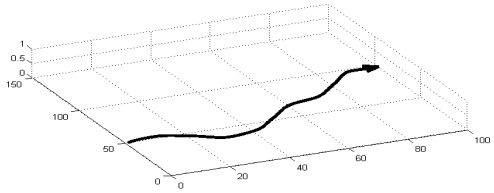


Instrumented Training

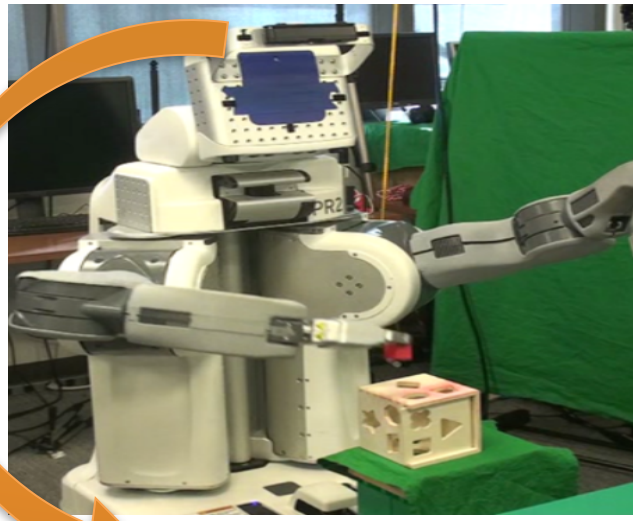
training time



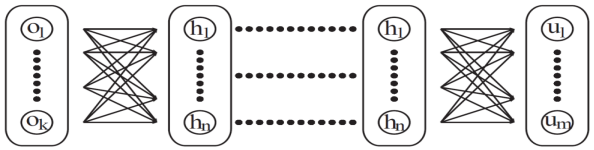
$\mathbf{x}_t \rightarrow \mathbf{u}_t$



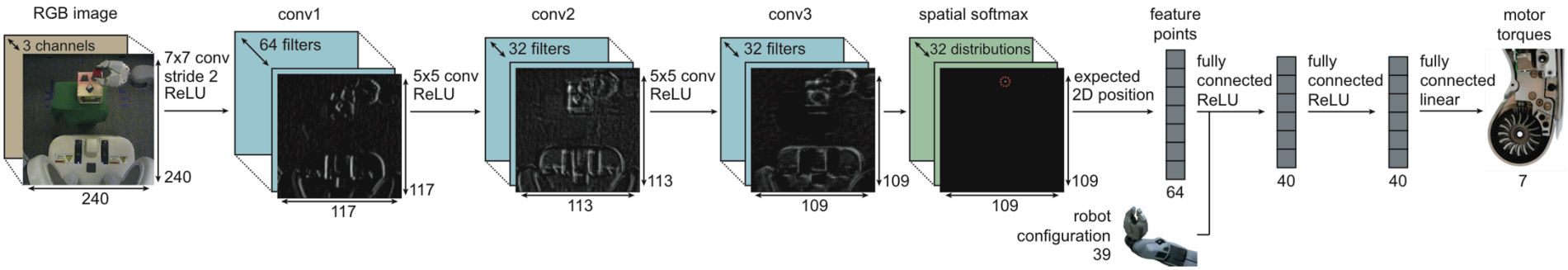
test time



$\mathbf{o}_t \rightarrow \mathbf{u}_t$

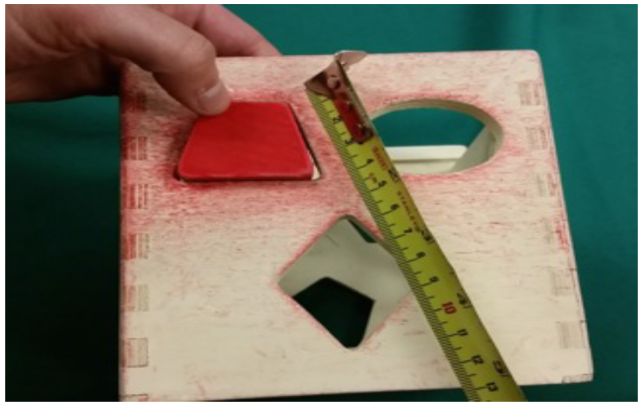


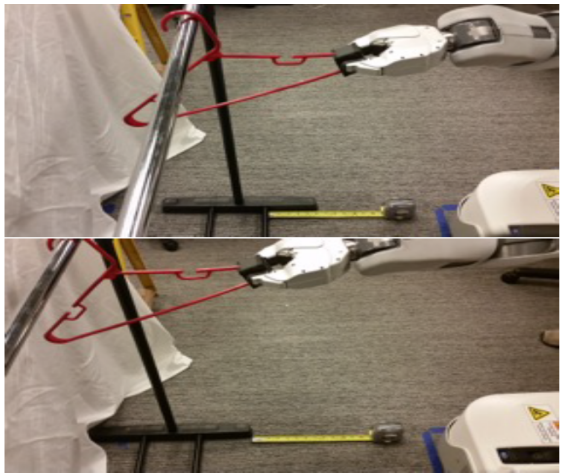
Architecture (92,000 parameters)



[Levine*, Finn*, Darrell, Abbeel, 2015, TR at: rll.berkeley.edu/deeplearningrobotics]

Experimental Tasks



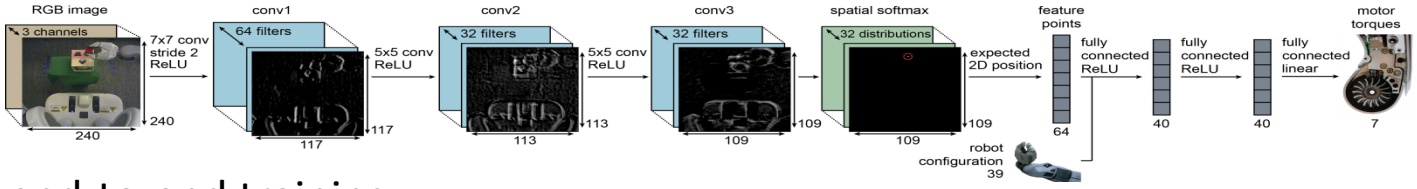


Learning

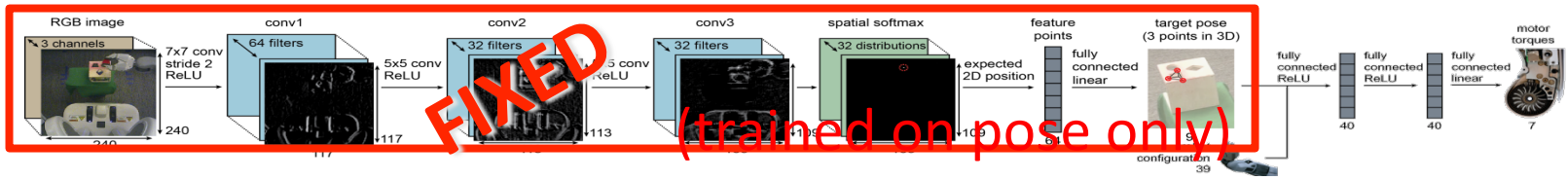
Learned Skills

[Levine*, Finn*, Darrell, Abbeel, 2015, TR at: rll.berkeley.edu/deeplearningrobotics]

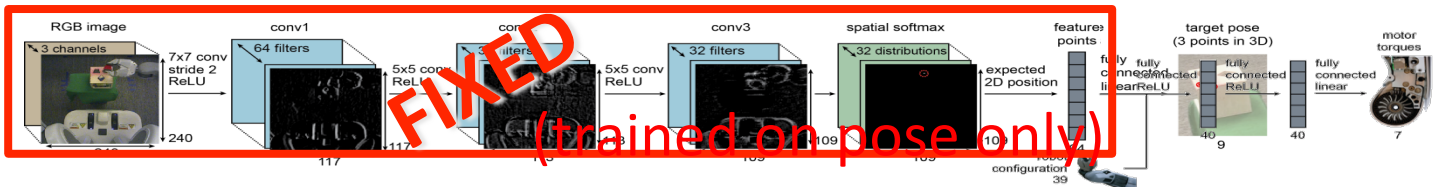
Comparisons



end-to-end training



pose prediction



pose features

Comparisons

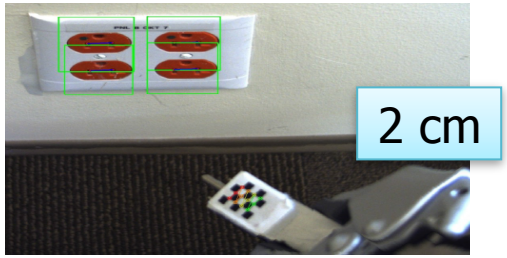
coat hanger	success rate
pose prediction	55.6%
pose features	88.9%
end-to-end training	**100%**

shape sorting cube	success rate
pose prediction	0%
pose features	70.4%
end-to-end training	**96.3%**

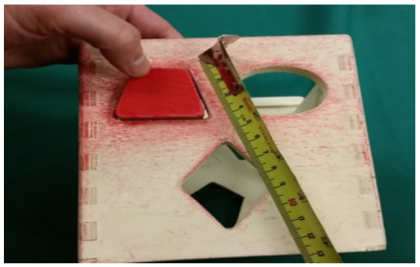
toy claw hammer	success rate
pose prediction	8.9%
pose features	62.2%
end-to-end training	**91.1%**

bottle cap	success rate
pose prediction	n/a
pose features	55.6%
end-to-end training	**88.9%**

network architecture	test error (cm)
softmax + feature points (**ours**)	**1.30 ± 0.73**
softmax + fully connected layer	2.59 ± 1.19
fully connected layer	4.75 ± 2.29
max-pooling + fully connected	3.71 ± 1.73

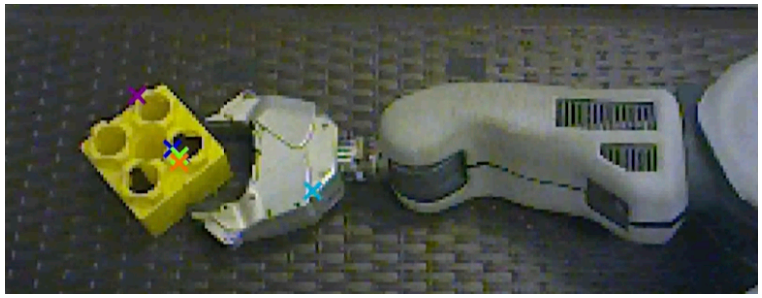


Meeussen et al. (Willow Garage)

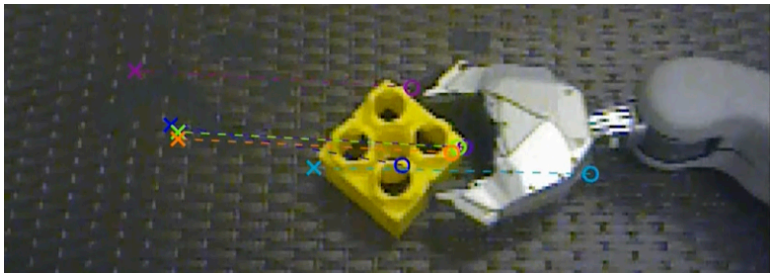


Visuomotor Learning Directly in Visual Space ?

Provide image that defines goal



Train controller in visual feature space

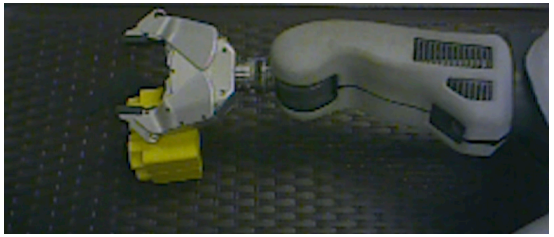


[Finn, Tan, Duan, Darrell, Levine, Abbeel, 2015]

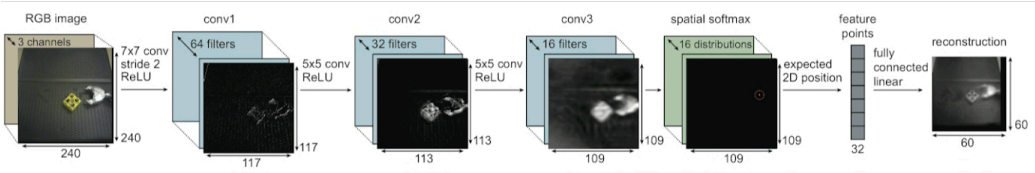
Visuomotor Learning Directly in Visual Space

1. Set target end-effector pose

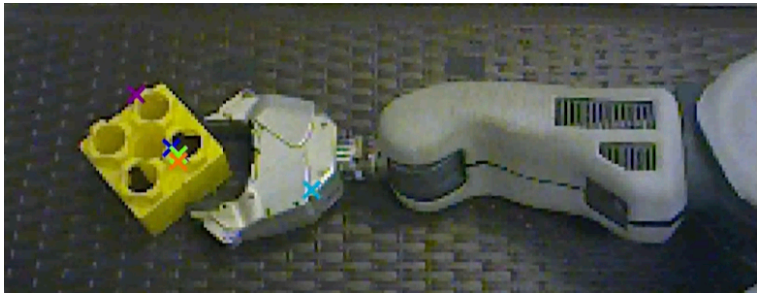
2. Train exploratory non-vision controller



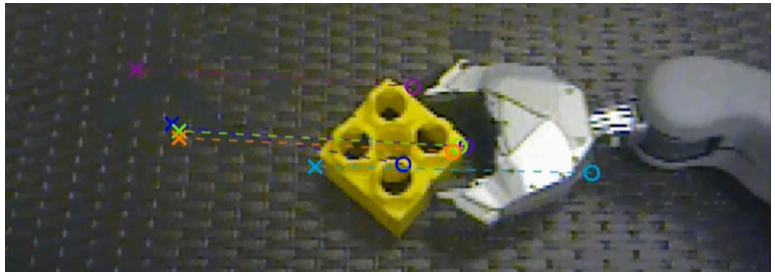
3. Learning visual features with collected images



4. Provide image that defines goal features



5. Train final controller in visual feature space



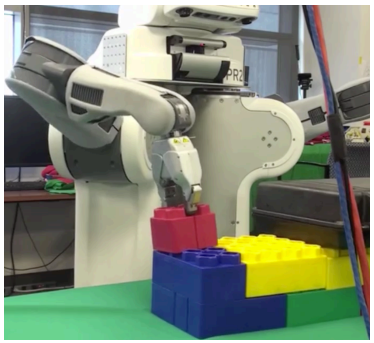
[Finn, Tan, Duan, Darrell, Levine, Abbeel, 2015]

Visuomotor Learning Directly in Visual Space

[Finn, Tan, Duan, Darrell, Levine, Abbeel, 2015]

Frontiers: Applications

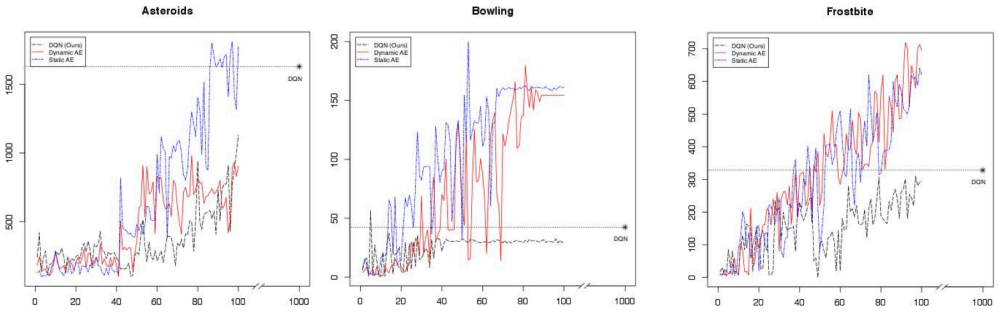
- Vision-based flight
- Locomotion
- Manipulation
- Natural language interaction
- Dialogue
- Program analysis



Frontiers: Foundations

- Shared and transfer learning

- Exploration



- Memory
  - Estimation
  - Temporal hierarchy / goal setting
- Tools / Experimentation
  - Stochastic computation graphs
  - Computation graph toolkit (CGT)