



Lecture 17: Imitation Learning

Bolei Zhou

The Chinese University of Hong Kong

Today's Outline

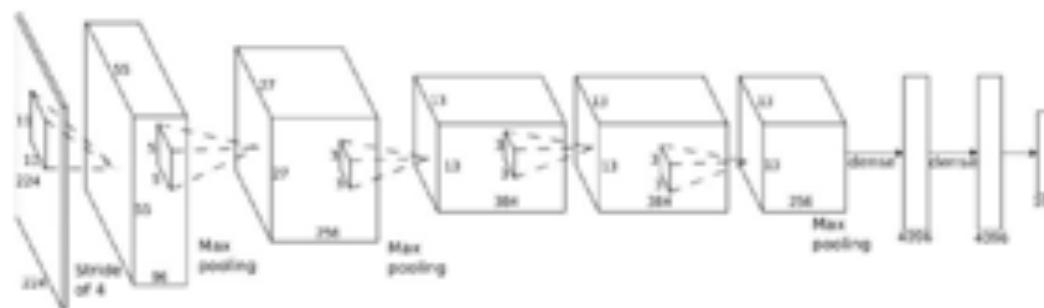
- Introduction on Imitation Learning
- Improving the model of the imitation learning
- Unifying imitation learning and reinforcement learning
- Case studies

Introduction on Imitation Learning

- Supervised learning of the policy network



\mathbf{o}_t

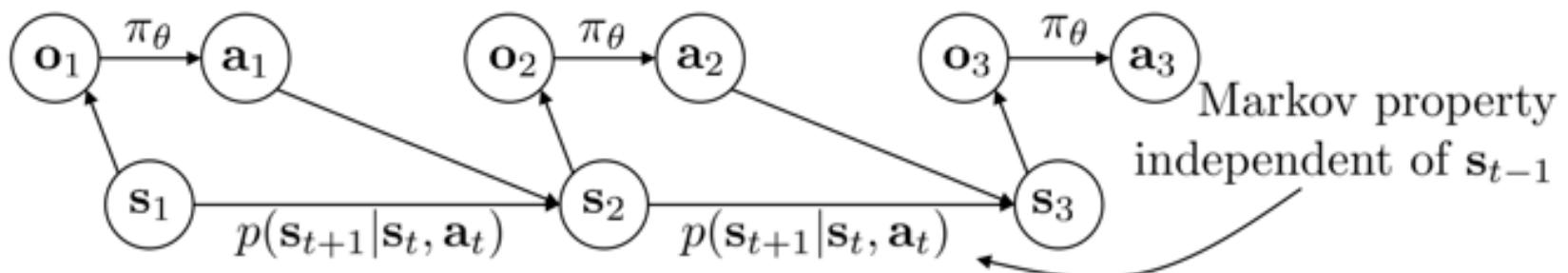


$\pi_\theta(\mathbf{a}_t | \mathbf{o}_t)$

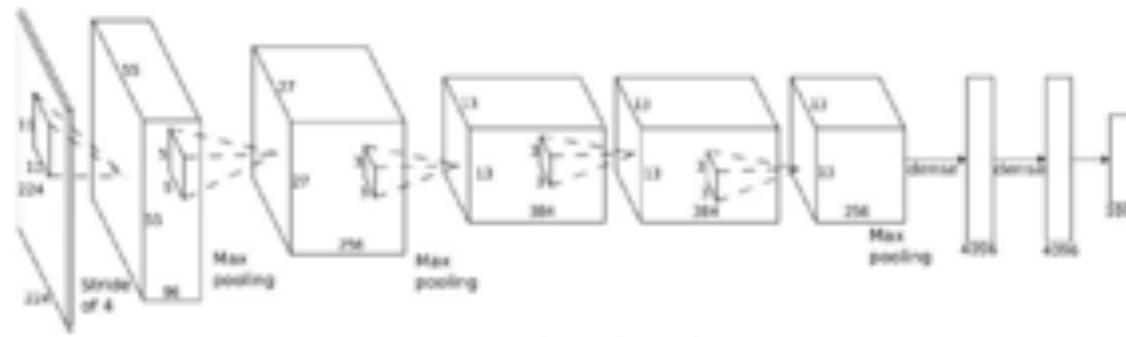


\mathbf{a}_t

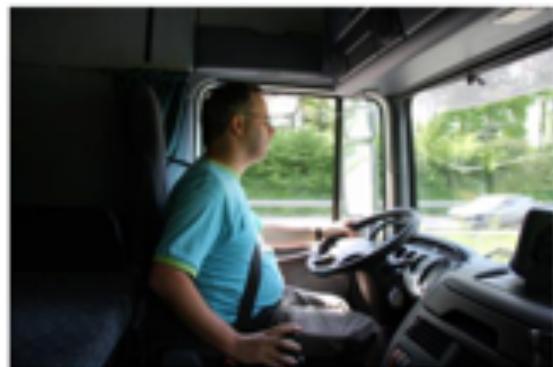
Providing the action ground-truth



Introduction on Supervised Imitation Learning



$$\pi_{\theta}(\mathbf{a}_t | \mathbf{o}_t)$$



\mathbf{o}_t
 \mathbf{a}_t

training
data

supervised
learning

$$\pi_{\theta}(\mathbf{a}_t | \mathbf{o}_t)$$

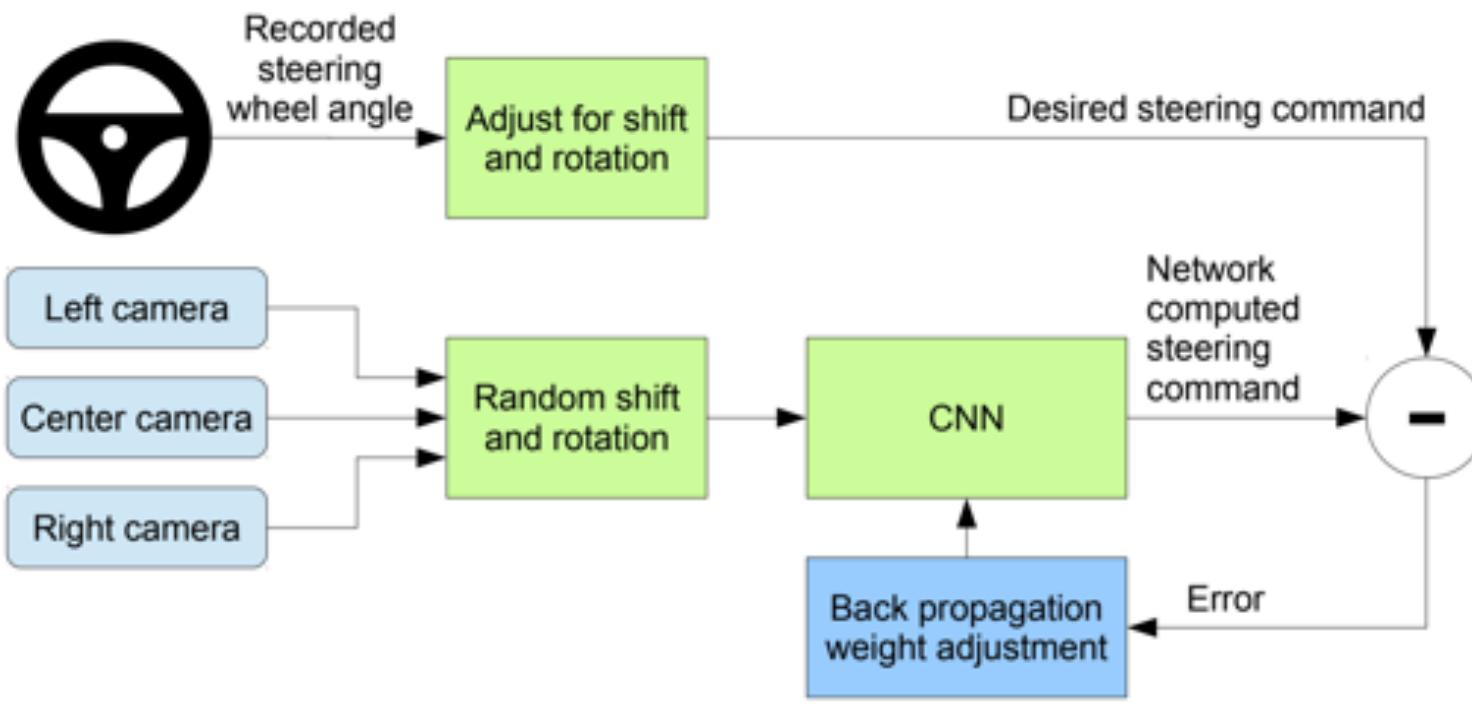
Case Study: End to End Learning for Self-Driving Cars

- DAVER-2 System By Nvidia: It works to some degree!



Case Study: End to End Learning for Self-Driving Cars

- 72 hours of driving data as training data



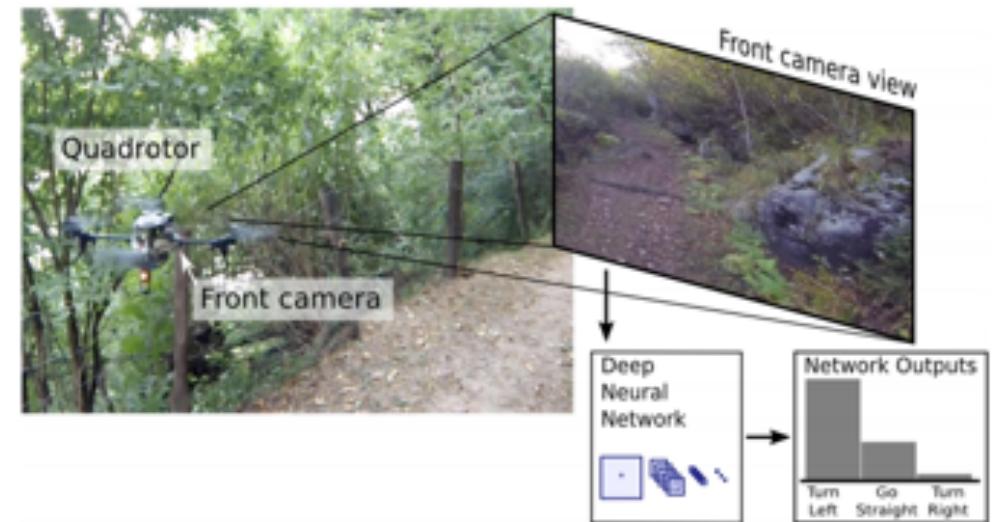
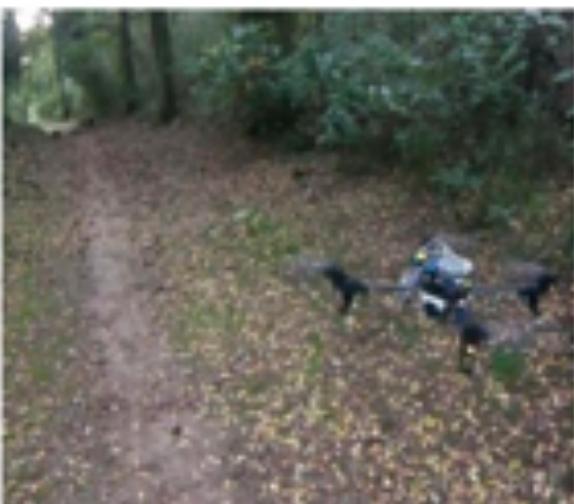
Example code of behavior clone: <https://github.com/Az4z3l/CarND-Behavioral-Cloning>



<https://www.youtube.com/watch?feature=oembed&v=Rmv643N3-yM>

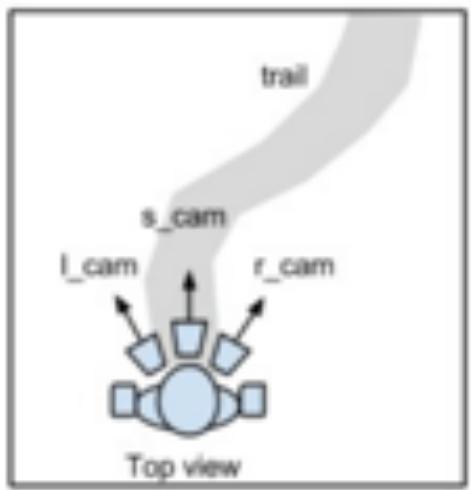
Case Study: Trail following as a classification

How to train a drone to fly

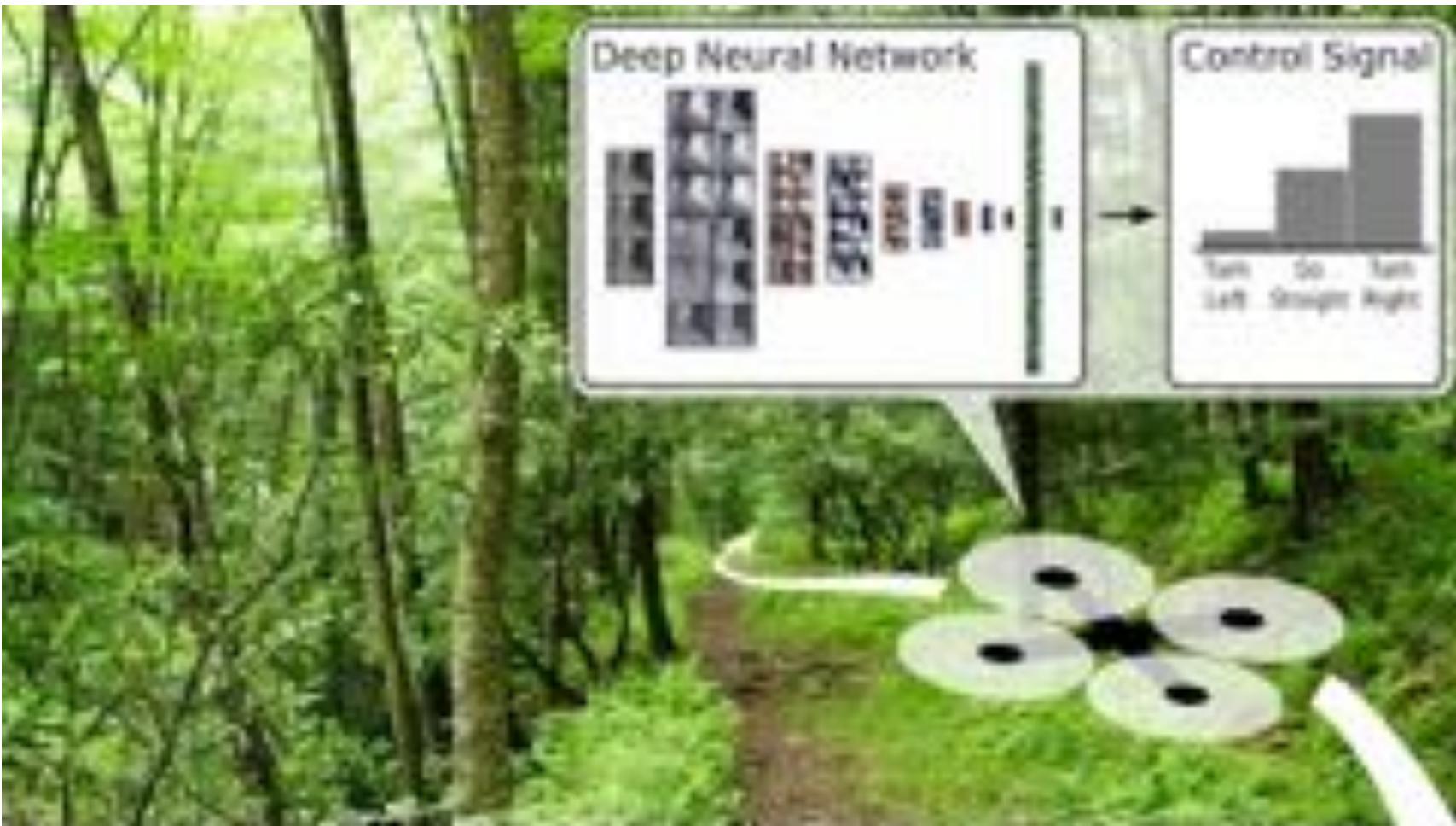


Case Study: Trail following as a classification

Human walking data



Case Study: Trail following as a classification



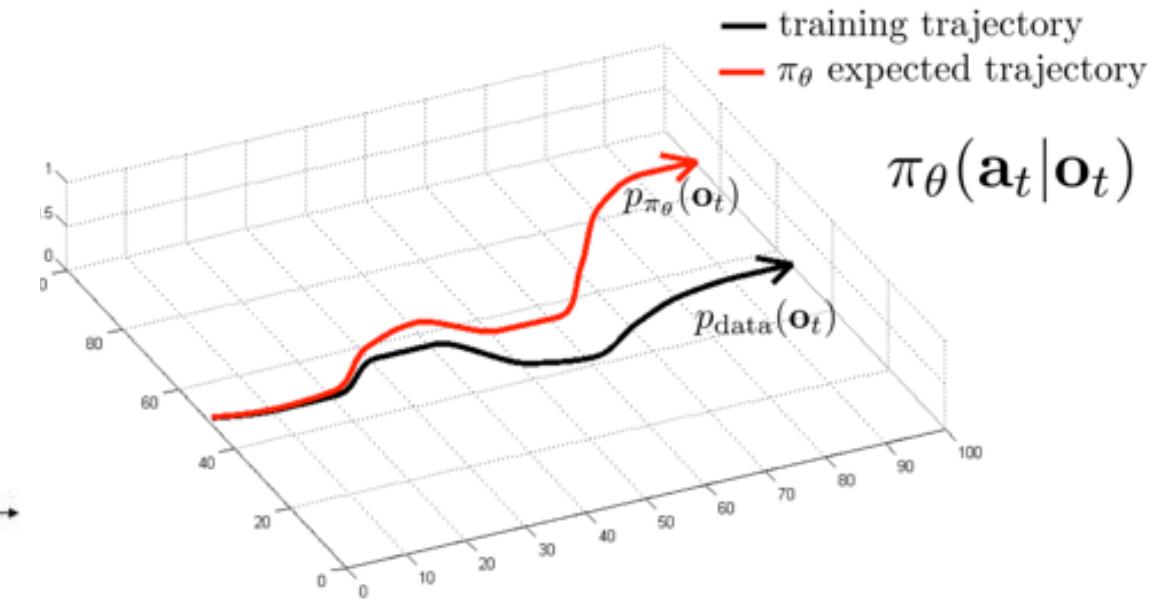
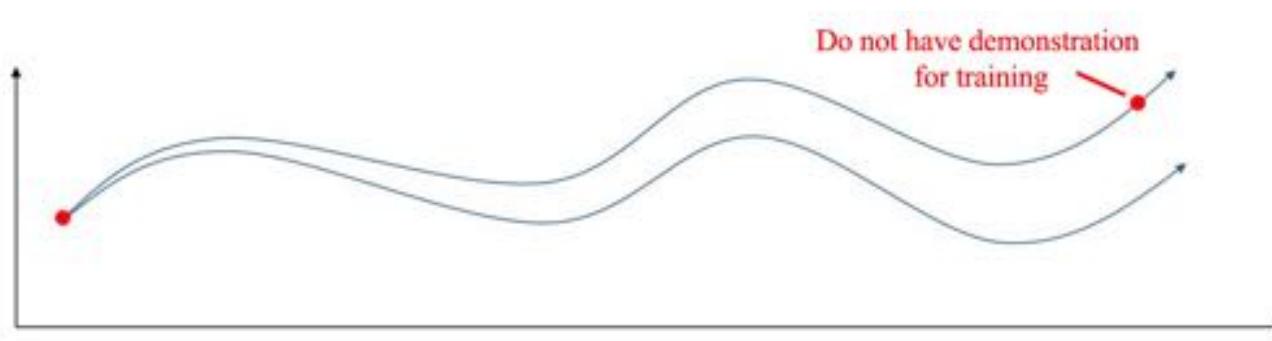
<https://youtu.be/umRdt3zGgpU>

Limitations of Supervised Imitation Learning

- The trained policy run into the off-course situations
- Mistakes grows quadratically in T: don't learn how to recover from failure

$$J(\hat{\pi}_{\text{sup}}) \leq T^2 \varepsilon$$

[Ross 2012]

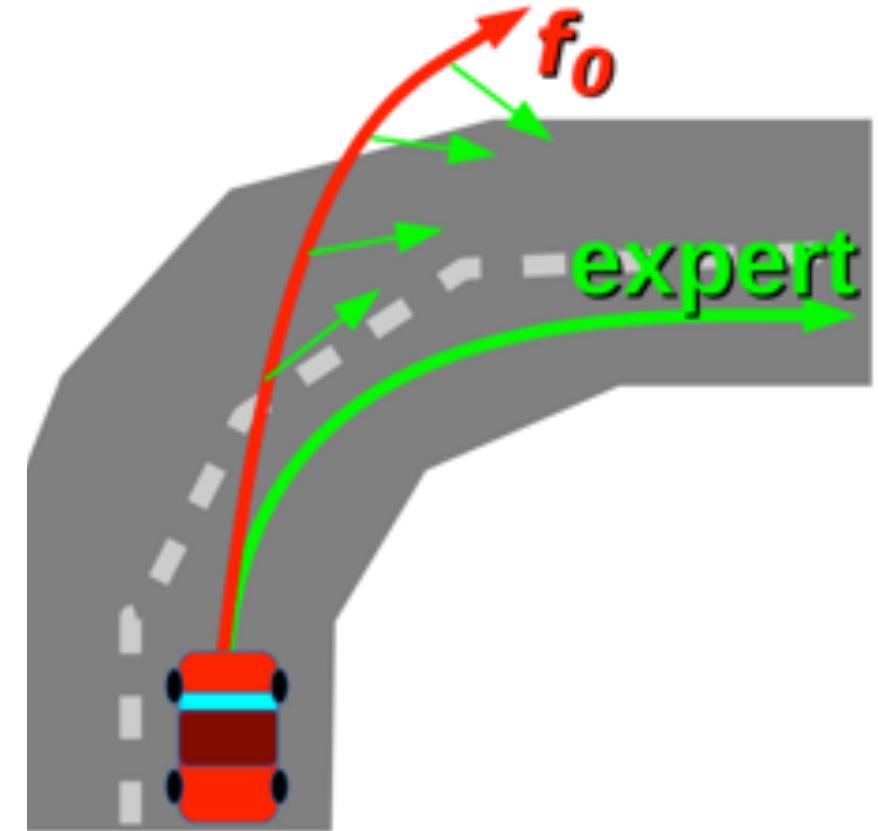


can we make $p_{\text{data}}(\mathbf{o}_t) = p_{\pi_\theta}(\mathbf{o}_t)$?

Annotating More On-Policy Data Iteratively

We can collect more training samples for those off-expert situations.

- run our current policy and observe.
- Then ask the human experts to label the possible actions again.
- This allows us to collect samples that we miss when we deploy the policy.



DAgger: Dataset Aggregation

can we make $p_{\text{data}}(\mathbf{o}_t) = p_{\pi_\theta}(\mathbf{o}_t)$?

idea: instead of being clever about $p_{\pi_\theta}(\mathbf{o}_t)$, be clever about $p_{\text{data}}(\mathbf{o}_t)$!

DAgger: Dataset Aggregation

goal: collect training data from $p_{\pi_\theta}(\mathbf{o}_t)$ instead of $p_{\text{data}}(\mathbf{o}_t)$

how? just run $\pi_\theta(\mathbf{a}_t | \mathbf{o}_t)$

but need labels \mathbf{a}_t !

- 
1. train $\pi_\theta(\mathbf{a}_t | \mathbf{o}_t)$ from human data $\mathcal{D} = \{\mathbf{o}_1, \mathbf{a}_1, \dots, \mathbf{o}_N, \mathbf{a}_N\}$
 2. run $\pi_\theta(\mathbf{a}_t | \mathbf{o}_t)$ to get dataset $\mathcal{D}_\pi = \{\mathbf{o}_1, \dots, \mathbf{o}_M\}$
 3. Ask human to label \mathcal{D}_π with actions \mathbf{a}_t
 4. Aggregate: $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_\pi$

Limitation with DAgger

- Need to query expert many times during the training
- Very expansive if it is human expert

- 
1. train $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ from human data $\mathcal{D} = \{\mathbf{o}_1, \mathbf{a}_1, \dots, \mathbf{o}_N, \mathbf{a}_N\}$
 2. run $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ to get dataset $\mathcal{D}_\pi = \{\mathbf{o}_1, \dots, \mathbf{o}_M\}$
 3. Ask human to label \mathcal{D}_π with actions \mathbf{a}_t
 4. Aggregate: $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_\pi$

DAgger with Other Expert Algorithms

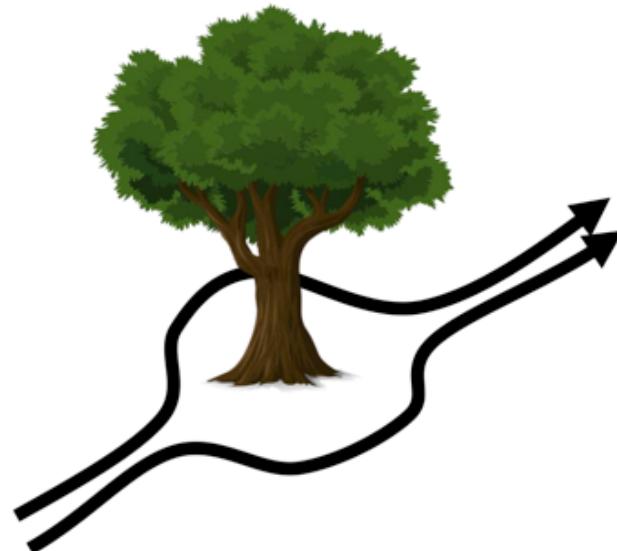
- Learn (quickly) to imitate other slow algorithms
- Game playing: we can compute suboptimal expert behavior with brute-force search offline, then use it during training time for a policy function to approximate
- Discrete optimizers: Many discrete optimization problems such as shortest path are very slow. Then use imitation learning to try to build shortest path optimizers that will run sufficiently efficiently in real time.

- 
1. train $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ from human data $\mathcal{D} = \{\mathbf{o}_1, \mathbf{a}_1, \dots, \mathbf{o}_N, \mathbf{a}_N\}$
 2. run $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ to get dataset $\mathcal{D}_\pi = \{\mathbf{o}_1, \dots, \mathbf{o}_M\}$
 3. Ask human to label \mathcal{D}_π with actions \mathbf{a}_t Ask other algorithm!
 4. Aggregate: $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_\pi$

Improving the Supervised Imitation Learning

How can we improve the policy model?

- Multimodal behavior
- Non-Markovian behavior

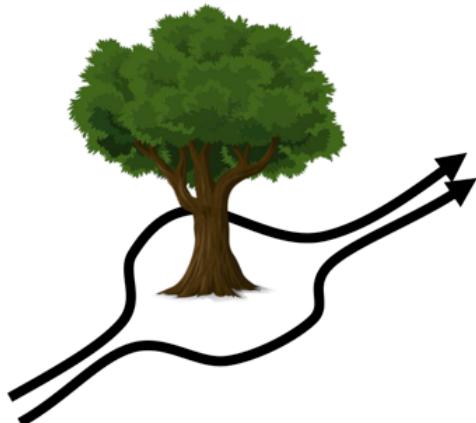


$$\pi_{\theta}(\mathbf{a}_t | \mathbf{o}_t)$$

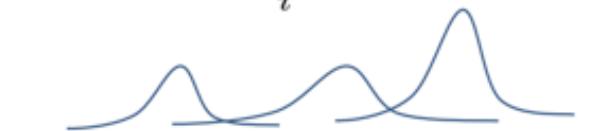
behavior depends only
on current observation

Improving the Supervised Imitation Learning

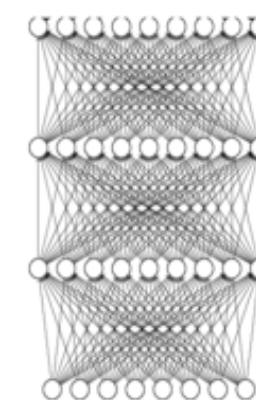
- Handling multimodal behavior
 - Output mixture of Gaussians
 - Latent variable models
 - Autoregressive discretization



$$\pi(\mathbf{a}|\mathbf{o}) = \sum_i w_i \mathcal{N}(\mu_i, \Sigma_i)$$



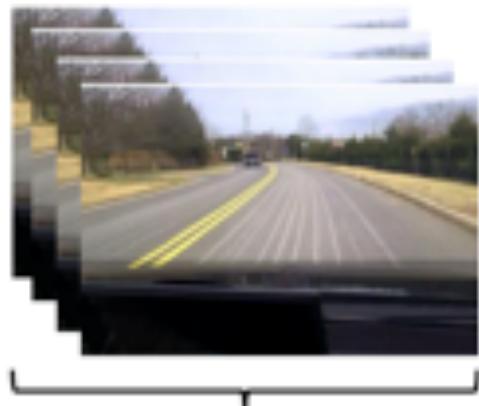
$$w_1, \mu_1, \Sigma_1, \dots, w_N, \mu_N, \sigma_N$$



Modeling the whole history

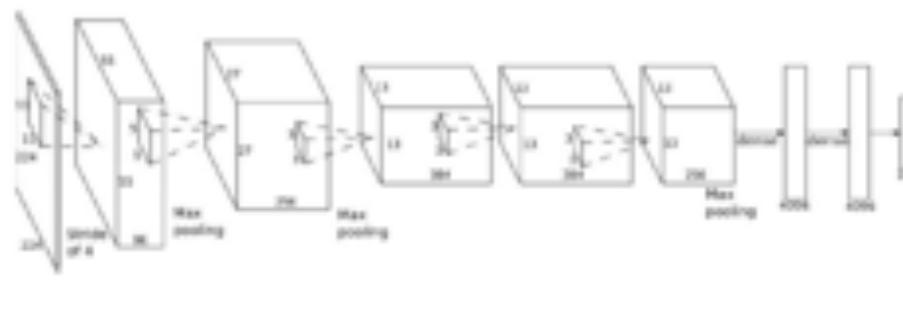
$$\pi_{\theta}(\mathbf{a}_t | \mathbf{o}_t)$$

behavior depends only
on current observation



$$\pi_{\theta}(\mathbf{a}_t | \mathbf{o}_1, \dots, \mathbf{o}_t)$$

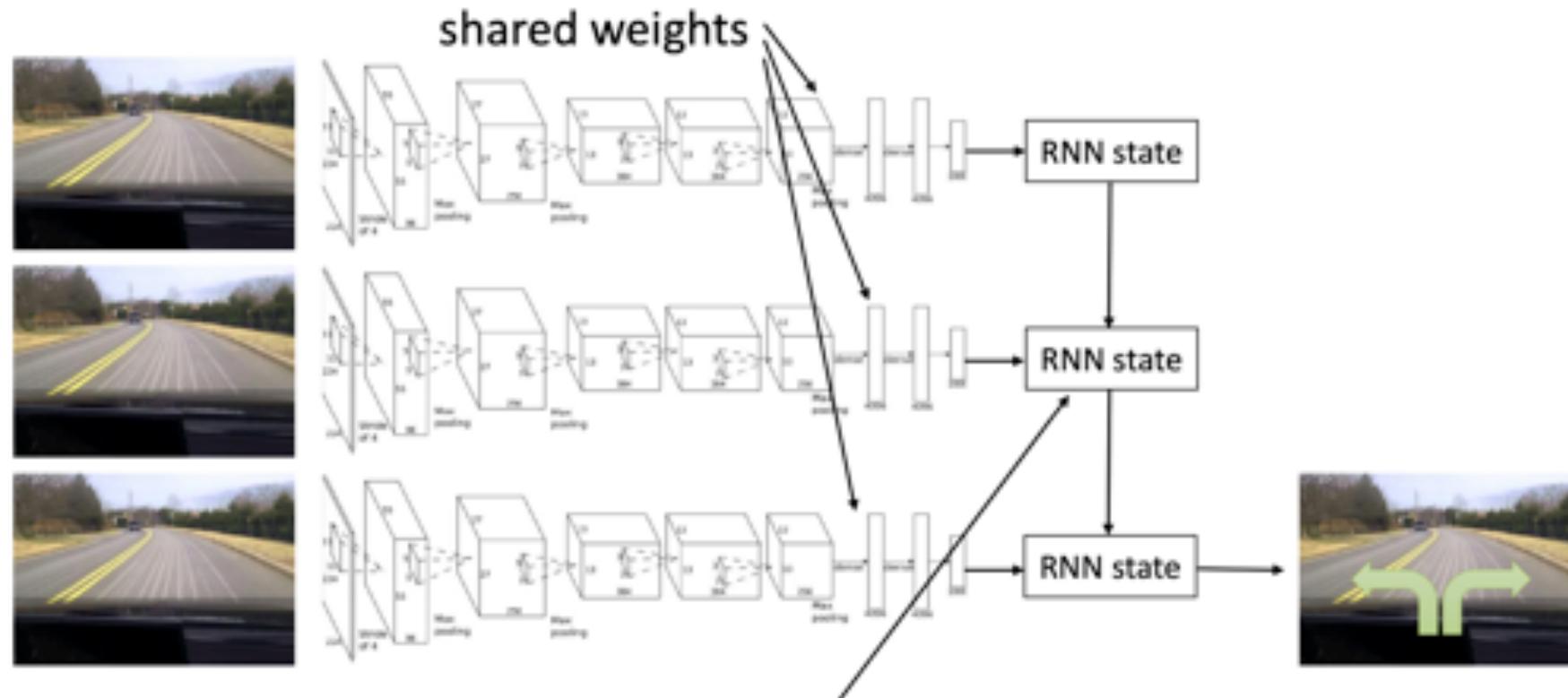
behavior depends on
all past observations



variable number of frames,
too many weights

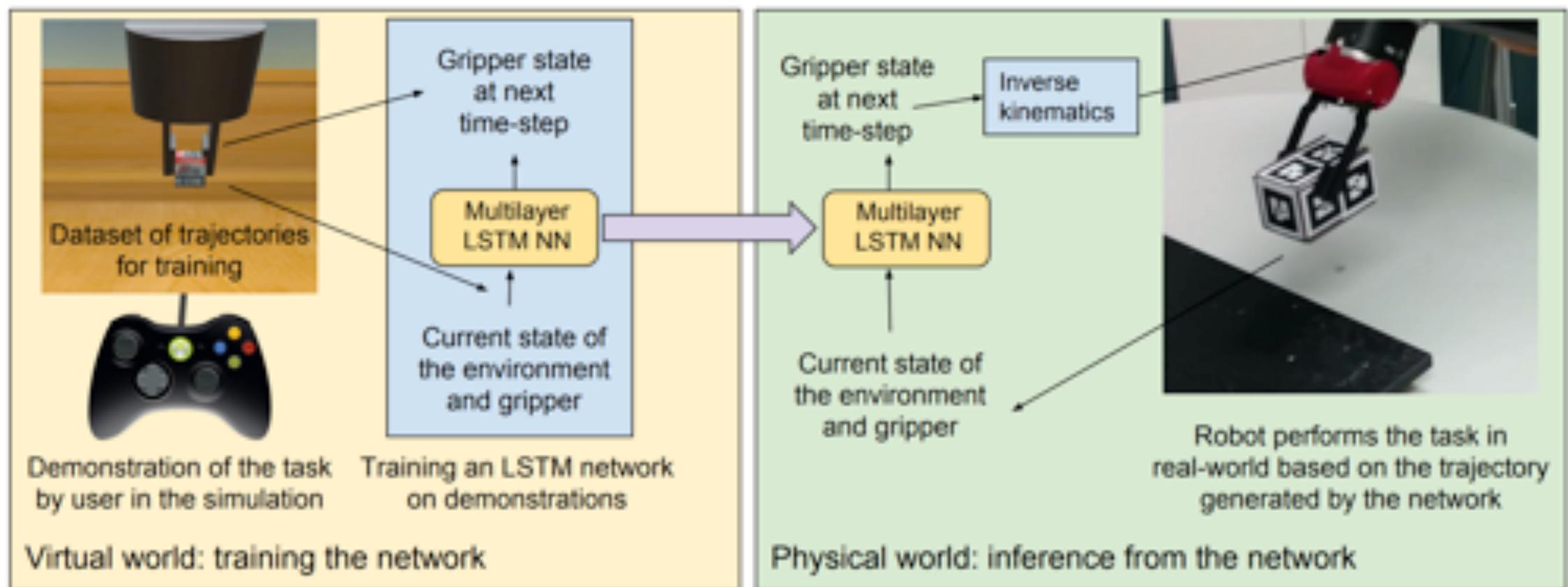
Modeling the whole history

- We can use a LSTM sequential model to encode the history

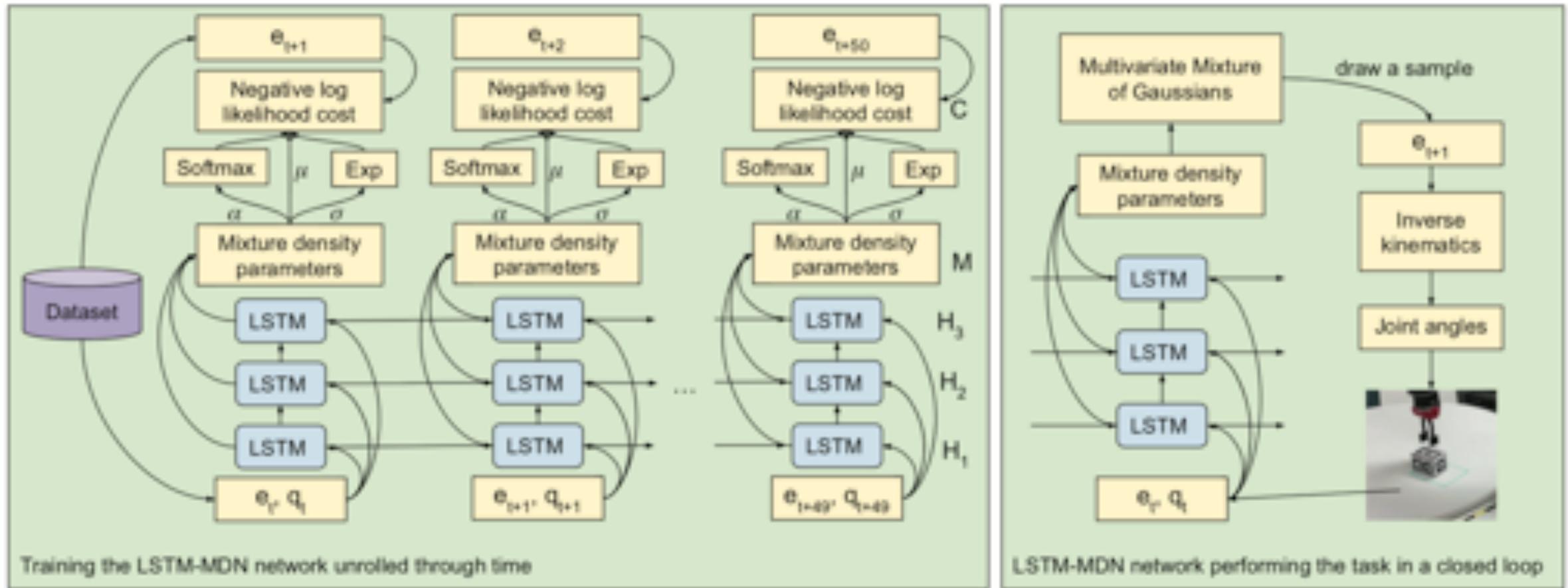


Typically, LSTM cells work better here

Case Study: Learning from demonstration using LSTM



Case Study: Learning from demonstration using LSTM



Rahmatizadeh et al. Learning real manipulation tasks from virtual demonstrations using LSTM. AAAI 2018

Scale up the data collection of human demonstration?

- Rahmatizadeh et al manually collected 650 demonstrations for Pick and place, and 1614 for Push to pose
- Make it like how we collect ImageNet through Crowd-sourcing!

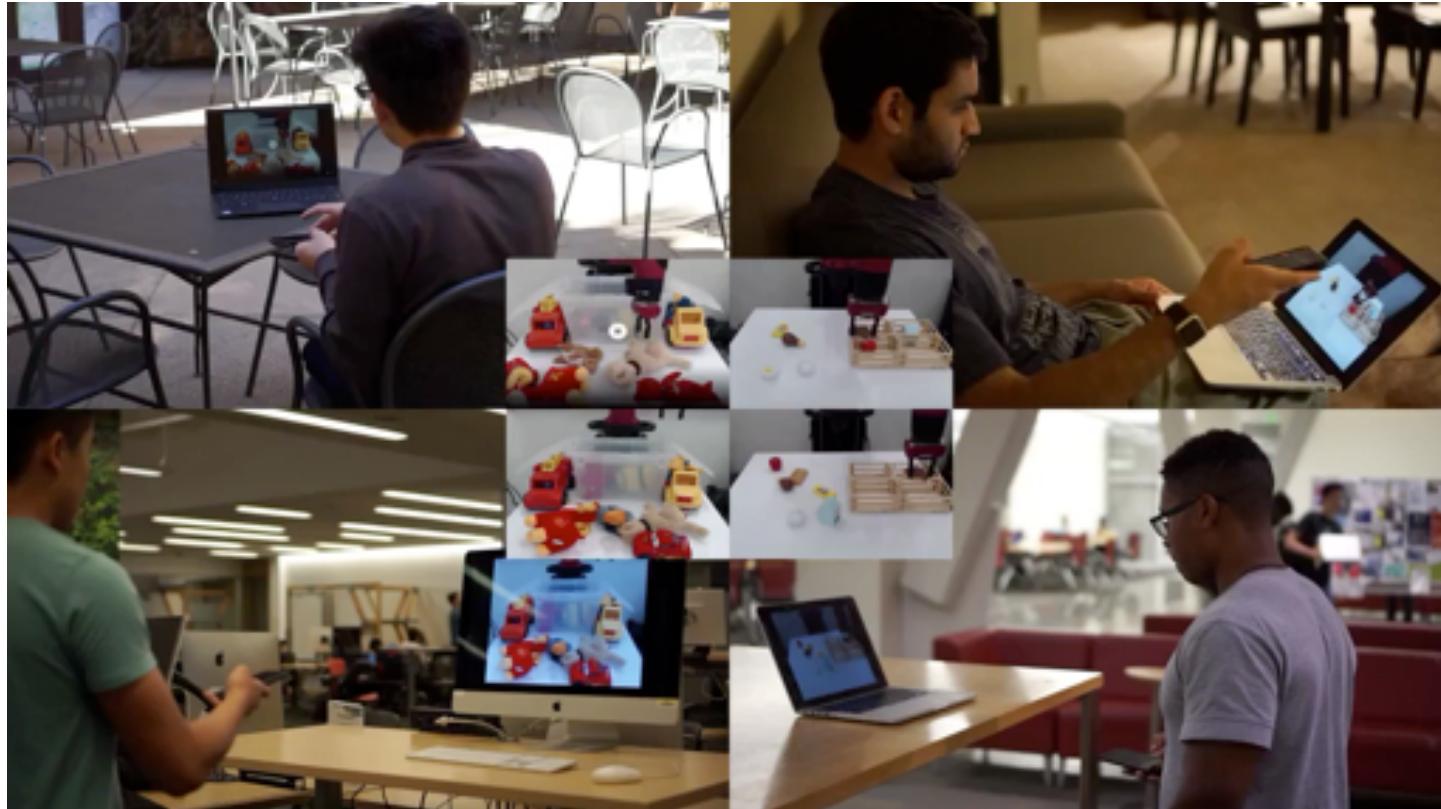


Gathering Demonstrations



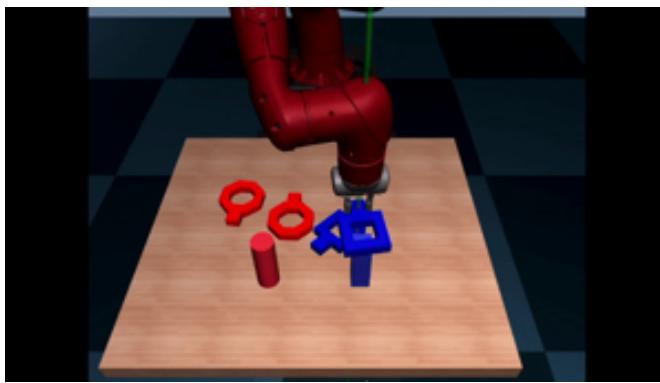
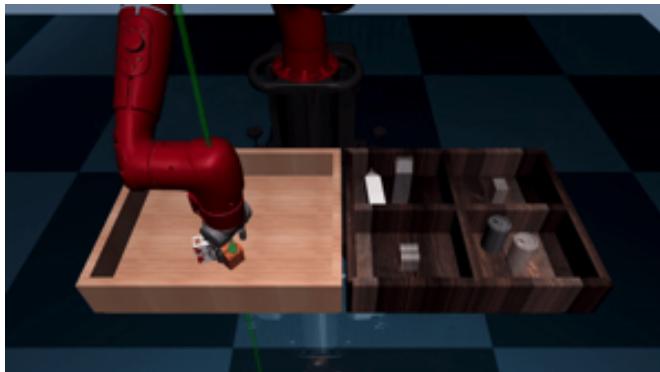
RoboTurk: Crowdsourcing Robotic Demonstrations

- RoboTurk is a crowdsourcing platform that is collecting human demonstrations of tasks such as “picking” and “assembly”;



RoboTurk: Crowdsourcing Robotic Demonstrations

- 137.5 hours of demonstration for two tasks
- 1071 successful picking demonstrations and 1147 assembly demonstrations



Quantitative Results					
Task	Number of Demonstrations				
	0	1	10	100	1000
Bin Picking (Can)	0±0	278±351	273±417	385±466	641±421
Assembly (Round)	0±0	381±467	663±435	575±470	775±388

Problems with Imitation Learning

- Humans need to provide data, which is typically in a limited amount
- Humans are not good at providing some kinds of actions
- Human can learn autonomously (unsupervised learning)



Imitation Learning vs. Reinforcement Learning

Imitation Learning

- Requires demonstrations
- Issue of distributional shift
- Simple stable supervised learning
- Only as good as the demo

Reinforcement Learning

- Requires reward function
- Must address exploration
- Potentially non-convergent
- Can become superhuman good

Can we get the best of both worlds?

What if we can have both demonstrations and rewards

Simplest Combination: Pretrain & Finetune

- Use the expert demonstration to initialize a policy (overcome exploration)
- Then apply RL to improve the policy and learn to deal with those off-course scenarios and go beyond performance of the demonstrator

1. collected demonstration data $(\mathbf{s}_i, \mathbf{a}_i)$  use demonstratin to help exploration
2. initialize π_θ as $\max_\theta \sum_i \log \pi_\theta(\mathbf{a}_i | \mathbf{s}_i)$
3. run π_θ to collect experience improve better than demonstration
4. improve π_θ with any RL algorithm

Simplest Combination: Pretrain & Finetune

Pretrain & finetune

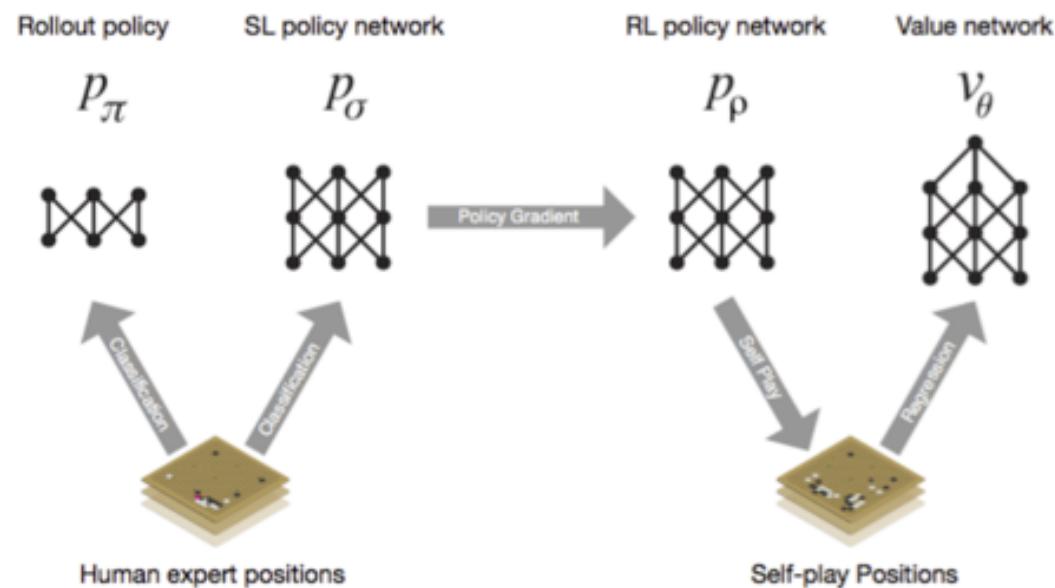
1. collected demonstration data $(\mathbf{s}_i, \mathbf{a}_i)$
2. initialize π_θ as $\max_\theta \sum_i \log \pi_\theta(\mathbf{a}_i | \mathbf{s}_i)$
-  3. run π_θ to collect experience
4. improve π_θ with any RL algorithm

vs. DAgger

- 
1. train $\pi_\theta(\mathbf{a}_t | \mathbf{o}_t)$ from human data $\mathcal{D} = \{\mathbf{o}_1, \mathbf{a}_1, \dots, \mathbf{o}_N, \mathbf{a}_N\}$
 2. run $\pi_\theta(\mathbf{a}_t | \mathbf{o}_t)$ to get dataset $\mathcal{D}_\pi = \{\mathbf{o}_1, \dots, \mathbf{o}_M\}$
 3. Ask human to label \mathcal{D}_π with actions \mathbf{a}_t
 4. Aggregate: $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_\pi$

Pretrain & Finetune for AlphaGo

- SL Policy Network: Train the policy network on 30 million moves from games played by human experts (could predict the human move 57% of the time)
- RL Policy Network: Then finetune the weights with policy gradients



Pretrain & Finetune for Starcraft2

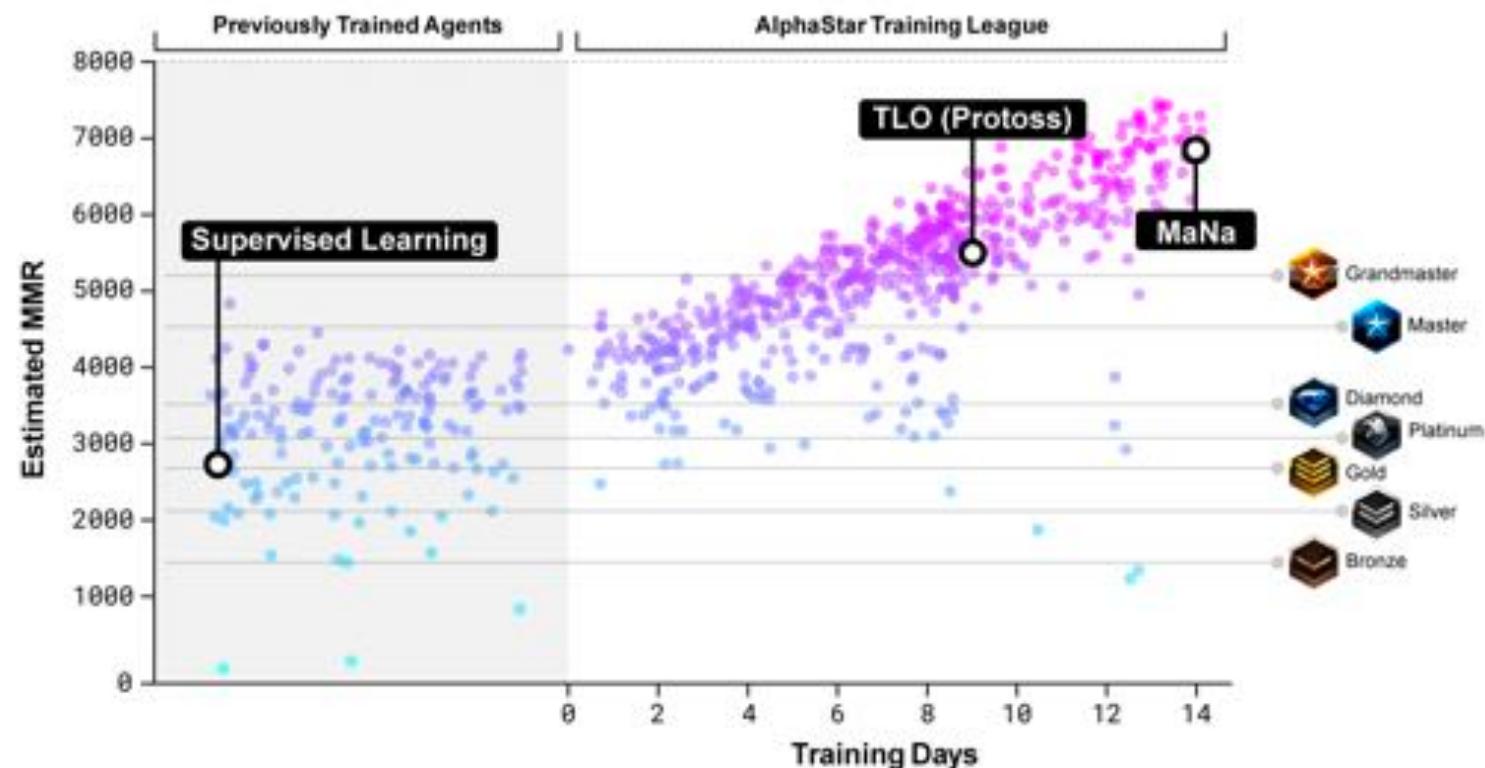
- Supervised learning from anonymised human games released by Blizzard.
- This allowed AlphaStar to learn, by imitation, the basic micro and macro-strategies used by players on the StarCraft ladder.
- This initial agent defeated the built-in “Elite” level AI - around gold level for a human player - in 95% of games.



<https://deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starcraft-ii/>

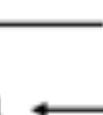
Pretrain & Finetune for Starcraft2

- Population-based and multi-agent reinforcement learning



<https://deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starcraft-ii/>

Problem with the Pretrain & Finetune

1. collected demonstration data $(\mathbf{s}_i, \mathbf{a}_i)$
2. initialize π_θ as $\max_\theta \sum_i \log \pi_\theta(\mathbf{a}_i | \mathbf{s}_i)$
3. run π_θ to collect experience  can be very bad (due to distribution shift)
4. improve π_θ with any RL algorithm  first batch of (very) bad data can destroy initialization

Can we avoid forgetting the demonstrations?

Solution: Off-policy Reinforcement Learning

- Off-policy RL can use any data
- We can use demonstrations as off-policy samples
 - Since demonstrations are provided as data in every iteration, they are never forgotten
 - Policy can still become better than the demos, since it is not forced to mimic them
- Off-policy policy gradient (with importance sampling)
- Off-policy Q-learning

Policy gradient with demonstrations

- Importance sampling on policy gradient

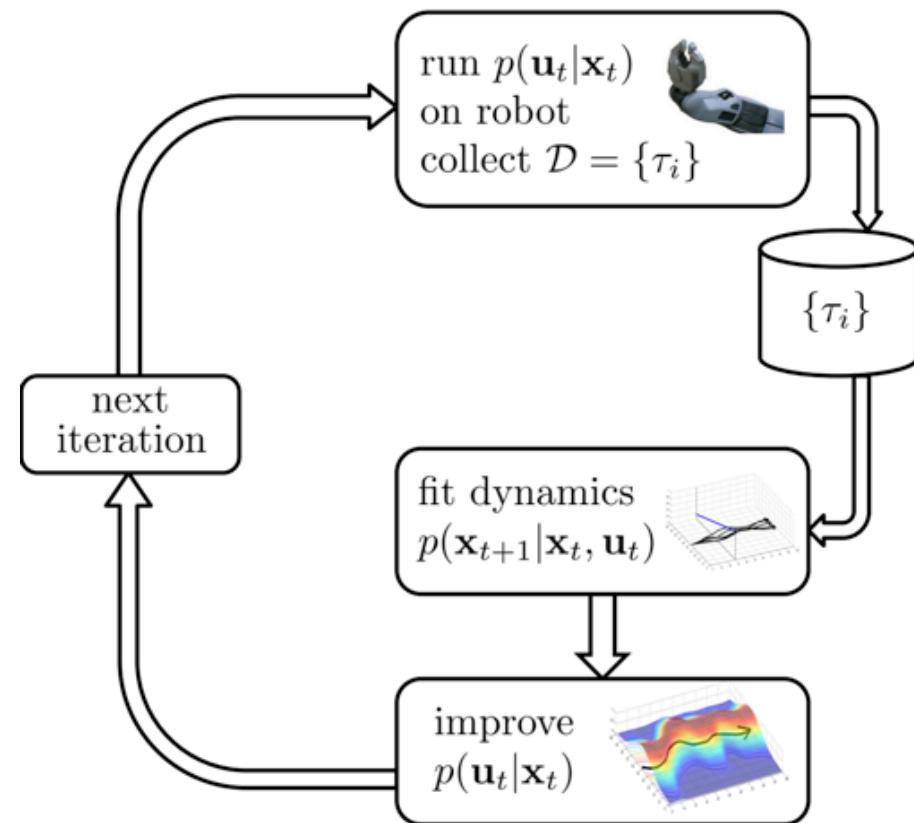
$$\nabla_{\theta} J(\theta) = \sum_{\tau \in \mathcal{D}} \left[\sum_{t=1}^T \nabla_{\theta} \log \pi_{\theta}(\mathbf{a}_t | \mathbf{s}_t) \left(\prod_{t'=1}^t \frac{\pi_{\theta}(\mathbf{a}_{t'} | \mathbf{s}_{t'})}{q(\mathbf{a}_{t'} | \mathbf{s}_{t'})} \right) \left(\sum_{t'=t}^T r(\mathbf{s}_{t'}, \mathbf{a}_{t'}) \right) \right]$$

from experience and demonstration

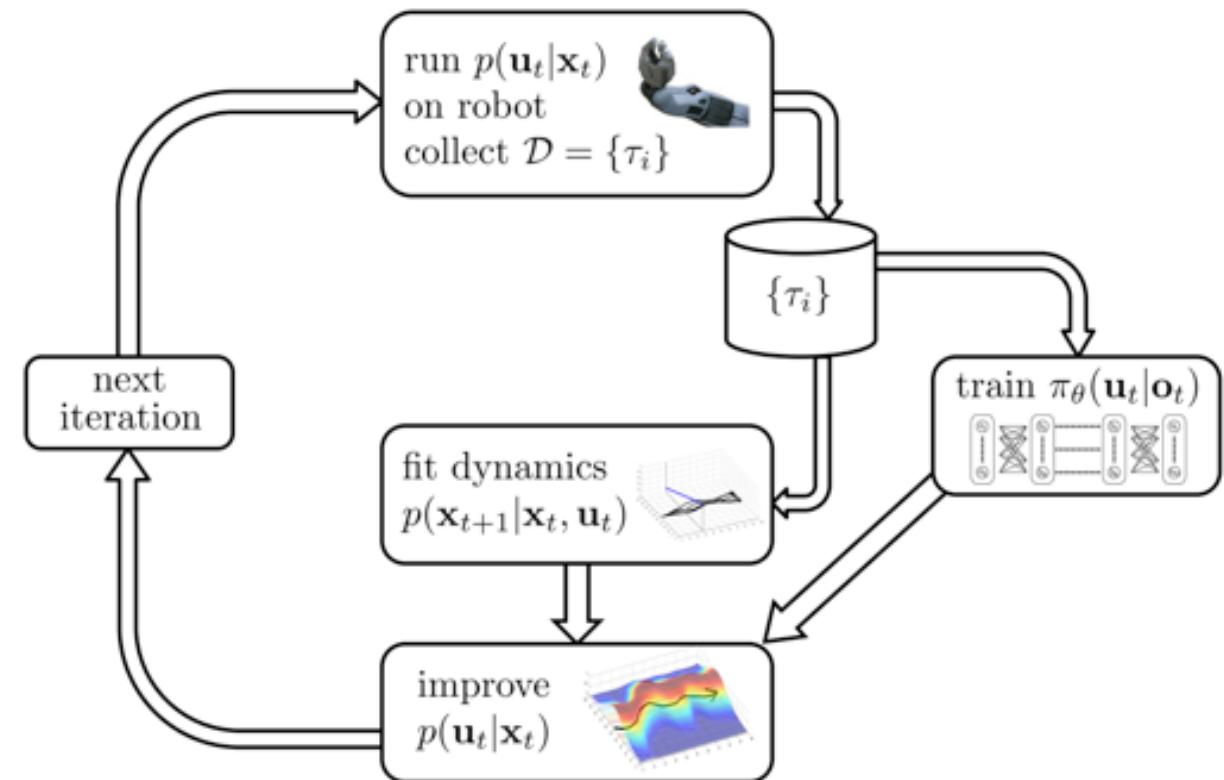
- Problem: which distribution did the demonstrations come from?
- Option 1: use supervised behavior cloning to approximate π_{demo}
- Option 2: assume Diract delta: $\pi_{demo}(\tau) = \frac{1}{N} \delta(\tau \in \mathcal{D})$

Guided Policy Search

Optimal control for trajectory optimization



Guide the training of policy network



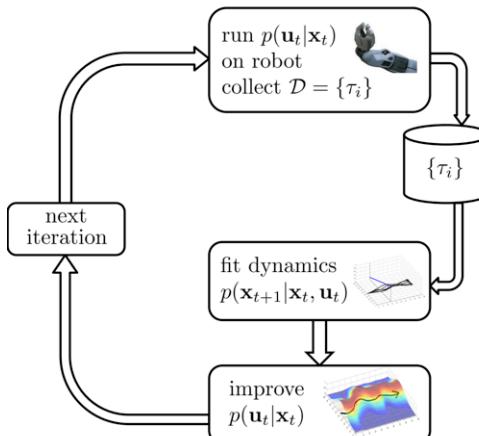
Review on model-based RL

- Optimal control for trajectory optimization (f is given)

$$\min_{u_1, \dots, u_t} \sum_{t=1}^T c(x_t, u_t) \text{ s.t. } x_t = f(x_{t-1}, u_{t-1})$$

- If f is not given, we learn locally linear models and use these models to solve for a time-varying linear Gaussian controller

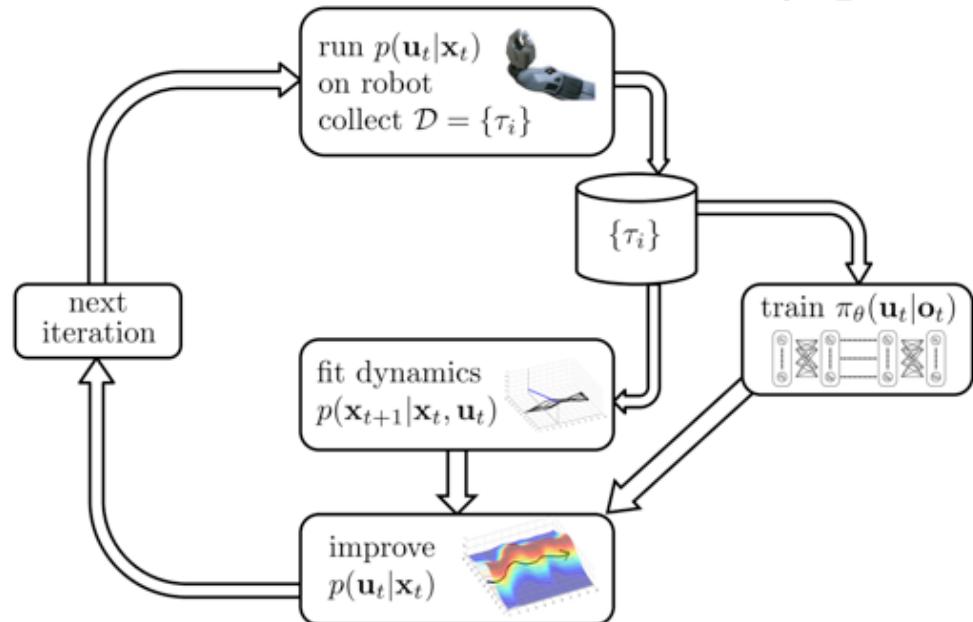
$$p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t) = \mathcal{N}(f(\mathbf{x}_t, \mathbf{u}_t))$$
$$f(\mathbf{x}_t, \mathbf{u}_t) = \mathbf{A}_t \mathbf{x}_t + \mathbf{B}_t \mathbf{u}_t$$



Guided policy search (GPS) formulation

$$\min_{u_1, \dots, u_t} \sum_{t=1}^T c(x_t, u_t) \text{ s.t. } x_t = f(x_{t-1}, u_{t-1}) \text{ s.t. } u_t = \pi_\theta(x_t)$$

$$\min_{p(\tau)} \sum_{t=1}^T E_{p(\mathbf{x}_t, \mathbf{u}_t)} [c(\mathbf{x}_t, \mathbf{u}_t) + \mathbf{u}_t^T \lambda_t + \rho_t D_{KL}(p(\mathbf{u}_t | \mathbf{x}_t) \| \pi_\theta(\mathbf{u}_t | \mathbf{x}_t))]$$

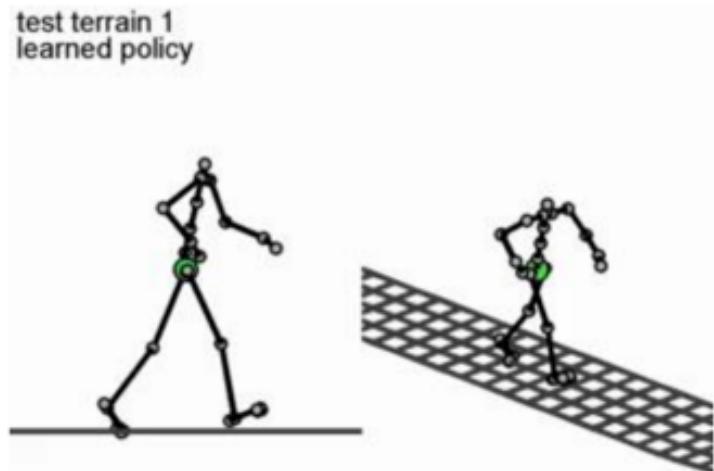


Dual gradient descend

1. Start with some initial choice of λ (by λ , we include λ_t corresponding to each time step)
2. Assign $\tau \leftarrow \arg \min_{\tau} \mathcal{L}(\tau, \theta, \lambda)$.
3. Assign $\theta \leftarrow \arg \min_{\theta} \mathcal{L}(\tau, \theta, \lambda)$.
4. Compute $\frac{dg}{d\lambda} = \frac{d\mathcal{L}}{d\lambda}(\tau, \theta, \lambda)$. Take a gradient step $\lambda \leftarrow \lambda + \alpha \frac{dg}{d\lambda}$
5. Repeat steps 2-4.

Guided policy search (GPS) Applications

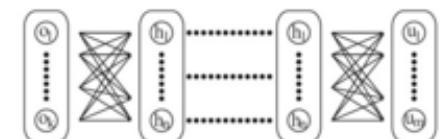
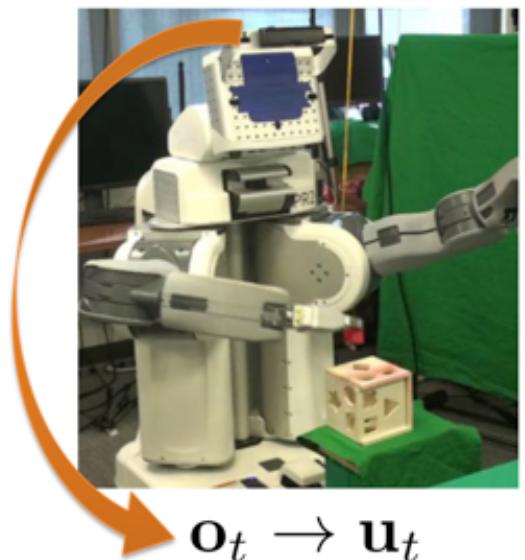
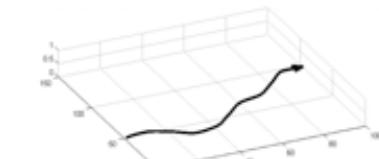
First demo of GPS:



<http://graphics.stanford.edu/projects/gpspaper/humanoid.mp4>
<http://graphics.stanford.edu/projects/gpspaper/video.mp4>

Levine and Koltun. ICML'13

End-to-End Training of Deep Visuomotor Policies

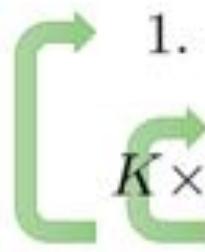


Sergey Levine, Chelsea Finn, Trevor Darrell,
Pieter Abbeel. JMLR'16

Q-learning with demonstrations

- Q-learning is already off-policy, no need to bother with importance weighting
- Simple solution: throw demonstrations into the replay buffer

initialize \mathcal{B} to contain the demonstration data

- 
1. collect dataset $\{(\mathbf{s}_i, \mathbf{a}_i, \mathbf{s}'_i, r_i)\}$ using some policy, add it to \mathcal{B}
 2. sample a batch $(\mathbf{s}_i, \mathbf{a}_i, \mathbf{s}'_i, r_i)$ from \mathcal{B}
 3. $\phi \leftarrow \phi - \alpha \sum_i \frac{dQ_\phi}{d\phi}(\mathbf{s}_i, \mathbf{a}_i)(Q_\phi(\mathbf{s}_i, \mathbf{a}_i) - [r(\mathbf{s}_i, \mathbf{a}_i) + \gamma \max_{\mathbf{a}'} Q_\phi(\mathbf{s}'_i, \mathbf{a}'_i)])$

Q-learning with demonstrations



<https://youtu.be/WGJwLfeVN9w>

Imitation learning as an auxiliary loss function

- Hybrid objective: we can combine RL and imitation learning objective

$$\frac{E_{\pi_\theta}[r(\mathbf{s}, \mathbf{a})] + \lambda \sum_{(\mathbf{s}, \mathbf{a}) \in \mathcal{D}_{\text{demo}}} \log \pi_\theta(\mathbf{a}|\mathbf{s})}{\text{RL} \qquad \qquad \qquad \text{Imitation learning}}$$

Issue: The weight needs careful tuning

Hybrid policy gradient

$$g_{aug} = \sum_{(s,a) \in \rho_\pi} \nabla_\theta \ln \pi_\theta(a|s) A^\pi(s,a) + \sum_{(s,a^*) \in \rho_D} \nabla_\theta \ln \pi_\theta(a^*|s) w(s,a^*)$$

policy gradient make policy closer to demonstrations

weighting function $w(s,a) = \lambda_0 \lambda_1^k \max_{(s',a') \in \rho_\pi} A^\pi(s',a') \quad \forall (s,a) \in \rho_D$



Hybrid Q-Learning

- Leverages a small set of demonstration data to massively accelerate the learning process

$$J(Q) = J_{DQ}(Q) + \lambda_1 J_n(Q) + \lambda_2 J_E(Q) + \lambda_3 J_{L2}(Q)$$

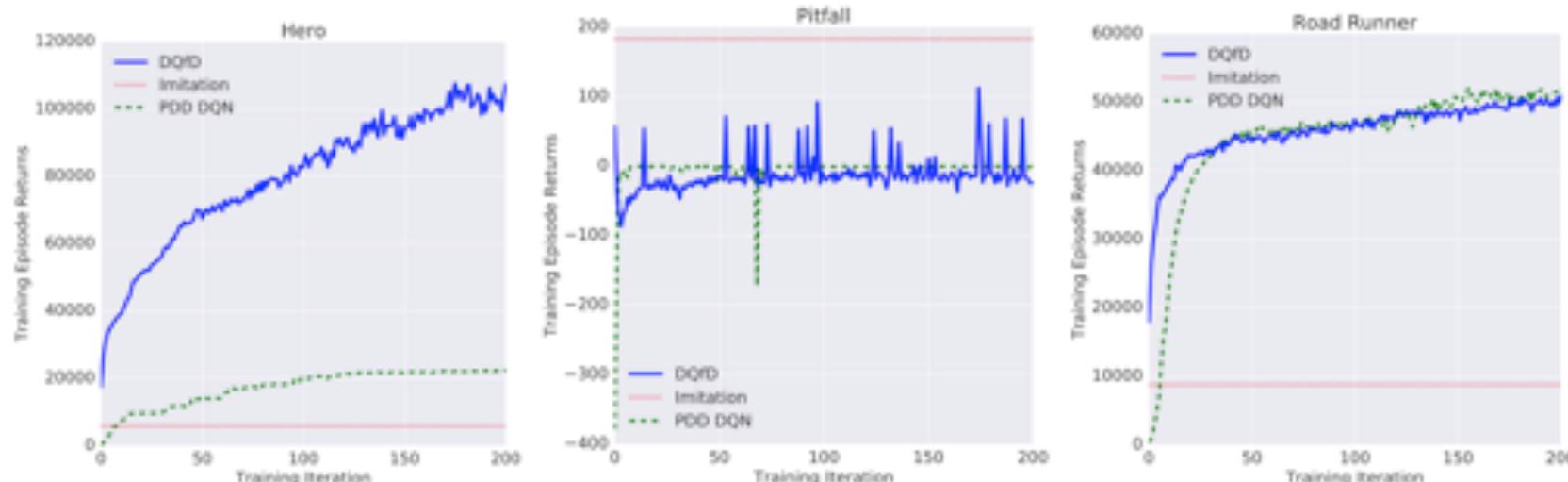
regularization loss

Q-learning loss

n-step Q-learning loss

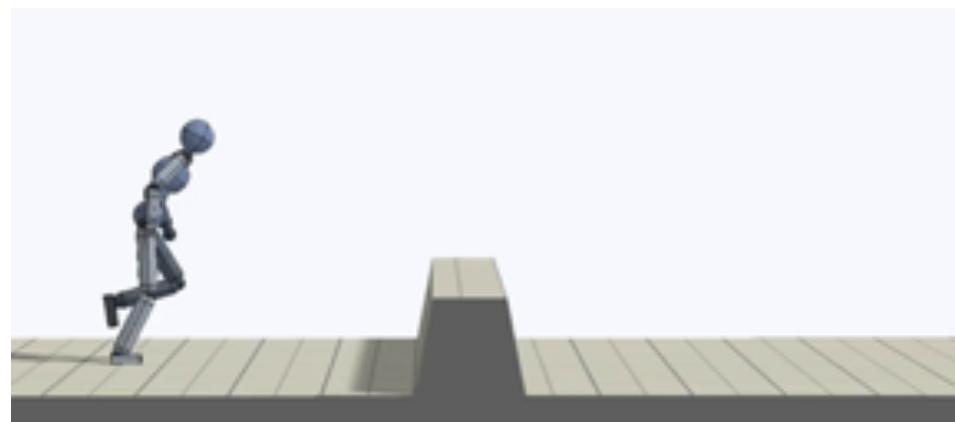
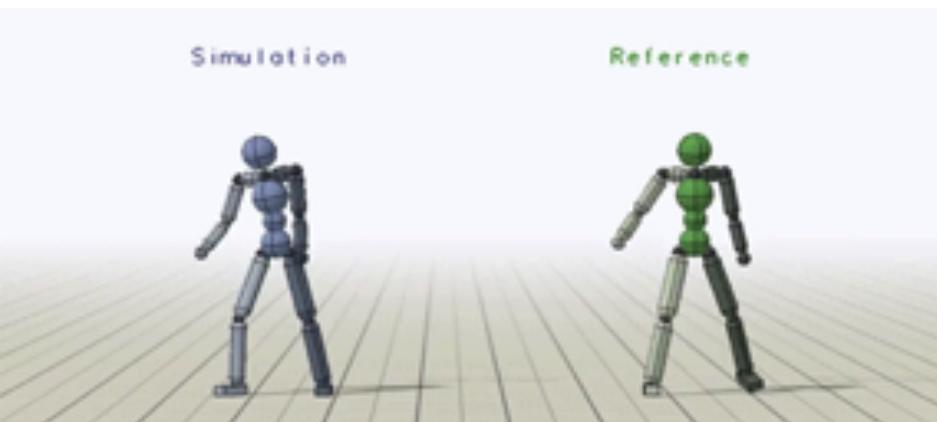
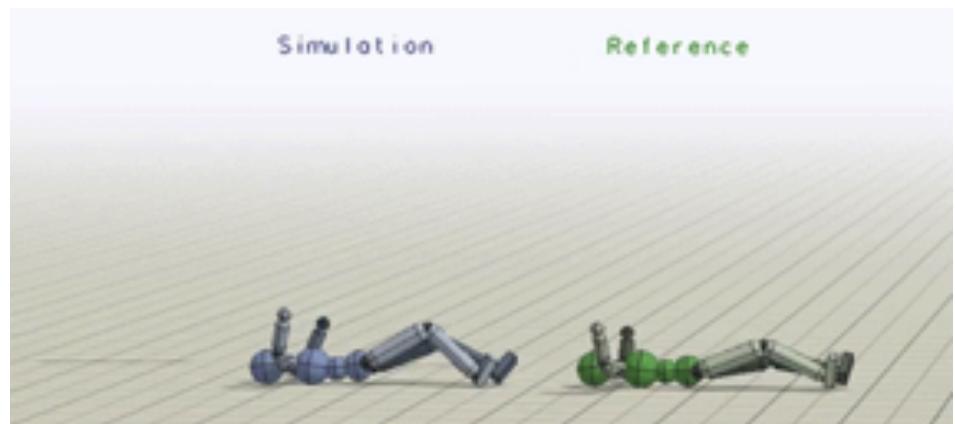
$$J_E(Q) = \max_{a \in A} [Q(s, a) + l(a_E, a)] - Q(s, a_E)$$

marginal loss on example action



Case Study: Motion Imitation

- To reproduce a given kinematic reference motion



Case Study: Motion Imitation

- Each reference motion is represented as a sequence of target poses $\hat{q}_0, \hat{q}_1, \dots, \hat{q}_T$,
- Reward function is defined as $r_t = \omega^I r_t^I + \omega^G r_t^G$.
- First term is imitation objective (imitate the reference motion), second term is task dependent
- Train with PPO

Case Study: Motion Imitation

- Imitation reward is carefully designed

$$r_t = \exp\left(-2\|\hat{q}_t - q_t\|^2\right).$$



$$r_t^I = w^p r_t^p + w^v r_t^v + w^e r_t^e + w^c r_t^c$$

$$w^p = 0.65, w^v = 0.1, w^e = 0.15, w^c = 0.1$$

$$r_t^p = \exp\left[-2\left(\sum_j \|\hat{q}_t^j \ominus q_t^j\|^2\right)\right]. \quad r_t^v = \exp\left[-0.1\left(\sum_j \|\hat{q}_t^j - \dot{q}_t^j\|^2\right)\right]. \quad r_t^e = \exp\left[-40\left(\sum_e \|\hat{p}_t^e - p_t^e\|^2\right)\right]. \quad r_t^c = \exp\left[-10\left(\|\hat{p}_t^c - p_t^c\|^2\right)\right].$$

Case Study: Motion Imitation



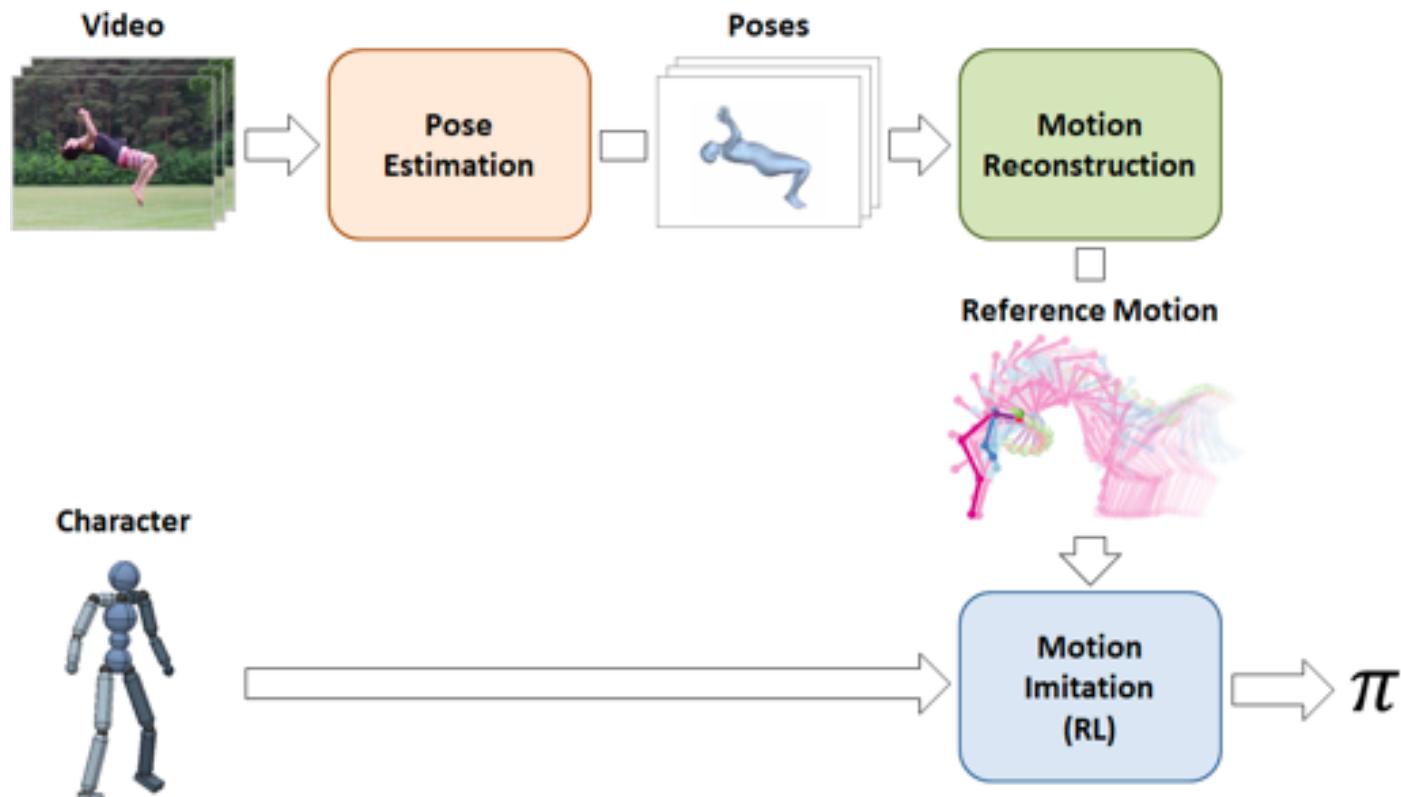
Case Study: Motion Imitation

- Follow-up work: how to get rid of MoCap data



Case Study: Motion Imitation

- Learning dynamics from videos



Case Study: Motion Imitation

Vision-based pose estimator



Case Study: Motion Imitation

- Learning dynamics from videos

<https://bair.berkeley.edu/blog/2018/10/09/sfv/>

Two Major Problems in Imitation Learning

- How to collect expert demonstrations
 - Crowdsourcing
 - Guided policy search or optimal control for trajectory optimization
- How to optimize the policy for off-course situations
 - Simulate those situations to collect new labels
 - Use off-policy learning with the already collected samples
 - Combine IL and RL

Summary on Imitation Learning

Pure imitation learning

- Easy and stable supervised learning
- Distributional shift
- No chance to get better than the demonstrations

Pure reinforcement learning

- Unbiased reinforcement learning, can get arbitrarily good
- Challenging exploration and optimization problem

Initialize & finetune

- Almost the best of both worlds
- ...but can forget demo initialization due to distributional shift

Pure reinforcement learning, with demos as off-policy data

- Unbiased reinforcement learning, can get arbitrarily good
- Demonstrations don't always help

Hybrid objective, imitation as an “auxiliary loss”

- Like initialization & finetuning, almost the best of both worlds
- No forgetting
- But no longer pure RL, may be biased, may require lots of tuning

Conclusion

- Imitation learning is an important research topic to bridge machine learning and human demonstration
 - Robot learning (learn from few samples)
 - Inverse reinforcement learning
 - Active learning
 - etc
- Recent survey: An Algorithmic Perspective on Imitation Learning: <https://arxiv.org/pdf/1811.06711.pdf>