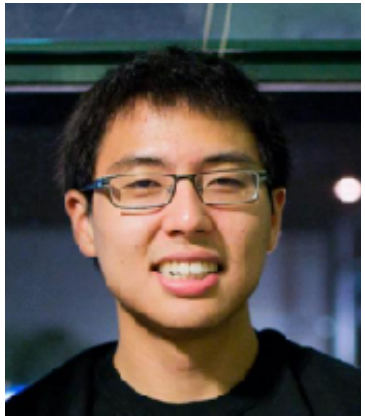
trying to make sense of control from pixels

Benjamin Recht
University of California, Berkeley

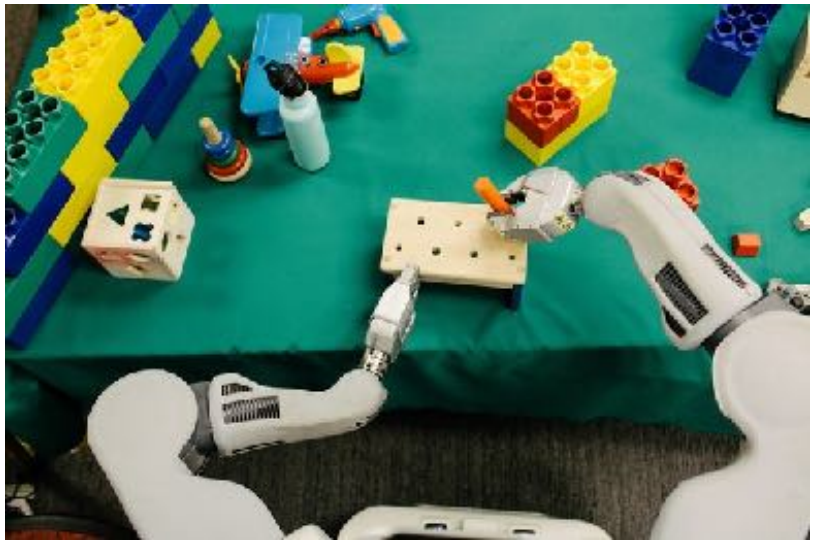
Collaborators

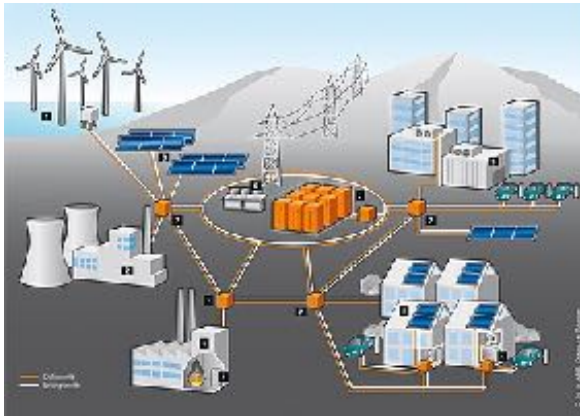




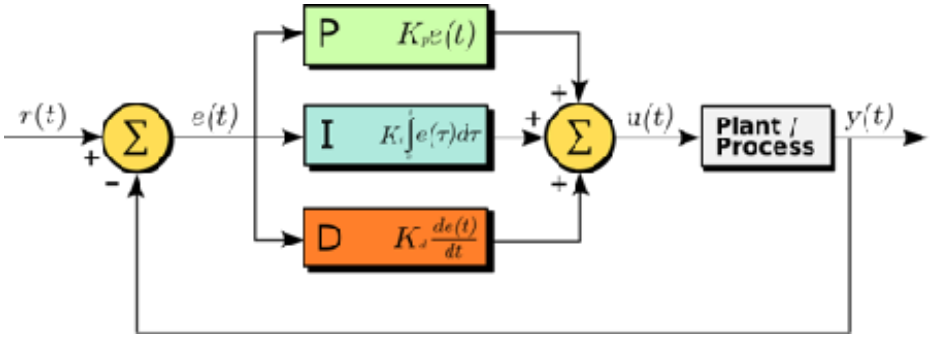

Joint work with Sarah Dean, Aurelia Guy, Horia Mania, Nikolai Matni, Rohan Sinha, and Stephen Tu, and Vickie Ye.

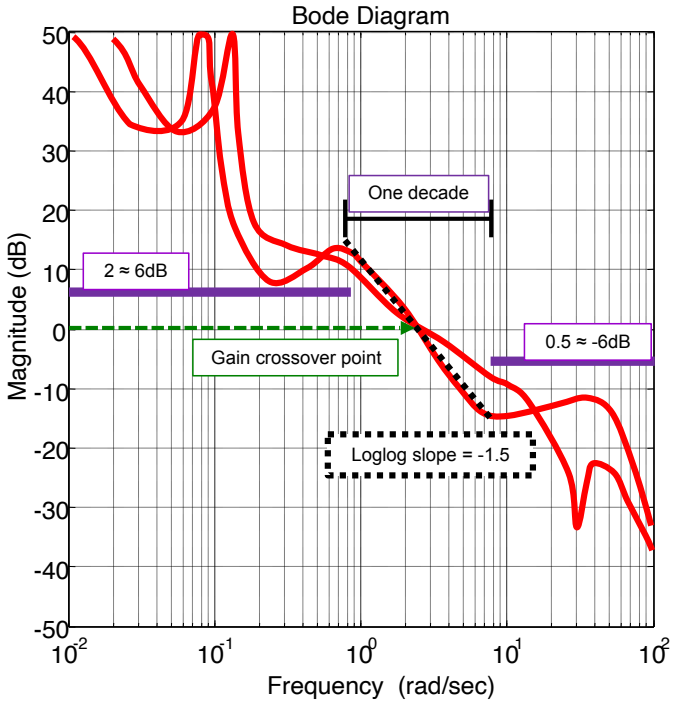




trustable, scalable, predictable

But PID control works…

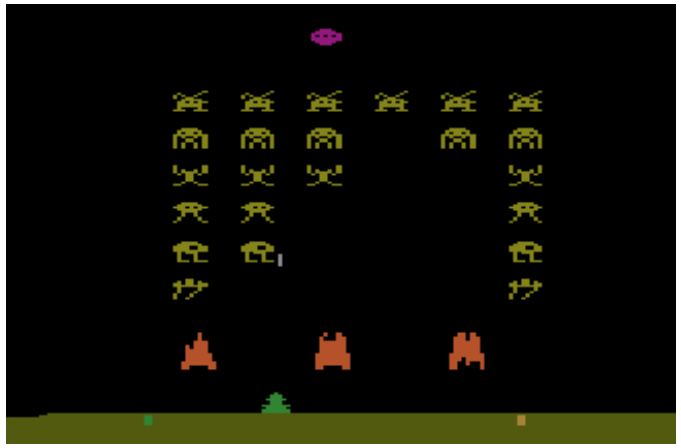


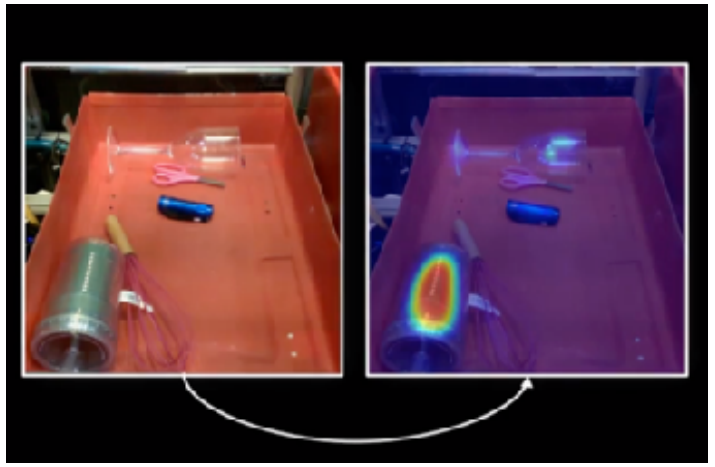


2 parameters suffice for 95% of all control applications.

How much needs to be modeled for more advanced control?

What role should **learning** play in advanced autonomous systems?

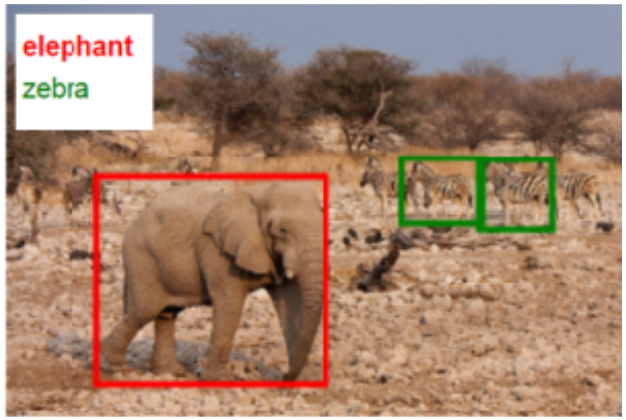


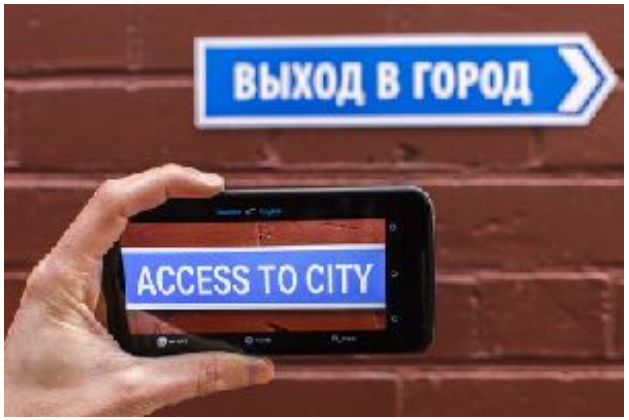


Policies map pixels to action

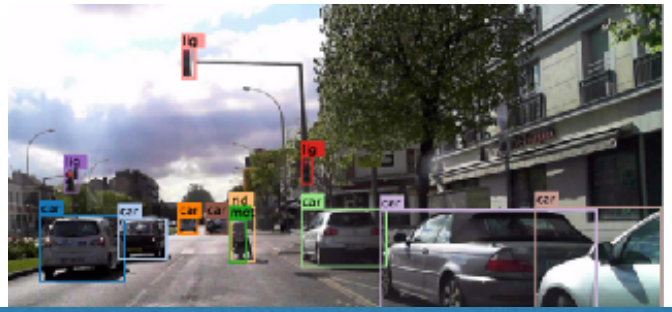
what is ML good for in control?

- Fundamentally, almost all machine learning successes are in *nonparametric prediction* (mostly classification).

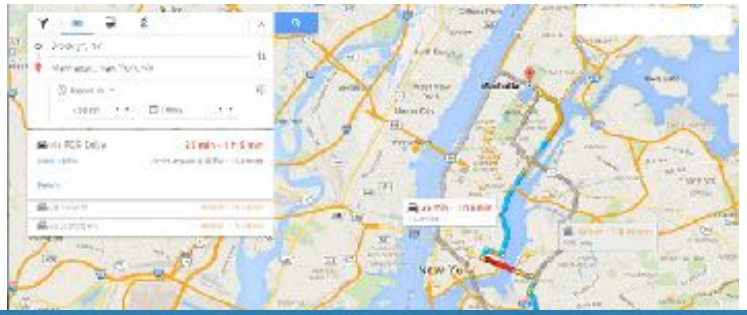




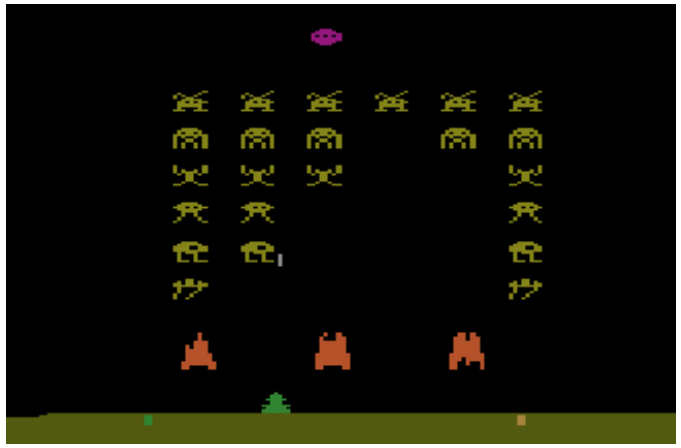
Perceptual sensors in the loop

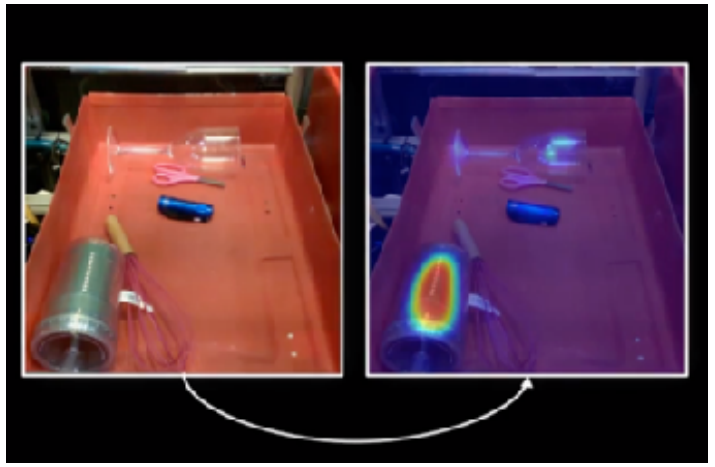


Forecasting in MPC



How to incorporate uncertain predictive perception in trustable, scalable, predictable autonomy?





Policies map pixels to action

Is there a reasonable approach to control with *perceptual sensors*?

- Output feedback vs state feedback
- Learning and distribution shift
- Mitigating distribution shift via control authority

Newton's Laws

$$q_{t+1} = q_t + v_t$$
$$v_{t+1} = v_t + a_t$$
$$m\, a_t = u_t$$

$$\text{minimize} \quad \mathbb{E}\left[\sum_{t=0}^{T} (x_t)_1^2 + r u_t^2\right]$$

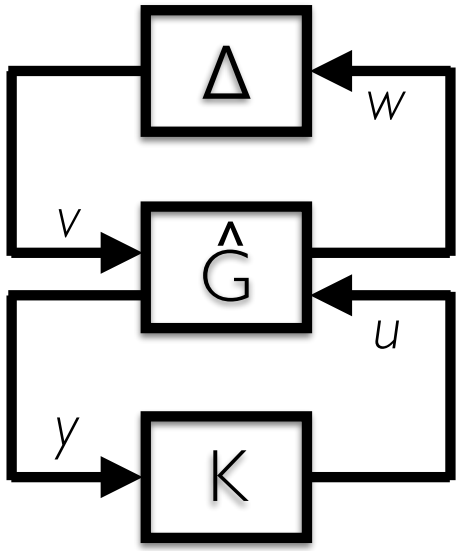
$$\text{subject to} \quad x_{t+1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x_t + \begin{bmatrix} 0 \\ 1/m \end{bmatrix} u_t + w_t$$

$$x_t = \begin{bmatrix} q_t \\ v_t \end{bmatrix}$$

Do we really need sophisticated learning in MDPs?

- **NO.** System ID in MDPs is relatively easy. We usually have good models. State-feedback well studied and fairly robust.

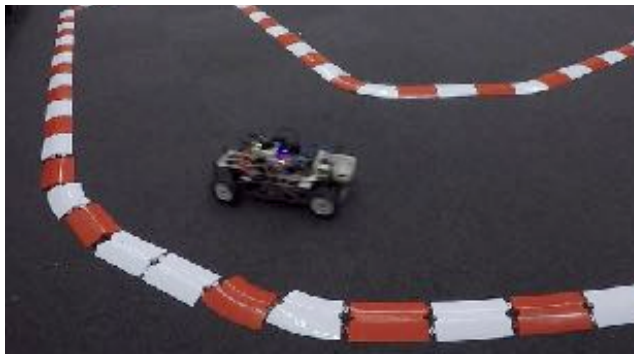
Adaptive Control



Certainty equivalent control achieves optimal regret. Robust control stabilizes low-data regimes.

Dean et al. FOCM 2019
Dean et al. Neurips 2018
Mania, Tu, R et al. Neurips 2019

Safe Exploration



Guaranteed safe execution and improved model performance within specified safe constraint sets.

Dean et al. ACC 2019

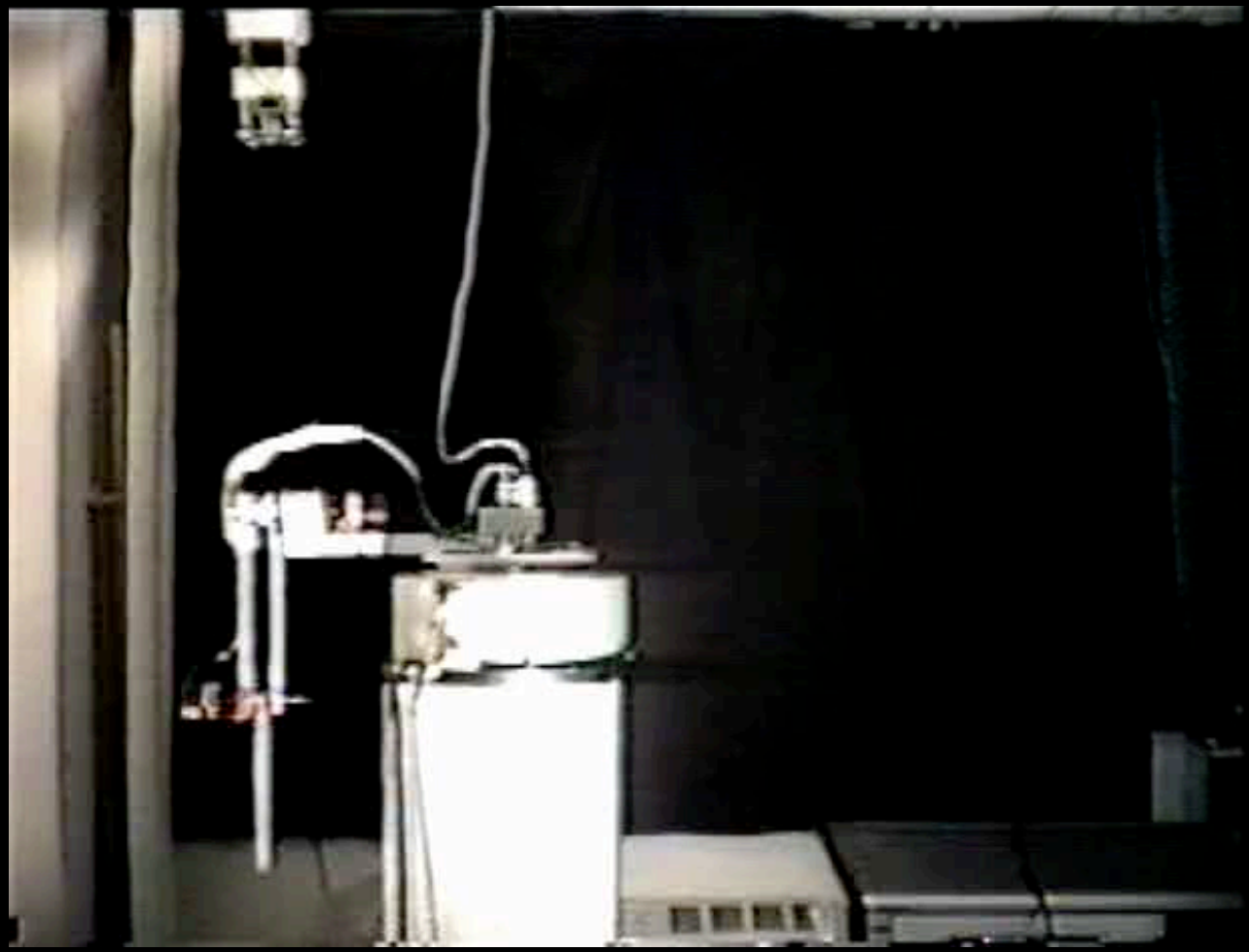
Simplifying RL

Simple, static, linear policies outperform deep RL and can be found with simple, standard optimization algorithms.

Mania, Guy, R. Neurips 2018

$$u(x_1, x_2) = 2a \sin x_1 + b x_2 F(x_1, x_2) \cos x_1$$

$$F(x_1, x_2) = \frac{2a+1}{4a}(2a \cos x_1 - 1) + \frac{x_2^2}{2}.$$



K. J. Åström, J. Aracil, F. Gordillo. "A family of smooth controllers for swinging up a pendulum." *Automatica*, 44:7, pp. 1841–1848, 2008.

Newton's Laws

$$q_{t+1} = q_t + v_t$$
$$v_{t+1} = v_t + a_t$$
$$m\, a_t = u_t$$

$$\text{minimize} \quad \mathbb{E}\left[\sum_{t=0}^{T} (x_t)_1^2 + r u_t^2\right]$$

$$\text{subject to} \quad x_{t+1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x_t + \begin{bmatrix} 0 \\ 1/m \end{bmatrix} u_t + w_t$$

$$x_t = \begin{bmatrix} q_t \\ v_t \end{bmatrix}$$

$z_t = h(x_t)$

f

$y_{t+1} = \begin{bmatrix} 1 & 0 \end{bmatrix} x_t + e_t$

perception map

LQG Regulators

$$\text{minimize} \quad \mathbb{E}\left[\sum_{t=0}^{T}(x_t)_1^2 + (x_t)_2^2 + u_t^2\right]$$

$$\text{subject to} \quad x_{t+1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x_t + \begin{bmatrix} 0 \\ 1/m \end{bmatrix} u_t + w_t$$

$$y_{t+1} = \begin{bmatrix} 1 & 0 \end{bmatrix} x_t + e_t$$

Assume e_t and w_t jointly Gaussian and uncorrelated

Optimal Solution:

1. Run optimal filter to estimate x_t from $\{y_1, \ldots, y_t\}$
2. Treat estimate as true and apply state feedback

This is the most common practice in control from outputs

LQG Regulators Can Be Fragile!

$$\text{minimize} \quad \mathbb{E}\left[\sum_{t=0}^{T}(x_t)_1^2 + (x_t)_2^2 + u_t^2\right]$$

$$\text{subject to} \quad x_{t+1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x_t + \begin{bmatrix} 0 \\ 1/m \end{bmatrix} u_t + w_t$$

$$y_{t+1} = \begin{bmatrix} 1 & 0 \end{bmatrix} x_t + e_t$$

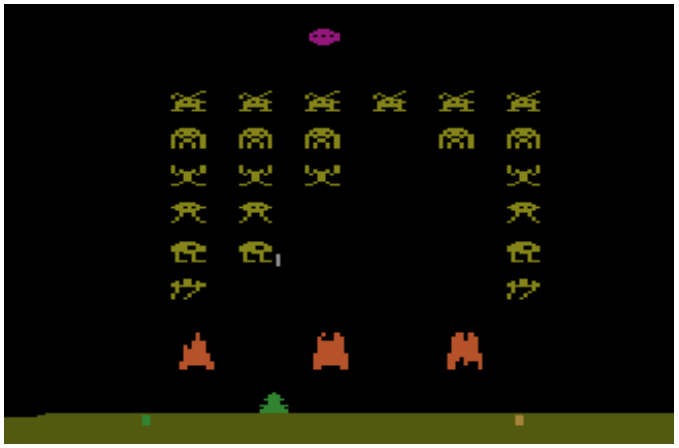
$$\Sigma_w = \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix}$$

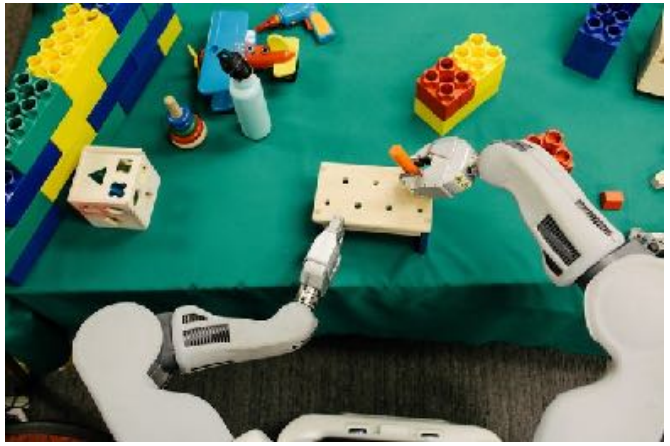
$$\sigma_e^2 = 10^{-10}$$

$$m = 1$$

If you misestimate *m* by more than 2e-5, closed loop with optimal LQG will be unstable!

Similar sensitivity to unmodeled sensor noise.





Q2: Is pixel-driven control best modeled by MDPs?

- NO. High dimensional state is problematic. Usually have some reasonable low-dimensional state in mind and want to map from pixel to state.

If you have imperfect state information, you have a POMDP problem, not an MDP problem

Train vision system

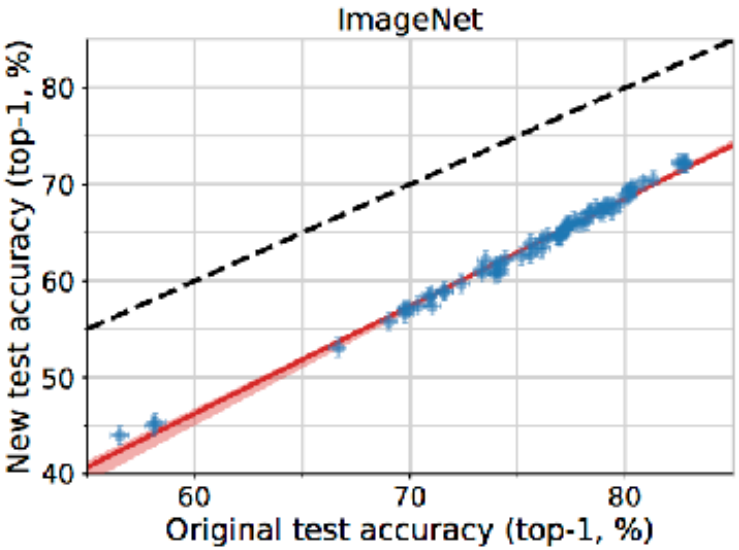
Deploy without vicon

Q3: Can we rely on standard ML for error quantification?

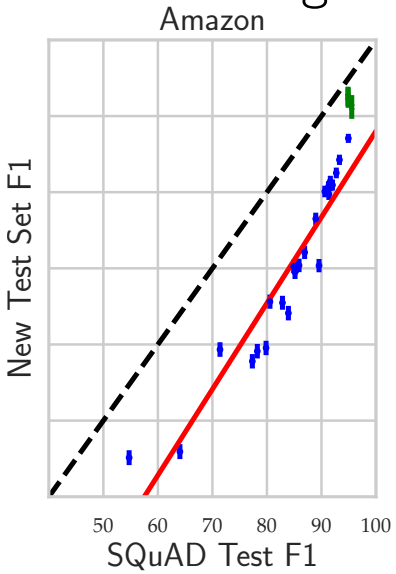
- NO. As soon as you change your policy, you change the distribution, and generalization goes out the window.

$$L_S - L_{S'} = \underbrace{(L_S - L_{\mathcal{D}})}_{\text{Adaptivity gap}} + \underbrace{(L_{\mathcal{D}} - L_{\mathcal{D}'})}_{\text{Distribution gap}} + \underbrace{(L_{\mathcal{D}'} - L_{S'})}_{\text{Generalization gap}}$$

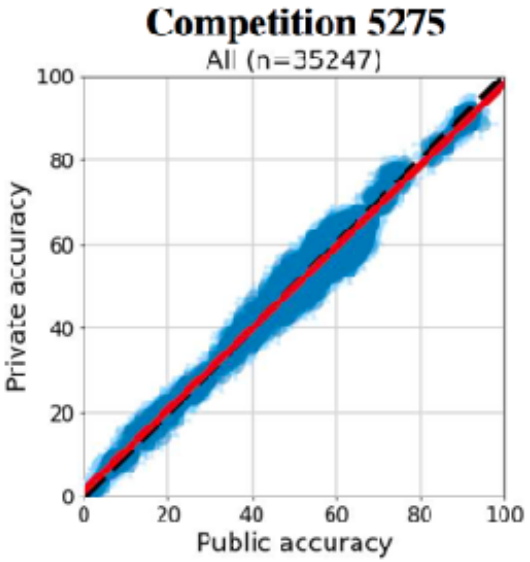
Image Classification



Question Answering



Kaggle



We *don't* see adaptive overfitting.

We *do* see significant fragility from distribution shift.

R. Et al, ICML 2019, Roelofs et. al, Neurips 2019, Shankar et al. 2019, Miller et al. 2020

Q1: Do we really need sophisticated learning in MDPs?

- **NO.** System ID in MDPs is relatively easy. We usually have good models. State-feedback well studied and fairly robust.

Q2: Is pixel-driven control best modeled by MDPs?

- **NO.** High dimensional state is problematic. Usually have some reasonable low-dimensional state in mind and want to map from pixel to state.

Q3: Can we rely on standard ML for error quantification?

- **NO.** As soon as you change your policy, you change the distribution, and generalization goes out the window.

Q1: Do we really need sophisticated learning in MDPs?

- NO. System ID in MDPs is relatively easy. We usually have good models. State-feedback well studied and fairly robust.

Today: using ideas from robust control and as little machine learning as possible design policies from pixels to action that are safe and performant.

Q3: Can we rely on standard ML for error quantification?

- NO. As soon as you change your policy, you change the distribution, and generalization goes out the window.

Camera

IMU

Encoders

Goal: drive as fast as possible around a loop provided by a single demonstration

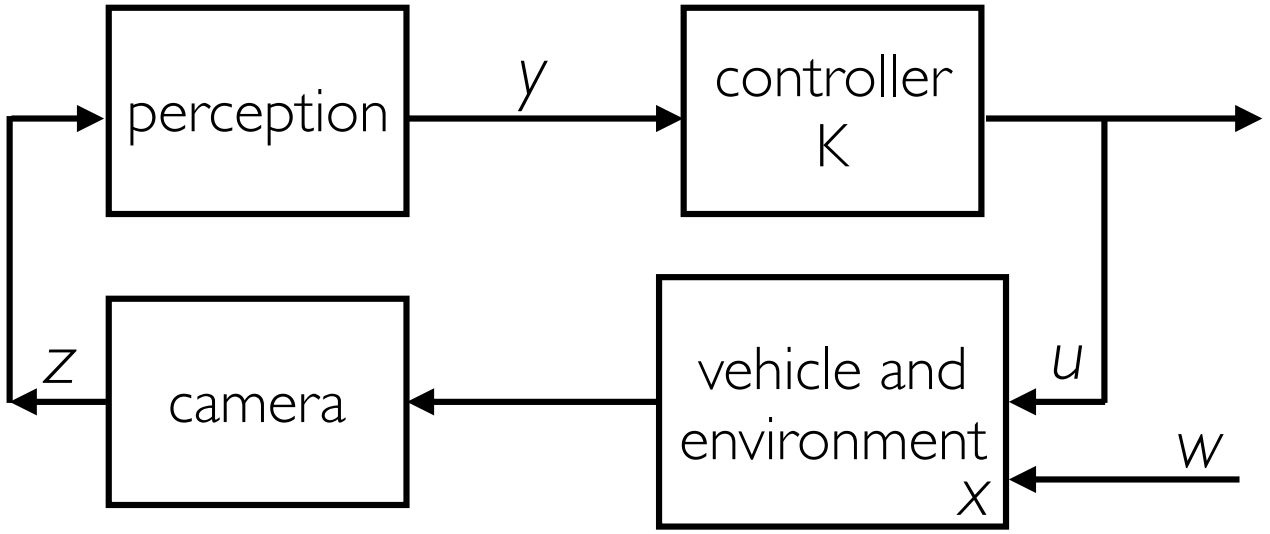
Challenging as there is no depth information and all of the coordinates are relative.

Related work

- Our inspiration here comes primarily from three groups working on autonomous racing:
  - Borrelli's MPC Lab at UC Berkeley
  - AutoRally at Georgia Tech (notably talking with Byron and Evangelos)
  - Scaramuzza's Robotics and Perception Lab at ETH

How to model these abstractions?

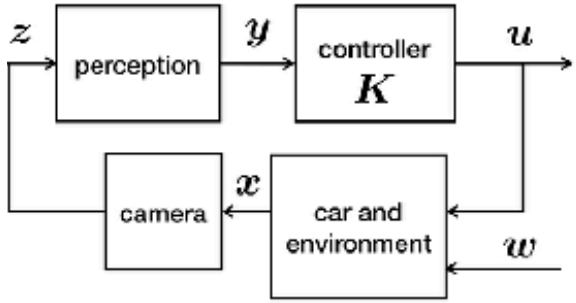
- ~~Unknown locally~~ linear dynamics $x_{k+1} = A_k x_k + B_k u_k + w_k$
- Appearance model $z_k = h(x_k)$
- Perception as virtual sensor $y_k = p(z_k) = C x_k + e_k$



Today:

- Machine learning black boxes as virtual sensors.
- How to use these robustly in feedback loops.
- What does this tell us about learning-control codesign?

Problem Setting: Output Feedback Control



- Perception as virtual sensor leads to familiar control setting

$$\min_{\mathbf{K}} \ \mathrm{cost}(\mathbf{x}, \mathbf{u})$$

$$\text{s.t. } x_{k+1} = Ax_k + Bu_k + Hw_k,$$

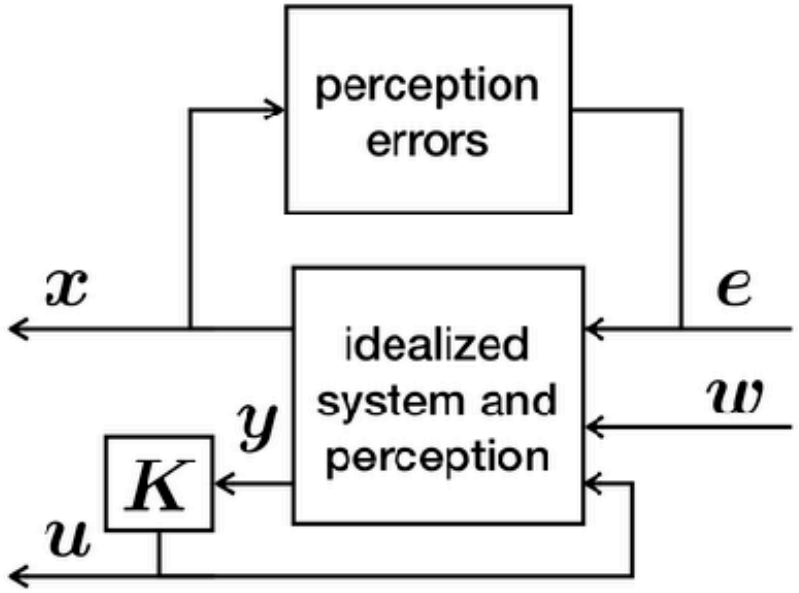
$$y_k = Cx_k + e_k,$$

$$u_k = \mathbf{K}(y_{0:k})$$

Well-studied solutions for various combinations of cost and error/noise models

Perception Errors and Output Feedback

$$e_k = p(h(x_k)) - Cx_k$$

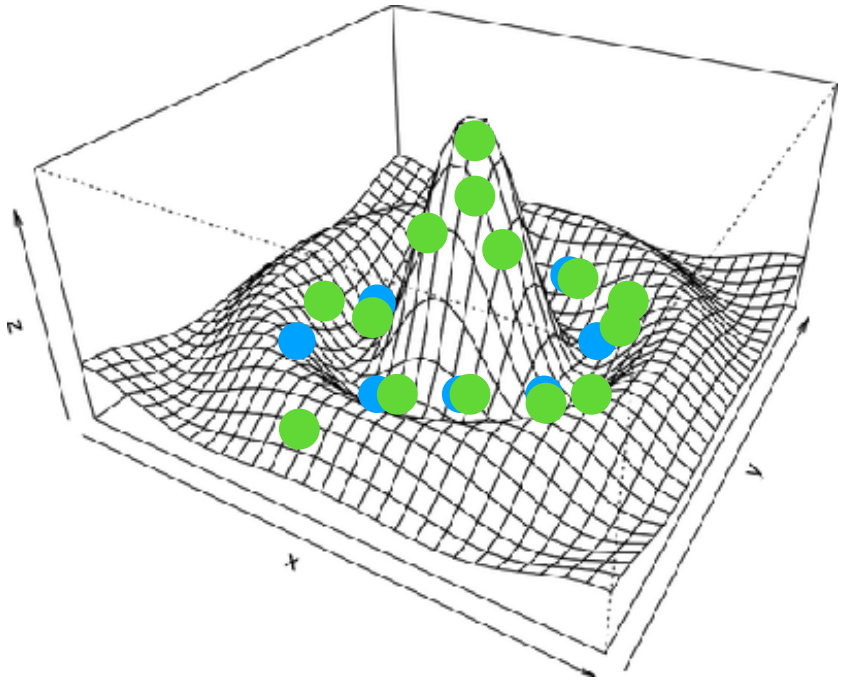


Suppose that $p \circ h$ - C is locally slope bounded within a radius of r around training datapoints. Define the safe set

$$X_\gamma = \bigcup_{(x_0,z_0)\in\mathcal{S}_0} \{x \in B_r(x_d) \ : \quad \|p \circ h(x_0) - Cx_0\| + S\|x - x_0\| \leq \gamma\}$$

Then for any x in X_γ, the perception error, e, has norm at most γ.

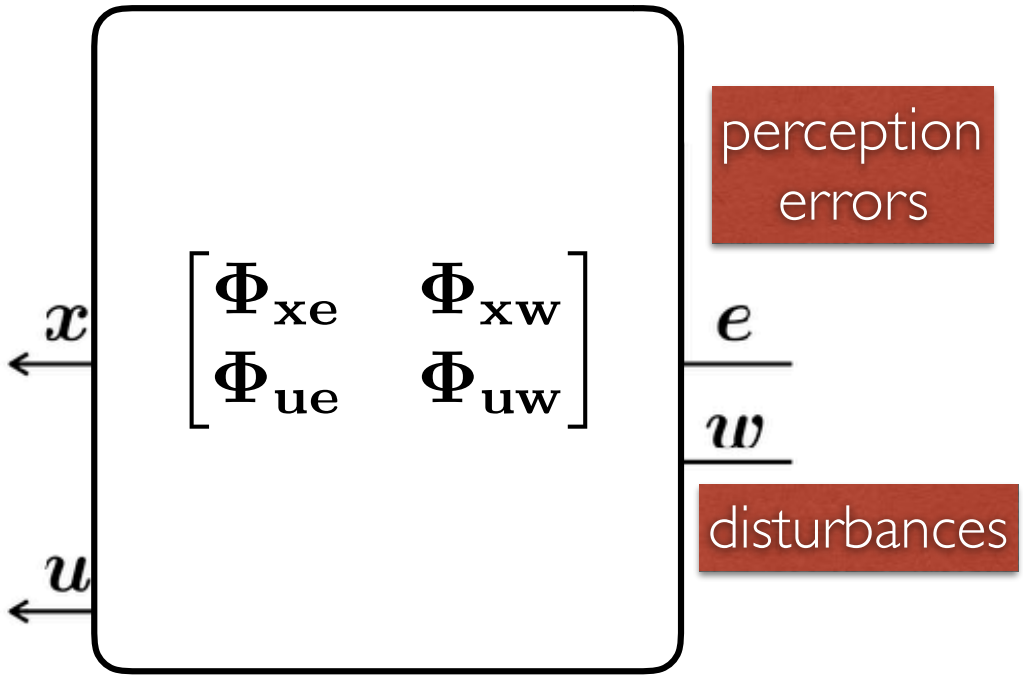
Generalization



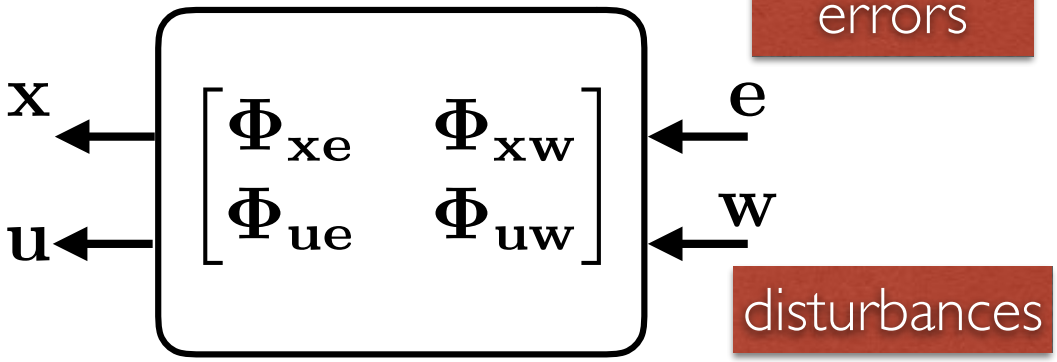
- Classic generalization results rely on statistical arguments about closeness of training and testing data
  - Usually assume same distribution
- The closed-loop distribution of states depends on the perception errors
- Idea: leverage control authority to ensure closeness

A System Level view of perception errors

$$e_k = p(h(x_k)) - Cx_k$$



Robust Generalization



x_d - training trajectories

$\hat{x}$ - nominal trajectories under design

Theorem 1: There exist B and ρ such that if the training error is bounded and the synthesized feedback system satisfies

$$\|\widehat{\mathbf{\Phi}}_{\mathbf{xe}}\| \leq B - \rho\|\hat{x} - x_d\|$$

we have that trajectories stay in the safe set X_γ and perception errors remain bounded.

Theorem 2: Without a more refined error model, staying near the training data is *necessary*.

Example: static filter and state feedback

Consider a controller of the form of static state feedback on the output of a Kalman Filter:

$$u_t = K\hat{x}_t$$
$$\hat{x}_{t+1} = A\hat{x}_t + Bu_t + L(y_t - C\hat{x}_t)$$

The sensitivity to measurements is given by frequency response elements

$$\mathbf{\Phi_{xe}}(t) = \begin{bmatrix} I & 0 \end{bmatrix} \begin{bmatrix} A - BK & BK \\ 0 & A - LC \end{bmatrix}^t \begin{bmatrix} 0 \\ L \end{bmatrix}$$

For our fragile LQG Example…

$$\text{minimize} \quad \mathbb{E}\left[\sum_{t=0}^{T}(x_t)_1^2 + (x_t)_2^2 + u_t^2\right]$$

$$\text{subject to} \quad x_{t+1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x_t + \begin{bmatrix} 0 \\ 1/m \end{bmatrix} u_t + w_t$$

$$y_{t+1} = \begin{bmatrix} 1 & 0 \end{bmatrix} x_t + v_t$$

$$\Sigma_w = \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix}$$

$$\sigma_v^2 = 10^{-10}$$

$$m = 1$$

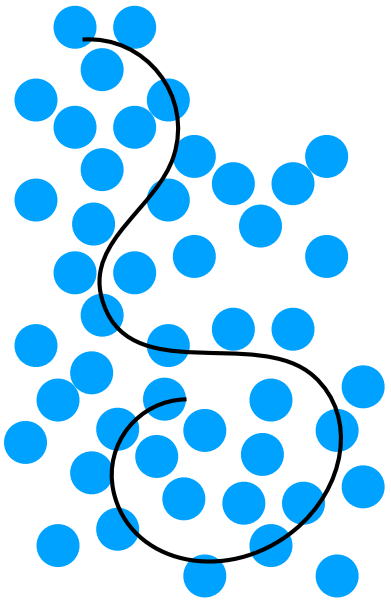
$$\|\widehat{\mathbf{\Phi}}_{\mathbf{xe}}\| = 2 \times 10^5$$

Training Strategies

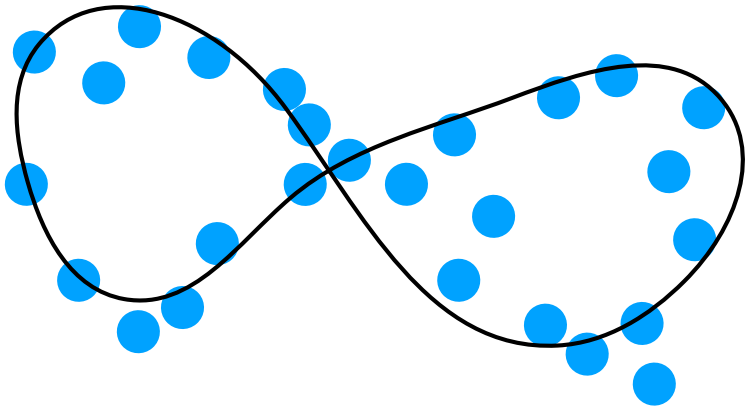
The training data affects the performance through $\|\widehat{\mathbf{x}} - \mathbf{x}_d\|$

We bound this quantity in two settings:

Dense Sampling



Imitation Learning

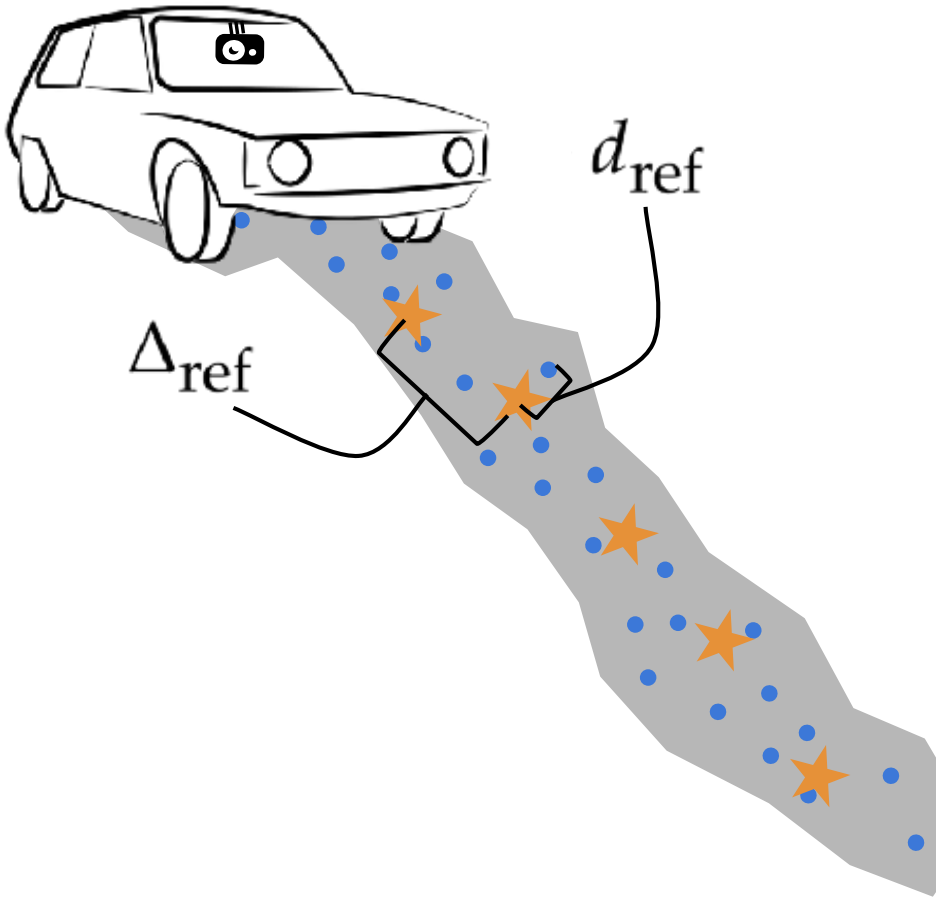


Training data should resemble desired closed-loop behavior!

Robust reference tracking

Model waypoints as disturbances

- Distance between waypoints: Δ_{ref}
- Distance between reference and training data. d_{ref}



Bounded perception errors provided:

$$S\|\mathbf{\Phi}_{xe}\| + \Delta_{\text{ref}}\|\mathbf{\Phi}_{xw}\| + d_{\text{ref}} \leq 1$$

difficulty of task

quality of perception and training data

Simulation Experiments

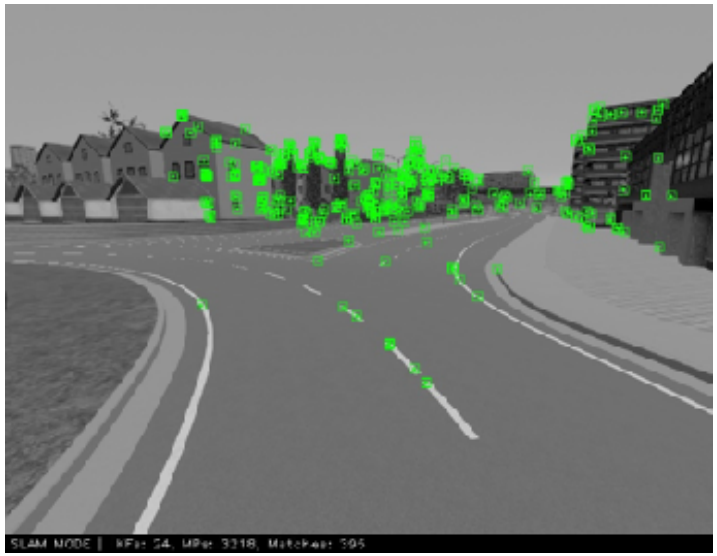
CARLA vehicle simulation platform

Waypoint tracking objective with
2D double integrator dynamics

Perception Maps

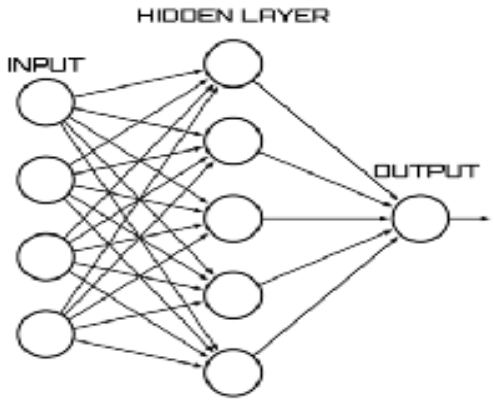
Visual odometry

- "trained" with SLAM (200 frames)



Convolutional Neural Network

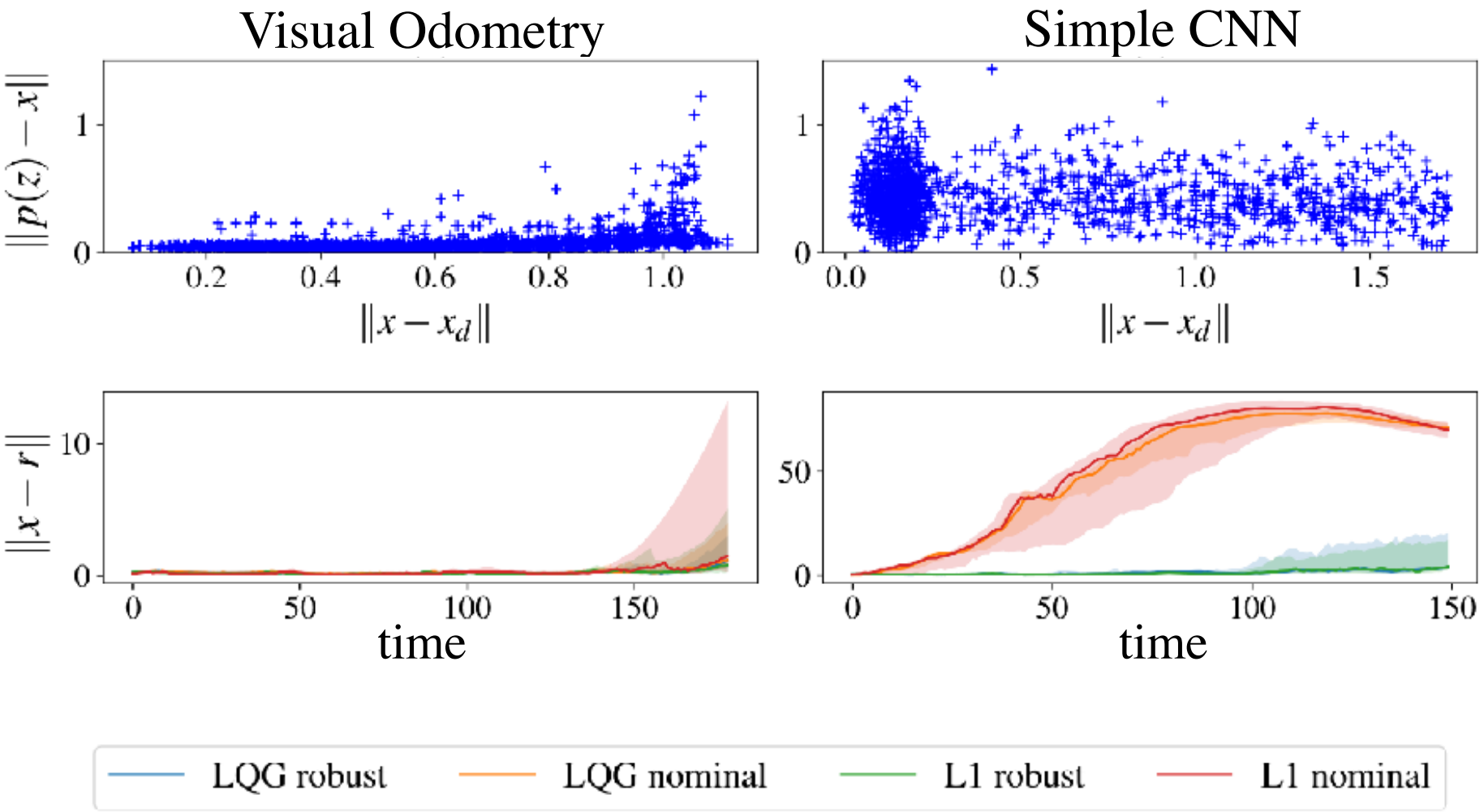
- single convolutional, ReLU activation, and max pooling layer
- trained with 30,000 frames



Simulation Experiments: Controller Synthesis

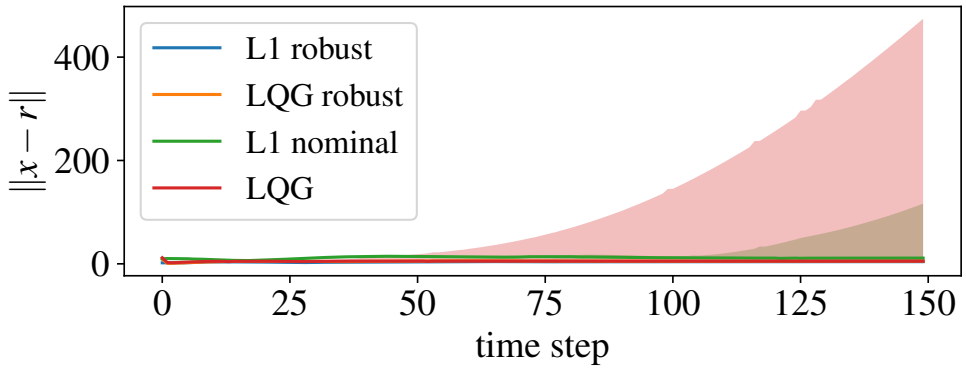
We formulate the waypoint tracking problem as an output-feedback control problem, and synthesize:

1. Nominal control disregarding measurement matrix errors
2. Robust control by constraining the norm $\|\widehat{\mathbf{\Phi}}_{\mathbf{xe}}\|$

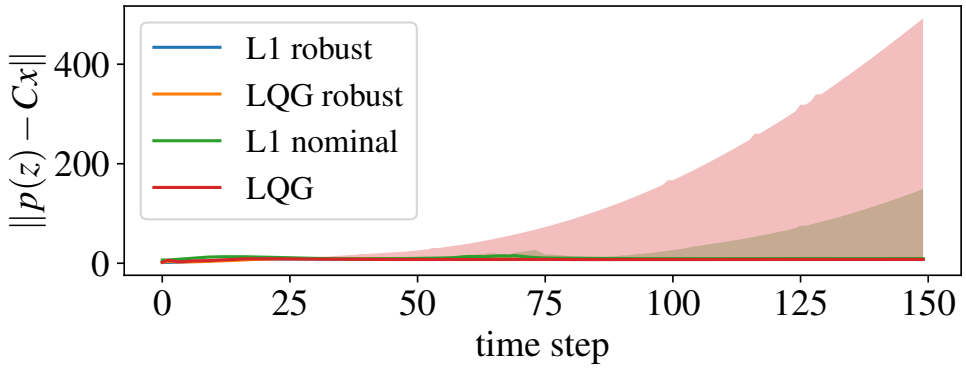

Visual Odometry
Simple CNN
$\|p(z) - x\|$
$\|x - x_d\|$
$\|x - r\|$
time
LQG robust
LQG nominal
L1 robust
L1 nominal

Experimental Results

Without robustness condition, close-loop system diverges

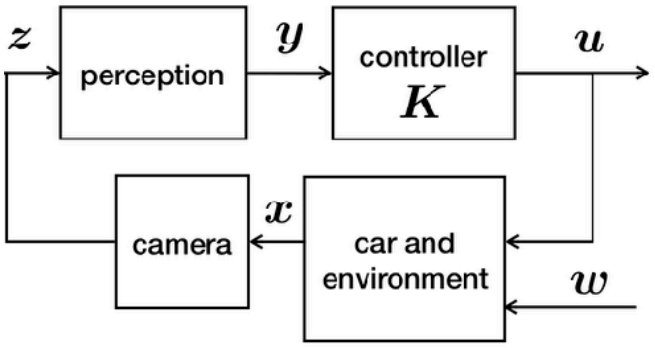


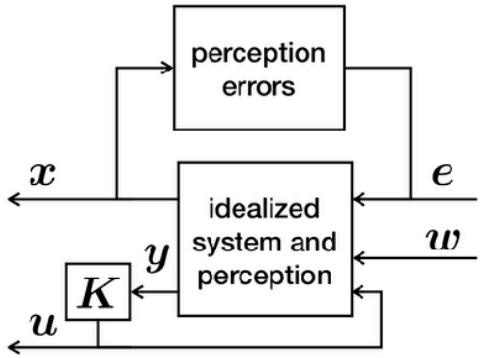
Due to increasing errors in perception



Towards understanding control from pixels

1. Perception map as virtual sensor
2. Leverage control to sidestep distribution shift
3. Separate learning components for end-to-end analysis

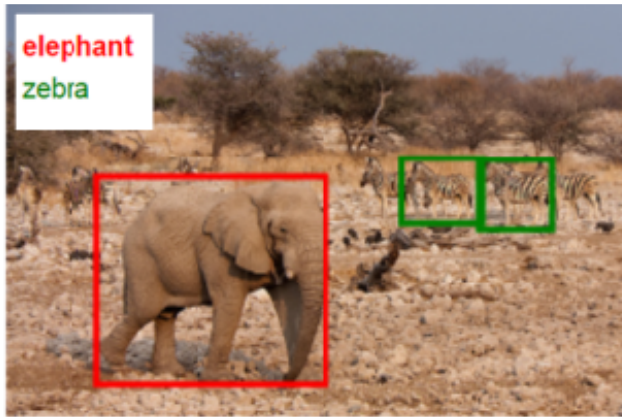


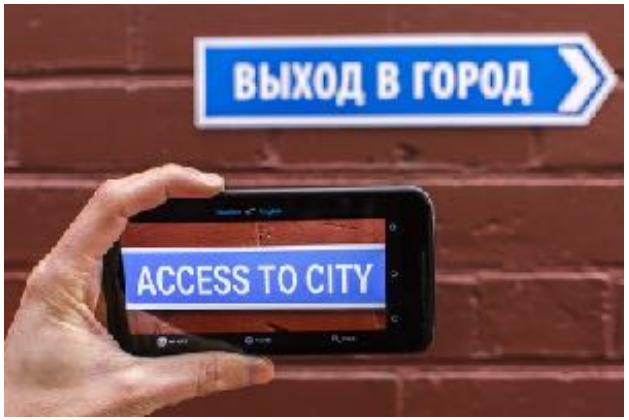


Robust Guarantees for Perception-Based Control.
S. Dean, N. Matni, B. Recht, V. Ye. arXiv:1907.03680

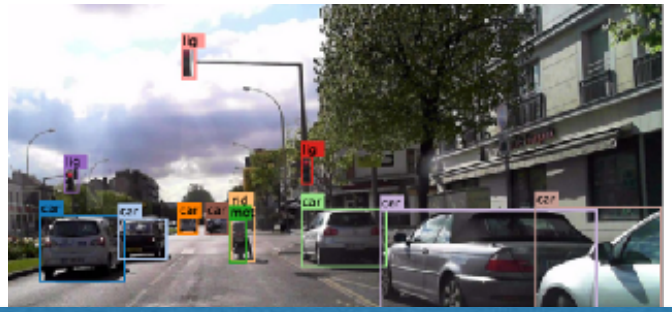
what is ML good for in control?

- Fundamentally, almost all machine learning successes are in *nonparametric prediction* (mostly classification).

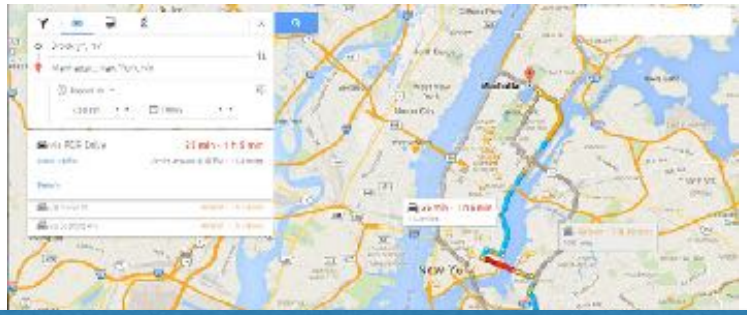




Perceptual sensors in the loop



Forecasting in MPC



How to incorporate uncertain predictive perception in trustable, scalable, predictable autonomy?

References

- "On the Sample Complexity of the Linear Quadratic Regulator." S. Dean, H. Mania, N. Matni, B. Recht, and S. Tu. *Journal of Foundations of Computational Mathematics,* 2019.
- "Least-squares Temporal Differencing for the Linear Quadratic Regulator" S. Tu and B. Recht. In ICML 2018.
- "Simple random search provides a competitive approach to reinforcement learning." H. Mania, A. Guy, and B. Recht. In NeurIPS 2018.
- "Regret Bounds for Robust Adaptive Control of the Linear Quadratic Regulator." S. Dean, H. Mania, N. Matni, B. Recht, and S. Tu. In NeurIPS 2018.
- "The Gap Between Model-Based and Model-Free Methods on the Linear Quadratic Regulator: An Asymptotic Viewpoint." S. Tu and B. Recht. In COLT 2019.
- "Safely Learning to Control the Constrained Linear Quadratic Regulator." S. Dean, S. Tu, N. Matni, and B. Recht. In ACC 2019.
- "Certainty Equivalent Control of LQR is Efficient." H. Mania, S. Tu, and B. Recht. In NeurIPS 2019.
- A Tour of Reinforcement Learning: The View from Continuous Control." B. Recht. *Annual Review of Control, Robotics, and Autonomous Systems.* 2(1): 253–279, 2019.
- "Do ImageNet Classifiers Generalize to ImageNet?" B. Recht, R. Roelofs, L. Schmidt, and V. Shankar. ICML 2019.
- "A systematic framework for natural perturbations from videos."V. Shankar, A. Dave, R. Roelofs, D. Ramanan, B. Recht, and L. Schmidt. `arXiv:1906.02168`
- "A Meta Analysis of Overfitting in Machine Learning." R. Roelofs, S. Fridovich-Keil, J. Miller, M. Hardt, B. Recht, and L. Schmidt. NeurIPS 2019.
- "The Effect of Natural Distribution Shift on Question Answering Models." J. Miller, K. Krauth, L. Schmidt, and B. Recht. *In preparation.*
- "Robust Guarantees for Perception-Based Control." S. Dean, N. Matni, B. Recht, V. Ye. `arXiv:1907.03680`