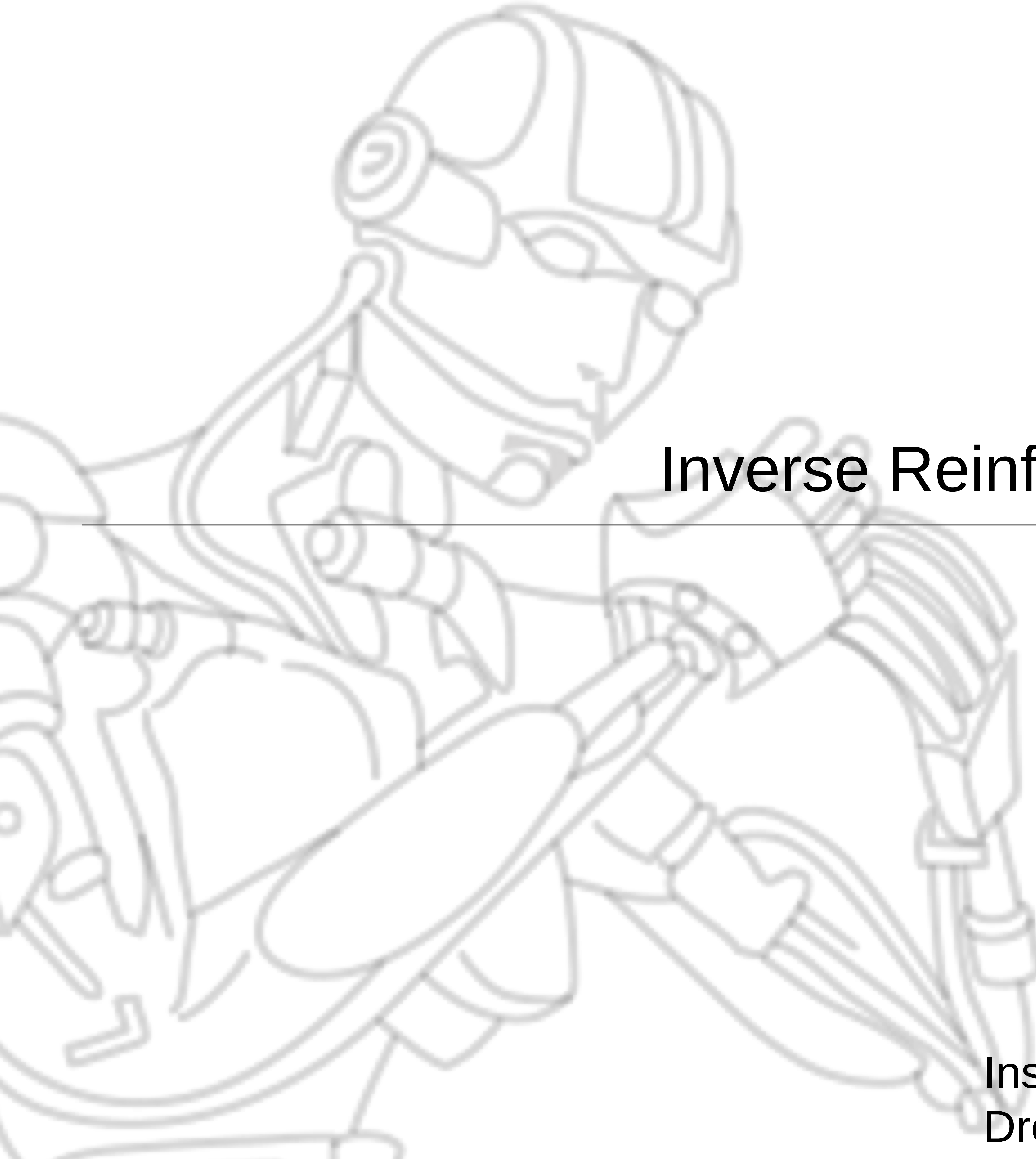


Imitation Learning
by
Inverse Reinforcement Learning

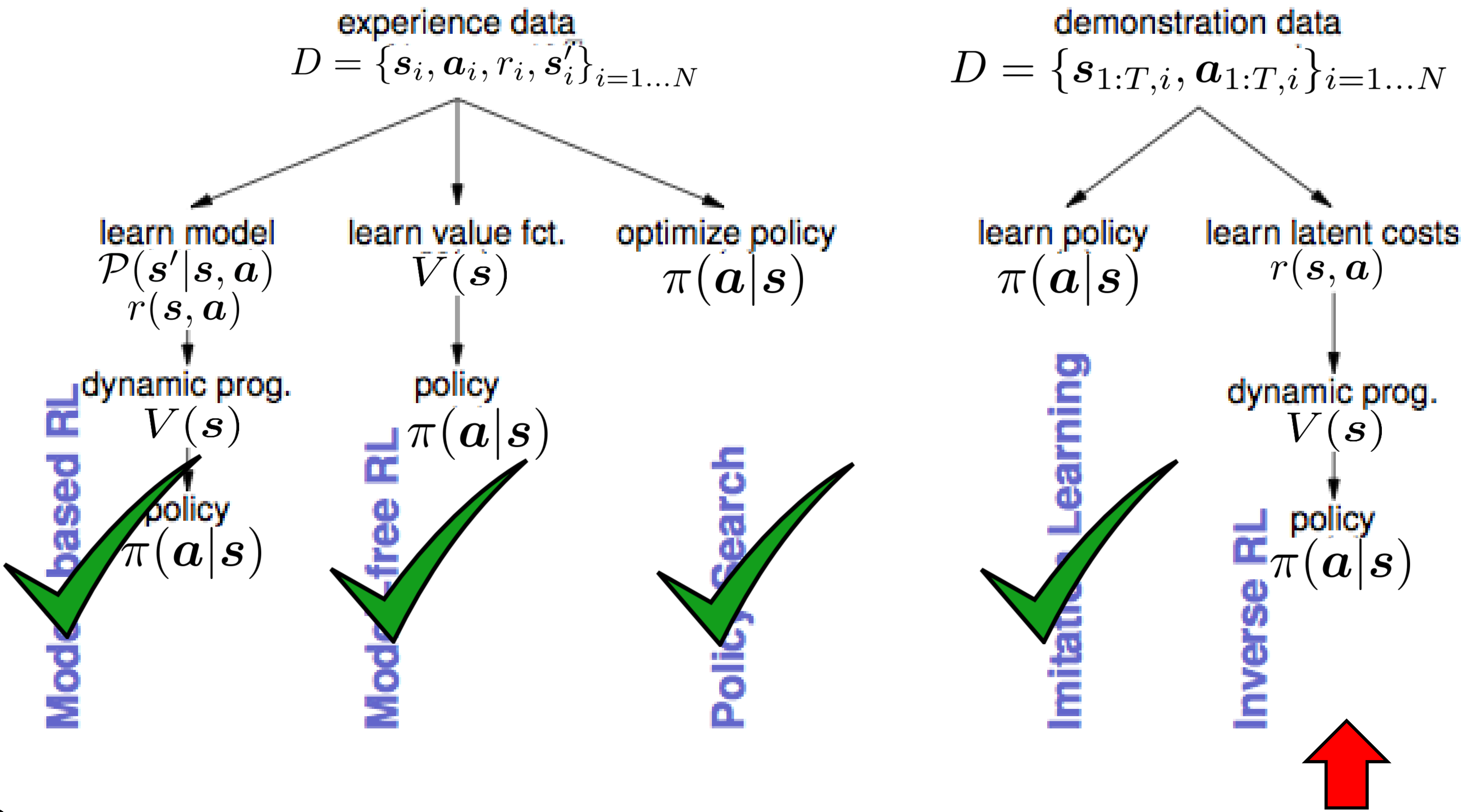
Jan Peters
Gerhard Neumann

Inspired by Slides from P. Abbeel,
Drew Bagnell and others

Purpose of this Lecture

- Learn an alternative approach to imitation learning
- What is the best way to imitate a teacher?
  - Learn its policy? Behavorial cloning
    - Needs a lot of demonstrations to generalize the behavior
  - Learn its intention / goals? Inverse Reinforcement Learning
    - Inverse Optimal Control, Inverse Optimal Planning
    - Determine the cost function of the teacher in order to obtain optimal behavior.
    - More concise description of behavior

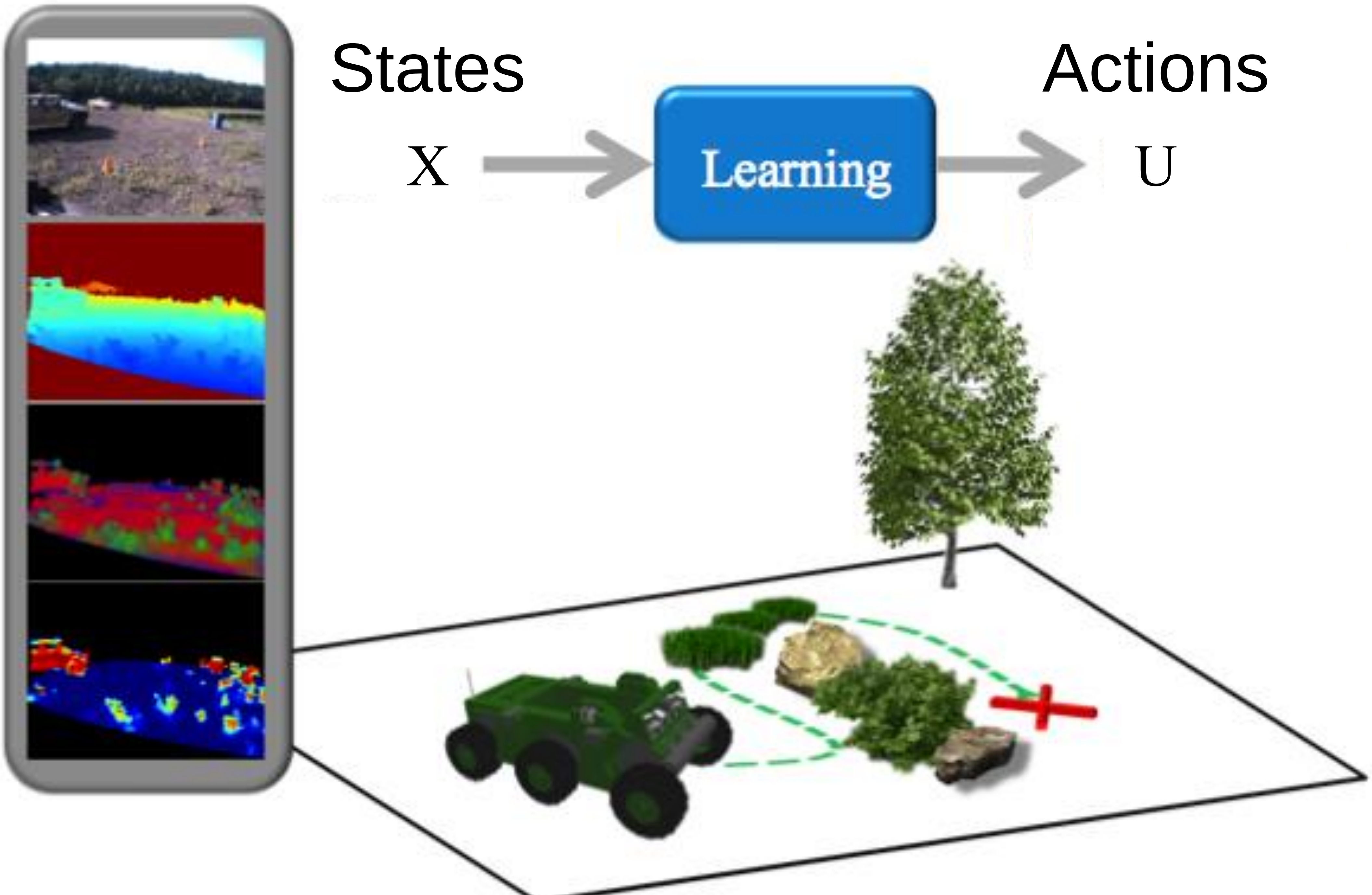
Bigger Picture



Outline of the Lecture

1. Introduction
    I. Comparison to Behavioral Cloning
2. Categories of IRL
    I. Maximum Margin
    II. Feature Matching by Max. Entropy
    III. Policy parametrized by rewards
3. Applications
4. Conclusion

Behavioral Cloning



What may be wrong here? Remember ALVINN?

There are difficulties involved with training "on-the-fly" with real images. If the network is not presented with sufficient variability in its training exemplars to cover the conditions it is likely to encounter when it takes over driving from the human operator, it will not develop a sufficiently robust representation and will perform poorly. In addition, the network must not solely be shown examples of accurate driving, but also how to recover (i.e. return to the road center) once a mistake has been made. Partial initial training on

ALVINN:
AN AUTONOMOUS LAND VEHICLE IN A NEURAL NETWORK

Dean A. Pomerleau
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213

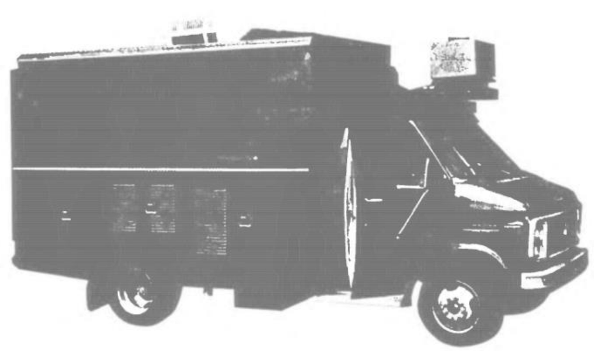
ABSTRACT

INTRODUCTION

NETWORK ARCHITECTURE

Disadvantages of Direct Imitation Learning

- Needs a lot of demonstrations to generalize
- High variability in the demonstrations
- Demonstrate how to recover from mistakes



Motivation for inverse RL

Apprenticeship learning/Imitation learning through inverse RL

Presupposition: reward function provides the most succinct and transferable definition of the task

Has enabled advancing the state of the art in various robotic domains

Modeling of other agents, both adversarial and cooperative

Scientific questions

Model animal and human behavior

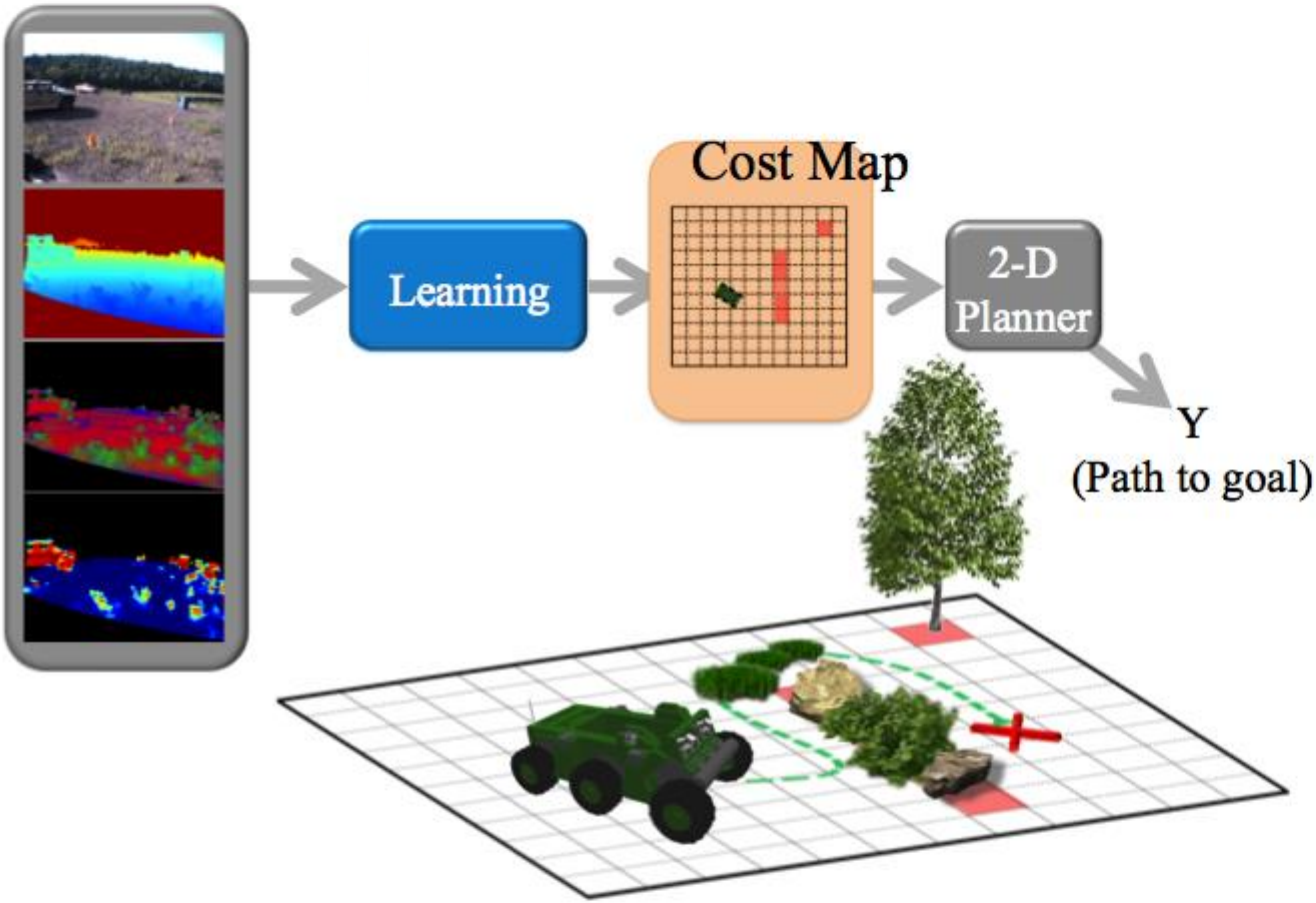
E.g., bee foraging, songbird vocalization. [See intro of Ng and Russell, 2000 for a brief overview.]

Meet Crusher...

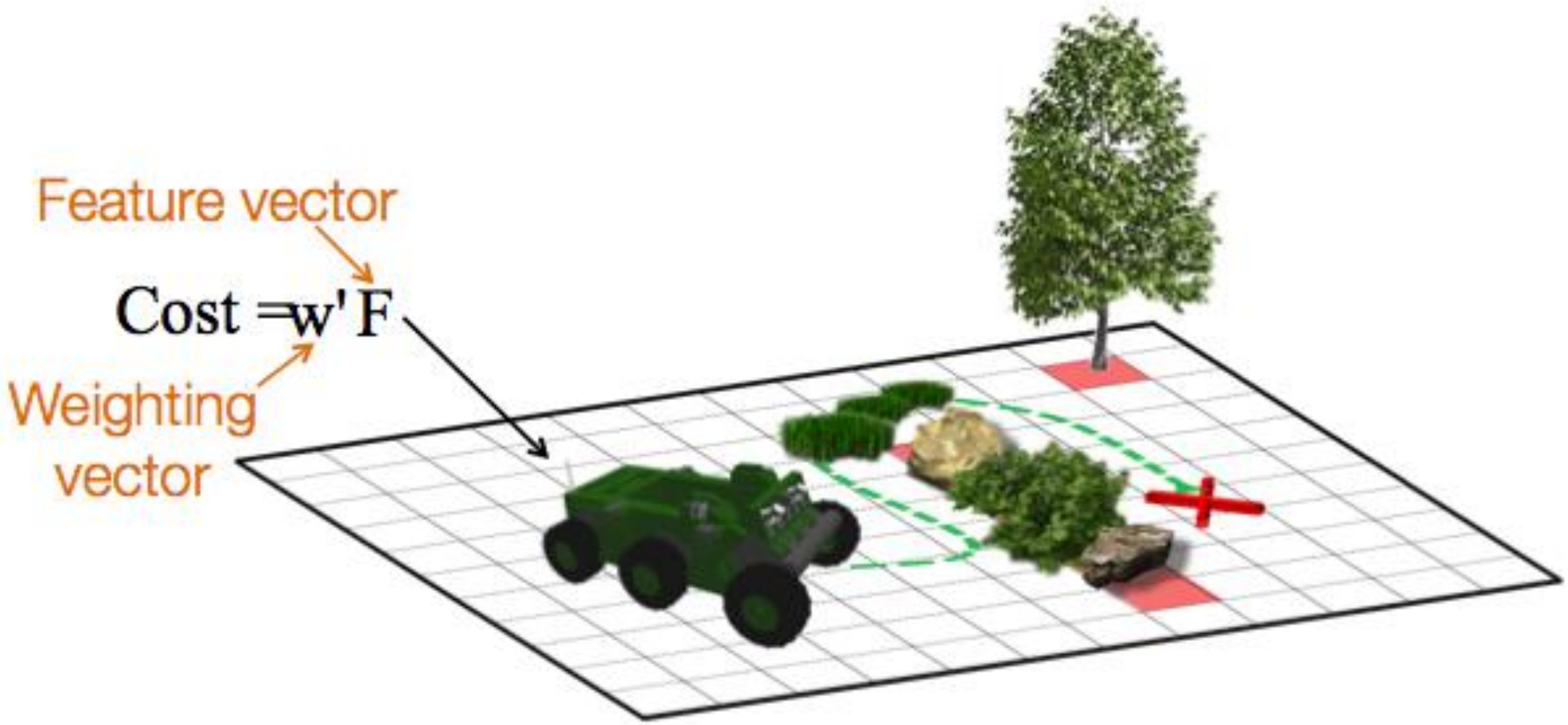
More Crusher pictures...

More Crusher pictures...

Inverse Reinforcement Learning

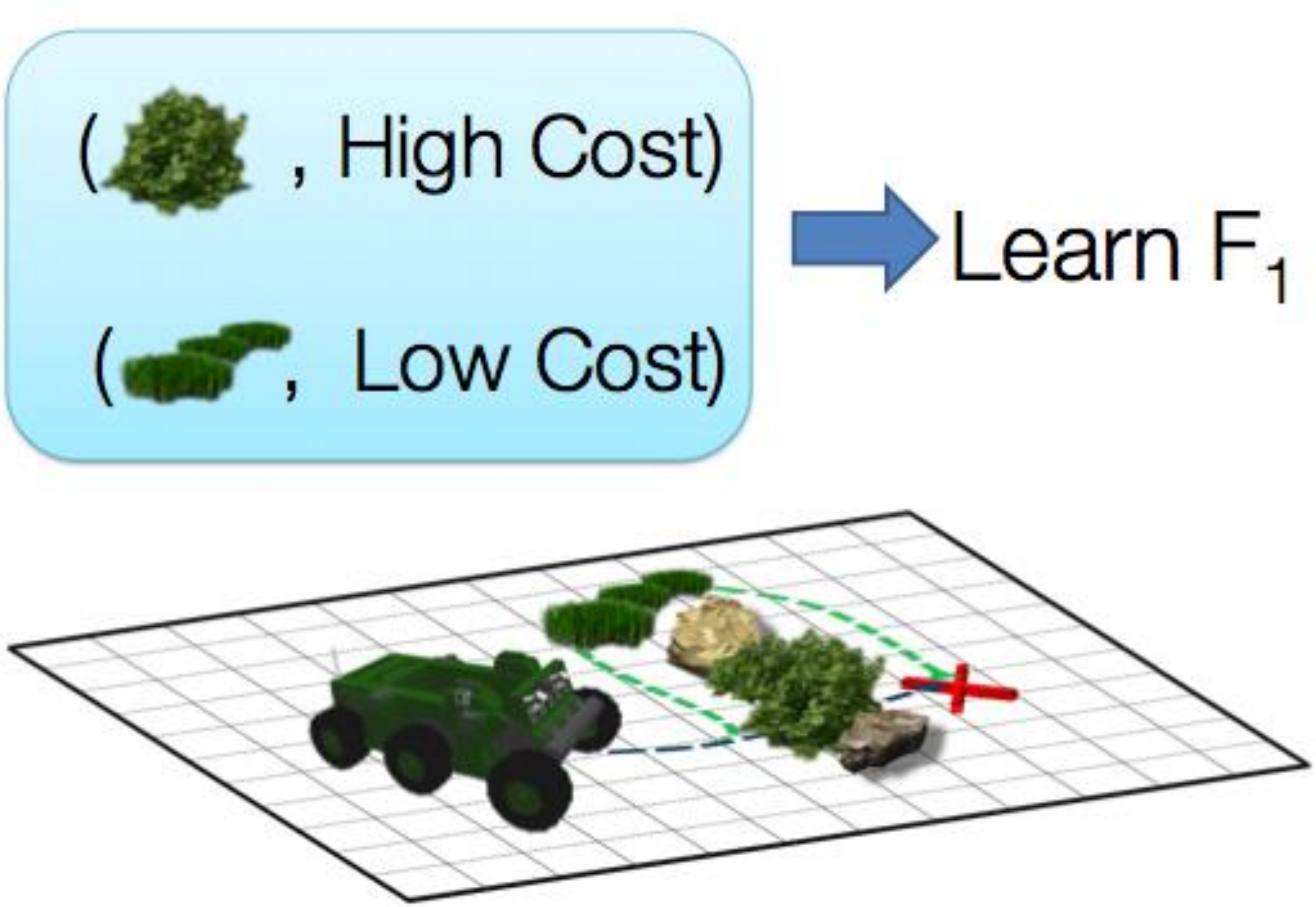


Recovering Cost!

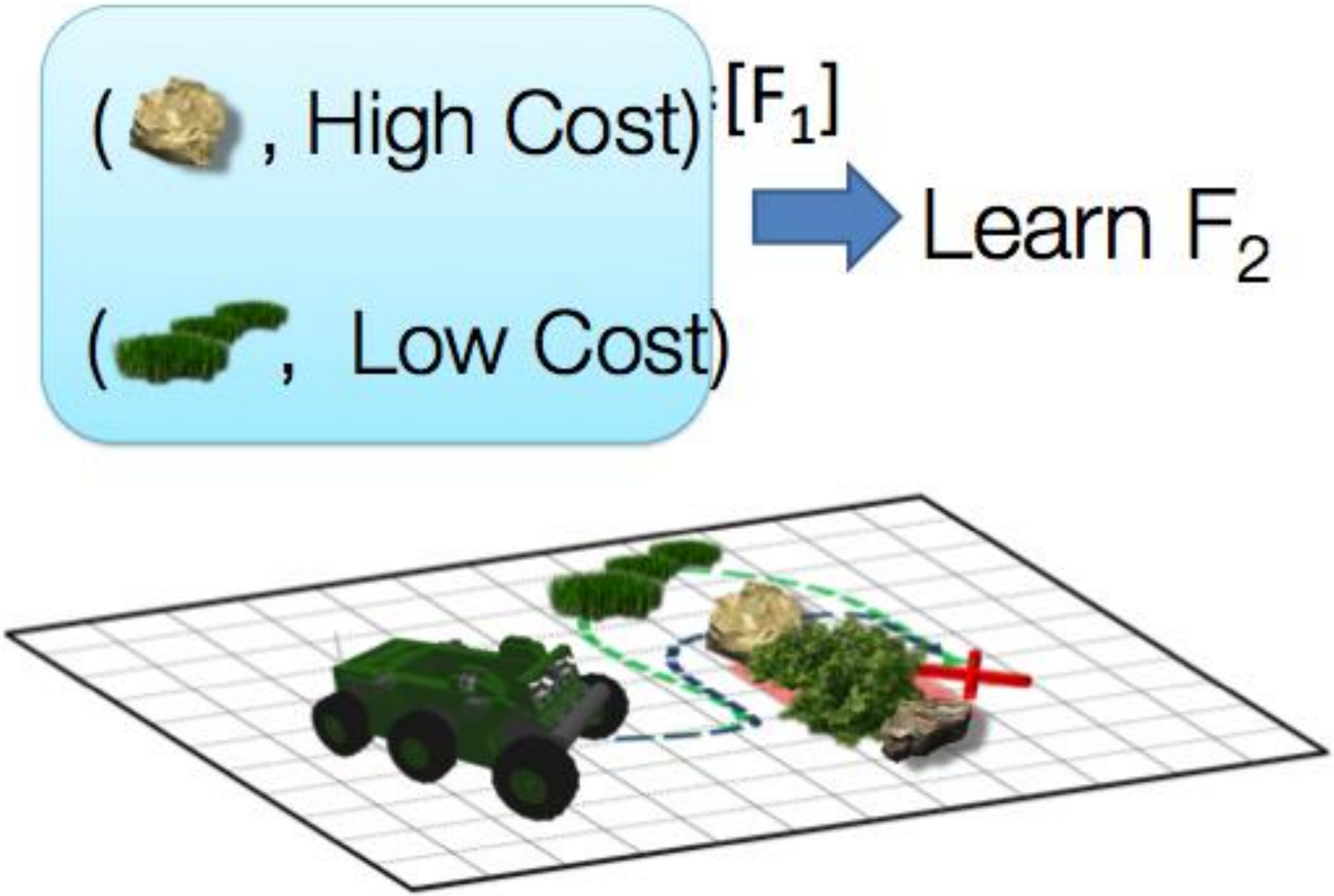


Ratliff, Bagnell, Zinkevich 2005
Ratliff, Bradley, Bagnell, Chestnutt, NIPS 2006
Silver, Bagnell, Stentz, RSS 2008

Recovering Cost!



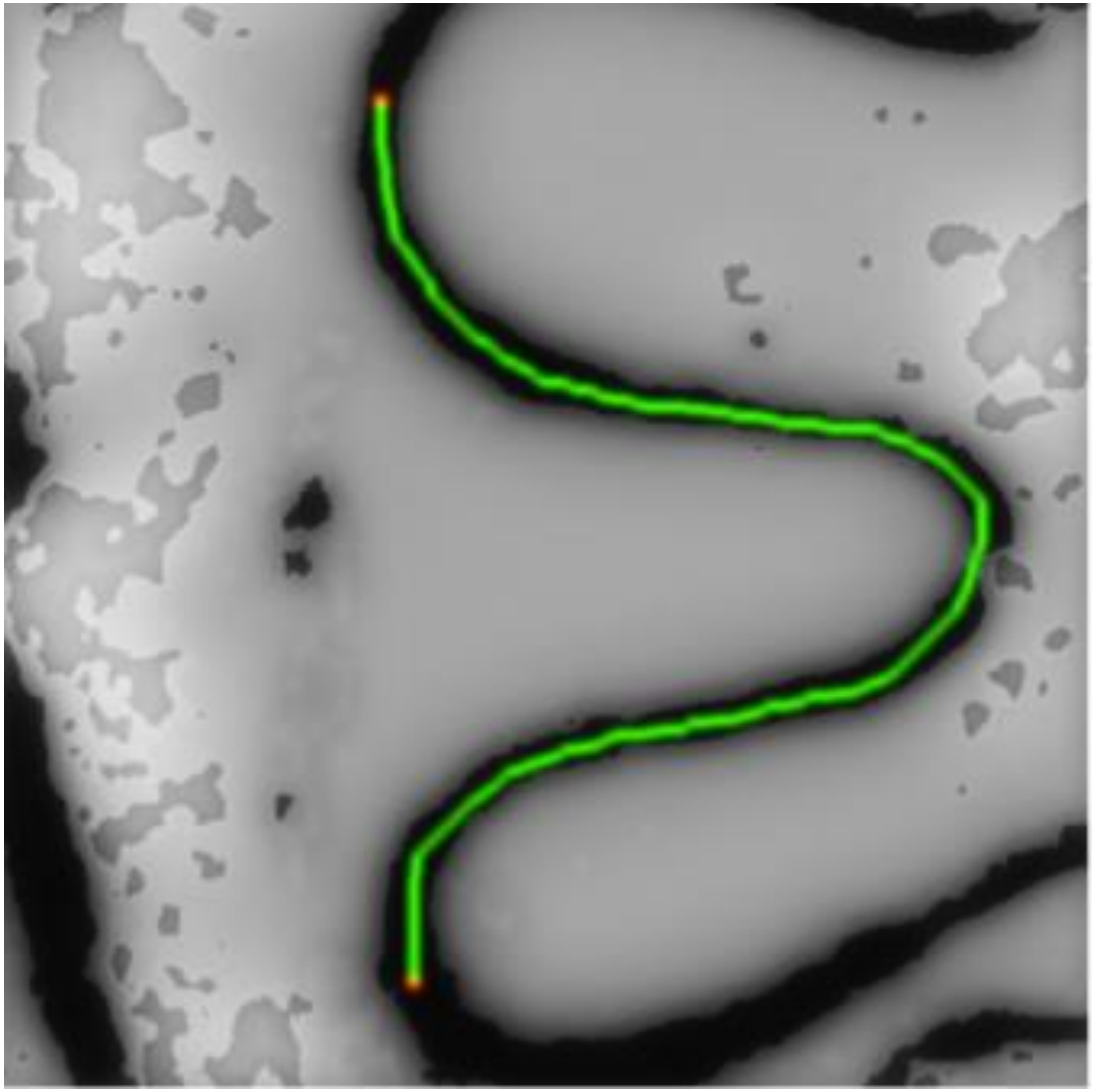
Recovering Cost!



Collect paths by teleoperation

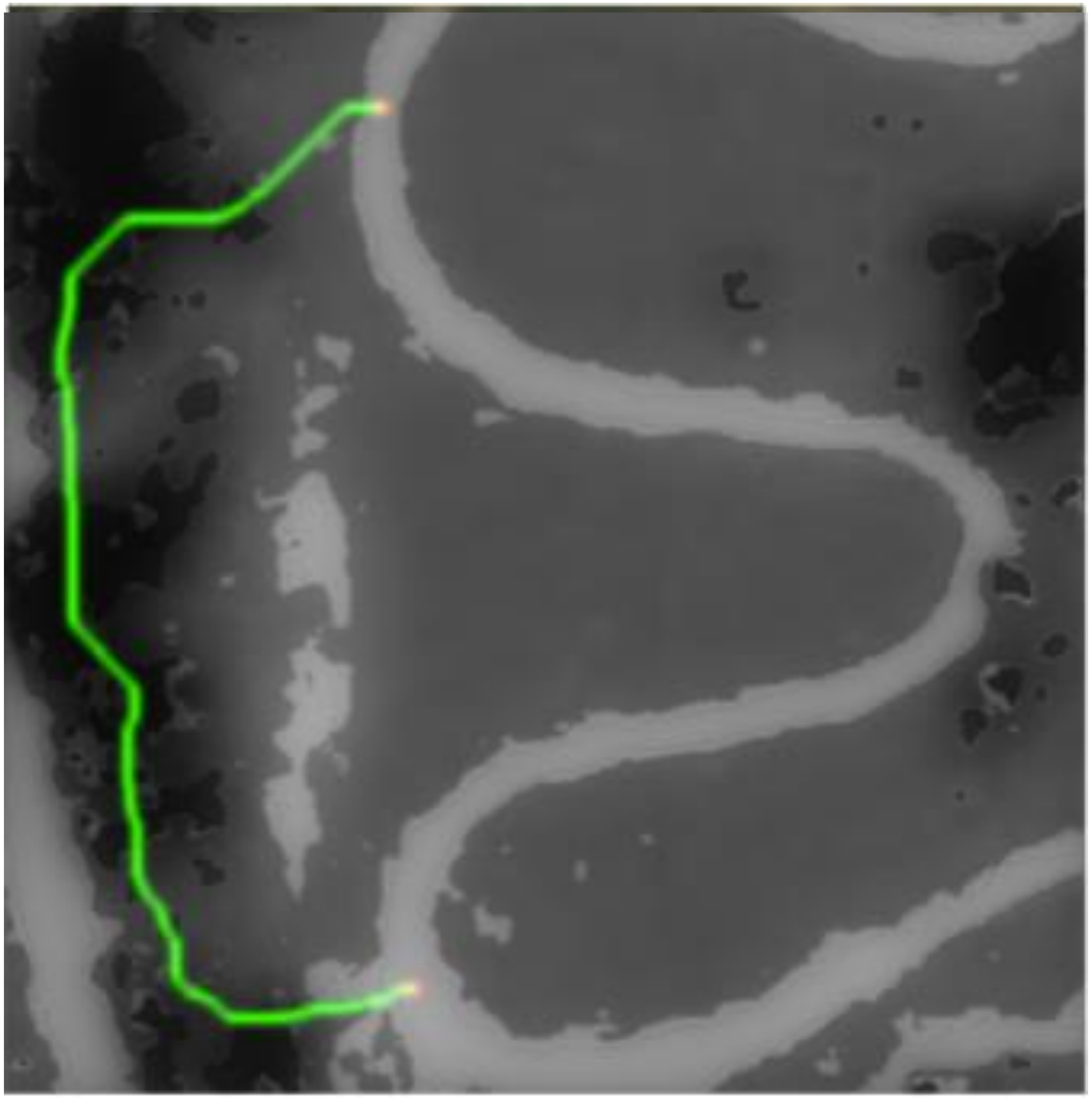
Training: Stay on the road

Test: Stay on the road



Training: Avoid the Road

Test: Avoid the Road




example path

High-level picture

Dynamics

Probability distribution over next states given current state and action

Describes desirability of being in a state.

Reward Function $r(s_t)$

Reinforcement Learning / Optimal Control

Controller/ Policy π

Prescribes action to take for each state

$$\text{argmax}_\pi \mathbb{E}_{p,\pi}\left[\sum_t \gamma^t r(s_t) | \pi\right]$$

Inverse RL:

- Given Policy and Model, can we recover R?
- More generally, given execution traces, can we recover r?

Outline of the Lecture

1. Introduction
   I. Comparison to Behavioral Cloning
2. Categories of IRL
   I. Maximum Margin
   II. Feature Matching by Max. Entropy
   III. Policy parametrized by rewards
3. Applications
4. Conclusion

Problem setup: Behavioral Cloning

Input:

Teacher's demonstrations: $D = \{\boldsymbol{s}_{1:T,i}, \boldsymbol{a}_{1:T,i}\}_{i=1\ldots N}$

Trace of the teacher's policy $\pi^*(\boldsymbol{a}|\boldsymbol{s})$

And its "long-term behavior" $\mu^*(\boldsymbol{s})$

Formulated as standard machine learning problem

Fix a policy class (neural network, decision tree, deep belief net, dynamical systems, …

Estimate a policy $\hat{\pi}(\boldsymbol{a}|\boldsymbol{s})$ from the training examples D

Problem:

There will always be an error in the estimation of the policy

Small error in the policy $\Rightarrow$ possibly large error in long-term behavior $\hat{\mu}(\boldsymbol{s})$

Problem setup: Inverse RL

Input:

Teacher's demonstrations: $D = \{\boldsymbol{s}_{1:T,i}, \boldsymbol{a}_{1:T,i}\}_{i=1\dots N}$

Trace of the teacher's policy $\pi^*(\boldsymbol{a}|\boldsymbol{s})$

And its "long-term behavior" $\mu^*(\boldsymbol{s})$

State and Action Space

Transition model: $p(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, \boldsymbol{a}_t)$

No reward function $r(\boldsymbol{s}_t)$

Inverse RL:

Can we recover $r(\boldsymbol{s}_t)$ that explains the policy $\pi^*(\boldsymbol{a}|\boldsymbol{s})$ (and its long-term behavior ?) $\mu^*(\boldsymbol{s})$

Apprenticeship Learning

Can we use $r(\boldsymbol{s}_t)$ to obtain a policy $\hat{\pi}(\boldsymbol{a}|\boldsymbol{s})$?

Inverse RL vs. Behavioral Cloning

Behavioral Cloning:

Simple to implement

No assumptions on the model/MDP

We might not reproduce the long term behavior

Representation: Policy

Hard to generalize

Needs many samples

Inverse RL:

Requires Planning / Solving an MDP

Hard for many interesting MDPs (e.g. high-DoF robots)

Representation: Reward

Compact description

Easy to transfer to new tasks

Basic principle

Find a reward function $r^*(\boldsymbol{s}_t)$ **which explains the expert behavior**

Assume expert is optimal w.r.t. to $r^*(\boldsymbol{s}_t)$

I.e., find $r^*(\boldsymbol{s}_t)$ **such that**

$$\mathbb{E}_{p,\pi}\left[\sum_t \gamma^t r^*(s_t) | \pi^*\right] \geq \mathbb{E}_{p,\pi}\left[\sum_t \gamma^t r^*(s_t) | \pi\right], \forall \pi$$

In fact a convex feasibility problem, but many challenges:

1. Ill-posed: $r^*(\boldsymbol{s}_t) = 0$ is a solution, reward function ambiguity
2. Limited Data: We typically only observe expert traces rather than the entire expert optimal policy --- how to compute left-hand side?
3. Optimality Assumption: Assumes the expert is indeed optimal --- otherwise infeasible
4. Computation: assumes we can enumerate all policies

Outline of the Lecture

1. Introduction
   I. Comparison to Behavioral Cloning
2. Categories of IRL
   I. Maximum Margin
   II. Feature Matching by Max. Entropy
   III. Policy parametrized by rewards
3. Applications
4. Conclusion

Feature based reward function

Lets assume the reward function is linear in some features,

I.e. $r(\boldsymbol{s}) = \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{s})$, where $\boldsymbol{\phi}(\boldsymbol{s})$ is a n-dimensional feature vector

$$
\begin{aligned}
\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(\boldsymbol{s}_t) | \pi\right] &= \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{s}_t) | \pi\right] \\
&= \boldsymbol{w}^T \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \boldsymbol{\phi}(\boldsymbol{s}_t) | \pi\right] \\
&= \boldsymbol{w}^T \boldsymbol{\psi}(\pi)
\end{aligned}
$$

where $\boldsymbol{\psi}(\pi)$ is the expected discounted feature vector of policy π

Subbing into: $\mathbb{E}_{p,\pi}\left[\sum_t \gamma^t r^*(s_t) | \pi^*\right] \geq \mathbb{E}_{p,\pi}\left[\sum_t \gamma^t r^*(s_t) | \pi\right], \forall \pi$

gives us: Find $\boldsymbol{w}^*$ such that $\boldsymbol{w}^{*T} \boldsymbol{\psi}(\boldsymbol{\pi}^*) \geq \boldsymbol{w}^{*T} \boldsymbol{\psi}(\boldsymbol{\pi}), \forall \pi$

Feature based reward function

$$\mathbb{E}_{p,\pi}\left[\sum_t \gamma^t r^*(s_t)|\pi^*\right] \geq \mathbb{E}_{p,\pi}\left[\sum_t \gamma^t r^*(s_t)|\pi\right], \forall\pi$$

$$\text{Find } \boldsymbol{w}^* \text{ such that } \boldsymbol{w}^{*T}\boldsymbol{\psi}(\boldsymbol{\pi}^*) \geq \boldsymbol{w}^{*T}\boldsymbol{\psi}(\boldsymbol{\pi}), \forall\pi$$

Feature expectations can be readily estimated from sample trajectories

- Solves limited data challenge
- The number of expert demonstrations required scales with the number of features in the reward function.

The number of expert demonstration required does not depend on

- Complexity of the expert's optimal policy
- Size of the state space

Basic principle

Find a reward function $r^*(\boldsymbol{s}_t)$ **which explains the expert behavior**

Assume expert is optimal w.r.t. to $r^*(\boldsymbol{s}_t)$

I.e., find $r^*(\boldsymbol{s}_t)$ **such that**

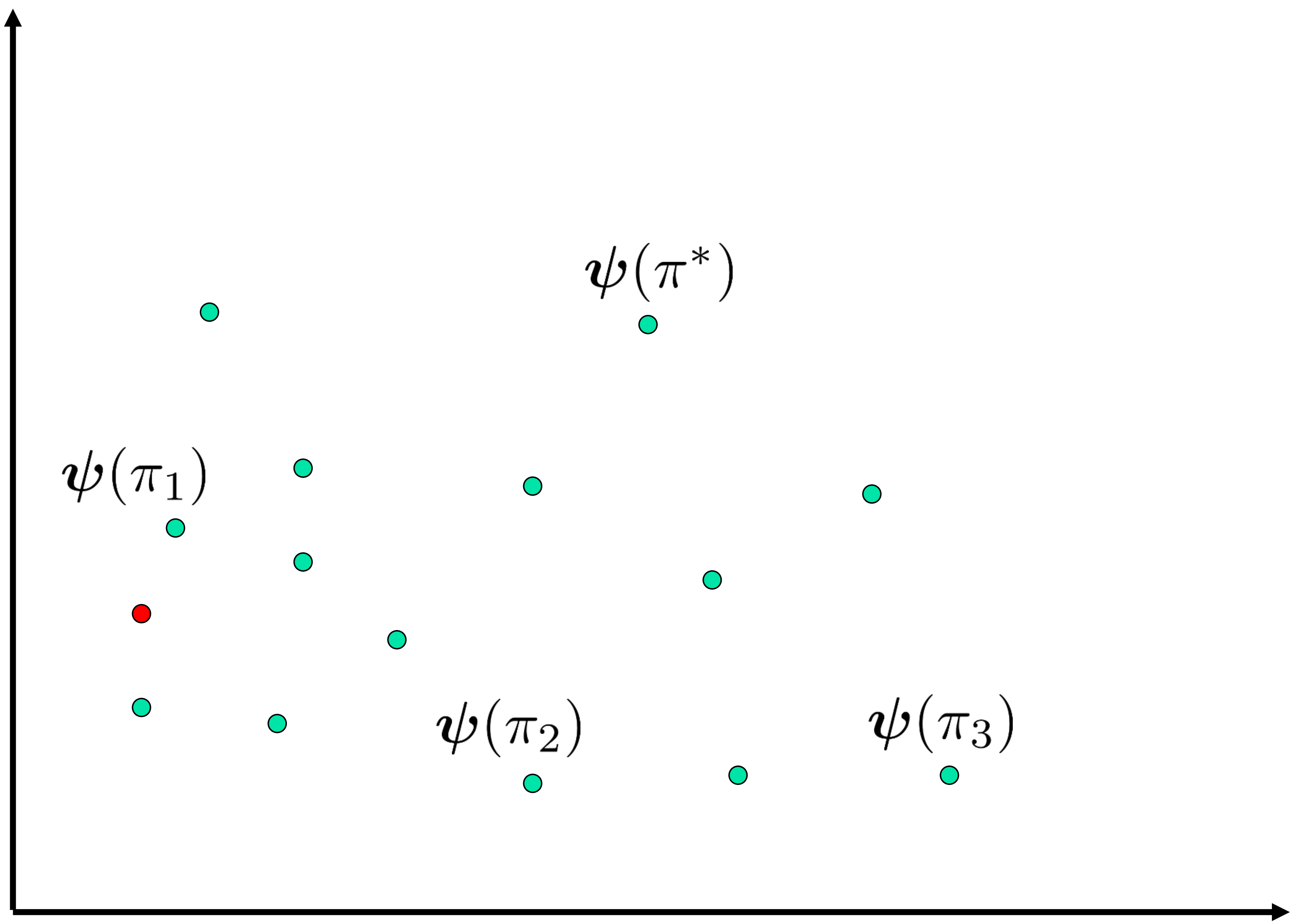
$$\mathbb{E}_{p,\pi}\left[\sum_t \gamma^t r^*(s_t) | \pi^*\right] \geq \mathbb{E}_{p,\pi}\left[\sum_t \gamma^t r^*(s_t) | \pi\right], \forall \pi$$

In fact a convex feasibility problem, but many challenges:

1. Ill-posed: $r^*(\boldsymbol{s}_t) = 0$ is a solution, reward function ambiguity
2. Limited Data: We typically only observe expert traces rather than the entire expert optimal policy --- how to compute left-hand side?
3. Optimality Assumption: Assumes the expert is indeed optimal --- otherwise infeasible
4. Computation: assumes we can enumerate all policies

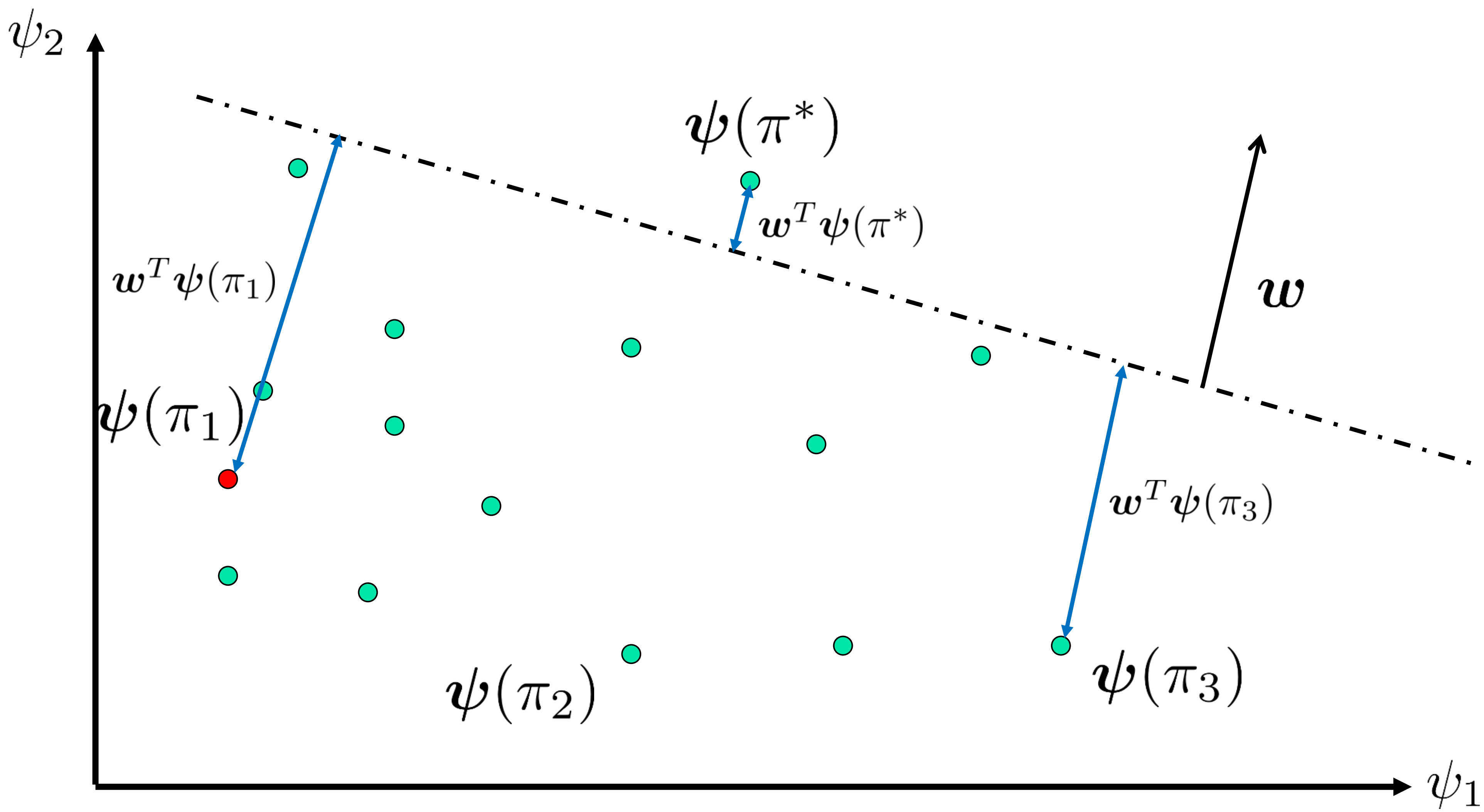
Constraint generation

Every policy has a corresponding feature expectation vector, which for visualization purposes we assume to be 2D



Constraint generation

Linear parametrization: We need to find a separating hyper-plane given by $\boldsymbol{w}$



Scalar product $\boldsymbol{w}^T\boldsymbol{\psi}$ gives us the (positive or negative) distance to the separating hyper-plane

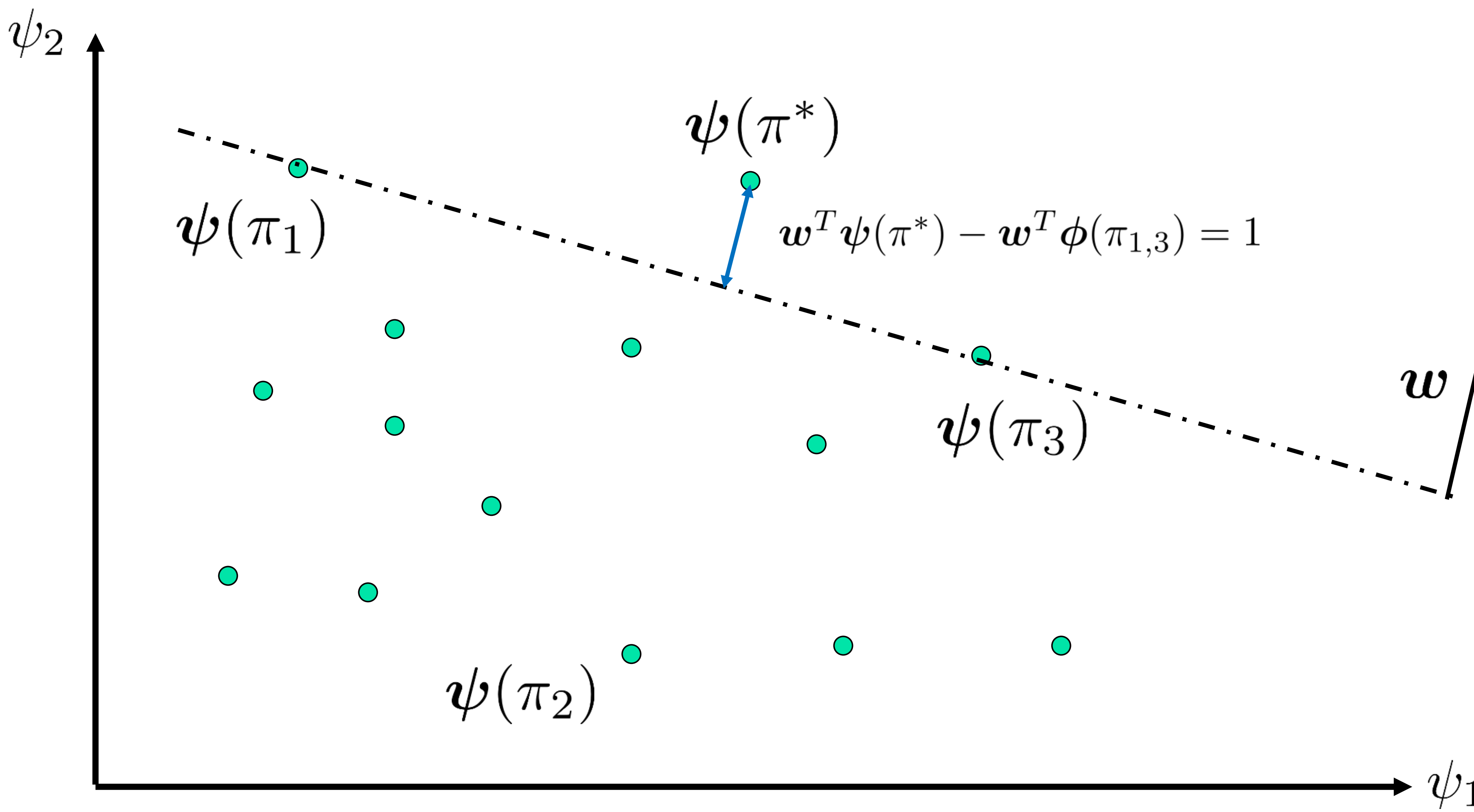
Ill-posed Problem

Standard max margin:

Smallest weight vector with predefined reward margin of 1

$$\begin{aligned} \min_{\boldsymbol{w}} \quad & \boldsymbol{w}^T \boldsymbol{w} \\ \text{s.t.} \quad & \boldsymbol{w}^T \boldsymbol{\psi}(\pi^*) \geq \boldsymbol{w}^T \boldsymbol{\psi}(\pi) + 1, \quad \forall \pi \end{aligned}$$

Max. margin solution



Similar interpretation as a support vector machine

Ill-posed Problem

Structured max margin:

Smallest weight vector

Margin depends on difference of policies $m(\pi^*, \pi)$

$$\begin{aligned} \min_{\boldsymbol{w}} \quad & \boldsymbol{w}^T \boldsymbol{w} \\ \text{s.t.} \quad & \boldsymbol{w}^T \boldsymbol{\psi}(\pi^*) \geq \boldsymbol{w}^T \boldsymbol{\psi}(\pi) + m(\pi^*, \pi), \quad \forall \pi \end{aligned}$$

Justification: margin should be larger for policies that are very different from π^*.

Example for $m(\pi^*, \pi)$ **:**

Sum of minimum distances from generated path the example path

Basic principle

Find a reward function $r^*(\boldsymbol{s}_t)$ which explains the expert behavior

Assume expert is optimal w.r.t. to $r^*(\boldsymbol{s}_t)$

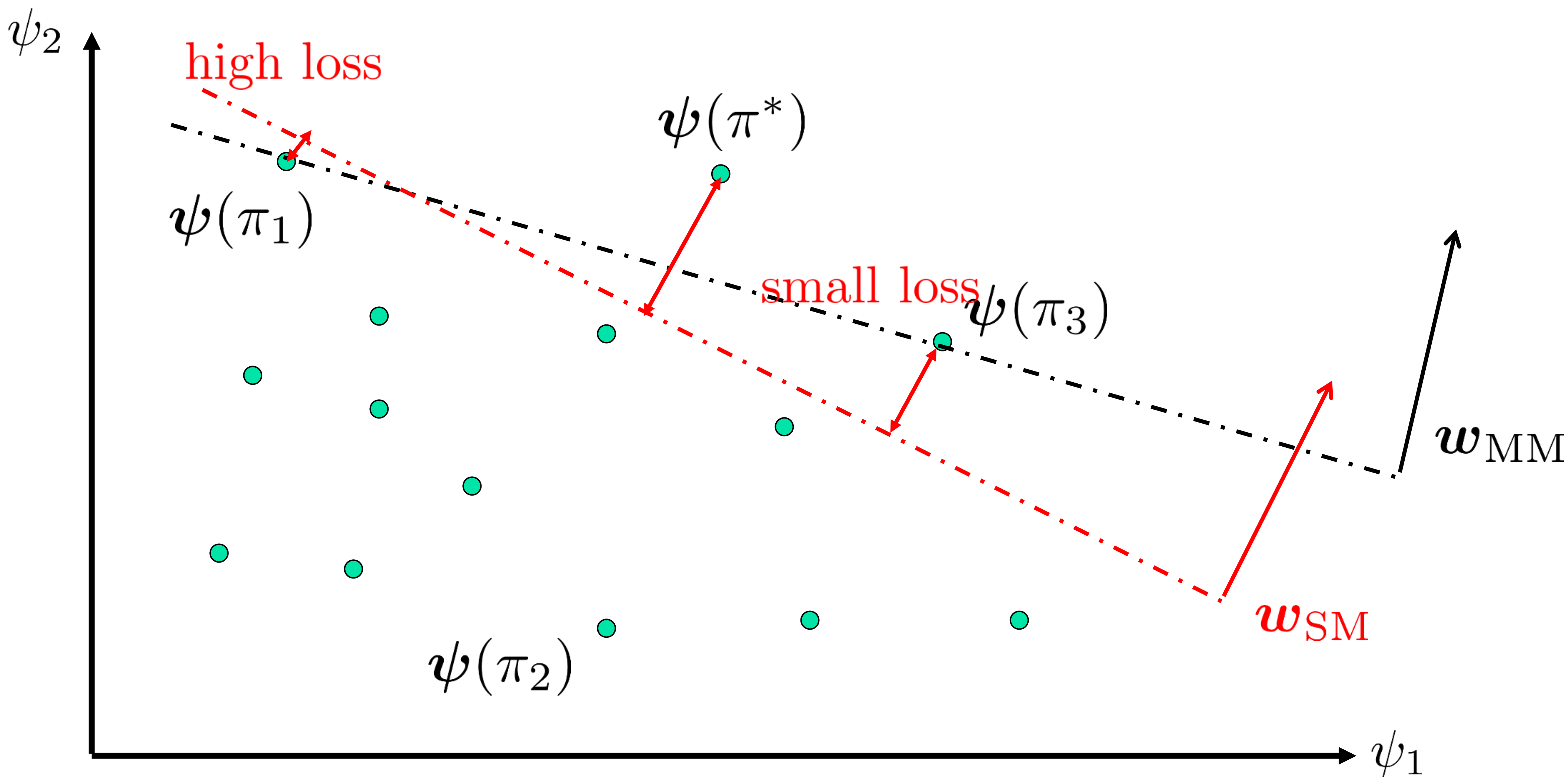
I.e., find $r^*(\boldsymbol{s}_t)$ such that

$$\mathbb{E}_{p,\pi}\left[\sum_t \gamma^t r^*(s_t) | \pi^*\right] \geq \mathbb{E}_{p,\pi}\left[\sum_t \gamma^t r^*(s_t) | \pi\right], \forall \pi$$

In fact a convex feasibility problem, but many challenges:

1. Ill-posed: $r^*(\boldsymbol{s}_t) = 0$ is a solution, reward function ambiguity
2. Limited Data: We typically only observe expert traces rather than the entire expert optimal policy --- how to compute left-hand side?
3. Optimality Assumption: Assumes the expert is indeed optimal --- otherwise infeasible
4. Computation: assumes we can enumerate all policies

Structured max margin solution



Expert suboptimality

Structured prediction max margin with slack variables:

Every constraint can be violated a bit

Minimize the amount of violation

$$
\begin{aligned}
\min_{\boldsymbol{w},\xi} \quad & \boldsymbol{w}^T\boldsymbol{w} + C\xi \\
\text{s.t.} \quad & \boldsymbol{w}^T\boldsymbol{\psi}(\pi^*) \geq \boldsymbol{w}^T\boldsymbol{\psi}(\pi) + m(\pi^*, \pi) - \xi, \quad \forall \pi
\end{aligned}
$$

Easy to extend to multiple MDPs

Resolved: access to π^*, ambiguity, expert suboptimality

One challenge remains: very large number of constraints

Ratliff et. al. use subgradient methods.

In this lecture: constraint generation

Constraint generation

Initialize $\Pi = \{\}$ **and then iterate** k = 1…

Solve

$$\begin{aligned} \boldsymbol{w}^{(k)} = \quad & \operatorname{argmin}_{\boldsymbol{w}} \min_{\xi} \boldsymbol{w}^T \boldsymbol{w} + C\xi \\ \text{s.t.} \quad & \boldsymbol{w}^T \boldsymbol{\psi}(\pi^*) \geq \boldsymbol{w}^T \boldsymbol{\psi}(\pi^{(i)}) + m(\pi^*, \pi^{(i)}) - \xi, \quad \forall \pi^{(i)} \in \Pi \end{aligned}$$

Find the most violated constraint

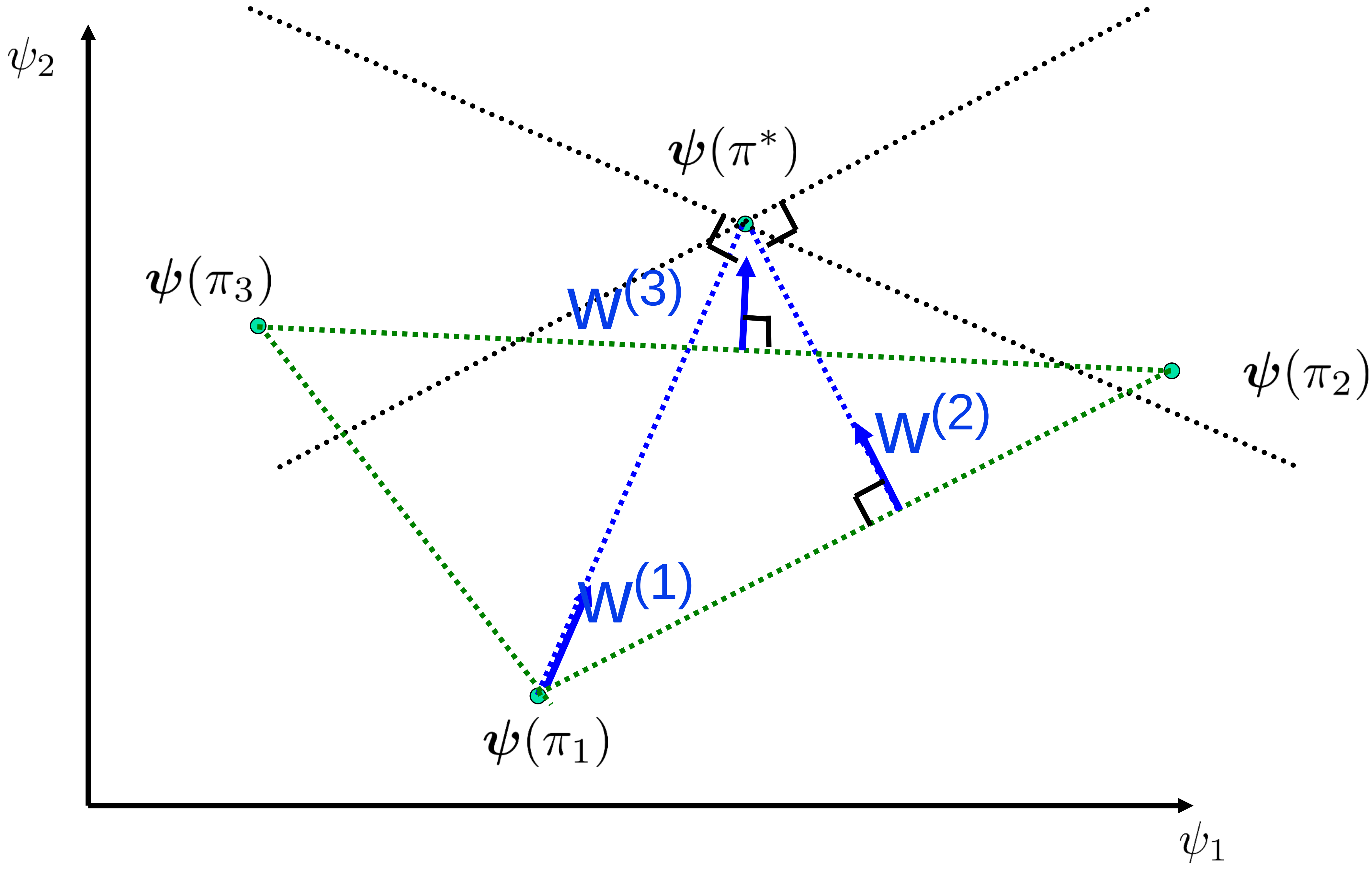
$$\pi^{(k)} = \max_{\pi} \boldsymbol{w}^{(k),T} \boldsymbol{\psi}(\pi) + m(\pi^*, \pi)$$

Compute optimal policy for the current estimate of the reward function (+ loss augmentation m), e.g., dynamic programming

Add $\pi^{(k)}$ **to set of policies** Π

If no constraint violations were found, we are done.

Algorithm example run

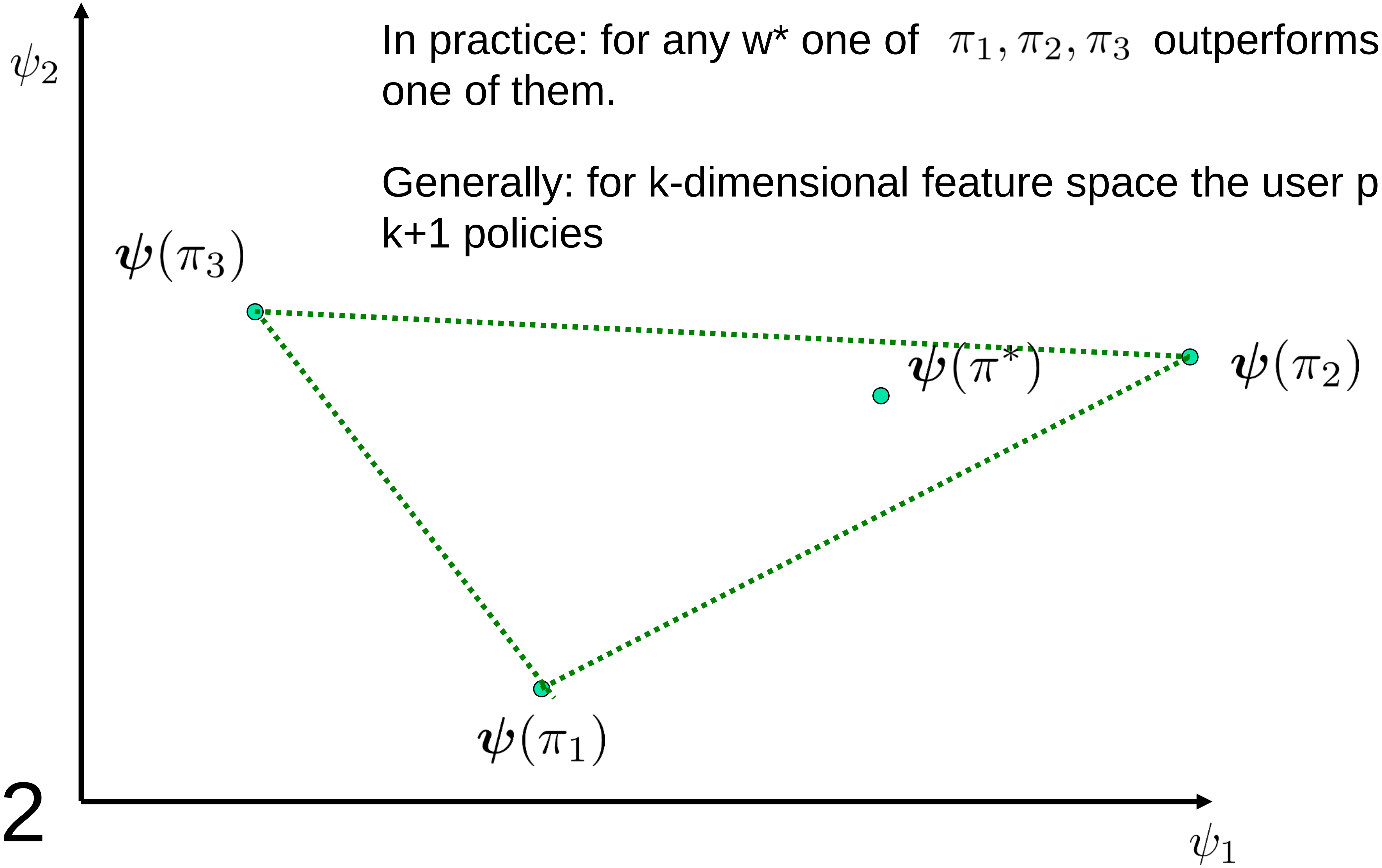


Suboptimal expert case

Can match expert by stochastically mixing between 3 policies

In practice: for any w* one of π_1, π_2, π_3 outperforms π^* → pick one of them.

Generally: for k-dimensional feature space the user picks between k+1 policies



Outline of the Lecture

1. Introduction
   I. Comparison to Behavioral Cloning
2. Categories of IRL
   I. Maximum Margin
   II. Feature Matching by Max. Entropy
   III. Policy parametrized by rewards
3. Applications
4. Conclusion

Classical Approach to Statistical Modeling: "The Principle of Maximum Entropy"

Premise: Statistical modeling should be performed with the least commitment possible.

- Predict using the probability distribution minimally committed/ maximally uncertain/*highest Shannon entropy*....
- ...subject to it agrees with known constraints
- “almost all” distributions are close to the MaxEnt one
- Uncertainty allows proper treatment of suboptimal demonstrations!

Maximum-entropy approach to inference

What is the maximum entropy distribution with given 1st and 2nd order and moments?

$$\operatorname{argmax}_p \qquad \underbrace{-\sum_x p(x) \log p(x)}_{\text{Entropy}}$$

$$\text{s.t.} \qquad \sum_x p(x)x = m_1, \quad \sum_x p(x)x^2 = m_2, \quad \sum_x p(x) = 1$$

Solution:

$$p(x) \propto \exp(\lambda_1 x + \lambda_2 x^2), \quad \text{where } \lambda_{1,2} \text{ are lagrangian multipliers}$$

That's a Gaussian! … which is of course well known that a Gaussian has maximum entropy of all distributions with given mean and variance

Maximum-entropy approach to IRL [Ziebart 2008]

Maximize entropy **over paths** **with a given feature expectation**

$$\operatorname{argmax}_p \quad -\sum_{\boldsymbol{\tau}} p(\boldsymbol{\tau}) \log p(\boldsymbol{\tau})$$

$$\text{s.t.} \quad \sum_{\boldsymbol{\tau}} p(\boldsymbol{\tau}) \boldsymbol{\psi}(\boldsymbol{\tau}) = \boldsymbol{\psi}(\pi^*), \quad \sum_{\boldsymbol{\tau}} p(\boldsymbol{\tau}) = 1$$

Solution: $p(\boldsymbol{\tau}) \propto \exp(\boldsymbol{w}^T \boldsymbol{\psi}(\boldsymbol{\tau}))$

w is now a lagrangian multiplier

We obtain a soft-max distribution over trajectories

Return of the trajectories: $R(\boldsymbol{\tau}) = \boldsymbol{w}^T \sum_t \gamma^t \boldsymbol{\phi}(\boldsymbol{s}_t) = \boldsymbol{w}^T \boldsymbol{\psi}(\boldsymbol{\tau})$

Problem: Does not take system dynamics into account

Trajectory could have huge return, but is very unlikely due to system dynamics

Maximum-Causal-entropy IRL [Ziebart 2010]

Maximize entropy of the policy with a given feature expectation

$$\operatorname{argmax}_\pi \quad -\sum_{t,a} \mu_t(s) \sum_a \pi_t(a|s) \log \pi_t(a|s)$$

Max. Caus. Ent

$$\text{s.t. } \forall t: \quad \sum_t \sum_s \mu_t(s)\phi(s) = \psi(\pi^*), \quad \sum_a \pi(a|s) = 1, \forall s$$

Match Features

$$\mu_t(s') = \sum_{s,a} \mu_{t-1}(s)\pi_{t-1}(a|s)p(s'|s,a),$$

State distribution consistency

State distribution at time step t has to be consistent with:

- state distribution and policy at time step t-1
- system dynamics

Maximum-Causal-entropy IRL [Ziebart 2010]

Solution: $\pi_t(a|s) \propto \exp\left(\boldsymbol{w}^T\boldsymbol{\phi}(s) + \mathbb{E}[V_{t+1}(s')|s,a]\right)$

$V_t(s)$ is again a Lagrangian multiplier

If we say $\boldsymbol{w}^T\boldsymbol{\phi}(s)$ is the reward, this is a soft-max over the Q-function

$$\pi_t(a|s) \propto \exp\left(Q(s,a;\boldsymbol{w})\right)$$

This is still a convex problem:

Solution can be obtained by optimizing dual function

Can be done (relatively) efficiently

Outline of the Lecture

1. Introduction
   I. Comparison to Behavioral Cloning
2. Categories of IRL
   I. Maximum Margin
   II. Feature Matching by Max. Entropy
   III. Policy parametrized by rewards
3. Applications
4. Conclusion

Reward function parameterizing the policy class

Alternatively, we can assume the policy is soft-max in a Q-function

$$\pi_t(a|s;r,\alpha) = \frac{\exp\left(\alpha Q^*(s,a;r)\right)}{\sum_a' \exp\left(\alpha Q^*(s,a';r)\right)}$$

where Q^* is the optimal Q-function for reward function $r(s)$, i.e.,

$$V^*(s;r) = \max_a Q^*(s,a;r)$$

$$Q^*(s,a;r) = r(s) + \mathbb{E}[V^*(s';r)|s,a]$$

Then we can evaluate the likelihood of seeing a set of state-action pairs as follows:

$$\log p(D|r,\alpha) = \sum_i \log \pi(a_i|s_i;r,\alpha)$$

Is equivalent to “a smarter” Behavior Cloning!

Reward function parameterizing the policy class

Ziebart's approach can also be shown to be equivalent!

Can be extended to Bayesian setup:

Put prior on parameters of reward function

$$p(r, \alpha | D) = \frac{p(D | r, \alpha) p(r, \alpha)}{p(D)}$$

- Ramachandran and Amir, AAAI2007: MCMC method to sample from this distribution
- Neu and Szepesvari, UAI2007: gradient method to optimize the likelihood [MAP]

Open Directions

- Open directions:
  - Active inverse RL,
  - Inverse RL with minmax control
  - Inverse RL with partial observability
  - Inverse RL with learning stages (rather than observing optimal policy)
  - Many more …
- Are you interested? We may have an excellent thesis for you!

Outline of the Lecture

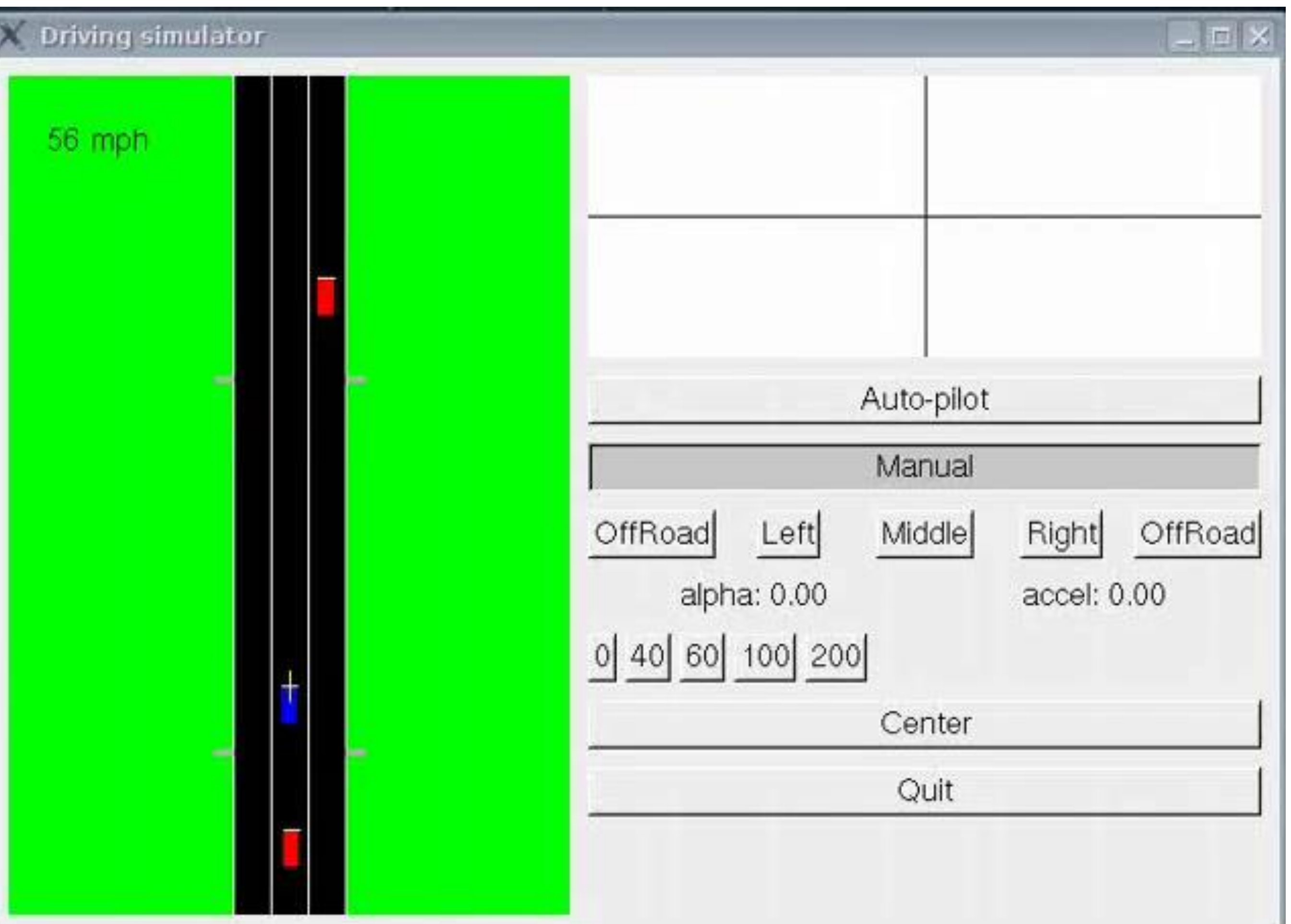
1. Introduction
   I. Comparison to Behavioral Cloning
2. Categories of IRL
   I. Maximum Margin
   II. Feature Matching by Max. Entropy
   III. Policy parametrized by rewards
3. Applications
4. Conclusion

Lecture outline

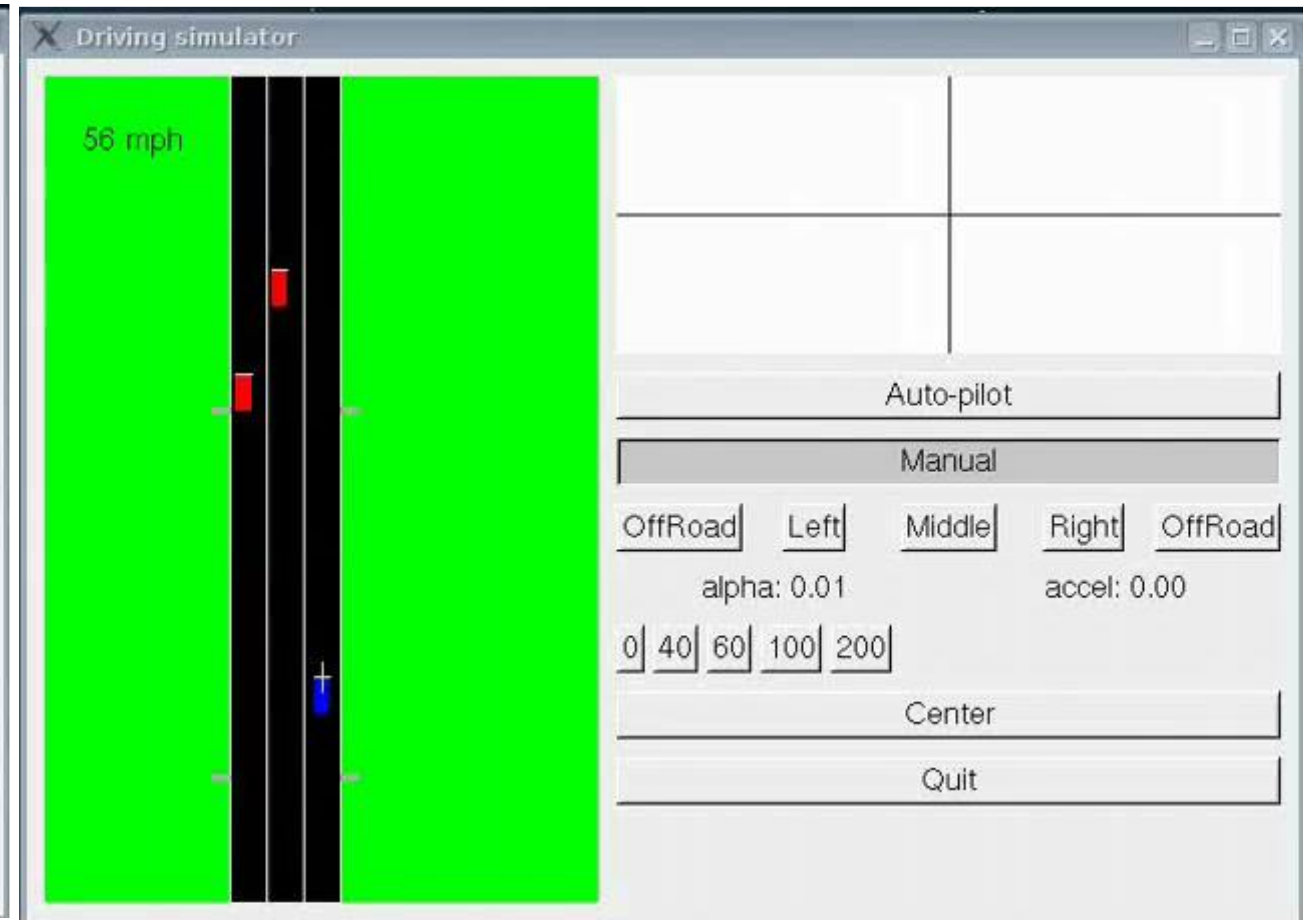
- Case studies:
  - (1) Highway driving,
  - (2) Crusher,
  - (3) Parking lot navigation,
  - (4) Route inference,
  - (5) Human path planning,
  - (6) Human inverse planning,
  - (7) Quadruped locomotion
  - (8) Helicopter Acrobatics

Highway driving

Teacher in Training World



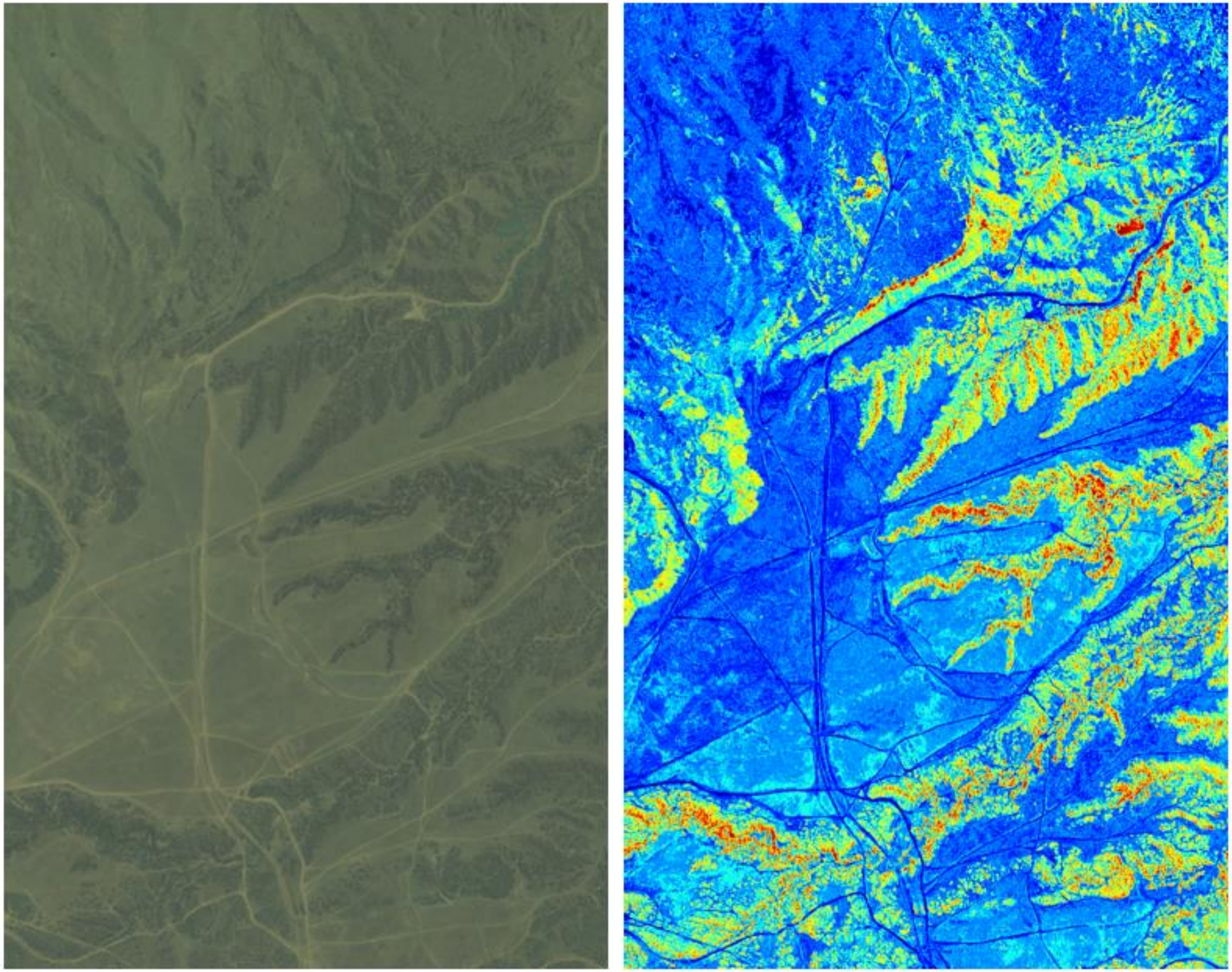
Learned Policy in Testing World



- Input:
  - Dynamics model / Simulator $P_{xu}(x_{t+1} \mid x_t, u_t)$
  - Teacher's demonstration: 1 minute in "training world"
  - Note: R* is unknown.
  - Reward features: 5 features corresponding to lanes/shoulders; 10 features corresponding to presence of other car in current lane at different distances

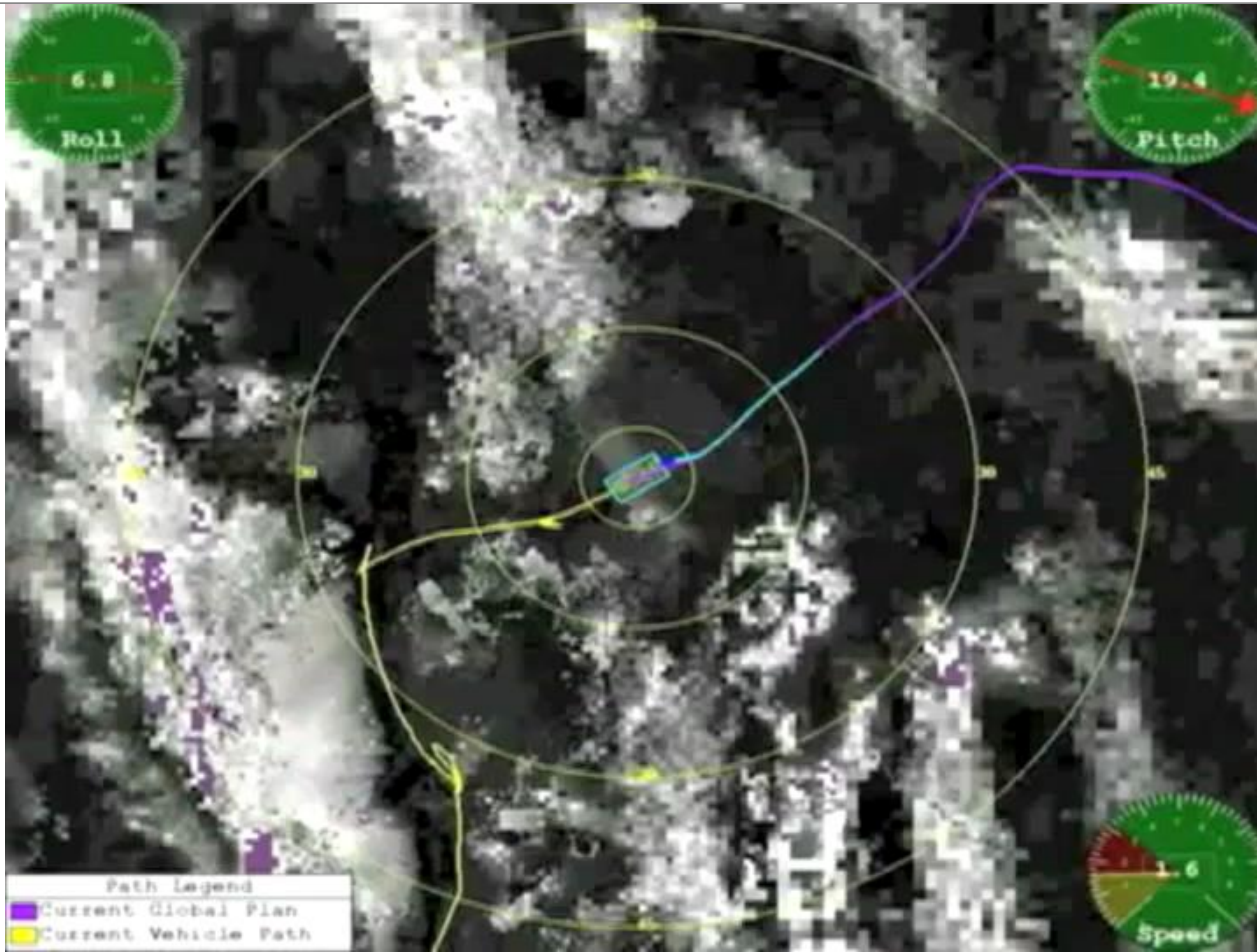
[Abbeel and Ng 2004]

Max margin



[Ratliff + al, 2006/7/8]

Max-margin



[Ratliff + al, 2006/7/8]

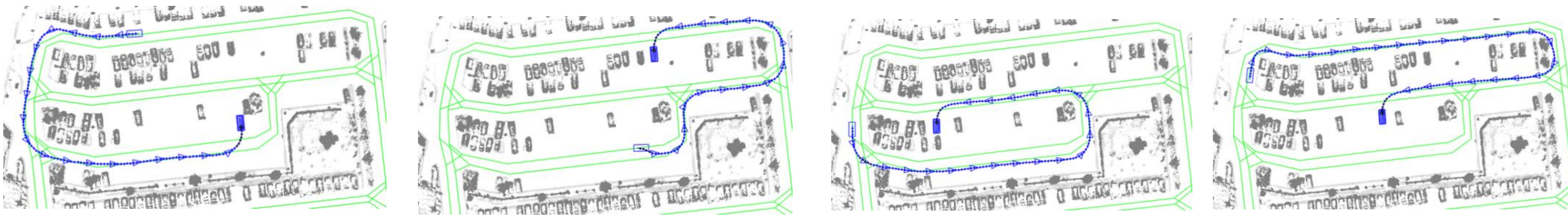
Parking lot navigation

- Reward function trades off:
  - Staying “on-road,”
  - Forward vs. reverse driving,
  - Amount of switching between forward and reverse,
  - Lane keeping,
  - On-road vs. off-road,
  - Curvature of paths.

[Abbeel et al., IROS 08]

Experimental setup

- Demonstrate parking lot navigation on “train parking lots.”



- Run our apprenticeship learning algorithm to find the reward function.
- Receive “test parking lot” map + starting point and destination.
- Find the trajectory that maximizes the learned reward function for navigating the test parking lot.

Nice driving style

Sloppy driving-style

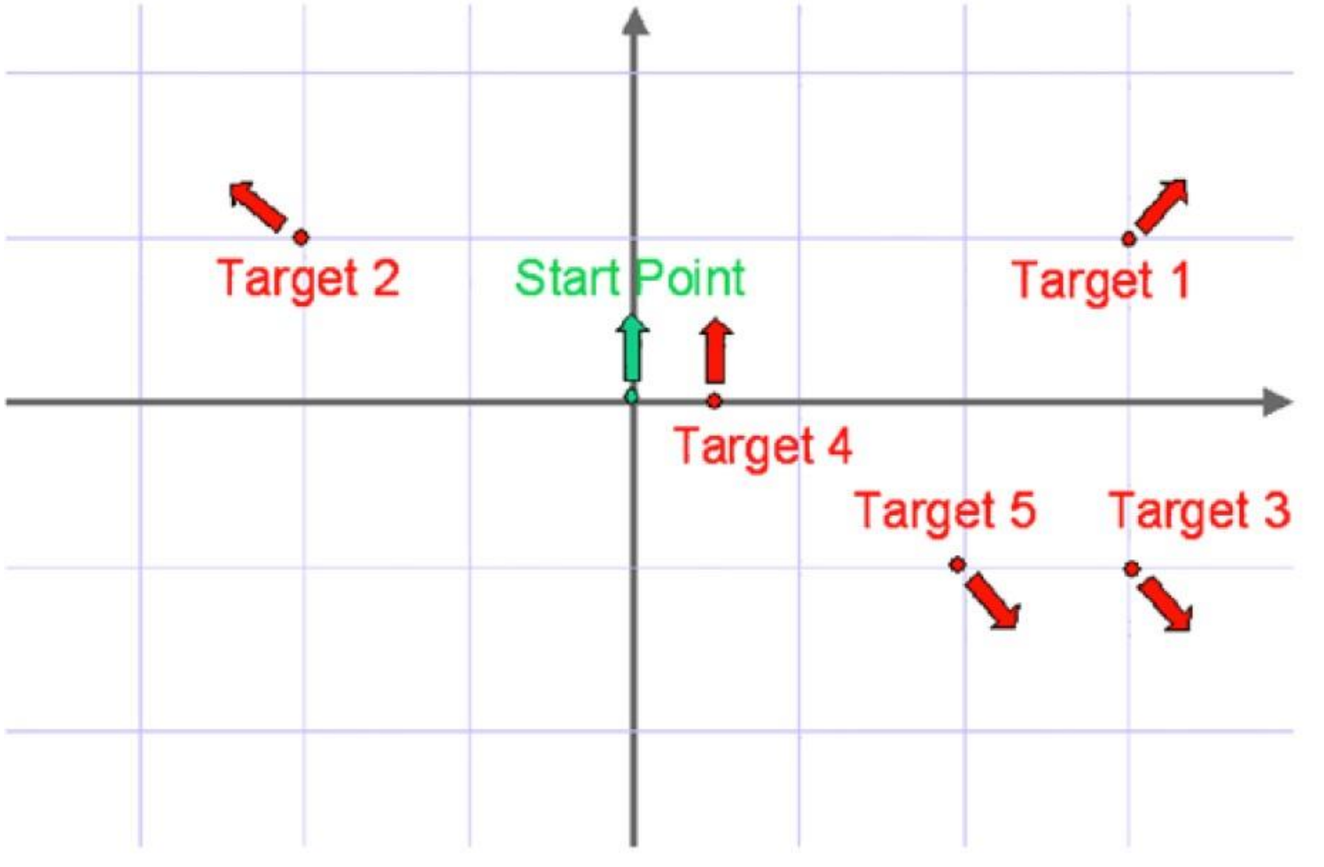
QuickTime™ and a
JVT/AVC Coding decompressor
are needed to see this picture.

“Don’t mind reverse” driving-style

QuickTime™ and a
JVT/AVC Coding decompressor
are needed to see this picture.

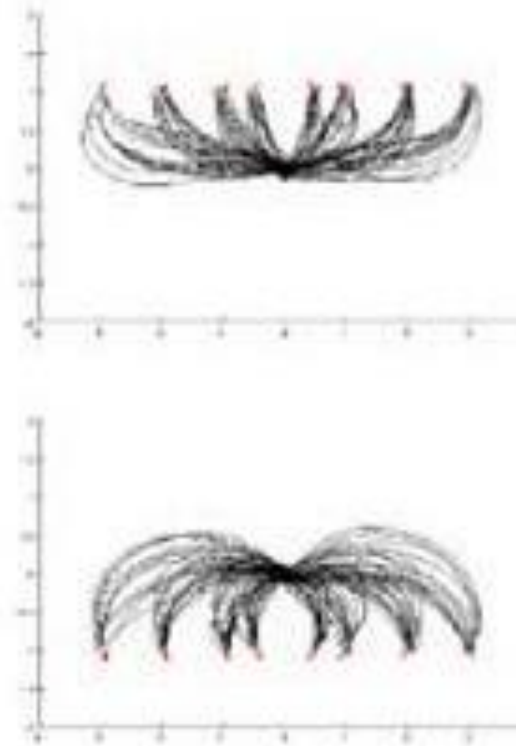
Human path planning

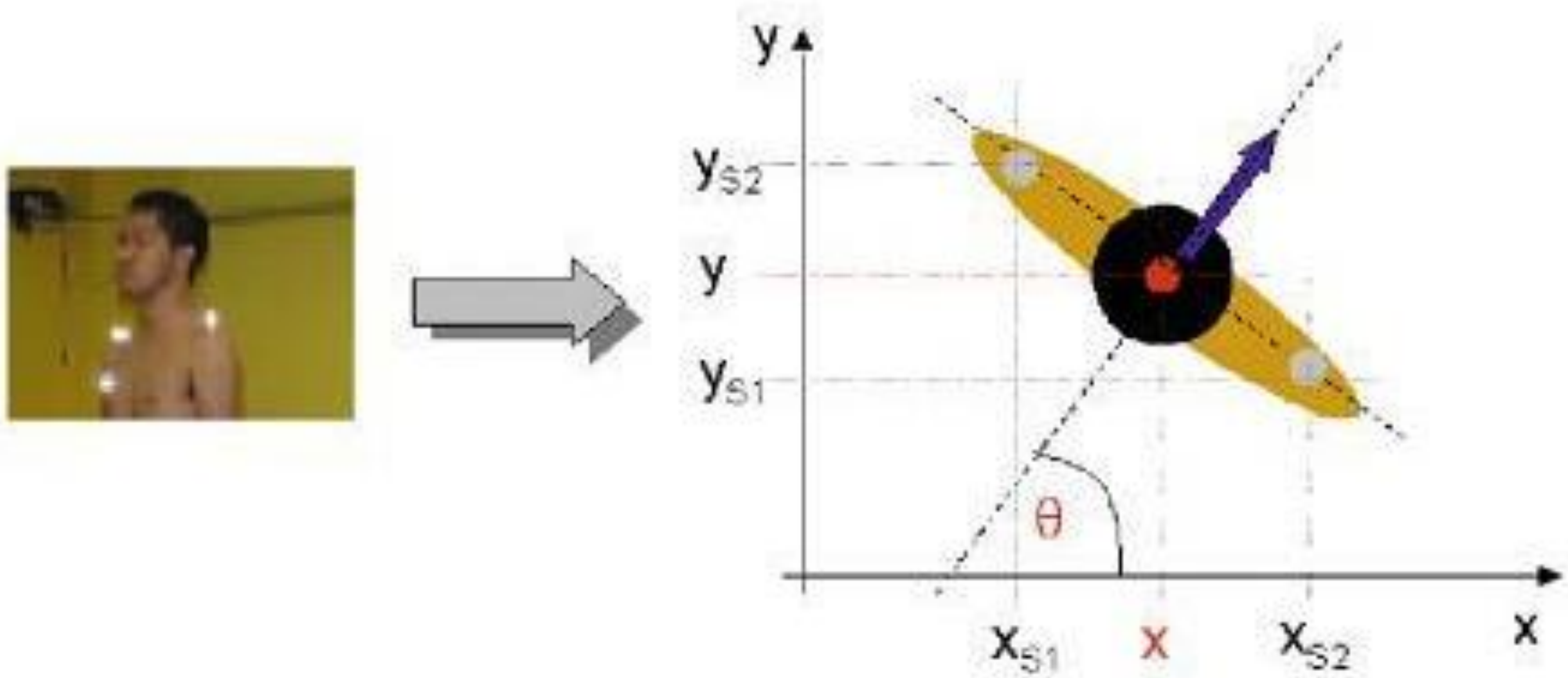
- Reward features:
  - Time to destination
  - (Forward acceleration)^2
  - (Sideways acceleration)^2
  - (Rotational acceleration)^2
  - Integral (angular error)^2



[Mombaur, Truong, Laumond, 2009]

Experimental Setup



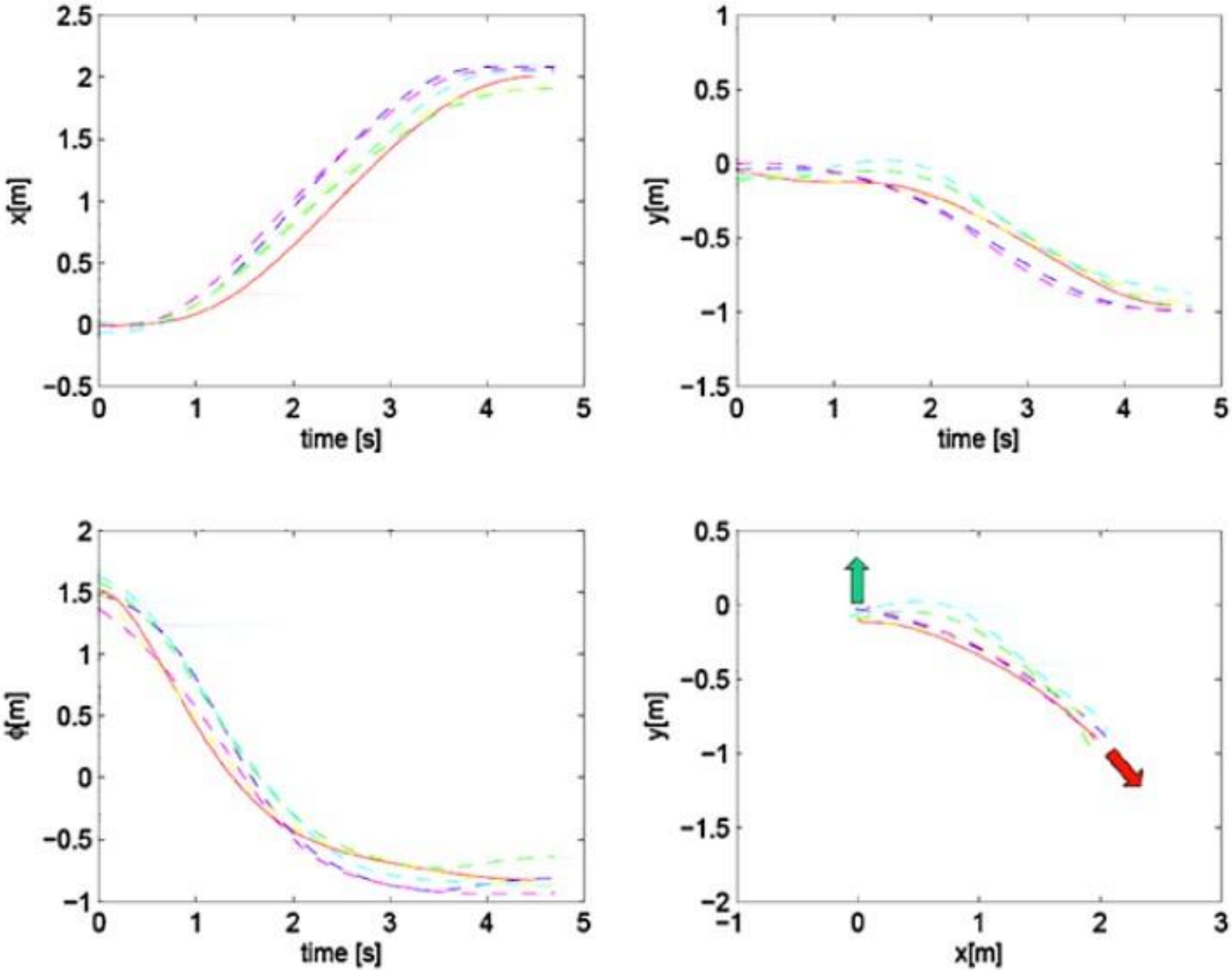


Human path planning

- Result:
  - Time to destination: 1
  - (Forward acceleration)^2 1.2
  - (Sideways acceleration)^2 1.7
  - (Rotational acceleration)^2 0.7
  - Integral (angular error)^2 5.2

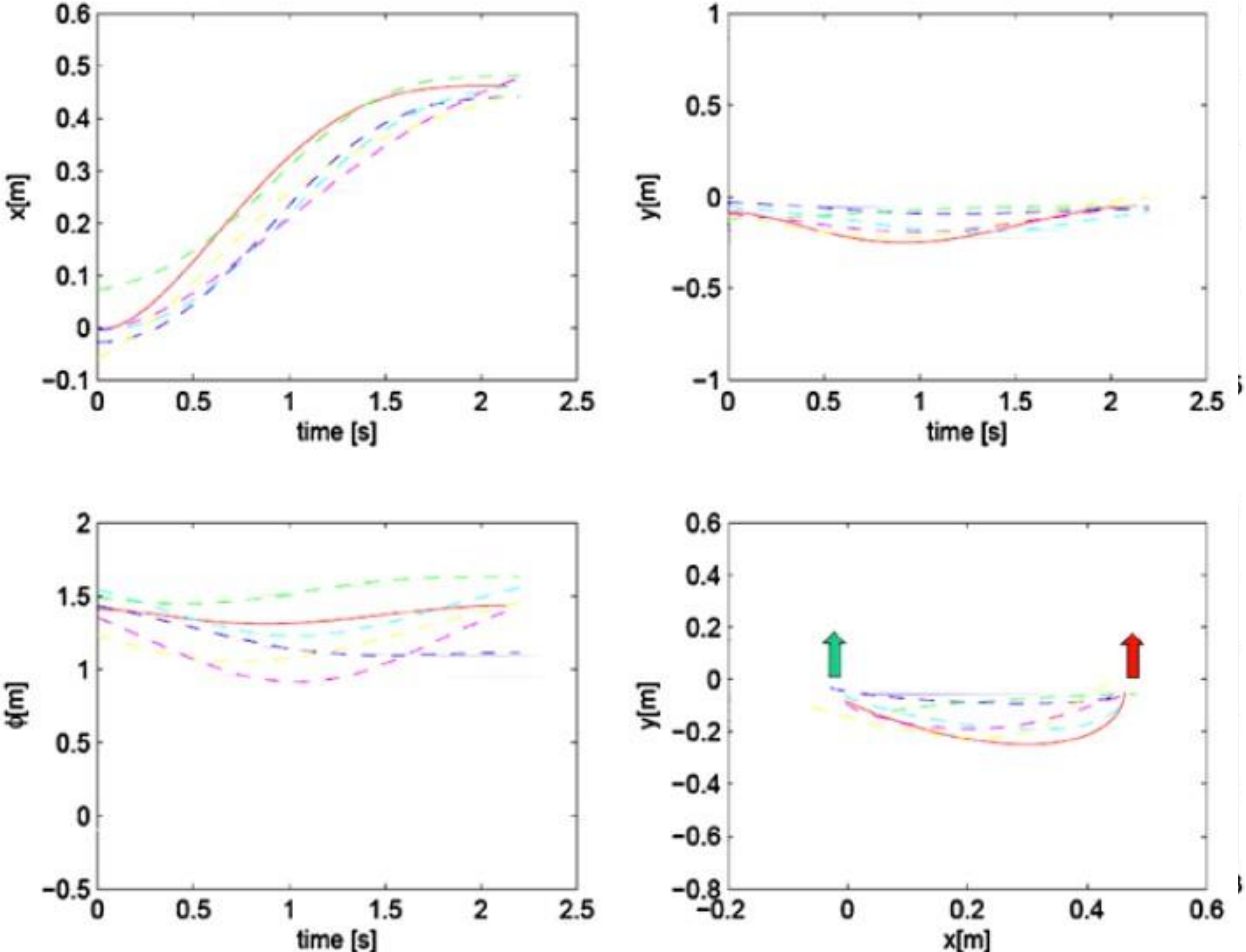
[Mombaur, Truong, Laumond, 2009]

Human path planning



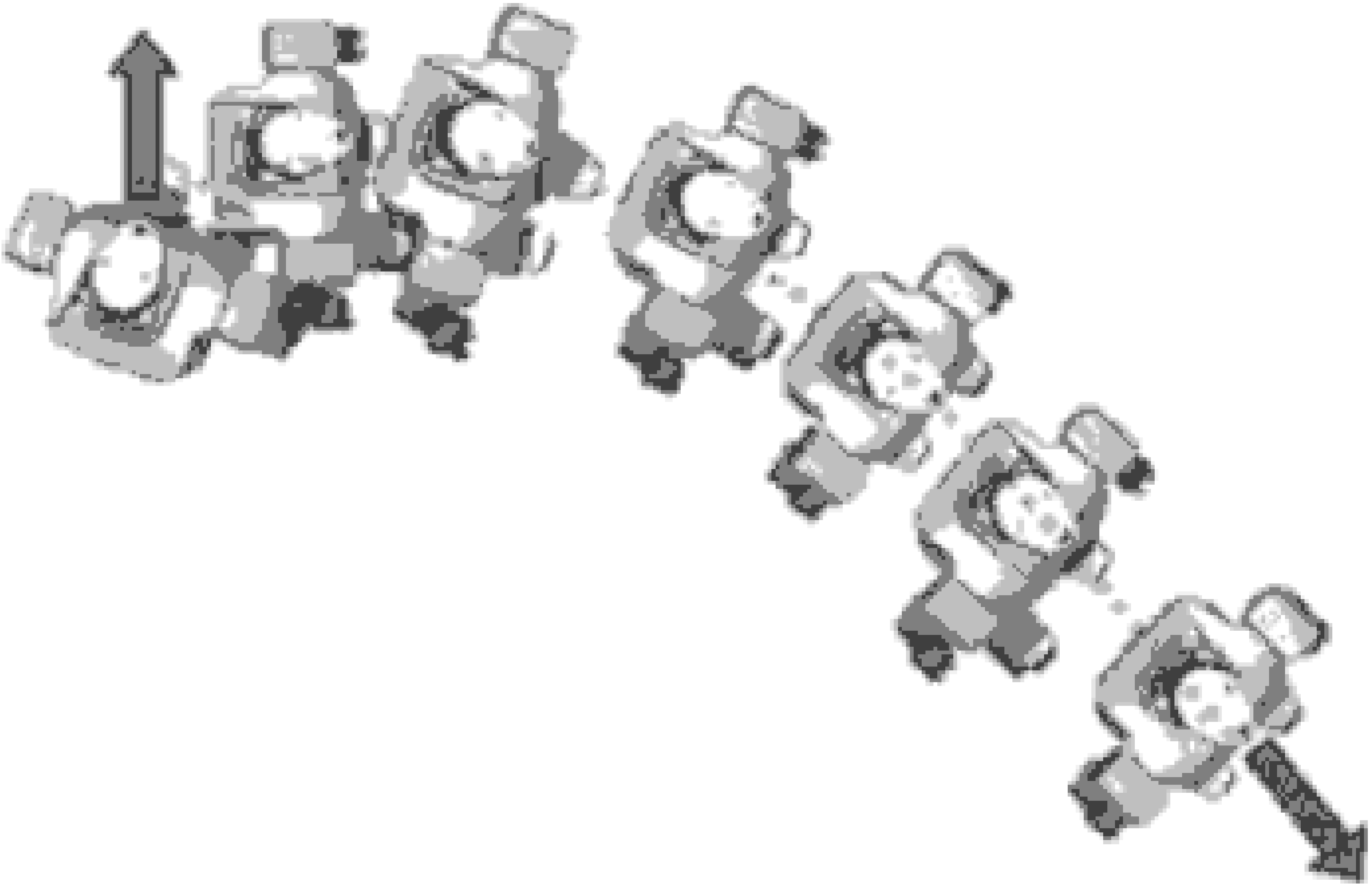
[Mombaur, Truong, Laumond, 2009]

Human path planning



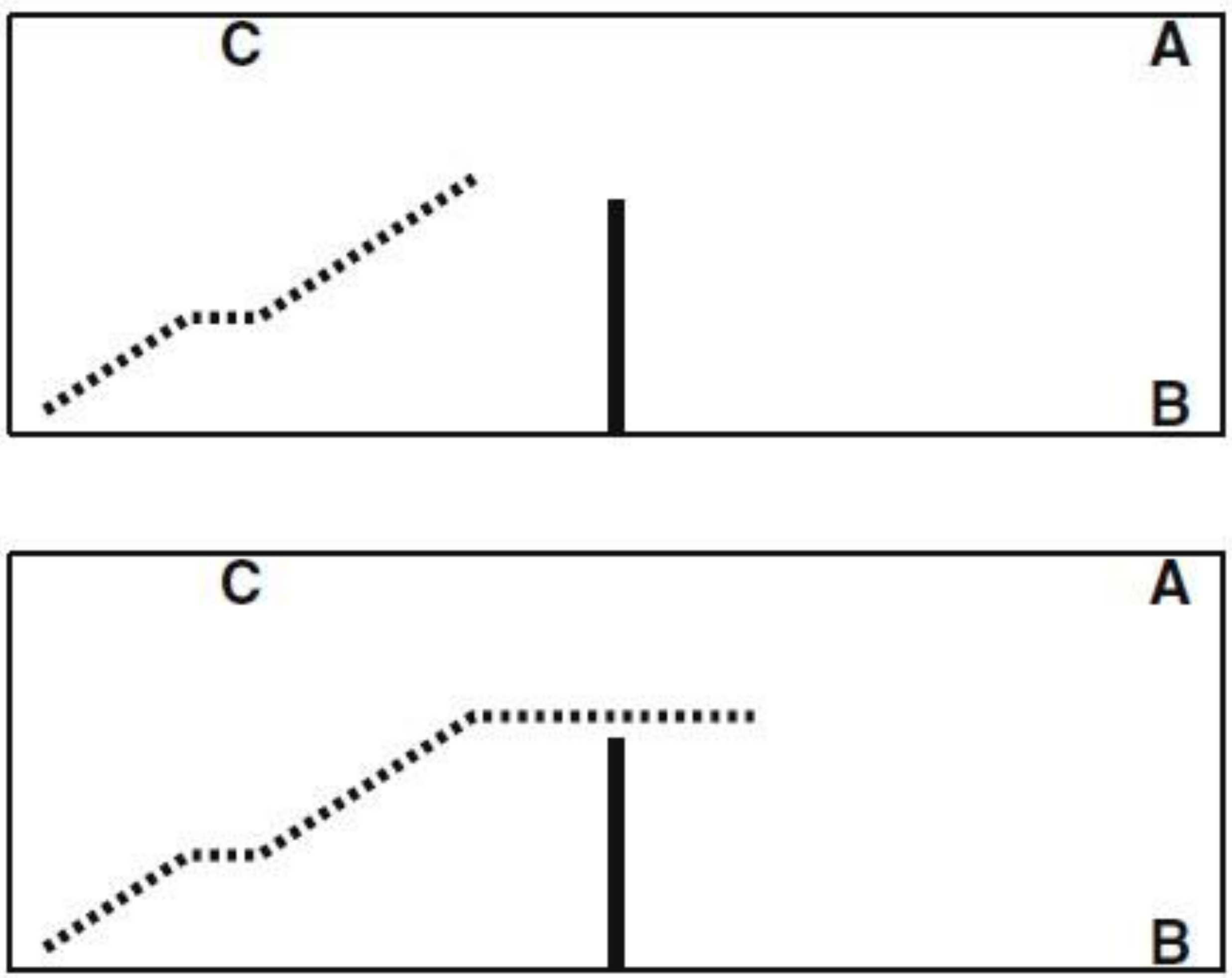
[Mombaur, Truong, Laumond, 2009]

Transfer to a Humanoid



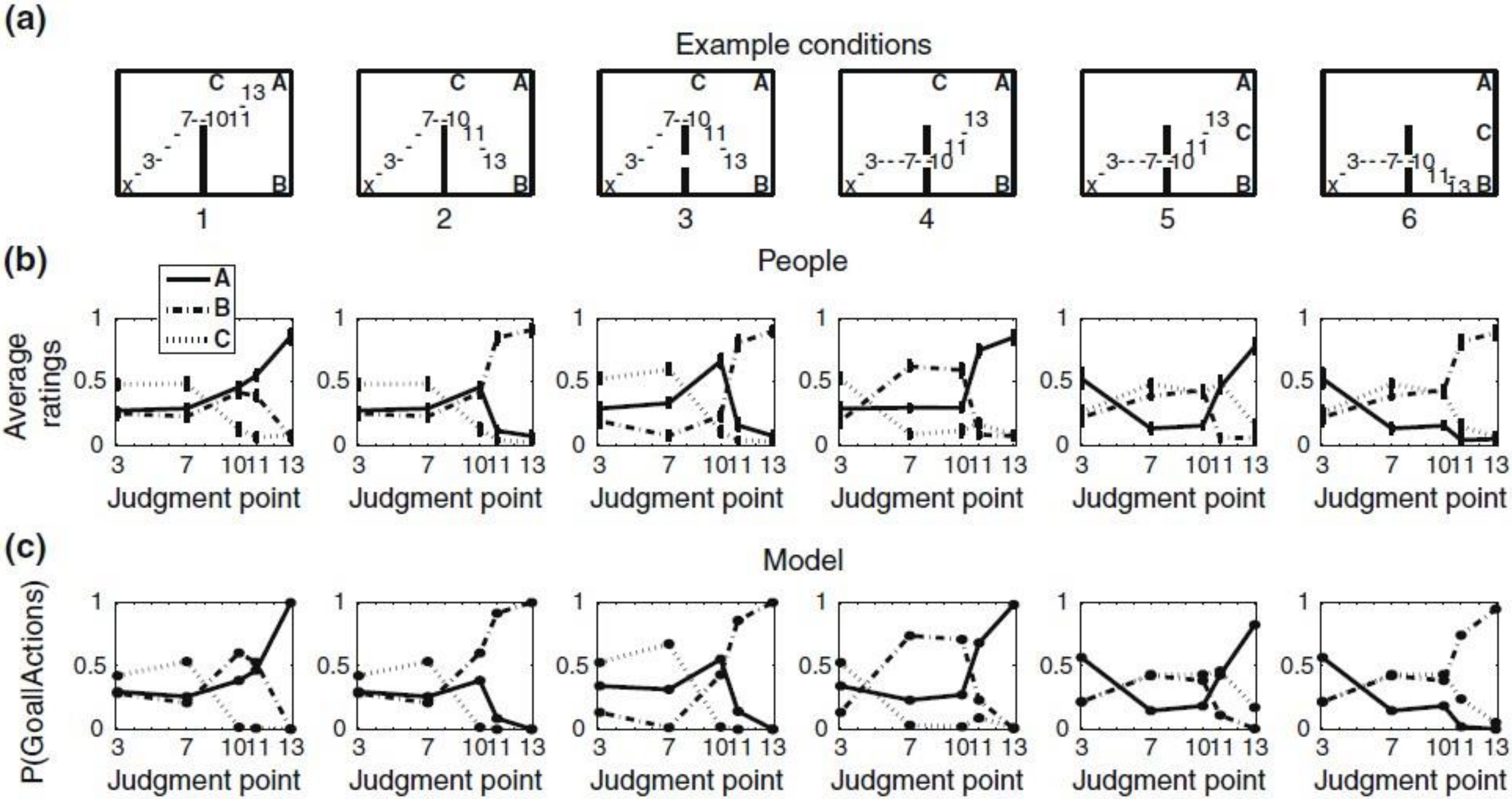
Goal inference

- Observe partial paths, predict goal. Goal could be either A, B, or C.
- + HMM-like extension: goal can change (with some probability over time).



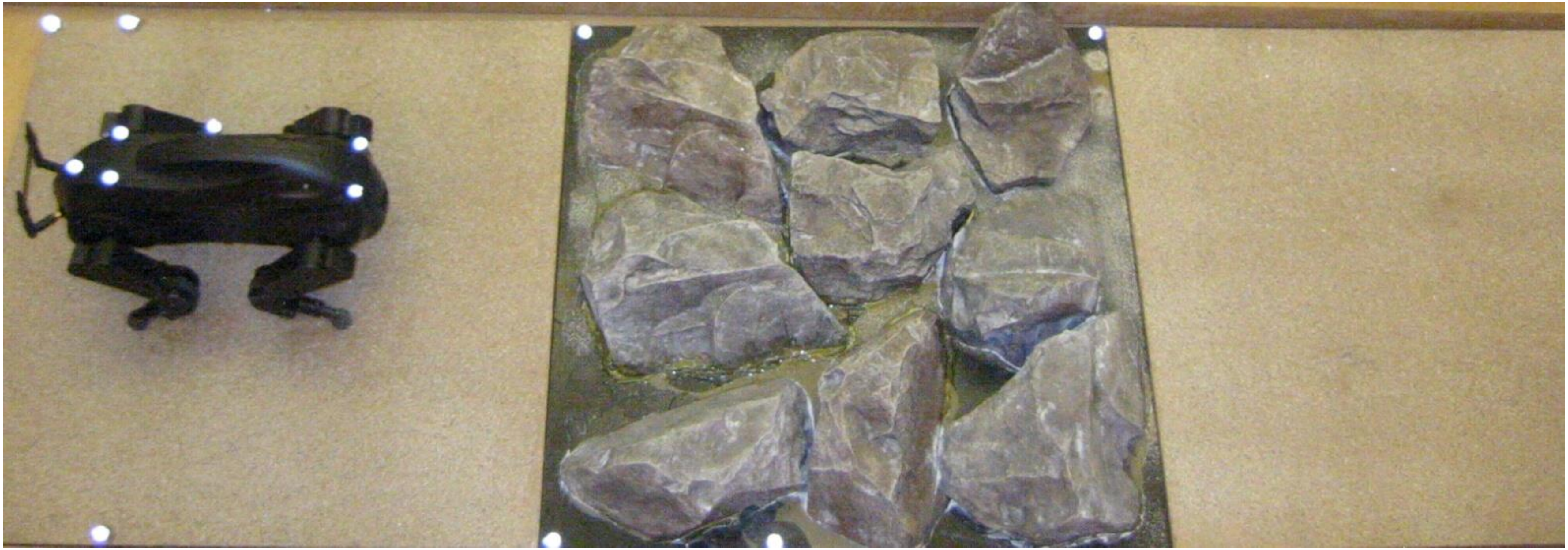
[Baker, Saxe, Tenenbaum, 2009]

Goal inference



[Baker, Saxe, Tenenbaum, 2009]

Quadruped

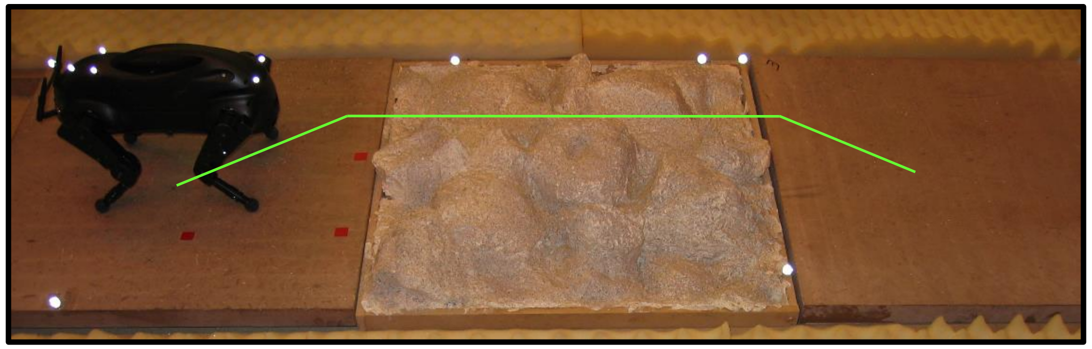


- Reward function trades off 25 features.

Hierarchical max margin [Kolter, Abbeel & Ng, 2008]

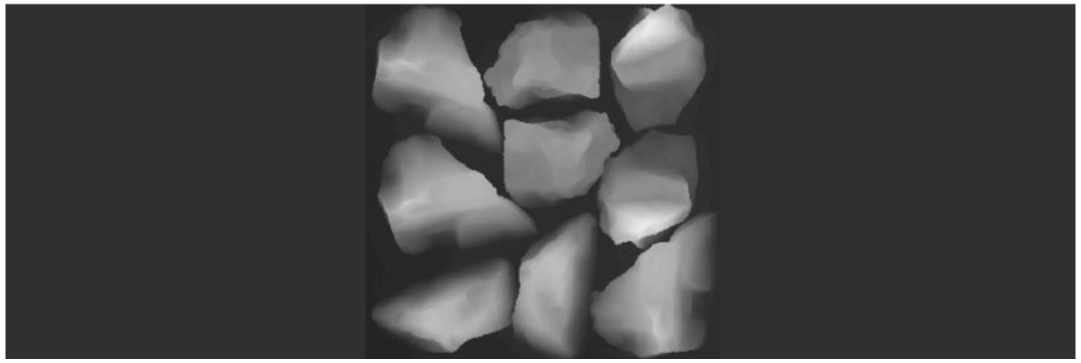
Experimental setup

- Demonstrate path across the “training terrain”



- Run the apprenticeship learning algorithm to find the reward function

- Receive “testing terrain”---height map.



- Find the optimal policy with respect to the learned reward function for crossing the testing terrain.

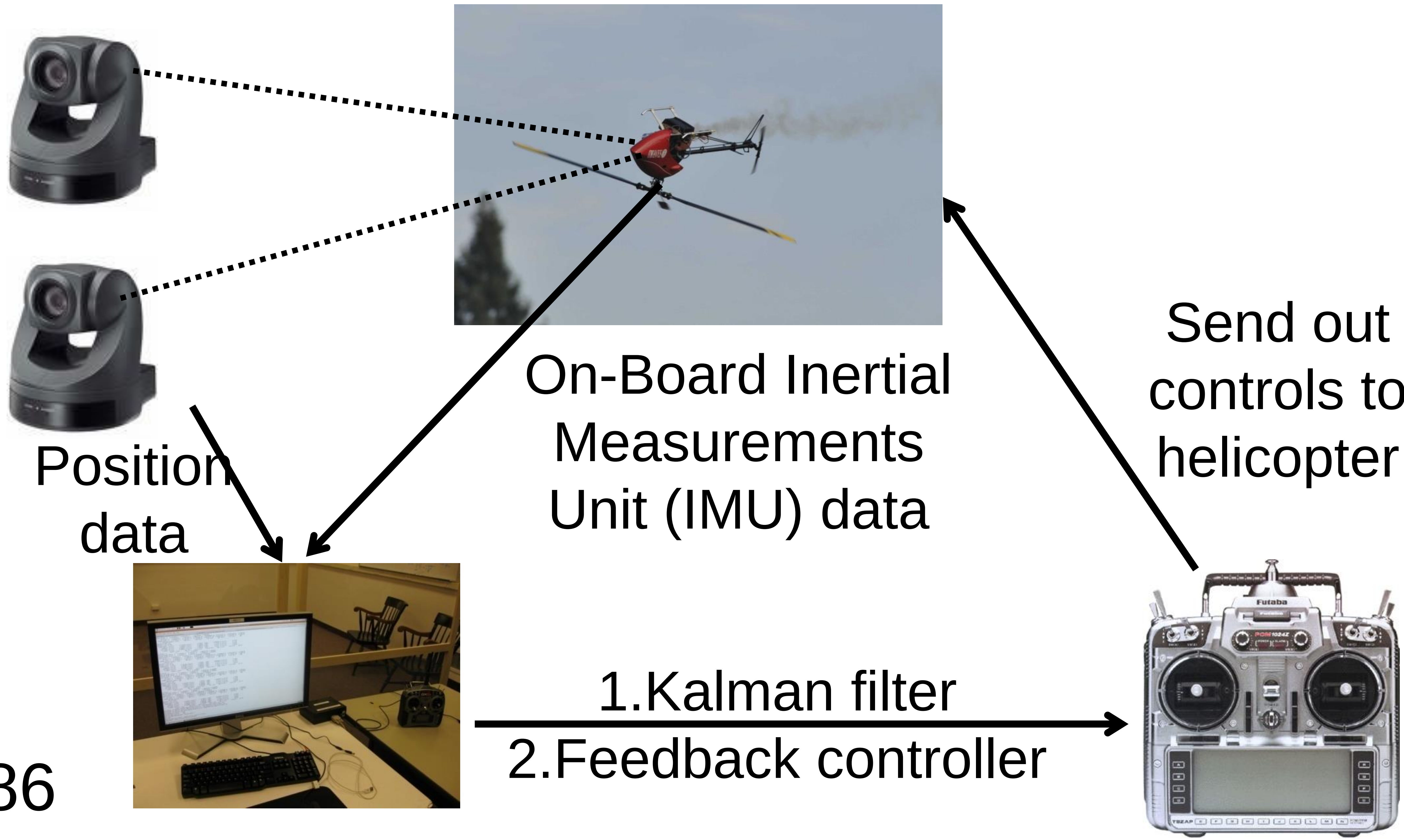
Little Dog: CMU Team

Ratliff + al, 2007

Remainder of lecture: extreme helicopter flight

- How does helicopter dynamics work
- Autonomous helicopter setup
- Application of inverse RL to autonomous helicopter flight

Autonomous helicopter setup

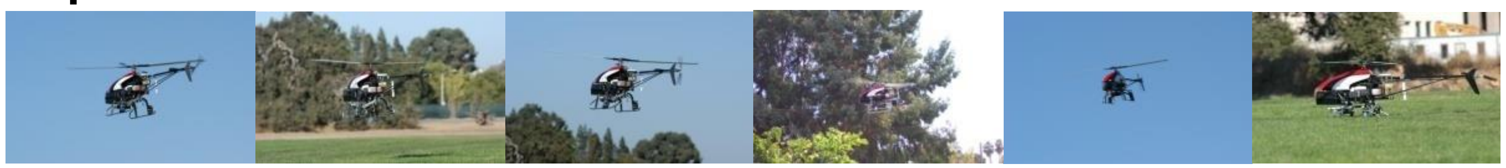


Helicopter dynamics

- 4 control inputs:
  - Main rotor collective pitch
  - Main rotor cyclic pitch (roll and pitch)
  - Tail rotor collective pitch

Experimental setup for the helicopter

1. Our expert pilot demonstrates the airshow several times.



2. Learn (by solving a joint optimization problem):

   - Reward function---trajectory.
   - Dynamics model---trajectory-specific local model.

3. Fly autonomously:

   - Inertial sensing + vision-based position sensing → (extended) Kalman filter
   - Receding horizon differential dynamic programming (DDP) feedback controller (20Hz)
   - **Learning to fly new aerobatics takes < 1 hour**

Results!

Summary

What you should know:

- Why is inverse RL useful / better than direct imitation learning?
- Algorithmic Challenges in IRL
- Different methods that use IRL, all are linear in features
- Why maximum margin?
- Why max. entropy?

Advertisement

Jan Peters &
Gerhard Neumann

TECHNISCHE
UNIVERSITÄT
DARMSTADT

Integrated Projektpraktikum

Lernende Roboter

⇕

Lernende Roboter: Integriertes Projekt Teil 1
Literature Review and Simulation Studies

Lernende Roboter: Integriertes Projekt Teil 2
Evaluation and Submission to a Conference

How does it fit in your course plan?

Lernende Roboter

⇕

Lernende Roboter: Integriertes Projekt Teil 1
Literature Review and Simulation Studies

Lernende Roboter: Integriertes Projekt Teil 2
Evaluation and Submission to a Conference

Starting at the Autonomous Systems Labs

"Taster course" as scientist

Among the most important questions ever:

continue the research road to a Ph.D. (=Dr.)?

The personal and professional advantages are enormous!

An exciting life:

follow *your* ideas & dreams...
actively acquire knowledge and refine it...
enjoy international conferences and visits with collaborators around the world...

However, it ain't for everybody!

Your Master's thesis will already decide on your chances!

- **Do you wanna figure out whether there is a researcher in YOU?**

Basic Idea: Be a researcher

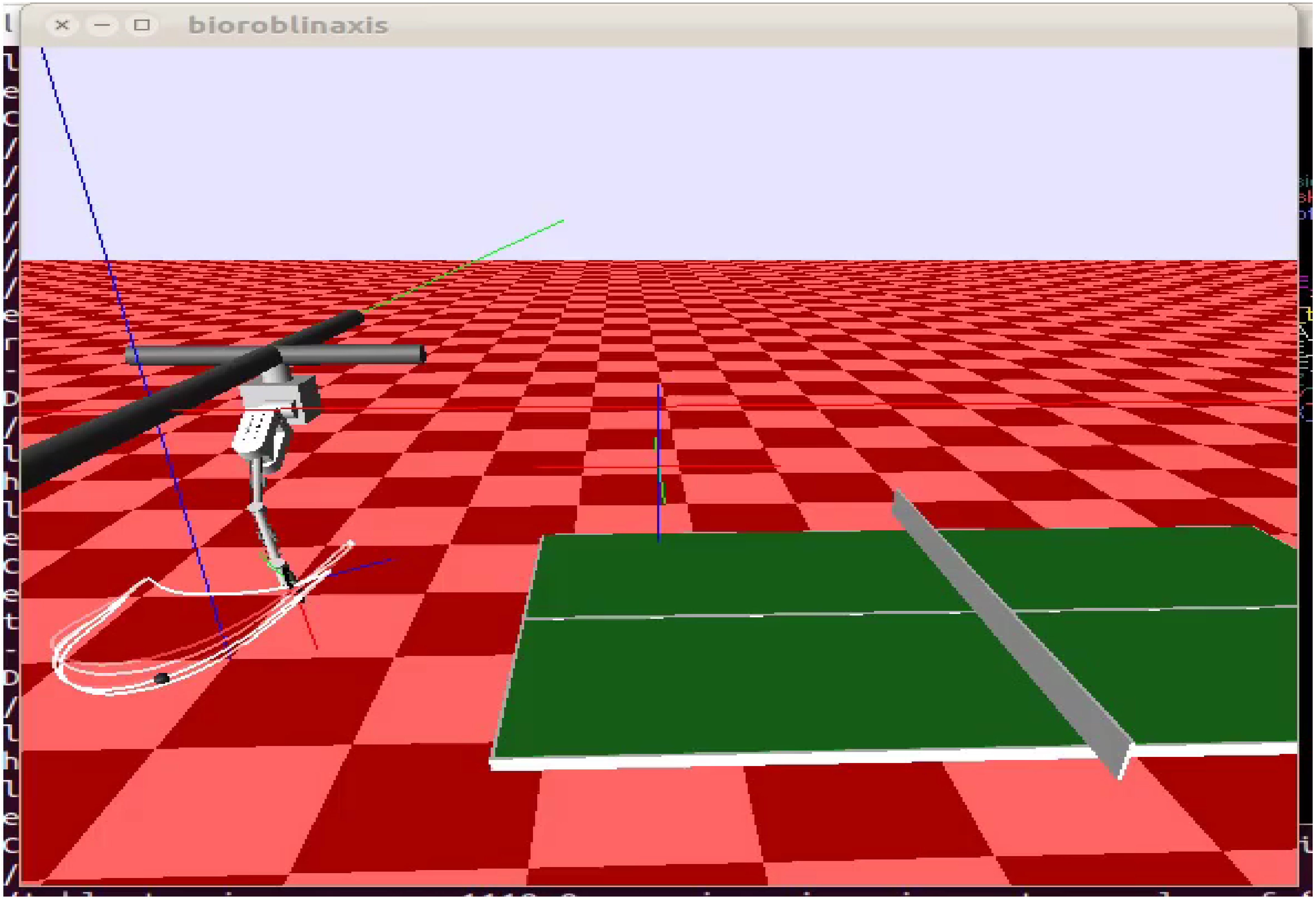
Mini-Class

- The lecture “Robot Learning” serves as background!
- We give you a few suggestions on platforms and algorithms.

Your idea becomes your project! This can only be fun!

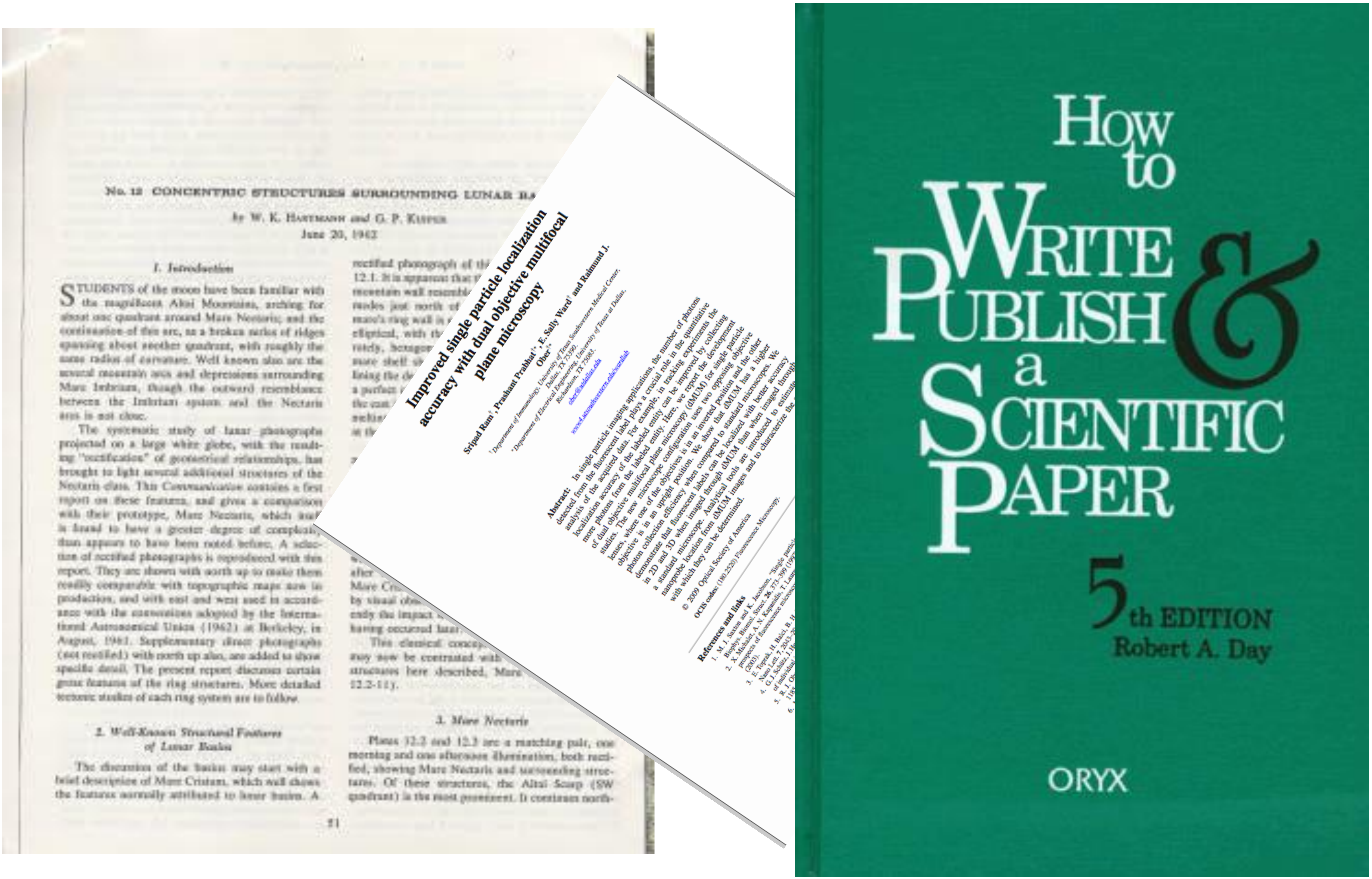
Your creativity is what will make it an amazing experience for both you & us!

Simulation



Simulation

Write a Scientific Paper



Do a mini-conference!

What we offer....

We offer you a glimpse how life as a researcher in robot learning is like!

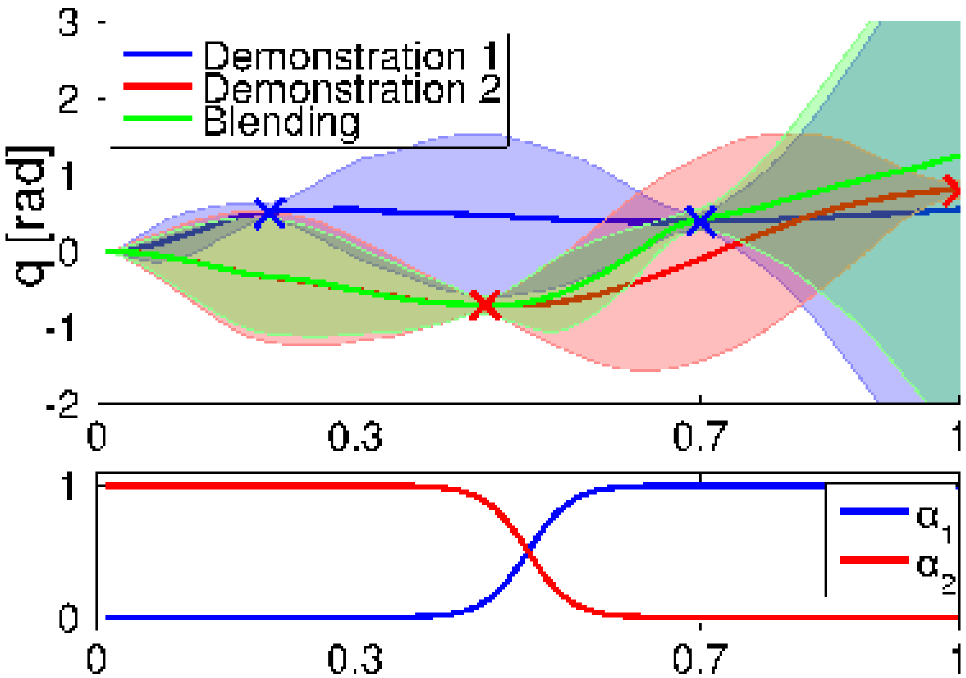
- Use the **knowledge from the robot learning** lecture right away!!!
- Decide what problem you are interested in and implement it in our simulator.
- Write a “**Scientific Paper**” as a team!
- Have a mini-conference at the semester’s end...
- Perfect start for your **Masters** or **Bachelors Theses**.

You are getting a taste how research life is like!

What we also offer....

Master and **Bachelor** thesis or also smaller projects

- For possible projects, see our homepage!
- Interested in your own topic? Just talk to us!



How does it fit in your course plan?

Related Classes:

- **Improve Foundations:** Robotik 1 (WS) + Robotik 2 (SoSe)
- **Useful Techniques:** Optimierung statischer und dynamischer Systeme
- **More (un-)supervised learning:**
  - Maschinelles Lernen: Statistische Methoden (SoSe),
  - Maschinelles Lernen: Statistische Methoden 2 (WS),
  - Maschinelles Lernen: Symbolische Ansätze (WS).
- **More Autonomous Systems:**
  - Intelligente Multi-Agent Systeme (SoSe) **(New Lecture!)**

Your way to the thesis…

Theses:

- Our class brings you right to **B.Sc. or M.Sc.** Thesis level (checkout our homepage)
- If you want to do your Ph.D. (=Dr) in Robot Learning, our classes plus all of the above are guaranteed to be optimal.
- Currently 19 Thesis are supervised by the Autonomous Systems Labs
- Many Master and Bachelor Theses end up in a **Publication!**

How does it fit in your course plan?

- **B.Sc. / M.Sc. Informatik:**
  - Computational Engineering (see Modulhandbuch), Not DKE
  - If you are strongly interested in machine learning you should:
    - Take ML: Statistical Methods for HCS credit
    - Take ML: Symbolische Methoden for DKE credit
    - Take RL for CE credit
- **M.Sc. in Autonome Systeme**