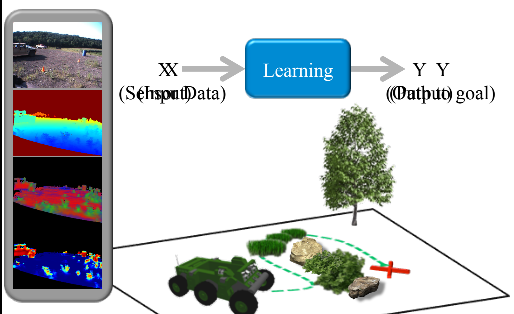
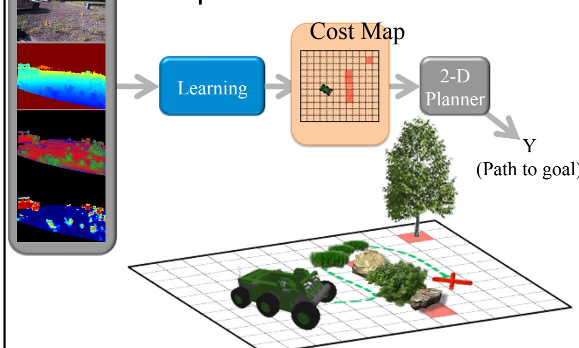
Inverse Optimal Control
(Inverse Reinforcement Learning)

Most slides by Drew Bagnell
Carnegie Mellon University
RI and ML
http://robotwhisperer.org



Optimal Control Solution



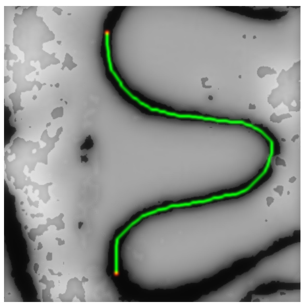
Mode 1: Training example

Mode 1: Training example

Mode 1: Learned behavior

Mode 1: Learned behavior

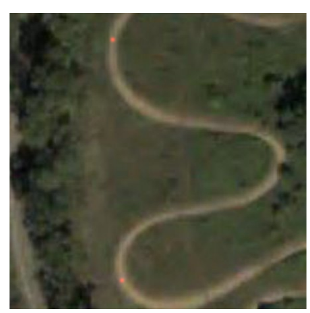
Mode 1: Learned cost map



Mode 2: Training example

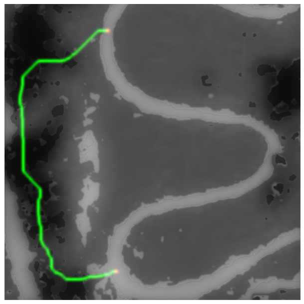
Mode 2: Training example

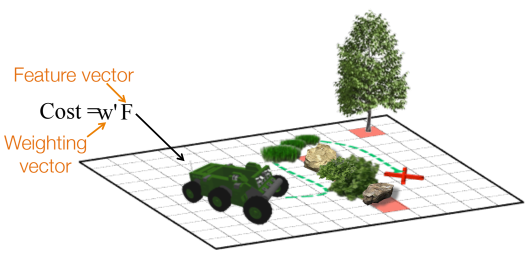
Mode 2: Learned behavior



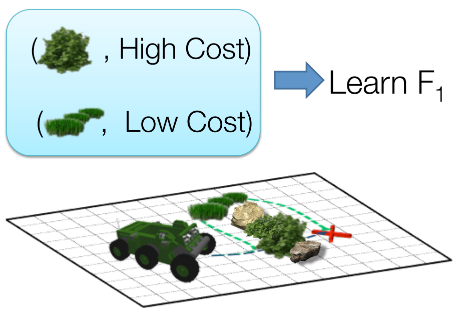
Mode 2: Learned behavior

Mode 2: Learned cost map

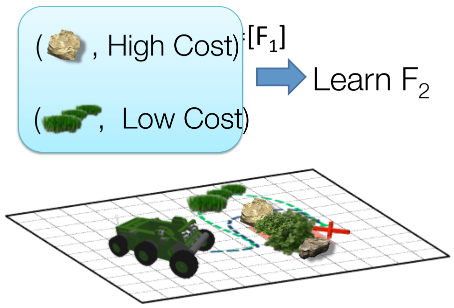




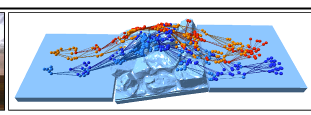
Ratliff, Bagnell, Zinkevich 2005
Ratliff, Bradley, Bagnell, Chestnutt, NIPS 2006
Silver, Bagnell, Stentz, RSS 2008



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



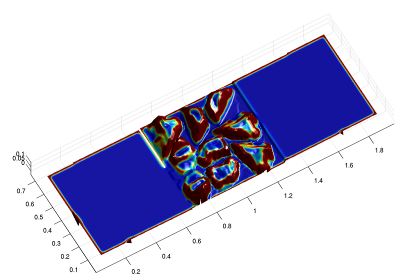
Ratliff, Bagnell, Zinkevich, ICML 2006
Ratliff, Bradley, Bagnell, Chestnutt, NIPS 2006
Silver, Bagnell, Stentz, RSS 2008



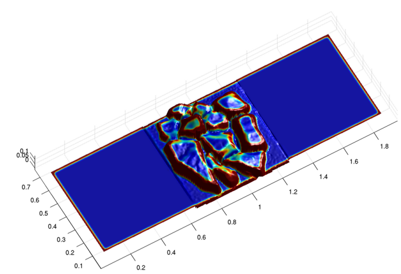
Ratliff, Bradley, Chesnutt, Bagnell 06

Zucker, Ratliff, Stolle, Chesnutt, Bagnell, Atkeson, Kuffner 09

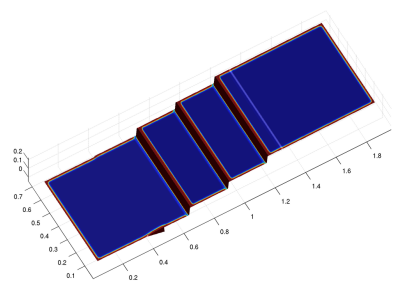
Learned Cost Function Examples



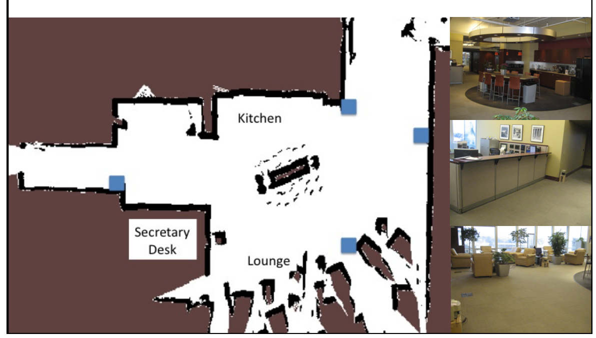
Learned Cost Function Examples



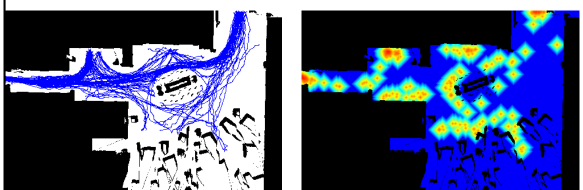
Learned Cost Function Examples



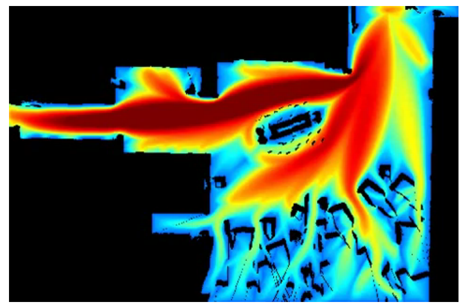
Pedestrian Trajectory Prediction



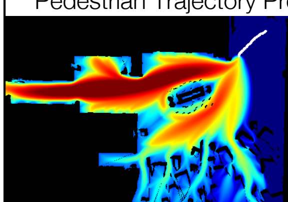
Pedestrian Trajectory Prediction



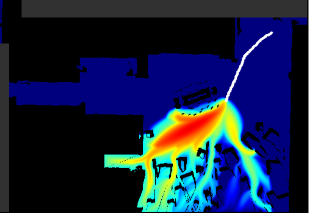
Pedestrian Trajectory Prediction



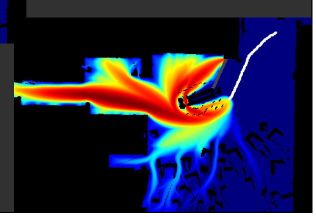
Pedestrian Trajectory Prediction



(Using O(1) Bayes' Rule for Goals)



Pedestrian Trajectory Prediction



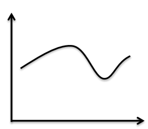
Staying out of People's Path

Car Prediction

Learning Manipulation Preferences

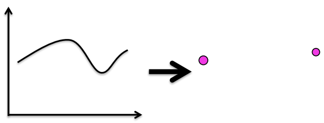
- ***Input:*** Human demonstrations of preferred behavior (e.g., moving a cup of water upright without spilling)
- ***Output:*** Learned cost function that results in trajectories satisfying user preferences

31



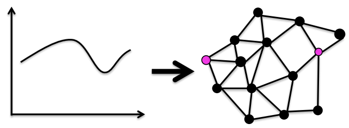
Demonstration(s)

32



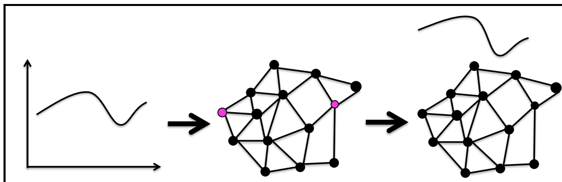
Demonstration(s) Graph

33



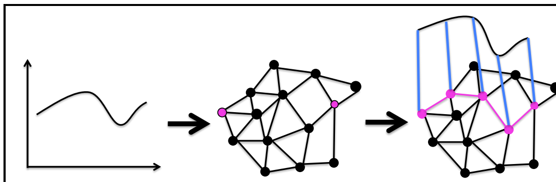
Demonstration(s) Graph

34



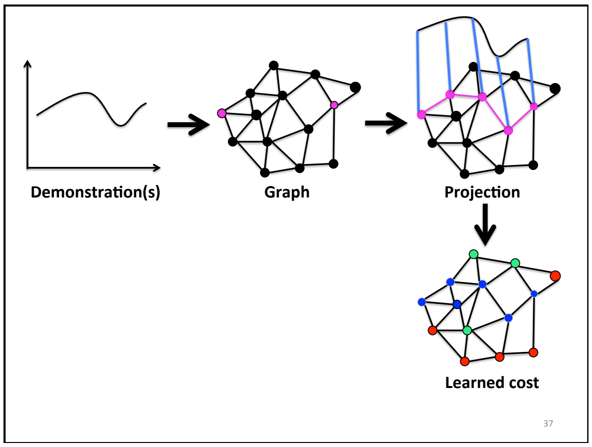
Demonstration(s) Graph Projection

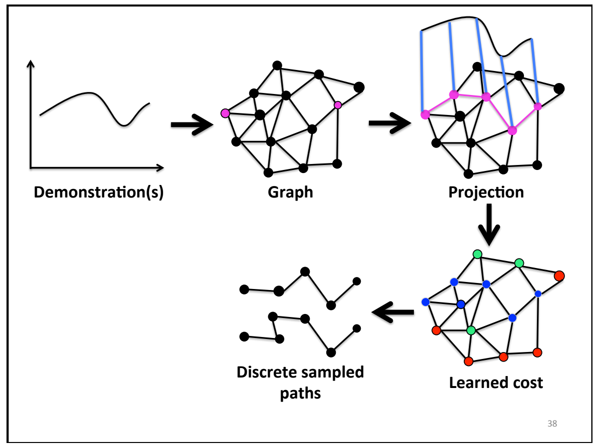
35

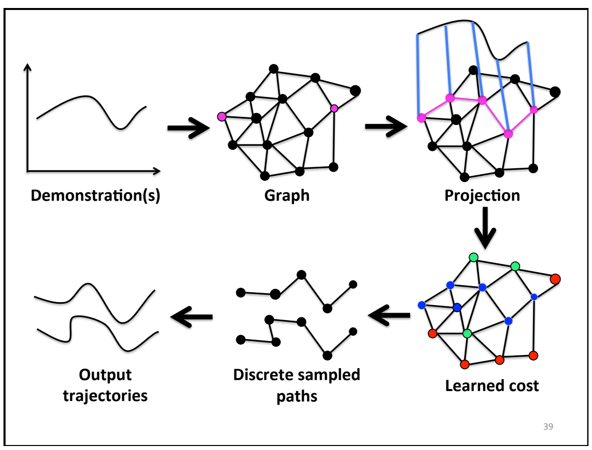


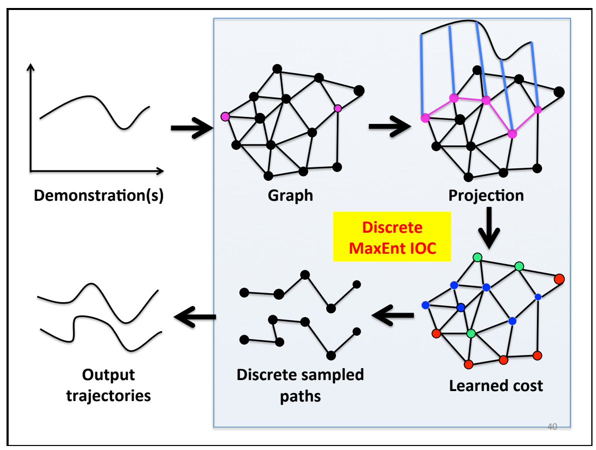
Demonstration(s) Graph Projection

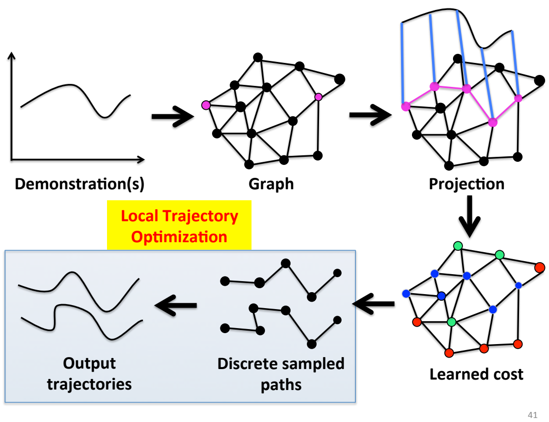
36


Demonstration(s)
Graph
Projection
Learned cost
37

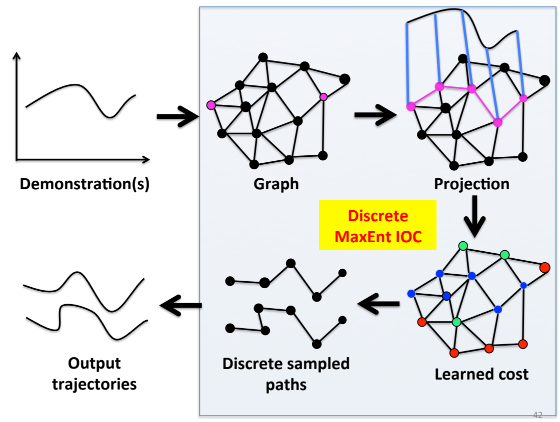

Demonstration(s)
Graph
Projection
Discrete sampled paths
Learned cost
38


Demonstration(s)
Graph
Projection
Output trajectories
Discrete sampled paths
Learned cost
39


Demonstration(s)
Graph
Projection
Discrete MaxEnt IOC
Output trajectories
Discrete sampled paths
Learned cost
40



41



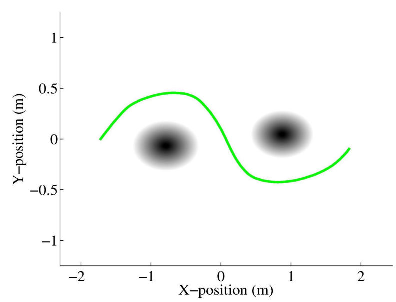
42

Discrete MaxEnt IOC

- ***Input:*** Expert demonstrations for a specific task, obstacle data, feature functions
- ***Output:*** Cost function (θ*), discrete path samples satisfying user preferences
- ***Steps:***
  1. Graph generation
  2. Projection
  3. Learning the cost function
  4. Sampling discrete paths from the graph

43

2D obstacle avoidance task

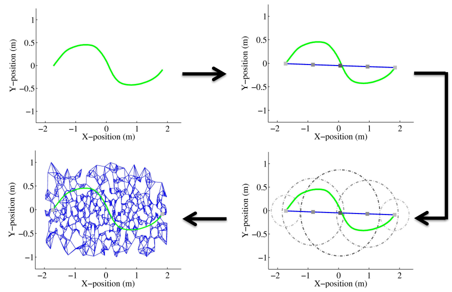


2D state: (x,y)

44

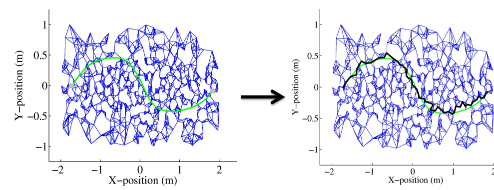
Graph generation

- ***Goal:*** Construct a graph in the robot's configuration space providing good coverage



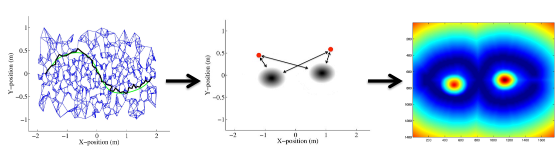
Projection

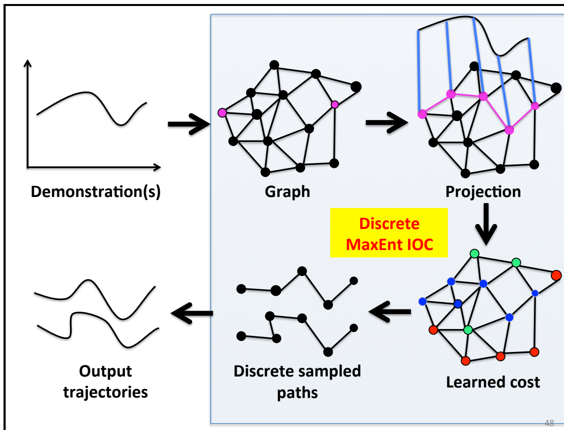
- ***Goal:*** Project the continuous demonstration onto the graph, resulting in a discrete graph path
- Use a **modified Dijkstra's** algorithm minimizing sum of:
  - Length of discrete path (Euclidean)
  - Distance to continuous demonstration

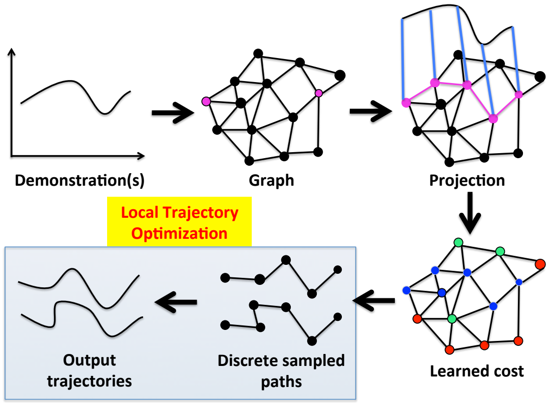


Learning the cost function

- ***Goal:*** Given projected demonstrations, learn the cost function
- Learn feature weights (θ^*) using **softened value iteration** on the discrete graph (MaxEnt IOC - Ziebart et al., 2008)
  - State dependent features (eg: Distance to obstacles)







Local Trajectory Optimization

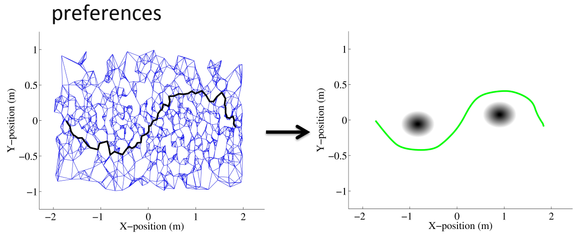
- **Sampled discrete graph paths** are **jerky** and cannot be executed on a real robot
- Naïve smoothing of the graph path can result in trajectories with high task-related cost
- We add a **smoothness regularizer** (squared velocity) to the **learned cost** function and optimize via local trajectory optimization (LTO)

$$\xi^* = \arg\min_{\xi} c[\xi]$$

$$c[\xi] = \underbrace{\frac{1}{2}\int_0^1 \| \frac{d}{dt}\xi(t) \|^2 \, dt}_{\text{Smoothness cost}} + \underbrace{\theta^* f[\xi]}_{\text{Learned cost}}$$

Local Trajectory Optimization (LTO)

- **Input:** Initial path (sampled graph path or smooth random path), learned cost function
- **Output:** Smoothed final trajectory satisfying human preferences



Experimental Results

Setup

- ***Binary*** state-dependent features (~95)
  - Histograms of distances to objects
  - Histograms of end-effector orientation
  - Object specific features (electronic vs non-electronic)
  - Approach direction w.r.t goal
- **Comparison**:
  - Human demonstrations
  - Obstacle avoidance planner (CHOMP)
  - Locally optimal IOC approach (similar to Max-Margin planning, Ratliff et. al., 2007)

53

Laptop task: Demonstration

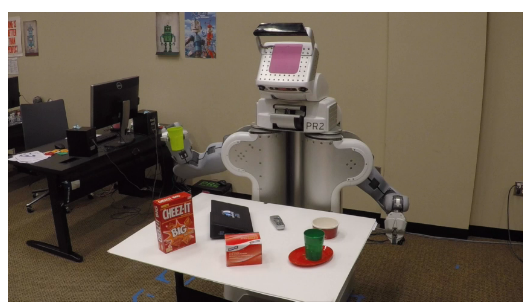
(Not part of training set)

54

Laptop task: LTO + Discrete graph path

55

Laptop task: LTO + Smooth random path



56

Statistics for Laptop task

Method	*% Points in collision*	*End-Effector normal deviation (deg.)*	*% Points above laptop*
Human Demonstration	2.7	7.4	2.1
Obstacle avoidance planner	12.9	18.2	17.3
Coarse, discrete graph sample	12.8	9.9	11.1
Local Trajectory Optimizer + Graph samples	4.0	5.3	1.2
Local Trajectory Optimizer + Random path	4.5	5.5	3.1

57