Machine Learning on Physical Robots

Prof. Peter Stone

Director, Learning Agents Research Group
Department of Computer Sciences
The University of Texas at Austin
Research Question

To what degree can autonomous intelligent agents learn in the presence of teammates and/or adversaries in real-time, dynamic domains?
Research Question

To what degree can autonomous intelligent agents learn in the presence of teammates and/or adversaries in real-time, dynamic domains?

- Autonomous agents
- Multiagent systems
- Machine learning
- Robotics
Autonomous Intelligent Agents

• They must **sense** their environment.
• They must **decide** what action to take ("think").
• They must **act** in their environment.
Autonomous Intelligent Agents

- They must sense their environment.
- They must decide what action to take (“think”).
- They must act in their environment.

Complete Intelligent Agents
Autonomous Intelligent Agents

- They must **sense** their environment.
- They must **decide** what action to take ("think").
- They must **act** in their environment.

**Complete Intelligent Agents**

- Interact with other agents (Multiagent systems)
Autonomous Intelligent Agents

- They must sense their environment.
- They must decide what action to take (“think”).
- They must act in their environment.

**Complete Intelligent Agents**

- Interact with other agents (Multiagent systems)
- Improve performance from experience (Learning agents)
Autonomous Intelligent Agents

- They must **sense** their environment.
- They must **decide** what action to take ("think").
- They must **act** in their environment.

**Complete Intelligent Agents**

- Interact with other agents  
  (Multiagent systems)
- Improve performance from experience  
  (Learning agents)

Autonomous Bidding, Cognitive Systems,  
Traffic management, **Robot Soccer**
RoboCup
Goal: By the year 2050, a team of humanoid robots that can beat the human World Cup champion team.
RoboCup

**Goal:** By the year 2050, a team of humanoid robots that can beat the human World Cup champion team.

- An international *research* initiative
RoboCup

**Goal:** By the year 2050, a team of humanoid robots that can beat the human World Cup champion team.

- An international *research* initiative

- Drives *research* in many areas:
  - Control algorithms; machine vision, sensing; localization;
  - Distributed computing; real-time systems;
  - Ad hoc networking; mechanical design;
RoboCup

**Goal:** By the year 2050, a team of humanoid robots that can beat the human World Cup champion team.

- An international *research* initiative

- Drives *research* in many areas:
  - Control algorithms; machine vision, sensing; localization;
  - Distributed computing; real-time systems;
  - Ad hoc networking; mechanical design;
  - Multiagent systems; machine learning; robotics
RoboCup

**Goal:** By the year 2050, a team of humanoid robots that can beat the human World Cup champion team.

- An international *research* initiative
- Drives *research* in many areas:
  - Control algorithms; machine vision, sensing; localization;
  - Distributed computing; real-time systems;
  - Ad hoc networking; mechanical design;
  - Multiagent systems; machine learning; robotics

**Several Different Leagues**
RoboCup Soccer

Small-sized League

Middle-sized League

Legged Robot League

Simulation League

Humanoid League

© 2003 The RoboCup Federation
Sony Aibo (ERS-210A, ERS-7)

- Electrostatic sensors
- Infrared range sensors
- 3 acceleration sensors (x, y, and z)
- Speaker and microphone
- Switch sensors
Sony Aibo (ERS-210A, ERS-7)

- **Color camera**
  - Resolution: 208 x 160
  - 30 frames per second

- **Wireless ethernet**
  - (802.11b)

- **On-board processor**
  - 576 MHz
  - 64 MB RAM
  - OS: Aperios + Open-R
  - Programming Language: C++
Sony Aibo (ERS-210A, ERS-7)

20 degrees of freedom

- head: 3 neck, 2 ears, 1 mouth
- 4 legs: 3 joints each
- tail: 2 DOF
Creating a team — Subtasks
Creating a team — Subtasks

- Vision
- Localization
- Walking
- **Ball manipulation** (kicking)
- Individual decision making
- Communication/coordination
Creating a team — Subtasks

- Vision
- Localization
- Walking
- Ball manipulation (kicking)
- Individual decision making
- Communication/coordination
Competitions

- Barely closed the loop by American Open *(May, ’03)*
Competitions

- Barely closed the loop by American Open (May, ’03)
- Improved significantly by Int’l RoboCup (July, ’03)
Competitions

- Barely closed the loop by American Open (May, ’03)
- Improved significantly by Int’l RoboCup (July, ’03)
- Won 3rd place at US Open (2004, 2005)
- Quarterfinalist at RoboCup (2004, 2005)
Competitions

- Barely closed the loop by American Open (May, ’03)
- Improved significantly by Int’l RoboCup (July, ’03)
- Won 3rd place at US Open (2004, 2005)
- Quarterfinalist at RoboCup (2004, 2005)

Highlights:
- Many saves: 1; 2; 3; 4;
- Lots of goals: CMU; Penn; Penn; Germany;
- A nice clear
- A counterattack goal
Post-competition: the research
Post-competition: the research

- Model-based joint control (Stronger, Stone)
- **Machine learning for fast walking** (Kohl, Stone)
- **Learning to acquire the ball** (Fidelman, Stone)
- **Learning sensor and action models** (Stronger, Stone)
- **Color constancy on mobile robots** (Sridharan, Stone)
- Robust particle filter localization (Sridharan, Kuhlmann, Stone)
- **Autonomous Color Learning** (Sridharan, Stone)
Policy Gradient RL to learn fast walk

Goal: Enable an Aibo to walk as fast as possible
Policy Gradient RL to learn fast walk

Goal: Enable an Aibo to walk as fast as possible

- Start with a parameterized walk
- Learn fastest possible parameters
Policy Gradient RL to learn fast walk

Goal: Enable an Aibo to walk as fast as possible

- Start with a parameterized walk
- Learn fastest possible parameters
- No simulator available:
  - Learn entirely on robots
  - Minimal human intervention
Walking Aibos

- Walks that “come with” Aibo are slow

- RoboCup soccer: 25+ Aibo teams internationally
  - Motivates faster walks
Walking Aibos

- Walks that “come with” Aibo are **slow**

- **RoboCup** soccer: *25+ Aibo teams* internationally
  - Motivates faster walks

<table>
<thead>
<tr>
<th>Hand-tuned gaits (2003)</th>
<th>Learned gaits</th>
</tr>
</thead>
<tbody>
<tr>
<td>230 mm/s</td>
<td>245</td>
</tr>
</tbody>
</table>
A Parameterized Walk

- Developed from scratch as part of UT Austin Villa 2003

- Trot gait with elliptical locus on each leg
Locus Parameters

- Ellipse length
- Ellipse height
- Position on $x$ axis
- Position on $y$ axis
- Body height
- Timing values

12 continuous parameters
Locus Parameters

- Ellipse length
- Ellipse height
- Position on $x$ axis
- Position on $y$ axis
- Body height
- Timing values

12 continuous parameters

- Hand tuning by April, '03: 140 mm/s
- Hand tuning by July, '03: 245 mm/s
Experimental Setup

- Policy $\pi = \{\theta_1, \ldots, \theta_{12}\}$, $V(\pi) =$ walk speed when using $\pi$
Experimental Setup

- Policy $\pi = \{\theta_1, \ldots, \theta_{12}\}$, $V(\pi) =$ walk speed when using $\pi$

- Training Scenario
  - Robots **time themselves** traversing fixed distance
  - Multiple traversals (3) per policy to account for **noise**
Experimental Setup

- Policy $\pi = \{\theta_1, \ldots, \theta_{12}\}$, $V(\pi) =$ walk speed when using $\pi$

- Training Scenario
  - Robots **time themselves** traversing fixed distance
  - Multiple traversals (3) per policy to account for **noise**
  - **Multiple robots** evaluate policies simultaneously
  - Off-board computer collects results, assigns policies
Experimental Setup

- Policy $\pi = \{\theta_1, \ldots, \theta_{12}\}$, $V(\pi) =$ walk speed when using $\pi$

- Training Scenario
  - Robots **time themselves** traversing fixed distance
  - Multiple traversals (3) per policy to account for **noise**
  - **Multiple robots** evaluate policies simultaneously
  - Off-board computer **collects results, assigns policies**

No human intervention except battery changes
Policy Gradient RL

• From $\pi$ want to move in direction of gradient of $V(\pi)$
Policy Gradient RL

- From $\pi$ want to move in direction of gradient of $V(\pi)$
  - Can’t compute $\frac{\partial V(\pi)}{\partial \theta_i}$ directly: estimate empirically
Policy Gradient RL

- From $\pi$ want to move in direction of gradient of $V(\pi)$
  - Can’t compute $\frac{\partial V(\pi)}{\partial \theta_i}$ directly: estimate empirically

- Evaluate neighboring policies to estimate gradient

- Each trial randomly varies every parameter
Policy Gradient RL

- From $\pi$ want to move in direction of gradient of $V(\pi)$
  - Can’t compute $\frac{\partial V(\pi)}{\partial \theta_i}$ directly: estimate empirically
- Evaluate neighboring policies to estimate gradient
- Each trial randomly varies every parameter
Experiments

- Started from **stable**, but fairly slow gait
- Used **3 robots** simultaneously
- Each iteration takes 45 traversals, $7\frac{1}{2}$ minutes
Experiments

- Started from **stable**, but fairly slow gait
- Used **3 robots** simultaneously
- Each iteration takes 45 traversals, $7\frac{1}{2}$ minutes

Before learning

After learning

- 24 iterations = **1080 field traversals**, $\approx$ **3 hours**
Results

Velocity of Learned Gait during Training

- Learned Gait (UT Austin Villa)
- Learned Gait (UNSW)
- Hand-tuned Gait (UNSW)
- Hand-tuned Gait (UT Austin Villa)
- Hand-tuned Gait (German Team)
Results

- Additional iterations didn’t help
- Spikes: evaluation noise? large step size?
# Learned Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Initial Value</th>
<th>$\epsilon$</th>
<th>Best Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Front ellipse:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(height)</td>
<td>4.2</td>
<td>0.35</td>
<td>4.081</td>
</tr>
<tr>
<td>(x offset)</td>
<td>2.8</td>
<td>0.35</td>
<td>0.574</td>
</tr>
<tr>
<td>(y offset)</td>
<td>4.9</td>
<td>0.35</td>
<td>5.152</td>
</tr>
<tr>
<td>Rear ellipse:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(height)</td>
<td>5.6</td>
<td>0.35</td>
<td>6.02</td>
</tr>
<tr>
<td>(x offset)</td>
<td>0.0</td>
<td>0.35</td>
<td>0.217</td>
</tr>
<tr>
<td>(y offset)</td>
<td>-2.8</td>
<td>0.35</td>
<td>-2.982</td>
</tr>
<tr>
<td>Ellipse length</td>
<td>4.893</td>
<td>0.35</td>
<td>5.285</td>
</tr>
<tr>
<td>Ellipse skew multiplier</td>
<td>0.035</td>
<td>0.175</td>
<td>0.049</td>
</tr>
<tr>
<td>Front height</td>
<td>7.7</td>
<td>0.35</td>
<td>7.483</td>
</tr>
<tr>
<td>Rear height</td>
<td>11.2</td>
<td>0.35</td>
<td>10.843</td>
</tr>
<tr>
<td>Time to move</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>through locus</td>
<td>0.704</td>
<td>0.016</td>
<td>0.679</td>
</tr>
<tr>
<td>Time on ground</td>
<td>0.5</td>
<td>0.05</td>
<td>0.430</td>
</tr>
</tbody>
</table>
Algorithmic Comparison, Robot Port

Before learning

After learning
Summary

- Used policy gradient RL to learn fastest Aibo walk
- All learning done on real robots
- No human intervention (except battery changes)
Outline

- Machine learning for fast walking (Kohl, Stone)
- Learning to acquire the ball (Fidelman, Stone)
- Learning sensor and action models (Stronger, Stone)
- Color constancy on mobile robots (Sridharan, Stone)
- Autonomous Color Learning (Sridharan, Stone)
Grasping the Ball

- **Three stages:** walk to ball; slow down; lower chin
- Head proprioception, IR chest sensor $\rightarrow$ ball distance
- Movement specified by 4 parameters
Grasping the Ball

- **Three stages:** walk to ball; slow down; lower chin

- Head proprioception, IR chest sensor $\rightarrow$ ball distance

- Movement specified by **4 parameters**

Brittle!
Parameterization

- **slowdown_dist**: when to slow down
- **slowdown_factor**: how much to slow down
- **capture_angle**: when to stop turning
- **capture_dist**: when to put down head
Learning the Chin Pinch

- **Binary, noisy** reinforcement signal: multiple trials
- **Robot evaluates self:** no human intervention
Results

• Evaluation of policy gradient, hill climbing, amoeba
## What it learned

<table>
<thead>
<tr>
<th>Policy</th>
<th>slowdown dist</th>
<th>slowdown factor</th>
<th>capture angle</th>
<th>capture dist</th>
<th>Success rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial</td>
<td>200mm</td>
<td>0.7</td>
<td>15.0°</td>
<td>110mm</td>
<td>36%</td>
</tr>
<tr>
<td>Policy gradient</td>
<td>125mm</td>
<td>1</td>
<td>17.4°</td>
<td>152mm</td>
<td>64%</td>
</tr>
<tr>
<td>Amoeba</td>
<td>208mm</td>
<td>1</td>
<td>33.4°</td>
<td>162mm</td>
<td>69%</td>
</tr>
<tr>
<td>Hill climbing</td>
<td>240mm</td>
<td>1</td>
<td>35.0°</td>
<td>170mm</td>
<td>66%</td>
</tr>
</tbody>
</table>
Instance of Layered Learning

- For domains too complex for tractably mapping state features $S \rightarrow$ outputs $O$
- Hierarchical subtask decomposition given: $\{L_1, L_2, \ldots, L_n\}$
- Machine learning: exploit data to train, adapt
- Learning in one layer feeds into next layer
Outline

• Machine learning for fast walking (Kohl, Stone)

• Learning to acquire the ball (Fidelman, Stone)

• Learning sensor and action models (Stronger, Stone)

• Color constancy on mobile robots (Sridharan, Stone)

• Autonomous Color Learning (Sridharan, Stone)
Learned Action/Sensor Models

• Mobile robots rely on **models of their actions and sensors**
  – Typically tuned **manually**: Time-consuming
Learned Action/Sensor Models

- Mobile robots rely on models of their actions and sensors
  - Typically tuned manually: Time-consuming

- Autonomous Sensor and Actuator Model Induction (ASAMI)
Learned Action/Sensor Models

- Mobile robots rely on **models of their actions and sensors**
  - Typically tuned **manually**: Time-consuming

- **Autonomous Sensor and Actuator Model Induction** (ASAMI)

- ASAMI is **autonomous**: no external feedback
  - Developmental robotics
Learned Action/Sensor Models

- Mobile robots rely on models of their actions and sensors
  - Typically tuned manually: Time-consuming

- Autonomous Sensor and Actuator Model Induction (ASAMI)

- ASAMI is autonomous: no external feedback
  - Developmental robotics

- Technique is implemented and tested in:
  - One-dimensional scenario: Sony Aibo ERS-7
  - Aibo in two-dimensional area
  - Second robotic platform: an autonomous car
Mobile robots rely on models of their actions and sensors.
Mobile robots rely on models of their actions and sensors.
Mobile robots rely on models of their actions and sensors.
General Methodology

- Action model, sensor model, world state unknown:

  Agent
  - Car Position
  - Car Velocity
  - Range Finder Readings
  - Camera Image

  Sensor Model
  - World State
    - Control Policy
    - Sensations
      - Throttle Position
      - Brake Position
      - Steering Position
    - Action Model
      - Action
      - Car Position
      - Car Velocity
        - Observations
          - Range Finder Readings
          - Camera Image

Peter Stone
General Methodology

- Given the robot’s actions and observations:

  ![Diagram showing Action Model and Sensor Model with World State Estimate]

  - Action Model
  - Sensor Model
  - World State Estimate
General Methodology

Given the robot's actions and observations:

- Action Model
- Sensor Model

Localization

World State Estimate
General Methodology

- Given the robot’s actions and observations:
General Methodology

- Given the robot’s actions and observations:

  **Inaccurate**
  
  - Action Model
  
  **Inaccurate**
  
  - Sensor Model

**World State Estimate**
General Methodology

- Given the robot’s actions and observations:

World State Estimate

Inaccurate Action Model

Inaccurate Sensor Model
General Methodology

• Given the robot’s actions and observations:

Accurate Action Model

Accurate Sensor Model

World State Estimate
The Task

- **Sensor model**: beacon height in image $\rightarrow$ distance
  - Mapping derived from camera specs not accurate
The Task

- **Sensor model**: beacon height in image $\mapsto$ distance
  - Mapping derived from camera specs not accurate

- **Action model**: parametrized walking, $W(x) \mapsto$ velocity
  - $x \in [-300, 300]$ is attempted velocity
  - Not accurate due to friction, joint behavior
Experimental Setup

- Aibo alternates walking forwards and backwards
  - Forwards: random action in $[0, 300]$
  - Backward phase: random action in $[-300, 0]$
  - Switch based on beacon size in image
Experimental Setup

- Aibo alternates walking forwards and backwards
  - Forwards: random action in $[0, 300]$  
  - Backward phase: random action in $[-300, 0]$  
  - Switch based on beacon size in image

- Aibo keeps self pointed at beacon
Learning Action and Sensor Models

- Both models provide info about the robot’s location

- **Sensor model**: observation $obs_k \mapsto$ location:
  
  $x_s(t_k) = S(obs_k)$
Both models provide info about the robot’s location

- **Sensor model**: observation $o_{bs_k}$ $\mapsto$ location:
  $$x_s(t_k) = S(o_{bs_k})$$

- **Action model**: action command $C(t)$ $\mapsto$ velocity:
  $$x_a(t) = x(0) + \int_0^t A(C(s)) \, ds$$
Learning Action and Sensor Models

- Both models provide info about the robot’s location

- **Sensor model**: observation $o_{bs_k} \mapsto$ location:
  \[ x_s(t_k) = S(o_{bs_k}) \]

- **Action model**: action command $C(t) \mapsto$ velocity:
  \[ x_a(t) = x(0) + \int_0^t A(C(s)) \, ds \]

- **Goal**: learn arbitrary continuous functions, $A$ and $S'$
Learning Action and Sensor Models

• Both models provide info about the robot’s location.

• **Sensor model**: observation $\text{obs}_k$ $\mapsto$ location:
  \[ x_s(t_k) = S(\text{obs}_k) \]

• **Action model**: action command $C(t)$ $\mapsto$ velocity:
  \[ x_a(t) = x(0) + \int_0^t A(C(s)) \, ds \]

• **Goal**: learn arbitrary continuous functions, $A$ and $S'$
  – Use polynomial regression as function approximator.
Learning Action and Sensor Models

- Both models provide info about the robot’s location

- **Sensor model:** observation $obs_k \mapsto$ location:
  \[ x_s(t_k) = S(obs_k) \]

- **Action model:** action command $C(t) \mapsto$ velocity:
  \[ x_a(t) = x(0) + \int_0^t A(C(s)) \, ds \]

- **Goal:** learn arbitrary continuous functions, $A$ and $S'$
  - Use polynomial regression as function approximator
  - Models learned in arbitrary units
Learning a Sensor Model

- Assume accurate action model
- Consider ordered pairs $(obs_k, x_a(t_k))$
- Fit polynomial to data
Learning a Sensor Model

- Assume accurate action model
- Consider ordered pairs \((obs_k, x_a(t_k))\)
- Fit polynomial to data
Learning an Action Model

- Assume accurate sensor model
- Plot $x_s(t)$ against time
Learning an Action Model

- Assume accurate sensor model
- Plot $x_s(t)$ against time

![Graph showing data points plotted against time](image)
Learning an Action Model

- Assume accurate sensor model is accurate
- Plot $x_s(t)$ against time
Learning an Action Model (cont.)

- Compute action model that minimizes the error
- Problem equivalent to another multivariate regression
Learning Both Simultaneously

- Both models improve via **bootstrapping**
  - Maintain two notions of location, $x_s(t)$ and $x_a(t)$
  - Each used to fit the other model
Learning Both Simultaneously

- Both models improve via **bootstrapping**
  - Maintain two notions of location, $x_s(t)$ and $x_a(t)$
  - Each used to fit the other model

- Use **weighted regression**
  - $w_i = \gamma^{n-i}, \gamma < 1$
  - Can still be computed incrementally
Learning Both Simultaneously

- Both models improve via **bootstrapping**
  - Maintain two notions of location, $x_s(t)$ and $x_a(t)$
  - Each used to fit the other model

- Use **weighted regression**
  - $w_i = \gamma^{n-i}, \gamma < 1$
  - Can still be computed incrementally

- Ramping up
Learning Both Simultaneously

- Over 2.5 min., $x_s(t)$ and $x_a(t)$ come into strong agreement
Experimental Results

- Run ASAMI for pre-set amount of time (2.5 minutes)
- Measure actual models with stopwatch and ruler
Experimental Results

- Run ASAMI for pre-set amount of time (2.5 minutes)
- Measure actual models with stopwatch and ruler
- Compare measured vs. learned after best scaling
Experimental Results

- Run ASAMI for pre-set amount of time (2.5 minutes)
- Measure actual models with **stopwatch and ruler**
- Compare measured vs. learned after best **scaling**

![Graphs showing comparison between measured and learned models](image-url)
Experimental Results

- Average fitness of model over 15 runs
Learning in Two Dimensions

- Robot learns while traversing rectangular field
  - Combinations of forward, sideways, and turning motion
  - Field has four color-coded cylindrical landmarks
Learning in Two Dimensions

- Robot learns while traversing rectangular field
  - Combinations of forward, sideways, and turning motion
  - Field has four color-coded cylindrical landmarks
Learning in Two Dimensions

- Robot learns while traversing rectangular field
  - Combinations of forward, sideways, and turning motion
  - Field has four color-coded cylindrical landmarks
2nd Robotic Platform: Autonomous Car

- **Self-driving car** provides many **challenges** for autonomous model learning

- Actions lead to accelerations, angular velocity:
  - Throttle, brake, and steering position

- Sensors provide information about pose and map:
  - Three-dimensional LIDAR

- Again learn both models starting without accurate estimate of either
3d LIDAR for Autonomous Cars

- The Velodyne LIDAR sensor:
  - 64 lasers return distance readings
  - Each laser is at a different vertical angle and different horizontal offset
  - Unit spins around vertical axis at 10Hz
Summary

- **ASAMI**: Autonomous, no external feedback
- Computationally **efficient**
- Starts with poor action model, no sensor model
  - Learns **accurate** approximations to both models
  - Models are to scale with each other
Outline

• Machine learning for fast walking (Kohl, Stone)

• Learning to acquire the ball (Fidelman, Stone)

• Learning sensor and action models (Stronger, Stone)

• Color constancy on mobile robots (Sridharan, Stone)

• Autonomous Color Learning (Sridharan, Stone)
Color Constancy

• Visual system’s ability to recognize true color across variations in environment
Color Constancy

- Visual system’s ability to recognize true color across variations in environment

- Challenge: Nonlinear variations in sensor response with change in illumination
Color Constancy

• Visual system’s ability to recognize true color across variations in environment

• Challenge: Nonlinear variations in sensor response with change in illumination

• Mobile robots:
  – Computational limitations
  – Changing camera positions
Vision Flowchart
Segmentation

- **Color Segmentation:**
  - Hand-label discrete colors.
  - Intermediate color maps.
  - NNr weighted average – Master color cube.
  - 128x128x128 color map – 2MB.
Sample Images
Sample Images
Sample Images
Sample Images
Our Goal

- Match current performance in **changing lighting**
- Experiments on ERS-210A robots
Training/Testin

Off-board training: Recognize 10 different colors

- Color cube: $128 \times 128 \times 128$ pixel values $\mapsto$ color label
- Nearest Neighbor/weighted average approach
Training/Testing

Off-board training: Recognize 10 different colors

- **Color cube**: $128 \times 128 \times 128$ pixel values $\mapsto$ color label
- **Nearest Neighbor**/weighted average approach

On-board testing:
Training/Testing

Off-board training: Recognize 10 different colors

- **Color cube**: $128 \times 128 \times 128$ pixel values $\rightarrow$ color label
- **Nearest Neighbor**/weighted average approach

On-board testing:

- **Segment** images using color map
Training/Testing

Off-board training: Recognize 10 different colors
  - Color cube: $128 \times 128 \times 128$ pixel values $\mapsto$ color label
  - Nearest Neighbor/weighted average approach

On-board testing:
  - Segment images using color map
  - Run-length encoding, region growing: detect markers
Training/Testing

**Off-board training:** Recognize *10 different colors*

- **Color cube:** 128 × 128 × 128 pixel values → color label
- **Nearest Neighbor*/weighted average approach

**On-board testing:**

- **Segment** images using color map
- Run-length encoding, region growing: detect markers
- Markers used for **Localization**
- Higher level strategies and **action selection**
Training/Testing

Off-board training: Recognize 10 different colors
- **Color cube**: 128 × 128 × 128 pixel values → color label
- **Nearest Neighbor**/weighted average approach

On-board testing:
- **Segment** images using color map
- Run-length encoding, region growing: detect markers
- Markers used for **Localization**
- Higher level strategies and **action selection**

**Real-time** color constancy without degradation
Approach

- Most previous: static cameras, few colors
Approach

• Most previous: static cameras, few colors

• Here: \textit{discrete} 2-illumination case: 1500lux vs. 400lux
Approach

• Most previous: static cameras, few colors

• Here: discrete 2-illumination case: 1500lux vs. 400lux

• Compare image pixel distributions (in normalized RGB)
Approach

• Most previous: static cameras, few colors

• Here: discrete 2-illumination case: 1500lux vs. 400lux

• Compare image pixel distributions (in normalized RGB)

• KL-divergence as similarity metric:
  – Given image, determine distribution in (r,g) space
  – Compare distribution A,B (N=64)

  \[
  KL(A, B) = - \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} (A_{i,j} \ln \frac{B_{i,j}}{A_{i,j}})
  \]

  – Small value ⇒ similar
Approach

• Most previous: static cameras, few colors

• Here: discrete 2-illumination case: 1500lux vs. 400lux

• Compare image pixel distributions (in normalized RGB)

• KL-divergence as similarity metric:
  – Given image, determine distribution in (r,g) space
  – Compare distribution A,B (N=64)

  \[
  KL(A, B) = - \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} (A_{i,j} \ln \frac{B_{i,j}}{A_{i,j}})
  \]

  – Small value ⇒ similar
  – Robust to large peaks in observed color distributions
Training Phase
Testing Phase
Results

- Test on *find-and-walk-to-ball* task
Results

− Test on find-and-walk-to-ball task

<table>
<thead>
<tr>
<th>Lighting transition</th>
<th>Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>15.2 ± 0.8</td>
</tr>
<tr>
<td>Bright/Dark</td>
<td>26.5 ± 1.7</td>
</tr>
<tr>
<td>Dark/Bright</td>
<td>20.1 ± 2.7</td>
</tr>
</tbody>
</table>

− Also tested intermediate illuminations; adversarial case
Results

- Test on *find-and-walk-to-ball* task

<table>
<thead>
<tr>
<th>Lighting transition</th>
<th>Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>15.2 ± 0.8</td>
</tr>
<tr>
<td>Bright/Dark</td>
<td>26.5 ± 1.7</td>
</tr>
<tr>
<td>Dark/Bright</td>
<td>20.1 ± 2.7</td>
</tr>
</tbody>
</table>

- Also tested *intermediate* illuminations; *adversarial* case
- On *ERS-7*, 3 illuminations ⇒ *whole range* of lab conditions
Results

- Test on *find-and-walk-to-ball* task

<table>
<thead>
<tr>
<th>Lighting transition</th>
<th>Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>15.2 ± 0.8</td>
</tr>
<tr>
<td>Bright/Dark</td>
<td>26.5 ± 1.7</td>
</tr>
<tr>
<td>Dark/Bright</td>
<td>20.1 ± 2.7</td>
</tr>
</tbody>
</table>

- Also tested *intermediate* illuminations; adversarial case
- On *ERS-7*, 3 illuminations ⇒ *whole range* of lab conditions
- Works in *real-time*
Autonomous Color Learning

- Color Constancy: more tediously created maps
  - Hand-labeling many images → hours of manual effort
Autonomous Color Learning

- Color Constancy: more tediously created maps
  - Hand-labeling many images → hours of manual effort

- Use the structured environment
  - Robot learns color distributions
Autonomous Color Learning

- Color Constancy: more tediously created maps
  - Hand-labeling many images → hours of manual effort

- Use the structured environment
  - Robot learns color distributions

- Comparable accuracy, 5 minutes of robot effort
Summary

- Learning on **physical robots**
  - No simulation, minimal human intervention
Summary

- Learning on **physical robots**
  - No simulation, minimal human intervention

- **Motion**: learning for fast walking

- **Behavior**: acquiring the ball

- **Localization**: ASAMI

- **Vision**: color constancy, autonomous color learning
Other Robotics Research

- TD learning for **strategy** (Stone, Sutton, Kuhlmann)
- **Collaborative surveillance** (Ahmadi, Stone)
- “Urban Challenge:” **autonomous vehicles** (Beeson et al.)
- Autonomous **traffic management** (Dresner, Stone)
Acknowledgements

Thanks to all the Students Involved!

- Dan Stronger, Nate Kohl, Peggy Fidelman, Mohan Sridharan
- Other members of the UT Austin Villa Legged Robot Team
- http://www.cs.utexas.edu/~AustinVilla
Acknowledgements

Thanks to all the Students Involved!

- Dan Stronger, Nate Kohl, Peggy Fidelman, Mohan Sridharan
- Other members of the UT Austin Villa Legged Robot Team
- http://www.cs.utexas.edu/~AustinVilla
- Fox Sports World for inspiration!