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

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

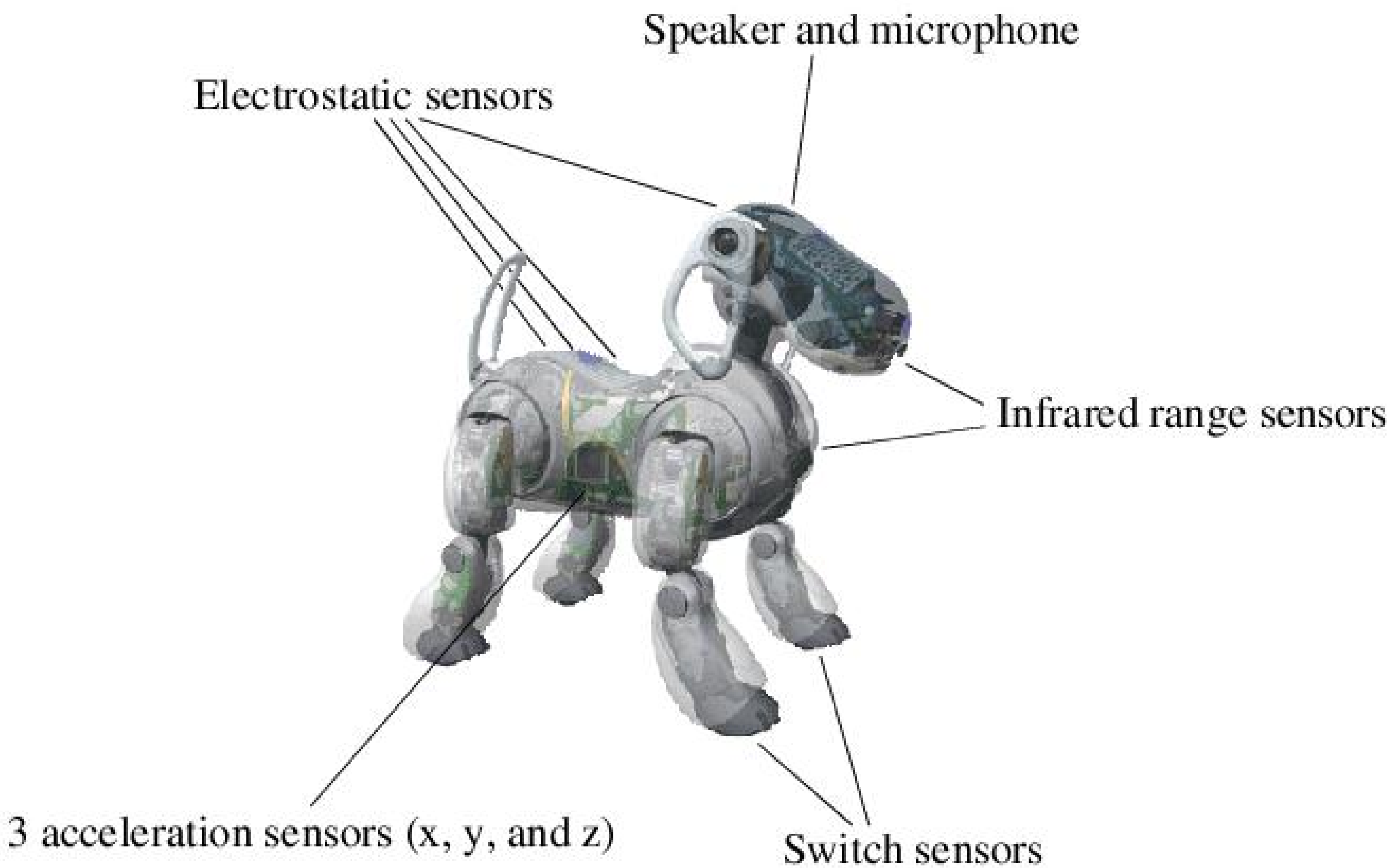
Middle-sized League

Legged Robot League

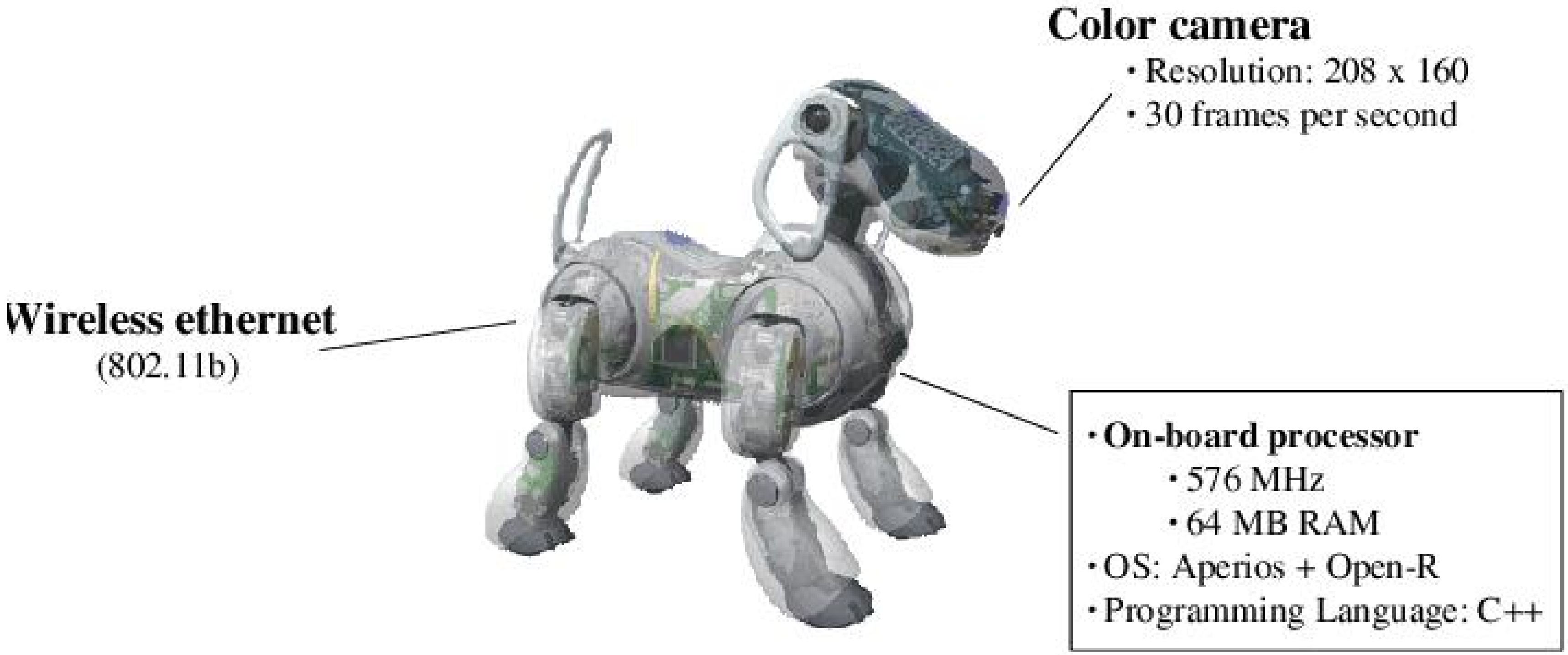
Simulation League

Humanoid League

Sony Aibo (ERS-210A, ERS-7)

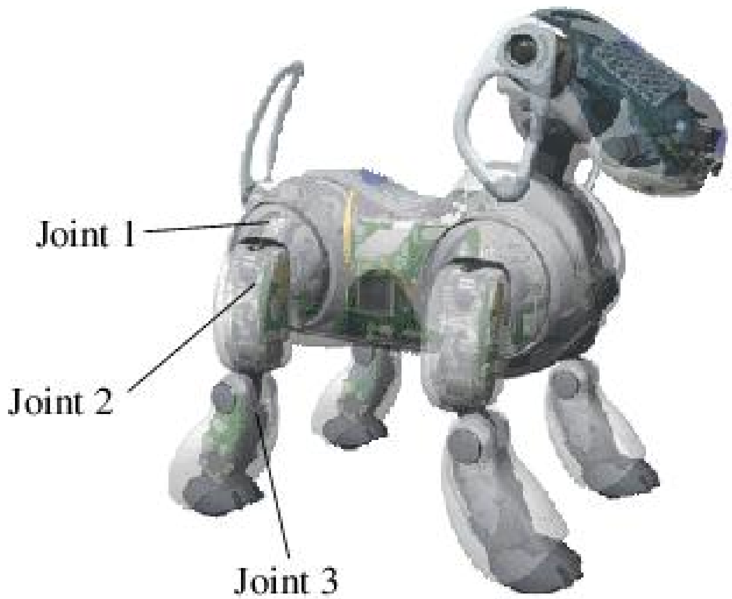


Sony Aibo (ERS-210A, ERS-7)



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

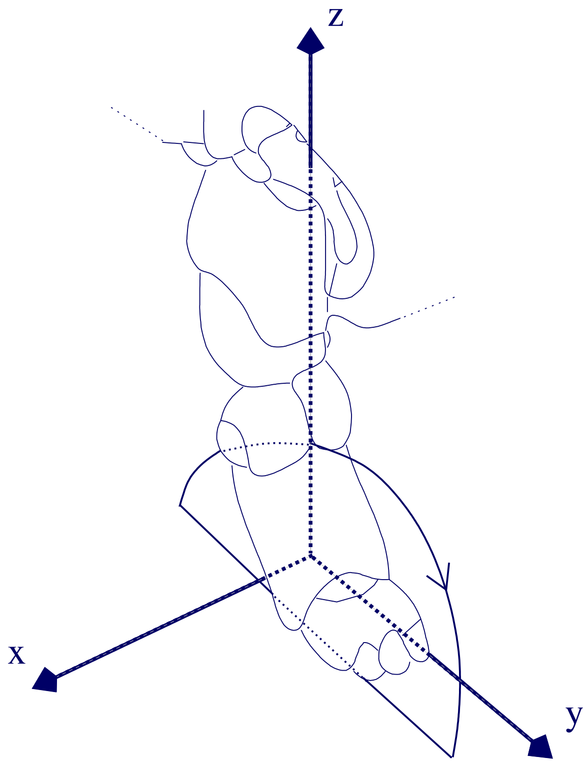
- Walks that "come with" Aibo are **slow**
- **RoboCup** soccer: **25+ Aibo teams** internationally
  - Motivates faster walks

Hand-tuned gaits (2003)			Learned gaits	
German Team	UT Austin Villa	UNSW	Hornby et al. (1999)	Kim & Uther (2003)
230 mm/s	**245**	**254**	**170**	**270** (± 5)

A Parameterized Walk

- Developed from scratch as part of **UT Austin Villa** 2003
- **Trot gait** with elliptical locus on each leg

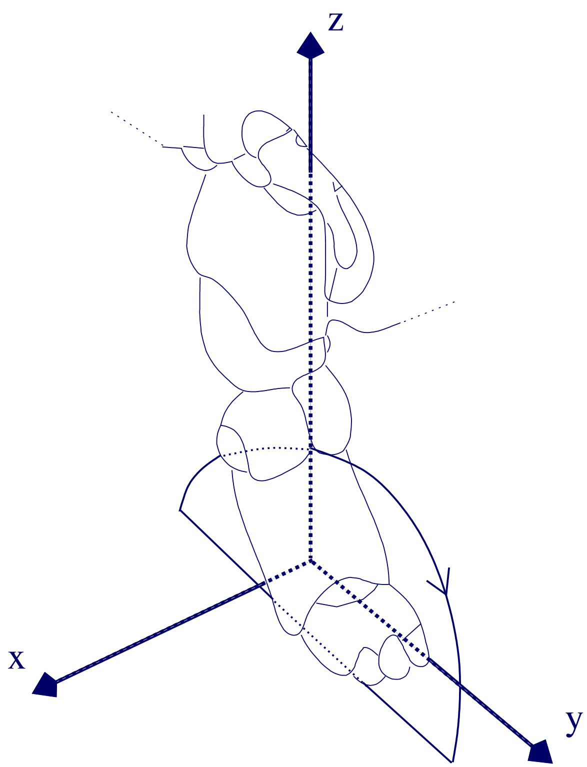
Locus Parameters



- 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 π

Experimental Setup

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

- 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 π
- 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 π
- 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 π want to move in direction of **gradient** of $V(\pi)$

Policy Gradient RL

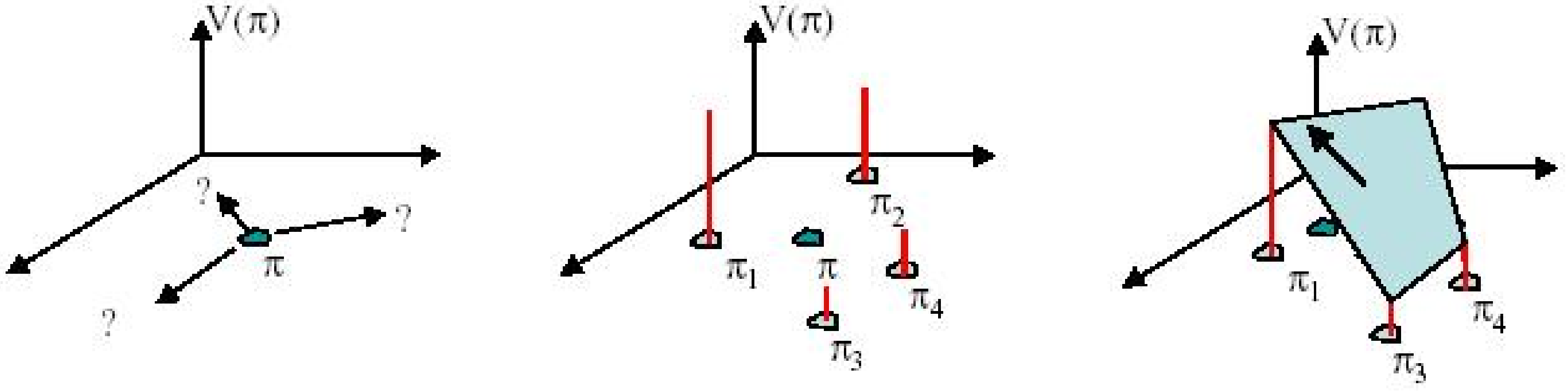
- From π want to move in direction of **gradient** of $V(\pi)$
  - Can't compute $\frac{\partial V(\pi)}{\partial \theta_i}$ directly: **estimate** empirically

Policy Gradient RL

- From π 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 π 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

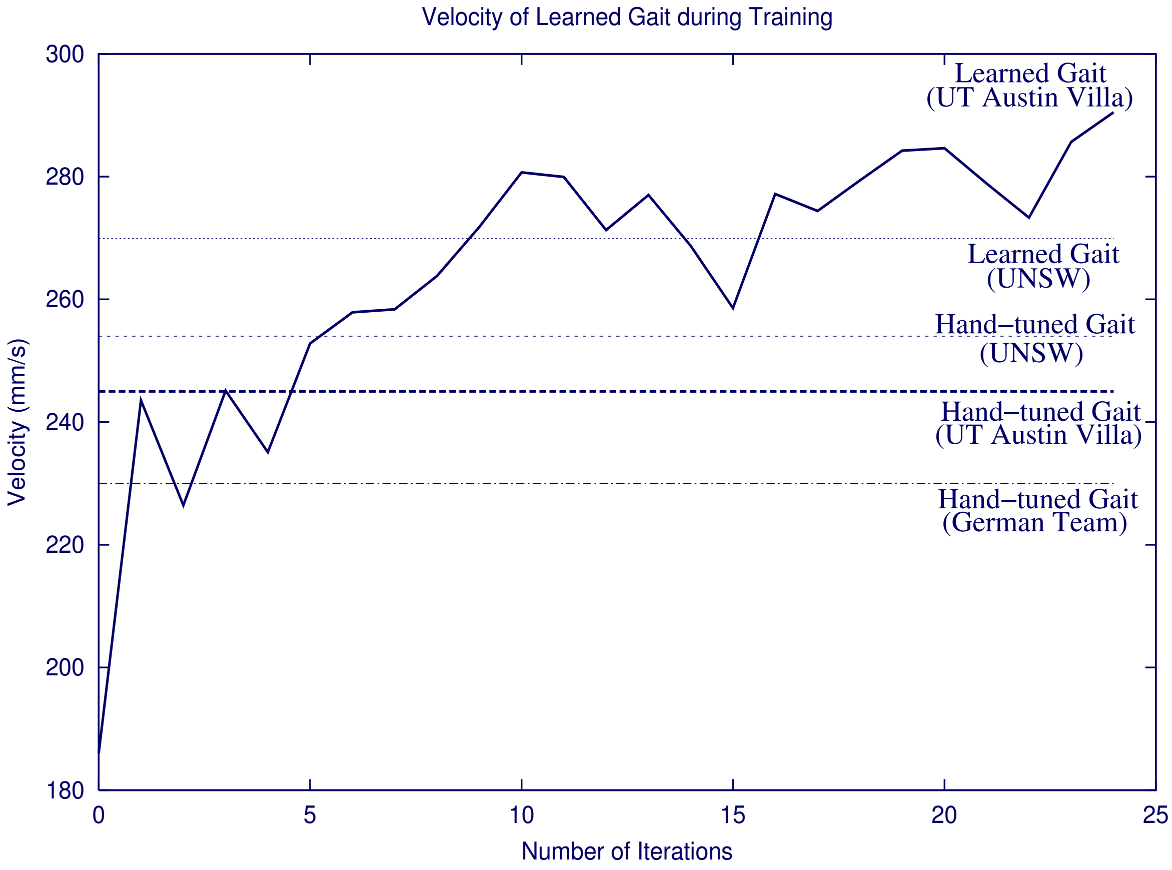
- Started from **stable**, but fairly slow gait
- Used **3 robots** simultaneously
- Each iteration takes 45 traversals, $7\frac{1}{2}$ minutes

Before learning

After learning

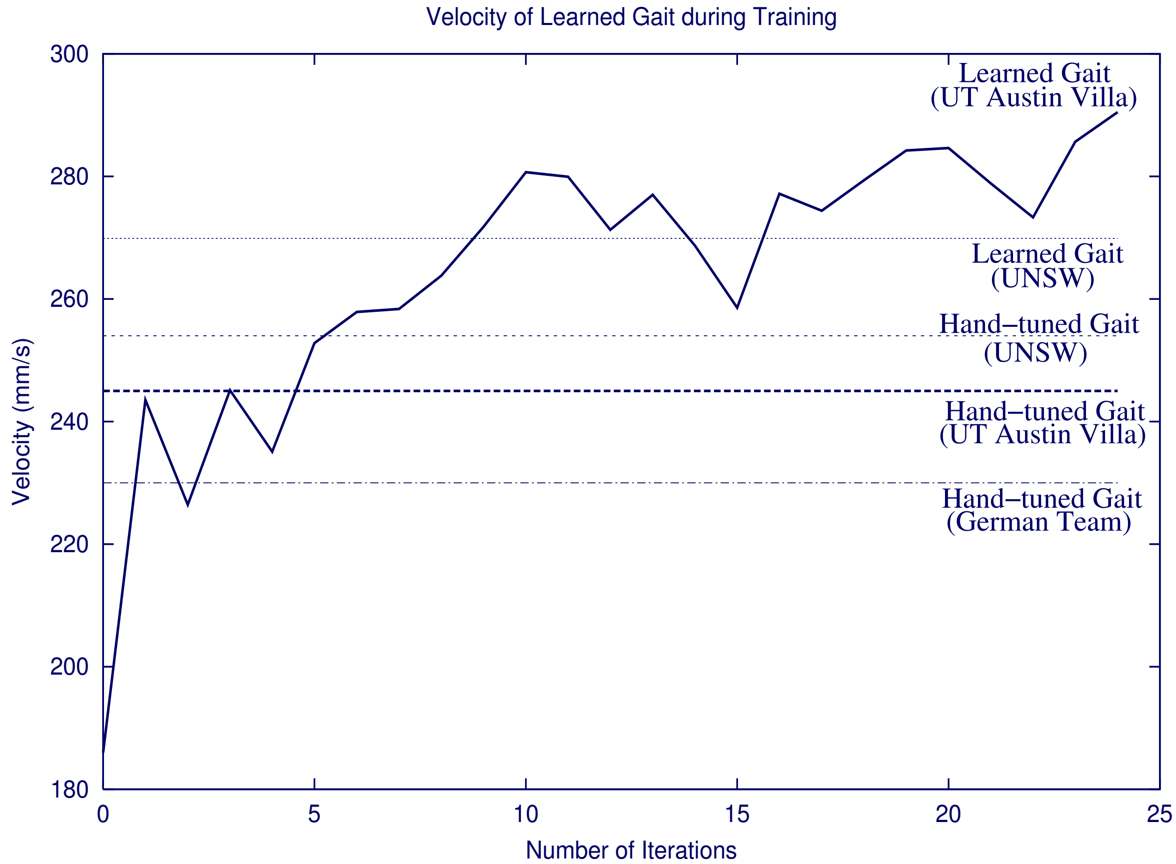
- 24 iterations = **1080 field traversals**, $\approx$ **3 hours**

Results



Velocity of Learned Gait during Training
300
280
260
240
220
200
180
0
5
10
15
20
25
Velocity (mm/s)
Number of Iterations
Learned Gait (UT Austin Villa)
Learned Gait (UNSW)
Hand–tuned Gait (UNSW)
Hand–tuned Gait (UT Austin Villa)
Hand–tuned Gait (German Team)

Results

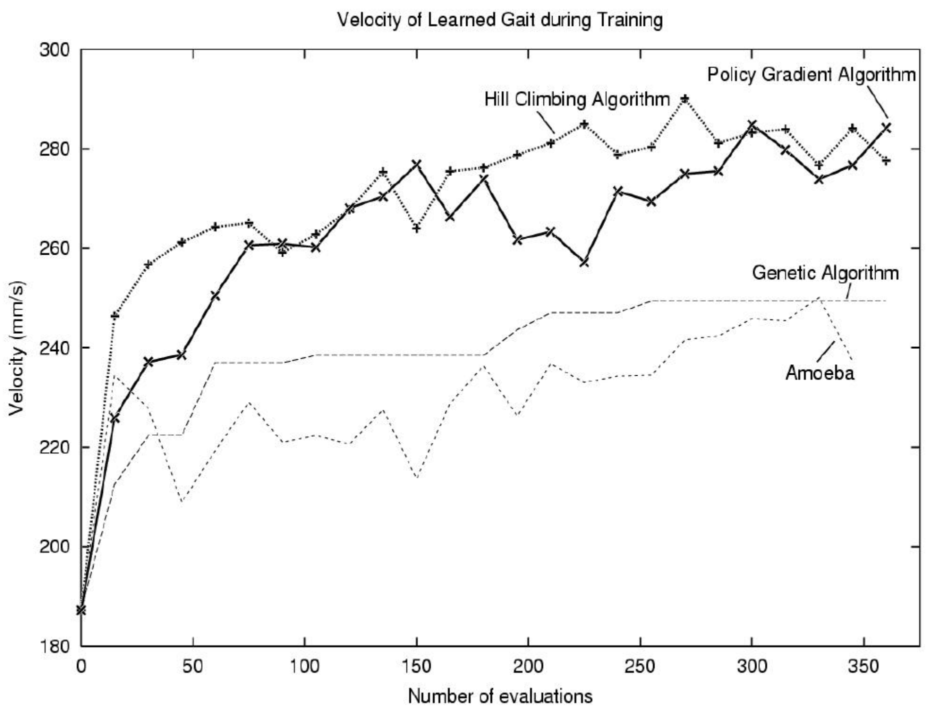


- Additional iterations didn't help
- Spikes: evaluation **noise**? large **step size**?

Learned Parameters

Parameter	Initial Value	ϵ	Best Value
Front ellipse:			
(height)	4.2	0.35	4.081
(x offset)	2.8	0.35	0.574
(y offset)	4.9	0.35	5.152
Rear ellipse:			
(height)	5.6	0.35	6.02
(x offset)	0.0	0.35	0.217
(y offset)	-2.8	0.35	-2.982
Ellipse length	4.893	0.35	5.285
Ellipse skew multiplier	0.035	0.175	0.049
Front height	7.7	0.35	7.483
Rear height	11.2	0.35	10.843
Time to move through locus	0.704	0.016	0.679
Time on ground	0.5	0.05	**0.430**

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 itervention** (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 $\mapsto$ ball distance
- Movement specified by **4 parameters**

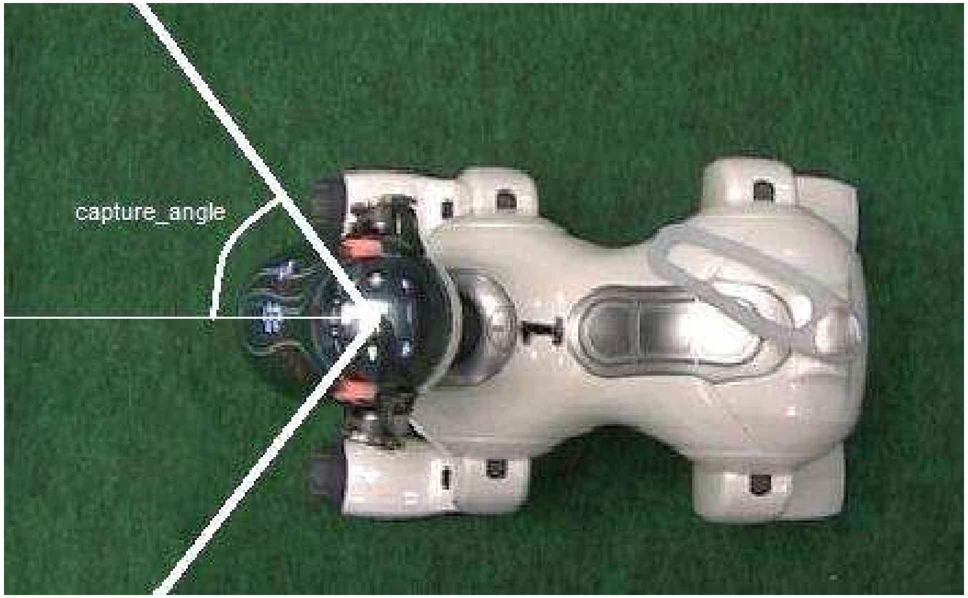
Grasping the Ball

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

$\boxed{\textbf{Brittle!}}$

Parameterization

- **slowdown_dist:** when to slow down
- **slowdown_factor:** how much to slow down
- **capture_angle:** when to stop turning



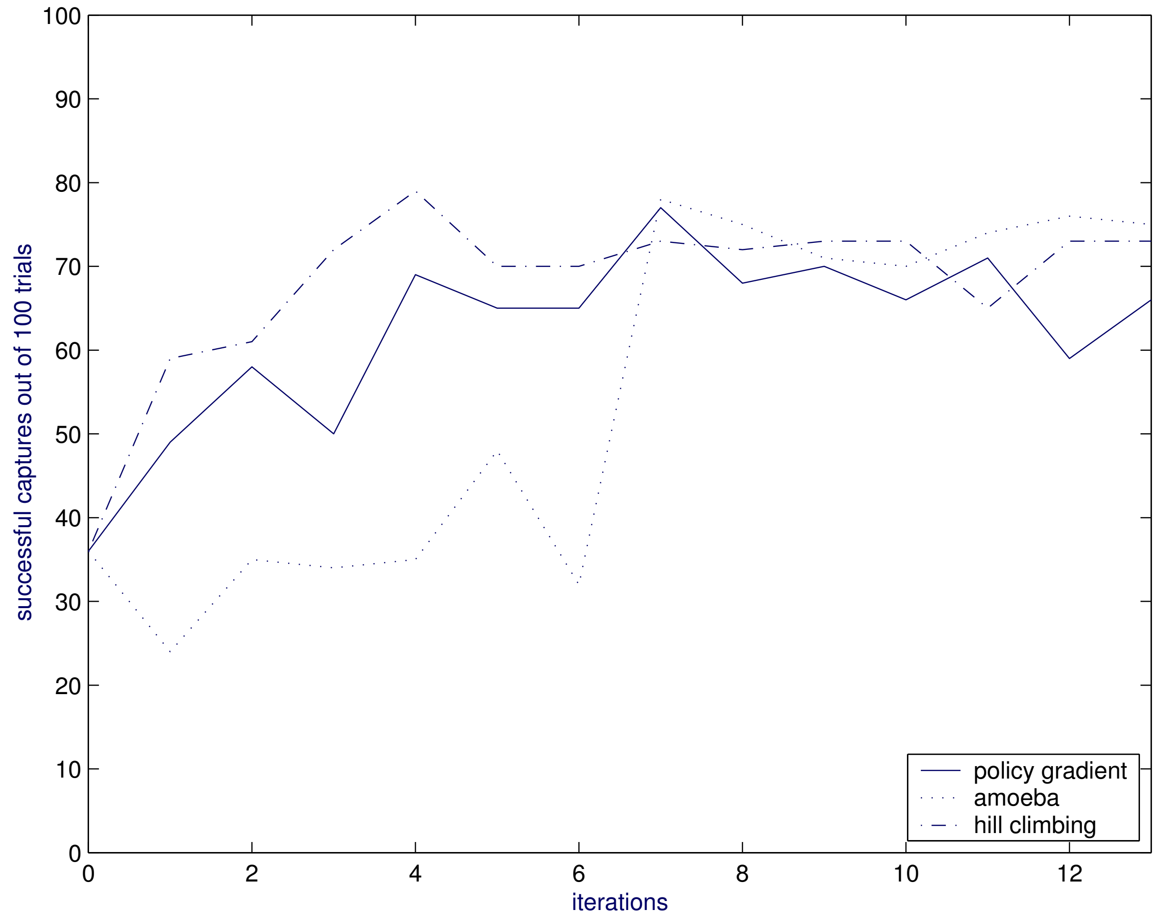
- **capture_dist:** when to put down head

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

Policy	slowdown dist	slowdown factor	capture angle	capture dist	Success rate
Initial	200mm	0.7	15.0^o	110mm	36%
Policy gradient	125mm	1	17.4^o	152mm	64%
Amoeba	208mm	1	33.4^o	162mm	69%
Hill climbing	240mm	1	35.0^o	170mm	66%

Instance of Layered Learning

- For domains too **complex** for tractably mapping state features $S \longmapsto$ 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
- **A**utonomous **S**ensor and **A**ctuator **M**odel **I**nduction (ASAMI)

Learned Action/Sensor Models

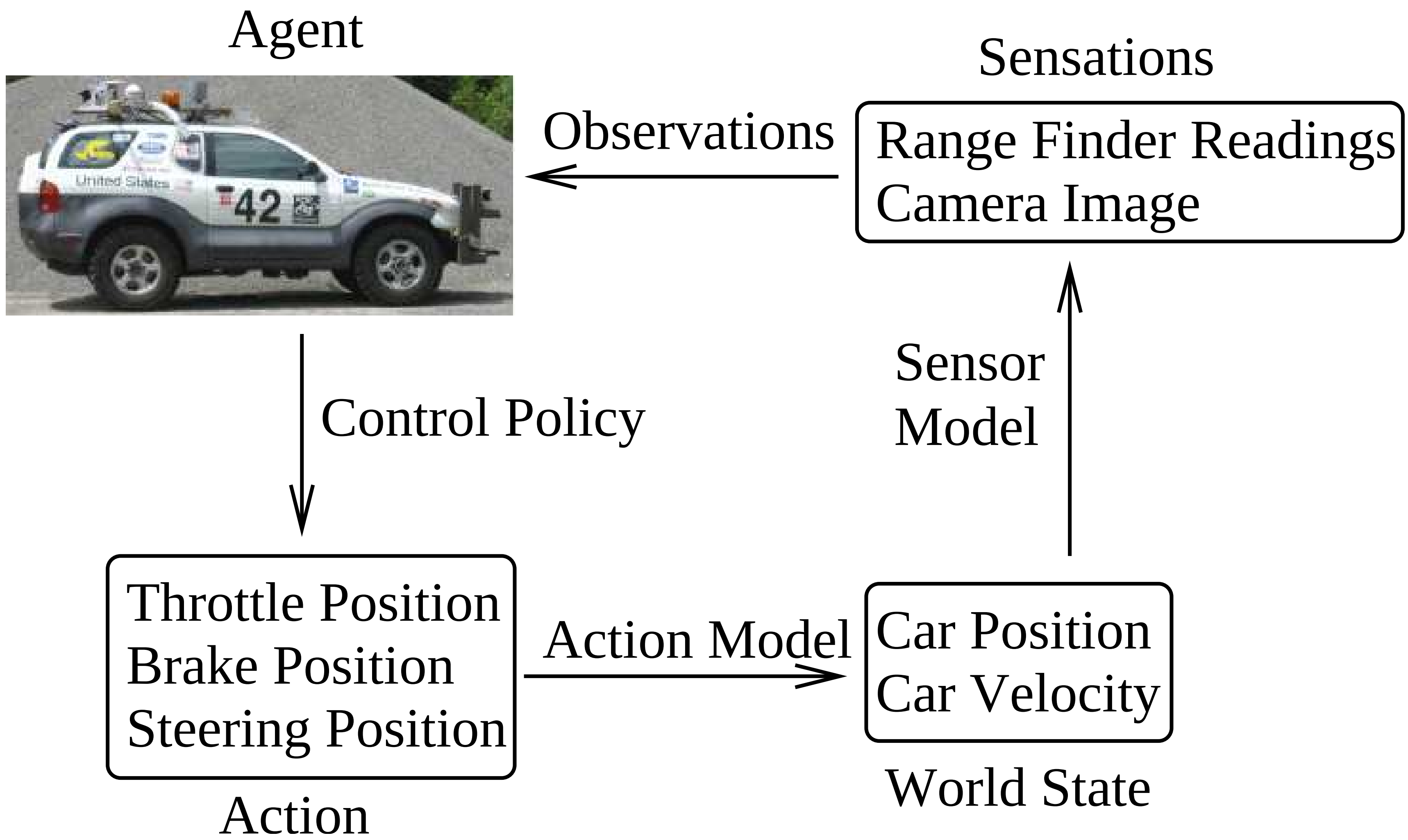
- Mobile robots rely on **models of their actions and sensors**
  - Typically tuned **manually**: Time-consuming
- **A**utonomous **S**ensor and **A**ctuator **M**odel **I**nduction (ASAMI)
- ASAMI is **autonomous**: no external feedback
  - Developmental robotics

Learned Action/Sensor Models

- Mobile robots rely on **models of their actions and sensors**
  - Typically tuned **manually**: Time-consuming
- **A**utonomous **S**ensor and **A**ctuator **M**odel **I**nduction (ASAMI)
- ASAMI is **autonomous**: no external feedback
  - Developmental robotics
- Techinique is implemented and tested in:
  - **One-dimensional scenario: Sony Aibo ERS-7**
  - Aibo in two-dimensional area
  - Second robotic platform: an autonomous car

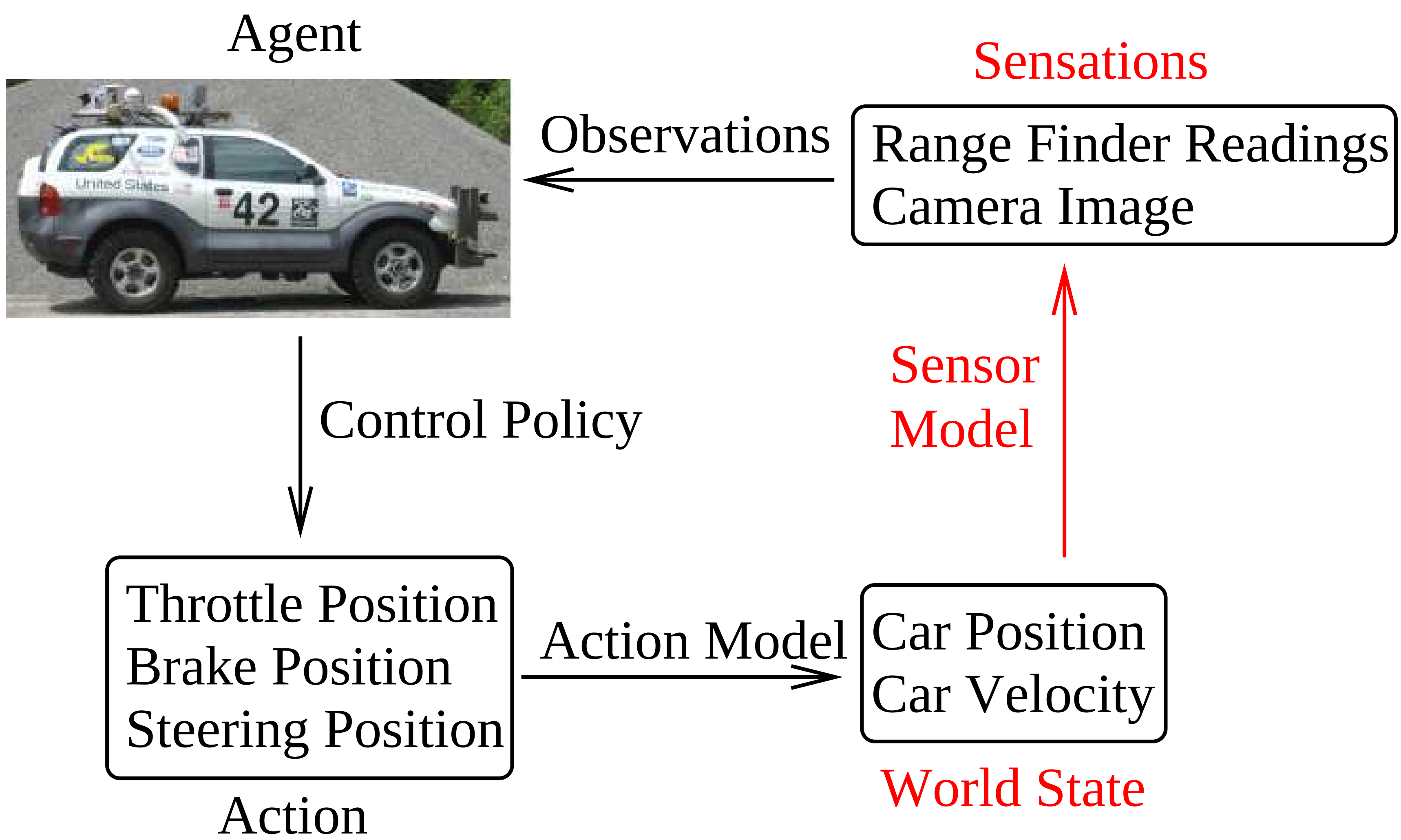
Action and Sensor Models

- Mobile robots rely on models of their actions and sensors



Action and Sensor Models

- Mobile robots rely on models of their actions and sensors



Action and Sensor Models

- Mobile robots rely on models of their actions and sensors

Agent

Sensations

Observations

Range Finder Readings
Camera Image

Control Policy

Sensor
Model

Throttle Position
Brake Position
Steering Position

Action Model

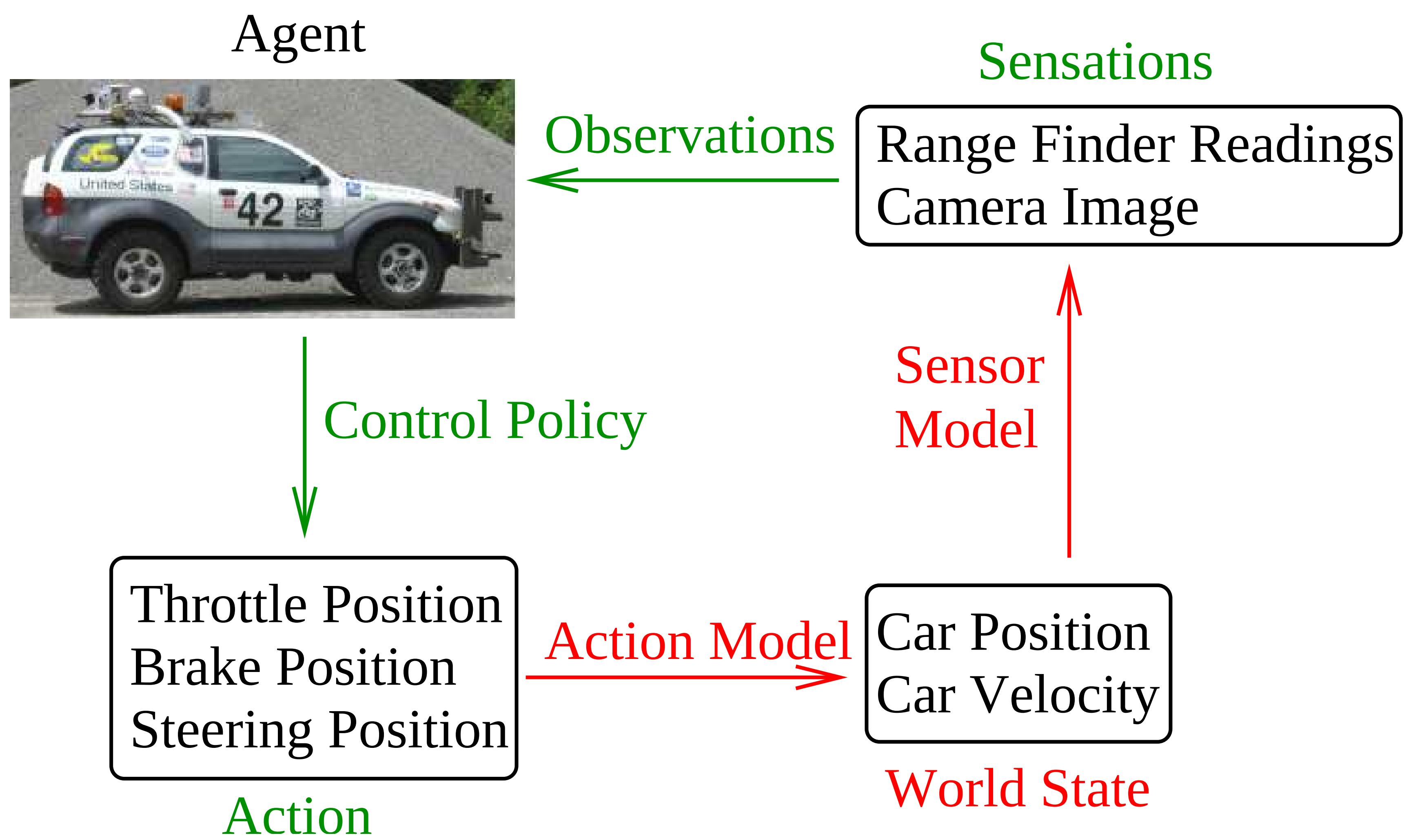
Car Position
Car Velocity

Action

World State

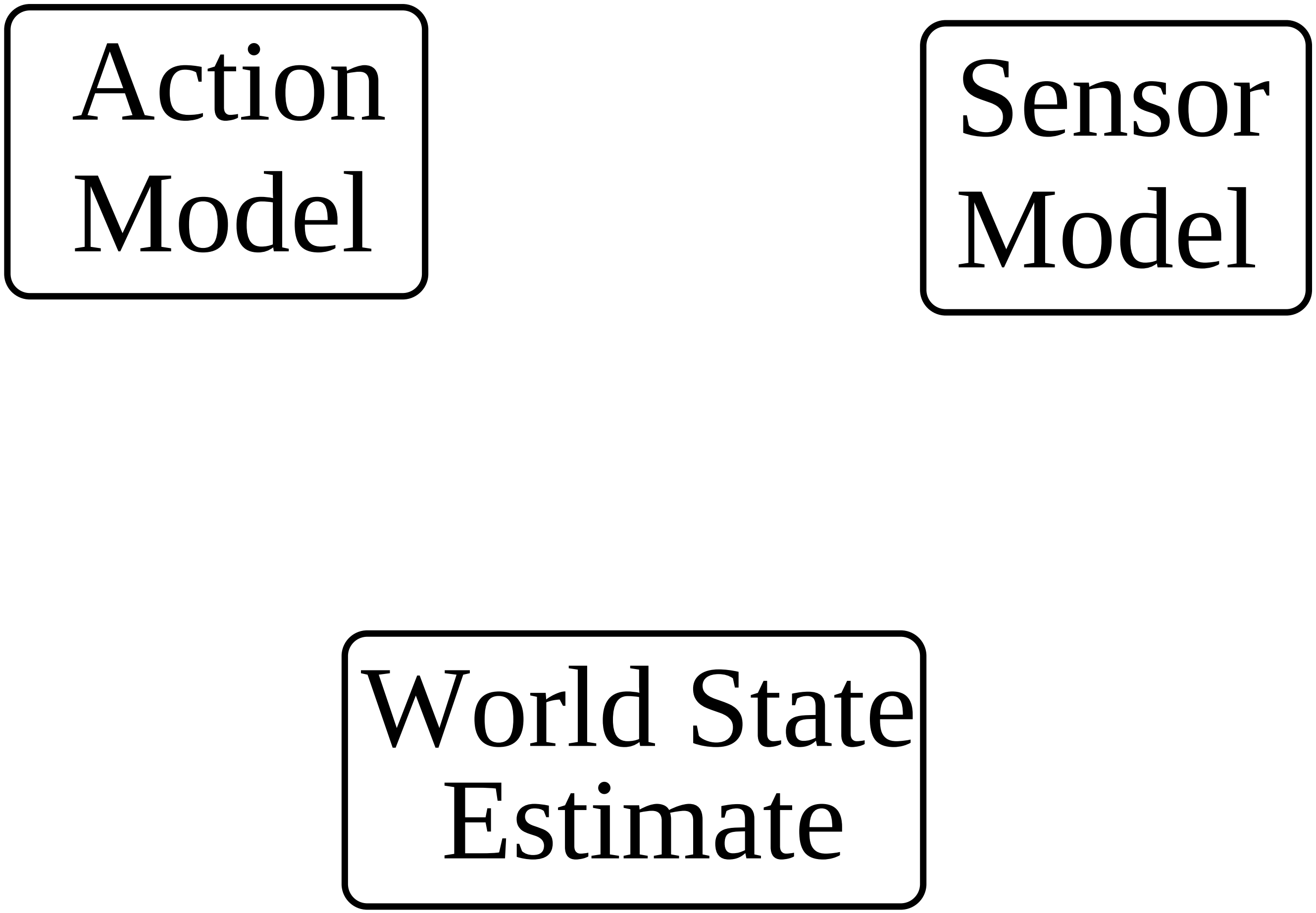
General Methodology

- Action model, sensor model, world state unknown:



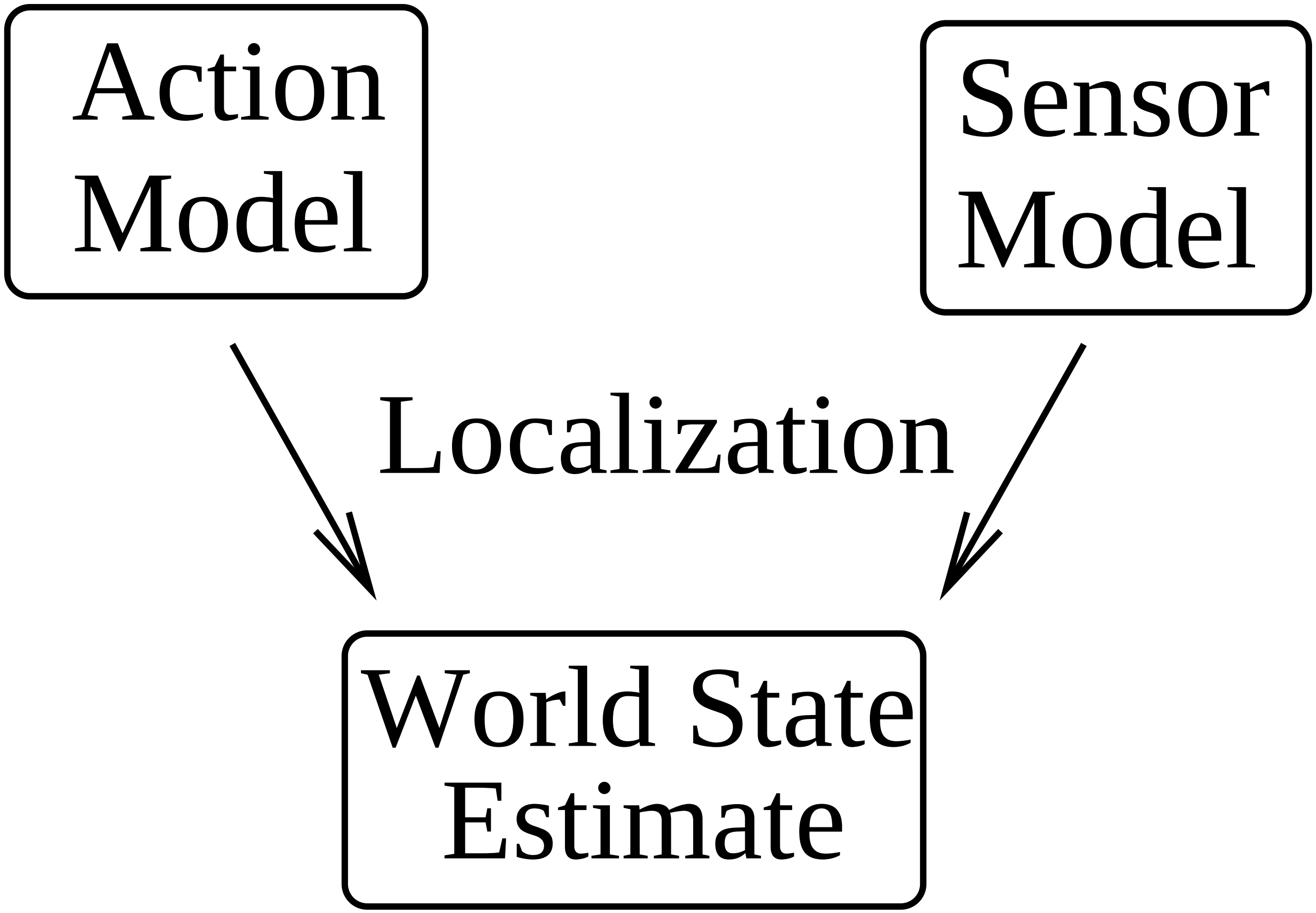
General Methodology

- Given the robot's actions and observations:



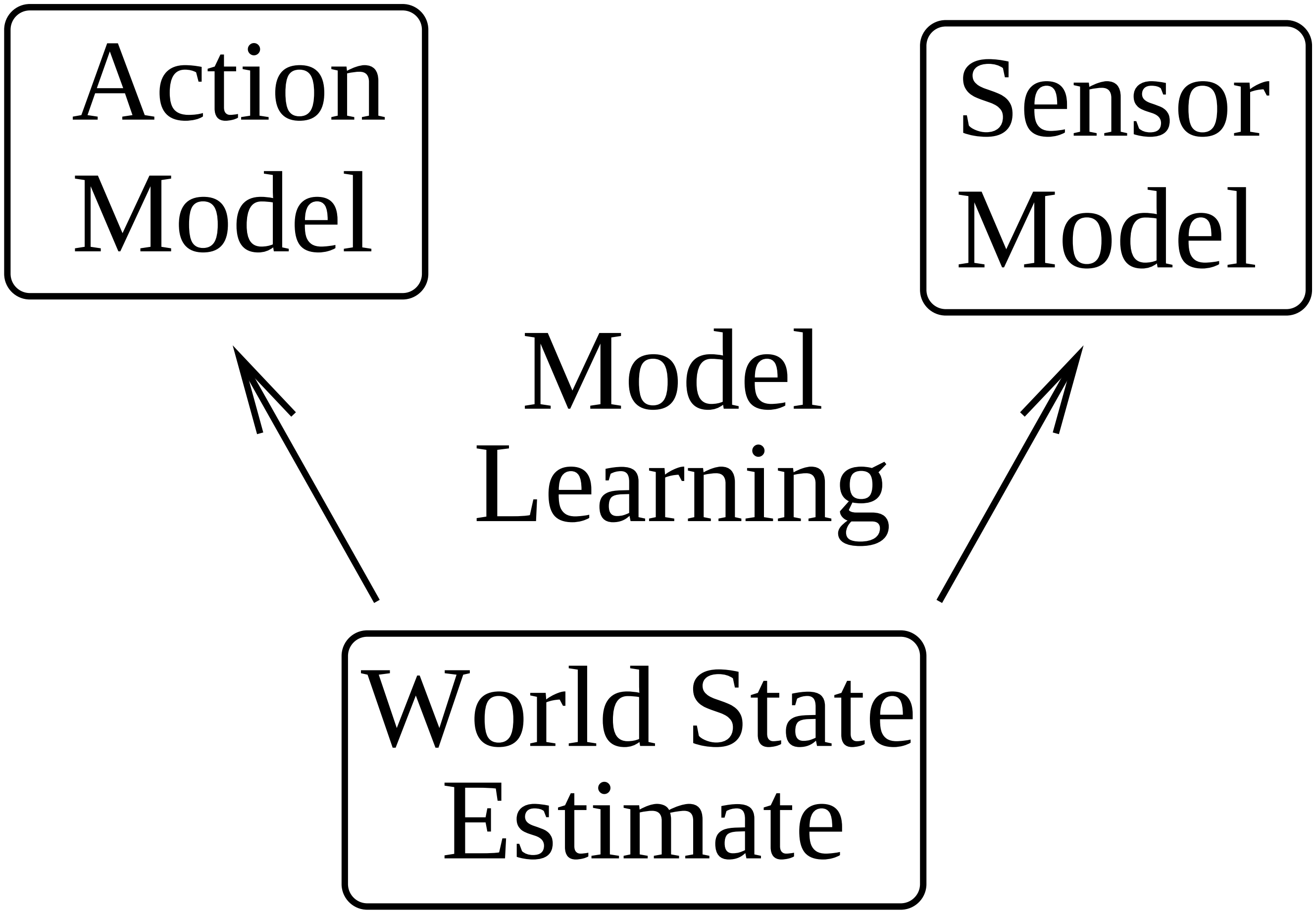
General Methodology

- Given the robot's actions and observations:



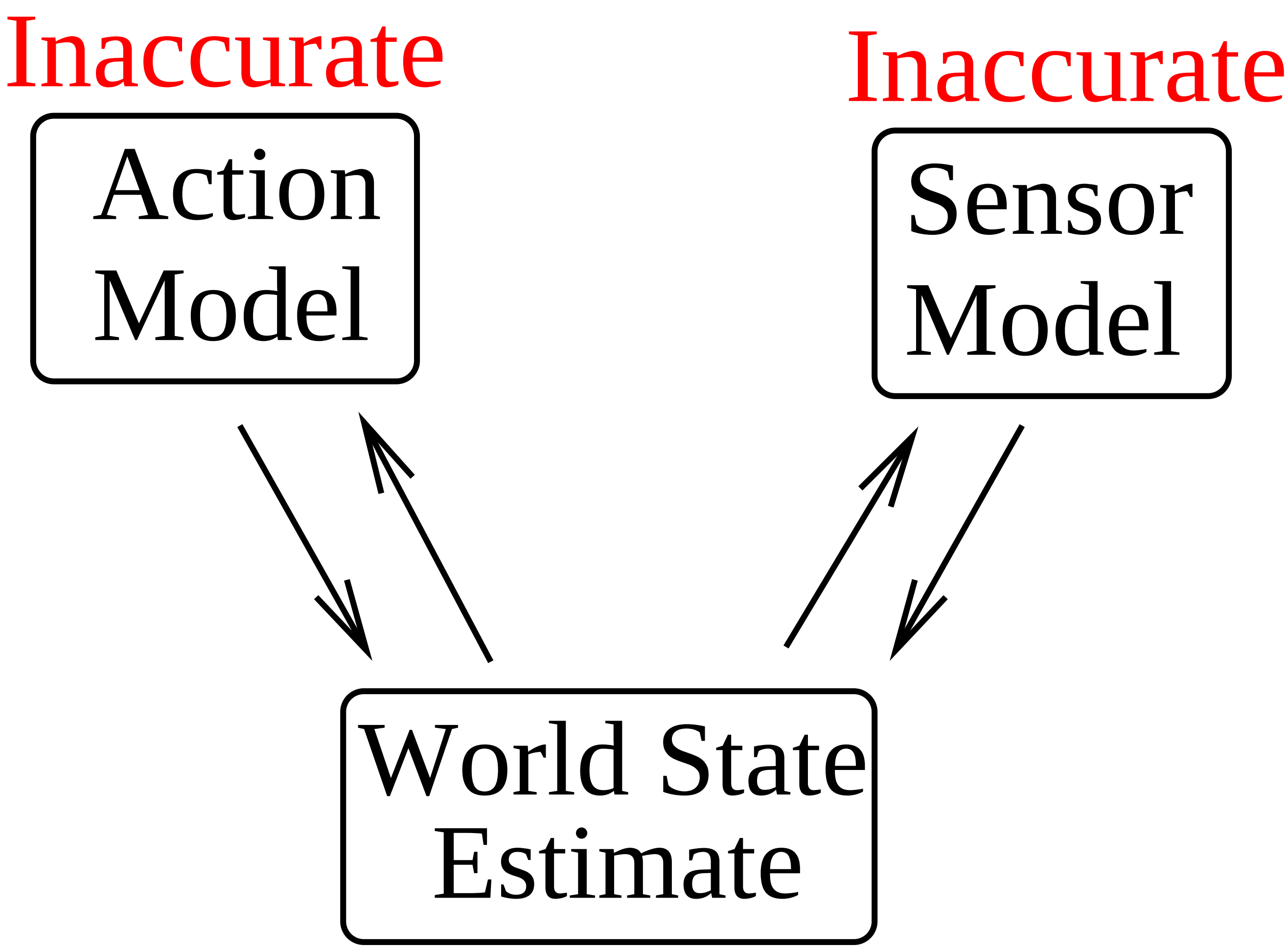
General Methodology

- Given the robot's actions and observations:



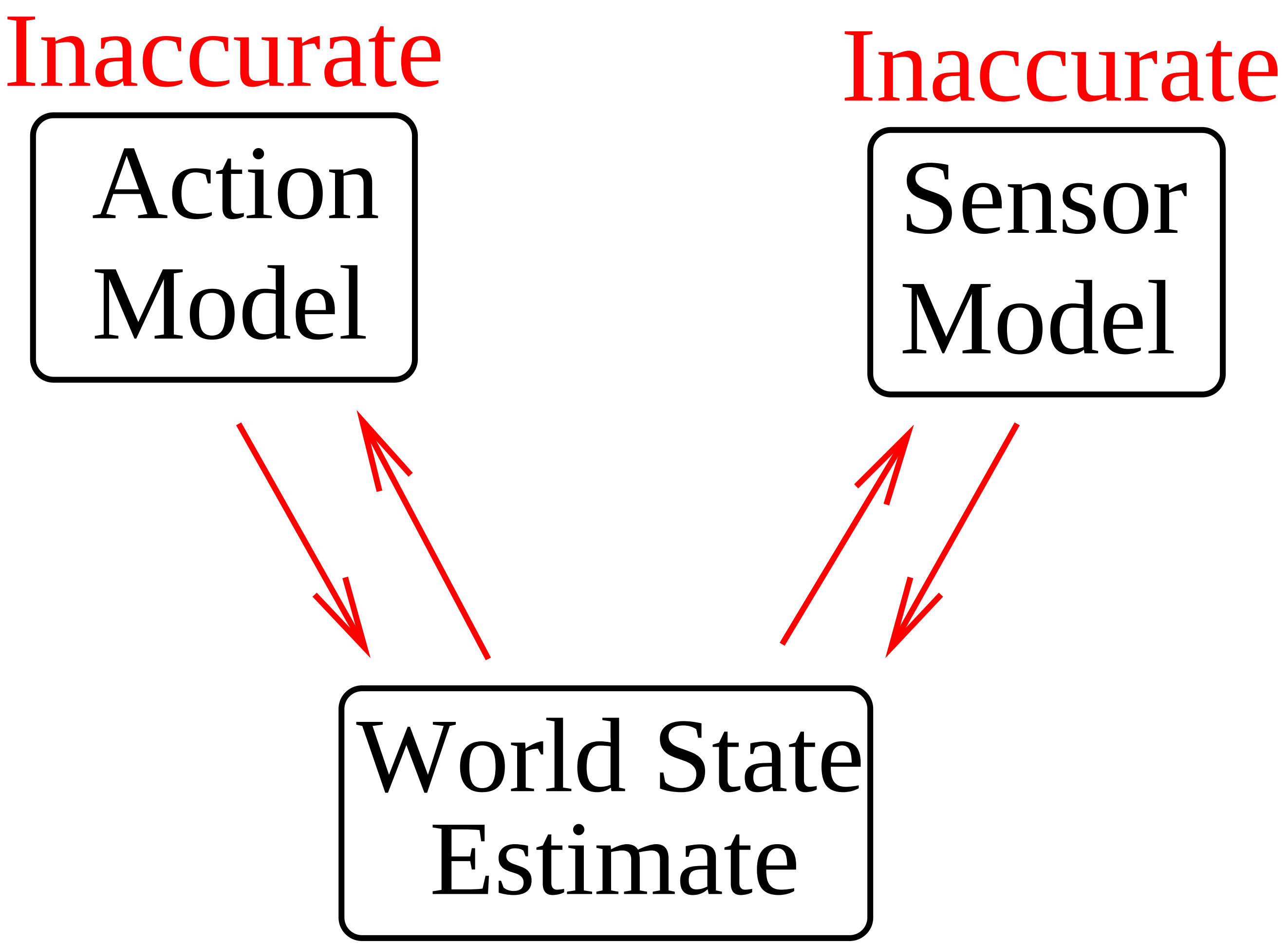
General Methodology

- Given the robot's actions and observations:



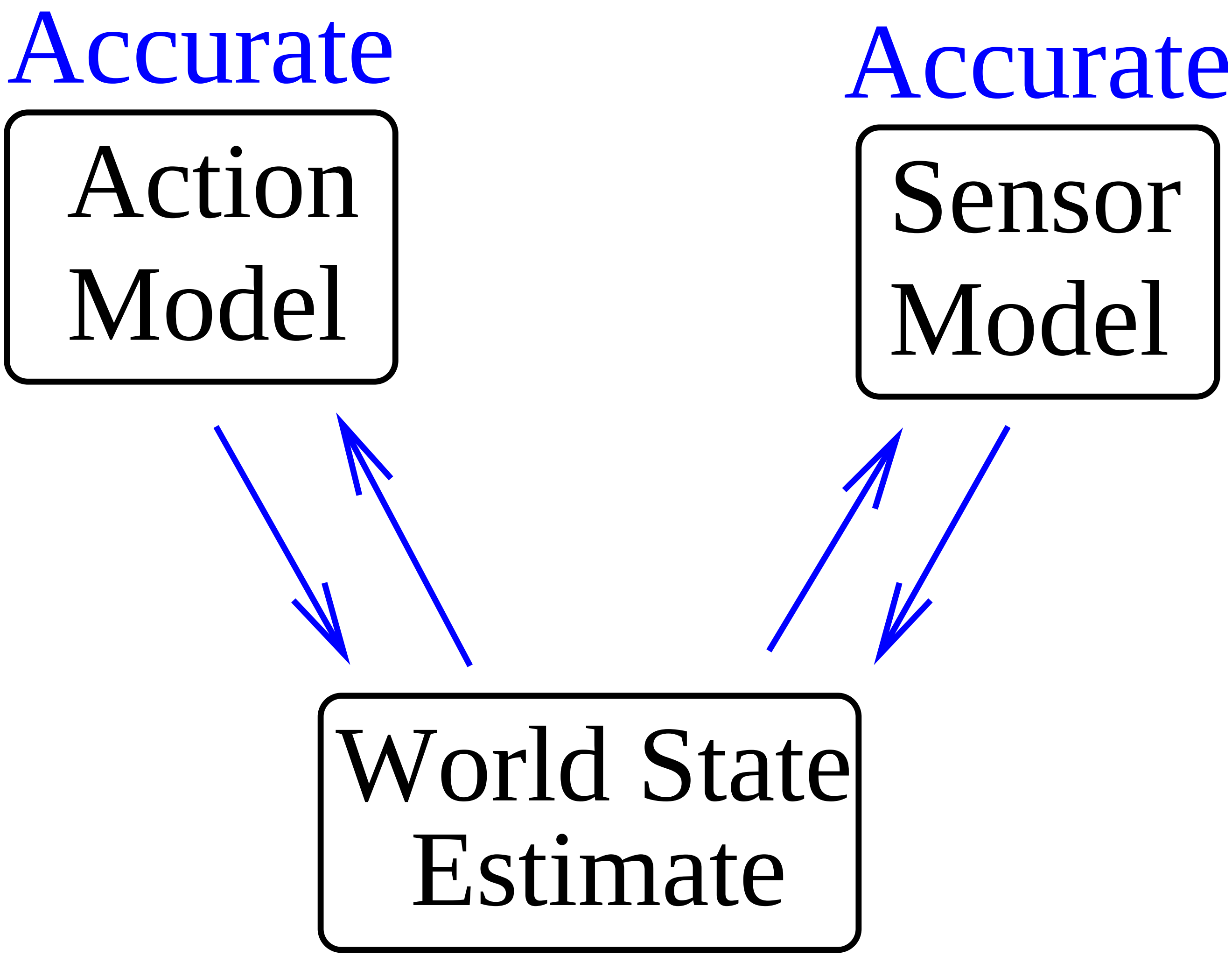
General Methodology

- Given the robot's actions and observations:

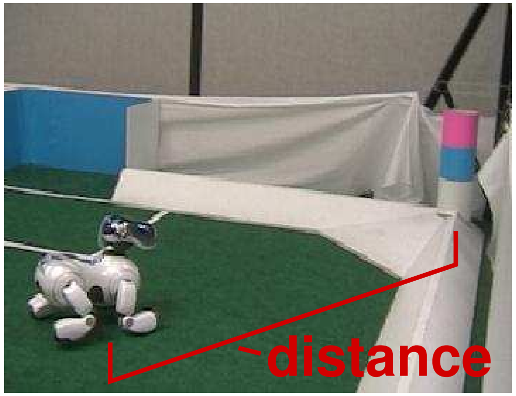


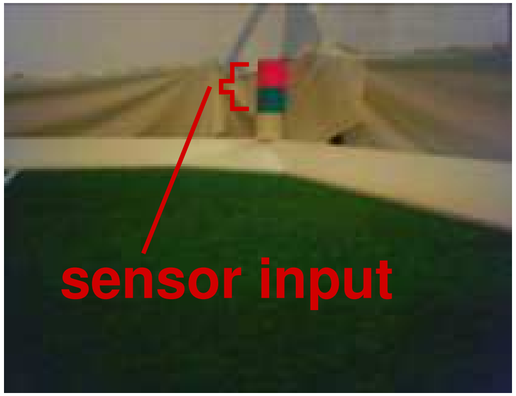
General Methodology

- Given the robot's actions and observations:



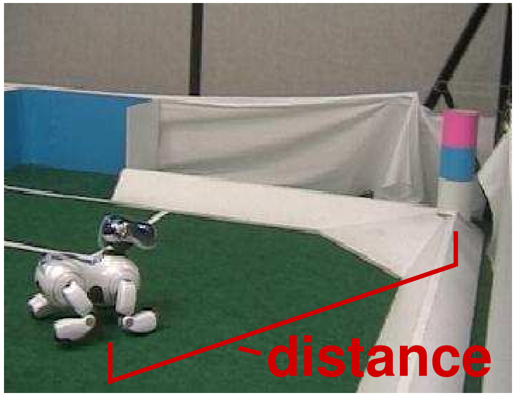
The Task

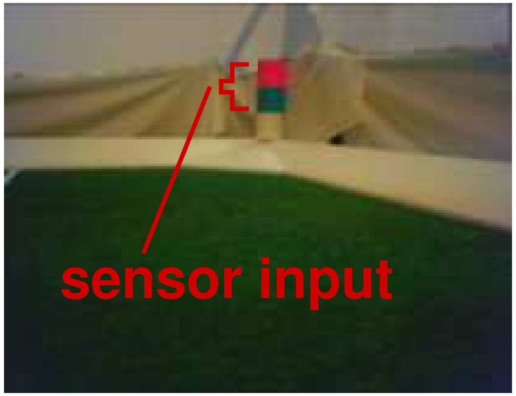




- **Sensor model:** beacon height in image $\mapsto$ distance
  - Mapping derived from camera specs not accurate

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:
$$x_s(t_k) = S(obs_k)$$

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:

$$x_a(t) = x(0) + \int_0^t A(C(s))\, ds$$

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

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

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:
$$x_a(t) = x(0) + \int_0^t A(C(s))\, ds$$

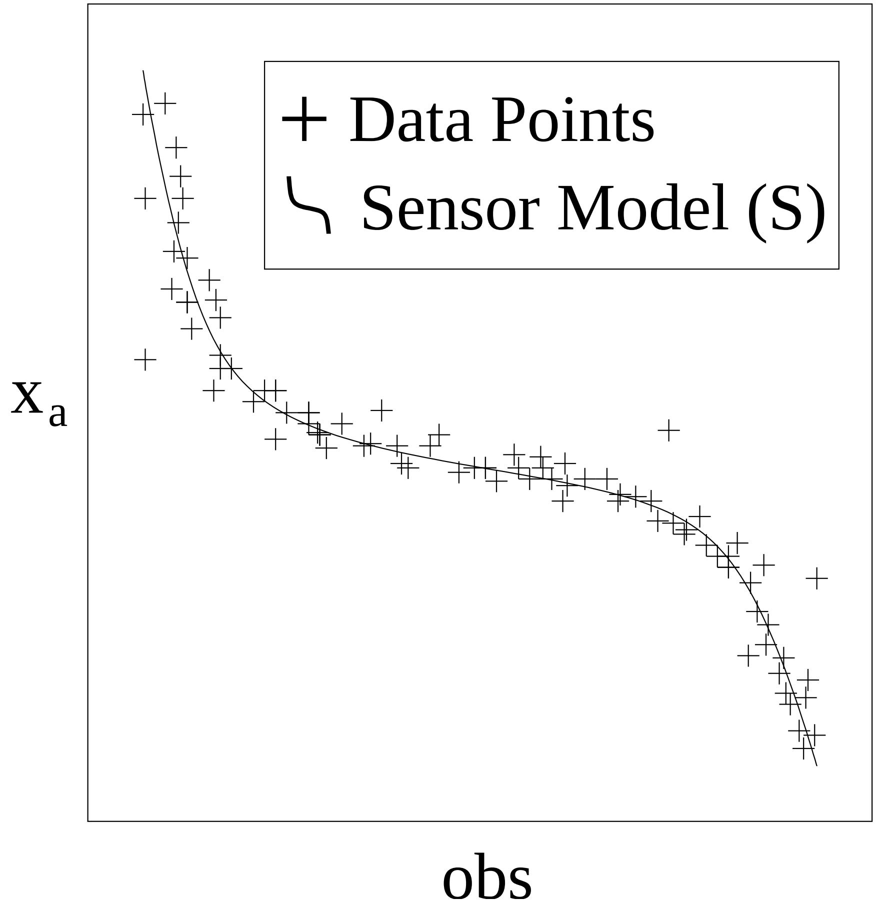
- **Goal:** learn arbitrary continuous functions, A and S
  – Use polynomial regression as function approximator
  – Models learned in arbitrary units

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

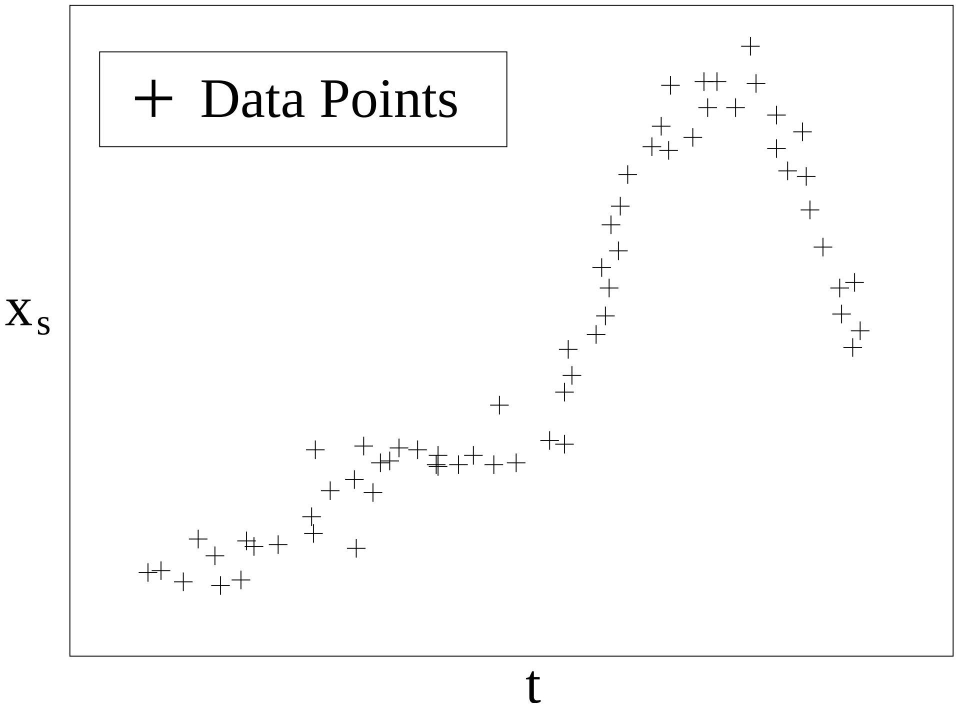


Learning an Action Model

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

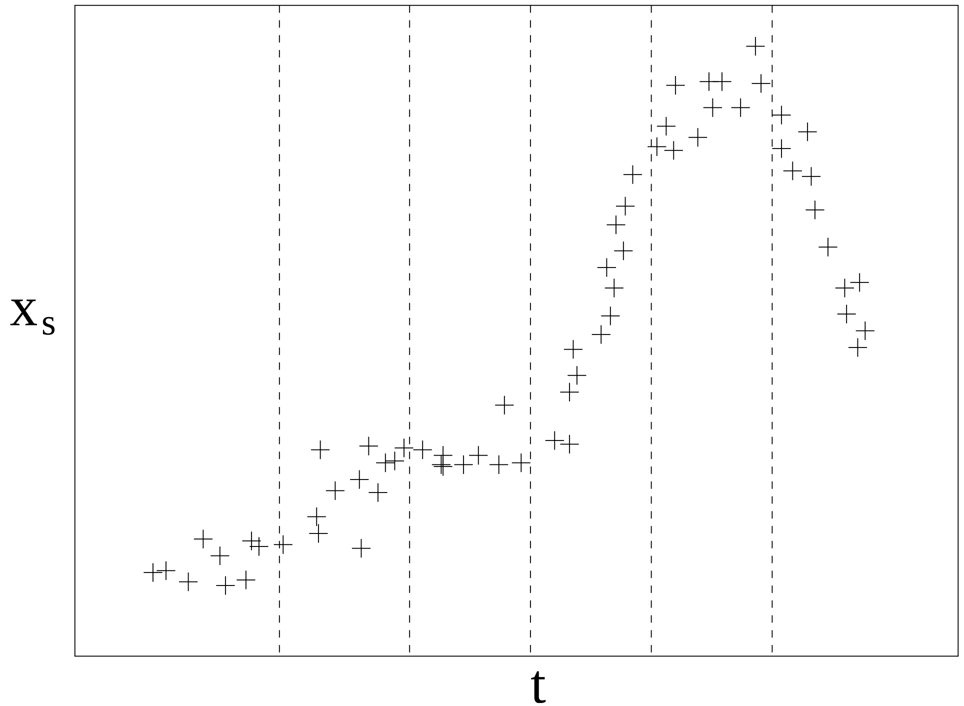
Learning an Action Model

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



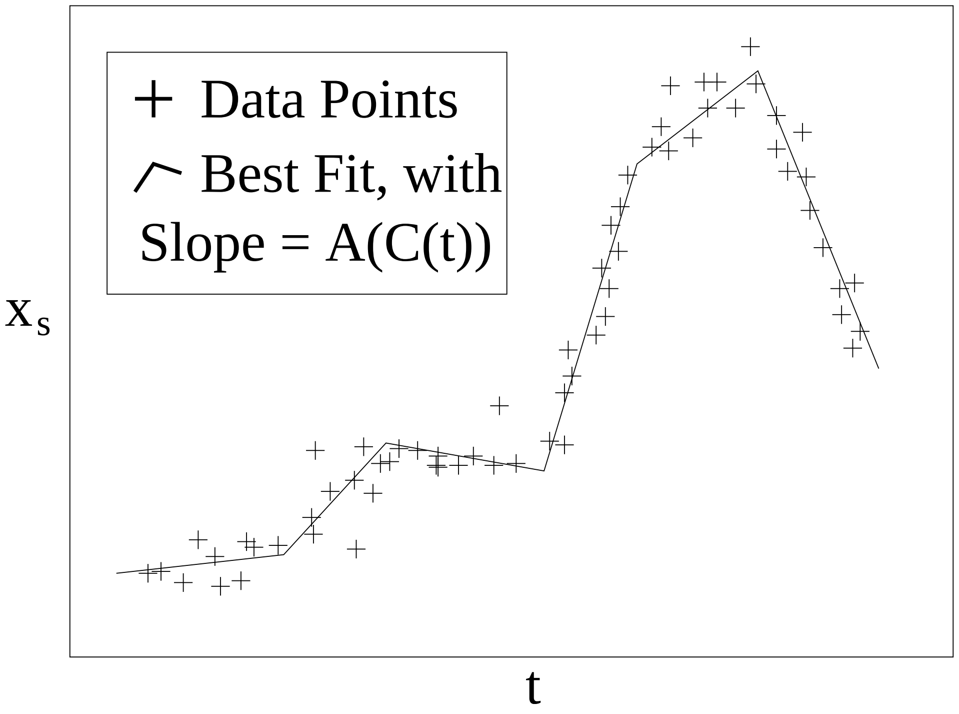
Learning an Action Model

- 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

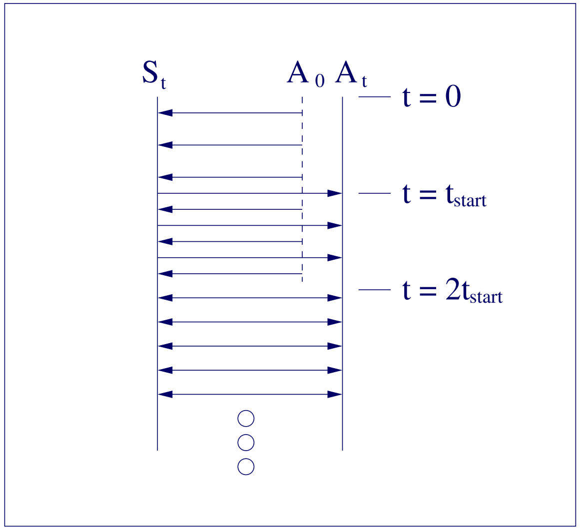
Learning Both Simultaneously

- 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

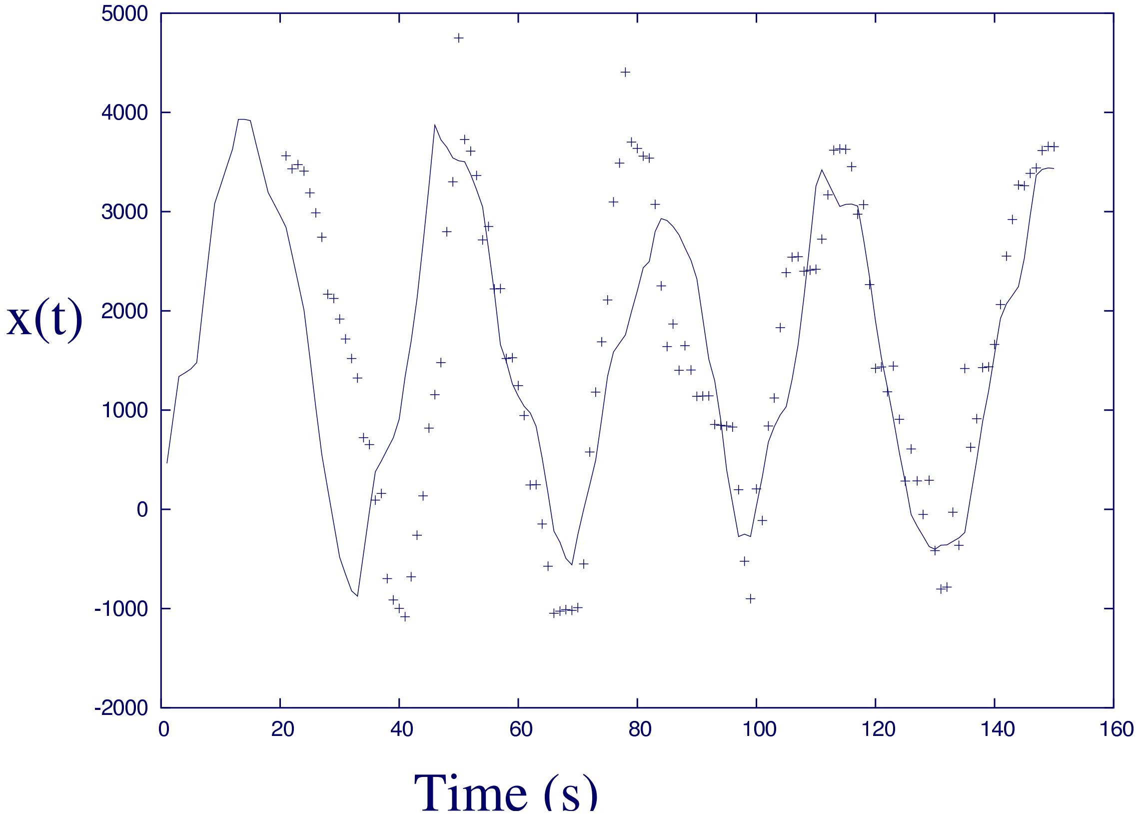
Learning Both Simultaneously

- 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

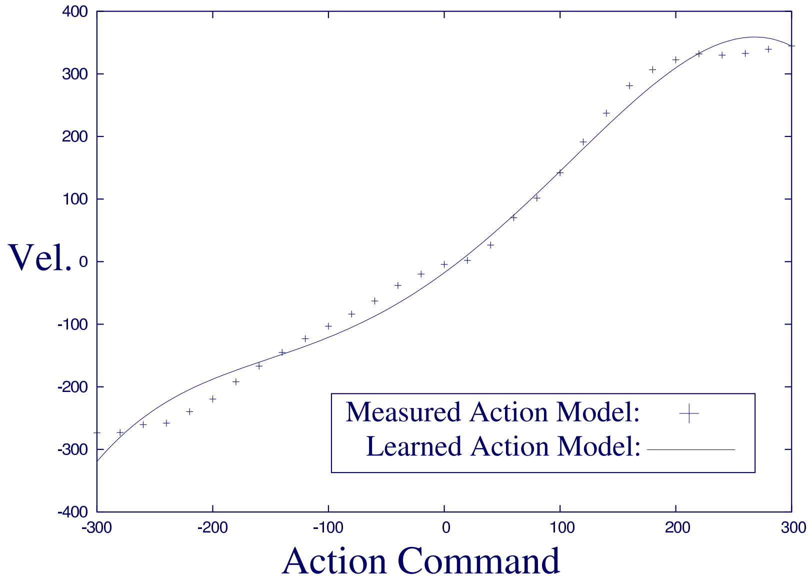
- Run ASAMI for pre-set amount of time (2.5 minutes)
- Measure actual models with **stopwatch and ruler**

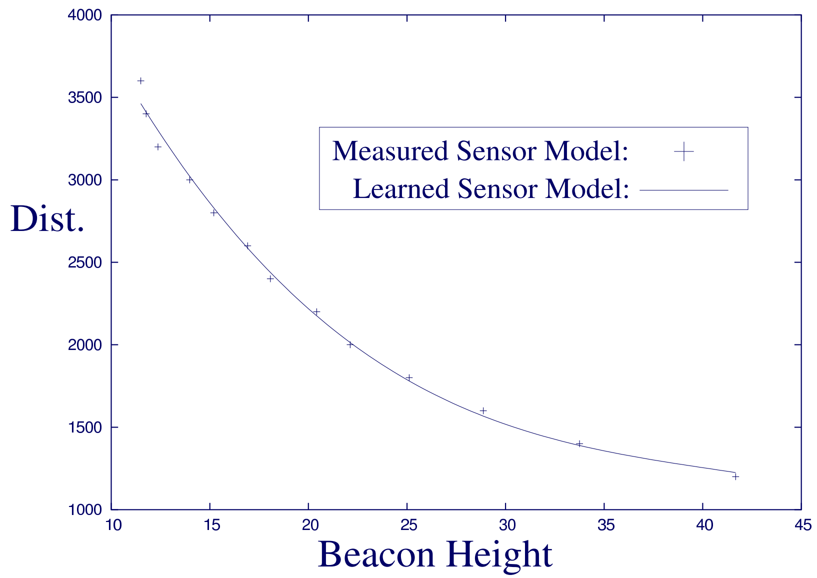
Experimental Results

- 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

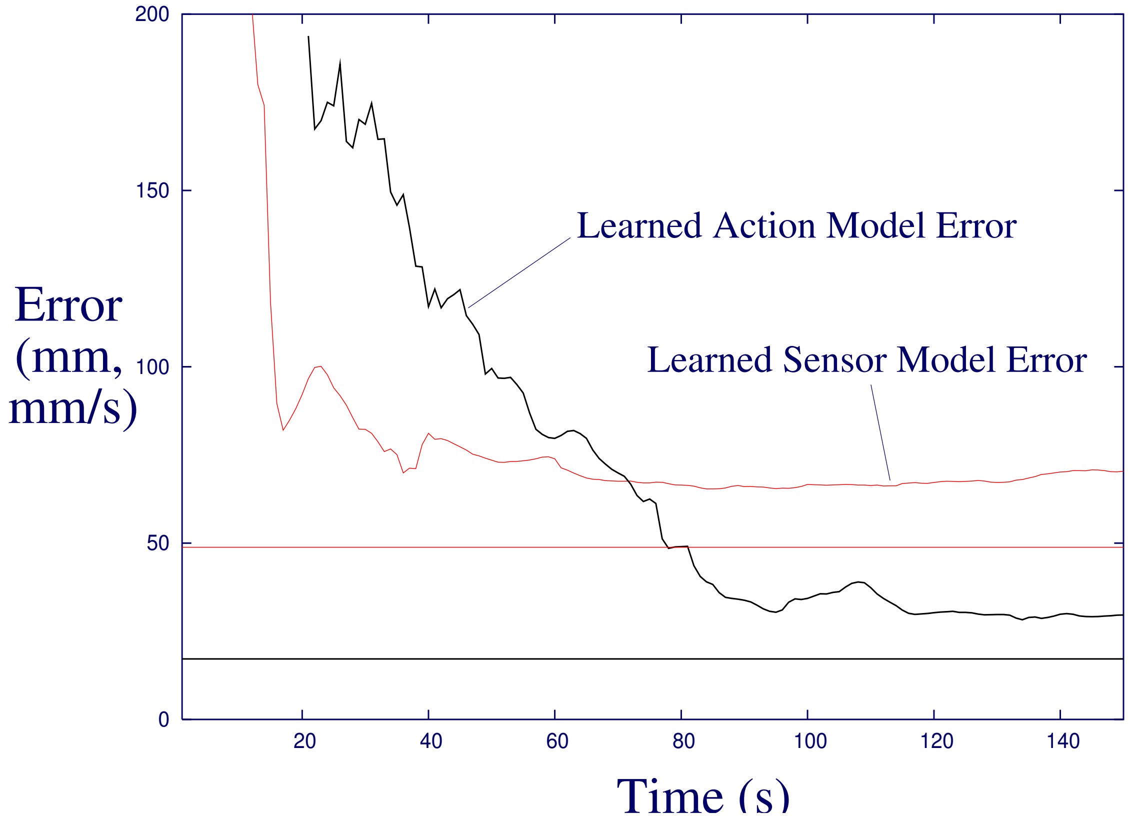
- Run ASAMI for pre-set amount of time (2.5 minutes)
- Measure actual models with **stopwatch and ruler**
- Compare measured vs. learned after best **scaling**





Experimental Results

- Average fitness of model over 15 runs

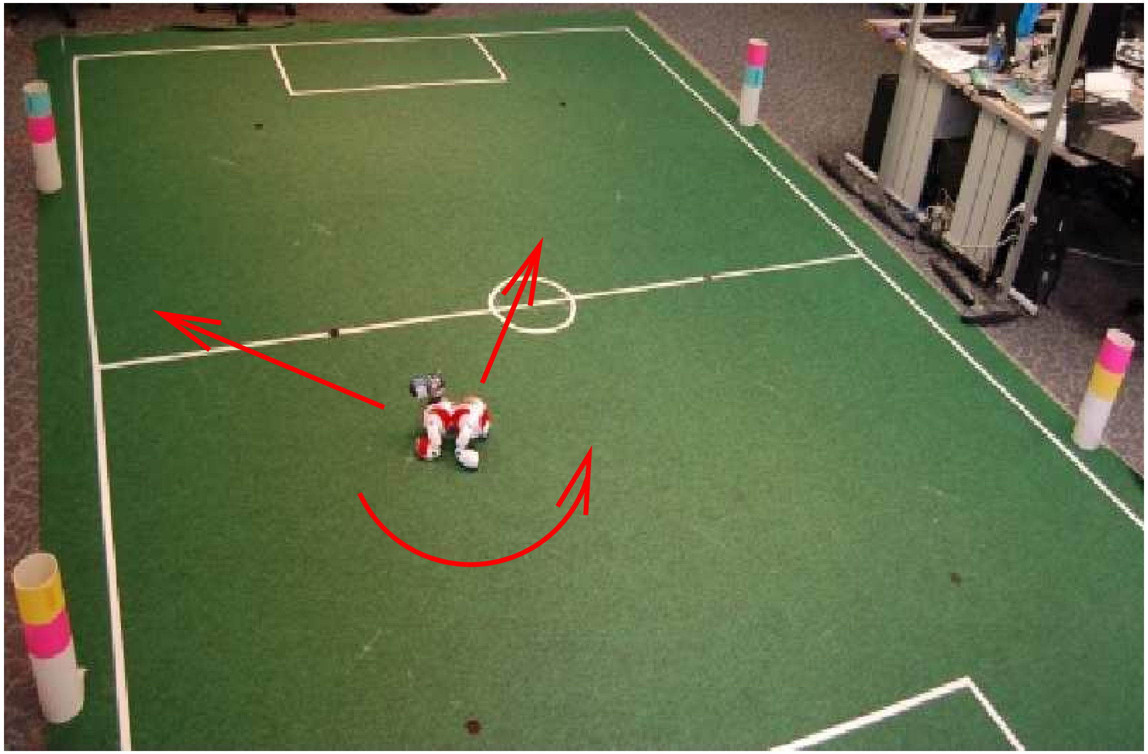


Learning in Two Dimensions

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

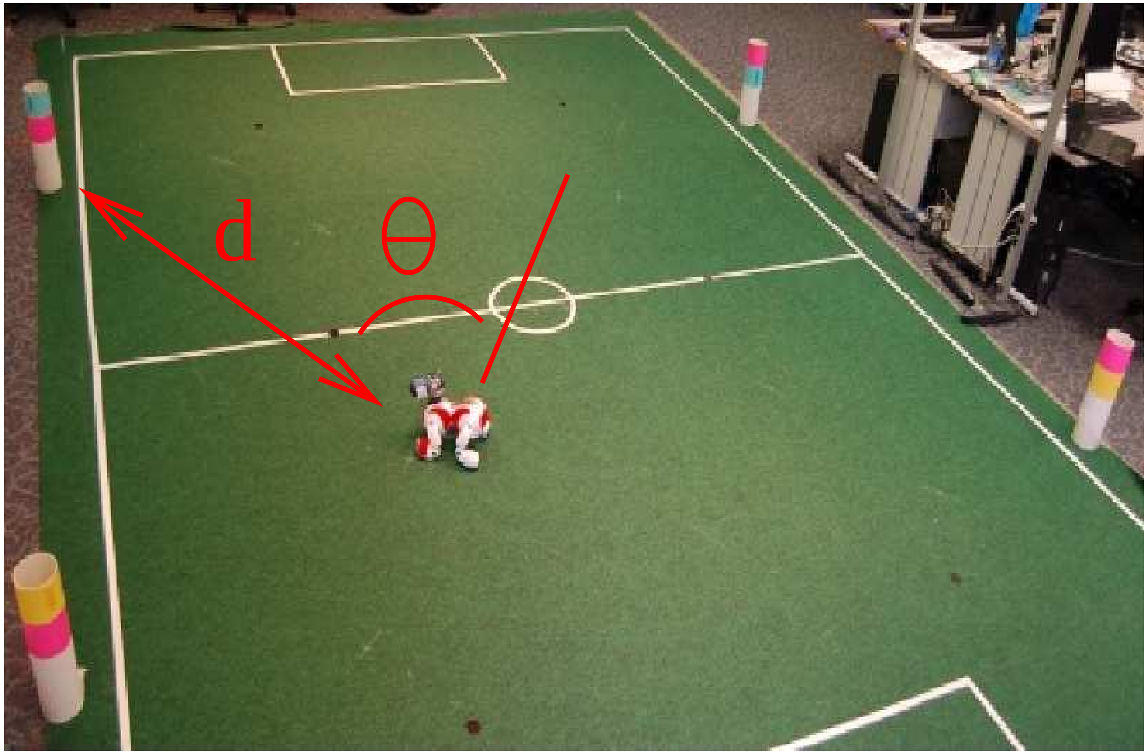
Learning in Two Dimensions

- 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



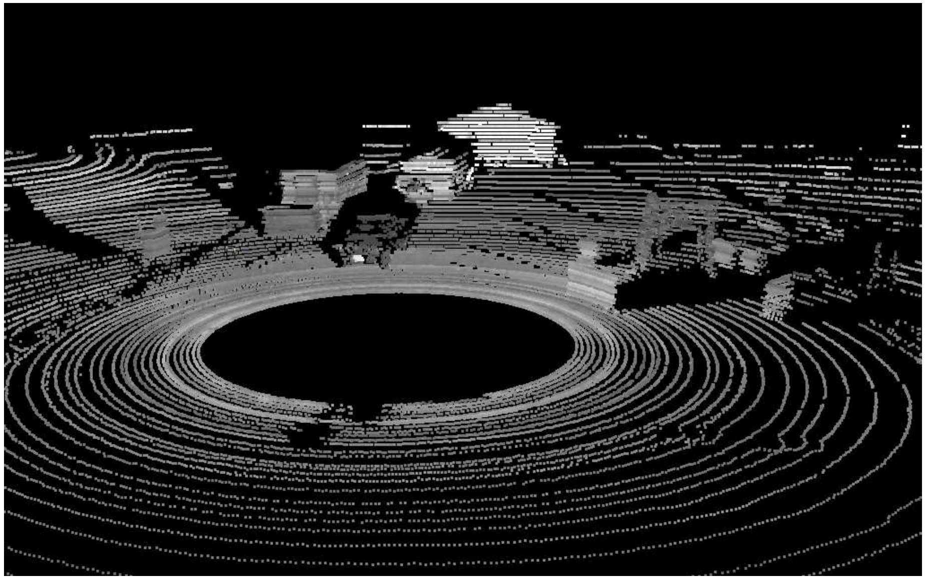
2nd Robotic Platform: Autonomous Car

- **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

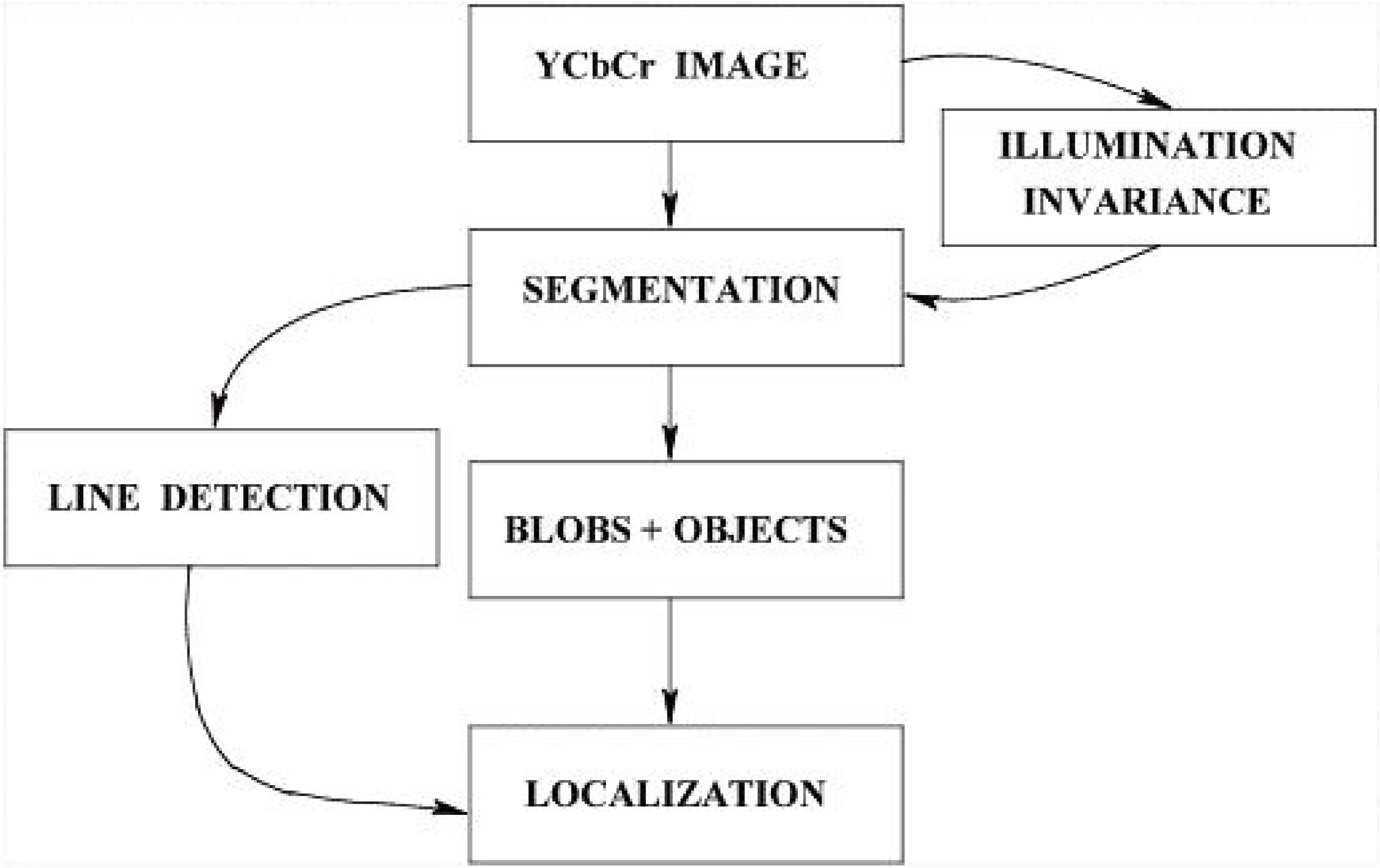
Color Constancy

- Visual system's ability to recognize **true color** across variations in environment
- Challenge: Nonlinear variations in sensor response with **change in illumination**

Color Constancy

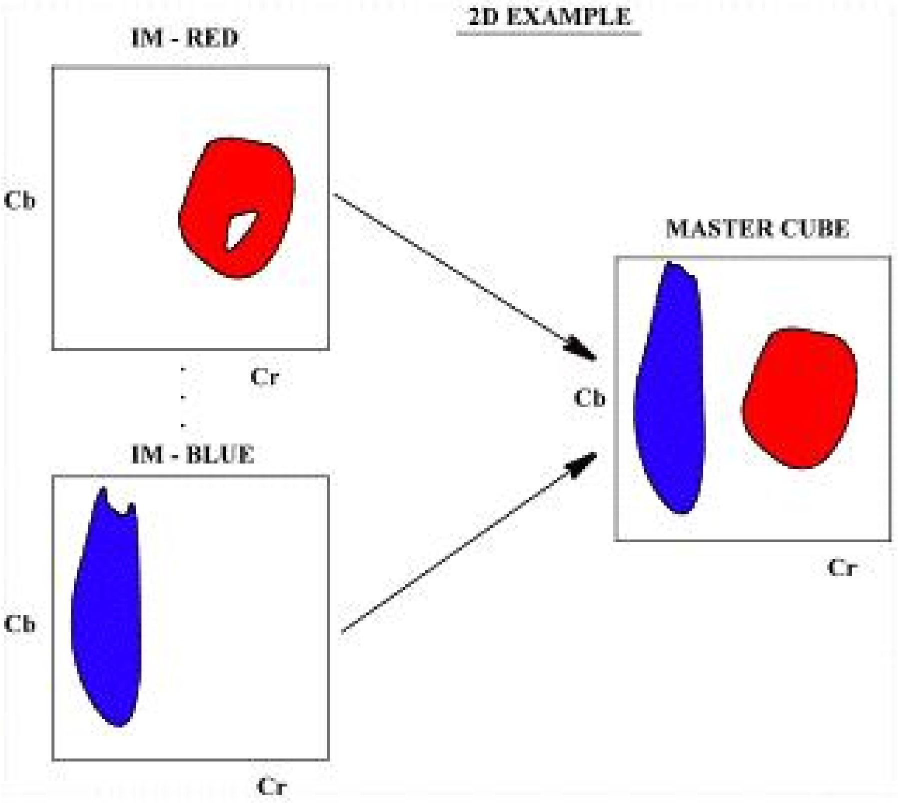
- Visual system's ability to recognize **true color** across variations in environment
- Challenge: Nonlinear variations in sensor response with **change in illumination**
- **Mobile robots**:
  - Computational limitations
  - Changing camera positions

Vision Flowchart



Segmentation

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

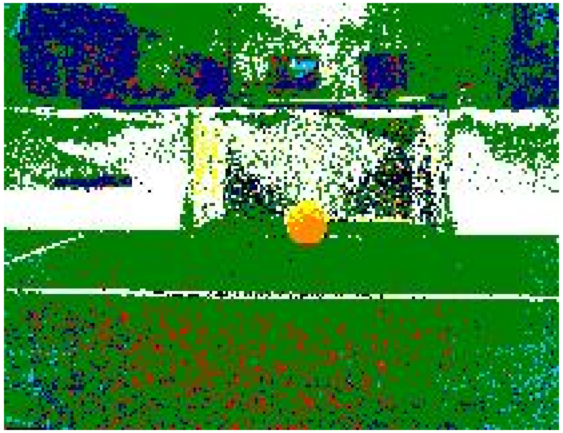


Sample Images

Sample Images

Sample Images

Sample Images



Our Goal

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

Training/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

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 $\mapsto$ 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**: $128 \times 128 \times 128$ 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**
- Markers used for **Localization**
- Higher level strategies and **action selection**

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**
- 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: **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 $\Rightarrow$ similar

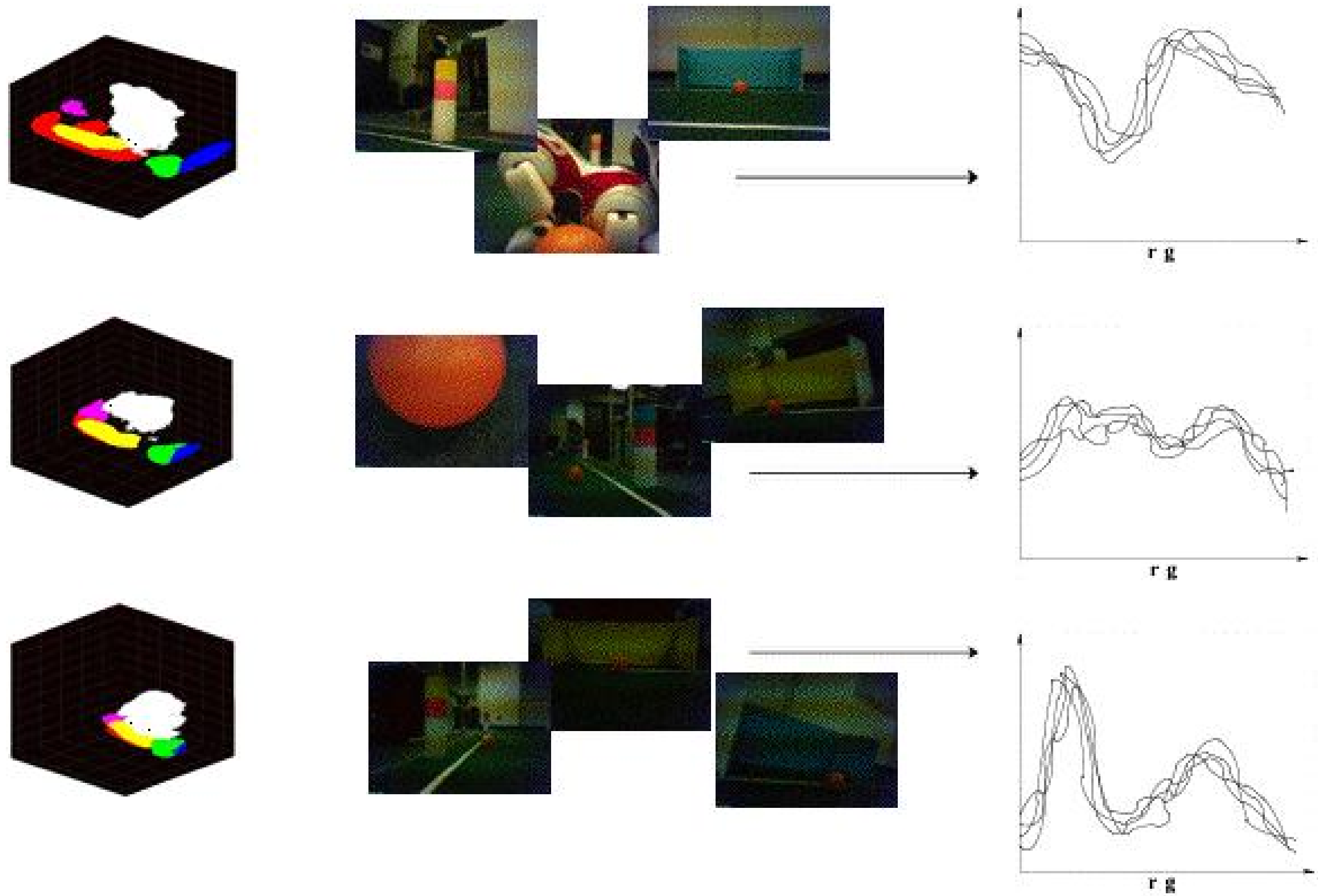
Approach

- 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)

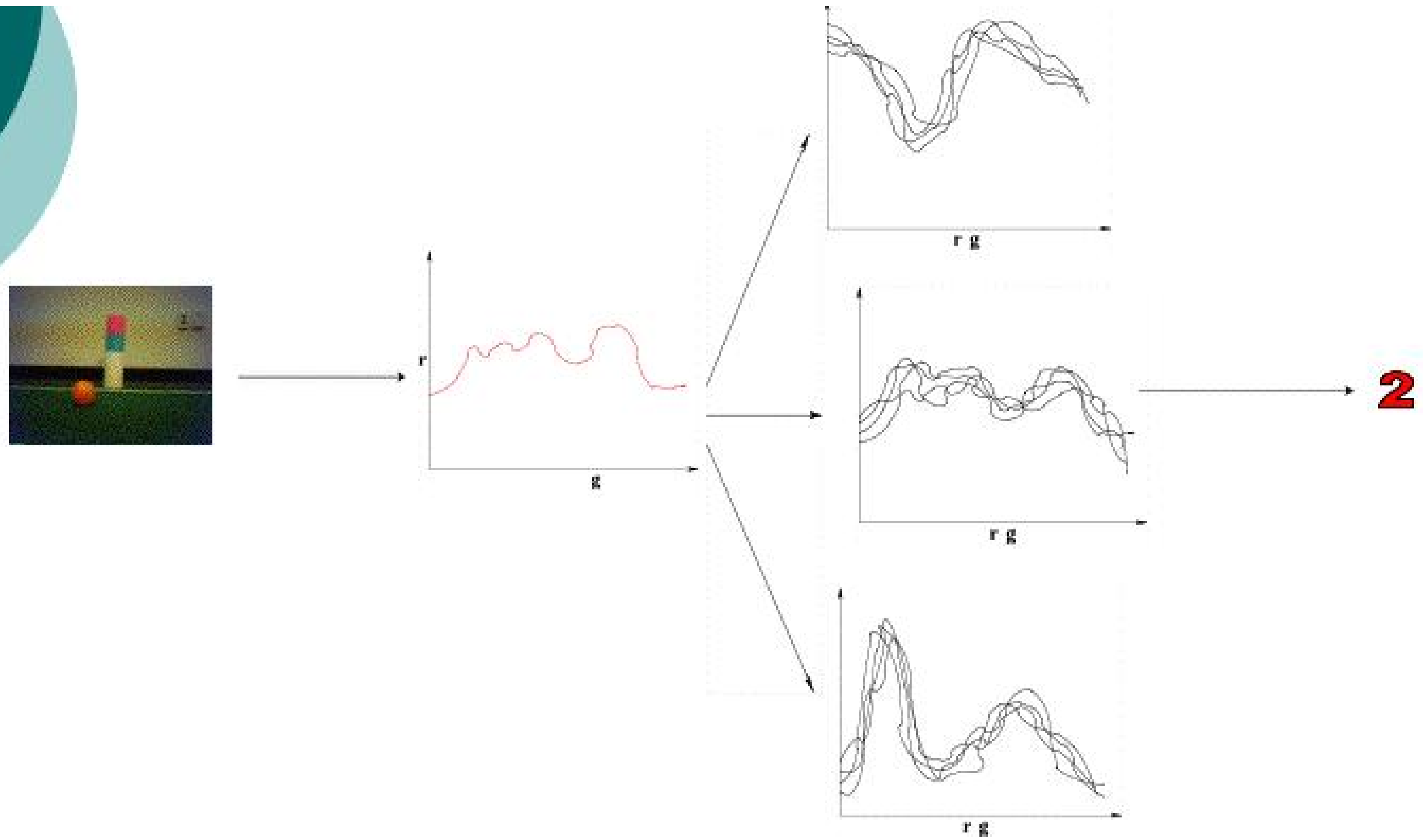
$$KL(A,B) = -\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left(A_{i,j}\ln\frac{B_{i,j}}{A_{i,j}}\right)$$

  - Small value $\Rightarrow$ similar
  - **Robust** to large peaks in observed color distrubutions

Training Phase



Testing Phase



Results

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

Results

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

Lighting transition	Time(sec)
None	15.2 ± 0.8
Bright/Dark	26.5 ± 1.7
Dark/Bright	20.1 ± 2.7

– Also tested **intermediate** illuminations; **adversarial** case

Results

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

Lighting transition	**Time(sec)**
None	15.2 ± 0.8
Bright/Dark	26.5 ± 1.7
Dark/Bright	20.1 ± 2.7

– Also tested **intermediate** illuminations; **adversarial** case

– On **ERS-7**, 3 illuminations $\Rightarrow$ **whole range** of lab conditions

Results

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

Lighting transition	**Time(sec)**
None	15.2 ± 0.8
Bright/Dark	26.5 ± 1.7
Dark/Bright	20.1 ± 2.7

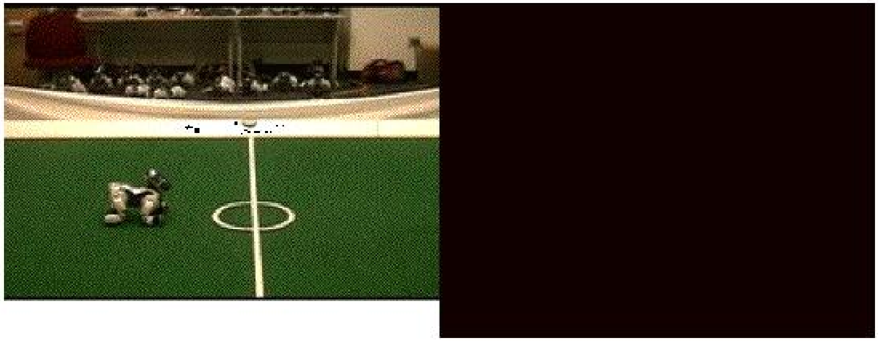
– Also tested **intermediate** illuminations; **adversarial** case
– On **ERS-7**, 3 illuminations $\Rightarrow$ **whole range** of lab conditions
– Works in **real-time**

Autonomous Color Learning

- Color Constancy: **more** tediously created maps
  - Hand-labeling many images $\longrightarrow$ hours of **manual effort**

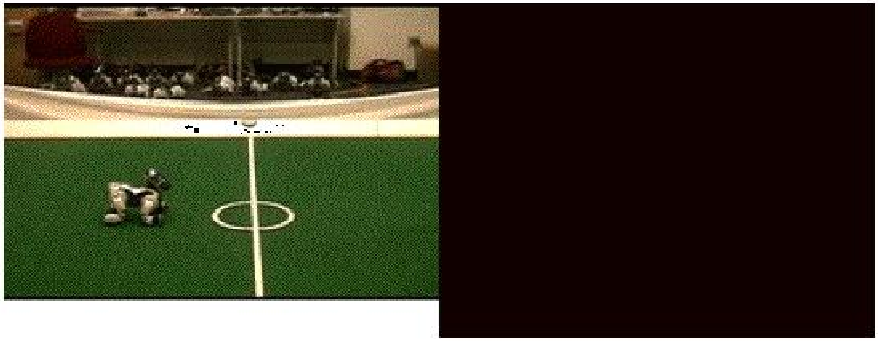
Autonomous Color Learning

- Color Constancy: **more** tediously created maps
  - Hand-labeling many images $\longrightarrow$ 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 $\longrightarrow$ 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!