CMRoboBits:
Hardware, Sensors, Basic Behaviors

Manuela Veloso
Douglas Vail
Scott Lenser

15-491
September 8, 2003

Outline

- Summary of robot hardware - sensors and actuators
- Sensors and AIBO sensors
- Closing the loop - sensors, behaviors
- Basic behavior state machine
- Handling sensor uncertainty
- Example of gsensor data and its use
- Adding a behavior
- Example behavior walk through
- Summary

Robot Hardware Summary

- AIBO ERS-210A built by Sony, commercially available.
- Output/actuators: 18 degrees of freedom with a continuous range of motion and a variety of other output mechanisms including sound and LEDs.
- Input/perception: A variety of sensors, the most important of which are a color CMOS camera, an accelerometer, a infrared proximity sensor, and joint position and load sensing.

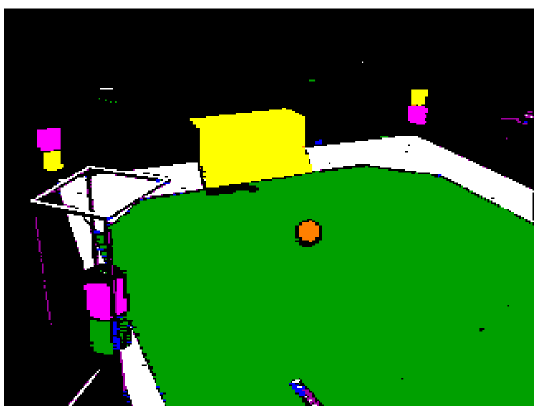
Types of Sensors

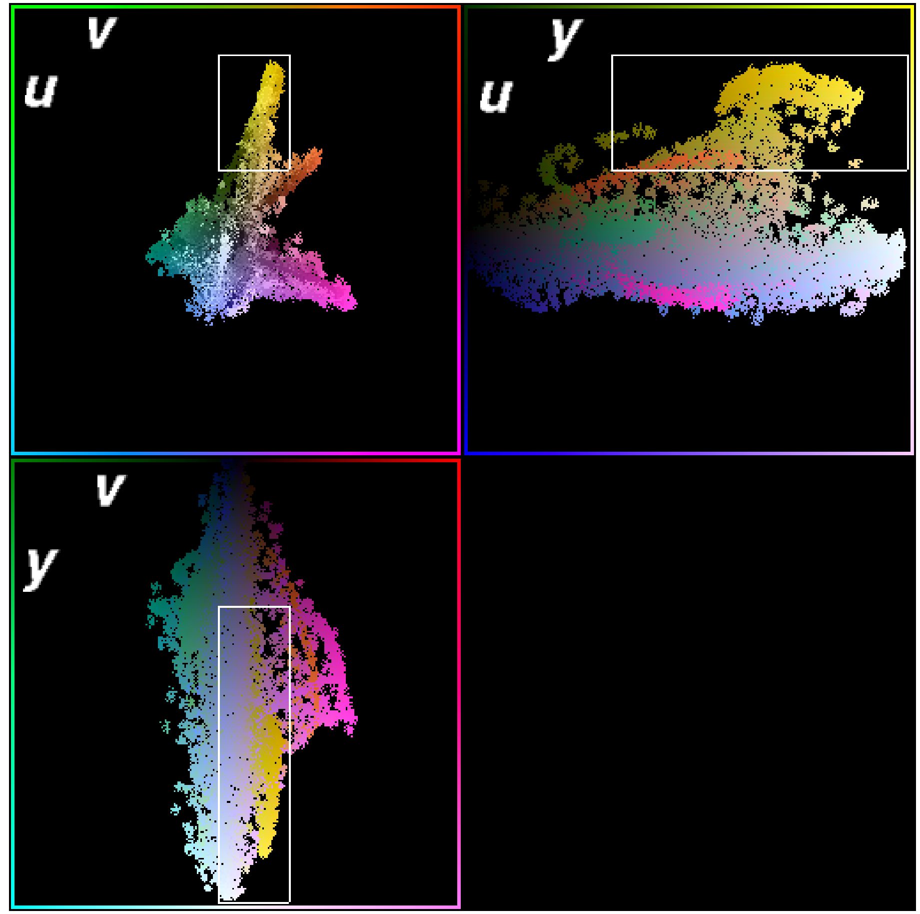
- Passive sensors - capture the environment as is
  - Vision camera, temperature sensor
- Active sensors - emit energy and capture result
  - Sonar: out sound, receives echo, and measures distance from time difference
  - IR: out infrared energy, and measures amount returned

Hardware Sensors - Primary

- Color CMOS camera
  - 176x144 pixel images in YUV color space
  - Use: detect objects in the environment
- Accelerometer to measure accelerations
  - Quite noisy
  - Use: detect if the robot has fallen over
- 4 buttons: 2 pressure sensitive buttons on the top of the head, 2 binary buttons (1 underneath the mouth and 1 on the back)
  - Use: start, pause, explicit input to the robot

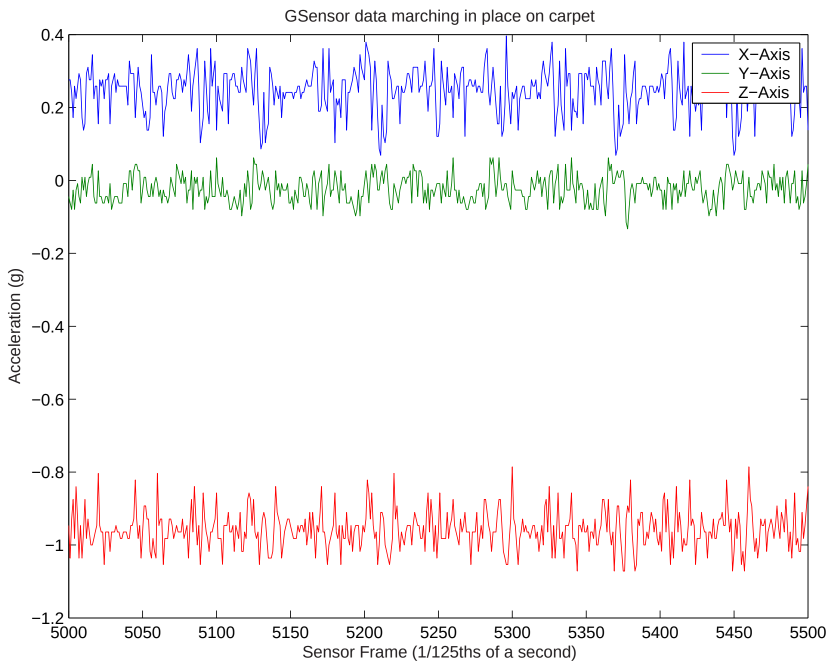
Example: Images, Pixels, YUV Space



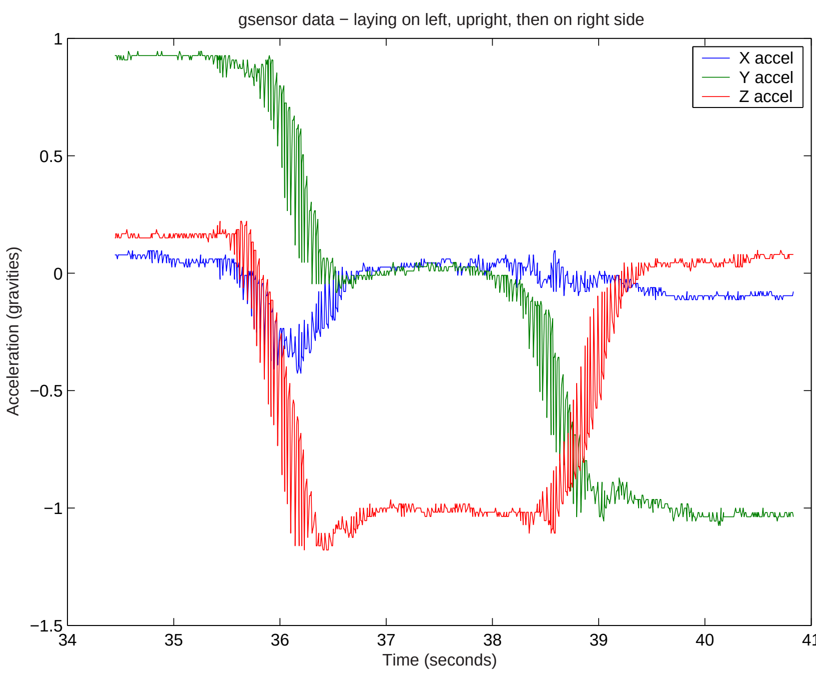


Example: Accelerometer, gsensor Data

Robot standing



Robot held on left side,
then rotated to upright
then rotated to right side down



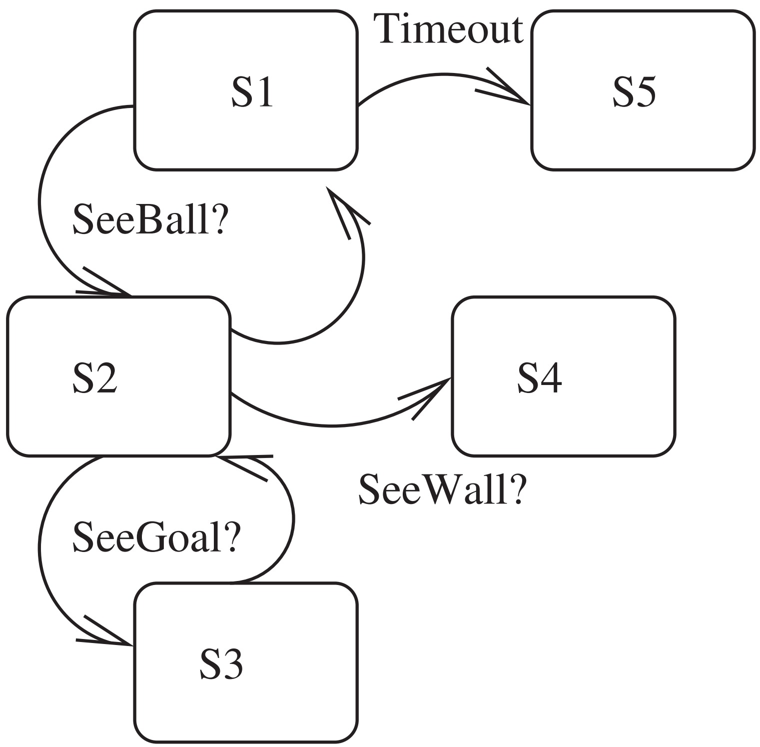
Hardware Sensors - Secondary

- An infrared range detector
  - A highly accurate distance measurement to object in front of robot's head
  - Distances can only be measured between about 10cm and 90cm in front of the robot.
- 4 binary contact switches, one in each paw
  - Use: detect contact with the ground
  - Extremely noisy – often do not fire even when the robot's weight is resting on the paw.
- Stereo microphones on the sides of the head.

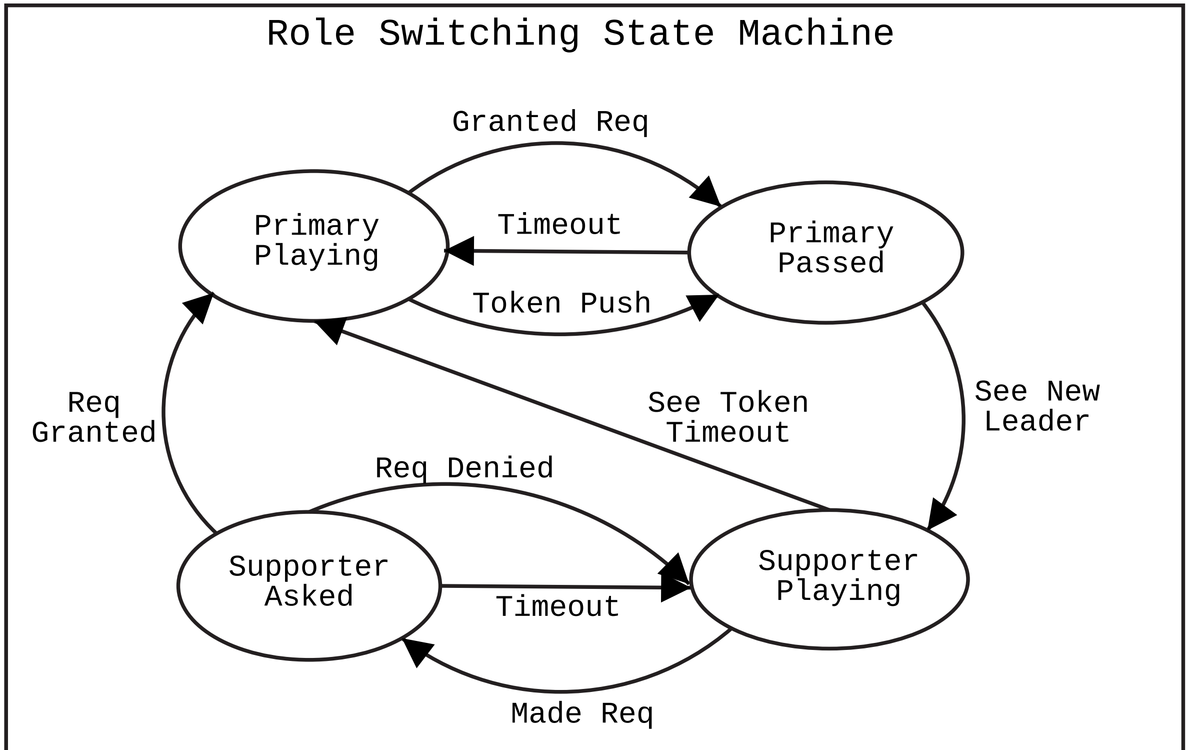
Closing the Loop: Sensors, Behaviors

- Execution is cyclic:
  - Each cycle begins when new sensor data becomes available
  - Each cycle ends when behaviors return a motion command
- We tie the execution cycle rate to the framerate of the vision camera.
  - Ideally, we get 25 frames per second
  - Ideally because slow image processing could reduce the framerate, so effective image processing is needed
  - Other sensors actually have a faster framerate, but vision is strongly tied to behaviors

Basic Sensor/Behavior State Machine



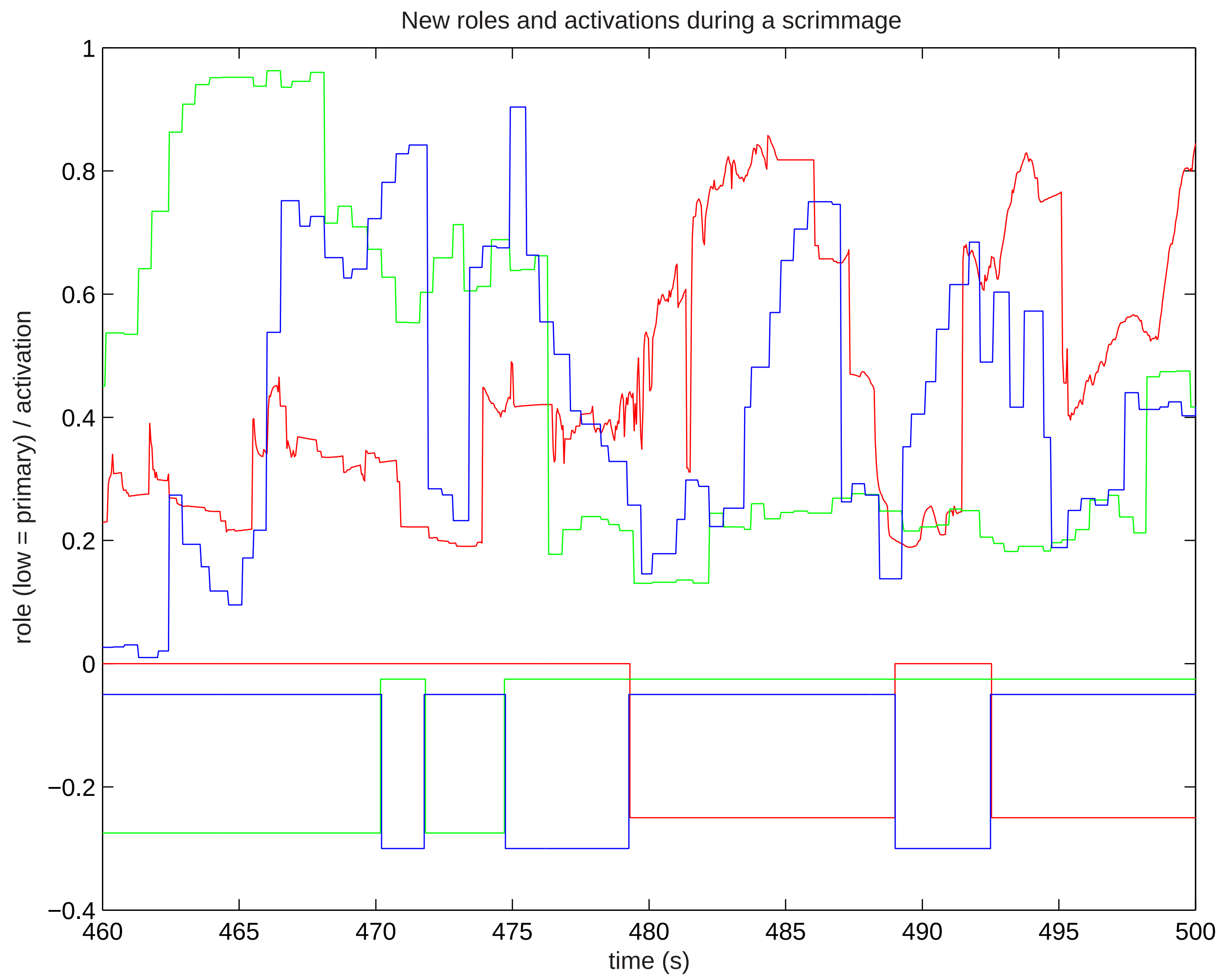
Example: Robot Soccer Role Assignment Behavior State Machine



Role switching is governed by:

MyActivation > LeaderActivation + CostOfSwitching

“Layered” Sensory Data Processing

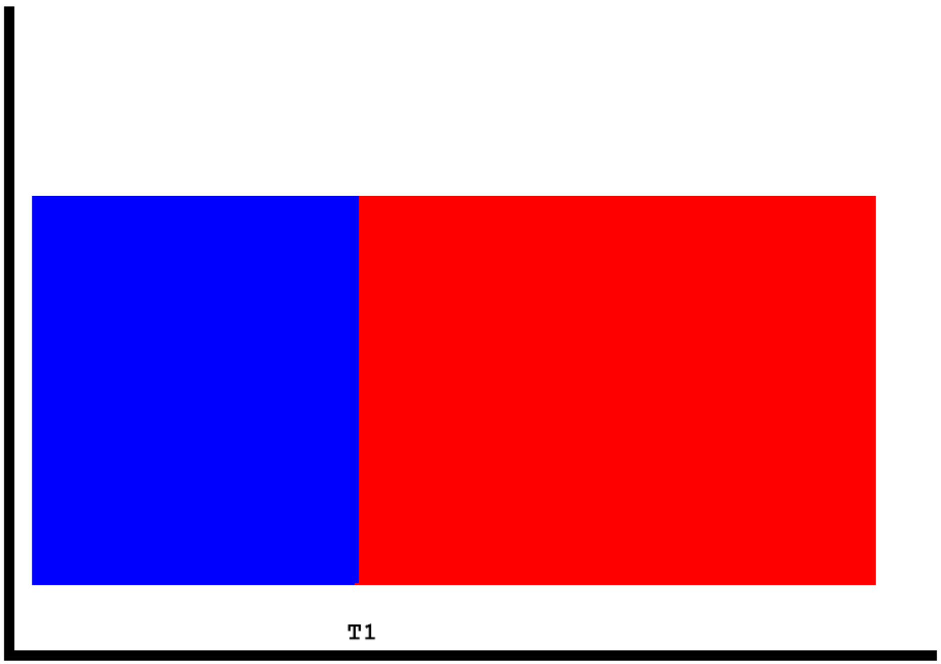


Sensor Uncertainty

- Sensor readinds are noisy
- Signal processing
- Sensor modeling

Handling Sensor Uncertainty - Thresholding

Sensory data is noisy, but decisions need to be made from the sensory data (e.g., SeeBall? CloseToWall?) One method: Thresholding.



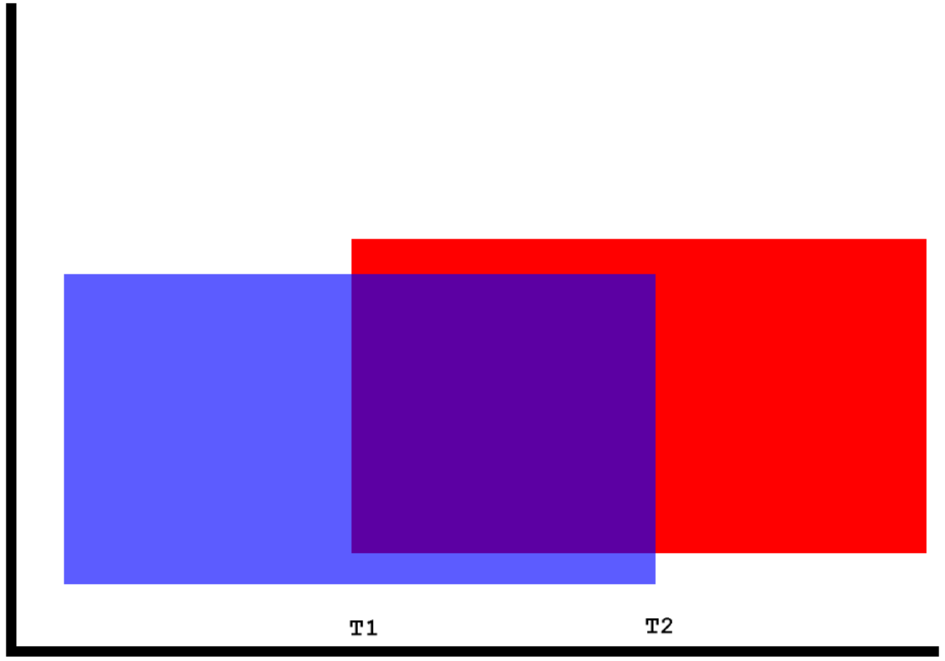
Handling Sensor Uncertainty - Single-Value Thresholding

- Using a single threshold returns two values and leads into a transition between two states.
- Is this a good idea with noisy sensors? Not really. Thresholding with noisy data on a single value may lead to behavior switching oscillation.

Handling Sensor Uncertainty - Hysteresis

- We need a method that will *wait* to switch between behaviors until it is *sure* that the transition between two states should occur.
- A new method: Hysteresis, *The lagging of an effect behind its cause (dictionary.com)*

Multiple Thresholding - Hysteresis



- The state switches from blue to red when values rise above T2. It switches from red to blue when values fall below T1.
- Hysteresis is an [often] better way of thresholding.
- Hysteresis leads to [some] latency in state changing.

Handling Sensor Uncertainty - Averaging

Exponential averaging combines multiple sensor readings and then decisions are made thresholding on the computed average.

$$avg_{n+1} = \alpha \cdot avg_n + (1 - \alpha) \cdot new\_value$$

Hysteresis and averaging represent a trade off between latency and noise tolerance.

Handling Sensor Uncertainty: Learning

- Sensor data can be used to learn specific states.
- E.g., an AIBO can learn to distinguish between different types of carpet or a table top *based on a window of gsensor data.*

Learning procedure:

- Gather some time (e.g., 3mns) of sample sensor data (e.g., gsensor) per state (e.g., on carpet, on table top)
- Generate “training examples”
  - Extract “good” features for the task (see next slide)
  - Create a training example as feature values and label
- Pass this training data to a supervised learning algorithm
- Run the trained classifier on the robot to classify new examples

Example: Gsensor Features for Learning

We use the variance in x, y, and z accelerations as well as the (x, y), (x, z), and (y, z) correlation coefficients over a 1 second window of data as features for a decision tree learner.

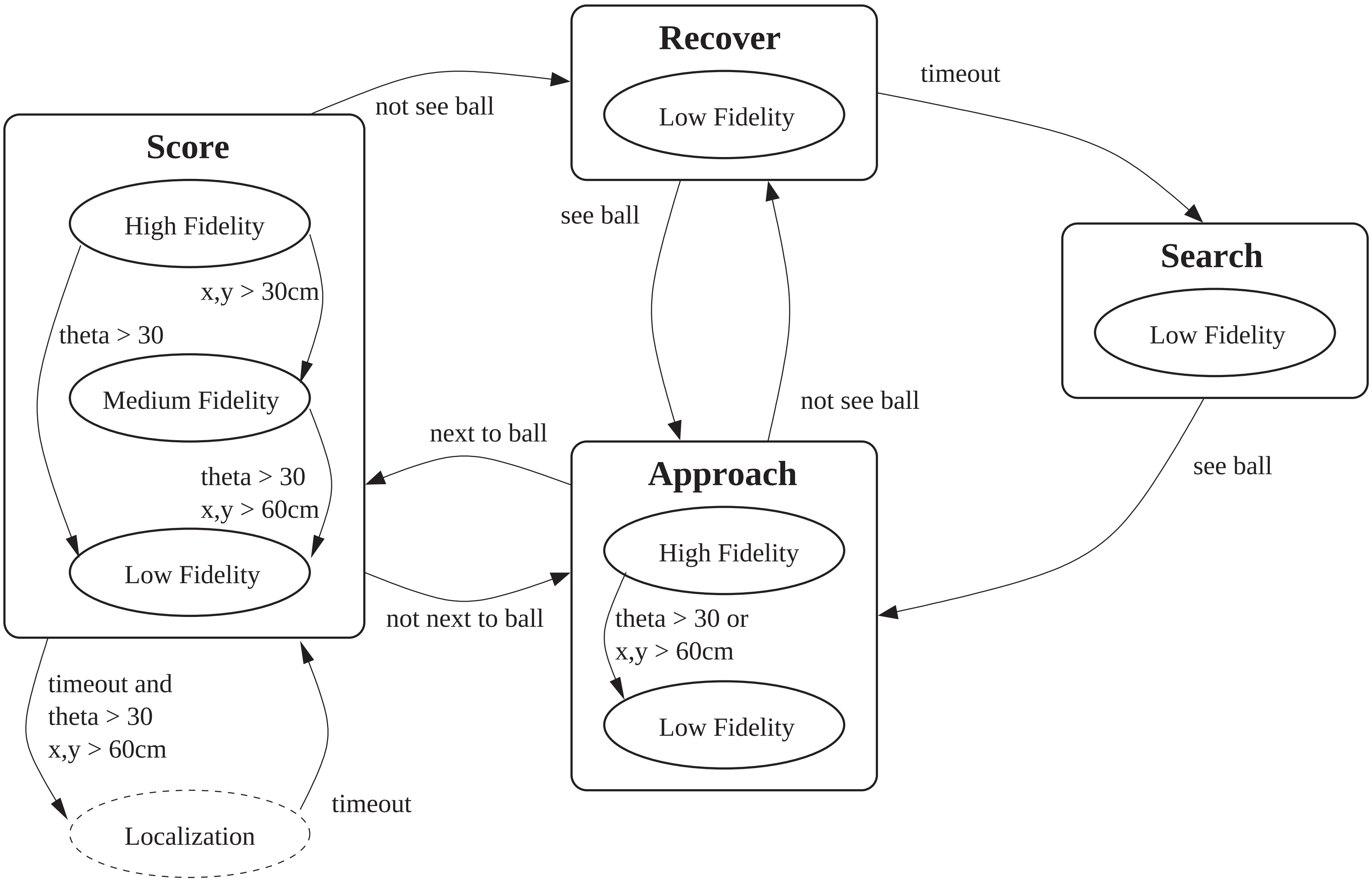
Feature vector:

$$< var(x), var(y), var(z), cor(x, y), cor(x, z), cor(y, z) >$$

$$var(x) = \frac{\sum_{i=1}^{125} (x_i - \overline{x})^2}{n-1}$$

$$cor(x, y) = \frac{\sum_{i=1}^{125} (x_i - \overline{x}) \cdot (y_i - \overline{y})}{\sqrt{var(x) \cdot var(y)}}$$

Reasoning About Sensor Uncertainty: Multi-Fidelity Behaviors



Multi-Fidelity Behaviors

Approach:

Low:

```
Run straight towards the ball.
```

High:

```
\item Skew approach to ball to get behind it
facing the goal
```

Score:

Low:

```
Until the robot sees the goal,
  Walk sideways around the ball.
Walk forward pushing the ball.
```

Medium-High:

```
Circle ball using shortest distance
If facing goal, push ball forwards.
```

AIBO Software Major Layers

- Aperios - OS, real-time, proprietary, developed by Sony
  - Organized as a set of cooperating processes, called objects in Aperios, which communicate via message passing
  - Provides system services

- OPEN-R - A set of routines and data structures used for controlling the robot, developed by Sony
  - Provides the basic interface to the robot hardware

Robot Development

- Aperios, OPEN-R, and related tools are freely available from Sony and are distributed as the OPEN-R SDK. The software runs on both Windows and Unix derivatives.

- Application - Additional processes, Aperios objects, can be created using the OPEN-R SDK to perform application specific functions.

Aperios Object Interactions

- Each object communicates with other objects by sending messages containing data, via shared memory regions.

- Message passing uses a hand-shake protocol.
  - Observer: indicates readiness to process messages by sending a ready message to the sender.
  - Sender: sends messages to observers when both the message and the observer are ready.
  - If the observer is not ready when the message is ready, the sender can either drop the message (normal case) or buffer it for later transmission.

Software Robot Interface

- The main system object: OVirtualRobotComm.
- Sensor values are obtained by subscribing to sensors with OVirtualRobotComm. After this is done, OVirtualRobotComm will send messages containing the sensor values.
- Most of the actuators are controlled by sending messages to OVirtualRobotComm. [We will talk about actuators and motion in the next class.]

Sensors

- Images are sent at the rate of 25Hz.
- Stereo microphone data is sent as buffers of data containing data sampled at up to 16KHz.
- General sensor data is sent together on one connection. Sensors are sampled every 8ms (every motor frame). Sensor readings are buffered and sent in messages containing data from 4 motor frames.

General Sensors

The following sensors are included in the general sensor reading message (all sensors reported in 8ms increments every 32ms):

- Sensed joint angles for every joint.
- PWM duty cycles for the motor actuating each joint. These values range -512 to 512.
- Accelerometer readings. These readings are quite noisy.
- Foot pad switch readings. These are the binary switches in the feet. These readings are very noisy.
- IR range reading. This reading is very accurate over the range of 10cm to 90cm.
- A temperature reading.

Code Walk Through

- agent/Motion/MotionInterface.h
- agent/Main/Makefile
- agent/Behaviors/SpinDog.h
- agent/Behaviors/SpinDog.cc

Summary

- In general, there is a large variety of robot sensors.
- The AIBO robot used in the class has a specific set of sensors.
- Sensor readings are input to the "robot program" (cognition).
- Behaviors are state machines.
- Behaviors focus sensory signal processing.
- Sensors have uncertainty that needs to be effectively handled, such as with hysteresis.
- Sensors values can be used to learn to classify the percepts.
- Behaviors can be sensitive to and reason about sensor uncertainty, such as multi-fidelity behaviors.
- The AIBO runs Aperios and OPEN-R. We provided and presented code to write a simple robot behavior.