See, AIBO; Run!

AIBO Motion and Vision Algorithms

Team Advisor: Ethan J. Tira-Thompson
Teaching Assistant: P. Matt Jennings
Team Members: Haoqian Chen,
Elena Glassman, Chengjou Liao,
Yantian Martin, Lisa Shank, Jon Stahlman

AIBO:

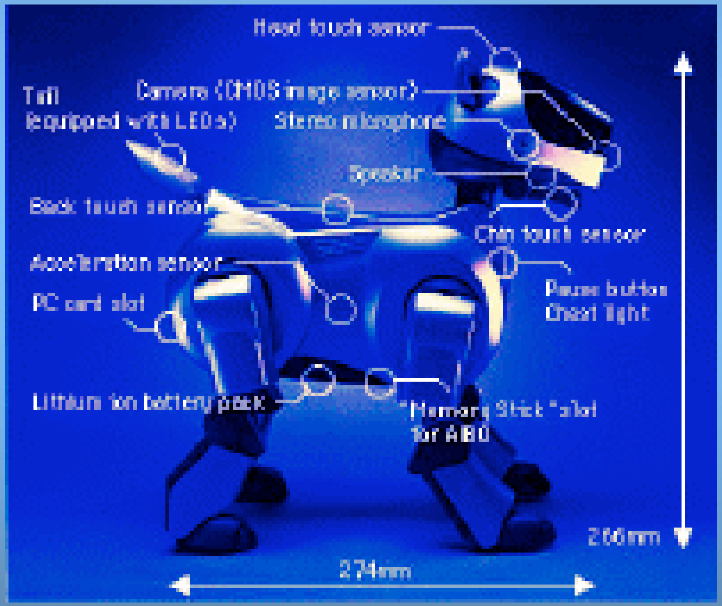
The Dog…
The Legend.

History of AIBO

- Sony’s Entertainment Robot
- AIBO – Artificially Intelligent roBOt
- Or aibo – Japanese for “companion”
- Originally intended for purchase by home users
- Found to be a relatively cheap, versatile platform that could be used by educators and researchers

AIBO: Technical Specifications

- 64 bit Processor
- 20 Degrees of Freedom
- Microphone
- Accelerometer
- Infrared Distance Sensor
- Pressure Sensors
- The Kitchen Sink



Tekkotsu

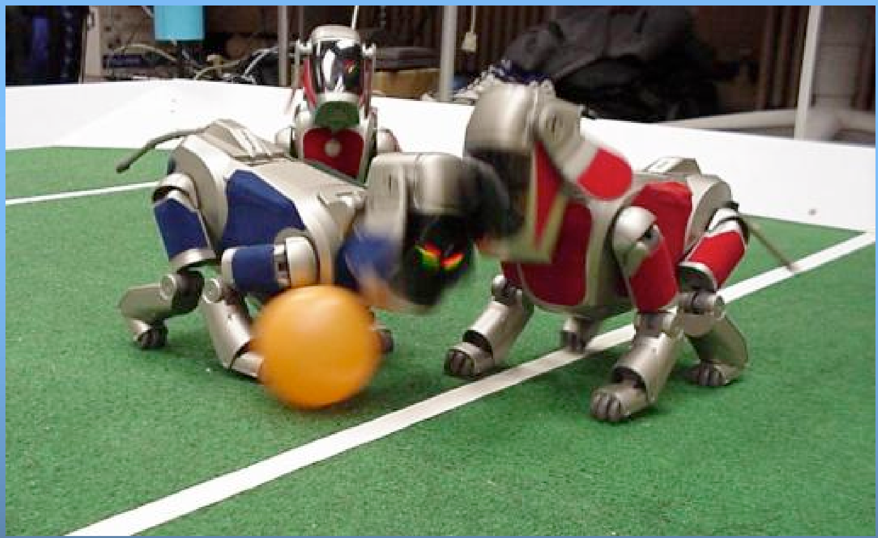
- Application framework for AIBO
- Under development at CMU's Robotics Lab
- TekkotsuMon - server-side interface to code running on robot
- To accomplish our goals, we built upon Tekkotsu platform
- Our advisor, Ethan Tira-Thompson, is a chief researcher for the project

Run, AIBO, *Run!*

- Originally attempted to stabilize image from AIBO's camera
  - Software methods
  - New walking motions
- Modified walk parameters
- Measured performance using accelerometers
- Stability vs. Speed

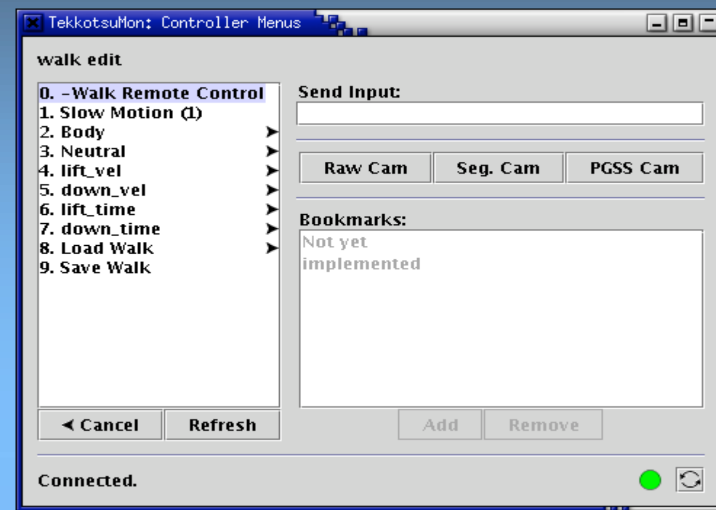
How Our Walk Was Developed

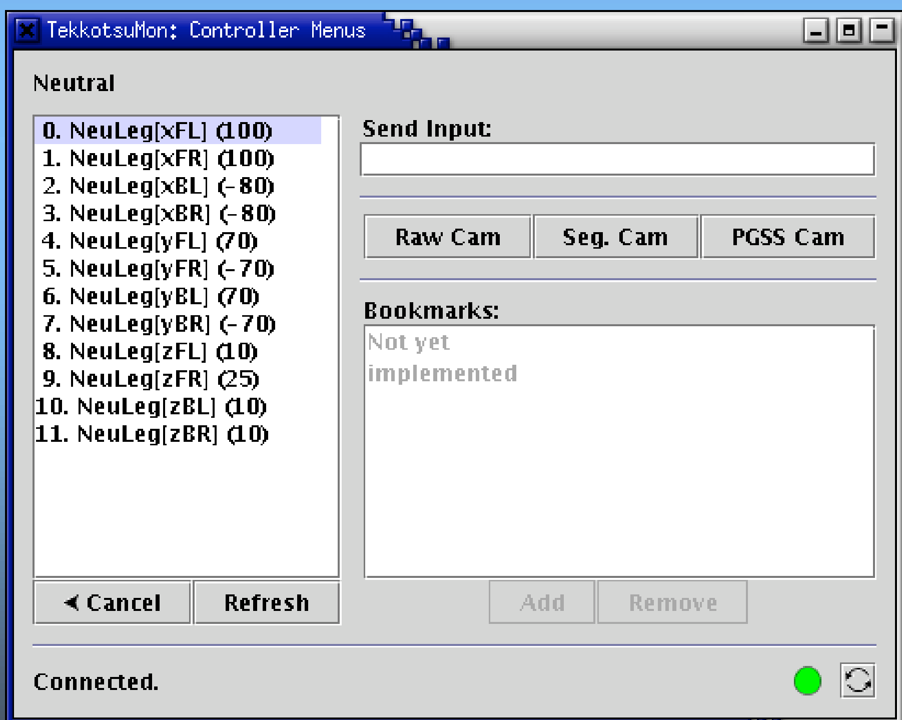
- Tekkotsu's default walk is taken from CMU's RoboCup soccer team
- Our new motion was created by modifying this walk's parameters

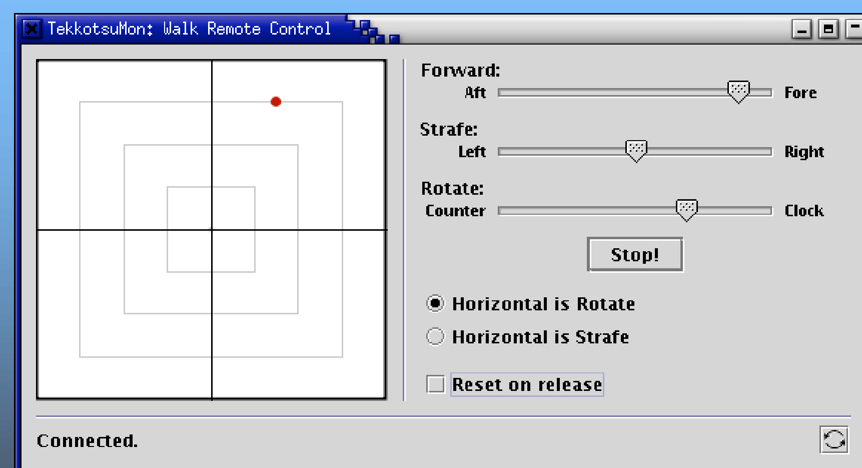


Get Your Move On!

- Using the TekkotsuMon GUI

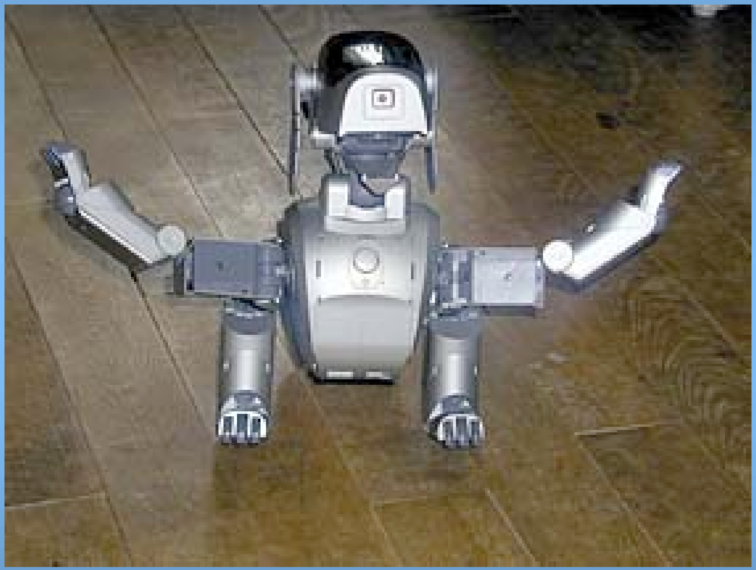






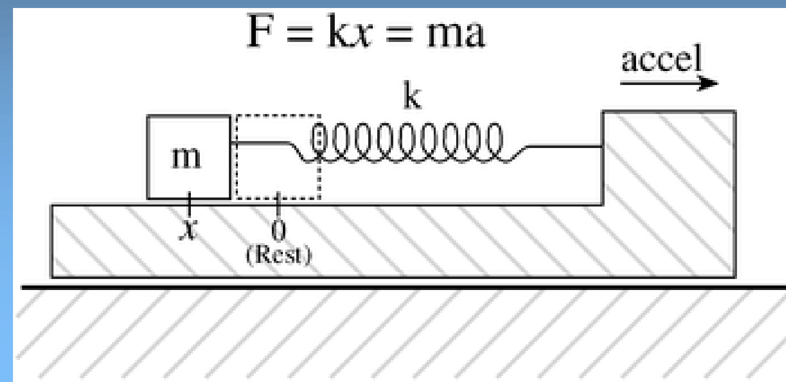
Changing Things Up: The Parameters

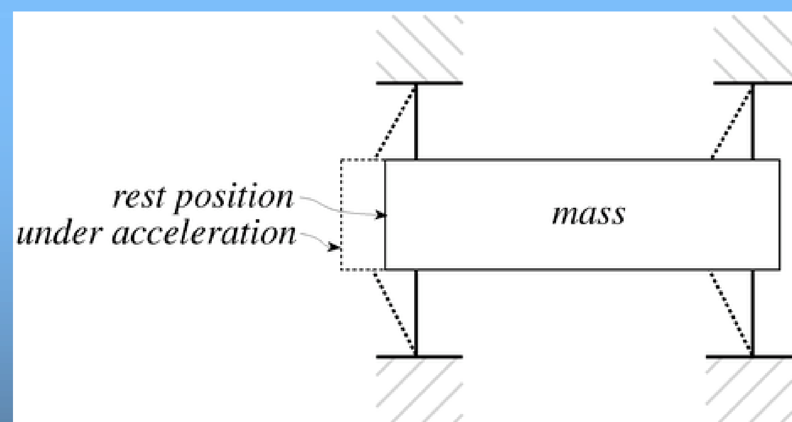
- Lift Velocity
- Down velocity
- Lift Time
- Down time
- Body Height/Angle
- Period
- Position Coordinates
- Hop and Sway



Accelerometers

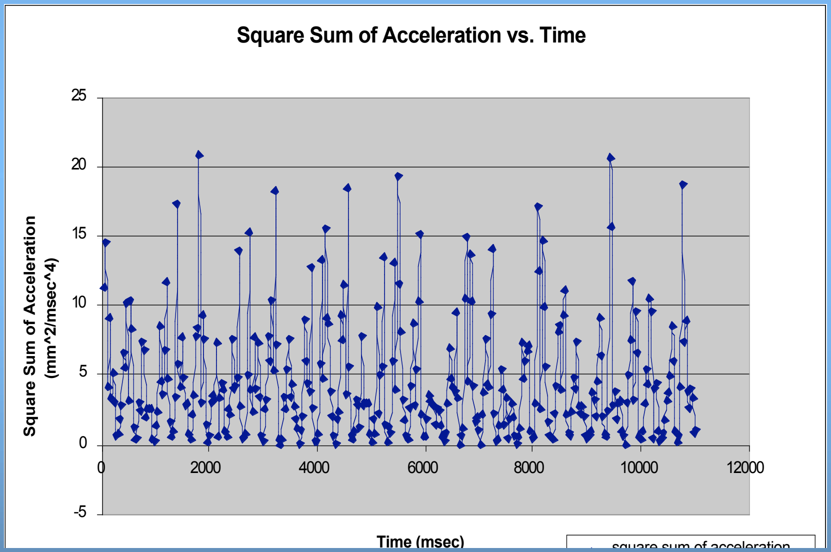
- Each consists of a mass/spring system
- Measure force and displacement on joints as robot walks
- We graphed the way force varies with time to evaluate stability of new walking algorithm



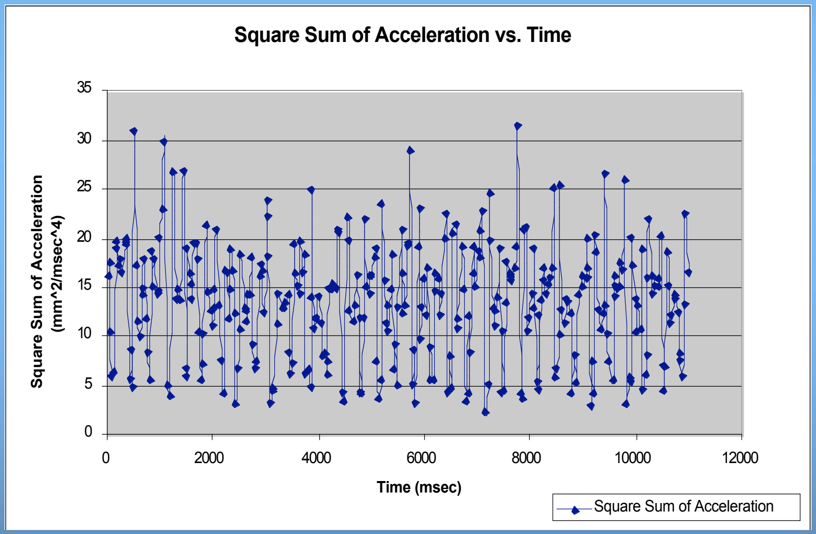


Our Upright Walk
vs.
CMU RoboCup Walk

- Upright



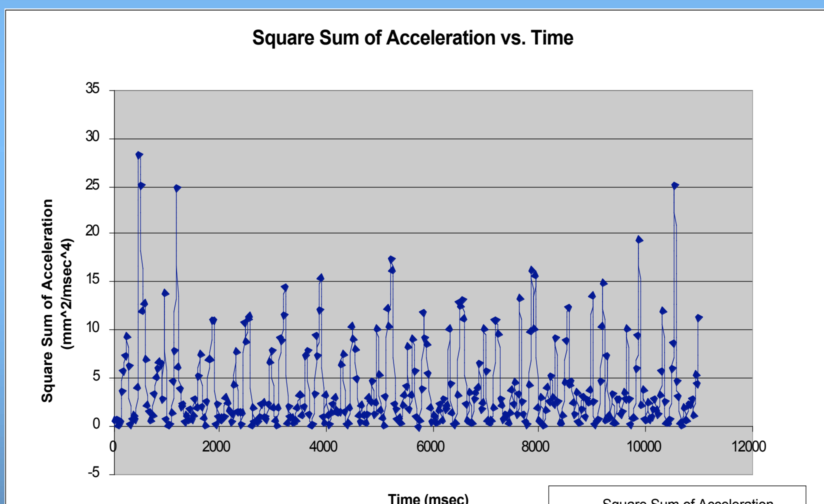
- RoboCup

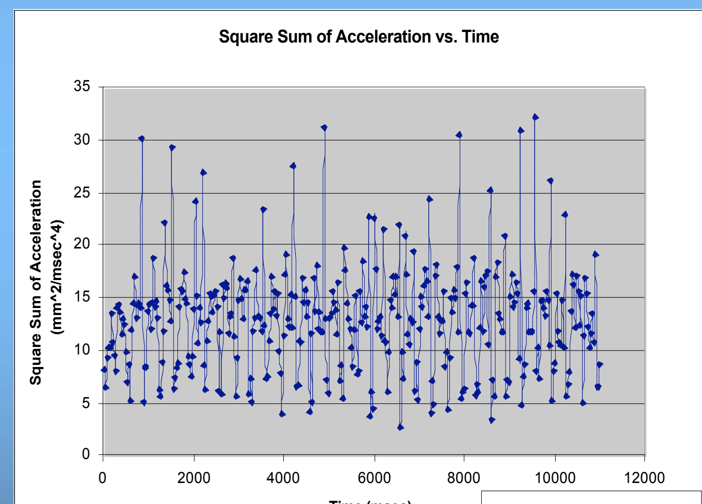


Quarter Speed

Our Upright Walk
vs.
CMU RoboCup Walk

- Upright
- RoboCup

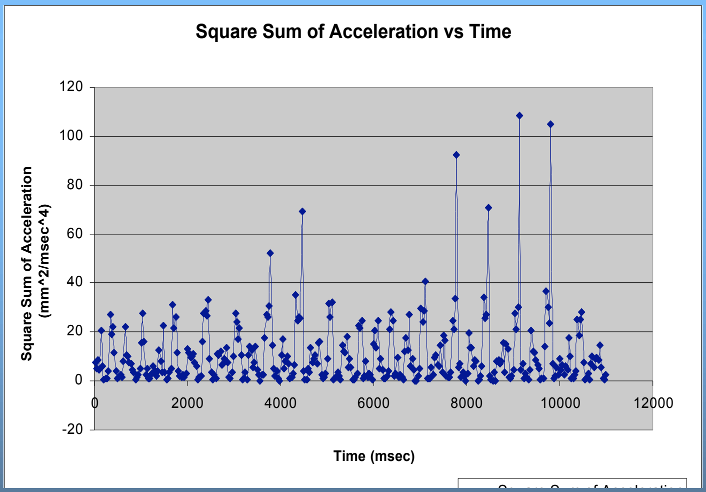


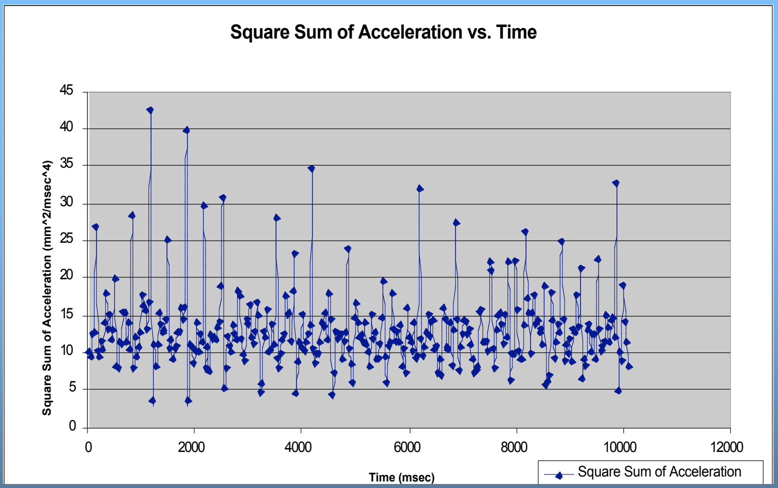


Half Speed

Our Upright Walk
vs.
CMU RoboCup Walk

- Upright
- RoboCup

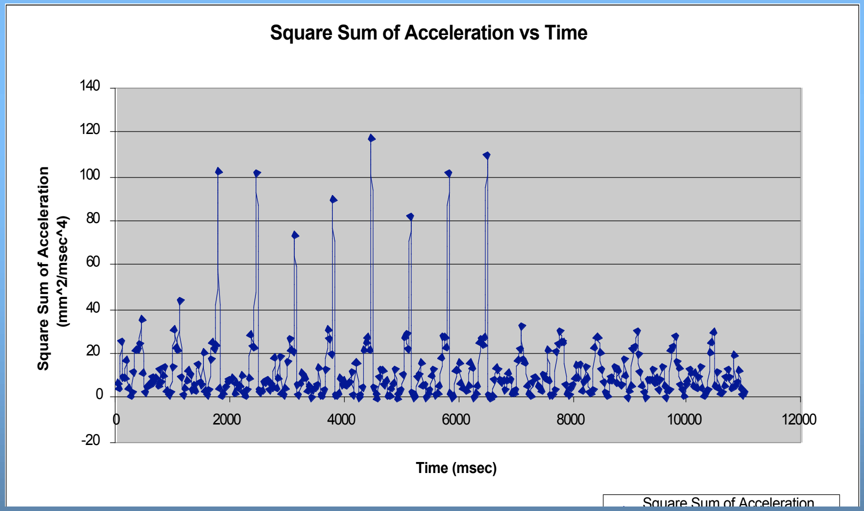




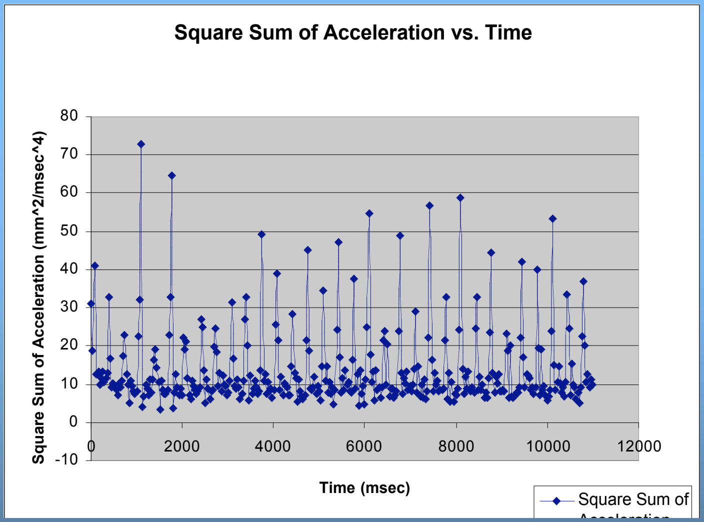
Three Quarter Speed

Our Upright Walk
vs.
CMU RoboCup Walk

- Upright



- RoboCup



Full Speed

See, AIBO, *See!*

- Developed an algorithm that allows AIBO to follow a pink line
- Gradually improved algorithm based on perceived weaknesses

How does AIBO see?

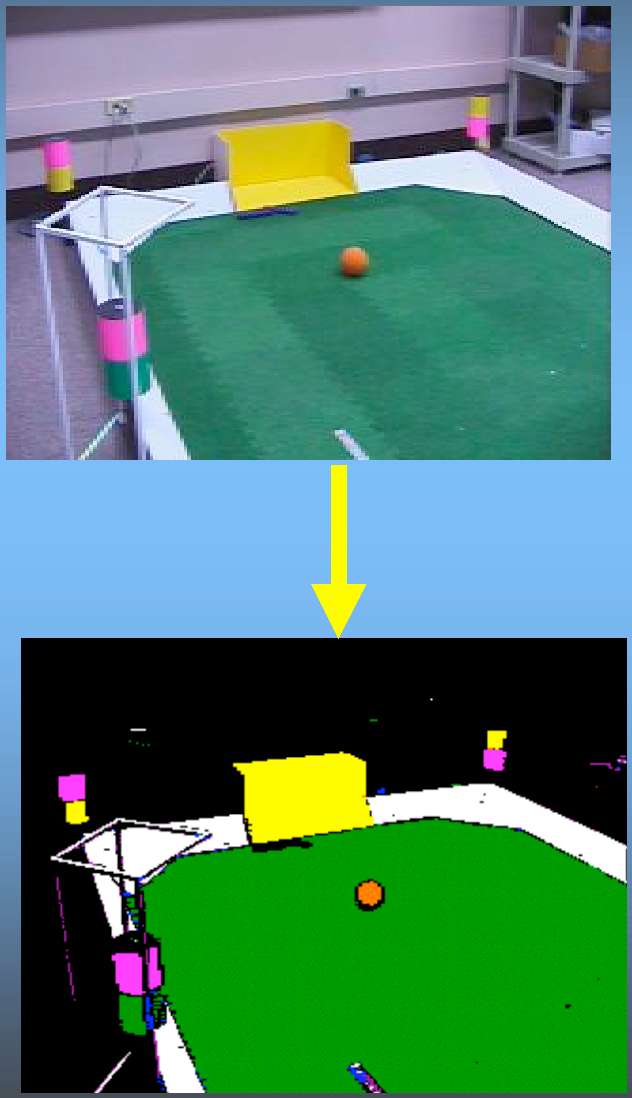
AIBO

CCD Camera
(YUV)

Raw Image
Translated
on
Computer
(RGB)

Segmented Vision

- AIBO camera captures YUV format
- Bitmapped images are too large to send over network efficiently, so the images must be compressed

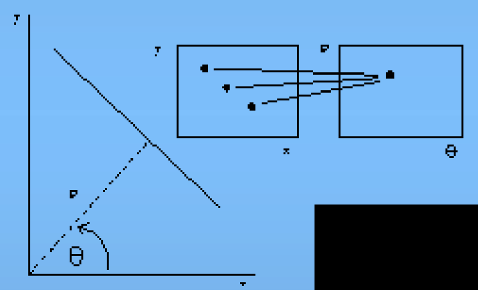


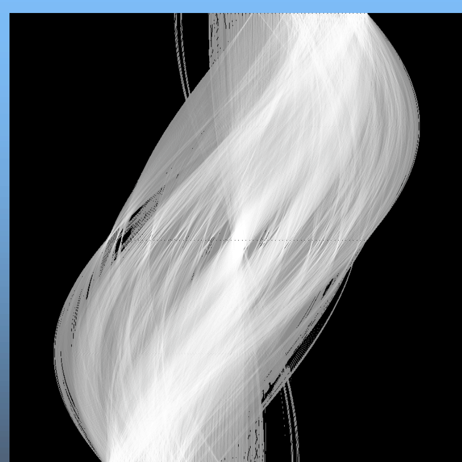
RLE (Run-Length Encoding)

- The data is then converted into a series of color run triplets
- Triplets are sent to the computer, where they are placed into an array
- Vision segmentation and run-length encoding is performed on board the AIBO

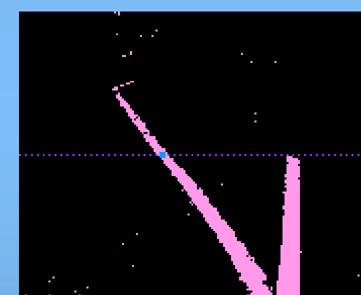
Line Following Algorithms

Hough Transformation





VS.



Hack Algorithm

Hough Transformation

- Represent a line in an image in a different way: Parameter Space
- Example – a line in image space can be represented as:

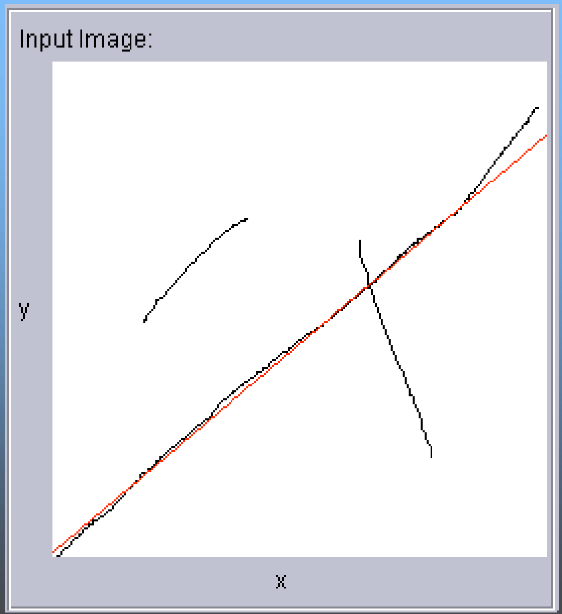
$$y = mx + y_0$$

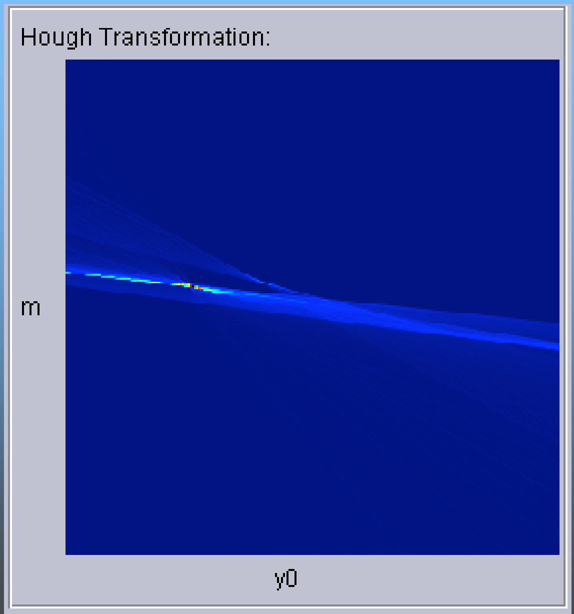
- The Hough method “transforms” this equation to parameter space:

$$y_0 = y - mx$$

Hough Transformation (cont'd)

- A discrete parameter space called an *accumulator* is created. All points in the original image are converted using Hough Transform into this parameter space.
- Points with the same slope and y-intercept are accumulated into the same cell of the accumulator.
- The highest cell is found, thus finding the most prominent line (marked in red).





Hough Transformation (cont'd)

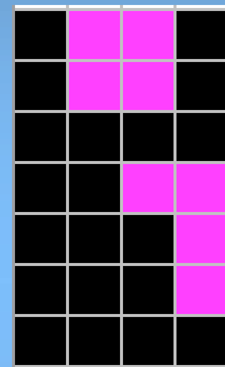
- Advantages:
  - Will almost always find the most prominent line
  - Ignores static and foreign objects (unless they have defined edges)
- Disadvantages:
  - Implementation in a robot is too computationally expensive
  - Processes too slow for a real-time image, like the AIBO's

A Hack Gains Greatness

- A *hack* is defined as an inelegant and usually temporary solution to a problem
- Ironically, the hack line following algorithm we developed (with the help of Alok Ladsariya) became our best solution.

The Basic Line Following Algorithm

- Take in decoded segmented vision
- Create RegionMap
  - First each pixel is set to pink or not-pink
  - Give each pink region a unique number… remember PaintBucket?
- Turn toward largest region on horizontal center line



-1	0	0	-1
-1	0	0	-1
-1	-1	-1	-1
-1	-1	0	0
-1	-1	-1	0
-1	-1	-1	0
-1	-1	-1	-1

-1	A	A	-1
-1	A	A	-1
-1	-1	-1	-1
-1	-1	B	B
-1	-1	-1	B
-1	-1	-1	B
-1	-1	-1	-1

Legend	
-1	Not Pink
0	Pink
A	1st Region
B	2nd Region

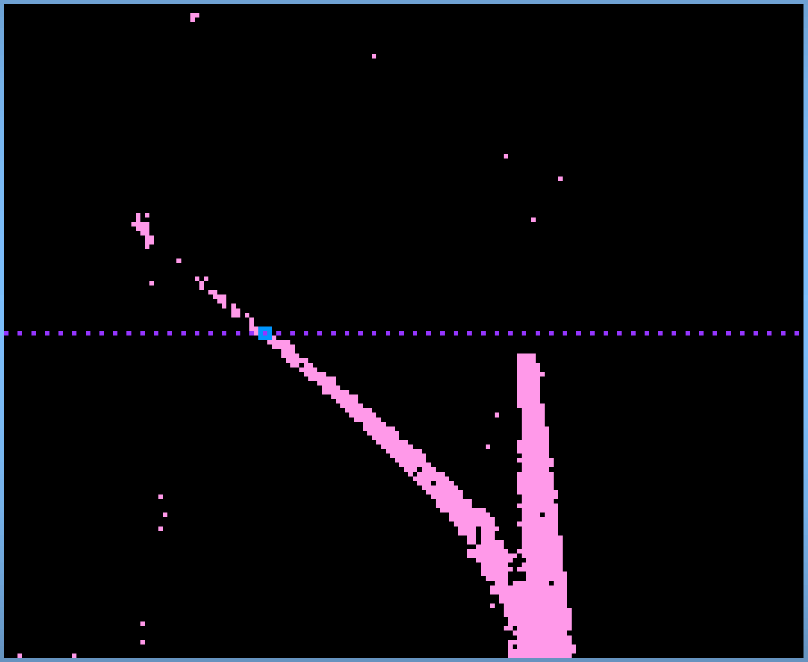
The Basic Line-Following Algorithm

- Primary target – displayed as blue dot
  - at intersection of largest region & center row
- Direction adjustment
  - Turns L if blue dot to L of center column
  - Turns R if blue dot to R of center column
- A few fundamental assumptions allow us to keep the algorithm simple
  - The line will be the largest region
  - The line will cross the center row *once*

Improvements

Definition: Lost = no regions exist in center row

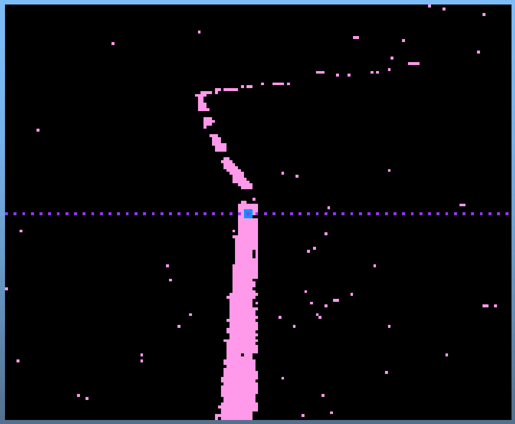
- If lost → stop forward motion & rotate
  - → determine rotation direction by which side of image contained dot more commonly
    - In last 15 frames
- Adjust direction with speed proportional to blue dot's distance from center column



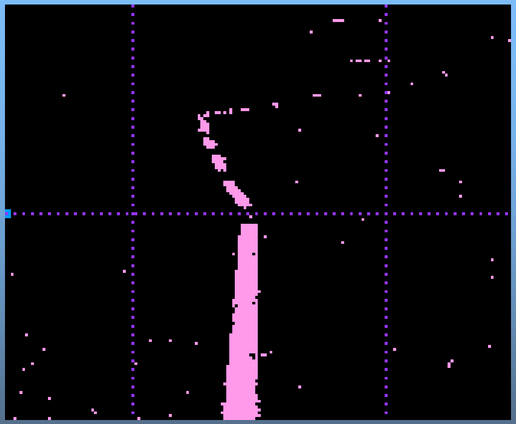
Algorithm Issues

Algorithm Issues

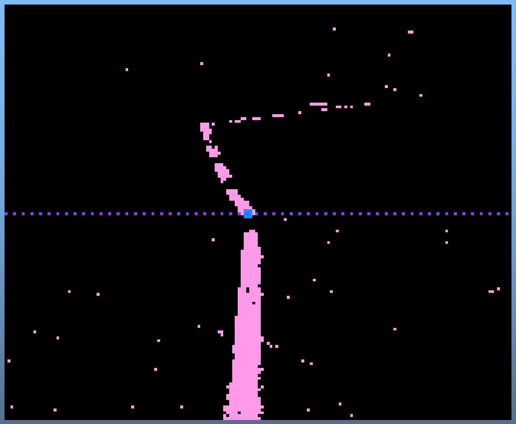
- A Dashed Line
  - The breaks in the line trigger the ‘lost’ behavior briefly
  - But even so, the break is not long enough to lose the dog completely.



Walking along happily…



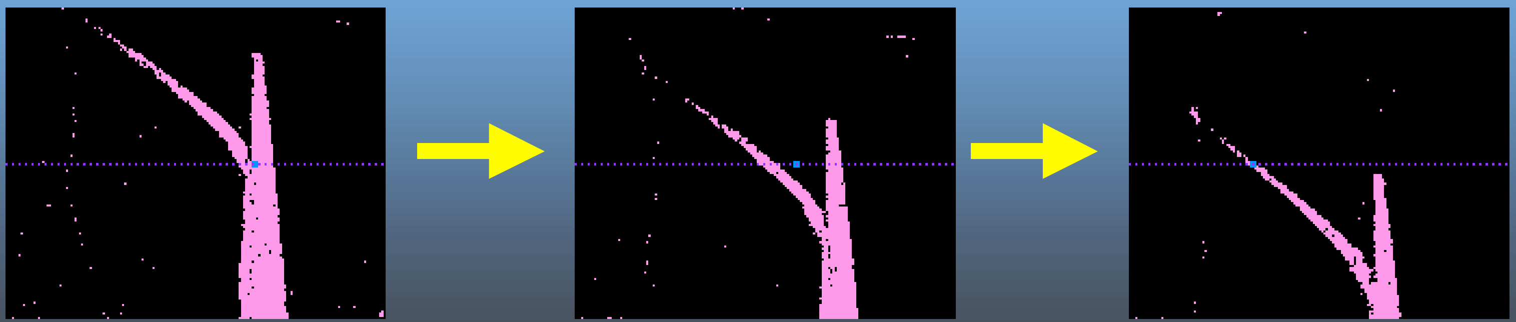
AHH!! BREAK IN THE LINE!!



Found line, walking along happily…

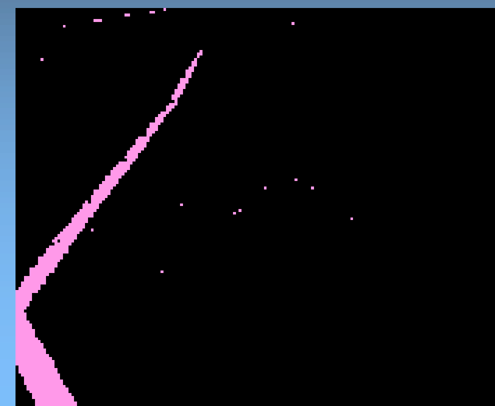
Algorithm Issues

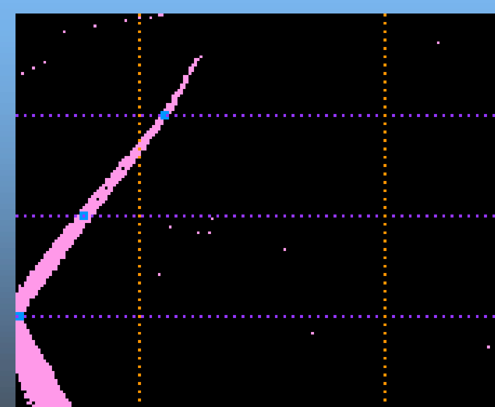
- Branching lines:
  - branch recognized as one region
  - the average taken of all x values of region's pixels in center row
  - once the shorter branch gets below center row, target jumps to longer branch (line)



Algorithm Issues

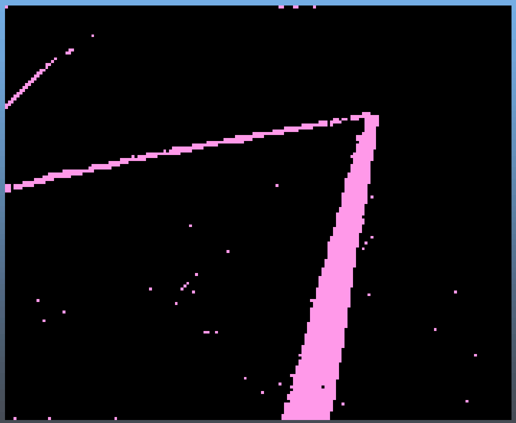
- Dog can lose line when
  - Special case: line slopes toward center of image
- Slope-Opposite-Direction Method
  - Two more blue dots
    - Above and below center
  - used to find slope of the line
  - If special case true:
    - Compromise and go *straight!*





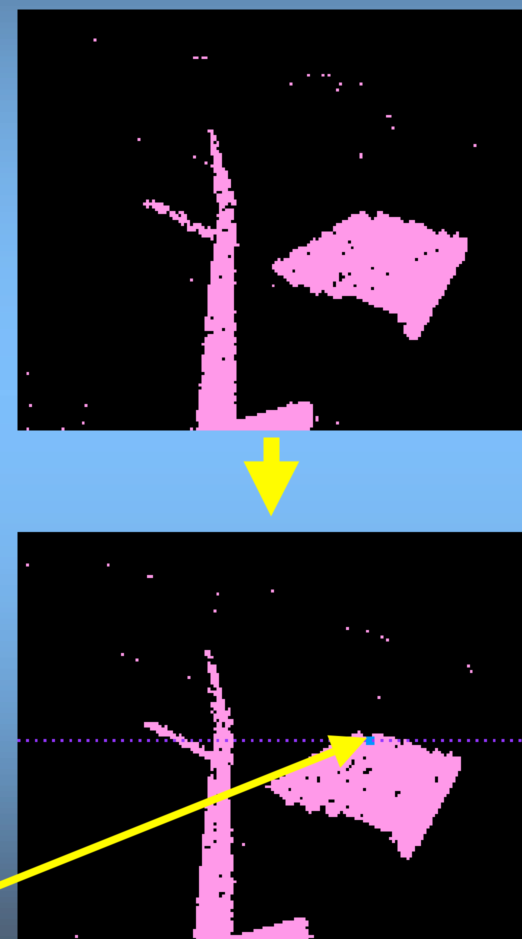
Unsolved Algorithm Issues

- Extremely sharp curves (< 90 deg.)
- Similar to branching issue (but more troublesome)
    - Current algorithm averages X values
    - Heads for center between two intersections of line with center row



Problem: Follow the Line!

- …that means don't follow the square.
- A possible solution: Shape Recognition
  - Feature Extraction
  - Describe regions



FAILURE!

In the Future: Feature Extraction

- Features easy to measure:
  - Area
  - Perimeter
  - Granularity Measurement
- And how can their measurements be combined → decision?

- Hypothesis: AIBO should choose
  - Largest
  - Most elongated
  - Least grainy

region in image.

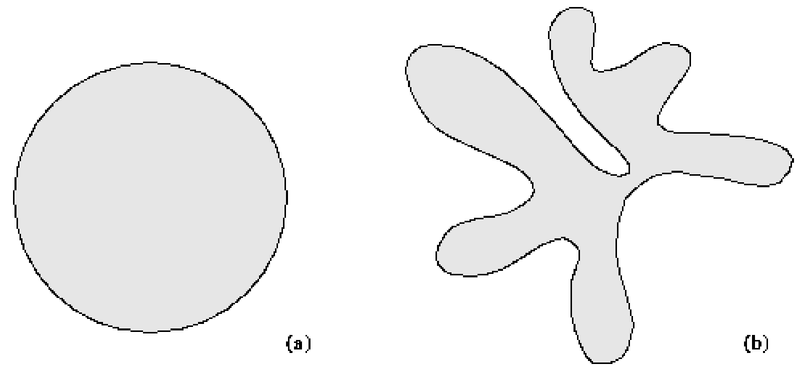
Elongation

- Ratio of Perimeter to Area of region
- In computer vision, called “Compactness”
- C = (P^2) / A

- Circle = most compact
- Way to differentiate between

  more compact shapes

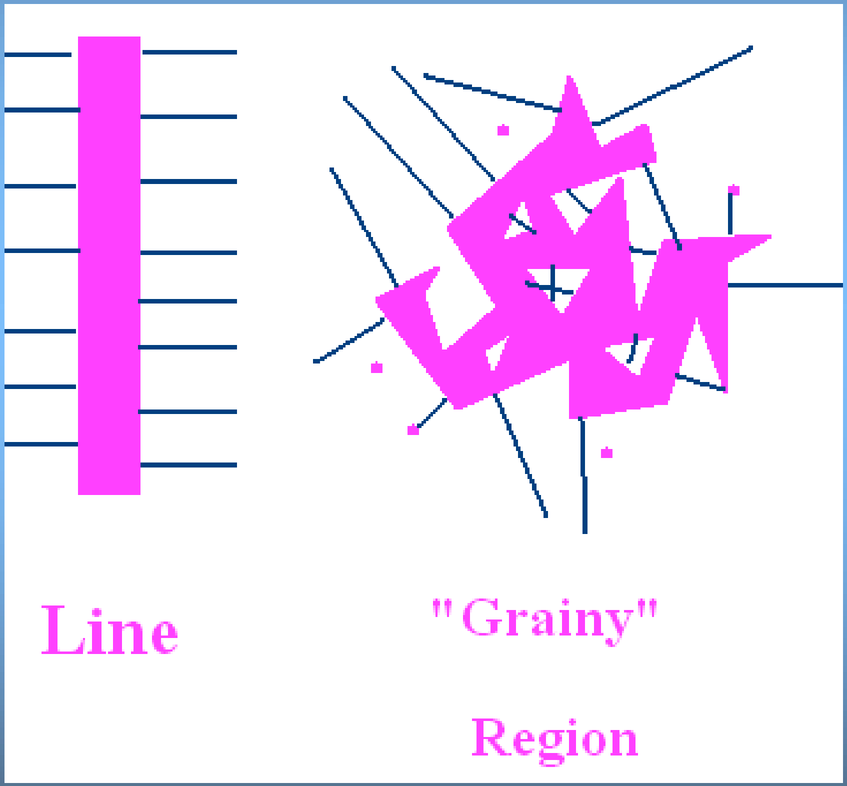
  and the line.



(a) Compact, (b) non-compact.

Granularity Measurement

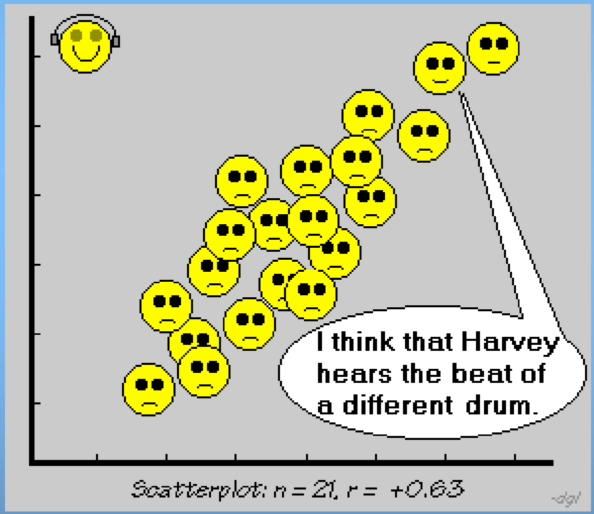
- Purpose: distinguish between
  - clear lines and grainy regions
- From each pixel at the boundary of a region
  - a ray extends until another pink pixel is encountered
- Granularity Measurement = sum of all rays' lengths



Standardization

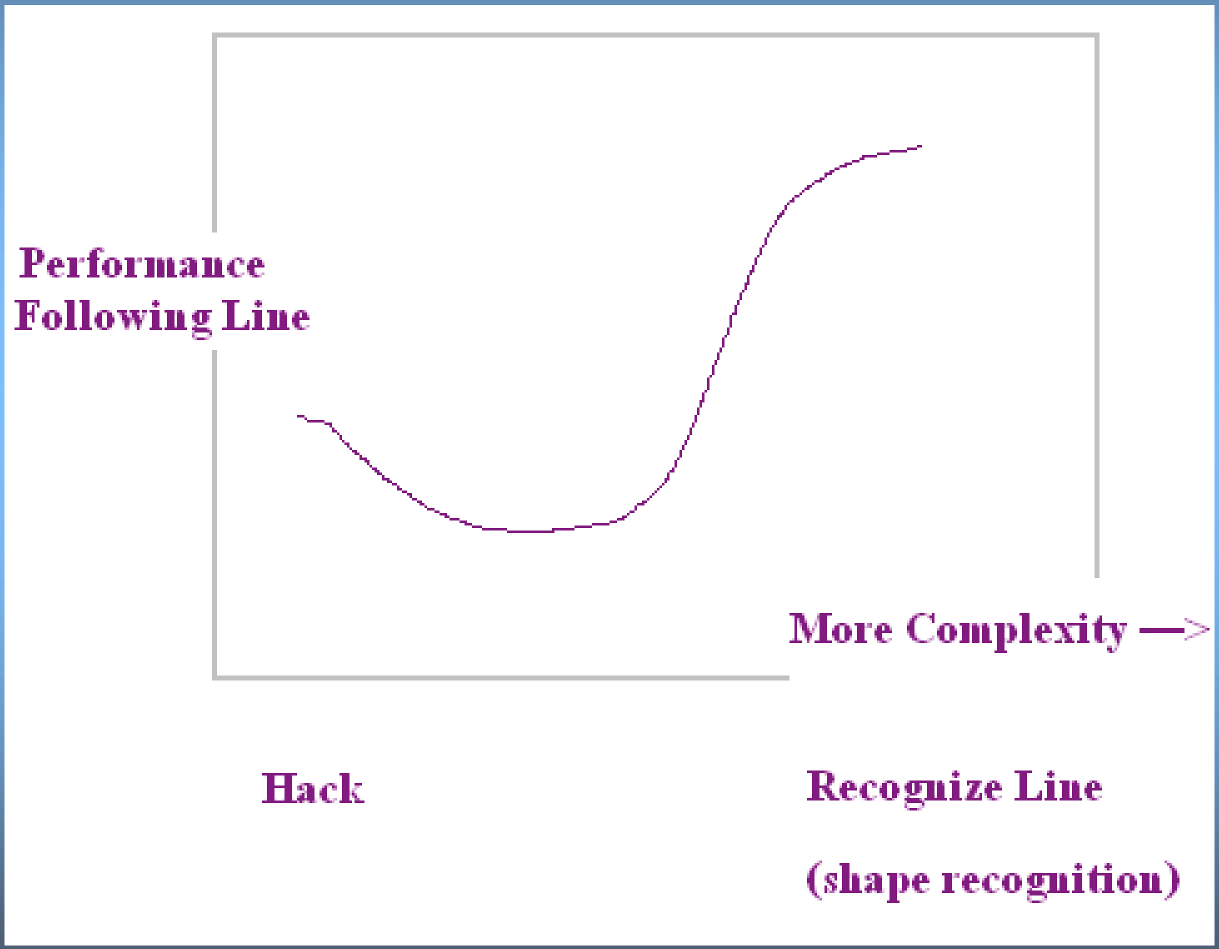
- To pick the best region, one can combine the three measurements
  - Largeness, Compactness, & Grainy Measure
- All have their own scales → standardization necessary
- Relevant → measurement for a region relative to all other regions.

•*How* different is this point?



www.neiu.edu/~lruecker/ smrm.htm

Performance vs. Complexity



Live Demo

How Different Walks Affect Line-Following

- Low velocity vs. high velocity
- Center of gravity affects slip potential
- Camera level affects:
  - Ease of getting lost
  - Precision of line following

Special Thanks to:

- Ethan
- Alok Ladsariya
- Dr. David Touretzky
- Greg Kesden
- Sony
- CMU School of Computer Science and Robotics Institute
- Manuela Veloso & CMU RoboCup
- PMatt

Bye!

Image Sources

www.dai.ed.ac.uk/HIPR2/hough.htm
www.dai.ed.ac.uk/CVonline/LOCAL_COPIES/MARBLE/medium/contours/feature.htm
www.physik.uni-osnabrueck.de/nonlinop/Hough/LineHough.html
www.ri.cmu.edu/labs/lab_60.html
www.nynatur.dk/artikler/artikel_tekster/robo_pets.html
www003.upp.so_net.ne.jp/studio_mm/kameo/kameo.html
www.opus.co.jp/products/aibo/
www.generation5.org/aibo.shtml
www.ohta-kyoko.com/aibo/aibo.html
www.vdelnevo.co.uk/
www.d2.dion.ne.jp/~narumifu/diary.html
www.21stcentury.co.uk/robotics/aibo.asp
digitalcamera.gr.jp/html/HotNews/backno/HotNews001001-31.ht
www.seaple.icc.ne.jp/~somari/aibo.htm
www-2.cs.cmu.edu/afs/cs/project/robosoccer/www/legged/legged-team.html#publications
ccrma-www.stanford.edu/CCRMA/Courses/252/sensors/node9.html
http://www.icaen.uiowa.edu/~dip/LECTURE/Shape3.html#scalar