Robotic Motion Planning: Tangent Bug and Sensors

Robotics Institute 16-735

http://voronoi.sbp.ri.cmu.edu/~motion

Howie Choset

http://voronoi.sbp.ri.cmu.edu/~choset

What's Special About Bugs

- Many planning algorithms assume global knowledge
- Bug algorithms assume only *local* knowledge of the environment and a global goal
- Bug behaviors are simple:
  - 1) Follow a wall (right or left)
  - 2) Move in a straight line toward goal
- Bug 1 and Bug 2 assume essentially tactile sensing
- Tangent Bug deals with finite distance sensing

Summary

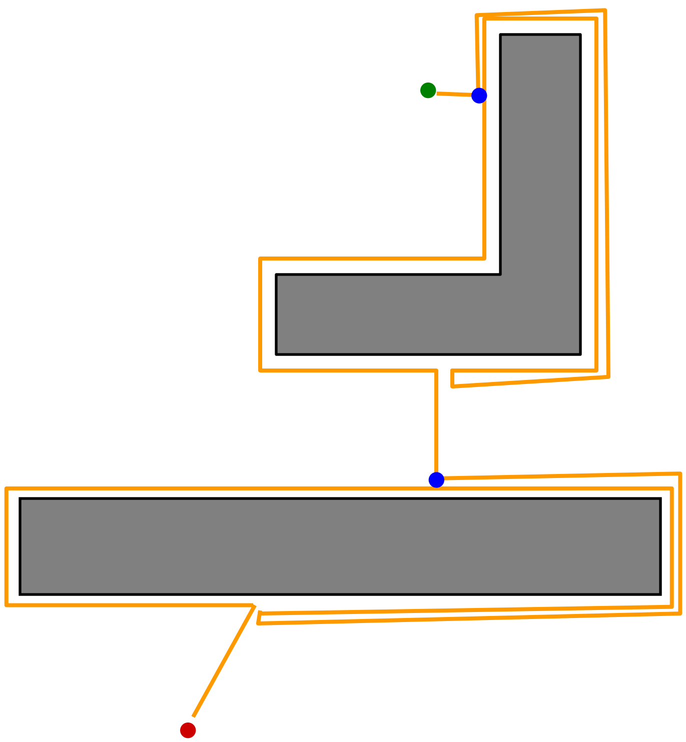
- Bug 1: safe and reliable
- Bug 2: better in some cases; worse in others
- Should understand the basic completeness proof
- Two behaviors: motion-to-goal, boundary following
- Tangent Bug: supports range sensing
- Sensors and control
  - should understand basic concepts and know what different sensors are

Bug 1

But some computing power!

- known direction to goal
- otherwise local sensing

walls/obstacles & **encoders**



"Bug 1" algorithm

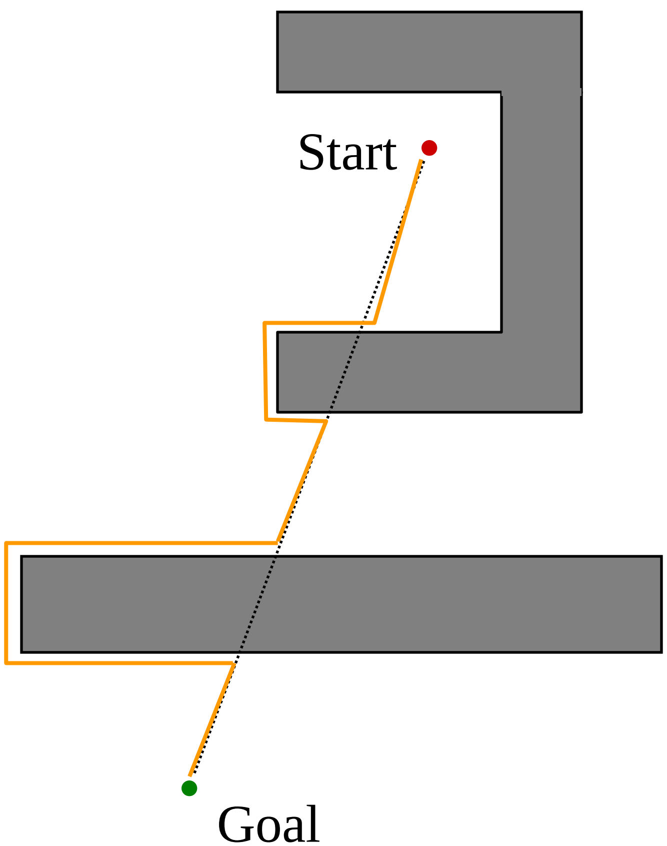
1) head toward goal

2) if an obstacle is encountered, circumnavigate it *and* remember how close you get to the goal

3) return to that closest point (by wall-following) and continue

Vladimir Lumelsky & Alexander Stepanov: Algorithmica 1987

A better bug?



"Bug 2" Algorithm

1) head toward goal on the *m-line*

2) if an obstacle is in the way, follow it until you encounter the m-line again ***closer to the goal***.

3) Leave the obstacle and continue toward the goal

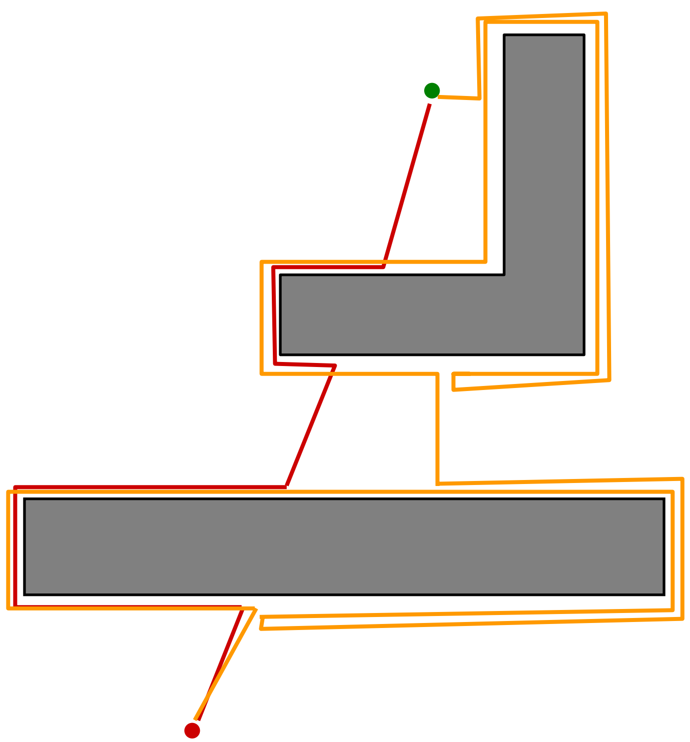
Better or worse than Bug1?

head-to-head comparison

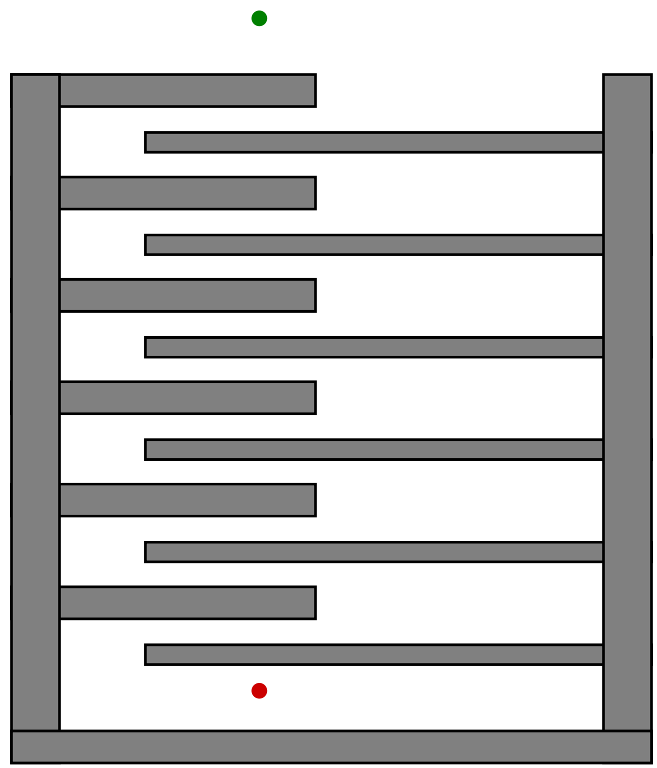
or thorax-to-thorax, perhaps

Draw worlds in which Bug 2 does better than Bug 1 (and vice versa).

Bug 2 beats Bug 1



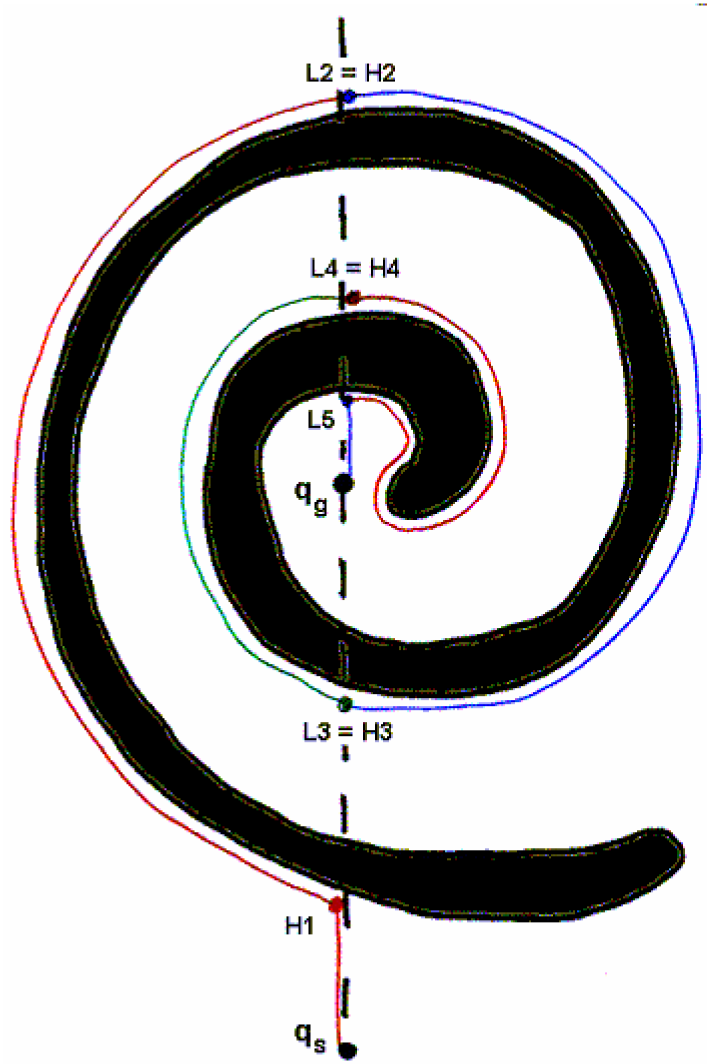
Bug 1 beats Bug 2

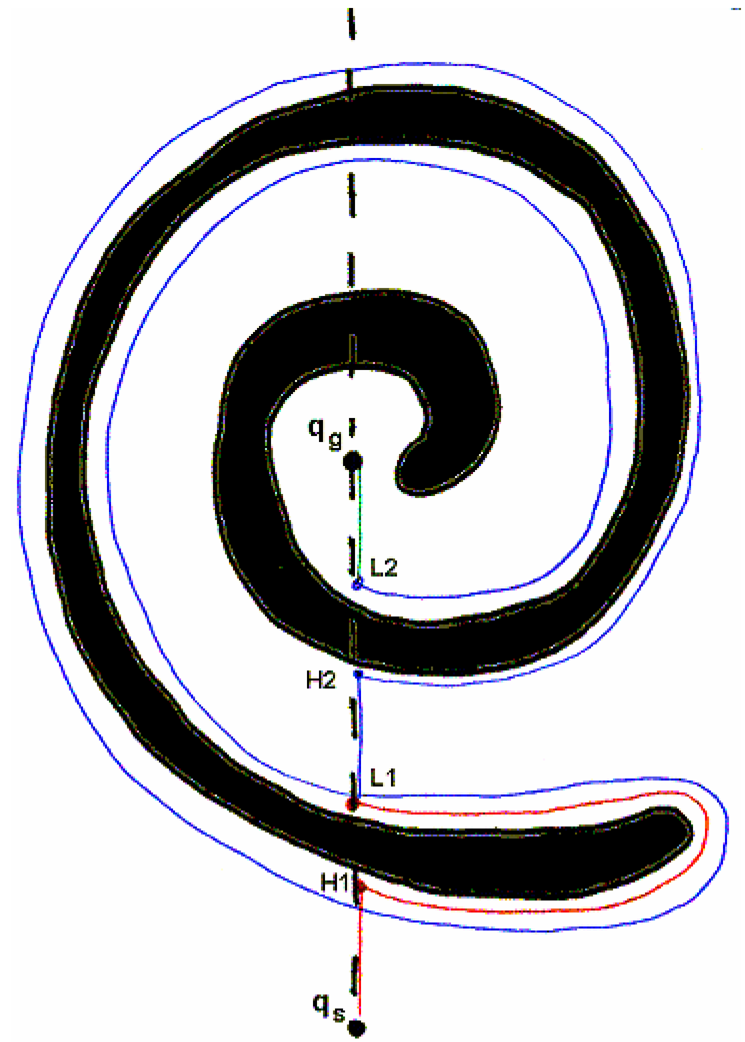


BUG 1 vs. BUG 2

- BUG 1 is an *exhaustive search algorithm*
  - it looks at all choices before commiting
- BUG 2 is a *greedy* algorithm
  - it takes the first thing that looks better
- In many cases, BUG 2 will outperform BUG 1, but
- BUG 1 has a more predictable performance overall

The Spiral





A More Realistic Bug

- As presented: global beacons plus contact-based wall following
- The reality: we typically use some sort of range sensing device that lets us look ahead (but has finite resolution and is noisy).
- Let us assume we have a range sensor

Raw Distance Function

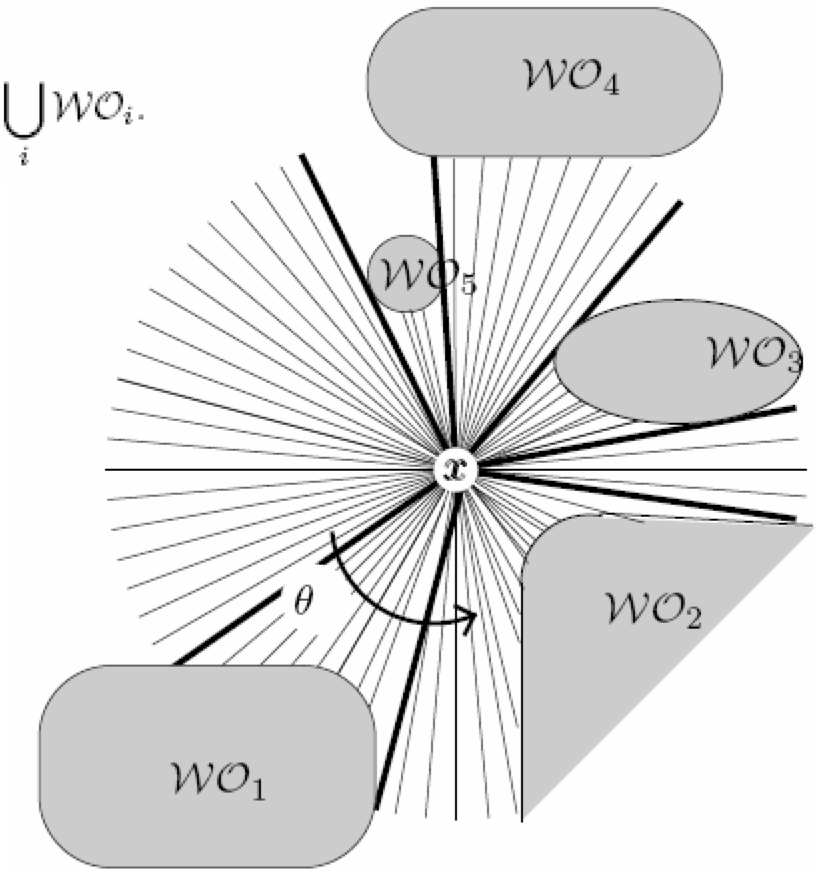
$$\rho(x,\theta) = \min_{\lambda\in[0,\infty]} d(x, x+\lambda[\cos\theta, \sin\theta]^T),$$

$$\text{such that} \quad x+\lambda[\cos\theta,\sin\theta]^T \in \bigcup_i \mathcal{WO}_i.$$

$$\rho : \mathbb{R}^2 \times S^1 \to \mathbb{R}$$

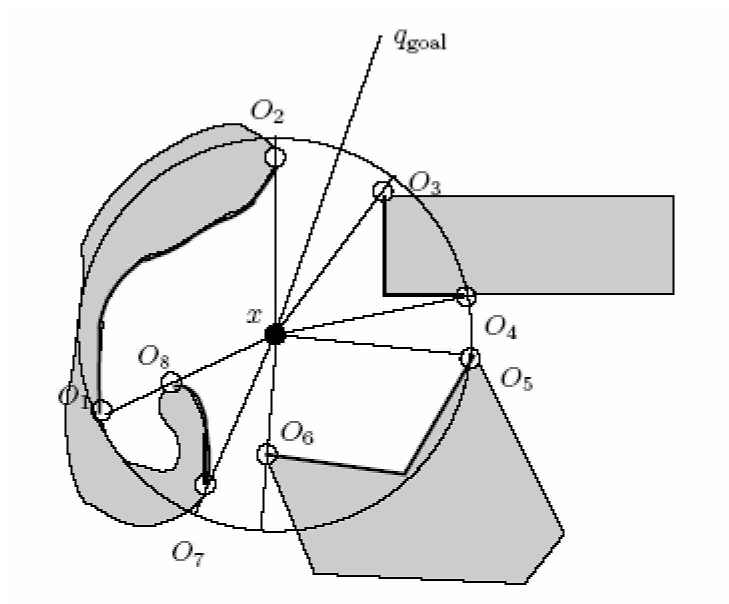
Saturated raw distance function

$$\rho_R(x,\theta) = \begin{cases} \rho(x,\theta), & \text{if } \rho(x,\theta) < R \\ \infty, & \text{otherwise.} \end{cases}$$

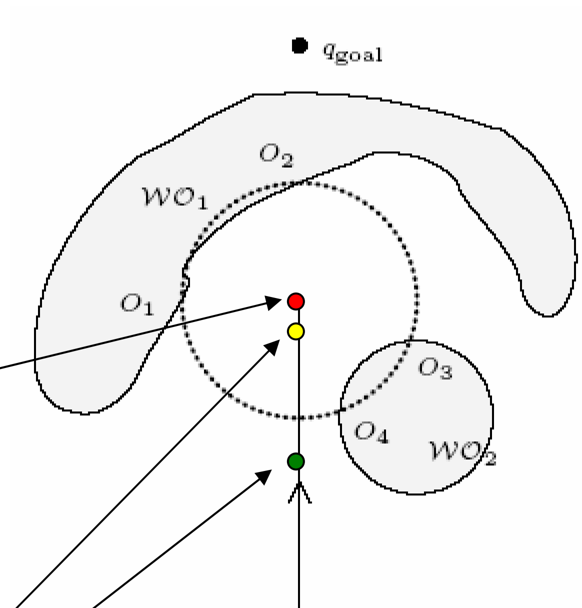


Intervals of Continuity

- Tangent Bug relies on finding endpoints of finite, conts segments of ρ_R



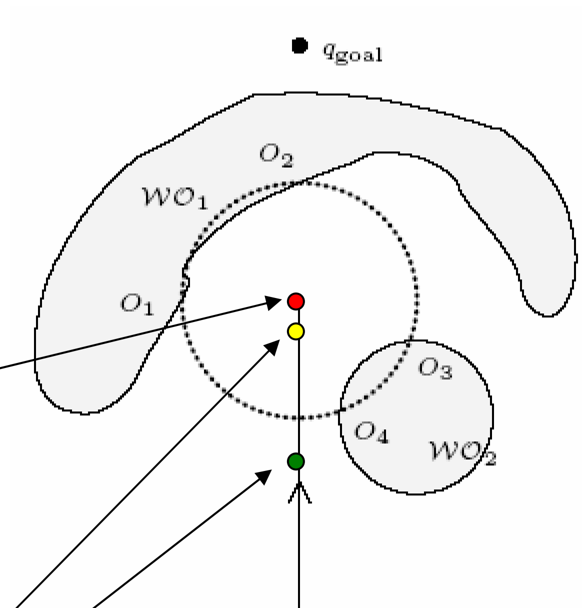
Motion-to-Goal Transition from Moving Toward goal to "following obstalces"



Transition

Currently, the motion-to-goal behavior "thinks" the robot can get to the goal

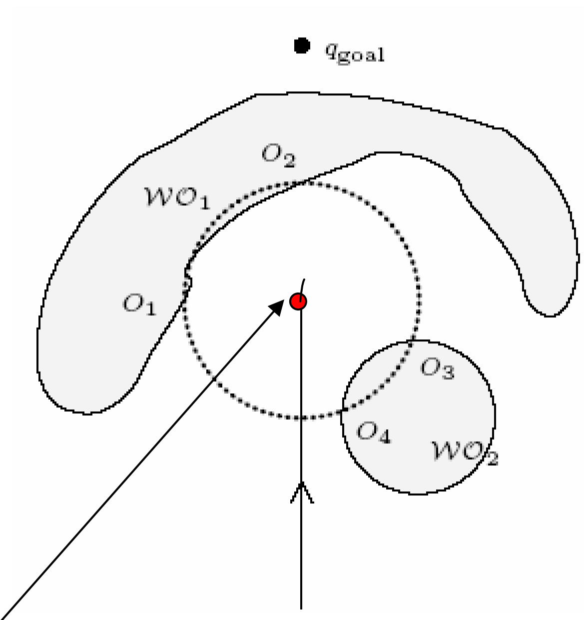
Motion-to-Goal Transition
Among Moving Toward goal to "following obstacles"



Transition

Currently, the motion-to-goal behavior "thinks" the robot can get to the goal

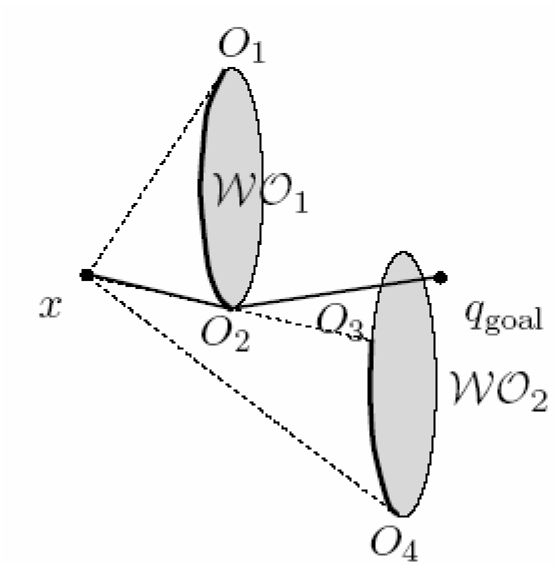
Motion-to-Goal Transition
Minimize Heuristic



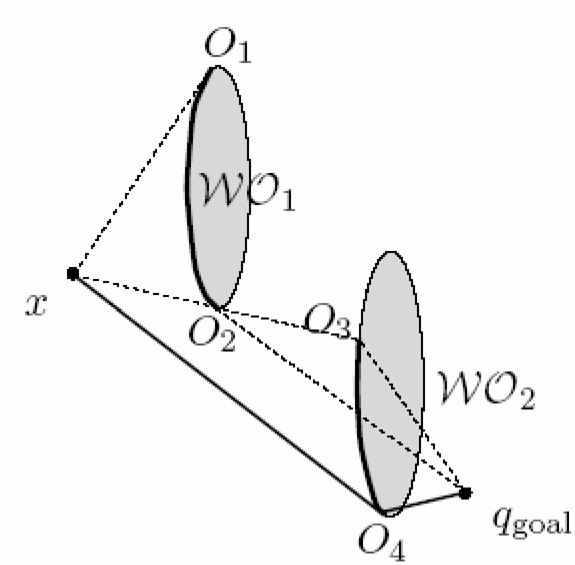
Now, it starts to see something --- what to do?
Ans: Choose the pt O_i that minimizes $d(x,O_i) + d(O_i,q_{goal})$

Minimize Heuristic Example

At x, robot knows only what it sees and where the goal is,



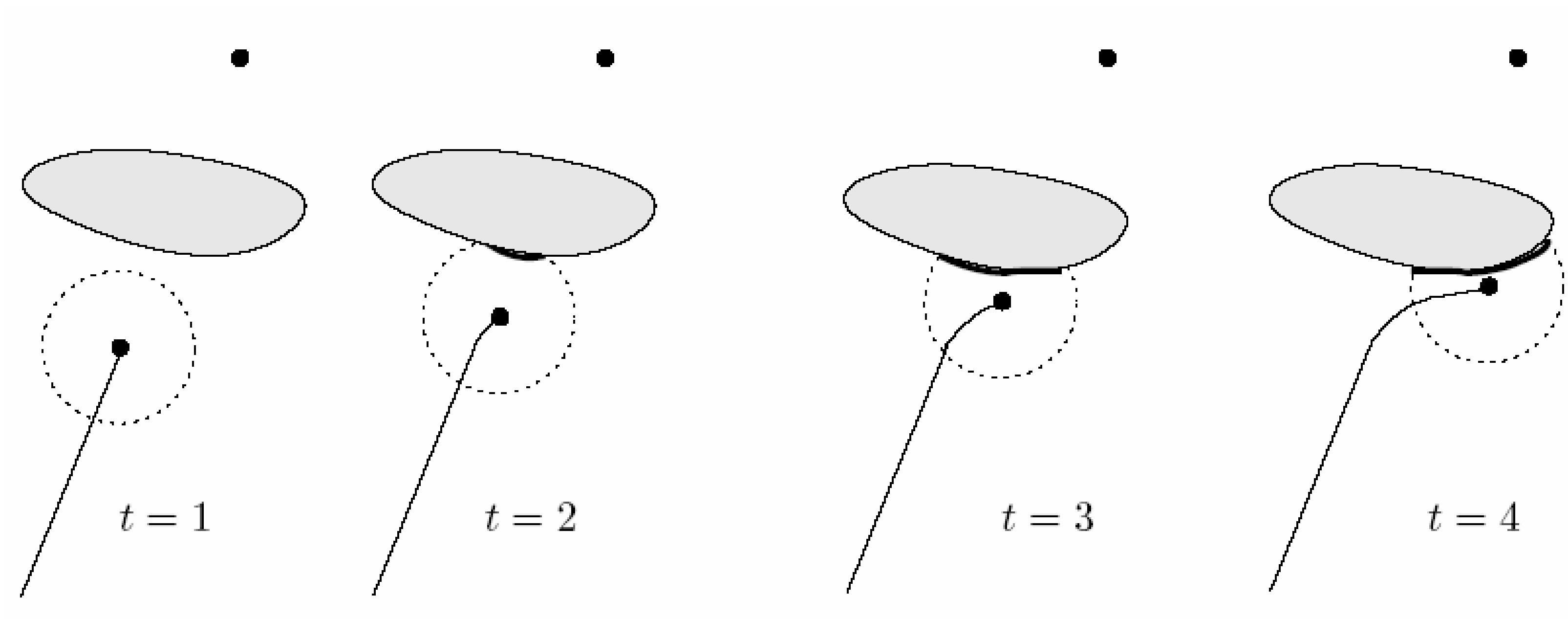
so moves toward O_2. Note the line connecting O_2 and goal pass through obstacle



so moves toward O_4. Note some "thinking" was involved and the line connecting O_4 and goal pass through obstacle

Choose the pt O_i that minimizes $d(x,O_i) + d(O_i,q_{goal})$

Motion To Goal Example



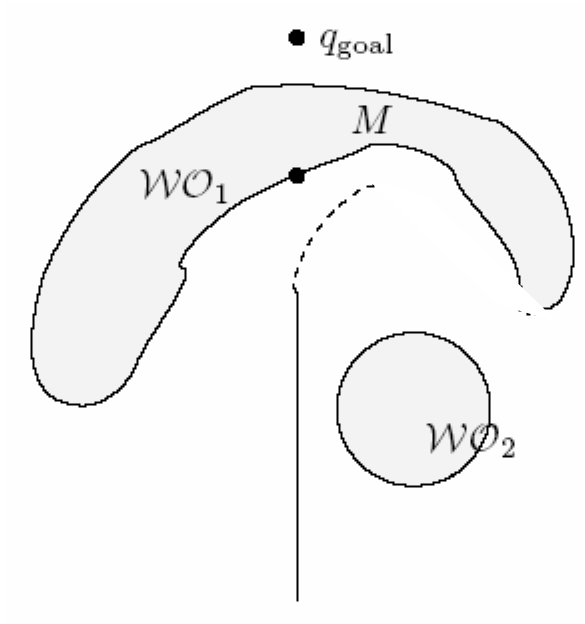
Choose the pt O_i that minimizes $d(x,O_i) + d(O_i,q_{goal})$

Transition *from* Motion-to-Goal

Choose the pt O_i that minimizes $d(x,O_i) + d(O_i,q_{goal})$

Problem: what if this distance starts to go up?

Ans: start to act like a BUG and follow boundary



M is the point on the "sensed" obstacle which has the shorted distance to the goal

Followed obstacle: the obstacle that we are currently sensing

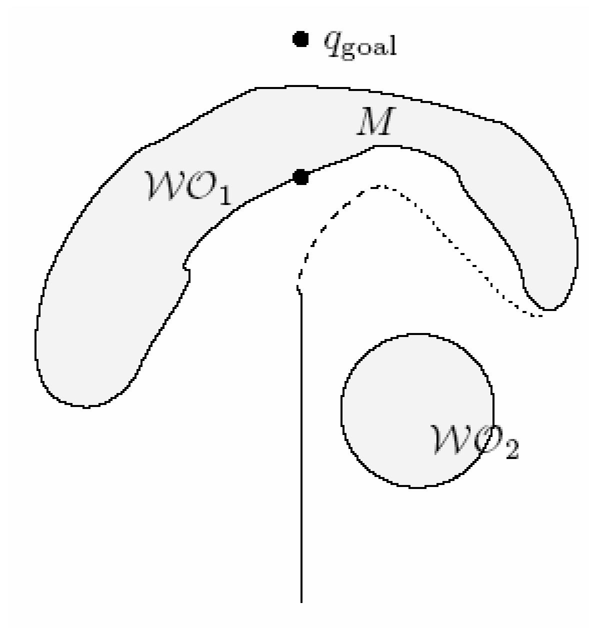
Blocking obstacle: the obstacle that intersects the segment $(1-\lambda)x + \lambda q_{\text{goal}} \quad \forall \lambda \in [0,1]$

They start as the same

Boundary Following

Move toward the O_i on the followed obstacle in the “chosen” direction

Maintain $d_{followed}$ and d_{reach}



M is the point on the “sensed” obstacle which has the shorted distance to the goal

Followed obstacle: the obstacle that we are currently sensing

Blocking obstacle: the obstacle that intersects the segment

They start as the same

d_{followed} and d_{reach}

- d_{followed} is the shortest distance between the sensed boundary and the goal

- d_{reach} is the shortest distance between *blocking* obstacle and goal (or my distance to goal if no blocking obstacle visible)

$$\Lambda = \{y \in \partial \mathcal{WO}_b : \lambda x + (1-\lambda) y \in \mathcal{Q}_{\text{free}} \quad \forall \lambda \in [0,1]\}.$$

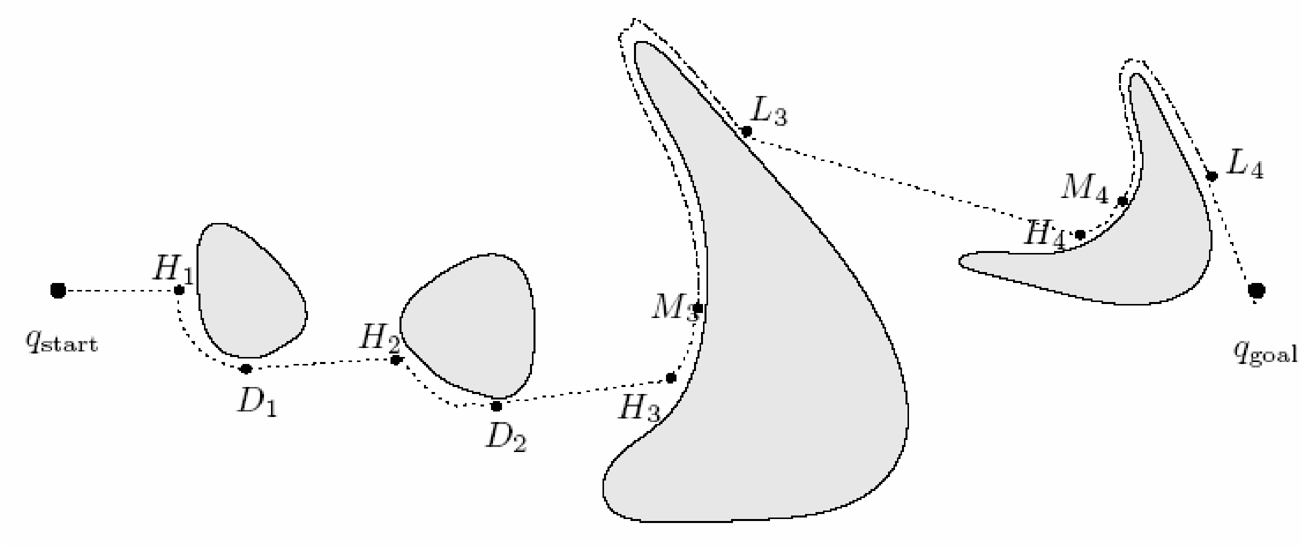
$$d_{\text{reach}} = \min_{c \in \Lambda} d(q_{\text{goal}}, c)$$

- Terminate boundary following behavior when $d_{\text{reach}} < d_{\text{followed}}$

- Initialize with $x = q_{\text{start}}$ and $d_{\text{leave}} = d(q_{\text{start}}, q_{\text{goal}})$

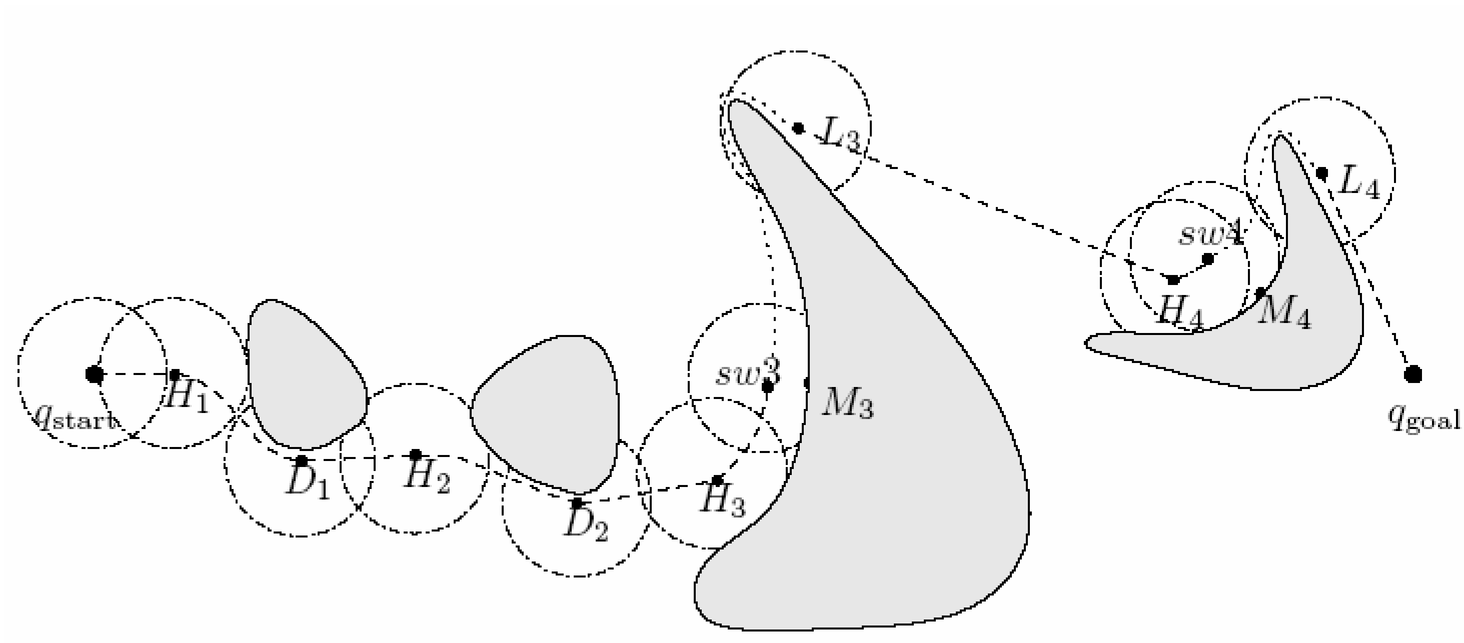
typo

Example: Zero Senor Range

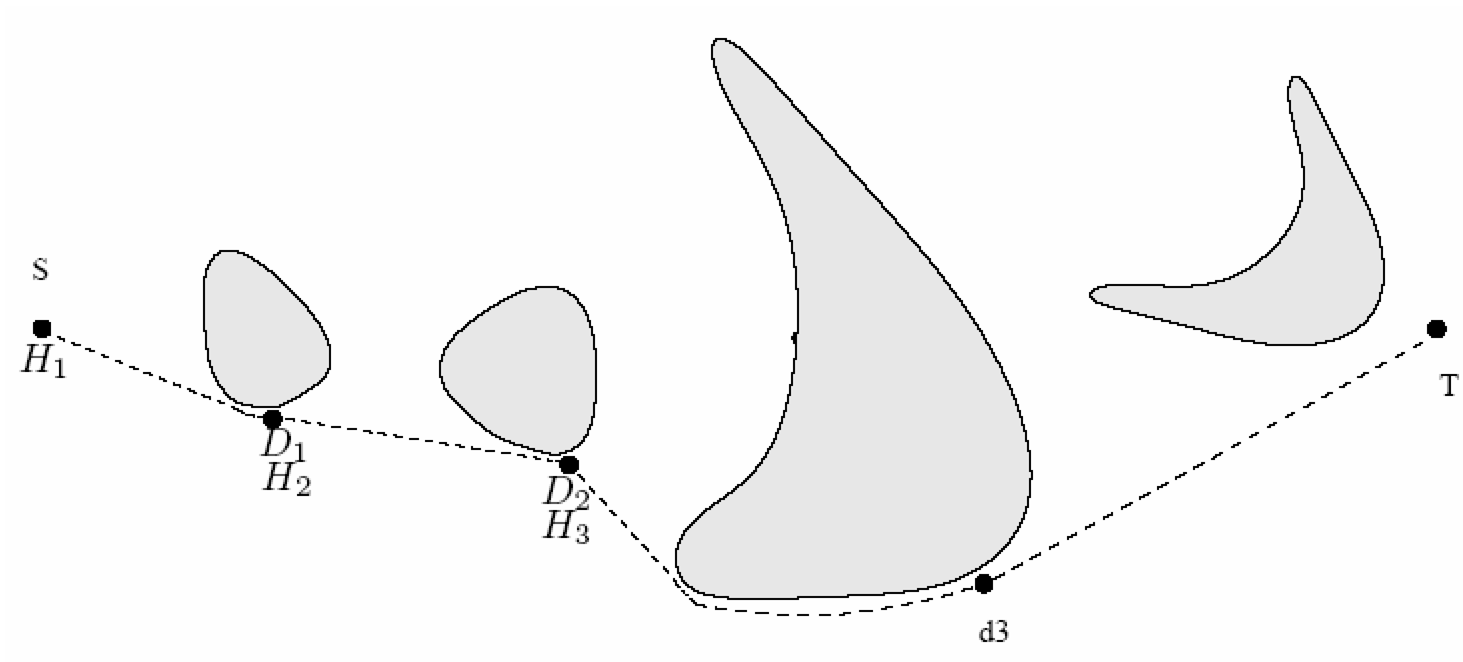


1. Robot moves toward goal until it hits obstacle 1 at H1
2. Pretend there is an infinitely small sensor range and the Oi which minimizes the heuristic is to the right
3. Keep following obstacle until robot can go toward obstacle again
4. Same situation with second obstacle
5. At third obstacle, the robot turned left until it could not increase heuristic
6. D_{followed} is distance between M_3 and goal, d_{reach} is distance between robot and goal because sensing distance is zero

Example: Finite Sensor Range

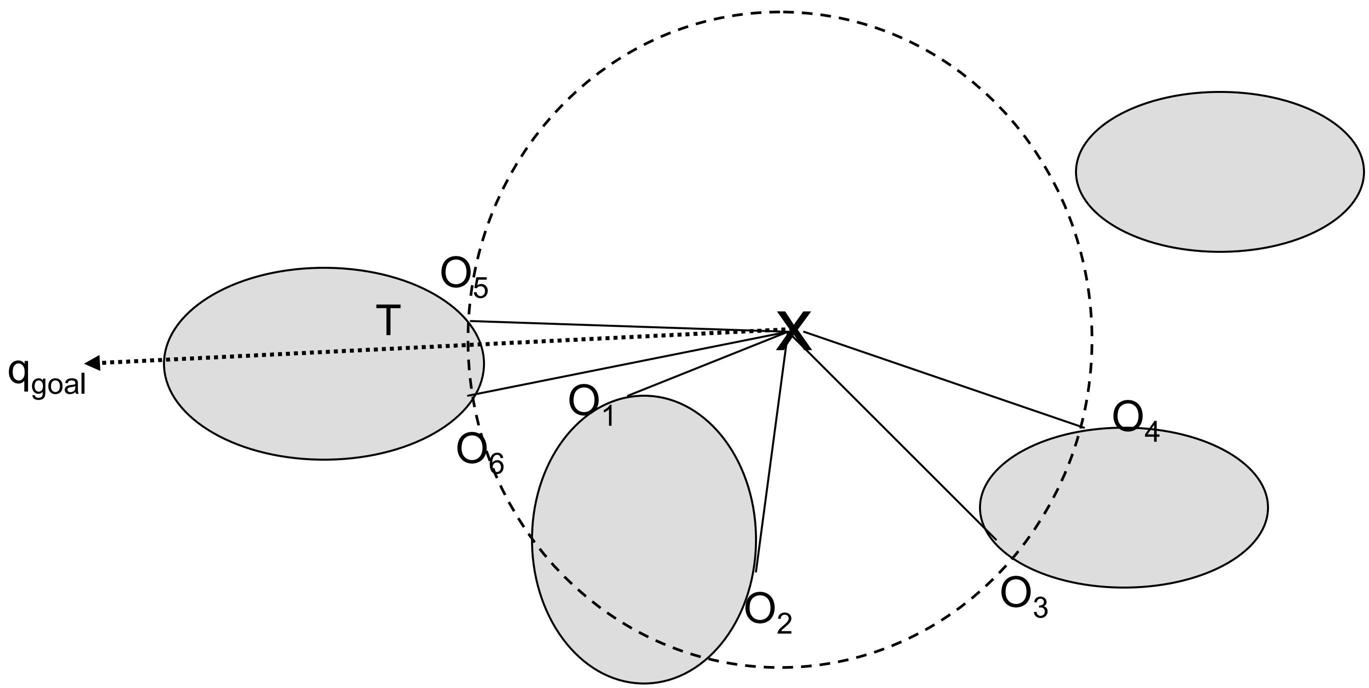


Example: Infinite Sensor Range



Tangent Bug

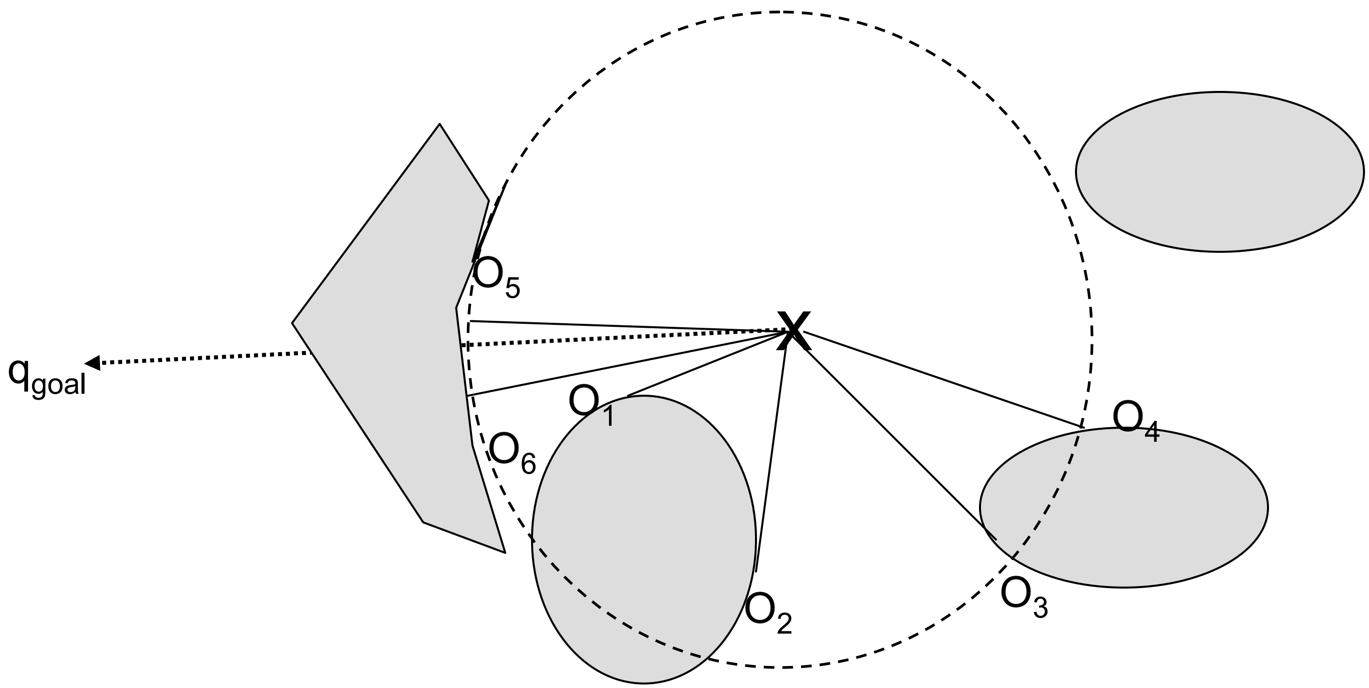
- Tangent Bug relies on finding endpoints of finite, conts segments of ρ_R



Now, it starts to see something --- what to do?
Ans: Choose the pt O_i that minimizes $d(x,O_i) + d(O_i,q_{goal})$
"Heuristic distance"

Tangent Bug

- Tangent Bug relies on finding endpoints of finite, conts segments of ρ_R



Problem: what if this distance starts to go up?
Ans: start to act like a BUG and follow boundary

The Basic Ideas

- A motion-to-goal behavior as long as way is clear or there is a visible obstacle boundary pt that decreases heuristic distance
- A boundary following behavior invoked when heuristic distance increases.
- A value $d_{followed}$ which is the shortest distance between the sensed boundary and the goal
- A value d_{reach} which is the shortest distance between *blocking* obstacle and goal (or my distance to goal if no blocking obstacle visible)
- Terminate boundary following behavior when $d_{reach} < d_{followed}$

Tangent Bug Algorithm

1) repeat
   a) Compute continuous range segments in view
   b) Move toward $n \in \{T, O_i\}$ that minimizes $h(x,n) = d(x,n) + d(n,q_{goal})$

   until
   a) goal is encountered, or
   b) the value of $h(x,n)$ begins to increase

2) follow boundary continuing in same direction as before repeating
   a) update $\{O_i\}$, d_{reach} and $d_{followed}$

   until
   a) goal is reached
   b) a complete cycle is performed (goal is unreachable)
   c) $d_{reach} < d_{followed}$

Note the same general proof reasoning as before applies, although the definition of hit and leave points is a little trickier.

Implementing Tangent Bug

- Basic problem: compute tangent to curve forming boundary of obstacle at any point, and drive the robot in that direction

- Let $D(x) = \min_c d(x,c) \quad c \in \cup_i WO_i$
- Let $G(x) = D(x) - W^*$ $\leftarrow$ some safe following distance
- Note that $\nabla G(x)$ points radially away from the object
- Define $T(x) = (\nabla G(x))$ the tangent direction
  - in a real sensor (we'll talk about these) this is just the tangent to the array element with lowest reading

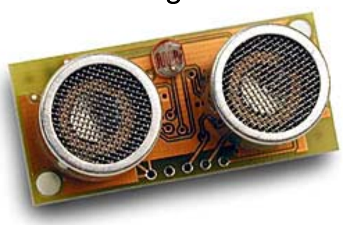
- We could just move in the direction T(x)
  - open-loop control

- Better is $\delta x = \mu (T(x) - \lambda (\nabla G(x)) G(x))$
  - closed-loop control (predictor-corrector)

Sensors!

Robots' link to the external world...

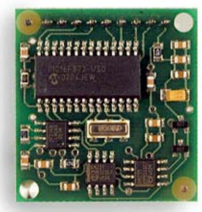
sonar rangefinder



gyro

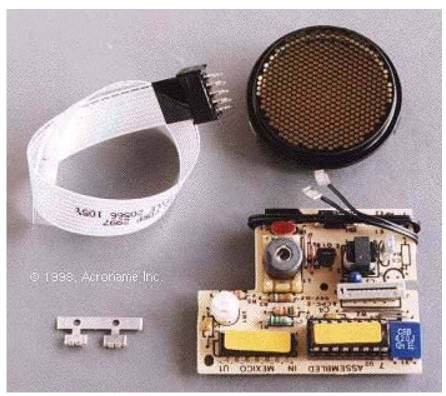
Sensors, sensors, sensors!

and tracking what is sensed: world models

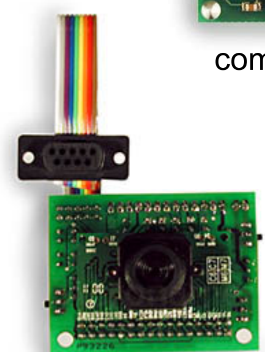


compass

IR rangefinder



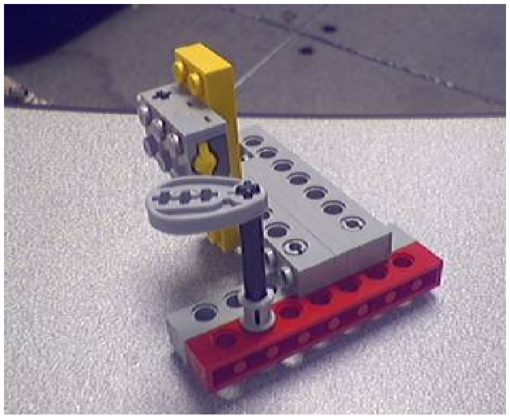
sonar rangefinder



CMU cam with on-board processing

odometry…

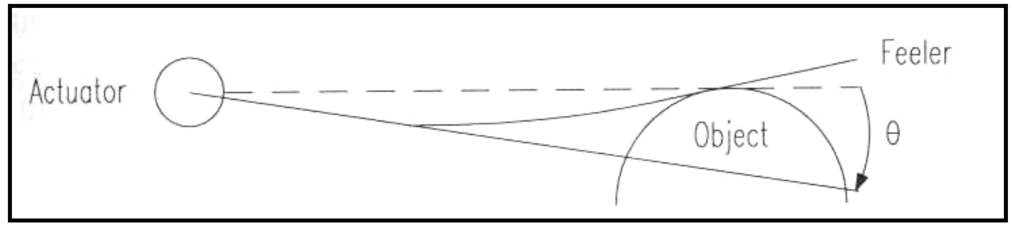
Tactile sensors

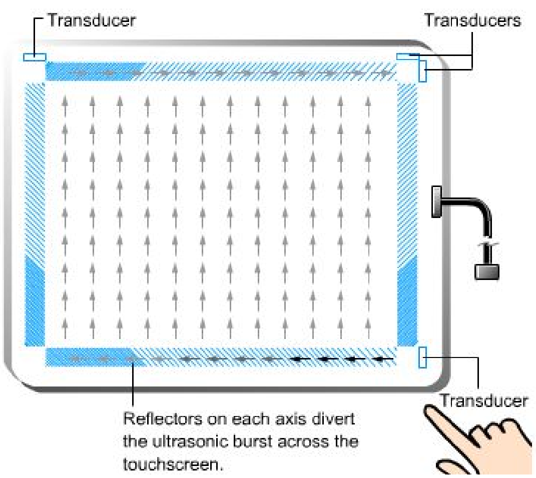


on/off switch

as a low-resolution encoder…

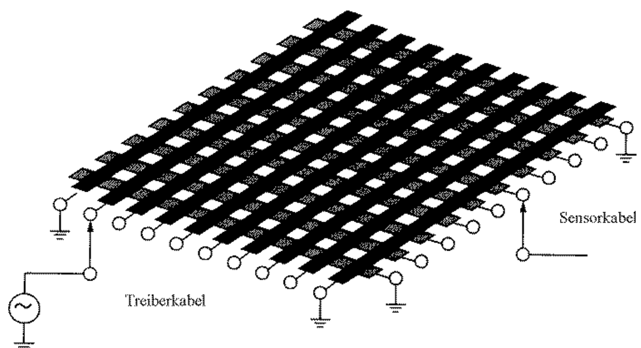
analog input: “Active antenna”





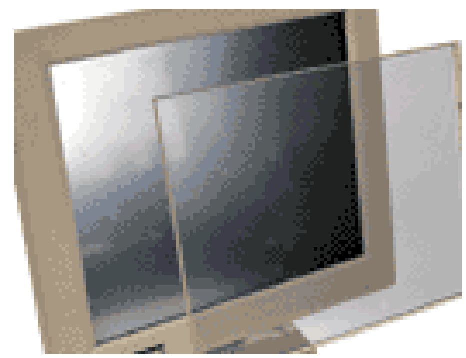
Surface acoustic waves

100% of light passes through



Capacitive array sensors

90% of light passes through



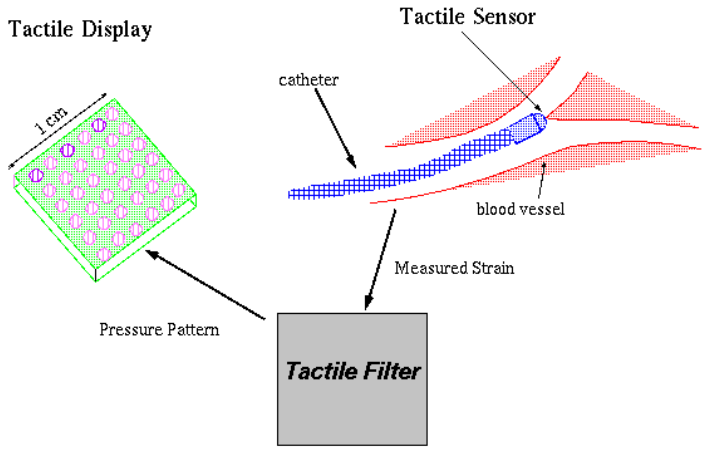
Resistive sensors

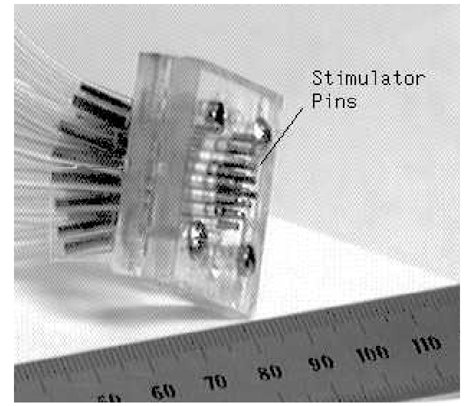
75% of light passes through

Tactile applications

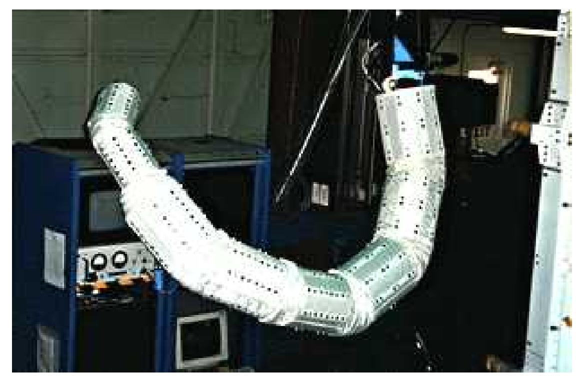
Medical teletaction interfaces

daVinci medical system





haptics

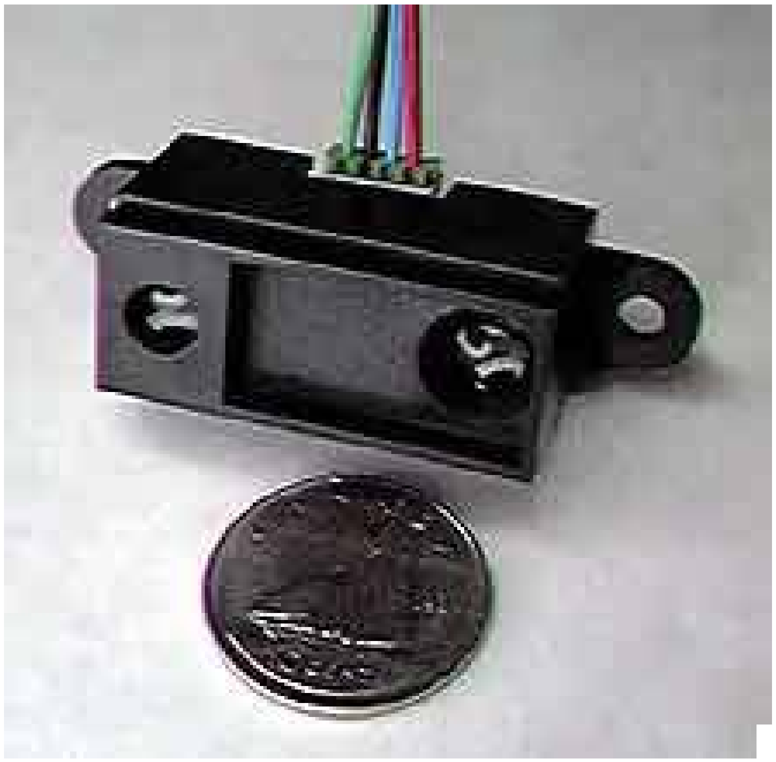


Robotic sensing

Merritt systems, FL

Infrared sensors

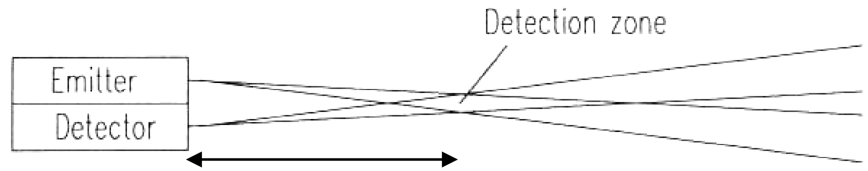
"Noncontact bump sensor"

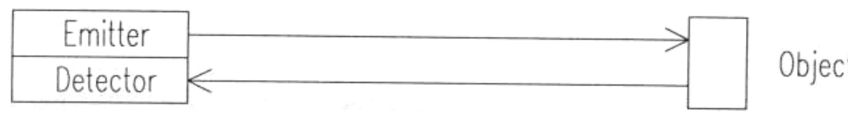


IR emitter/detector pair

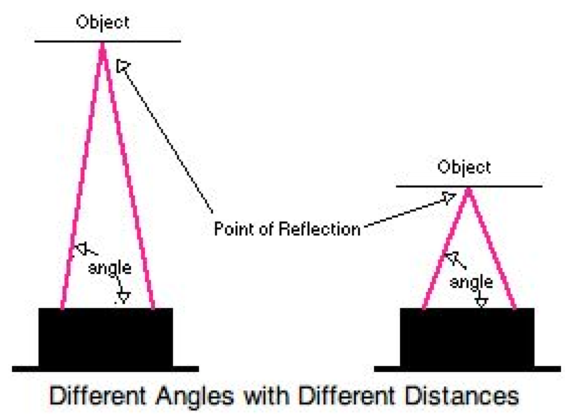
IR detector

"object-sensing" IR

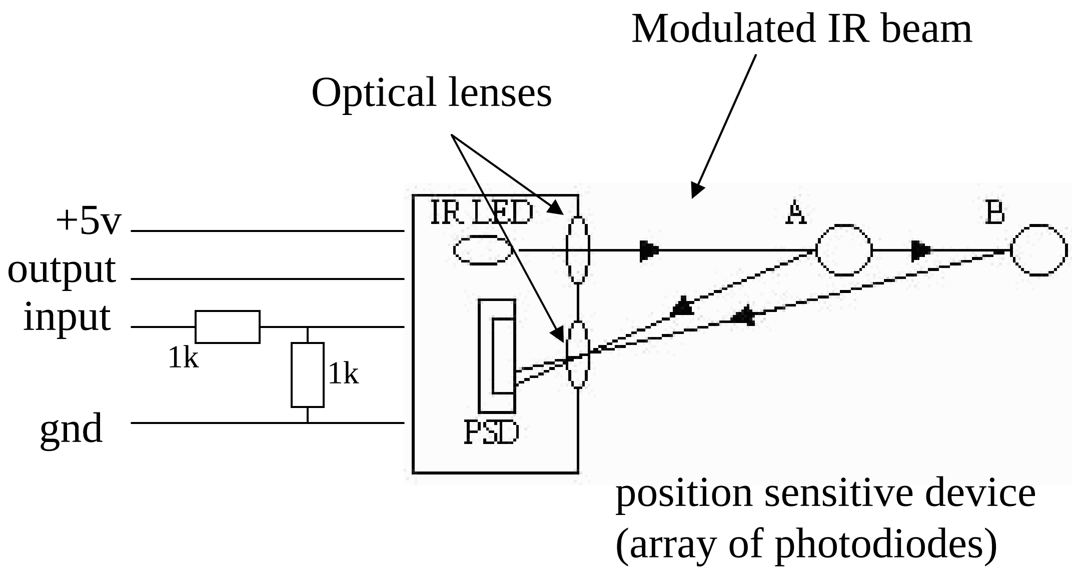




diffuse distance-sensing IR



Digital Infrared Ranging



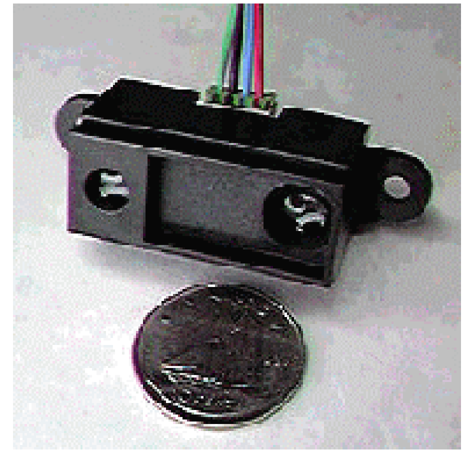
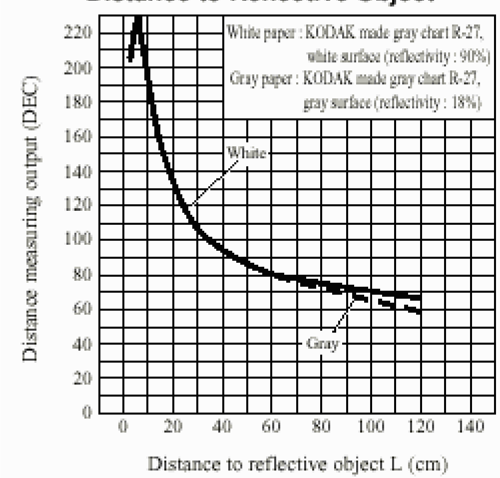


Fig. 1 Distance Measuring Output vs. Distance to Reflective Object



- Optics to covert horizontal distance to vertical distance
- Insensitive to ambient light and surface type
- Minimum range ~ 10cm
- Beam width ~ 5deg
- Designed to run on 3v -> need to protect input
- Uses Shift register to exchange data (clk in = data out)
- Moderately reliable for ranging

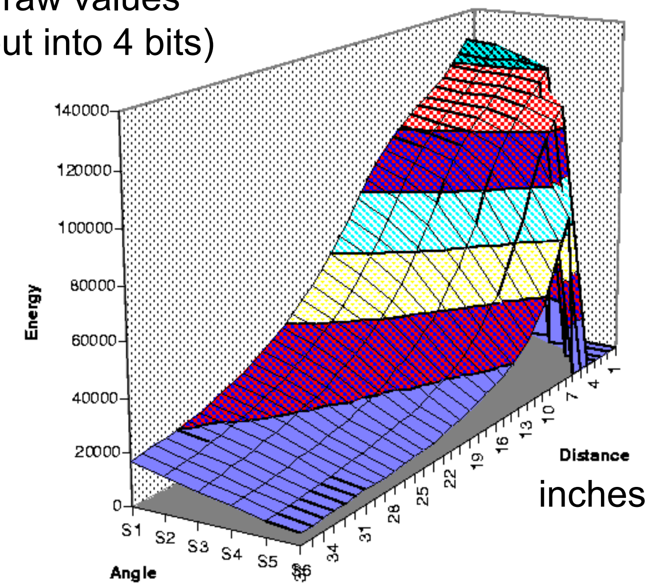
Infrared calibration

The response to white copy paper
(a dull, reflective surface)

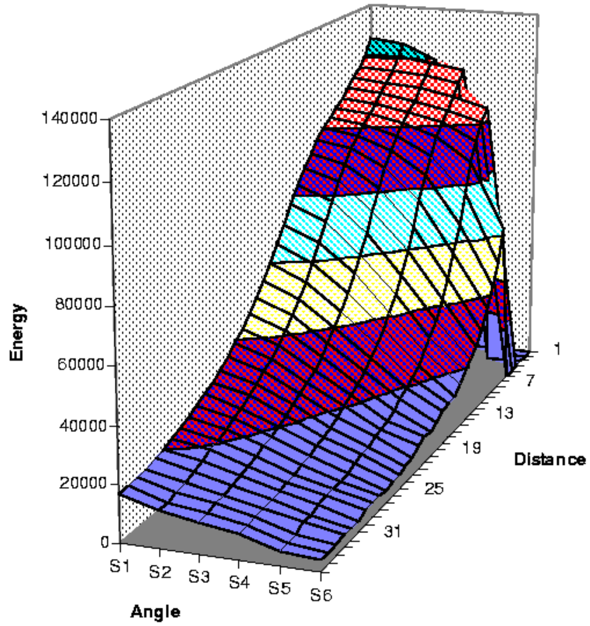
(1) sensing is based on light intensity.

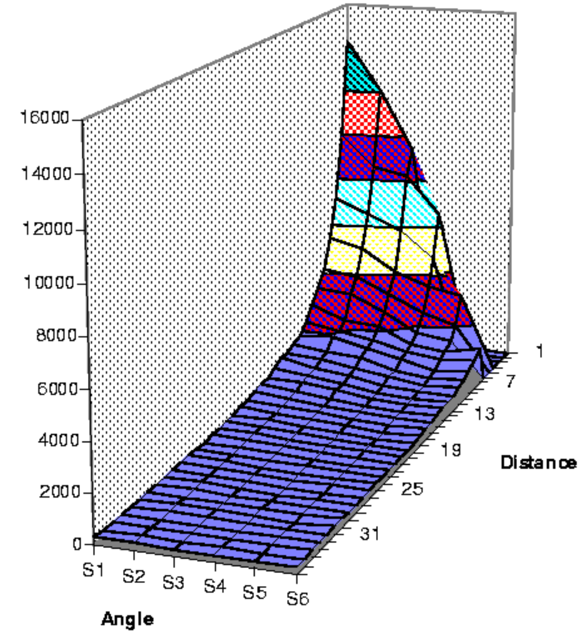
(2) sensing is based on angle received.

raw values
(put into 4 bits)



15º increments





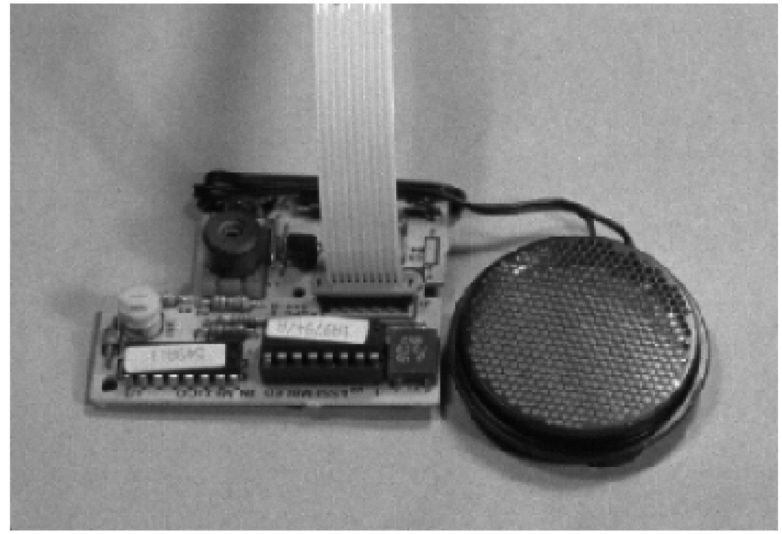
in the dark

fluorescent light

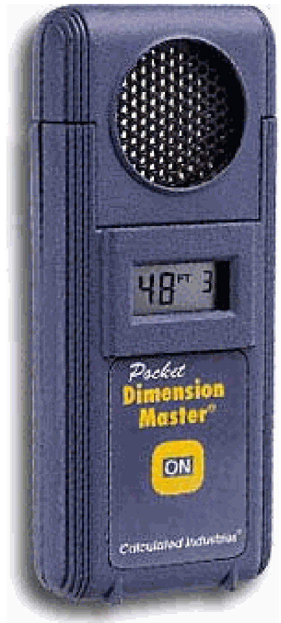
incandescent light

Polaroid Ultrasonic Sensor

•



Mobile Robot

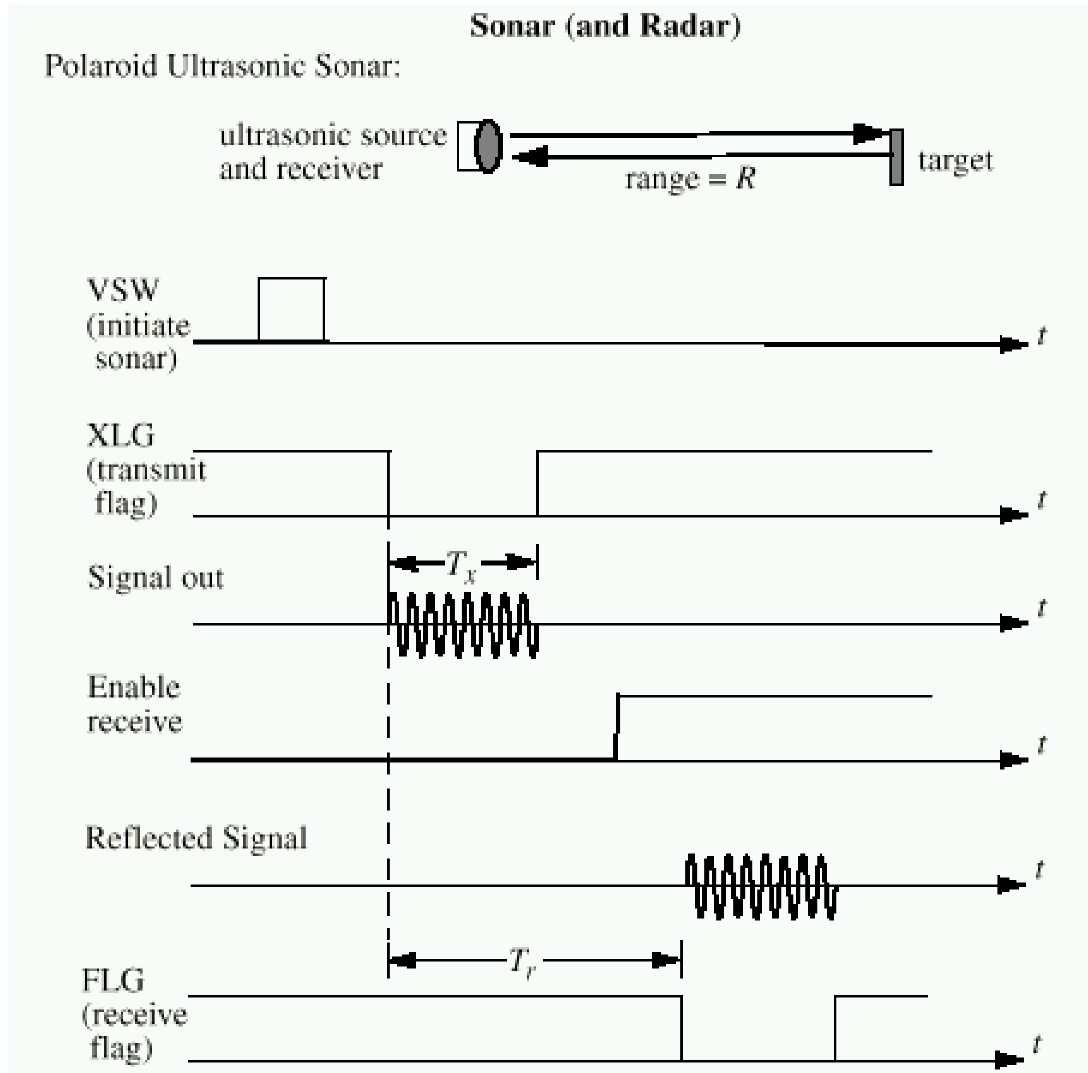


Electric
Measuring Tape

Focus for Camera

http://www.robotprojects.com/sonar/scd.htm

- Digital Init
- Chirp
  - 16 high to low
  - -200 to 200 V
- Internal Blanking
- Chirp reaches object
  - 343.2 m/s
  - Temp, pressure
- Echoes
  - Shape
  - Material
- Returns to Xducer
- Measure the time

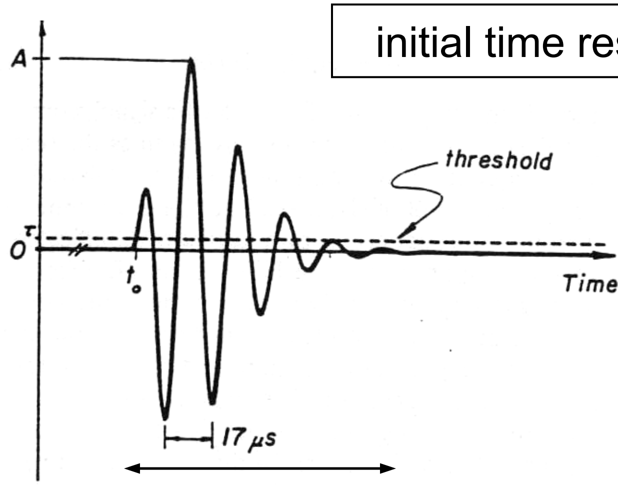


Issues

- Azimuth Uncertainty
- Specular Reflections
- Multipass
- Highly sensitive to temperature and pressure changes
- Minimum Range

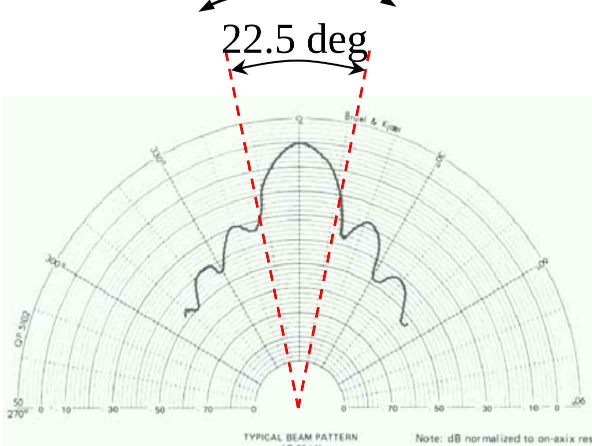
Sonar modeling

initial time response

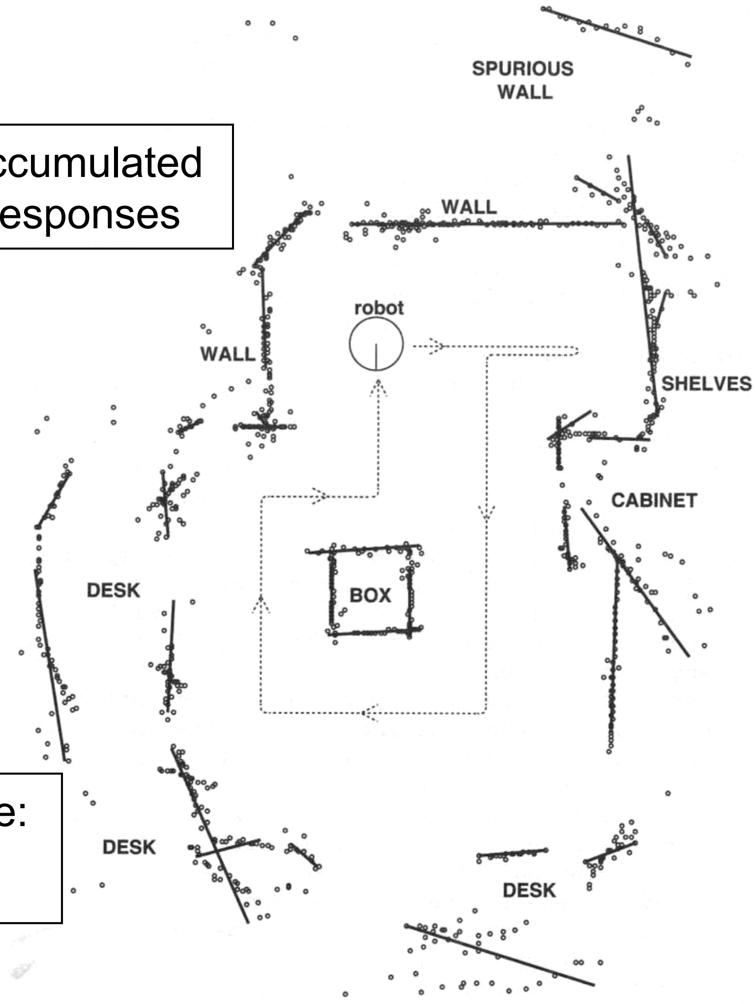


blanking time

cone width



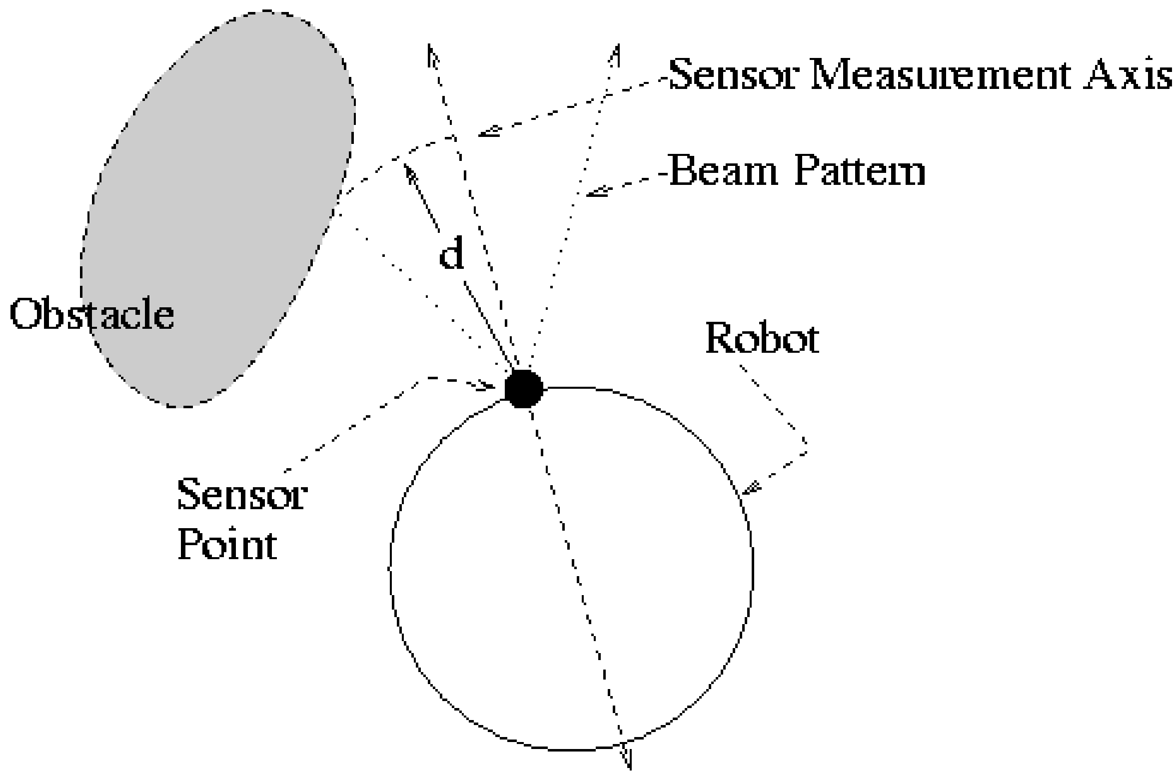
accumulated responses



spatial response:

Not Guassian!

(Naïve) Sensor Model

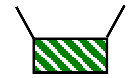


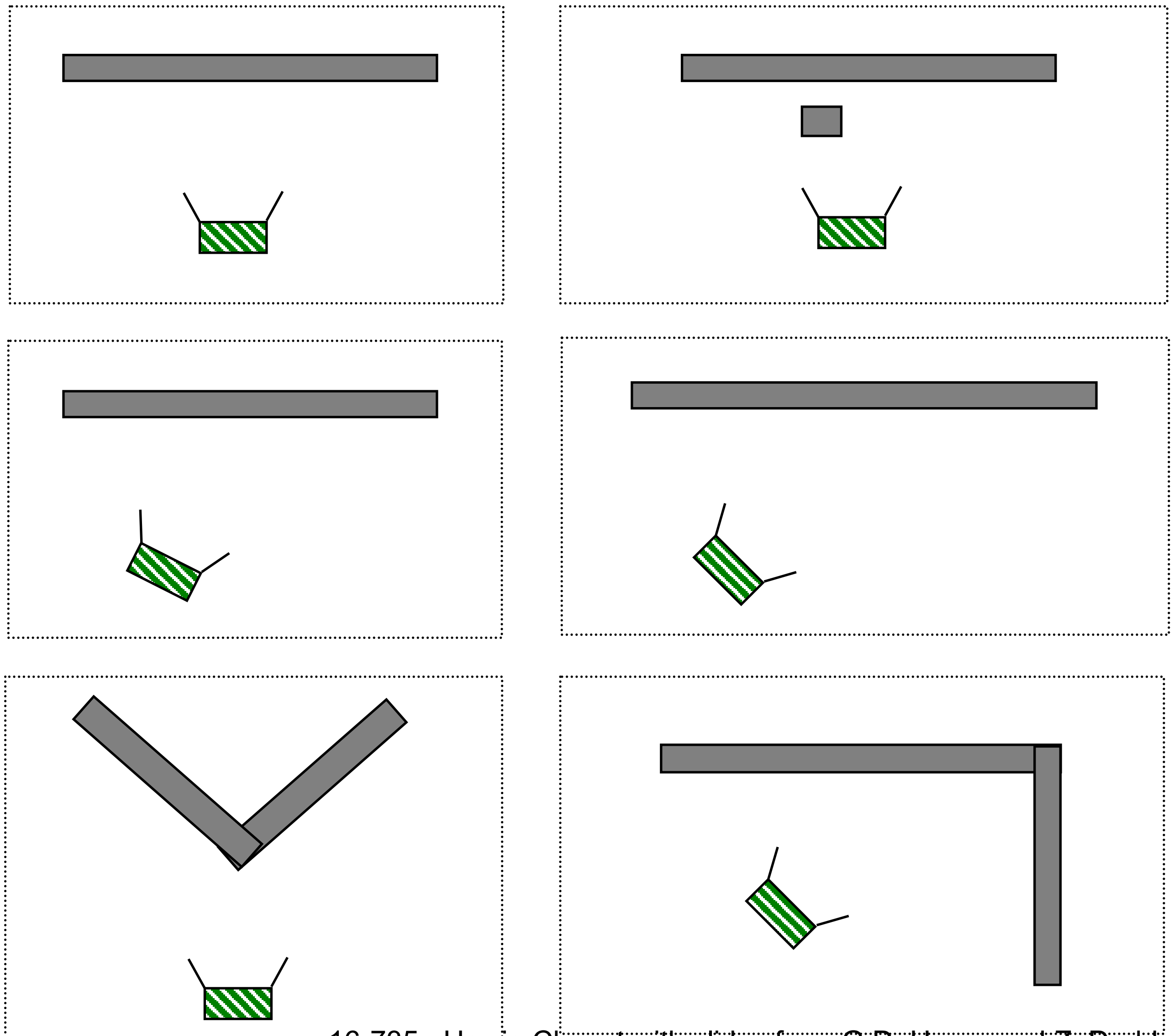
Acoustic Physics

- In air, sound dissipates proportionally to distance squared
  - Sound also attenuates according to humidity and temperature
- Solid objects are acoustic reflectors
- Due to the relatively large wavelength of sound, most solid objects are acoustic mirrors

Sonar effects


walls (obstacles)


sonar

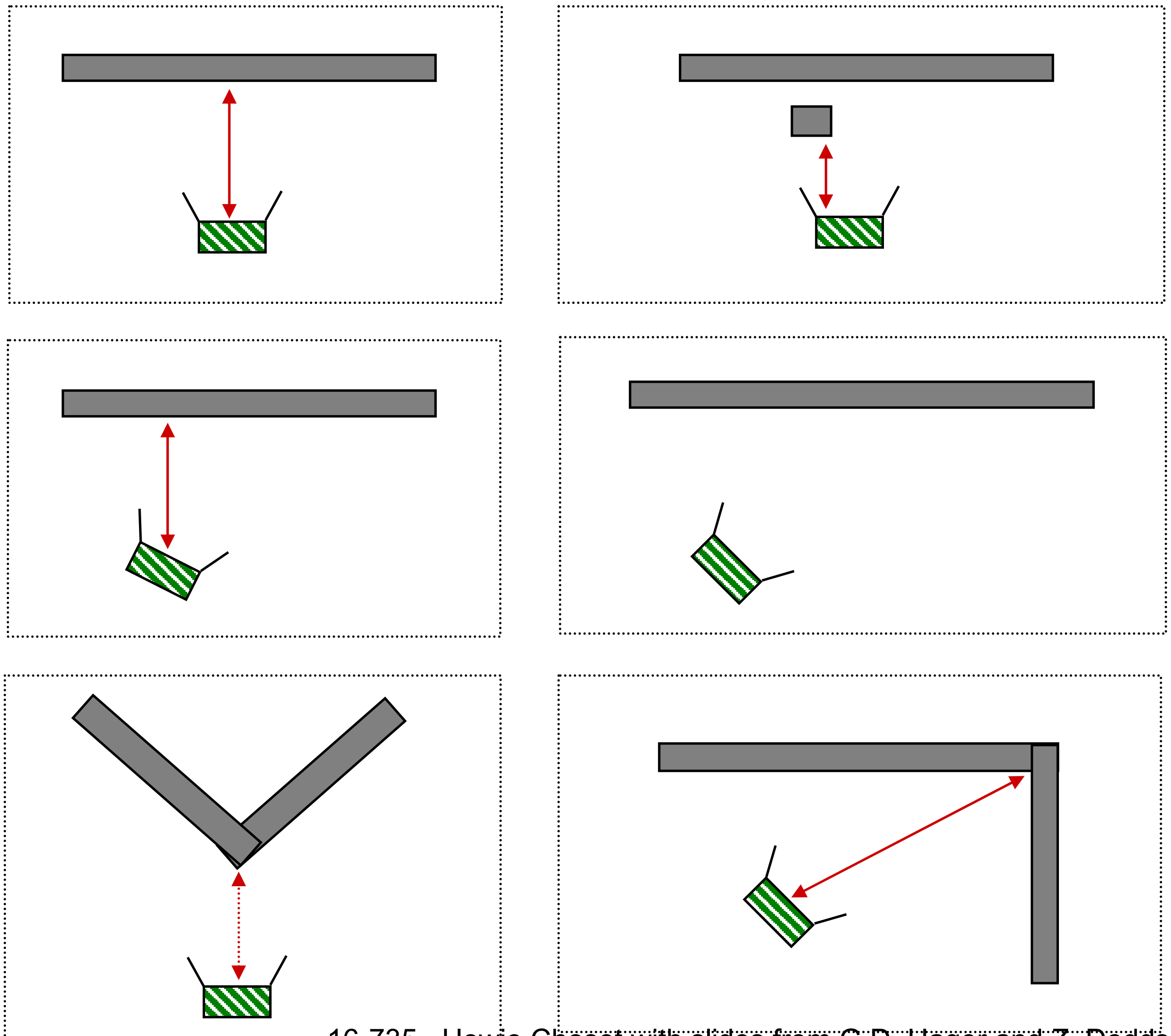


Draw the range reading that the sonar will return in each case…

Sonar effects

walls (obstacles)

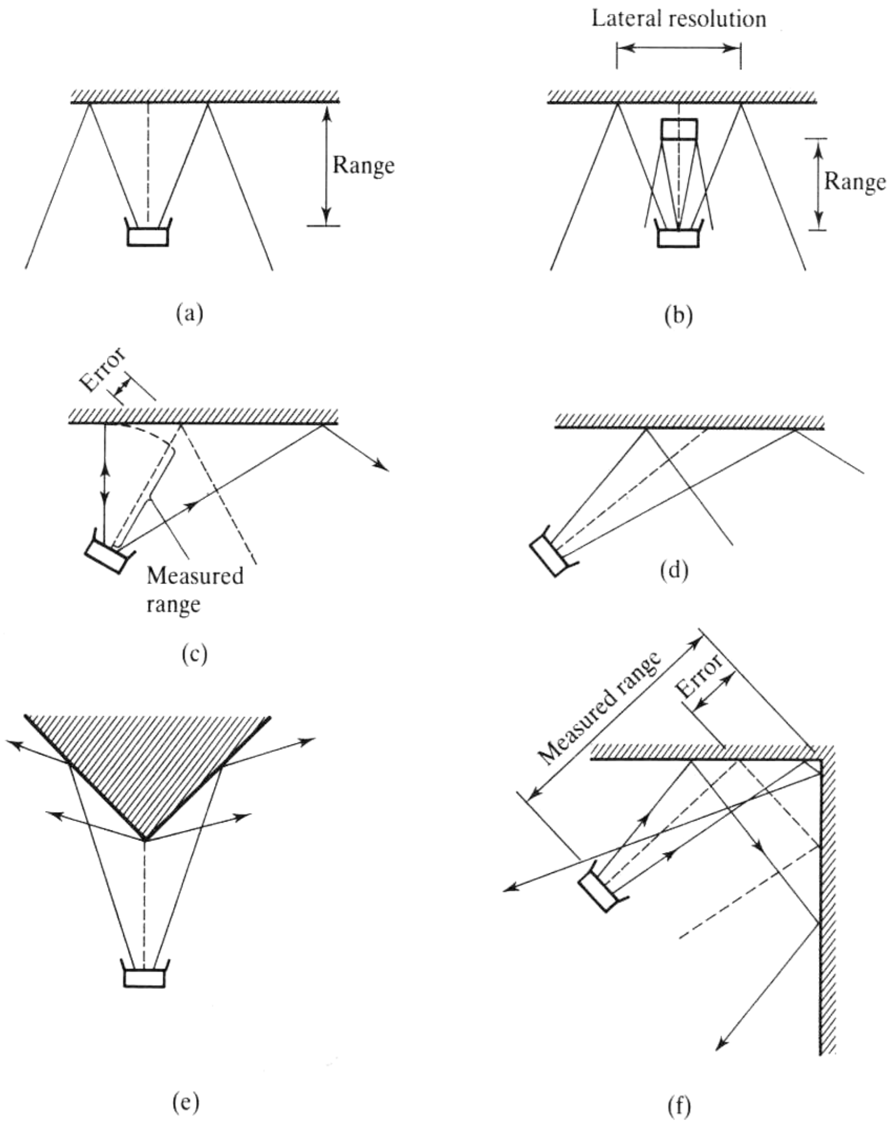
sonar



Draw the range reading that the sonar will return in each case…

holding a sponge…

Sonar effects



(a) Sonar providing an accurate range measurement

(b-c) Lateral resolution is not very precise; the closest object in the beam's cone provides the response

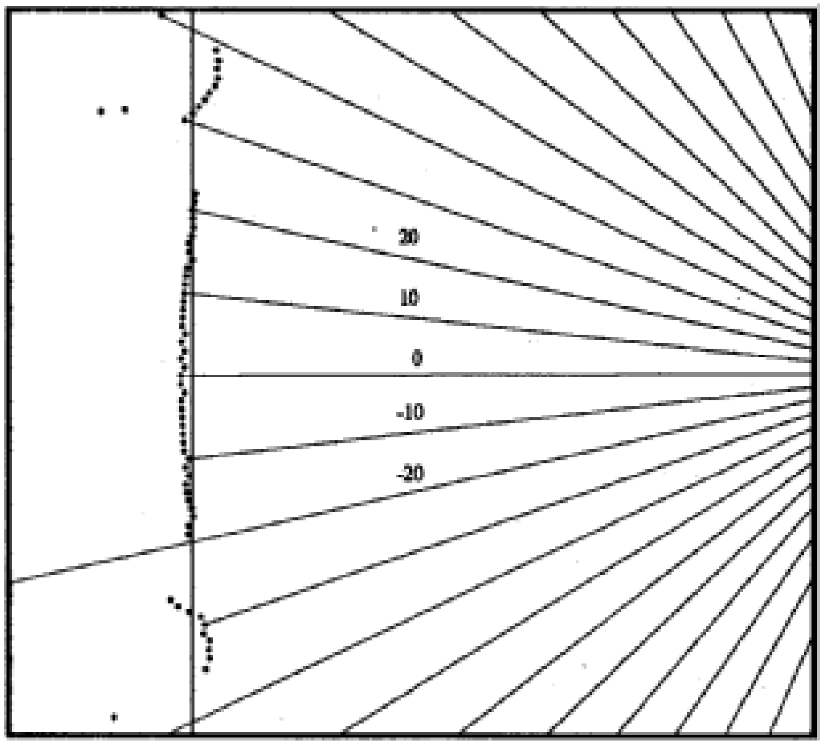
(d) Specular reflections cause walls to disappear

(e) Open corners produce a weak spherical wavefront

(f) Closed corners measure to the corner itself because of multiple reflections --> sonar ray tracing

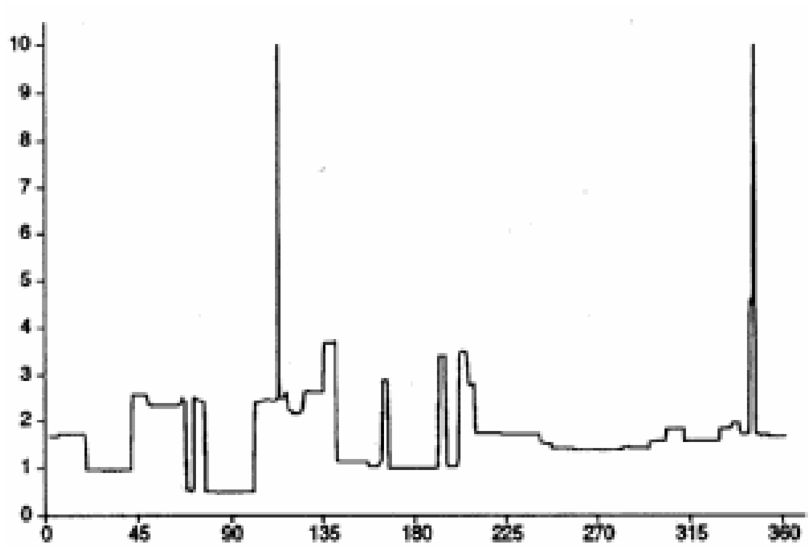
resolution: time / space

Region of Constant Depth (Leonard)
Typical Wall Response



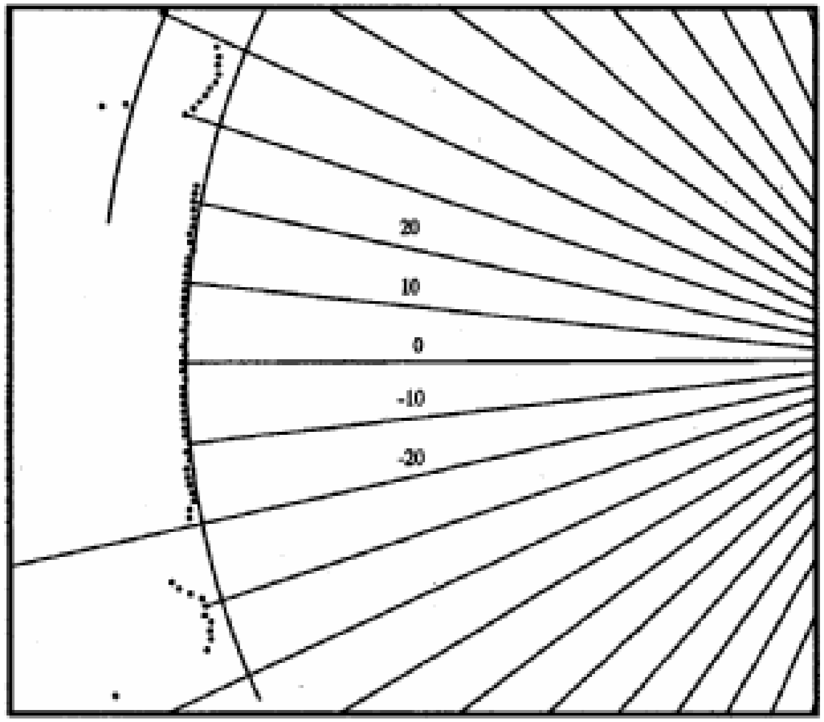
- It is tempting to try to fit a line to the data and call that the wall
- Many approaches have done this in the past

A Different Representation



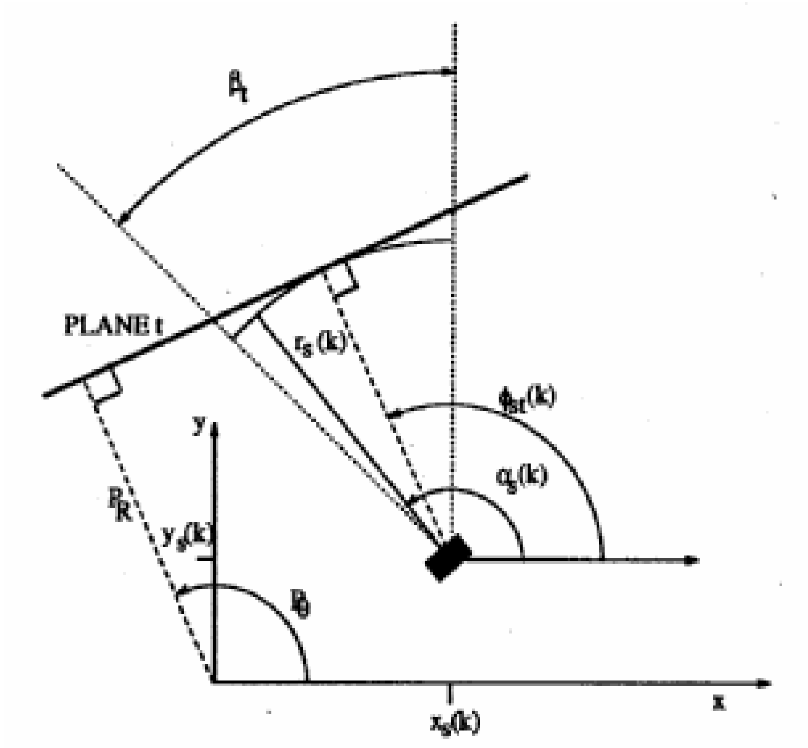
- Plot of range vs angle for the original scan.
- There are a number of regions where the depth remains constant
- These should correspond to arcs in a Cartesian representation

Region of Constant Depth



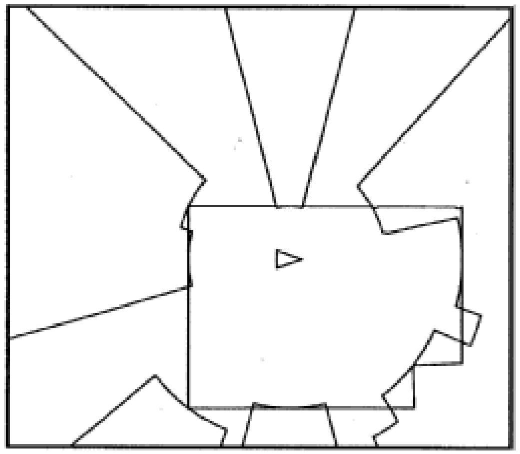
- An arc is fit through the closest response belonging to the RCD
- This should correspond to a perpendicular surface

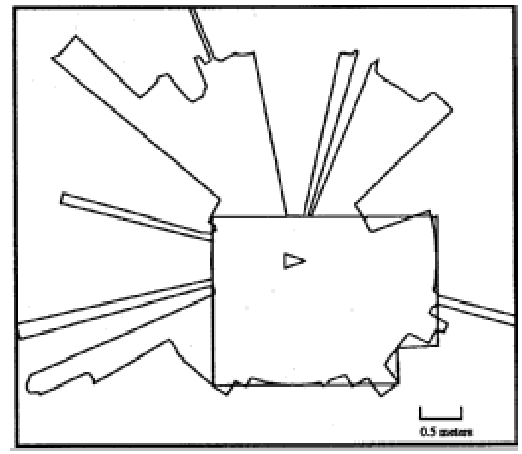
Wall Response Prediction



- Perpendicular line to the target is the center of the RCD
- The RCD should extend in either direction $\beta_t/2$ which corresponds to the beam width as well as the ability of the target to reflect acoustic energy

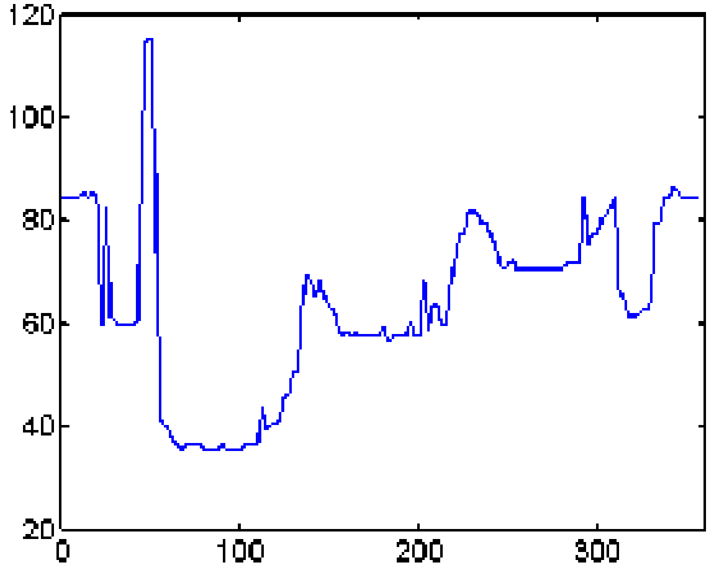
Compare Prediction with Data

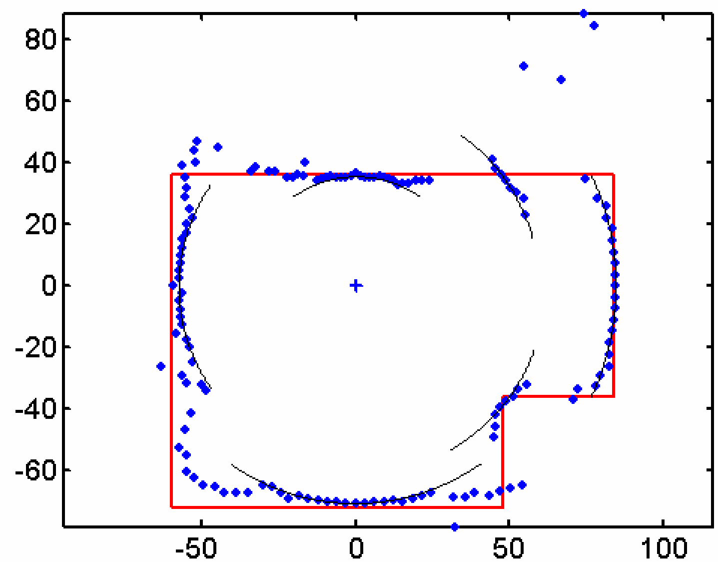




- Close correspondence
- Our model seems sufficient

But does it *really* work?



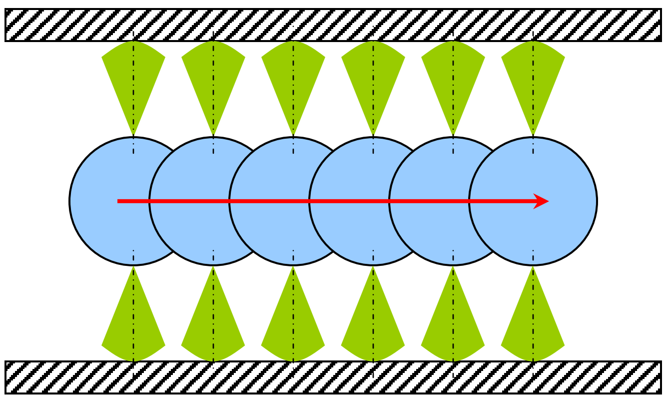


- 240 sonar scans rotating ~1.5 degrees between each
- Decent results for a cardboard room and 1 inch resolution

ATM: Improve Azimuth Resolution

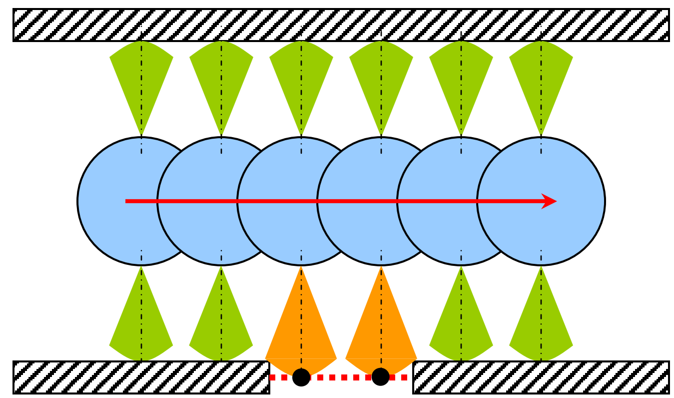
Wide Sonar Beam Can Merge Obstacles

<u>Passing a corridor</u>



No Problem!

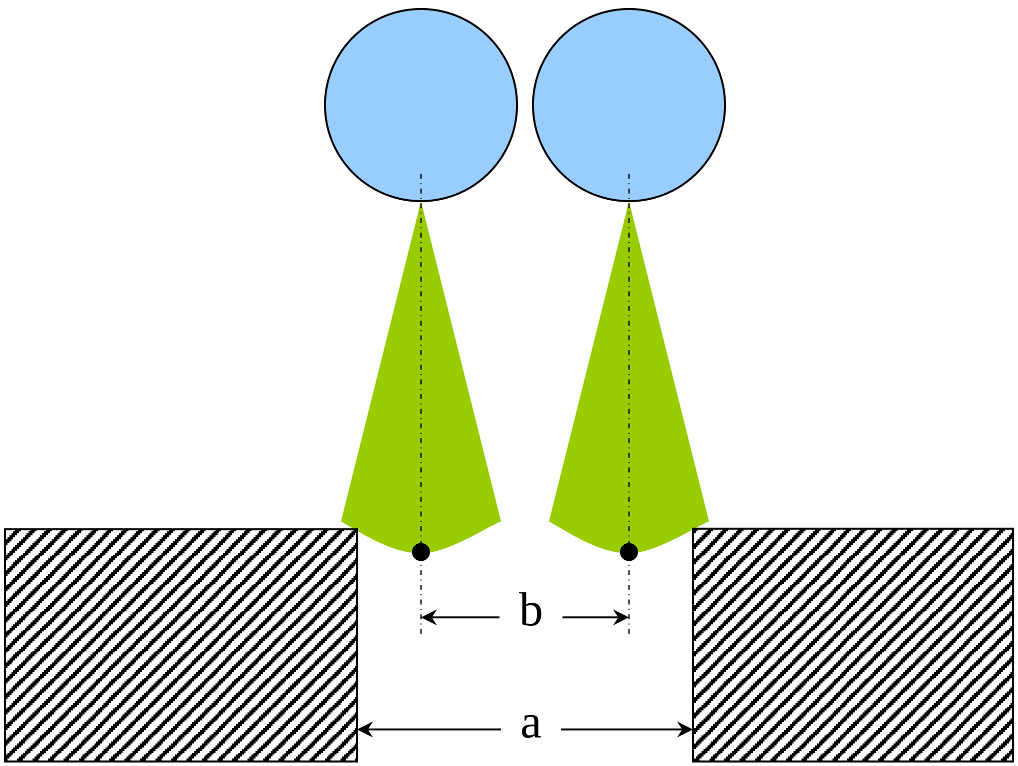
<u>Neglecting the corner</u>

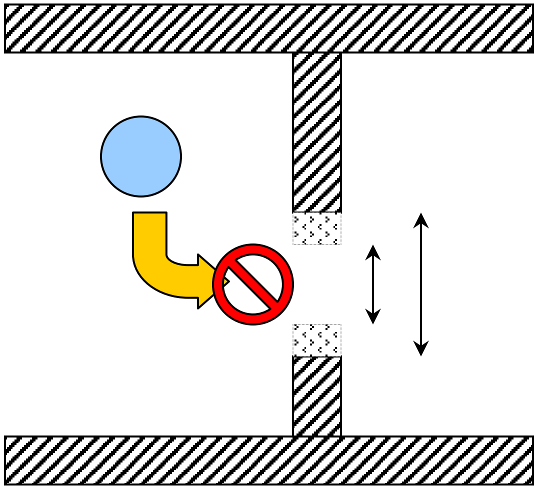


Overlook the Gap!

Center Line Model – Problem

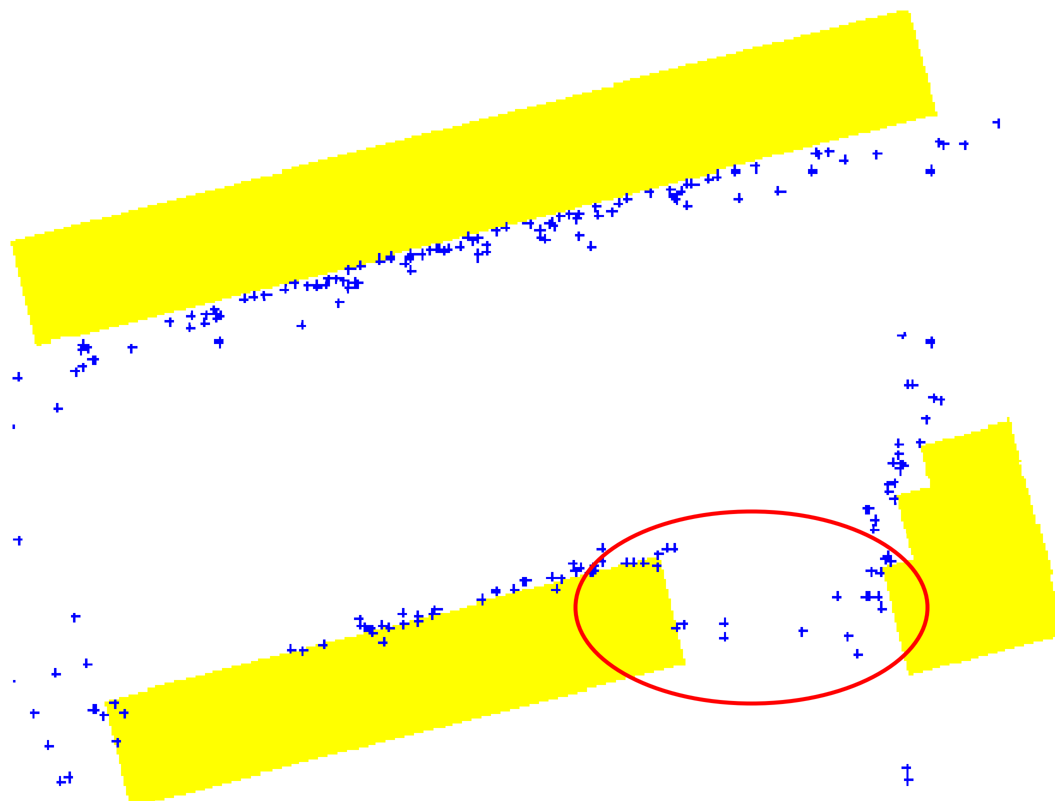
- The robot might believe the passageway is too narrow for it to pass through





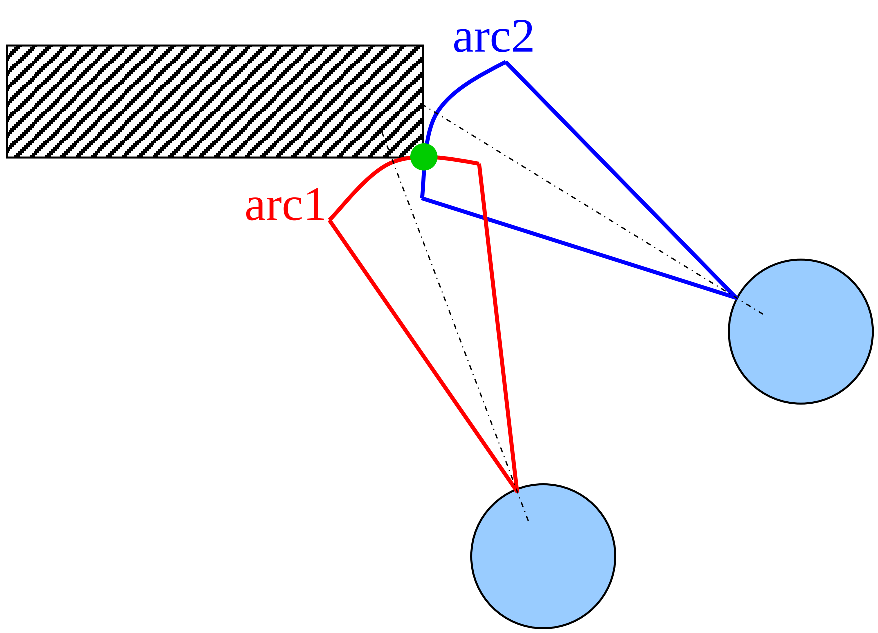
Center Line Model – Real Data

- Center Line Model gives the robot a false impression of the world
  - The robot perceives that an opening does not exist!!!



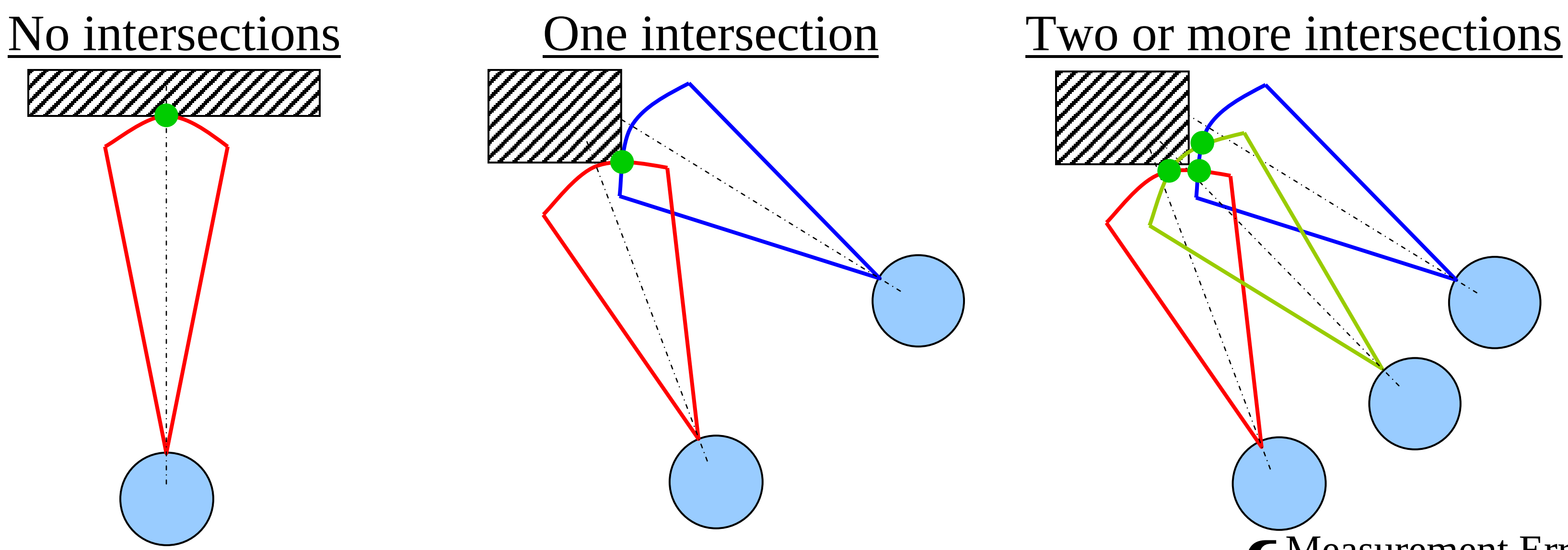
ATM: Arc Intersection

- Arc Intersection of two or more arcs
  - The point of reflection can lie anywhere on arc 1, arc 2, …, arc n
  - If many sonar arcs intersect at one point, the probability that it is in an obstacle becomes quite high



ATM Median Intersection

- The median of all intersections on one arc serves as a good approximation of the point
  - Median is in the cluster of intersections
  - Median is robust with respect to noise



Assume <u>on Center Line</u>

Assume <u>at the Intersection</u>

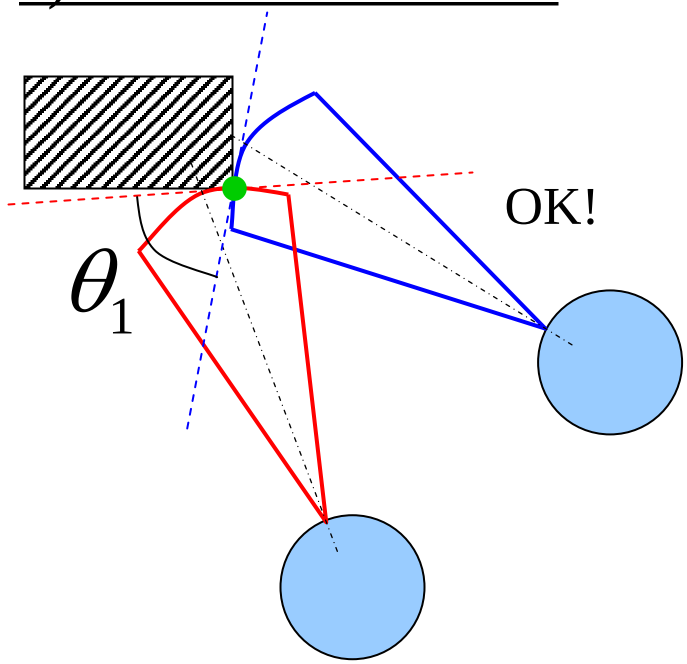
Errors { Measurement Error? Dead-Reckoning? Some Noise?

➔ <u>Take a Median</u>

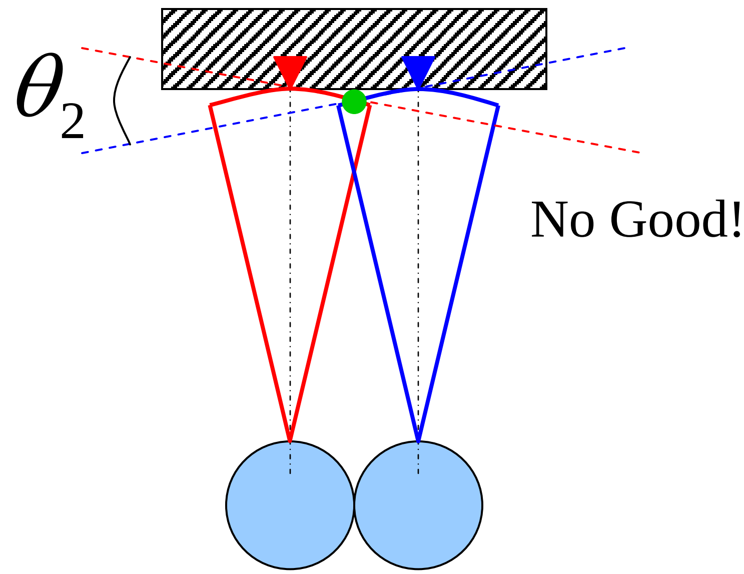
ATM Transversal Intersection

- Robot do not consider all intersections, just “Stable” intersections

a) Stable intersection



b) Unstable intersection

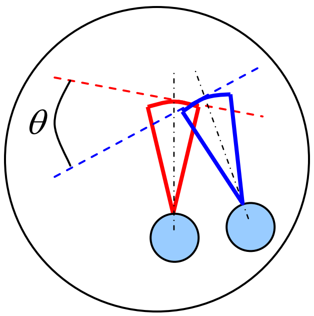


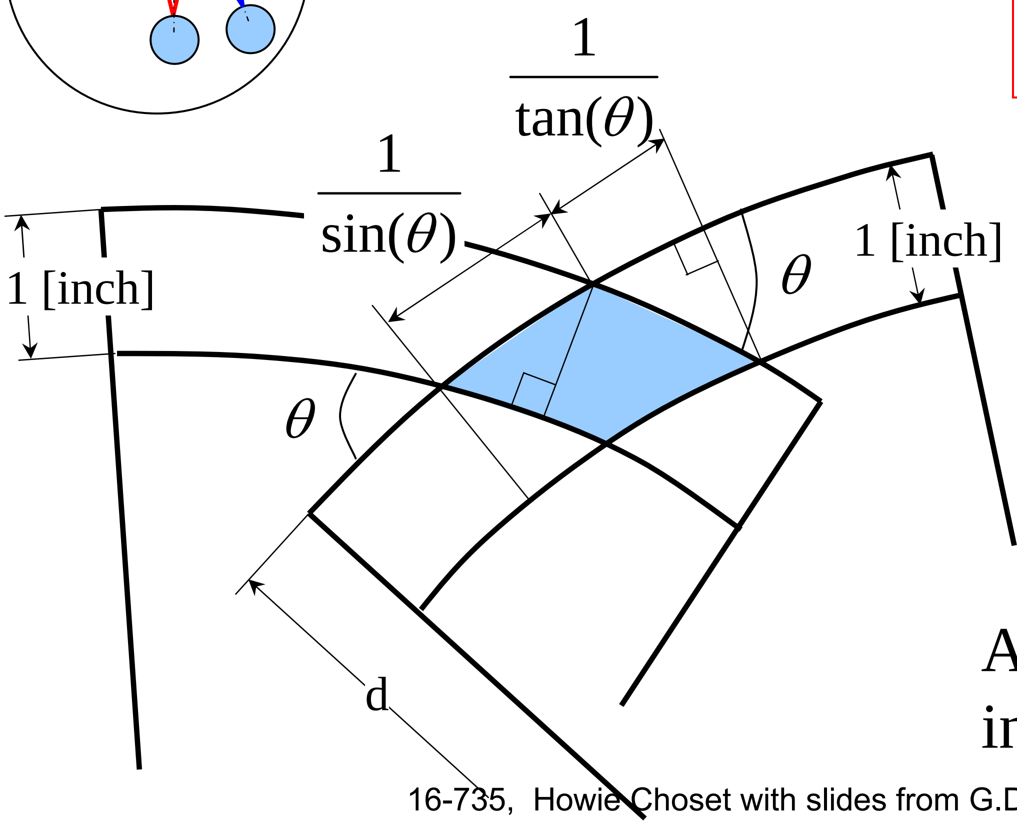
a) is more STABLE than b). Because $\theta_1 > \theta_2$

➔ How big does θ have to be?

Transversal Intersection – Why 30 deg?

- It is enough for our Nomad robots to use 30 deg as the threshold for transversal intersections





$$Arc(d) = d \times \frac{22.5}{180}\pi[inch]$$

$$Len(\theta) = \frac{1}{\sin(\theta)} + \frac{1}{\tan(\theta)}[inch]$$

$$Arc(100'') = 100'' \times \frac{22.5}{180}\pi \approx 39.2[inch]$$

$$Len(30°) = \frac{1}{\sin(30°)} + \frac{1}{\tan(30°)} \approx 3.73[inch]$$

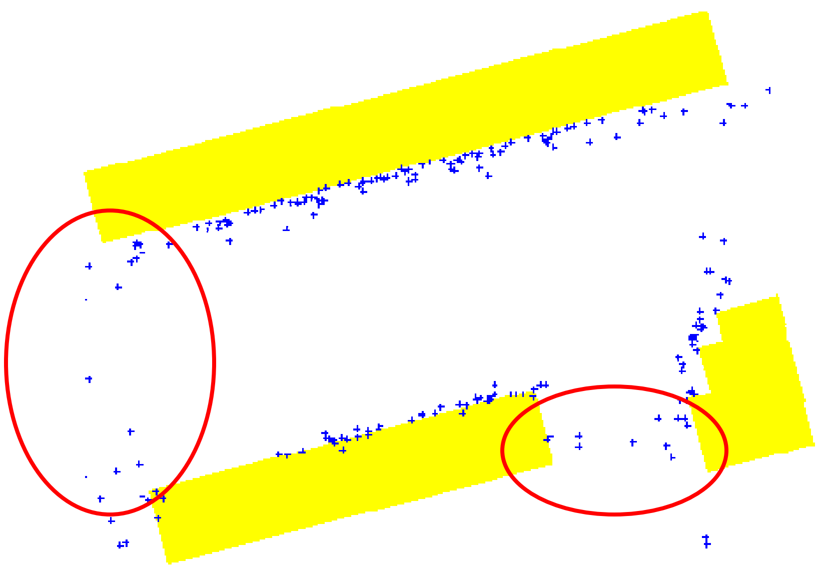
$$\frac{Len(30°)}{Arc(100'')} < \frac{1}{10}$$

At least a 10-fold improvement in resolution!! (at d=100 [inch])

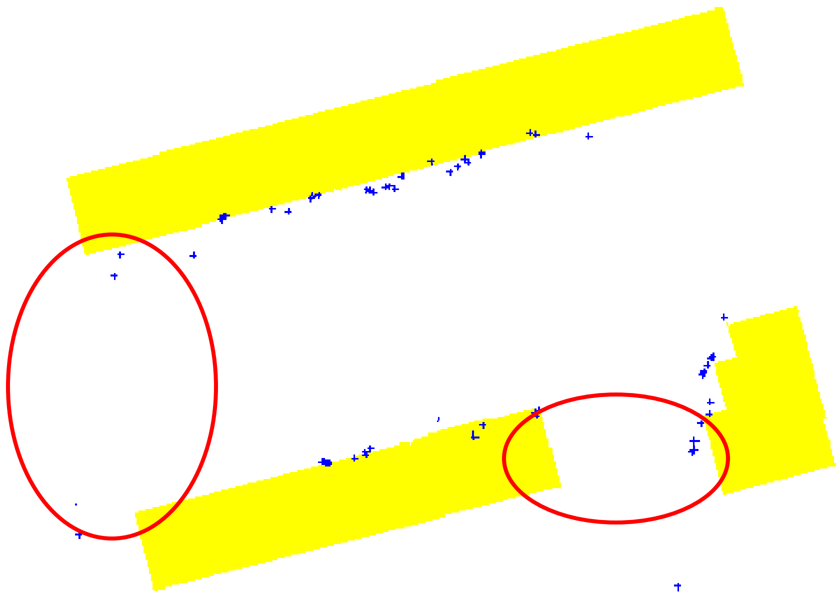
Real Data

- The points that ATM method generates present a more accurate view of the environment

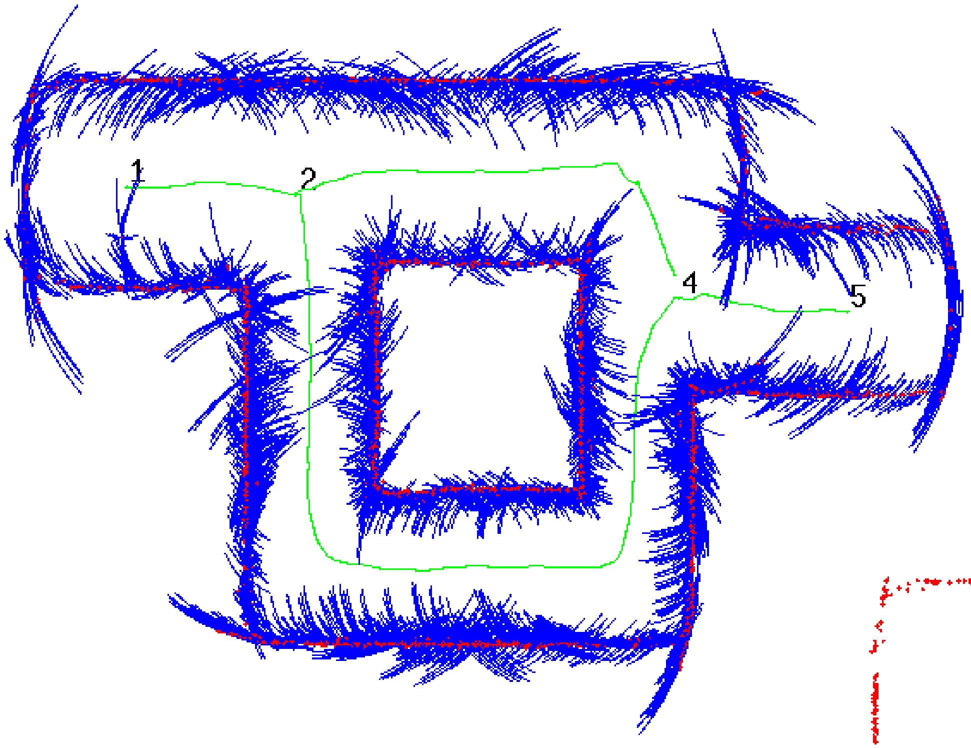
Center Line Model

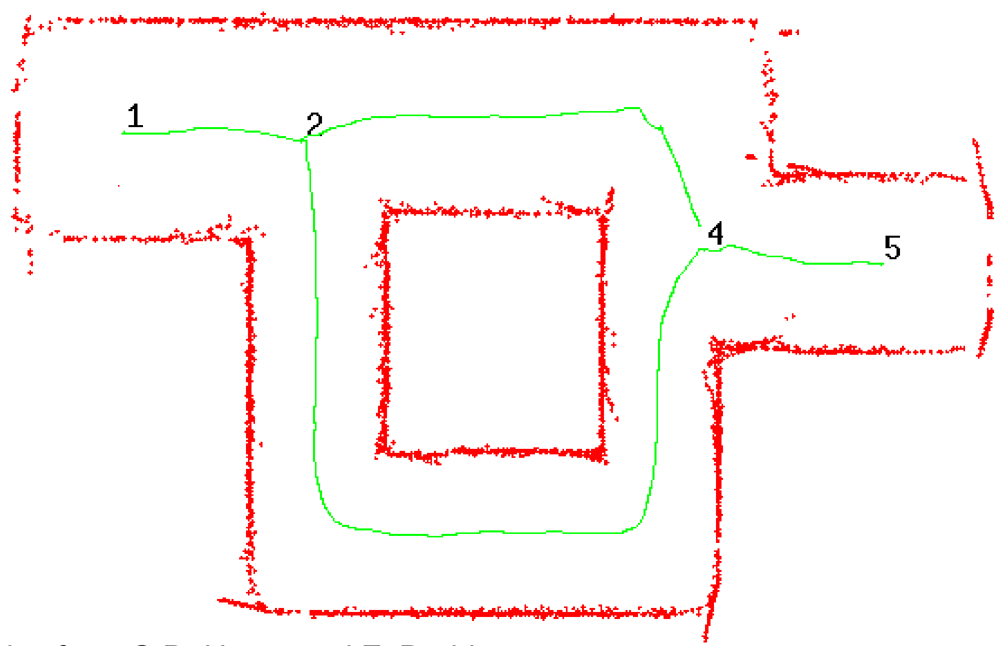


ATM Method



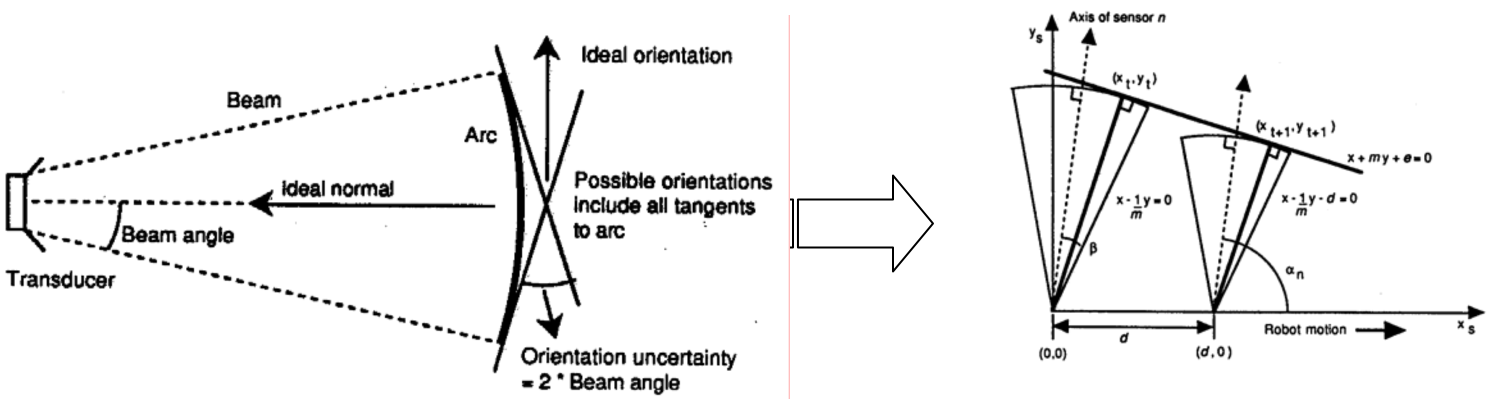
Mapping Example



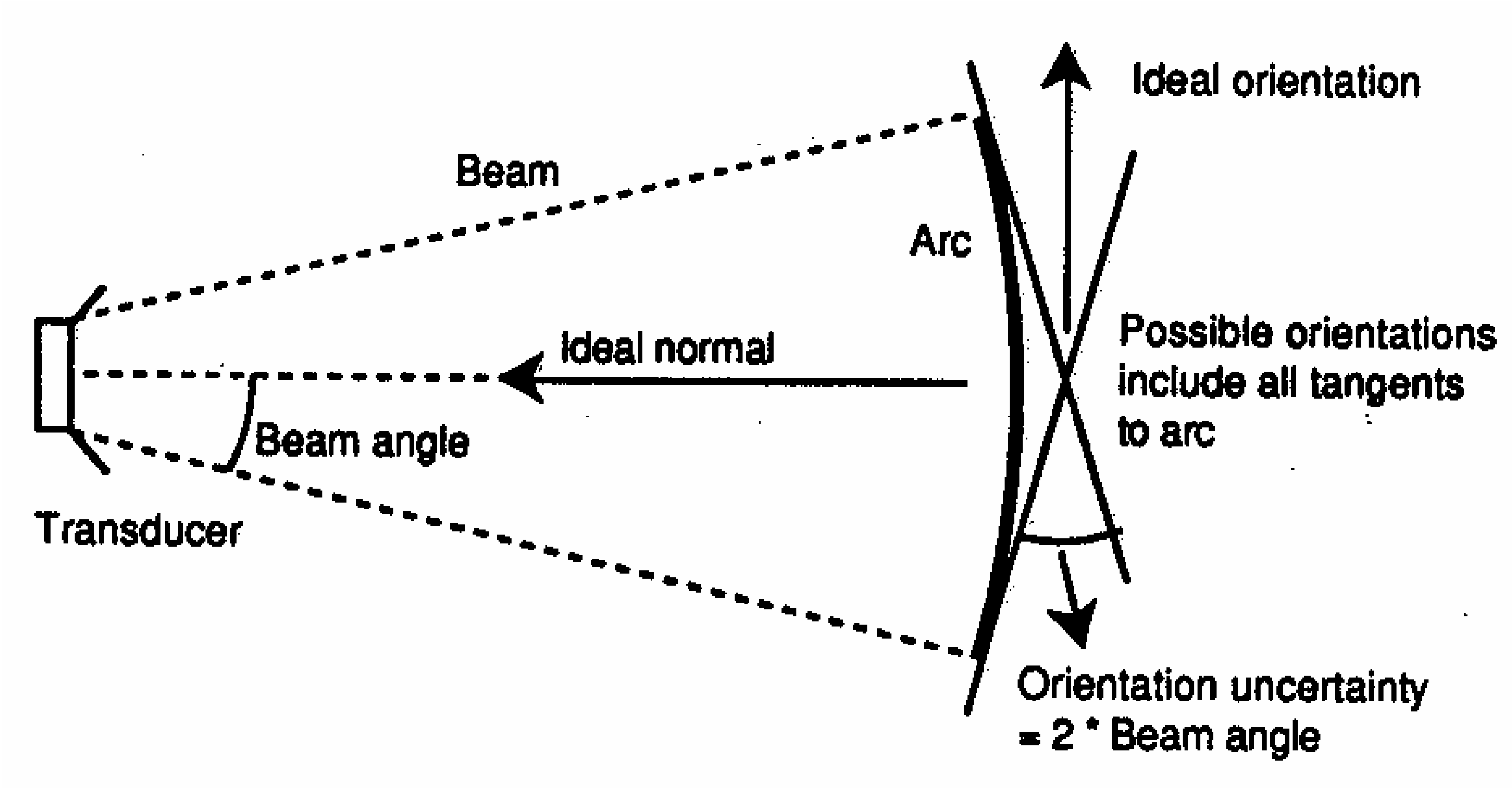


McKerrow Method Overview

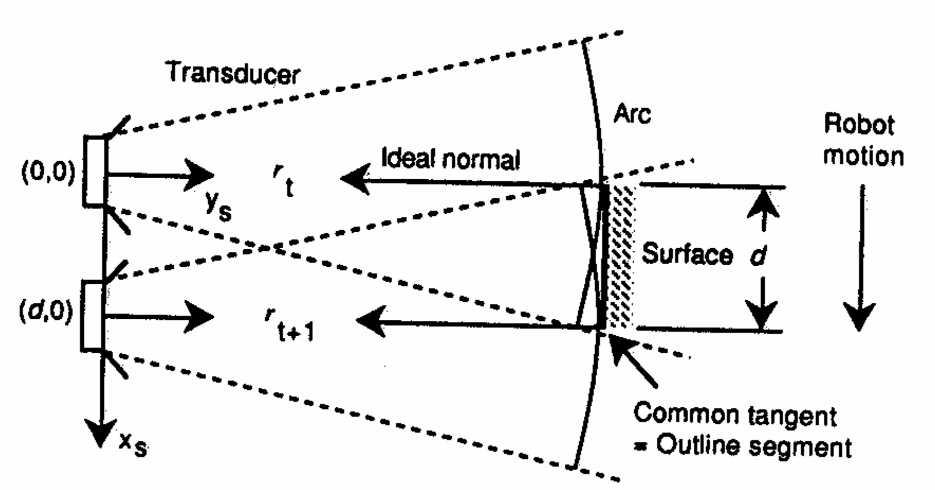
- Obstacle is tangent to closest sonar arc
- Use previous sonar sensor data
- Obstacle is tangent to both arcs
- Create outline segments
- Combine segments   Outlined obstacle

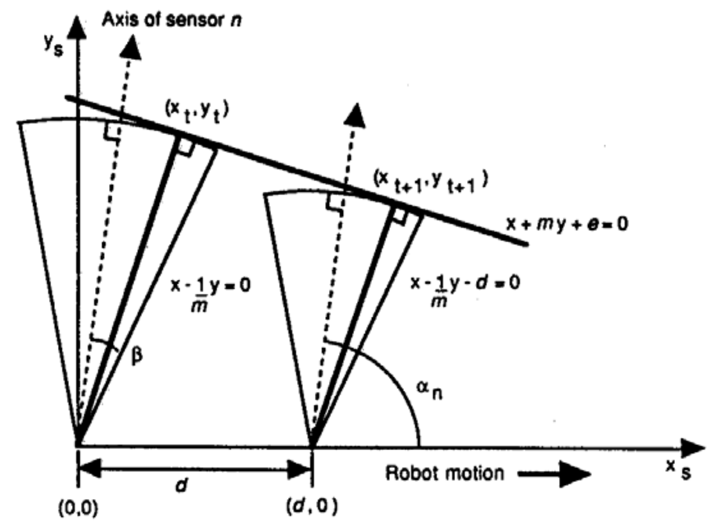


Single Sonar Reading



Two Sonar Readings

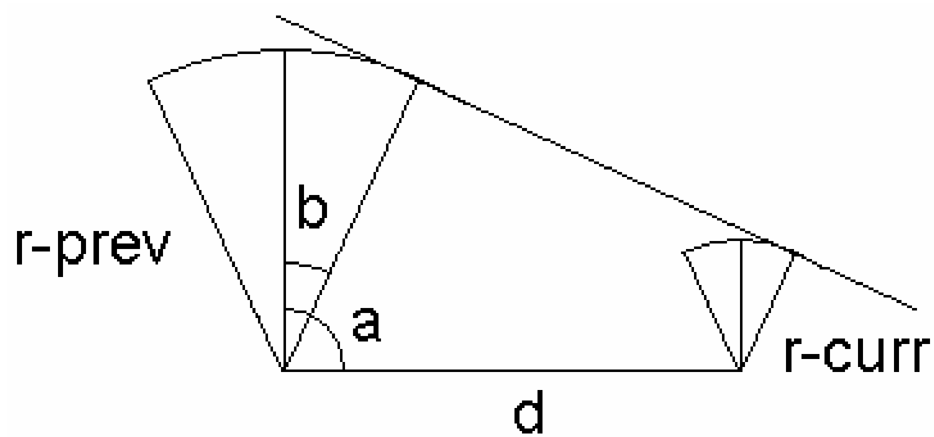




Common Tangent Condition

- d = distance between
  arc centers
- r = radius of arc
- b = beam angle
- a = angle of beam
  axis to x-axis

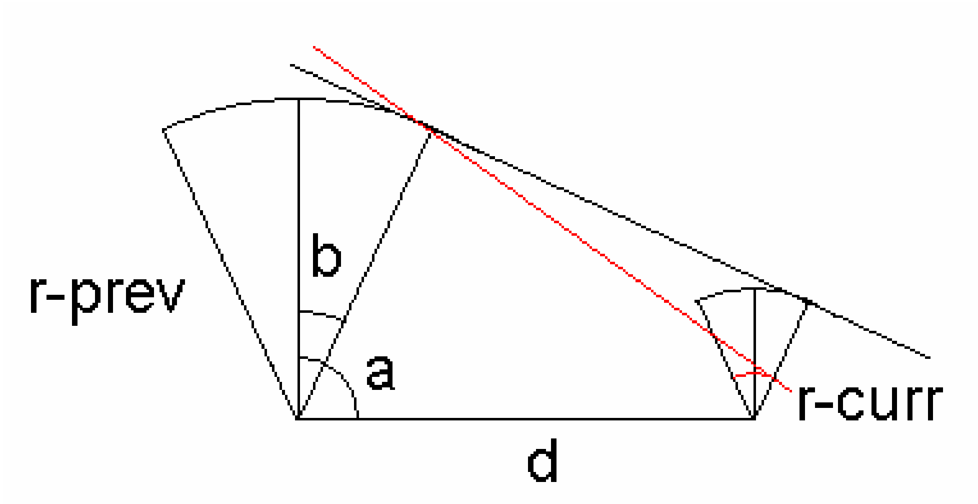
- $-d * \cos(a - b) < r_c - r_p < -d * \cos(a + b)$



Common Tangent Condition

- d = distance between
  arc centers
- r = radius of arc
- b = beam angle
- a = angle of beam
  axis to x-axis

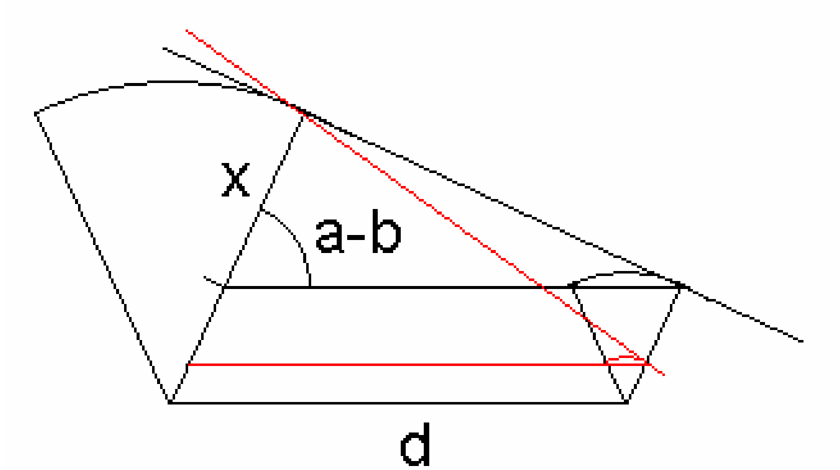
- $-d * \cos(a - b) < r_c - r_p < -d * \cos(a + b)$



Common Tangent Condition

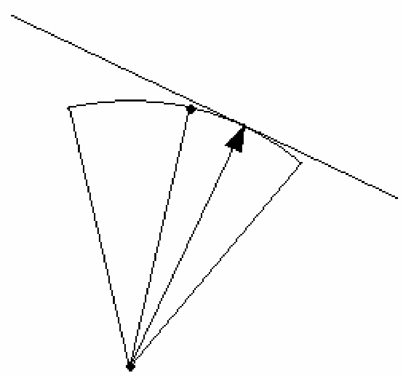
- d = distance between
  arc centers
- r = radius of arc
- b = beam angle
- a = angle of beam
  axis to x-axis

- $-d * \cos(a - b) < r_c - r_p < -d * \cos(a + b)$



Calculate Intersection Points

- Tangent to arc = Perpendicular
  to Range Vector
- Common tangent will have slope m
- Range vector will have slope –1/m
- Knowing arc centers and slope, equations describing range vectors can be found
- Using range vector equations and radii, points of intersection can be found



Example – World

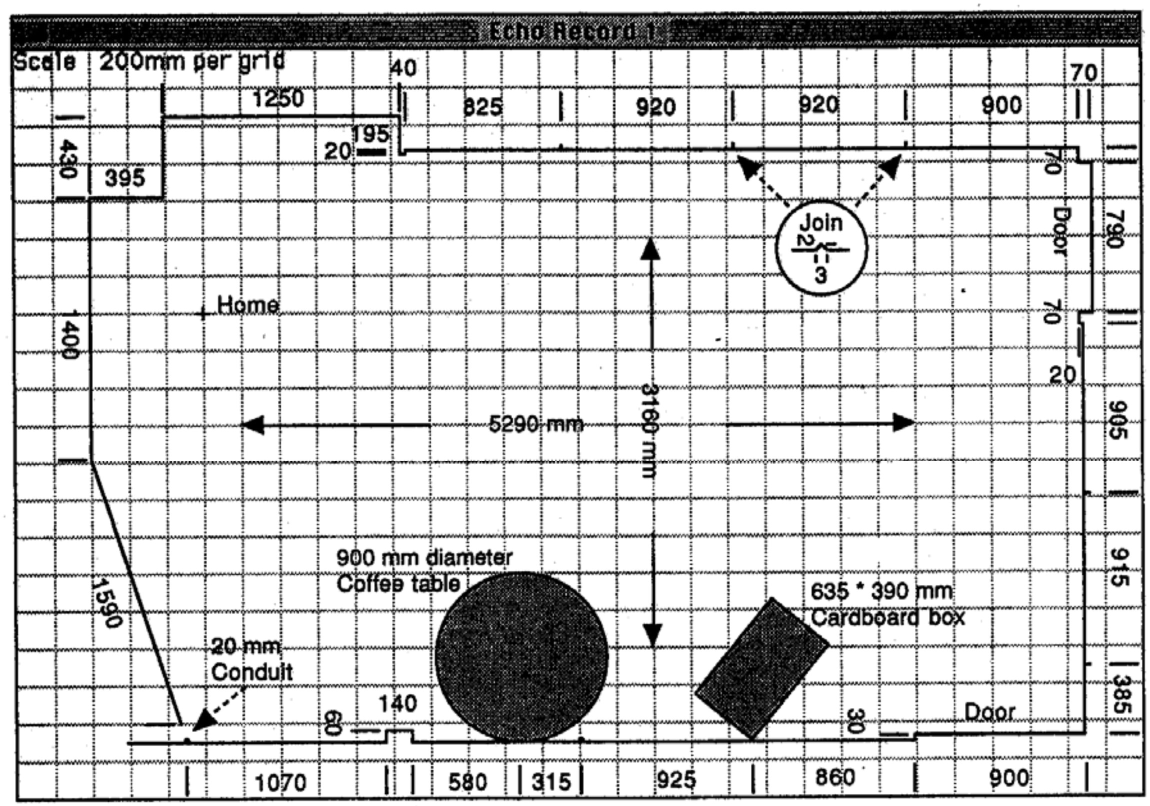


Fig. 6. Dimensions of room for experiments in millimetres.

Example – Cluttered

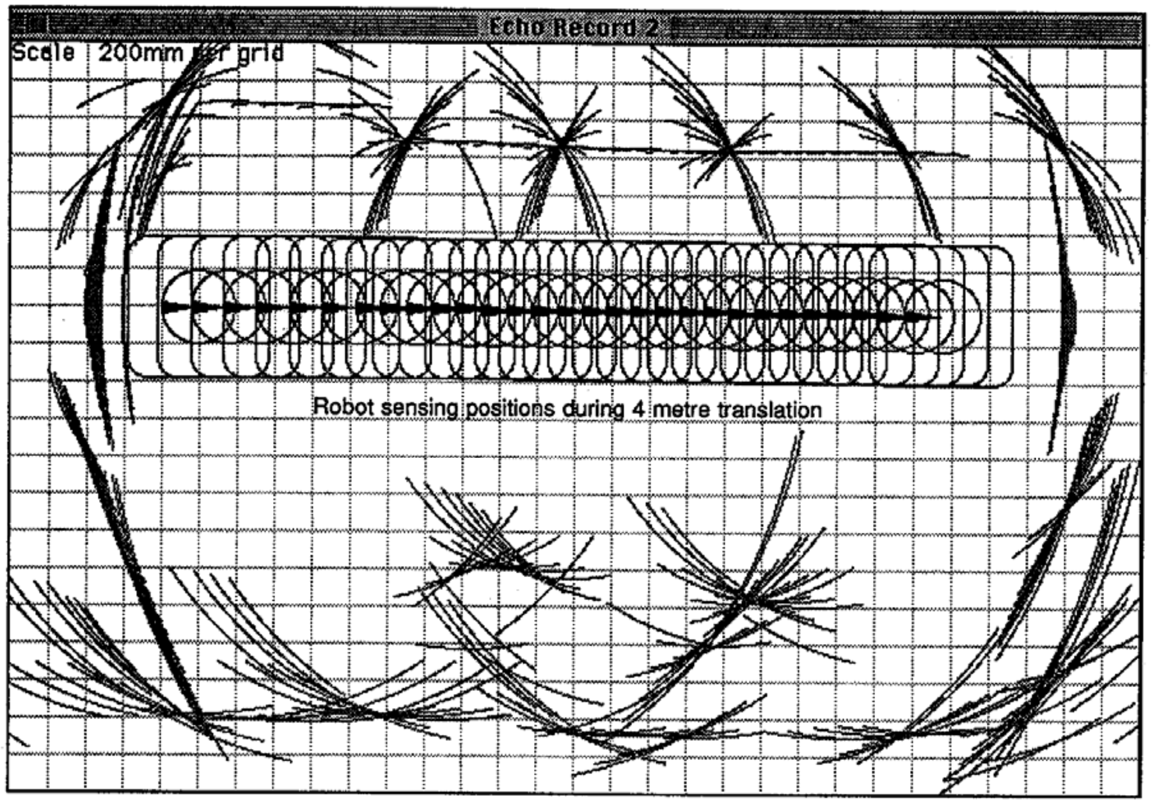


Fig. 8. Outline segment map derived from arc map in Fig. 7.

Example – Clearer Picture

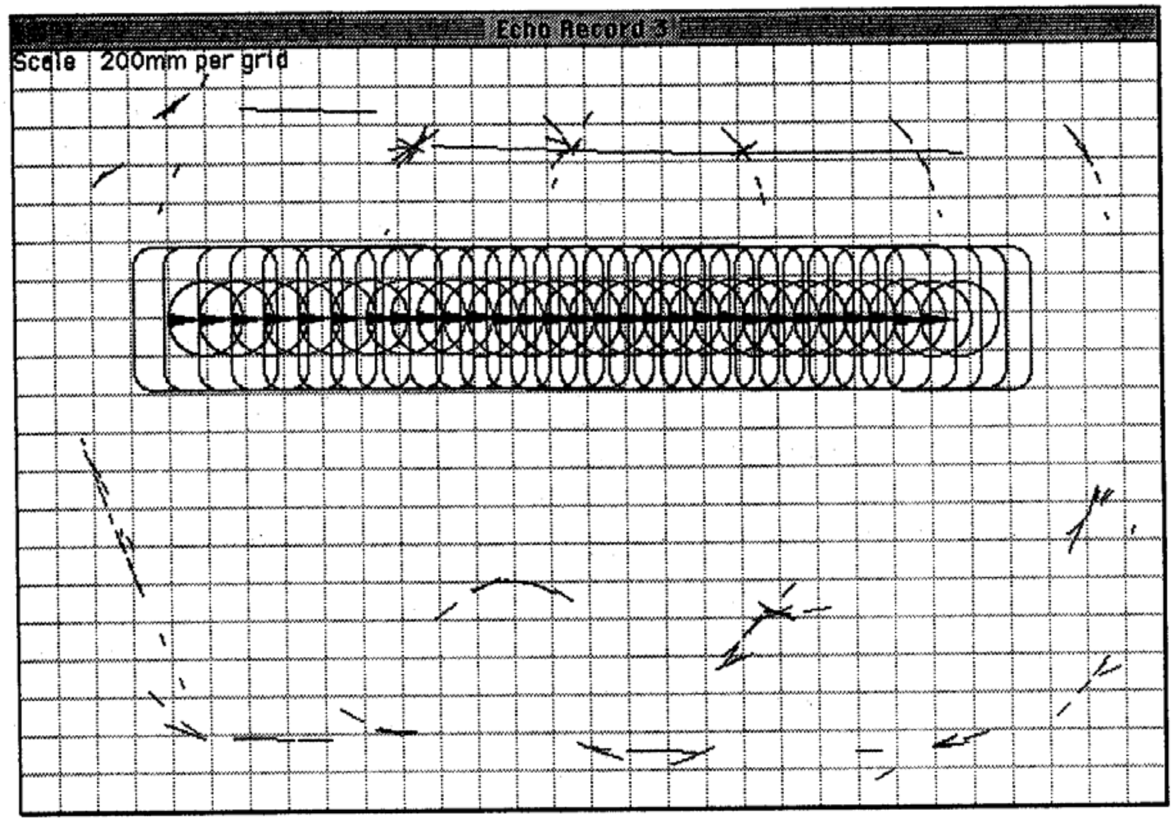
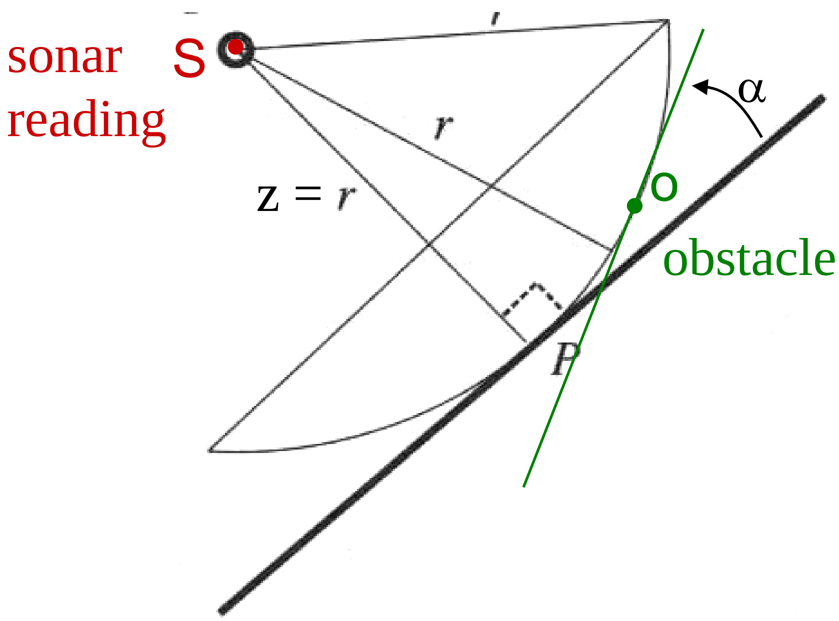


Fig. 7. Arc map of room in Fig. 6.

Sonar Modeling

response model (Kuc)

$$h_R(t, z, a, \alpha) = \frac{2c\cos\alpha}{\pi a \sin\alpha}\sqrt{1 - \frac{c^2(t - 2z/c)^2}{a^2 \sin^2\alpha}}$$



- Models the response, h_R, with

  c = speed of sound

  a = diameter of sonar element

  t = time

  z = orthogonal distance

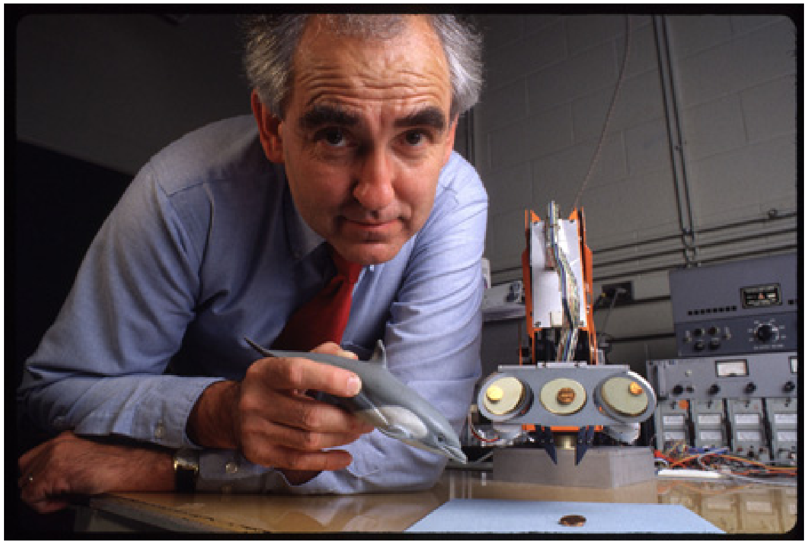
  α = angle of environment surface

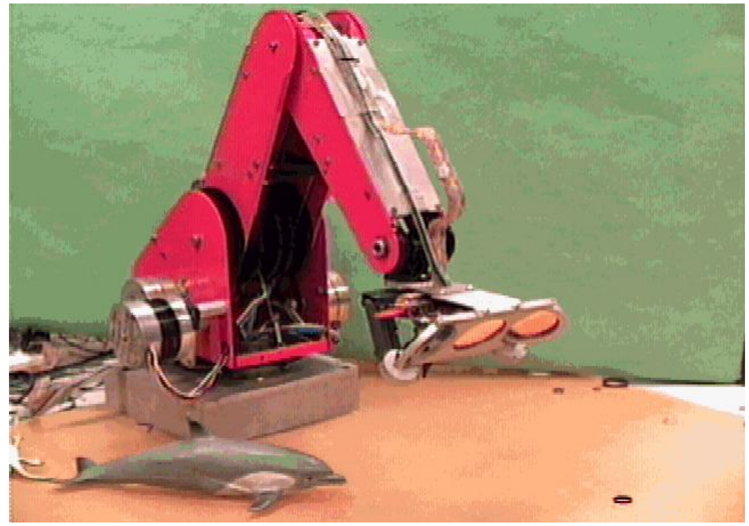
- Then, allow uncertainty in the model to obtain a probability:

$$\boxed{p(S \mid o)}$$

chance that the sonar reading is S, given an obstacle at location o

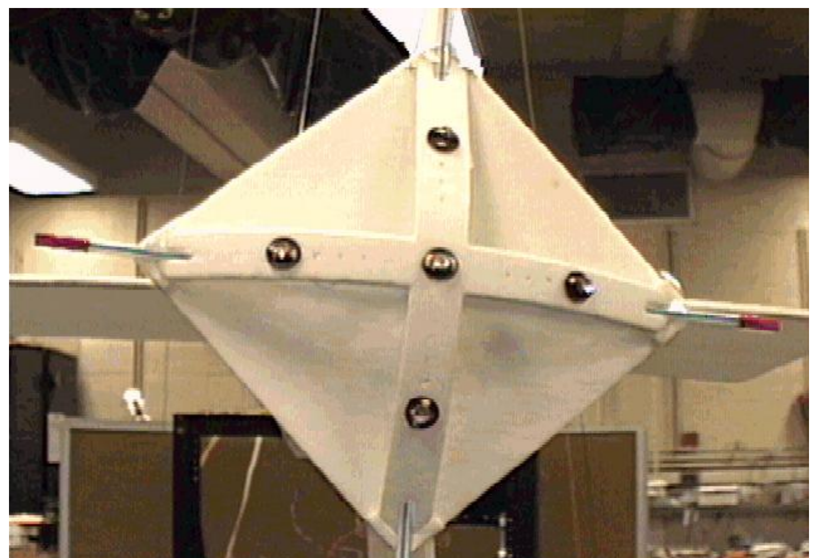
Roman Kuc's animals



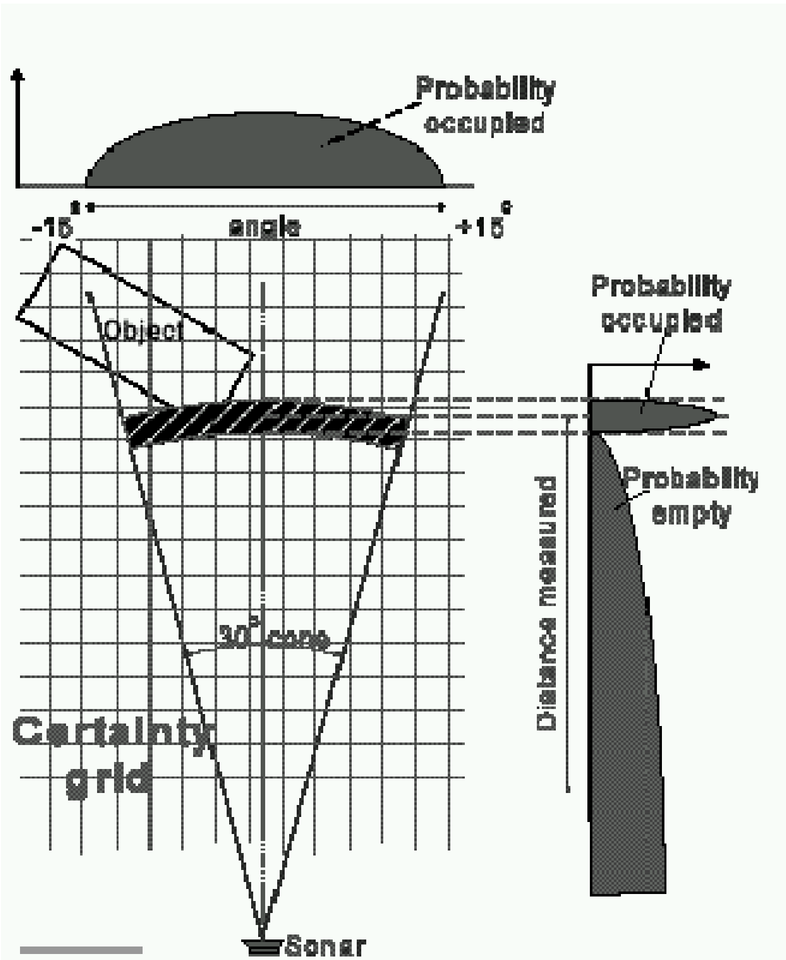


Rodolph

Robat



Certainty Grid Approach

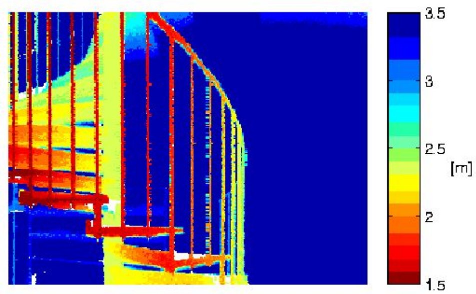


Combine info with Bayes Rule (Morevac and Elfes)

Laser Ranging

LIDAR

Sick Laser



LIDAR map

Sick laser
range finder