Probabilistic Mapping

Robot Challenge 2011
Happylab

Simon Coupland
simonc@dmu.ac.uk

Overview

- Motivation
- Sonar Modelling
- Occupancy Grids
- Applications

Motivation

- Measurements are never accurate
  - System measurements:
    - Surfaces change
    - Motors stall
  - Environment measurements
    - Wind, humidity and temperature
    - Material properties

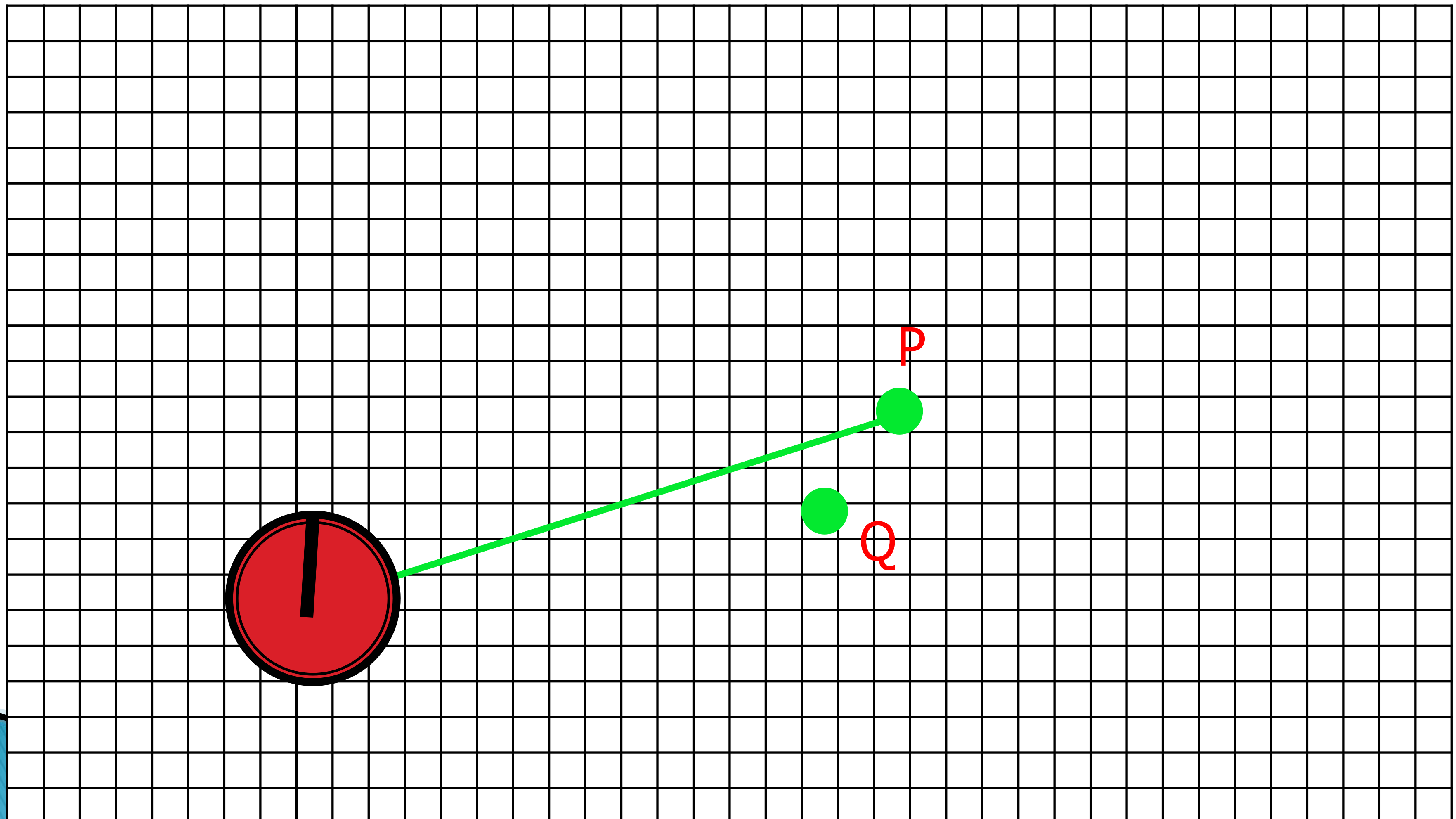
Solution

- Form probabilistic models of:
  - Sensor
  - Environment

Sonar Model

- Why do we need a sonar model?
  - Readings are inaccurate
  - A probabilistic model of the sonar allows us to model the inaccuracies
  - Say we get a distance reading of 2.23 metres at a heading of 70°
  - We can the ask questions like:
    - Given the reading, what is the probability that there is an obstacle at (21.2,42.1)?
- General question:
  - Given the potential for uncertainty in the sonar reading at P what do I know about Q?

Sonar Model



Sonar Model

- We need a model of a sonar sensor
- But what should its properties be?

Sonar Model

- We need a model of a sonar sensor
- But what should its properties be?

- The area immediately around the sonar reading should mark the obstacle taking into account:
  - Uncertainty in the distance reading
  - Uncertainty in angle
- Grid squares in this area have a probability of 1
- This is region 1

Sonar Model

- We need a model of a sonar sensor
- But what should its properties be?

- The area in between the sonar and obstacle should be free of obstacles
- Grid squares in this area have a probability of 0
- This is region 2

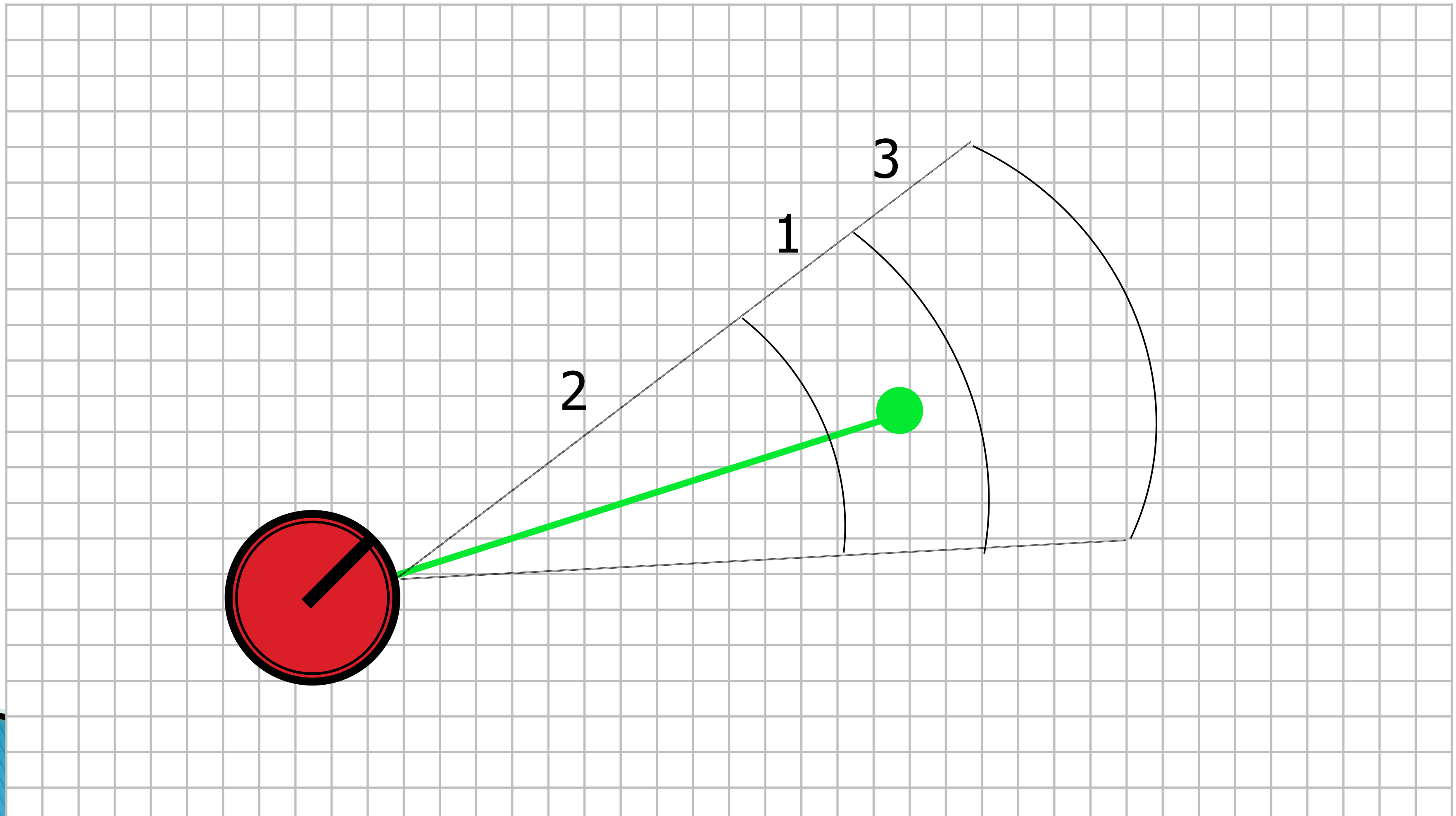
Sonar Model

- We need a model of a sonar sensor
- But what should its properties be?

- The area in on the other side of obstacle we have no knowledge of
- Grid squares in this area have a probability of 0.5
- This is region 3

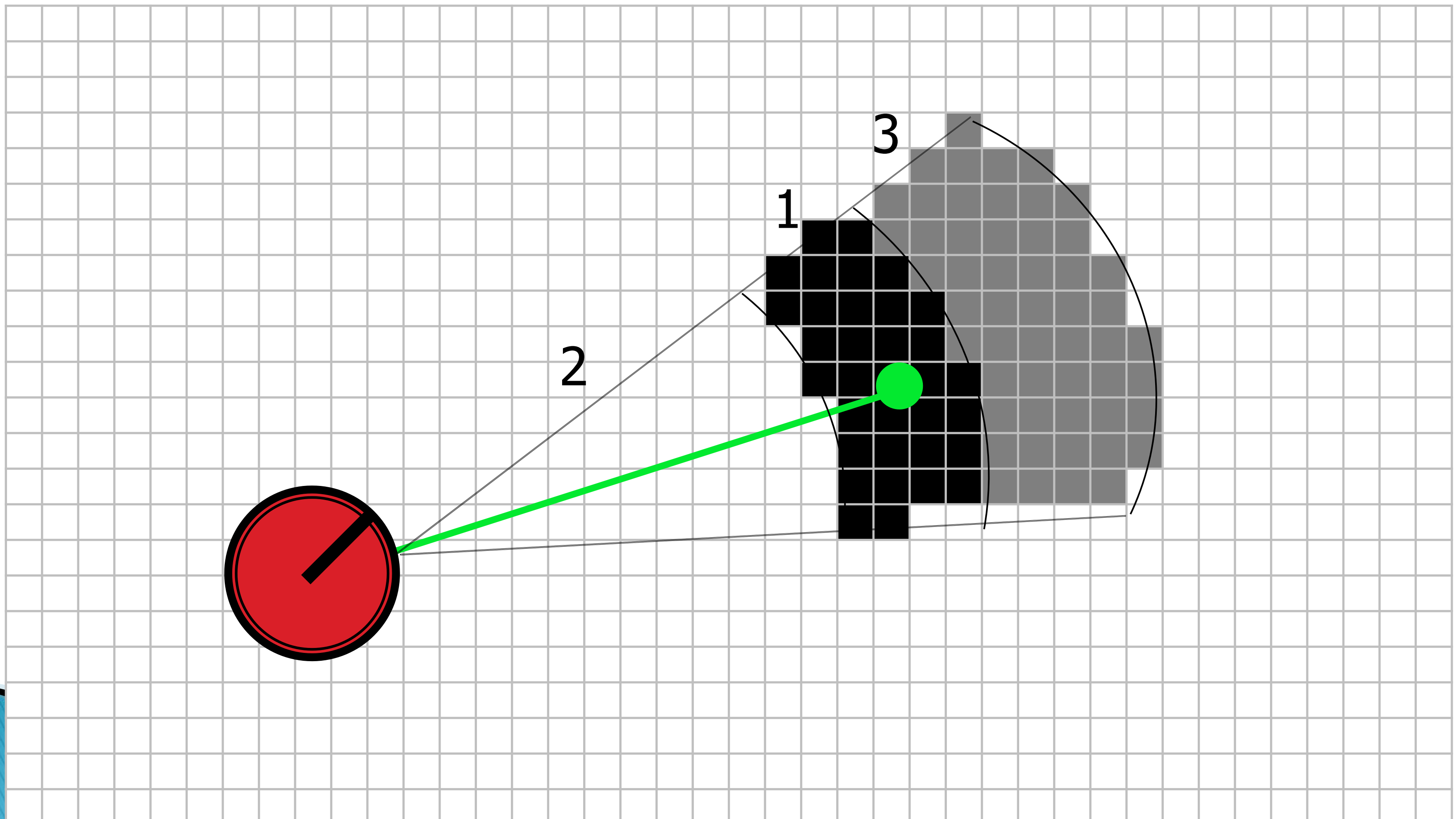
Sonar Model

- Simplest model



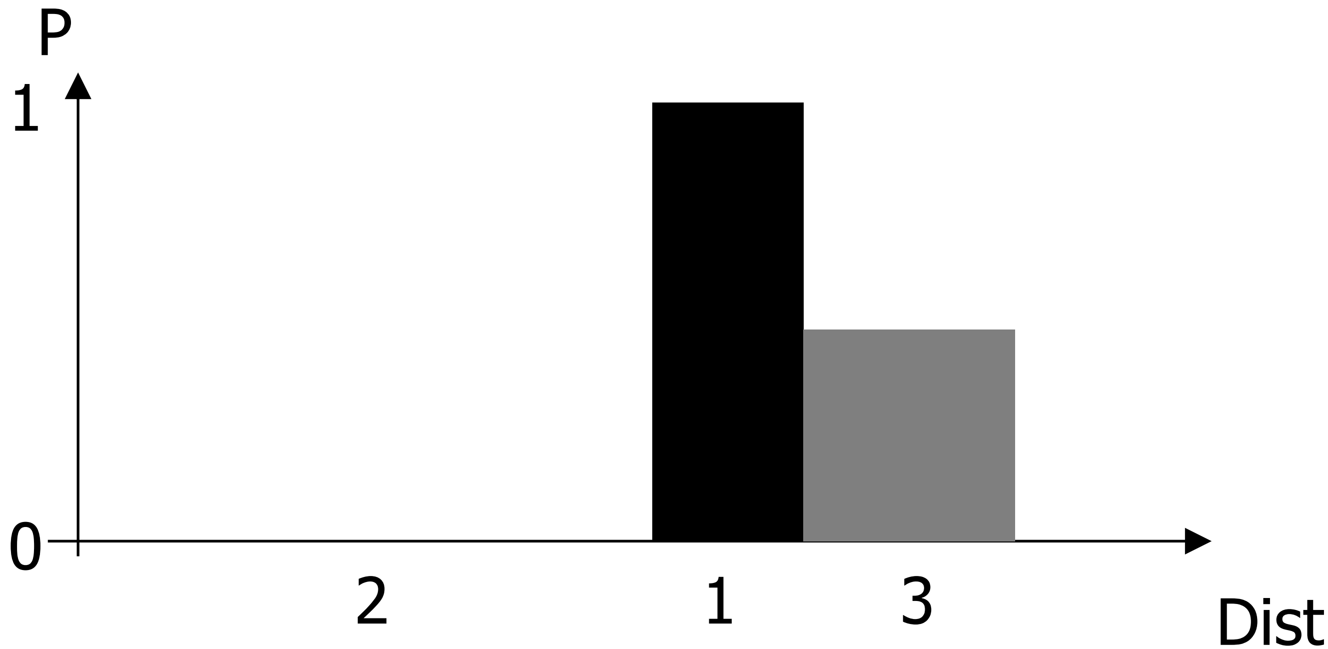
Sonar Model

- Simplest model



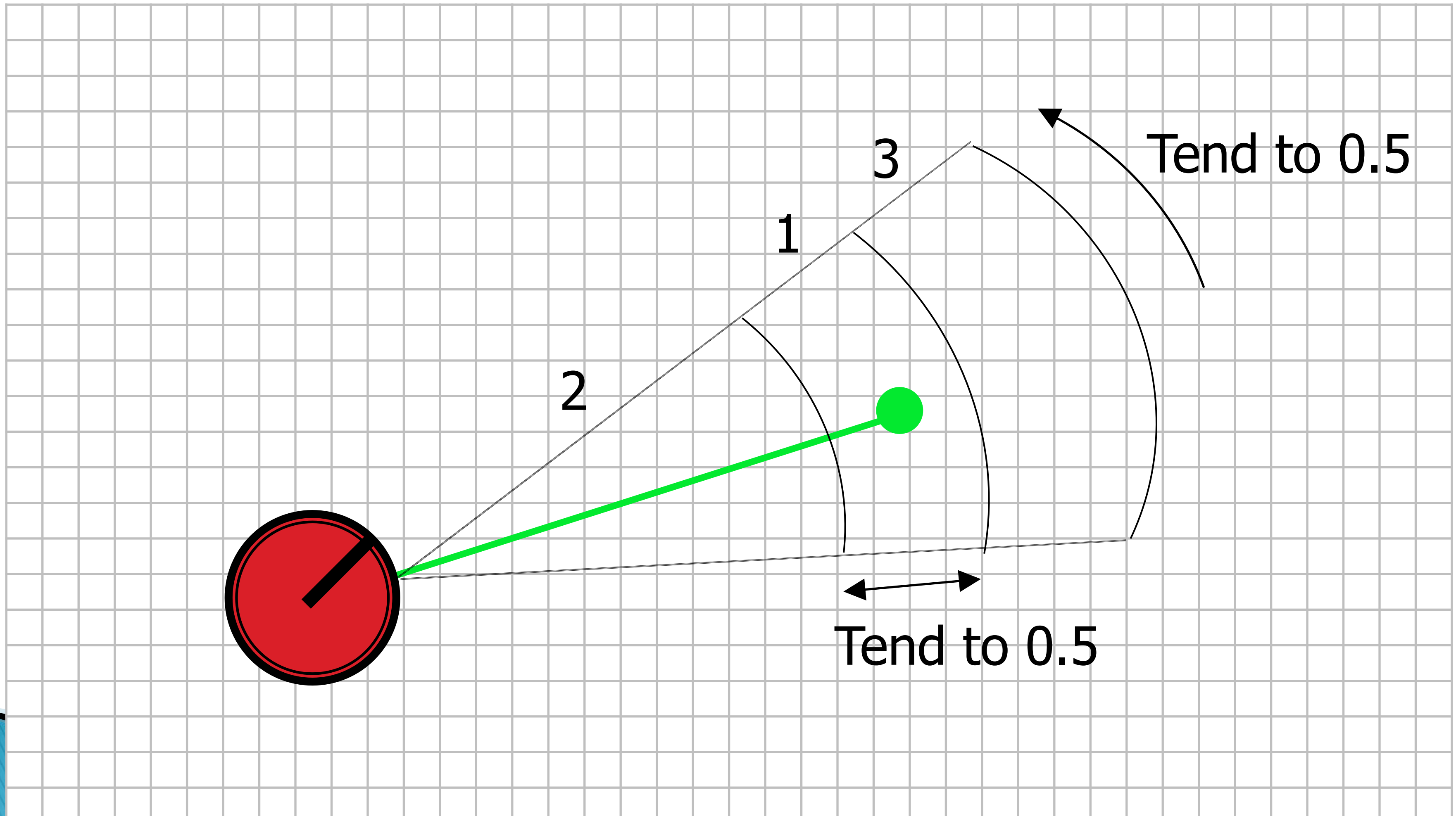
Sonar Model

- Simplest model



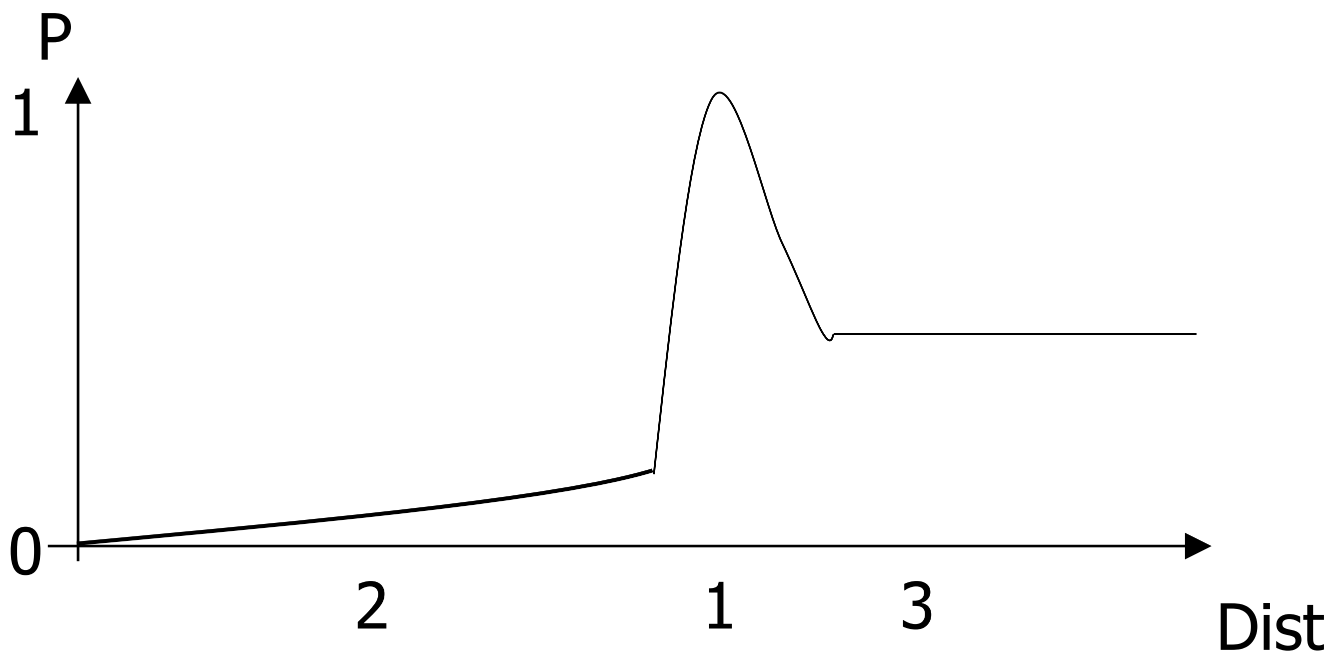
Sonar Model

- Improving this model



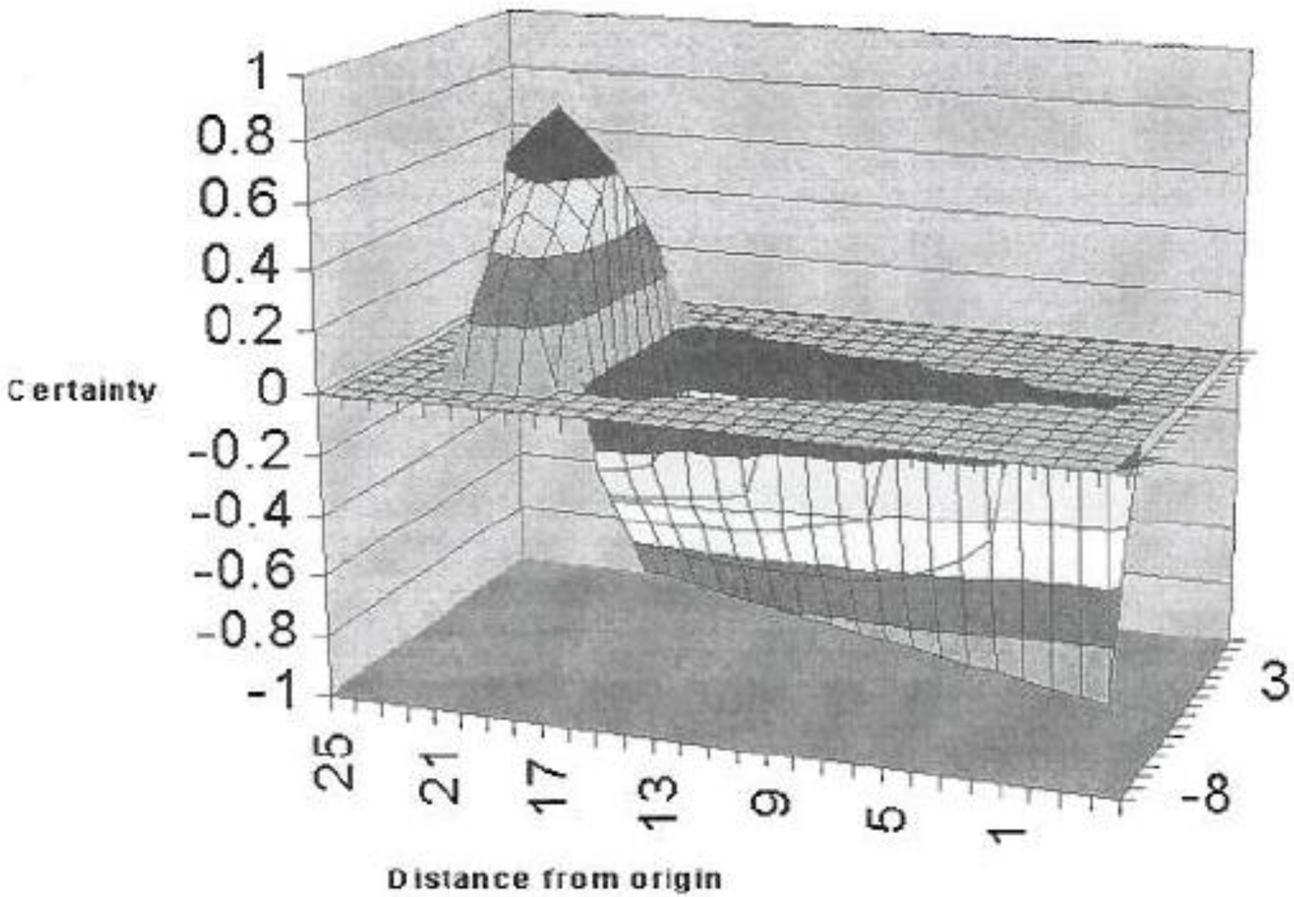
Sonar Model

- More complex model



Sonar Model

- Example from Murphy page 379

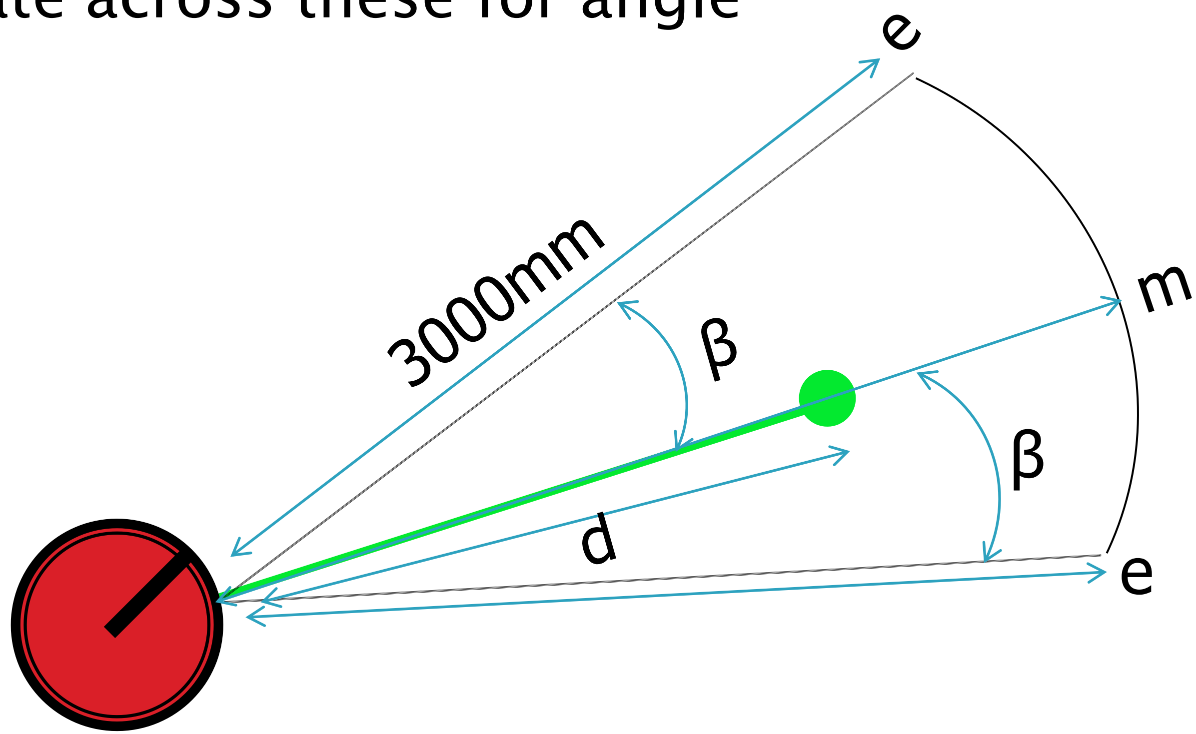


Sonar Model

- How to get create a model like this?
  - Use parametric functions
    - Bell curve
    - Triangle
    - Trapezoid
  - Parameters
    - Angle
    - Distance
  - Combine these functions and plot in polar space

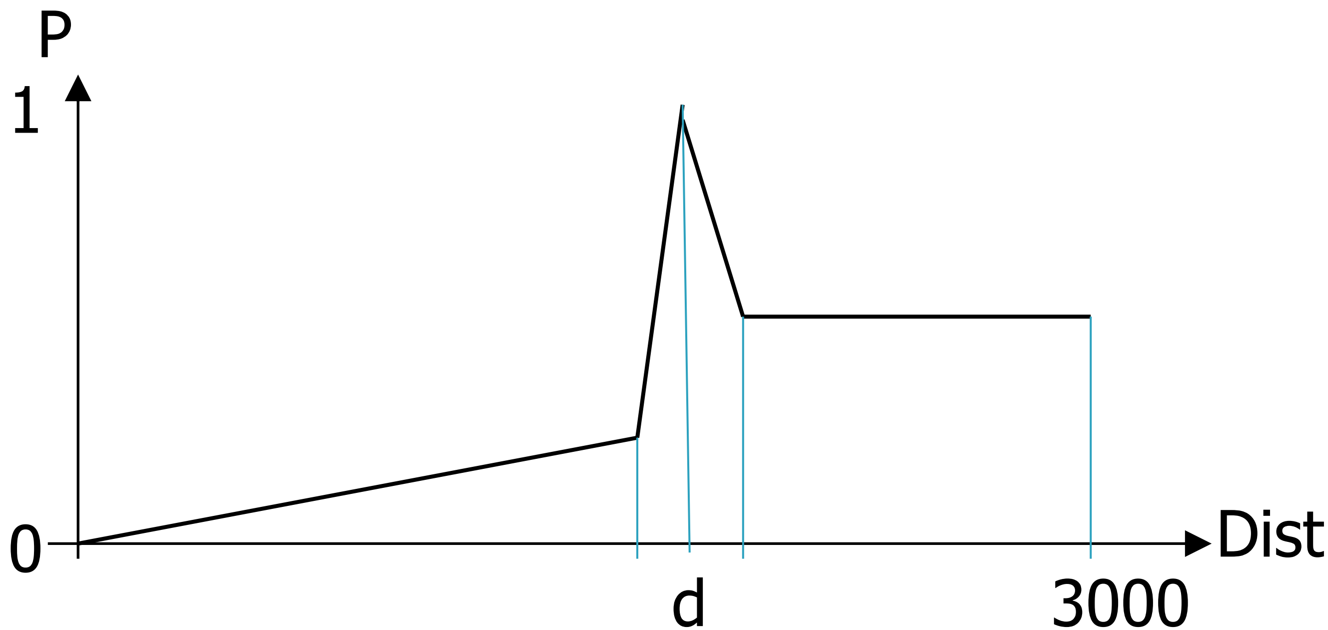
Sonar Model

- Example using linear interpolation
  - Use two linear functions of distance m and e
  - Interpolate across these for angle



Sonar Model

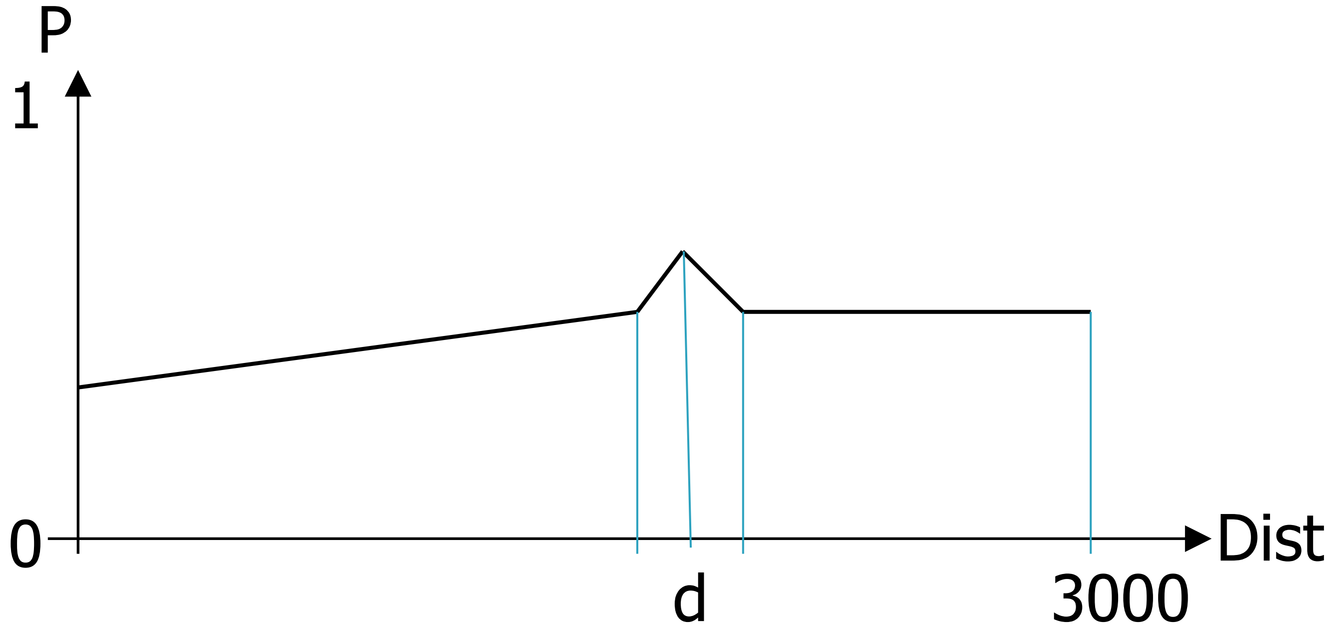
- Function m



- Points = {(0,0),(max(0,d−150),0.25), (d,1.0), (min(d+150,3000),0.5), (3000,0.5)}

Sonar Model

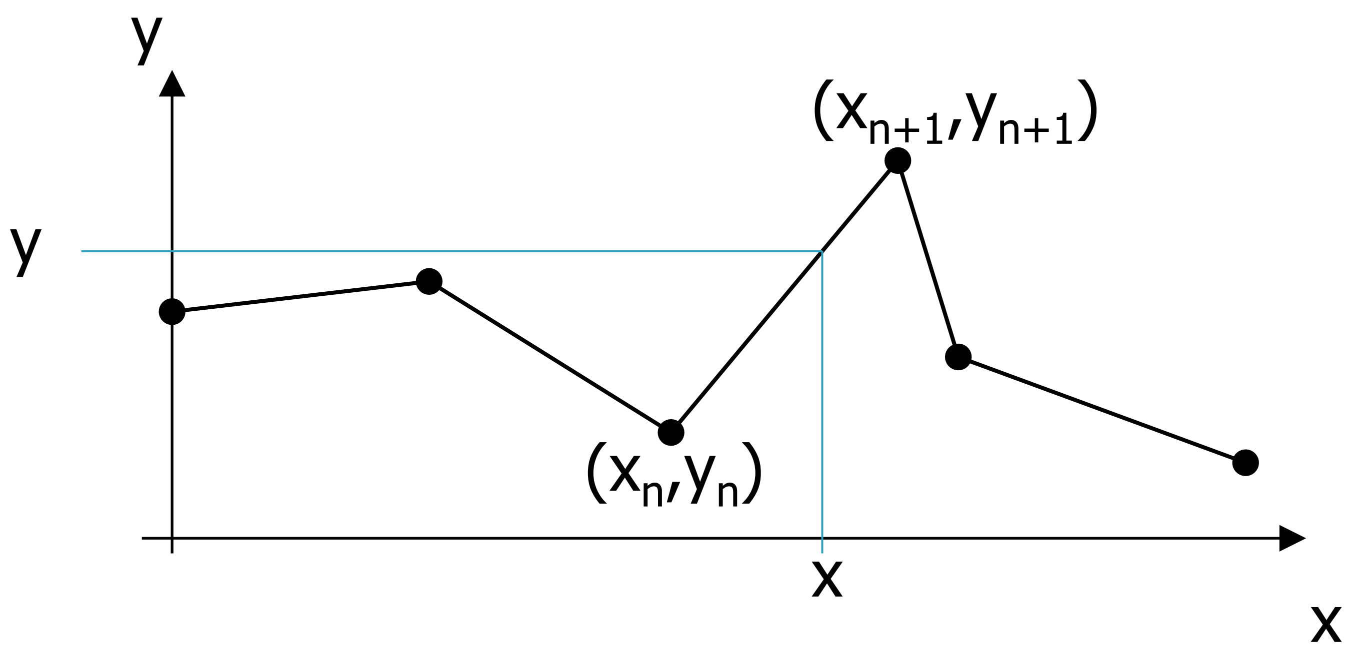
- Function e



- Points = {(0,0.4),(max(0,d−150),0.5), (d,0.6), (min(d+150,3000),0.5), (3000,0.5)}

Sonar Model

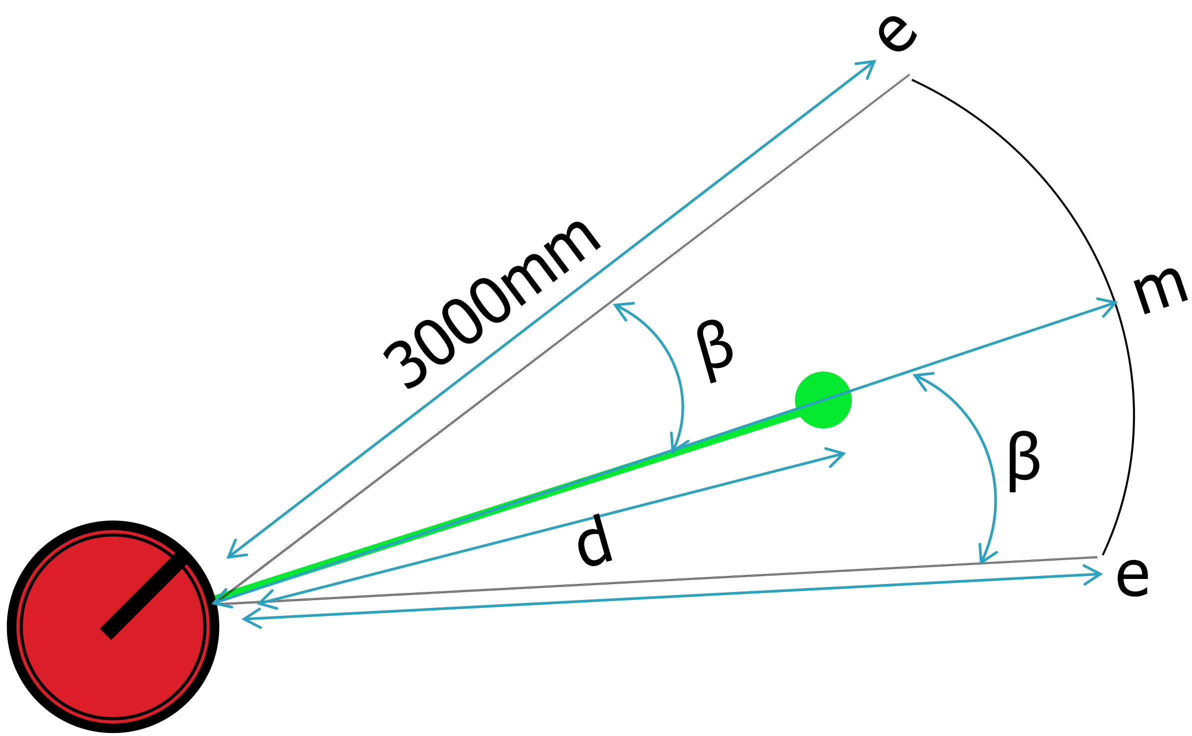
- Linear interpolation



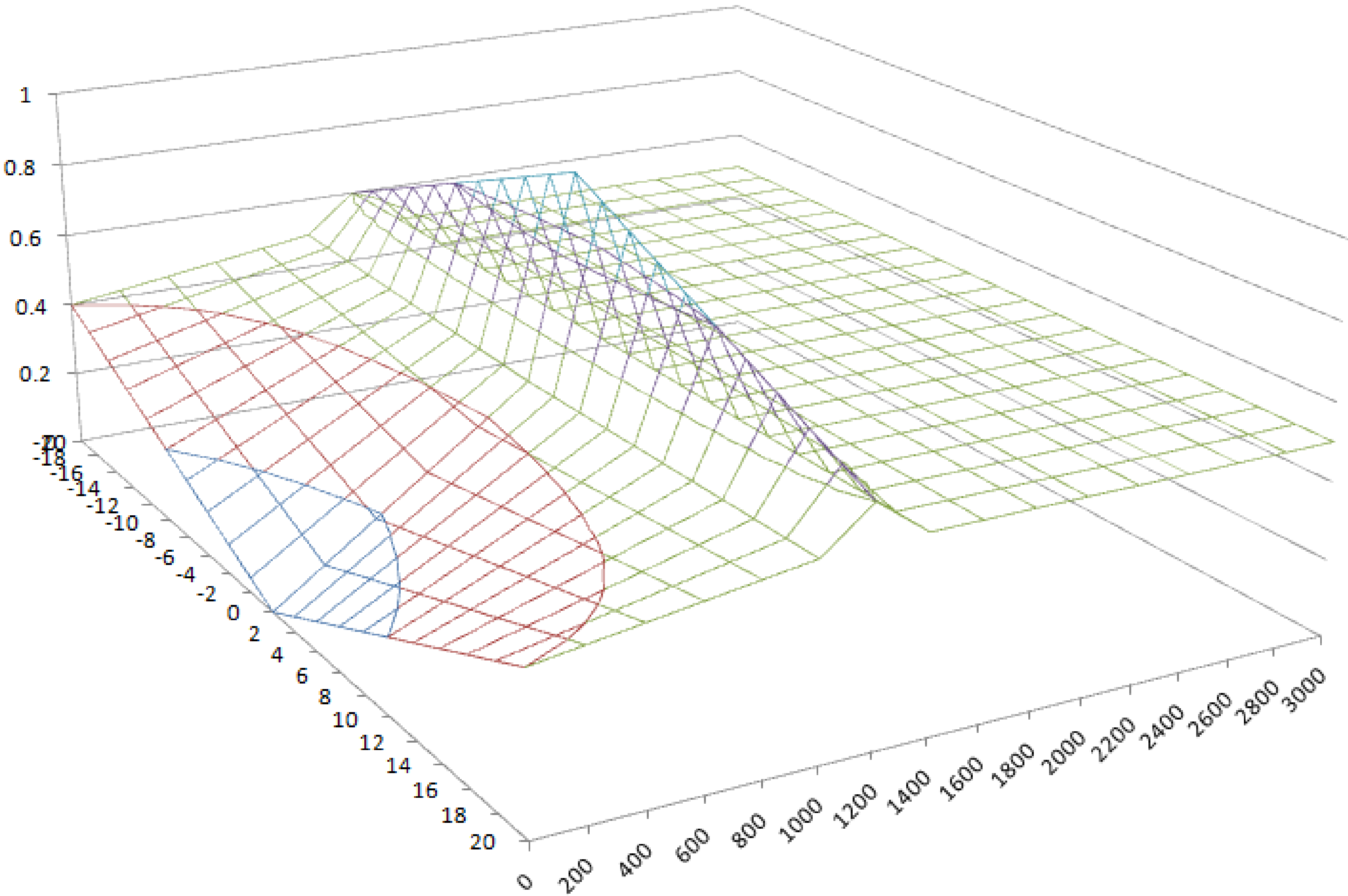
$$y = y_n + (x - x_n) \times \frac{y_{n-1} - y_n}{x_{n-1} - x_n}$$

Sonar Model

- Recall

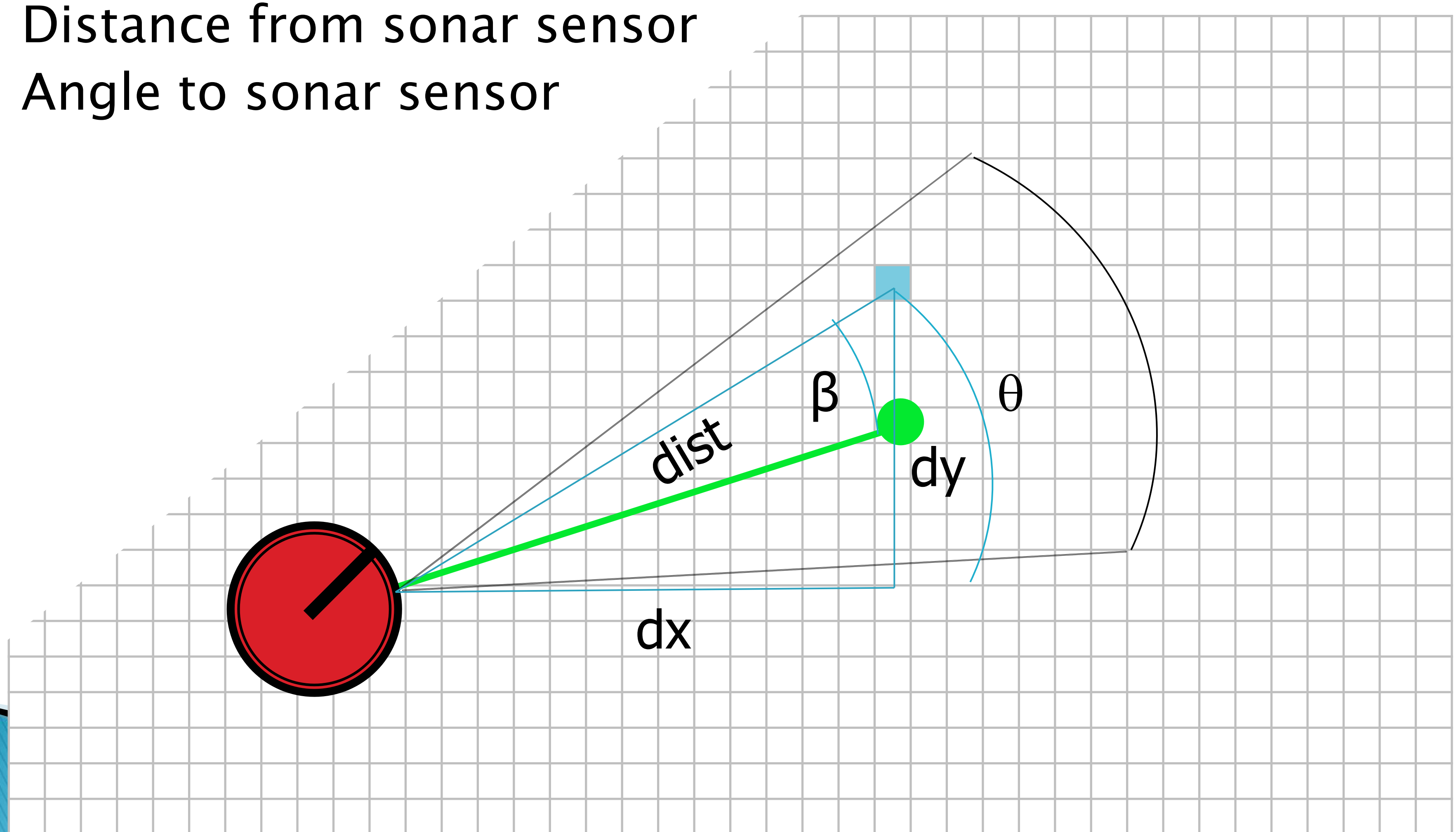


In Cartesian Co-ord Space



Polar Space

- For each grid square we calculate:
  - Distance from sonar sensor
  - Angle to sonar sensor



Polar Space

- Solutions:

$$dist = \sqrt{dx^2 + dy^2}$$

$$\theta = arctan2(dy, dx)$$

$$\beta = sonar\theta - \theta$$

Sonar Model

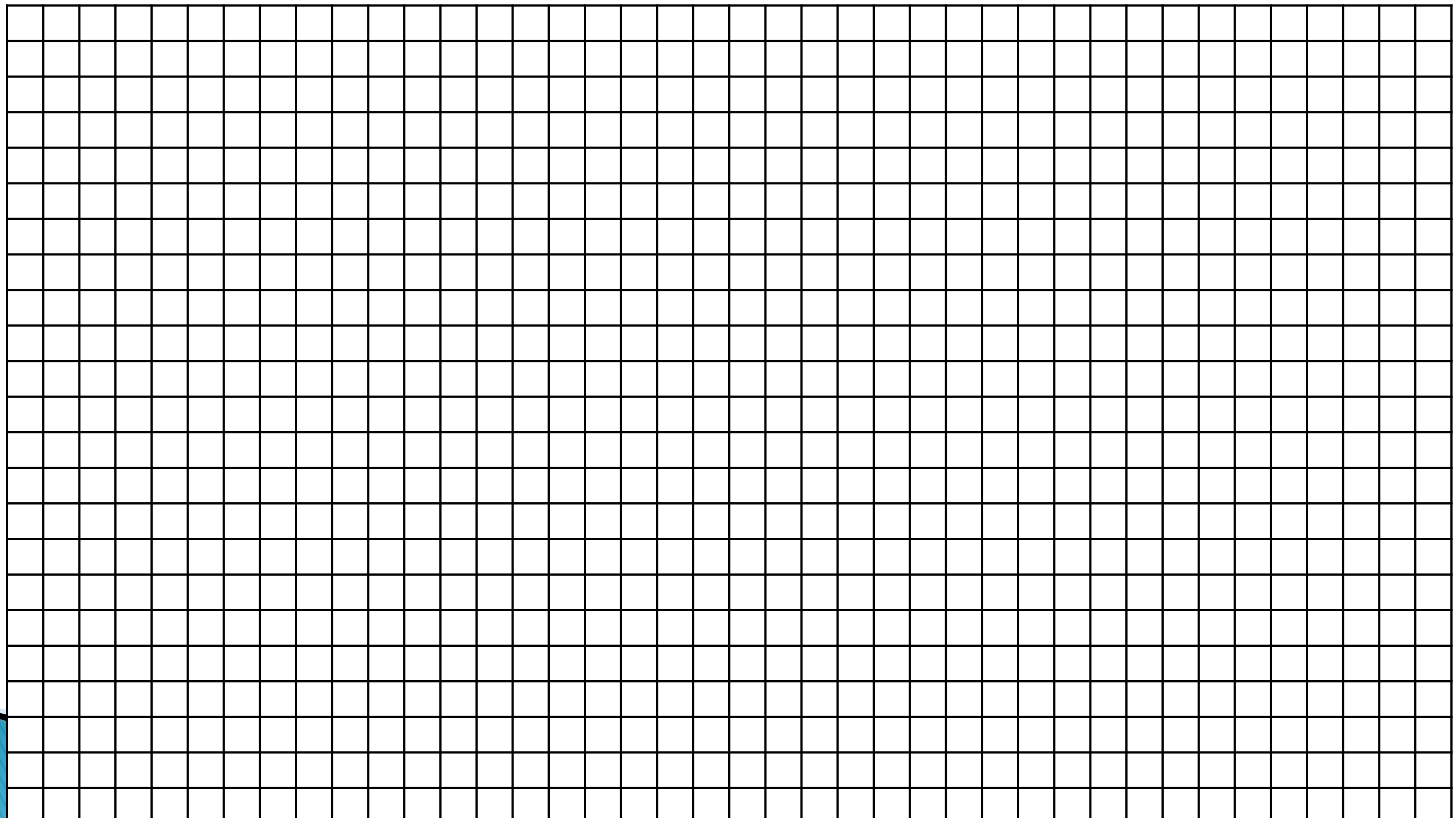
- Demo

The Occupancy Grid

- Current map model is binary:
  - A point in space is either occupied or not
  - True whether scattergram or a line based map
- Range readings are binary
- However there are uncertainties:
  - Wind
  - Humidity
  - Obstacle material properties
- Occupancy grids attempt to mitigate some of these

The Occupancy Grid

- Divide mapping area into a grid



The Occupancy Grid

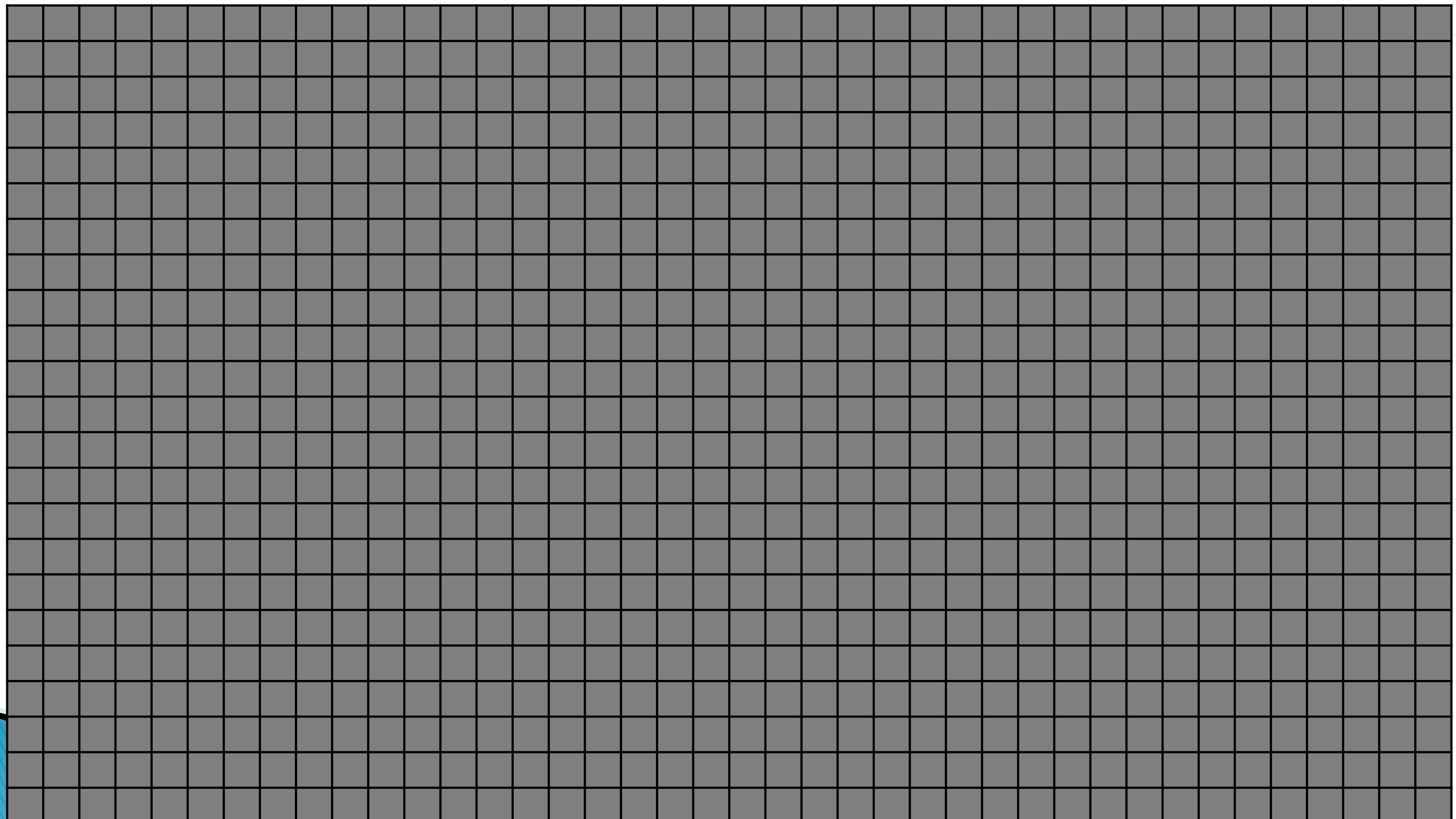
- Grid size is proportional to computational complexity:
  - How big is the area you are working in?
    - 5m X 5m
  - Let a grid square cover 5cm X 5cm area
    - 100 x 100 squares
    - 10000 squares

The Occupancy Grid

- So each grid square represents an area of some size
- Each square is assigned a probability
- This is the probability that the square is occupied
- 0 = definitely not occupied (white)
- 1 = definitely occupied (black)
- 0.5 = Equal chance of being occupied or unoccupied (grey)
- All squares are initialised to 0.5

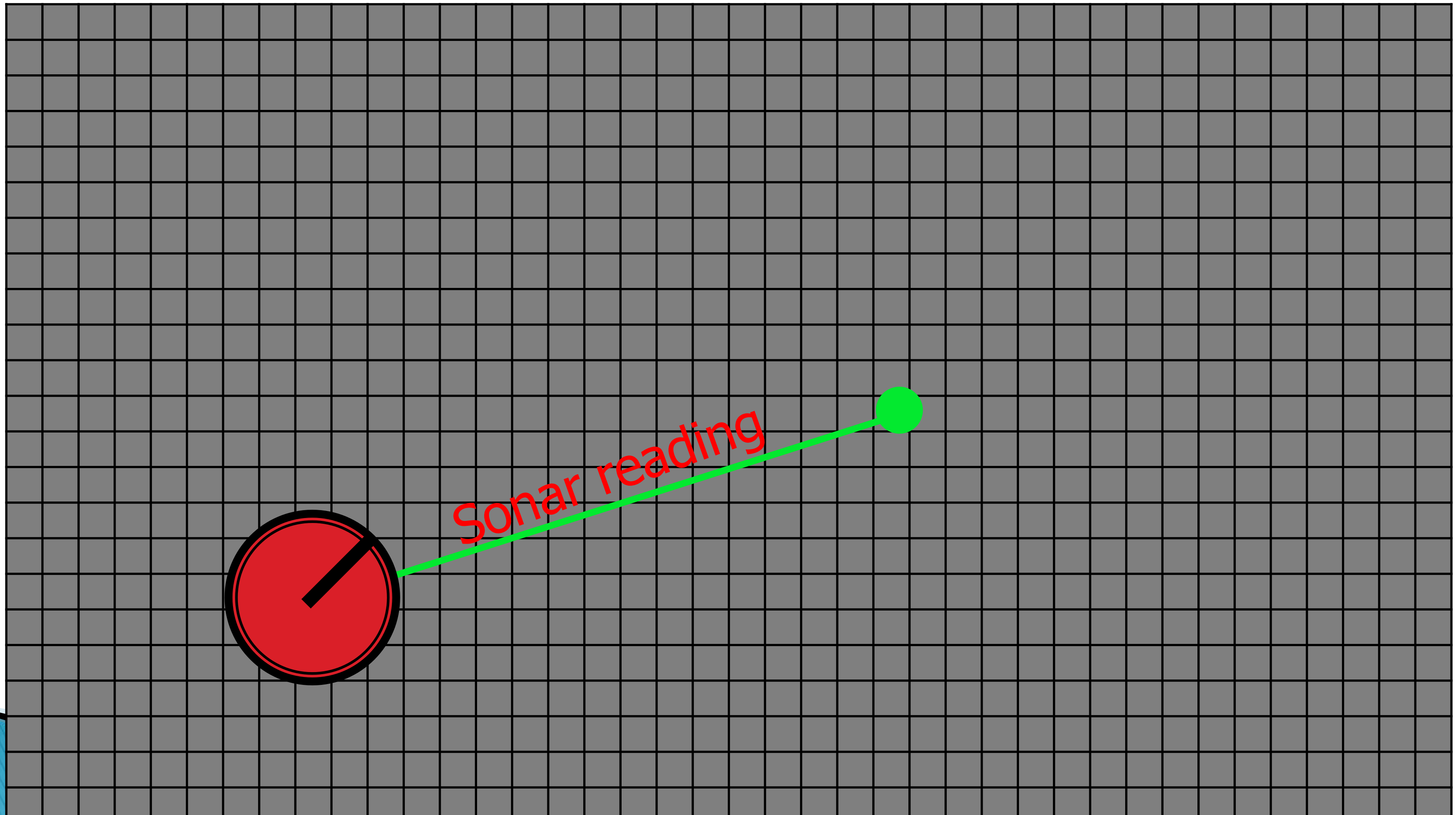
The Occupancy Grid

- Initial occupancy grid



The Occupancy Grid

- But how to do we update it?



The Occupancy Grid

- Apply our sonar model based on:
  - Current odometry position
  - Sonar readings
- Application uses Bayes' theorem

Bayes' Theorem

- **Conditional probability:**
- H is a hypothesis (something we wish to test the truth of), E is the available evidence, then:

$$p(H|E) = \frac{p(E|H) \times p(H)}{p(E)}$$

- p(E | H) is the likelihood of the data, given the hypothesis
- p(H) is the prior probability of the hypothesis.
- p(E) is the prior probability of the evidence (used to normalise the probabilities, 1–p(H))
- p(H | E) is the posterior probability of the hypothesis – the probability that H is true given the evidence E.

Bayes' Theorem

- **How does this relate to mapping?**
- The hypothesis is that a given grid square is occupied
- The occupancy grid holds the probability that a grid square is occupied – p(H) – prior probability
- p(E) is the second prior probability, the probability that the grid square is empty given by 1.0 – p(H)
- p(E | H) is the likelihood of the data, given the hypothesis given by the sonar model.
- p(H | E) is the posterior probability of the hypothesis – the probability that H (there is an obstacle) is true given the evidence E (new sonar reading).

Bayes' Theorem

- So we can now update the occupancy grid when we get a single reading
- If we only use this method we overwrite our hard gained evidence
- We need to use this previous evidence and **update** rather than **overwrite**

Recursive Bayes' Theorem

- Using the recursive form of Bayes' we get:

$$p(H|E_t) = \frac{p(E_t|H) \times p(H|E_{t-1})}{p(E_t|H) \times p(H|E_{t-1}) + p(E_t|\neg H) \times p(\neg H|E_{t-1})}$$

- Terms are the same. t and t-1 refers to current time and previous time

Simple Worked Example

- Consider a grid square centred over 250,140
- Initial value is 0.5

- Sonar reading taken
- Sonar model gives this square p = 0.67

$$p(H|E_1) = \frac{0.67 \times 0.5}{0.67 \times 0.5 + 0.33 \times 0.5}$$

- New value = 0.67

Simple Worked Example

- Grid square centres over 250,140
- Value is 0.67

- Sonar reading taken
- Sonar model gives this square p = 0.71

$$p(H|E_2) = \frac{0.71 \times 0.67}{0.71 \times 0.67 + 0.29 \times 0.33}$$

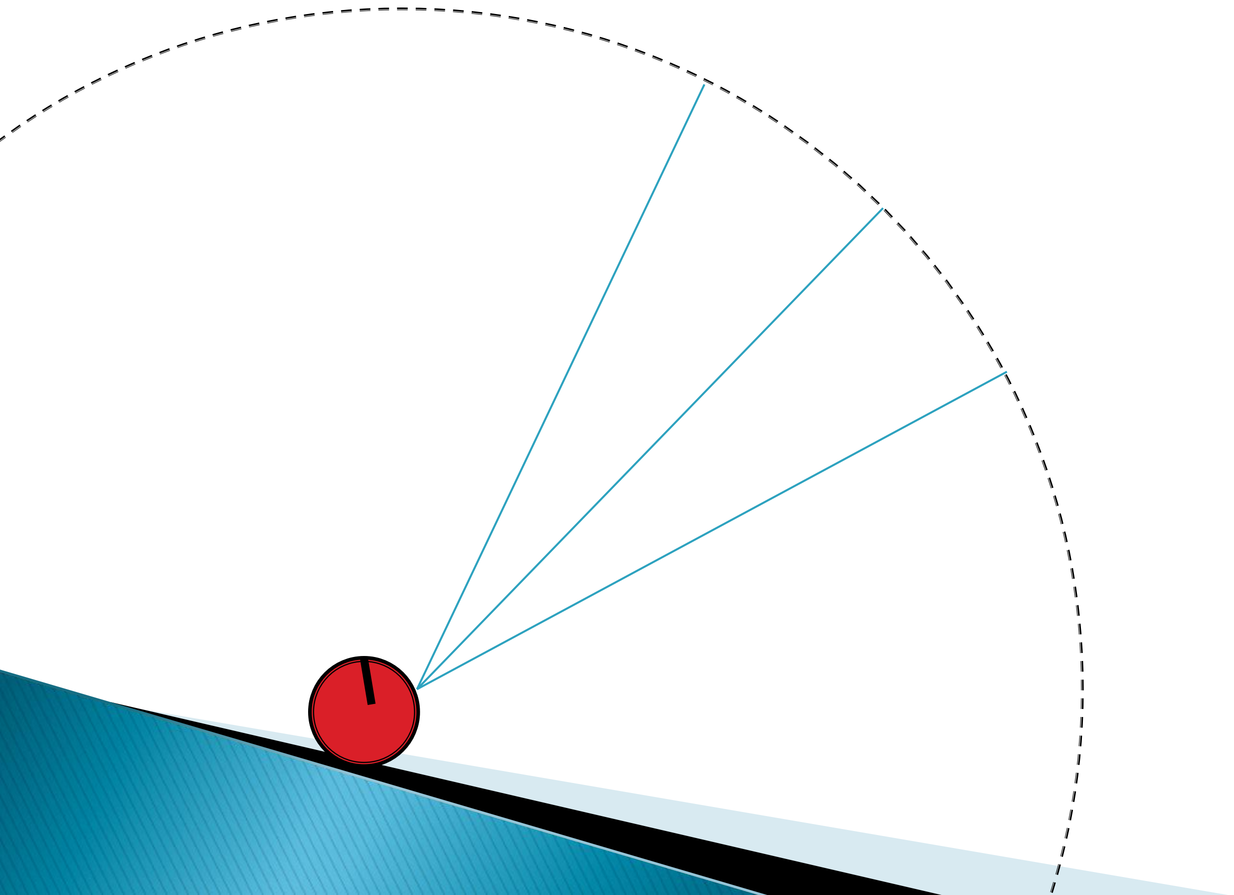
- New value = 0.83

Simple Worked Example

- This calculation needs to be performed for every square inside the sonar ‘cone’

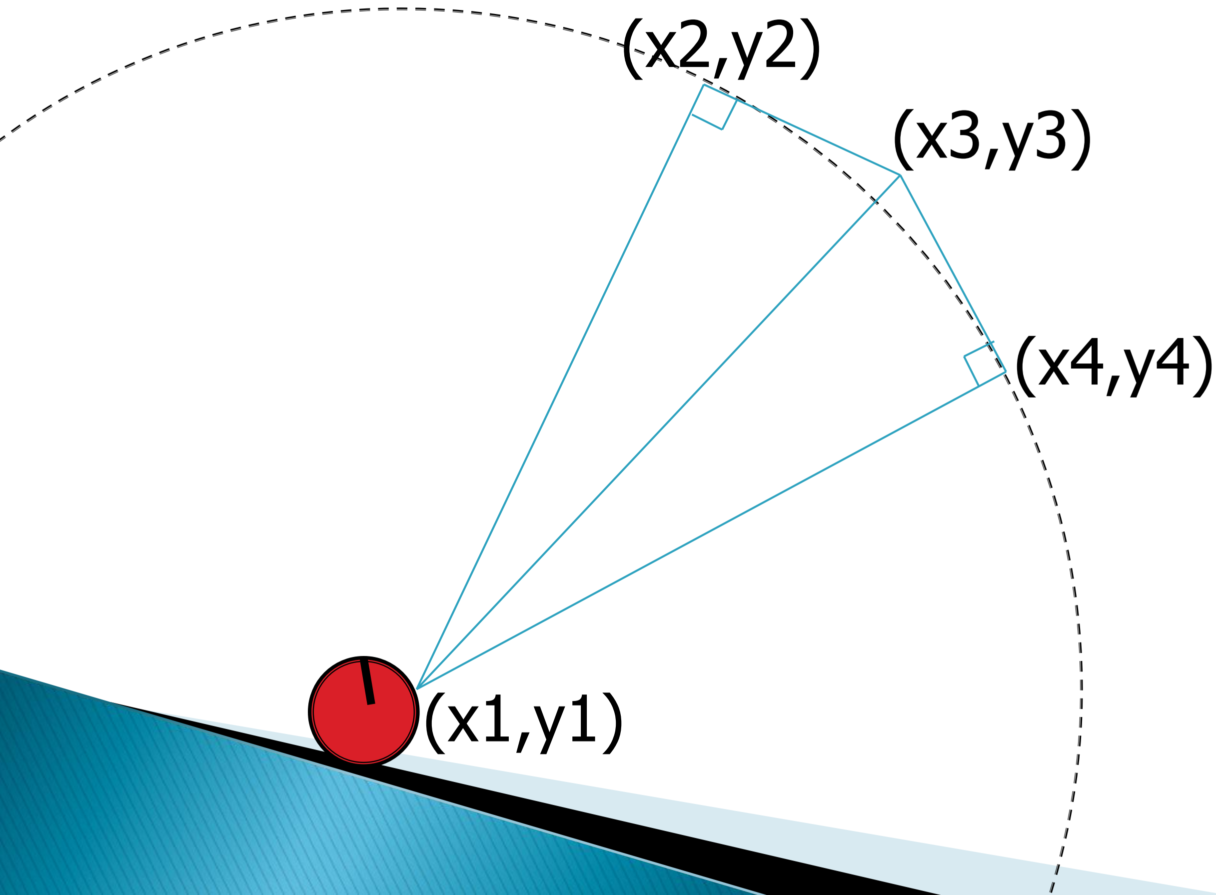
Implementation Details

- Sonar ‘cone’ bounding box
- One quick approach:



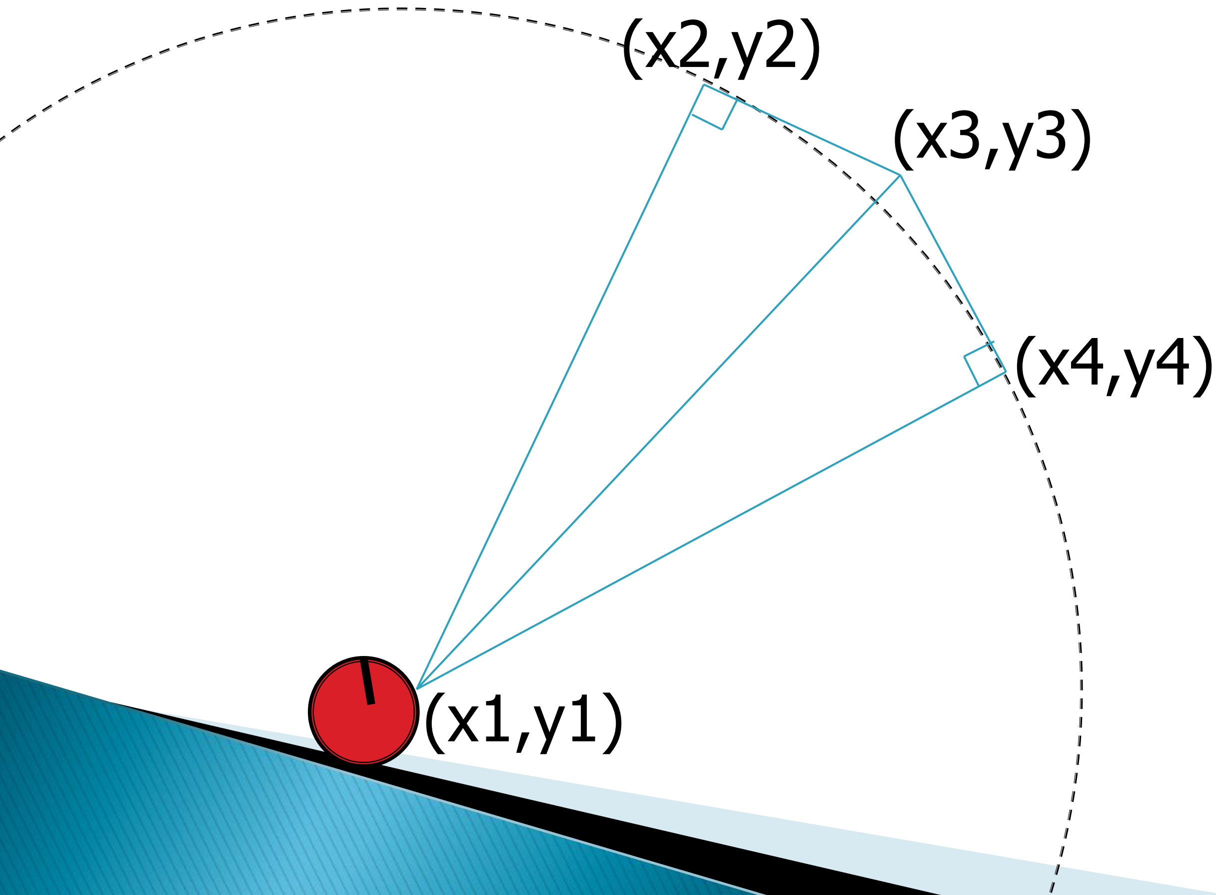
Implementation Details

- Sonar ‘cone’ bounding box
- One quick approach:



Implementation Details

- Sonar 'cone' bounding box
- One quick approach:

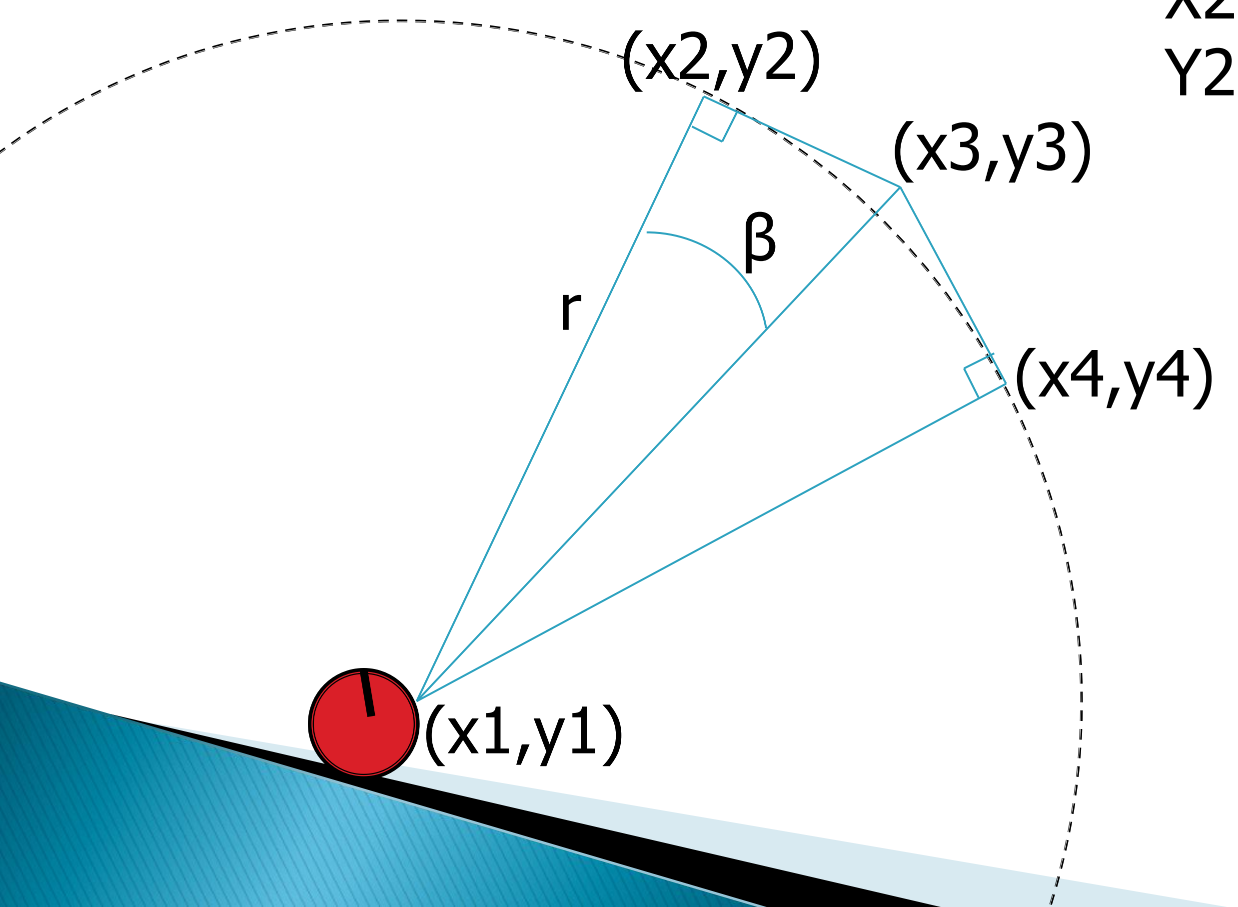


X1 = robotX + sonar offset
Y1 = roboyY + sonar offset

Implementation Details

- Sonar ‘cone’ bounding box
- One quick approach:

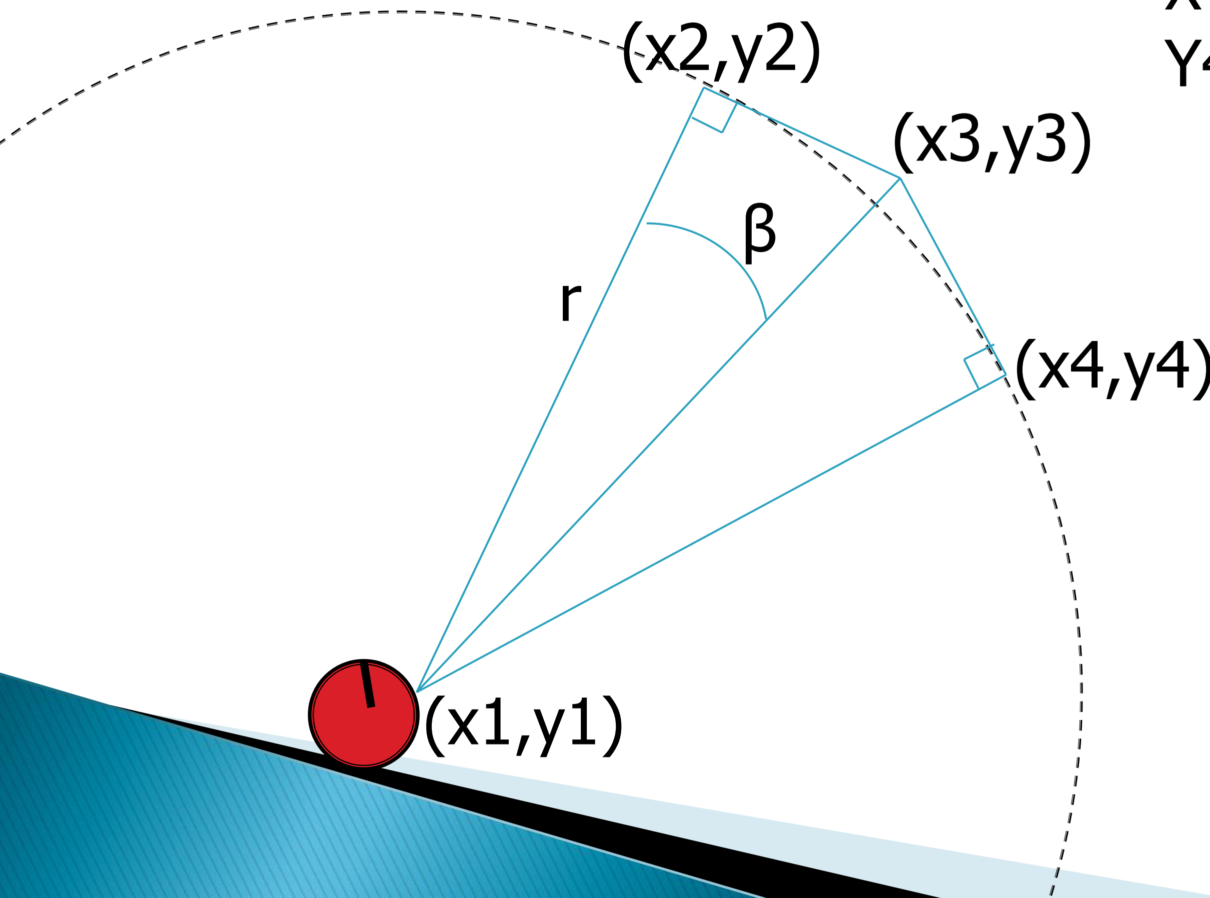
X2 = r X cos(sonarTh – β) + x1
Y2 = r X sin(sonarTh – β) + y1



Implementation Details

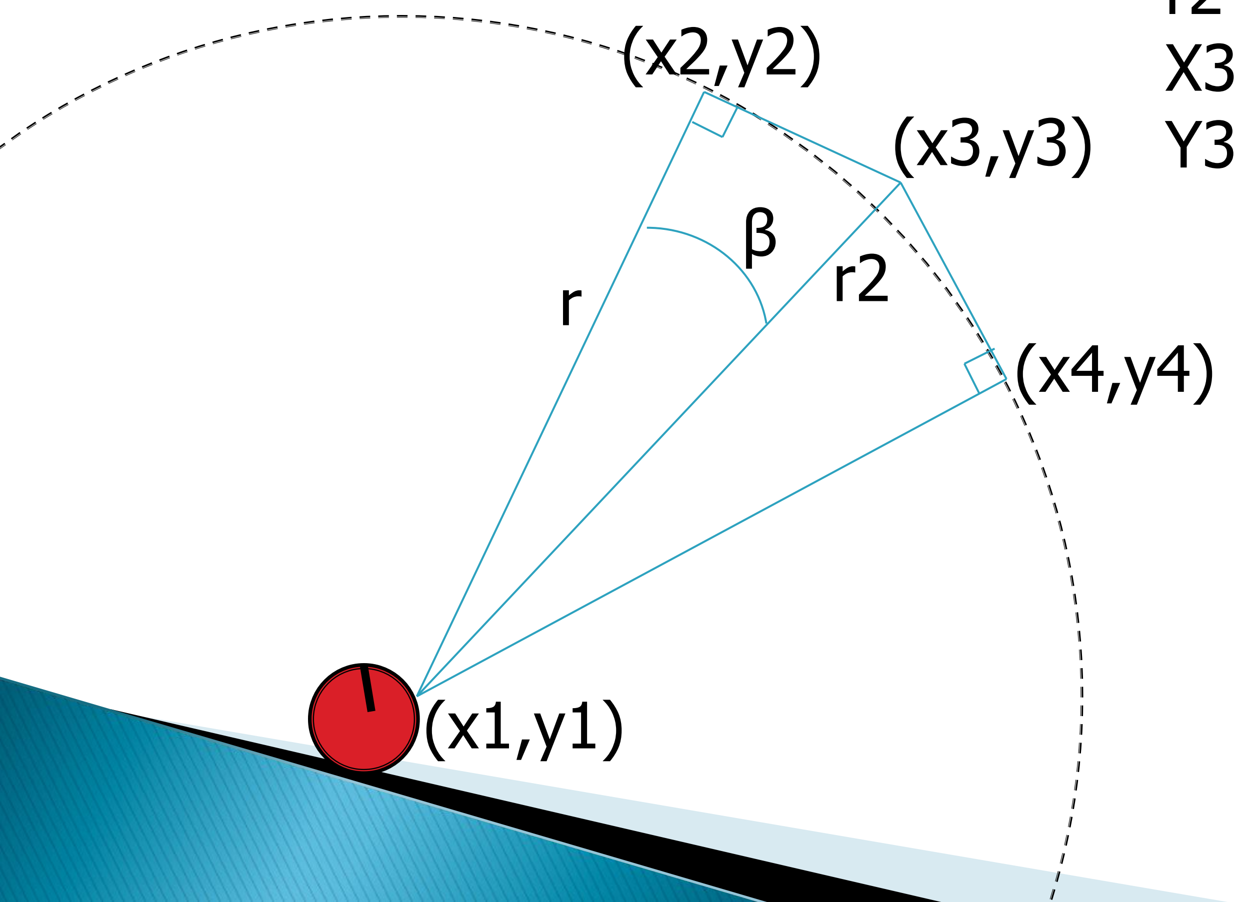
- Sonar 'cone' bounding box
- One quick approach:

X4 = r X cos(sonarTh + β) + x1
Y4 = r X sin(sonarTh + β) + y1



Implementation Details

- Sonar 'cone' bounding box
- One quick approach:



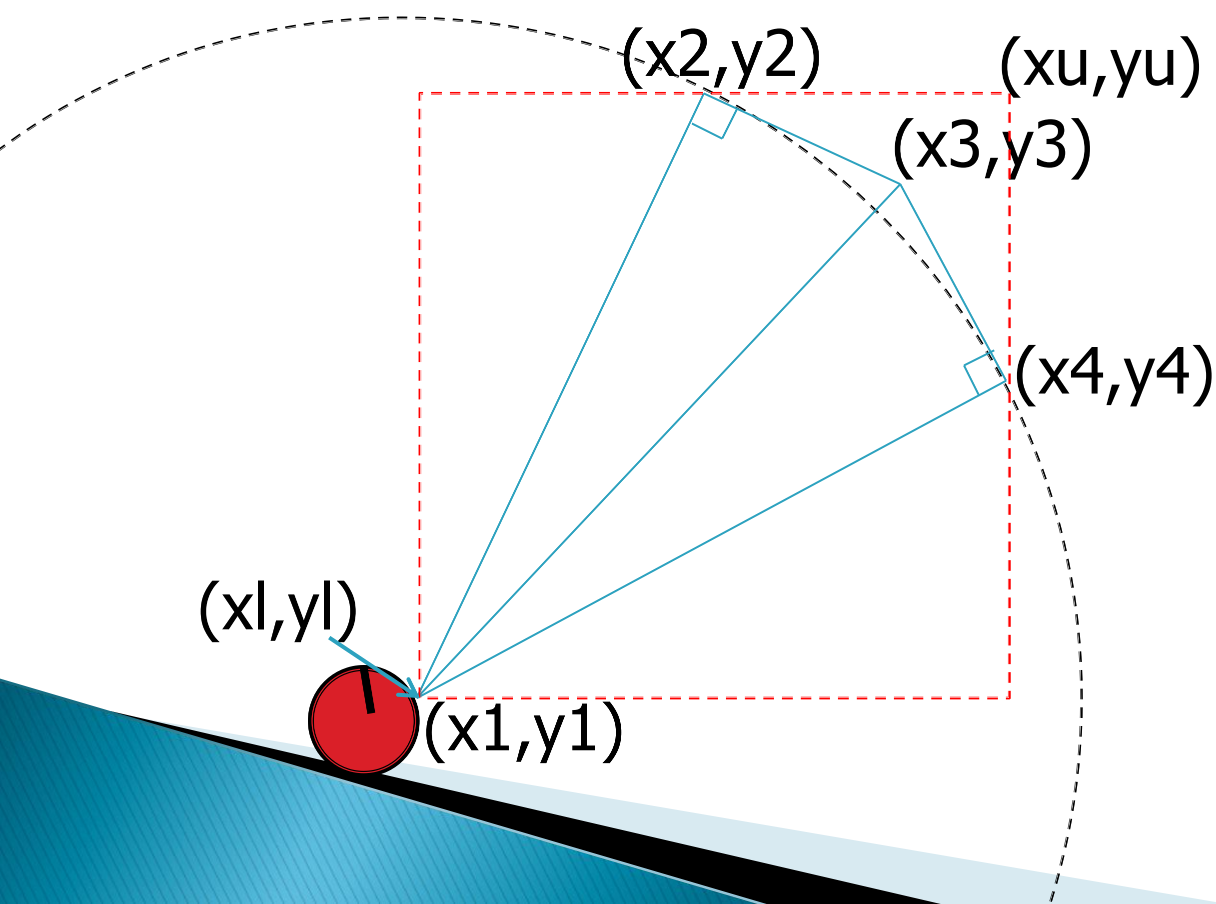
$r2 = r / \cos(\beta)$

$X3 = r2 \times \cos(sonarTh) + x1$

$Y3 = r2 \times \sin(sonarTh) + y1$

Implementation Details

- Sonar 'cone' bounding box
- One quick approach:



xl = min(x1,x2,x3,x4)
yl = min(y1,y2,y3,y4)

xu = max(x1,x2,x3,x4)
yu = max(y1,y2,y3,y4)

Which gives our BB

Can now write a for loop to iterate over the required squares in the grid

Implementation Details

- Sonar ‘cone’ bounding box
- Only iterate over grid squares in the BB
- Only update grid square values when they fall inside the sonar ‘cone’

Implementation Details

- Decision to be made
- We have multiple sensors
- Each with it's own pose
- Each has to be moved to a global pose
- Many options here
  - Option 1
  - Get sonar reading
  - Apply to grid
  - Rotate & translate grid
  - Repeat for all sensors

Implementation Details

- Decision to be made
- We have multiple sensors
- Each with it's own pose
- Each has to be moved to a global pose
- Many options here
  - Option 2
  - Get sonar reading
  - Rotate & translate
  - Apply to global grid
  - Repeat for all sensors

Implementation Details

- Decision to be made
- We have multiple sensors
- Each with it's own pose
- Each has to be moved to a global pose
- Many options here
  - Option 3
  - Get sonar reading
  - Apply to single local grid
  - Repeat for all sensors
  - Rotate & translate
  - Apply to global grid

Occupancy Grids

- Demo

Applications

- An occupancy grid holds a probabilistic model of the environment built up from a number a sonar readings over time

- We can use this to identify our position using a process called Monte-Carlo Localization

Summary

- Robot perceptions are full of inaccuracies
- Application of probability techniques can mitigate the affect of these on decision making