Probabilistic Robotics

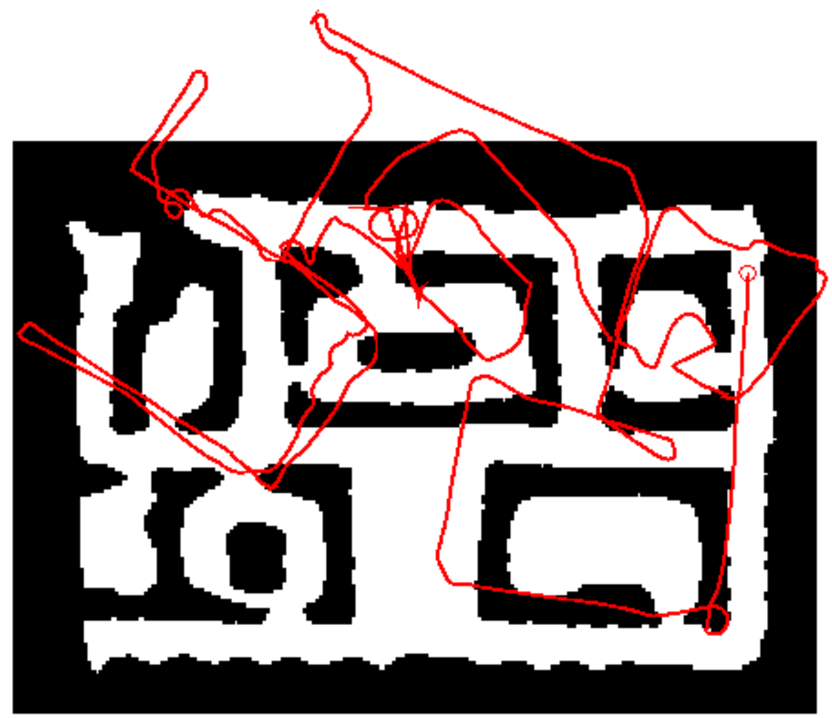
Probabilistic Motion and Sensor Models

Some slides adopted from: Wolfram Burgard, Cyrill Stachniss,

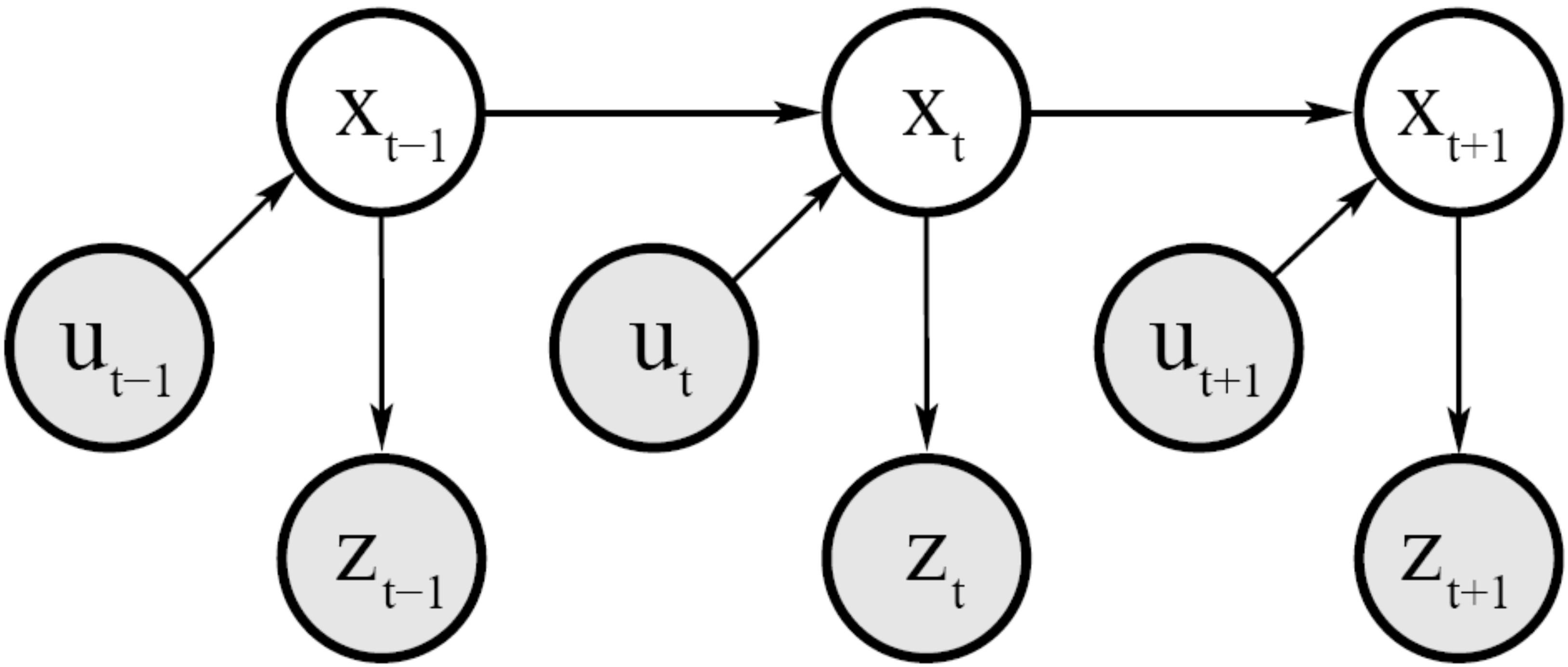
Maren Bennewitz, Kai Arras and Probabilistic Robotics Book

Robot Motion

- Robot motion is inherently uncertain.
- How can we model this uncertainty?



Dynamic Bayesian Network for Controls, States, and Sensations



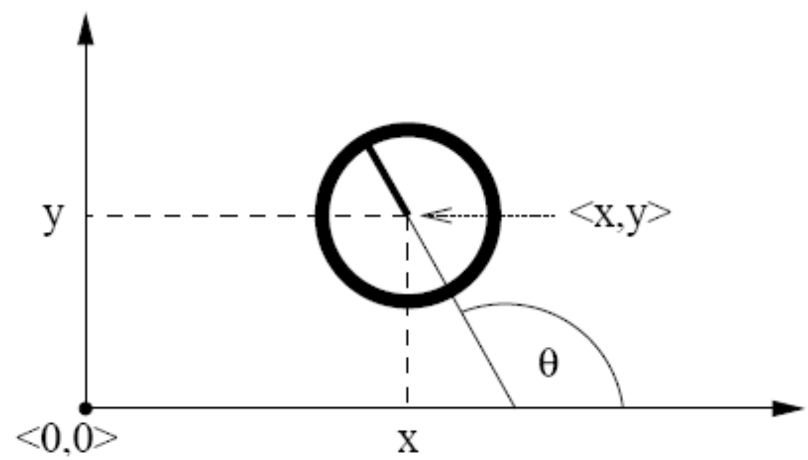
Probabilistic Motion Models

- To implement the Bayes Filter, we need the transition model $p(x \mid x', u)$.
- The term $p(x \mid x', u)$ specifies a posterior probability, that action u carries the robot from x' to x.
- In this section we will specify, how $p(x \mid x', u)$ can be modeled based on the motion equations.

Coordinate Systems

- In general the configuration of a robot can be described by six parameters.
- Three-dimensional Cartesian coordinates plus three Euler angles pitch, roll, and tilt.
- Throughout this section, we consider robots operating on a planar surface.

- The state space of such systems is three-dimensional (x,y,θ).



Typical Motion Models

- In practice, one often finds two types of motion models:
  - **Odometry-based**
  - **Velocity-based** (**dead reckoning**)
- Odometry-based models are used when systems are equipped with wheel encoders.
- Velocity-based models have to be applied when no wheel encoders are given.
- They calculate the new pose based on the velocities and the time elapsed.

Example Wheel Encoders

These modules require +5V and GND to power them, and provide a 0 to 5V output. They provide +5V output when they "see" white, and a 0V output when they "see" black.

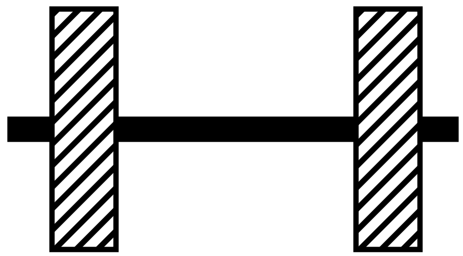
These disks are manufactured out of high quality laminated color plastic to offer a very crisp black to white transition. This enables a wheel encoder sensor to easily see the transitions.

Source: http://www.active-robots.com/

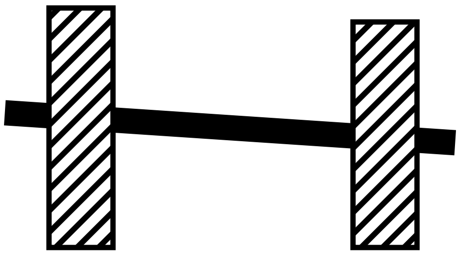
Dead Reckoning

- Derived from “deduced reckoning.”
- Mathematical procedure for determining the present location of a vehicle.
- Achieved by calculating the current pose of the vehicle based on its velocities and the time elapsed.

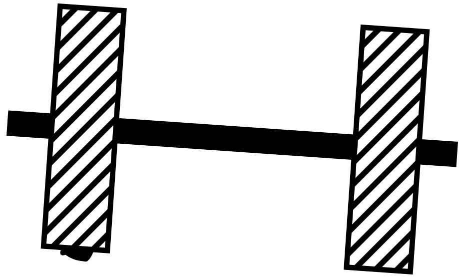
Reasons for Motion Errors



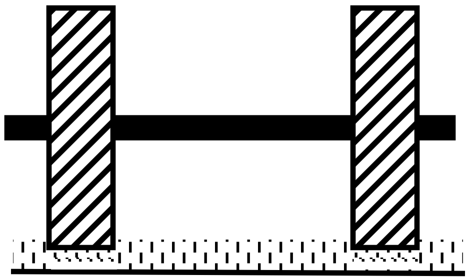
ideal case



different wheel diameters



bump



carpet

and many more …

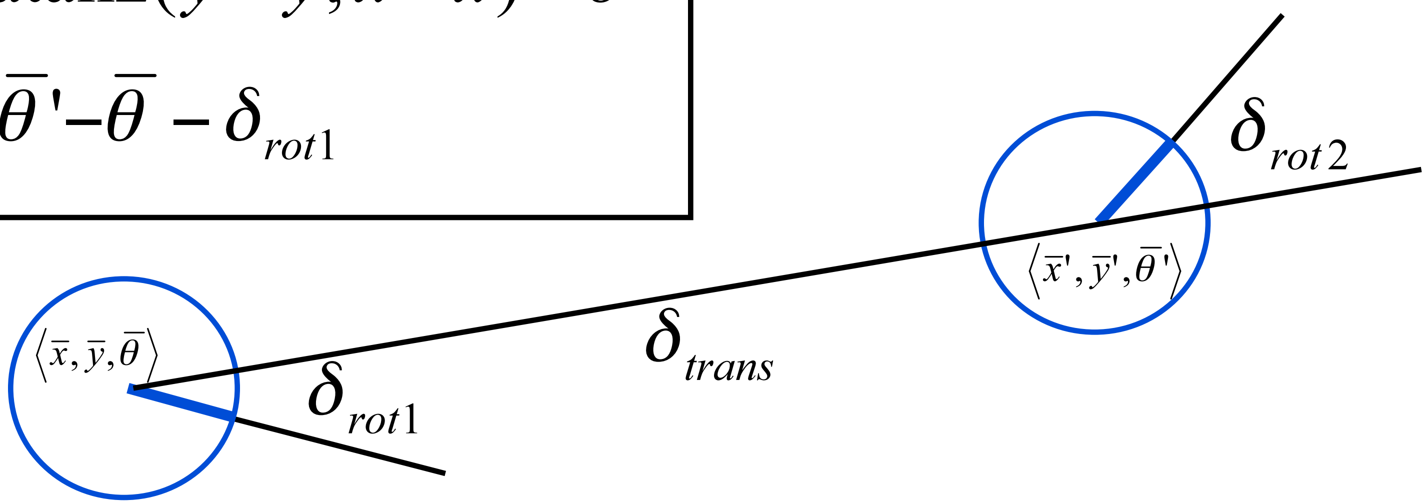
Odometry Model

- Robot moves from $\langle \bar{x}, \bar{y}, \bar{\theta} \rangle$ to $\langle \bar{x}', \bar{y}', \bar{\theta}' \rangle$.
- Odometry information $u = \langle \delta_{rot1}, \delta_{rot2}, \delta_{trans} \rangle$.

$$\delta_{trans} = \sqrt{(\bar{x}' - \bar{x})^2 + (\bar{y}' - \bar{y})^2}$$

$$\delta_{rot1} = \operatorname{atan2}(\bar{y}' - \bar{y}, \bar{x}' - \bar{x}) - \bar{\theta}$$

$$\delta_{rot2} = \bar{\theta}' - \bar{\theta} - \delta_{rot1}$$



The atan2 Function

- Extends the inverse tangent and correctly copes with the signs of x and y.

$$\text{atan2}(y, x) = \begin{cases} \text{atan}(y/x) & \text{if } x > 0 \\ \text{sign}(y)\ (\pi - \text{atan}(|y/x|)) & \text{if } x < 0 \\ 0 & \text{if } x = y = 0 \\ \text{sign}(y)\ \pi/2 & \text{if } x = 0, y \neq 0 \end{cases}$$

Noise Model for Odometry

- The measured motion is given by the true motion corrupted with noise.

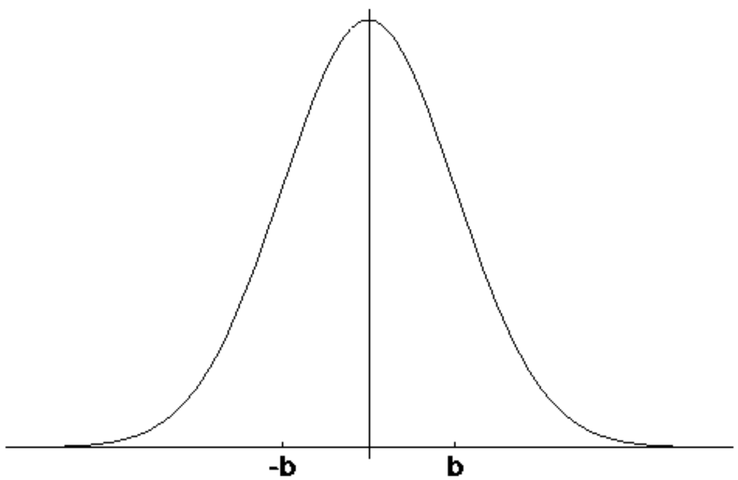
$$\hat{\delta}_{rot1} = \delta_{rot1} + \varepsilon_{\alpha_1 |\delta_{rot1}| + \alpha_2 |\delta_{trans}|}$$

$$\hat{\delta}_{trans} = \delta_{trans} + \varepsilon_{\alpha_3 |\delta_{trans}| + \alpha_4 |\delta_{rot1} + \delta_{rot2}|}$$

$$\hat{\delta}_{rot2} = \delta_{rot2} + \varepsilon_{\alpha_1 |\delta_{rot2}| + \alpha_2 |\delta_{trans}|}$$

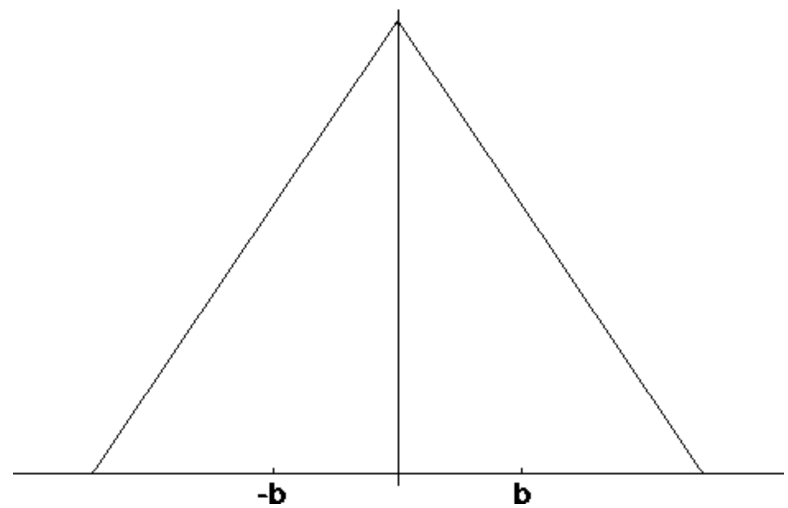
Typical Distributions for Probabilistic Motion Models

Normal distribution



$$\varepsilon_{\sigma^2}(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\frac{x^2}{\sigma^2}}$$

Triangular distribution



$$\varepsilon_{\sigma^2}(x) = \begin{cases} 0 \text{ if } |\mathrm{x}| > \sqrt{6\sigma^2} \\ \frac{\sqrt{6\sigma^2} - |x|}{6\sigma^2} \end{cases}$$

Calculating the Probability (zero-centered)

- For a normal distribution

1. Algorithm **prob_normal_distribution**(a,b):
2. return $\frac{1}{\sqrt{2\pi\, b^2}} \exp\left\{-\frac{1}{2}\frac{a^2}{b^2}\right\}$

- For a triangular distribution

1. Algorithm **prob_triangular_distribution**(a,b):
2. return $\max\left\{0, \frac{1}{\sqrt{6}\, b} - \frac{|a|}{6\, b^2}\right\}$

Calculating the Posterior Given x, x', and u

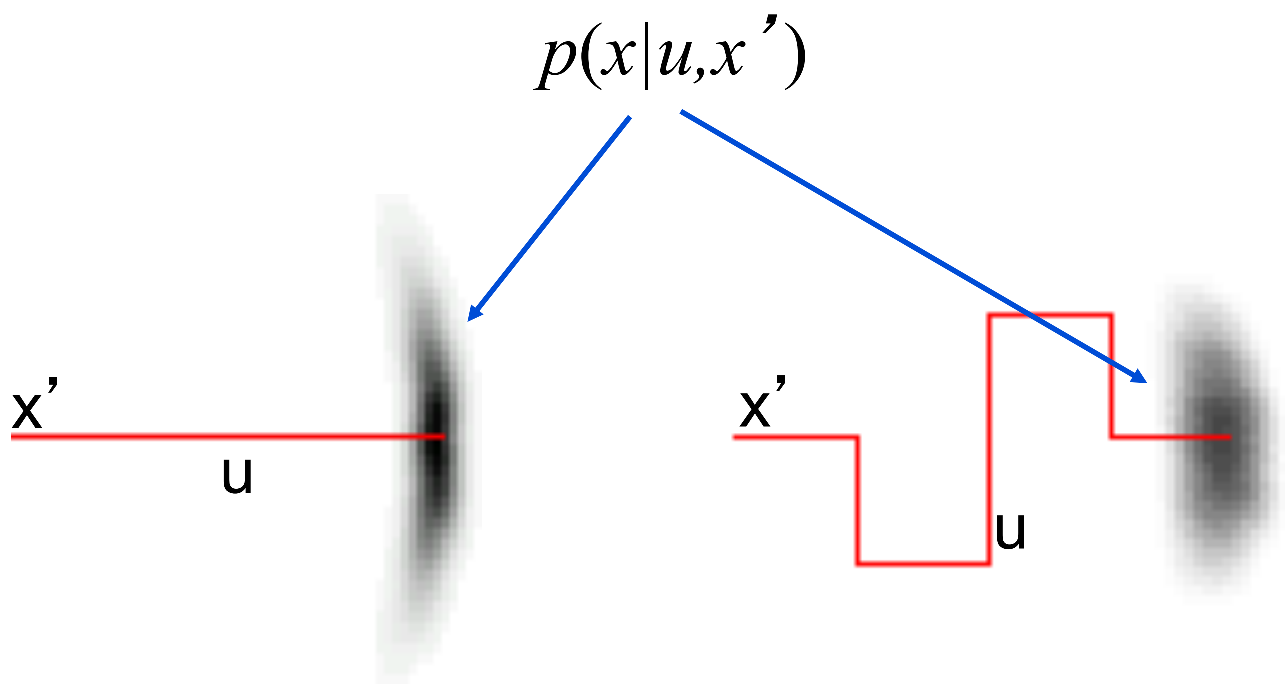
1. Algorithm **motion_model_odometry(x,x',u)**
2. $\delta_{trans} = \sqrt{(\bar{x}'-\bar{x})^2 + (\bar{y}'-\bar{y})^2}$
3. $\delta_{rot1} = \text{atan2}(\bar{y}'-\bar{y}, \bar{x}'-\bar{x}) - \bar{\theta}$
4. $\delta_{rot2} = \bar{\theta}' - \bar{\theta} - \delta_{rot1}$
5. $\hat{\delta}_{trans} = \sqrt{(x'-x)^2 + (y'-y)^2}$
6. $\hat{\delta}_{rot1} = \text{atan2}(y'-y, x'-x) - \bar{\theta}$
7. $\hat{\delta}_{rot2} = \theta' - \theta - \hat{\delta}_{rot1}$
8. $p_1 = \text{prob}(\delta_{\text{rot1}} - \hat{\delta}_{\text{rot1}}, \alpha_1 |\hat{\delta}_{\text{rot1}}| + \alpha_2 \hat{\delta}_{\text{trans}})$
9. $p_2 = \text{prob}(\delta_{\text{trans}} - \hat{\delta}_{\text{trans}}, \alpha_3 \hat{\delta}_{\text{trans}} + \alpha_4 (|\hat{\delta}_{\text{rot1}}| + |\hat{\delta}_{\text{rot2}}|))$
10. $p_3 = \text{prob}(\delta_{\text{rot2}} - \hat{\delta}_{\text{rot2}}, \alpha_1 |\hat{\delta}_{\text{rot2}}| + \alpha_2 \hat{\delta}_{\text{trans}})$
11. return $p_1 \cdot p_2 \cdot p_3$

odometry values (u)

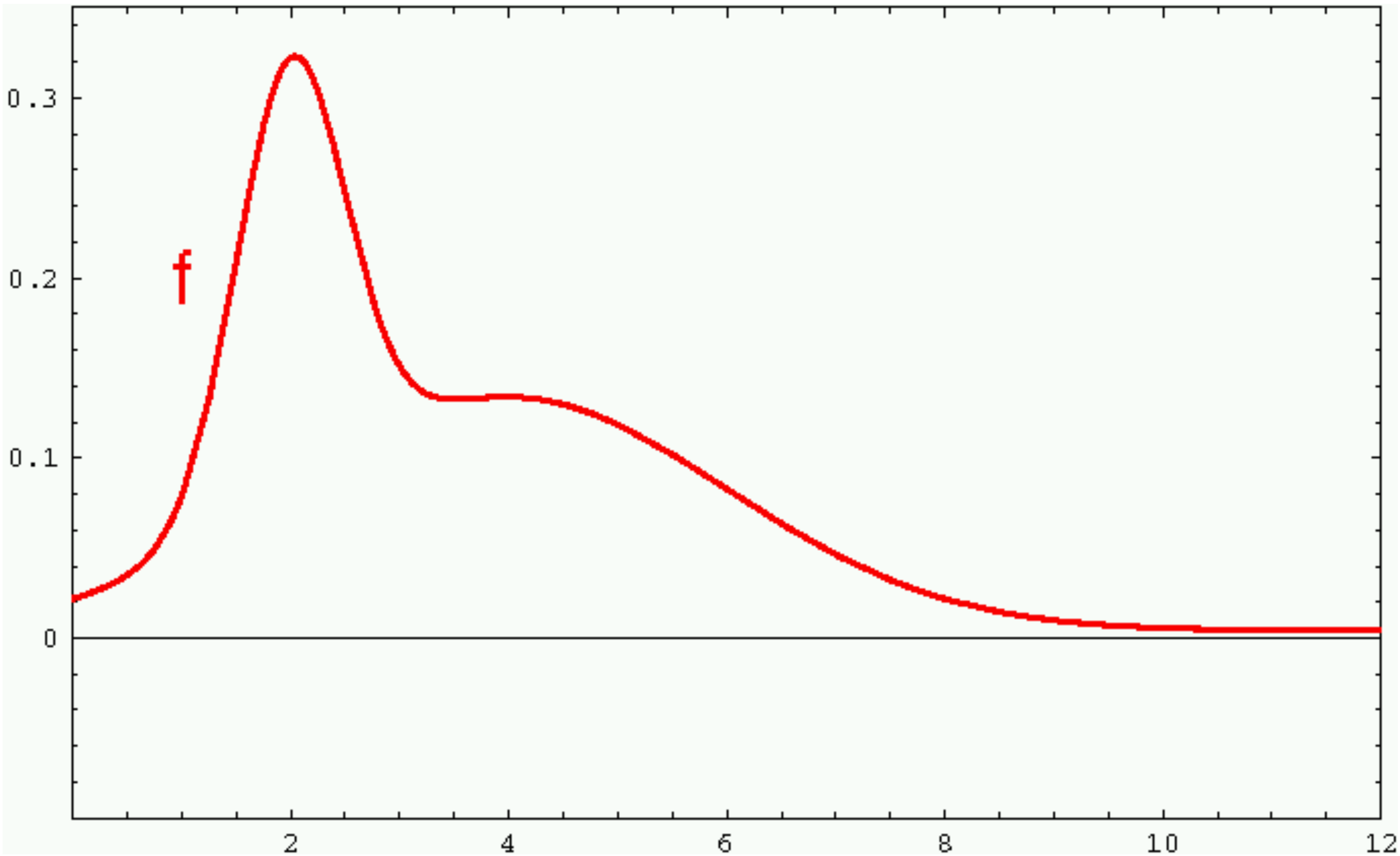
values of interest (x,x')

Application

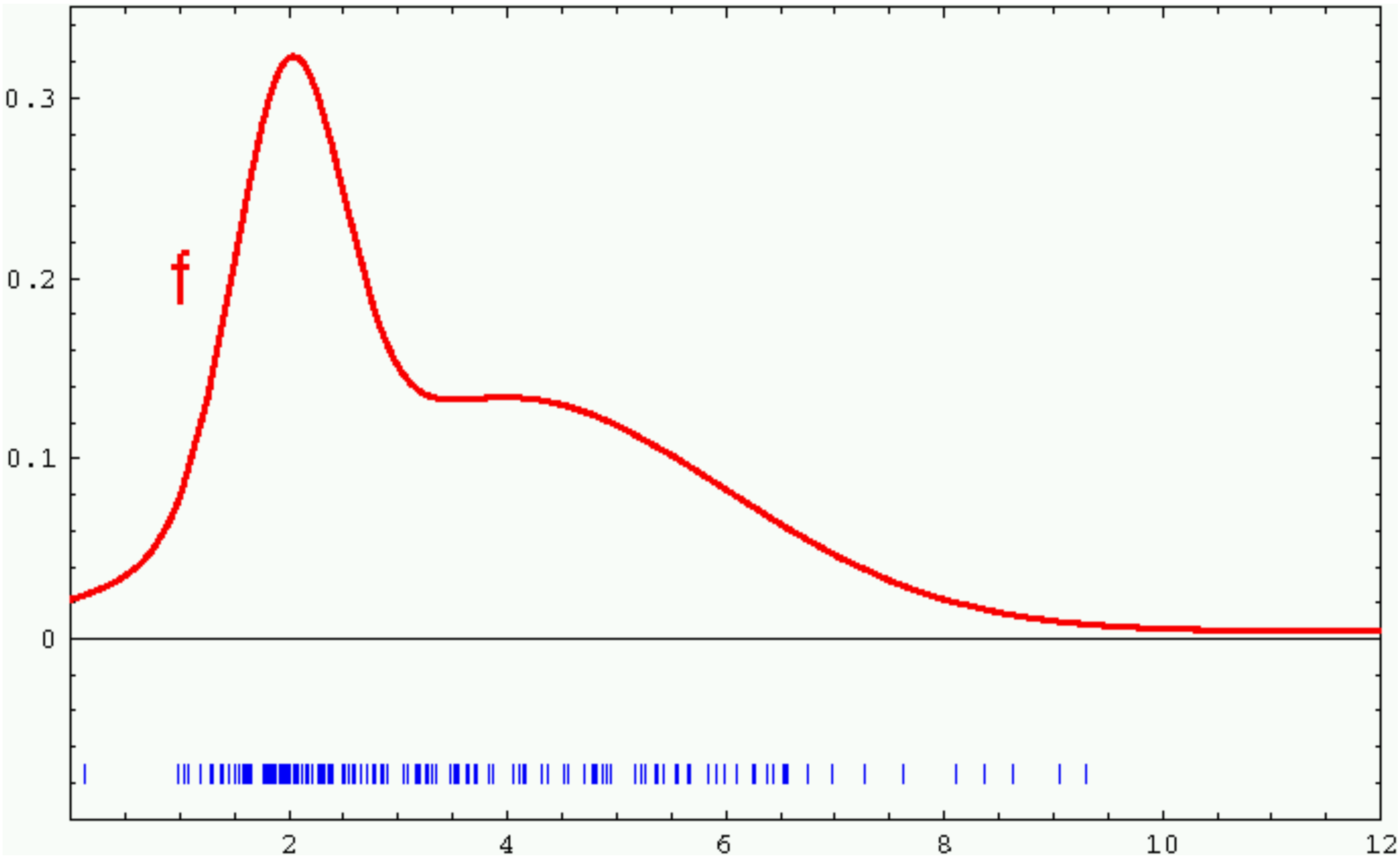
- Repeated application of the sensor model for short movements.
- Typical banana-shaped distributions obtained for 2d-projection of 3d posterior.



Sample-based Density Representation



Sample-based Density Representation



How to Sample from Normal or Triangular Distributions?

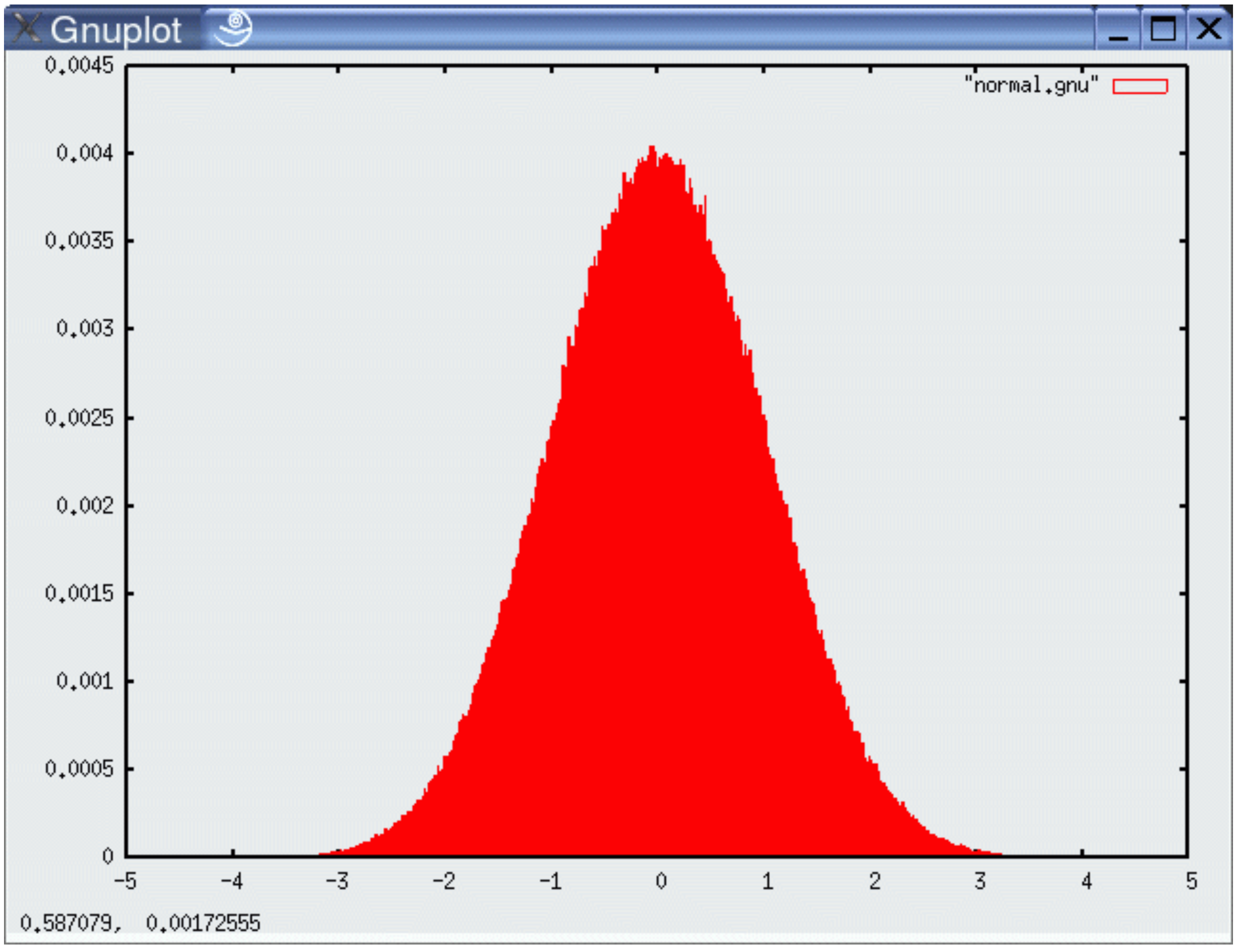
- Sampling from a normal distribution

1. Algorithm **sample_normal_distribution**(b):
2. return $\frac{1}{2}\sum_{i=1}^{12} rand(-b,b)$

- Sampling from a triangular distribution

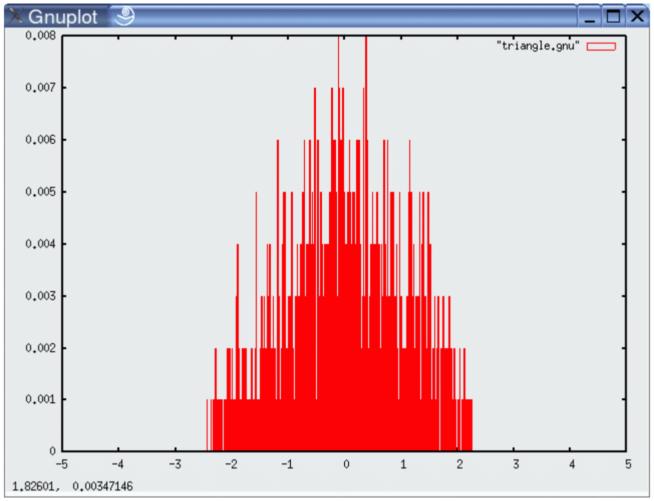
1. Algorithm **sample_triangular_distribution**(b):
2. return $\frac{\sqrt{6}}{2}\left[\mathbf{rand}(-b,b)+\mathbf{rand}(-b,b)\right]$

Normally Distributed Samples

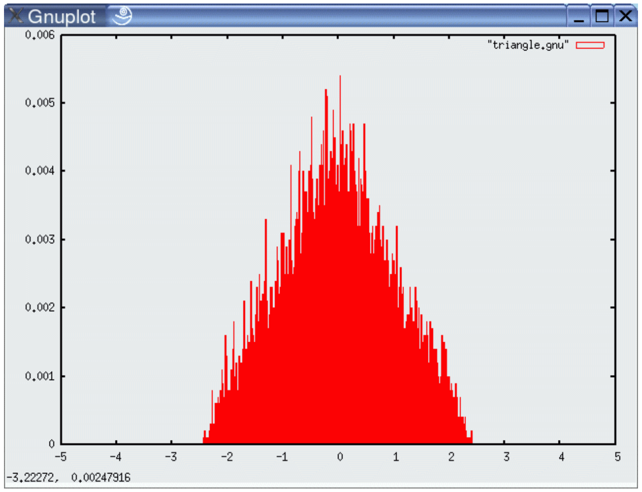


10^6 samples

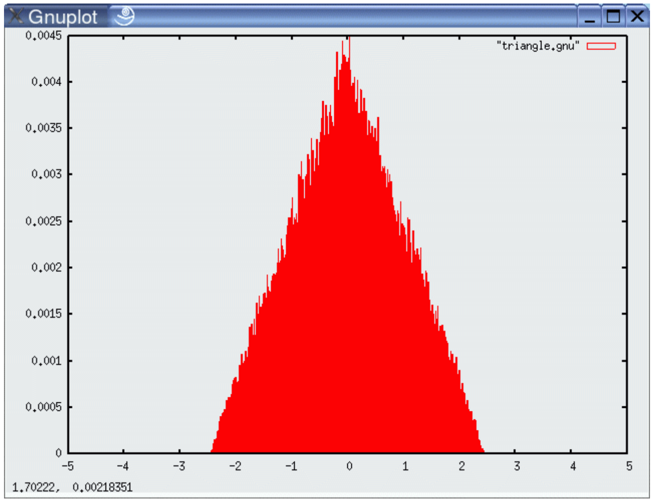
For Triangular Distribution



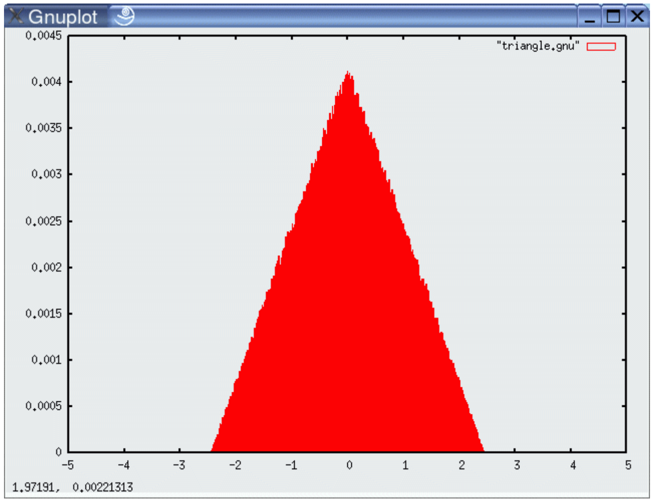
10^3 samples



10^4 samples



10^5 samples



10^6 samples

Rejection Sampling

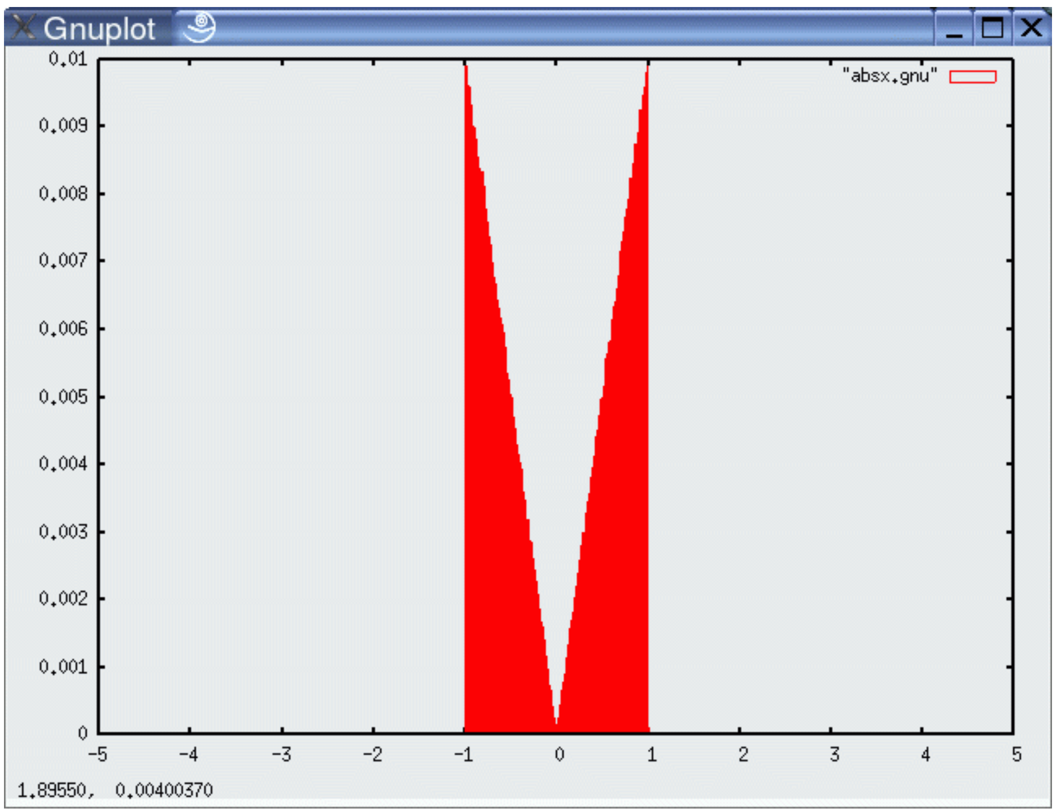
- Sampling from arbitrary distributions

Algorithm **sample_distribution**(f,b):
repeat
    $x = \mathbf{rand}(-b, b)$
    $y = \mathbf{rand}(0, \max\{f(x) \mid x \in (-b, b)\})$
until $(y \leq f(x))$
return x

Example

- Sampling from

$$f(x) = \begin{cases} \text{abs}(x) & x \in [-1;1] \\ 0 & \text{otherwise} \end{cases}$$



Sample Odometry Motion Model

1. Algorithm **sample_motion_model**(u, x):

   $u = \langle \delta_{rot1}, \delta_{rot2}, \delta_{trans} \rangle, x = \langle x, y, \theta \rangle$

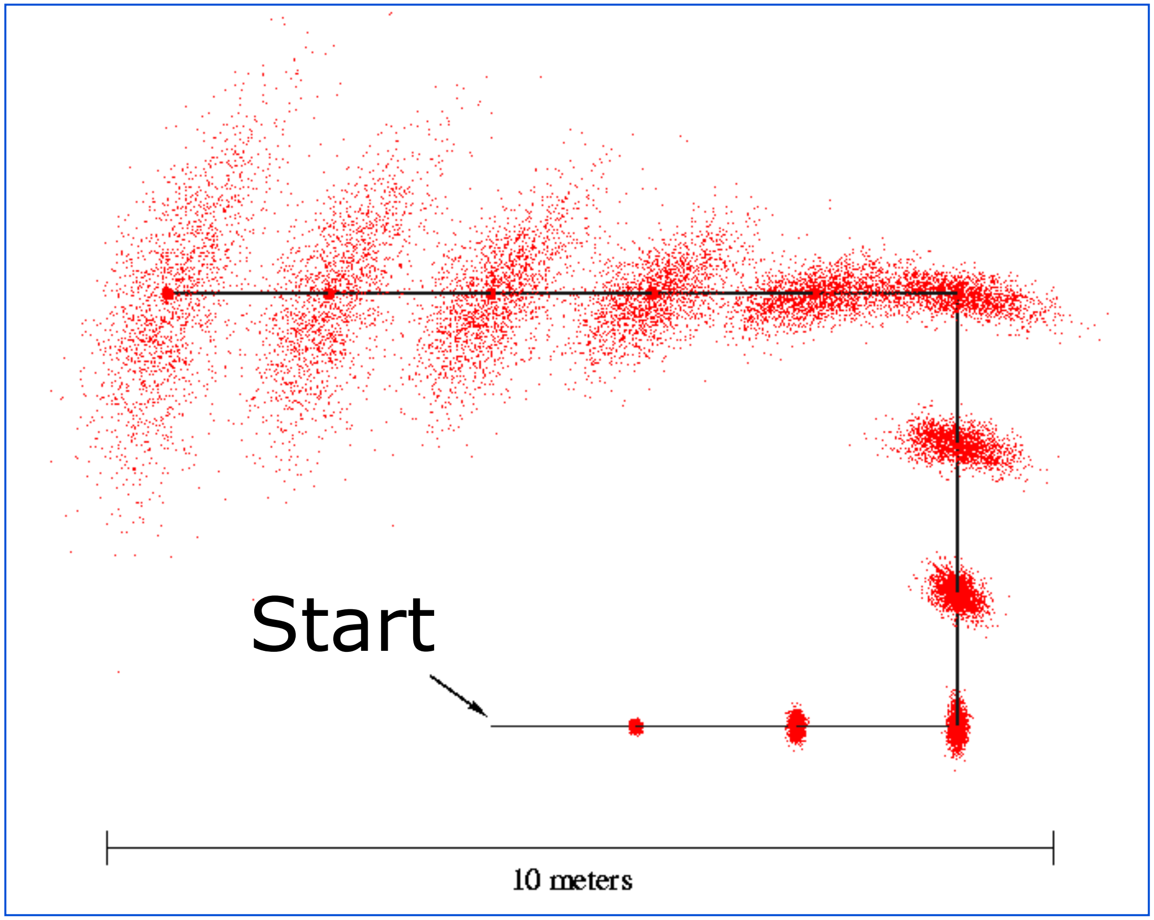
1. $\hat{\delta}_{rot1} = \delta_{rot1} + \text{sample}(\alpha_1 |\delta_{rot1}| + \alpha_2 \, \delta_{trans})$
2. $\hat{\delta}_{trans} = \delta_{trans} + \text{sample}(\alpha_3 \, \delta_{trans} + \alpha_4 \, (|\delta_{rot1}| + |\delta_{rot2}|))$
3. $\hat{\delta}_{rot2} = \delta_{rot2} + \text{sample}(\alpha_1 |\delta_{rot2}| + \alpha_2 \, \delta_{trans})$

4. $x' = x + \hat{\delta}_{trans} \cos(\theta + \hat{\delta}_{rot1})$
5. $y' = y + \hat{\delta}_{trans} \sin(\theta + \hat{\delta}_{rot1})$
6. $\theta' = \theta + \hat{\delta}_{rot1} + \hat{\delta}_{rot2}$

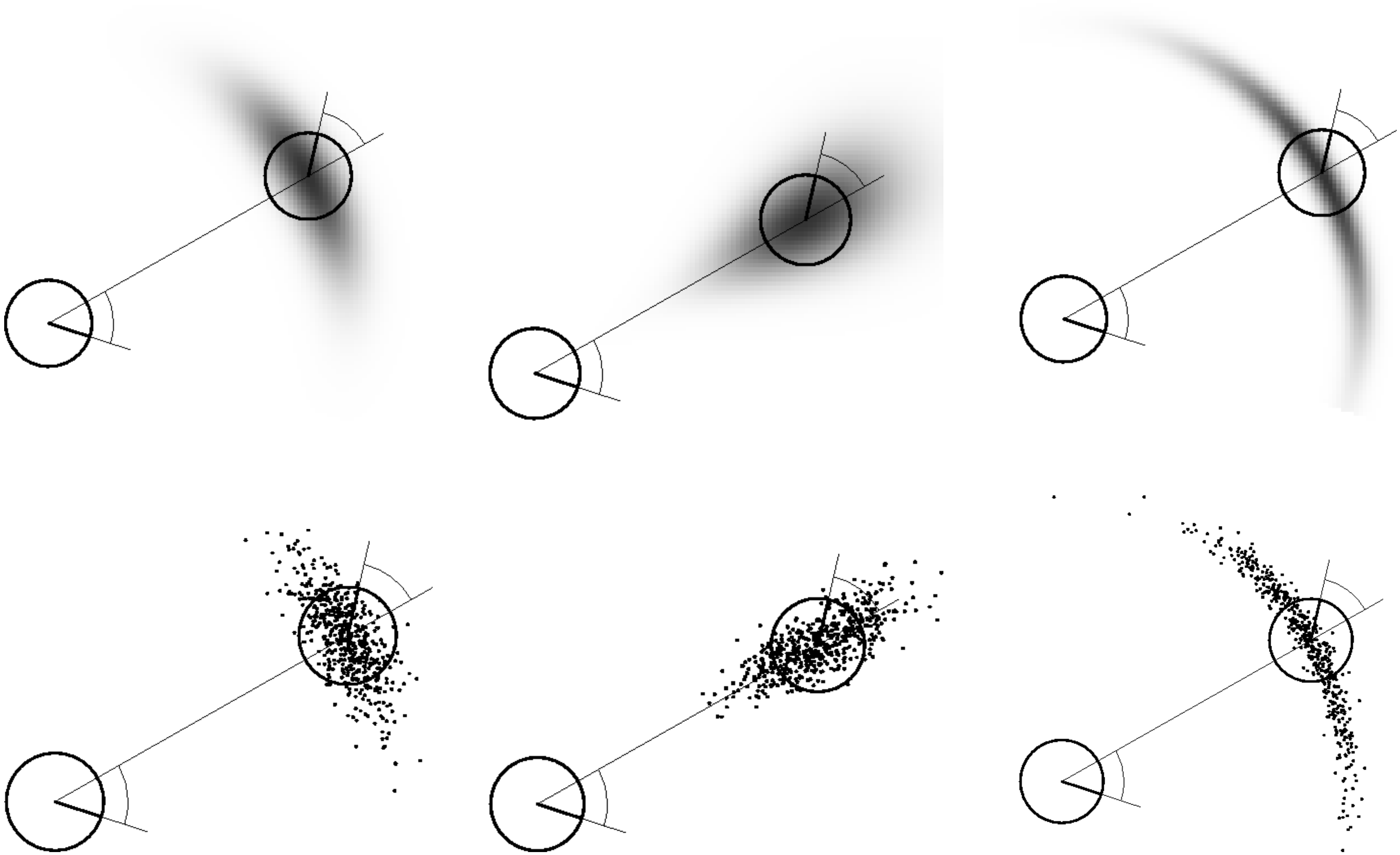
7. Return $\langle x', y', \theta' \rangle$

sample_normal_distribution

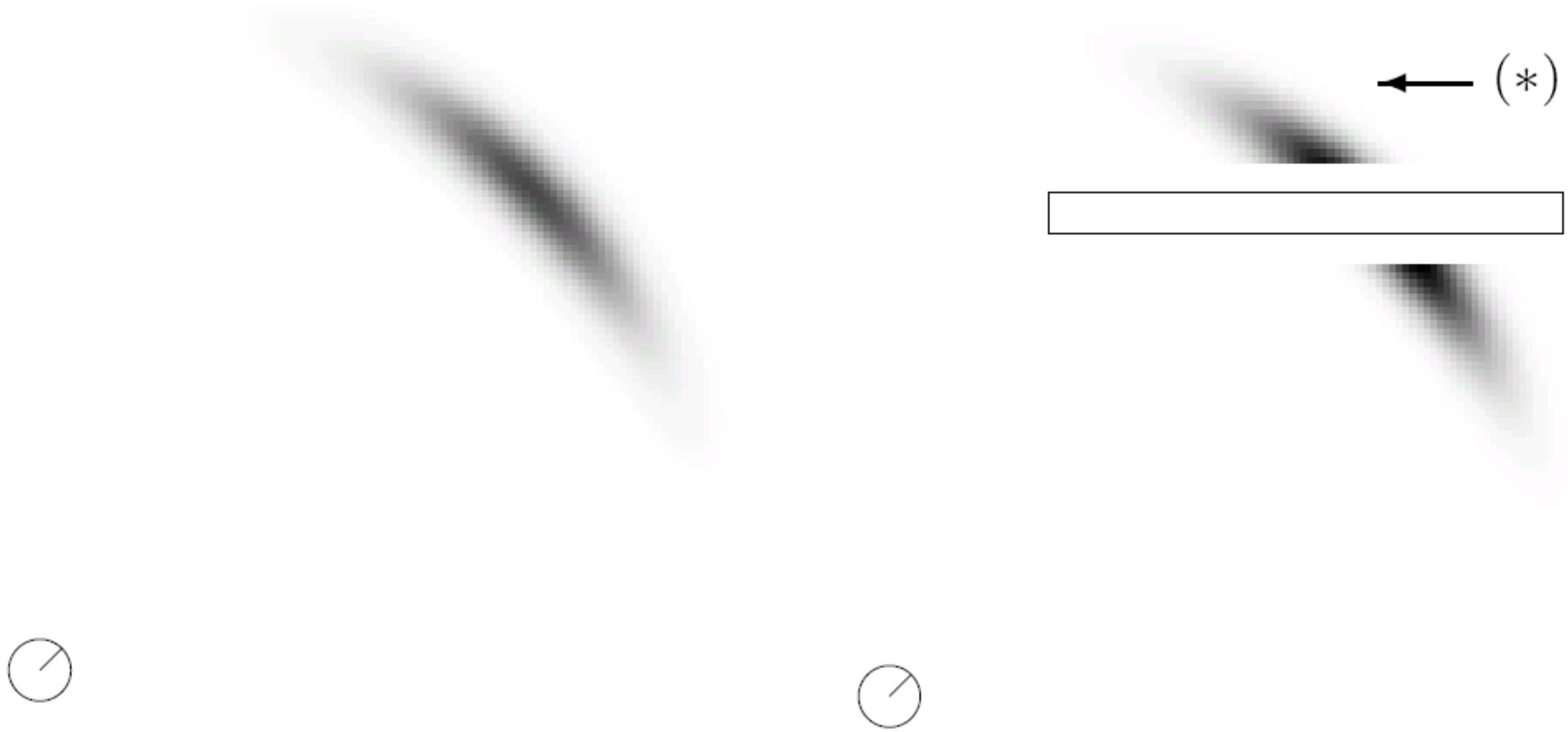
Sampling from Our Motion Model



Examples (Odometry-Based)



Map-Consistent Motion Model



$$p(x \mid u, x') \neq p(x \mid u, x', m)$$

Approximation: $p(x \mid u, x', m) = \eta \; p(x \mid m) \; p(x \mid u, x')$

In the presence of the map : samples cannot be at occupied cells

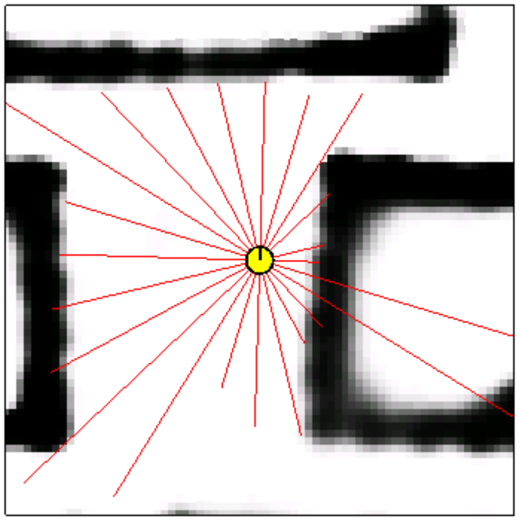
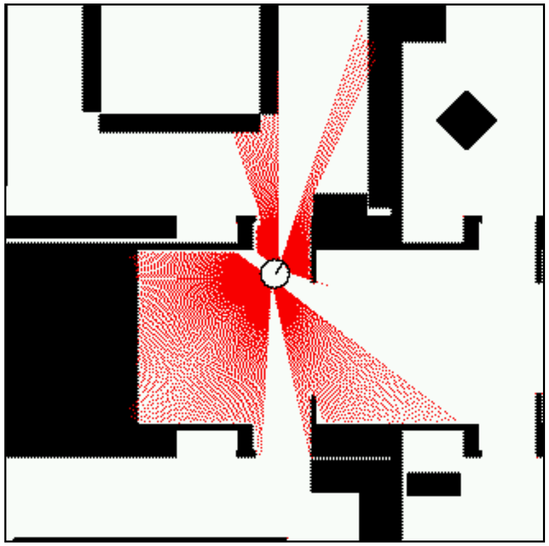
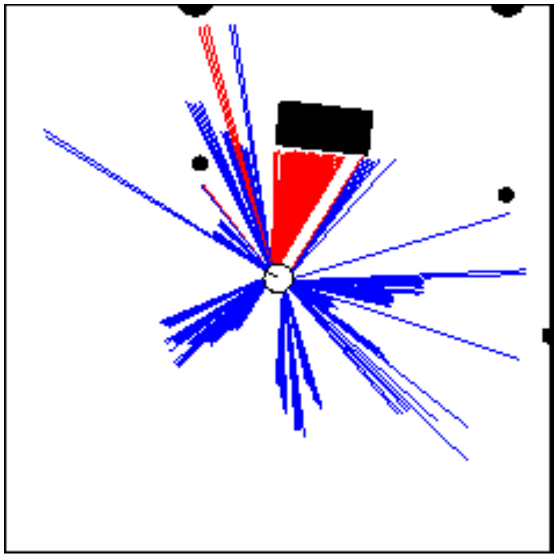
Summary

- We discussed motion models for odometry-based and velocity-based systems
- We discussed ways to calculate the posterior probability $p(x| x', u)$.
- We also described how to sample from $p(x| x', u)$.
- Typically the calculations are done in fixed time intervals Δt.
- In practice, the parameters of the models have to be learned.
- We also discussed an extended motion model that takes the map into account.

Sensors for Mobile Robots

- Contact sensors: Bumpers
- Internal sensors
  - Accelerometers (spring-mounted masses)
  - Gyroscopes (spinning mass, laser light)
  - Compasses, inclinometers (earth magnetic field, gravity)
- Proximity sensors
  - Sonar (time of flight)
  - Radar (phase and frequency)
  - Laser range-finders (triangulation, tof, phase)
  - Infrared (intensity)
- Visual sensors: Cameras
- Satellite-based sensors: GPS

Proximity Sensors

- The central task is to determine *P(z|x)*, i.e., the probability of a measurement *z* given that the robot is at position *x*.
- **Question**: Where do the probabilities come from?
- **Approach**: Let’s try to explain a measurement.

Beam-based Sensor Model

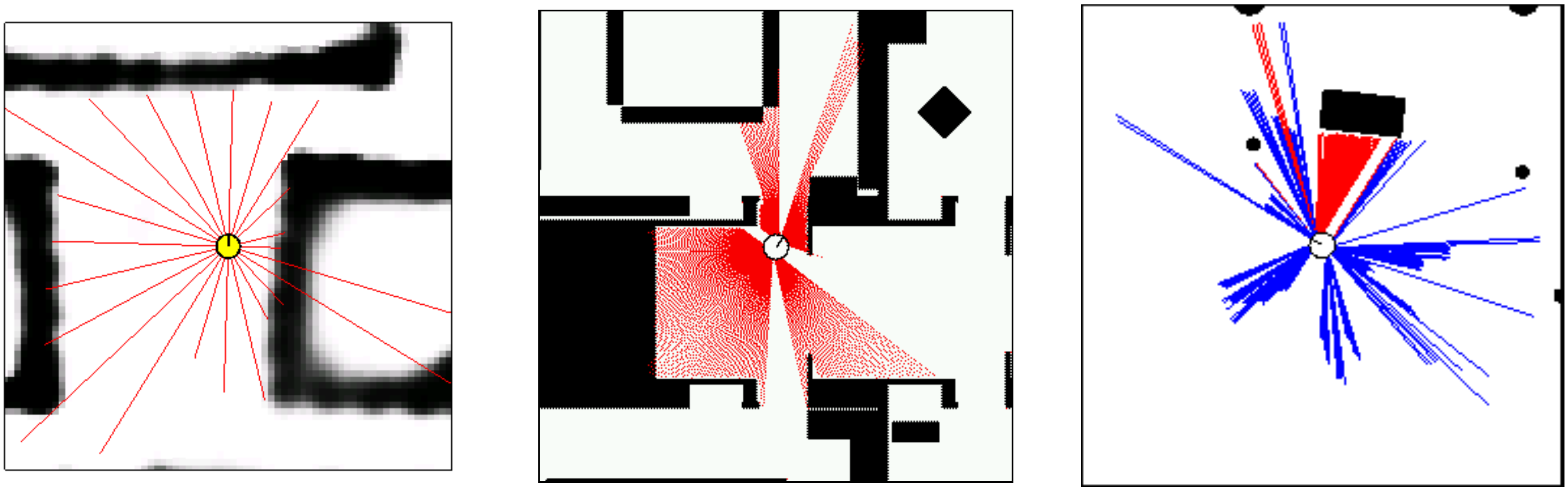
- Scan z consists of K measurements.

$$z = \{z_1, z_2, \ldots, z_K\}$$

- Individual measurements are independent given the robot position.

$$P(z \mid x, m) = \prod_{k=1}^{K} P(z_k \mid x, m)$$

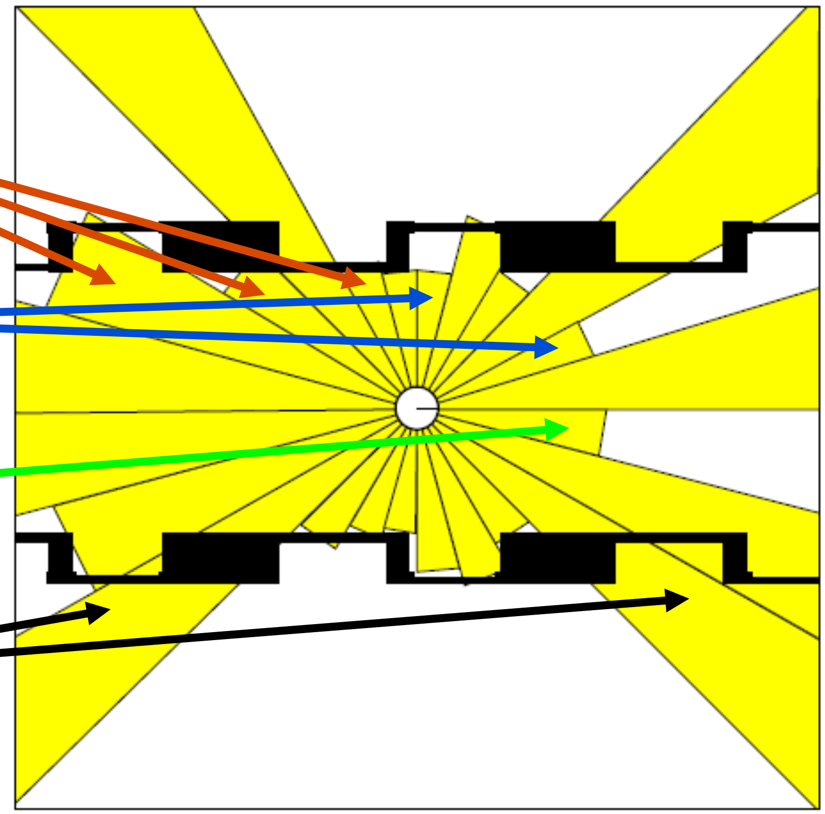
Beam-based Sensor Model



$$P(z \mid x, m) = \prod_{k=1}^{K} P(z_k \mid x, m)$$

Typical Measurement Errors of an Range Measurements

1. Beams reflected by obstacles
2. Beams reflected by persons / caused by crosstalk
3. Random measurements
4. Maximum range measurements

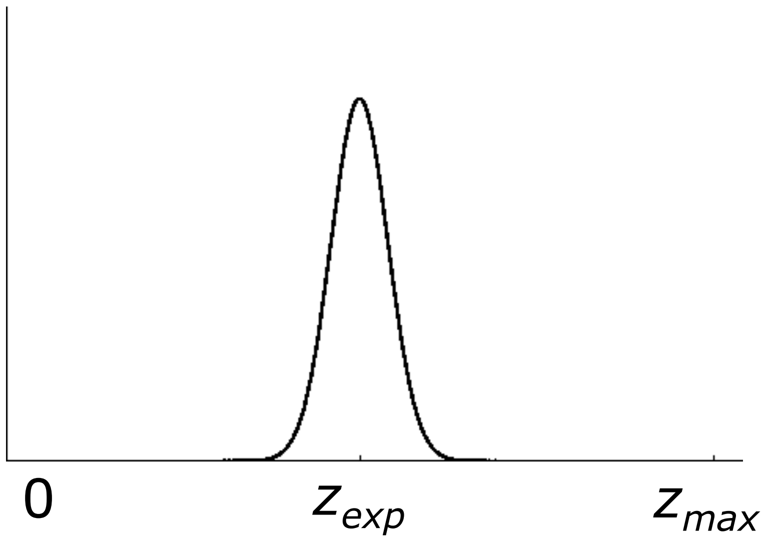


Proximity Measurement

- Measurement can be caused by …
  - a known obstacle.
  - cross-talk.
  - an unexpected obstacle (people, furniture, …).
  - missing all obstacles (total reflection, glass, …).
- Noise is due to uncertainty …
  - in measuring distance to known obstacle.
  - in position of known obstacles.
  - in position of additional obstacles.
  - whether obstacle is missed.

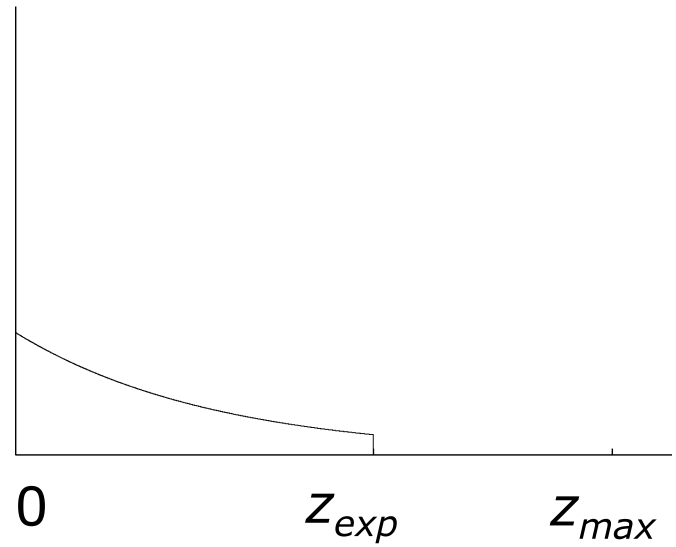
Beam-based Proximity Model

Measurement noise



$$P_{hit}(z \mid x, m) = \eta \frac{1}{\sqrt{2\pi b}} e^{-\frac{1}{2}\frac{(z - z_{exp})^2}{b}}$$

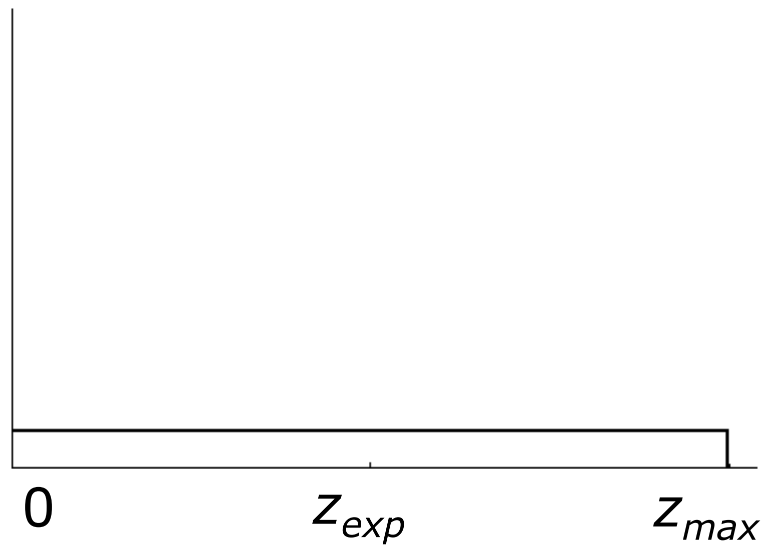
Unexpected obstacles



$$P_{unexp}(z \mid x, m) = \begin{cases} \eta \, \lambda \, e^{-\lambda z} & z < z_{exp} \\ 0 & otherwise \end{cases}$$

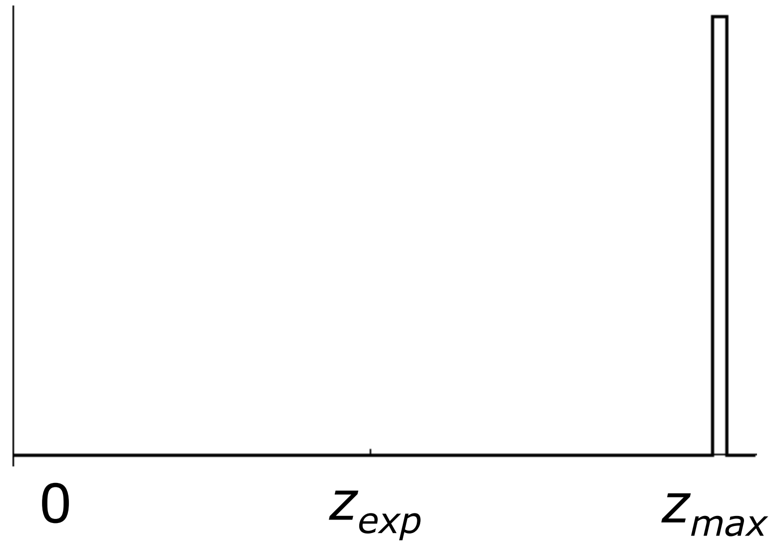
Beam-based Proximity Model

Random measurement



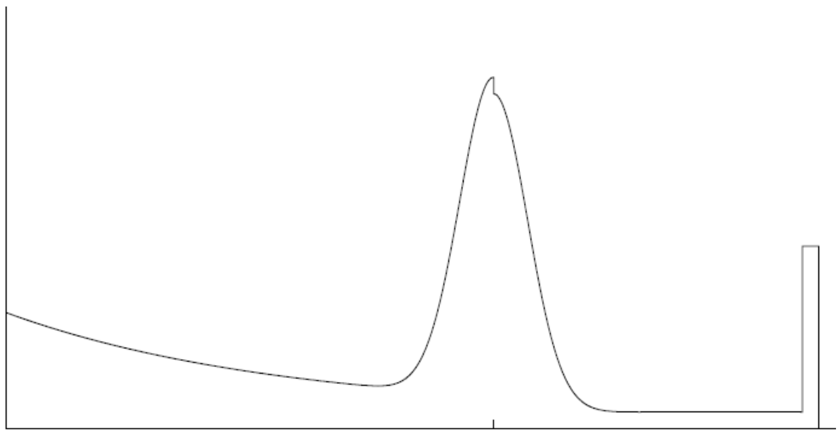
$$P_{rand}(z \mid x, m) = \eta \frac{1}{z_{\max}}$$

Max range



$$P_{\max}(z \mid x, m) = \eta \frac{1}{z_{small}}$$

Resulting Mixture Density

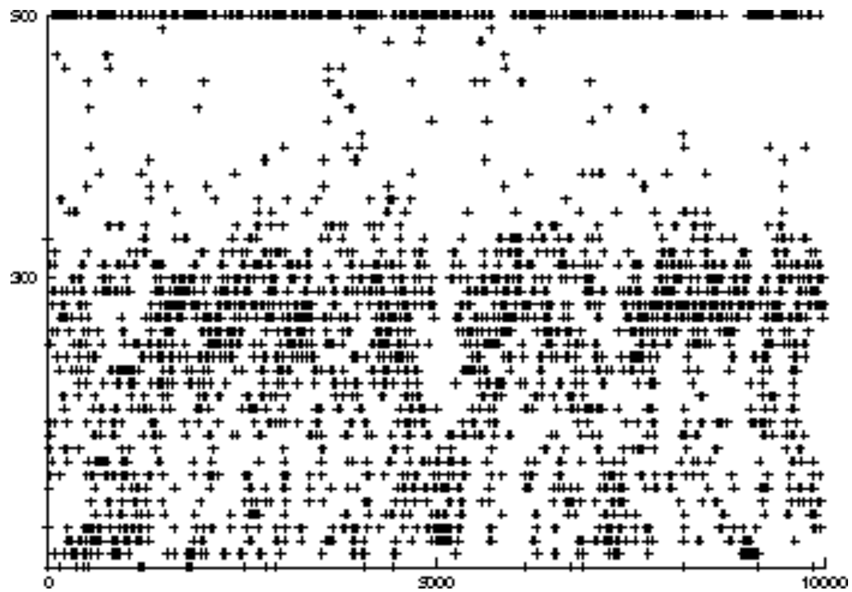


$$P(z \mid x, m) = \begin{pmatrix} \alpha_{\text{hit}} \\ \alpha_{\text{unexp}} \\ \alpha_{\max} \\ \alpha_{\text{rand}} \end{pmatrix}^T \cdot \begin{pmatrix} P_{\text{hit}}(z \mid x, m) \\ P_{\text{unexp}}(z \mid x, m) \\ P_{\max}(z \mid x, m) \\ P_{\text{rand}}(z \mid x, m) \end{pmatrix}$$

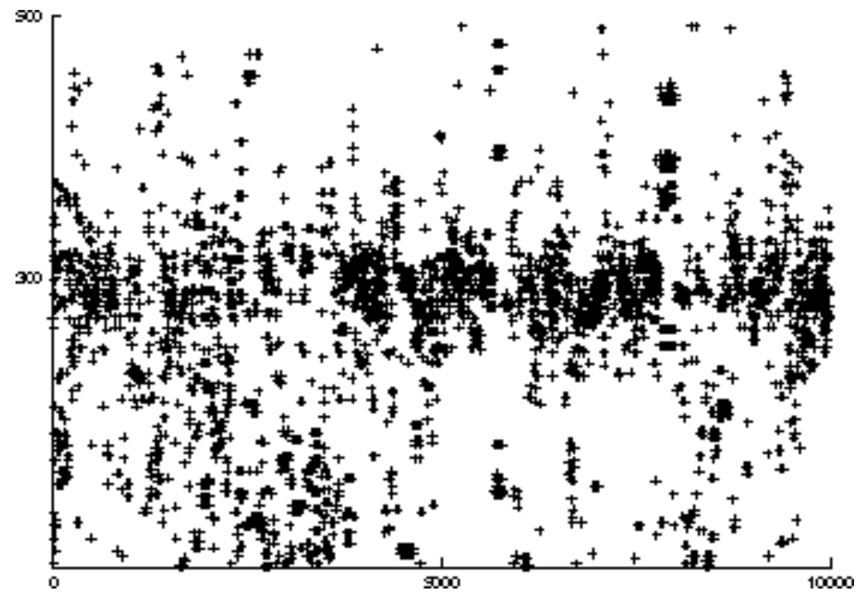
How can we determine the model parameters?

Raw Sensor Data

Measured distances for expected distance of 300 cm.



Sonar



Laser

Approximation

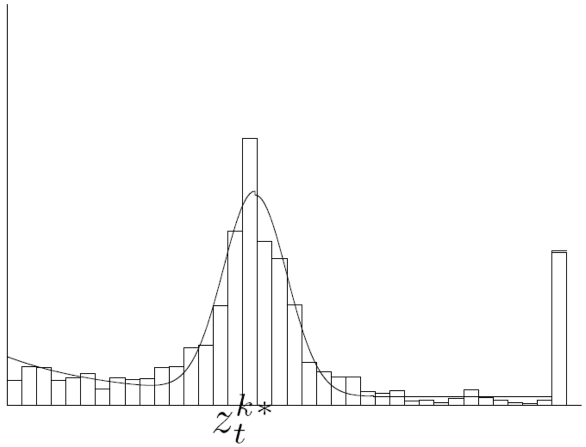
- Maximize log likelihood of the data

$$P(z \mid z_{\exp})$$

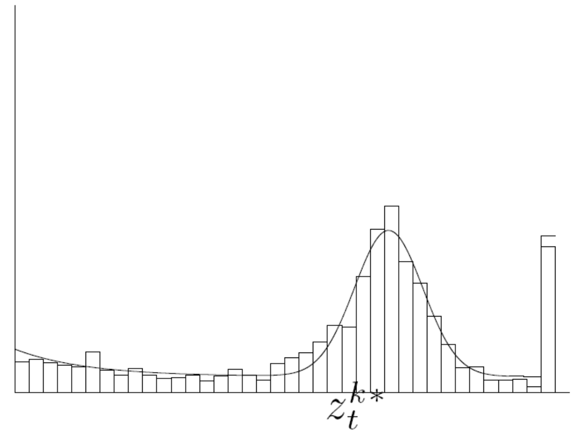
- Search space of n-1 parameters.
  - Hill climbing
  - Gradient descent
  - Genetic algorithms
  - Expectation maximization
  - ML estimate of the parameters

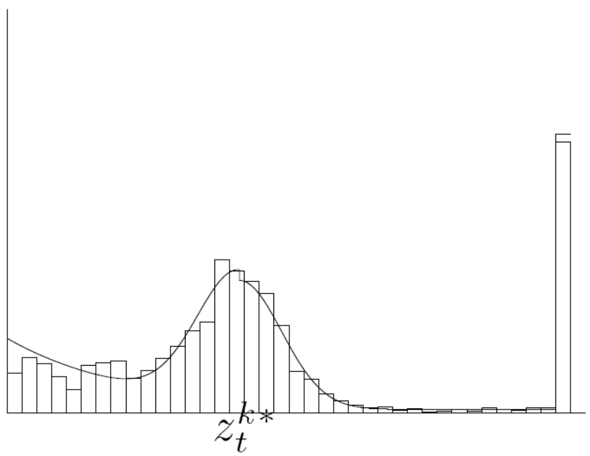
- Deterministically compute the n-th parameter to satisfy normalization constraint.

Approximation Results

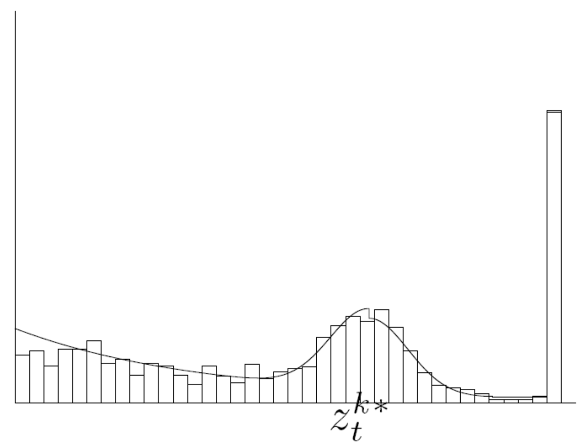


Laser





Sonar

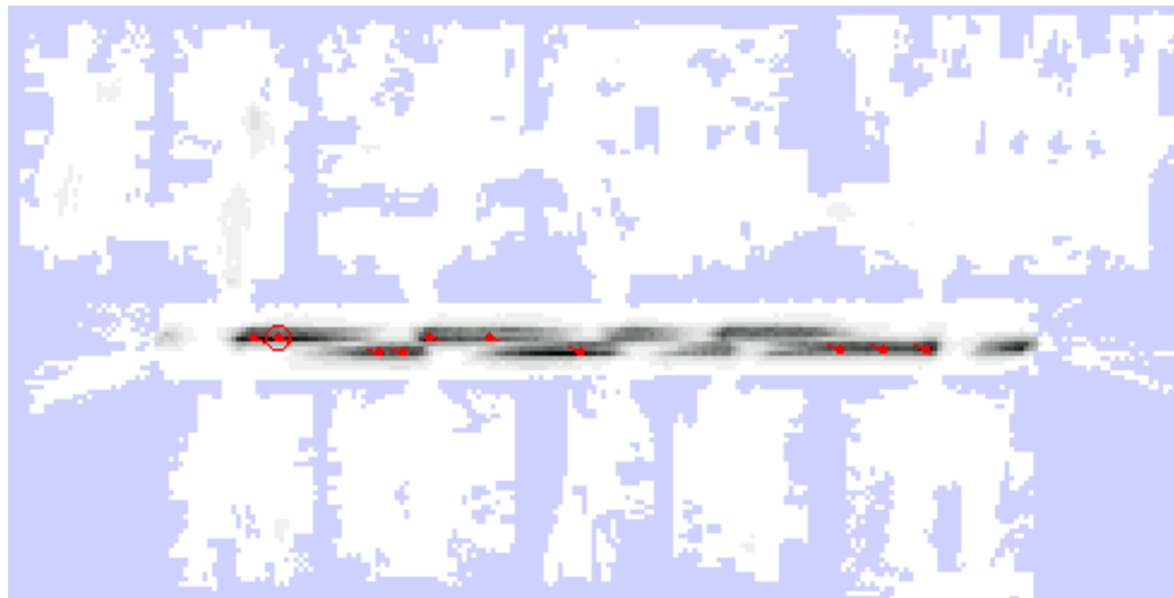


300cm

400cm

Example

z



$P(z|x,m)$

Measurements along corridor are more likely

Scan and likelihood evaluated along corridor

Summary Beam-based Model

- Assumes independence between beams.
  - Justification?
  - Overconfident!
- Models physical causes for measurements.
  - Mixture of densities for these causes.
  - Assumes independence between causes. Problem?
- Implementation
  - Learn parameters based on real data.
  - Different models should be learned for different angles at which the sensor beam hits the obstacle.
  - Determine expected distances by ray-tracing.
  - Expected distances can be pre-processed.

Scan-based Model

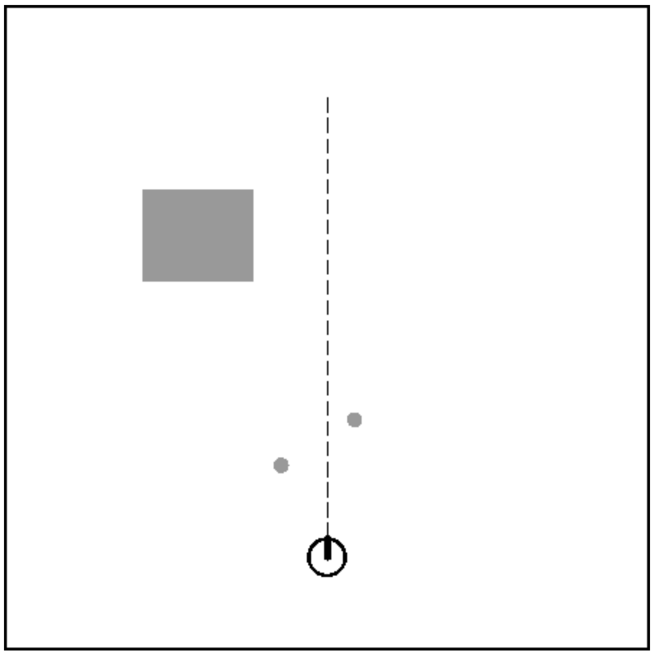
- Beam-based model is …
  - not smooth for small obstacles and at edges.
  - not very efficient.
  - Small change in pose – large change in likelihood

- Idea: Instead of following along the beam, just check the end point.

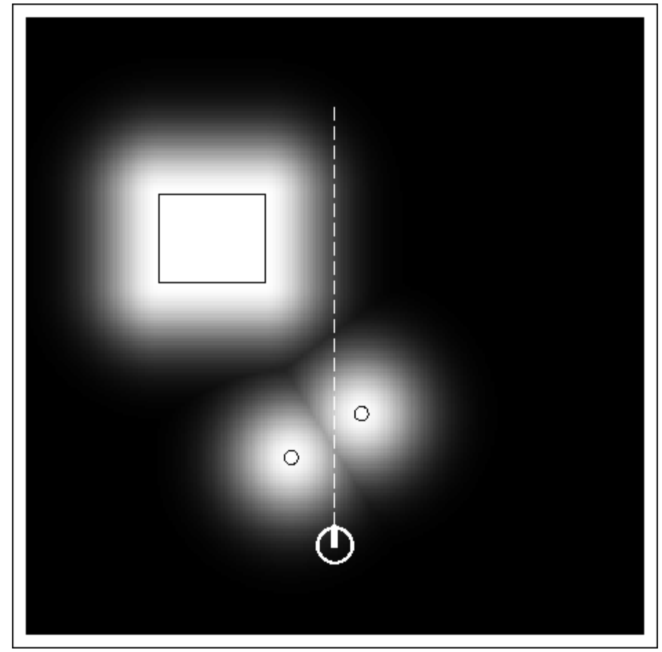
Scan-based Model

- Probability is a mixture of …
  - a Gaussian distribution with mean at distance to closest obstacle,
  - a uniform distribution for random measurements, and
  - a small uniform distribution for max range measurements.
- Again, independence between different components is assumed.

Example

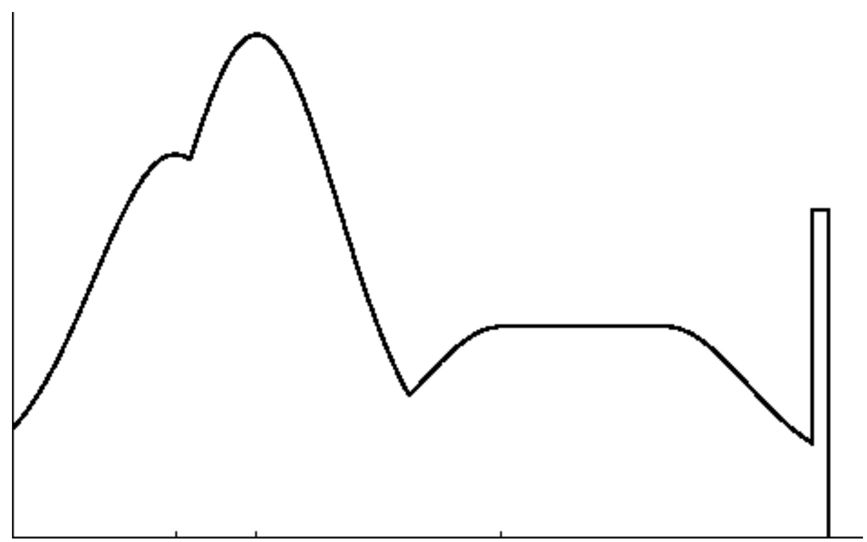


Map m



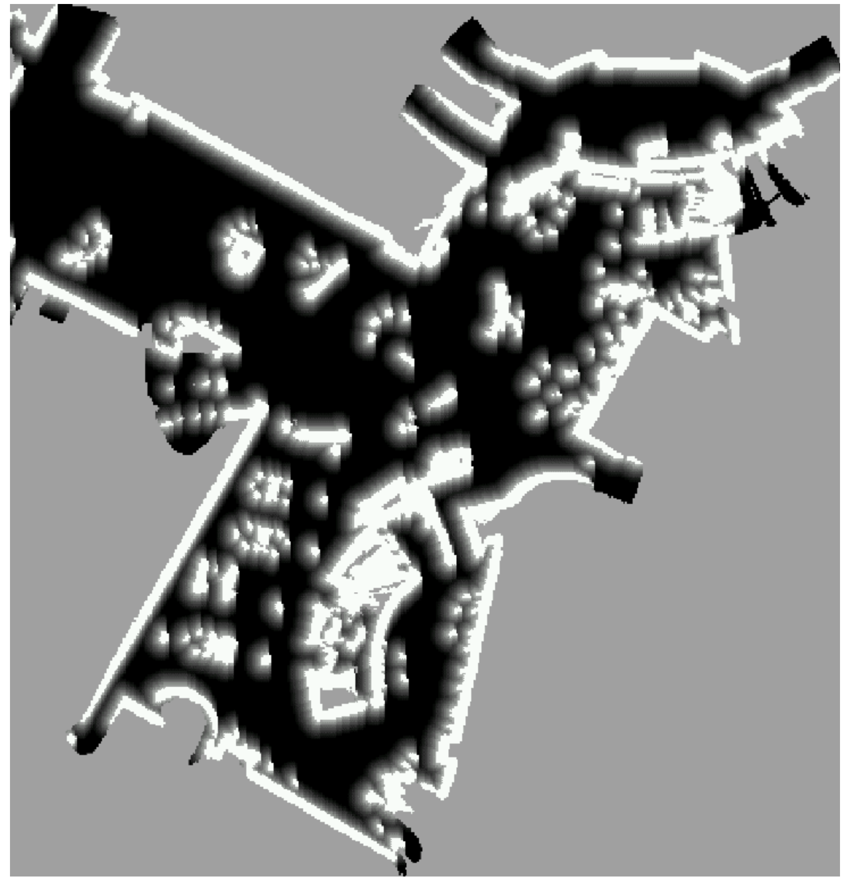
Likelihood field

$P(z|x,m)$



San Jose Tech Museum

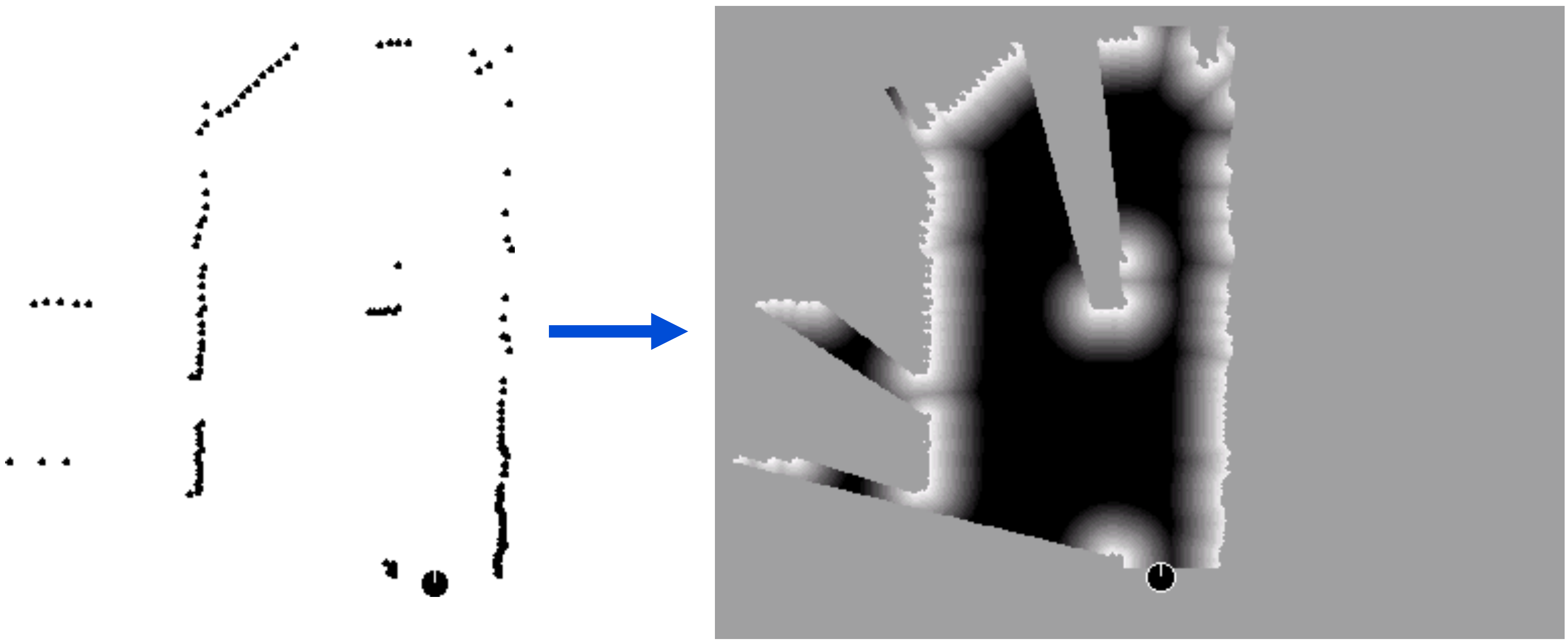
Occupancy grid map



Likelihood field

Scan Matching

- Extract likelihood field from scan and use it to match different scan.



Additional Models of Proximity Sensors

- Map matching (sonar,laser): generate small, local maps from sensor data and match local maps against global model.

- Scan matching (laser): map is represented by scan endpoints, match scan into this map.

- Features (sonar, laser, vision): Extract features such as doors, hallways from sensor data.

Landmarks

- Active beacons (*e.g.*, radio, GPS)
- Passive (*e.g.*, visual, retro-reflective)
- Standard approach is triangulation

- Sensor provides
  - distance, or
  - bearing, or
  - distance and bearing.

Distance and Bearing

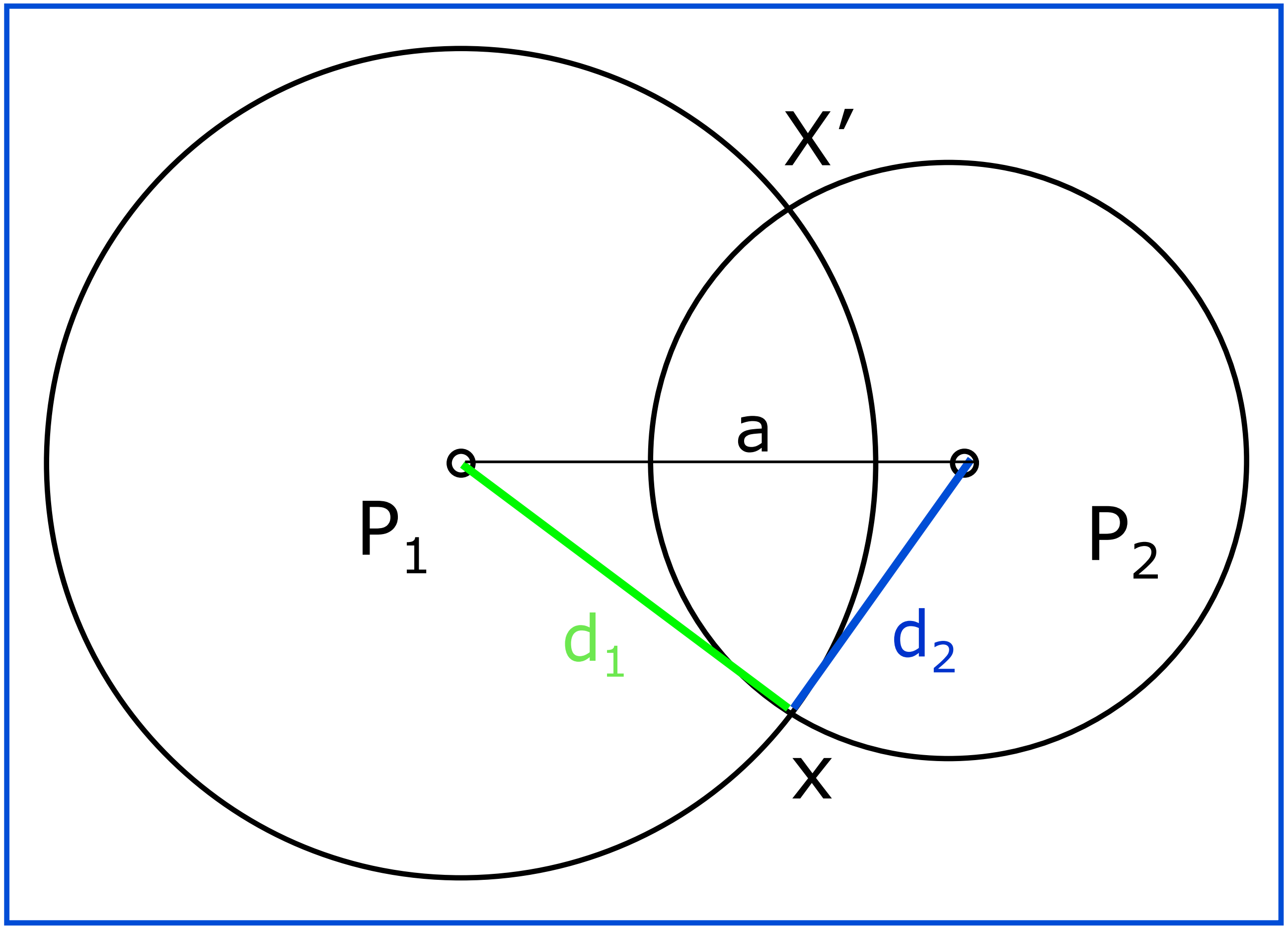
Probabilistic Model

1. Algorithm **landmark_detection_model**(z,x,m):
   $z = \langle i, d, \alpha \rangle, x = \langle x, y, \theta \rangle$
2. $\hat{d} = \sqrt{(m_x(i) - x)^2 + (m_y(i) - y)^2}$
3. $\hat{a} = \text{atan2}(m_y(i) - y, m_x(i) - x) - \theta$
4. $p_{\text{det}} = \text{prob}(\hat{d} - d, \varepsilon_d) \cdot \text{prob}(\hat{\alpha} - \alpha, \varepsilon_\alpha)$
5. Return $z_{\text{det}} p_{\text{det}} + z_{\text{fp}} P_{\text{uniform}}(z \mid x, m)$

Computing likelihood of landmark measurement

Distances Only
No Uncertainty

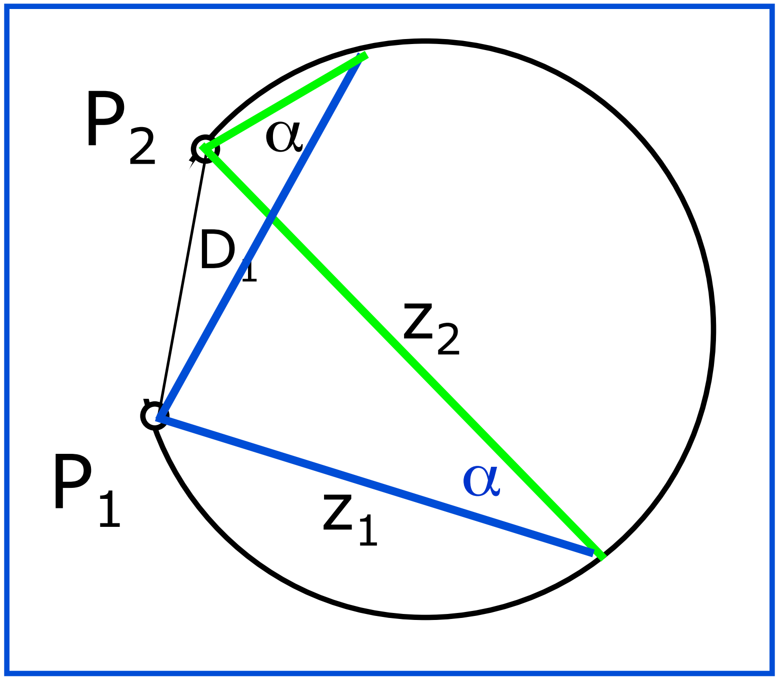
$$x = (a^2 + d_1^2 - d_2^2)/2a$$
$$y = \pm\sqrt{(d_1^2 - x^2)}$$

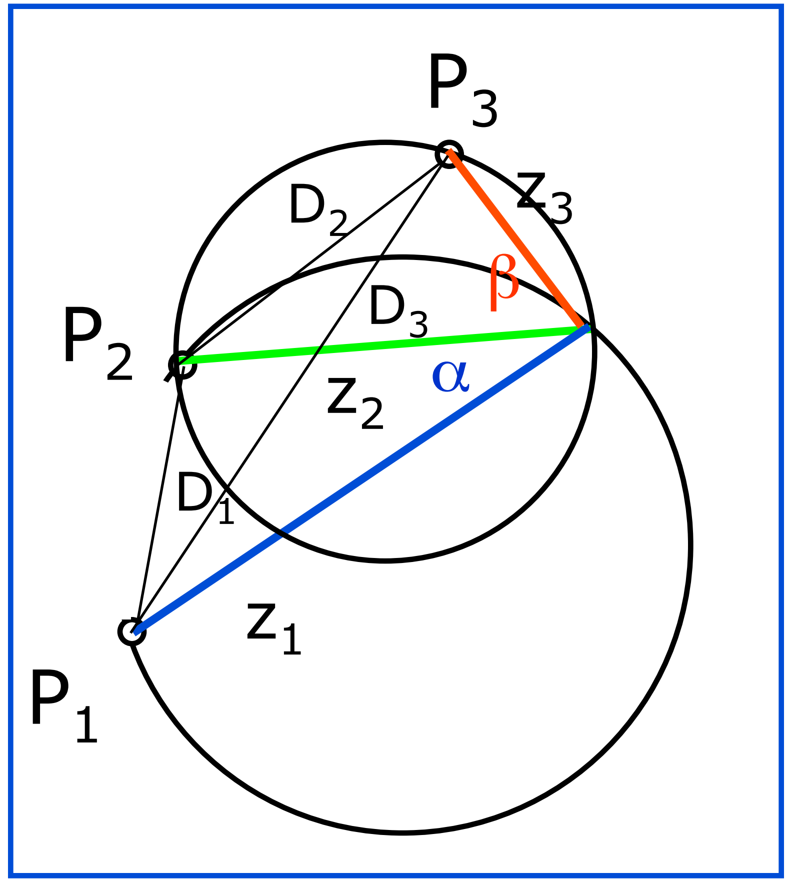


$P_1 = (0,0)$

$P_2 = (a,0)$

Bearings Only
No Uncertainty





Law of cosine

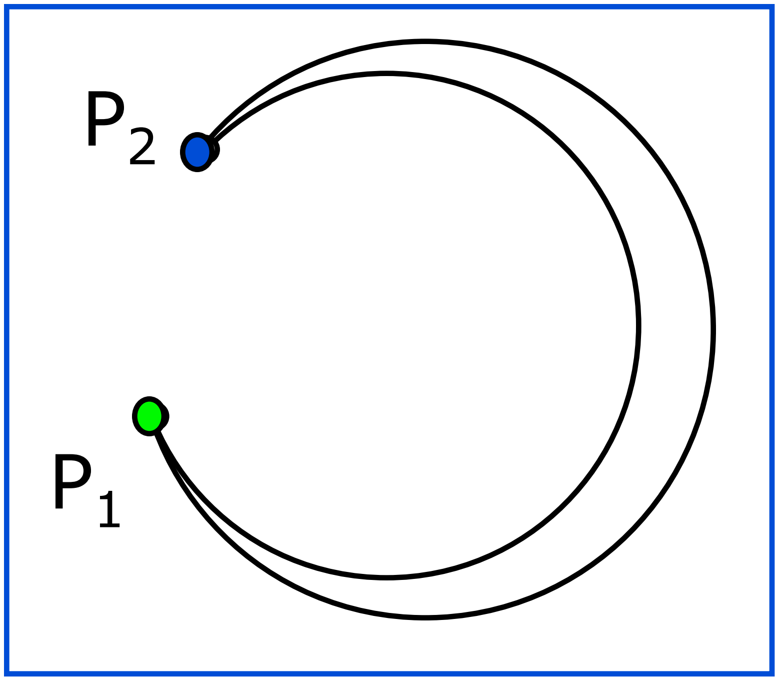
$$D_1^2 = z_1^2 + z_2^2 - 2\, z_1 z_2 \cos\alpha$$

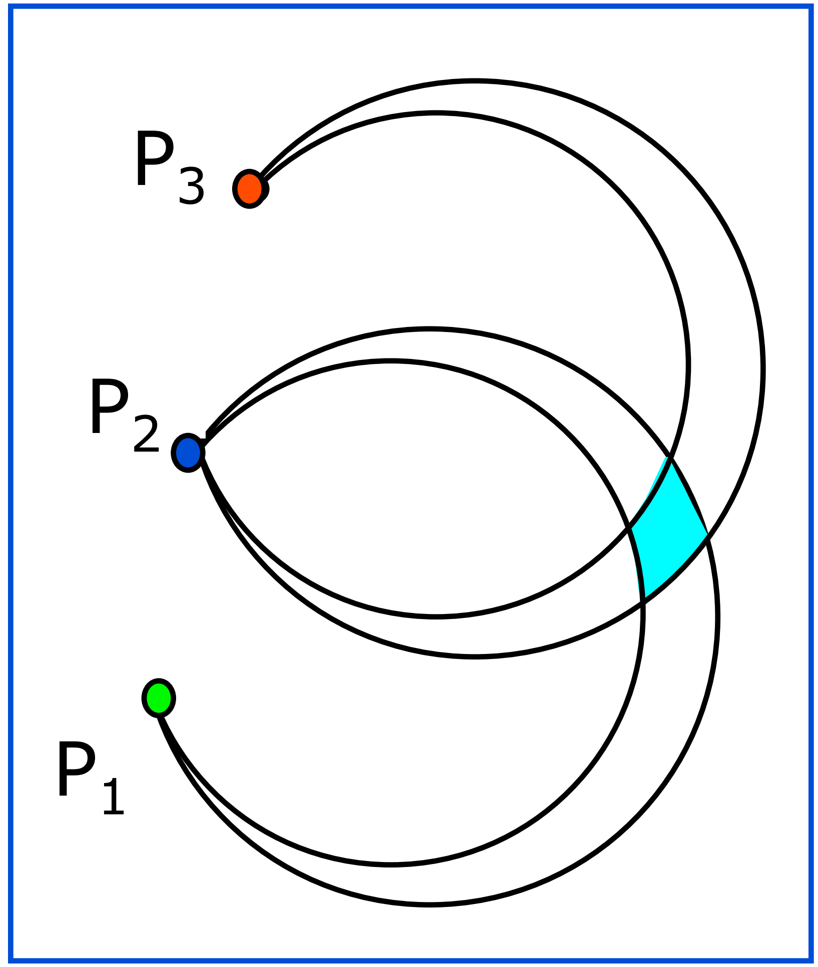
$$D_1^2 = z_1^2 + z_2^2 - 2\, z_1 z_2 \cos(\alpha)$$

$$D_2^2 = z_2^2 + z_3^2 - 2\, z_1 z_2 \cos(\beta)$$

$$D_3^2 = z_1^2 + z_3^2 - 2\, z_1 z_2 \cos(\alpha + \beta)$$

Bearings Only With Uncertainty





Most approaches attempt to find estimation mean.

Summary of Sensor Models

- Explicitly modeling uncertainty in sensing is key to robustness.
- In many cases, good models can be found by the following approach:
  1. Determine parametric model of noise free measurement.
  2. Analyze sources of noise.
  3. Add adequate noise to parameters (eventually mix in densities for noise).
  4. Learn (and verify) parameters by fitting model to data.
  5. Likelihood of measurement is given by "probabilistically comparing" the actual with the expected measurement.
- This holds for motion models as well.
- It is extremely important to be aware of the underlying assumptions!