Cognitive Robotics

04. Motion Models

AbdElMoniem Bayoumi, PhD

Fall 2022

Acknowledgment

- These slides have been created by Wolfram Burgard, Dieter Fox, Cyrill Stachniss and Maren Bennewitz

Where are we?

Bayes Filters: Framework

Given:

- Stream of observations *z* and action data *u:*

$$d_{1:t} = \{u_1, z_1 \ldots u_t, z_t\}$$

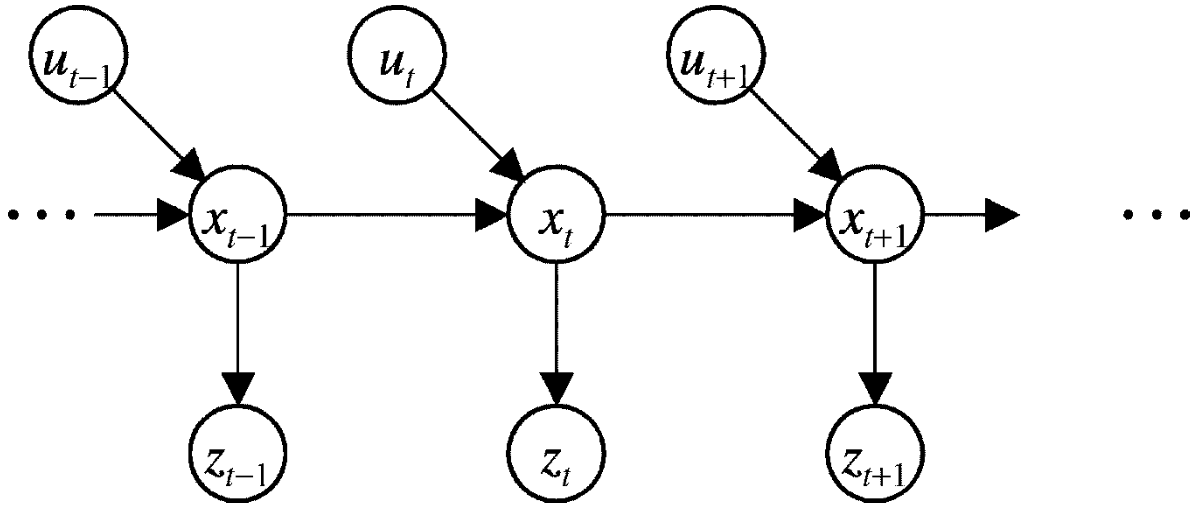
- Sensor model $P(z|x)$
- Action model $P(x|u,x')$
- Prior probability of the system state $P(x)$

Goal:

- Estimate the state X of a dynamical system
- Belief: posterior of the state

$$Bel(x_t) = P(x_t \mid u_1, z_1 \ldots, u_t, z_t)$$

Markov Assumption



$$p(z_t \mid x_{0:t}, z_{1:t-1}, u_{1:t}) = p(z_t \mid x_t)$$

$$p(x_t \mid x_{0:t-1}, z_{1:t-1}, u_{1:t}) = p(x_t \mid x_{t-1}, u_t)$$

Bayes Filters

z = observation
u = action
x = state

$$\boxed{Bel(x_t)} = P(x_t \mid u_1, z_1 \ldots, u_t, z_t)$$

Bayes $$= \eta \; P(z_t \mid x_t, u_1, z_1, \ldots, u_t) \; P(x_t \mid u_1, z_1, \ldots, u_t)$$

Markov $$= \eta \; P(z_t \mid x_t) \; P(x_t \mid u_1, z_1, \ldots, u_t)$$

Total prob. $$= \eta \; P(z_t \mid x_t) \int P(x_t \mid u_1, z_1, \ldots, u_t, x_{t-1}) \; P(x_{t-1} \mid u_1, z_1, \ldots, u_t) \; dx_{t-1}$$

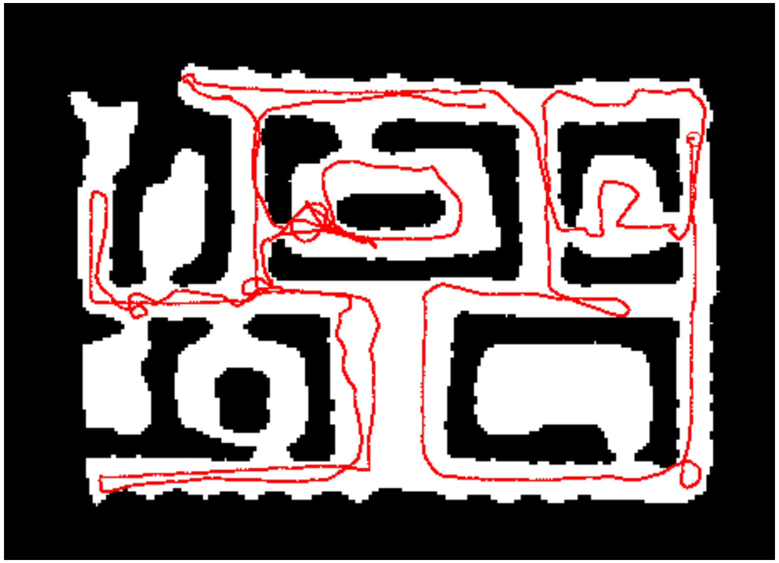
Markov $$= \eta \; P(z_t \mid x_t) \int P(x_t \mid u_t, x_{t-1}) \; P(x_{t-1} \mid u_1, z_1, \ldots, u_t) \; dx_{t-1}$$

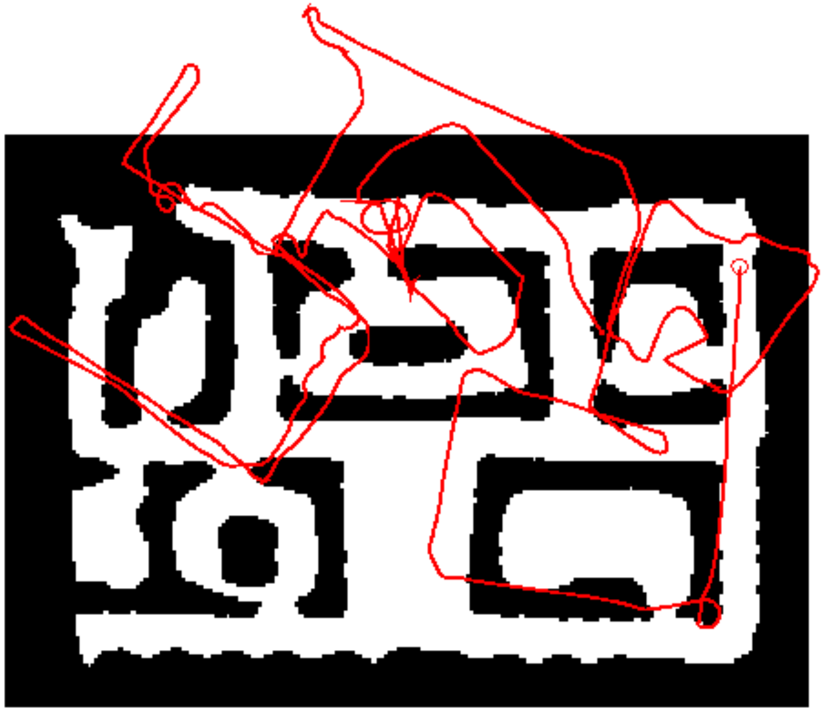
Markov $$= \eta P(z_t \mid x_t) \int P(x_t \mid u_t, x_{t-1}) \, P(x_{t-1} \mid u_1, z_1, \ldots, u_{t-1}, z_{t-1}) \; dx_{t-1}$$

$$\boxed{= \eta \; P(z_t \mid x_t) \int P(x_t \mid u_t, x_{t-1}) \; Bel(x_{t-1}) \; dx_{t-1}}$$

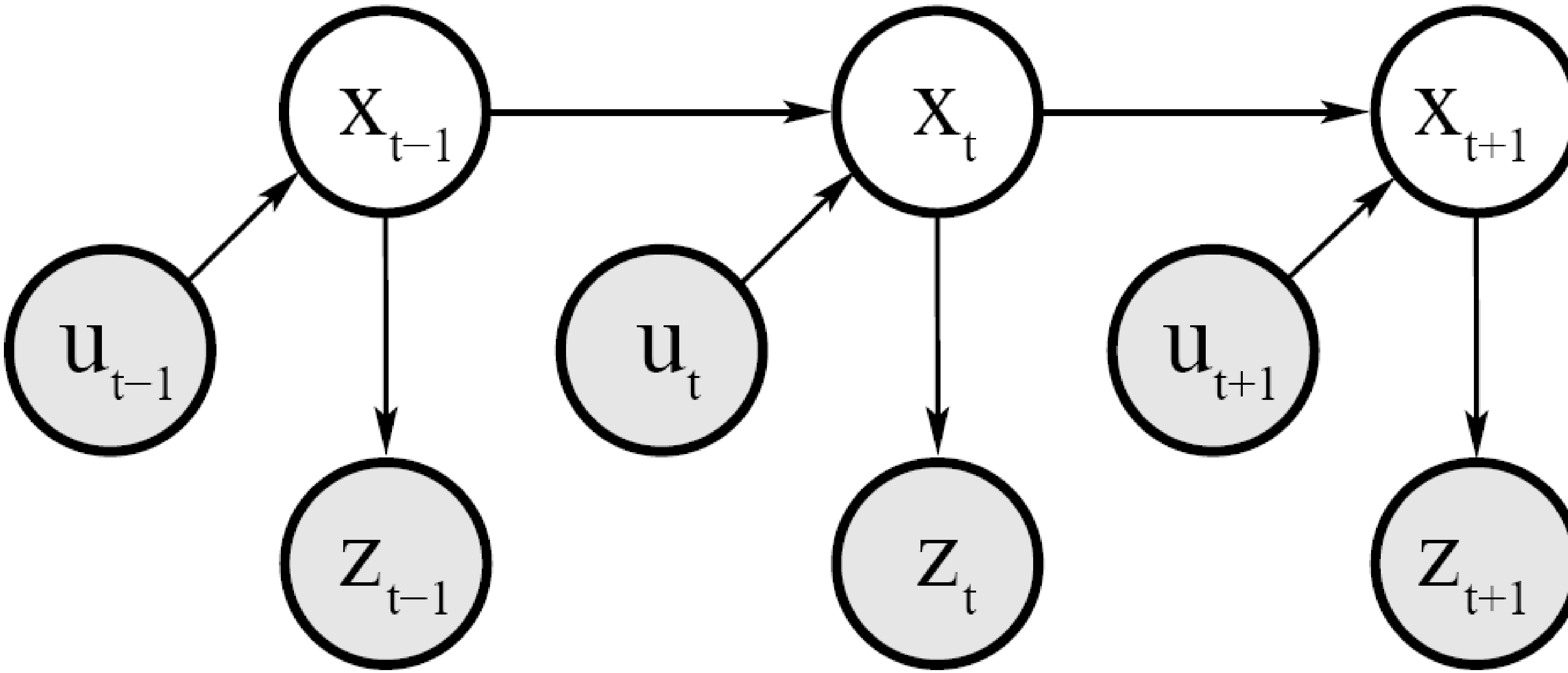
Robot Motion

- Robot motion is inherently uncertain
- The error accumulates
- How can we model this uncertainty?





Dynamic Bayesian Network
Controls, States, and Measurements

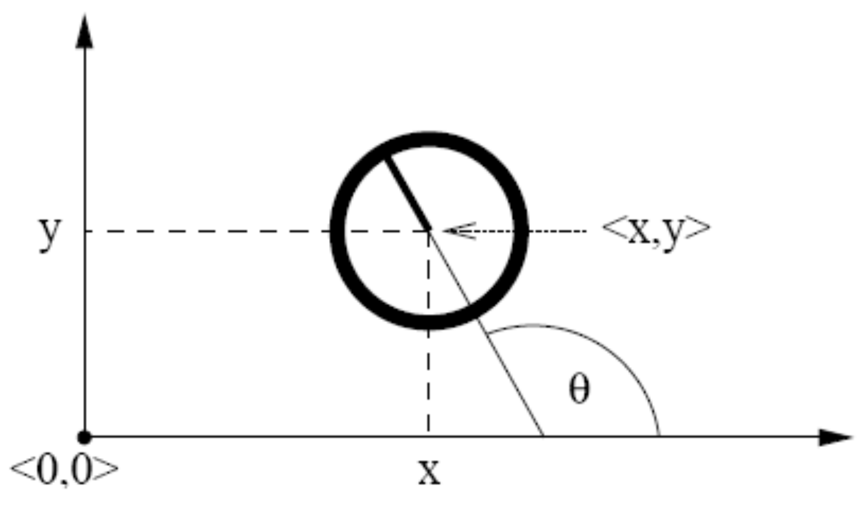


Probabilistic Motion Models

- To implement the Bayes Filter, we need the transition model $p(x_t \mid x_{t-1}, u_t)$
- The term $p(x_t \mid x_{t-1}, u_t)$ specifies a posterior probability, that action u_t carries the robot from x_{t-1} to x_t
- $p(x_t \mid x_{t-1}, u_t)$ can be modeled based on the motion equations and the uncertain outcome of the movements

Coordinate Systems

- Configuration of a typical wheeled robot in 3D can be described by six parameters
- 3D Cartesian coordinates plus the three Euler angles for roll, pitch, and yaw
- For simplicity, we consider robots operating on a planar surface throughout this section
- The state space of such systems is 3D: (x,y,θ)

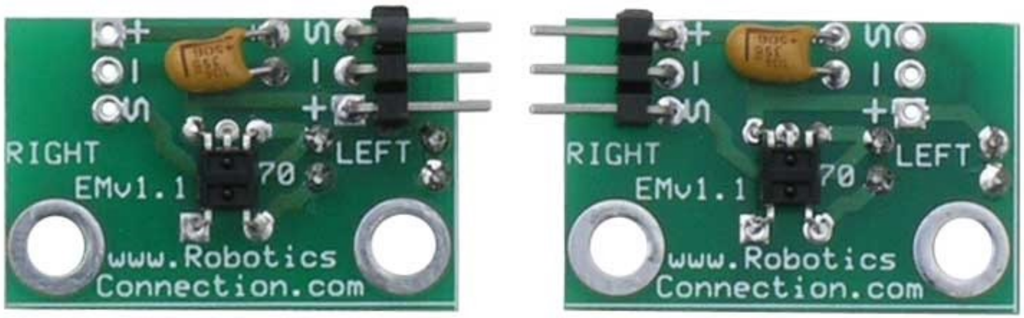


Typical Motion Models

- In practice, one mainly finds two types of motion models
- **Odometry-based:** when systems are equipped with wheel/joint encoders
- **Velocity-based** (dead reckoning): Calculate the new pose based on the velocities and the time elapsed

Example Wheel Encoders

- Measure how much a wheel turns and in which direction
- No absolute position measurement
- Increments can be integrated to pose



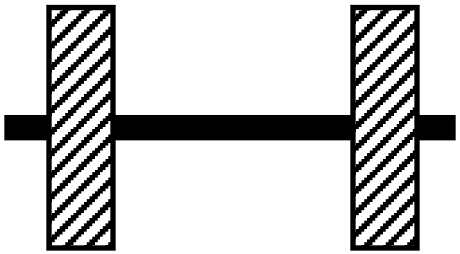
Black areas on transparent disk interrupt light falling on photo sensor

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

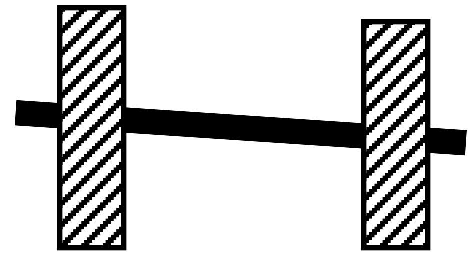
Dead Reckoning

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

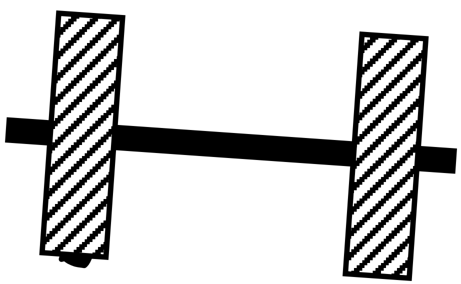
Some Reasons for Motion Errors of Wheeled Robots



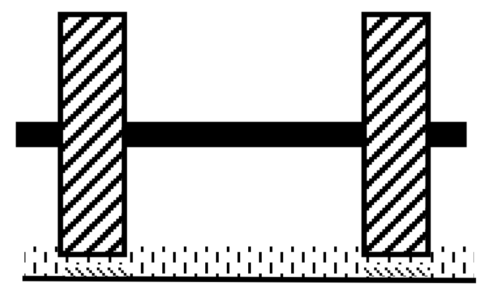
ideal case



different wheel diameters



bump

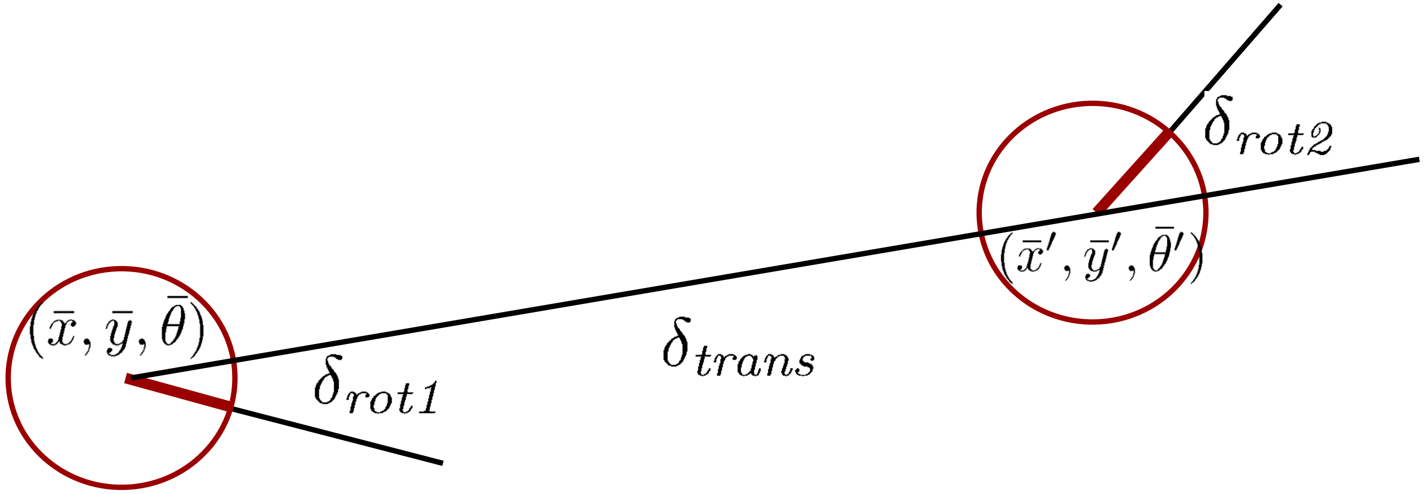


carpet

Odometry Model

- Motion from $(\bar{x}, \bar{y}, \bar{\theta})$ to $(\bar{x}', \bar{y}', \bar{\theta}')$
- Odometry information $u = (\delta_{rot1}, \delta_{trans}, \delta_{rot2})$

$$
\begin{aligned}
\delta_{trans} &= \sqrt{(\bar{x}' - \bar{x})^2 + (\bar{y}' - \bar{y})^2} \\
\delta_{rot1} &= \operatorname{atan2}(\bar{y}' - \bar{y}, \bar{x}' - \bar{x}) - \bar{\theta} \\
\delta_{rot2} &= \bar{\theta}' - \bar{\theta} - \delta_{rot1}
\end{aligned}
$$



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:

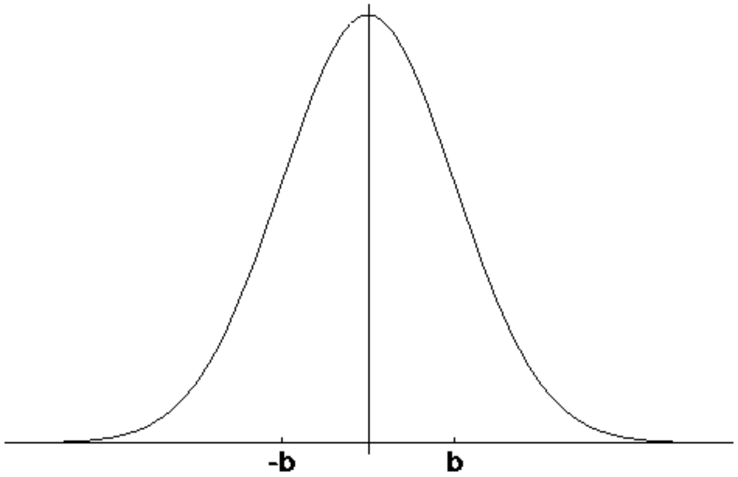
$$\hat{\delta}_{rot1} = \delta_{rot1} + \varepsilon_{\alpha_1 |\delta_{rot1}| + \alpha_2 |\delta_{trans}|}$$

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

$$\hat{\delta}_{rot2} = \delta_{rot2} + \varepsilon_{\alpha_5 |\delta_{rot2}| + \alpha_6 |\delta_{trans}|}$$

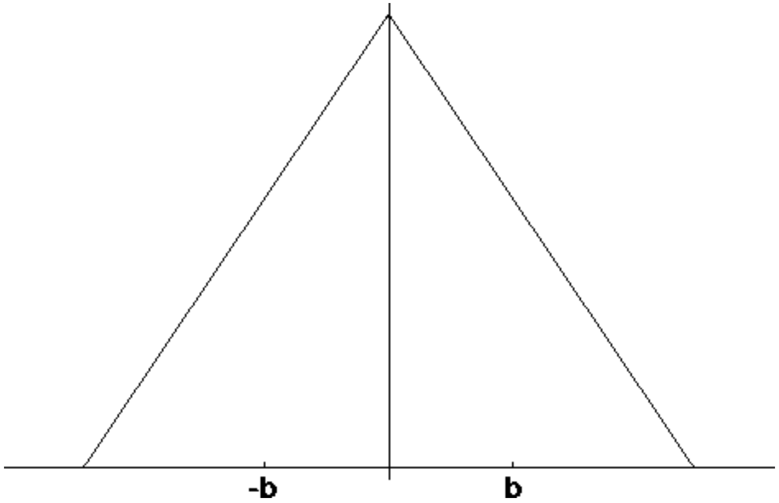
Typical Distributions for Probabilistic Motion Models

Normal distribution



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

Triangular distribution



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

Error Distribution with Zero Mean and Given Variance

- For a normal distribution

query point

1. Algorithm **prob_normal_distribution**(a,b):

std. deviation

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 Odometry

poses odometry

1. Algorithm **motion_model_odometry**$(\boxed{\boldsymbol{x}, \boldsymbol{x}'}, \boxed{\bar{\boldsymbol{x}}, \bar{\boldsymbol{x}}'})$
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. $\hat{\delta}_{rot1} = \text{atan2}(y'-y, x'-x) - \theta$
7. $\hat{\delta}_{rot2} = \theta' - \theta - \hat{\delta}_{rot1}$
8. $p_1 = \text{prob}(\delta_{\text{rot1}} - \hat{\delta}_{\text{rot1}}, \alpha_1 |\delta_{\text{rot1}}| + \alpha_2 \delta_{\text{trans}})$
9. $p_2 = \text{prob}(\delta_{\text{trans}} - \hat{\delta}_{\text{trans}}, \alpha_3 \delta_{\text{trans}} + \alpha_4 (|\delta_{\text{rot1}}| + |\delta_{\text{rot2}}|))$
10. $p_3 = \text{prob}(\delta_{\text{rot2}} - \hat{\delta}_{\text{rot2}}, \alpha_5 |\delta_{\text{rot2}}| + \alpha_6 \delta_{\text{trans}})$
11. return $p_1 \cdot p_2 \cdot p_3$

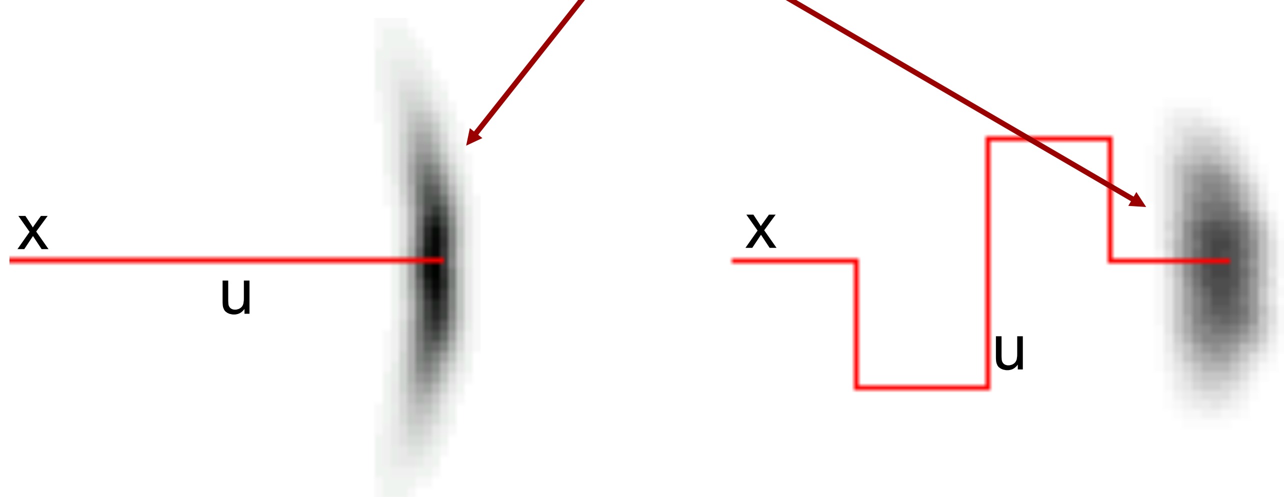
from odometry

values of interest (**x**,**x'**)

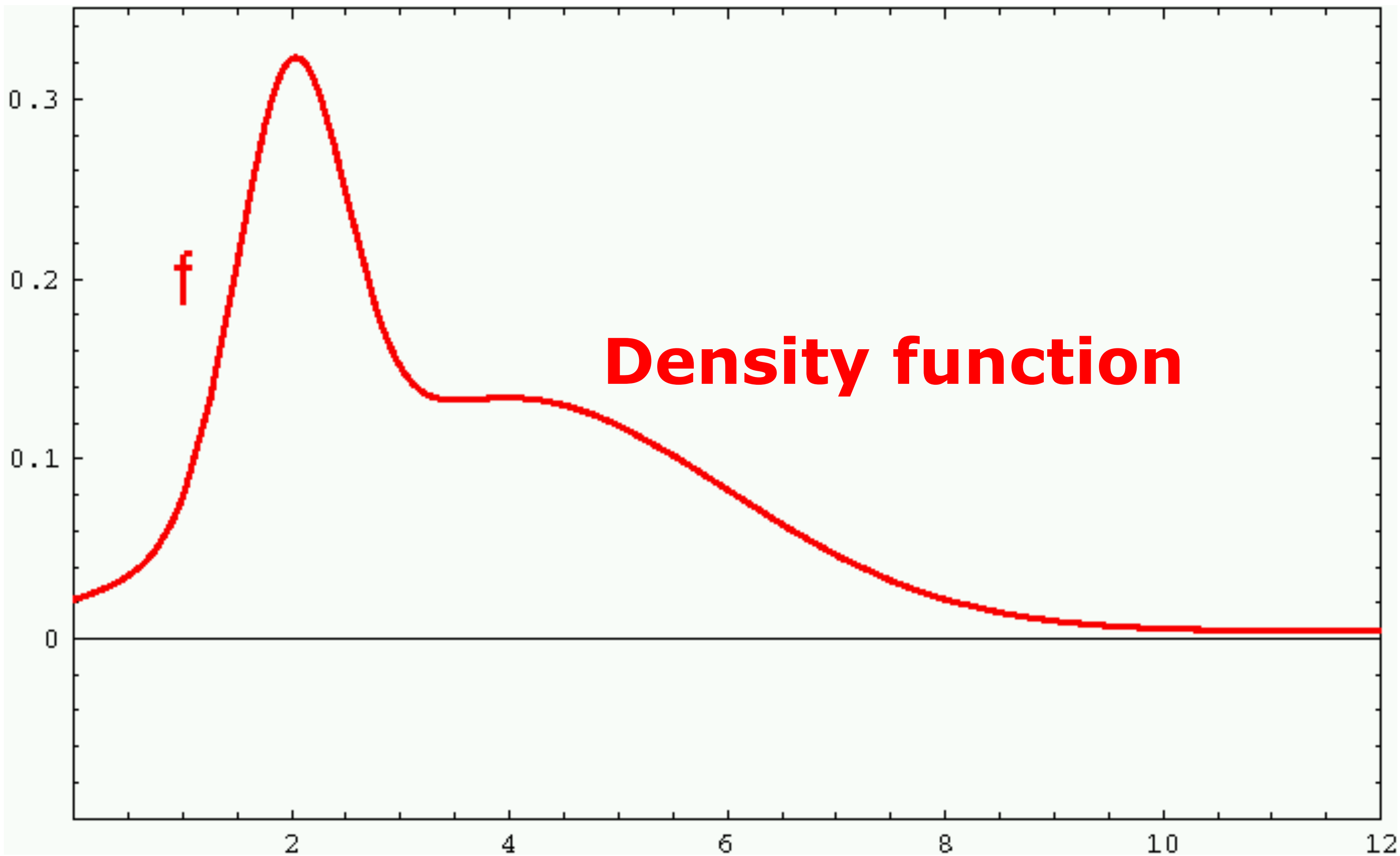
Application

- Repeated application of the motion model for short movements
- Typical banana-shaped distributions obtained for the 2D projection of the 3D posterior

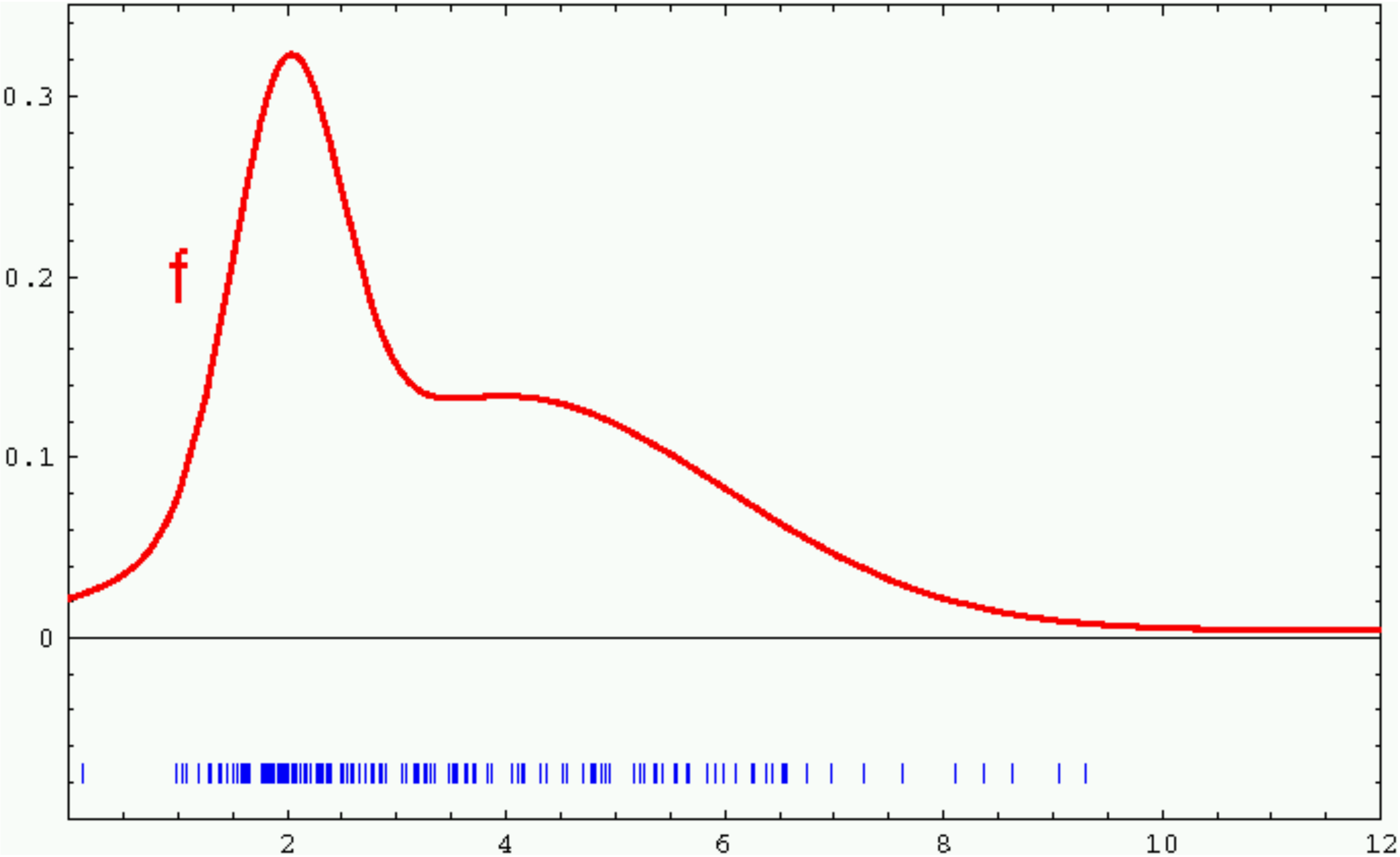
$$p(\boldsymbol{x}' | \boldsymbol{u}, \boldsymbol{x})$$



Sample-Based Density Representation



Sample-Based Density Representation

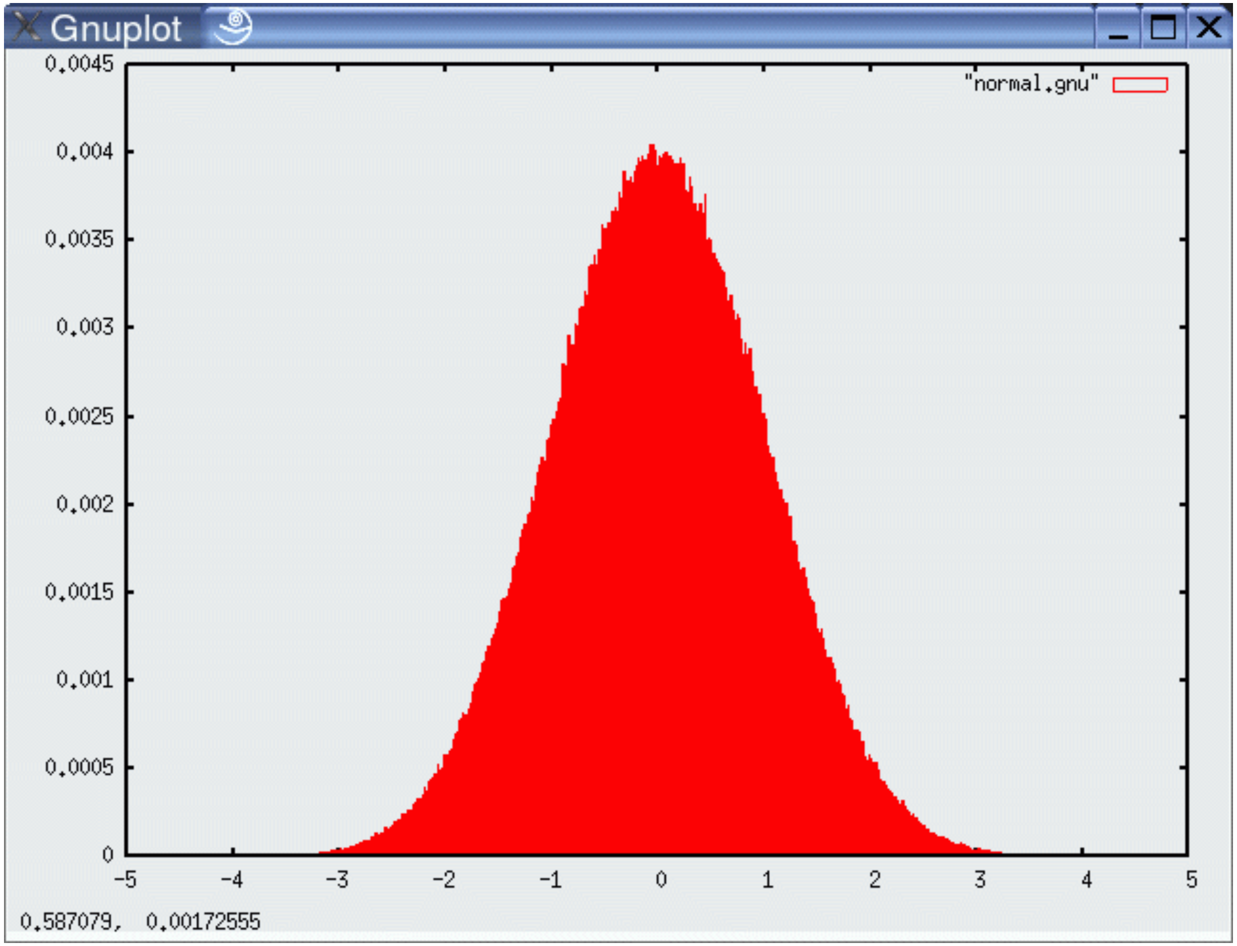


How to Sample from a Normal Distribution?

- Sampling from a normal distribution

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

Normally Distributed Samples



10^6 samples

How to Sample from Normal or Triangular Distributions?

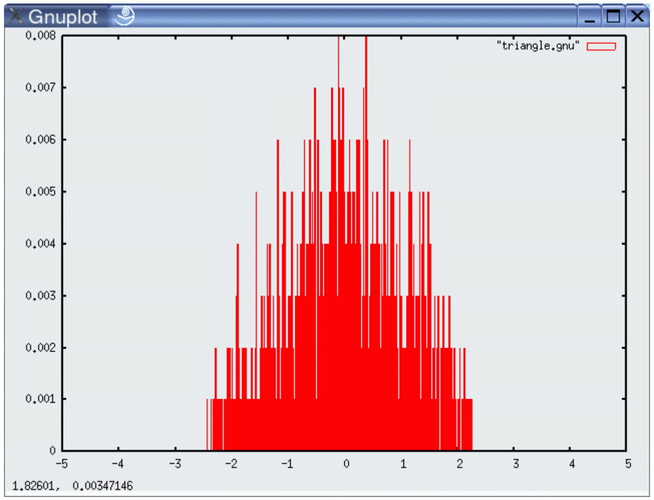
- Sampling from a normal distribution

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

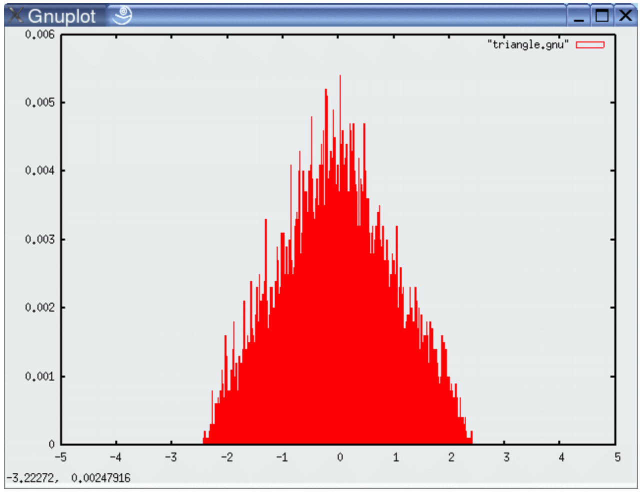
- Sampling from a triangular distribution

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

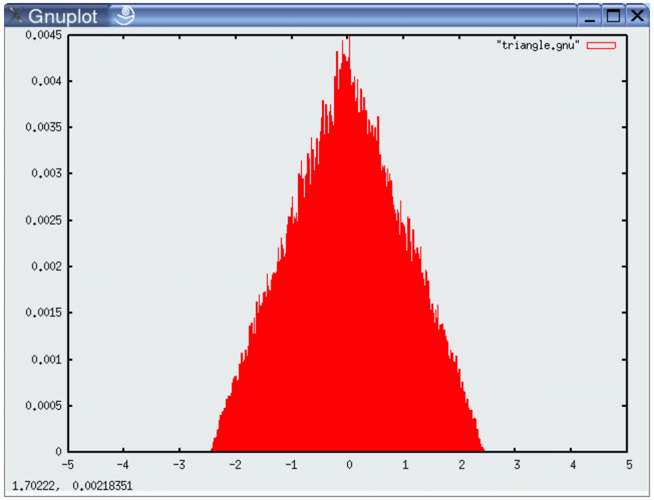
For Triangular Distribution



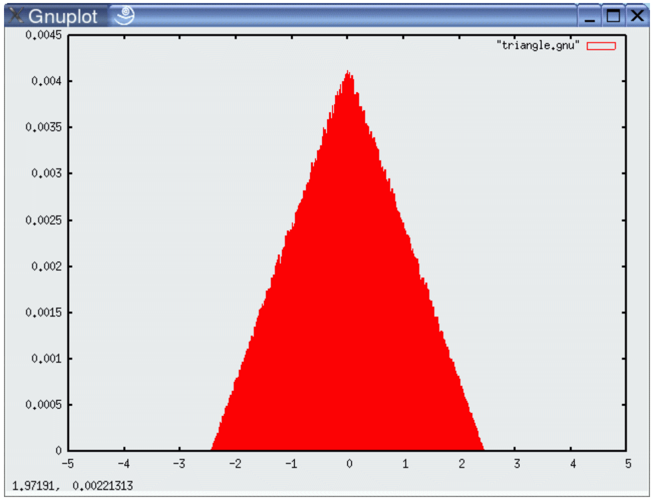
10^3 samples



10^4 samples

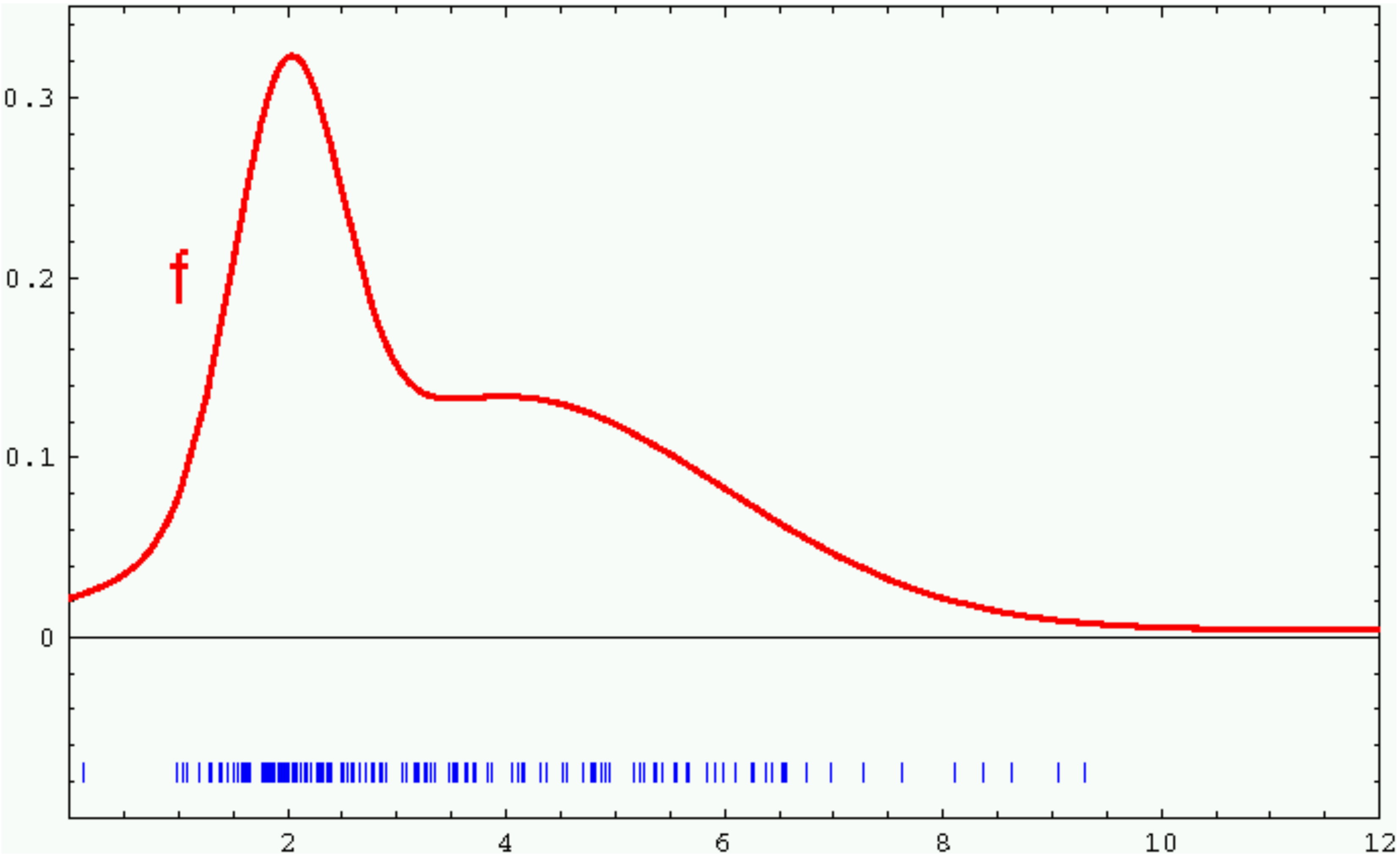


10^5 samples



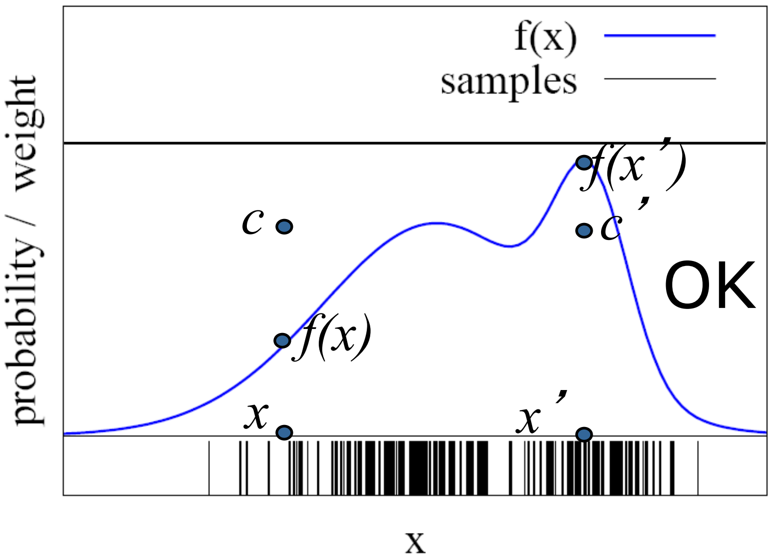
10^6 samples

How to Obtain Samples from Arbitrary Functions?



Rejection Sampling

- Sampling from arbitrary distributions
- Sample x from a uniform distribution from $[-b,b]$
- Sample c from $[0, max\, f]$
- if $f(x) > c$ keep the sample
  otherwise reject the sample



Rejection Sampling

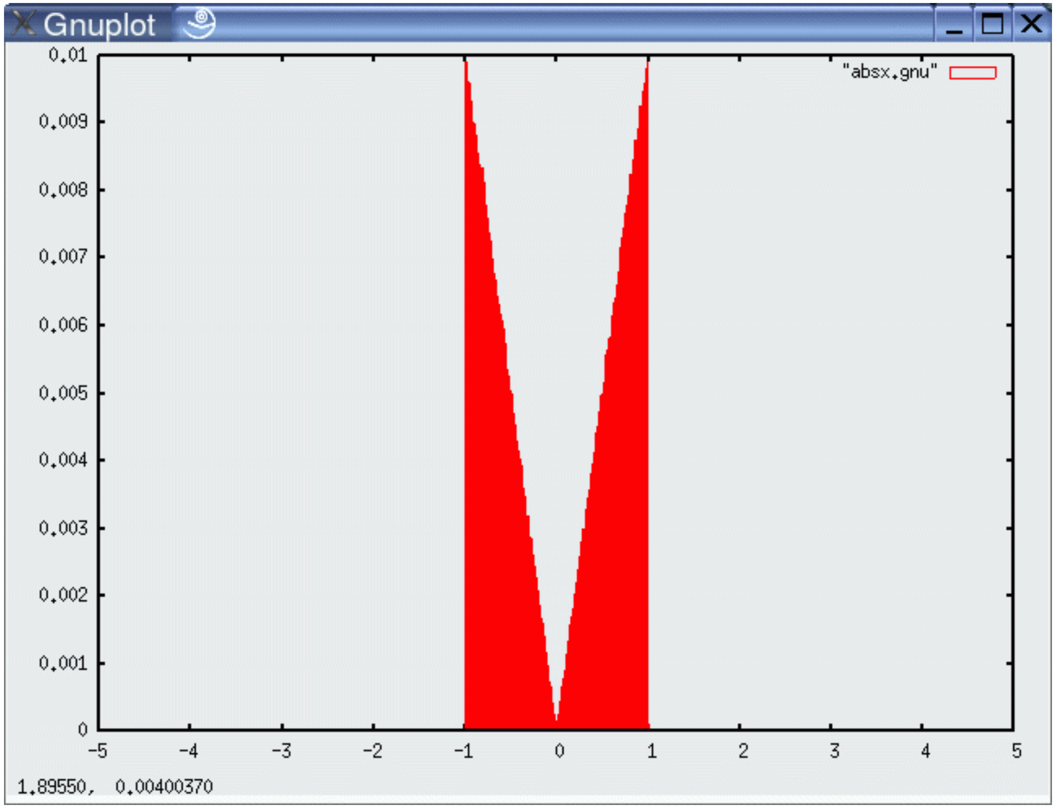
Sampling from arbitrary distributions:

1. Algorithm **sample_distribution**(f, b):
2. repeat
3. $x = \mathbf{rand}(-b, b)$
4. $y = \mathbf{rand}(0, \max\{f(x) \mid x \in [-b, b]\})$
5. until $y \leq f(x)$
6. 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):

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

old pose

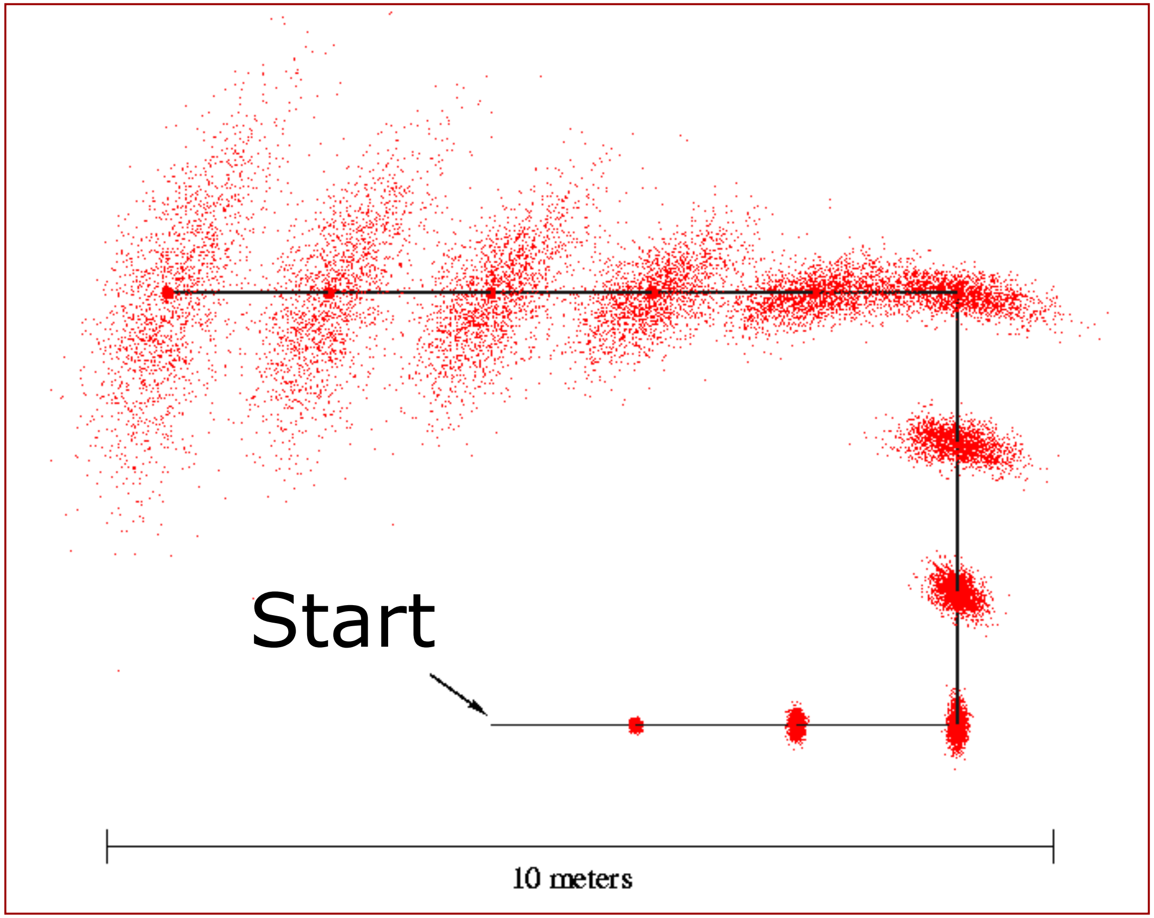
1. $\hat{\delta}_{rot1} = \delta_{rot1} + \text{sample}(\alpha_1 \,|\, \delta_{rot1} \,|+ \alpha_2 \; \delta_{trans})$
2. $\hat{\delta}_{trans} = \delta_{trans} + \text{sample}(\alpha_3 \; \delta_{trans} + \alpha_4 \; (|\, \delta_{rot1} \,| + |\, \delta_{rot2} \,|))$
3. $\hat{\delta}_{rot2} = \delta_{rot2} + \text{sample}(\alpha_5 \,|\, \delta_{rot2} \,|+ \alpha_6 \; \delta_{trans})$

sample_normal_distribution

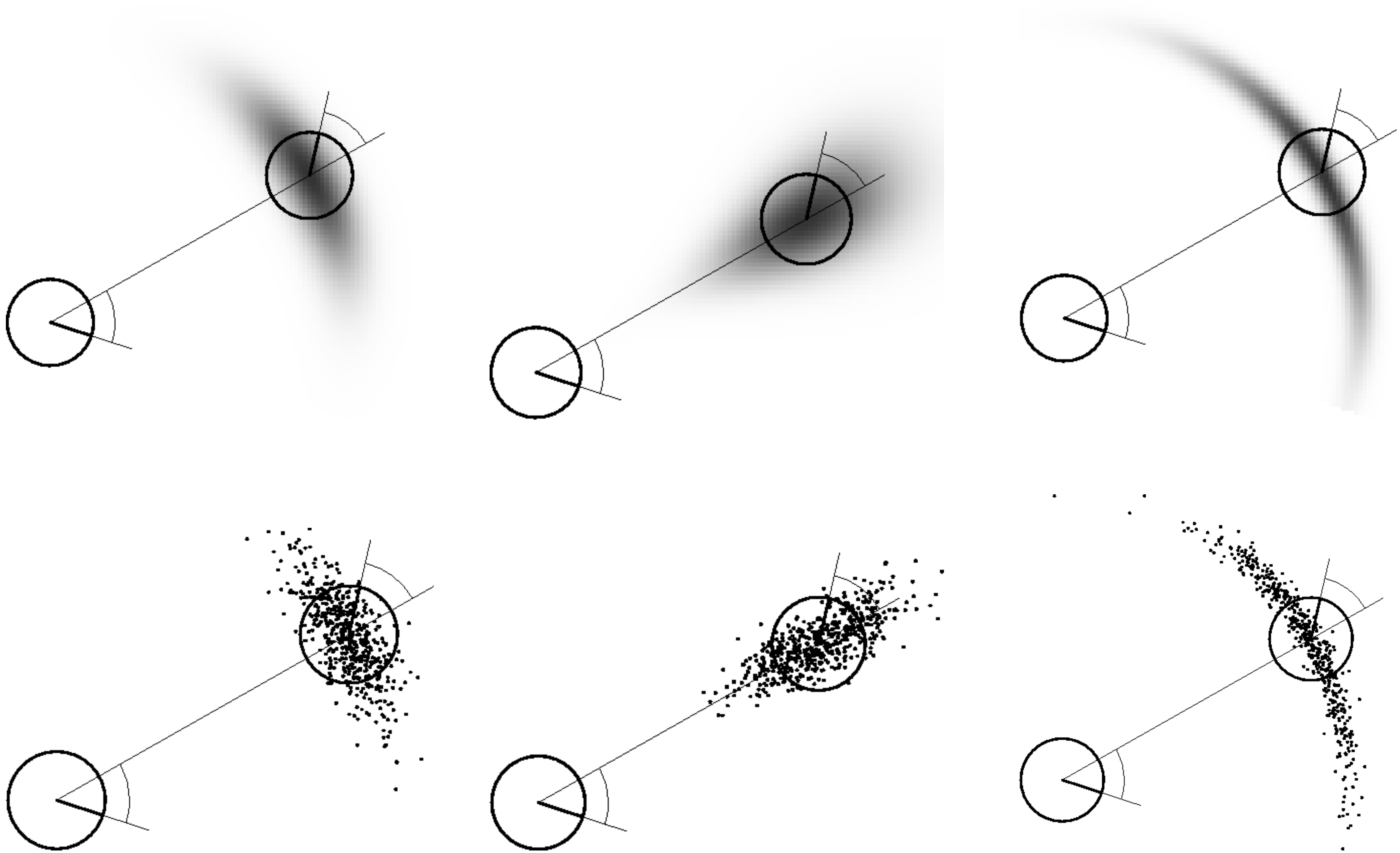
4. $x' = x + \hat{\delta}_{trans} \cos(\theta + \hat{\delta}_{rot1})$
5. $y' = y + \hat{\delta}_{trans} \sin(\theta + \hat{\delta}_{rot1})$
6. $\theta' = \theta + \hat{\delta}_{rot1} + \hat{\delta}_{rot2}$

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

Sampling from Motion Model



Examples (Odometry-Based)



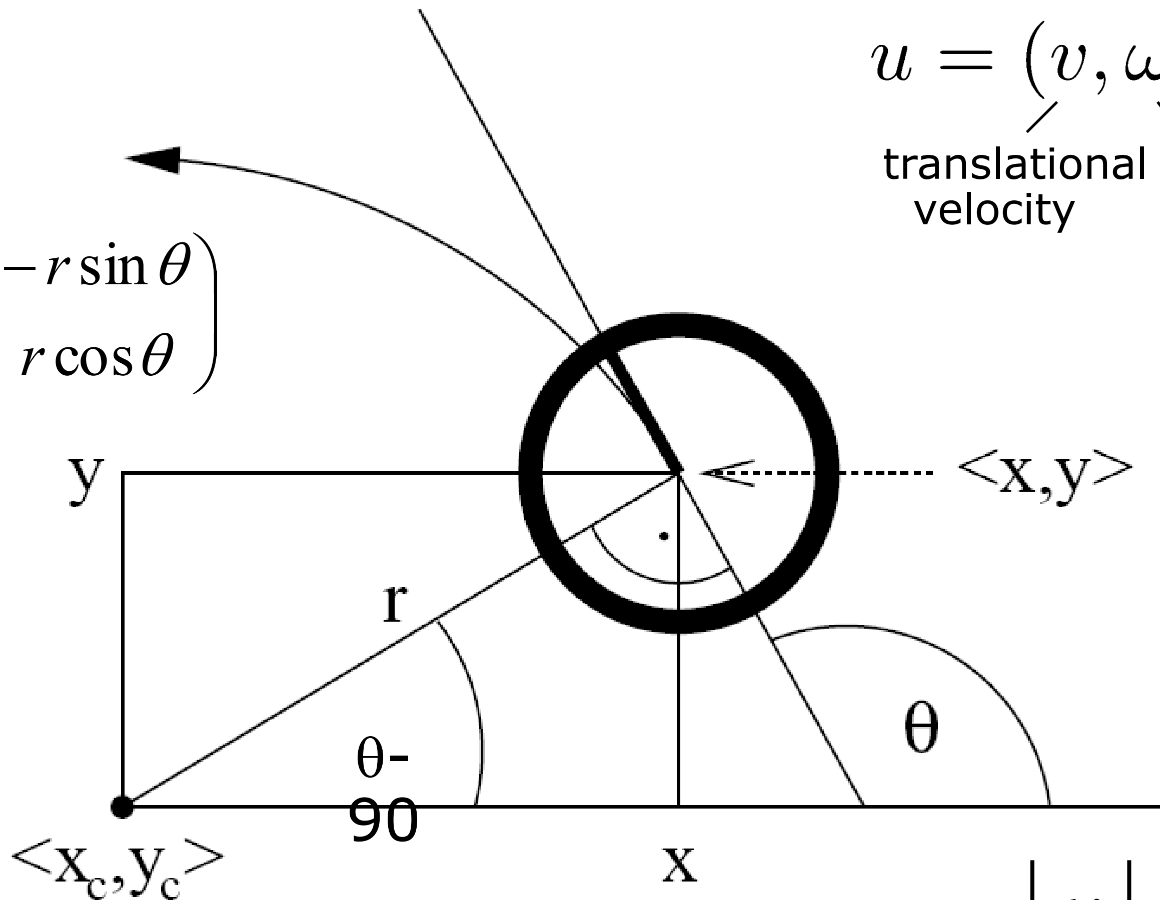
high translational noise

high rotational noise

Velocity-Based Model

$$u = (\underbrace{v}_{\text{translational velocity}}, \underbrace{\omega}_{\text{rotational velocity}})^T$$

$$\begin{pmatrix} x_c \\ y_c \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} -r\sin\theta \\ r\cos\theta \end{pmatrix}$$



instantaneous
center of rotation

$$r = \left|\frac{v}{\omega}\right|$$

Motion Equation

- Robot moves from (x, y, θ) to (x', y', θ')
- Velocity information $u = (v, \omega)$

$$\begin{pmatrix} x' \\ y' \\ \theta' \end{pmatrix} = \begin{pmatrix} x_c + \frac{v}{\omega} \sin(\theta + \omega\Delta t) \\ y_c - \frac{v}{\omega} \cos(\theta + \omega\Delta t) \\ \omega\Delta t \end{pmatrix}$$

$$\begin{pmatrix} x' \\ y' \\ \theta' \end{pmatrix} = \begin{pmatrix} x \\ y \\ \theta \end{pmatrix} + \begin{pmatrix} -\frac{v}{\omega} \sin\theta + \frac{v}{\omega} \sin(\theta + \omega\Delta t) \\ \frac{v}{\omega} \cos\theta - \frac{v}{\omega} \cos(\theta + \omega\Delta t) \\ \omega\Delta t \end{pmatrix}$$

Noise Model for the Velocity-Based Model

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

$$\hat{v} = v + \varepsilon_{\alpha_1|v|+\alpha_2|\omega|}$$

$$\hat{\omega} = \omega + \varepsilon_{\alpha_3|v|+\alpha_4|\omega|}$$

- Discussion: What is the disadvantage of this noise model?

Noise Model for the Velocity-Based Model

- The $(\hat{v}, \hat{\omega})$-circle constrains the final orientation (2D manifold in a 3D space)

- Better approach:

$$\hat{v} = v + \varepsilon_{\alpha_1|v|+\alpha_2|\omega|}$$

$$\hat{\omega} = \omega + \varepsilon_{\alpha_3|v|+\alpha_4|\omega|}$$

$$\hat{\gamma} = \varepsilon_{\alpha_5|v|+\alpha_6|\omega|}$$

↑

term to account for the final rotation

Motion Including 3rd Parameter

$$x' = x - \frac{\hat{v}}{\hat{\omega}} \sin\theta + \frac{\hat{v}}{\hat{\omega}}\ sin(\theta + \hat{\omega}\Delta t)$$

$$y' = y + \frac{\hat{v}}{\hat{\omega}} \cos\theta - \frac{\hat{v}}{\hat{\omega}}\ cos(\theta + \hat{\omega}\Delta t)$$

$$\theta' = \theta + \hat{\omega}\Delta t + \hat{\gamma}\Delta t$$

to account for the final rotation

Equation for the Velocity Model

$$x_{t-1} = (x, y, \theta)^T$$
$$x_t = (x', y', \theta')^T$$

some constant

Center of circle:

$$\begin{pmatrix} x^* \\ y^* \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} -\lambda \sin\theta \\ \lambda \cos\theta \end{pmatrix} = \begin{pmatrix} \frac{x+x'}{2} + \mu(y - y') \\ \frac{y+y'}{2} + \mu(x' - x) \end{pmatrix}$$

some constant (the center of the circle lies on a ray half way between x and x' and is orthogonal to the line between x and x')

Equation for the Velocity Model

$$x_{t-1} = (x, y, \theta)^T$$
$$x_t = (x', y', \theta')^T$$

some constant

Center of circle:

$$\begin{pmatrix} x^* \\ y^* \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} -\lambda \sin\theta \\ \lambda \cos\theta \end{pmatrix} = \begin{pmatrix} \frac{x+x'}{2} + \mu(y - y') \\ \frac{y+y'}{2} + \mu(x' - x) \end{pmatrix}$$

Allows us to solve the equations to:

$$\mu = \frac{1}{2} \frac{(x - x')\cos\theta + (y - y')\sin\theta}{(y - y')\cos\theta - (x - x')\sin\theta}$$

Sampling from Velocity Model

Algorithm sample_motion_model_velocity(u_t, x_{t-1})**:**

$u = (v, \omega)$

$\hat{v} = v + \textbf{sample}(\alpha_1 v^2 + \alpha_2 \omega^2)$

$\hat{\omega} = \omega + \textbf{sample}(\alpha_3 v^2 + \alpha_4 \omega^2)$

$\hat{\gamma} = \textbf{sample}(\alpha_5 v^2 + \alpha_6 \omega^2)$

$x' = x - \frac{\hat{v}}{\hat{\omega}} \sin\theta + \frac{\hat{v}}{\hat{\omega}} \ sin(\theta + \hat{\omega}\Delta t)$

$y' = y + \frac{\hat{v}}{\hat{\omega}} \cos\theta - \frac{\hat{v}}{\hat{\omega}} \ cos(\theta + \hat{\omega}\Delta t)$

$\theta' = \theta + \hat{\omega}\Delta t + \hat{\gamma}\Delta t$

return $x_t = (x', y', \theta')^T$

Posterior Probability for Velocity Model

Algorithm motion_model_velocity(x_t, u_t, x_{t-1})**:**

$u = (v, \omega)$

$$\mu = \frac{1}{2}\,\frac{(x - x')\cos\theta + (y - y')\sin\theta}{(y - y')\cos\theta - (x - x')\sin\theta}$$

$$x^* = \frac{x + x'}{2} + \mu(y - y')$$

$$y^* = \frac{y + y'}{2} + \mu(x' - x)$$

// instantaneous center of rotation

$$r^* = \sqrt{(x - x^*)^2 + (y - y^*)^2}$$

// distance to center

$$\Delta\theta = \operatorname{atan2}(y' - y^*, x' - x^*) - \operatorname{atan2}(y - y^*, x - x^*)$$

$$\hat{v} = \frac{\Delta\theta}{\Delta t}\, r^*$$

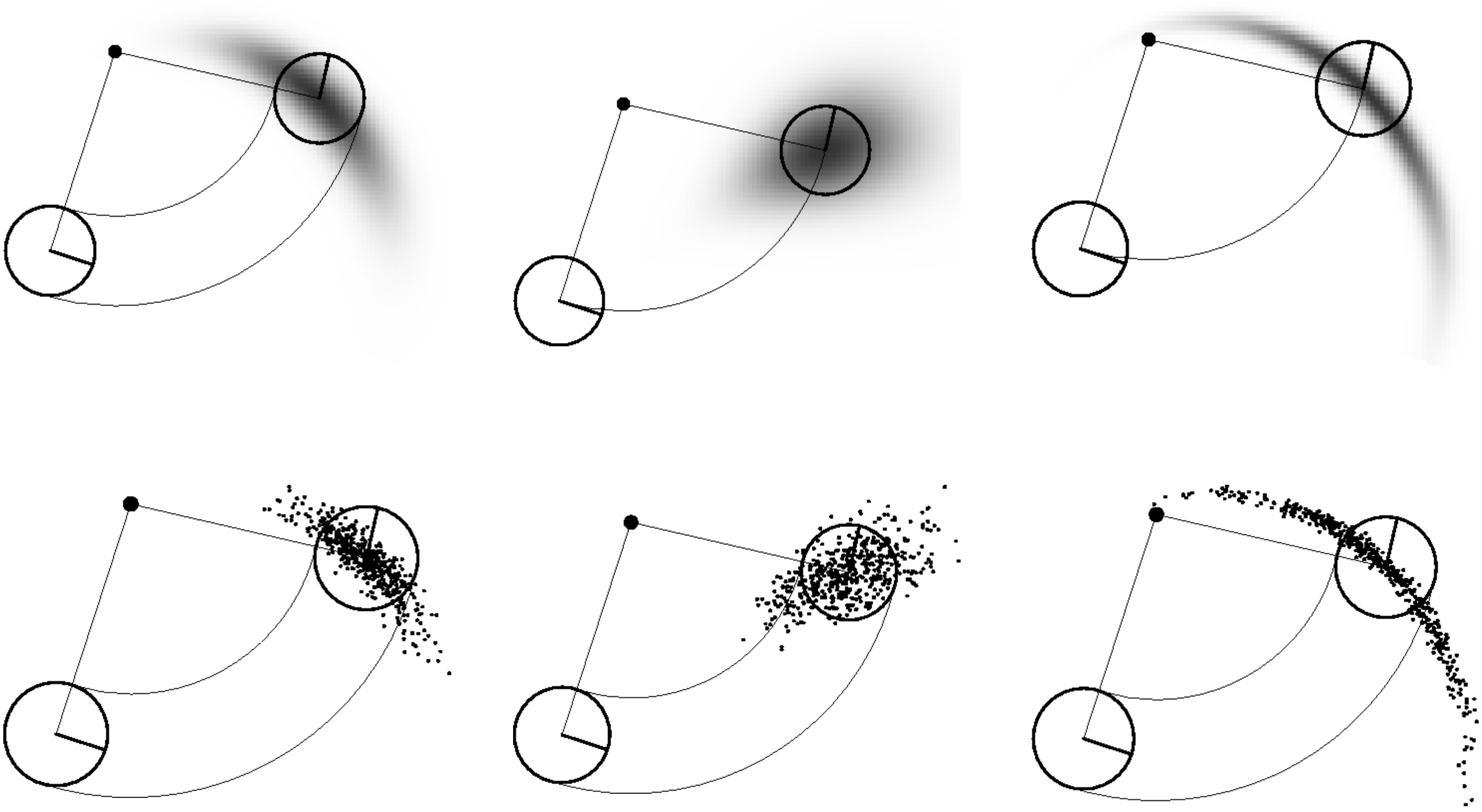
$$\hat{\omega} = \frac{\Delta\theta}{\Delta t}$$

$$\hat{\gamma} = \frac{\theta' - \theta}{\Delta t} - \hat{\omega}$$

// compute motion error (deviation from control u)

return $\mathbf{prob}(v - \hat{v}, \alpha_1 v^2 + \alpha_2 \omega^2) \cdot \mathbf{prob}(\omega - \hat{\omega}, \alpha_3 v^2 + \alpha_4 \omega^2) \cdot \mathbf{prob}(\hat{\gamma}, \alpha_5 v^2 + \alpha_6 \omega^2)$

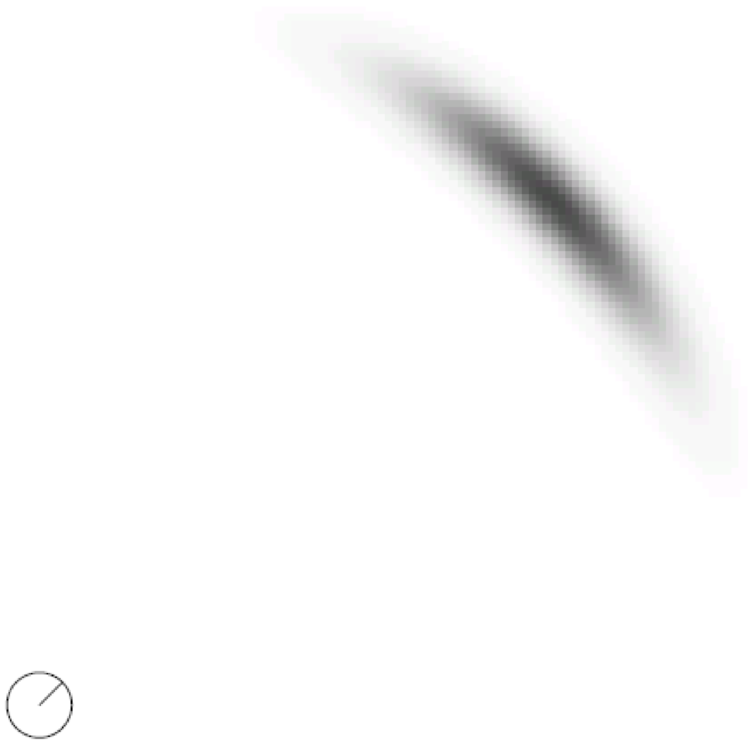
Examples (Velocity-Based)

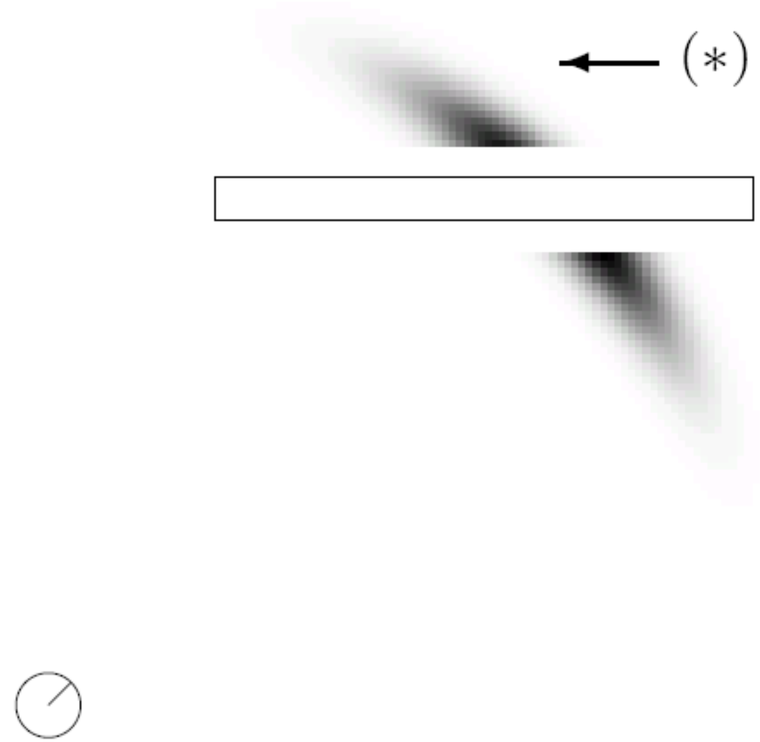


high translational noise

high rotational noise

Map-Consistent Motion Model





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

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

Summary

- We discussed motion models for odometry-based and velocity-based systems
- Calculations are done in fixed time intervals
- We discussed ways to calculate the posterior probability $p(x' | x, u)$
- We also described how to sample from $p(x' | x, u)$
- In practice, the parameters of the motion models have to be learned

Acknowledgment

- These slides have been created by Wolfram Burgard, Dieter Fox, Cyrill Stachniss and Maren Bennewitz