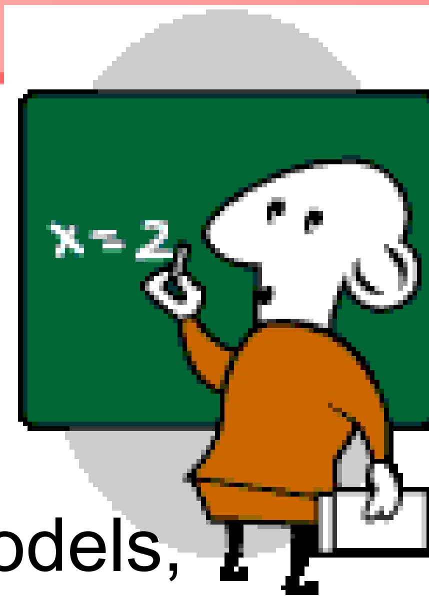
Probabilistic Models of Robot Motion

Department of Electrical and Electronics Engineering
Dr. Afşar Saranlı

Lecture slides heavily use material from the textbook and
Sebastian Thrun, Lecture Slides; http://www.probabilistic-robotics.org/

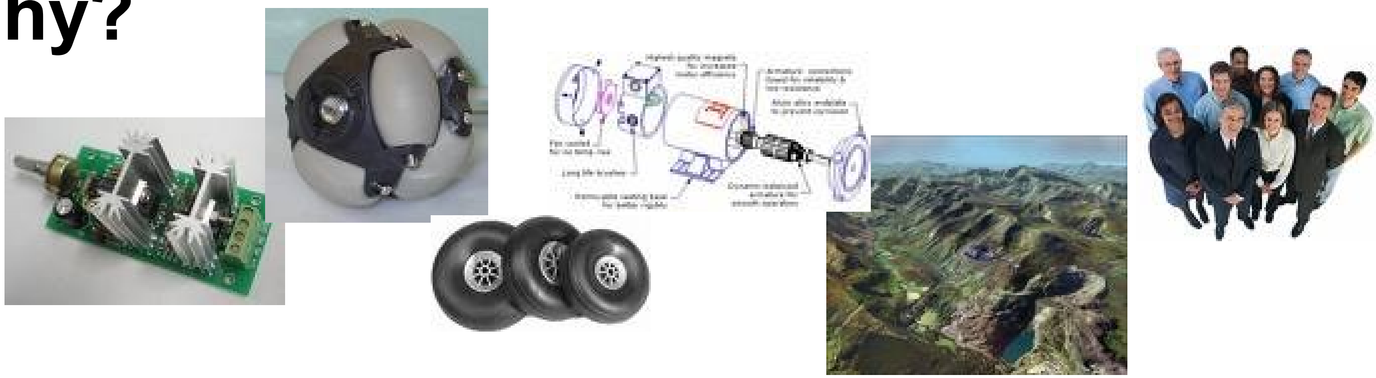
What we will discuss



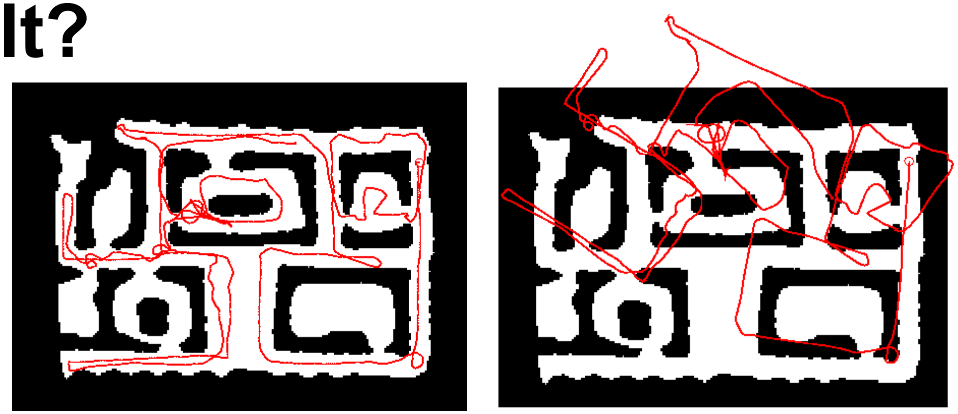
- Realize the inherent uncertainty in the motion of a robot,
- Discuss general principles of motion models,
- Examine "*odometry motion model*" for 2D robots,
- Examine "*velocity motion model*" for 2D robots,
- Discuss generalization to "robots with dynamics" and other devices in 3D that we may call "robots"

Uncertainty in Robot Motion

- Robot motion is inherently uncertain.
- **Why?**



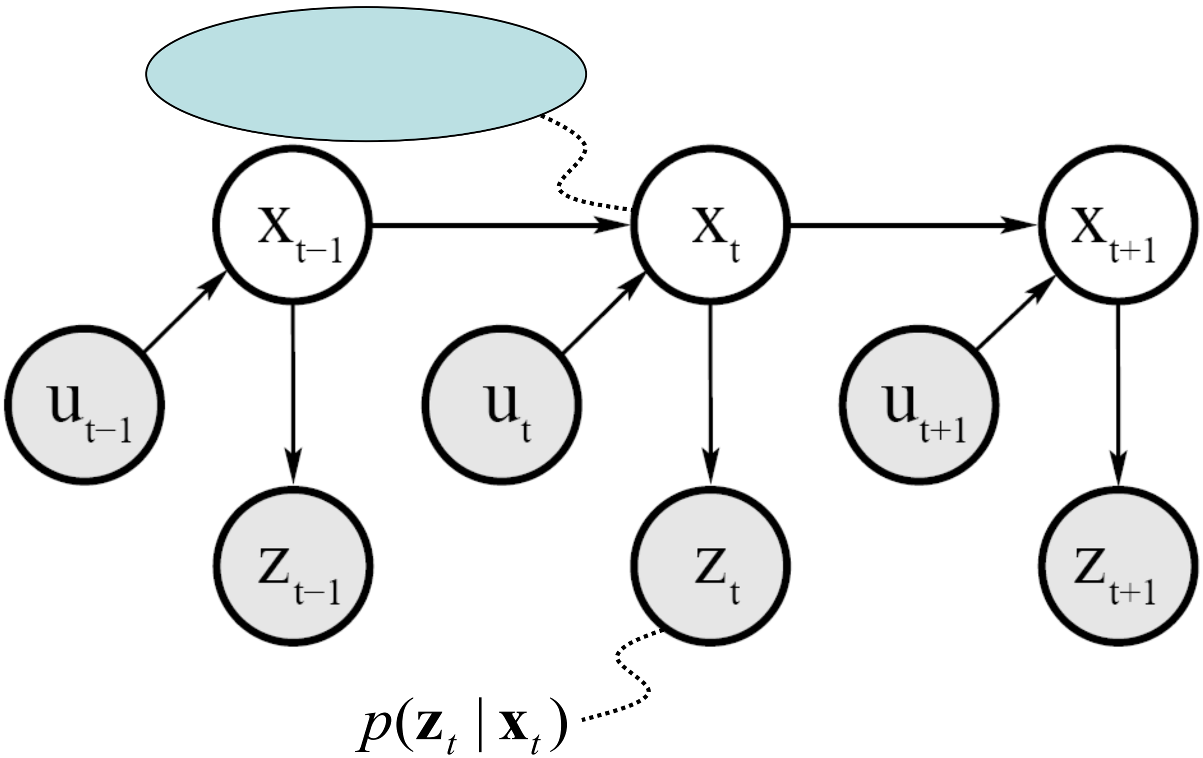
- **The result?**



- **Question:** How can we model this uncertainty?

Dynamic Bayes Network

- **Remember the Markovian interaction of random variables:**

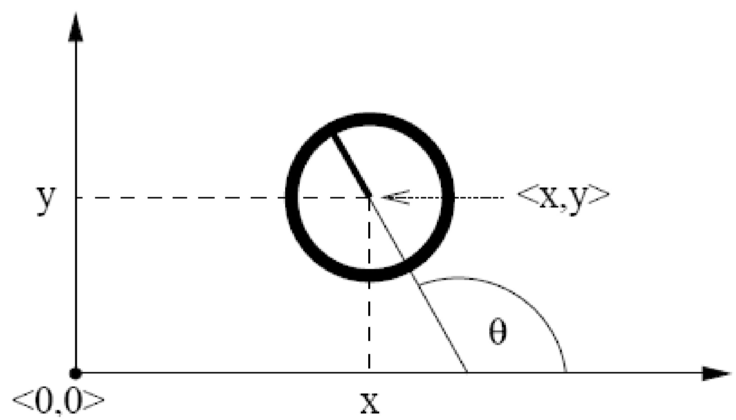


Probabilistic Model of Motion

- To implement the Bayes Filter, we need the state 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 (pdf), that action u carries the robot from x_{t-1} to x_t.
- In this section we will specify, how $p(x_t \mid x_{t-1}, u_t)$ can be modeled based on the planar (2D) kinematic equations of motion.

Coordinate Systems

- In general (3D space) the configuration of a solid body robot can be described by six parameters.
- 3 cartesian coordinates (position) plus 3 Euler angles yaw, pitch, roll.
- Throughout this section, we consider robots operating on a planar surface.
- We will also ignore "dynamics", i.e. speed variables.
- The *state space* of such system is three-dimensional: (x,y,θ) and is known as the *robot pose*.



Typical Motion Models?

- In general, there are no typical motion models!
- E.g., a helicopter UAV robot may need an *entire book* just for the motion model!!
- However, due to robustness of probabilistic framework, crude models may work in practice,
- E.g., simple motion models are used in *Radar Target Tracking* to track actual planes.
- In this course: Planar robots with wheels
  (As an example to suggest the method for others)

Typical Motion Models?

- For our special example case: Two popular types of motion models:

Odometry-based / Velocity-based

- Odometry-based models are preferred when systems are equipped with *wheel encoders*.
- *They use <u>Encoder counters</u> for wheel rotation to estimate pose change in a given short time interval.*
- Velocity-based models have to be applied when no wheel encoders are given.
- *They calculate the new pose based on the motor velocities and the elapsed time.*

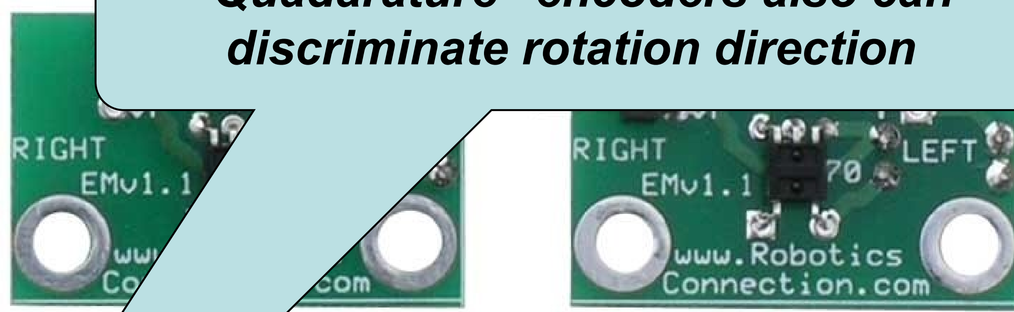
Other Possibilities?

- Other possibilities exist:
- Treat some sensors as “better motion models”
- E.g. Inertial Sensing (accelerometers and gyroscopes)

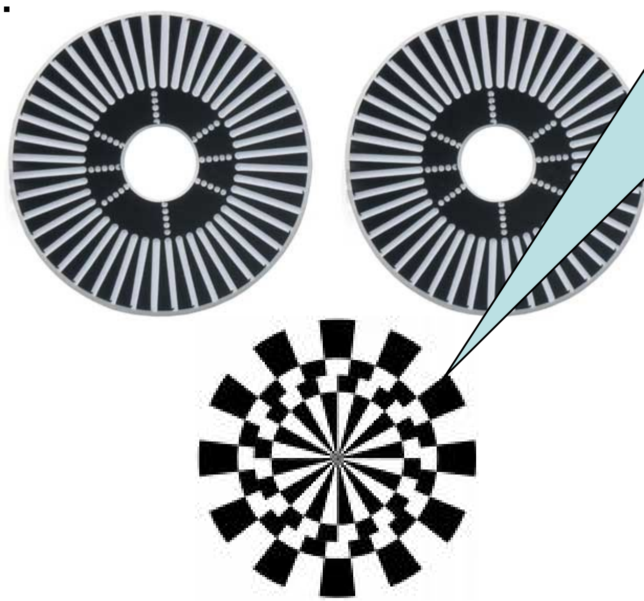
- *Interesting question:* Using a sensor either as part of the motion model, or use it as part of the sensor model? Which one to choose?
- *Example:* In so-called “Visual-Inetial Navigation”, IMU is used as a “motion model” while camera frames are used as measurements using a “camera measurement model”.

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.



"Quadarature" encoders also can discriminate rotation direction



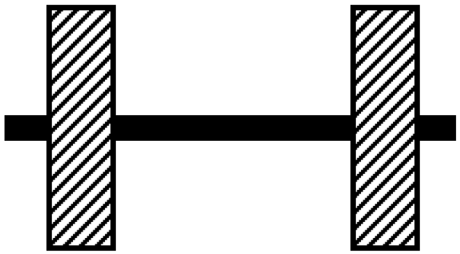
These disks are manufactured out of high quality laminated color plastic to offer a very crisp black to white transition. This enables a wheel encoder sensor to easily see the transitions.

Source: http://www.active-robots.com/, http://mitros.org

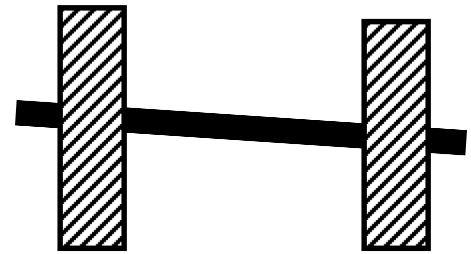
Dead Reckoning

- Velocity motion model is also called "*dead reckoning*",
- *Deterministic Mathematical procedure for crudely computing the current pose of a vehicle.*
- Achieved by calculating the current pose of the vehicle based on the previous pose, motor command (velocities) and the time elapsed.
- Supplemented by a "noise model"
- We will see how it works…
- Term sometimes also used for deterministic (integration) methods using wheel encoders…

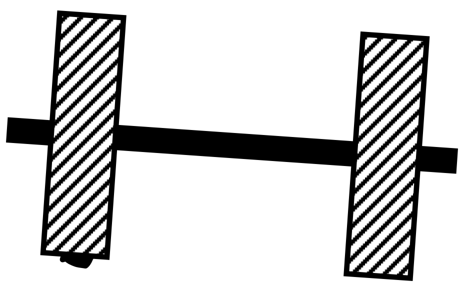
Some Reasons for Odometry Errors



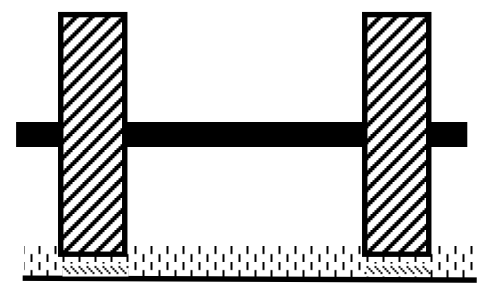
ideal case



different wheel diameters



bump



slippage

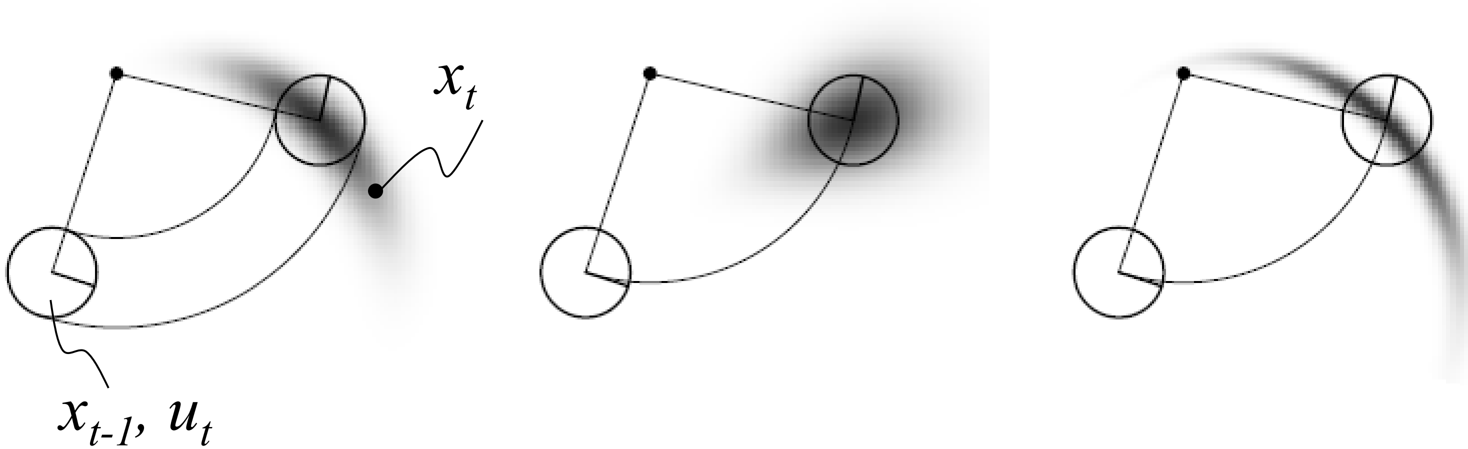
and possibly more …

Roadmap for Deriving A Motion Model

- Two approaches:
  - Calculate, analytically, the $p(x_t \mid x_{t-1}, u_t)$ distribution, (needed for Gaussian approaches – theoretically more complex but numerically efficient)
  - Draw samples from the $p(x_t \mid x_{t-1}, u_t)$ distribution, (sometimes called "Monte-Carlo Methods". used by e.g., particle filters – theoretically much simpler but numerically costly)
- Fundamental approach for *Analytic Calculation*:
  - Assume analytic density functions for errors, (gaussian, triangular)
  - To compute $p(x_t \mid x_{t-1}, u_t)$: For given (x_t, x_{t-1}, u_t) analytically write the error value expressions as a function of (x_t, x_{t-1}, u_t) ,
  - Assume error components independent,
  - Derive (by using closed form density expression for the error) and multiply individual error component probabilities

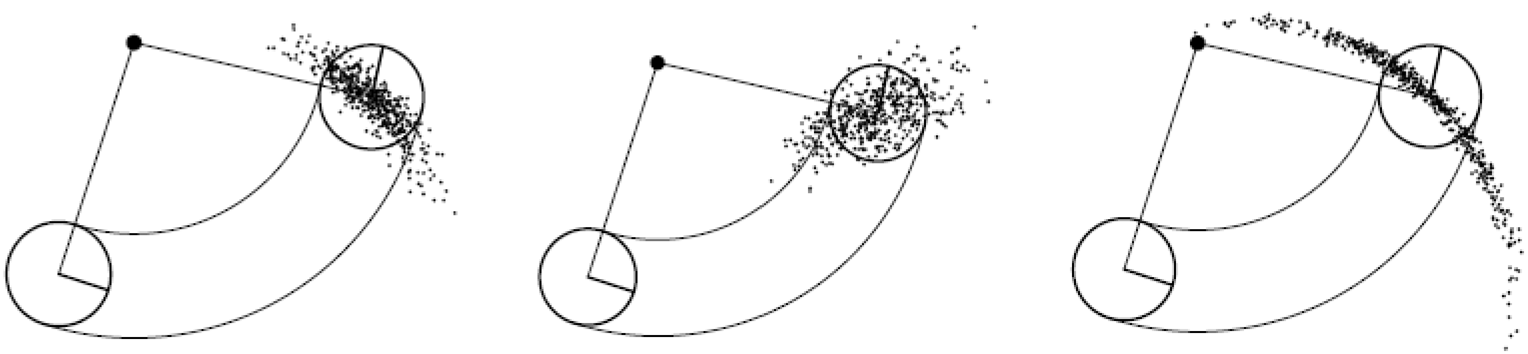
Roadmap for Deriving A Motion Model

- Analytic (closed form) model:
- **For any given** (x_t, x_{t-1}, u_t), ***compute*** ***the value*** $p(x_t \mid x_{t-1}, u_t)$**:**



Roadmap for Deriving A Motion Model

- Fundamental approach for sampling:
  - **Given (x_{t-1}, u_t), *__sample__* (x_t) according to $p(x_t \mid x_{t-1}, u_t)$:**
    - Instantiate (sample) errors according to given distributions,
    - Add error values to odometry or velocities,
    - Find the *erronous* “realization” (x_t).
    - This is “one sample” from the model. Draw enough samples to “see” or represent the distribution:



Odometry Motion Model

Let us start by deriving an analytic (closed form) Motion Model…

Odometry Motion Model

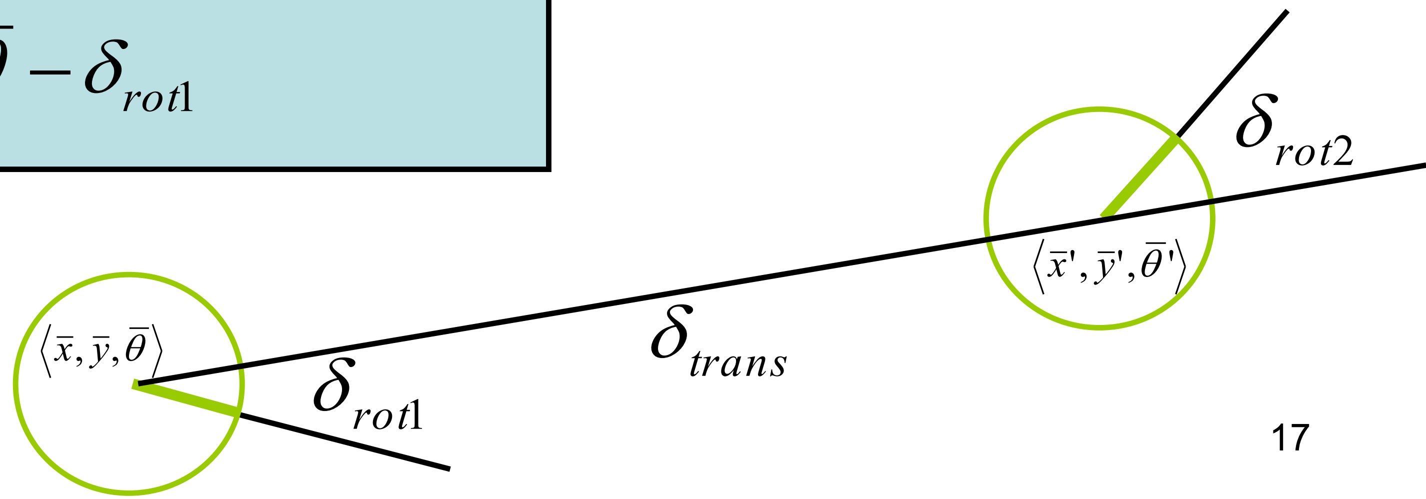
- According to odometry information (command u_t): Robot moves from $\langle \bar{x}, \bar{y}, \bar{\theta} \rangle$ to $\langle \bar{x}', \bar{y}', \bar{\theta}' \rangle$ from time *t-1* to *t*.
- Represent odometry information $u = \langle \delta_{rot1}, \delta_{rot2}, \delta_{trans} \rangle$

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

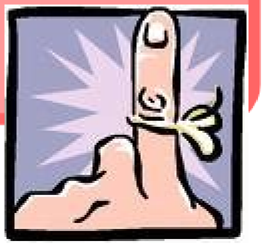
$$\delta_{rot1} = \operatorname{atan2}(\bar{y}' - \bar{y}, \bar{x}' - \bar{x}) - \bar{\theta}$$

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

"Bar" on the variables indicates noisy odometry treated as "command" u_t



The atan2 function



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

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

Noise model for Odometry

- ***Assumption:*** The "*measured" motion* is given by the "*true" motion corrupted with indep. noise*.
- Hence "true" motion is given by:

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

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

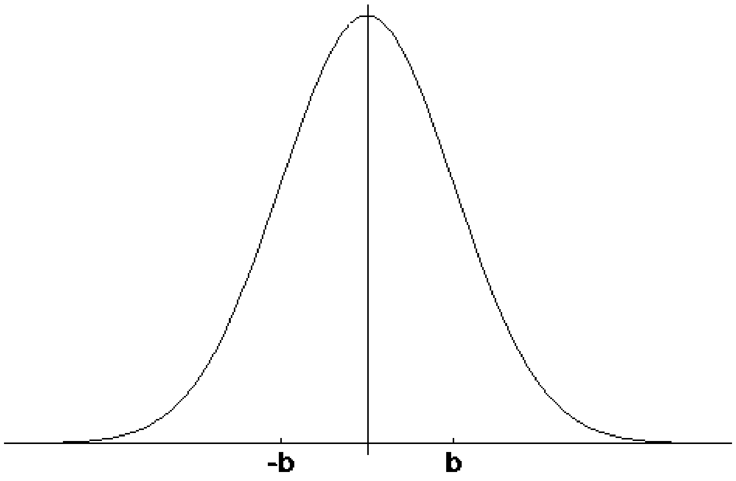
$$\hat{\delta}_{rot2} = \delta_{rot2} + \varepsilon_{\alpha_1 |\delta_{rot2}| + \alpha_2 |\delta_{trans}|}$$

ε_b is zero-mean noise with standard deviation *b*

- Here: $\alpha_1, \ldots, \alpha_4$ are robot specific noise parameters

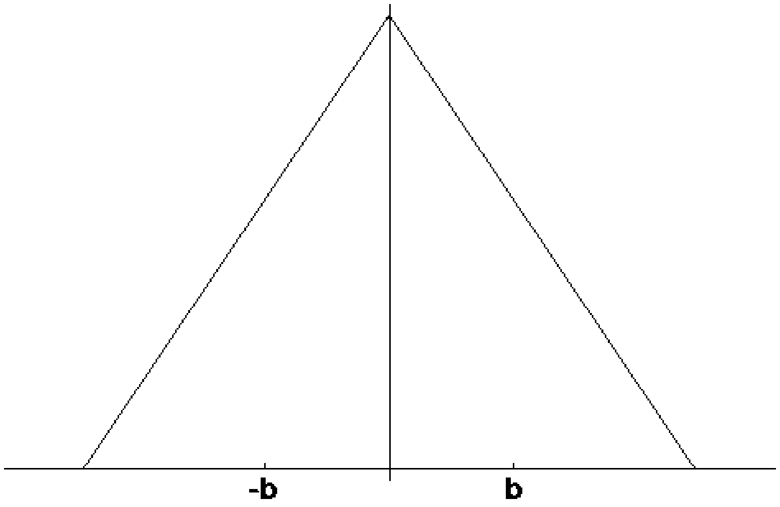
Density Functions for Errors

"Normal" (Gaussian) density function



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

Triangular density function



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

… assuming x is the error variable

Calculating the Error Probability

- For a zero-mean normal density

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

"error" x

Std. Dev. σ

- For a zero-mean triangular density

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 $p(x'| x,u)$

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. $p_1 = \text{prob}(\quad , \alpha_1 |\hat{\delta}_{\text{rot1}}| + \alpha_2 \hat{\delta}_{\text{trans}})$
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 measured "control" values (u) *"Commanded Motion"*

current and hypothetical next state (x, x') given. *"true motion"*

"motion error"

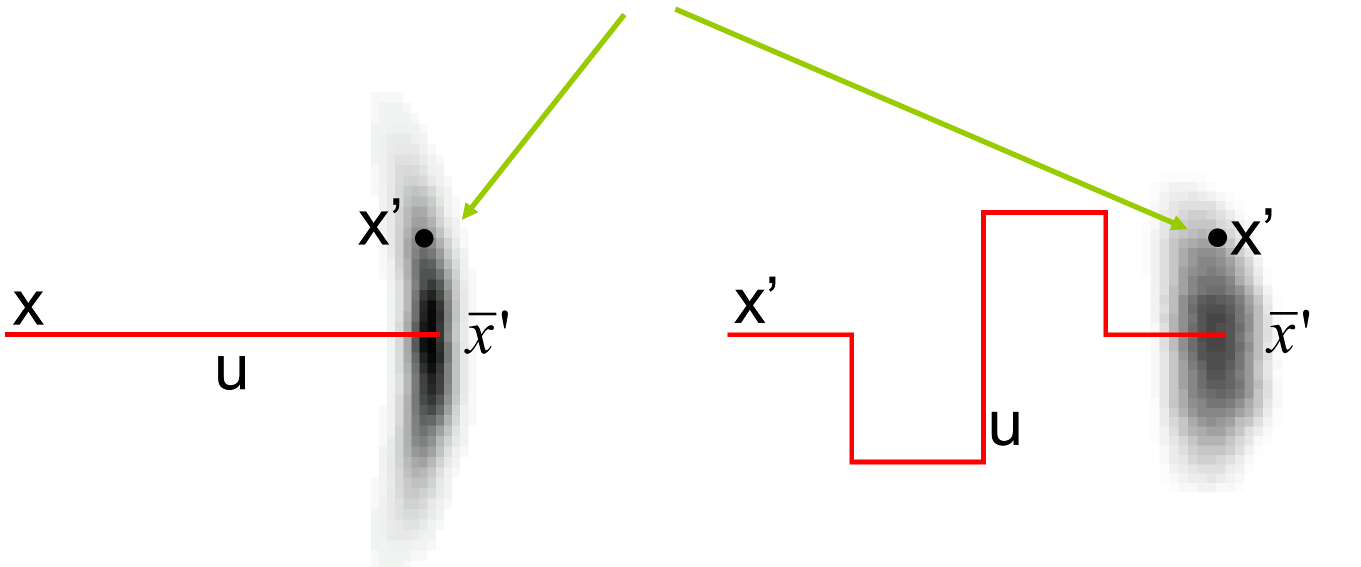
then: closed form probabilities of individual error components

Independence assumption on the error components

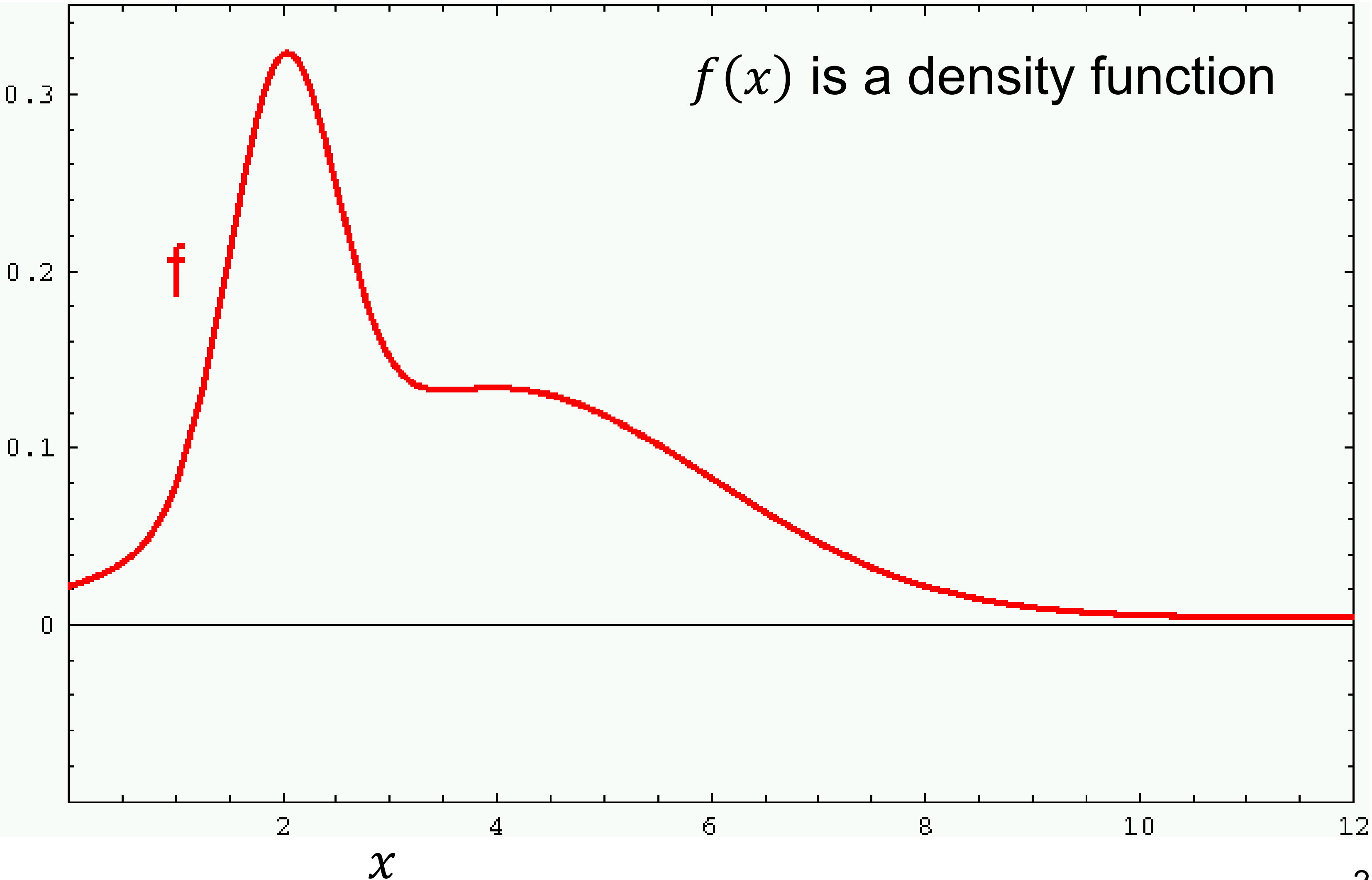
Application

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

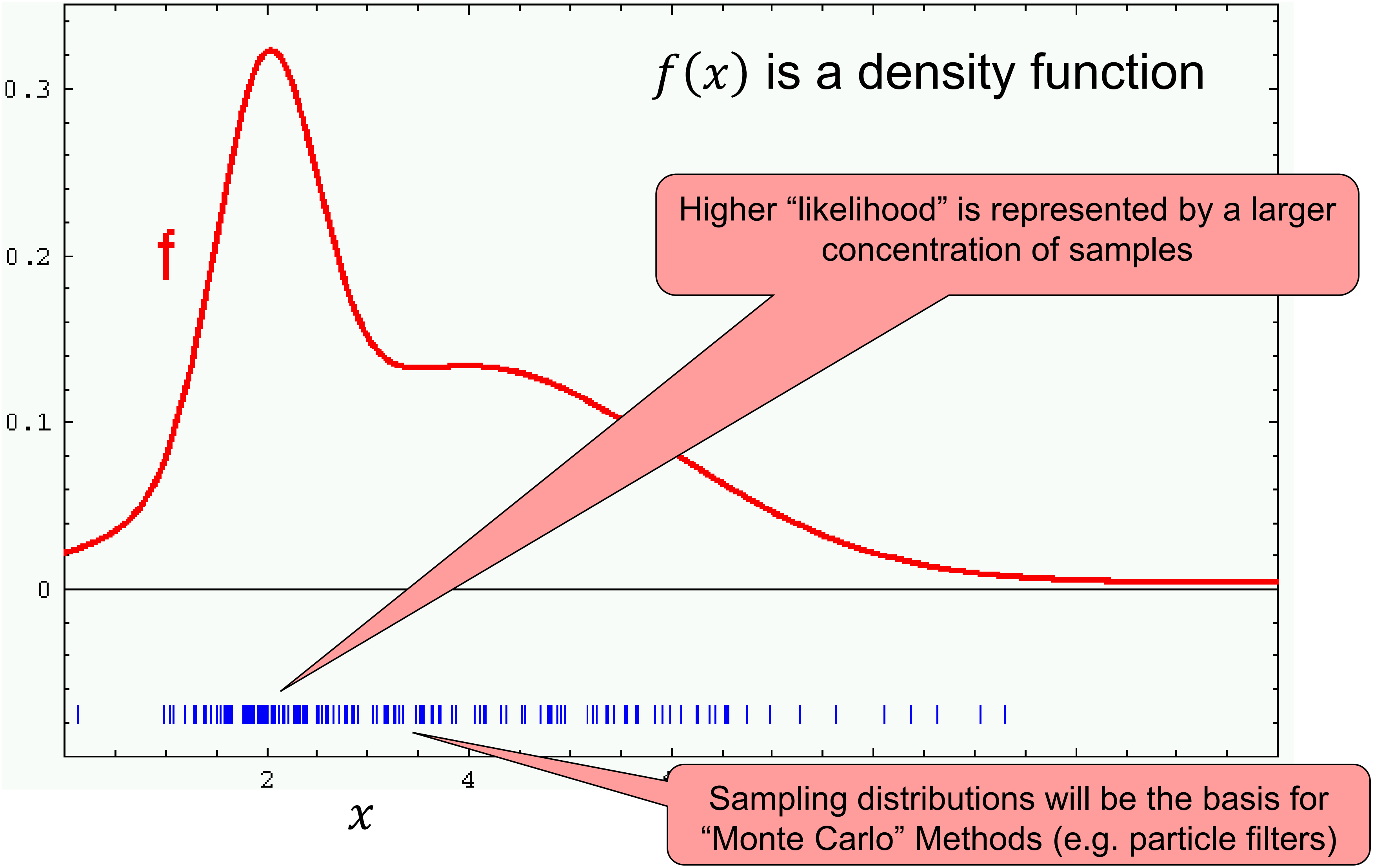
$$p(x'|x,u)$$



Sample based Density Representation



Sample based Density Representation



How to sample?

- Sampling from a Gaussian (Normal) distribution

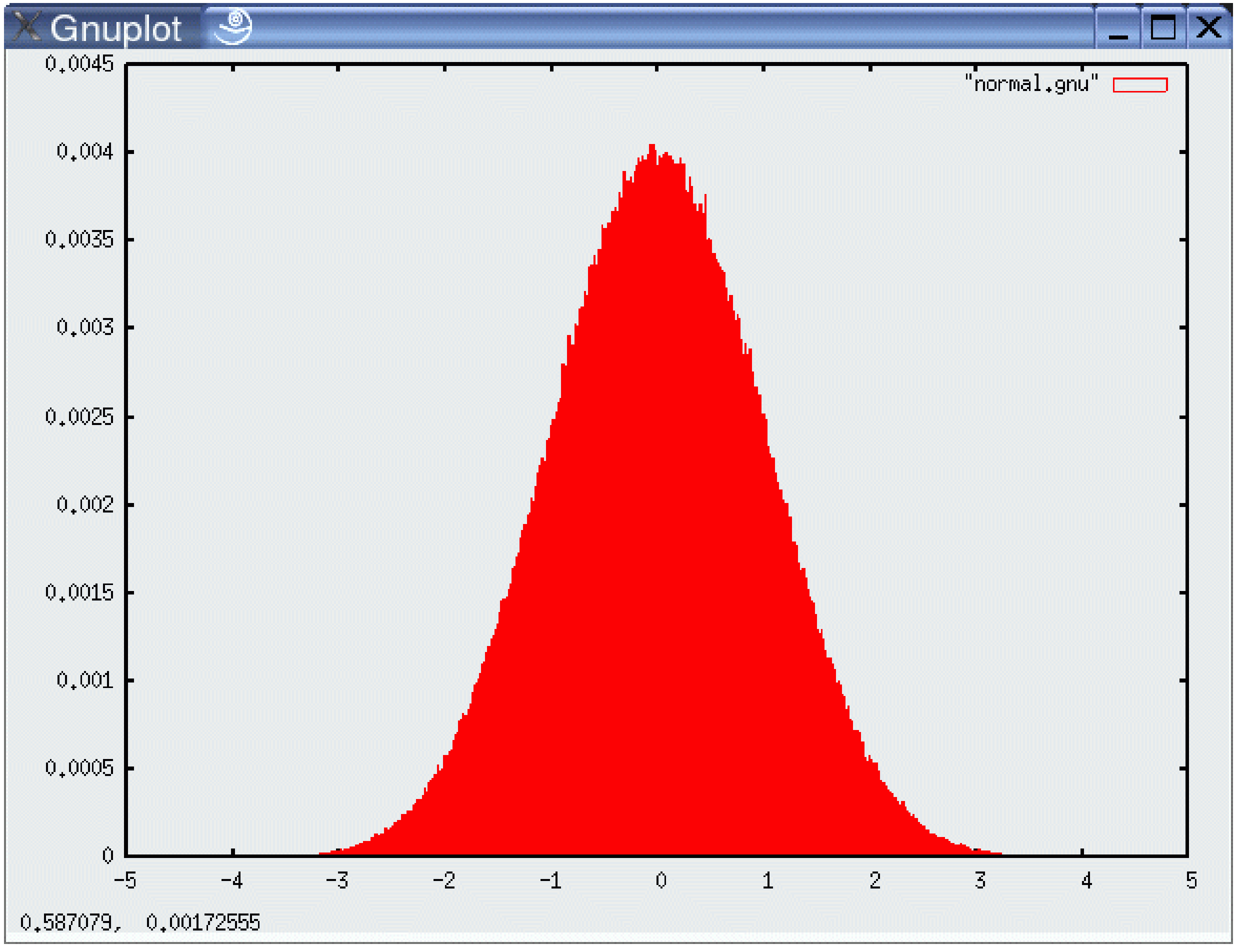
Exploits the "Central Limit Theorem"!

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

- Sampling from a triangular distribution

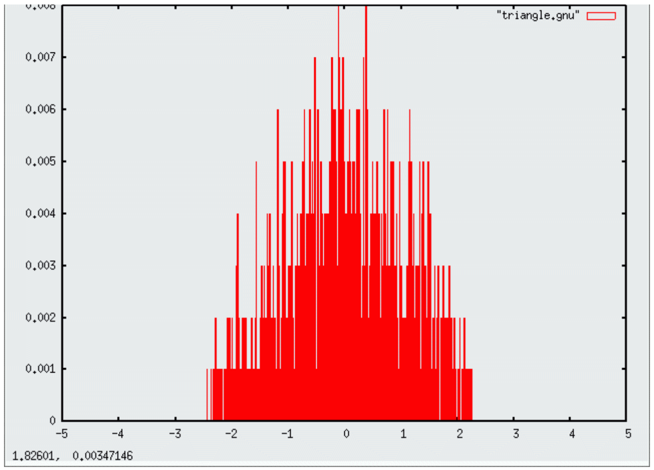
1. Algorithm **sample_triangular_distribution**(b):
2. return $\frac{\sqrt{6}}{2}\left[\mathbf{rand}(-b, b) + \mathbf{rand}(-b, b)\right]$

Illustration: Gaussian Distribution

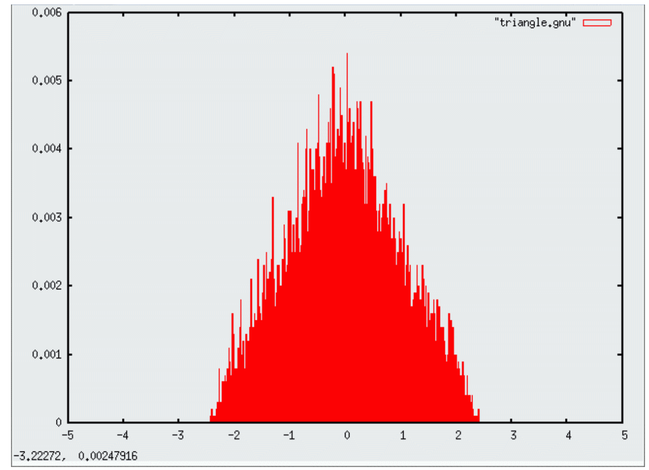


10^6 samples

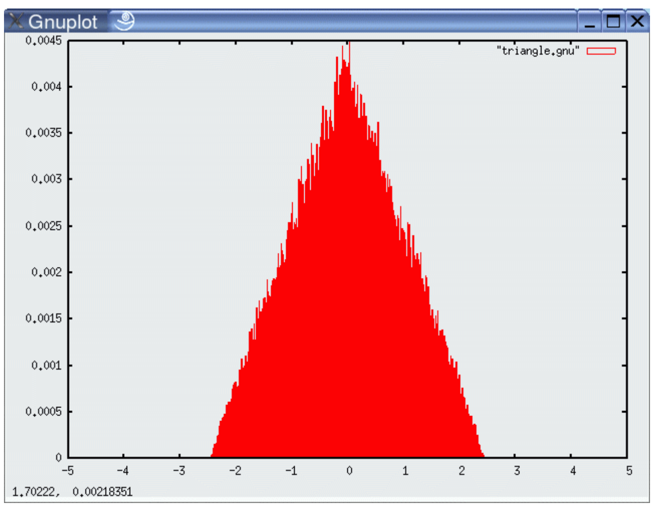
Illustration: Triangular Distribution



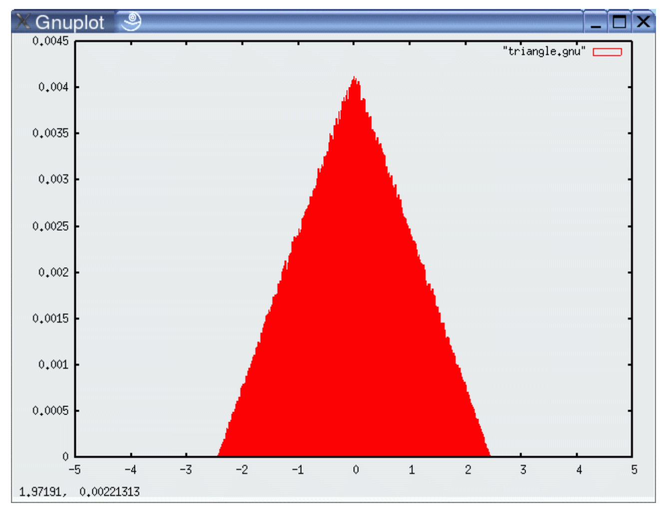
10^3 samples



10^4 samples



10^5 samples



10^6 samples

Rejection Sampling

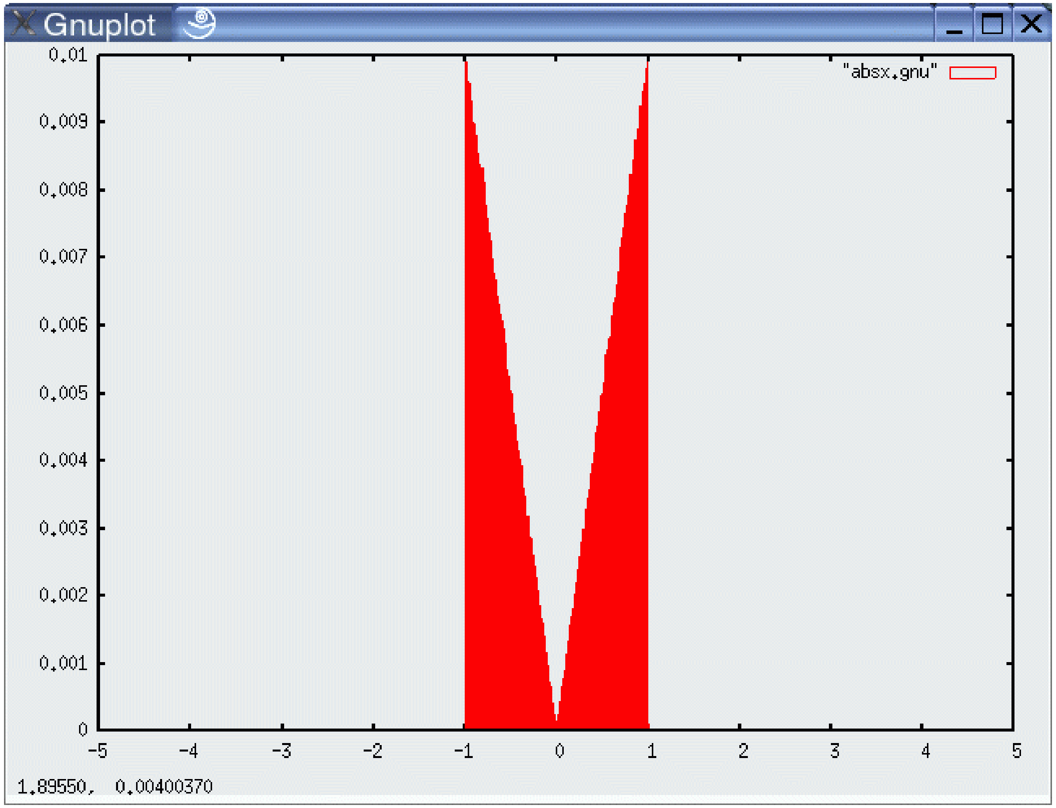
- Sampling from an arbitrary distribution $f(b)$

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

Example

- Sampling from

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



Sampling from the Odometry Model

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 $\langle x', y', \theta' \rangle$

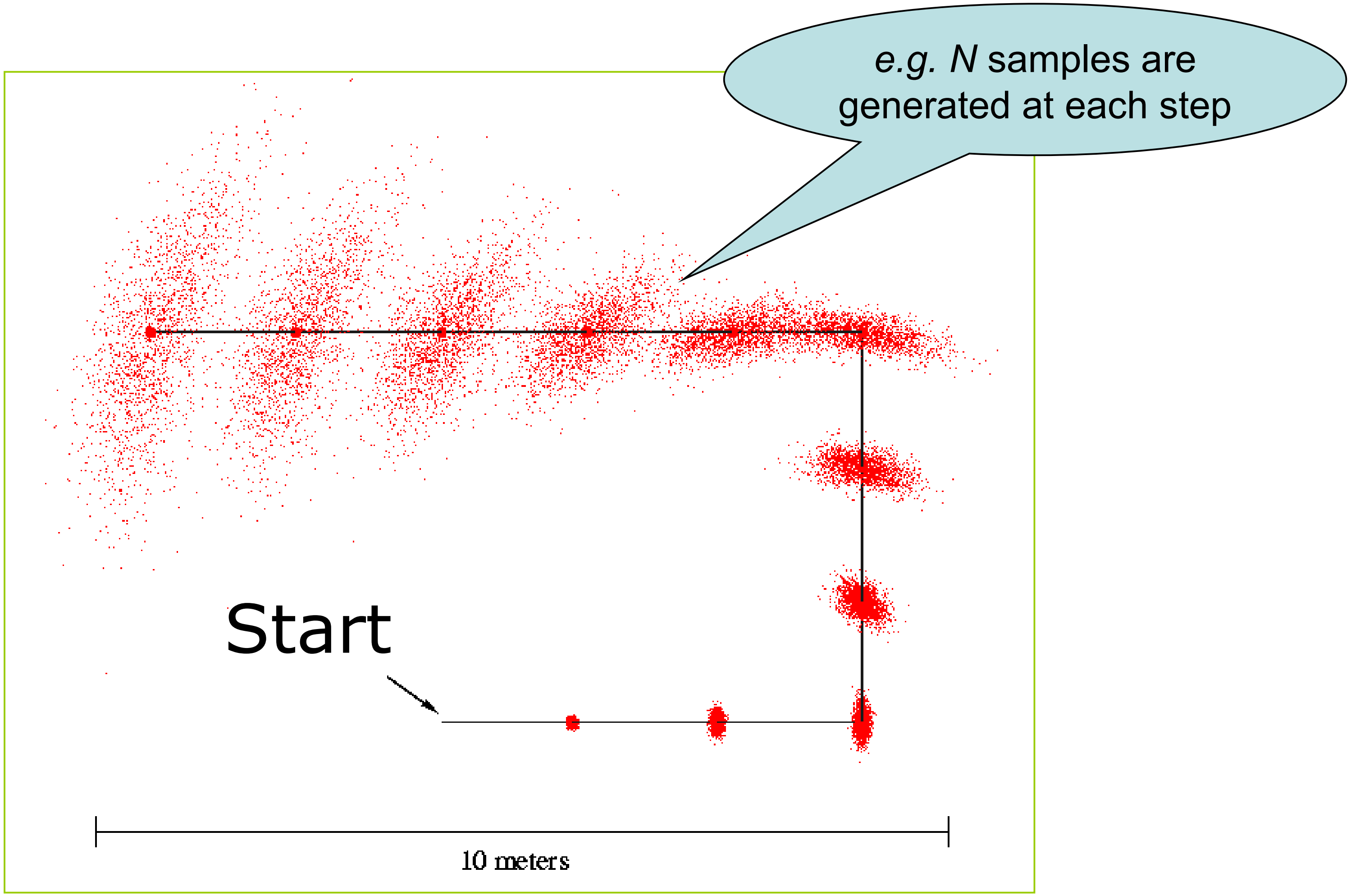
Given current state and current controls

Generate a sample of “noisy controls”

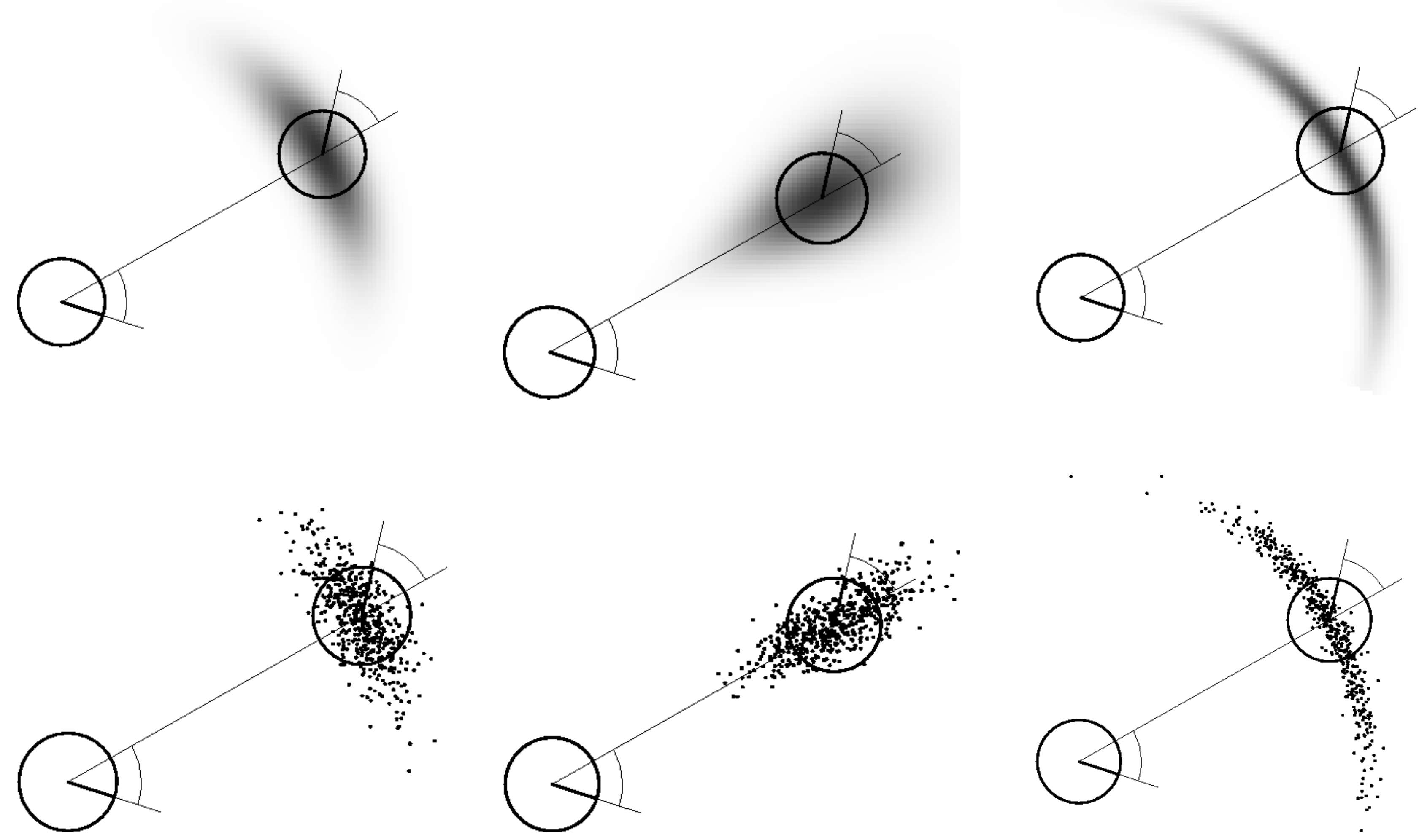
Find a sample of “noisy next state

Example: “sample_normal_distribution”

Illustration: Multiple Steps



Examples: Odometry Based



Velocity Motion Model - Geometry

$$u = \begin{pmatrix} v \\ w \end{pmatrix}$$ command

Linear velocity

Angular velocity

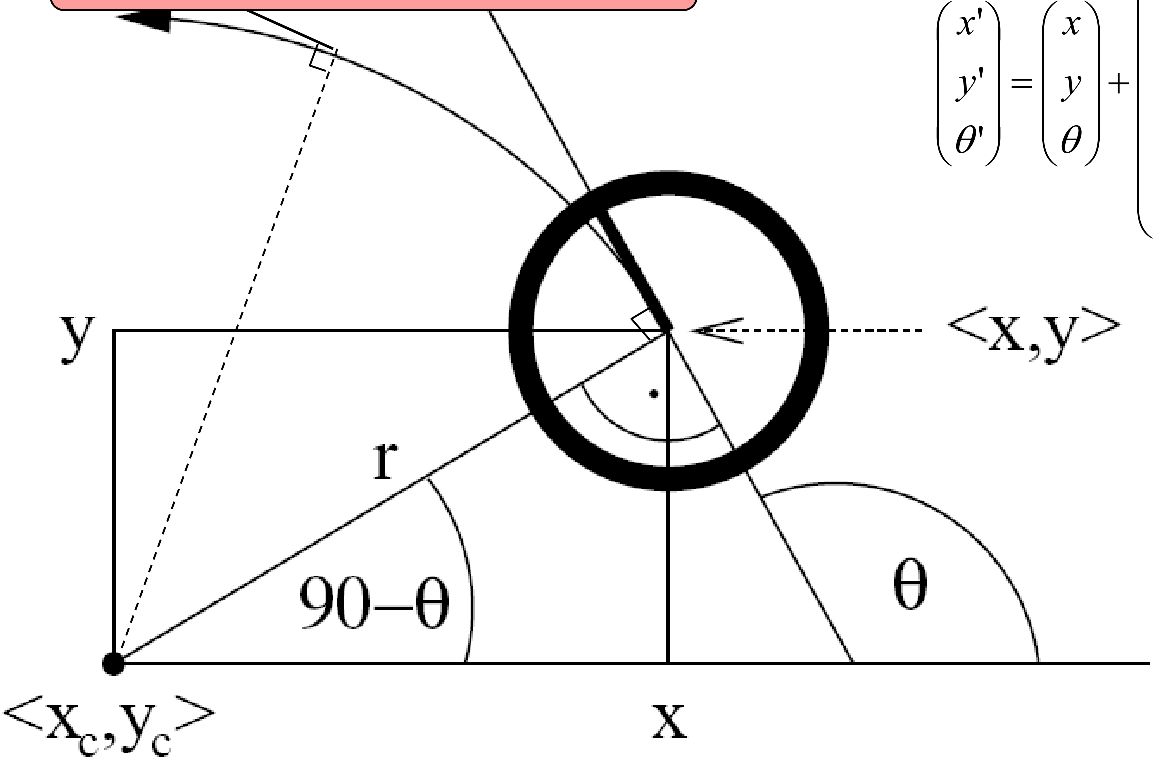
$$r = \left|\frac{v}{w}\right|$$

$$x_c = x - \left|\frac{v}{w}\right| \sin(\theta)$$

$$y_c = y + \left|\frac{v}{w}\right| \cos(\theta)$$

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

"Ideal" kinematic motion



Actual (Noisy) Motion

"command noise" and the noisy command:

$$\begin{pmatrix} \hat{v} \\ \hat{w} \end{pmatrix} = \begin{pmatrix} v \\ w \end{pmatrix} + \begin{pmatrix} \varepsilon_{\alpha_1|v|+\alpha_2|w|} \\ \varepsilon_{\alpha_3|v|+\alpha_4|w|} \end{pmatrix}$$

Forward Model!!

Hence the "noisy" actual motion:

$$\begin{pmatrix} x' \\ y' \\ \theta' \end{pmatrix} = \begin{pmatrix} x \\ y \\ \theta \end{pmatrix} + \begin{pmatrix} -\left|\frac{\hat{v}}{\hat{w}}\right| \sin(\theta) + \left|\frac{\hat{v}}{\hat{w}}\right| \sin(\theta + \hat{w}\Delta t) \\ \left|\frac{\hat{v}}{\hat{w}}\right| \cos(\theta) - \left|\frac{\hat{v}}{\hat{w}}\right| \cos(\theta + \hat{w}\Delta t) \\ \hat{w}\Delta t + \hat{\gamma}\Delta t \end{pmatrix}$$

A 3rd independent noise term on rotation to disturb the perfect "circular path" assumption

Sampling from the Velocity Model

- The *forward model:*

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

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

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

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

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

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

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

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

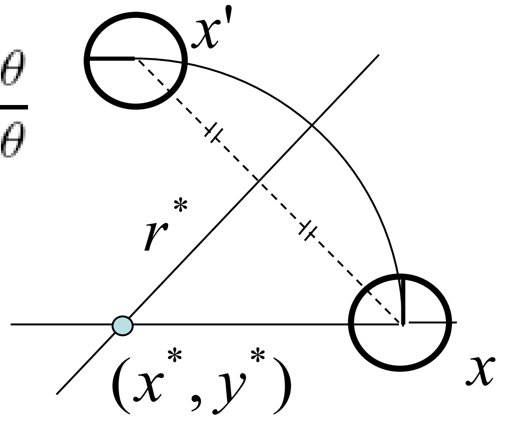
Calculating the Posterior $p(\mathbf{x}'|\,\mathbf{x},\mathbf{u})$

Inverse Model*: Given ($\mathbf{x}'$, $\mathbf{x}$, $\mathbf{u}$) find *pdf value:

From (x,x'), the "*center of circle*":

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

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



Sketch:

- Then find the radius of rotation *r*,
- Then find the rotation angle Δθ,
- Then find *v'* and *w' (from x,x')…*
- Then find the error between (v',w') and and those from ***u***, hence the error,
- Then find the probability of the error!

Calculating the Posterior $p(\mathbf{x}' \mid \mathbf{x}, \mathbf{u})$

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

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

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

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

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

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

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

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

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

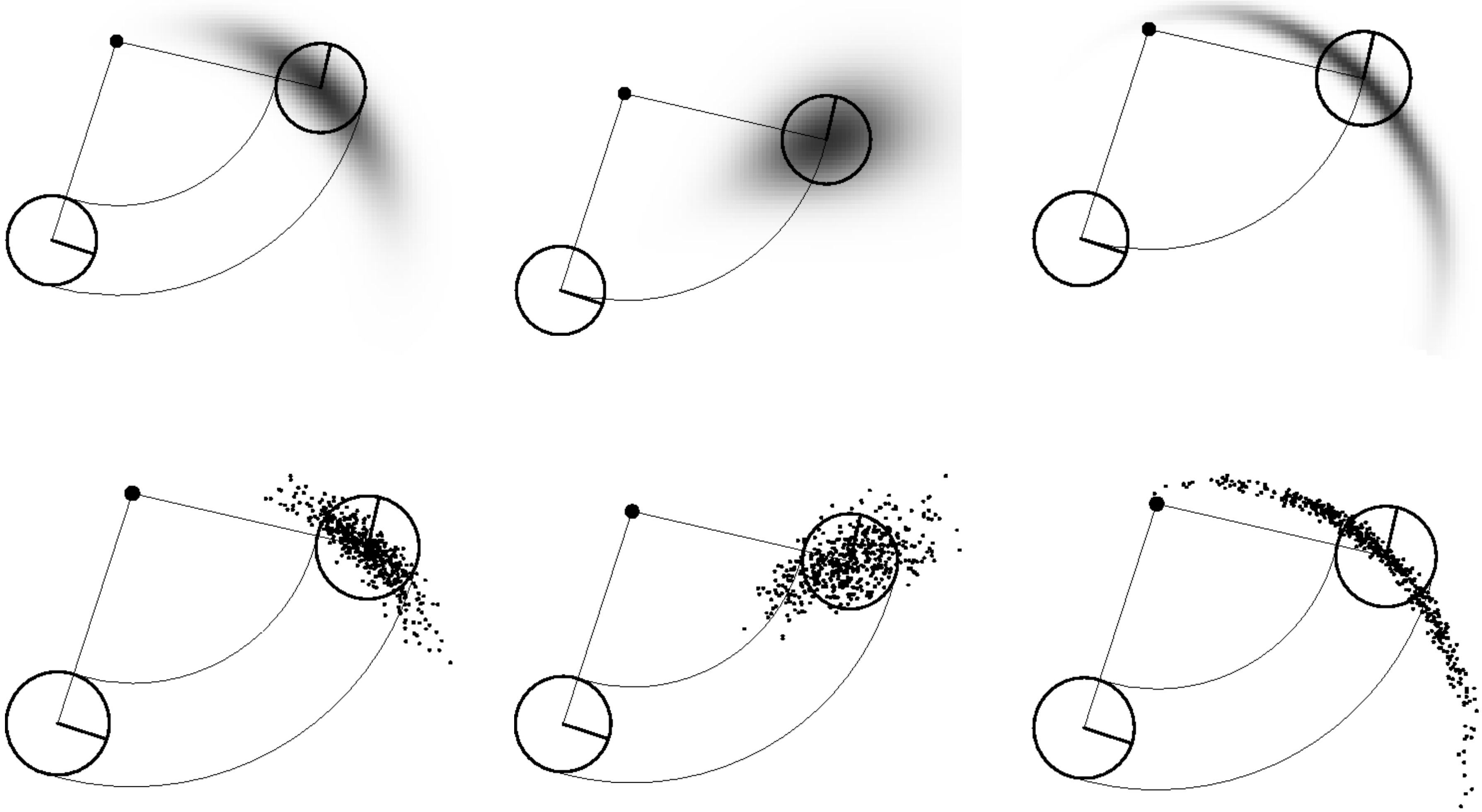
return $\mathbf{prob}(v - \hat{v}, \alpha_1|v| + \alpha_2|\omega|) \cdot \mathbf{prob}(\omega - \hat{\omega}, \alpha_3|v| + \alpha_4|\omega|)$
$\cdot\ \mathbf{prob}(\hat{\gamma}, \alpha_5|v| + \alpha_6|\omega|)$

Command u gives us (v, w).

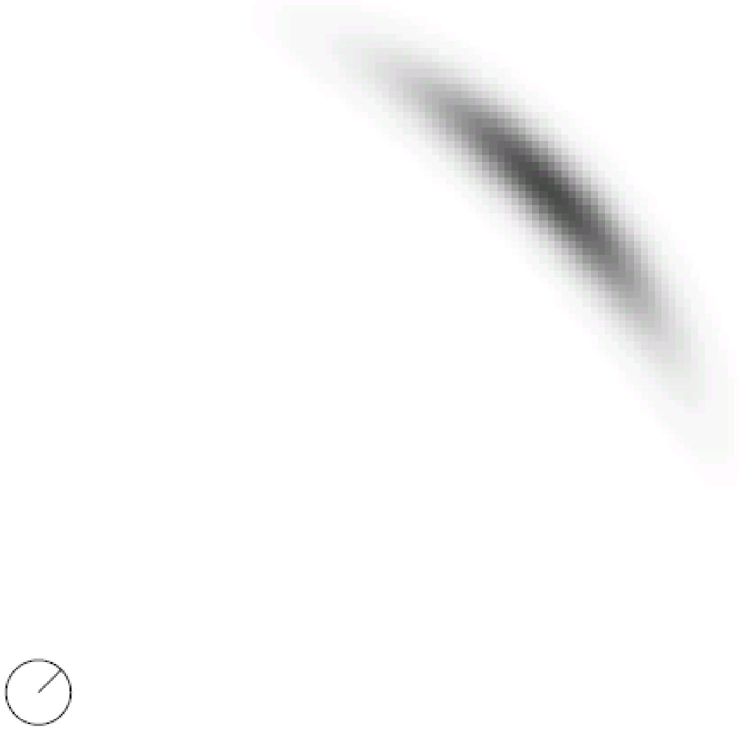
$(\hat{v}, \hat{w})$ calculated from x_t and x_{t-1}

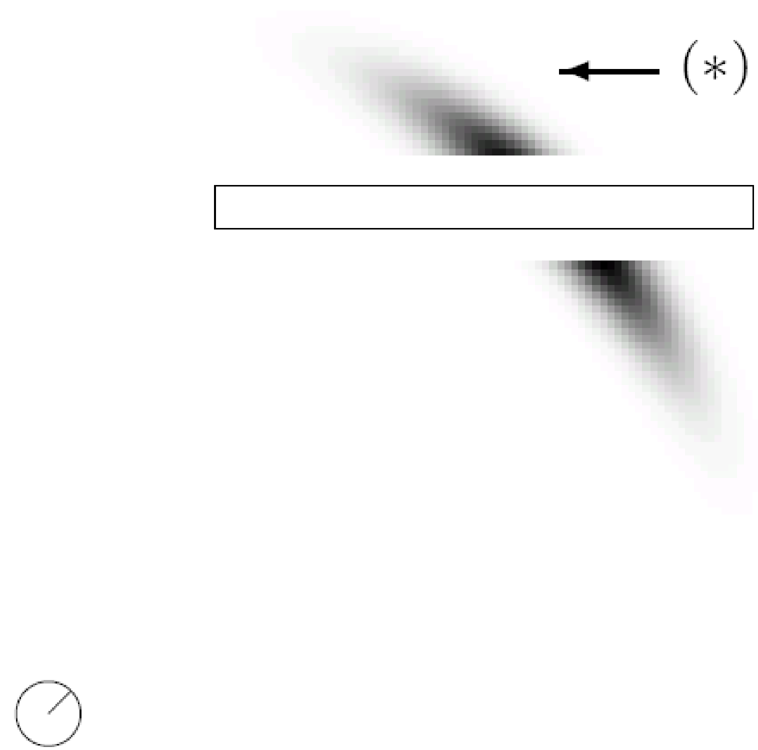
“velocity errors”

Examples: Velocity Based



Map-Consistent Models





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

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

But this implies the robot can pass through a wall!!

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 from the actual robot experiments,
- We also briefly mentioned an extended motion model that takes the map into account.