MEN791
Autonomous Unmanned Vehicles

Motion Models

CONTACT

Ulsan National Institute of Science and Technology

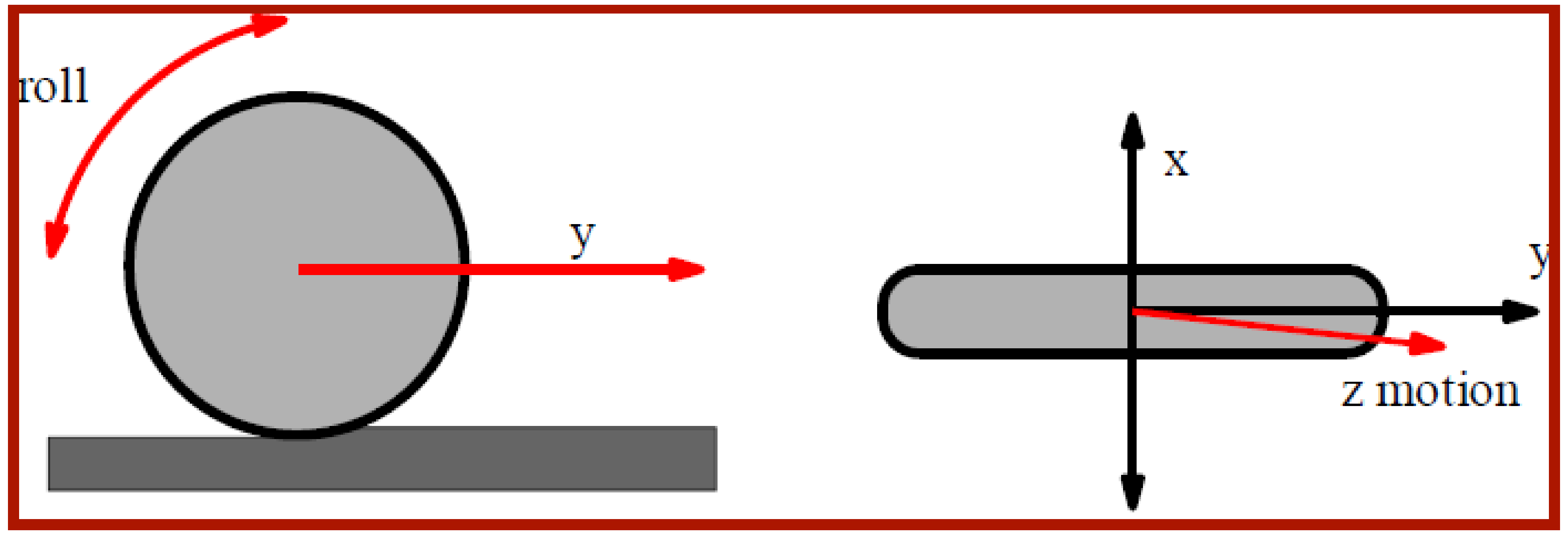
Address 50 UNIST-gil, Ulju-gun, Ulsan, 44919, Korea
Tel. +82 52 217 0114 **Web.** www.unist.ac.kr

School of Mechanical, Aerospace, and Nuclear Engineering
5th Engineering Building Room 801-4
Tel. +82 52 217 3048 **Web.** https://sites.google.com/site/aslunist/

Locomotion of Wheeled Robots

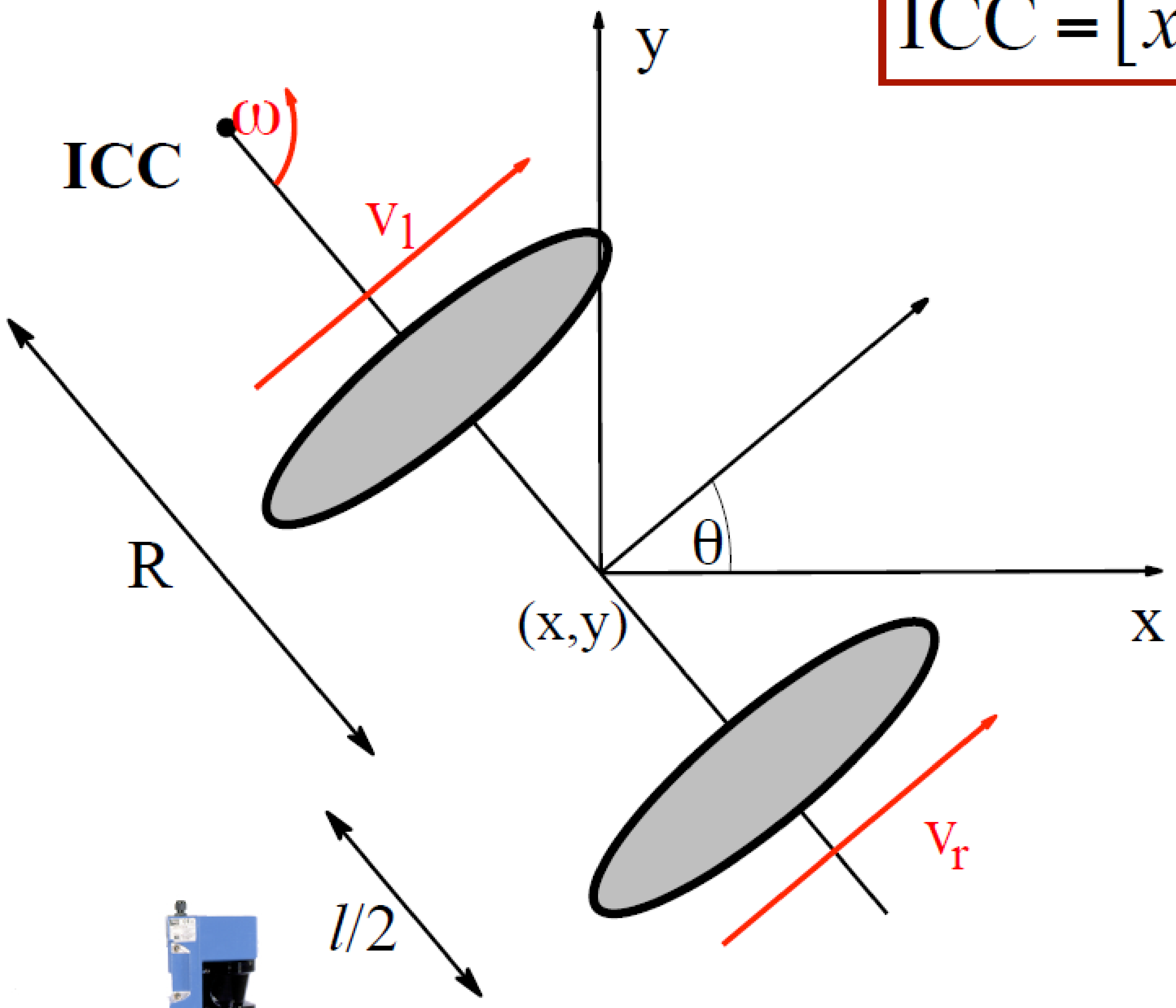
Locomotion (Oxford Dict.):
Power of motion from place to place

- Differential drive (AmigoBot, Pioneer 2-DX)
- Car drive (Ackerman steering)
- Synchronous drive (B21)
- XR4000
- Mecanum wheels



we also allow wheels to rotate around the z axis

Differential Drive



$$\boxed{\mathrm{ICC} = [x - R\sin\theta,\ y + R\cos\theta]}$$

$$\omega(R + l/2) = v_r$$

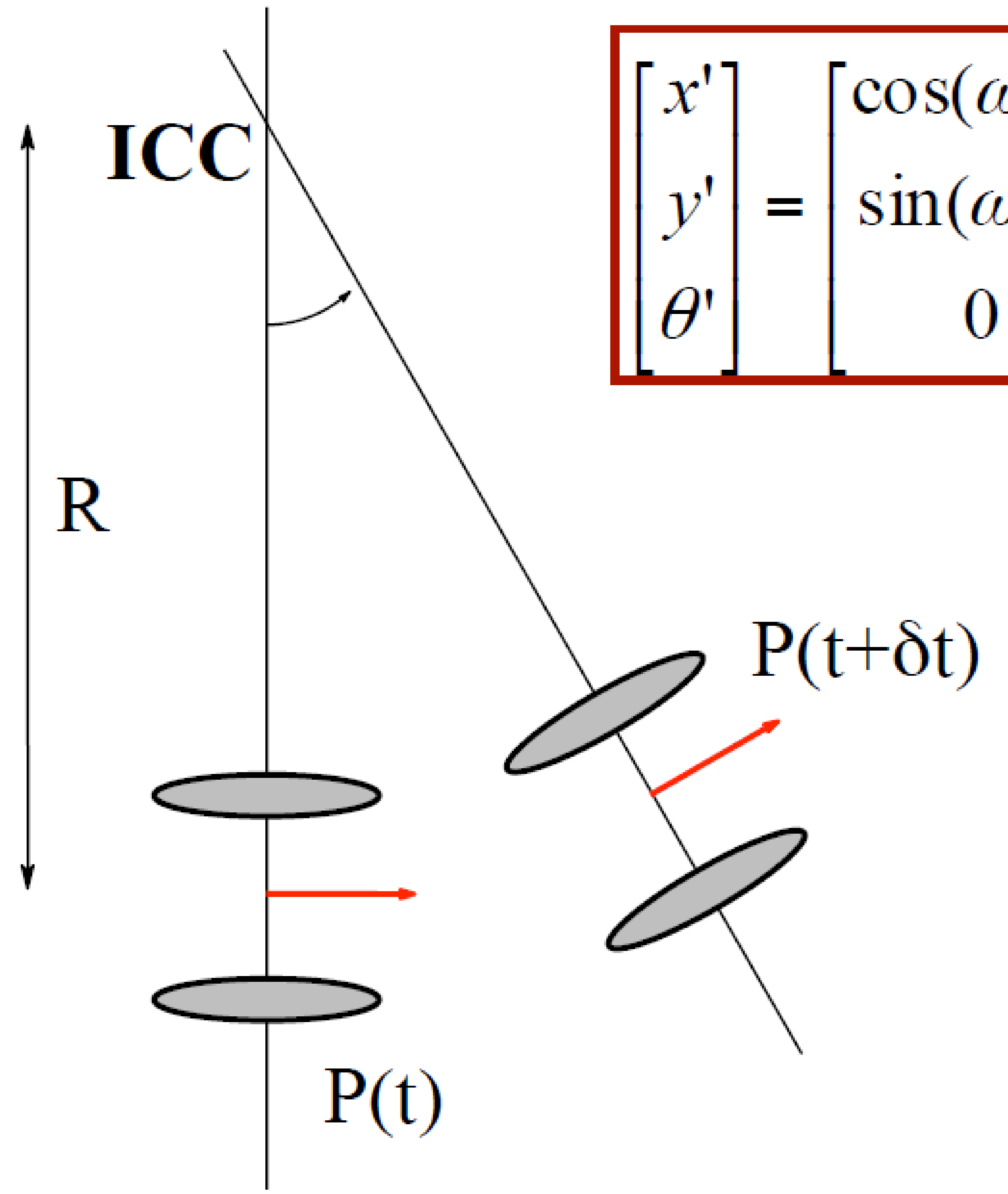
$$\omega(R - l/2) = v_l$$

$$R = \frac{l}{2}\frac{(v_l + v_r)}{(v_r - v_l)}$$

$$\omega = \frac{v_r - v_l}{l}$$

$$v = \frac{v_r + v_l}{2}$$

Differential Drive: Forward Kinematics



$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} \cos(\omega\delta t) & -\sin(\omega\delta t) & 0 \\ \sin(\omega\delta t) & \cos(\omega\delta t) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x - ICC_x \\ y - ICC_y \\ \theta \end{bmatrix} + \begin{bmatrix} ICC_x \\ ICC_y \\ \omega\delta t \end{bmatrix}$$

$$x(t) = \int_0^t v(t') \cos[\theta(t')]dt'$$

$$y(t) = \int_0^t v(t') \sin[\theta(t')]dt'$$

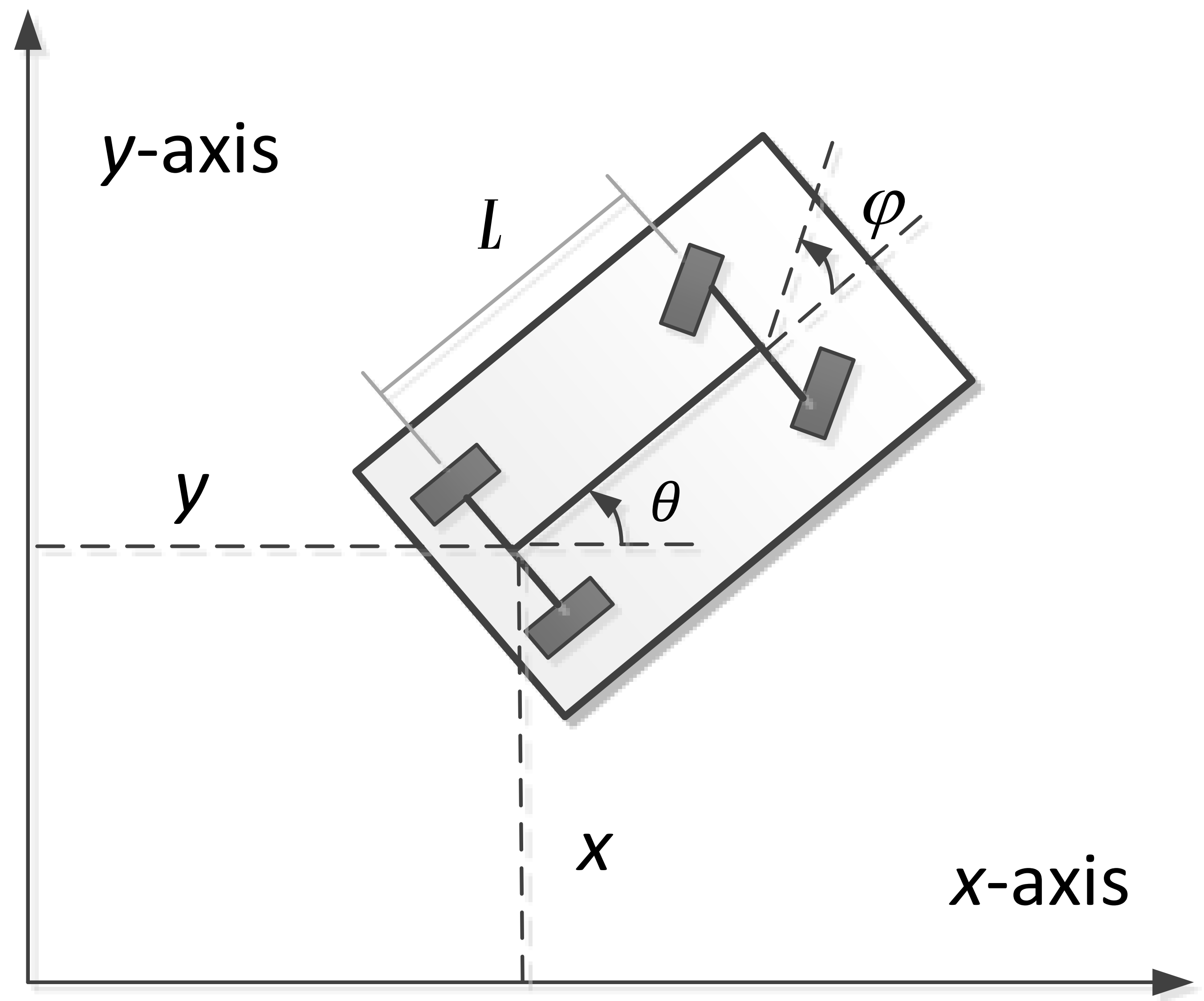
$$\theta(t) = \int_0^t \omega(t')\, dt'$$

$$x(t) = \frac{1}{2}\int_0^t [v_r(t') + v_l(t')] \cos[\theta(t')]dt'$$

$$y(t) = \frac{1}{2}\int_0^t [v_r(t') + v_l(t')] \sin[\theta(t')]dt'$$

$$\theta(t) = \frac{1}{l}\int_0^t [v_r(t') - v_l(t')]\, dt'$$

Car example



$$\dot{x} = v \ \cos\theta$$
$$\dot{y} = v \ \sin\theta$$
$$\dot{\theta} = (v/L) \ \tan\varphi$$
$$|\varphi| < \Phi$$

State $q = \begin{pmatrix} x \\ y \\ \theta \end{pmatrix}$ **Controls** $u = \begin{pmatrix} v \\ \varphi \end{pmatrix}$

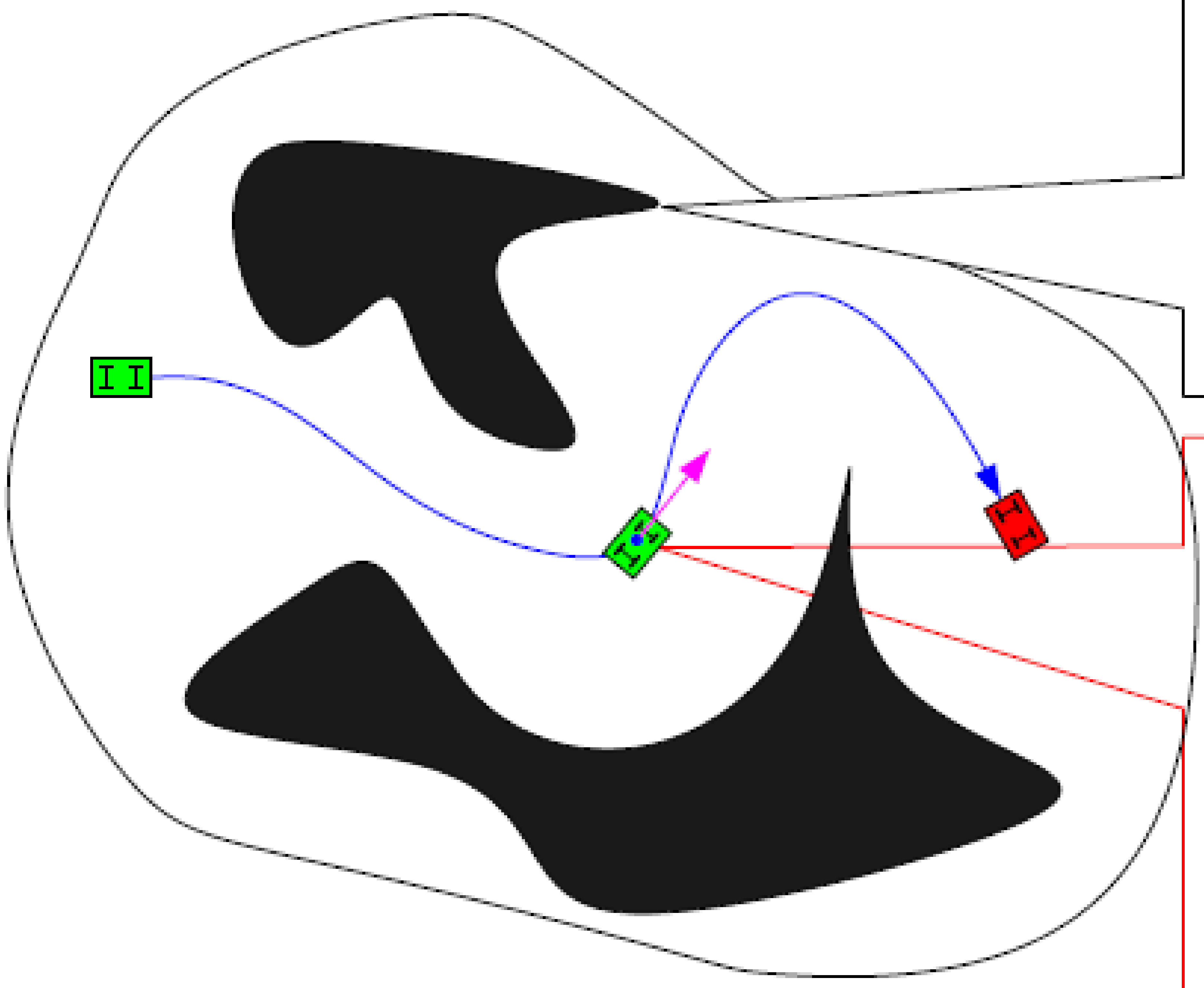
Sytem equation $\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} v \ \cos\theta \\ v \ \sin\theta \\ (v/L) \ \tan\varphi \end{pmatrix}$

- The car is a *non-holonomic* system: not all DoFs are controlled,
  $\dim(u) < \dim(q)$
  We have the *differential constraint* $\dot{q}$:

$$\dot{x}\sin\theta - \dot{y}\cos\theta = 0$$

  "A car cannot move directly lateral."

- Analogy to dynamic systems: Just like a car cannot instantly move sidewards, a dynamic system cannot instantly change its position q: the current change in position is *constrained* by the current velocity $\dot{q}$.



Configuration Constraints

$\{g_e(q) = 0; g_i(q) > 0\}$

- Collision avoidance
- Complicated geometry
- High Dimension

Differential Constraints

$\{h_e(q,\dot{q}) = 0; h_i(q,\dot{q}) > 0\}$

$\{k_e(q,\dot{q},\ddot{q}) = 0; k_i(q,\dot{q},\ddot{q}) > 0\}$

- Restricted velocities and accelerations
- Nonlinear constraints

- 2-D or 3-D UAV kinematic model
  - In order to deliver guidance inputs to low-level SAS/CAS, 2 or 3-D kinematic model for UAV is considered

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\psi} \\ \dot{v} \\ \dot{\omega} \end{pmatrix} = f(\mathbf{x}, \mathbf{u}) = \begin{pmatrix} v\cos\psi \\ v\sin\psi \\ \omega \\ -\frac{1}{\tau_v}v + \frac{1}{\tau_v}u_v \\ -\frac{1}{\tau_\omega}\omega + \frac{1}{\tau_\omega}u_\omega \end{pmatrix}$$

$$|u_v - v_0| \leq v_{max}$$
$$|u_\omega| \leq \omega_{max}$$

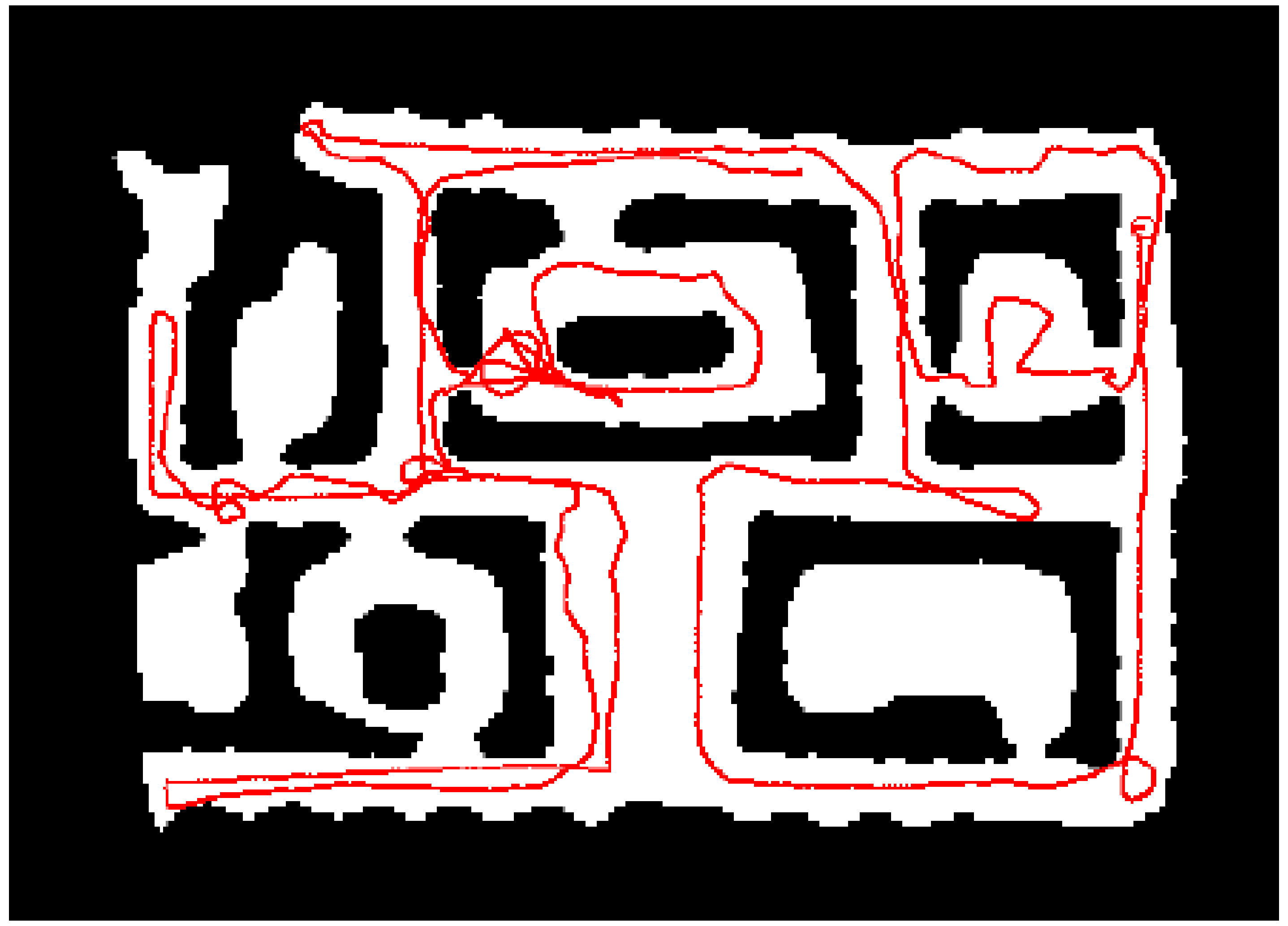
This can be discretised by Euler integration into:

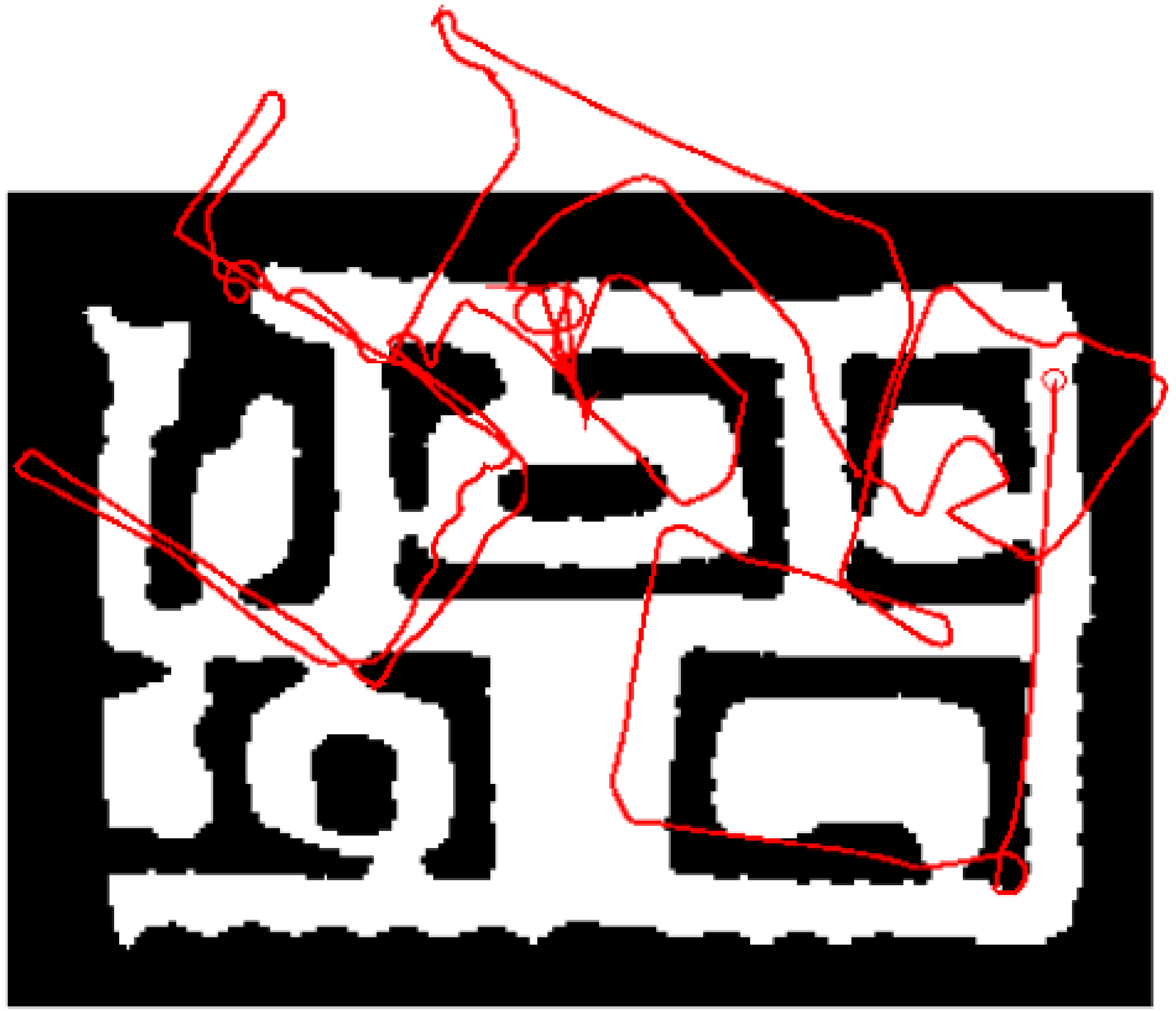
$$\mathbf{x}_{k+1} = f_d(\mathbf{x}_k, \mathbf{u}_k) = \mathbf{x}_k + T_s f(\mathbf{x}_k, \mathbf{u}_k)$$

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{\chi} \\ \dot{\gamma} \\ \dot{v} \\ \dot{\omega}_\chi \\ \dot{\omega}_\gamma \end{pmatrix} = f(\mathbf{x}, \mathbf{u}) = \begin{pmatrix} v\cos\chi\cos\gamma \\ v\sin\chi\cos\gamma \\ v\sin\gamma \\ \omega_\chi \\ \omega_\gamma \\ -\frac{1}{\tau_v}v + \frac{1}{\tau_v}u_v \\ -\frac{1}{\tau_{\omega_\chi}}\omega_\chi + \frac{1}{\tau_{\omega_\chi}}u_{\omega_\chi} \\ -\frac{1}{\tau_{\omega_\gamma}}\omega_\gamma + \frac{1}{\tau_{\omega_\gamma}}u_{\omega_\gamma} \end{pmatrix}$$

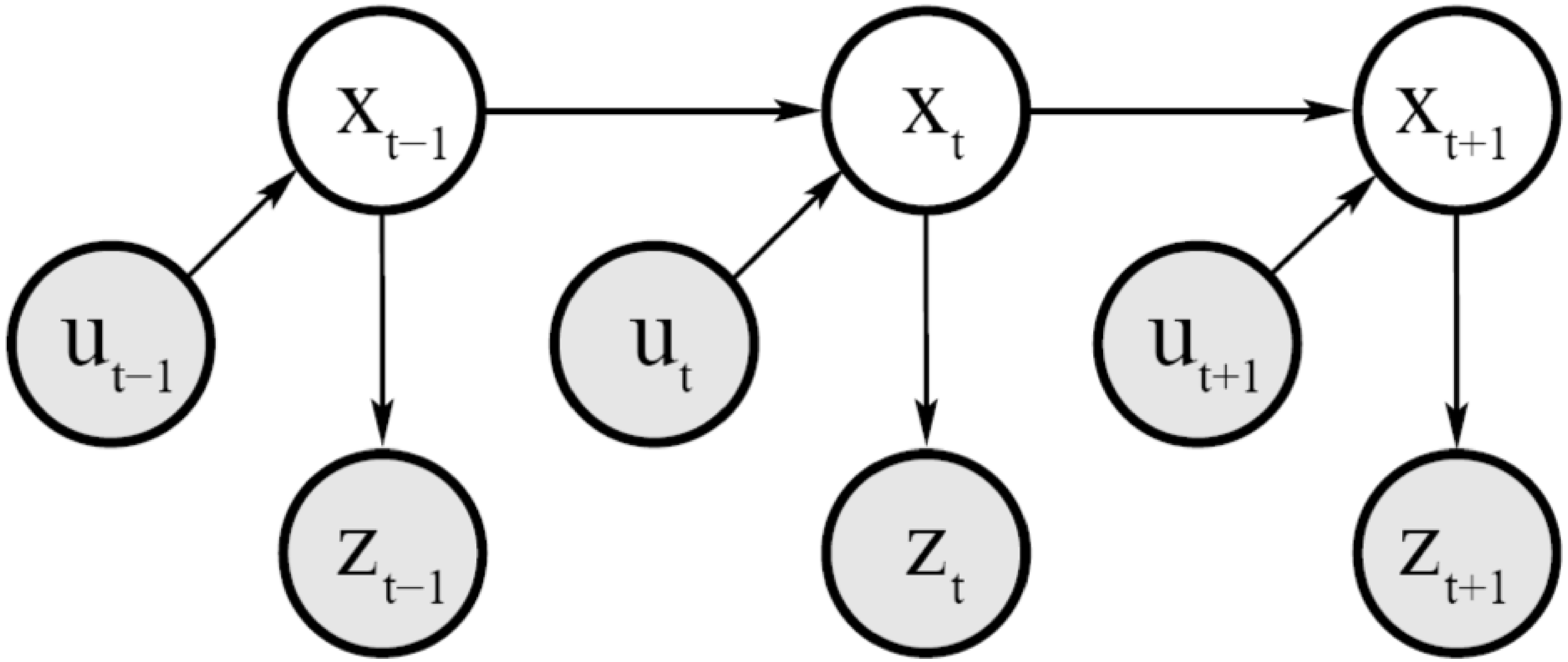
Robot Motion

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



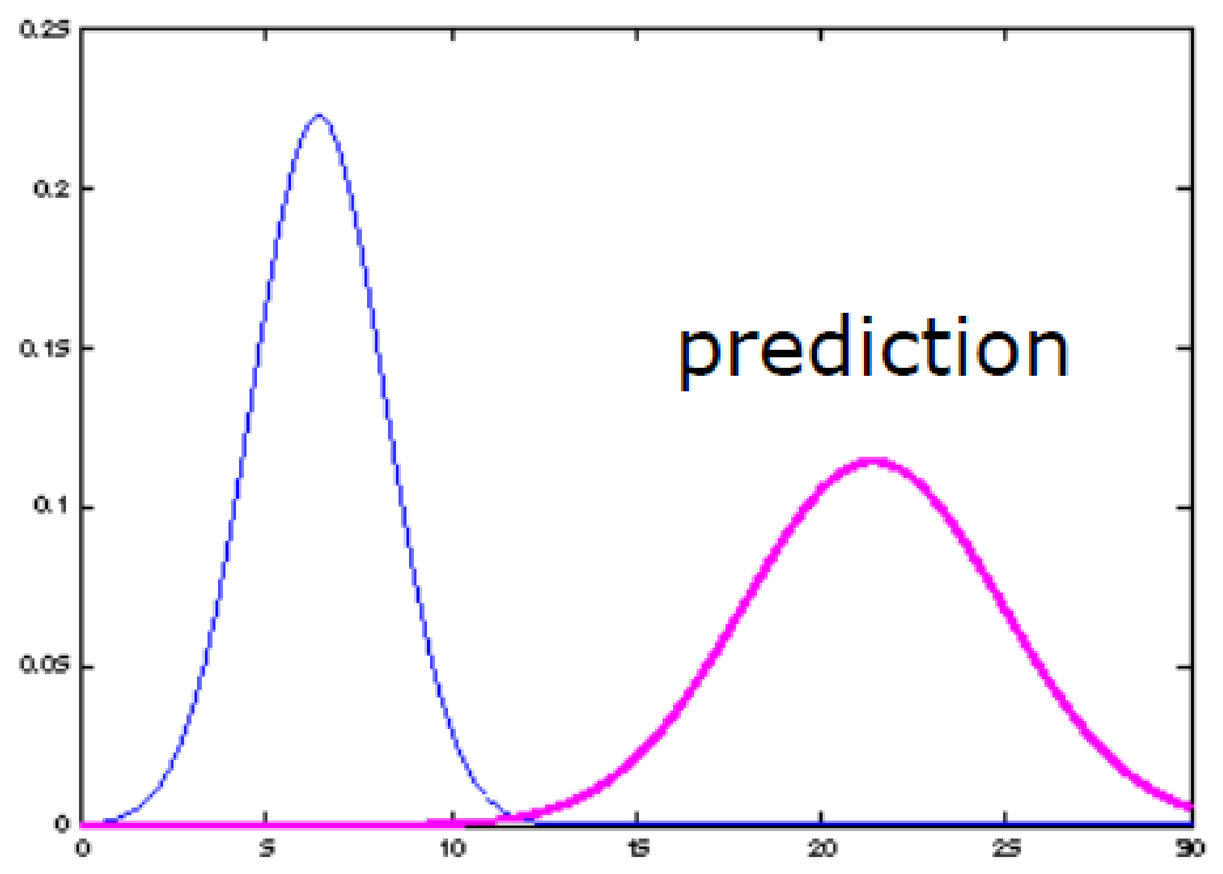


Dynamic Bayesian Network for Controls, States, and Sensations



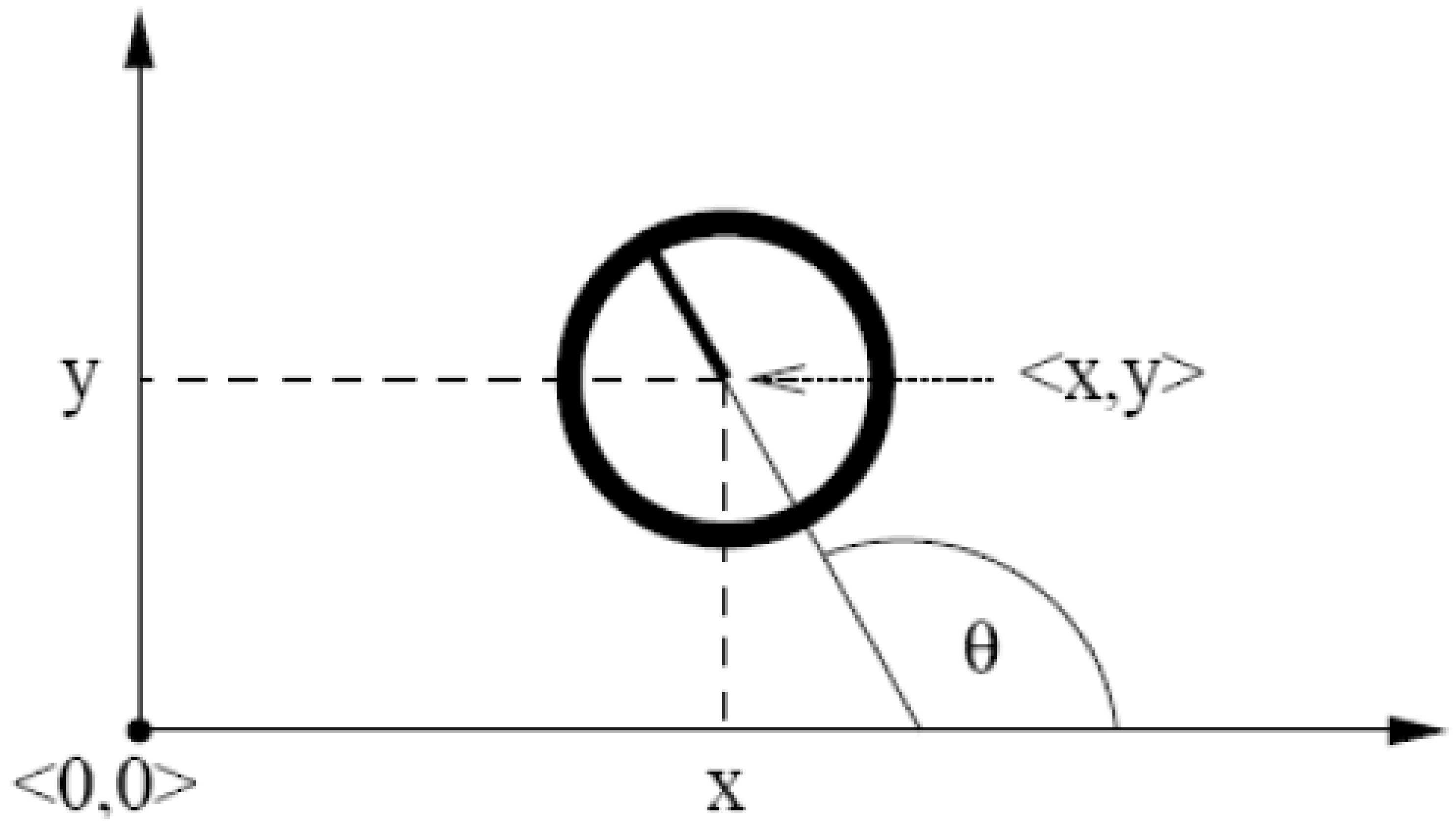
Probabilistic Motion Models

- To implement the Bayes Filter, we need the transition model $p(x_t \mid x_{t-1}, u_t)$
- The term $p(x_t \mid x_{t-1}, u_t)$ specifies a posterior probability, that action u 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 motion equations.



Coordinate Systems

- The configuration of a typical wheeled robot in 3D can be described by six parameters.
- Three-dimensional Cartesian coordinates plus three Euler angles roll, pitch, and yaw.
- Throughout this section, we consider robots operating on a planar surface.
- The state space of such systems is three-dimensional (x, y, θ).



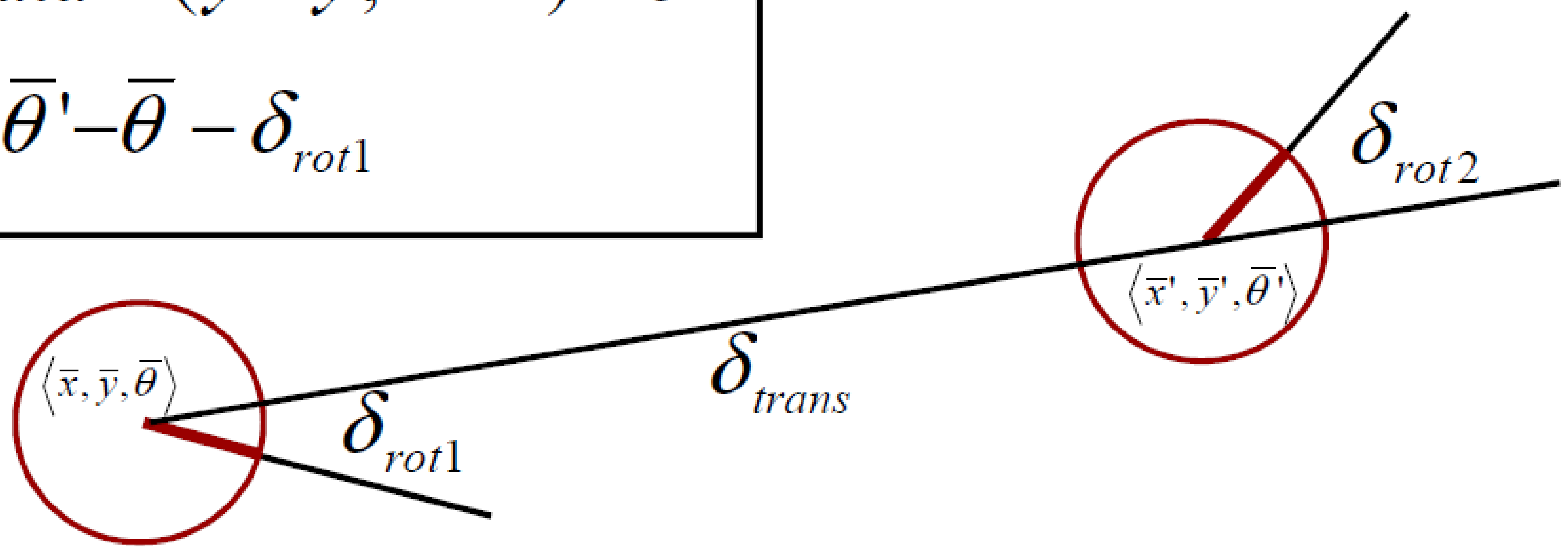
Typical Motion Models

- 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.

Odometry Model

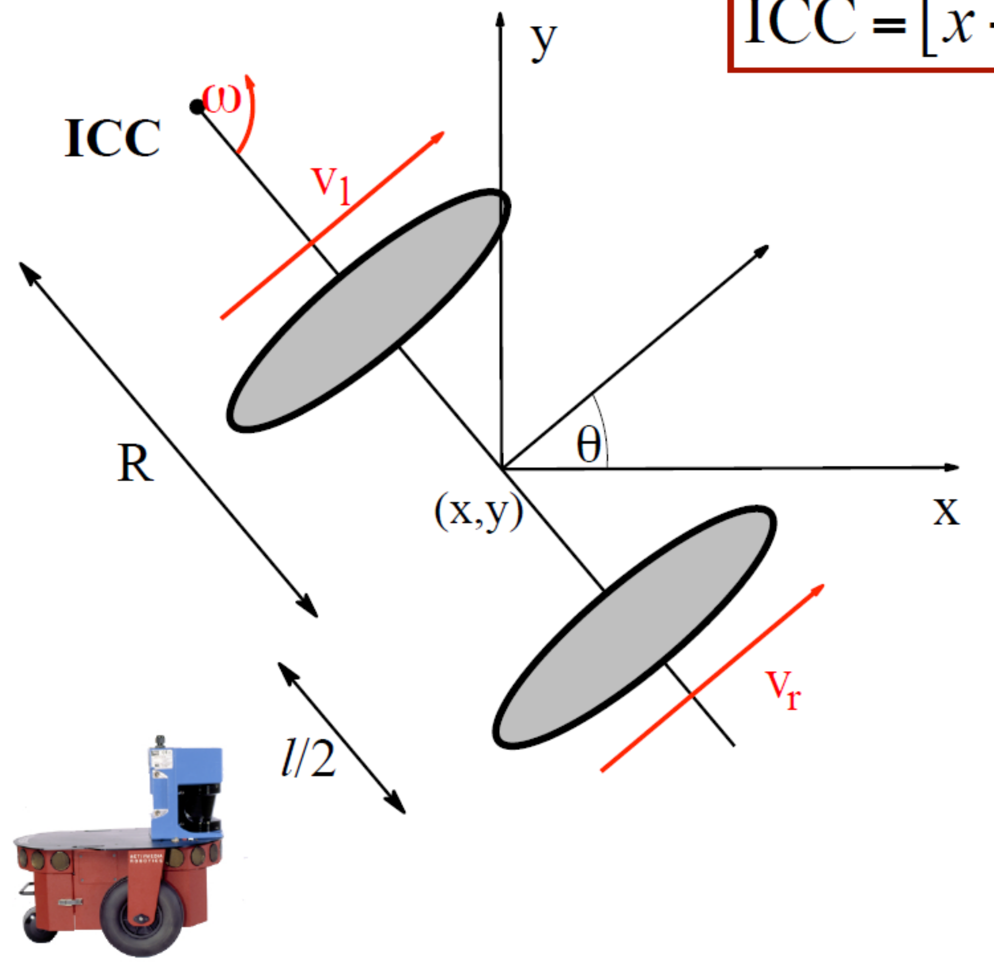
- Robot moves from $\langle \bar{x}, \bar{y}, \bar{\theta} \rangle$ to $\langle \bar{x}', \bar{y}', \bar{\theta}' \rangle$.
- Odometry information $u = \langle \delta_{rot1}, \delta_{rot2}, \delta_{trans} \rangle$.

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



$$\mathbf{X}' = f(x, y, \theta, \Delta s_r, \Delta s_l) = \mathbf{X} + \begin{bmatrix} \frac{\Delta s_r + \Delta s_l}{2} \cos\left(\theta + \frac{\Delta s_r - \Delta s_l}{2l}\right) \\ \frac{\Delta s_r + \Delta s_l}{2} \sin\left(\theta + \frac{\Delta s_r - \Delta s_l}{2l}\right) \\ \frac{\Delta s_r - \Delta s}{l} \end{bmatrix}$$

Differential Drive



$$\boxed{ICC = [x - R\sin\theta, y + R\cos\theta]}$$

$$\omega(R + l/2) = v_r$$
$$\omega(R - l/2) = v_l$$
$$R = \frac{l}{2}\frac{(v_l + v_r)}{(v_r - v_l)}$$
$$\omega = \frac{v_r - v_l}{l}$$
$$v = \frac{v_r + v_l}{2}$$

Noise Model for Odometry

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

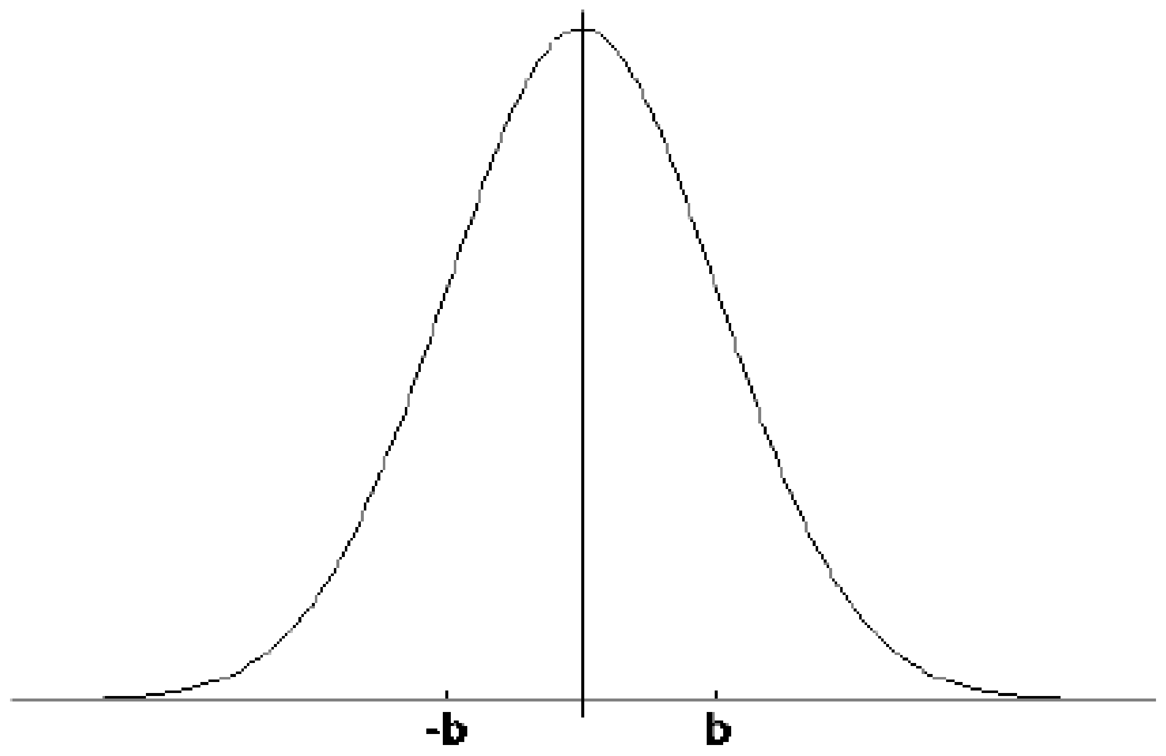
$$\hat{\delta}_{rot1} = \delta_{rot1} + \varepsilon_{\alpha_1 |\delta_{rot1}| + \alpha_2 |\delta_{trans}|}$$

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

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

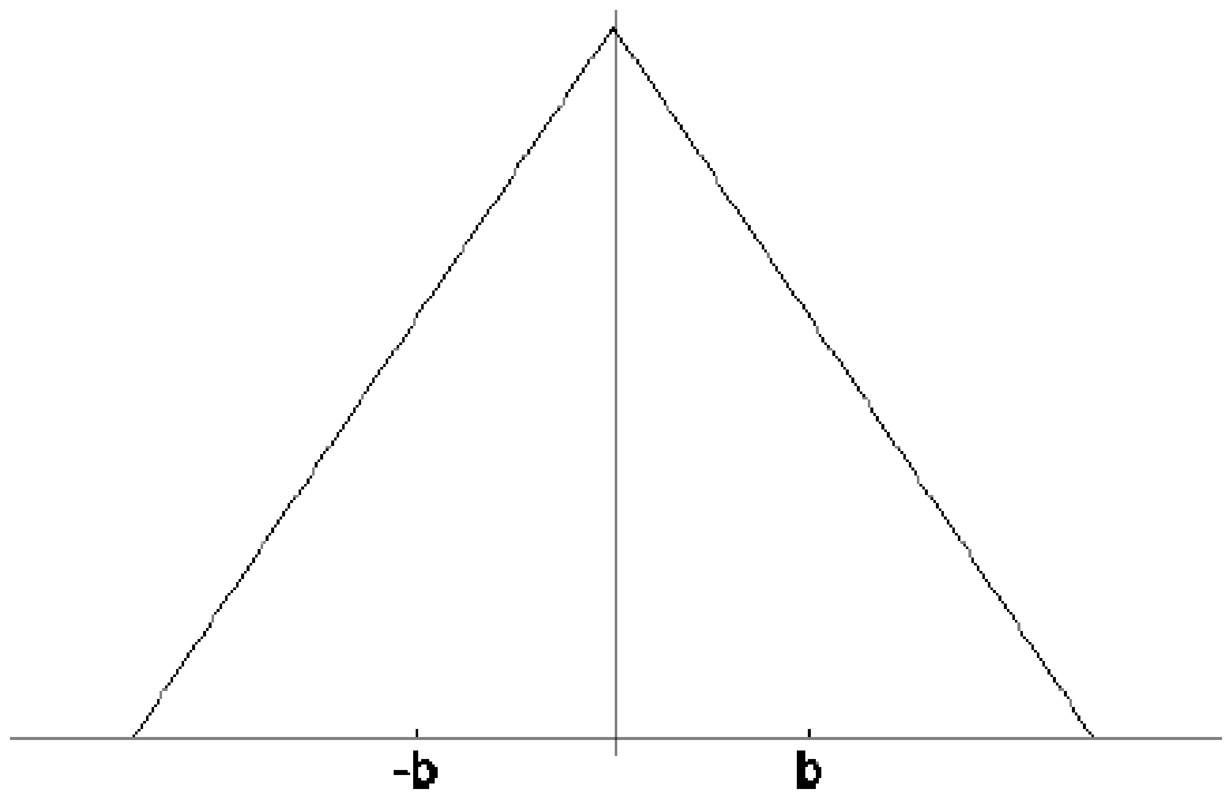
Typical Distributions for Probabilistic Motion Models

Normal distribution



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

Triangular distribution



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

Calculating the Posterior Given x, x', and Odometry

odometry

Relative motion information as measured by the robot's internal odometry.

1. Algorithm **motion_model_odometry**$(x, x', \boxed{\bar{x}, \bar{x}'})$
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. $\hat{\delta}_{rot1} = \text{atan2}(y'-y, x'-x) - \bar{\theta}$
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 $p_1 \cdot p_2 \cdot p_3$

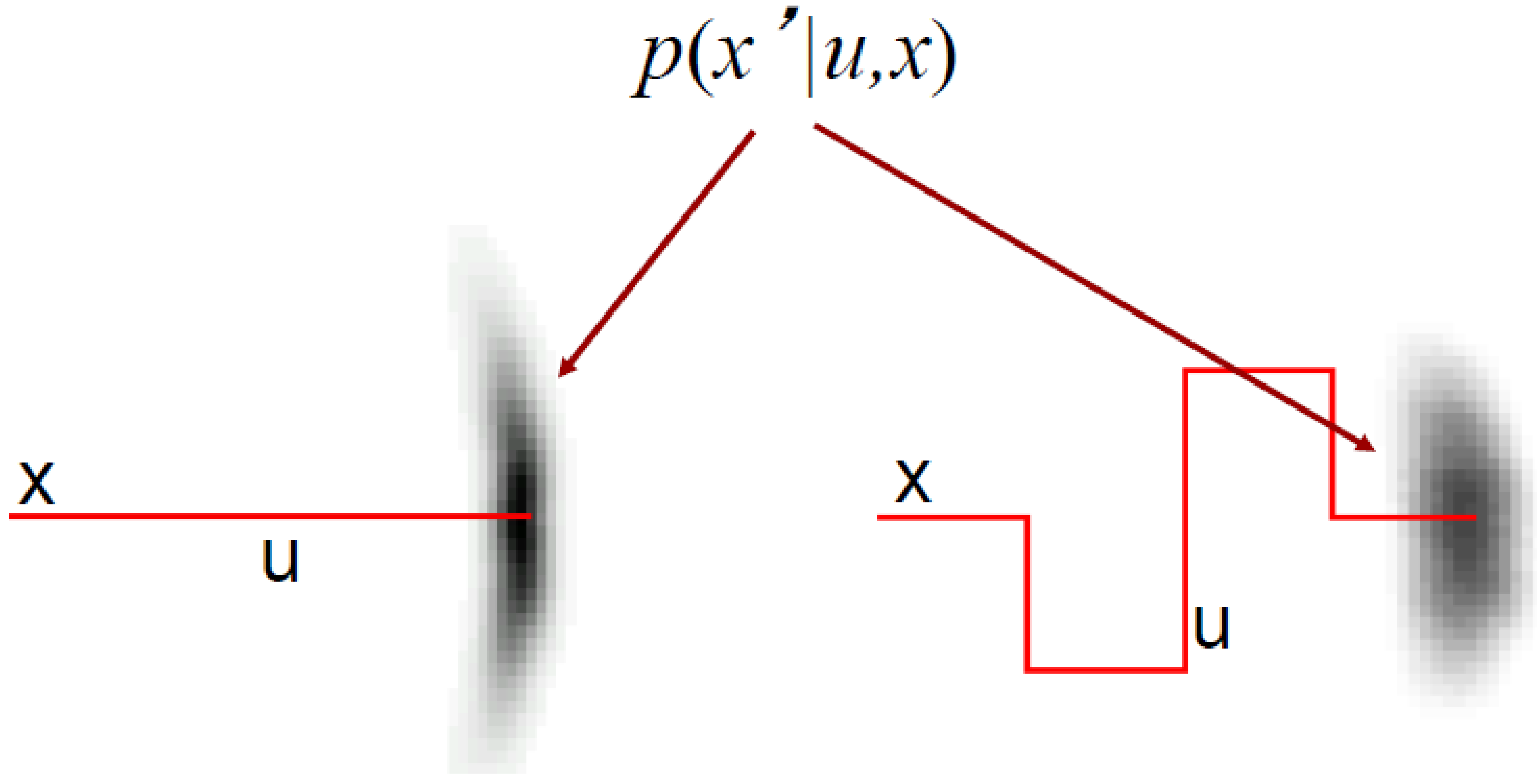
odometry params (u)

values of interest (x,x')

Errors are assumed to be independent!

Application

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



Sample Odometry Motion Model

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

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

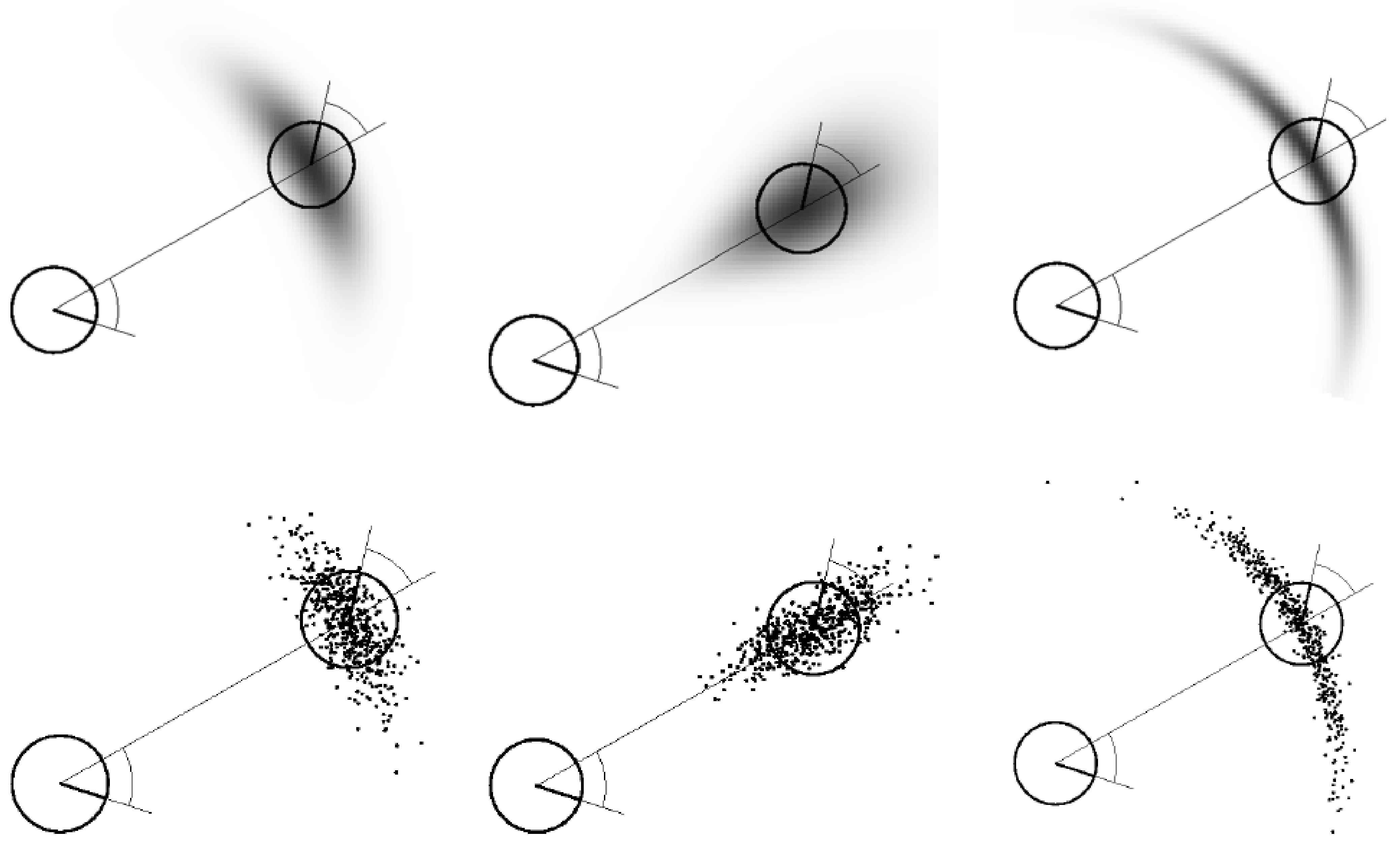
1. $\hat{\delta}_{rot1} = \delta_{rot1} + \text{sample}(\alpha_1 |\delta_{rot1}| + \alpha_2 \delta_{trans})$
2. $\hat{\delta}_{trans} = \delta_{trans} + \text{sample}(\alpha_3 \delta_{trans} + \alpha_4 (|\delta_{rot1}| + |\delta_{rot2}|))$
3. $\hat{\delta}_{rot2} = \delta_{rot2} + \text{sample}(\alpha_1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

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

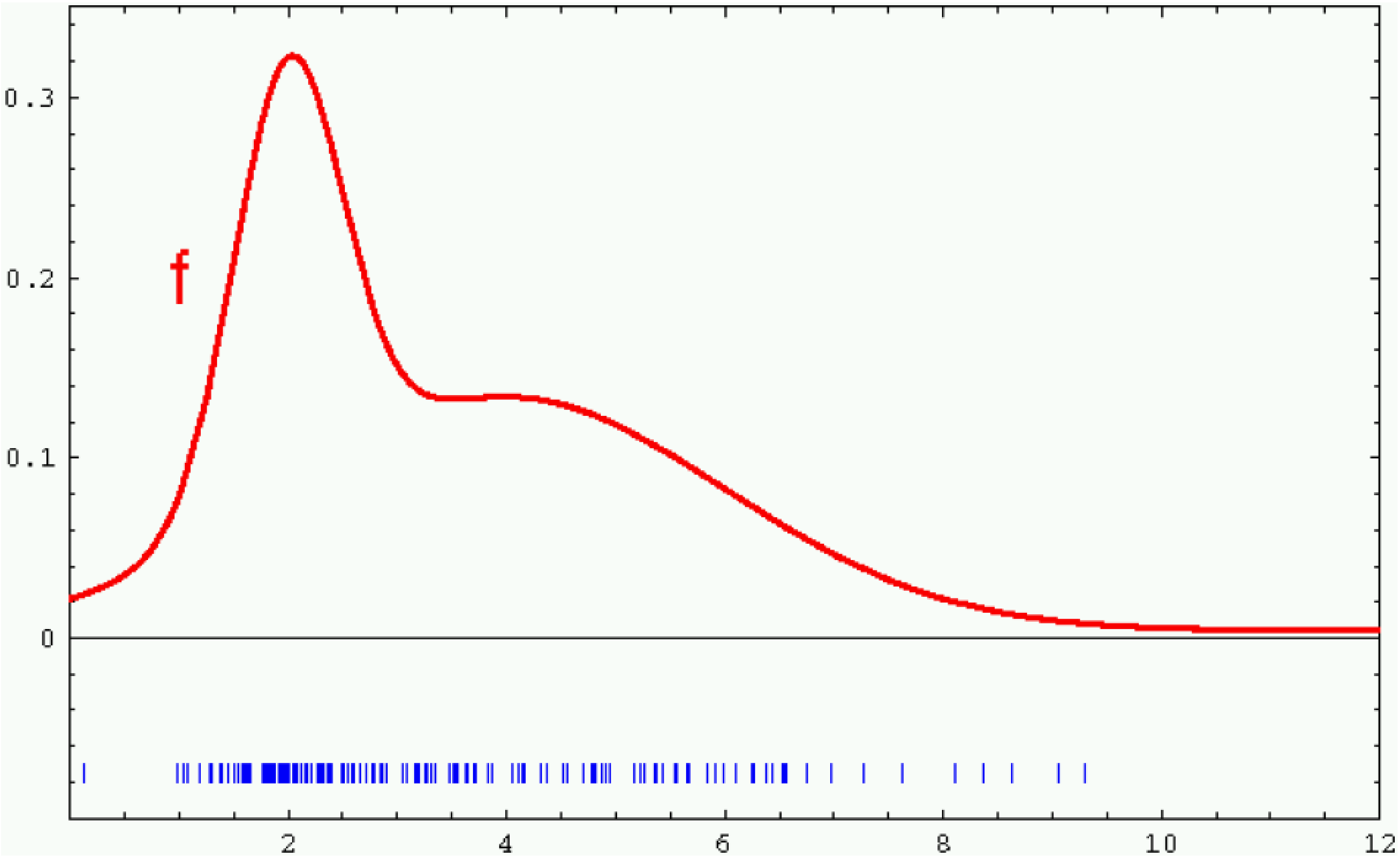
7. Return $\langle x', y', \theta' \rangle$

sample_normal_distribution

Examples (Odometry-Based)



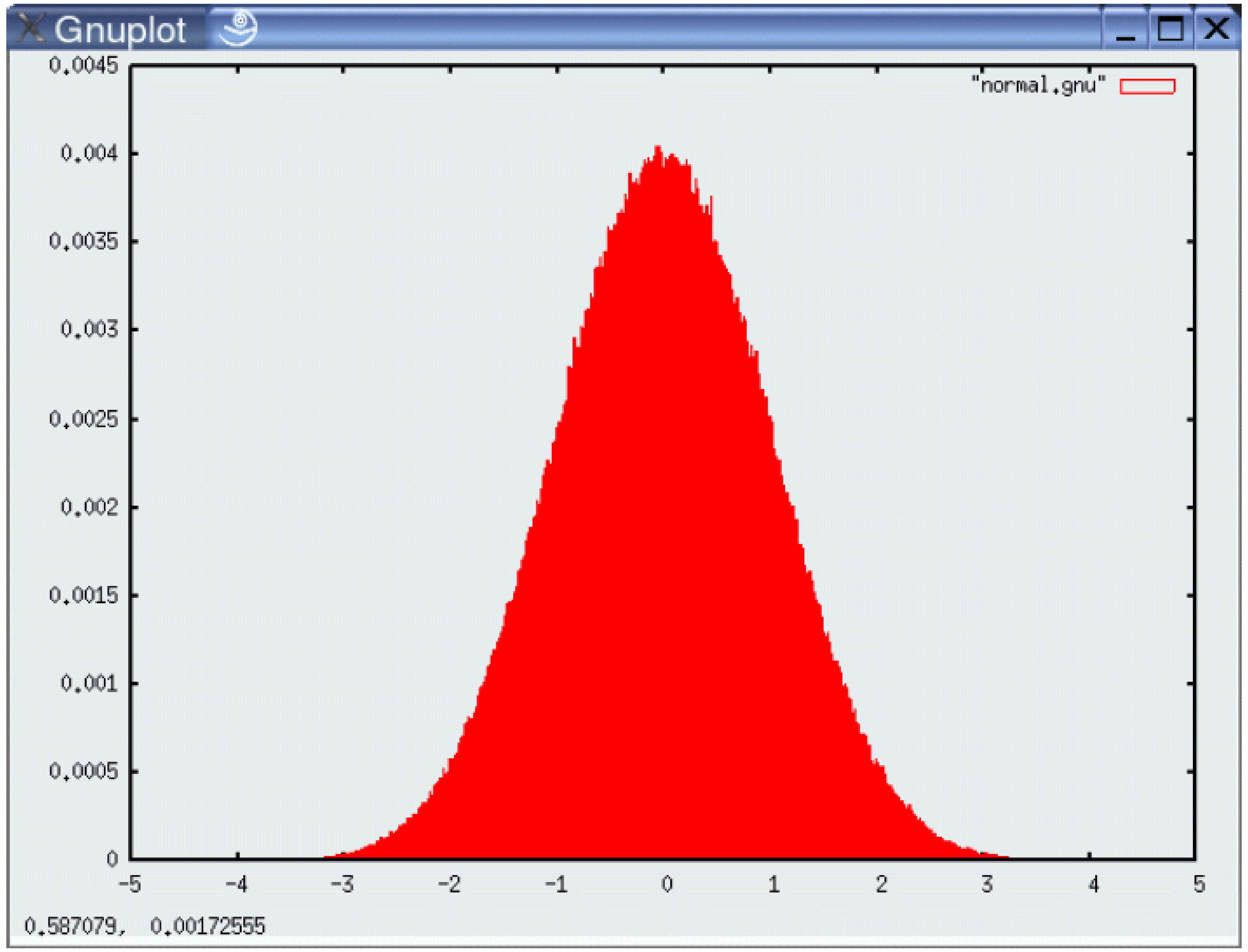
Sample-Based Density Representation



How to Sample from Normal Distributions?

- Sampling from a normal distribution

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



10^6 samples

C 또는 Matlab등 대부분의 컴퓨터 언어는 (0, 1) 구간의 값을 균일한 확률분포로 취하는 랜덤 변수를 만드는 루틴을 제공하고 있다. 이러한 루틴을 사용하여 얻어지는 랜덤변수로부터 정규 분포(가우시안 분포)를 갖는 랜덤 변수를 다음과 같이 만들 수 있다.

ξ 를 (0, 1)구간의 균일분포 랜덤변수라고 하자. ξ 의 평균값과 분산은 다음과 같이 구해진다.

$$E[\xi] = \frac{1}{2}$$

$$E[(\xi - \frac{1}{2})^2] = \frac{1}{12}$$

다음으로

$$\xi^* = \sum_{i=1}^{12} \xi_{k+1} - 6 \tag{25}$$

과 같이 12개의 랜덤 변수 값으로부터 새로운 랜덤변수 ξ^* 를 정의하자. 그러면, 확률분포가 어떻든 간에 많은 수의 랜덤변수가 더해지면 정규분포에 가까워진다는 Central Limit Theorem에 의해 ξ^* 의 확률분포는 정규분포에 가깝게 됨을 보일 수 있으며, 이때의 평균값은 0이고 분산은 1이 된다. 즉,

$$\xi^* \sim N(0,1)$$

이 된다.

How to Sample from Normal or Triangular Distributions?

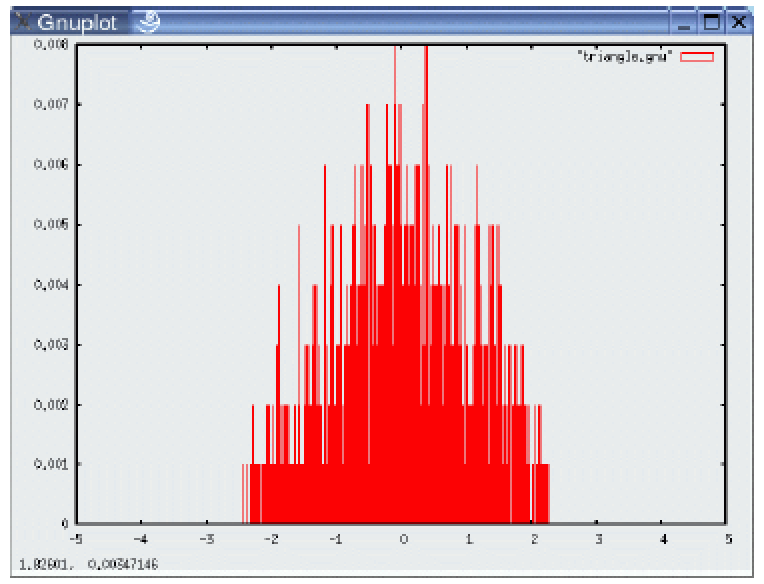
- Sampling from a normal distribution

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

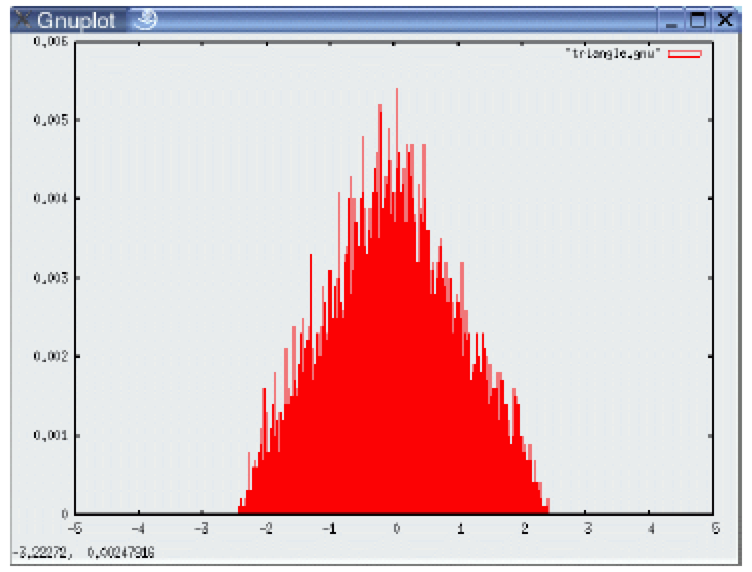
- Sampling from a triangular distribution

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

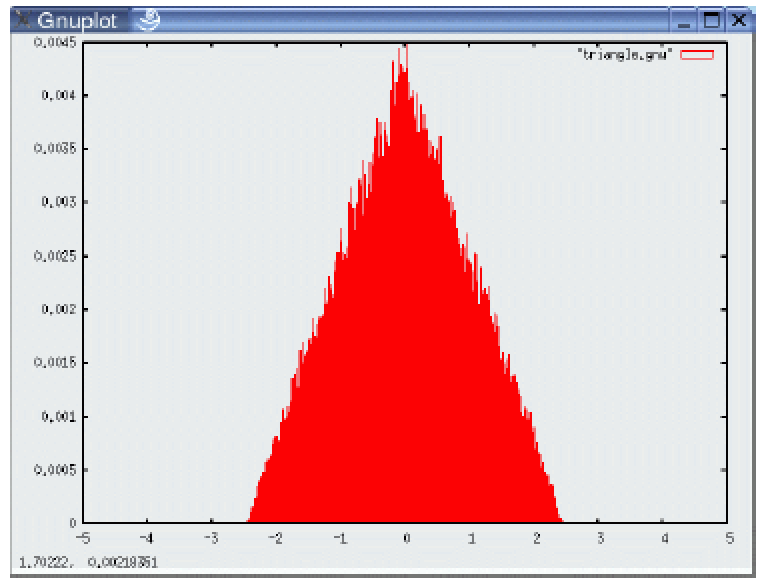
For Triangular Distribution



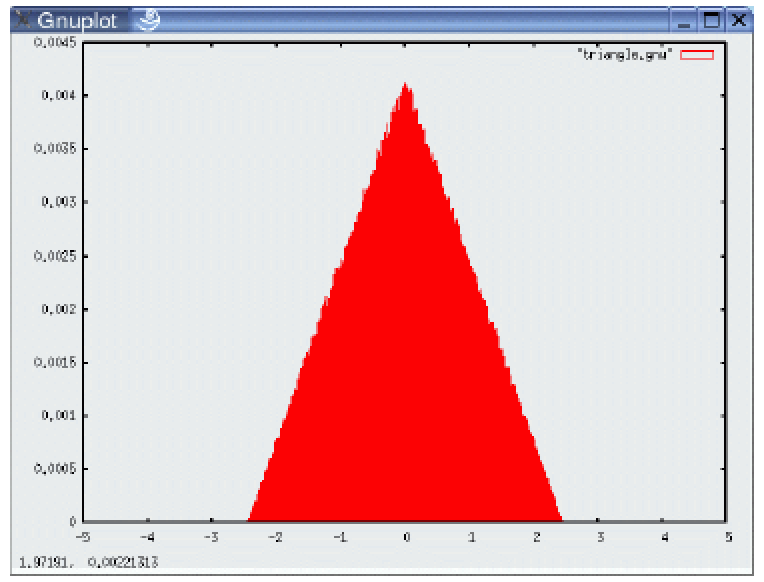
10^3 samples



10^4 samples

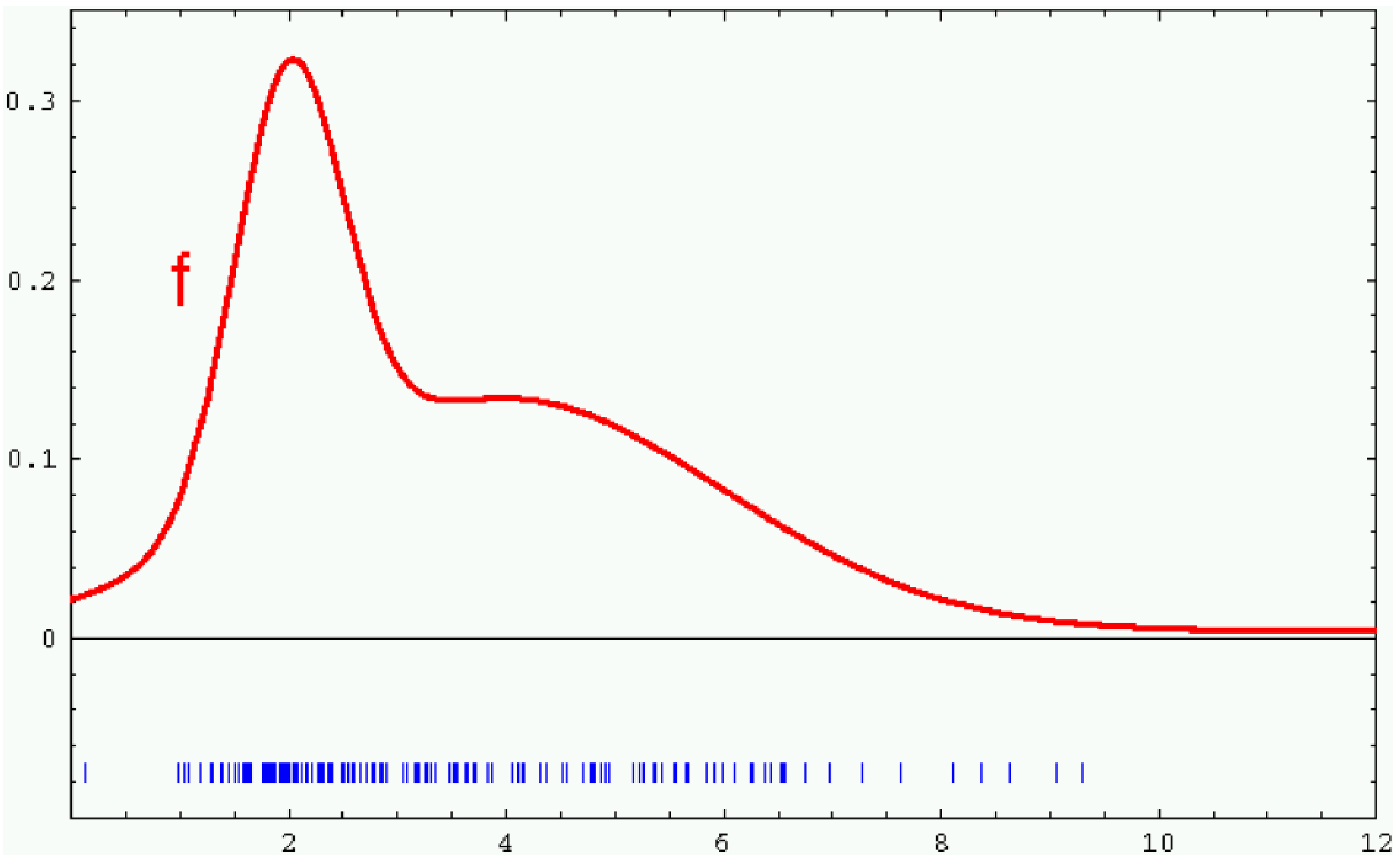


10^5 samples



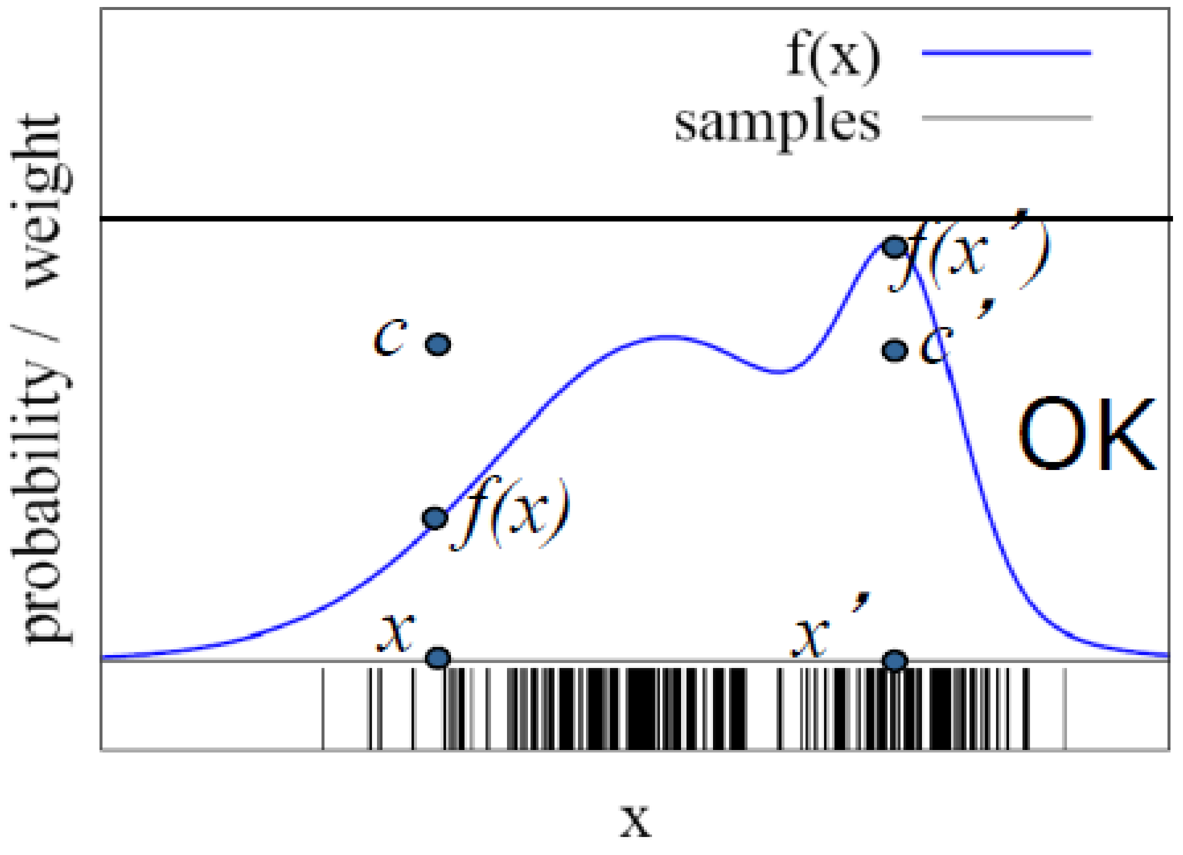
10^6 samples

How to Obtain Sample from Arbitrary Functions?

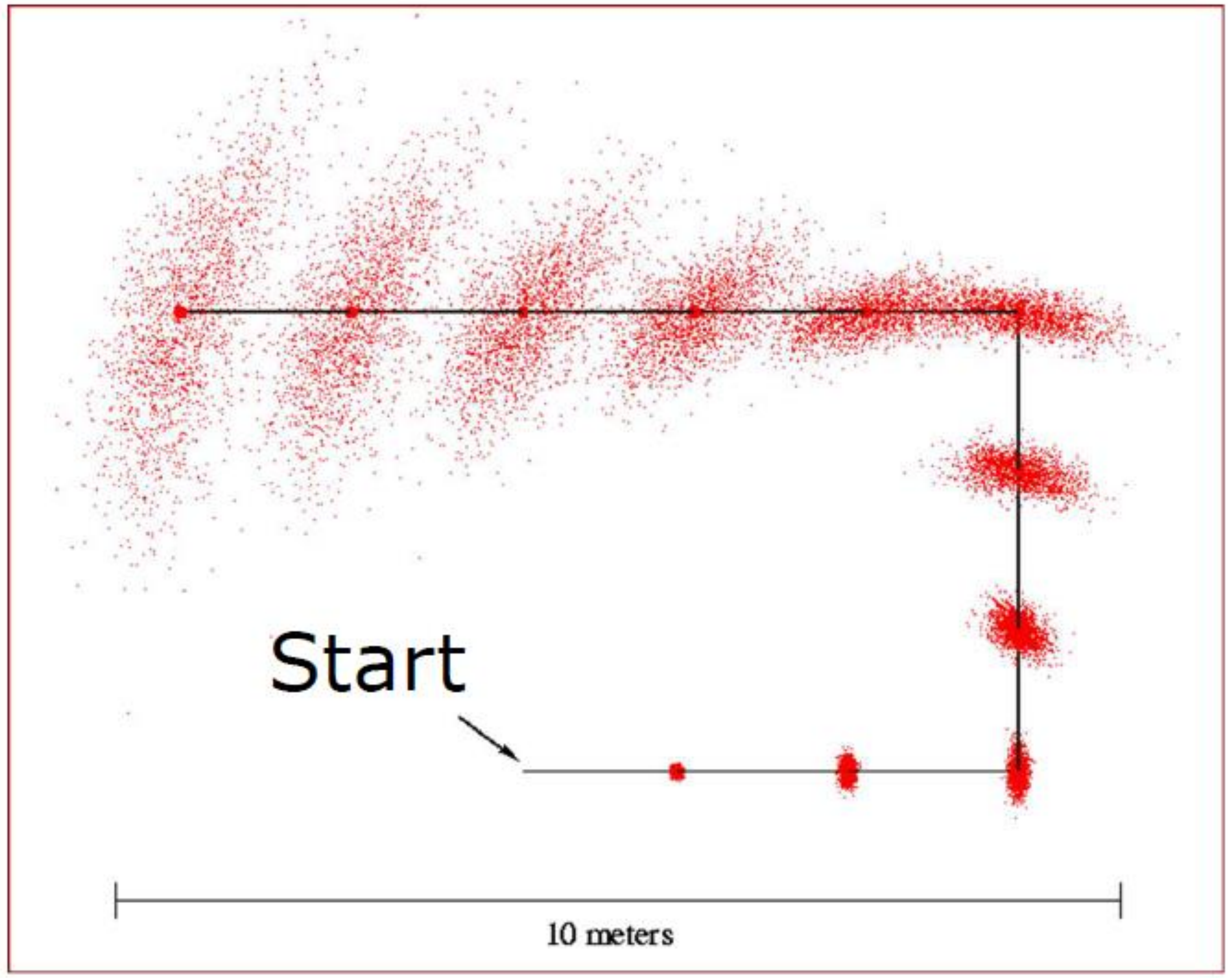


Rejection Sampling

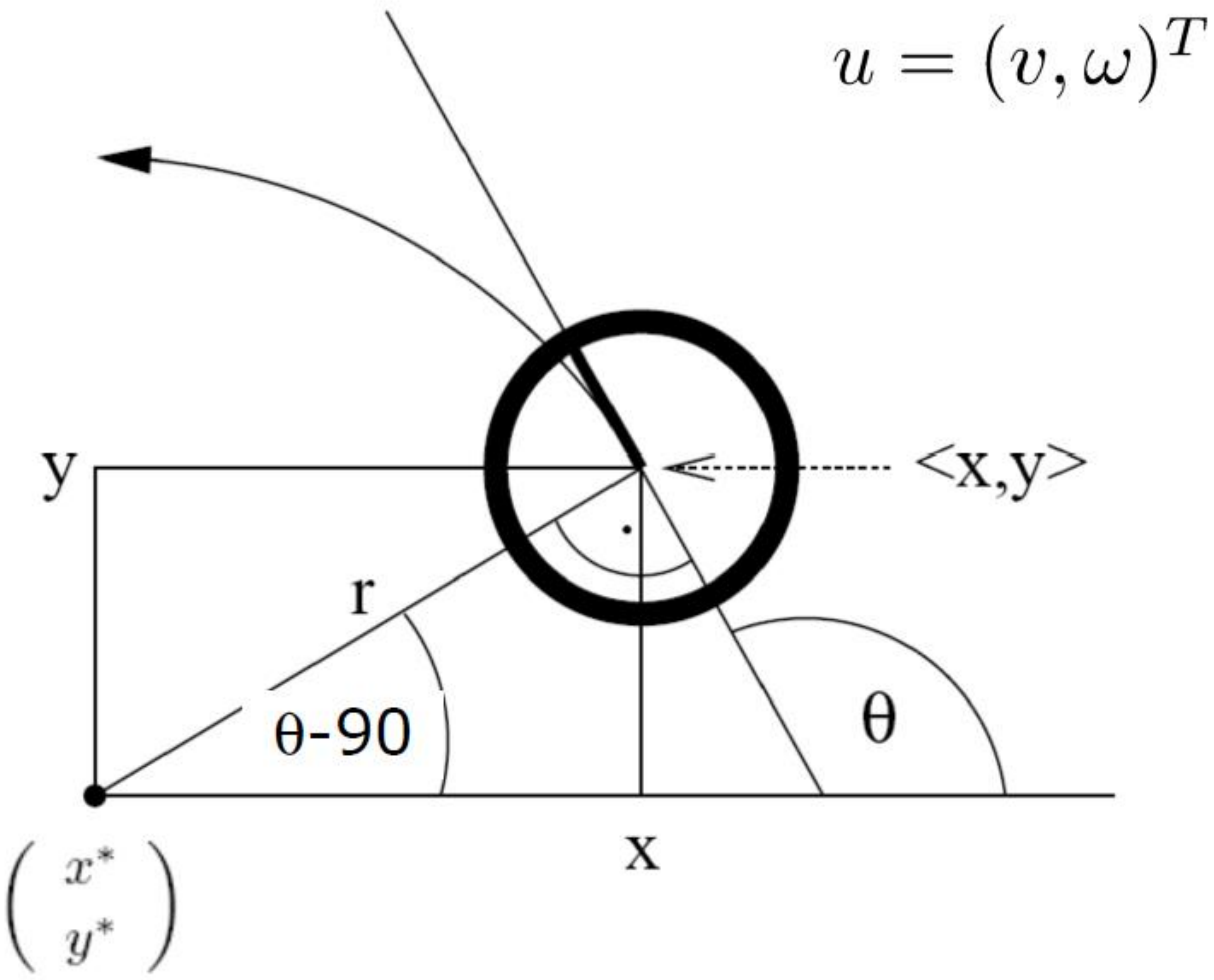
- Sampling from arbitrary distributions
- Sample x from a uniform distribution from $[-b,b]$
- Sample c from $[0, max\ f]$
- if $f(x) > c$ keep the sample
  otherwise reject the sample



Sampling from Our Motion Model



Velocity-Based Model



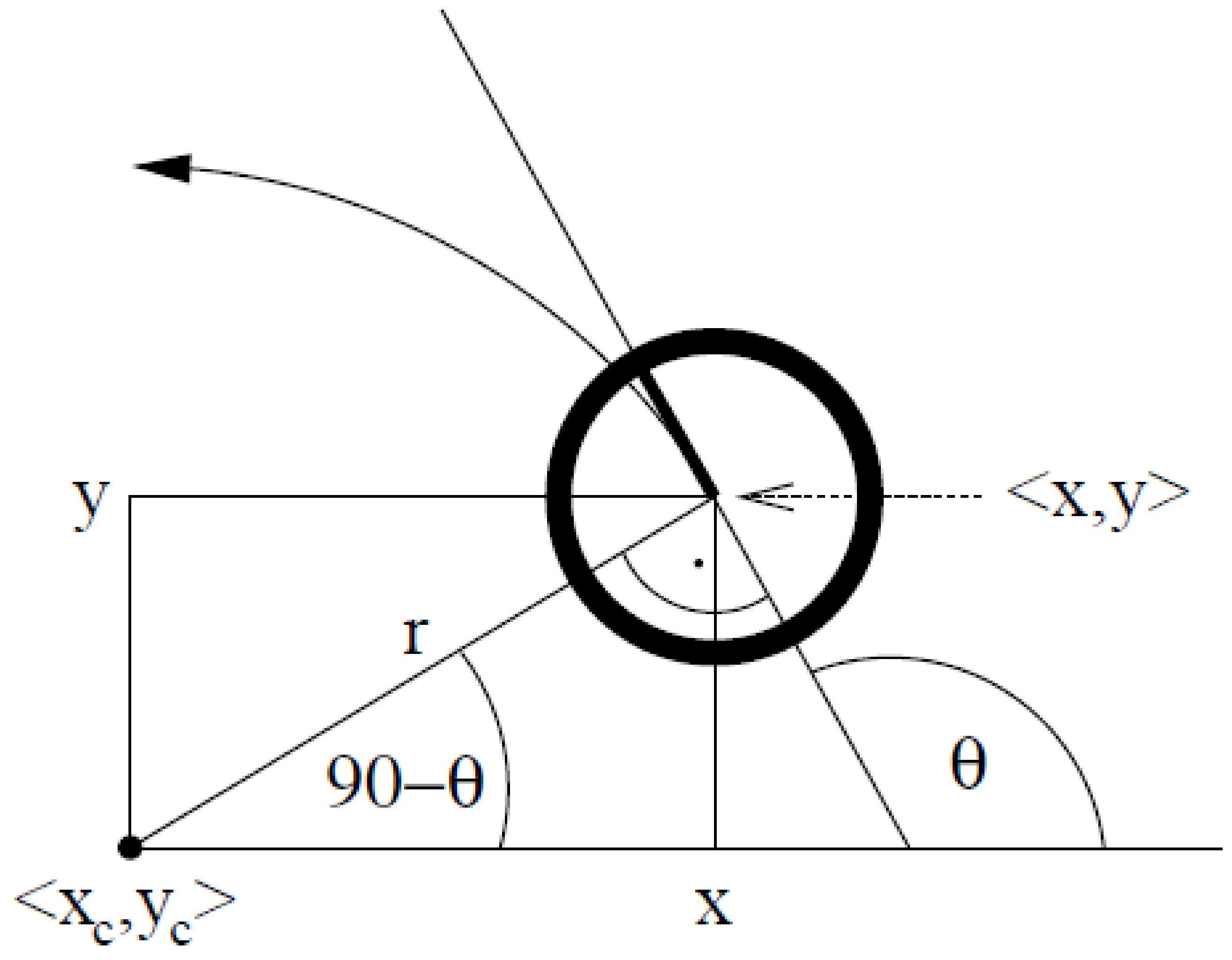
Dead Reckoning

- 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.
- Historically used to log the position of ships.

Let $x_{t-1} = (x, y, \theta)^T$ be the initial pose of the robot, and suppose we keep the velocity constant at $(v \;\; \omega)^T$ for some time Δt. As one easily shows, the center of the circle is at

$$x_c = x - \frac{v}{\omega} \sin\theta \tag{5.7}$$

$$y_c = y + \frac{v}{\omega} \cos\theta \tag{5.8}$$



The variables $(x_c \;\; y_c)^T$ denote this coordinate. After Δt time of motion, our ideal robot will be at $x_t = (x', y', \theta')^T$ with

$$\begin{aligned} \begin{pmatrix} x' \\ y' \\ \theta' \end{pmatrix} &= \begin{pmatrix} x_c + \frac{v}{\omega} \sin(\theta + \omega\Delta t) \\ y_c - \frac{v}{\omega} \cos(\theta + \omega\Delta t) \\ \theta + \omega\Delta t \end{pmatrix} \\ &= \begin{pmatrix} x \\ y \\ \theta \end{pmatrix} + \begin{pmatrix} -\frac{v}{\omega} \sin\theta + \frac{v}{\omega} \sin(\theta + \omega\Delta t) \\ \frac{v}{\omega} \cos\theta - \frac{v}{\omega} \cos(\theta + \omega\Delta t) \\ \omega\Delta t \end{pmatrix} \end{aligned} \tag{5.9}$$

Noise Model for the Velocity-Based Model

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

$$
\boxed{
\begin{aligned}
\hat{v} &= v + \varepsilon_{\alpha_1 |v| + \alpha_2 |\omega|} \\
\hat{\omega} &= \omega + \varepsilon_{\alpha_3 |v| + \alpha_4 |\omega|}
\end{aligned}
}
$$

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

Noise Model for the Velocity-Based Model

- The 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

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

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

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

Term to account for the final rotation

Posterior Probability for Velocity Model

Algorithm motion_model_velocity(x_t, u_t, x_{t-1}): $p(x_t \mid x_{t-1}, u_t)$

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

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

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

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

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

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

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

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

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

Sampling from Velocity Model

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

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

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

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

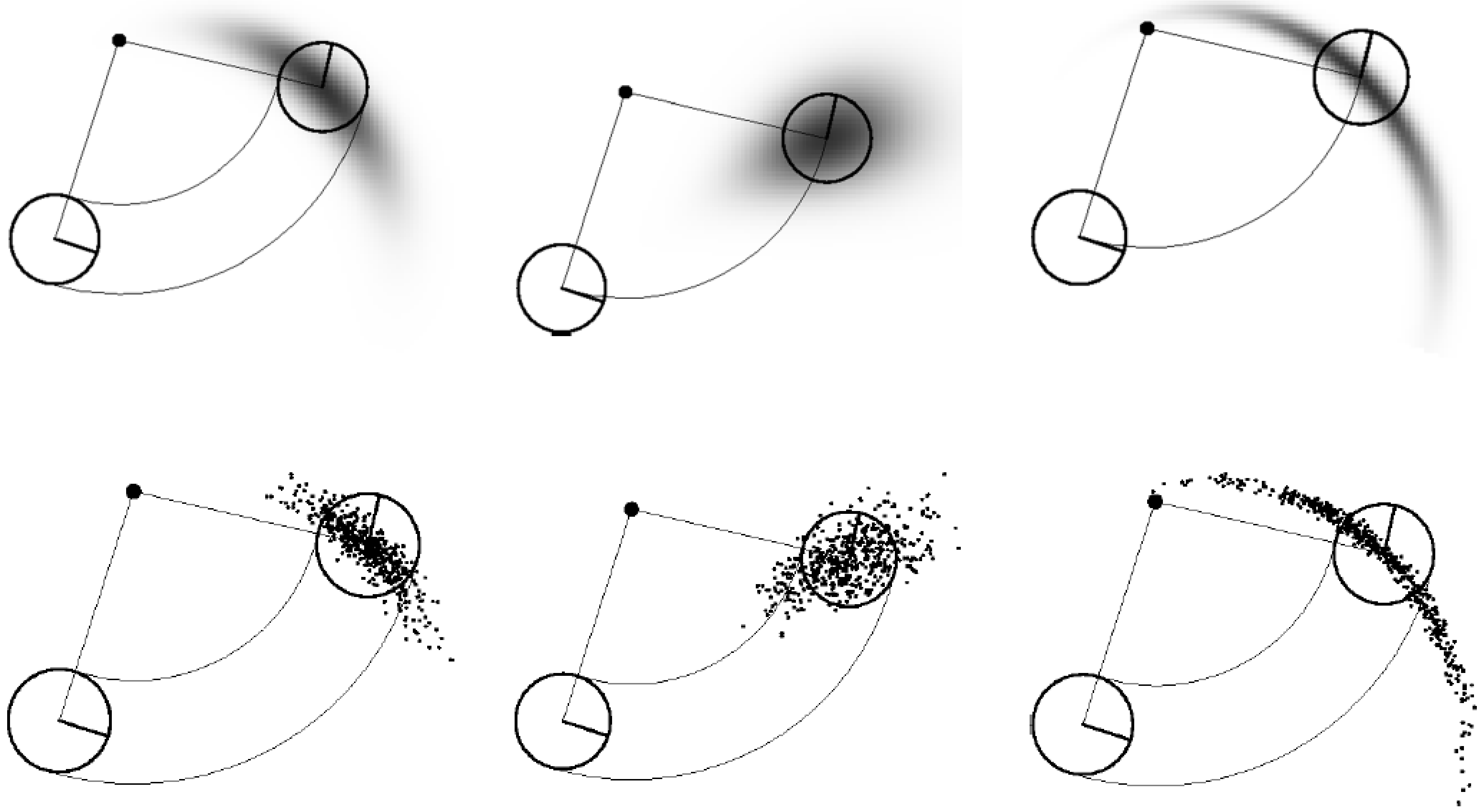
$x' = x - \frac{\hat{v}}{\hat{\omega}} \sin\theta + \frac{\hat{v}}{\hat{\omega}} \, sin(\theta + \hat{\omega}\Delta t)$

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

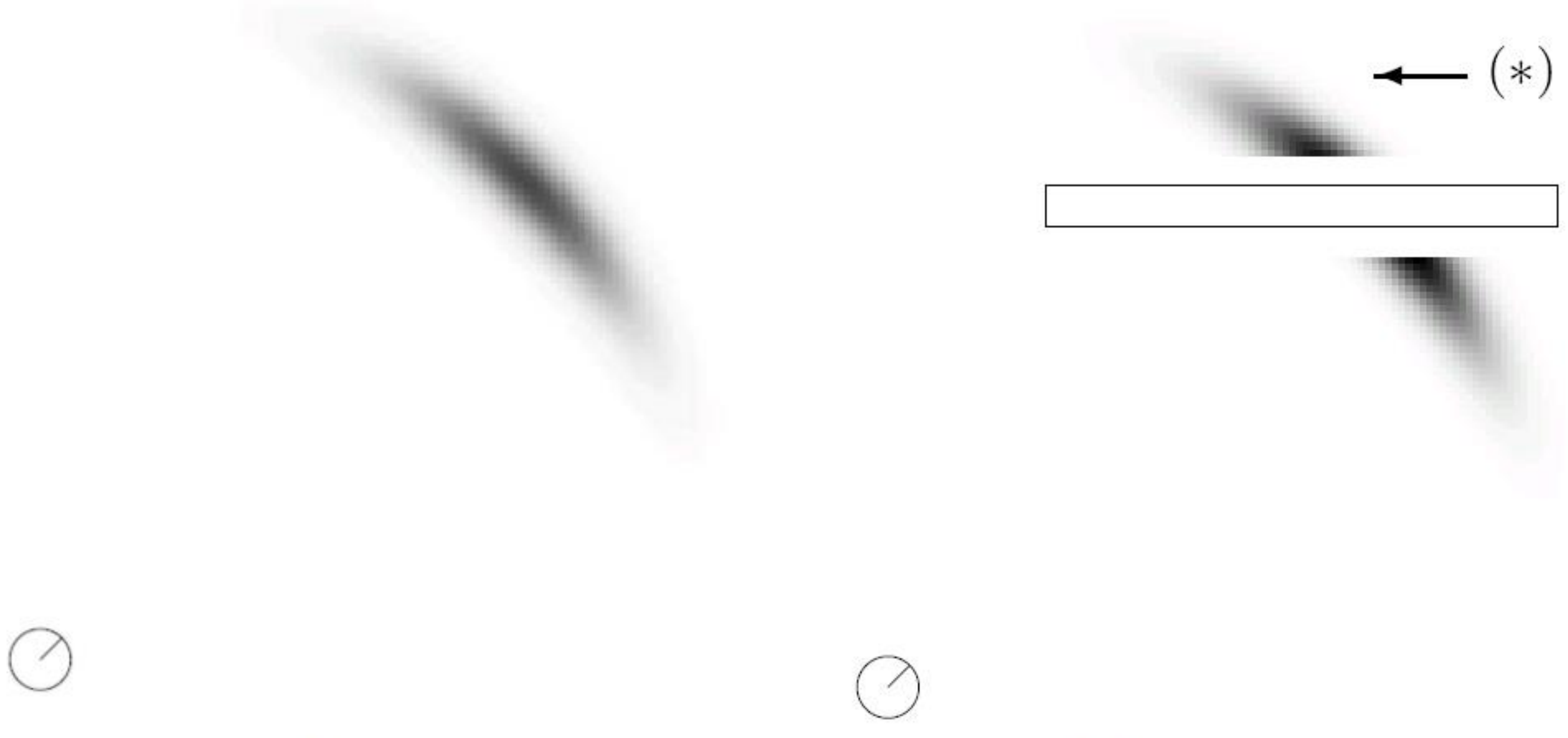
$\theta' = \theta + \hat{\omega}\Delta t + \hat{\gamma}\Delta t$

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

Examples (Velocity-Based)



Map-Consistent Motion Model



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

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

Summary

- We discussed motion models for odometry-based and velocity-based systems
- We discussed ways to calculate the posterior probability $p(x' \mid x, u)$.
- We also described how to sample from $p(x' \mid x, u)$.
- 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.

ULSAN NATIONAL INSTITUTE OF SCIENCE AND TECHNOLOGY
2009

THANK YOU