Introduction to Mobile Robotics and Robot Operating System (ROS)

Lecture 3. Kinematic models of wheeled robots.

Probabilistic motion models

Oleg Shipitko,

March 2021

girafe ai

Outline

1. Kinematic models of wheeled robots
    a. Differential drive
    b. Tricycle
    c. Ackermann principle
    d. Omni- and mecanum-wheels
2. Probabilistic motion models
    a. Odometry-based model
    b. Speed control based model

RECURSIVE BAYESIAN POSE ESTIMATION

$$p(\mathbf{x_t}|map, \mathbf{z_t}, \mathbf{u_t}) = C \cdot p(\mathbf{z_t}|\mathbf{x_t}, map) \int_S p(\mathbf{x_t}|\mathbf{u_t}, \mathbf{x_{t-1}}) p(\mathbf{x_{t-1}}|map, \mathbf{z_{t-1}}, \mathbf{u_{t-1}}) d\mathbf{x_{t-1}}$$

C — normalization coefficient

S — the probabilistic space of robot poses

$p(\mathbf{z_t}|\mathbf{x_t}, map)$ — observation (measurement) model

$p(\mathbf{x_t}|\mathbf{u_t}, \mathbf{x_{t-1}})$ — motion model

$p(\mathbf{x_{t-1}}|map, \mathbf{z_{t-1}}, \mathbf{u_{t-1}})$ — previous system state (robot pose)

KALMAN FILTER

Prediction:

$$\hat{\mathbf{x}}_t = \mathbf{F}_t \hat{\mathbf{x}}_{t-1} + \mathbf{B}_k \vec{\mathbf{u}_t}$$

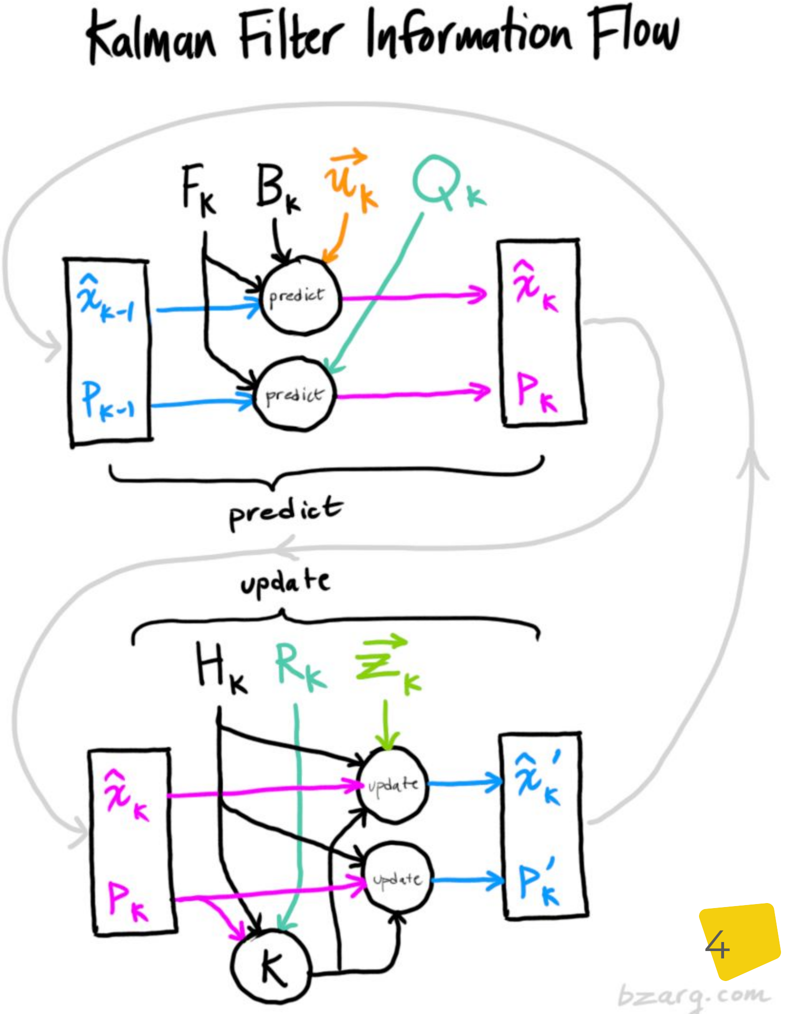
$$\hat{\Sigma}_t = \mathbf{F}_t \Sigma_{t-1} \mathbf{F}_t^T + \mathbf{Q}_t$$

Correction:

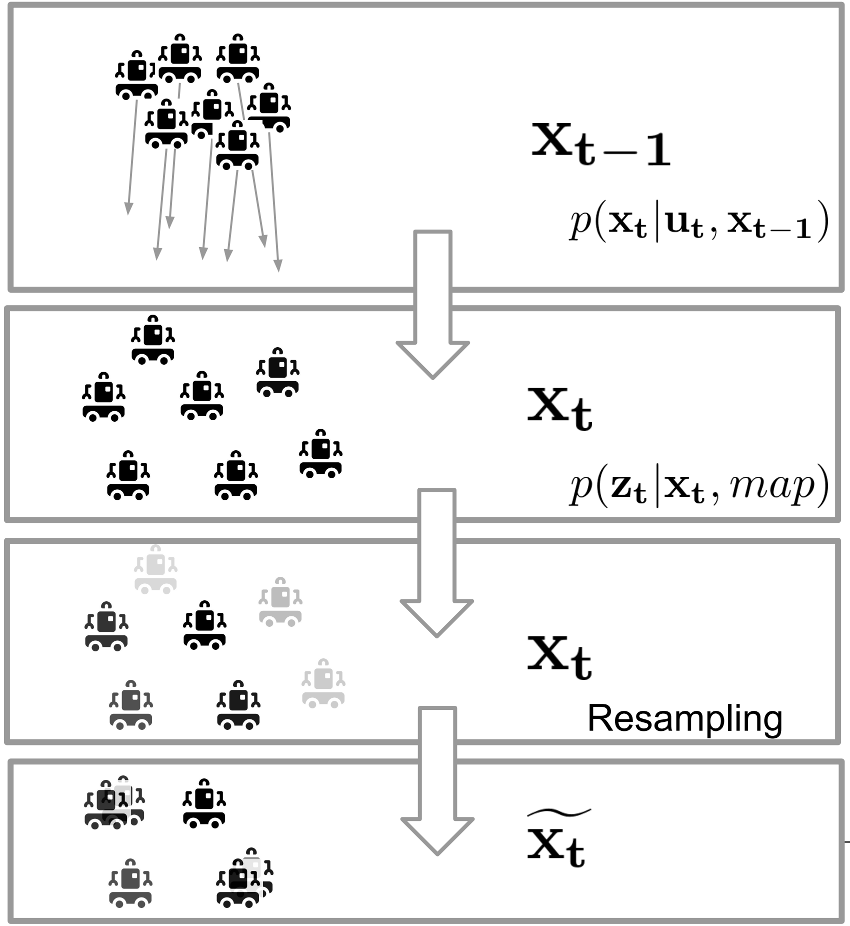
$$\mathbf{K}' = \hat{\Sigma}_t \mathbf{H}_t^T (\mathbf{H}_t \hat{\Sigma}_t \mathbf{H}_t^T + \mathbf{R}_t)^{-1}$$

$$\mathbf{x}'_t = \mathbf{H}_t \hat{\mathbf{x}}_t + \mathbf{K}' (\vec{\mathbf{z}_t} - \mathbf{H}_t \hat{\mathbf{x}}_t)$$

$$\Sigma'_t = \hat{\Sigma}_t - \mathbf{K}' \mathbf{H}_t \hat{\Sigma}_t$$



PARTICLE FILTER



Algorithm 1 Generic Monte-Carlo localization algorithm

procedure MCL($\mathbf{x_{t-1}}, m, \mathbf{u_t}, \mathbf{z_t}$)
  $\{\mathbf{x_t^n}\} = \{\widetilde{\mathbf{x_t^n}}\} = \emptyset$
  for $n = 1$ to N do
    sample $x_t^n \sim p(\mathbf{x_t}|\mathbf{u_t}, \mathbf{x_{t-1}^n})$ — Motion model
    $w_t^n = p(\mathbf{z_t}|\mathbf{x_t^n}, map)$ — Observation model
    $\{\widetilde{\mathbf{x_t^n}}\} = \{\widetilde{\mathbf{x_t^n}}\} + \langle x_t^n, w_t^n \rangle$
  end for
  for $n = 1$ to N do — Resampling
    draw i with probability $\propto \widetilde{w_t^i}$
    $\{\mathbf{x_t^n}\} = \{\mathbf{x_t^n}\} + \langle \widetilde{x_t^i}, \widetilde{w_t^i} \rangle$
  end for
  return $\{\mathbf{x_t^n}\}$
end procedure

RECURSIVE BAYESIAN POSE ESTIMATION

$$p(\mathbf{x_t}|map, \mathbf{z_t}, \mathbf{u_t}) = C \cdot p(\mathbf{z_t}|\mathbf{x_t}, map) \int_S p(\mathbf{x_t}|\mathbf{u_t}, \mathbf{x_{t-1}}) p(\mathbf{x_{t-1}}|map, \mathbf{z_{t-1}}, \mathbf{u_{t-1}}) d\mathbf{x_{t-1}}$$

C — normalization coefficient

S — the probabilistic space of robot poses

$p(\mathbf{z_t}|\mathbf{x_t}, map)$ — observation (measurement) model

$p(\mathbf{x_t}|\mathbf{u_t}, \mathbf{x_{t-1}})$ — motion model

$p(\mathbf{x_{t-1}}|map, \mathbf{z_{t-1}}, \mathbf{u_{t-1}})$ — previous system state (robot pose)

RECURSIVE BAYESIAN POSE ESTIMATION

$$p(\mathbf{x_t}|map, \mathbf{z_t}, \mathbf{u_t}) = C \cdot p(\mathbf{z_t}|\mathbf{x_t}, map) \int_S p(\mathbf{x_t}|\mathbf{u_t}, \mathbf{x_{t-1}}) p(\mathbf{x_{t-1}}|map, \mathbf{z_{t-1}}, \mathbf{u_{t-1}}) d\mathbf{x_{t-1}}$$

C — normalization coefficient

S — the probabilistic space of robot poses

$p(\mathbf{z_t}|\mathbf{x_t}, map)$ — observation (measurement) model

$p(\mathbf{x_t}|\mathbf{u_t}, \mathbf{x_{t-1}})$ — motion model

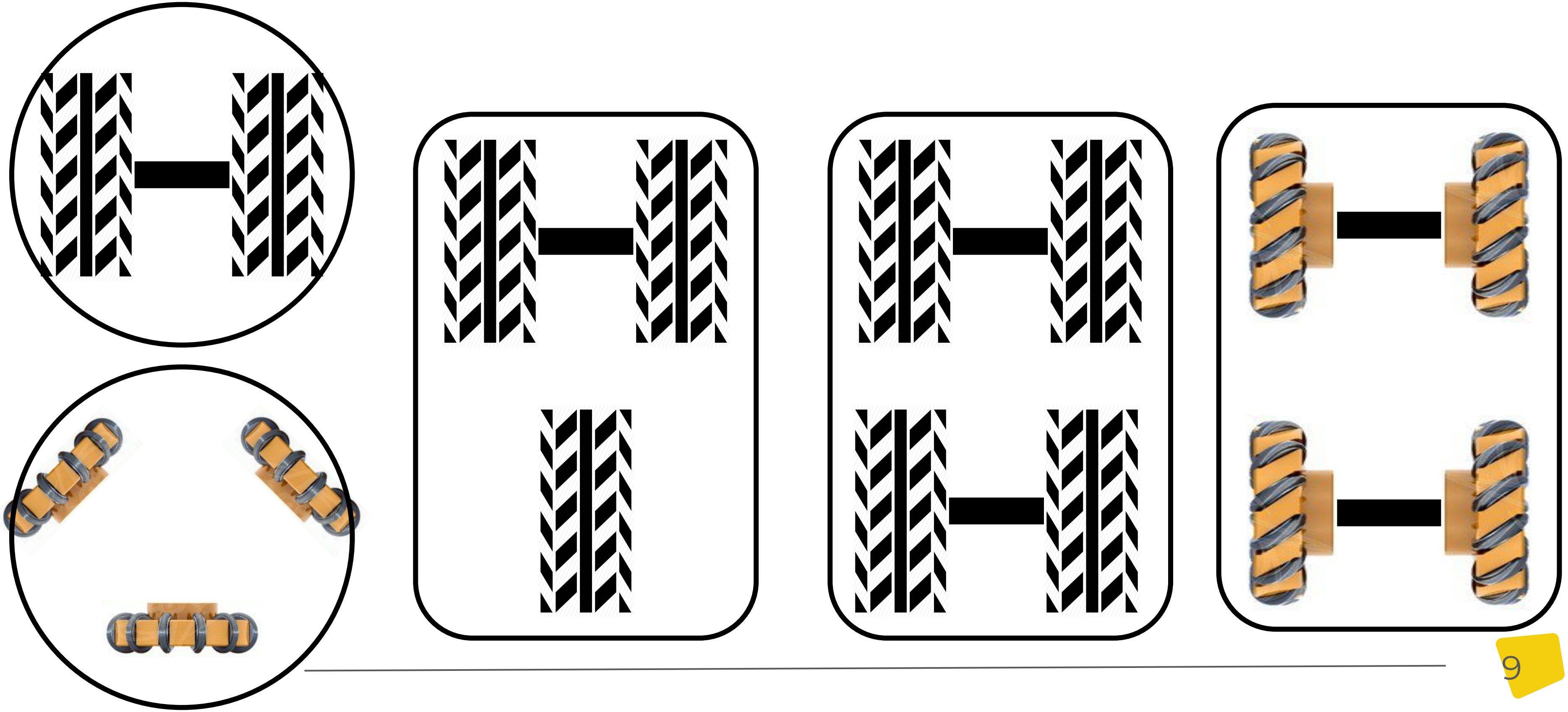
$p(\mathbf{x_{t-1}}|map, \mathbf{z_{t-1}}, \mathbf{u_{t-1}})$ — previous system state (robot pose)

Kinematic models of wheeled robots

girafe
ai

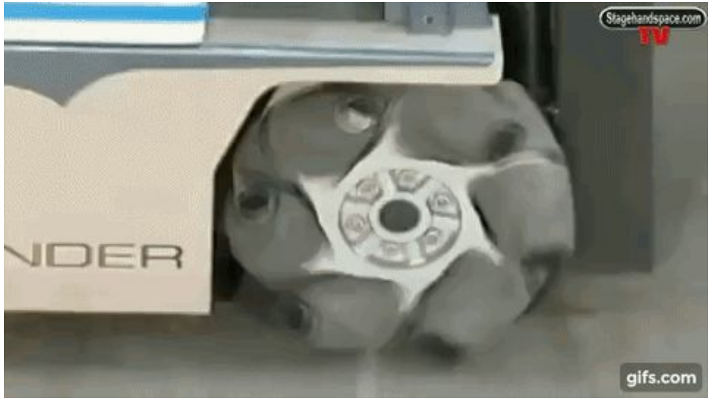
01

(some) TYPES OF WHEELED KINEMATICS

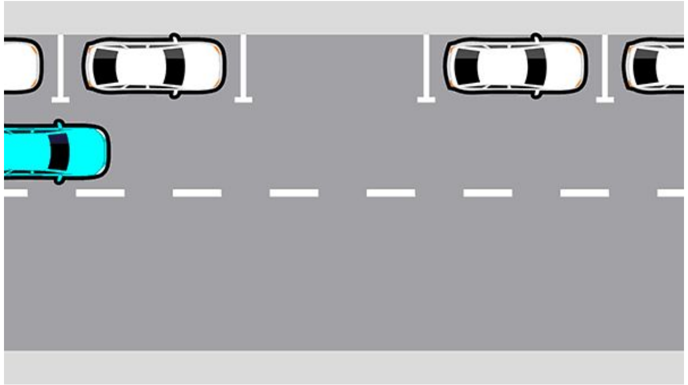


HOLONOMIC SYSTEMS

- A robot is called **holonomic** if the number of **controlled** degrees of freedom = the **total** number of degrees of freedom.



- A **nonholonomic system** is a mechanical system on which, in addition to geometric ones, kinematic constraints are also superimposed.
- Mathematically, nonholonomic constraints are expressed by non-integrable equations.



HOLONOMIC SYSTEMS

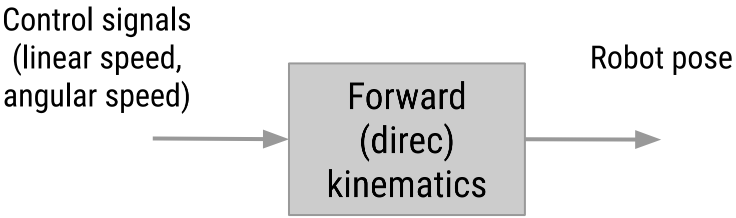
- **Holonomic constraints** limit the allowed state space (geometry).
- For instance, if there is a truck and a trailer, **not all angles** between them **are possible**. This is a **holonomic constraint**.

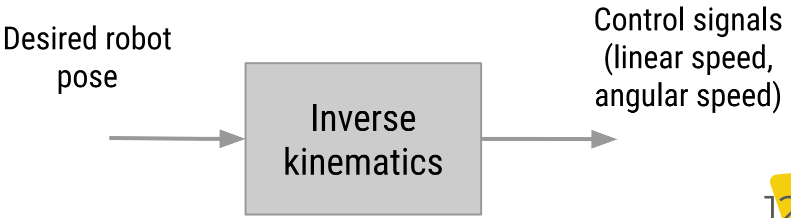
- **Nonholonomic constraints** limit the control space relative to the current state.
- For instance, a car **can not move sideway**.

DIRECT AND INVERSE KINEMATICS PROBLEMS

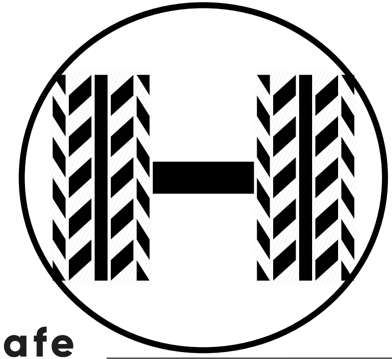
- The **direct kinematics problem** — having control parameters (for example, wheel speeds) and motion time, find the position into which the robot has moved.

- The **inverse kinematics problem** is to find the control parameters that move the robot into a given position in a given time.





DIFFERENTIAL DRIVE

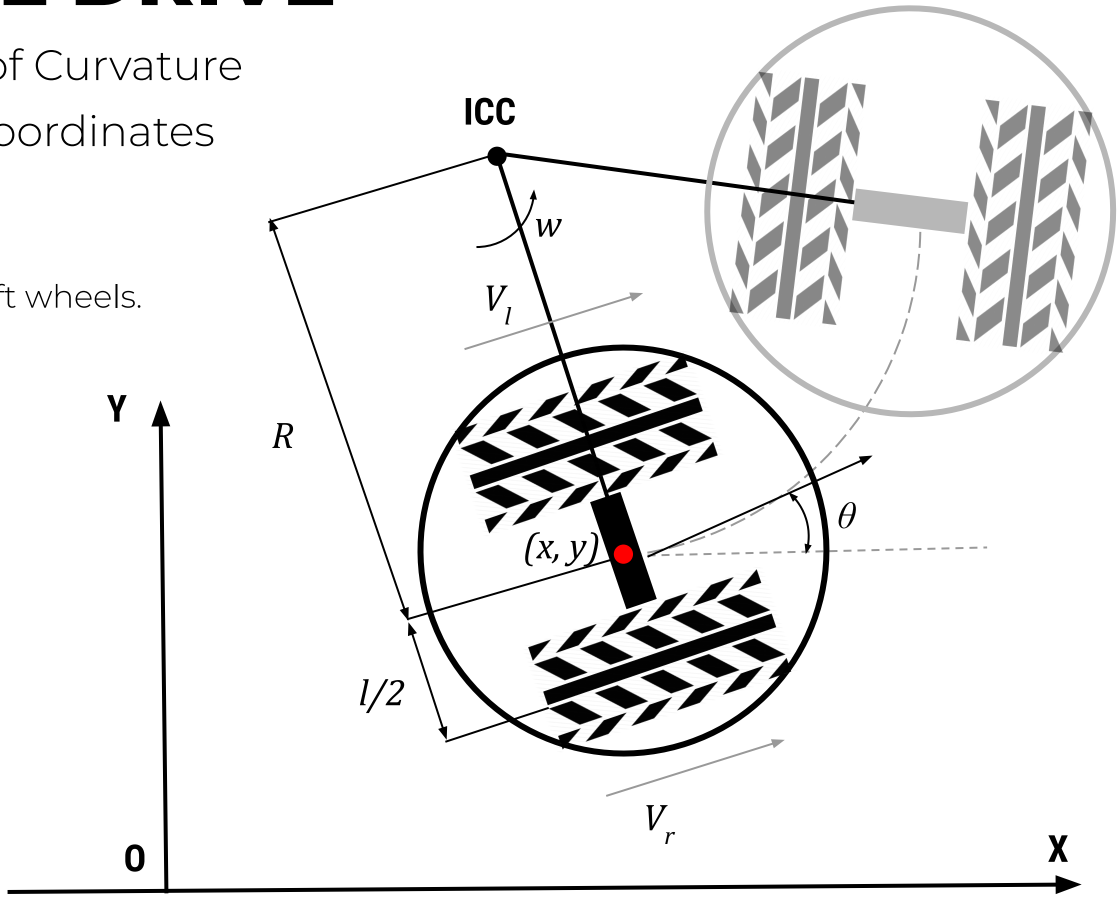


DIFFERENTIAL DRIVE

ICC – Instantaneous Center of Curvature

(x, y, θ) – wheel axle center coordinates

$\left.\begin{matrix} V_r \\ V_l \end{matrix}\right\}$ – the speed of the right and left wheels. Controlled parameters.



DIFFERENTIAL DRIVE

ICC – Instantaneous Center of Curvature

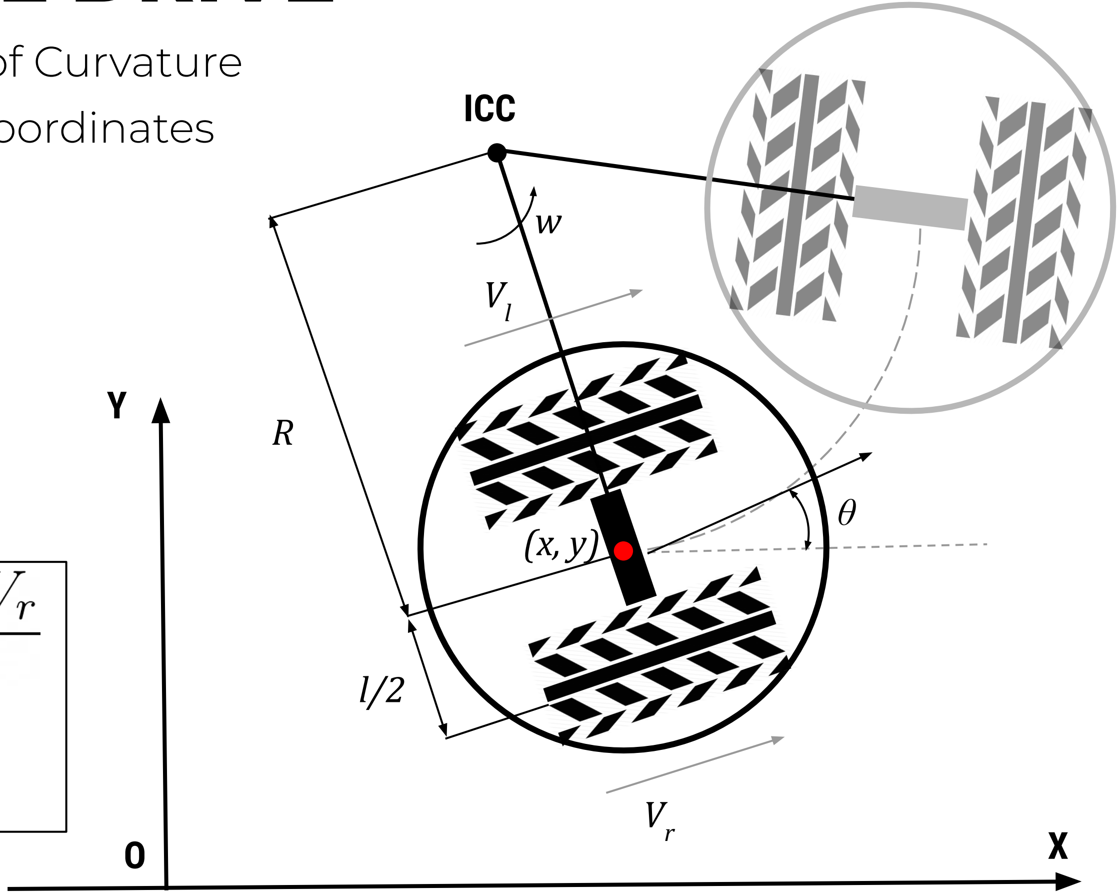
(x, y, θ) – wheel axle center coordinates

$$w(R + \frac{l}{2}) = V_r$$

$$w(R - \frac{l}{2}) = V_l$$

$$w = \frac{V_r - V_l}{l} \qquad V = \frac{V_l + V_r}{2}$$

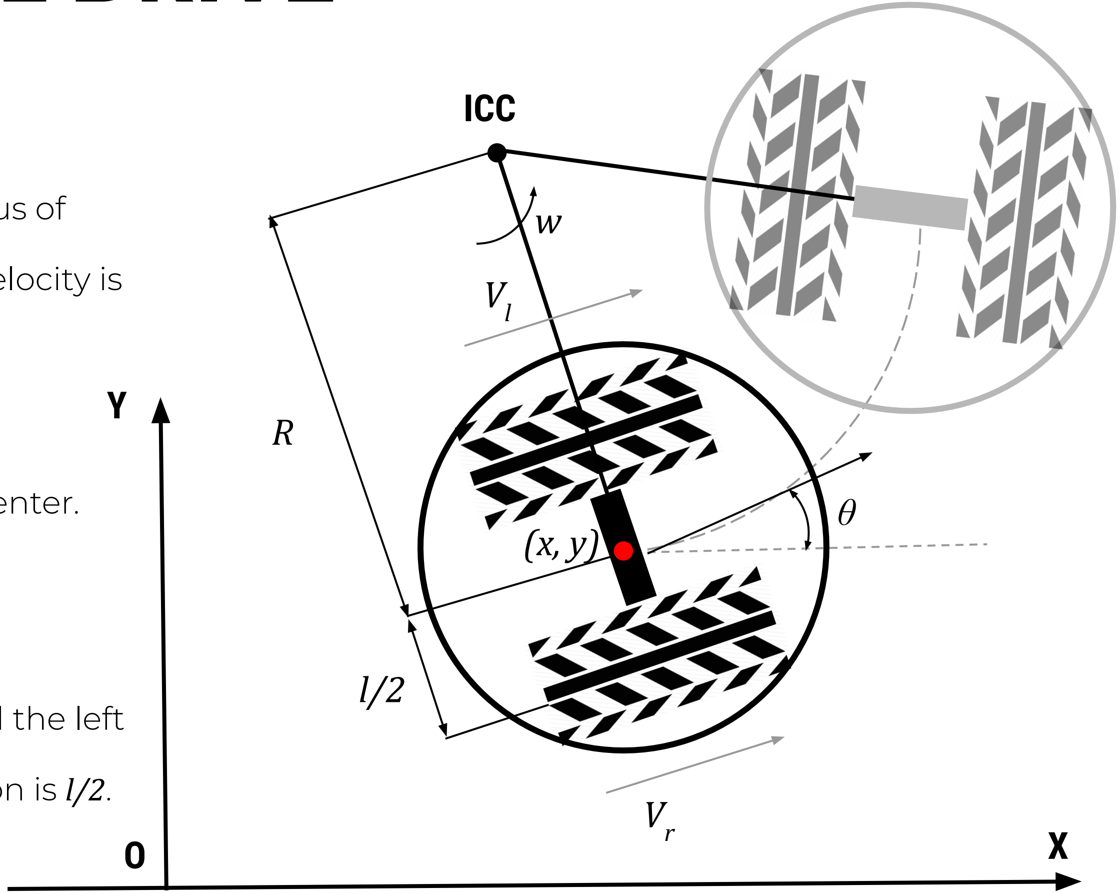
$$R = \frac{l}{2}\frac{V_r + V_l}{V_r - V_l}$$



DIFFERENTIAL DRIVE

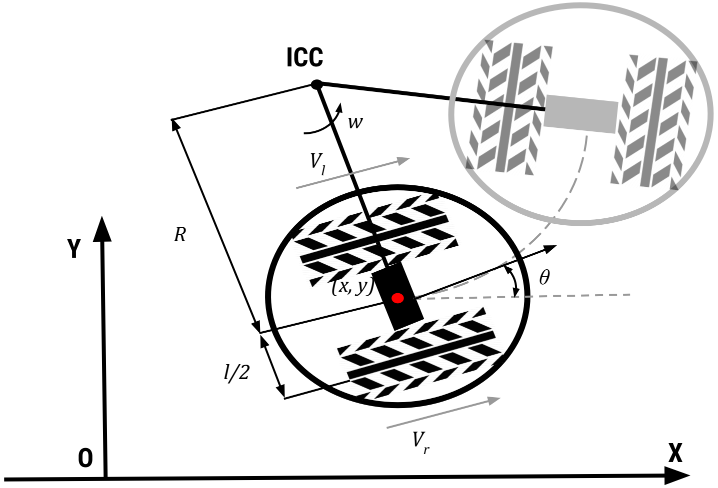
3 types of motion:

- ❑ $V_l = V_r$ — linear motion. The radius of rotation is **infinity**. The angular velocity is **zero**.
- ❑ $V_l = -V_r$ — rotation around the center. The radius of rotation is zero.
- ❑ $V_l = 0$ ($V_r = 0$) — rotation around the left (right) wheel. The radius of rotation is $l/2$.



DIFFERENTIAL DRIVE FORWARD KINEMATICS

$$ICC = [x - R\sin(\theta), y + R\cos(\theta)]$$

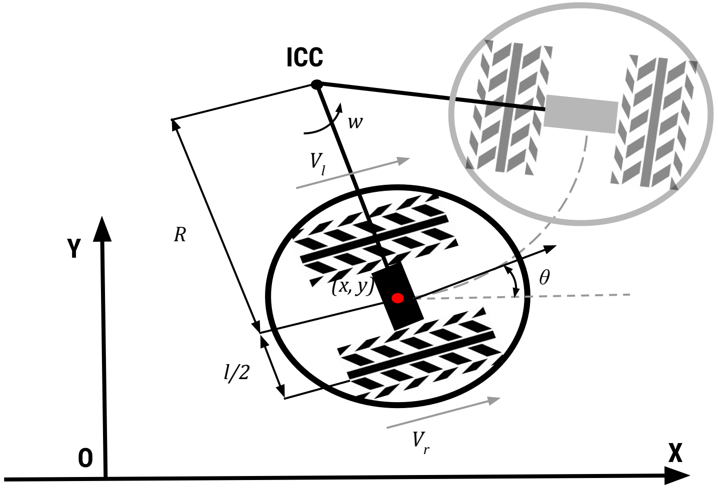


At the time moment **t+δt** the robot pose is defined as:

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

DIFFERENTIAL DRIVE FORWARD KINEMATICS

$$x(t) = \int_0^t V(t)cos[\theta(t)]dt$$
$$y(t) = \int_0^t V(t)sin[\theta(t)]dt$$
$$\Theta(t) = \int_0^t \omega(t)dt$$

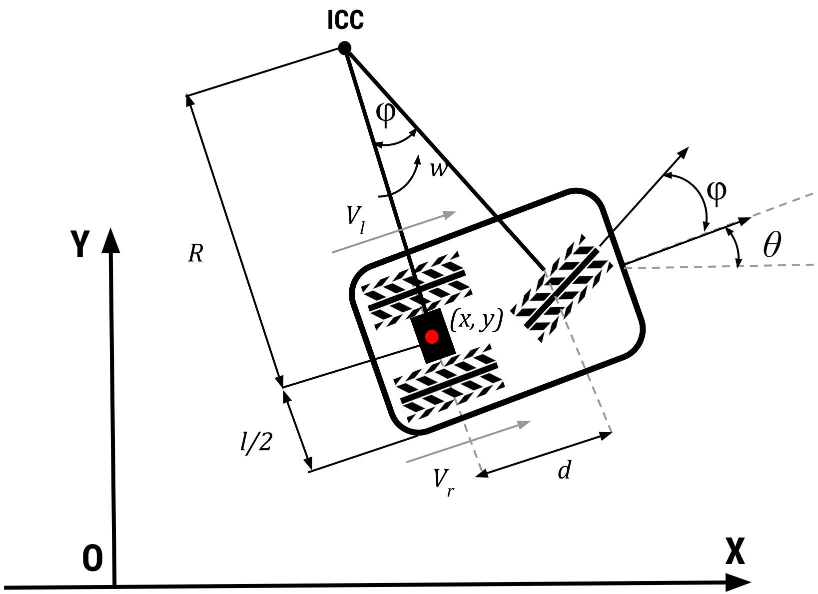


$$x(t) = \frac{1}{2}\int_0^t [v_r(t) + v_l(t)]cos[\theta(t)]dt$$
$$y(t) = \frac{1}{2}\int_0^t [v_r(t) + v_l(t)]sin[\theta(t)]dt$$
$$\Theta(t) = \frac{1}{l}\int_0^t [v_r(t) - v_l(t)])dt$$

TRICYCLE

$$ICC = [x - R\sin(\theta), y + R\cos(\theta)]$$

$$R = \frac{d}{\tan \ \varphi}$$



At the time moment **t+δt** the robot pose is defined as:

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

TRICYCLE

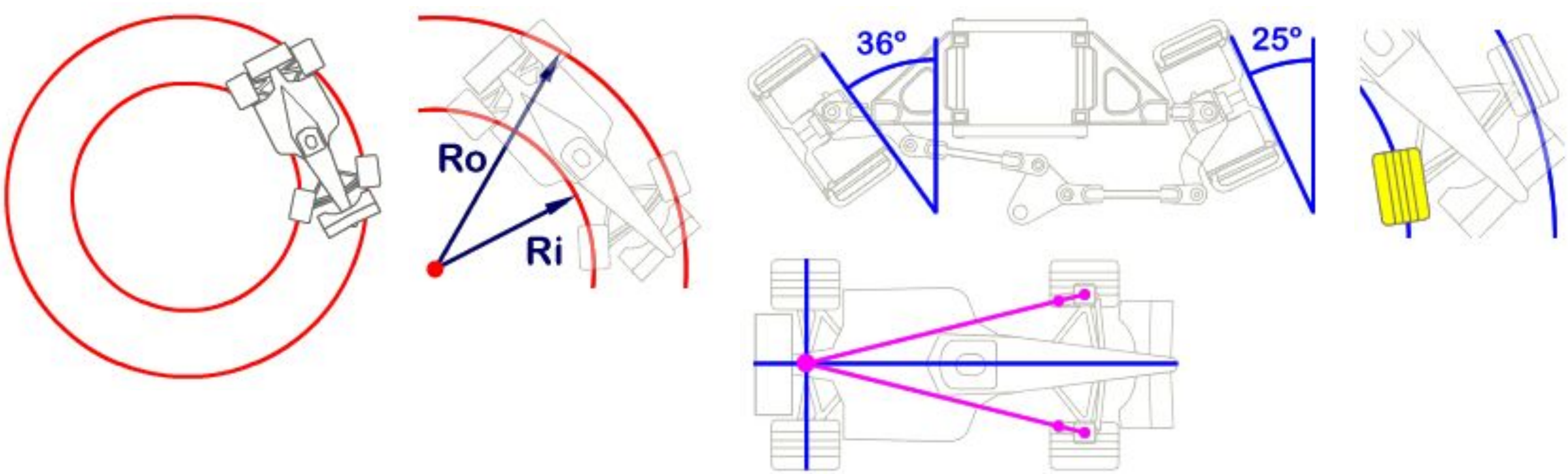
Features:

- Can not rotate in place
- When using 4 wheels, a differential for the rear wheels and an Ackermann steering geometry for the steering wheels is required

girafe ai

ACKERMANN STEERING PRINCIPLE

Steering geometry principle designed to allow steering wheels to go around circles of different radii and to avoid wheel slip.



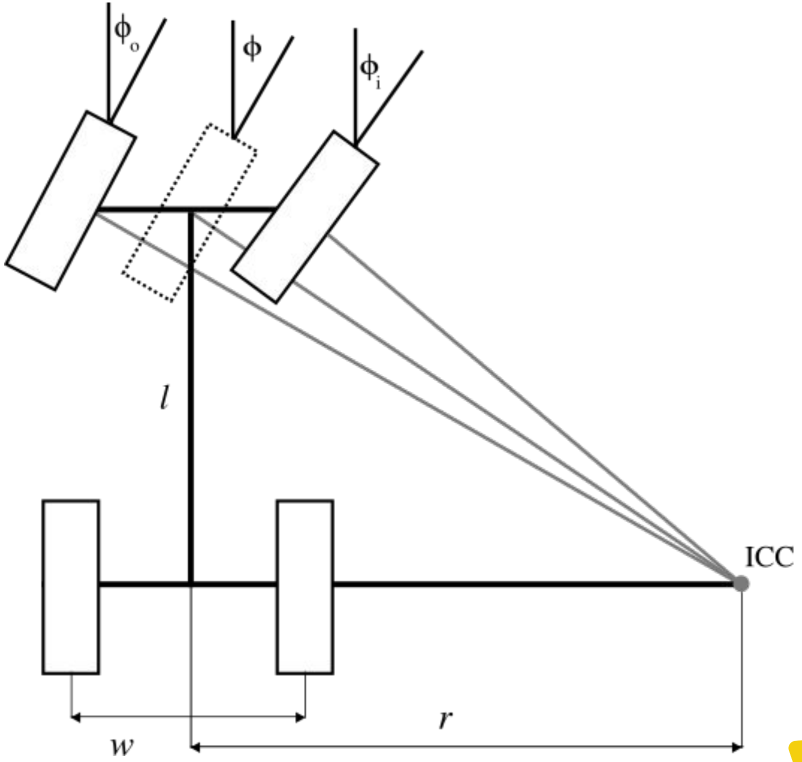
Source: http://www.rc-auto.ru/articles_tuning/id/445/

ACKERMANN STEERING PRINCIPLE

$$\tan(\phi) = \frac{l}{r}$$

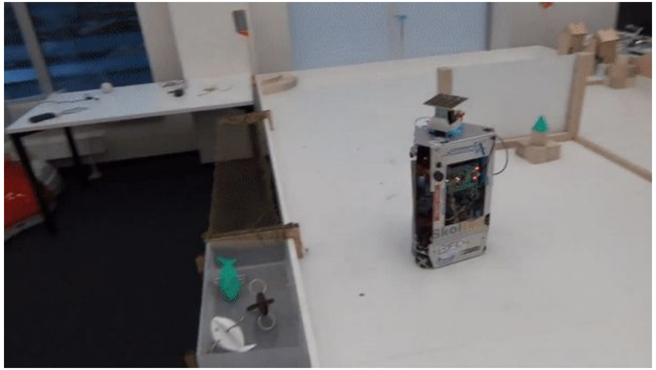
$$\tan(\phi_i) = \frac{l}{r - \frac{w}{2}}$$

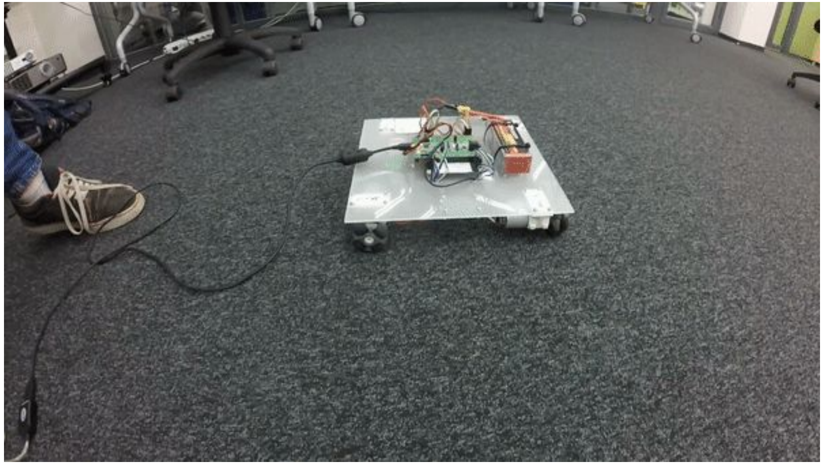
$$\tan(\phi_o) = \frac{l}{r + \frac{w}{2}}$$



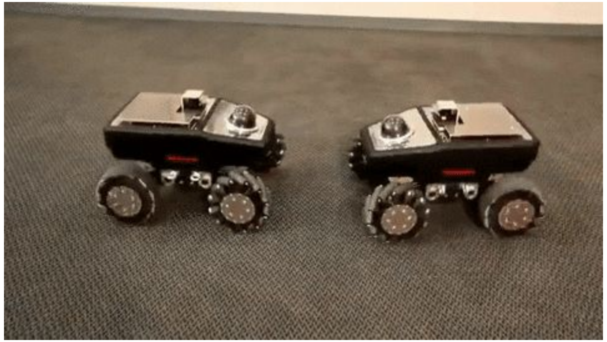
Source: http://www.rc-auto.ru/articles_tuning/id/445/

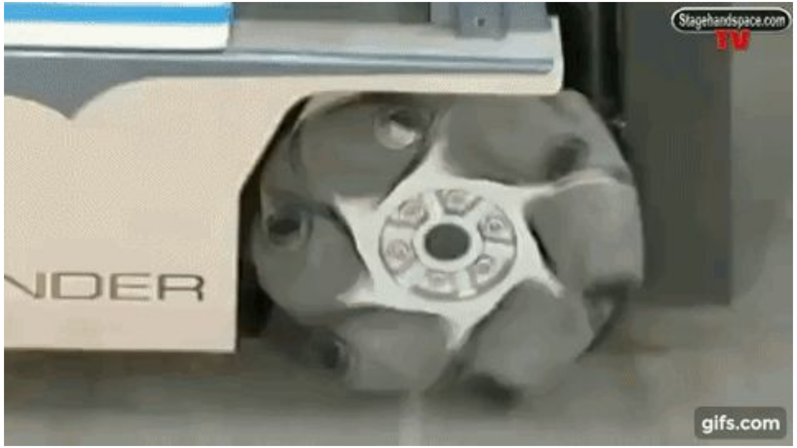
OMNIDIRECTIONAL WHEELS





MECANUM WHEELS (ILON WHEEL, SWEDISH WHEEL)

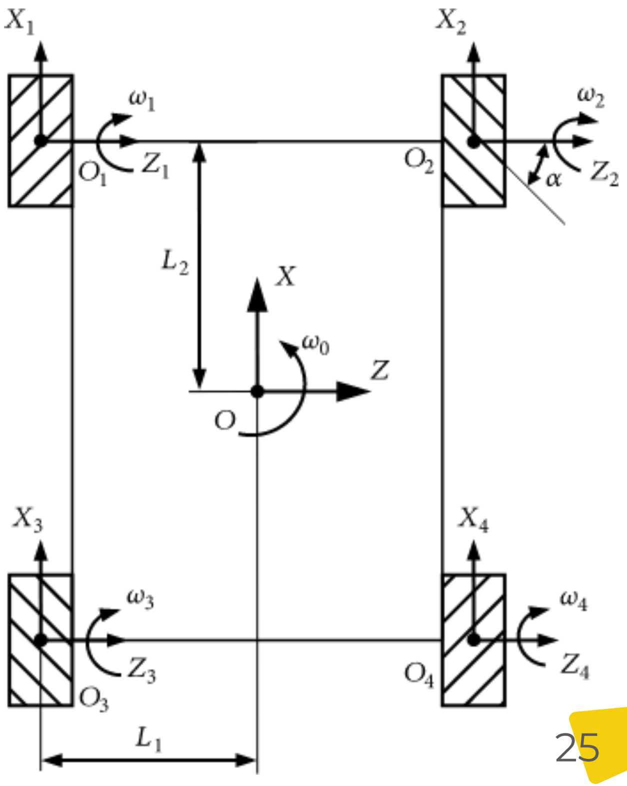




MECANUM WHEELS (ILON WHEEL, SWEDISH WHEEL)

$$\begin{bmatrix} v_x \\ v_z \\ \omega_0 \end{bmatrix} = \frac{r}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & -1 & 1 \\ -\frac{1}{L_1 + L_2} & \frac{1}{L_1 + L_2} & -\frac{1}{L_1 + L_2} & \frac{1}{L_1 + L_2} \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix}$$

Motion type	ω_1	ω_2	ω_3	ω_4
Straight	ω	ω	ω	ω
Perpendicular	ω	$-\omega$	$-\omega$	ω
45º motion	0	ω	ω	0
In place rotation	ω	$-\omega$	ω	$-\omega$



Li, Yunwang, et al. "Modeling and kinematics simulation of a Mecanum wheel platform in RecurDyn." *Journal of Robotics* 2018 (2018).

Probabilistic motion models

girafe
ai

02

RECURSIVE BAYESIAN POSE ESTIMATION

$$p(\mathbf{x_t}|map, \mathbf{z_t}, \mathbf{u_t}) = C \cdot p(\mathbf{z_t}|\mathbf{x_t}, map) \int_S p(\mathbf{x_t}|\mathbf{u_t}, \mathbf{x_{t-1}})p(\mathbf{x_{t-1}}|map, \mathbf{z_{t-1}}, \mathbf{u_{t-1}})d\mathbf{x_{t-1}}$$

C — normalization coefficient

S — the probabilistic space of robot poses

$p(\mathbf{z_t}|\mathbf{x_t}, map)$ — observation (measurement) model

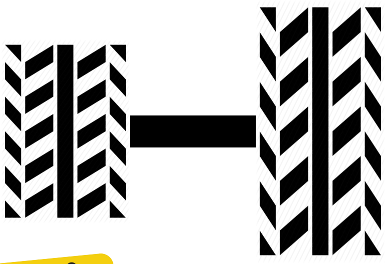
$p(\mathbf{x_t}|\mathbf{u_t}, \mathbf{x_{t-1}})$ — motion model

$p(\mathbf{x_{t-1}}|map, \mathbf{z_{t-1}}, \mathbf{u_{t-1}})$ — previous system state (robot pose)

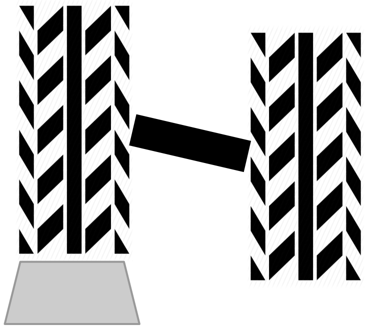
WHY DO WE NEED PROBABILISTIC MOTION MODELS?

- Actuators, like sensors, are not absolutely accurate.
- External factors also affect the precision of motion.

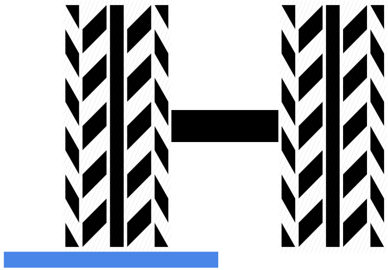
Difference in wheel diameters



Obstacles



Slippage on various surfaces



PROBABILISTIC MOTION MODELS

In practice, there are 2 types of motion models:

- ❑ **Odometry**-based
- ❑ **Speed** control based (dead reckoning)

Historically was used in ships navigation

- ❑ Odometry-based models are used when the robot is equipped with wheels encoders.
- ❑ Speed-based models are used when there are no encoders. They are based on calculating the traveled distance given the speed and travel time.

girafe ai

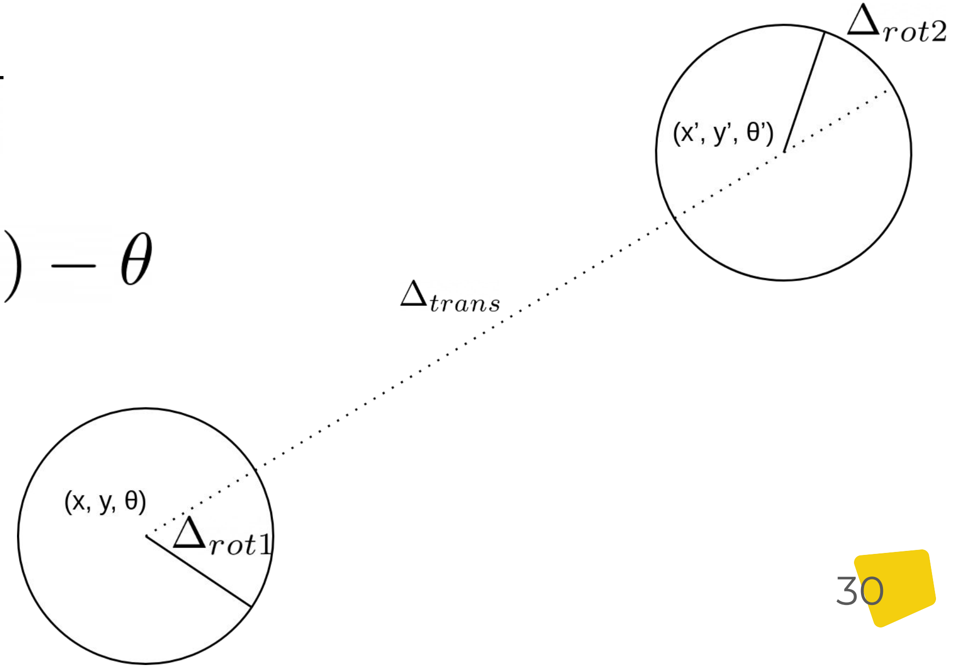
ODOMETRY-BASED MODEL

- The robot is moving from (x, y, θ) to (x', y', θ')
- Encoders provide the following information: $u_t = (\Delta_{trans}, \Delta_{rot1}, \Delta_{rot2})$

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

$$\Delta_{rot1} = atan2(y' - y, x' - x) - \theta$$

$$\Delta_{rot2} = \theta' - \theta - \Delta_{rot1}$$



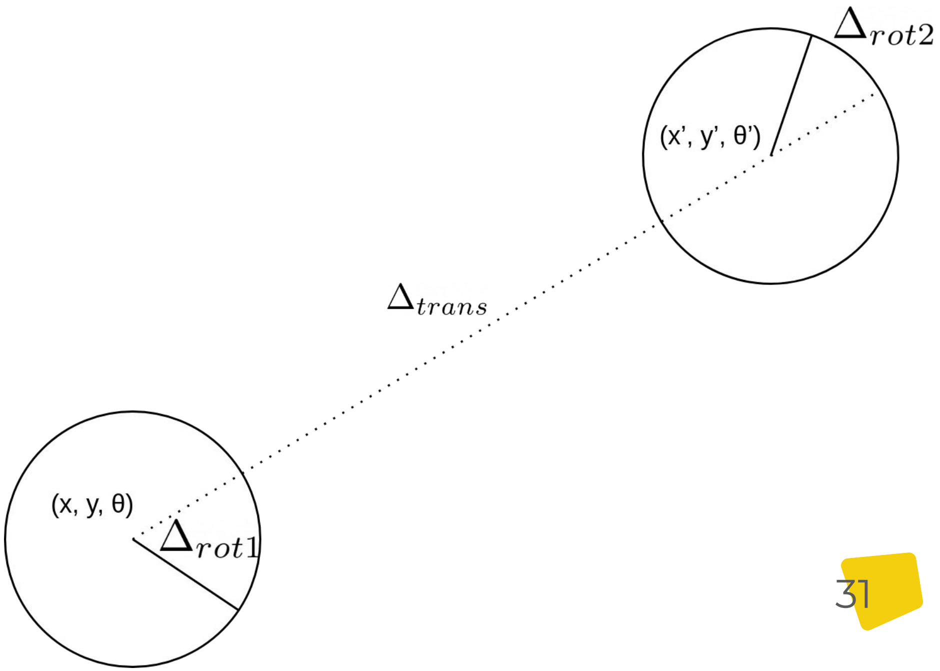
NOISE MODEL

Real motion is prone to error (noise):

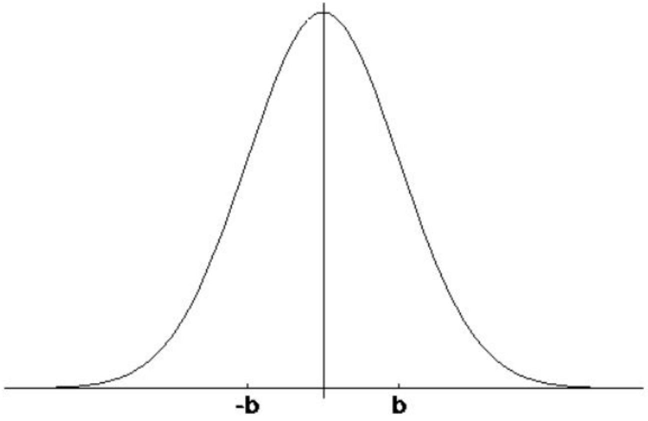
$$\hat{\Delta}_{trans} = \Delta_{trans} + \eta_1$$

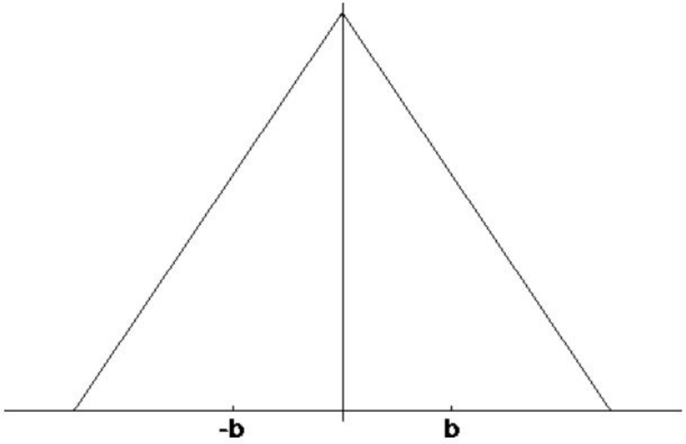
$$\hat{\Delta}_{rot1} = \Delta_{rot1} + \eta_2$$

$$\hat{\Delta}_{rot2} = \Delta_{rot2} + \eta_3$$



NOISE MODEL





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

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

girafe ai

NOISE MODELING

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

2. return $\frac{1}{\sqrt{2\pi\, b^2}} \exp\left\{-\frac{1}{2}\frac{a^2}{b^2}\right\}$

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

2. return $\max\left\{0, \frac{1}{\sqrt{6}\, b} - \frac{|a|}{6\, b^2}\right\}$

girafe ai

SAMPLING FROM NOISE MODEL

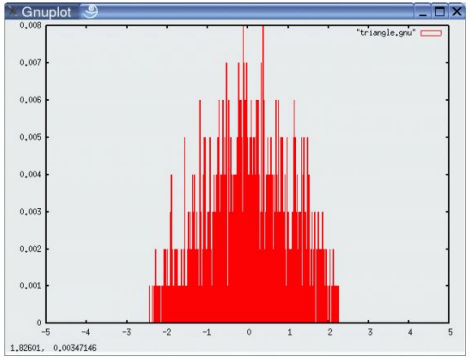
1. Algorithm **sample_normal_distribution**(b):

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

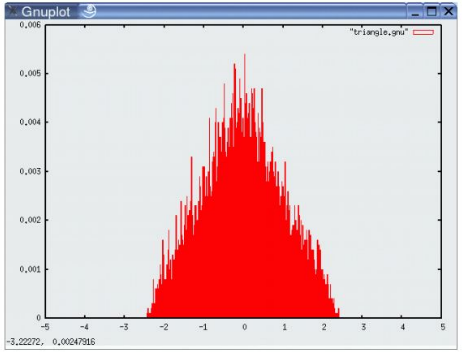
1. Algorithm **sample_triangular_distribution**(b):

2. return $\frac{\sqrt{6}}{2}\,[\mathrm{rand}(-b, b) + \mathrm{rand}(-b, b)]$

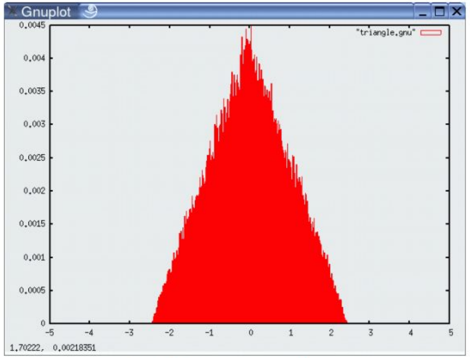
SAMPLING FROM NOISE MODEL



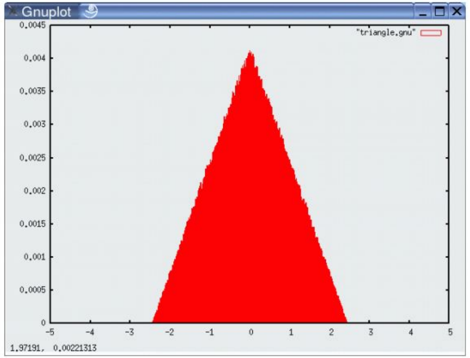
10^3 samples



10^4 samples



10^5 samples



10^6 samples

POSE POSTERIOR DISTRIBUTION ESTIMATION

hypotheses odometry

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

odometry params (**u**)

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

girafe ai

SAMPLING FROM MOTION MODEL

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

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

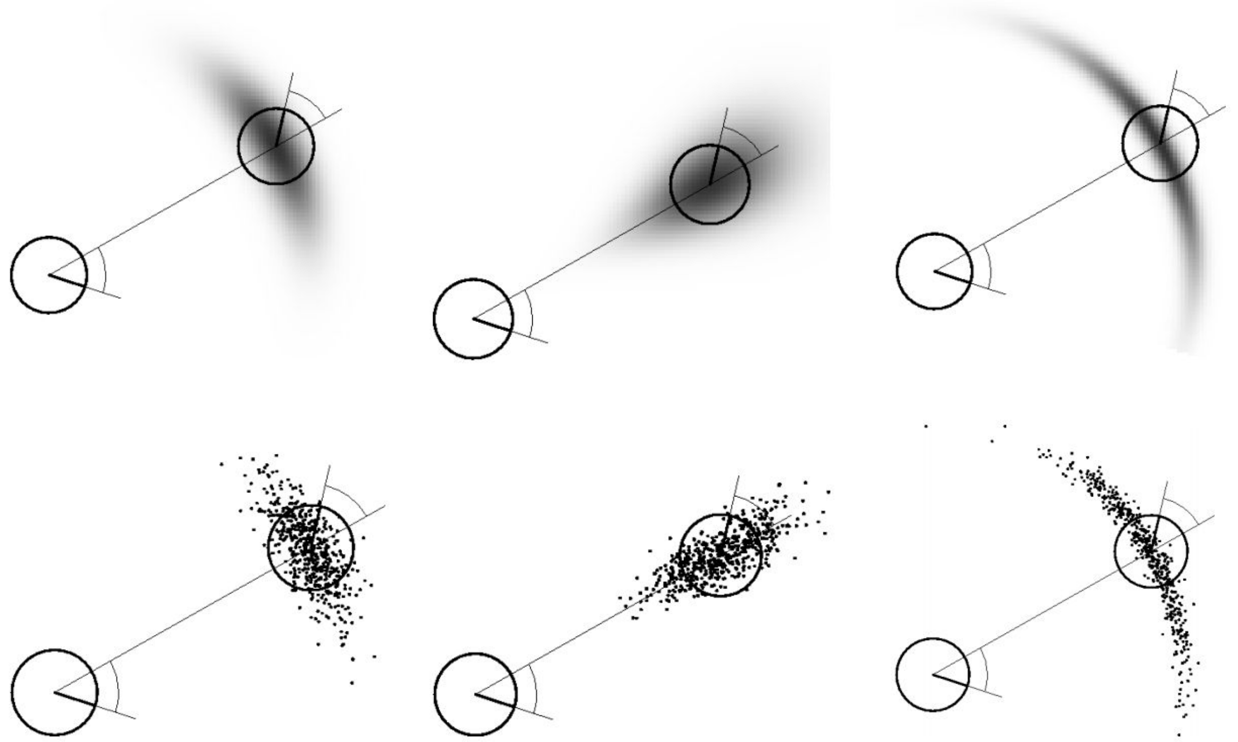
1. $\hat{\delta}_{rot1} = \delta_{rot1} + \text{sample}(\alpha_1 |\delta_{rot1}| + \alpha_2\, \delta_{trans})$
2. $\hat{\delta}_{trans} = \delta_{trans} + \text{sample}(\alpha_3\, \delta_{trans} + \alpha_4\, (|\delta_{rot1}| + |\delta_{rot2}|))$
3. $\hat{\delta}_{rot2} = \delta_{rot2} + \text{sample}(\alpha_1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

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

EXAMPLE OF ODOMETRY-BASED MOTION MODEL

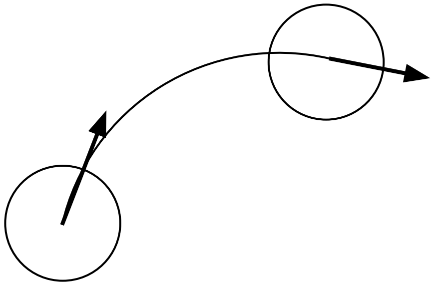


SPEED-BASED MOTION MODEL

- Such a model assumes that we control the parameters of the robot's motion — linear and angular velocity
- The robot moves along a circular arc
- Control signals (speeds) are subject to noise

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

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



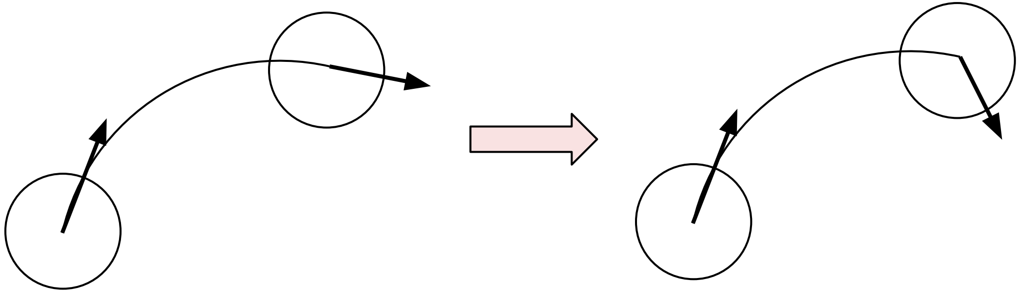
SPEED-BASED MOTION MODEL

- To allow the final turn, a third motion parameter is introduced

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

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

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



SPEED-BASED MOTION MODEL

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

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

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

SAMPLING FROM SPEED-BASED MOTION MODEL

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

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

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

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

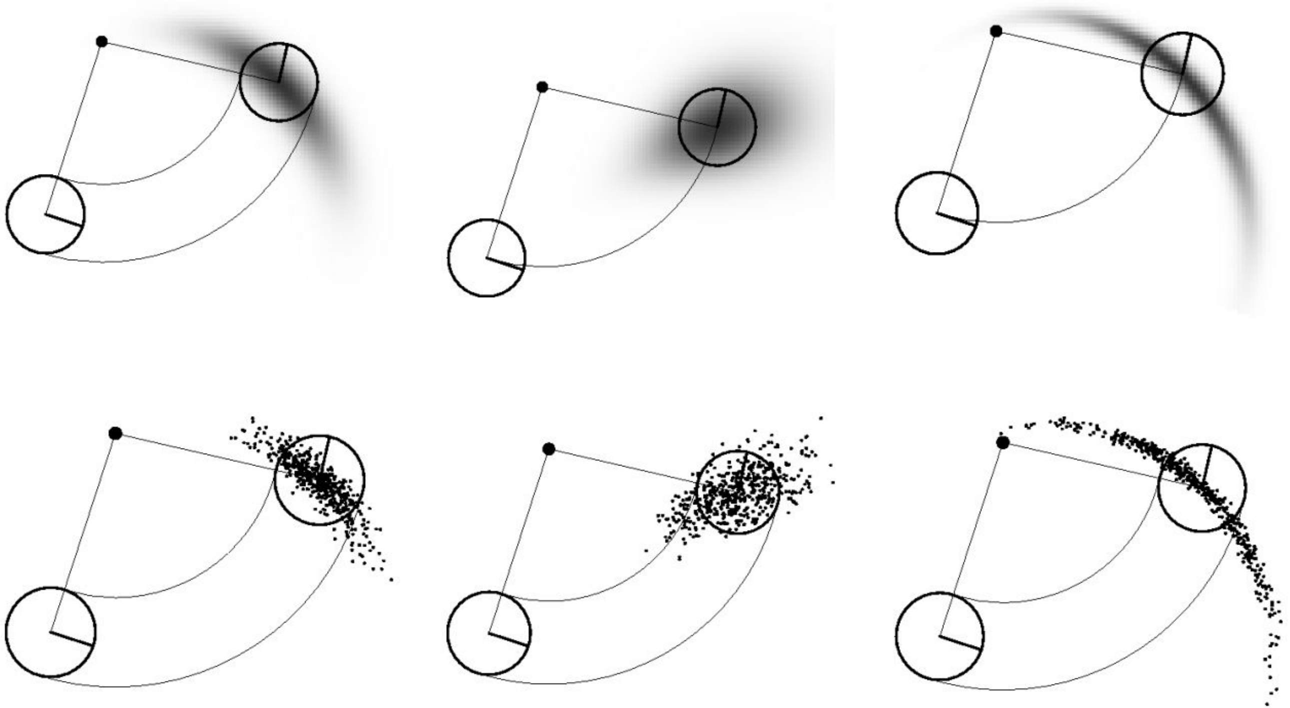
$x' = x - \frac{\hat{v}}{\hat{\omega}} \sin\theta + \frac{\hat{v}}{\hat{\omega}}\ sin(\theta + \hat{\omega}\Delta t)$

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

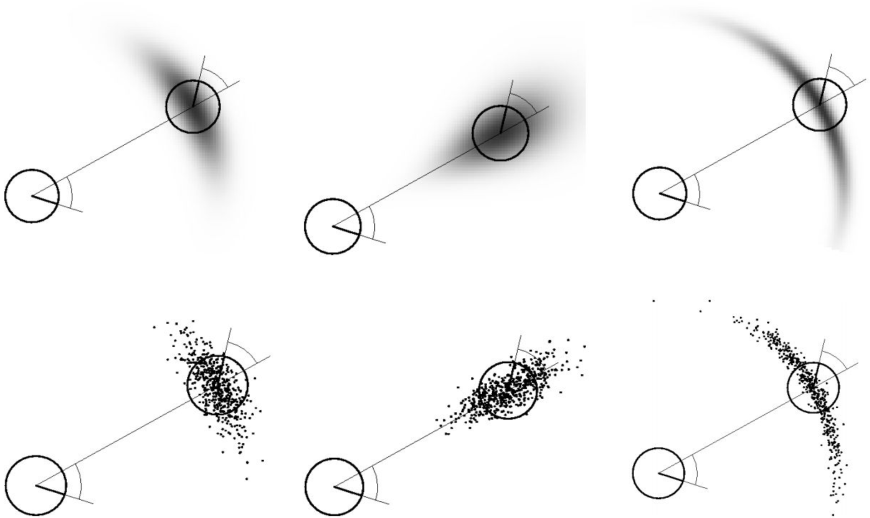
$\theta' = \theta + \hat{\omega}\Delta t + \hat{\gamma}\Delta t$

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

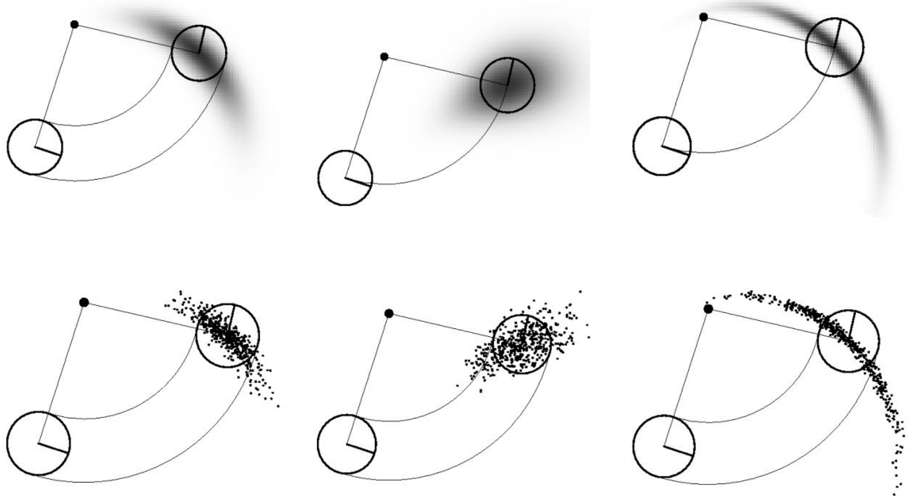
EXAMPLE OF SPEED-BASED MOTION MODEL



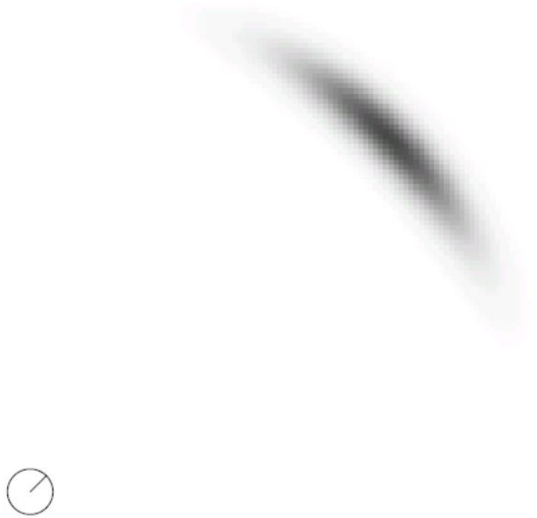
ODOMETRY-BASED MODEL

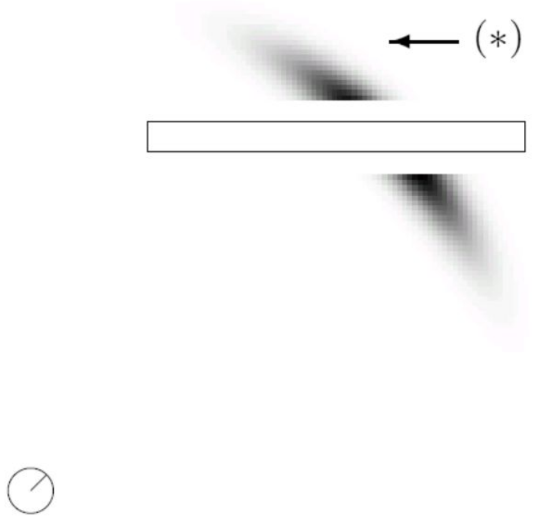


SPEED-BASED MODEL



MOTION MODELS ACCOUNTING FOR ENVIRONMENT





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

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

ADDITIONAL RESOURCES

1. Differential Drive Kinematics
2. Probabilistic Robotics (in Notion). Chapter 5.
3. Mobility: wheels and whegs

Thanks for attention!

Questions? Additions? Welcome!