5 – Kinematics & Odometry

Robotics and Computer Vision (BPC-PRP)

Course supervisor: Ing. Adam Ligocki, Ph.D.

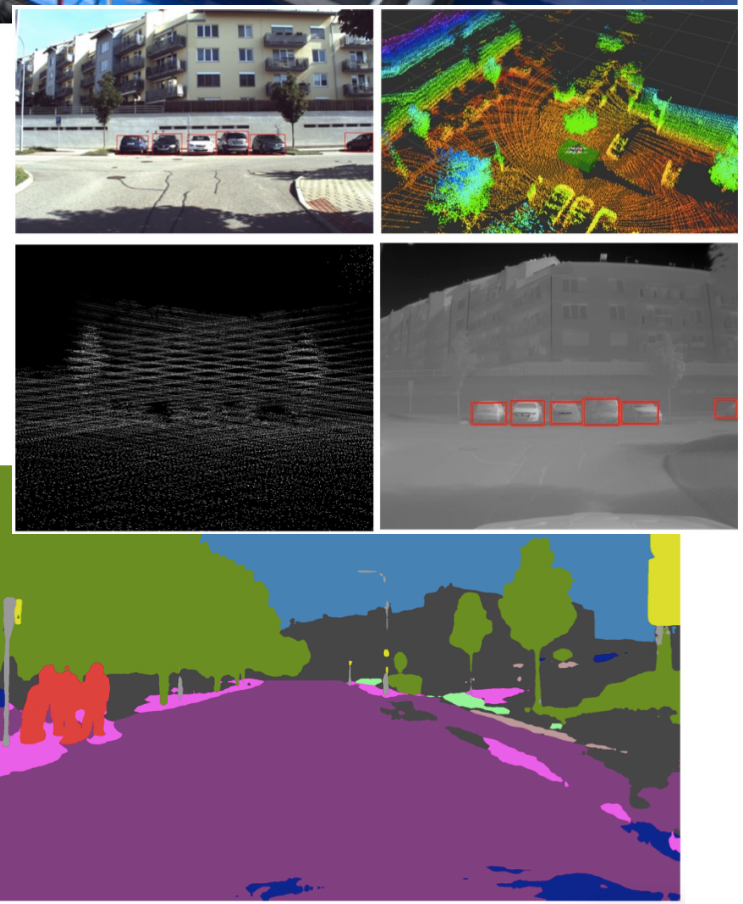
Ing. Adam Ligocki, Ph.D.

Brno University of Technology
2025

Robotics and AI

Profile

Robotics and AI



Ing. Adam Ligocki, Ph.D.

Position: Assistant Professor

Research: Data Fusion

Room: SE1.102

Web: https://www.vut.cz/lide/adam-ligocki-154791

Background:

- Artificial Intelligence
- Neural Networks
- Software Development

Kinematics and Odometry

Kinematics is the study of how a robot's parts move relative to each other and to world. It focuses on describing the positions, velocities, and accelerations of these parts without considering the forces that cause them. In simpler terms, it's about understanding how a robot moves so we can control the robot's overall motion.

Forward kinematics involves figuring out a robot's final position based on individual part movements.

Inverse kinematics works in reverse by calculating the movements of each part to reach a desired position or orientation.

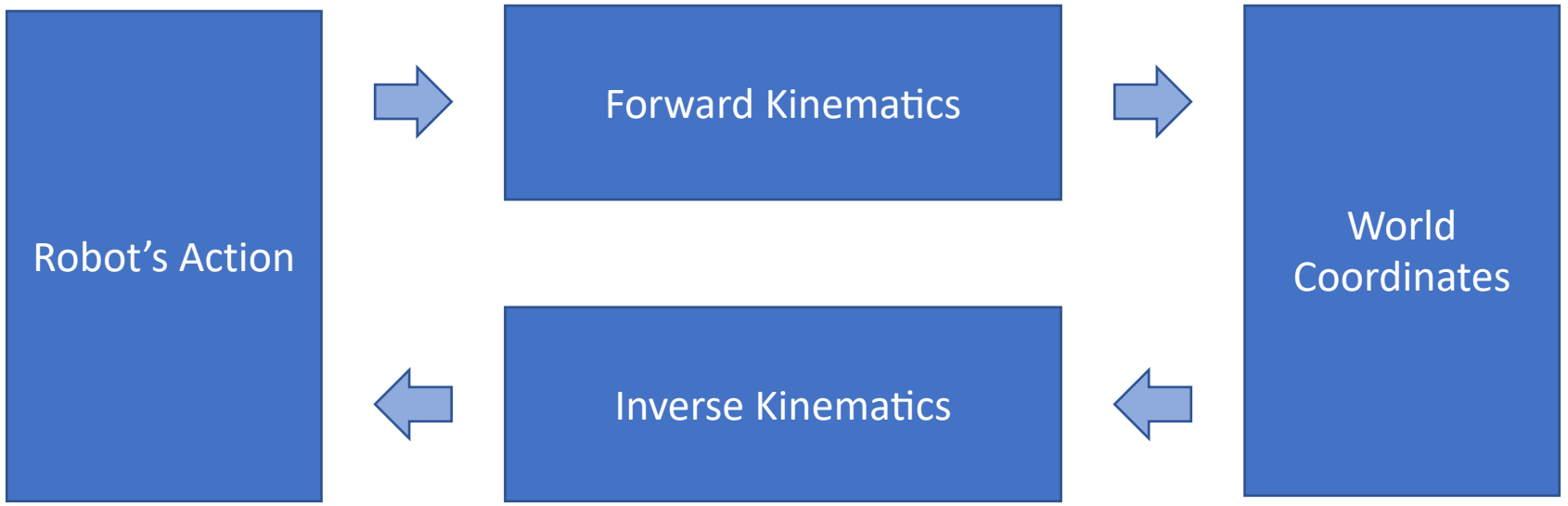
Odometry is a method to estimate a robot's position and orientation by tracking how much its wheels have turned. It commonly relies on wheel encoders that measure rotations, translating those rotations into distance traveled. Using these measurements over time, odometry helps a robot keep track of its location without relying on external systems.

Kinematics

Kinematics

Forward kinematics: how the robot's joints (wheels) movement effects it's world coordinates.

Inverse kinematics: how to move robot's joints (wheels) to reach required world's coordinates.



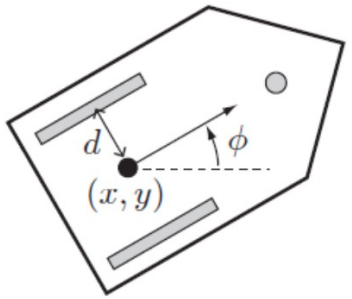
Type of Drives (Chassis)

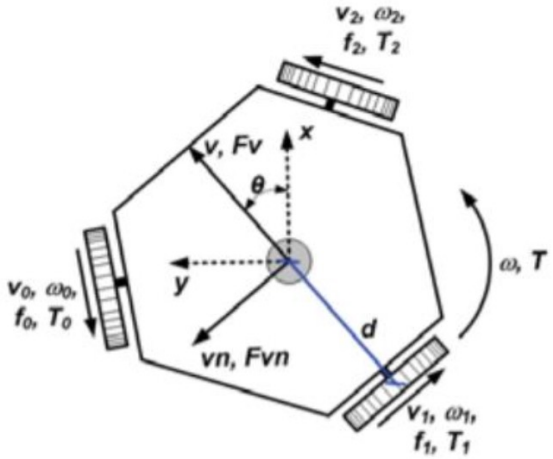
Differential

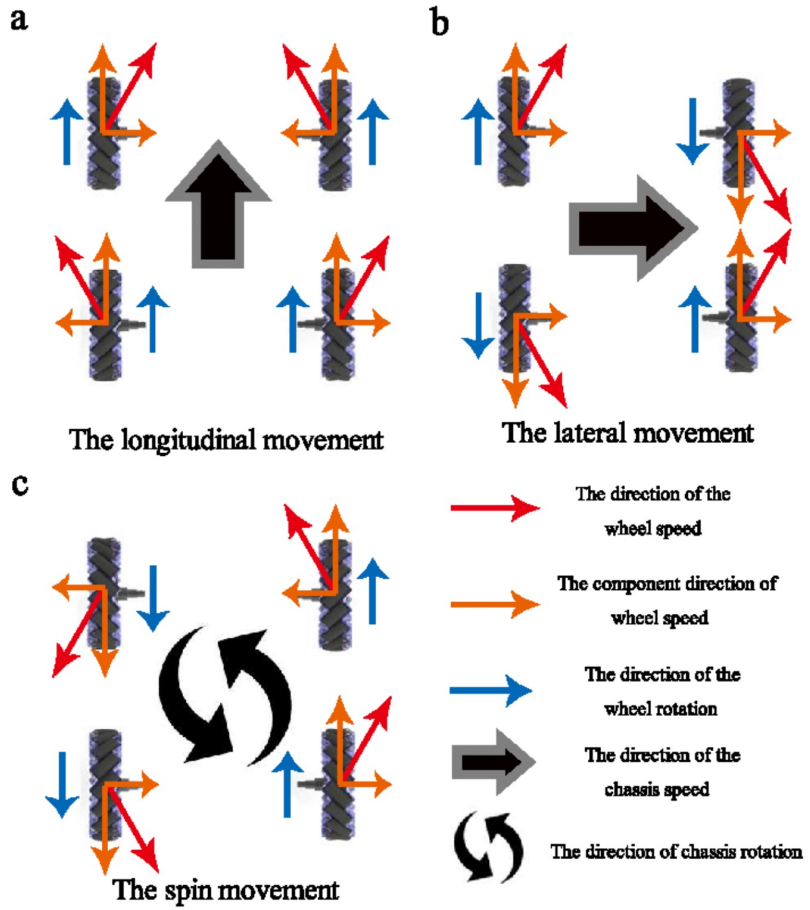
Omnidirectional

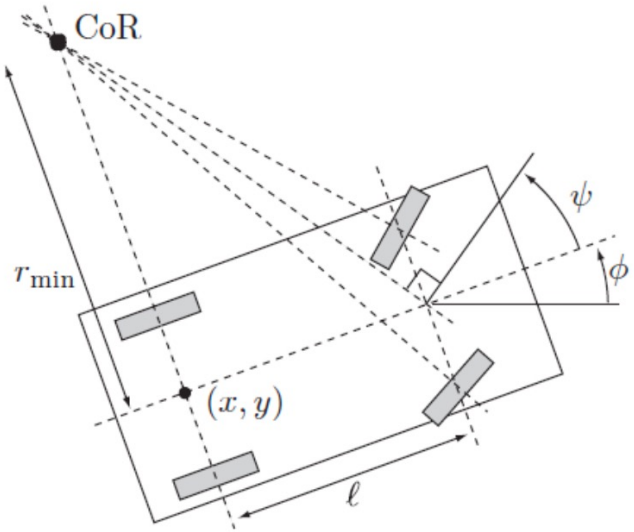
Ackerman

…



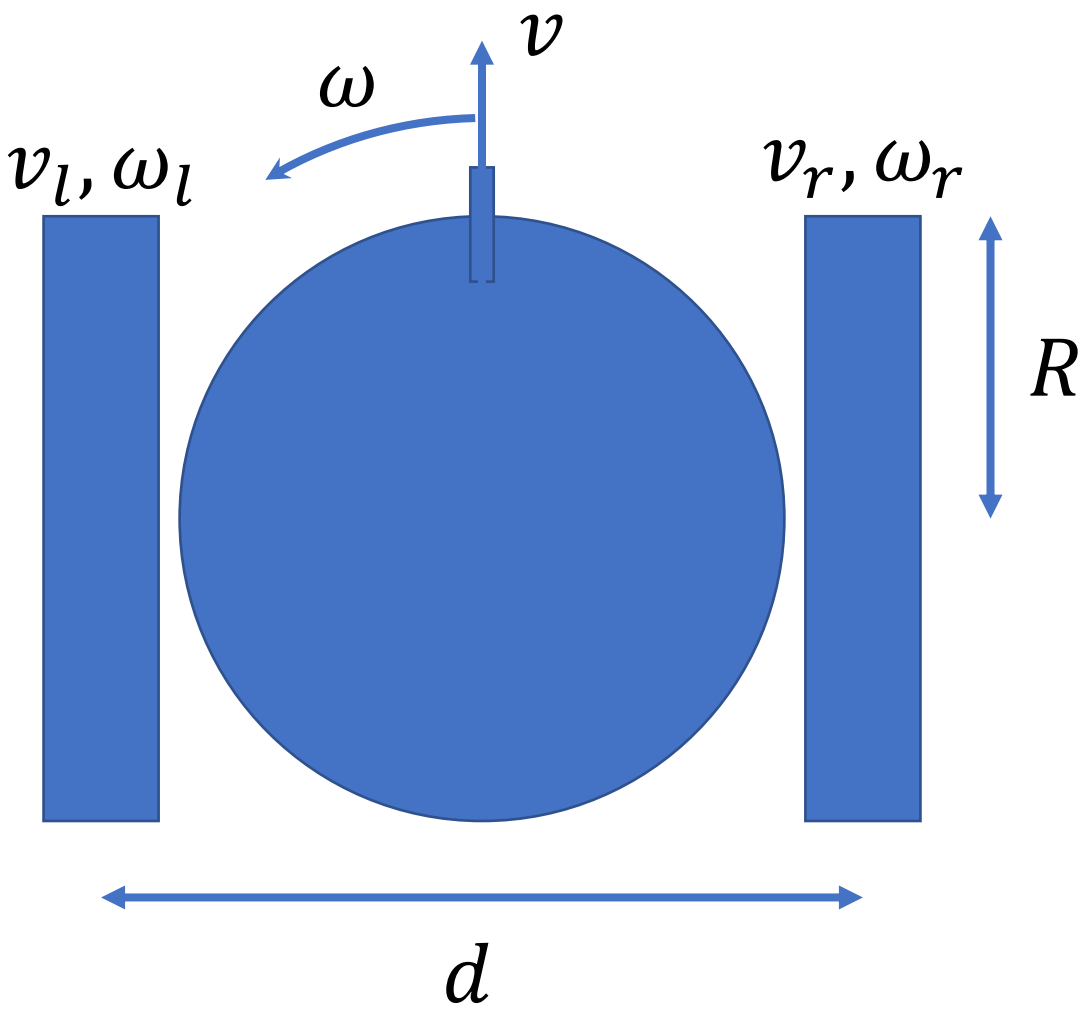






Electronics and Robotics Club. The ERC Handbook: Drive Mechanism [online]. BITS Goa, 2020 [cit. 2022-02-14]. Available at: https://erc-bpgc.github.io/handbook/mechanical/drive_mechanism/
https://en.wikipedia.org/wiki/Omni_wheel
https://www.researchgate.net/figure/The-principle-of-omnidirectional-chassis-movement_fig2_328995037

Differential Chassis

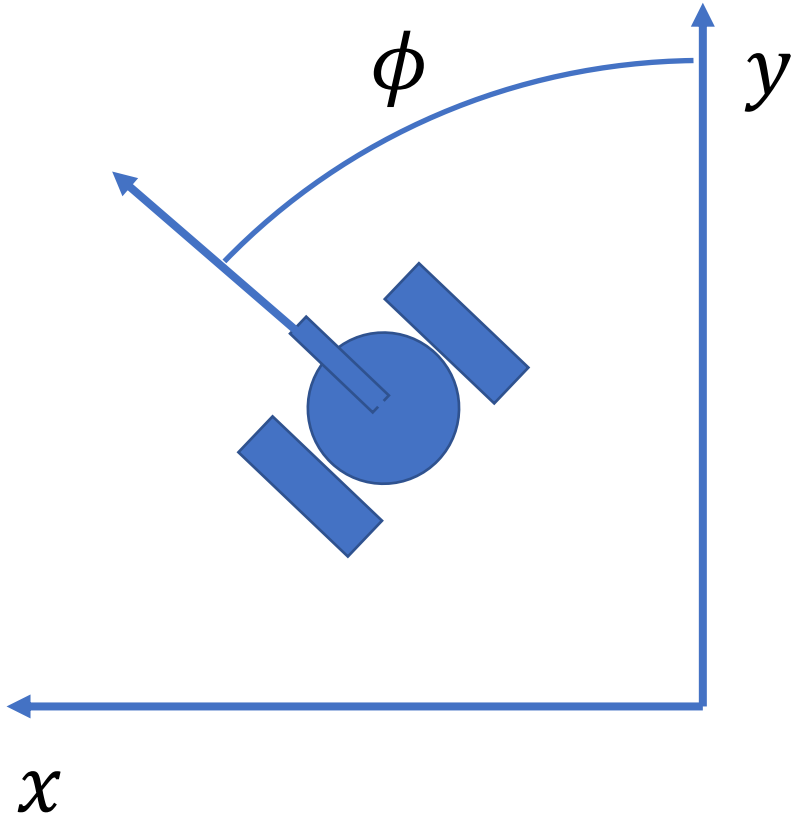


$$\omega_x * \mathrm{R} = v_x$$

World‘s Coordinates

World position is described by 3 degrees of freedom (DoF).

Expressed by the 3x1 vector $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\phi})$.



Differential Chassis

Holonomic vs Non-Holonomic

Degrees of Freedom (DoF)

refers to the number of independent parameters or coordinates needed to uniquely describe a system's position or configuration.

Degrees of Motion (DoM)

describes the separate directions or axes along which a system or object can move (rotations and translations).

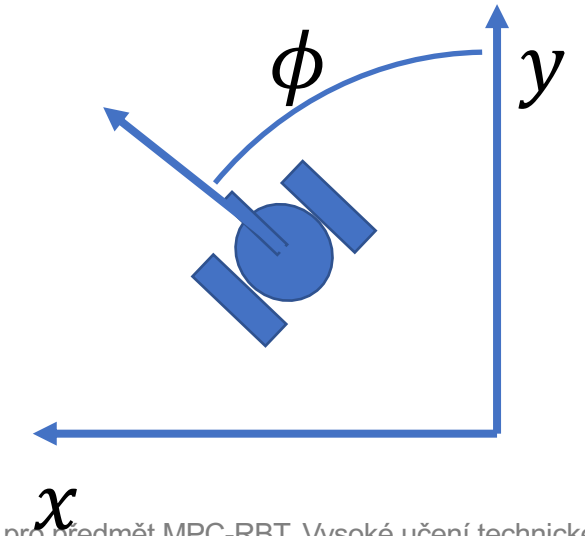
Also called Differentiabe Degrees of Freedom (DDoF)

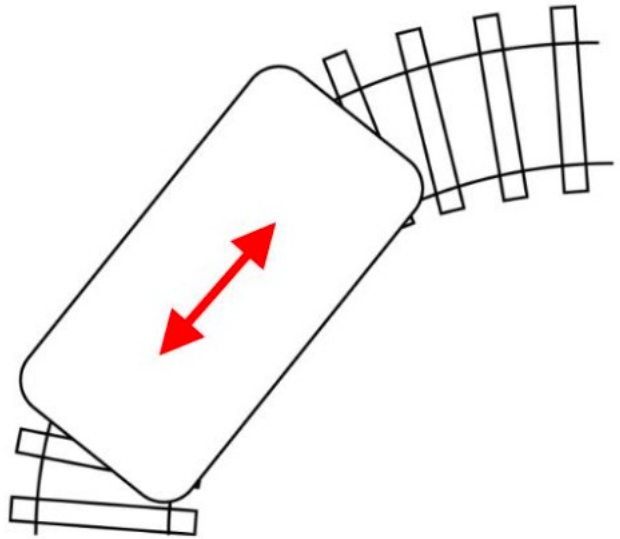
Holonomic Robot

DoF = DoM (DDoF)

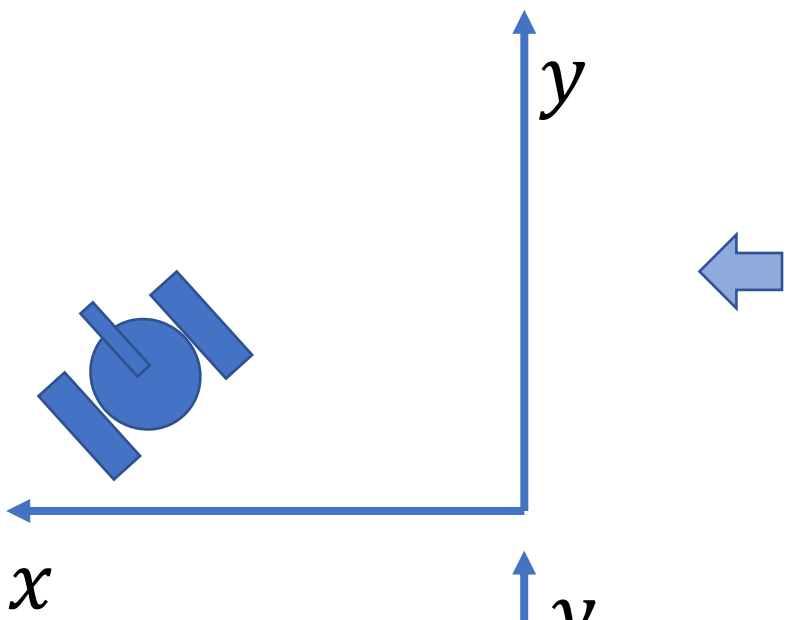
Non-Holonomic Robot

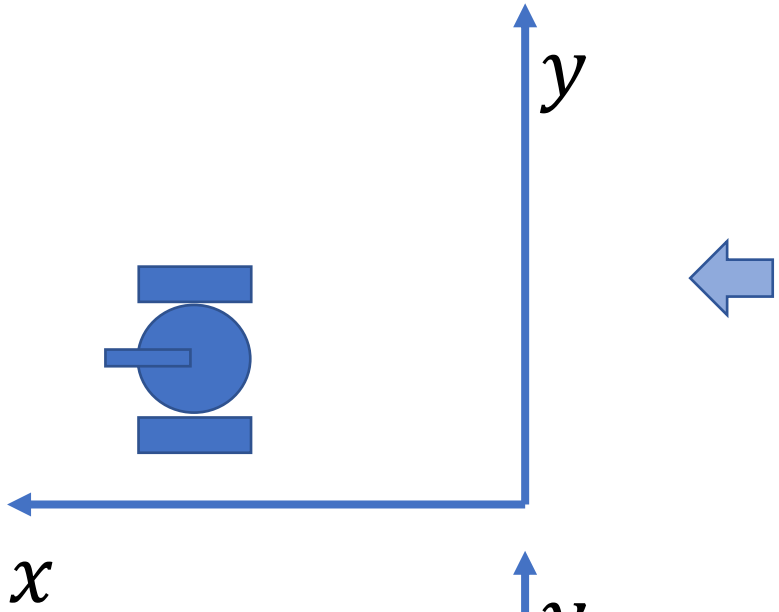
DoF > DoM

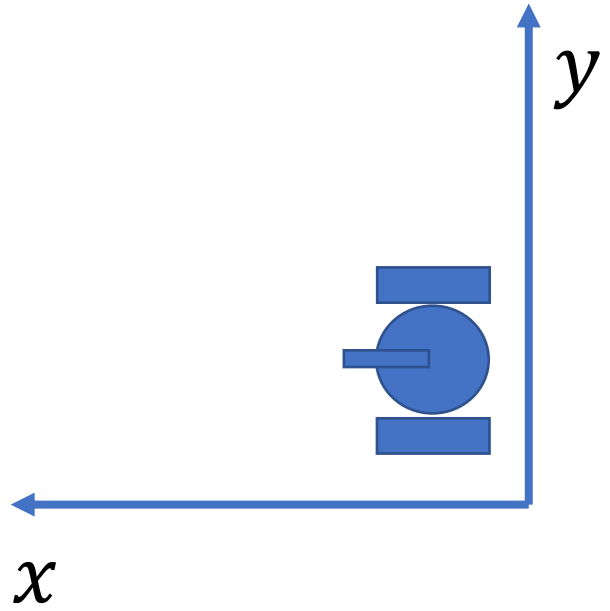


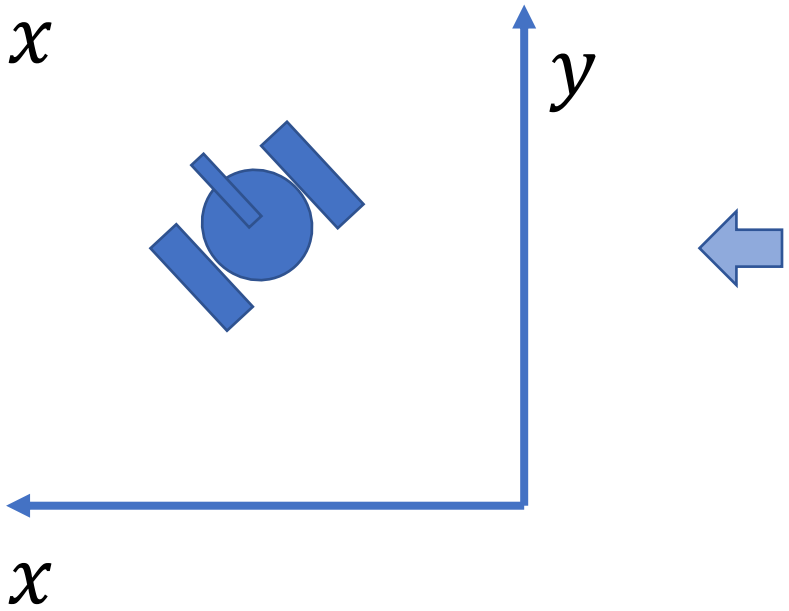


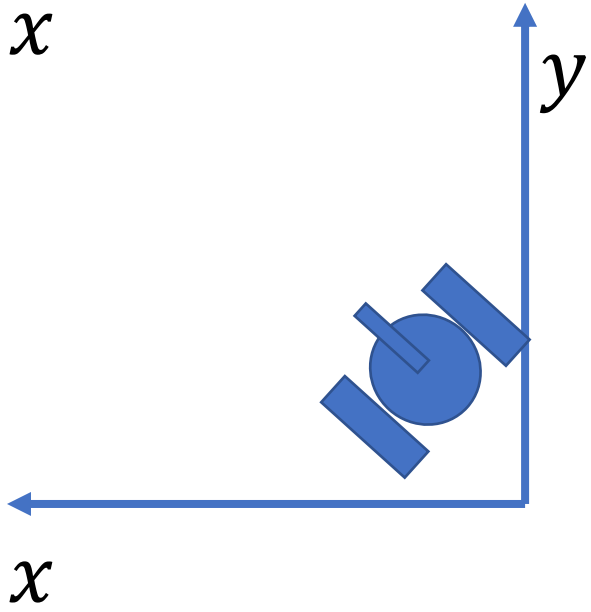
Holonomic vs Non-Holonomic











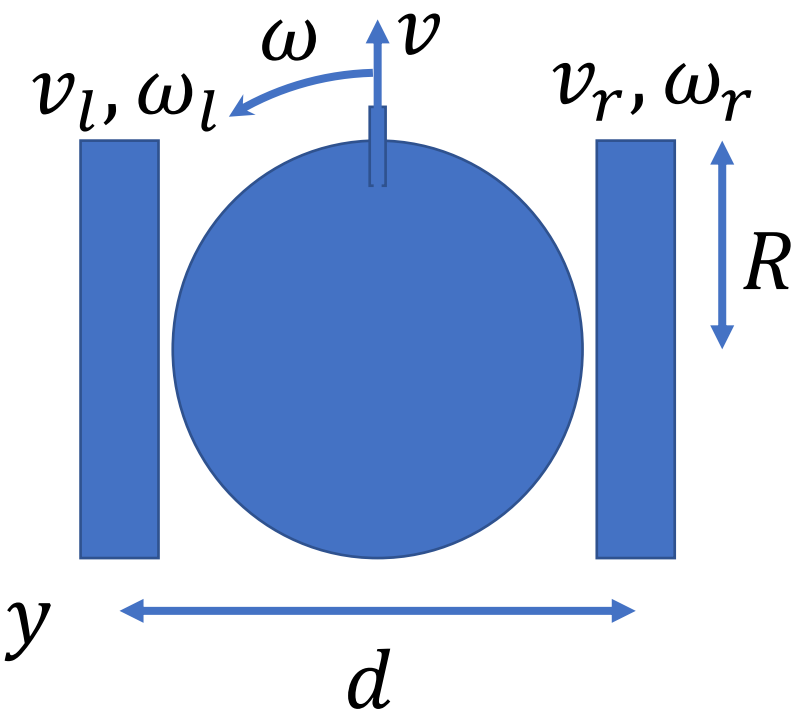
Depends on the order

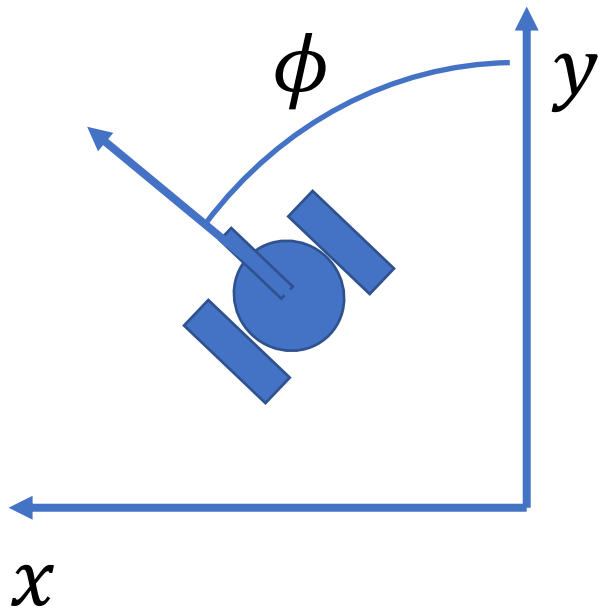
HRABEC, Jakub. Kinematika mobilních robotů: Stručný úvod do problematiky. Prezentace pro předmět MPC-RBT. Vysoké učení technické v Brně, 2021.

Forward Kinematics

$$\dot{x} = v * \cos\phi$$
$$\dot{y} = v * \sin\phi$$
$$\dot{\phi} = \omega$$

$$\dot{x} = \frac{R}{2}(\omega_r + \omega_l) * \cos\phi$$
$$\dot{y} = \frac{R}{2}(\omega_r + \omega_l) * \sin\phi$$
$$\dot{\phi} = \frac{R}{L}(\omega_r - \omega_l)$$





Inverse (Backward) Kinematics

$$\dot{x} = v * \cos\phi$$
$$\dot{y} = v * \sin\phi$$
$$\dot{\phi} = \omega$$

$$v = \frac{R}{2}(\omega_r + \omega_l) \rightarrow \frac{2v}{R} = \omega_r + \omega$$
$$\omega = \frac{R}{L}(\omega_r - \omega_l) \rightarrow \frac{\omega L}{R} = \omega_r - \omega_l$$

$$\dot{x} = \frac{R}{2}(\omega_r + \omega_l) * \cos\phi$$
$$\dot{y} = \frac{R}{2}(\omega_r + \omega_l) * \sin\phi$$
$$\dot{\phi} = \frac{R}{L}(\omega_r - \omega_l)$$

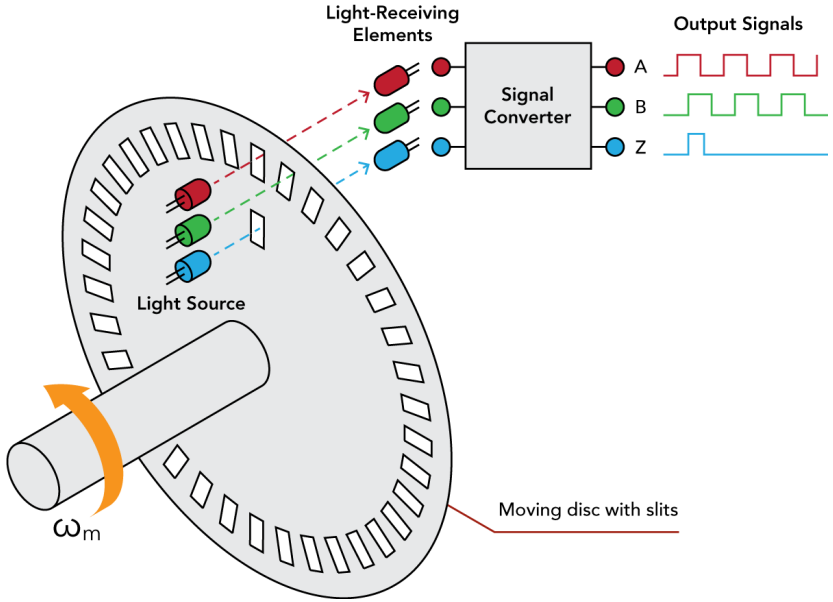
$$\boldsymbol{v_r = \frac{2v + \omega L}{2R}}$$
$$\boldsymbol{v_l = \frac{2v - \omega L}{2R}}$$

What is the dimension of the v_r?

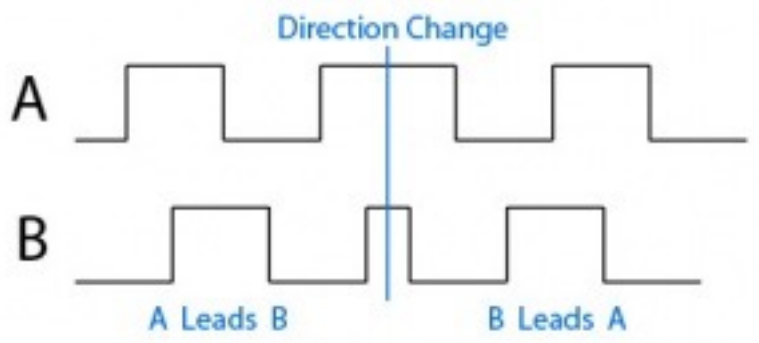
Odometry

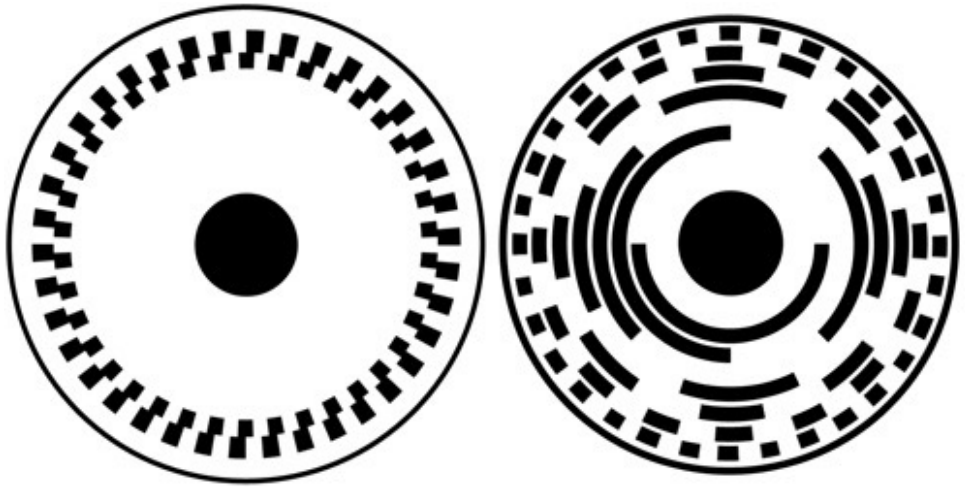
- Method to estimate the robot’s position & orientation over time.
- Relies on wheel encoder data measuring distance travelled.
- Useful for navigation, especially in indoor or GPS-denied environments.
- Wheel rotation -> distance traveled (using wheel radius & encoder counts).
- Small increments in distance combined to update overall robot pose.
- Constantly tracks (x, y, θ) in a chosen coordinate frame.

Encoders

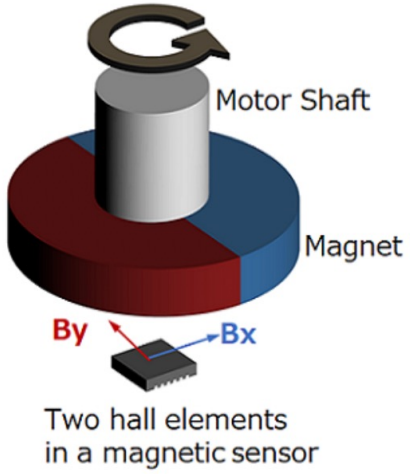


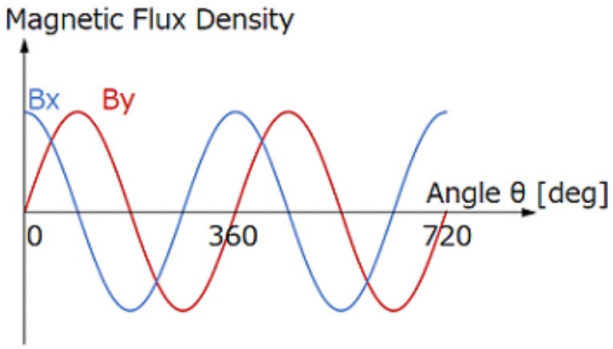
Quadrature Encoder





Relative vs Absolute

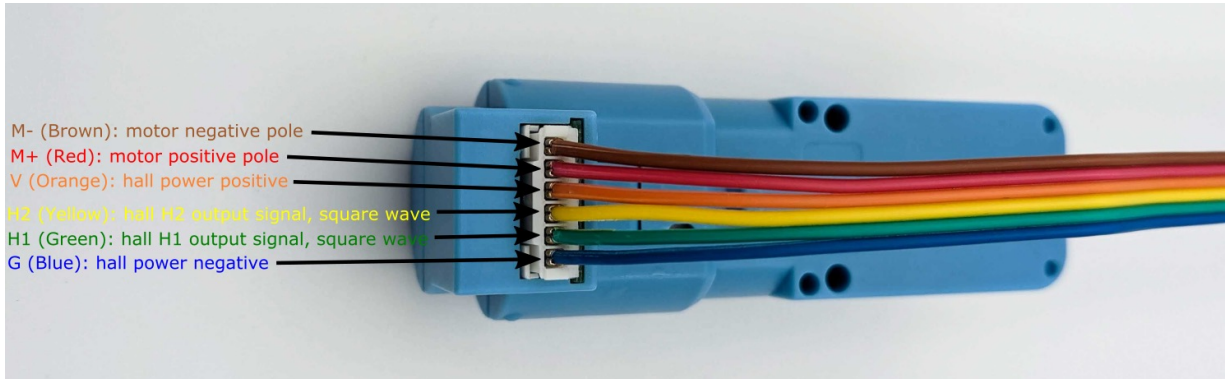


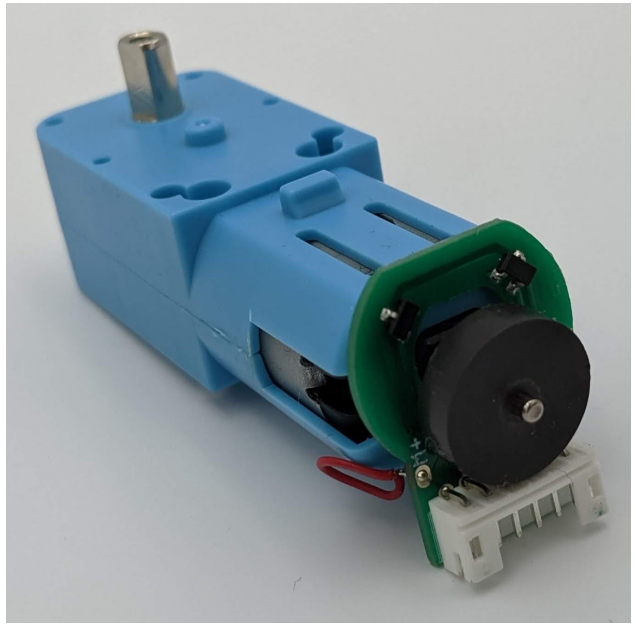


Magnetic Encoder

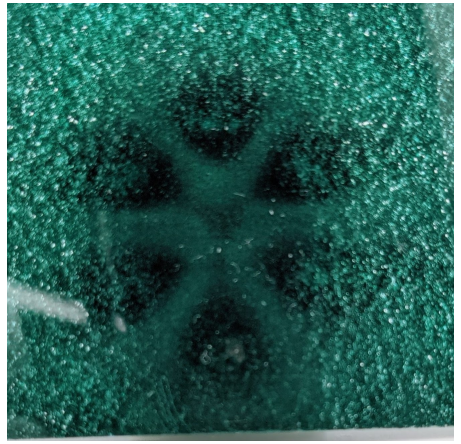
https://www.futek.com/Incremental-Encoder-Signal-Converter
https://www.phidgets.com/docs/Encoder_Guide
https://www.akm.com/global/en/products/rotation-angle-sensor/tutorial/magnetic-encoder/

DG01D-E Motor with Encoder





3 periods per motor rotation
1:48 gear ratio
2 channels
2 edges
3 * 48 * 2 * 2 = 576 pulses per single output axis rotation rotation



https://cz.mouser.com/ProductDetail/SparkFun/ROB-16413?qs=vmHwEFxEFR%2F20T1Ah9zr5w%3D%3D
https://ros-mobile-robots.com/DG01D-E-motor-with-encoder/

Odometry

Left and right wheel speed

$$\Delta d_l = R * \Delta\phi_l$$
$$\Delta d_r = R * \Delta\phi_r$$

$$\Delta d = \frac{\Delta d_l + \Delta d_r}{2}$$
$$\Delta\theta = \frac{\Delta d_r - \Delta d_l}{L}$$

$\Delta\phi$ is change in wheel's rotation
Related to encoder's ticks

$$\Delta d_l = R * \Delta\phi_l$$
$$\Delta d_r = R * \Delta\phi_r$$

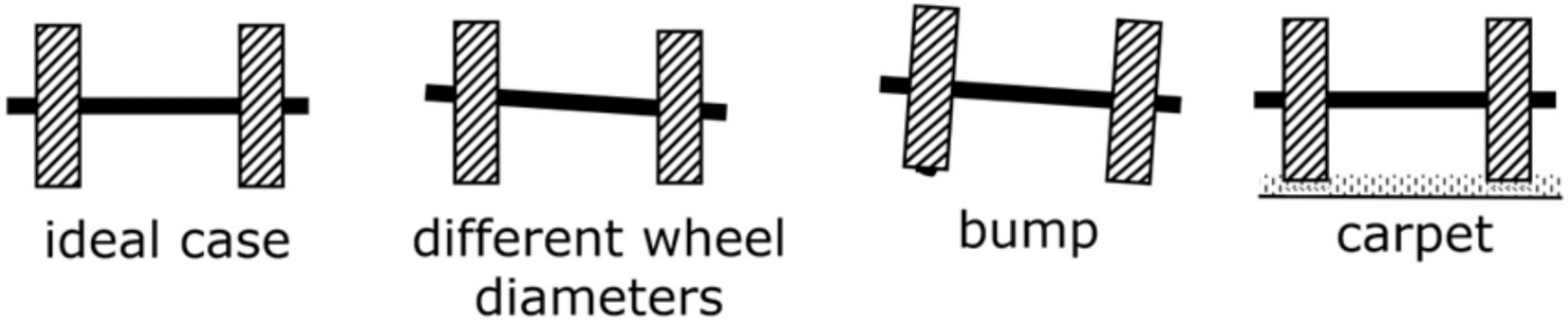
$$\boldsymbol{x_{k+1} = x_k + \Delta d * \cos(\theta_k + \frac{\Delta\theta}{2})}$$

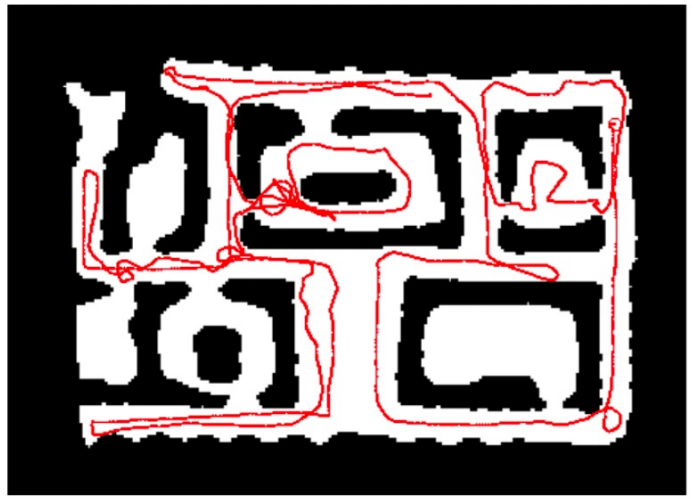
$$\boldsymbol{x_{k+1} = x_k + \Delta d * \sin(\theta_k + \frac{\Delta\theta}{2})}$$

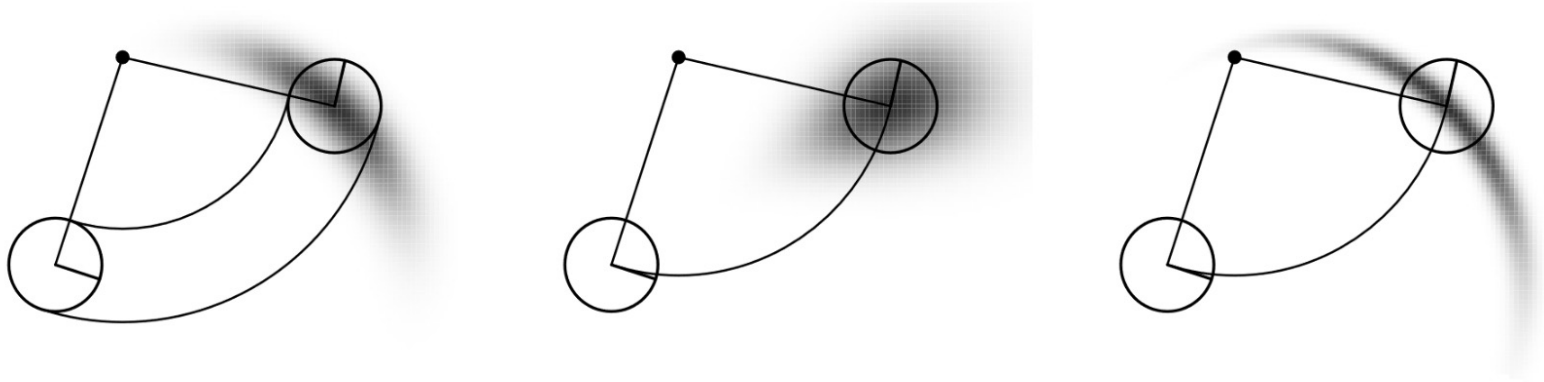
$$\boldsymbol{\theta_{k+1} = \theta_k + \Delta\theta}$$

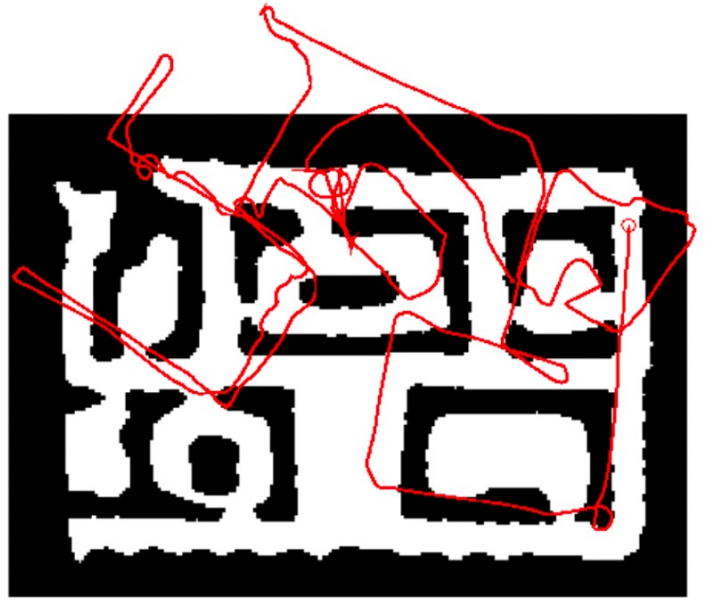
- De facto forward kinematics implementation
- Works for small steps (any time encoder data are available)

Errors









BURGARD, Wolfram et al. Probabilistic Motion Models [online]. Uni Freiburg, 2011 [cit. 2022-02-16]. Available at: http://ais.informatik.uni-freiburg.de/teaching/ss11/robotics/slides/06-motion-models.pdf
THRUN, Sebastian, Wolfram BURGARD and Dieter FOX. *Probabilistic robotics*. Massachusetts: MIT Press, 2006. ISBN 978-0-262-20162-9.

Practical Implementation Details

Calibrate

- Adjust **R** and **L** via experimental measurements
- Identify systematic biases in encoders
- Driving in a straight line or rotating in place and comparing measured vs. expected distance/angle.

Consider

- Sampling rate: how often you read encoder values
- Filter and smooth data to reduce noise
- Maintain up-to-date robot states in a consistent coordinate frame

Limitate Error Sources

- Wheel slip or uneven floor can cause encoder misreadings
- Variations in wheel size or friction
- Integration error accumulates over time
- Periodic recalibration or sensor fusion often needed (e.g., IMU, vision)

Advanced: Kalman Filter

Combines predictions from a model with real-world sensor data to reduce noise and improve accuracy.

- **Predictive Power:** Uses a motion model to predict the next state of the robot.
- **Measurement Update:** Incorporates sensor data to correct the prediction.
- **Noise Reduction:** Filters out measurement noise and sensor inaccuracies.
- **Smooth Trajectory:** Helps avoid abrupt jumps in position estimation, leading to more accurate and reliable odometry.

$$\bar{\mu}_t = g(u_t, \mu_{t-1})$$

$$\bar{\sum}_t = G_t \sum_{t-1} G_t^T + R_t$$

$$K_t = \bar{\sum}_t H_t^T (H_t \bar{\sum}_t H_t^T + Q_t)^{-1}$$

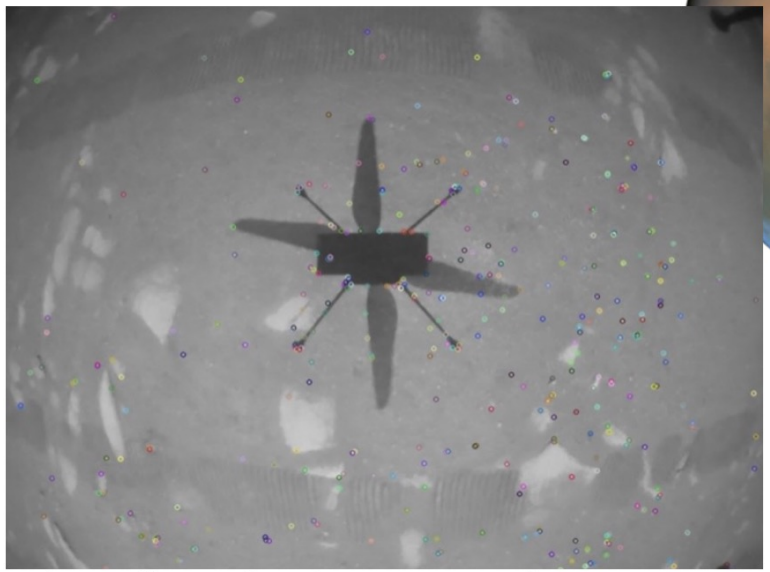
$$\mu_t = \bar{\mu}_t + K_t(z_t - h(\bar{\mu}_t))$$

$$\sum_t = (I - K_t H_t \bar{\sum}_t)$$

Advanced: Visual-Inertial Odometry

- Combines visual data (e.g., from a camera) with inertial data (e.g., from an IMU) to estimate a robot's position and orientation.
- Utilizes both the visual cues from the environment and motion data from accelerometers and gyroscopes.
- **Visual Data:** Tracks features in images to estimate movement.
- **Inertial Data:** Provides high-frequency motion updates, even when visual tracking is poor.
- **Fusion Algorithm:** Often uses an Extended Kalman Filter or similar method to merge the two data sources.

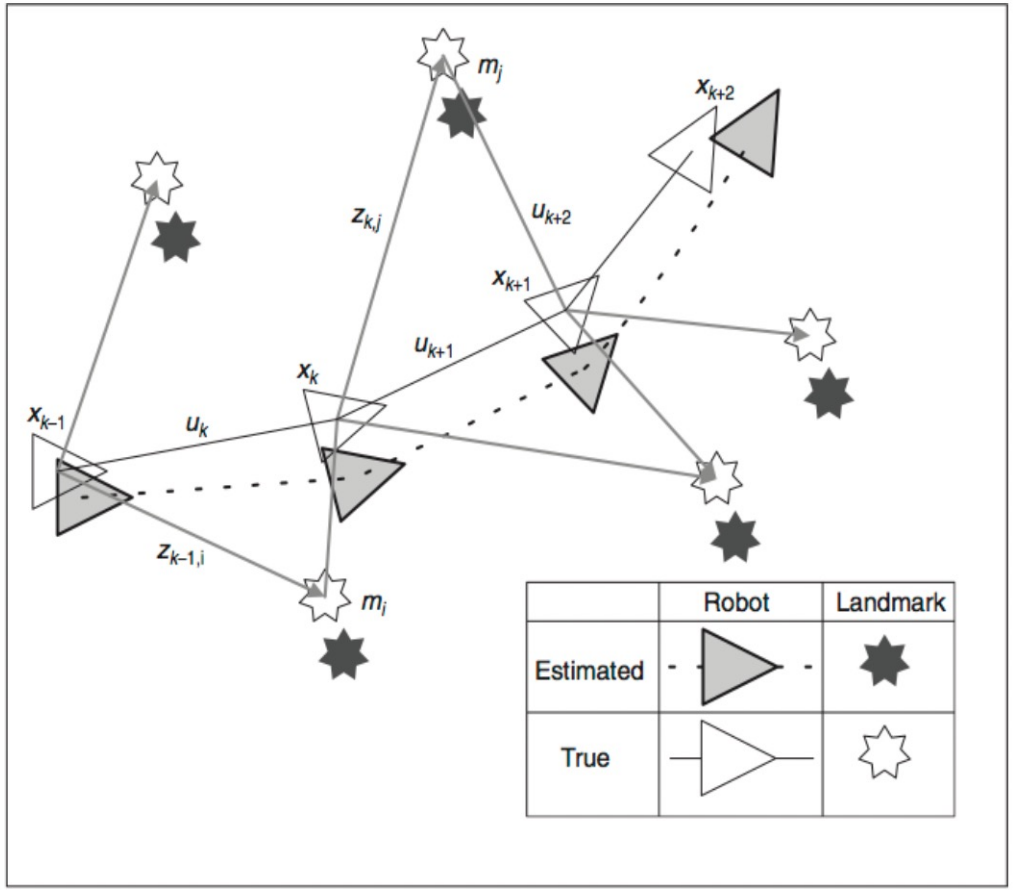
www.kosmonautix.cz



https://www.therobotreport.com/ingenuity-helicopter-state-estimation-localization/

Advanced: SLAM (Simultaneous Localization and Mapping)

- SLAM is a method used by robots and autonomous vehicles to build a map of an unknown environment while simultaneously keeping track of their location within it.
- Integrates sensor data (e.g., LIDAR, cameras, IMU) to construct a real-time map and determine position.
- **Mapping:** Uses sensor data to create a spatial representation of the environment.
- **Localization:** Continuously updates the robot's position on the generated map.
- **Loop Closure:** Detects previously visited locations to improve map accuracy and reduce drift.



http://gopalmenon.github.io/Raspberri-Pi-GoPiGo-Robot-EKF-SLAM-Manuscript/images/EssSlam.png

Adam Ligocki

ligocki@vutbr.cz

https://www.linkedin.com/in/adamligocki/

https://github.com/adamek727

Robotics and AI Research Group

https://github.com/Robotics-BUT

Brno University of Technology
Faculty of Electrical Engineering and Communication
Department of Control and Instrumentation