German Research Center for Artificial Intelligence

ROBOT ODOMETRY

M. Sc. Mihaela Popescu
Robotics Innovation Center
DFKI Bremen

Prof. Dr. Dr. h.c. Frank Kirchner
Arbeitsgruppe Robotik, Universität Bremen
`https://robotik.dfki-bremen.de/`
`robotics@dfki.de`

8th November, 2022 – Bremen, Deutschland

Universität Bremen

German Research Center for Artificial Intelligence

Universität Bremen

Introduction

German Research Center for Artificial Intelligence

Definition

"Odometry is the use of data from motion sensors to estimate the change in position over time." *https://en.wikipedia.org/wiki/Odometry*

German Research Center for Artificial Intelligence

Definition

"Odometry is the use of data from motion sensors to estimate the change in position over time." *https://en.wikipedia.org/wiki/Odometry*

The word *Odometry* comes from the Greek words:

- *hodos* meaning travel or journey, and
- *metron* meaning measure.

German Research Center for Artificial Intelligence

Definition

"Odometry is the use of data from motion sensors to estimate the change in position over time." *https://en.wikipedia.org/wiki/Odometry*

The word *Odometry* comes from the Greek words:

- *hodos* meaning travel or journey, and
- *metron* meaning measure.

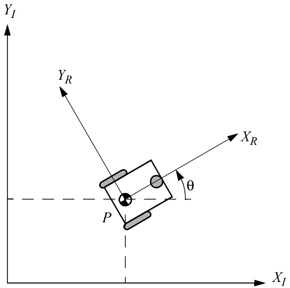
Odometry is also known as *dead reckoning*.

Universität Bremen

German Research Center for Artificial Intelligence

Robot Pose

- For simplicity, we assume that the mobile robot navigates on a two-dimensional plane.
- In this case, the robot position and orientation is described by the following state vector:
  - position (x_R, y_R)
  - orientation θ.



Universität Bremen

Types of Localization

- **Relative localization** (also called incremental localization or pose tracking)
  The change of position at successive points in time is incrementally calculated and integrated relative to the starting position. ⇒ *Odometry*

- **Global localization**
  The robot position is determined with respect to an external reference system, e.g. to a given map or a global coordinate system. ⇒ *Localization*

German Research Center for Artificial Intelligence

Types of Localization

- **Relative localization** (also called incremental localization or pose tracking)
  The change of position at successive points in time is incrementally calculated and integrated relative to the starting position. $\Rightarrow$ *Odometry*

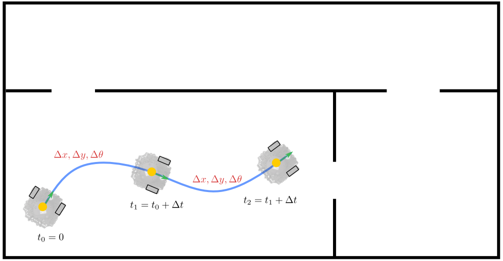
- **Global localization**
  The robot position is determined with respect to an external reference system, e.g. to a given map or a global coordinate system. $\Rightarrow$ *Localization*

$\rightarrow$ For more precision, the two methods are often combined.

German Research Center for Artificial Intelligence

Odometry

- Where is the robot relative to its previous position?



Incremental Localization

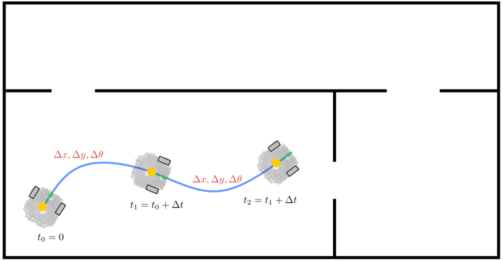
German Research Center for Artificial Intelligence

Odometry

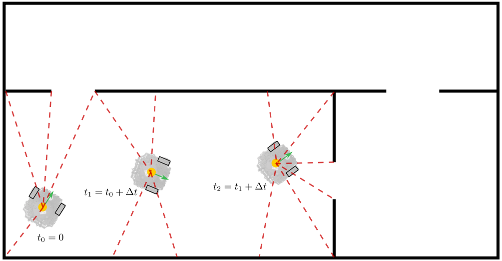
- Where is the robot relative to its previous position?

Localization

- Where is the robot in the world? *(future lecture on localization)*



Incremental Localization



Absolute Localization

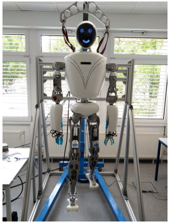
Universität Bremen

German Research Center for Artificial Intelligence

Applications

The position of mobile robots is required for tasks such as:

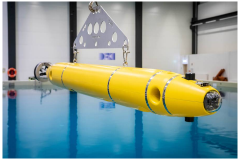
- mapping an environment
- path planning
- autonomous navigation
- ...



Humanoid Robot

Space Rover

Smart Car



Underwater Robot

Odometry Models

Odometry Basic Concept

- Odometry is used in robotics by wheeled or legged robots to estimate their position relative to a starting location.

German Research Center for Artificial Intelligence

Odometry Basic Concept

- Odometry is used in robotics by wheeled or legged robots to estimate their position relative to a starting location.
- Basic concept:
  - Develop a mathematical model of how the motion of the robot's wheels, joints, etc. induce motion of the vehicle itself.
  - Integrate (add up) these motions over time to calculate the robot current position with respect to the previous (initial) location.

What sensors can be used to measure the robot's motion?

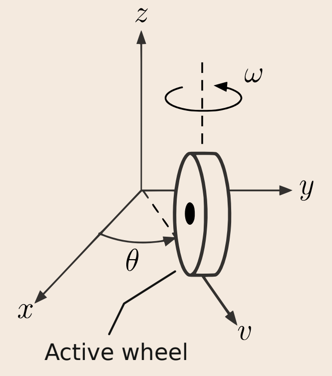
- **Wheel encoder** (wheel odometry) $\rightarrow$ in this lecture
- Joint encoders (leg odometry)
- IMU: Accelerometer and Gyroscope (inertial odometry)
- Camera (visual odometry)
- Laser scanner (lidar odometry)

$\Rightarrow$ Sensor fusion

The odometry estimation varies by vehicle design:

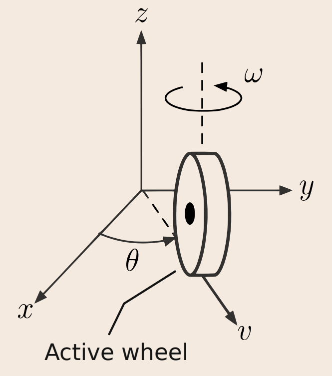
- Unicycle robot model
- Spherical robot
- Bicycle robot model
- Differential drive robot
- Ackermann steering vehicle
- Omnidirectional robot
- ...

German Research Center for Artificial Intelligence

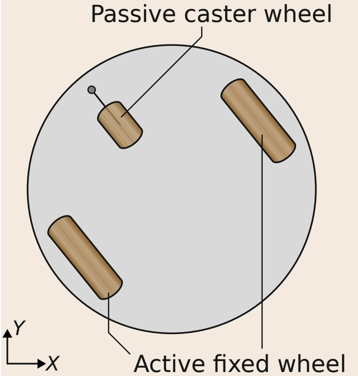


Unicycle Model: single upright wheel rolling without sideways slip.

German Research Center for Artificial Intelligence

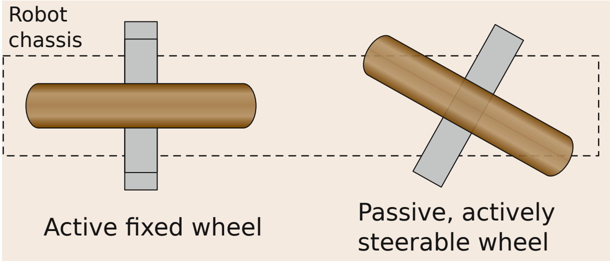


Unicycle Model: single upright wheel rolling without sideways slip.



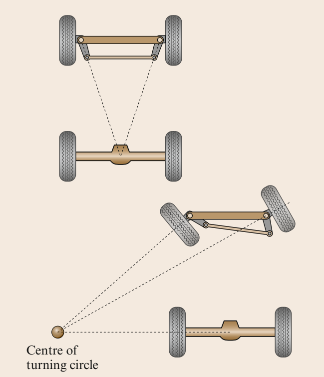
Differential Drive Model: two independently driven wheels and a caster wheel.

Universität Bremen



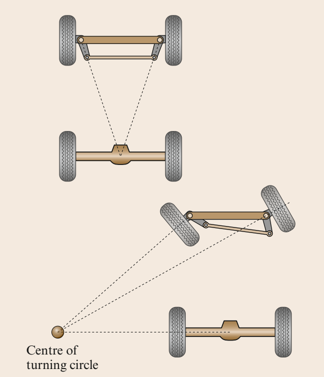
Bicycle model: steerable front wheel and active rear wheel.

German Research Center for Artificial Intelligence

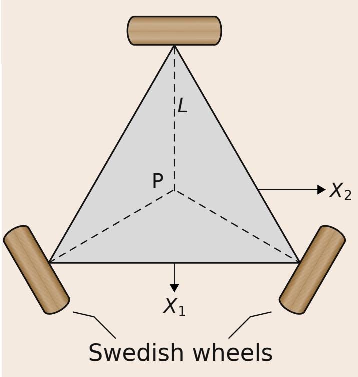


Ackermann Steering Model: two fixed active back wheels and two passive front wheels steered at different angles.

Universität Bremen



Ackermann Steering Model: two fixed active back wheels and two passive front wheels steered at different angles.



Omnidirectional Model: three or more Swedish wheels.

German Research Center for Artificial Intelligence

Robot Motion Constraints

- Based on the motion they can perform, wheeled robots can be:
  - Holonomic (omnidirectional robot)
  - Nonholonomic (car-like robot)

Universität Bremen

German Research Center for Artificial Intelligence

Robot Motion Constraints

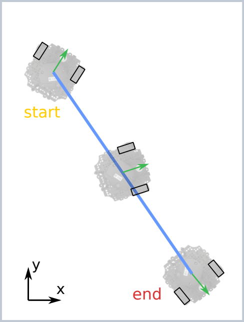
- Based on the motion they can perform, wheeled robots can be:
  - Holonomic (omnidirectional robot)
  - Nonholonomic (car-like robot)
- This depends in part on the robot wheels type:
  - Nonholonomic mobile robots employ conventional wheels without sideways slip.
  - Holonomic mobile robots employ Swedish wheels that allow sideways sliding.

Mobile robot with Swedish wheels.

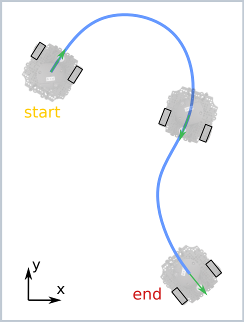
German Research Center for Artificial Intelligence

Robot Motion Constraints

- Nonholonomic robots have a constraint on velocity, which means that they can not have pure lateral velocity.
- Despite the velocity constraint, nonholonomic robots can reach any configuration (x, y, θ) in an obstacle-free plane.



Holonomic robot



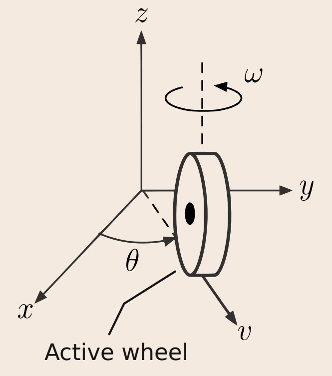
Nonolonomic robot

Universität Bremen

Unicycle Model

Unicycle Model

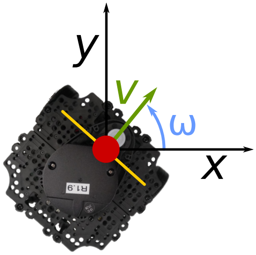
- Composed by a single upright wheel rolling without sideways slip.
- This type of robot is unstable without dynamic control to maintain balance.
- The robot state is described by its position (x, y) and orientation θ.
- We can control its linear velocity v and angular velocity ω.



German Research Center for Artificial Intelligence

Unicycle Kinematic Model

- Consider the TurtleBot to be a point robot located at the middle distance between the two wheels (red dot).
- The velocity of the robot is represented by the linear velocity $v\left[\frac{m}{s}\right]$ and angular velocity $\omega\left[\frac{rad}{s}\right]$.



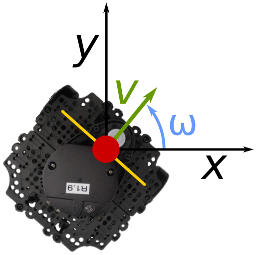
Universität Bremen

German Research Center for Artificial Intelligence

Unicycle Kinematic Model

- Consider the TurtleBot to be a point robot located at the middle distance between the two wheels (red dot).
- The velocity of the robot is represented by the linear velocity $v \left[\frac{m}{s}\right]$ and angular velocity $\omega \left[\frac{rad}{s}\right]$.
- The robot incremental motion is described as a:
  - rotation of $\Delta\theta = \omega\Delta t$ radians counterclockwise and
  - translation of $\Delta d = v\Delta t$ meters forward,

  where Δt is the sampling time.



Universität Bremen

German Research Center for Artificial Intelligence

Numerical Example

- Consider the following linear and angular velocity:

$$v = 2\ m/s$$

$$\omega = \pi\ rad/s$$

$$\Delta t = 0.1\ s$$

- The robot starts at the initial position $(x_0, y_0) = (0, 0)$ and orientation $\theta_0 = 0$.
- The robot first executes the angular motion, then the linear displacement.

German Research Center for Artificial Intelligence

Solution of Numerical Example

- Compute the robot incremental pose (position x, y and orientation θ):

$$x = x_0 + \Delta d \cos\theta$$

$$y = x_0 + \Delta d \sin\theta$$

$$\theta = \theta_0 + \Delta\theta$$

German Research Center for Artificial Intelligence

Solution of Numerical Example

- Compute the robot incremental pose (position x, y and orientation θ):

$$x = x_0 + \Delta d \cos\theta \qquad x = x_0 + v\Delta t \cos\theta$$
$$y = x_0 + \Delta d \sin\theta \qquad y = x_0 + v\Delta t \sin\theta$$
$$\theta = \theta_0 + \Delta\theta \qquad \theta = \theta_0 + \omega\Delta t$$

German Research Center for Artificial Intelligence

Solution of Numerical Example

- Compute the robot incremental pose (position x, y and orientation θ):

$$x = x_0 + \Delta d \cos\theta \qquad x = x_0 + v\Delta t \cos\theta$$

$$y = x_0 + \Delta d \sin\theta \qquad y = x_0 + v\Delta t \sin\theta$$

$$\theta = \theta_0 + \Delta\theta \qquad \theta = \theta_0 + \omega\Delta t \qquad \theta = 0 + \pi * 0.1 = \pi/10 \approx 0.3141 \ rad$$

German Research Center for Artificial Intelligence

Solution of Numerical Example

- Compute the robot incremental pose (position x, y and orientation θ):

$$x = x_0 + \Delta d \cos\theta \quad x = x_0 + v\Delta t \cos\theta \quad x = 0 + 2 * 0.1 \cos(\pi/10) \approx 0.1902\ m$$

$$y = x_0 + \Delta d \sin\theta \quad y = x_0 + v\Delta t \sin\theta \quad y = 0 + 2 * 0.1 \sin(\pi/10) \approx 0.0618\ m$$

$$\theta = \theta_0 + \Delta\theta \quad \theta = \theta_0 + \omega\Delta t \quad \theta = 0 + \pi * 0.1 = \pi/10 \approx 0.3141\ rad$$

German Research Center for Artificial Intelligence

Solution of Numerical Example

- Compute the robot incremental pose (position x, y and orientation θ):

$$x = x_0 + \Delta d \cos\theta \qquad x = x_0 + v\Delta t \cos\theta \qquad x = 0 + 2 * 0.1 \cos(\pi/10) \approx 0.1902\ m$$

$$y = x_0 + \Delta d \sin\theta \qquad y = x_0 + v\Delta t \sin\theta \qquad y = 0 + 2 * 0.1 \sin(\pi/10) \approx 0.0618\ m$$

$$\theta = \theta_0 + \Delta\theta \qquad \theta = \theta_0 + \omega\Delta t \qquad \theta = 0 + \pi * 0.1 = \pi/10 \approx 0.3141\ rad$$

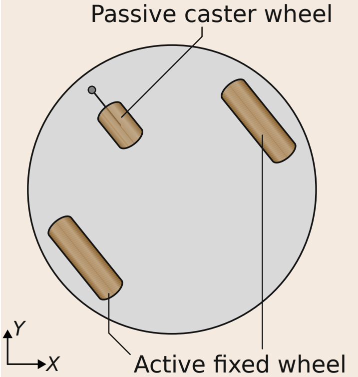
- At every time step, the current robot pose can be computed incrementally by reapplying the formulas above with the previously computed robot pose.

Universität Bremen

Differential Drive Model

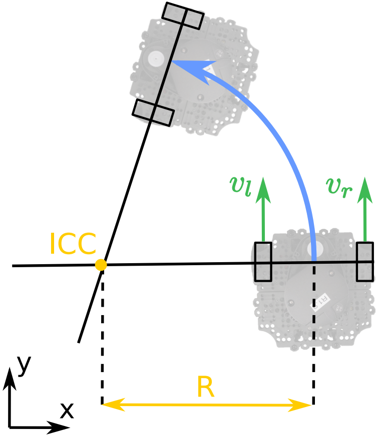
Differential Drive Model

- It has two driveable wheels which are independently controllable, mounted along a common axis.
- One or more low-friction caster wheels are used to keep the robot horizontal.



Differential Drive Model

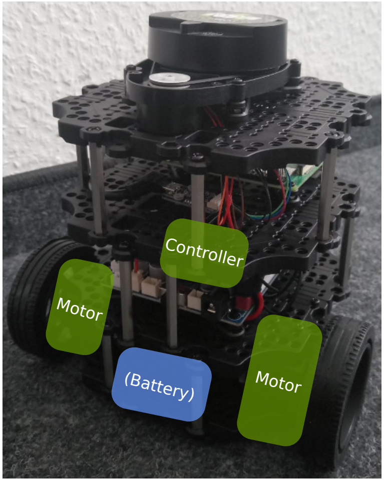
- The vehicle rotates around a point known as the ICC (**I**nstantaneous **C**enter of **C**urvature) that lies on the wheels' common axis.
- The robot state is described by its position (x, y) and orientation θ.
- We can control the robot's linear velocity v and angular velocity ω, by setting the linear velocity of the left wheel $\boldsymbol{v_l}$ and right wheel $\boldsymbol{v_r}$.



From a point robot to two wheels

Instead of a point robot, the TurtleBot is now modeled as a robot with:

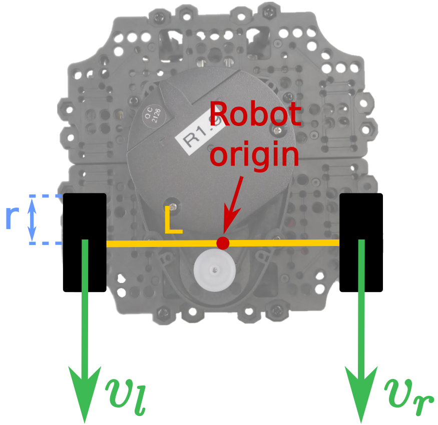
- two wheels,
- two motors, and
- a motor controller.



German Research Center for Artificial Intelligence

From a point robot to two wheels

- The left and right wheel speeds are denoted by v_l and v_r.
- The wheels have radius r.
- The common axis where the wheels are mounted has length L (wheel base).
- The robot origin is at the middle distance between the two wheels.

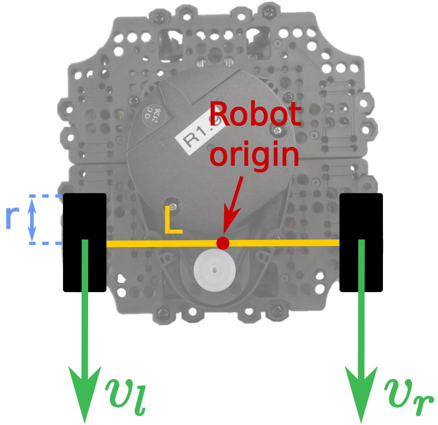


Universität Bremen

German Research Center for Artificial Intelligence

Differential Drive Model

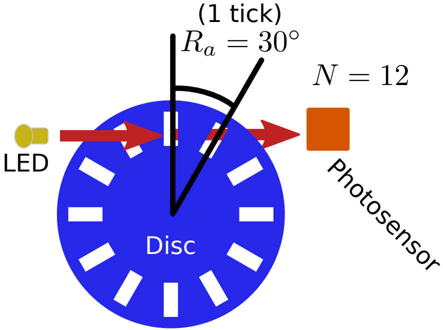
- Both wheels move with equal velocity: → straight line.
- Both wheels move with opposite velocity: → turn in place.
- One wheel moves slower than the other: → turn in the direction of the slower wheel following a curved path.



Universität Bremen

Wheel Encoder

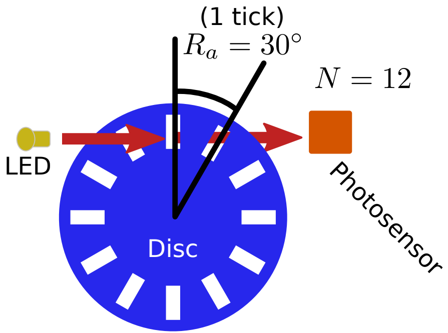
- The encoder outputs the number of resolutions n (tick counts) that the motor has turned in a time interval Δt.



German Research Center for Artificial Intelligence

Wheel Encoder

- The encoder outputs the number of resolutions n (tick counts) that the motor has turned in a time interval Δt.
- We know:
  - the encoder number of ticks N
  - angular resolution
    $\boldsymbol{R_a = 2\pi / N}$ $[rad/tick]$
  - on-ground distance resolution
    $\boldsymbol{R_d = 2\pi r / N}$ $[m/tick]$

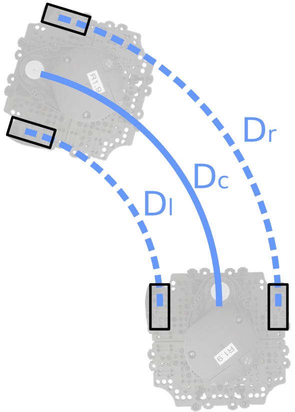


Universität Bremen

German Research Center for Artificial Intelligence

How far has the robot traveled?

- The robot wheels follow an arc trajectory.
- The distance traveled by the left and right wheels is given by the formula:
  $D_l = D_r = R_d * n$

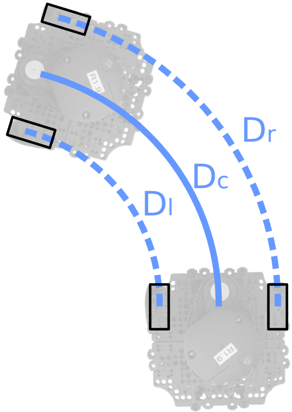


Universität Bremen

German Research Center for Artificial Intelligence

How far has the robot traveled?

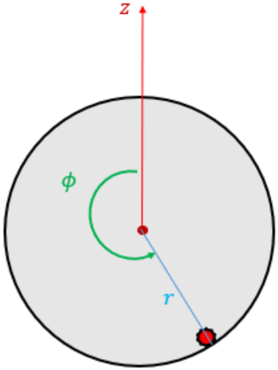
- The robot wheels follow an arc trajectory.
- The distance traveled by the left and right wheels is given by the formula:
  $D_l = D_r = R_d * n$
- The distance traveled by the robot is the average of the left and right wheel distances:
  $D_c = \frac{D_l + D_r}{2}$



German Research Center for Artificial Intelligence

Wheel Velocities

- We denote by ϕ the wheel orientation with respect to the world z-axis.



Wheel orientation ϕ and radius r.
Source: `https://www.roboticsbook.org/S52_diffdrive_actions.html`

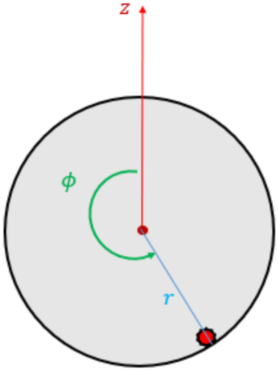
Universität Bremen

German Research Center for Artificial Intelligence

Wheel Velocities

- We denote by ϕ the wheel orientation with respect to the world z-axis.
- The wheel rotation speed is given by:

$$\dot{\phi} = \frac{R_a * n}{\Delta t}$$



Wheel orientation ϕ and radius r.
Source: https://www.roboticsbook.org/S52_diffdrive_actions.html

Universität Bremen

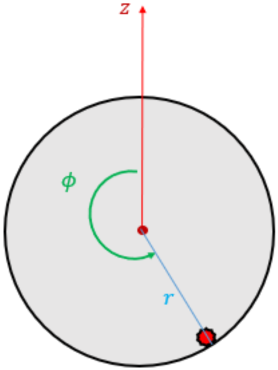
Wheel Velocities

- We denote by ϕ the wheel orientation with respect to the world z-axis.
- The wheel rotation speed is given by:

$$\dot{\phi} = \frac{R_a * n}{\Delta t}$$

- Relationship between the wheel rotation speed and linear wheel velocity:

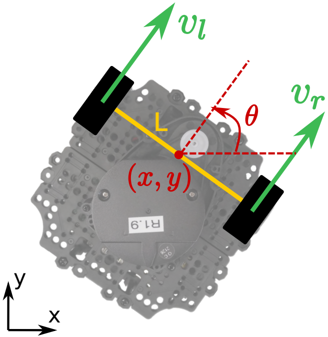
$$v = r\dot{\phi}$$



Wheel orientation ϕ and radius r.
Source: https://www.roboticsbook.org/S52_diffdrive_actions.html

How to compute the robot's location?

- Known: wheel speed control inputs (v_l, v_r) and the initial robot pose (x_0, y_0, θ_0).



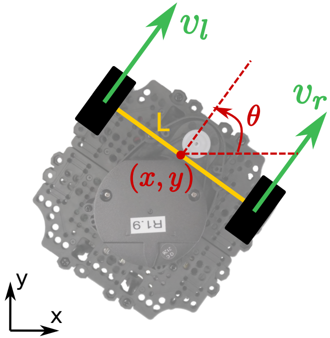
How to compute the robot's location?

- Known: wheel speed control inputs (v_l, v_r) and the initial robot pose (x_0, y_0, θ_0).
- Incrementally estimate the robot state using the following motion model for the time interval Δt:

$$x = x_0 + \frac{(v_r + v_l)}{2} \Delta t * cos\theta$$

$$y = y_0 + \frac{(v_r + v_l)}{2} \Delta t * sin\theta$$

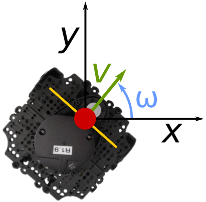
$$\theta = \theta_0 + \frac{(v_r - v_l)}{L} \Delta t$$



German Research Center for Artificial Intelligence

How to compute the wheel speeds?

- Known: desired linear velocity v and angular velocity ω.
- What are the left and right wheel speeds (v_l, v_r)?



German Research Center for Artificial Intelligence

How to compute the wheel speeds?

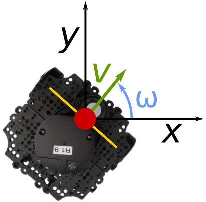
- Known: desired linear velocity v and angular velocity ω.
- What are the left and right wheel speeds (v_l, v_r)?
- Exploit the unicycle model:

Unicycle model

$$x = x_0 + v\Delta t \cos\theta$$

$$y = x_0 + v\Delta t \sin\theta$$

$$\theta = \theta_0 + \omega\Delta t$$



German Research Center for Artificial Intelligence

How to compute the wheel speeds?

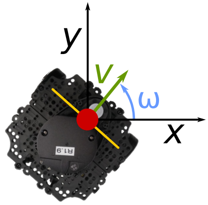
- Known: desired linear velocity v and angular velocity ω.
- What are the left and right wheel speeds (v_l, v_r)?
- Exploit the unicycle model:

Unicycle model

$$x = x_0 + v\Delta t \cos\theta$$
$$y = x_0 + v\Delta t \sin\theta$$
$$\theta = \theta_0 + \omega\Delta t$$

Differential drive model

$$x = x_0 + \frac{(v_r + v_l)}{2}\Delta t * cos\theta$$
$$y = y_0 + \frac{(v_r + v_l)}{2}\Delta t * sin\theta$$
$$\theta = \theta_0 + \frac{(v_r - v_l)}{L}\Delta t$$



German Research Center for Artificial Intelligence

How to compute the wheel speeds?

Unicycle model

$$x = x_0 + v\Delta t \cos\theta$$

$$y = x_0 + v\Delta t \sin\theta$$

$$\theta = \theta_0 + \omega\Delta t$$

Differential drive model

$$x = x_0 + \frac{(v_r + v_l)}{2}\Delta t * cos\theta$$

$$y = y_0 + \frac{(v_r + v_l)}{2}\Delta t * sin\theta$$

$$\theta = \theta_0 + \frac{(v_r - v_l)}{L}\Delta t$$

Universität Bremen

German Research Center for Artificial Intelligence

How to compute the wheel speeds?

Unicycle model

$$x = x_0 + v\Delta t \cos\theta$$

$$y = x_0 + v\Delta t \sin\theta$$

$$\theta = \theta_0 + \omega\Delta t$$

Differential drive model

$$x = x_0 + \frac{(v_r + v_l)}{2}\Delta t * cos\theta$$

$$y = y_0 + \frac{(v_r + v_l)}{2}\Delta t * sin\theta$$

$$\theta = \theta_0 + \frac{(v_r - v_l)}{L}\Delta t$$

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

$$\omega = \frac{(v_r - v_l)}{L}$$

German Research Center for Artificial Intelligence

How to compute the wheel speeds?

Unicycle model	*Differential drive model*
$x = x_0 + v\Delta t \cos\theta$	$x = x_0 + \frac{(v_r + v_l)}{2}\Delta t * cos\theta$
$y = x_0 + v\Delta t \sin\theta$	$y = y_0 + \frac{(v_r + v_l)}{2}\Delta t * sin\theta$
$\theta = \theta_0 + \omega\Delta t$	$\theta = \theta_0 + \frac{(v_r - v_l)}{L}\Delta t$

$$v = \frac{(v_r + v_l)}{2} \quad \Rightarrow \quad 2v = v_r + v_l$$

$$\omega = \frac{(v_r - v_l)}{L} \quad \Rightarrow \quad \omega L = v_r - v_l$$

German Research Center for Artificial Intelligence

How to compute the wheel speeds?

Unicycle model

$$x = x_0 + v\Delta t \cos\theta$$

$$y = x_0 + v\Delta t \sin\theta$$

$$\theta = \theta_0 + \omega\Delta t$$

Differential drive model

$$x = x_0 + \frac{(v_r + v_l)}{2}\Delta t * cos\theta$$

$$y = y_0 + \frac{(v_r + v_l)}{2}\Delta t * sin\theta$$

$$\theta = \theta_0 + \frac{(v_r - v_l)}{L}\Delta t$$

$$v = \frac{(v_r + v_l)}{2} \Rightarrow 2v = v_r + v_l \Rightarrow v_l = \frac{2v - \omega L}{2}$$

$$\omega = \frac{(v_r - v_l)}{L} \Rightarrow \omega L = v_r - v_l \Rightarrow v_r = \frac{2v + \omega L}{2}$$

German Research Center for Artificial Intelligence

How to compute the wheel speeds?

Unicycle model

$$x = x_0 + v\Delta t \cos\theta$$

$$y = x_0 + v\Delta t \sin\theta$$

$$\theta = \theta_0 + \omega\Delta t$$

Differential drive model

$$x = x_0 + \frac{(v_r + v_l)}{2}\Delta t * cos\theta$$

$$y = y_0 + \frac{(v_r + v_l)}{2}\Delta t * sin\theta$$

$$\theta = \theta_0 + \frac{(v_r - v_l)}{L}\Delta t$$

$$v = \frac{(v_r + v_l)}{2} \Rightarrow 2v = v_r + v_l \Rightarrow v_l = \frac{2v - \omega L}{2} \Rightarrow \dot{\phi}_l = \frac{2v - \omega L}{2r}$$

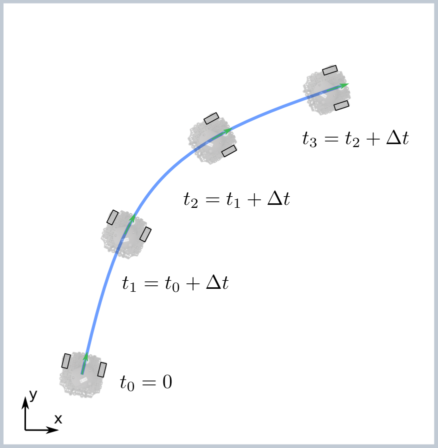
$$\omega = \frac{(v_r - v_l)}{L} \Rightarrow \omega L = v_r - v_l \Rightarrow v_r = \frac{2v + \omega L}{2} \Rightarrow \dot{\phi}_r = \frac{2v - \omega L}{2r}$$

Odometry Drift

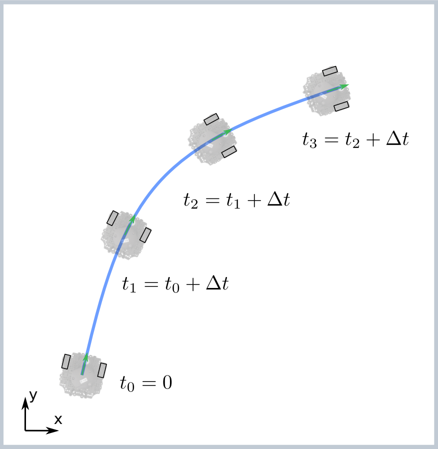
German Research Center for Artificial Intelligence

Real World Problems

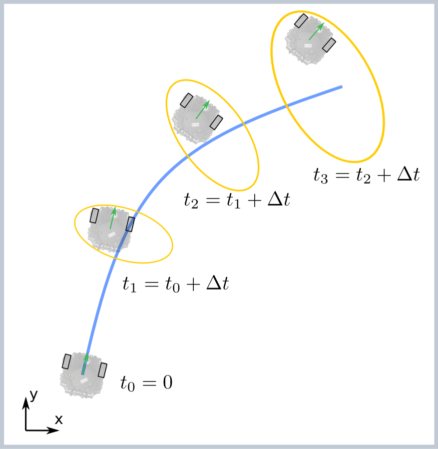
- For our models, we assume the robot drives in a perfect world where we can measure exactly how far the robot moved.
- In time, the odometry drifts away from the real robot position due to **errors incrementally adding up** at every time step $\rightarrow$ the error is unbounded!
- A source of errors is the **integration of velocity measurements over time** to give position estimates (velocity is assumed to be constant for the time interval Δt).



Ideal case (no odometry drift)



Ideal case (no odometry drift)



Real case (with odometry drift)

Odometry Drift

Examples

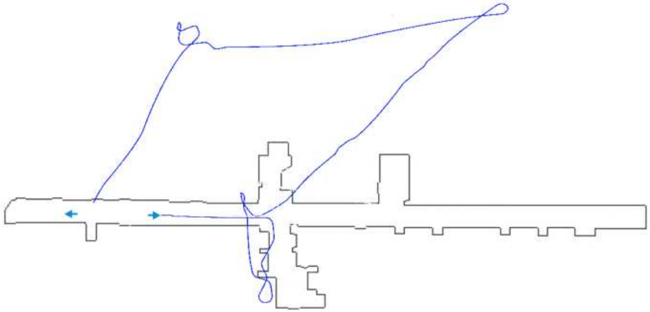
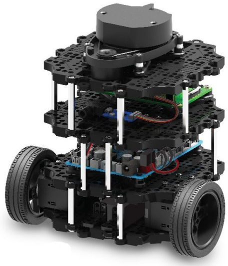


Figure – Result of odometry when driving through the corridor of an office building. Errors in the turning angle estimation θ lead to very large errors in the Cartesian position estimation (x, y).
(Source: Mobile Roboter, Chapter 4: Fortbewegung)

German Research Center for Artificial Intelligence

Odometry for Wheeled Robots

- Problems that can occur in wheel odometry:
  - Incorrect wheel size (e.g. wheel compaction).
  - Wheel slippage on icy or wet surfaces.
  - Losing contact to the ground.
  - Encoder inaccuracies (e.g. low resolution).



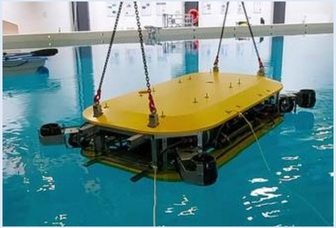
TurtleBot 3 Burger.

Odometry for Flying, Walking and Diving Robots

Odometry can be hard to obtain in different environments:

- Wind/water currents move flying/diving robots.
- More difficult to calculate on walking robots (forward kinematics).

Microcopter
TZI/Uni Bremen Student project
Autonomous QuadCopter 2010



AUV Cuttlefish
DFKI, 2021

Mantis
DFKI, 2016

Conclusions and Further Reading

German Research Center for Artificial Intelligence

Summary

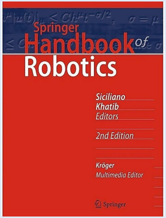
- Odometry provides the robot's **incremental change** in position and orientation.
- For wheel odometry computation, the robot **motion model** is needed.
- Some of the most important wheeled robot models are the **unicycle model**, the **differential drive**, the bicycle model, the Ackermann steering model, and the omnidirectional drive.
- **Odometry drift** occurs due to time integration of velocity measurements and failures of wheel odometry (e.g. wheel slip, inaccurate wheel dimensions).
- Robots often use both **local and global localization** methods.

German Research Center for Artificial Intelligence

Additional Literature

Springer Handbook of Robotics (English)

- Chapter 20.1: Odometry



Source: https://link.springer.com/referencework/10.1007%2F978-3-540-30301-5

Mobile Roboter (German)

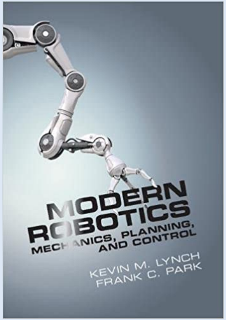
- Chapter 4.1, 4.2: Fortbewegung

Source: https://link.springer.com/book/10.1007/978-3-642-01726-1

Additional Literature

Modern Robotics
(textbook and video lectures)

- Chapter 13.3.1: Modeling of Nonholonomic Wheeled Mobile Robots
  https://youtu.be/fPHVhlRFFCk
- Chapter 13.4: Odometry
  https://youtu.be/eQ9EOZvp9jw



Source: http://hades.mech.northwestern.edu/images/7/7f/MR.pdf

Next: Mapping.