DTU

Software for Autonomous Systems
SFfAS-31391:

Autonomous Guided Vehicles

Lecturer, Course Coordinator: Evangelos Boukas—PhD

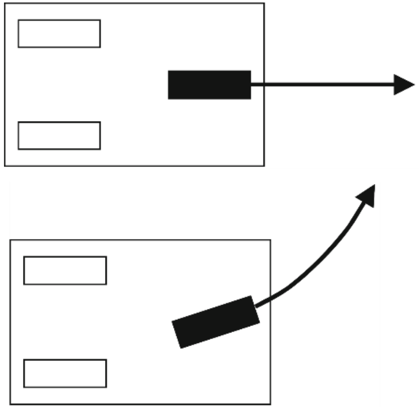
Outline

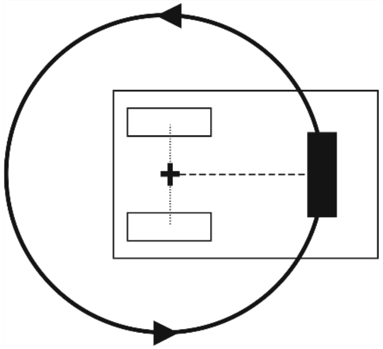
- Types of Locomotion
- Differential Drive Kinematics – (One Pager)
- AGV System as a whole!
- Localization & Mapping
- Navigation – Path Planning

- And of course we'll be Hands on!!!

Types of Mobile Robot Locomotion - Steering

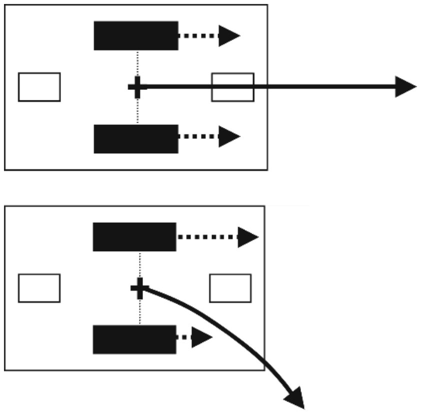
- Single Wheel
  - Working Principle
    - Single wheel for Driving and Steering
  - Pros
    - Linear and Angular Velocities Decoupled
  - Cons
    - Cannot handle Complex Terrains

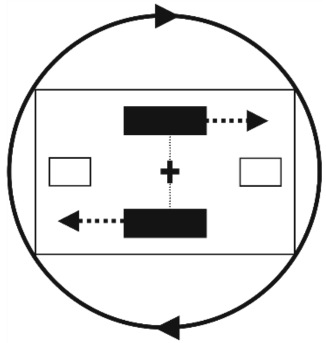




Types of Mobile Robot Locomotion - Steering

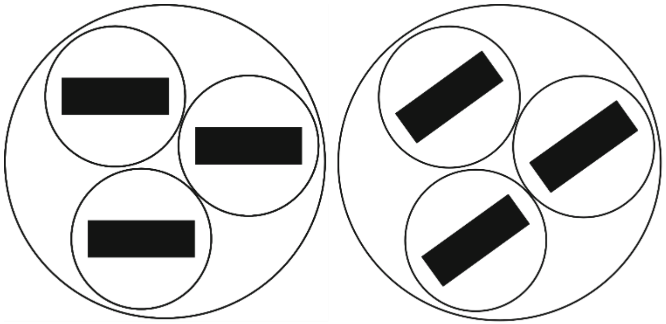
- Single Wheel
- Differential
  - Working Principle
    - Two fixed driving wheels
    - One caster wheel
  - Pros
    - Simplicity
  - Cons
    - Difficulty of calculating odometry/ Driving Straight





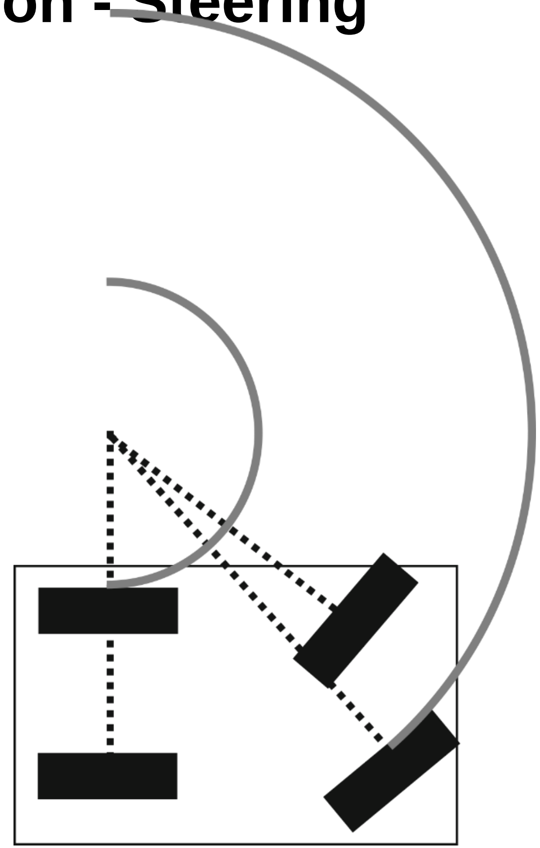
Types of Mobile Robot Locomotion - Steering

- Single Wheel
- Differential
- Synchro Drive
  - Working Principle
    - Three wheels rotating and driving identically
  - Pros
    - Almost holonomic
  - Cons
    - Has to stop to rotate



Types of Mobile Robot Locomotion - Steering

- Single Wheel
- Differential
- Synchro Drive
- Ackerman
  - Working Principle
    - All wheels on the tangent of circle
  - Pros
    - Different motor for drive and steering
  - Cons
    - Planning is difficult, non holonomic

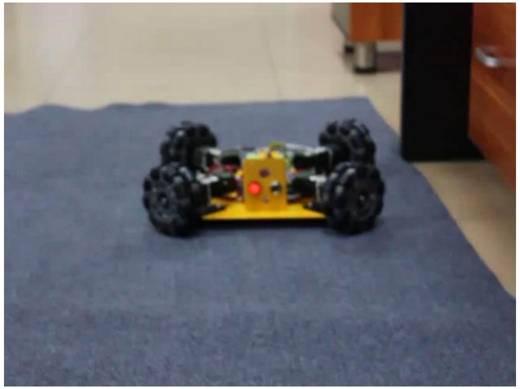


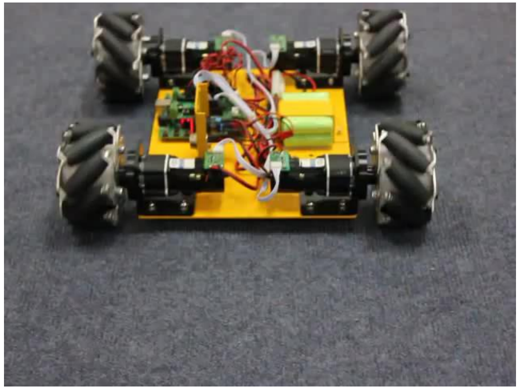
Types of Mobile Robot Locomotion - Steering

- Single Wheel
- Differential
- Synchro Drive
- Ackerman
- Skid Steering
  - Working Principle
    - Differential, no caster
  - Pros
    - Rugged, Robust
  - Cons
    - Wheel Odometry is veeery hard

Types of Mobile Robot Locomotion - Steering

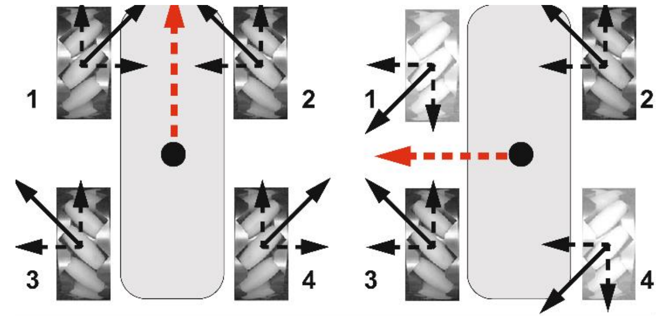
- Single Wheel
- Differential
- Synchro Drive
- Ackerman
- Skid Steering
- Omni-Directional - Mecanum Wheels

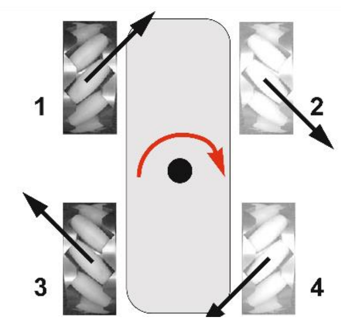




Types of Mobile Robot Locomotion - Steering

- Single Wheel
- Differential
- Synchro Drive
- Ackerman
- Skid Steering
- Omni-Directional - Mecanum Wheels
  - Working Principle
    - Perpendicular vector
  - Pros
    - Holonomic
  - Cons
    - Hard to get grip(friction)





Differential Drive Kinematics

- Forward:

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = 2\pi r \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ -\frac{1}{d} & \frac{1}{d} \end{bmatrix} \begin{bmatrix} \dot{\theta}_L \\ \dot{\theta}_R \end{bmatrix}$$

where:

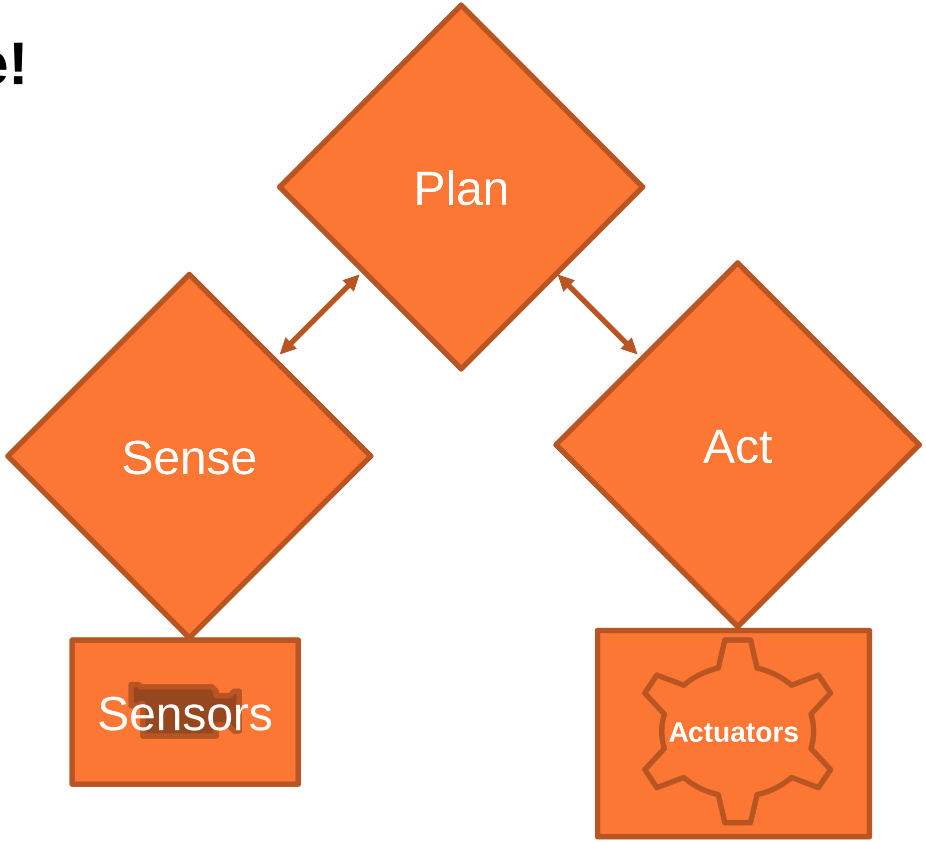
v	is the vehicle's linear speed (equals ds/dt or $\dot{s}$),
ω	is the vehicle's rotational speed (equals $d\varphi/dt$ or $\dot{\varphi}$),
$\dot{\theta}_{L,R}$	are the individual wheel speeds in revolutions per second,
r	is the wheel radius,
d	is the distance between the two wheels.

- Inverse

$$\begin{bmatrix} \dot{\theta}_L \\ \dot{\theta}_R \end{bmatrix} = \frac{1}{2\pi r} \begin{bmatrix} 1 & -\frac{d}{2} \\ 1 & \frac{d}{2} \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}$$

AGV System as a whole!

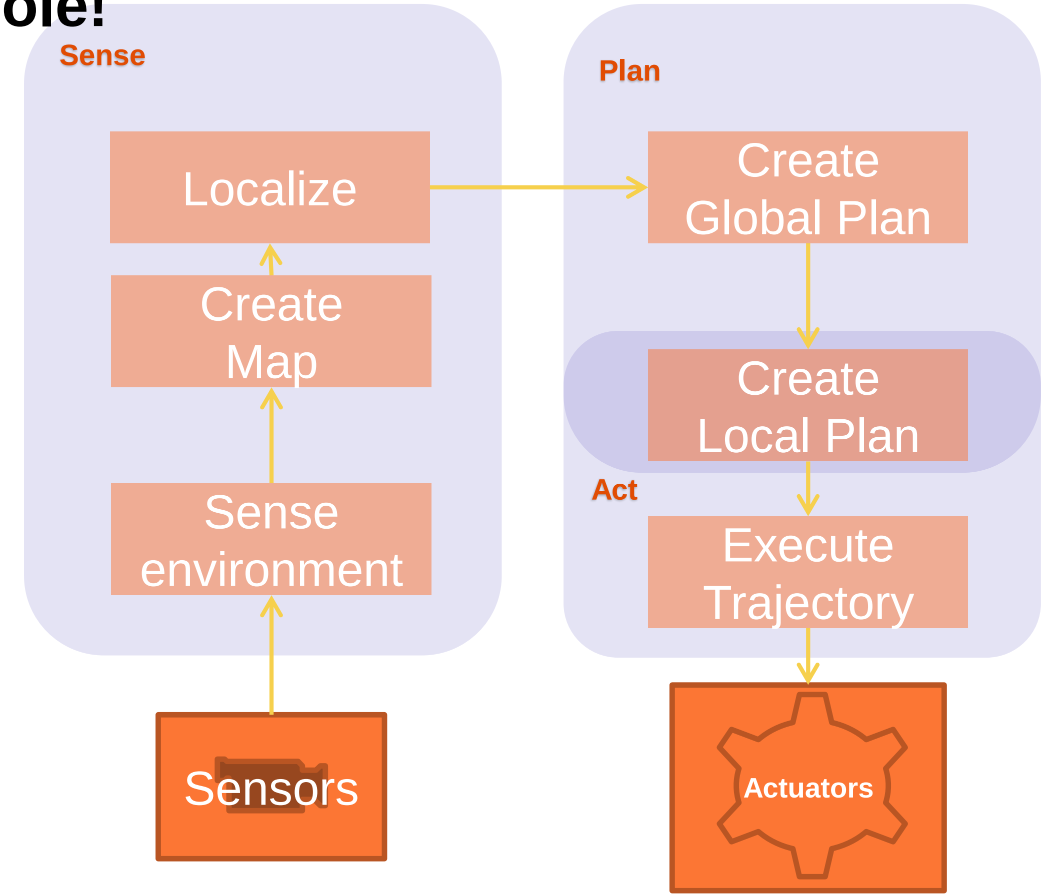
- Robot Software Architecture (Sense→Plan→Act)



AGV System as a whole!

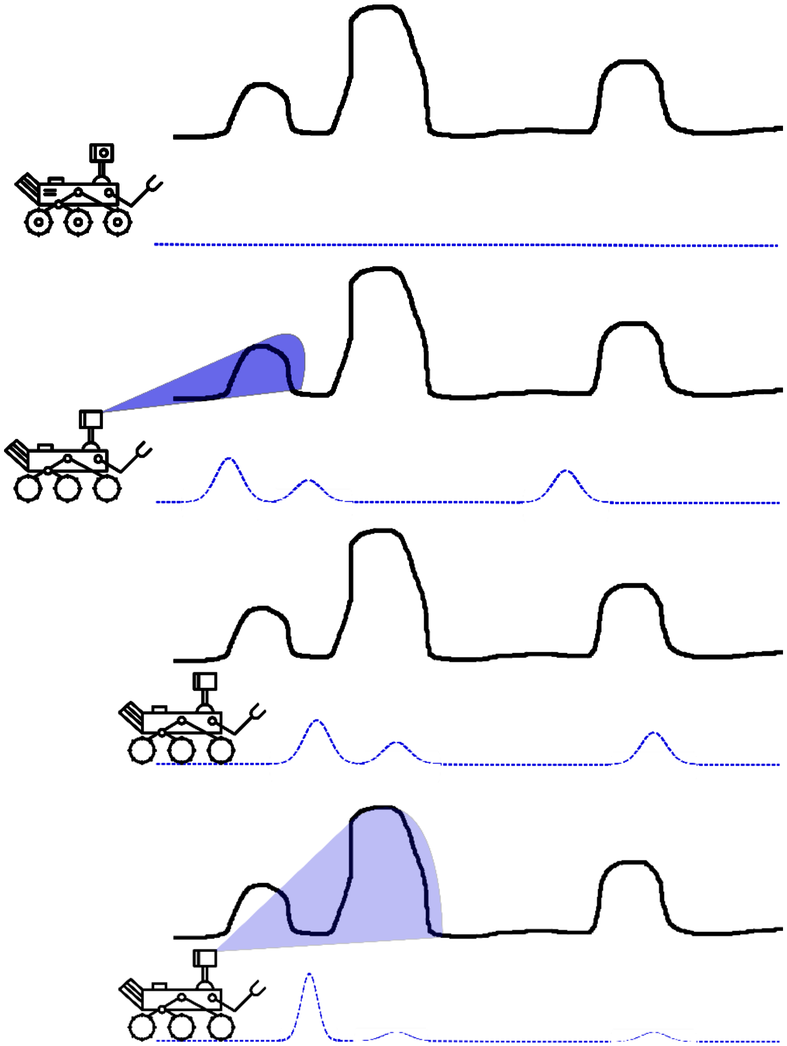
- Robot Software Architecture (Sense→Plan→Act)

1. Use Sensors
   a) Mapping
   b) Localization
2. Plan Global Route
3. Execute Path
   a) Plan Local Trajectory
   b) Execute Velocity Commands



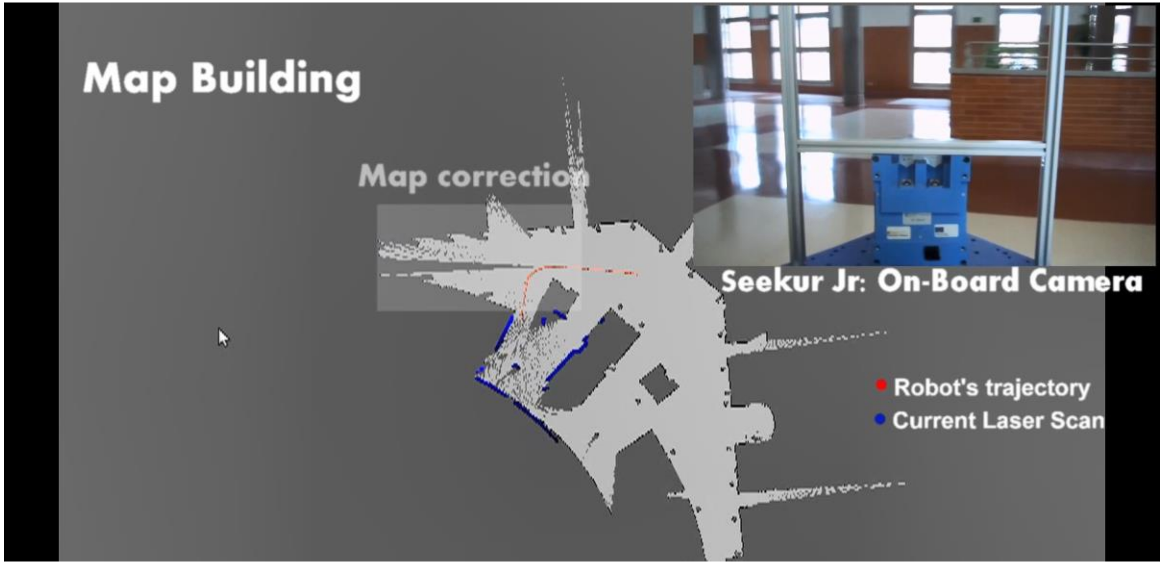
AGV – Localization

- Assume an AGV wandering in 2D
- Initially it doesn't know where it is
- By sensing, it gets an estimation
- When moving,
  it just propagates this estimation
- By sensing again,
  it gets a better estimation
- The same applies in **3D!**



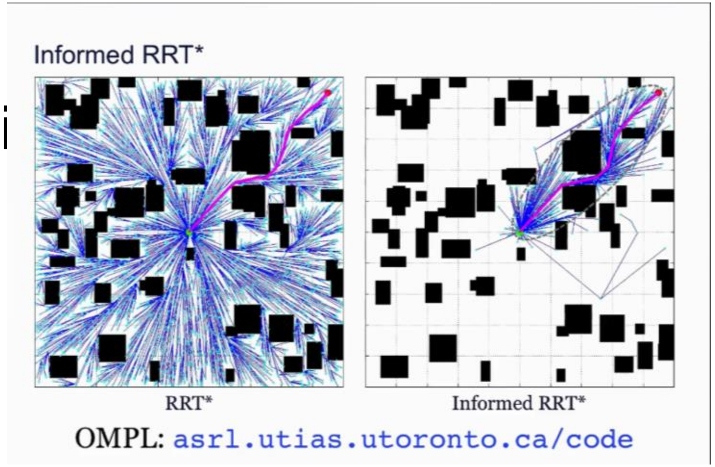
AGV Mapping

- Initial Map Generation
- Map Accumulation
- Map Refinement

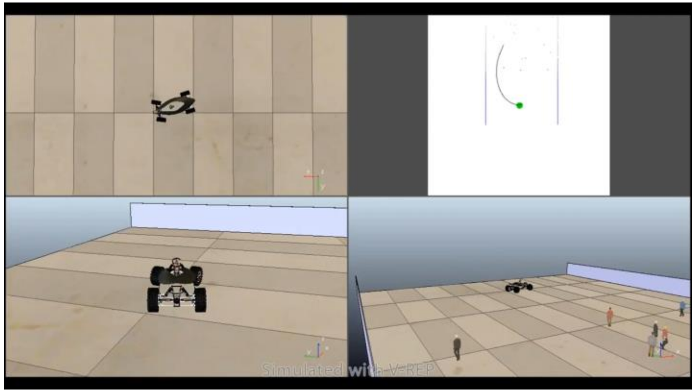


- When Combining the Localization and Mapping we get: “Simultaneous Localization and Mapping” - SLAM

AGV Navigation - Path Planni

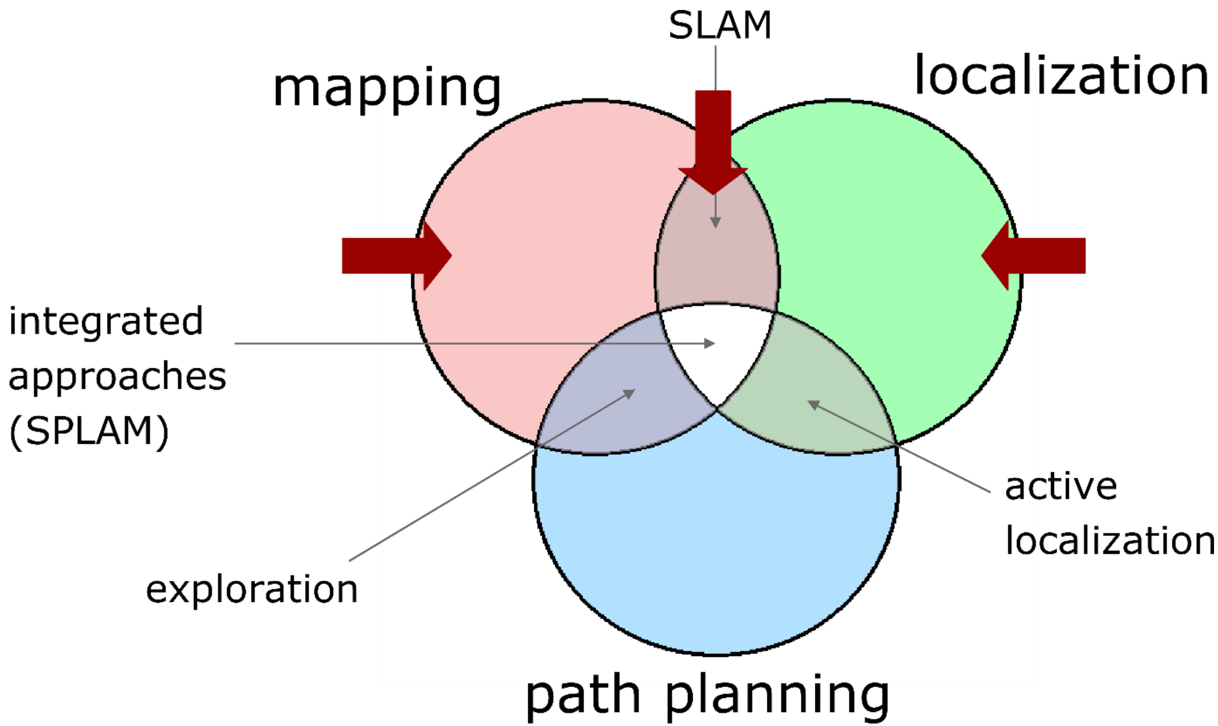


- Global Path Planning
  - Operates on Global Map
  - Find the best Global trajectory



- Local Path Planning
  - Operates on Local Map
  - Generate small navigation spline to overcome obstacles
  - Send arc velocity commands to robot

Overview of Nanigation



Overview of Navigation

To navigate a robot we need:

- A map
- A localization module
- A path planning module

These components are sufficient if:

- The map fully reflects the environment
- The environment is static
- There are no errors in the estimate

However:

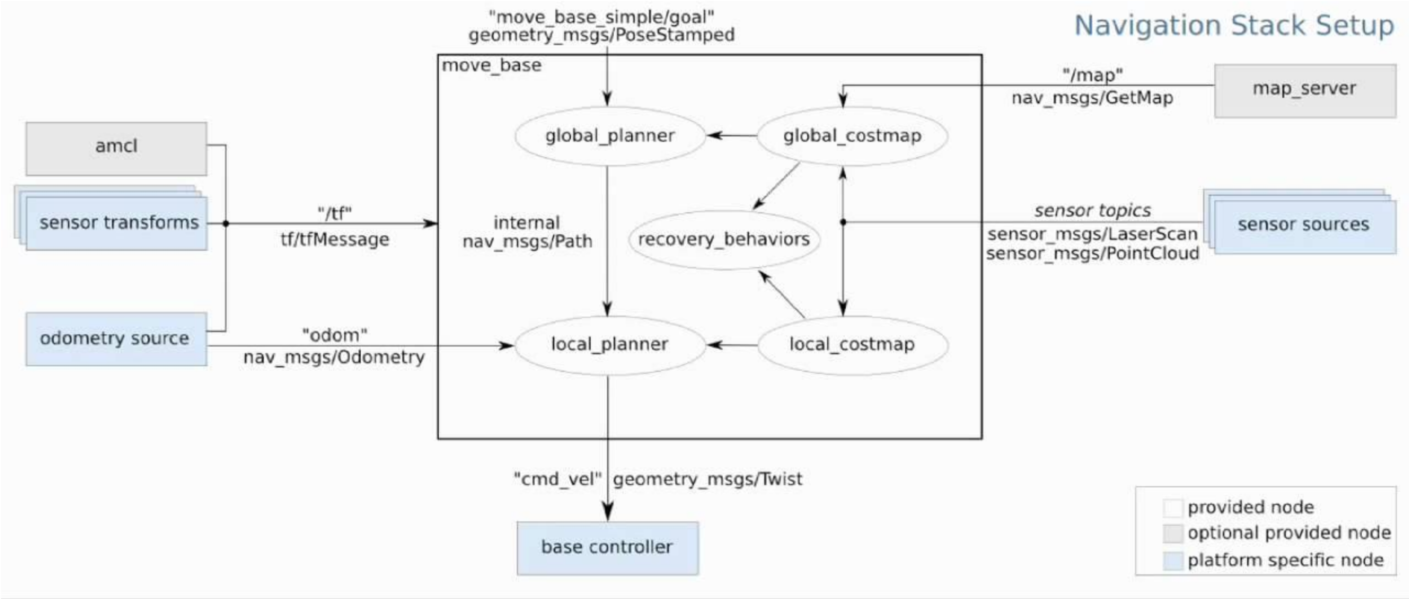
- The environment changes (e.g. opening/closing doors)
- It is dynamic (things might appear/disappear from the perception range of the robot)
- The estimate is “noisy”

Thus we need to complement our ideal design with other components that address these issues, namely:

- Obstacle-Detection/Avoidance
- Local Map Refinement, based on the most recent sensor reading.

ROS Navigation

- Map provided by a “Map Server”
- Each module is a node
- Planner has a layered architecture (local and global planner)
- Obstacle sensing refined on-line by appropriate modules (local and global



Localizing a robot on ROS

- ROS implements the Adaptive Monte Carlo Localization algorithm
  - AMCL uses a particle filter to track the position of the robot
  - Each pose is represented by a particle.
  - Particles are
    - Moved according to (relative) movement measured by the odometry
    - Suppressed/replicated based on how well the laser scan fits the map, given the position of the particle.
  - The localization is integrated in ROS by emitting a transform from a map-frame to the odom frame that “corrects” the odometry.

Localizing a robot on ROS

- AMCL relies on a laser

  Unless you want to spend 5K euro,
  you will not get a laser, so your robot will not localize with this procedure

- However…

  You can get a kinect/xtion sensor, that provides data useful to simulate a laser scanner (how?)

  These data can then be plugged in AMCL et voila’ you get your system running.

Mobile Robots in ROS

- Let’s do some hands on…

Find it on the DTU Learn Page

Mapping in ROS

- There are gazillions of SLAM algorithms around.
- ROS uses GMapping, which implements a particle filter to track the robot trajectories.

Find it on the DTU Learn Page

SumUP

- We learned about AGVs (mobile Robots),
- We about Localization and Mapping
- We learned about Path Planning
- We implemented a mobile robot in ROS

Software for Autonomous Systems
SFfAS-31391:

Autonomous Guided Vehicles

Lecturer, Course Coordinator: Evangelos Boukas—PhD