ROS NAVIGATION STACK

Lecturer
Dr. Emilio José Sánchez Tapia
Carlos Suárez Zapico

tecnun

MOBILE ROBOTS

ROS Navigation Stack

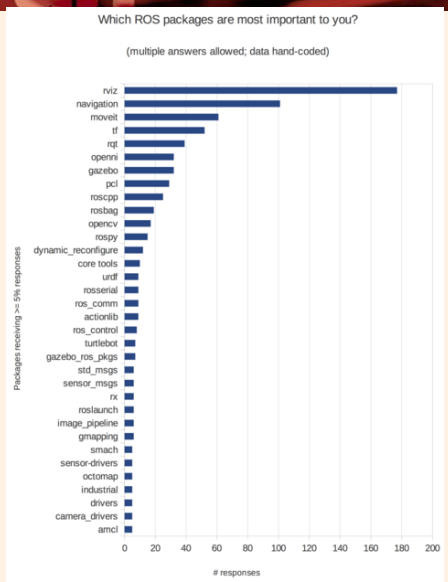
- **Main objective**
  1. Move a robot from A to B
  2. while avoiding obstacles
  3. and not getting lost

- **Content**
  - A **set of ROS nodes and algorithms** for autonomously moving the robot from one location to another, avoiding all obstacles
  - Comes with an **implementation of algorithms** related to **performing autonomous navigation**

Emilio Sánchez-

3

ROS most used packages



ROS Introduction

Emilio Sánchez-

4

Navigation System Overview

Navigation

Mapping	• gmapping • Other options: Hector SLAM and Cartographer
Localization	• Adaptive Monte Carlo Localization (AMCL)
Path Planning	• move_base package • Local planner and costmap • Global planner and costmap
Control	

Emilio Sánchez-

5

Introduction: Mapping

- What do you need to make a robot **navigate**?
  - A map
  - The map is a representation of the environment created from sensor data e.g. 2D lasers
  - By moving the robot around the environment, we can build a 2D map
- DEMO: Teleoperation and Mapping

Emilio Sánchez-

6

Introduction: Localization

- Why and what do you need to **localize** your robot in the map?
  - The robot needs to know where it is to be able to do something useful e.g. bring you coffee or beer
  - For navigation you need to know the position and orientation i.e. pose. This is known as **Localization**
- DEMO: Localization
  - Green arrows: Location guesses of the robot made by the localization algorithm in order to figure out where the robot is in the map
  - The arrows will concentrate on the most likely location when you move the robot

Emilio Sánchez-

7

Introduction: Path Planning

- What do you need to **navigate** your robot in the map?
  - Something to tell the robot **where** to go
  - And **how** to get there
  - This is called **Path Planning**
    - Input: current location of the robot, and location where it needs to go
    - Output: best and fastest path
- DEMO: Path Planning

Emilio Sánchez-

8

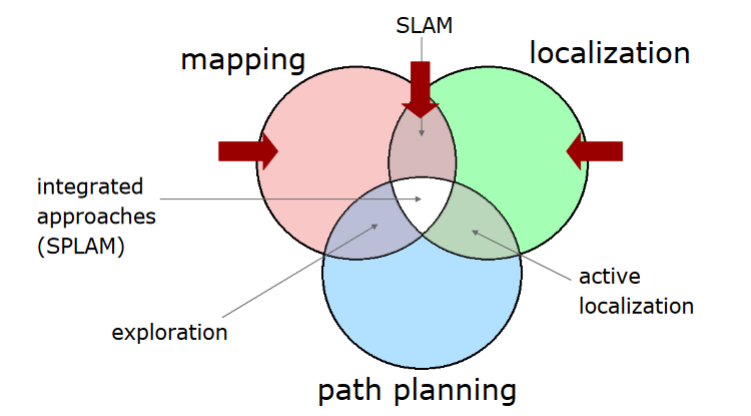
Introduction: Obstacle Avoidance

- What if something not in the map moves into the robot's path?
  - The obstacle avoidance system divides the map into smaller pieces, which, based on sensor data, updates in real-time
  - The path plan is updated accordingly
- DEMO: Obstacle Avoidance

Emilio Sánchez-

9

ROS NAVIGATION STACK



Sensor&Actuators

Emilio Sánchez-

10

ROS NAVIGATION STACK

1- Exploration

2- SLAM (Pose estimation and Mapping)

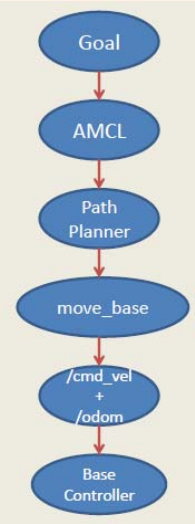
3- Localization

4- Motion Planning

ROS Navigation Stack Packages

Package/Component	Description
map_server	offers map data as a ROS Service
gmapping	provides laser-based SLAM
amcl	a probabilistic localization system
global_planner	implementation of a fast global planner for navigation
local_planner	implementations of the Trajectory Rollout and Dynamic Window approaches to local robot navigation
move_base	links together the global and local planner to accomplish the navigation task

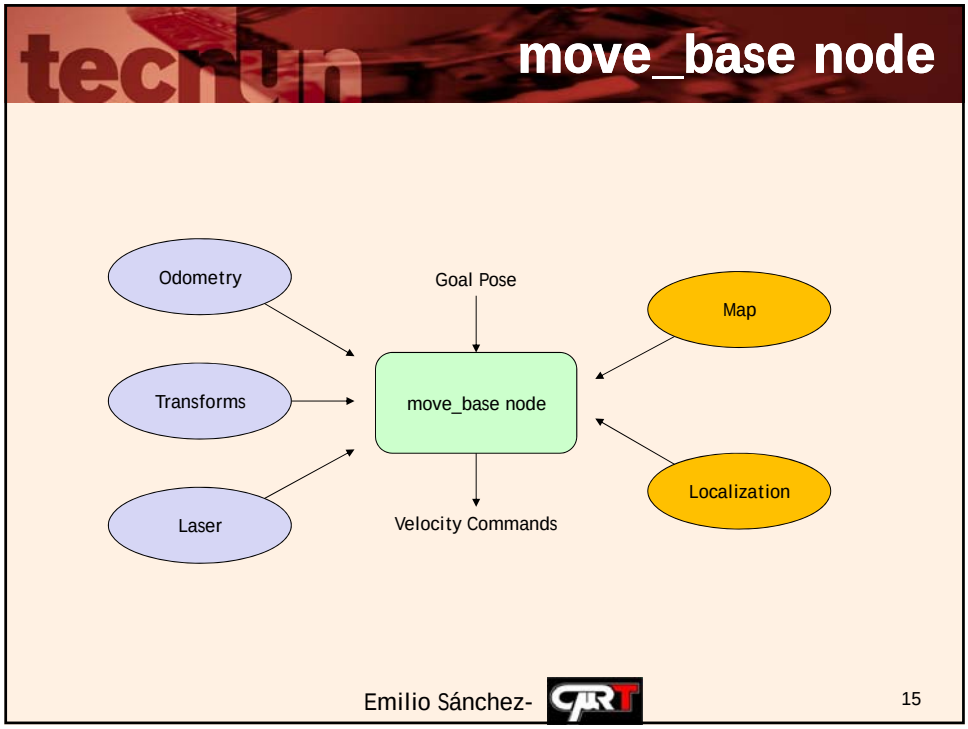
Navigation steps after mapping



Hardware Requirement

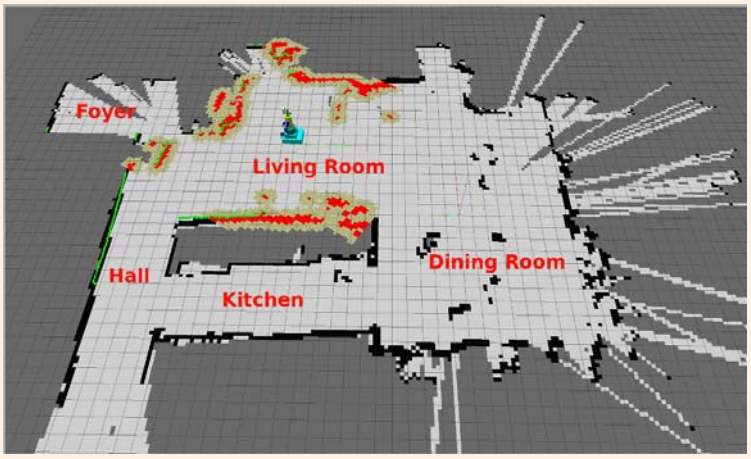
Three main hardware requirements

- The navigation stack can only handle a **differential drive** and **holonomic wheeled** robots. It can also do certain things with biped robots, such as localization, as long as the robot does not move sideways
- A **planar laser** must be mounted on the mobile base of the robot to create the map and localization. Alternatively, you can generate something equivalent to laser scans from other sensors (Kinect for example)
- Its performance will be best on robots that are nearly **square or circular shaped footprint.**


move_base node
Odometry
Transforms
Laser
Goal Pose
move_base node
Velocity Commands
Map
Localization
Emilio Sánchez-
15


GMAPPING
Lecturer
Dr. Emilio José Sánchez Tapia
Carlos Suárez Zapico

SLAM(Simultaneous Localization And Mapping)



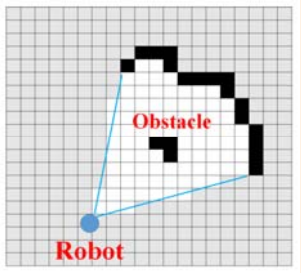
Grid maps

- Discretize the world into cells
- Grid structure is rigid
- The area corresponds to a cell is either completely free or occupied

Gmapping

- **Input**
- Odometry (/odom)
- Laserscan (/scan)
- Movement Commands (/cmd_vel)

- **Output**
- Occupancy grid map



Sensor&Actuators Emilio Sánchez- 19

Exploration

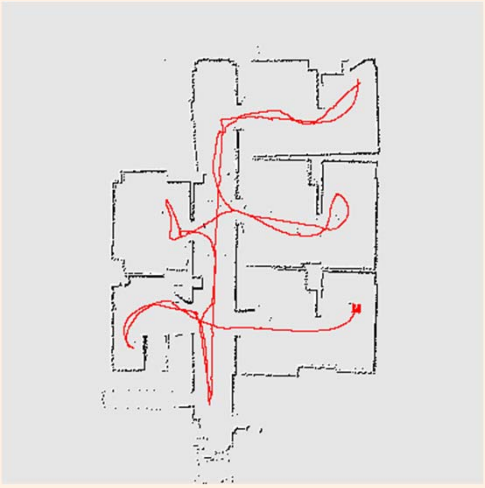
Follow a strategy to explore the surrounding environment and create a detailed map.

- Teleoperation

- Random Navigation

ROS Introduction Emilio Sánchez- 20

SLAM



Conventional naming frames

Navigation Stack Frames

For the navigation stack to work properly, the robot needs to publish the following tf relationships:

/map → /odom → /base_footprint → /base_link → /base_laser_link

- **map** – the coordinate frame fixed to the map
- **odom** – the self consistent coordinate frame using the odometry measurements only (this will not change on localization updates)
  - The map → odom transform is published by amcl or gmapping
- **base_footprint** – the base of the robot at zero height above the ground
- **base_link** – the base link of the robot, placed at the rotational center of the robot
- **base_laser_link** – the laser sensor

Hardness of Mapping Problem

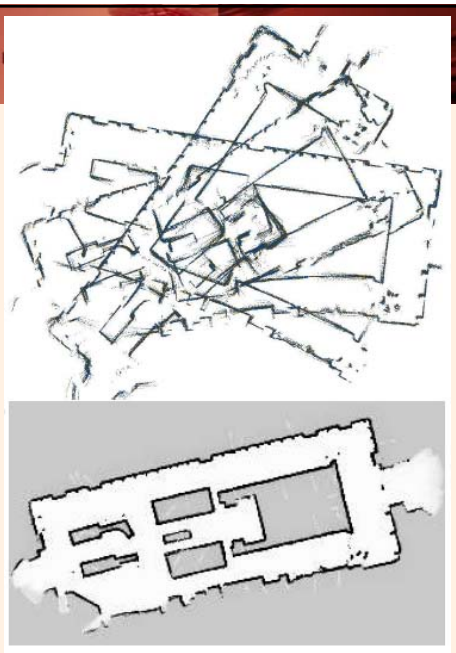
Size- the larger the environment compare with perception range

Noise in perception and actuation –if laser and actuators noisy-free it would be much easy

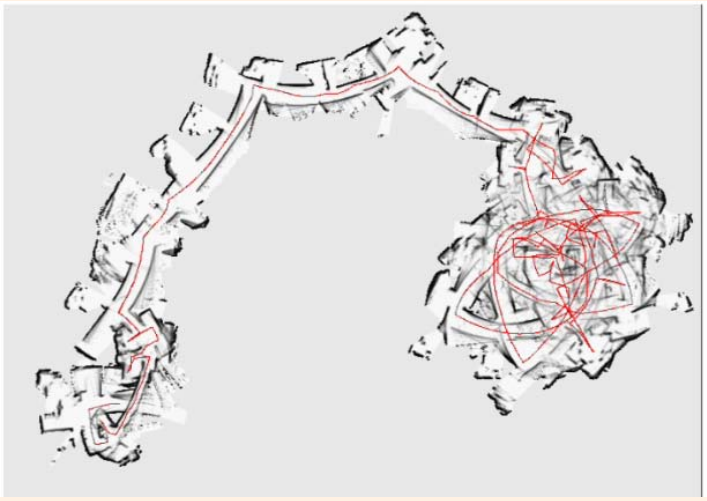
Perceptual Ambiguity- the more frequent different places look alike, more difficult to establish correspondence between different locations.

Cycles- when closing a cycle the accumulated odometric error can be huge

Mapping



Mapping with raw odometry



Sensor&Actuators Emilio Sánchez- 25

Mapping with Scan Matching

Sensor&Actuators Emilio Sánchez- 26

rosrun map_server map_server mymap.yaml

Gmapping simulation

$ export TURTLEBOT3_MODEL=Burger

Execute Gazebo simulator

$ roslaunch turtlebot3_gazebo turtlebot3_world.launch

Execute rviz to visualize map

$ roslaunch turtlebot3_gazebo turtlebot3_gazebo_rviz.launch

Teleoperation on Gazebo

$ roslaunch turtlebot3_teleop turtlebot3_teleop_key.launch

Launch SLAM

$ roslaunch turtlebot3_slam turtlebot3_slam.launch slam_methods:=gmapping

Save the map

$ rosrun map_server map_saver -f ~/map

Relaunch map

rosrun map_server map_server mymap.yaml

Sensor&Actuators Emilio Sánchez- 27