Robotics &
Internet-of-Things

ROS

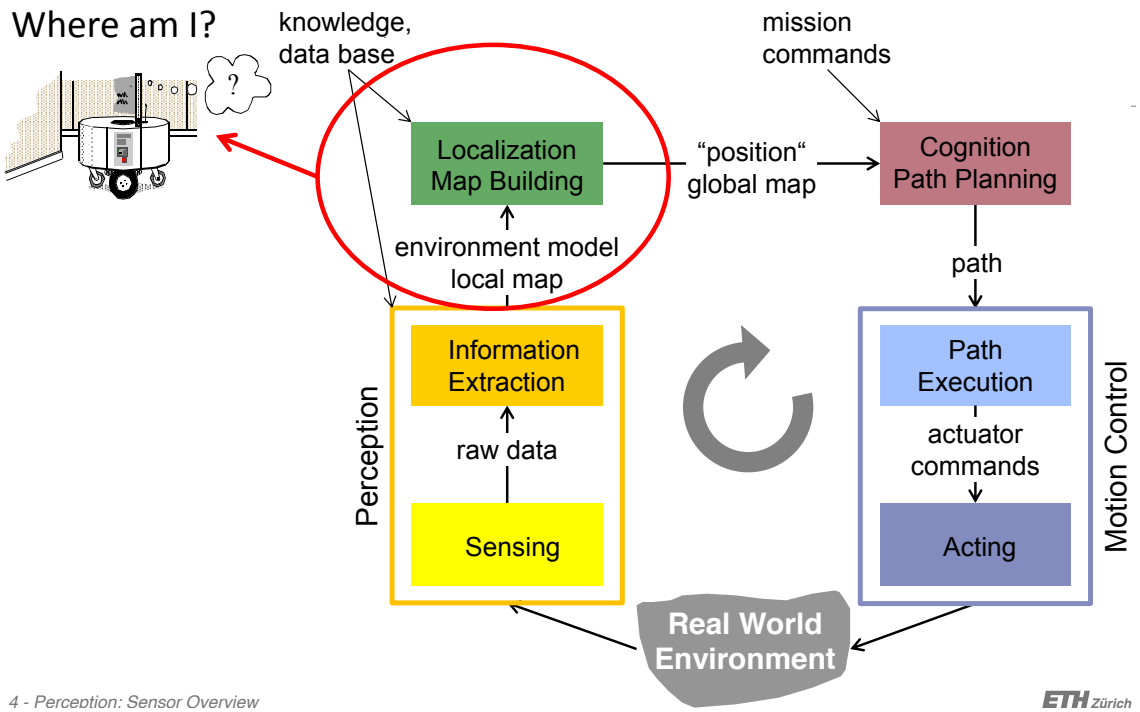
Udemy Course
Robot Operating System
Basics, Motion, and OpenCV

Prof. Anis Koubaa

Map-Based Navigation

https://www.udemy.com/user/anis-koubaa/

MAP-BASED NAVIGATION



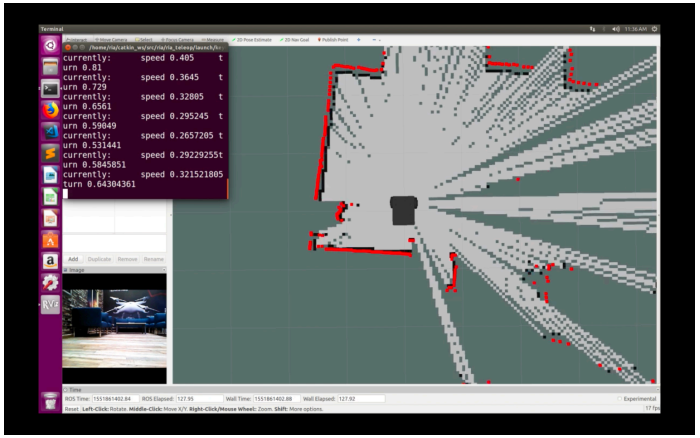
Reference:
http://www.asl.ethz.ch/education/master/mobile_robotics

MAP-BASED NAVIGATION

- Robot navigation
  - Where am I?
  - Where am I going?
  - How do I get there?
- **Localization:** it helps the robot to know where he is
- **Mapping:** the robot needs to have a map of its environment to be able to recognize where he has been moving around so far
- **Motion planning or path planning:** to plan a path, the target position must be well- defined to the robot, which require an appropriate addressing scheme that the robot can understand

BUILDING A MAP: SIMULTANEOUS LOCALIZATION AND MAPPING (SLAM)

- It is the process of building a map using range sensors (e.g. laser sensors, 3D sensors, ultrasonic sensors) while the robot is moving around and exploring an unknown area
- Sensor Fusion: This process uses filtering techniques like Kalman filters or praticle filters



NAVIGATION IN ROS

- Three main packages of the navigation stack
  - **move_base:** makes the robot navigate in a map and move to move to a goal pose with respect to a given reference frame
  - **mapping:** creates maps using laser scan data
  - **amcl:** responsible for localization using an existing map.

Robotics &
Internet-of-Things

ROS

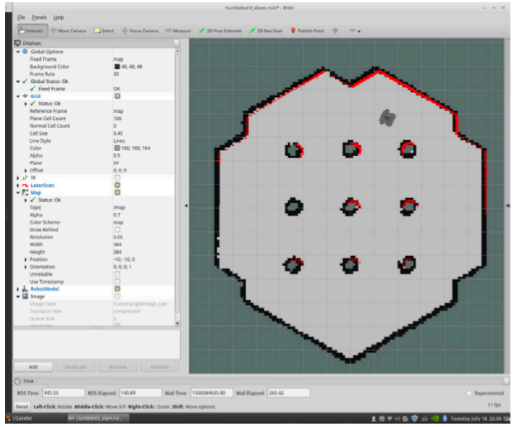
Udemy Course
Robot Operating System
Basics, Motion, and OpenCV

Prof. Anis Koubaa

SLAM Demo

https://www.udemy.com/user/anis-koubaa/

TURTLEBOT3 SLAM DEMO



- Start Turtlebot3 Waffle
  - export TURTLEBOT3_MODEL=waffle
  - roslaunch turtlebot3_gazebo turtlebot3_house.launch
- open **gmapping** SLAM application
  - export TURTLEBOT3_MODEL=waffle_pi
  - roslaunch turtlebot3_slam turtlebot3_slam.launch slam_methods:=gmapping
- Teleop the robot
  - export TURTLEBOT3_MODEL=waffle_pi
  - roslaunch turtlebot3_teleop turtlebot3_teleop_key.launch
- Save the Map
  - rosrun map_server map_saver -f ~/map

Occupancy Grid Map

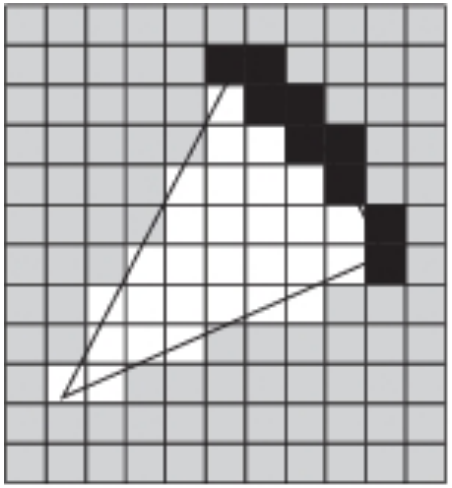
- A map is a grid (matrix) of cell
- A cell can be **empty** or **occupied**
- Depending on resolution, cell size can be 5 to 50 cm
- Each cell hold a probability of occupancy 0% to 100%
- Areas that are unknown are marked as -1

EXAMPLE OF MAP

SLAM Approaches in ROS

- There are several SLAM approaches in ROS
  - **gmapping** which contains a ROS wrapper for OpenSlam's Gmapping
  - **cartographer**, which is a system developed by Google that provides real-time simultaneous localization and mapping (SLAM) in 2D and 3D across multiple platforms and sensor configurations.
  - **hector_slam** which is an other SLAM approach that can be used without odomtery

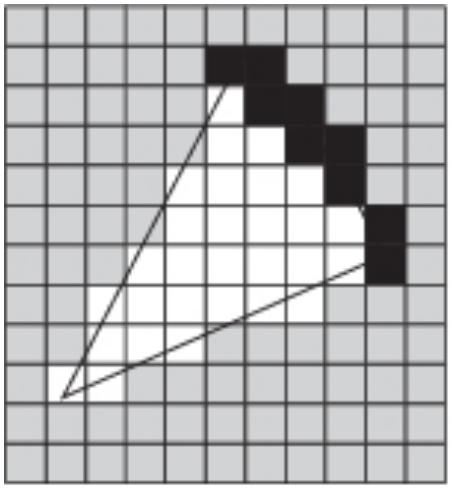
Marking and Clearing



- The map is built using a SLAM algorithm.
- Cell has three possible states
  - Unknown
  - Empty
  - Occupied
- It is based on the process of **marking** and **clearing**
  - **Marking:** a cell is marked as obstacle
  - **Clearing:** a cell is marked as empty
- ray-tracing: used to find empty cell.

OCCUPANCY GRID IN ROS

- The map is built using a SLAM algorithm.
- Cell has three possible states
  - Unknown
  - Empty
  - Occupied



Robotics &
Internet-of-Things

ROS

Udemy Course
Robot Operating System
(Part II)
Navigation and SLAM

Prof. Anis Koubaa

Understand Maps in ROS

https://www.udemy.com/user/anis-koubaa/

Robotics & Internet-of-Things

ROS

Udemy Course
Robot Operating System
(Part II)
Navigation and SLAM

Prof. Anis Koubaa

Understand ROS Nodes and Launch File used for SLAM

https://www.udemy.com/user/anis-koubaa/

Robotics &
Internet-of-Things

ROS

Udemy Course
Robot Operating System
(Part II)
Navigation and SLAM

Prof. Anis Koubaa

Map-Navigation Demo

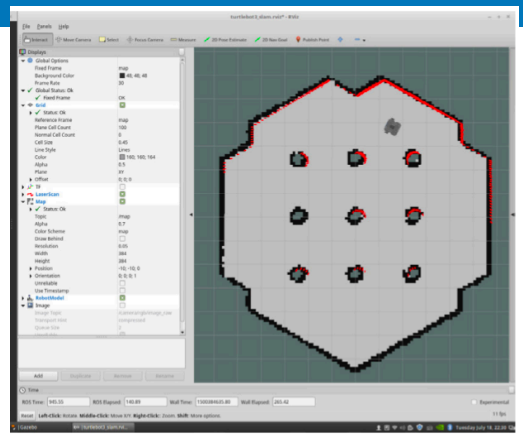
https://www.udemy.com/user/anis-koubaa/

Navigation Demo Summary

- We first need to manually set the initial location of the robot in the map using **2D Pose Estimate**
- We send goal poses (location+orientation) using 2D Nav Goal, so that the robot moves to it.
- The navigation stack has two motion planners:
  - **Global Path Planner:** plans a **static obstacle-free** path from the location of the robot to the goal location
  - **Local Path Planner:** execute the planned trajectory and avoids **dynamic** obstacle.

/http://emanual.robotis.com/docs/en/platform/turtlebot3/simulation

TURTLEBOT3 NAVIGATION DEMO



- Start Turtlebot3 Waffle
  - export TURTLEBOT3_MODEL=waffle
  - roslaunch turtlebot3_gazebo turtlebot3_house.launch
- open **navigation** application
  - export TURTLEBOT3_MODEL=waffle_pi
  - roslaunch turtlebot3_navigation turtlebot3_navigation.launch map_file:=/home/ros/tb3_house_map.yaml

/http://emanual.robotis.com/docs/en/platform/turtlebot3/simulation

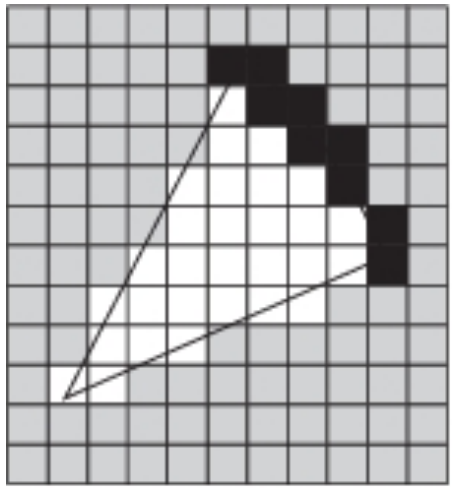
Navigation Demo Summary

- We first need to manually set the initial location of the robot in the map using **2D Pose Estimate**
- We send goal poses (location+orientation) using 2D Nav Goal, so that the robot moves to it.
- The navigation stack has two motion planners:
  - **Global Path Planner:** plans a **static obstacle-free** path from the location of the robot to the goal location
  - **Local Path Planner:** execute the planned trajectory and avoids **dynamic** obstacle.

/http://emanual.robotis.com/docs/en/platform/turtlebot3/simulation

Marking and Clearing

- It is based on the process of **marking** and **clearing**
  - **Marking:** a cell is marked as obstacle
  - **Clearing:** a cell is marked as empty
- ray-tracing: used to find empty cell.



Robotics &
Internet-of-Things

ROS

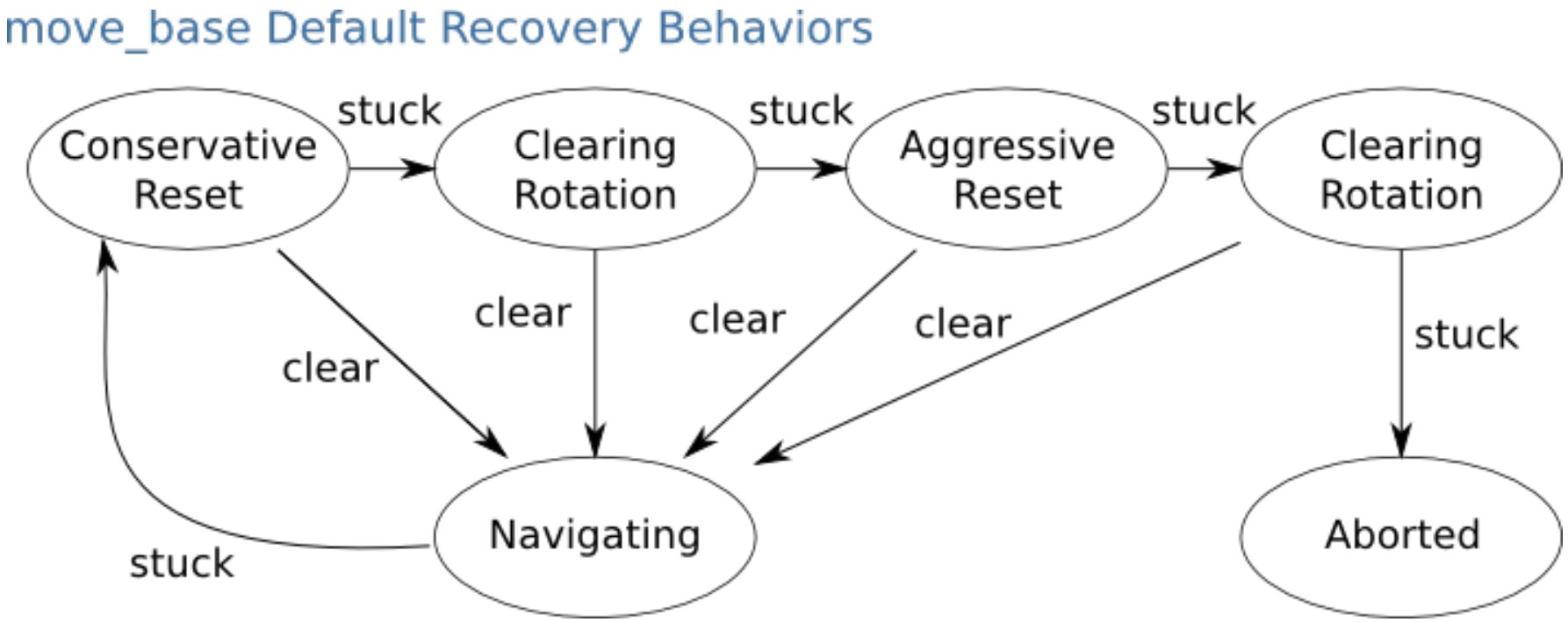
Udemy Course
Robot Operating System
(Part II)
Navigation and SLAM

Prof. Anis Koubaa

Recovery Behaviors
move_base behaviors

https://www.udemy.com/user/anis-koubaa/

MOVE_BASE BEHAVIORS



Robotics & Internet-of-Things

ROS

Udemy Course
Robot Operating System (Part II)
Navigation and SLAM

Prof. Anis Koubaa

Write a ROS Node for Map-Based Navigation

https://www.udemy.com/user/anis-koubaa/

Udemy Course
Robot Operating System
(Part II)
Navigation and SLAM

Prof. Anis Koubaa

The Recovery Behavior Demo

https://www.udemy.com/user/anis-koubaa/

Robotics & Internet-of-Things

ROS

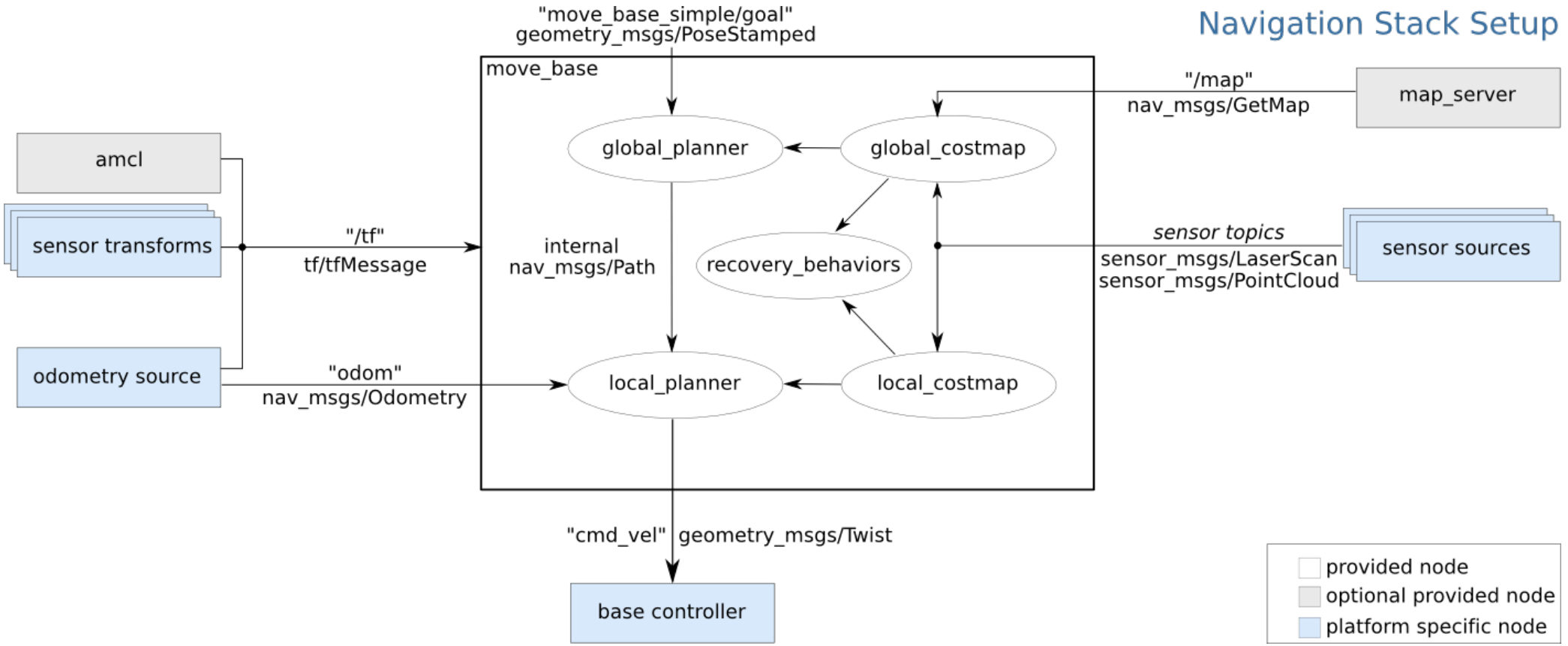
Udemy Course
Robot Operating System
(Part II)
Navigation and SLAM

Prof. Anis Koubaa

Robot Setup to Support the ROS Navigation Stack

https://www.udemy.com/user/anis-koubaa/

ROBOT SETUP

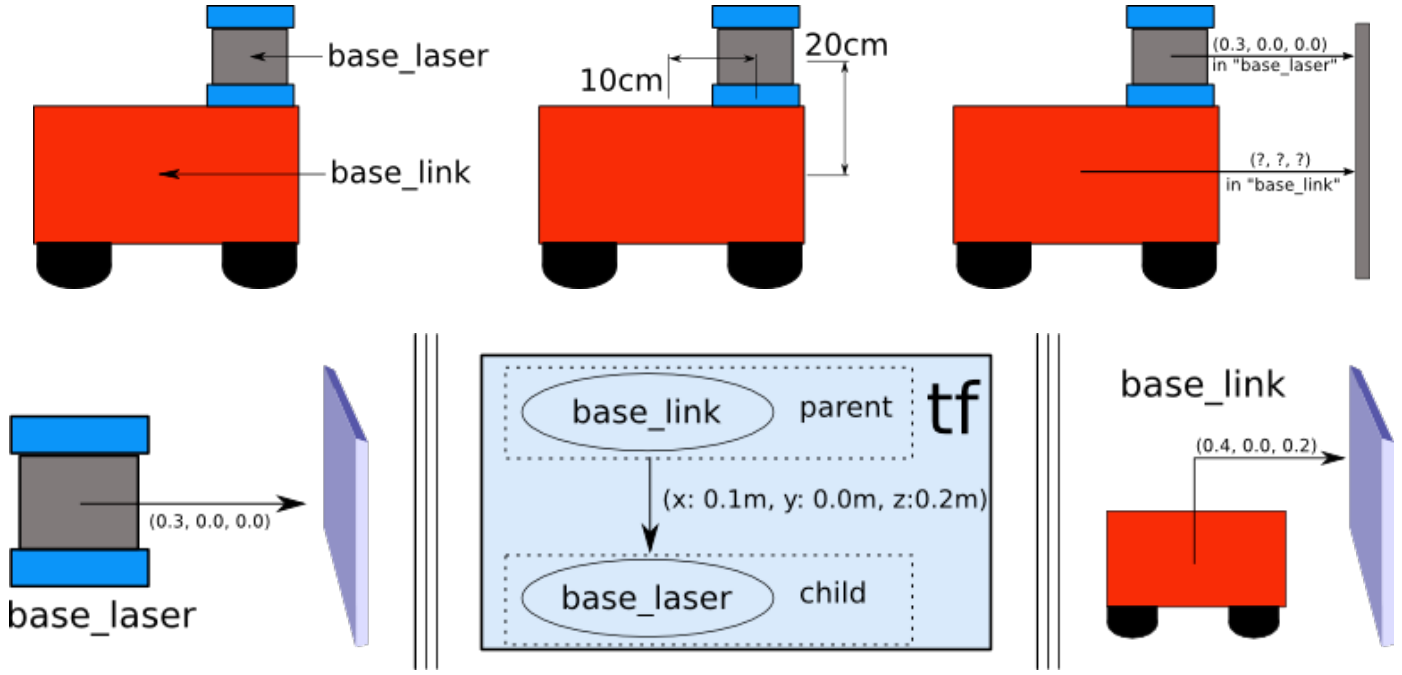

Navigation Stack Setup
"move_base_simple/goal"
geometry_msgs/PoseStamped
move_base
"/map"
nav_msgs/GetMap
map_server
global_planner
global_costmap
amcl
sensor transforms
"/tf"
tf/tfMessage
internal
nav_msgs/Path
recovery_behaviors
sensor topics
sensor_msgs/LaserScan
sensor_msgs/PointCloud
sensor sources
odometry source
"odom"
nav_msgs/Odometry
local_planner
local_costmap
"cmd_vel"
geometry_msgs/Twist
base controller
provided node
optional provided node
platform specific node

Robot Setup

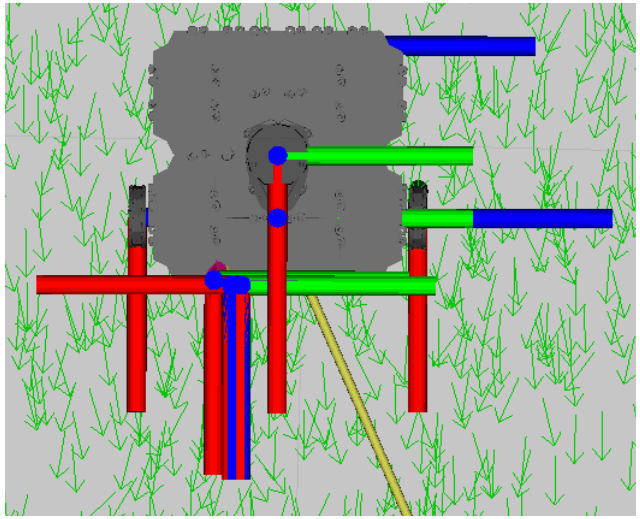
- ROS
  - requires ROS to be installed

Robot Setup

- Transform Configuration
  - a robot must publish tf transform to describe the relation between coordinate frames



URDF: Robot Description Language.



```xml
<?xml version="1.0" ?>
<robot name="turtlebot3_burger" xmlns:xacro="http://ros.org/wiki/xacro">
  <xacro:include filename="$(find turtlebot3_description)/urdf/common_properties.xacro"/>
  <xacro:include filename="$(find turtlebot3_description)/urdf/turtlebot3_burger.gazebo.xacro"/>

  <link name="base_footprint"/>

  <joint name="base_joint" type="fixed">
    <parent link="base_footprint"/>
    <child link="base_link"/>
    <origin xyz="0.0 0.0 0.010" rpy="0 0 0"/>
  </joint>

  <link name="base_link">
    <visual>
      <origin xyz="-0.032 0 0.0" rpy="0 0 0"/>
      <geometry>
        <mesh filename="package://turtlebot3_description/meshes/bases/burger_base.stl" scale="0.001 0.001 0.001"/>
      </geometry>
      <material name="light_black"/>
    </visual>

    <collision>
      <origin xyz="-0.032 0 0.070" rpy="0 0 0"/>
      <geometry>
        <box size="0.140 0.140 0.143"/>
      </geometry>
    </collision>

    <inertial>
      <origin xyz="0 0 0" rpy="0 0 0"/>
      <mass value="8.2573504e-01"/>
      <inertia ixx="2.2124416e-03" ixy="-1.2294101e-05" ixz="3.4938785e-05"
               iyy="2.1193702e-03" iyz="-5.0120904e-06"
               izz="2.0064271e-03" />
    </inertial>
  </link>
```

http://wiki.ros.org/urdf/Tutorials

BROADCASTING A TRANSFORM

Toggle line numbers

```
1  #include <ros/ros.h>
2  #include <tf/transform_broadcaster.h>
3
4  int main(int argc, char** argv){
5    ros::init(argc, argv, "robot_tf_publisher");
6    ros::NodeHandle n;
7
8    ros::Rate r(100);
9
10   tf::TransformBroadcaster broadcaster;
11
12   while(n.ok()){
13     broadcaster.sendTransform(
14       tf::StampedTransform(
15         tf::Transform(tf::Quaternion(0, 0, 0, 1), tf::Vector3(0.1, 0.0, 0.2)),
16         ros::Time::now(),"base_link", "base_laser"));
17     r.sleep();
18   }
19 }
```

Robot Setup

- Sensor Information
  - The navigation stack uses information from sensors to avoid obstacles in the world
  - Two types of sensor messages
    - sensor_msgs/LaserScan
    - sensor_msgs/PointCloud
- Supported sensors
  - URG, SICK

ROBOT SETUP

nav_msgs/Odometry Message

- **Odomtery Information**
  - The navigation stack requires that odometry information be published using tf and the nav_msgs/Odometry message
  - http://wiki.ros.org/navigation/Tutorials/RobotSetup/Odom

```
Header header
  uint32 seq
  time stamp
  string frame_id
string child_frame_id
geometry_msgs/PoseWithCovariance pose
  geometry_msgs/Pose pose
    geometry_msgs/Point position
      float64 x
      float64 y
      float64 z
    geometry_msgs/Quaternion orientation
      float64 x
      float64 y
      float64 z
      float64 w
  float64[36] covariance
geometry_msgs/TwistWithCovariance twist
  geometry_msgs/Twist twist
    geometry_msgs/Vector3 linear
      float64 x
      float64 y
      float64 z
    geometry_msgs/Vector3 angular
      float64 x
      float64 y
      float64 z
  float64[36] covariance
```

Robot Setup

- **Base Controller**
  - The navigation stack assumes that it can send velocity commands using a geometry_msgs/Twist message
  - Twist message is assumed to be in base coordinate frame
  - Special node receives cmd_vel command and convert it to motor command

Robot Setup

- **Mapping (map_server)**
  - Navigation stack can work with or without map