Hochschule
Bonn-Rhein-Sieg
University of Applied Sciences

Software Development Project

Final Presentation

Wei-Chan Hsu, Torsten Jandt, Ramesh Kumar, Danning Wang

July 10, 2017

Introduction

Software Development Project

- Object-oriented software development
- Agile software development
- Unified modeling language (UML)
- Refactoring
- Software development in robotics

Basic Navigation Test

- Environment: Workspaces, waypoints and obstacles.
- Task specification: Sequence of poses.

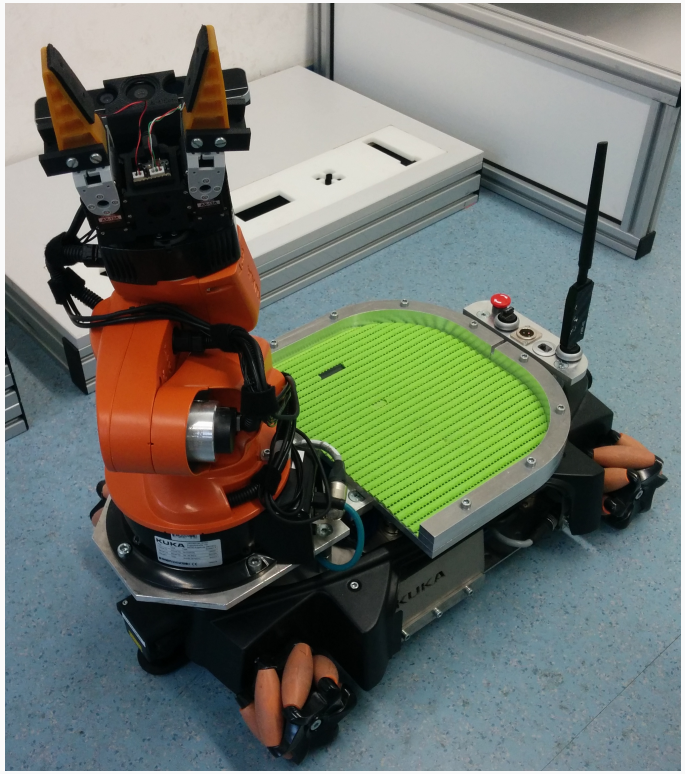
Challenges

- **Perception:** Accessing and processing sensor data.
- **Mapping:** Building map of the environment.
- **Localization:** Pose inside map.
- **Path planning:** Determine sequence of poses between waypoints.
- **Motion control:** Execution of path.

KUKA youBot

The youBot is a mobile manipulator designed for education and research purposes. It comes with fully open interfaces and API.

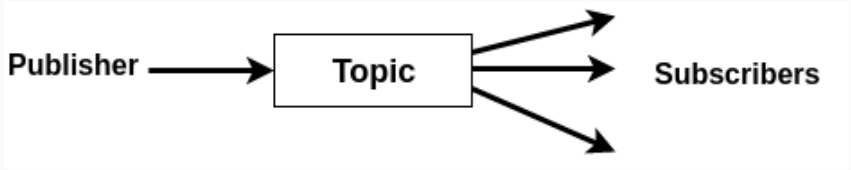
- Omnidirectional, four-wheeled
- 5-DOF manipulator with a two-finger gripper
- On-board PC with CPU, 2GB memory, 32GB SSD drive
- Sensors: vision sensors, rangefinders



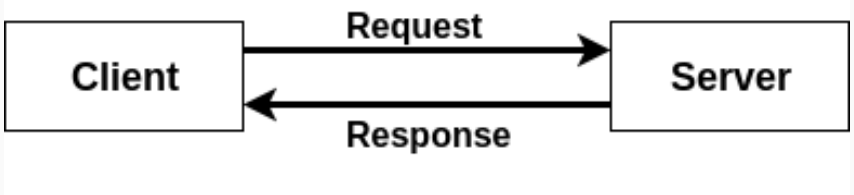
Robot Operating System (ROS)

Set of software and libraries.

- **Node**: A process using ROS.
- **Topic**: Message queue, used for communication between nodes.



- **Service**: Offers synchronous service calls.



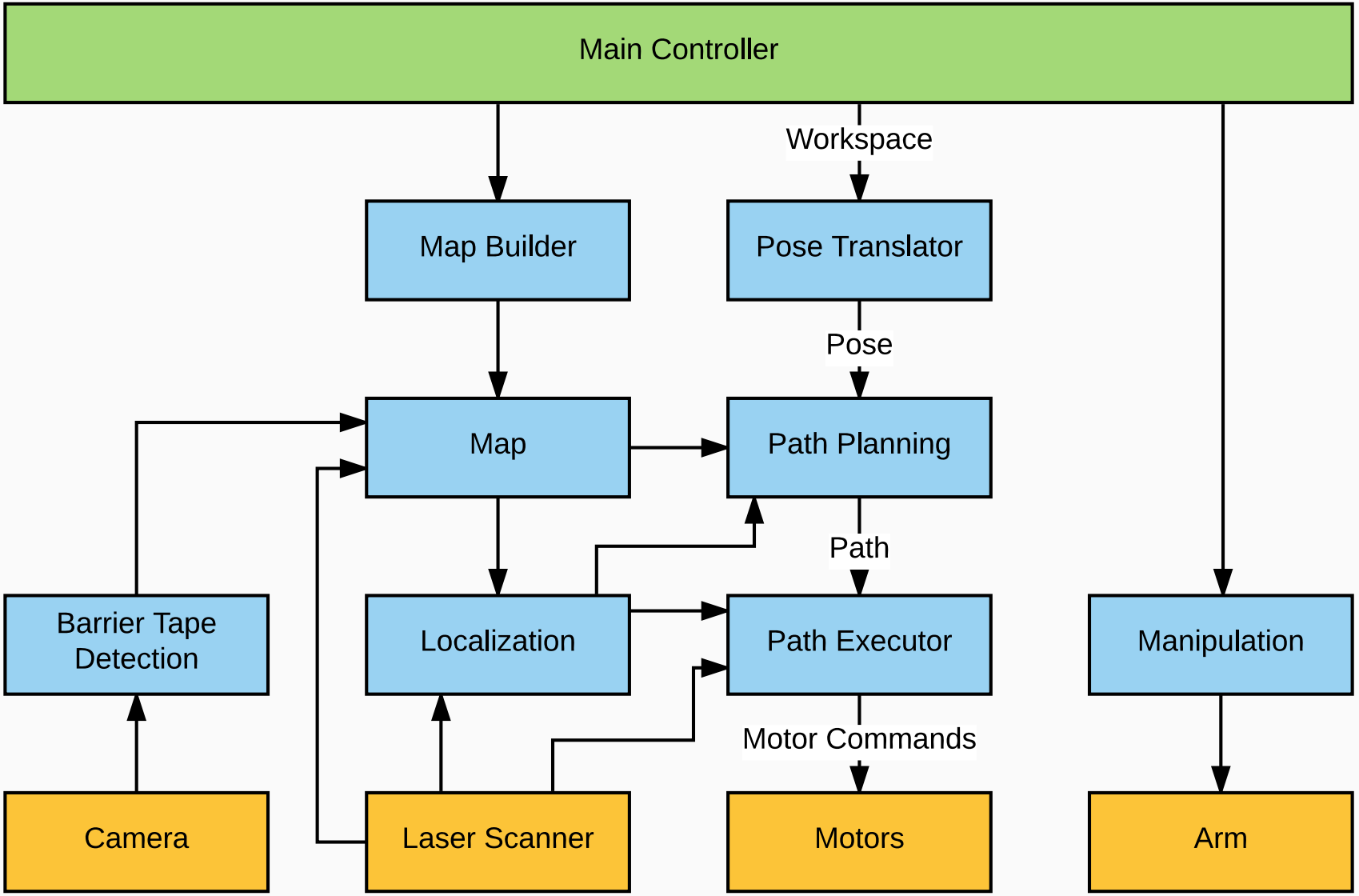
- **Actionlib**
  - Provides client interface to send requests to server
  - Client and server communicate with messages:
    - Goal
    - Feedback
    - Result

Approach

Approach

- Divide problem into smaller parts.
- Each part is defined by
  - It's function and
  - Interface
- Parts are replaceable.

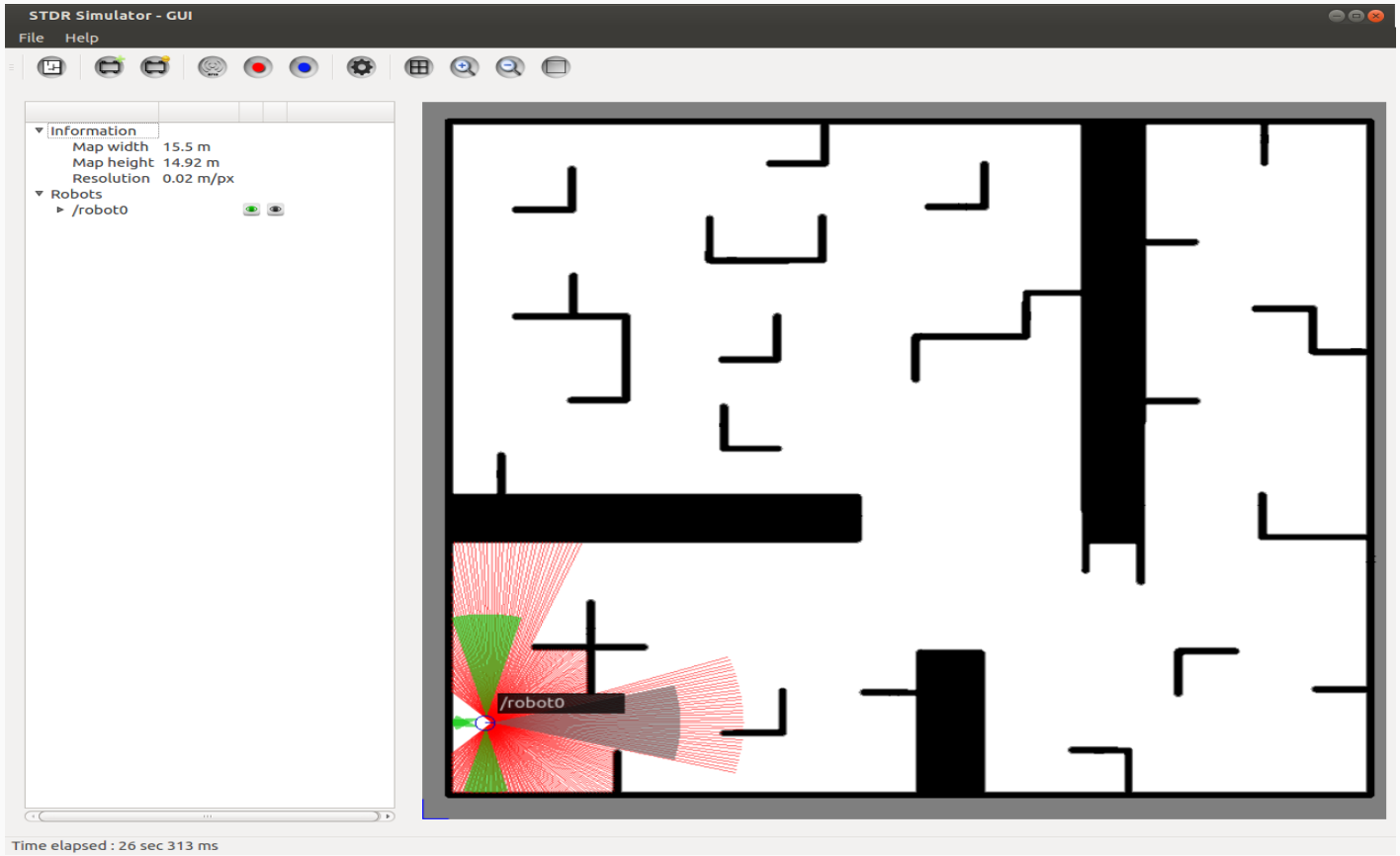
Software Modules



Realization

Simulation

- Simple Two Dimensional Robot (STDR) simulator
- Tasks performed:
  - Map Building
  - Localization



Map building I

- Gmapping is used to build 2D occupancy grid map
- RAO-BLACKWELLIZED Mapping
  - Individual map is associated to every sample
  - Each map is built given the observations and the trajectory
  - ????
- Map Server
  - Provides map saver utility, to save generated map in files(yaml and pgm)
  - Offers map data as a ROS Service

Localization I

- Adaptive Monto Carlo Localization(AMCL) is used to localize the robot
- Uses particle filter to track the pose of robot
  - Distribute samples according to initial pose
  - For each particle, predict next pose from motion model and add random noise
  - Update each particleś weight based on likelihood of getting the sensor readings from that particleś hypothesis
  - Resample new set of particles according to its weight
- What it needs?
  - Laser scans
  - Initial pose
  - Transforms
  - Map

Localization I

- **Problems**
  - Amcl could not find laser data on /scan topic
  - Amcl node crushes after some time
  - Amcl is not working
- **Solutions**
  - Remap /scan_front and /scan_rear to /scan topic
  - Because of incorrect transformations provided by stdr simulator
  - Parameter odom_model_type should be omnicorrected

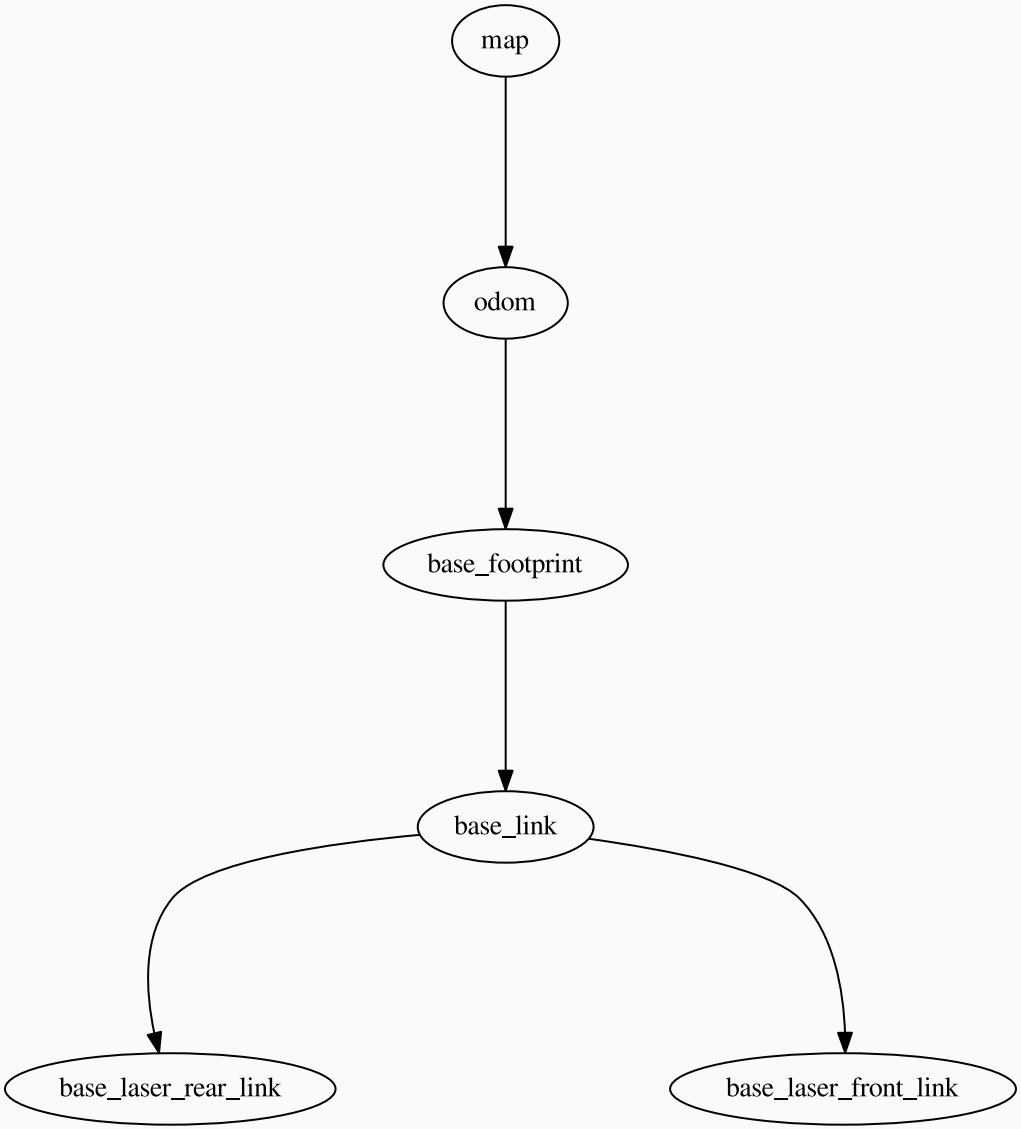
youBot Driver

- **ROS Wrapper**
  - Allows to write ROS programs for controlling youBot
  - Provides an interface between youBot driver and ROS framework
  - Allows to move the base and arm by sending ROS messages
- **List of Drivers**
  - Drive base
  - Laser scanners
  - Arm
  - Joystick
  - Transformations

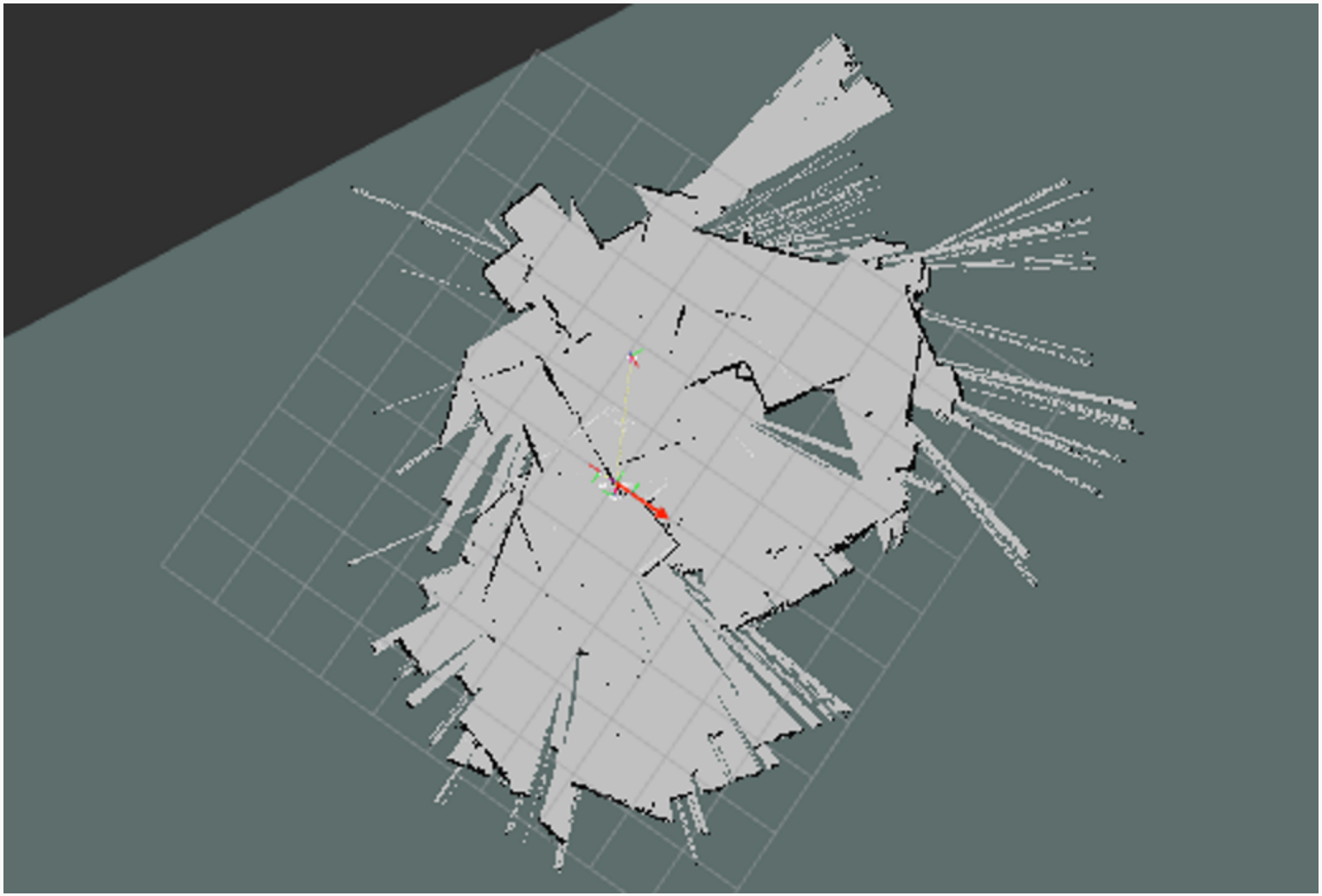
Map building II

- **Problems**
  - Messy Map
- **Solutions**
  - Incorrect transformation between laser frame and base link

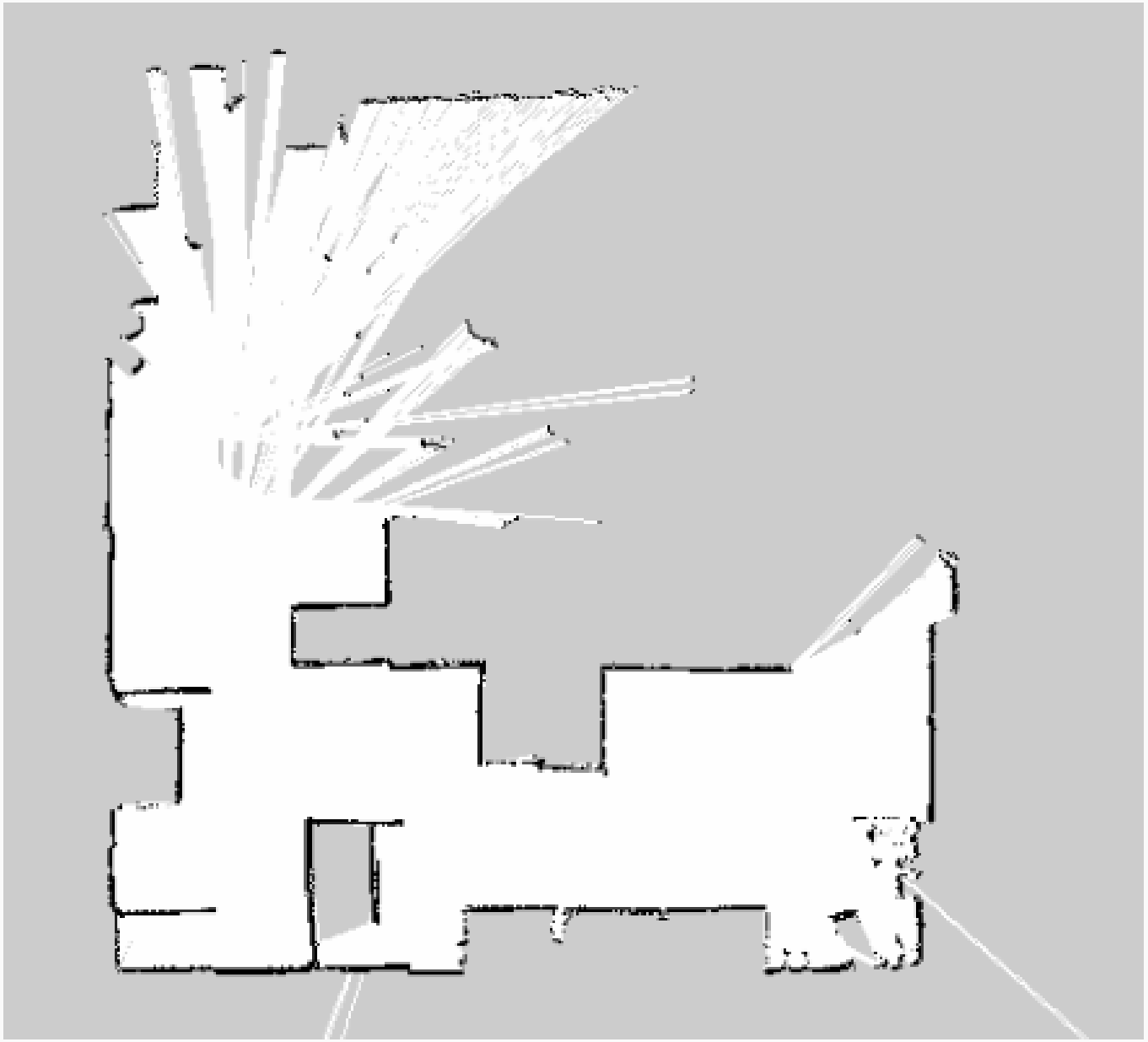
Map building II



Map building III



Map building IV



Localization II

Navigation

- Requirements
  - Map (map_server)
  - Localization (amcl)
  - Odometry source
  - Transforms
  - Sensor sources
  - Goal (move_base)
- Components
  - Planners: global, local
  - Costmaps: global, local
- Output: Velocity command (cmd_vel)

Navigation - Local Planner

- Planner: dwa_local_planner
- Given: plan, costmap, odom
- Generates costs of transversing through map grids
- Output: Velocity command

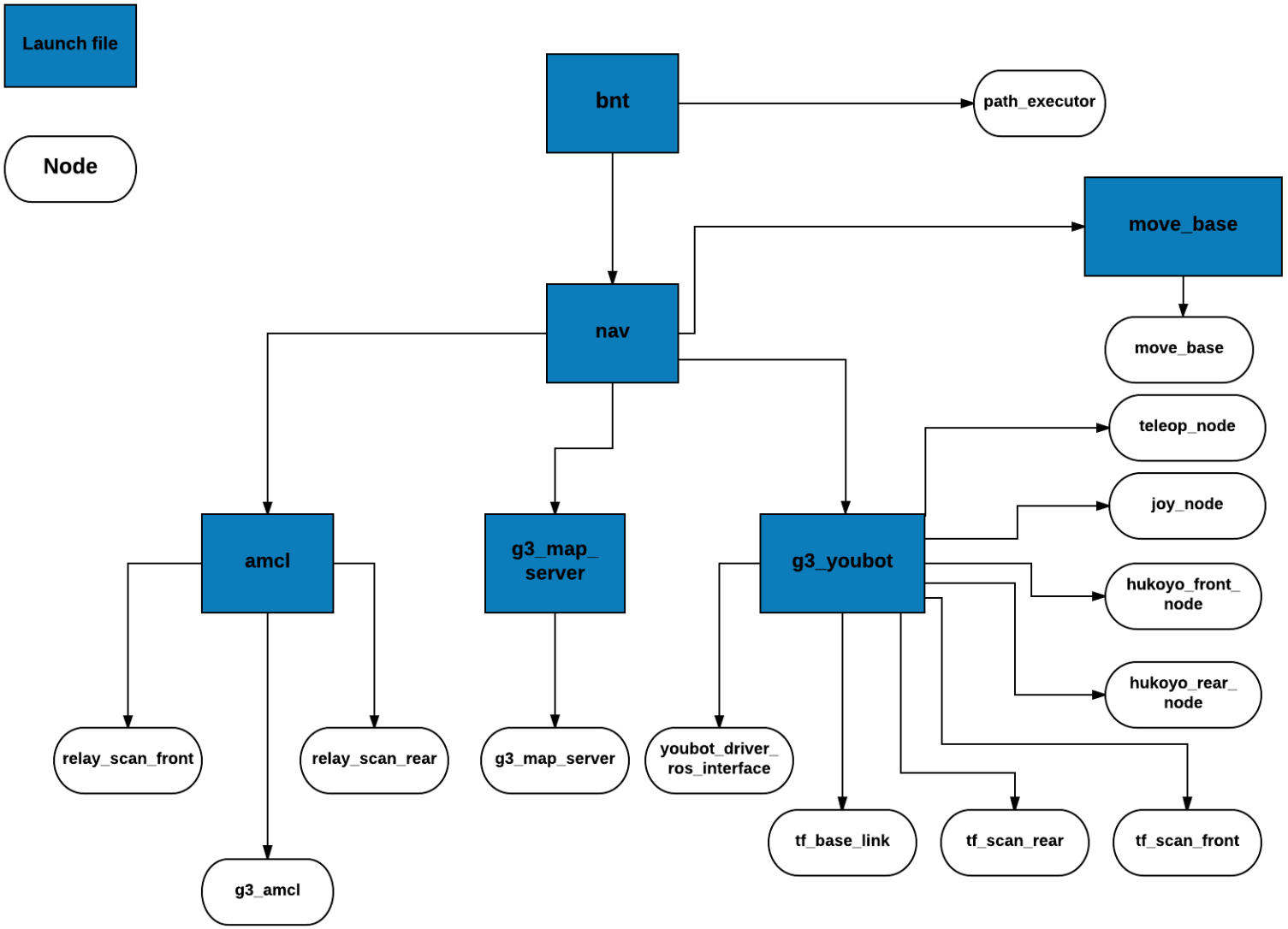
BNT.py

The node acts as path executor that reads a set of user inputs and convert them to move_base_msgs.

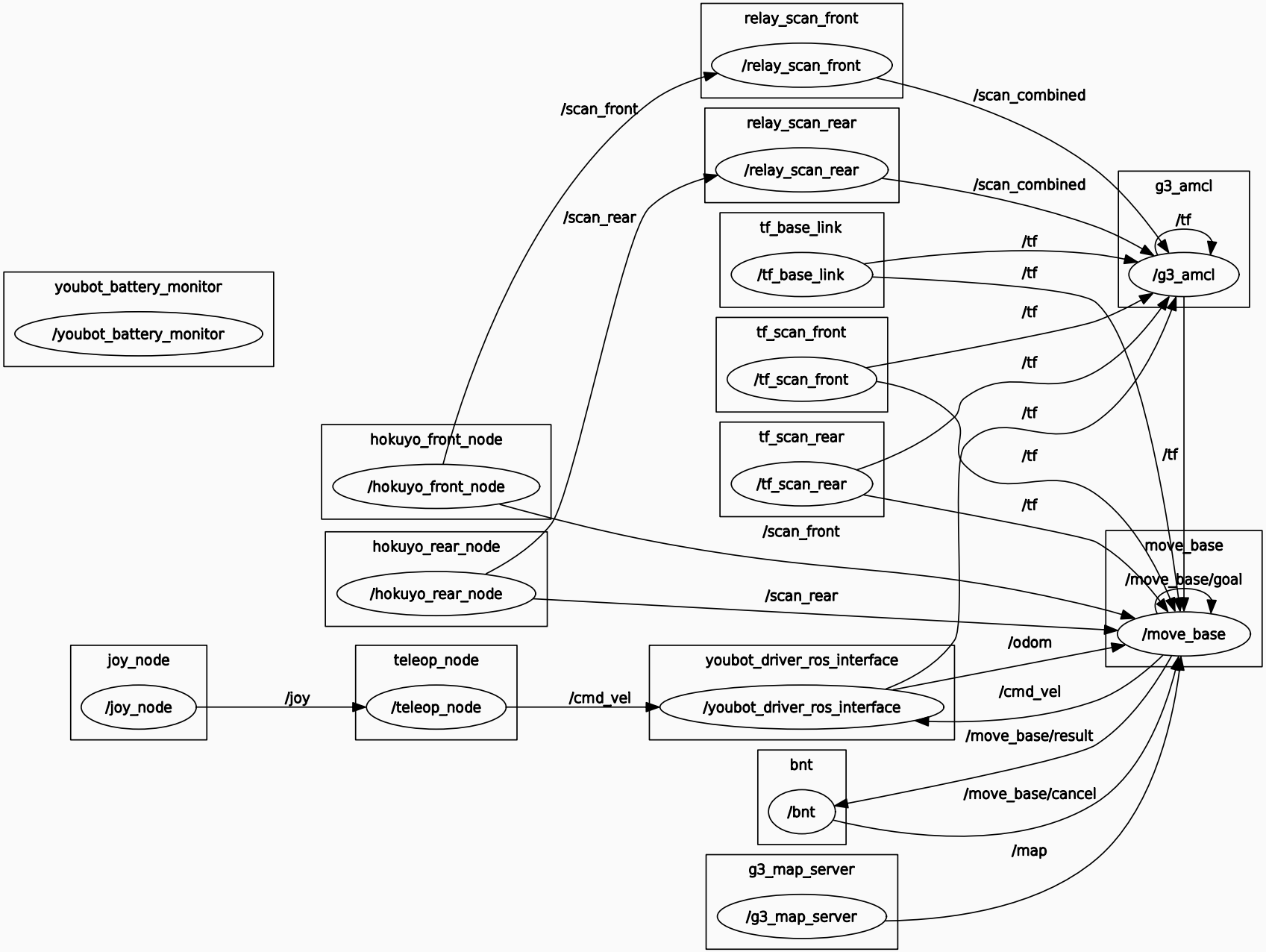
- Class: Position, Pose, Environment, Workspace, PathExecutor
- Functions:
  - Reads user inputs
  - Reads workspace from file
  - Converts workspace to move_base_msgs
  - Clears cost map
  - Sends goal message

Results

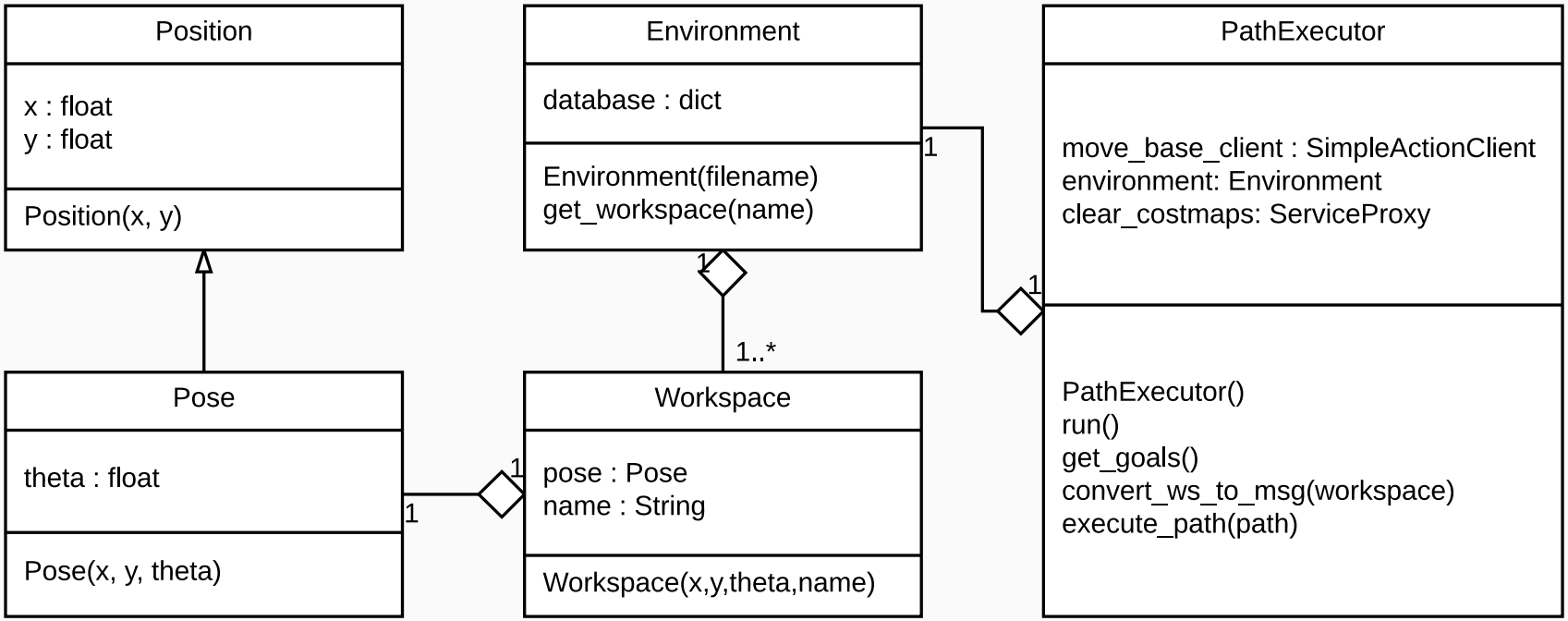
Launch Files



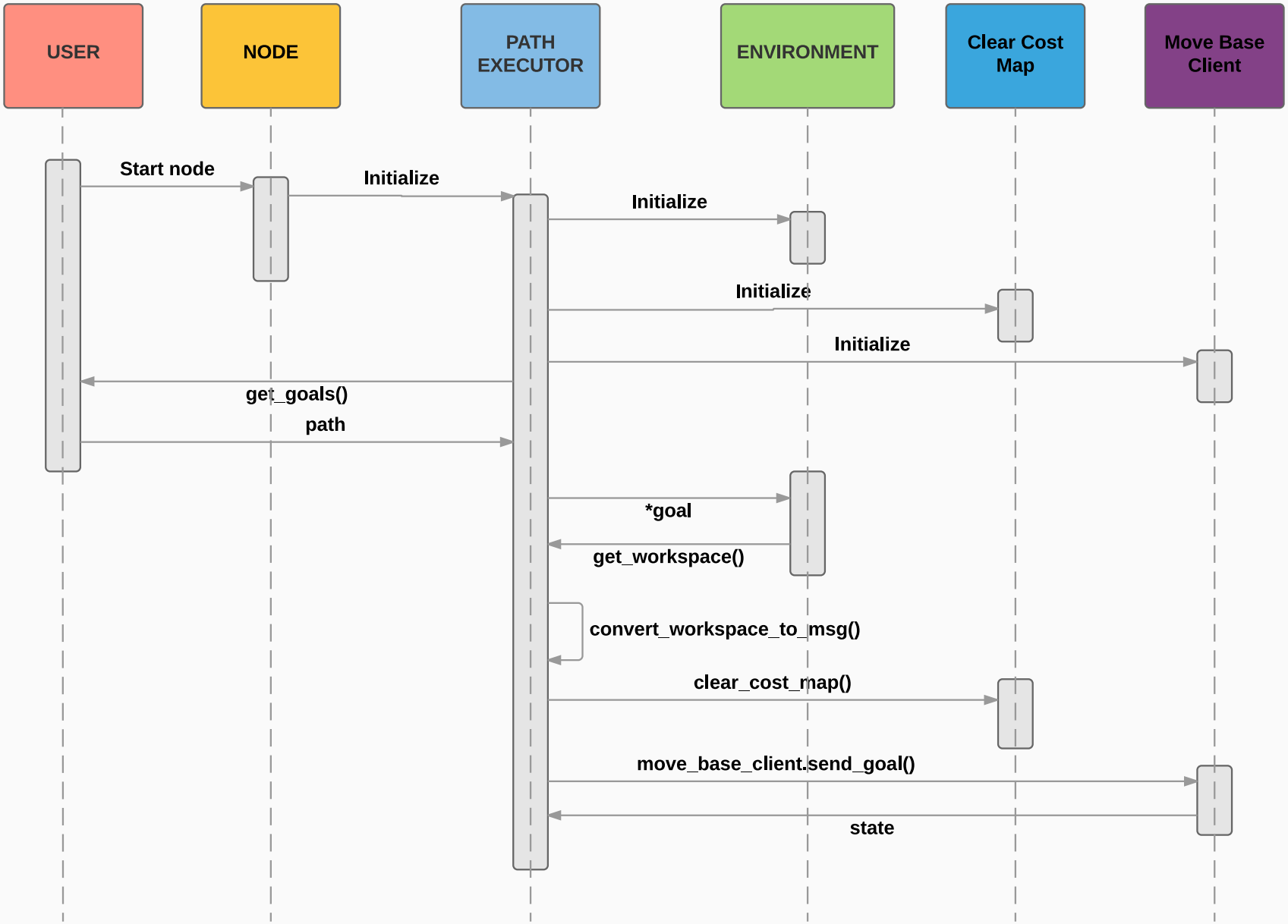
RQT Graph


relay_scan_front
/relay_scan_front
/scan_front
/scan_combined
relay_scan_rear
/relay_scan_rear
/scan_rear
/scan_combined
g3_amcl
/tf
/g3_amcl
tf_base_link
/tf_base_link
/tf
/tf
youbot_battery_monitor
/youbot_battery_monitor
tf_scan_front
/tf_scan_front
/tf
/tf
hokuyo_front_node
/hokuyo_front_node
tf_scan_rear
/tf_scan_rear
/tf
/tf
/tf
/scan_front
hokuyo_rear_node
/hokuyo_rear_node
/scan_rear
move_base
/move_base/goal
/move_base
/odom
joy_node
/joy_node
/joy
teleop_node
/teleop_node
/cmd_vel
youbot_driver_ros_interface
/youbot_driver_ros_interface
/cmd_vel
/move_base/result
bnt
/bnt
/move_base/cancel
/map
g3_map_server
/g3_map_server

Class Diagram



Sequence Diagram



Conclusions

Conclusions

- ROS is a powerful and flexible framework for robot software development.
- Navigation was analyzed and applied to youBot.
- The task contains mapping, localization, path planning, motion execution.
- Mapping and localization were realized in simulation, while all the subtasks were executed on youBot in experiments.
- The subtasks were successively carried out, such that navigation of robot was resolved.
- The robot was able to navigate around the lab by user input of a series of workspace.

Future Work

- User interface can be improved.
- Enable the robot to detect barrier tapes.
- The parameters can be retuned to improve the performace of robot.