Multi-Robot Control (MRC)

Week 6: Action Server w/ MATLAB
Multi-Robot Localization
Action Server with MATLAB

• End goal
  - ROS “action” interface
  - Client provides a goal
  - Server provides
    - Updates to the client during execution
    - Interface for suspending or canceling the action
    - Confirmation when complete
move_base Action Server
1.1.2 Action API

The `move_base` node provides an implementation of the SimpleActionServer (see actionlib documentation), that takes in goals containing geometry_msgs/PoseStamped messages. You can communicate with the move_base node over ROS directly, but the recommended way to send goals to move_base if you care about tracking their status is by using the SimpleActionClient. Please see actionlib documentation for more information.

Action Subscribed Topics

`move_base/goal (move_base_msgs/MoveBaseActionGoal)`
A goal for move_base to pursue in the world.

`move_base/cancel (actionlib_msgs/GoalID)`
A request to cancel a specific goal.

Action Published Topics

`move_base/feedback (move_base_msgs/MoveBaseActionResult)`
Feedback contains the current position of the base in the world.

`move_base/status (actionlib_msgs/GoalStatusArray)`
Provides status information on the goals that are sent to the move_base action.

`move_base/result (move_base_msgs/MoveBaseActionResult)`
Result is empty for the move_base action.
MATLAB Action Client

See gitlab example: action_goal_ex.m
From Last Week

ROS Graph – Single TB3, mapping
From Last Week

Single TB3, tf tree

Recorded at time: 38.934

map

Broadcaster: /turtlebot3_slam_gmapping
Average rate: 23.333
Buffer length: 0.3
Most recent transform: 38.937
Oldest transform: 38.637

odom

Broadcaster: /gazebo
Average rate: 32.895
Buffer length: 0.304
Most recent transform: 38.907
Oldest transform: 38.603

base_footprint

Broadcaster: /robot_state_publisher
Average rate: 10000.0
Buffer length: 0.0
Most recent transform: 0.0
Oldest transform: 0.0

base_link

Broadcaster: /robot_state_publisher
Average rate: 32.895
Buffer length: 0.304
Most recent transform: 38.907
Oldest transform: 38.603

wheel_left_link

imu_link

caster_back_link

base_scan

wheel_right_link
Running Two TB3s Simultaneously

What could go wrong?
ROS Names

Names refer to nodes, topics, services, actions etc. Organized hierarchically, similar to file system, e.g.,

- / (global namespace)
- /cmd_vel
- /robot0/cmd_vel
- /robot1/cmd_vel

Encapsulation and name resolution

- base
- relative/name
- /global/name
- ~private/name
Namespace Encapsulation via `<group>`

```xml
<node pkg="rosserial_python"
name="turtlebot3_core">

<node pkg="hls_driver"
name="turtlebot3_lds">
```
Namespace Encapsulation via `<group>`

```xml
<node pkg="rosserial_python" name="turtlebot3_core"/>
<node pkg="hls_driver" name="turtlebot3_lds">
</node>

<group ns="tb3_0">
    <node pkg="rosserial_python" name="turtlebot3_core">
    </node>
    <node pkg="hls_driver" name="turtlebot3_lds">
    </node>
</group>

<group ns="tb3_1">
    <node pkg="rosserial_python" name="turtlebot3_core">
    </node>
    <node pkg="hls_driver" name="turtlebot3_lds">
    </node>
</group>
```

```xml
/tb3_0
/tb3_1
```
tf and tf2 – transform library

- Keep track of multiple coordinate frames
- Maintain relationship between frames in tree structure
- Enables transform of points, vectors, etc. between any two frames at any desired time!

TB3: turtlebot3bringup turtlebot3_robot.launch
LAP: turtlebot3bringup turtlebot3_remote.launch
tf for multiple robots

We have same problem – how to have multiple copies of the same tf tree without conflict?

• tf_prefix – analogous to namespace

TB3: ROS_NAMESPACE=tb3_0 roslaunch turtlebot3Bringup turtlebot3_robot.launch
    multi_robot_name:="tb3_0" set_lidar_frame_id:="tb3_0/base_scan"

LAP: ROS_NAMESPACE=tb3_0 roslaunch turtlebot3Bringup turtlebot3_remote.launch
    multi_robot_name:=tb3_0
TB3: ROS_NAMESPACE=tb3_0 roslaunch turtlebot3_bringup turtlebot3_robot.launch
multi_robot_name:="tb3_0" set_lidar_frame_id:="tb3_0/base_scan"

```xml
<launch>
  <arg name="multi_robot_name" default=""/>
  <arg name="set_lidar_frame_id" default="base_scan"/>

  <include file="$(find turtlebot3_bringup)/launch/turtlebot3_core.launch">
    <arg name="multi_robot_name" value="$(arg multi_robot_name)"/>
  </include>

  <include file="$(find turtlebot3_bringup)/launch/turtlebot3_lidar.launch">
    <arg name="set_lidar_frame_id" value="$(arg set_lidar_frame_id)"/>
  </include>

  <node pkg="turtlebot3_bringup" type="turtlebot3_diagnostics" name="turtlebot3_diagnostics" output="screen"/>
</launch>
```

```xml
<launch>
  <arg name="multi_robot_name" default=""/>

  <node pkg="rosserial_python" type="serial_node.py" name="turtlebot3_core" output="screen">
    <param name="port" value="/dev/ttyACM0"/>
    <param name="baud" value="115200"/>
    <param name="tf_prefix" value="$(arg multi_robot_name)"/>
  </node>
</launch>
```
LAP: ROS_NAMESPACE=tb3_0 roslaunch turtlebot3_bringup turtlebot3_remote.launch
multi_robot_name:=tb3_0

```xml
<launch>
  <arg name="model" default="$(env TURTLEBOT3_MODEL)" doc="model type [burger, waffle, waffle_pi]"/>
  <arg name="multi_robot_name" default=""/>
  <include file="$(find turtlebot3_bringup)/launch/includes/description.launch.xml">
    <arg name="model" value="$(arg model)" />
  </include>

  <node pkg="robot_state_publisher" type="robot_state_publisher" name="robot_state_publisher">
    <param name="publish_frequency" type="double" value="50.0" />
    <param name="tf_prefix" value="$(arg multi_robot_name)"/>
  </node>
</launch>
```
Setting tf_prefix

TB3: ROS_NAMESPACE=tb3_0 roslaunch turtlebot3_bringup turtlebot3_robot.launch
multi_robot_name:="tb3_0" set_lidar_frame_id:="tb3_0/base_scan"

LAP: ROS_NAMESPACE=tb3_0 roslaunch turtlebot3_bringup turtlebot3_remote.launch
multi_robot_name:=tb3_0
Result – Two Distinct Namespaces

/tb3_1
- /tb3_1/joy_node
- /tb3_1/joy
- /tb3_1/teleop_twist_joy
- /tb3_1/cmd_vel
- /tb3_1/turtlebot3_core
- /tb3_1/map_server
- /tb3_1/turtlebot3_ids
- /tb3_1/sensor_state
- /tb3_1/joint_states
- /tb3_1/robot_state_publisher
- /tb3_1/scan
- /tb3_1/mu
- /tb3_1/amcl
- /tb3_1/firmware_version
- /tb3_1/turtlebot3_diagnostics
- /tb3_1/particlecloud

/tb3_0
- /tb3_0/joy_node
- /tb3_0/joy
- /tb3_0/teleop_twist_joy
- /tb3_0/cmd_vel
- /tb3_0/turtlebot3_core
- /tb3_0/map_server
- /tb3_0/turtlebot3_ids
- /tb3_0/sensor_state
- /tb3_0/joint_states
- /tb3_0/robot_state_publisher
- /tb3_0/firmware_version
- /tb3_0/amcl
- /tb3_0/particlecloud

/tf
- /tf_static

/tb3_1/scan
/tb3_1/mu
/tb3_1/amcl
/tb3_1/turtlebot3_diagnostics
/tb3_1/particlecloud
Two tf trees with common map

[Diagram of two tf trees with a common map]
Two Robot Localization: Initial