Multi-Robot Control (MRC)

W4: Towards Robot Control
Plan of the Week

Topics
- MATLAB-ROS
  - Communicating with ROS nodes: topics and services
- MATLAB Control
  - Subscribe and Wait
  - Callbacks
- Robot Simulation Environment: Gazebo

Activities
- Read PRR Ch 6
- Do tutorials: Gazebo and MATLAB-RST
- Do Assignment 4
Towards MATLAB Control

We need MATLAB to “speak” ROS

• Publish cmd_vel and subscribe to odom
Towards MATLAB Control

We need MATLAB to “speak” ROS

- Publish cmd_vel and subscribe to odom
Towards MATLAB Control

We need MATLAB to “speak” ROS

• Publish cmd_vel and subscribe to odom
Towards MATLAB Control

We need MATLAB to “speak” ROS
- Publish cmd_vel and subscribe to odom

geometry_msgs/Point Message

File: geometry_msgs/Point.msg

Raw Message Definition

```
# This contains the position of a point in free space
float64 x
float64 y
float64 z
```

Compact Message Definition

```
float64 x
float64 y
float64 z
```

geometry_msgs/Pose

```
geometry_msgs/pose
c768r64[36] covariance
```

nav_msgs/Odometry

```
geometry_msgs/Twist
```

Gazebo Simulated Environment and Robot Goal Waypoint (X,Y)
Towards MATLAB Control

We need MATLAB to “speak” ROS

- Publish `cmd_vel` and subscribe to `odom`

```matlab
message.Pose.Pose.X;
message.Pose.Pose.Y;
```
ROS ↔ MATLAB

ROS command line vs. MATLAB for calling a service

```matlab
% Porting ROS commands to MATLAB
rosinit

% rosservice call /turtle1/teleport_absolute 1 1 1.57
rosservice info /turtle1/teleport_absolute
teleClient = rossvcclient('/turtle1/teleport_absolute')
teleMsg = rosmessage(teleClient)
teleMsg.X = 1;
teleMsg.Y = 1;
teleMsg.Theta = 1.57
teleResp = call(teleClient,teleMsg,'Timeout',3)
```
ROS ↔ MATLAB

ROS command line vs. MATLAB for publishing a topic

% rostopic pub -1 /turtle1/cmd_vel geometry_msgs/Twist -- '[4.0, 0.0, 0.0]'

to = 1.5; % timeout
t1Pub = rospublisher('/turtle1/cmd_vel', rostype.geometry_msgs_Twist);
t1Msg = rosmessage(t1Pub);
t1Msg.Linear.X = 4.0;
send(t1Pub, t1Msg)
pause(to);
ROS and Gazebo Simulation
Waypoint Guidance and Control

Problem Statement
• Given estimate of current state: \( x, y, \theta \)
  and waypoint (goal position): \( x_g, y_g \)
• Find commands to achieve the goal
  – Linear velocity: \( u \)
  – Angular velocity: \( \omega \)

Guidance
• Determine desired heading: \( \theta_g \)
  – Geometry - \text{atan2}
• Determine desired speed:
  – Proportional to distance (d)
    with max. value \( u = \min(K_u(d), u_{\text{nom}}) \)

Control
• Proportional heading \( \omega = K_h(\theta_g - \theta) \)

End Condition: \( d < d_{\text{tol}} \)
function dist=turtle_waypoint(X,Y)

roslit;

% Setup publisher/subscriber
poseSub = rossubscriber('/turtle1/pose');
cmdPub = rospublisher('/turtle1/cmd_vel');
cmdMsg = rosmesage(cmdPub);

% Get the current pose
poseMsg = receive(poseSub)

cmdMsg.Linear.X = ??;
cmdMsg.Angular.Z = ??;
send(cmdPub,cmdMsg);

roshutdown

return
Implementation: Callback Function

Event driven function
- When something happens...
- Execute the function

Subscribe with callback function

rossubscriber
Subscribe to messages on a topic

Syntax
```
sub = rossubscriber(topicname)
sub = rossubscriber(topicname, msgtype)
sub = rossubscriber(topicname, callback)
sub = rossubscriber(topicname, msgtype, callback)
```
% Setup subscription - which implements our controller.
% We pass the publisher, the message to publish and the goal as
% additional parameters to the callback function.
odmsub = rossubscriber('sim_p3at/odom',{@p3odomCallback,cmdpub,cmdmsg,goal})

function p3odomCallback(~,message,cmdpub,cmdmsg,goal)
% Waypoint control - this function is called when we receive an odom msg.

% Publish the velocity commands
cmdmsg.Linear.X = v;
cmdmsg.Angular.Z = w;
send(cmdpub,cmdmsg);
Summary

Reading: PRR Ch 6 – Robots and Simulators
Tutorials: Gazebo (w/ ROS) and MATLAB RST

Assignment
• 1) Using MATLAB to publish topics and call services
  – Porting your letter generating shell script
• 2) MATLAB waypoint control of a turtle
  – Subscribe and wait
• 3) Gazebo simulation
  – Drive a robot with a script
• 4) MATLAB waypoint control of a simulated robot
  – Callback function