SC-635 Advanced Topics in Mobile Robotics

Experiment Module : Custom messages and trilateration

January 29, 2021

Systems and Control Engineering

Indian Institute of Technology Bombay

Overview

1. Custom messages

Recapitulation

- Concepts : **nodes**, **topics**, **messages**, **publishing**, **subscribing**, **rosgraph**
- Tools : **roscore**, **rosrun**, **roslaunch**, **rostopic**, **rqt_graph**
- Created a workspace **sc635_workspace** : One time exercise
- Created a **'pub_sub'** project
- Created a **Launch file** and passed arguments from launch file
- Kobuki Mobile Base : Robot model simulation using gazebo
- Controlling Kobuki robot
  - **/odom**
  - **/mobile_base/commands/velocity**

Custom messages

In lab assignment 1 we scripted

- Name publisher node with **String** message type
- Roll number publisher node with **Numeric** message type

We created separate nodes to publish each type.

With custom messages we can encode specialized information such as collection of simple message types.
Some examples:

- Collection of robot poses in multi-agent setting
- Club output of multiple sensors together in one message
- Sensor parameters in one message

Custom message: Student_info

Lets create a custom message *student_info* with following fields

- Name
- Roll number

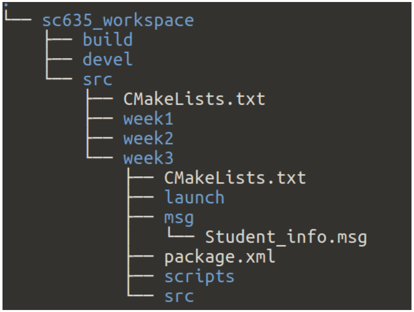
Custom messages are usually placed under the msg directory

Student_info.msg contents

```
string name
int64 roll_number
```

Directory structure

The directory structure with msg directory should appear as follows:



```
.
└── sc635_workspace
    ├── build
    ├── devel
    └── src
        ├── CMakeLists.txt
        ├── week1
        ├── week2
        └── week3
            ├── CMakeLists.txt
            ├── launch
            ├── msg
            │   └── Student_info.msg
            ├── package.xml
            ├── scripts
            └── src
```

Modifications 1

We notify *catkin_make* by modifying the contents of *package.xml* file

```
1  <?xml version="1.0"?>
2  <package format="2">
3    <name>template_a3</name>
4    <version>0.0.0</version>
5    <description>The week3 package</description>
6     ...
7    <buildtool_depend>catkin</buildtool_depend>
8    <build_depend>rospy</build_depend>
9
10   <build_depend>message_generation</build_depend>
11   <exec_depend>message_runtime</exec_depend>
12
13     <build_export_depend>rospy</build_export_depend>
14   <exec_depend>rospy</exec_depend>
15
16   <export>
17     <!-- Other tools can request additional information be placed here -->
18   </export>
19 </package>
```

Line 10,11 are added to let the build system know that we want to use custom messages.

Modifications 2

We also need to modify CMakeLists.txt file

```
cmake_minimum_required(VERSION 2.8.3)
project(week3)
## Components
find_package(catkin REQUIRED COMPONENTS
  rospy
  roscpp
  std_msgs                          ## modified
  message_generation                ## modified
)
## Generate messages in the 'msg' folder
add_message_files(
    FILES
    Student_info.msg                ## modified
)
## Generate added messages and services with any dependencies listed here
generate_messages(
  DEPENDENCIES
  std_msgs                          ## modified
)
## catkin specific configuration
catkin_package(
 CATKIN_DEPENDS message_runtime     ##modified
)

## Specify additional locations of header files
include_directories(
  ${catkin_INCLUDE_DIRS}
)
```

Custom message: Publisher

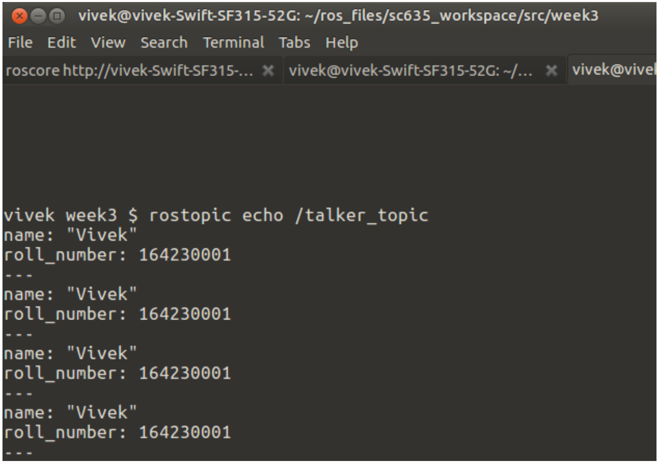
Lets use the custom message by writing a publisher:

```
1  #!/usr/bin/env python
2  import rospy
3  from week3.msg import Student_info
4  
5  def talker():
6      rospy.init_node('talker_node', anonymous=True)
7      pub = rospy.Publisher('talker_topic', Student_info, queue_size=10)
8  
9      rate = rospy.Rate(1)
10     while not rospy.is_shutdown():
11         t = Student_info("Vivek", 164230001)
12         pub.publish(t)
13         rospy.loginfo("Sent␣a␣message!")
14         rate.sleep()
15 
16 if __name__ == '__main__':
17     try:
18         talker()
19     except rospy.ROSInterruptException:
20         pass
```

In line 11, we are creating a custom message.

Custom message : echo

We expect to see the following:


vivek@vivek-Swift-SF315-52G: ~/ros_files/sc635_workspace/src/week3
File Edit View Search Terminal Tabs Help
roscore http://vivek-Swift-SF315-...
vivek@vivek-Swift-SF315-52G: ~/...
vivek@vivel

```
vivek week3 $ rostopic echo /talker_topic
name: "Vivek"
roll_number: 164230001
---
name: "Vivek"
roll_number: 164230001
---
name: "Vivek"
roll_number: 164230001
---
name: "Vivek"
roll_number: 164230001
---
```

Custom message: Trilateration

An example custom message that carries trilateration data:

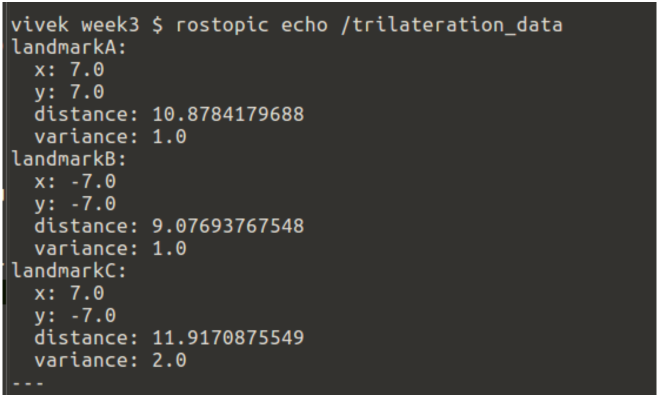
file **Landmark.msg**

```
float32 x
float32 y
float32 distance
float32 variance
```

file **Trilateration.msg**

```
Landmark landmarkA
Landmark landmarkB
Landmark landmarkC
```

Trilateration message : echo



```
vivek week3 $ rostopic echo /trilateration_data
landmarkA:
  x: 7.0
  y: 7.0
  distance: 10.8784179688
  variance: 1.0
landmarkB:
  x: -7.0
  y: -7.0
  distance: 9.07693767548
  variance: 1.0
landmarkC:
  x: 7.0
  y: -7.0
  distance: 11.9170875549
  variance: 2.0
---
```

Reference **Landmark.msg**

```
float32 x
float32 y
float32 distance
float32 variance
```

Summary

- Concepts : **nodes**, **topics**, **messages**, **publishing**, **subscribing**, **rosgraph**
- Tools : **roscore**, **rosrun**, **roslaunch**, **rostopic**, **rqt_graph**
- Created a workspace **sc635_workspace** : One time exercise
- Created a **'pub_sub'** project
- Created a **Launch file** and passed arguments from launch file

- Kobuki Mobile Base : Robot model simulation using gazebo
- Controlling Kobuki robot
  - **/odom**
  - **/mobile_base/commands/velocity**

- **Custom messages**