Tutorial on Robot Operating System (ROS)

Mayank Mittal

Indian Institute of Technology, Kanpur

mayankm@iitk.ac.in

June 7, 2018

Requirements

- Ubuntu 16.04LTS
- ROS Kinetic Kame
- OpenCV 3.1 *(pre-installed with ros full desktop installation)*

Objective

In this tutorial, you will learn to write a C++ node that shall read an image file and publish it using OpenCV and ROS

Step 1: Installing ROS packages

Install following dependencies: `roscpp`, `image_transport`, `cv_bridge`

Instruction

```
$ sudo apt-get install ros-kinetic-image-transport
        ros-kinetic-cv-bridge
```

Step 2: Setup catkin workspace

Create a directory

Instruction

```
$ mkdir -p ~/catkin_ws/src
```

Create catkin workspace

Instruction

```
$ cd ~/catkin_ws/src
$ catkin_init_workspace
```

Step 2: Build catkin workspace

Build the empty workspace

Instruction

```
$ cd ..
$ catkin_make
```

Step 3: Creating a package

Move to `src` directory

Instruction

```
$ cd ~/catkin_ws/src
```

Create catkin package

Instruction

```
$ catkin_create_pkg image_pub roscpp
        image_transport cv_bridge
```

Step 2: Build catkin workspace

Build the empty workspace

Instruction

```
$ cd ..
$ catkin_make
```

Step 4: Metadata (A first Glance)

Move to package `image_pub/` directory and view the `package.xml` file

Instruction

```
$ cd ~/catkin_ws/src/image_pub
$ vim package.xml
```

View `CMakeLists.txt` file

Instruction

```
$ vim CMakeLists.txt
```

Step 5: Import an image

Download an image from the internet and move it to package's `src` directory

Instruction

```
$ mv ~/Dowmloads/<image-name>.png
       ~/catkin_ws/src/image_pub/src
```

Step 5: Writing first node

Open package's `src directory`

Instruction

```
$ cd ~/catkin_ws/src/image_pub/src
```

In new text editor start writing your node

Instruction

```
$ vim image_file_publisher.cpp
```

Step 5: Writing first node

Write the following header files

Instruction

```
#include "ros/ros.h"
#include <ros/console.h>
#include <ros/package.h>

#include <image_transport/image_transport.h>
#include <opencv2/highgui/highgui.hpp>
#include <cv_bridge/cv_bridge.h>

#include <sstream>
#include <stdio.h>

using namespace std;
```

Step 5: Writing first node

Write the `main` function

Instruction

```
int main(int argc, char **argv) {

  return 0;
}
```

Step 5: Writing first node

Initialize the ROS node handle

Instruction

```
int main(int argc, char **argv) {
  ros::init(argc, argv, "file_publisher");

  // creating nodehandle
  ros::NodeHandle nh;

  return 0;
}
```

Step 5: Writing first node

Write parameters to the node

Instruction

```
...
  // creating nodehandle
  ros::NodeHandle nh;

  // fetching parameters for the node
  int frequency = 10;
  string file = "/full/path/to/<image-name>.png";
...
```

Step 5: Writing first node

Define a image publisher

Instruction

```
...
  string file = "/full/path/to/<image-name>.png";

  // defining publisher using image_transport
  image_transport::ImageTransport it(nh);
  image_transport::Publisher pub =
                  it.advertise("image", frequency);
...
```

Step 5: Writing first node

Read image

Instruction

```
...
  //read input from the argument passed to file
  cv::Mat image = cv::imread(file,
                  CV_LOAD_IMAGE_COLOR);
...
```

Step 5: Writing first node

Convert to `sensor_msgs/Image.h`

Instruction

```
...
  sensor_msgs::ImagePtr msg =
        cv_bridge::CvImage(std_msgs::Header(),
    "bgr8", image).toImageMsg();
  msg->header.frame_id = "camera_link";

...
```

Step 5: Writing first node

Publish data

Instruction

```
...
  ros::Rate loop_rate(frequency);

  while (nh.ok()) {
    msg->header.stamp = ros::Time::now();
    pub.publish(msg);
    ros::spinOnce();
    loop_rate.sleep();
    ROS_INFO("Image from File Published");
  }
  return 0;
}
```

The End