Task 1.2: Problem Statement

blogpost-style, resources, task-1-2

Smit e-Yantra Staff

Oct '21

Task 1.2

Overview

In this task, we will be learning the following:

- Getting started with ROS
- Subscribing video frames from a ROS topic
- Publishing messages over a ROS topic

In the previous task, we learned how to detect an ArUco marker from a static image, the next problem is to detect the ArUco marker from a video stream from a camera in the simulator Gazebo. The result has to be published on a *rostopic*.

Prerequisites

Installations

- Update the strawberry_stacker package from github, follow the process to do so

```
cd ~/catkin_ws/src/strawberry_stacker
git add .
git commit -m "Put any message here"
git pull origin master
```

- Build your workspace

```
cd ~/catkin_ws
catkin build
```

Learning resources

- By this time, it is assumed you have completed the task 1.1 and attempted the bonus task.
- We highly recommend you follow **ROS tutorials** (following till 1.1.13 will be enough for this task) before proceeding with the actual task as the learning curve might be steep in the start.
- This task will require the knowledge of *rosnodes*, *rostopic*, publisher & subscriber model
- Skip to main content sion for tips and tricks will be scheduled soon after the launch, do attend the session for getting familiar with ROS

Problem statement

- Create a *rosnode* named *marker_detection* in a python script, which will detect a moving ArUco marker from the video feed of camera and will publish the id, position and orientation of the marker on a *rostopic /marker_info*
- You need to subscribe to a *rostopic* named */camera/camera/image_raw* to read camera the video frames from camera
- Apply the ArUco detection on these frames and publish the results on the *rostopic /marker_info* in the message type *task_1/Marker*

Procedure

- Complete the boiler plate script provided to you in the *scripts* folder
- Launch the Gazebo world by typing the following command

  ```
  roslaunch task_1 task1_2.launch
  ```

  Once the simulation window launches, you should see a static camera in air poiting downwards and there will be an ArUco marker moving in a pattern.

- Run your python script in a separate terminal to start detecting and publishing the ArUco details

  ```
  rosrun task_1 marker_detection.py
  ```

  > Note: To avoid manually typing the rosrun command for every iteration, you can start the rosnode in the launch file itself, to do that add the following lines in the `task_1.2.launch` file in the launch folder. Make sure you add the line before the `</launch>` line.

  ```
  <node name="marker_detection" type="marker_detection.py" pkg="task_1" />
  ```

Expected output

As soon as you start running the *marker_detection* node, the *rosnode* should start publishing the data over the *rostopic /marker_info* with the frequency of **10hz**

Recording and Submission

Recording Logs

- ROS allows us to record a log of the messages that occurred in a given time period. This is like recording a data stream. The ROS utility which does this is called **rosbag** , and the command to capture the data is `rosbag record`.
- You can run the `rosbag record` command separately on the command line. But to not loose any data you will have to start recording precisely at the same moment you start publishing messages. Hence it is a much more preferable option to include the rosbag recording in the launch file itself. It has already been added in the launch file and you need to enable it using the arg `record:="true"`
- Before recording the rosbag, make sure you have completed the task and you are ready with the expected output
- To record your submission, you need to enable recording by typing the following command

  ```
  roslaunch task_1 task1_2.launch record:="true" rec_name:="aruco_detection.bag"
  ```

- This will record and generate a bag file named *aruco_detection.bag*
- Make sure the recording is complete, to verify that the recording was done, look for this message in the terminal after 30 seconds

  ```
  [rosbag_record_aruco-6] process has finished cleanly
  ```

- After the recording is done, you will find the bag file in the folder named *bag_files* in the package

  > Note: bag files with the same name will be overwritten by the rosbag utility without a prompt/warning. Make sure you provide proper name for each iteration if you want to save them all.

Skip to main content

- Verify that your bag file is properly recorded by using the `rosbag info` command followed by the absolute or relative path of the file. To do so, enter the following command…

  ```
  roscd task_1/bag_files
  rosbag info <NameOfBagFile>.bag
  ```

  Verify that the desired topics `/marker_info` & `/gazebo/model_states_throttle` are mentioned.

- You can use the `rosbag play` command to see the messages you've recorded in the same order and rate. You can verify this by running the and playing your bag file

  ```
  rosbag play <NameOfBagFile>.bag
  ```

Submission instructions

- Rename all your python scripts with a prefix <SS_team_id>, for example the script name is marker_detection.py and your team id is 1234, then rename the file as **SS_1234_marker_detection.py**
- Also rename the bag file to <SS_team_id>.bag, eg. **SS_1234.bag**
- Compress the python script along with the bag file **directly** to a zip file and name it with the file name <SS_team_id>.zip, for eg. **SS_1234.zip**
- Overall, your directory should look like this

  ```
  __SS_1234.zip
     |__SS_1234_marker_detection.py
     |__SS_1234.bag
  ```

- Submit the zip file on portal

All the best !

Task 1 : Getting Started with ArUco and ROS

CLOSED ON OCT 23, '21

UNLISTED ON OCT 23, '21