Intel Realsense Cameras to do SLAM

This project involves using Intel Realsense D415 and T265 cameras to achieve SLAM (D435 camera should also work with this). Specifically, this is done using **Occupancy-mapping**, a branch of realsense-ros.

Getting Started

These instructions will get you a copy of the project up and running on your local machine for development and testing purposes.

Prerequisites

This was done using Jetson Nano (Ubuntu 18.04.4 LTS) since Realsense was originally made to be run with Ubuntu and thus has easier setup.

Installing

If you have RealSense SDK and ROS installed and have set up a catkin workspace, skip to step 3.

1. Install Realsense

In order to install the RealSense SDK, we must first add the Intel repository. We will be using the tutorials from JetsonHacks, which uses convenience script to perform these actions.

- From a Terminal, clone the repository, and switch over to that repositories directory:

```
git clone https://github.com/JetsonHacksNano/installLibrealsense
cd installLibrealsense
```

- Install the sdk:

```
./installLibrealsense.sh
```

- Start realsense-viewer:

```
realsense-viewer
```

If you are having difficulty, follow the Jetsonhacks tutorial on installing Realsense until 7:30 min mark.

Make sure you get it running on the Realsense viewer before moving on to next steps.

2. Install ROS

- Clone repository on to the Jetson:

```
git clone https://github.com/JetsonHacksNano/installROS.git
cd installROS
```

- Run installROS

```
./installROS.sh -p ros-melodic-desktop
```

If you don't specificy package, then it'll install ros-melodic-ros-base by default.

- Setup Catkin Workspace

```
./setupCatkinWorkspace.sh optional-workspace-name
```

where optional-workspace-name is the name and path of the workspace to be used.

The default workspace name is catkin_ws. If a path is not specified, the default path is the current home directory. This script also sets up some ROS environment variables.

3. Install RealSense Wrapper for ROS

If already have a **src** folder in your catkin workspace, skip this bullet.

- Make a src folder in your workspace:

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

- Navigate to src and clone the occupancy-mapping branch:

```
cd ~/catkin_ws/src/
git clone -b occupancy-mapping https://github.com/IntelRealSense/realsnse-ros.git
cd installRealSenseROS
./installRealSenseROS catkin-workplace-name
```

catkin-workplace-name is the path to the catkin_workspace to place the RealSense ROS package. If no catkin workplace name is specified, the script defaults to ~/catkin_ws.

4. Set the Parameters

The parameters to specify or fine-tune include resolution, HeightOfInterestMin, HeightOfInterestMax, DepthOfInterestMin, and DepthOfInterestMax.

- Navigate to occupancy's launch directory: From the catkin directory, this would be

```
cd src/realsense-ros/occupancy/launch
```

- Open the occupancy.launch file with your favorite editor
- Edit the parameters.

Note: the produced map seems to be very sensitive to HeightOfInterestMin and HeightOfInterestMax parameters.

HeightOfInterestMin/Max: space above and below the camera that is considered when determining if the space is occupied. This depends on where the camera is mounted [on your robot] and the height of the robot.

For example min. is the height above ground (including some margin) and max. is the height of the robot minus height of the camera (plus some margin). So, HeightOfInterestMin = "-0.5" means that the camera is located 50cm above the ground.

5. [Optional] Set Camera Serial Numbers When working with multiple cameras, it will be useful to enter the camera serial numbers into the launch file so that they can be identified correctly.

- Navigate to realsense2_camera launch folder.
- Open the rs_d400_and_t265.launch file using vim/your favorite editor
- Edit the **serial_no_camera1** and **serial_no_camera2** arguments with camera serial number (12-digits) found on the cameras themselves.

How to Run

1. Navigate to your catkin workspace

- Open a terminal window and enter

```
cd name-of-your-catkin-workspace
```

If you followed the jetsonhacks tutorials, the default name of your catkin workspace should be **catkin_ws**

2. Source the Devel

This step can only be done in the root catkin directory

NOTE: You must source the devel in every terminal window you open

```
source devel/setup.bash
```

3. Set ROS Network This step involves setting the ROS Network.

Note: This must be done in every window you use. See below.

- For first terminal window:

```
export ROS_HOSTNAME=localhost
export ROS_MASTER_URI=http://localhost:11311
roscore
```

- For all other terminal windows that you use:

```
export ROS_HOSTNAME=localhost
export ROS_MASTER_URI=http://localhost:11311
```

4. Run the launch file

```
roslaunch occupancy occupancy_live_rviz.launch
```

6. View the Map

Once RViz is running, make sure the Map checkbox is ticked under Displays (on the left half of the screen). If Map is not one of the displays listed, you can add it (button at the bottom left side)

```
Add -> By display type -> rviz -> Map
```

You can specificy what map topic you want. (grid_map, octomap_grid, grid_prob_map, etc). grid_map works quite well, but it is definitely worth experimenting.

7. [Optional] Add Other Displays

To add the D415 depth camera display:

```
Add -> By topic -> /d400 (what you named d415 camera) -> /depth -> /image_rect_raw -> double click image
```

That's it! Good Luck! See the below section for some tips on common errors and mistakes!

Saving the Map

1. Install the map-server package:

```
sudo apt-get install ros-melodic-map-server
```

2. Saving the map to file

- Open a new terminal and navigate to the folder where you want to save your map

- Source the devel:

```
    source devel/setup.bash
```

- Set up the ROS network if required (see above sections for instructions).
- Run the following command:

```
    rosrun map_server map_saver map:=[namespace/topic]
```

Map will only be saved if you enter the right namespace/topic for the location of the map. Verify the namespace/topic by checking the tree and nodes.

For example, if using grid-map topic in rtab-map, you would run the following:

```
rosrun map_server map_saver map:=/rtabmap/grid_map
```

If the terminal hangs after "", map_server is not subscribed to right topic. Re-check the tree and how the topics/nodes are organized and try again.

General Tips & Common Errors

This section will discuss potential and common errors.

1. control_transfer returned error, index: 768, error: No data available, number: 61

Generally, you can ignore this error (apparently the developers haven't removed this error).

2. ResourceNotFound: ___

For this error, you probably haven't installed what the error is pointing to (ex. realsense2_camera).

Ex. ResourceNotFound: realsense2_camera Fix: Run the following and try again

```
    sudo apt install ros-melodic-realsense2-camera
```

If you are unsure of the install command, Google it! :)

3. roslaunch command runs but hangs almost immediately

This error generally has to do with ROS network settings. Might arise if you're working from home, new location, etc.

Make sure you've set the ROS_HOSTNAME, ROS_MASTER_URI, etc. See Step 3 under How to Run.

If you're still having issues, see ROS Network Setup.

4. Not seeing Map/Warning/Errors in RViz, RTabmap

If you're not seeing the map, make sure you've followed Step 6 above. Sometimes, it can be an error of the transform (TF)

Make sure the nodes and topics are as listed in the "Nodes and Topics" png file.

- "WARNING control_transfer returned error, index 768, error: No data available, number: 61" can be ignored

5. Other Errors

Here are some general tips:

- Quit the process and run again
- Wait a bit longer (sometimes it takes a while to get both cameras up and running)
- Source the devel in catkin root directory:

```
source devel/setup.bash
```

- Unplug cameras, replug, and try again
- Ignore Errors (some errors will fix itself or don't really impact the running of program)
- Look it up! There's a lot of documentation on errors people have faced and how they have fixed it! Hopefully some of these work :)