6 Pure visual 2D mapping and navigation

depthimage_to_laserscan： http://wiki.ros.org/depthimage_to_laserscan

depthimage_to_laserscan source code: https://github.com/ros-perception/depthimage_to_laserscan

6.1 Introduction

depthimage_to_laserscan takes a depth image(float-encoded meters or preferably uint16-encoded millimeters for OpenNI devices) and generates a 2D laser scan based on the provided parameters. depthimage_to_laserscan uses deferred subscriptions and does not subscribe to images or camera information until a user scans.

The depthimage_to_laserscan function package converts depth images into lidar data, and its mapping and navigation functions are the same as lidar. Note: The scanning range of the depth camera is not 360°.

6.2 Use

Note: The pure depth mapping navigation in this section is not effective and is not recommended.

Note: When building a map, the slower the speed, the better the effect(note that if the rotation speed is slower), the effect will be poor if the speed is too fast.

According to different models, you only need to set the purchased model in [.bashrc], X1(ordinary four-wheel drive) X3(Mike wheel) X3plus(Mike wheel mechanical arm) R2(Ackerman differential) and so on. Section takes X3 as an example

Open the [.bashrc] file

```
sudo vim .bashrc
```

Find the [ROBOT_TYPE] parameter and modify the corresponding model

```
export  ROBOT_TYPE=X3    # ROBOT_TYPE: X1 X3 X3plus R2 X7
```

6.2.1 Mapping

Start command(robot side)

```
roslaunch yahboomcar_nav astrapro_bringup.launch
```

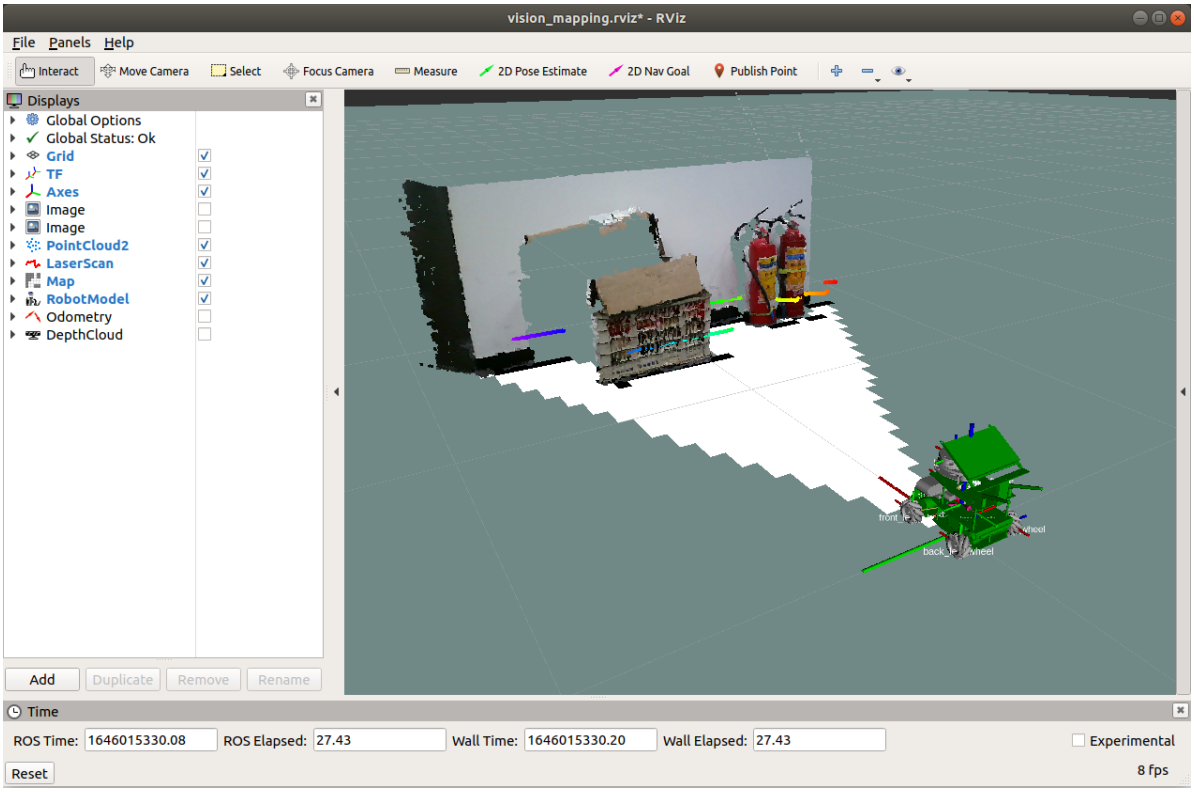
Mapping command(robot side)

```
roslaunch  yahboomcar_nav  yahboomcar_map.launch  use_rviz:=false
map_type:=gmapping
```

- [use_rviz] parameter: whether to enable rviz visualization.
- [map_type] parameter: set the mapping algorithm [gmapping].

Open the visual interface(virtual machine side)

```
roslaunch yahboomcar_nav view_vision_mapping.launch
```



6.2.2 Control the robot

- The keyboard controls the movement of the robot

```
rosrun teleop_twist_keyboard teleop_twist_keyboard.py  # system integration
roslaunch yahboomcar_ctrl yahboom_keyboard.launch      # custom
```

- The handle controls the movement of the robot

There may be some scattered points during the mapping process. If the mapping environment is well closed and regular, the movement is slow, and the scattering phenomenon is much smaller.

6.2.3 Map save

```
# The first way
rosrun map_server map_saver -f ~/yahboomcar_ws/src/yahboomcar_nav/maps/my_map
# second way
bash ~/yahboomcar_ws/src/yahboomcar_nav/maps/map.sh
```

The map will be saved to ~/yahboomcar_ws/src/yahboomcar_nav/maps/ folder, a pgm image, a yaml file.

map.yaml

```
image : map.pgm
resolution : 0.05
origin: [-15.4,-12.2,0.0]
negate : 0
occupied_thresh : 0.65
free_thresh: 0.196
```

Parameter parsing:

- image: The path of the map file, which can be an absolute path or a relative path
- resolution: the resolution of the map, m/pixel
- origin: The 2D pose(x, y, yaw) of the lower left corner of the map, where yaw is rotated counterclockwise(yaw=0 means no rotation). Many parts of the system currently ignore the yaw value.
- negate: whether to reverse the meaning of white/black, free/occupy(the interpretation of the threshold is not affected)
- occupied_thresh: Pixels with an occupancy probability greater than this threshold will be considered fully occupied.
- free_thresh: Pixels with an occupancy probability less than this threshold will be considered completely free.

6.2.4 Navigation

Start command(robot side)

```
roslaunch yahboomcar_nav astrapro_bringup.launch
```

Navigation commands(robot side)

```
roslaunch  yahboomcar_nav  yahboomcar_navigation.launch  use_rviz:=false
map:=house
```

- [use_rviz] parameter: whether to enable rviz visualization.
- [map_type] parameter: set the mapping algorithm [gmapping].

Open the visual interface(virtual machine side)

```
roslaunch yahboomcar_nav view_navigate.launch
```

1. Single point navigation

- Use the [2D Pose Estimate] of the [rviz] tool to set the initial pose until the position of the car in the simulation is consistent with the position of the actual car.
- Click the [2D Nav Goal] of the [rviz] tool, and then select the target point on the map where there are no obstacles, release the mouse to start the navigation, only one target point can be selected, and it will stop when it arrives.

2. Multi-point navigation

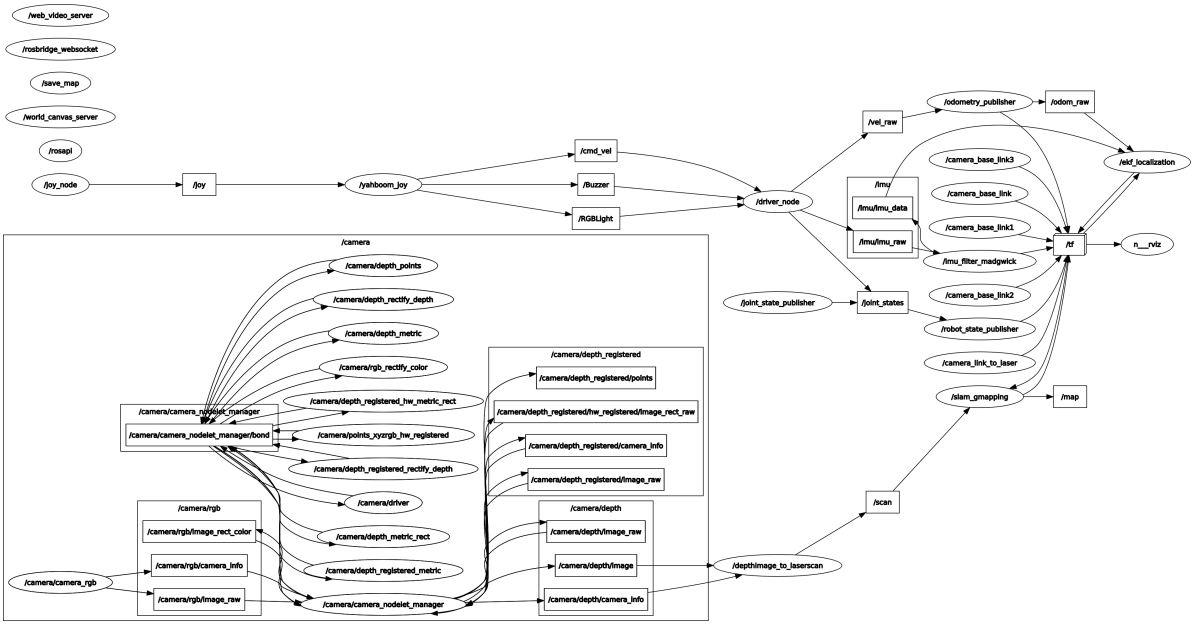
- Same as the first step of single-point navigation, first set the initial pose of the car.
- Click the [Publish Point] of the [rviz] tool, and then select the target point on the map where there are no obstacles, release the mouse to start the navigation, you can click the [Publish Point] again, and then select the point, the robot will point and point cruising in between.
- When using the [2D Pose Estimate] tool of the [rviz] tool to set the initial pose of the car, the multi-point navigation function is automatically canceled.

6.3 Topics and Services

Subscribe to topics	**type**	**describe**
image	sensor_msgs/Image	Enter an image. This can be in floating point or raw uint16 format. For OpenNI devices, uint16 is the native representation, which is more efficient to process. This is usually /camera/depth/image_raw. If your image is distorted, you should remap this subject to image_rect. OpenNI cameras generally have little distortion, so corrections for this application can be skipped.
camera_info	sensor_msgs/CameraInfo	Camera information for the associated image.
Post a topic	**type**	**describe**
scan	sensor_msgs/LaserScan	Output laser scan. and will output a range array containing NAN and +-INF.

Node view

```
rqt_graph
```



6.4 configuration parameters

parameter	type	Defaults	describe
scan_height	int	1 pixel	The number of pixel rows used to generate the laser scan. For each column, the scan will return the minimum value of the vertically centered pixel in the image.
scan_time	double	1/30.0Hz(0.033s)	Scan interval time(seconds). Typically, a rate of 1.0/frame. This value is not easily calculated from consecutive messages, so it is up to the user to set it correctly.
range_min	double	0.45m	The minimum range to return(in meters). Ranges smaller than this value will be output as -Inf.
range_max	double	10.0m	The maximum range returned(in meters). Ranges larger than this value will be output as +Inf.
output_frame_id	str	camera_depth_frame	The frame id of the laser scan. For point clouds from "optical" frames in Z-forward, the value should be set to the corresponding frame in X-forward and Z-up

6.5 TF transformation

Required TF Transform	**describe**
laser-->base_link	The transformation between the lidar coordinate system and the base coordinate system, generally published by robot_state_publisher or static_transform_publisher
base_link-->odom	The transformation between the map coordinate system and the robot's odometer coordinate system to estimate the robot's pose in the map
Published TF Transform	**describe**
map-->odom	The transformation between the map coordinate system and the robot's odometer coordinate system to estimate the robot's pose in the map

View tf tree

```
rosrun rqt_tf_tree rqt_tf_tree
```