

7. Hector mapping algorithm

7. Hector mapping algorithm

7.1. Introduction

7.2. Use

7.2.1. Start

7.2.2. Controlling the robot

7.2.3. Map saving

7.3. Topics and services

7.4. Configuration parameters

7.5. TF transformation

hector_slam: http://wiki.ros.org/hector_slam

hector_slam/Tutorials: http://wiki.ros.org/hector_slam/Tutorials/SettingUpForYourRobot

hector_mapping: http://wiki.ros.org/hector_mapping

map_server: https://wiki.ros.org/map_server

7.1. Introduction

Features: hector_slam does not need to subscribe to odometry/odom messages. It uses the Gauss-Newton method and directly uses lidar to estimate odometry information. However, when the robot is fast, slipping will occur, causing deviations in the mapping effect and placing high demands on sensors. When building a map, set the car's rotation speed as low as possible.

There is no method to use the odom coordinate system, taken from Wiki.

2. Use without odom frame

If you do not require the use of a odom frame (for example because your platform does not provide any usable odometry) you can directly publish a transformation from map to base_link:

```
<param name="pub_map_odom_transform" value="true"/>
<param name="map_frame" value="map" />
<param name="base_frame" value="base_frame" />
<param name="odom_frame" value="base_frame" />
```

7.2. Use

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

According to different models, you only need to set the purchased model in [.bashrc], X1 (normal four-wheel drive) X3 (Mailun) Take X3 as an example

```
#Raspberry Pi 5 master needs to enter docker first, please perform this step
#If running the script into docker fails, please refer to ROS/07, Docker tutorial
~/run_docker.sh
```

Open the [.bashrc] file

```
sudo vim .bashrc
```

Find the [ROBOT_TYPE] parameters and modify the corresponding car model

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

7.2.1. Start

Start the command (robot side). For the convenience of operation, this section takes [mono + laser + yahboomcar] as an example.

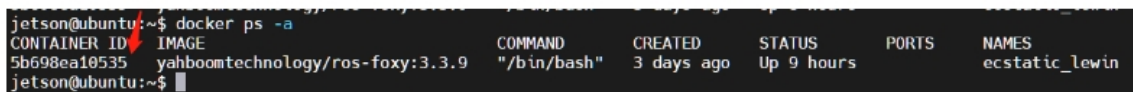
```
roslaunch yahboomcar_nav laser_bringup.launch # laser + yahboomcar
roslaunch yahboomcar_nav laser_usb_bringup.launch # mono + laser + yahboomcar
roslaunch yahboomcar_nav laser_astrapro_bringup.launch # Astra + laser + yahboomcar
```

Mapping command (robot side)

<PI5 needs to open another terminal to enter the same docker container

1. In the above steps, a docker container has been opened. You can open another terminal on the host (car) to view:

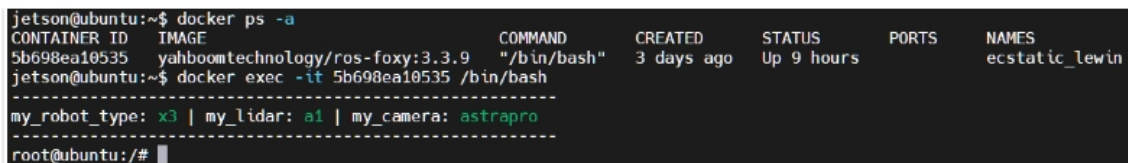
```
docker ps -a
```



CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
5b698ea10535	yahboomtechnology/ros-foxy:3.3.9	"/bin/bash"	3 days ago	Up 9 hours		ecstatic_lewin

2. Now enter the docker container in the newly opened terminal:

```
docker exec -it 5b698ea10535 /bin/bash
```



```
jetson@ubuntu:~$ docker ps -a
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS   NAMES
5b698ea10535   yahboomtechnology/ros-foxy:3.3.9    "/bin/bash"            3 days ago    Up 9 hours                   ecstatic_lewin
jetson@ubuntu:~$ docker exec -it 5b698ea10535 /bin/bash
-----
my_robot_type: x3 | my_lidar: a1 | my_camera: astrapro
-----
root@ubuntu:/#
```

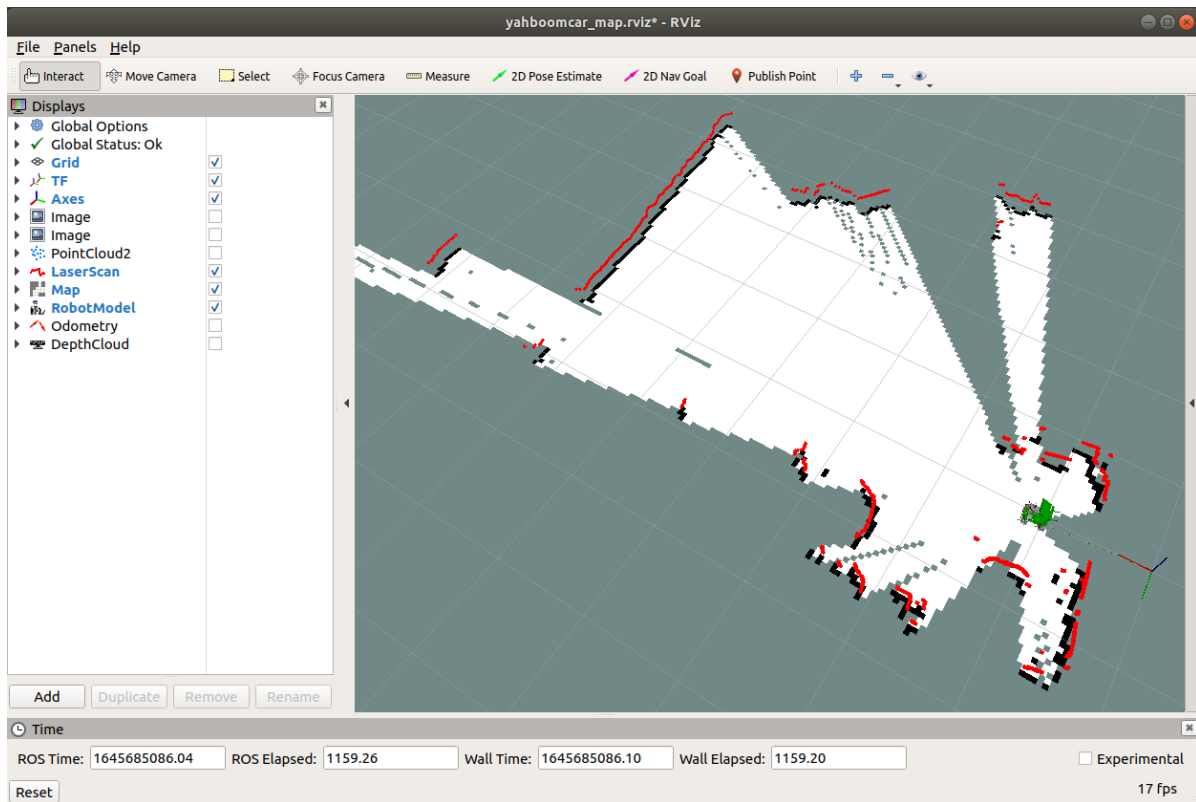
After successfully entering the container, you can open countless terminals to enter the container.

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

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

Turn on the visual interface (virtual machine side)

```
roslaunch yahboomcar_nav view_map.launch
```



The gap at the back of the robot is due to the obstruction caused by the installation position of the display screen, so a certain range of radar data is blocked. The shielding range can be adjusted, or it can not be blocked according to the actual situation. For specific operations, see [01. Radar Basic Course].

7.2.2. Controlling the robot

- Keyboard controls robot movement

```
roslaunch teleop_twist_keyboard teleop_twist_keyboard.py # System integration
roslaunch yahboomcar_ctrl yahboom_keyboard.launch # Custom
```

- Control the robot movement with the handle

Make the robot cover the area to be mapped and the map should be as closed as possible.

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

7.2.3. Map saving

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

The map will be saved to the ~/yahboomcar_ws/src/yahboomcar_nav/maps/ folder, a pgm image and 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 analysis:

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

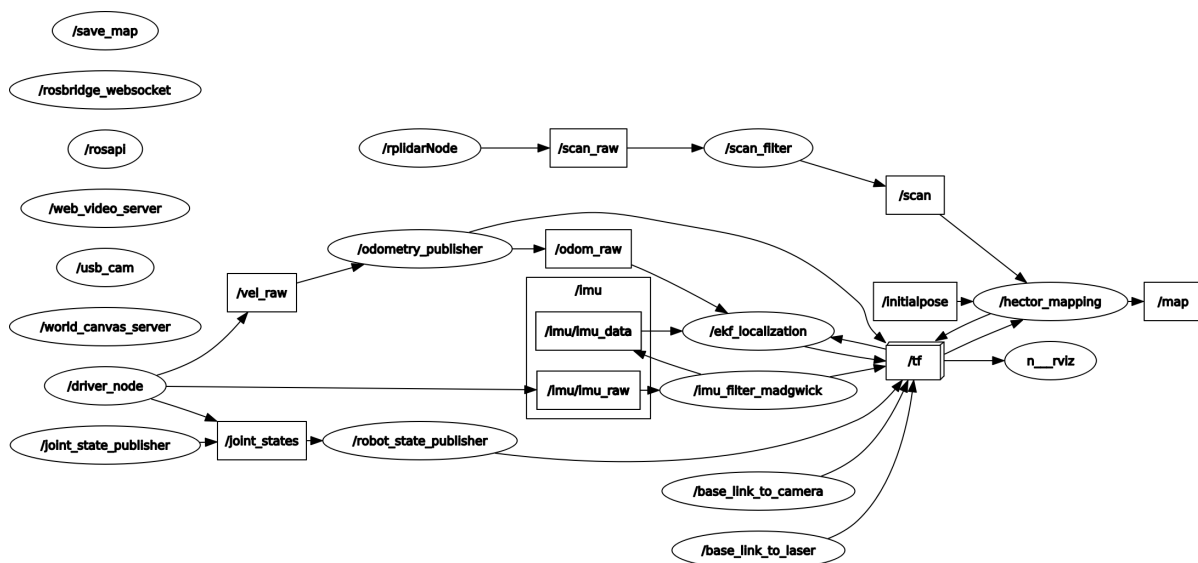
7.3. Topics and services

Topic Subscription	Type	Description
scan	sensor_msgs/LaserScan	Depth data of lidar scan
syscommand	std_msgs/String	System command. If the string equals "reset", the map and robot pose are reset to the initial state
Topic Post	Type	Description
map_metadata	nav_msgs/MapMetaData	Publish map Metadata
map	nav_msgs/OccupancyGrid	Publish map raster data
slam_out_pose	geometry_msgs/PoseStamped	Covariance-free robot pose estimation
poseupdate	geometry_msgs/PoseWithCovarianceStamped	Robot pose estimation with Gaussian uncertainty estimation

Topic Subscription	Type	Description
Service	Type	Description
dynamic_map	nav_msgs/GetMap	Get map data
reset_map	std_srvs/Trigger	Call this service to reset the map, and hector will create a brand new map from scratch. Note that this will not restart the robot's pose, it will restart from the last recorded pose.
pause_mapping	std_srvs/SetBool	Call this service to stop/start processing laser scans.
restart_mapping_with_new_pose	hector_mapping/ResetMapping	Call this service to reset the map, robot's pose, and resume the map (if paused)

Node view

rqt_graph



7.4. Configuration parameters

Parameters	Type	Default value	Description
~base_frame	String	"base_link"	Robot base coordinate system, used for positioning and laser scanning data transformation
~map_frame	String	"map"	The coordinate system of the map
~odom_frame	string	"odom"	Odometer coordinate system (essentially the coordinate system pointed to by map)
~map_resolution	Double	0.025(m)	Map resolution, edge length of grid cells
~map_size	Int	1024	The size of the map
~map_start_x	double	0.5	/The position of the origin of map [0.0, 1.0] on the x-axis relative to the grid map
~map_start_y	double	0.5	/The position of the origin of /map [0.0, 1.0] on the y-axis relative to the grid map
~map_update_distance_thresh	double	0.4(m)	The threshold for map update, which is calculated from the first update on the map until the straight distance reaches this parameter value and is updated again
~map_update_angle_thresh	double	0.9(rad)	The threshold for map update, starting from the first update on the map and updating again after the rotation reaches this parameter value

Parameters	Type	Default value	Description
~map_pub_period	double	2.0	Map release cycle
~map_multi_res_levels	int	3	Map multi-resolution grid levels
~update_factor_free	double	0.4	Used to update the map of free cells, the range is [0.0, 1.0]
~update_factor_occupied	double	0.9	Used to update the map of occupied units, the range is [0.0, 1.0]
~laser_min_dist	double	0.4(m)	The minimum distance of laser scanning points. Scanning points smaller than this value will be ignored
~laser_max_dist	double	30.0(m)	The maximum distance of laser scanning points. Scanning points smaller than this value will be ignored
~laser_z_min_value	double	-1.0(m)	Minimum height relative to the lidar, scan points below this value will be ignored
~laser_z_max_value	double	1.0(m)	The maximum height relative to the lidar, scan points above this value will be ignored
~pub_map_odom_transform	bool	true	Whether to publish the coordinate transformation between map and odom
~output_timing	bool	false	Process the output timing information of each laser scan through ROS_INFO
~scan_subscribe_queue_size	int	5	Queue size for scan subscribers

Parameters	Type	Default value	Description
~pub_map_scanmatch_transform	bool	true	Whether to publish the coordinate transformation between scanmatcher and map
~tf_map_scanmatch_transform_frame_name	String	"scanmatcher_frame"	The coordinate system name of scanmatcher

7.5, TF transformation

Required TF transformation	Description
laser-->base_link	Usually a fixed value, the transformation between the lidar coordinate system and the base coordinate system, generally published by robot_state_publisher or static_transform_publisher
Released TF Transform	Description
map-->odom	The current estimate of the robot pose within the map frame (only provided if parameter "pub_map_odom_transform" is true).

View tf tree

```
roslaunch rqt_tf_tree rqt_tf_tree
```

