

Communication with Marvelmind devices using ROS (Robot Operating System).

Version 2022.11.20

Note: for integration with ROS 2 see [this separate document](#).

Marvelmind supplies ROS package **marvelmind_nav**, which is able to communicate with mobile beacon or modem and provide received location and other data. We have released and tested latest version of the package for ROS Melodic (under Ubuntu 18.04) and ROS Noetic (under Ubuntu 20.04). The ROS package is also available in source repository by link:

https://bitbucket.org/marvelmind_robotics/ros_marvelmind_package

To install the package in the ROS system from source, at first create catkin workspace as described here:

http://wiki.ros.org/catkin/Tutorials/create_a_workspace

Then create the directory for the package by executing commands in the terminal:

```
$ cd ~/catkin_ws/src
$ mkdir marvelmind_nav
```

Then copy downloaded sources from the repository into the created directory.

Before running the software from the package, you should execute following command from the 'catkin_ws' directory:

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

After this, you can build and install Marvelmind ROS package: execute from 'catkin_ws' directory:

```
$ catkin_make
$ catkin_make install
```

Now we are ready to run the software, but before it we need to prepare the marvelmind system.

Use another PC with dashboard software to build map as described in the operating manual:

http://marvelmind.com/pics/marvelmind_navigation_system_manual.pdf

Then connect the mobile beacon (hedgehog) by the USB cable to the USB port of your ROS machine.

Execute command in terminal to find the virtual serial port used by the hedgehog:

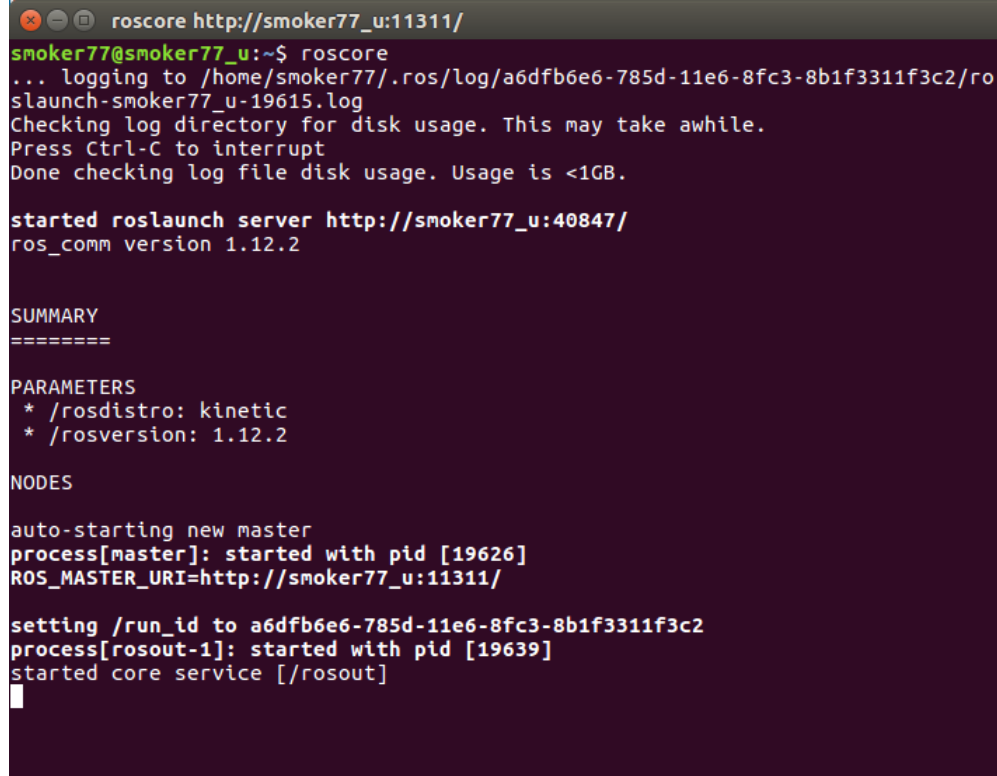
```
$ ls /dev/ttyACM*
```

In most cases, the hedgehog connects to "/dev/ttyACM0", this port is used by default in the Marvelmind ROS software. If no ports found by this command, try another one:

```
$ ls /dev/ttyUSB*
```

Before running Marvelmind ROS software, run the ROS server in separate terminal:

```
$ roscore
```



```
roscore http://smoker77_u:11311/
smoker77@smoker77_u:~$ roscore
... logging to /home/smoker77/.ros/log/a6dfb6e6-785d-11e6-8fc3-8b1f3311f3c2/ro
slaunch-smoker77_u-19615.log
Checking log directory for disk usage. This may take awhile.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://smoker77_u:40847/
ros_comm version 1.12.2

SUMMARY
=====

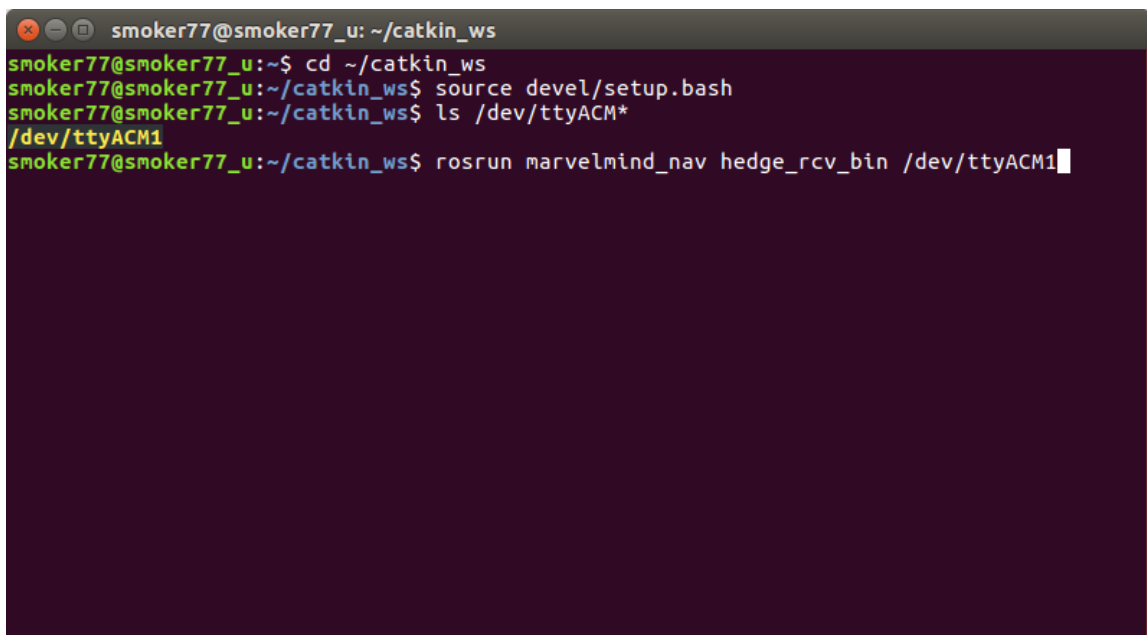
PARAMETERS
* /rostdistro: kinetic
* /rosverison: 1.12.2

NODES

auto-starting new master
process[rosmaster]: started with pid [19626]
ROS_MASTER_URI=http://smoker77_u:11311/

setting /run_id to a6dfb6e6-785d-11e6-8fc3-8b1f3311f3c2
process[rosout-1]: started with pid [19639]
started core service [/rosout]
```

Then run the node '**hedge_rcv_bin**' for receiving data from hedgehog as shown on following screenshot. Note the parameter of the running program is '/dev/ttyACM1', the name of virtual serial port detected by previous command. If the port is '/dev/ttyACM0', this parameter can be skipped.



```
smoker77@smoker77_u: ~/catkin_ws
smoker77@smoker77_u:~$ cd ~/catkin_ws
smoker77@smoker77_u:~/catkin_ws$ source devel/setup.bash
smoker77@smoker77_u:~/catkin_ws$ ls /dev/ttyACM*
/dev/ttyACM1
smoker77@smoker77_u:~/catkin_ws$ rosrn marvelmind_nav hedge_rcv_bin /dev/ttyACM1
```

In second command line parameter you can specify baudrate of the serial port, for example: **hedge_rcv_bin /dev/ttyACM1 115200**. For connection via USB it doesn't mean anything but with connection via UART it should correspond to mobile beacon (or modem) baudrate setting.

You can get message like 'unable to open serial connection' even if the serial port is present. This may mean you have no permissions to access this port. You can get all permissions by command **'sudo chmod 0777 /dev/ttyACM0'**. But you will lose the permissions after next reboot. For permanent permissions you can add user to **dialout** group as described here: <https://askubuntu.com/questions/58119/changing-permissions-on-serial-port>

If the node successfully receives data from hedgehog, it outputs the location data to the terminal as shown on screenshot:

```

smoker77@smoker77_u: ~/catkin_ws
[ INFO] [1473625993.222961259]: 218663, 60, X=0.770 Y= 0.440 Z=0.800 flags=2
[ INFO] [1473625993.282959329]: 218722, 59, X=0.770 Y= 0.450 Z=0.800 flags=2
[ INFO] [1473625993.342963385]: 218783, 61, X=0.780 Y= 0.420 Z=0.800 flags=2
[ INFO] [1473625993.402958895]: 218842, 59, X=0.770 Y= 0.440 Z=0.800 flags=2
[ INFO] [1473625993.462963279]: 218902, 60, X=0.770 Y= 0.440 Z=0.800 flags=2
[ INFO] [1473625993.522961667]: 218961, 59, X=0.770 Y= 0.450 Z=0.800 flags=2
[ INFO] [1473625993.582958058]: 219021, 60, X=0.770 Y= 0.440 Z=0.800 flags=2
[ INFO] [1473625993.642960960]: 219081, 60, X=0.770 Y= 0.440 Z=0.800 flags=2
[ INFO] [1473625993.702960428]: 219140, 59, X=0.770 Y= 0.450 Z=0.800 flags=2
[ INFO] [1473625993.762960499]: 219200, 60, X=0.770 Y= 0.440 Z=0.800 flags=2
[ INFO] [1473625993.822961710]: 219260, 60, X=0.770 Y= 0.440 Z=0.800 flags=2
[ INFO] [1473625993.882954247]: 219320, 60, X=0.770 Y= 0.440 Z=0.800 flags=2
[ INFO] [1473625993.942941914]: 219379, 59, X=0.770 Y= 0.440 Z=0.800 flags=2
[ INFO] [1473625994.002959652]: 219440, 61, X=0.770 Y= 0.440 Z=0.800 flags=2
[ INFO] [1473625994.062957609]: 219499, 59, X=0.770 Y= 0.450 Z=0.800 flags=2
[ INFO] [1473625994.122957416]: 219559, 60, X=0.770 Y= 0.440 Z=0.800 flags=2
[ INFO] [1473625994.182951399]: 219619, 60, X=0.770 Y= 0.450 Z=0.800 flags=2
[ INFO] [1473625994.242958338]: 219678, 59, X=0.770 Y= 0.440 Z=0.800 flags=2
[ INFO] [1473625994.302958989]: 219738, 60, X=0.770 Y= 0.440 Z=0.800 flags=2
[ INFO] [1473625994.342958514]: 219797, 59, X=0.770 Y= 0.440 Z=0.800 flags=2
[ INFO] [1473625994.422957060]: 219858, 61, X=0.770 Y= 0.440 Z=0.800 flags=2
[ INFO] [1473625994.462990439]: 219917, 59, X=0.770 Y= 0.440 Z=0.800 flags=2
[ INFO] [1473625994.522990435]: 219976, 59, X=0.770 Y= 0.440 Z=0.800 flags=2

```

First value in square brackets is a ROS timestamp, then hedgehog timestamp in milliseconds, time (in milliseconds) between position samples, coordinates X,Y,Z in meters, and byte of flags.

The node 'hedge_rcv_bin' works as ROS publisher, it sends the message with location data to the topics named 'hedge_pos', 'hedge_pos_a' and 'hedge_pos_ang'.

'hedge_pos_ang' is most new version of the topic; it includes address of mobile beacon and orientation angle of paired beacons.

Following table lists all topics and data available via these topics:

Topic	Message field	Type	Description
hedge_pos_ang	address	uint8	Address of mobile beacon
	timestamp_ms	uint32	Timestamp of location, milliseconds
	x_m	float64	X coordinate, meters
	y_m	float64	Y coordinate, meters
	z_m	float64	Z coordinate, meters

	flags	uint8	flags of location
	angle	float64	Orientation angle of paired beacons, degrees
beacon_pos_a			
	address	uint8	Address of stationary beacon
	x_m	float64	X coordinate, meters
	y_m	float64	Y coordinate, meters
	z_m	float64	Z coordinate, meters
beacon_distance			
	address_hedge	uint8	Address of mobile beacon
	address_beacon	uint8	Address of stationary beacon
	distance_m	float64	Raw distance from mobile to stationary beacon, meters
hedge_imu_fusion			
	timestamp_ms	int64	Timestamp of IMU fusion data, milliseconds
	x_m	float64	(X,Y,Z) coordinates of mobile beacon by IMU fusion. meters.
	y_m	float64	
	z_m	float64	
	qw	float64	Orientation quaternion of mobile beacon (qw,qx,qy,qz). Normalized ($qw^2+qx^2+qy^2+qz^2=1$)
	qx	float64	
	qy	float64	
	qz	float64	
	vx	float64	(vx, vy, vz) – speed vector of mobile beacon calculated by IMU fusion, meters/s
	vy	float64	
	vz	float64	
	ax	float64	(ax, ay, az) – acceleration of mobile beacon meters/s ²
	ay	float64	
	az	float64	
hedge_imu_raw			
	timestamp_ms	int64	Timestamp of raw IMU data, milliseconds
	acc_x	int16	(acc_x, acc_y, acc_z) – raw accelerometer data, 1 mg/LSB
	acc_y	int16	
	acc_z	int16	
	gyro_x	int16	(gyro_x, gyro_y, gyro_z) – raw gyroscope data, 0.0175 dps/LSB
	gyro_y	int16	
	gyro_z	int16	
	compass_x	int16	(compass_x, compass_y, compass_z) – raw compass data (only for HW4.9 beacons). X,Y: 1100 LSB/Gauss Z: 980 LSB/Gauss
	compass_y	int16	
	compass_z	int16	
hedge_quality			
	address	uint8	Address of the mobile beacon beacon
	quality_percents	uint8	Quality of location, percents
hedge_telemetry			
	battery_voltage	float64	Battery voltage of the mobile beacon, volts
	rsi_dbm	int8	RSSI (radio signal strength), dBm

marvelmind_waypoint	total_items	uint8	Total number of waypoint program items (N)
	item_index	uint8	Index of this waypoint item (0...N-1)
	movement_type	uint8	Type of action (6 = move to specified point)
	param1	int16	Parameter 1 (depends from movement_type) X coordinate of waypoint, cm if type= 6
	param2	int16	Parameter 2 (depends from movement_type) Y coordinate of waypoint, cm if type= 6
	param3	int16	Parameter 3 (depends from movement_type) Z coordinate of waypoint, cm if type= 6

The package also contains another node '**subscriber_test**', which is working as ROS subscriber and receiving data from all the topics. This node can be used for test purposes and as basis for user software.

Run the '**subscriber_test**' node in separate terminal as shown on following screenshot:

```

smoker77@smoker77_u: ~/catkin_ws
smoker77@smoker77_u:~$ cd ~/catkin_ws
smoker77@smoker77_u:~/catkin_ws$ source devel/setup.bash
smoker77@smoker77_u:~/catkin_ws$ rosrn marvelmind_nav subscriber_test

```

The running '**subscriber_test**' node outputs to the terminal received location data from the topic '/hedge_pos' as shown on the next screenshot:

```

smoker77@smoker77_u: ~/catkin_ws
[ INFO] [1473627322.843188052]: Hedgehog data: 1548426, X=0.720 Y= 0.570 Z=0.800 flags=2
[ INFO] [1473627322.903167517]: Hedgehog data: 1548486, X=0.700 Y= 0.580 Z=0.800 flags=2
[ INFO] [1473627322.963161788]: Hedgehog data: 1548546, X=0.710 Y= 0.590 Z=0.800 flags=2
[ INFO] [1473627323.023166165]: Hedgehog data: 1548606, X=0.710 Y= 0.590 Z=0.800 flags=2
[ INFO] [1473627323.083168992]: Hedgehog data: 1548665, X=0.720 Y= 0.570 Z=0.800 flags=2
[ INFO] [1473627323.143147253]: Hedgehog data: 1548725, X=0.700 Y= 0.580 Z=0.800 flags=2
[ INFO] [1473627323.203169379]: Hedgehog data: 1548785, X=0.720 Y= 0.570 Z=0.800 flags=2
[ INFO] [1473627323.263170405]: Hedgehog data: 1548844, X=0.720 Y= 0.570 Z=0.800 flags=2
[ INFO] [1473627323.323166049]: Hedgehog data: 1548904, X=0.710 Y= 0.570 Z=0.800 flags=2
[ INFO] [1473627323.383165544]: Hedgehog data: 1548964, X=0.720 Y= 0.580 Z=0.800 flags=2
[ INFO] [1473627323.443162007]: Hedgehog data: 1549024, X=0.710 Y= 0.590 Z=0.800 flags=2
[ INFO] [1473627323.503178015]: Hedgehog data: 1549084, X=0.720 Y= 0.570 Z=0.800 flags=2
[ INFO] [1473627323.563168653]: Hedgehog data: 1549143, X=0.710 Y= 0.580 Z=0.800 flags=2
[ INFO] [1473627323.623166626]: Hedgehog data: 1549204, X=0.710 Y= 0.590 Z=0.800 flags=2
[ INFO] [1473627323.683168008]: Hedgehog data: 1549263, X=0.710 Y= 0.590 Z=0.800 flags=2
[ INFO] [1473627323.743163334]: Hedgehog data: 1549322, X=0.720 Y= 0.580 Z=0.800 flags=2
[ INFO] [1473627323.803167859]: Hedgehog data: 1549382, X=0.720 Y= 0.580 Z=0.800 flags=2
[ INFO] [1473627323.863170595]: Hedgehog data: 1549442, X=0.710 Y= 0.590 Z=0.800 flags=2
[ INFO] [1473627323.923186361]: Hedgehog data: 1549502, X=0.090 Y= 0.920 Z=0.800 flags=2
[ INFO] [1473627323.983184846]: Hedgehog data: 1549561, X=0.710 Y= 0.580 Z=0.800 flags=2
[ INFO] [1473627324.043182927]: Hedgehog data: 1549621, X=0.710 Y= 0.590 Z=0.800 flags=2
[ INFO] [1473627324.103198350]: Hedgehog data: 1549681, X=0.700 Y= 0.600 Z=0.800 flags=2

```

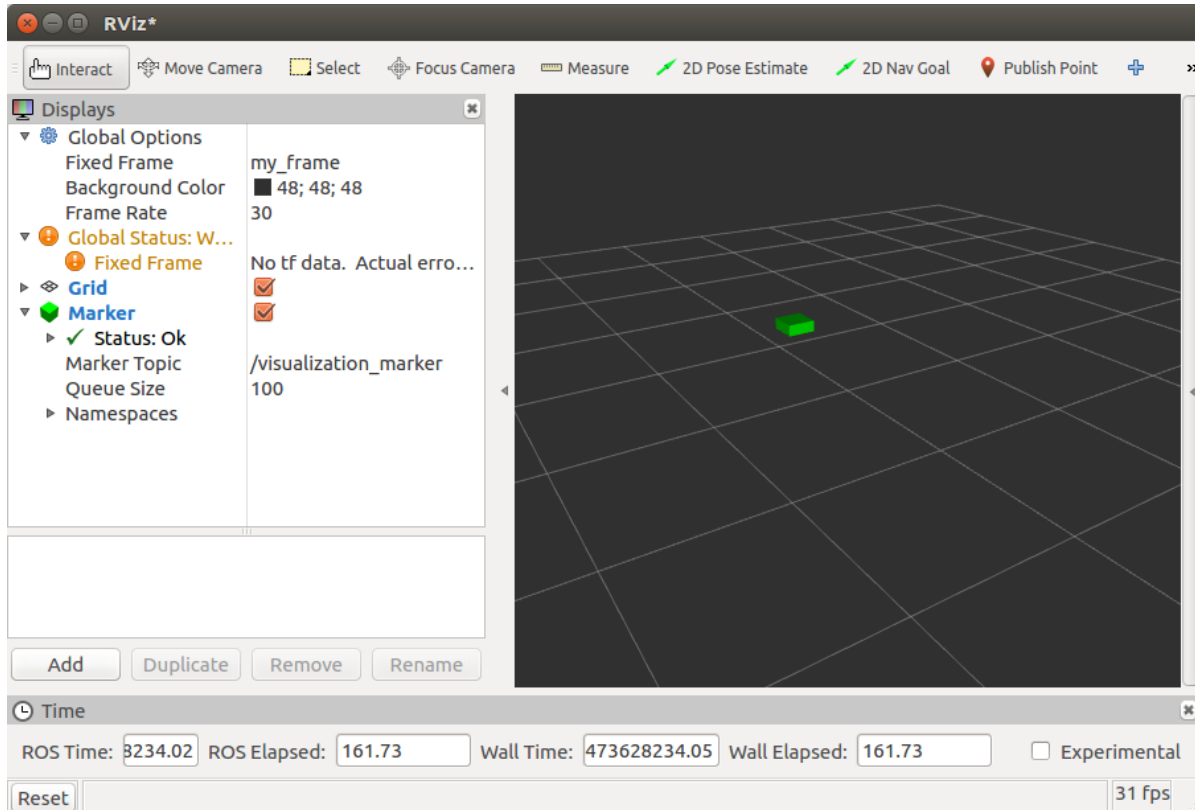
In addition this node works as publisher and sends data to topic "**visualization_marker**". This allows to view the position in the standard ROS software '**rviz**'.

Run the '**rviz**' in separate terminal as shown on screenshot:

```
smoker77@smoker77_u: ~  
smoker77@smoker77_u:~$ rosrn rviz rviz  
[ INFO] [1473628071.154944507]: rviz version 1.12.1  
[ INFO] [1473628071.154989001]: compiled against Qt version 5.5.1  
[ INFO] [1473628071.155001615]: compiled against OGRE version 1.9.0 (Ghadamon)  
[ INFO] [1473628071.662983695]: Stereo is NOT SUPPORTED  
[ INFO] [1473628071.663080950]: OpenGL version: 3 (GLSL 1.3).
```

The GUI window, shown on the next screenshot, should appear.

To see the hedgehog, make sure the '**Fixed frame**' parameter has value '**my_frame**', and the marker '**visualization marker**' is connected.



The next screenshot shows the dashboard window on another computer with system, used for the described above testing of ROS.

Dashboard - robots management V4.74 ultimate

File Language View Firmware Substrate Help

Marvelmind robotics

clear map
Dots timeout: 5
freeze screen

HIDE	13	14	19
13		2.95	
14	2.95		
19	2.53	1.40	

SCAN

modem V5.20.24 3/3

Submap 0 +

unfreeze map

save map load map

device 5-12

Connected: COM4 X: 4.97, Y:-1.18 Rate: 16 Hz 547 total, 0 failed (0%)

read all write all

Starting beacon trilateration (0..255)	0
Starting set of beacons	14: 13:0:0
Location update rate	16 Hz
Maximum speed, m/s (0.1..60.0)	5.0
Movement filtering	disabled
Supply voltage, V	5.05
Time from reset, h:m:s	07:28:52 R
Temperature of air, °C (-20..60)	23
RSSI, dBm	-47
Camera frequency, MHz	433.400
Device address (0..99)	1
Channel	0
Parameters of radio	(+) expand
Interfaces	(+) expand
Limitation distances	auto
Map gluing	disabled
Gluing ready	no
Submap X shift, m (-100.00..100.00)	0.00
Submap Y shift, m (-100.00..100.00)	0.00
Submap rotation, degrees (-360..360)	0

reset sleep wake up
CTRL deep sleep default