Université Paul Sabatier

Master Intelligence Artificielle et
Reconnaissance des Formes
Master Robotique : Décision et Commande

User Manual - Navigation Between Markers

Mobile Robot Navigation

Authors:
Thibaut Aghnatios
Marine Bouchet
Bruno Dato
Tristan Klempka
Thibault Lagoute

Tutors:
Frédéric Lerasle
Michaël Lauer
Michel Taix

30 March 2017

Document tracking

Name	Major Version	Minor Version	Creation Date	Last version
User Manual - Navigation Between Markers	A	5	30/03/2017	5/04/2017

Document authors

Redaction	Integration	Review	Validation
Bruno Dato Thibaut Aghnatios Marine Bouchet	Bruno Dato Marine Bouchet	Bruno Dato	??

Document validation

Validation	Name	Date	Visa

Broadcast list

User Manual - Navigation Between Markers is distributed to all clients and external stakeholders.

Review history

Version	Additions or modifications	Author	Date
A.0	Document creation	Bruno Dato	30/01/2017
A.1	Sections 1 and 2	Bruno Dato	1/04/2017
A.2	Section 1.4.5	Thibaut Aghnatios	02/04/2017
A.3	Section 1.4.3	Thibaut Aghnatios	04/04/2017
A.4	Sections 1.4 and 2.3	Marine Bouchet	04/04/2017
A.5	Review	Bruno Dato	05/04/2017

Contents

1 Prerequisites

1.1 Equipment

- TurtleBot 2
- AR markers

1.2 Software

To be able to use any TurtleBot 2 with all the basic features, you need to complete the following tutorials :

- Turtlebot Installation
- PC Installation
- Network Configuration

You also need the following software and additional package :

- GIT [Installation]
- Package *ar_track_alvar*

```
> sudo apt-get install ros-indigo-ar-track-alvar
```

1.3 Workspace

1.3.1 Build workspace

You need a ROS workspace (catkin workspace) to build our project before executing it. If you are running the ball search on the TurtleBot PC you have to create another workspace on the TurtleBot PC. In the case your are running it on a remote PC, you have to create the workspace on this PC.

Place you where you want to build the workspace and execute the following commands :

```
> mkdir -p /catkin_ws/src
> cd /catkin_ws/src
> catkin_init_workspace
> cd ..
> catkin_make
```

Then, in $\sim$ /*.bashrc*, add the following lines (it is normal if some of them are already there) :

```
#Initialisation Turtlebot kinect
export TURTLEBOT_3D_SENSOR=kinect
#ROS Version
source /opt/ros/indigo/setup.bash
source <YOUR_PATH>/catkin_ws/devel/setup.bash
#Select corresponding TurtleBot on your network
export ROS_MASTER_URI=http://<IP_OF_TURTLEBOT>:11311
```

1.3.2 Download package

Now, you need to download the package containing the source code. Place you in your workspace (catkin_ws), and execute the following commands :

```
> cd src
> git clone https://github.com/Projet-Navigation-UPS/TurtleBot-pkgs
```

1.3.3 Build executables

Now that you have downloaded the source code, you just need to compile to build the executable files. Put you in your workspace (catkin_ws) and run the command :

```
> catkin_make
```

Several red lines must appear in the compilation description, it means that the executables we need have been created.

1.4 Map and markers configuration

1.4.1 Environment map

You need yo create the map of the environment within you navigate if it is not already available in the folder **/catkin_ws/src/TurtleBot-pkgs/turtlebot_proj_nav/map**. To create the map, we use the **turtlebot_navigation** package which provides a SLAM mode (Tutorial link).We recommend to put the robot in a parallel orientation with a wall to facilitate the future definitions of markers orientations. After turning on the TurtleBot and its laptop, execute the following commands the TurtleBot laptop :

```
> roslaunch turtlebot_bringup minimal.launch
> roslaunch turtlebot_navigation gmapping_demo.launch
```

Then, on a remote computer, execute the visualization of the SLAM :

```
> roslaunch turtlebot_rviz_launchers view_navigation.launch
```

To make the robot move thanks to your keyboard and explore the environment, execute :

```
> roslaunch turtlebot_teleop keyboard_teleop.launch --screen
```

Once the map is satisfying for the navigation, on another terminal you have to save it :

```
> rosrun map_server map_saver -f <PATH>/catkin_ws/src/TurtleBot-pkgs/turtlebot
_proj_nav/map/my_map
```

1.4.2 Markers disposition

Within our project we have used 16×16 markers placed 3 m away minimum from each other. We put their centers 31cm above the ground so the kinect-marker is as parallel as possible to the ground.

1.4.3 Graph of the markers

To choose towards which marker (AR marker) to move when one has been detected and is still too far from the final goal to go directly to it, we use a graph representing all the markers defined in an XML format file. Each node of the graph has different properties:

- Id: number corresponding to the AR marker;
- Label: name of the node, each node corresponds to an AR marker;
- PositionX: coordinate along the x-axis of the known environment map;
- PositionY: coordinate along the y-axis of the known environment map;
- Orientation: angle between the normal of the AR marker and the x-axis of the map (between 0 and 2π).

The numbers allow the markers to be located in the graph. The coordinates (x, y) of the markers make it possible to know the positions in the map. The orientation allow to command the robot to move in front and at a certain distance from the marker to avoid walls. For each marker that you will use in your navigation, you need to define a node with its properties and all its links with the others. The costs for then links have to be integers, you can use the distances between the marker in cm for example.

You can use the visualizer RVIZ to get the positions and orientations of the markers in the map by display geometry poses in the map.

1.4.4 Markers static transforms publisher

In this section, you will need to modify *navigation.launch*. AR markers transforms need to be published. These transforms represent where the markers are in our scene in the system. Use one *static_transform_publisher* for each marker. In the *args* field put the position (x,y,z), the orientation (i,j,k,w), the transform parent (always */map*) and the name of the marker transform (*/marker_X*) of the marker.
More info can be found at :
wiki.ros.org/tf#static_transform_publisher

Be careful, for each marker, the related mark must have its z-axis on the normal of the marker and its x-axis pointing upwards.

1.4.5 Visibility map

First, to generate the visibility map, it is necessary to previously have a map of the environment. This map is created virtually or by using the mapping available on the Turtlebot, used in the *map_server*.

The node *visib_pgmwriter_node.cpp* must not be modified, all configurations are done directly in the *visib_init.cpp* file. Indeed, the node launchs the function $Writing\_map\_visib()$ that creates in a PGM file (Plain PGM: $http://netpbm.sourceforge.net/doc/pgm.html\#plainpgm$) all markers defined in the *graph.xml* located in the *rsc* folder. So for a given map size and for the configurations performed correctly in *visib_init.cpp*, just change the position and orientation of our markers in the *graph.xml* so that the new visibility map is automatically generated by running our *visib_pgmwriter_node.cpp* node again.

```
> rosrun turtlebot_proj_nav visib_pgmwriter_node.cpp
```

For a different map scale, see the developer manual.

2 Navigation Between Markers

First, turn on the TurtleBot (there is a switch button on the side of the robot base). Then, turn on the TurtleBot PC. We will now launch all the ROS nodes that we need to run our application.

2.1 On the TurtleBot PC

2.1.1 Basic features

If you are using the TurtleBot PC, open two terminals and chronologically execute the following commands to activate the minimal features and the vision features, one on each terminal :

```
> roslaunch turtlebot_bringup minimal.launch
```

```
> roslaunch turtlebot_bringup 3dsensor.launch
```

2.1.2 Navigation

To launch all the navigation node, execute the following command :

```
> roslaunch turtlebot_proj_nav navigation.launch
```

You can now see on the visualizer RVIZ the map, the TurtleBot and all the markers.

2.2 On a remote PC

2.2.1 Basic features

To execute the navigation from a remote PC, fist you have to ssh to the TurtleBot PC to launch the minimal and vision features. Open a first terminal and write the following commands :

```
> ssh turtlebot@<TURTLEBOTP_IP>
```

```
> roslaunch turtlebot_bringup minimal.launch
```

Then, in a second terminal :

```
> ssh turtlebot@<TURTLEBOTP_IP>
```

```
> roslaunch turtlebot_bringup 3dsensor.launch
```

2.2.2 Navigation

To run the navigation nodes on a remote laptop execute the following commands :

```
> roslaunch turtlebot_proj_launch navigation.launch
```

You can now see on the visualizer RVIZ the map, the TurtleBot and all the markers.

2.3 Behaviour of the navigation

The navigation takes place between an initial position, the real position of the robot, or in the visibility zone of a marker, and a final position:

Scenario	A	B
Start	Absolute initial position	Unknown initial position but in a visibility zone of a marker
Arrival	Final Position	Final Position

These scenarii are (B bis an extension of A):

B If no input position
    The robot performs a marker search in its visual field
    As long as no marker in view
        The robot turns on itself
        It performs another marker search in its visual field

A As long as the goal is further than the next marker in the goal direction
    The robot moves to the most visibility zone of the marker
    ... close in the direction of the goal
    The robot performs a marker search in its visual field
    As long as there is no marker in view
        The robot turns on itself
        It performs another marker search in its visual field
    The robot moves to the goal