Autonoship Simulation User Guide

1. Introduction

The development of autonoship simulation package aims at building a testing environment for ship collision avoidance and autonoship system. The simulation can support a controllable own ship with sensors including radar, camera, GPS, and IMU, and multiple controllable target ships.

This project includes the autonoship gazebo simulation package and usable ship dynamics and radar plugins. An installation guidance is contained.

2. System Requirements

This simulation is built on Ubuntu 16.04 (http://releases.ubuntu.com/16.04/) and ROS kinetic (http://wiki.ros.org/kinetic/Installation/Ubuntu). For installation and usage, please follow the instructions in the file *autonoship_guidance.html*

3. Simulation environment

3.1. Existing scenarios

All launch files are in the directory: <workspace_name>/src/autonoship_simulation/launch. There are 17 different testing scenarios in the folder launch named as "autonoship_scenario**.launch". You can run these testing scenarios by declaring it in the launch process by:

```
roslaunch autonoship_simulation autonoship_gazebo.launch scenario:=scenario8
```

3.2. Adjustment of time flow

The time flow can be adjusted in the file:

<workspace_name>/src/autonoship_simulation/worlds/autonoship.world

The actual time flow in the simulation can be calculated by:

$$< real\_time\_update\_rate > * < max\_step\_size >$$
$$= 5000 * 0.001$$
$$= 5.0\ (secs\ in\ simulation / sec\ in\ real\ world)$$

4. Parameters of sensors

Parameters of all plugins can be found in the files:

<workspace_name>/src/autonoship_simulation/autonoship_gazebo/urdf/autonoship.launch*

5. Running the simulation

1) To control the ownship with keyboard ("w, a, s, d, q, e"), in a terminal, run:
   roslaunch autonoship_simulation autonoship_gazebo.launch

2) To run a certain testing scenario (e.g. scenario8), run:
   roslaunch autonoship_simulation autonoship_gazebo.launch scenario:=scenario8

3) To control targetships, publish message in the terminal:

rostopic pub targetship1/u2 std_msgs/Float64 10000000

4) To test the radar module, echo the radar feedback in the terminal:
   rostopic echo ownship/logical_camera

5) To change the RPM of the radar to 80, run in a terminal:
   rostopic pub ownship/rpm std_msgs/Float64 80

6) A radar illustration node is provided to show the position of targets in a plot. Run:
   rosrun autonoship_simulation radar_reader.py

7) To test the radar tracking module, echo the state of a targetship (e.g. targetship1):
   rostopic echo ownship/targetship1/state