Cartographer ROS for Fetch Robotics Platforms Documentation

Release 1.0.0

The Cartographer Authors

Nov 23, 2017

Contents

Cartographer is a system that provides real-time simultaneous localization and mapping (SLAM) in 2D and 3D across multiple platforms and sensor configurations. This repository provides Cartographer SLAM for Fetch Robotics platforms via Cartographer ROS.

CHAPTER 1

Building & Installation

Installation has been tested on Ubuntu 14.04 (Trusty) with ROS Indigo, but may also work on Ubuntu 16.04 (Xenial) with ROS Kinetic. We recommend using wstool and rosdep. For faster builds, we also recommend using Ninja.

```
# Install wstool and rosdep.
sudo apt-get update
sudo apt-get install -y python-wstool python-rosdep ninja-build

# Create a new workspace in 'catkin_ws'.
mkdir catkin_ws
cd catkin_ws
wstool init src

# Merge the cartographer_fetch.rosinstall file and fetch code for␣
↪dependencies.
wstool merge -t src https://raw.githubusercontent.com/googlecartographer/
↪cartographer_fetch/master/cartographer_fetch.rosinstall
wstool update -t src

# Install deb dependencies.
# The command 'sudo rosdep init' will print an error if you have already
# executed it since installing ROS. This error can be ignored.
sudo rosdep init
rosdep update
rosdep install --from-paths src --ignore-src -r --rosdistro=${ROS_DISTRO} -y

# Build and install.
catkin_make_isolated --install --use-ninja
source install_isolated/setup.bash
```

CHAPTER 2

Running the demo

Now that Cartographer and Cartographer's integration with Fetch Robotics platforms are installed, download the example bag to a known location, in this case `~/Downloads`, and use `roslaunch` to bring up the demo:

```
# Download the 2D example bag.
wget -P ~/Downloads https://storage.googleapis.com/cartographer-public-data/
↪bags/fetch/cartographer_freight_demo.bag

# Launch the 2D demo.
roslaunch cartographer_fetch offline_freight_2d.launch bag_filenames:=${HOME}
↪/Downloads/cartographer_freight_demo.bag

# Pure localization demo: We use the larger run as map and the smaller as␣
↪localization.
wget -P ~/Downloads https://storage.googleapis.com/cartographer-public-data/
↪bags/fetch/cartographer_freight_demo.bag
wget -P ~/Downloads https://storage.googleapis.com/cartographer-public-data/
↪bags/fetch/cartographer_freight_demo_2.bag
# Generate the map: Run the next command, wait until cartographer_offline_
↪node finishes.
roslaunch cartographer_fetch offline_freight_2d.launch bag_filenames:=${HOME}
↪/Downloads/cartographer_freight_demo_2.bag
# Run pure localization:
roslaunch cartographer_fetch demo_freight_localization.launch \
   bag_filename:=${HOME}/Downloads/cartographer_freight_demo.bag \
   map_filename:=${HOME}/Downloads/cartographer_freight_demo_2.bag.pbstream

# Download the 2D example bag with simulated data.
wget -P ~/Downloads https://storage.googleapis.com/cartographer-public-data/
↪bags/fetch/cartographer_freight_simulation_demo.bag

# Launch the 2D simulation demo.
roslaunch cartographer_fetch demo_freight_simulation.launch bag_filename:=$
↪{HOME}/Downloads/cartographer_freight_simulation_demo.bag
```

The launch files will bring up `roscore` and `rviz` automatically.