

SLAM Processing

Guide to processing SLAM data recorded into rosbags by a VLX1

- [Intro](#)
- [Requirements](#)
 - [Required programs](#)
- [Setup](#)
 - [Sources](#)
 - [Docker](#)
- [Process](#)
 - [Starting the Docker Container](#)
 - [Setting up the Environment](#)
 - [Setting Up ROS](#)
 - [ROS Commands](#)
 - [Dataset Structure](#)
 - [ROSBAG Structures](#)

Intro

These are my adventures in getting processing to work from VLX recordings.

Requirements

Requirements

Required programs

List of required programs + resources

- tmux
- docker - ros:noetic-ros-core container
- dataset from a VLX1

Setup

Setup

Sources

Sources used while getting this up and running:

ROS

Cartographer

Cartographer install: <https://google-cartographer-ros.readthedocs.io/en/latest/compilation.html>

Cartographer dependencies issue: https://github.com/cartographer-project/cartographer_ros/issues/1726

Cartographer build issues: <https://stackoverflow.com/questions/32801638/cmake-error-at-cmakelists-txt30-project-no-cmake-c-compiler-could-be-found>

All the dependencies which were problematic during the build process:

```
# Install Boost iostreams
sudo apt install libboost-iostreams-dev

# Install Ceres
sudo apt install libceres-dev

# Install Lua
sudo apt install liblua5.2-dev

# Install Protobuf
sudo apt install libprotobuf-dev protobuf-compiler

# Install Cairo
```

```
sudo apt install libcairo2-dev
```

```
# Install PCL Conversions
```

```
sudo apt install ros-noetic-pcl-conversions
```

```
# Install tf2_eigen
```

```
sudo apt install ros-noetic-tf2-eigen
```

```
# Install URDF
```

```
sudo apt install ros-noetic-urdf
```

```
# Install Google Mock (GMock)
```

```
sudo apt install libgmock-dev
```

```
# Install Google Test (if needed)
```

```
sudo apt install libgtest-dev
```

```
cd /usr/src/gtest
```

```
sudo cmake .
```

```
sudo make
```

```
sudo cp *.a /usr/lib
```

```
# Install RViz
```

```
sudo apt install ros-noetic-rviz
```

```
# Install roslint
```

```
sudo apt install ros-noetic-roslint
```

```
# Install PCL ROS
```

```
sudo apt install ros-noetic-pcl-ros
```

Setup

Docker

The ros:noetic-ros-core container is already pulled and created on the system. If it's not running it can be started again with

```
rkitec@gibsonhh:~$ docker start rosbag_container
```


Process

Starting the Docker Container

Start the ROS Noetic docker container.

```
rkitect@gibsonhh:~$ docker start rosbag_container
```

It should already have the volumes set

- rosbag_workspace:/root - Contains all changes made to the ROS environment after initial pull
- /mnt/taketwo/Web/nextcloud/data/rkitect/files/Documents/GI360/Ros Trials/AIA Conference/:/mnt/DataSets - Contains the AIA Conference DataSet that I scanned

Setting up the Environment

You will likely need multiple terminals running to see everything you are wanting to monitor. To do this you can either run tmux outside of the docker with multiple sessions, or multiple terminal windows that are executing the `rosbag_container` bash. Whichever method you use, you will need to ensure the rosbag_container container is already running. After some consideration, it's recommended to always use tmux so that you can continue working and maintain your command list if you get disconnected or want to work from a different workstation.

Start a ROSCORE Instance

You will need roscore running to perform any playback of data stored in bags. The best approach that I've found so far is to start roscore in a tmux session so that it is persistent without needing to always have a roscore terminal running.

Check to see if the roscore tmux session already exists

```
rkitect@gibsonhh:~$ tmux list-sessions
```

You should get a list of current tmux sessions.

```
rkitect@gibsonhh:~$ tmux list-sessions
invokewebai: 1 windows (created Wed Oct 16 09:56:34 2024)
rkitect@gibsonhh:~$
```

If roscore isn't listed, then continue. If it is listed, then roscore should be running already. You should hop into that tmux session to verify.

Start a new tmux session title roscore

```
tmux new -t roscore
```

Bash into the rosbag_container docker instance

```
rkitect@gibsonhh:~$ docker exec -it rosbag_container bash
root@03efcb8f4879:/#
```

Source the ROS environment

```
root@03efcb8f4879:/# source /opt/ros/noetic/setup.bash
```

Run ROSCORE

```
root@03efcb8f4879:/# roscore
... logging to /root/.ros/log/179ffe12-8f26-11ef-988c-0242ac1c0002/roslaunch-03efcb8f4879-255.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://03efcb8f4879:44425/
ros_comm version 1.17.0

SUMMARY
=====

PARAMETERS
* /rostdistro: noetic
* /rosversion: 1.17.0

NODES

auto-starting new master
process[master]: started with pid [263]
ROS_MASTER_URI=http://03efcb8f4879:11311/

setting /run_id to 179ffe12-8f26-11ef-988c-0242ac1c0002
process[rosout-1]: started with pid [273]
started core service [/rosout]
```

You can leave this up if you want to observe, or you can exit it by typing Ctrl+B. The tmux session will remain running. You can verify this by using `tmux list-sessions`

```
rkitect@gibsonhh:~$ tmux list-sessions
invokewebai: 1 windows (created Wed Oct 16 09:56:34 2024)
roscore-5: 1 windows (created Sun Oct 20 15:57:43 2024) (group roscore)
rkitect@gibsonhh:~$
```

If you want to reattach the roscore session you can use

```
rkitect@gibsonhh:~$ tmux a -t roscore
```

Existing Sessions

You can check for any existing tmux sessions for roscore using the list-sessions option for tmux

```
rkitect@gibsonhh:~$ tmux list-sessions
invokewebai: 1 windows (created Wed Oct 16 09:56:34 2024)
rosbag: 1 windows (created Tue Oct 22 22:24:00 2024)
roscore-5: 1 windows (created Sun Oct 20 15:57:43 2024) (group roscore)
rkitect@gibsonhh:~$
```

If you see the session you want to use go ahead and load into it.

New Sessions

If you don't see the session you want to continue using, or just want to start a new session, continue here. To start new sessions for running ROS commands or viewing output, use one of the following methods.

TMUX

```
rkitect@gibsonhh:~$ tmux new -t rosdocker1 #add a new number as needed or name appropriately
```

Once you are in the tmux session you can skip down to Bashing into the rosbag_container docker

Multi-Terminal

In each new terminal window, execute bash for the rosbag_container container

Bashing into the ROSBAG_CONTAINER docker

```
rkitect@gibsonhh:~$ docker exec -it rosbag_container bash  
root@03efcb8f4879:/#
```

You can now continue on to [Setting up the ROS Environment](#)

Process

Setting Up ROS

This process uses ROS Noetic because it was the latest version of ROS which still uses ROS1.

Once you're in the container bash, source the ROS Environment

```
source /opt/ros/noetic/setup.bash
```

ROS Commands

Below are some useful ROS Commands that you'll be using and needing to get information

- `roscore` - starts the roscore service required to playback and subscribe to bag topics
- `rosbag info` - lists information for a bag including the included topics

```
root@03efcb8f4879:/mnt/DataSets/internal/bags# rosbag info bag_laser_horiz_0.bag
path:      bag_laser_horiz_0.bag
version:    2.0
duration:   60.0s
start:      Sep 29 2022 16:07:41.83 (1664467661.83)
end:        Sep 29 2022 16:08:41.82 (1664467721.82)
size:       36.2 MB
messages:   3017
compression: bz2 [69/69 chunks; 68.63%]
uncompressed: 52.6 MB @ 898.2 KB/s
compressed:  36.1 MB @ 616.4 KB/s (68.63%)
types:      velodyne_msgs/VelodyneScan [50804fc9533a0e579e6322c04ae70566]
topics:     /laser_horiz/packets 3017 msgs : velodyne_msgs/VelodyneScan
root@03efcb8f4879:/mnt/DataSets/internal/bags#
```

- `rosbag play` - plays back the data recorded in a bag file

```
root@03efcb8f4879:/mnt/DataSets/internal/bags# rosbag play bag_laser_vert_0.bag
[ INFO] [1729462384.630717691]: Opening bag_laser_vert_0.bag

Waiting 0.2 seconds after advertising topics... done.

Hit space to toggle paused, or 's' to step.

[RUNNING] Bag Time: 1664467662.912766 Duration: 1.048908 / 59.992893
```

-

Dataset Structure

A quick rundown of the dataset folder structure. This is not the ROSBAG topic structure. This includes my understanding of each item to the best of my ability to find information on them.

- anchors -contains files from anchors which were set with the VLX during recording
 - anchor_poses.txt - location to add your manually surveyed control points. These should correspond with the anchors recorded in anchor_poses_recorded.txt
 - anchor_poses_recorded.txt - anchors recorded on the VLX during recording will be shown here.
 - anchors.txt
- cam - raw photos from each camera in DNG format
 - preview - thumbnails of each camera shot in JPG format
- dataset.json - config information for the scan itself
- info - location information for each camera shot recorded in .JSON format
- internal - recorded point information location
 - artifacts - IMU's local trajectory bag recording
 - bags - velodyne's point cloud bag recordings + sensor recordings
 - trajectory_slam.bag - ? just a single entry. unsure if velodyne or IMU recording
- logs - various log files recorded during recording
- maps - quality map recording and config data
- qualitymap_rec.png - quality map recorded during scan
- sensor_frame.xml
- sensorfarm.xml.sig
- trolley.cer
- wifi

ROSBAG Structures

These are the individual .bag files recorded from the VLX and what information can be found inside them.

File	Location	Creator	Info	Topics	Types
trajectory_slam.bag	/internal/	navvis	It's the path of something...	/path	nav_msgs/Path
trajectory_local.bag	/internal/artifacts /	IMU	the local path of the VLX during recording	/tf_trajectory	tf2_msgs/Path

bag_<timestamp>.bag	/internal/bags/	navvis	Mixed information from various sensors during scan	/beacon_data /cam0/temperature /cam1/temperature /cam2/temperature /cam3/temperature /controller/parameter_descriptions /controller/parameter_updates /controller/trigger /detected_hook_size /diagnostics /flir_sys_init /hw_trigger_button_pressed /image_saving /imu/acc_pure /imu/delta_vel_orient /imu/imu_raw/data /imu/imu_sync_status /imu/magnetic_field /joint_states /laser_horiz/temperature /laser_vert/temperature /target_fps /trigger_event /trigger_event_header /trigger_processed /usable_trajectory_start_time /user_interaction /watchdog/diagnostics /watchdog/status	bluetooth_beacon_logger/BleBeacon diagnostic_msgs/DiagnosticArray dynamic_reconfigure/Configure/Config dynamic_reconfigure/Configure/ConfigDescription geometry_msgs/Vector3Stamped navvis_msgs/HookSize navvis_msgs/ImageSaving navvis_msgs/ImuDeltaVelOrient navvis_msgs/ImuSyncStatus navvis_msgs/TriggerEvent navvis_msgs/UserInteraction navvis_msgs/WatchdogDiagnostics navvis_msgs/WatchdogStatus sensor_msgs/Imu sensor_msgs/JointState sensor_msgs/MagneticField sensor_msgs/Temperature std_msgs/Bool std_msgs/Header std_msgs/Time std_msgs/UInt32
bag_laser_horiz_#.bag	/internal/bags/	velodyne		/laser_horiz/packets	velodyne_msgs/VelodyneScan
bag_laser_vert_#.bag	/internal/bags	velodyne		/laser_vert/packets	velodyne_msgs/VelodyneScan