Overview

> This subsystem is still in experimentation phase and WILL change.

The autonomous navigation of the rover is supported by the use of an **Intel RealSense D435i** depth camera. This sensor provides synchronized RGB, depth, and inertial measurements (IMU), which are essential for perception and mapping tasks.

The software interface for the camera is provided through the **realsense-ros ROS2 wrapper** package realsenseai/realsense-ros: ROS Wrapper for RealSense™ Cameras, which enables integration of the camera with the ROS 2 ecosystem. This repository has been cloned into the workspace and is included as a package. Additionally, the **librealsense SDK** realsenseai/librealsense: RealSense SDK, which provides the low-level drivers and APIs for the camera—is also included as a separate package.

The primary package responsible for SLAM (Simultaneous Localization and Mapping) and map generation is `rtt_slam`. Within this package, all required dependencies are declared in the `package.xml` file, ensuring proper integration with the rest of the ROS 2 system.

The process takes the RGB and depth frames along with the IMU data and produces an STVL (Spatio-Temporal Voxel Layer) map as the output Using an External Costmap Plugin (STVL) — Nav2 1.0.0 documentation. The Nav2 framework is utilized to produce the map and for autonomous navigation.

To launch SLAM:

```
ros2 launch rtt_slam slam.launch.py
```

The launch file starts:

1. Realsense camera drivers
2. RTAB-Map odometry and SLAM
3. IMU filtering
4. Navigation2 (Nav2)
5. Costmap Visualization utilities

Launch File Documentation

The launch file defines and orchestrates all nodes and processes required for perception, SLAM, and data streaming. It ensures proper initialization order and parameter configuration.

The SLAM system is comprised of four launch files:

1. camera_launch.yaml

Responsible for starting the depth camera.

The launch file:

- Loads RealSense configuration from `config/camera_realsense.yaml`
- Starts the RealSense camera driver

Configuration:

- Enables the gyroscope, accelerometer, IMU synchronization, and point cloud generation.
- Aligns depth images to the RGB camera.
- Reduces image resolution for improved stability and performance.

2. rtabmap_launch.yaml

Responsible for localization and map generation.

The launch file:

- Loads RTAB-Map configuration from `config/rtabmap.yaml`
- Remaps camera RGB, depth, and camera info topics
- Starts RGB-D odometry
- Starts RTAB-Map SLAM
- Starts the Madgwick IMU filter
- Publishes a static transform between `base_link` and `camera_link`

Configuration:

- Uses synchronized RGB and depth images from the RealSense camera.
- Uses filtered IMU orientation estimates for odometry.
- Deletes any existing RTAB-Map database on startup.
- Optionally launches the RTAB-Map visualization tool.

3. navigation_launch.yaml

Responsible for autonomous navigation and costmap generation.

The launch file:

- Loads navigation parameters from `config/navigation.yaml`

- Starts the Nav2 navigation stack
- Starts costmap visualization nodes

Configuration:

- Uses the RTAB-Map occupancy grid as the global map.
- Uses a voxel-based local costmap generated from RealSense point clouds.
- Enables obstacle marking and clearing.
- Uses the DWB local planner for path following.

4. slam_launch.yaml

Responsible for launching the complete SLAM system.

The launch file:

- Launches `camera_launch.yaml`
- Launches `rtabmap_launch.yaml`
- Launches `navigation_launch.yaml`

Configuration:

- Combines perception, localization, mapping, and navigation into a single launch command.
- Ensures all required subsystems are started together.

Video Streaming to Basestation

```
ExecuteProcess(
    cmd=['/bin/bash', os.path.expanduser('~/stream.sh')],
)
```

A separate script (`stream.sh`) is executed to stream camera data to the base station.

Streaming Script Documentation

This script uses **GStreamer** to transmit video over UDP.

Paragraph Breakdown:

```
rosimagesrc ros-topic="/camera/camera/color/image_raw"
```

Captures image directly from a ROS topic.

```
videoconvert ! video/x-raw,format=I420
```

Converts image format for encoder compatibility

```
x264enc tune=zerolatency speed-preset=superfast
```

Encodes video using H.264 with low-latency settings

```
rtph264pay pt=96
```

Packages encoded video into RTP packets

```
udpsink host=145.126.xx.XXX port=4500
```

Sends the stream to the base station via UDP

RViz2

RViz2 is used as the primary visualization interface for the rover system. It allows real-time monitoring of sensor data such as RGB images, depth maps, point clouds, and TF frames. Additionally, it provides tools to visualize occupancy grids and SLAM outputs, aiding in debugging and system validation. It is used to test the implementation of the mapping and the working of the camera.

Revision #12
Created 2026-04-14 09:34:47 UTC by Dmytro Khorsun

Updated 2026-06-01 14:50:06 UTC by Narendra Setty