SLAMR01 Core

Create workspace folders used by SLAMR01 application

```
cd ~/workspace
mkdir rtk_logs ros_home
```

Application Dependencies

System Dependencies

```
sudo apt install ffmpeg

#
# Other Notes:
#

# libi2c must be manually build from source rather than using apt install (see further below)
#sudo apt install libi2c-dev

# opencv-python is built from source and not installed using apt
#  (see Jetson installation instructions for ROS on other page)

# PyQt5 (probably already installed from debian repo called python3-pyqt5)
```

Standard ROS Package Dependencies

```
sudo apt install ros-noetic-ddynamic-reconfigure ros-noetic-camera-calibration-parsers ros-noetic-camera-info-
manager ros-noetic-image-common

#
# NOTE: cv-bridge must not be installed using standard ROS package and should be build from source
#       (see Jetson install instructions for ROS on other page)
#
```

In addition to the standard ROS packages above, there are some packages that are not enforced as dependencies but are needed to support some basic functionality:

Octomap packages :

```
cd <catkin_workspace>/src/dependencies

# octomap_msgs
# (using commit dcaaf62bd0 Jul 1 18:56:48 2020 at time of writing)
git clone https://github.com/OctoMap/octomap_msgs.git

# octomap_ros
# (using commit fb462a9e5 Dec 14 00:12:35 2021 at time of writing)
git clone https://github.com/OctoMap/octomap_ros.git

# octomap_rviz_plugins
# (using commit c766489f62 Jun 2 17:40:07 2022 at time of writing)
git clone https://github.com/OctoMap/octomap_rviz_plugins.git
```

image_pipeline packages: https://github.com/ros-perception/image_pipeline.git

branch : noetic

NOTE: this must be built manually because it links to our custom build of OpenCV (DO NOT use apt install of ros-noetic-image-pipeline)

```
cd <catkin_workspace>/src/dependencies

# image_proc from image_pipeline
#  (using commit aa6437b197 -- May 30 17:37:03 2022 -- at time of writing
git clone https://github.com/ros-perception/image_pipeline.git
cd image_pipeline
git checkout noetic
```

realsense-ros package: https://github.com/IntelRealSense/realsense-ros.git

You would want to use the latest tag for noetic.

link to reference: https://confluence.xrcs.jhuapl.edu/confluence/x/RYOpB

depends packages:

ddynamic_reconfigure : **https://github.com/pal-robotics/ddynamic_reconfigure.git**

```
cd <catkin_workspace>/src/dependencies

# ROS wrapper for realsense
git clone https://github.com/IntelRealSense/realsense-ros.git

# checkout the last version for ROS1:
#  (goes to commit f400d682 Nov 15 03:20:29 2021 at time of this writing)
git checkout `git tag | sort -V | grep -P "^2.\d+\.\d+" | tail -1`

#
# NOTE: when installing with JetPack 5.0.1 the following commit was also used and worked fine:
#  commit e4938bbb Mar 23 12:29:11 2022
#
```

Aruco Tag package :

```
cd <catkin_workspace>/src/dependencies

 # Aruco library
git clone https://github.com/pal-robotics/aruco_ros.git

# checkout the noetic branch or the tag <version>
cd aruco_ros
git checkout tags/3.0.3
```

Primary Application

semantic segmentation package (**objectrecognition** APL):

you will need to download the weights trained for this inference model. You should place them in a folder called

~/workspace/models_weights/

The links to the weights: https://confluence.xrcs.jhuapl.edu/confluence/x/NwIsBg

```
cd <catkin_workspace>/src

# semantic segmentation
git clone https://bitbucket.xrcs.jhuapl.edu/scm/slamr01/objectrecognition.git
```

```
cd <catkin-workspace>/src

# haptic
git clone https://gitlab.jhuapl.edu/slamr01/Haptic.git

# Manually build libi2c
#
#  A manual build is needed because using "sudo apt install libi2c-dev"
#  does not include the python wrapper for libi2c
#

# Can build libi2c using the code saved in haptic repo:
cd Haptic/depends/libi2c-master
sudo python setup.py install
cd <catkin-workspace>/src

# Or build libi2c using the latest code from GitHub:
cd ~/programs/extern/libi2c
git clone https://github.com/amaork/libi2c.git
cd libi2c
sudo python setup.py install
cd <catkin-workspace>/src


# navmap
git clone https://gitlab.jhuapl.edu/slamr01/navmap.git

# rviz plugin
git clone https://gitlab.jhuapl.edu/slamr01/rviz_plugin_waypoint_selector.git

# core
git clone https://gitlab.jhuapl.edu/slamr01/core.git

# install Python dependencies for core (pip installs)
# you may need to disconnect from APLNIS for this step
cd core
pip install -r requirements.txt

cd <catkin-workspace>
catkin build core


#
# Apply bug fix for python-openal (see instructions below)
#
```

.bashrc

Add the following line to .bashrc in order to automatically source the catkin workspace (where <catkin-workspace> is the path to your top level catkin workspace directory):

source <catkin-workspace>/devel/setup.bash

FIX FOR FORCING AUDIO THROUGH JETSON WHEN REMOTE SSHING:

add the following 2 lines to ~/.bashrc:

alias force_audio='xprop -root -f PULSE_SERVER 8s -set PULSE_SERVER "<string-output>"'

force_audio

and replace <string-output> above with the string output provided when running the following command:

xprop -root PULSE_SERVER

Example:

xprop -root PULSE_SERVER

outputs

PULSE_SERVER(STRING) = "{dbfef1aa0b064bcf9d30ec3ad0886edb}unix:/run/user/1000/pulse/native"

therefore add to ~/.bashrc:

alias force_audio='xprop -root -f PULSE_SERVER 8s -set PULSE_SERVER {dbfef1aa0b064bcf9d30ec3ad0886edb}unix:/run/user/1000/pulse/native'

Bug Fix for python-openal

If you encounter the following error while running core node:

```
[ERROR] [1670431770.604798]: bad callback: <bound method AudioNode.play_audio_callback of <__main__.AudioNode
object at 0xffffaa875df0>>
Traceback (most recent call last):
  File "/opt/ros/noetic/lib/python3/dist-packages/rospy/topics.py", line 750, in _invoke_callback
    cb(msg)
  File "/mnt/slamr01_sd/workspace/catkin_ws/src/core/nodes/devices/audio_node.py", line 151, in
play_audio_callback
    self.sink.play(sources)
  File "/home/slamr01/.local/lib/python3.8/site-packages/openal/audio.py", line 518, in play
    al.alSourcePlayv(_to_ctypes(sids, al.ALuint), len(sids))
ctypes.ArgumentError: argument 1: <class 'TypeError'>: wrong type
```

This is caused by a bug in OpenAL. To fix this problem, in file "openal/audio.py" and under function "def play(self, sources)" (line 518) make the following changes:

```
# bug: comment out the line below (source file line 518), which has the function argument order reversed
#al.alSourcePlayv(_to_ctypes(sids, al.ALuint), len(sids))
# fix: replace with the line below, which corrects the function argument order
al.alSourcePlayv(len(sids), _to_ctypes(sids, al.ALuint))
```

Note:

- the file "openal/audio.py" is probably located at the path "~/.local/lib/python3.8/site-packages/openal/audio.py"
- you can use the following command to search for it: "sudo find / -name audio.py | grep openal/audio.py"

Other Notes

navmap_core package has the main launch files that runs the mapping and planner nodes with it's corresponding configurations.

It depends with capra packages.

EXAMPLES:

```
# mapping only with sensors
roslaunch navmap_core rtabmap_mapping_rs.launch

# Mapping playback from rosbags
roslaunch navmap_core rtabmap_mapping_playback_rs.launch rosbag_file_path:= {path_to_rosbag}

# mapping + APL RTK with sensors on platform
roslaunch navmap_core rtk_rtabmap_mapping.launch

# Mapping + APL RTK playback from rosbags
roslaunch navmap_core rtk_rtabmap_mapping_playback_rs.launch rosbag_file_path:= {path_to_rosbag}
```