DLI Edge AI and Robotics Teaching Kit

Module 4 Lab:   
**Autonomous JetBot Navigation with ROS and Gazebo**

This lab differs from the starter notebooks from the [NVIDIA-AI-IOT/Jetbot](https://github.com/NVIDIA-AI-IOT/jetbot) project on Github as it uses the Robot Operating System 2 (ROS2) as the middleware application to interface with the JetBot. Most of this instruction has been adapted from the [JetBot\_ROS](https://github.com/dusty-nv/jetbot_ros) Project.

This lab assumes you have loaded the most recent SD\* image for the Jetson Nano and have configured the Operating System. If needed, refer back to Module 1.3 for instructions on setting up the Jetson Nano.

\*Note - Many third party JetBot kits such as the [Sparkfun Jetbot AI](https://learn.sparkfun.com/tutorials/assembly-guide-for-sparkfun-jetbot-ai-kit/5-software-setup-guide-from-nvidia) Kit may already provide a pre-flashed SD card and is configured with drivers that support the hardware accessories in their kits. This lab makes use of containers and should not interfere with those drivers.

SSH into your Jetson Nano, then follow the steps below to download the jetbot\_ros project and run the container.\*\*

\*\*Virtual JetBot Note – if you do not have a JetBot and plan to launch a virtual JetBot with Gazebo, a headless connection is not recommended. You will get much better performance with the Gazebo Display if you directly connect a monitor, keyboard and mouse to the Jetson Nano 2GB. You can unplug the camera if an extra USB port is needed as the Virtual JetBot will be using a virtual camera.

**Load the jetbot\_ros project and run the container:**

Make a workspace directory\*

**mkdir workspace**

Navigate to that directory\*

cd /workspace

Make a src directory\*

mkdir src

Navigate to that directory\*

cd /src

\*This is a personal preference. You could opt to name this directory something else.

Clone the Jetbot\_Ros project from Github into the workspace/src directory you just created.

**sudo** **git clone** [**https://github.com/dusty-nv/jetbot\_ros**](https://github.com/dusty-nv/jetbot_ros)

Navigate to the jetbot\_ros directory

**cd jetbot\_ros**

Execute the run.sh script which will download and run the applicable Docker Jetbot\_ROS container that matches the l4t version of your SD image.

**docker/run.sh**

The run.sh script contains all the commands to download any needed libraries, build and run the container. It will download the appropriate container based on your image. Different versions of the container will run different versions of ROS2. As of November 2021, if you have the l4t 32.6.1 image, this will download the Foxy-r32.6.1 container.

Once the container is run, you the prompt will be set to the inside of the container similar to:

Text

Description automatically generated

**Launch the JetBot ROS Nodes:**

If you **do not** have a physical JetBot:

**ros2 launch jetbot\_ros gazebo\_world.launch.py**

If you have a physical JetBot, run:

**ros2 launch jetbot\_ros jetbot\_nvidia.launch.py**

This launches the nodes and to show the virtual JetBot or to prepare the physical JetBot.

Now let’s control the JetBot. Open another terminal and SSH into your JetBot. The type the following to get into your container:

**sudo docker exec -it jetbot\_ros /bin/bash**

Run the following to test controlling the JetBot with your keyboard  
:

**ros2 launch jetbot\_ros teleop\_keyboard.launch.py**

Text

Description automatically generated

**Data Collection:**

Next we will launch another launch file for data collection. This will launch the teleop\_keyboard node so that we can move the JetBot and the Data Collection Node. Run the following command inside the jetbot\_ros container:

**ros2 launch jetbot\_ros data\_collection.launch.py**

Begin to drive the robot with your keyboard and press the ‘c’ key to capture images. As you capture images, annotate the captured images by clicking on the center of the path in the pop-up window. Capturing a good range of images including when the JetBot is off the track, will help train the JetBot to get back on the track when navigating.

By default, the data is collected and saved to /workspace/src/jetbot\_ros/data/datasets/.

Press **Ctrl + C** when done collecting data. Now it’s time train the model.

You could opt to take this data and train on a different machine by running the jetbot\_ros container and copying the data to that machine. Let’s proceed with training on this Jetson Nano. We need to train a model and generate the best\_model.pth file that we need for navigation.

Train the Model with the Data you Collected:

Run the command to change to the dnn directory:

**cd /workspace/src/jetbot\_ros/jetbot\_ros/dnn**

Next, run this command to call the code needed to train and create a model:

**python3 train.py --data /workspace/src/jetbot\_ros/data/datasets/20211018-160950/**

It may take a while for the training to complete. Once it is done, you can now run the model.

**Reset the JetBot Position**

If you are using the virtual JetBot, reset the JetBot’s position by pressing the **Ctrl + R** keys in Gazebo.

**Launch Autonomous Navigation using the trained model:**

If you are still in the dnn directory, you can change back to the workspace directory:

**cd /workspace**

Launch the command to have the Jetson navigate autonomously. Make sure you use the path to the model:

**ros2 launch jetbot\_ros nav\_model.launch.py model:=/workspace/src/jetbot\_ros/data/models/202106282129/best\_model.pth**

**Deliverables:**

* **Dataset.zip:** A zipped file of the dataset folder that contains the images you collected and used for training.
* **best\_model.pth:** The generated model file from training the model using the images you collected.