File Types/extensions used in the folder

1) .py: Python script file used for writing Python code.
2) .sh: Shell script file used for writing command-line scripts in Linux.
3) .dae: COLLADA file format for exchanging 3D digital assets.
4) .stl: Stereolithography file format used for 3D printing and CAD models.
5) .rosinstall: ROS workspace configuration file for managing dependencies and source code locations.
6) .rviz: RViz configuration file for saving visualization settings in ROS.
7) .xml: Extensible Markup Language file used for storing and transporting data.
8) .urdf.xacro: XML macro file for generating URDF files in ROS.
9) .txt: Plain text file used for unformatted text.
10) .md: Markdown file used for writing formatted text using Markdown syntax.

Husky_demo/

ROS 2 build
build/
install/
log/
packages
src/isaac_ros_*
src/realsense-ros
src/ros-foxglove-bridge
src/husky
src/navigation
src/nvorn
src/xbox_test_keyboard
src/joint_state_publisher
internal data
host_gate

.gitignore — Ignores files and directories in version control

docs/ — contains the images for the tutorial

Readme.md — contains the tutorial

isaac_ros/ — Main Folder and the directory for the workstation

husky_demo.sh (Bash Script) — Installs and runs the Isaac ROS demo on a workstation or NVIDIA Jetson device. Launches Isaac Sim.

extensive summary of what it does

1. Initializes paths and environment variables and checks for necessary directories.

2. Installs Isaac ROS Demo either on workstation or nano:
   a. Workstation → Installs the demo on a desktop environment.
      - Checks dependencies.
      - Sets up ROS.
      - Launches Isaac ROS and Husky simulation environment.
      - Copies Fast DDS configuration.
   b. Jetson → Uses Jetson nano
      - Checks the Jetson LVT version.
      - Installs necessary packages.
      - Configures Docker images for Isaac ROS.
      - Sets up the demo on the Jetson device.
      - Builds ROS packages and runs them.

3. Launches Isaac SIM using $ISAAC_SIM_PATH/python.sh launching the simulation script husky_isaac_sim.py.

4. Launches ROS:
   a. Workstation → IF HIL_DEMO is not set, runs Isaac ROS and Husky by executing scripts/run_dev.sh.
   b. Jetson → sourcing the ROS setup file and launching the appropriate ROS launch file.

5. Scripts

   1. run_dev.sh: isaac_ros_common/scripts.
      - Sets up and runs the Docker environment for Isaac ROS.

   2. husky_isaac_sim.py: $ISAAC_DEMO_LOCAL_PATH/scripts.
      - Launches the Isaac SIM application and runs the simulation.

   3. python.sh: $ISAAC_SIM_PATH/

   4. fastdds.xml: $ISAAC_DEMO_LOCAL_PATH/scripts.
      - Copies the Fast DDS configuration to the .ros folder if necessary.

   5. rosinstall/husky_workstation.rosinstall: in $ISAAC_DEMO_LOCAL_PATH.
      - Pulls or update ROS packages for workstation installation.

   6. rosinstall/husky_robot.rosinstall: in $ISAAC_DEMO_LOCAL_PATH.
      - Pulls or updates ROS packages for Jetson installation.

build/ **install/** — *After Colcon build*

CMakeLists.txt — Configures build, installs directories, and sets up ament package.

package.xml — Defines the Husky Isaac SIM package, dependencies, and maintainer information.

src/
Main Folder and the directory for the workstation

scripts/
Scripts for husky

run_in_docker.sh — Sets up Husky demo in Docker:
- handles dependencies installation,
- builds the workspace,
- launches the demo with ROS2 and Isaac Sim.

Calls the following functions:
1. DDS configuration: fastdds.xml
2. Environment Setup: source install/setup.bash
3. launches allinone.launch.py or husky.launch.py

fastdds.xml — An XML configuration file for Fast Data Distribution Service, setting up UDPv4 transport for Real-Time Publish-Subscribe communication.

husky_isaac_sim.py — Runs the simulation environment in Isaac Sim:
- loads environments
- configures ROS2 communication
- handles URDF robot descriptions.

uses following extensions:
1. husky_action_graphs.py,
2. omni.isaac.kit,
3. omni.isaac.core,
4. rclpy.

husky_action_graphs.py — Creating action graphs in Isaac Sim, including camera and differential controller graphs for the Husky robot.

uses Isaac Sim extensions:
1. omni.graph.core,
2. omni.isaac.core.utils,
3. omni.isaac.ros2_bridge,
4. pxr.UsdGeom (for usds).

meshes/
meshes for cameras

d435.dae — info about depth camera

zed2.stl — 3D Model for the depth camera.

rosinstall/
From GitHub clones necessary dependencies and NVIDIA Isaac ROS Packages

husky_robot.rosinstall — *Make sure the website is up-to-date*

List of packages:
1. isaac_ros_common,
2. isaac_ros_nitros,
3. isaac_ros_nvblox,
4. isaac_ros_visual_slam,
5. isaac_ros_apriltag,
6. ros-foxglove-bridge,
7. realsense-ros,
8. xacro.

husky_workstation.rosinstall — *Make sure the website is up-to-date*

List of packages:
1. isaac_ros_common,
2. isaac_ros_nvblox,
3. Husky robot repository: humble-devel
5. xacro.

rviz/
meshes for cameras

navigation.rviz — Sets up RViz to visualize navigation data with customized displays and panel layouts.

urdf.rviz — configures RViz to visualize URDF models with specific displays and customized panel layouts.

launch/
Python Scripts

Accessories/

d435.urdf.xacro — Defines the Intel RealSense D435 camera model: its physical and visual properties, and optionally using mesh files.

zed.urdf.xacro — Defines the ZED stereo camera model; optionally using mesh files.

sensor_arch.urdf.xacro — Defines a sensor arch mount for the Husky robot

urdf/
XML format used in ROS to describe the physical configuration of a robot enabling simulation, visualization, and control.:
- joints,
- links,
- visual properties,
- physical properties.

empty.urdf — For testing purposes

husky_macro.urdf.xacro — Contains macro definitions for the Husky robot and reusable blocks of URDF code for defining:
1. links,
2. joints,
3. sensors, etc.

decorations.urdf.xacro — Defines decorative elements for the Husky robot (aesthetic or nonfunctional parts)

wheel.urdf.xacro — Defines Husky robot's configuration for integration with Isaac Sim:
- paths to mesh files
- additional macros.

husky.isaac.xacro — Defines Husky robot's configuration for integration with Isaac Sim:
- paths to mesh files
- additional macros.

allinone.launch.py — Combines multiple launch files into one

robot_display.launch.py — description + display

display.launch.py — Configures and launches nodes for visualizing the robot state in RViz.

description.launch.py —
- Sets up robot_state_publisher node to publish robot state to the ROS system
- Uses xacro to process URDF files

husky.launch.py — Combines multiple launch files into one

nvblox.launch.py —
- Launches the nvblox node for 3D mapping and perception.
- Loads parameters from configuration files and sets up necessary namespacing.

nvblox_examples_bringup
nvblox_ros package

nav2_isaac_sim.launch.py — Configures and launches the Nvblox node for 3D mapping and perception.