Installation and Usage Guide for CARLA–ROS 2 Integration (Verified Fork Edition with Reusable Setup Scripts)

Sören, Nikita, Iman
Prepared for Intelligent Robots Lab

Version 1.3 – October 2025

Abstract. This verified installation and usage guide provides a reliable, step-by-step procedure for integrating the CARLA autonomous driving simulator with ROS 2 using the `carla_ros_bridge`. While an installation guide is also available on the official CARLA site, this document uses the well-tested `ttgamage/carla-ros-bridge` fork (branch `humble`), which has proven to work cleanly with ROS 2 Humble. The guide covers both native Ubuntu and Windows 11 with WSL 2, explains how to record and inspect simulation data, includes automation support for VS Code, and adds reusable shell scripts to standardize setup and daily operations. All paths and examples are generic and avoid personal data.

1 Overview and Roadmap

CARLA (*Car Learning to Act*) is an open-source simulator for autonomous driving research. ROS 2 (*Robot Operating System*) provides a middleware for modular robotics applications.

The `carla_ros_bridge` connects CARLA with ROS 2, enabling:

- Real-time streaming of sensor data (LiDAR, camera, IMU, etc.),
- Vehicle control via ROS topics/services,
- Recording of topics for offline analysis and dataset generation.

1.1 Workflow Overview

Step 1: Install prerequisites and environment (Ubuntu or Windows + WSL 2).
Step 2: Install CARLA simulator.
Step 3: Install ROS 2 (Humble preferred).
Step 4: Build the bridge from the **ttgamage** fork (`humble` branch).
Step 5: Create reusable setup/run scripts (this guide provides templates).
Step 6: Verify connectivity and record simulation data.
Step 7: (Optional) Automate with VS Code tasks.

2 System Requirements and Compatibility

- **OS:** Ubuntu 22.04 LTS (recommended) or Windows 11 with WSL 2 (Ubuntu 22.04).
- **CARLA:** 0.9.15 or later.
- **ROS 2:** Humble Hawksbill.
- **Python:** 3.10 or later.
- **GPU:** NVIDIA GPU with recent driver (CUDA-capable). Check with `nvidia-smi`.
- **Storage:** $\geq$ 20 GB free.

Note on the bridge repository:

- Official docs exist; however, for ROS 2 Humble, the fork `https://github.com/ttgamage/carla-ros-bridg` (`humble` branch) is recommended due to better compatibility.

3 Installation on Ubuntu (Native)

3.1 1. Enable Repositories and Update

```
sudo apt update && sudo apt install -y software-properties-common
sudo add-apt-repository universe
sudo apt update && sudo apt upgrade -y
```

3.2 2. Install Dependencies

```
sudo apt install -y build-essential git python3-venv python3-pip \
  wget curl gnupg lsb-release
```

3.3 3. Create a Python Virtual Environment

```
mkdir -p ~/carla_project
cd ~/carla_project
python3 -m venv carla_venv
source carla_venv/bin/activate
```

Tip: Always activate this environment before Python tasks.

3.4 4. Install CARLA Simulator

```
wget https://carla-releases.s3.eu-west-3.amazonaws.com/Linux/CARLA_0
    .9.15.tar.gz
tar -xvzf CARLA_0.9.15.tar.gz
```

Start CARLA (open a new terminal, **Terminal A**):

```
./CarlaUE4.sh
```

If dependencies are missing:

```
sudo apt install -y libomp5 libvulkan1
```

3.5 5. Install ROS 2 Humble (Verified Key/Repo Commands)

```
sudo apt install -y locales
sudo locale-gen en_US en_US.UTF-8
sudo update-locale LC_ALL=en_US.UTF-8 LANG=en_US.UTF-8
export LANG=en_US.UTF-8

sudo mkdir -p /usr/share/keyrings
sudo curl -sSL https://raw.githubusercontent.com/ros/rosdistro/master/ros
    .key \
   -o /usr/share/keyrings/ros-archive-keyring.gpg

echo "deb [arch=$(dpkg --print-architecture) \
signed-by=/usr/share/keyrings/ros-archive-keyring.gpg] \
http://packages.ros.org/ros2/ubuntu $(. /etc/os-release && echo
    $UBUNTU_CODENAME) main" \
| sudo tee /etc/apt/sources.list.d/ros2.list > /dev/null

sudo apt update
sudo apt install -y ros-humble-desktop
```

Add sourcing to `.bashrc`:

```
echo 'source /opt/ros/humble/setup.bash' >> ~/.bashrc
```

3.6 6. Build the ROS–CARLA Bridge (ttgamage Fork)

```
sudo apt install -y python3-colcon-common-extensions
mkdir -p ~/carla_ros_ws/src
cd ~/carla_ros_ws/src
git clone --recurse-submodules -b humble https://github.com/ttgamage/
    carla-ros-bridge.git
cd ..
rosdep update
rosdep install --from-paths src --ignore-src -r -y
colcon build --symlink-install
```

Source the workspace:

```
source ~/carla_ros_ws/install/setup.bash
```

If headers are missing:

```
sudo apt install -y ros-humble-tf2-eigen
colcon build
```

4 Installation on Windows 11 + WSL 2

4.1 1. Enable WSL 2 and Install Ubuntu

```
wsl --install -d Ubuntu-22.04
```

Update the new system:

```
sudo apt update && sudo apt upgrade -y
```

4.2 2. Install ROS 2 in WSL 2

Repeat the steps from Section 3.5 inside WSL 2.

4.3 3. Install CARLA on Windows

Download the Windows build (e.g., to `C:\CARLA_0.9.15\WindowsNoEditor`) and run:

```
CarlaUE4.exe
```

4.4 4. Connect CARLA (Windows) and ROS 2 (WSL 2)

In WSL 2 (replace with your Windows host IP):

```
export CARLA_HOST=<Windows_IP>
export CARLA_PORT=2000
ping <Windows_IP>
```

4.5 5. Build the Bridge in WSL 2 (ttgamage Fork)

```
sudo apt install -y python3-colcon-common-extensions
mkdir -p ~/carla_ros_ws/src
cd ~/carla_ros_ws/src
git clone --recurse-submodules -b humble https://github.com/ttgamage/
    carla-ros-bridge.git
cd ..
rosdep update
rosdep install --from-paths src --ignore-src -r -y
colcon build --symlink-install
source install/setup.bash
```

5 Reusable Setup and Run Scripts (Recommended)

To standardize daily use, create the following scripts in a project folder (e.g., `/carla_project/scripts`). Make them executable with `chmod +x`.

5.1 1. Project Environment Script (`setup_env.sh`)

```
#!/usr/bin/env bash
set -euo pipefail

# --- Optional: activate Python venv for tooling (not required for ROS
    itself)
if [ -f "$HOME/carla_project/carla_venv/bin/activate" ]; then
  # shellcheck disable=SC1091
  source "$HOME/carla_project/carla_venv/bin/activate"
fi

# --- ROS 2 Humble
# shellcheck disable=SC1091
source /opt/ros/humble/setup.bash

# --- ROS workspace (adjust if different)
if [ -f "$HOME/carla_ros_ws/install/setup.bash" ]; then
  # shellcheck disable=SC1091
  source "$HOME/carla_ros_ws/install/setup.bash"
fi

# --- Optional: ensure a deterministic RMW implementation
export RMW_IMPLEMENTATION=rmw_cyclonedds_cpp

# --- Optional: CARLA host/port (set if connecting to remote/Windows)
export CARLA_HOST=${CARLA_HOST:-127.0.0.1}
export CARLA_PORT=${CARLA_PORT:-2000}
```

5.2 2. Start CARLA (Linux) (`run_carla.sh`)

```
#!/usr/bin/env bash
set -euo pipefail
# From within the CARLA extracted folder:
# ./run_carla.sh
if [ -x "./CarlaUE4.sh" ]; then
  exec ./CarlaUE4.sh
else
  echo "Error: run from CARLA folder containing ./CarlaUE4.sh" >&2
  exit 1
fi
```

5.3 3. Start Bridge (run_bridge.sh)

```
#!/usr/bin/env bash
set -euo pipefail
# shellcheck disable=SC1091
source "$HOME/carla_project/scripts/setup_env.sh"
exec ros2 launch carla_ros_bridge carla_ros_bridge.launch.py
```

5.4 4. Spawn Example (spawn_objects.sh)

```
#!/usr/bin/env bash
set -euo pipefail
# shellcheck disable=SC1091
source "$HOME/carla_project/scripts/setup_env.sh"

DEFS_JSON="${1:-/path/to/sensors.json}"
exec ros2 run carla_spawn_objects carla_spawn_objects \
  --ros-args --param "objects_definition_file:=${DEFS_JSON}"
```

5.5 5. Record and Inspect (record_start.sh, record_info.sh)

```
#!/usr/bin/env bash
# record_start.sh
set -euo pipefail
# shellcheck disable=SC1091
source "$HOME/carla_project/scripts/setup_env.sh"
OUT="${1:-carla_session}"
exec ros2 bag record -a -o "$OUT"
```

```
#!/usr/bin/env bash
# record_info.sh
set -euo pipefail
# shellcheck disable=SC1091
source "$HOME/carla_project/scripts/setup_env.sh"
BAG_DIR="${1:-carla_session}"
exec ros2 bag info "$BAG_DIR"
```

5.6 6. Helpful One-Off Checks

```
# Echo a topic exactly once (Humble syntax):
ros2 topic echo /clock --once

# List CARLA-related topics:
ros2 topic list | grep carla || true
```

6 Running and Recording (Using Scripts or Manually)

6.1 A. Using the Scripts

1. **Terminal A:** Start CARLA (Linux): `./run_carla.sh`
2. **Terminal B:** Start the bridge: `./run_bridge.sh`
3. **Terminal C:** Spawn sensors/vehicles: `./spawn_objects.sh /path/to/sensors.json`
4. **Terminal D:** Record all topics: `./record_start.sh carla_session`

6.2 B. Manual Commands (Reference)

```
# Terminal A (Linux):
./CarlaUE4.sh

# Terminal B:
source ~/carla_ros_ws/install/setup.bash
ros2 launch carla_ros_bridge carla_ros_bridge.launch.py

# Terminal C:
ros2 run carla_spawn_objects carla_spawn_objects \
  --ros-args --param objects_definition_file:=/path/to/sensors.json

# Terminal D:
ros2 bag record -a -o carla_session
```

7 Automating with VS Code

Create `.vscode/tasks.json` in your project root:

```
{
  "version": "2.0.0",
  "tasks": [
    {
      "label": "Run CARLA (Linux)",
      "type": "shell",
      "command": "./run_carla.sh",
      "problemMatcher": []
    },
    {
      "label": "Run ROS Bridge",
      "type": "shell",
      "command": "bash -lc 'source ~/carla_project/scripts/setup_env.sh &&
           ros2 launch carla_ros_bridge carla_ros_bridge.launch.py'",
      "problemMatcher": []
    },
    {
      "label": "Record All Topics",
```

```
    "type": "shell",
    "command": "bash -lc 'source ~/carla_project/scripts/setup_env.sh &&
        ros2 bag record -a -o carla_session'",
    "problemMatcher": []
  }
 ]
}
```

Tip: The `bash -lc` form ensures your environment loads in VS Code's integrated terminal.

8 Known Issues and Fixes

- `ros2`**: command not found** — Ensure the environment is sourced (`source /opt/ros/humble/setup.bash`) or use `setup_env.sh`.
- **Missing** `tf2_eigen.hpp` — Install: `sudo apt install -y ros-humble-tf2-eigen`, then rebuild with `colcon`.
- **Permission denied** — Make scripts executable: `chmod +x *.sh`.
- **Bridge not connecting** — Verify `CARLA_HOST`, port `2000`, and that CARLA is running.
- **Stale build caches** — `rm -rf build install log && colcon build -symlink-install`.
- **WSL networking** — Allow TCP 2000–2002 in Windows Firewall; test with `ping` and (if installed) `nc -z <IP> 2000`.

9 Appendix: Recommended Terminal Layout

Terminal	Purpose
A	Start CARLA simulator
B	Run ROS bridge
C	Spawn sensors/vehicles
D	Record or playback data

Tips

- Keep ROS 2 and CARLA versions synchronized to avoid API mismatches.
- Use short paths without spaces for reliable scripting.
- Store project scripts under a dedicated `scripts` folder and version control them.