Day 1 Notes — TurtleBot3 + RTAB-Map Simulation Project

1. Important ROS Commands for the Project

Environment, build, and sourcing:

```
# Create workspace
mkdir -p ~/Desktop/turtlebot3_ws/src
cd ~/Desktop/turtlebot3_ws
colcon build --symlink-install
source /opt/ros/humble/setup.bash
source ~/Desktop/turtlebot3_ws/install/setup.bash

# Set model (Burger | Waffle | Waffle Pi)
echo 'export TURTLEBOT3_MODEL=waffle_pi' >> ~/.bashrc
source ~/.bashrc
```

Launch Gazebo world and robot:

```
# Gazebo with TurtleBot3 default world
export TURTLEBOT3_MODEL=waffle_pi
ros2 launch turtlebot3_gazebo turtlebot3_world.launch.py
```

Teleop control:

```
# Option A: launched from full_sim.launch.py using xterm (requires: sudo apt install xterm)
# Option B: run manually in its own terminal
ros2 run turtlebot3_teleop teleop_keyboard
```

RTAB-Map (simulation-safe, LiDAR-only mode):

```
# RTAB-Map in LiDAR-only mode (simulation does not provide depth frames)
ros2 run rtabmap_slam rtabmap --delete_db_on_start --ros-args   -p use_sim_time:=true -p frame_
```

Unified bring-up: Gazebo + RTAB-Map + RViz + Teleop + Map Saver:

```
# Custom master launch created in our package (turtlebot3_rtabmap)
# Starts Gazebo, RTAB-Map (LiDAR mode in sim), RViz, teleop, and map_saver_server
ros2 launch turtlebot3_rtabmap full_sim.launch.py
```

RViz, TF, and topics:

```
# Visualize TF tree (generates frames.pdf)
ros2 run tf2_tools view_frames
xdg-open frames.pdf

# Verify active topics and data rates
ros2 topic list
ros2 topic info /scan
ros2 topic hz /map
```

Map saving (Nav2 map_saver service):

```
# Ensure maps folder exists
mkdir -p ~/Desktop/turtlebot3_ws/maps

# Start map saver (already included in full_sim.launch.py)
# To save the latest map explicitly:
ros2 service call /map_saver/save_map nav2_msgs/srv/SaveMap "{map_topic: '/map', map_url: '/hom
```

```
# RTAB-Map internal database (debug/analysis):
# ~/.ros/rtabmap.db  (written automatically on shutdown)
```

2. Issues and Problems Occurred

Issue: Wrong apt package syntax and missing packages

Details / Fix: Initial apt commands used placeholder angle-brackets and some packages like turtlebot3_slam were not present. We corrected installs and created a custom package turtlebot3_rtabmap with our own launch files.

Issue: Real robot bringup failed (serial /dev/ttyACM0 and LDS)

Details / Fix: Tried launching hardware node; failed because we are running simulation. Solution: use Gazebo bringup only, avoid hardware drivers.

Issue: Package 'rtabmap_ros' found but libexec missing

Details / Fix: Newer ROS2 distributions split executables (e.g., rtabmap_slam/rtabmap). We changed launch to call the correct executable and rebuilt/sourced the workspace.

Issue: RViz error: Frame [map] does not exist

Details / Fix: map→odom TF wasn't published because RTAB-Map was waiting for inputs. We enabled LiDAR-only mode and drove the robot with teleop; TF then showed map→odom.

Issue: Depth missing in Gazebo for waffle/waffle_pi

Details / Fix: Gazebo's TurtleBot3 models do not provide /camera/depth/image_raw. RTAB-Map stalled when subscribe_depth=true. We disabled depth (and optionally RGB) in simulation and ran LiDAR-only.

Issue: Teleop didn't open in separate window

Details / Fix: full_sim.launch.py uses prefix='xterm -e'. If xterm isn't installed it silently fails. Fix: sudo apt install xterm or run teleop manually in another terminal.

Issue: Graph optimization warnings (GTSAM IndeterminantLinearSystemException)

Details / Fix: Low-texture simulated scenes and limited constraints make the graph underconstrained. These warnings are non-fatal; maps still update and save.

Issue: Map files not appearing in maps/ folder

Details / Fix: map_saver_server is a service; it does not auto-save on exit. We created the maps folder and used a service call to save my_map.{pgm,yaml}. RTAB-Map's own DB is written to ~/.ros/rtabmap.db.

3. Technical Aspects — Explanation of Key Concepts

RTAB-Map overview: RTAB-Map (Real■Time Appearance■Based Mapping) performs loop■closure■capable SLAM. It accepts RGB■D, stereo, or LiDAR + odometry inputs. In ROS2 Humble, the main executable is rtabmap from the rtabmap_slam package.

Sensor combinations used today: In Gazebo, TurtleBot3 Burger has only LiDAR; Waffle adds RGB; Waffle Pi hardware adds RGB■D. Gazebo's stock models do not publish depth, so we operated in LiDAR■only mode to guarantee reliable mapping.

Why subscribe_depth and subscribe_rgb are disabled in simulation: When depth is enabled but not published, RTAB■Map blocks waiting for frames and never creates the map frame. RGB■only can also stall due to synchronization and low■texture images. Disabling both forces a deterministic LiDAR■only pipeline.

TF frames for navigation: The minimal chain is map → odom → base_footprint → base_link. RTAB■Map publishes map→odom after it processes scans and odometry. RViz should use Fixed Frame = map for global visualization.

Use of approx_sync: approx_sync=true allows RTAB■Map to accept messages with slightly different timestamps, which is helpful when camera/LiDAR/odom rates don't perfectly align. In pure LiDAR mode it remains harmless.

Occupancy grid vs point cloud maps: /map is a 2D nav_msgs/OccupancyGrid used by Nav2. RTAB■Map can also publish /cloud_map (3D point clouds) when depth is available; in LiDAR■only mode this is limited.

Map persistence: rtabmap.db vs PGM/YAML: RTAB■Map saves an internal database (~/.ros/rtabmap.db) automatically; it stores poses, features, and raw data for later analysis. To use the map with Nav2, we save an occupancy grid via the map_saver service to a .pgm image with a .yaml metadata file.

full_sim.launch.py structure: Our launch integrates: Gazebo world, RTAB■Map (LiDAR mode), RViz with preset config, teleop (via xterm), and nav2_map_server's map_saver_server. This provides a one■command workflow to explore and save a map.

4. Summary to Put in Resume — Day 1

- Set up ROS2 Humble workspace and installed TurtleBot3, RTAB■Map, Nav2, and SLAM Toolbox packages.
- Built a custom package (turtlebot3_rtabmap) and a master launch (full_sim.launch.py) that starts Gazebo, RTAB■Map, RViz, teleop, and a map saver.
- Diagnosed and resolved SLAM bring■up issues: missing executables, map frame not existing, teleop window failures, and Gazebo's lack of depth images.
- Configured RTAB■Map for simulation■safe LiDAR■only SLAM (subscribe_scan=true; subscribe_depth=false; subscribe_rgb=false; approx_sync=true).
- Verified TF tree (map→odom), inspected topics, and produced 2D occupancy maps from LiDAR; saved maps using nav2_map_server service.
- Documented a repeatable workflow and created study notes detailing commands, concepts, and troubleshooting steps.