NYU TANDON SCHOOL OF ENGINEERING

AI4CE

Self Drive Group 2 Final Presentation

Zihan Liu, Raman Jha, Jason Luo, Shreyansh Bhardwaj

Data Collection

- Collecting Data - Still use the rosbag
  - A rosbag assembles all the info we want into a single file, which is convenient

```
zihan@daniel:~/catkin_s25/src/NYU-Self-Drive_S25/bags$ rosbag info testrun1_2025-02-27-16-52-33.bag
path:        testrun1_2025-02-27-16-52-33.bag
version:     2.0
duration:    2:29s (149s)
start:       Feb 27 2025 16:52:33.90 (1740675153.90)
end:         Feb 27 2025 16:55:03.42 (1740675303.42)
size:        671.7 MB
messages:    6663
compression: none [761/761 chunks]
types:       geometry_msgs/Twist [9f195f881246fdfa2798d1d3eebca84a]
             nav_msgs/Odometry   [cd5e73d190d741a2f92e81eda573aca7]
             sensor_msgs/Image   [060021388200f6f0f447d0fcd9c64743]
topics:      /camera/image    760 msgs    : sensor_msgs/Image
             /cmd_vel        1500 msgs    : geometry_msgs/Twist
             /odom           4403 msgs    : nav_msgs/Odometry
```

Exploration

Launch File Structure we are going to have:

Rosbag Recording Node

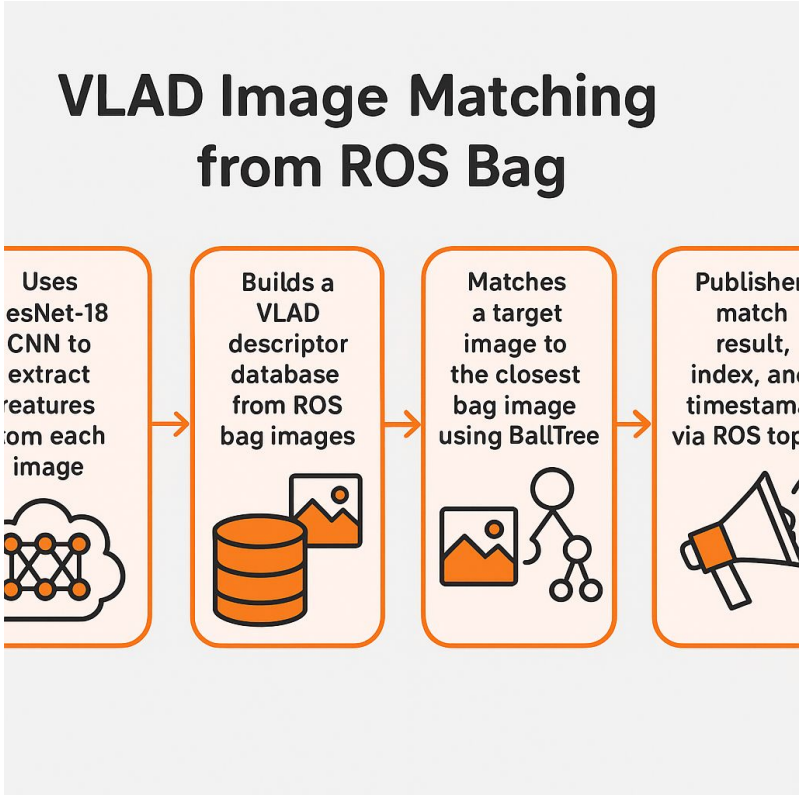
- It launches a node from the `rosbag` package using the `record` type.
- It records the topics `/odom`, `/camera/image`, and `/cmd_vel`.
- The recorded data is saved to a bag file with a filename specified by the `bag_filename` argument. The file is saved in the path `$(env HOME)/catkin_s25/src/NYU-Self-Drive_S25/bags/testrun1.bag`.

Turtlebot Teleoperation

- It includes another launch file from the `turtlebot3_teleop` package (`turtlebot3_teleop_key.launch`).
- This included launch file is used to enable keyboard teleoperation for Turtlebot3.

Navigation - CNN Feature extraction

- Uses **ResNet-18 CNN** to extract features from each image.
- Builds a **VLAD descriptor database** from ROS bag images.
- Matches a **target image** to the closest bag image using **BallTree**.
- Publishes match result, index, and timestamp via ROS topics.



Odometry-Based Shortest Path to Image

Extracts odometry poses and timestamps from ROS bag.

Finds odometry index closest to a target image timestamp.

Builds graph:

- Sequential connections between odometry points.
- **Loop closures**: connects spatially close but temporally distant nodes.

Computes shortest path from start to target using Dijkstra's algorithm.

Move the turtlebot

Extracts odometry + image timestamps from a bag file

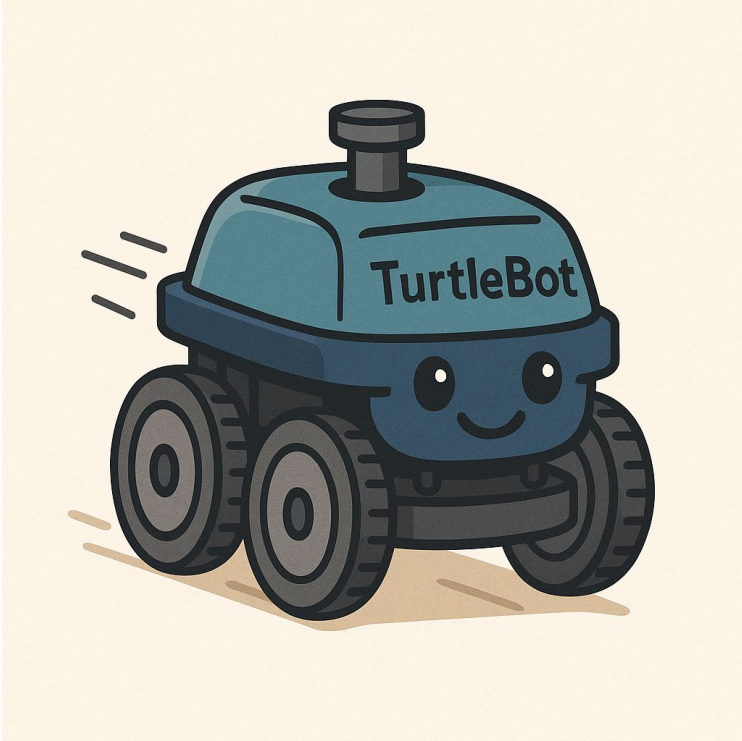
Finds shortest path to a goal image using loop-closing graph

Filters waypoints and detects **turning angles** from path geometry

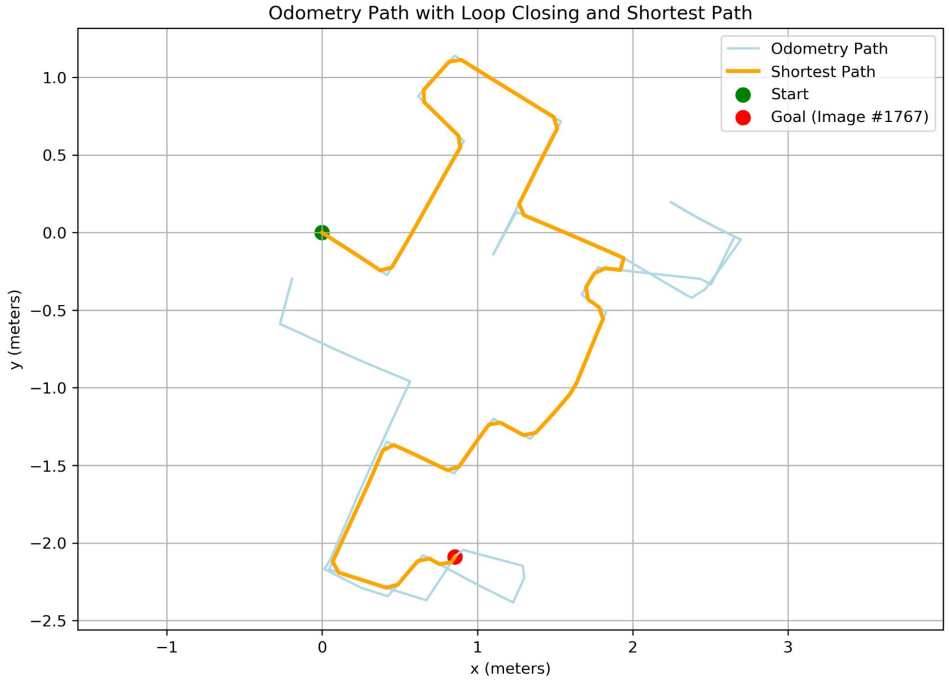
Publishes velocity commands (`/cmd_vel`) for:

- Smooth **angular turns** (ramp-up strategy)
- Straight-line **forward motion**

Executes the **full navigation plan** to reach image location autonomously



Shortest Path



Thank You