Stereo Visual SLAM System in a CARLA Simulation Environment

Integration of Stereo Vision, IMU Fusion & 3D Mapping

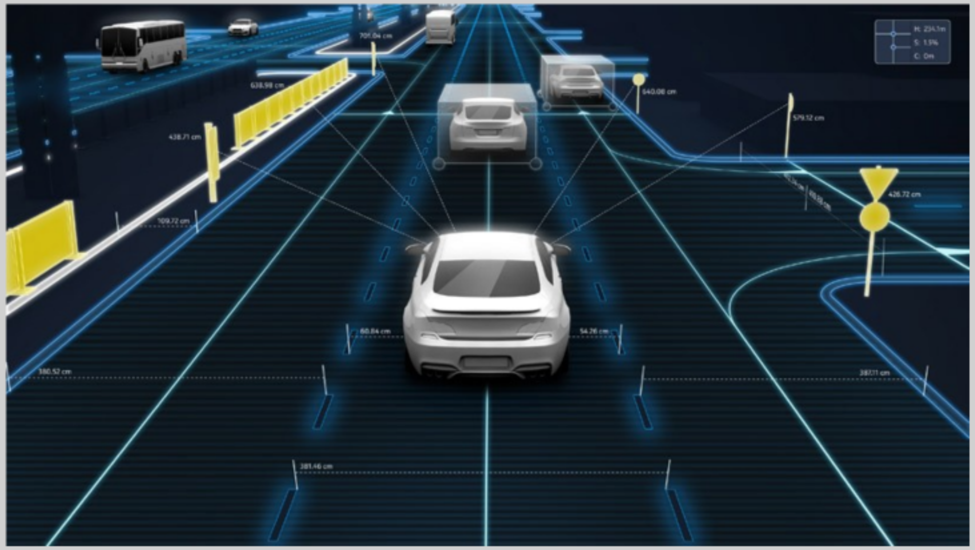
ECE 544 - Mobile Robots
Sankalp Rajeev

Introduction

Visual SLAM is the process of building a map and estimating trajectory using camera input

This project implements a full Stereo Visual SLAM pipeline with:

- *Depth estimation from stereo*
- *IMU integration using a 9D Kalman Filter*
- *Pose estimation via PnP*
- *Loop closure detection*



Simulated in CARLA using a self-driving vehicle in urban scenes

Key tools & libraries:

- *Python, OpenCV, NumPy, CARLA Simulator, Open3D*

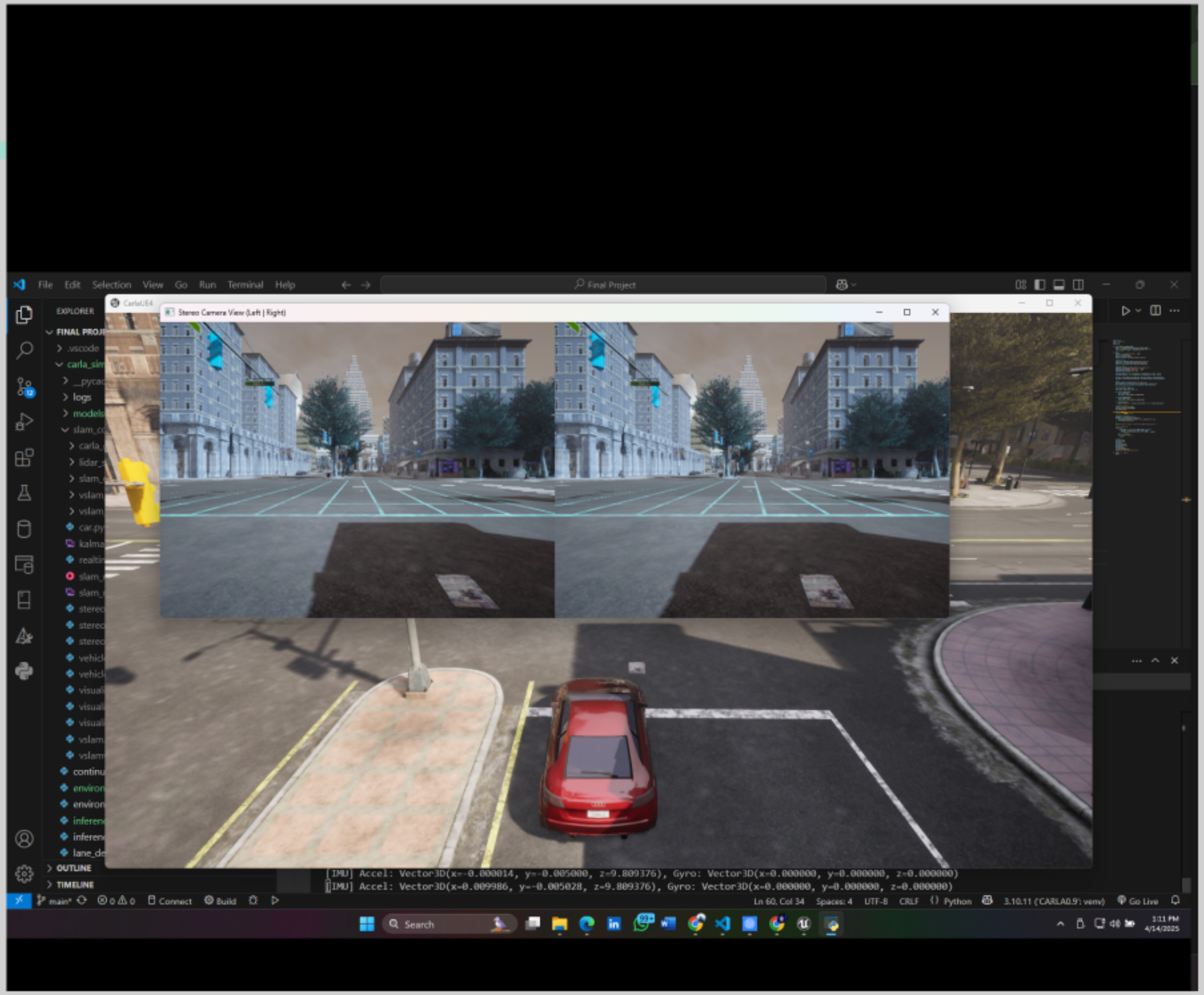
Vehicle Sensor Configuration in CARLA

Vehicle Setup:

- *Spawned a standard CARLA vehicle*
- *Enabled built-in autopilot*

Attached Sensors:

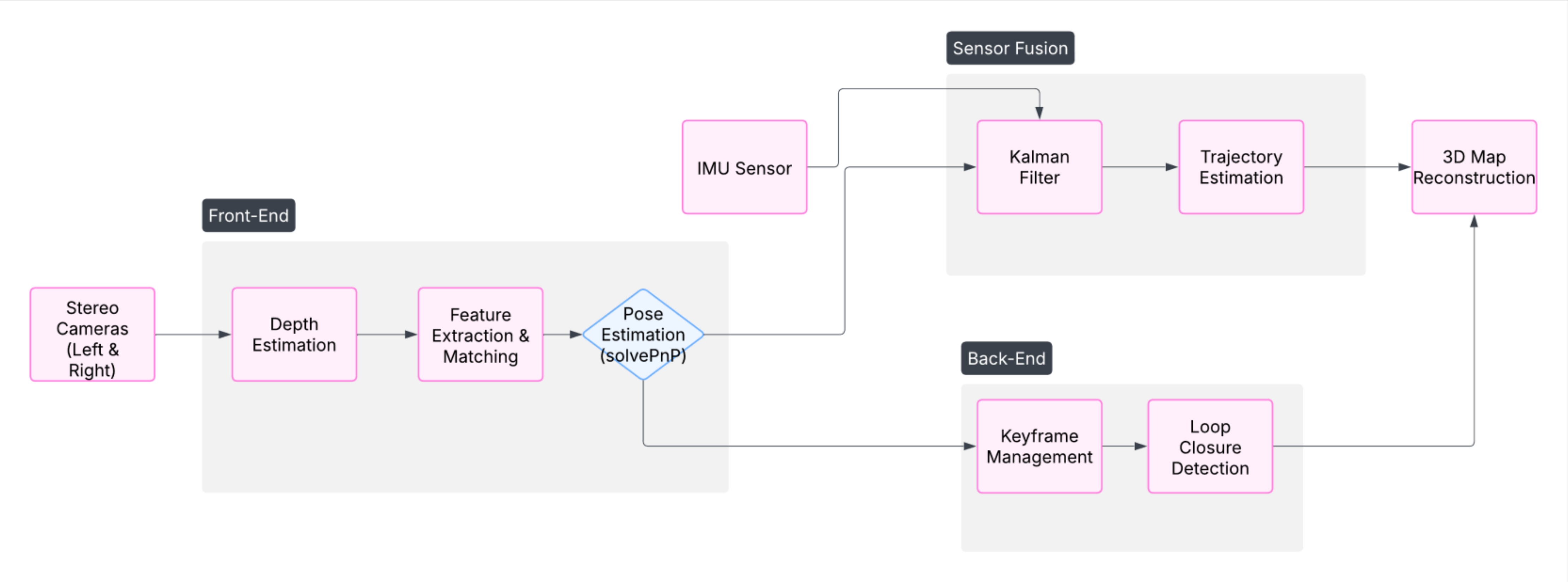
- *Left RGB Camera*
  - *Resolution: 640×480*
  - *Position: Front-left, focal baseline = 0.4 m*
- *Right RGB Camera*
  - *Synchronized with left for stereo vision*
- *IMU Sensor*
  - *Captures 3-axis acceleration and angular velocity*
  - *Used for Kalman Filter prediction*

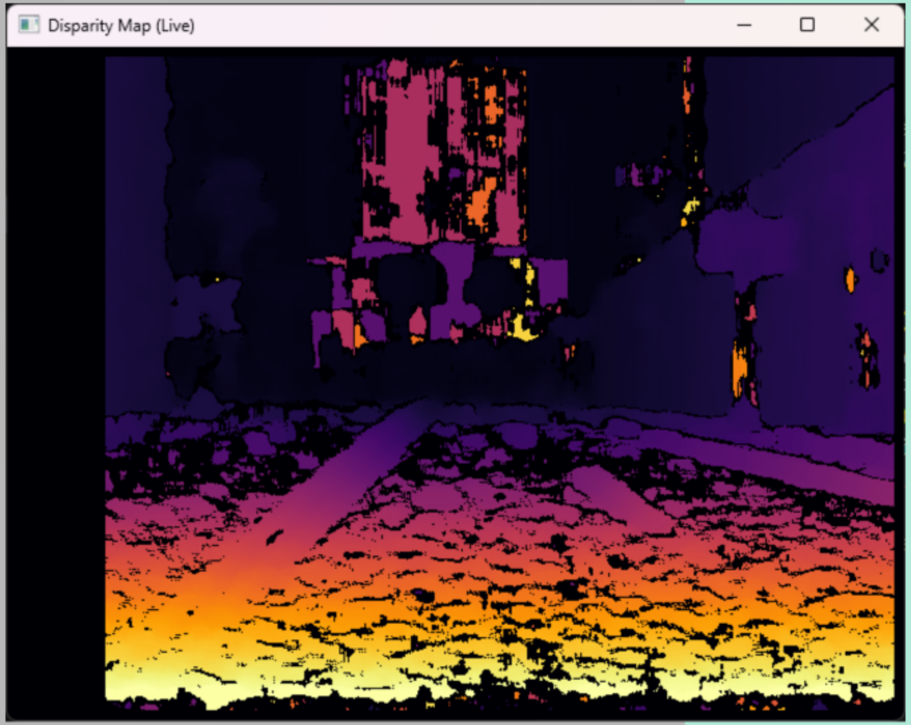

File Edit Selection View Go Run Terminal Help
Final Project
EXPLORER
FINAL PROJECT
Stereo Camera View (Left | Right)
OUTLINE
TIMELINE

```
[IMU] Accel: Vector3D(x=-0.000014, y=-0.005000, z=9.809376), Gyro: Vector3D(x=0.000000, y=0.000000, z=0.000000)
[IMU] Accel: Vector3D(x=0.009986, y=-0.005028, z=9.809376), Gyro: Vector3D(x=0.000000, y=0.000000, z=0.000000)
```

Ln 60, Col 34 Spaces: 4 UTF-8 CRLF Python
Search

System Architecture



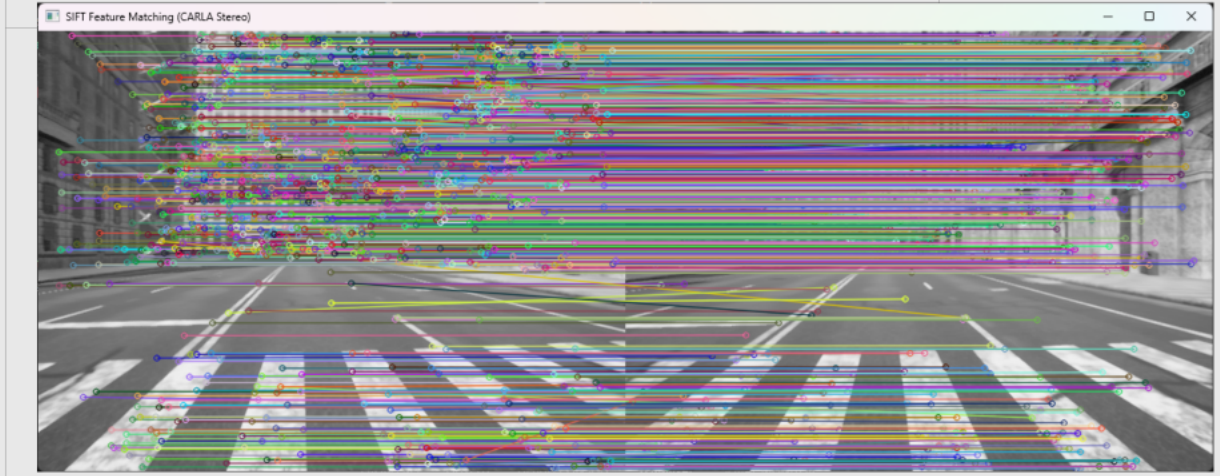


Depth Estimation Methods

*Depth estimation is achieved using StereoSGBM (Semi-Global Block Matching), which computes disparity maps from stereo images. The depth is calculated by the formula: Depth = (focal length * baseline) / disparity, allowing for accurate real-time calculations essential for spatial understanding.*

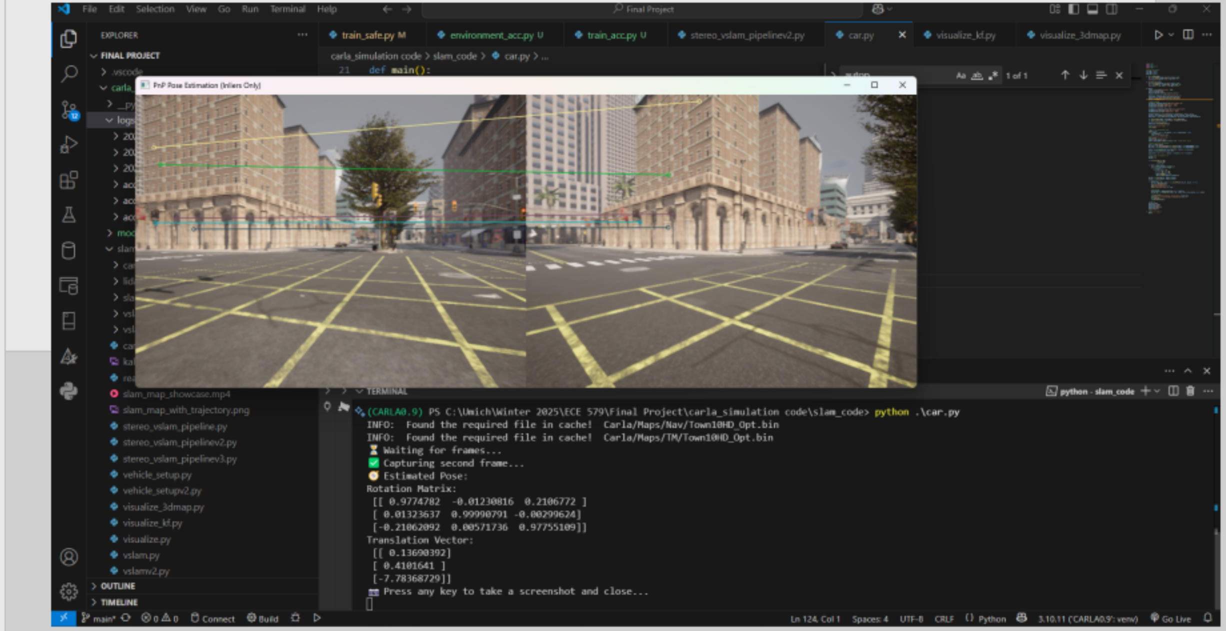
Feature Detection and Matching Algorithms

- *Keypoints extracted using SIFT and ORB*
- *Matched using FLANN-based matcher + KNN ratio test*
- *Ensures consistent tracking of features between frames*
- *Robust against scale and rotation changes*



Pose Estimation using PnP

- *Matched 3D–2D correspondences across frames*
- *Depth from stereo used to recover 3D points*
- *Pose computed using PnP with RANSAC*
- *Returns rotation (R) and translation (t) of the current camera*
- *Used to update trajectory in real-world coordinates*

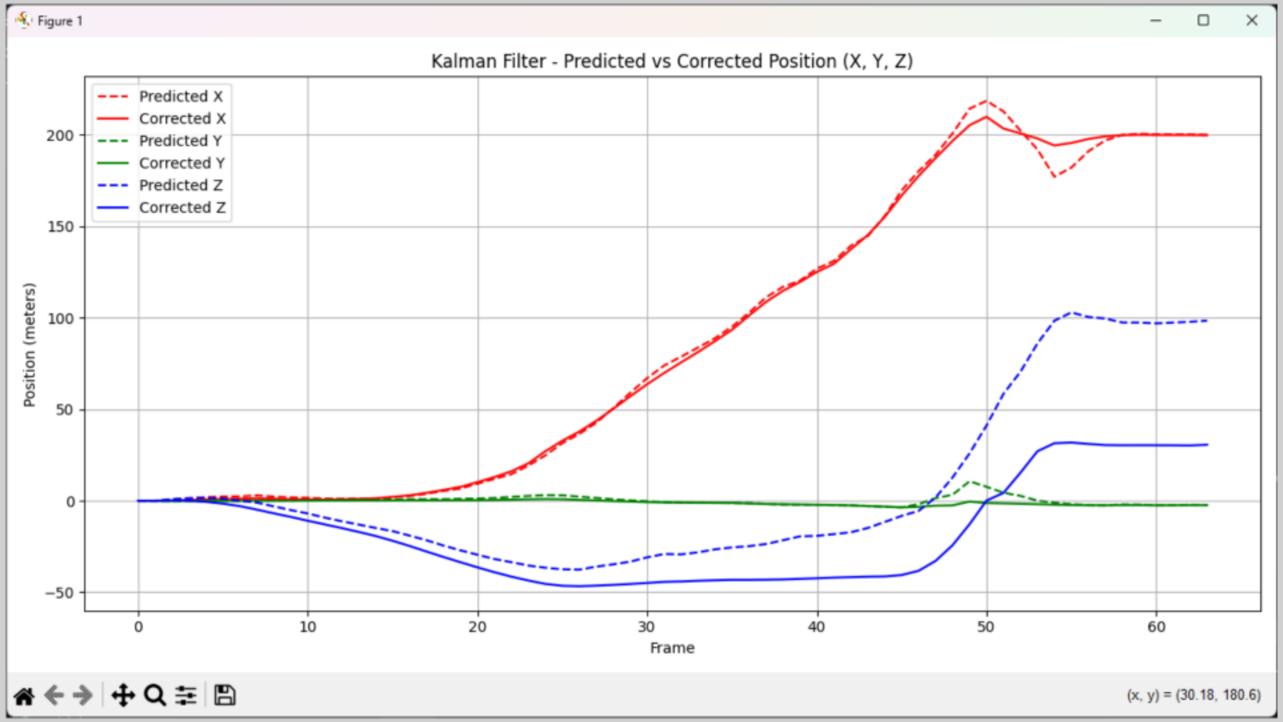


```
21    def main():
```

```
(CARLA0.9) PS C:\Umich\Winter 2025\ECE 579\Final Project\carla_simulation code\slam_code> python .\car.py
INFO:  Found the required file in cache!  Carla/Maps/Nav/Town10HD_Opt.bin
INFO:  Found the required file in cache!  Carla/Maps/TM/Town10HD_Opt.bin
⏳ Waiting for frames...
✅ Capturing second frame...
🧭 Estimated Pose:
Rotation Matrix:
 [[ 0.9774782  -0.01230816  0.2106772 ]
 [ 0.01323637  0.99990791 -0.00299624]
 [-0.21062092  0.00571736  0.97755109]]
Translation Vector:
 [[ 0.13690392]
 [ 0.4101641 ]
 [-7.78368729]]
⌨ Press any key to take a screenshot and close...
```

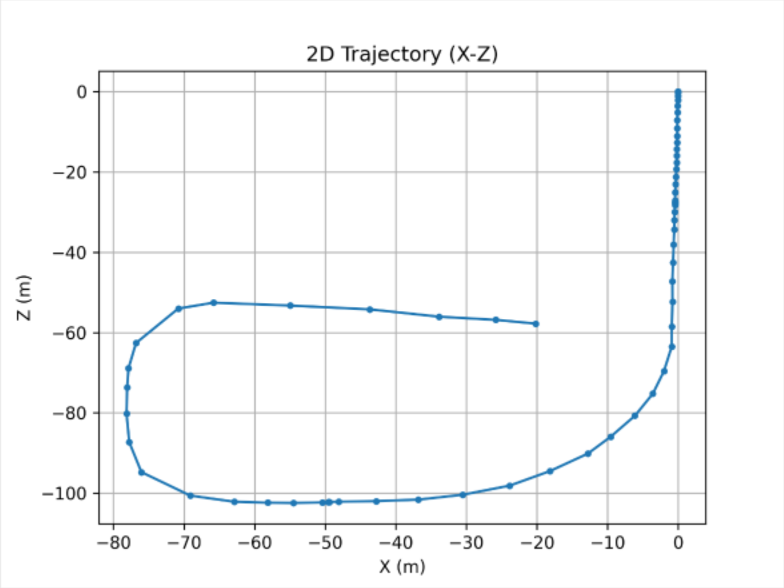
Kalman Filter Implementation

- *9D Kalman Filter fuses:*
  - *Linear acceleration (IMU)*
  - *Angular velocity (IMU)*
  - *Visual pose estimates (SLAM)*
- *Predicts position, velocity, orientation in real time*
- *Visual data used to correct drift from inertial prediction*



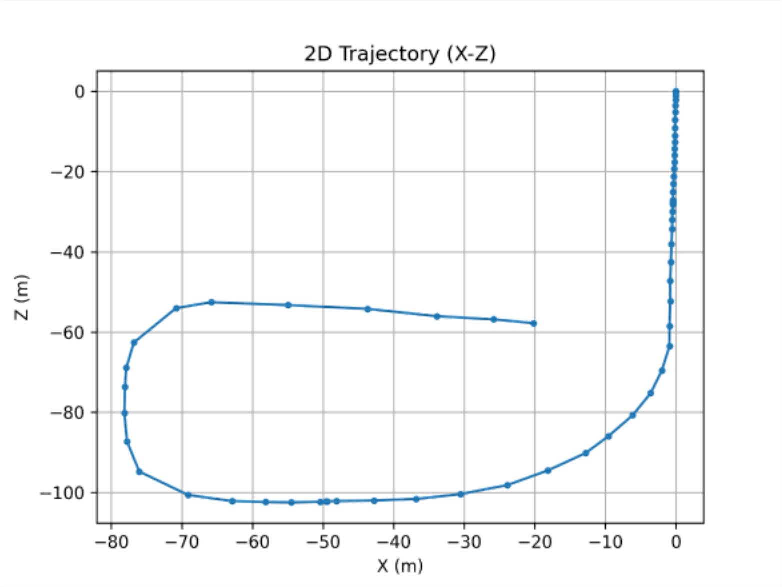
Keyframe Management and Loop Closure

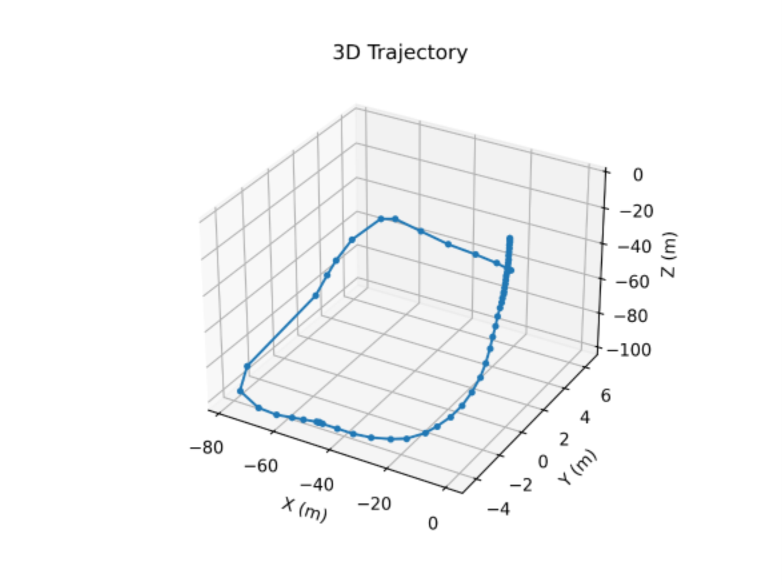
- Keyframes stored during significant motion
- Each new keyframe's descriptor is compared with all past keyframes
- If >30 good matches are found ; loop closure is flagged
- Enables drift detection when re-visiting previous locations
- Currently: loop is detected but not yet used for map correction



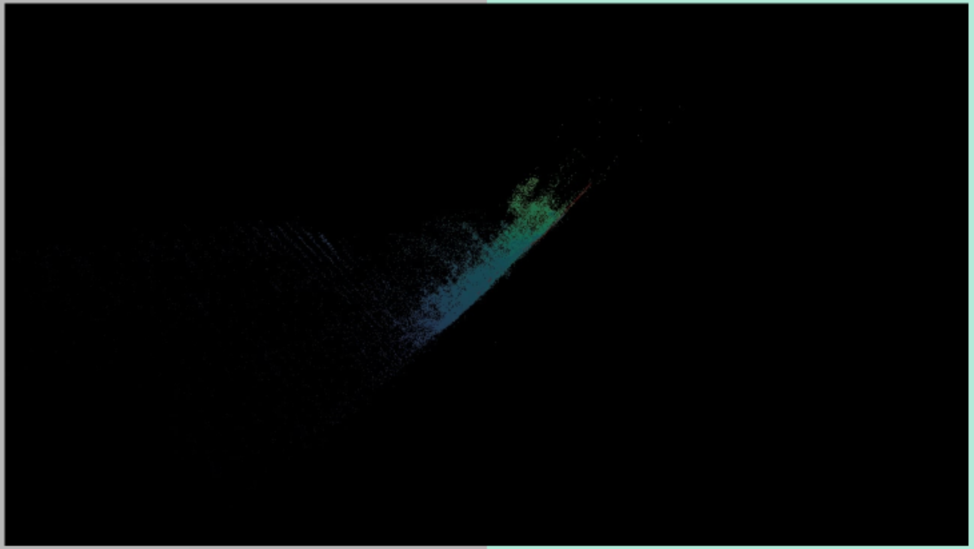
Visualization of 2D and 3D Trajectories

- *2D trajectory shows lateral pose drift*
- *3D trajectory confirms correct height tracking*
- *Consistent movement even without global optimization*


2D Trajectory (X-Z)


3D Trajectory

3D Point Cloud Mapping



- *Mapped environment using Open3D*
- *Colors indicate Z-height for clarity*
- *Data downsampled and cleaned for visualization.*

DEARBORN
CECS

Thank You!