Day - 25

1. Introduction

A method for achieving high-precision depth estimation combines Real-Time Kinematic (RTK) or Post-Processed Kinematic (PPK) Global Navigation Satellite System (GNSS) with stereo vision using two cameras. Here is the detailed steps to set up an experimental system that integrates these technologies to develop and test a depth perception system.

2. System Components

2.1 GNSS Components

- Two RTK-capable GNSS Modules: Recommended model: u-blox ZED-F9P.
- GNSS Antennas: Compatible with the GNSS modules.
- Radio Link: Optional for real-time corrections, but PPK can be used for post-processing if not available.
- Data Processing Unit: Laptop or Raspberry Pi for processing GNSS data.

2.2 Stereo Vision Components

- Two Identical USB Cameras: Alternatively, a stereo camera module like Arducam or ZED stereo camera.
- Rigid Camera Mounting Frame: Ensures a fixed and known baseline distance.
- Calibration Checkerboard: Printable checkerboard for stereo calibration.

2.3 Processing Setup

Computer with Required Software:

- Python with OpenCV, NumPy, and Matplotlib libraries.
- RTKLIB for GNSS data processing.
- Optional: ROS (Robot Operating System) for real-time data handling.

3. Step-by-Step Experimental Setup

3.1 GNSS Base Station Setup

- Set up one GNSS module in a fixed known position.
- Collect raw satellite data to provide differential corrections.

3.2 GNSS Rover Setup

- Mount the second GNSS module on a mobile platform (handheld or on a toy car).
- Connect the rover GNSS to a laptop or Raspberry Pi for real-time or post-processed data collection.

3.3 Camera Rig Assembly

- Mount two cameras on a rigid frame with a fixed distance between them (typically 10-20 cm).
- Ensure cameras are parallel and stable.
- Calibrate the stereo camera system using OpenCV to obtain intrinsic and extrinsic parameters, distortion coefficients, and disparity mapping capabilities.

3.4 Data Synchronization

- Start GNSS and camera data collection simultaneously for rough synchronization.
- For improved accuracy, consider using hardware triggers or timestamp alignment methods.

3.5 Data Collection Process

- Move the rover around objects within the environment (indoors or outdoors).
- Record synchronized GNSS raw data and stereo image sequences.

4. Data Processing and Integration

4.1 GNSS Data Processing

- Process GNSS data using RTKLIB to obtain centimeter-level position accuracy.
- If real-time correction is desired, set up an RTK base-rover communication link or use NTRIP corrections if available.

4.2 Stereo Vision Processing

- Use OpenCV to process stereo image pairs.
- Generate disparity maps and compute depth information from stereo images.
- Convert disparity data to 3D point clouds for object localization.

4.3 Data Fusion

- Merge GNSS position data with the 3D points from stereo vision to obtain precise object locations.
- Analyze the alignment and accuracy between stereo-derived depth and GNSS positional data.
- Visualize the combined data using 3D plotting libraries like Matplotlib.

5. Evaluation and Testing

5.1 Precision and Accuracy Assessment

- Measure the precision of stereo depth perception independently.
- Compare the stereo system measurements against the GNSS-derived positions.
- Evaluate error margins and consistency across different environmental conditions.

5.2 Parameter Variation Testing

- Change stereo camera baseline distance to observe effects on depth resolution.
- Experiment with different GNSS update rates and correction methods.
- Analyze how camera resolution and lighting affect stereo processing accuracy.

6. Potential Extensions

- Integrate Inertial Measurement Units (IMUs) for enhanced accuracy during GNSS signal loss.
- Explore visual odometry methods for improving trajectory tracking.
- Implement real-time data fusion using ROS for dynamic applications.
- Extend the setup to outdoor environments with more complex objects and obstacles.

7. References

- RTKLIB: https://github.com/tomojitakasu/RTKLIB
- u-blox ZED-F9P Documentation: https://www.u-blox.com/en/product/zed-f9p-module
- OpenCV Stereo Calibration: https://docs.opencv.org/4.x/d9/d0c/group__calib3d.html
- Stereo Vision with OpenCV Example: https://docs.opencv.org/4.x/dd/d53/tutorial_py_depthmap.html