PROJECT REPORT – VISUAL INERTIAL ODOMETRY FOR AUTONOMOUS DRONE NAVIGATION

Introduction : Odometry in Robotics refers to estimating not only the distance traveled, but the entire trajectory of a moving robot. In Visual Odometry (VO) we have a camera rigidly attached to a moving object (such as a car or a drone), and our job is to construct a 6 degree of freedom trajectory using the video stream coming from this camera(s). First corner points are detected in an image (FAST algo), then these points are tracked (KLT algo) and then projected to virtual 3D environment(point cloud) from where position is estimated using geometric transformations. There are two types of Visual Odometry namely Monocular VO and Stereo VO. For Stereo VO, a calibrated stereo camera pair is used which helps compute the feature depth between images at various time points. Computed output is actual motion (on scale). If faraway features are tracked then it degenerates to monocular case and needs IMU signal to estimate depth. Also for mini bots when relative camera distance is very less compared to distance of image. Visual Inertial Odometry (VIO) is the combination of IMU in the Visual Odometry system in which the visual information and inertial measurements are combined (using Extended Kalman Filter) to achieve an accurate measurement. There are two ways to fuse the visual and inertial measurement namely loosely coupled approach and tightly coupled approach. The loosely coupled approach separately estimates the image motions and inertial measurements, and then fuses these two estimates to obtain the final estimate. The tightly coupled approach fuses the visual and inertial data directly at the measurement level to jointly estimate all IMU and camera states. Generally, the tightly coupled approach is better in term of accuracy and robustness in motion estimation but less energy efficient.

Methodology :

The MARS logger records camera frames at 16.5Hz and IMU readings at 50+ Hz from smartphone and the two data streams are synced to one clock. Relative frequency is hardware dependent.

- frame_timestamps.txt
- gyro_accel.csv
- movie_metadata.csv
- movie.mp4

The frame_timestamps contains timestamp of each recorded video frame, the gyro_accel.csv and movie_metadata.csv contains the IMU and VO data respectively.

ROS (Robot Operating System) is an open source software development kit for robotics applications. ROS offers a standard software platform to developers across industries that will carry them from research and prototyping all the way through to deployment and production. ROS uses data format called bag. The collected data from the android app needs to be converted to bag file (using Kalibr library) in order to find a path through the camera.

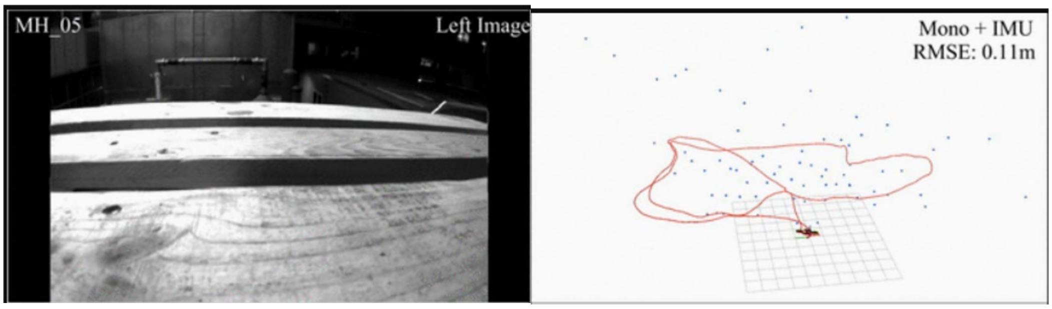
```
[ ] b4 = bagreader('/content/drive/MyDrive/bag file/demo.bag')
    b4.topic_table

[INFO]  Successfully created the data folder /content/drive/MyDrive/bag file/demo.
```

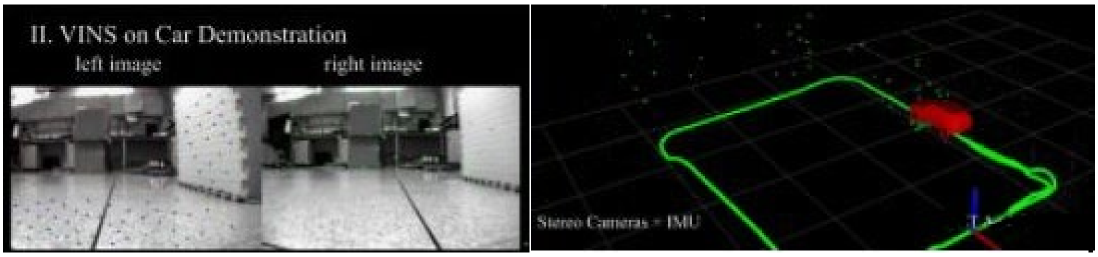
	Topics	Types	Message Count	Frequency
0	/cam0/image_raw	sensor_msgs/Image	270	16.529472
1	/imu0	sensor_msgs/Imu	832	50.646059

In the bag data file we can see that the frequency of IMU data is nearly 3 times that of the camera data.

ROS is launched from terminal. First a particular configuration is loaded then the required bag file is feeded. Cases of monocular VIO and stereo VIO are shown below :



Monocular VIO for drone dataset



Stereo VIO for car dataset

Observation :

Visual Inertial Odometry gives a very good estimation of the trajectory of the system. The sample datasets came up with ground truth values by laser tracking in indoor environment which we see almost as a superimposition. Also the plot is real time using the recent point cloud which demonstrates its feasibility in real-world applications like augmented reality based games. However running VIO on low power hardware may not be feasible because of heavy computation.

Challenges and Future Work :

Apart from computational complexity, VIO has other challenges like camera caliberation (dependent on camera hardware like focal length, aperture, ..), initial state estimation (important for accurate trajectory) and dataset creation. Thus, VIO implementation is hardware dependent and thus different for each such type.

Future work involves streaming of sensor values directly from a ROS node to simulate trajectory in real time. Also barometer can be used to aid in initial state estimation.