ECE 276A Project 1: Orientation Tracking

Winston Chou
PID: A17460970

***Abstract*—This paper presents a post-analysis optimization approach for processing sensor data collected from Inertial Measurement Units (IMUs) and cameras. The study focuses on refining sensor readings and estimating the orientation of a rigid body based on the sensor setup. A gradient descent method is employed to optimize quaternion readings, while a Motion Model and Observation Model are utilized to determine sensor orientation. The processed orientation data is then used to generate a cylindrical projection panorama from multiple camera frames. The results demonstrate improved accuracy in orientation tracking and panoramic image creation, showcasing the effectiveness of the proposed methods in sensor fusion and computer vision applications.**

Keywords—orientation tracking, sensor fusion, IMU, quaternions, optimization, gradient descent, panorama

I. Introduction

Orientation tracking is crucial tasks in robotics, computer vision, and augmented reality applications. This paper addresses the challenge of extracting rotations from Inertial Measurement Unit (IMU) data and utilizing this information with camera images to generate a comprehensive panorama of the surrounding environment.

The experimental setup consists of three key sensors: an IMU capturing angular velocity and linear acceleration, a VICON sensor providing ground truth rotation matrices, and a camera recording images corresponding to the setup's orientation. Each sensor reading comes with a timestamp, enabling precise temporal alignment of the data.

A. Our approach involves several steps

1) Quaternion Estimation: Calculate quaternions using the IMU's angular velocity data and the time difference between measurements, employing a motion model to predict subsequent quaternion values.

2) Observation Model: Utilize an observation model to predict sensor measurements based on the current state. The IMU's acceleration values are used to construct this model, assuming the body undergoes pure rotation and experiences an acceleration of [0, 0, −g].

3) Gradient Descent Optimization: The estimated quaternions are refined using a gradient descent method. This process compares predicted acceleration values with actual IMU data and incorporates smoothing based on the difference between consecutive orientation values.

4) Rotation Matrix Calculation: The optimized quaternions are used to compute rotation matrices, which are essential for mapping the camera's view to a panoramic image.

5) Image Projection: The camera's field of view (60° horizontally, 45° vertically) is converted into cartesian coordinates using spherical coordinate system. Then, it is projected onto a cylindircal suface wrapping around the sphere using the calculated rotation matrices. This process involves coordinate system transformation and image projection.

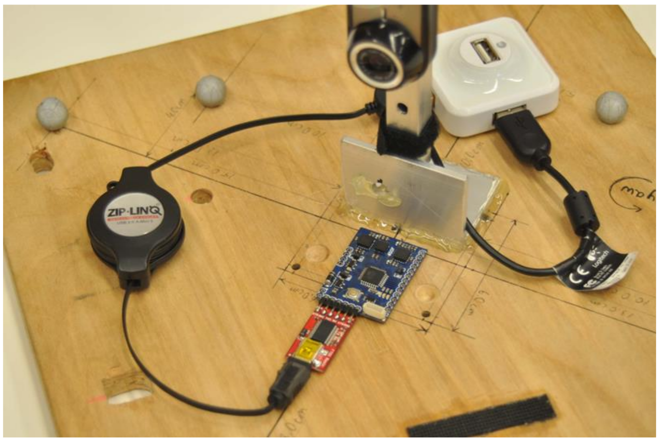

Fig. 1. Sensor Setup. A ZIP-LINKS retractable cable management system (black circular device), connected to a blue development board and a red USB-to-Serial converter. A white USB hub provides power and connectivity. Several gray VICON tracking markers are strategically placed around the setup for precise motion capture. [Source: ECE276A_PR1.pdf]

The significance of this work lies in its ability to accurately reconstruct environments using sensor data and camera images while minimizing noise in IMU readings. By combining angular velocity and linear acceleration data, we achieve more precise rotation matrices, resulting in higher-quality panoramic reconstructions.

II. Problem Formulation

The orientation tracking problem can be formulated using three key mathematical models:

A. Motion Model

The system uses quaternions to represent body-frame orientation, where q_t represents the orientation at time t. The motion model predicts the next quaternion q_{t+1} using:

$$q_{t+1} = f(q_t, \tau_t\omega_t) := q_t \circ e^{\left[0, \frac{\tau_t\omega_t}{2}\right]} \qquad (1)$$

where ω_t is the angular velocity measured from the IMU, and τ_t is the time difference between measurements. The initial quaternion q_0 is set to [1, 0, 0, 0].

B. Observation Model

The linear acceleration prediction is calculated using:

$$[0, a_t] = h(q_t) := q_t^{-1} \circ [0,0,0,-g] \circ q_t \qquad (2)$$

where q_t is the quaternion at time t, and gravity in world-frame z axis, which $g = 9.81\ m/s^2$.

C. Cost Function

The optimization problem is defined by a cost function that combines two terms:

$$c(q_{1:T}) = \frac{1}{2}\Sigma_{t=0}^{T-1}\left\|2\log\left(q_{t+1}^{-1} \circ f(q_t, \tau_t\omega_t)\right)\right\|_2^2 + \frac{1}{2}\Sigma_{t=1}^{T}\|a_t - h(q_t)\|_2^2 \qquad (3)$$

where the first term measures the error between the estimated orientation and the motion model prediction, and the second term measures the error between the acceleration measurements and the observation model prediction. ω_t is the angular velocity measured from the IMU, a_t is the linear acceleration measured from the IMU, and τ_t is the time difference between measurements.

D. Gradient Descent Optimization

The quaternions are optimized iteratively using:

$$q_{k+1} = q_k - \alpha \nabla c(q_{1:k}) \tag{4}$$

where α is the step size and k represents the iteration number. This optimization process aims to minimize noise in the IMU data by incorporating both angular velocity and linear acceleration measurements.

When minimizing $c(q)$, we need to keep q_t as a unit quaternion. Hence, we have a constrained optimization problem:

$$\min_{q_{1:T}} c(q_{1:T}) \qquad s.t. \|q_t\|_2 \tag{5}$$

III. Technical Approach

The implementation of the orientation tracking is presented as follows:

A. Sensor Data Preprocessing

The preprocessing of IMU sensor data begins with a systematic bias removal process. My initial approach attempted to dynamically determine the stationary period by analyzing VICON data, where a stationary state would be indicated by the rotation matrix approximating an identity matrix. To account for sensor noise, I implemented a threshold condition: $Error\ Threshold\ * I \geq I - R$ to check for all dataset, including the testset. However, this dynamic approach was simplified by adopting a fixed window of the first 300 samples, which proved sufficient for capturing the stationary state of the setup.

During this stationary period, we compute the average angular velocity and linear acceleration values. These averaged values represent the sensor bias and are subsequently subtracted from all sensor measurements to obtain clean, unbiased readings.

$$value = (raw - bias) * scale\ factor$$

$$scale\ factor = V_{ref}/1023/sensitivity$$

where we can find corresponding scale factors and sensitivities for accelerometer and gyroscope in "IMU_reference.pdf" and IMU specification sheets.

The acceleration measurements undergo further calibration steps to ensure accurate representation of the physical motion. The acceleration data is first converted to standard gravity units by applying a multiplication factor of $g = 9.81\, m/s^2$. To address the directional mismatch inherent in the sensor setup, the measurements along all three axes (x, y, and z) are inverted through multiplication by -1. Since the experimental setup undergoes pure rotation, the only acceleration experienced by the system is the downward gravitational force. Therefore, we incorporate this effect by adding a $-g$ term to the z-coordinate acceleration measurements. This comprehensive preprocessing pipeline ensures the sensor data accurately reflects the physical motion of the system while minimizing the impact of systematic errors and biases.

B. Quaternion and Acceleration Computation

The quaternion computation begins with calculating the exponential of the angular velocity and time duration product. Using equation 1 (Motion Model), I process the corrected angular velocity data to obtain predicted quaternions. These quaternions are then utilized to compute predicted acceleration according to equation 2 (Observation Model), which involves quaternion multiplication with the gravity vector.

From the quaternion values, I extract Euler angles using the Transforms3d library, which enables comparison with the VICON data ground truth for datasets 1 through 9. The comparison plot demonstrates the relationship between calculated Euler angles and VICON measurements across time steps. The visualization shows both the computed acceleration values and Euler angle comparisons, providing a comprehensive view of the system's orientation estimation accuracy.

C. Gradient Descent Optimization

The optimization process employs the cost function defined in equation 3 to refine the quaternion values, bringing them closer to the sensor's measured acceleration. I implemented a vectorized version of the gradient descent algorithm (equation 4) that efficiently completes 300 iterations within a 9-second timeframe. The optimized quaternions undergo another round of acceleration and Euler angle calculations.

D. Panorama Creation

The panorama creation begins with processing timestamped camera images and their corresponding rotation matrices to arrange them in world-frame coordinates. The camera's specifications include a 60° horizontal and 45° vertical field of view, with images sized at 320×240 pixels. The process starts by establishing a spherical coordinate system using a 2D matrix. The horizontal range is divided into 320 parts between 60° of view, while the vertical range spans 240 parts between 45° of view. These spherical coordinates are then transformed into Cartesian coordinates using standard transformation equations.

$$x = \rho \cos(\theta) \sin(\phi)$$

$$y = \rho \sin(\theta) \sin(\phi)$$

$$z = \rho \cos(\phi)$$

The rotation matrices are applied to map each image to its corresponding world frame coordinates.

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}_{\{w\}} = R_{wb} \begin{bmatrix} x \\ y \\ z \end{bmatrix}_{\{b\}}$$

E. Cylindrical Projection:

The world frame coordinates undergo conversion to a spherical coordinate system using three key equations:

1) The radius (ρ) is calculated as the square root of the sum of squared x, y, and z coordinates

$$\rho = \sqrt{x^2 + y^2 + z^2}$$

2) The azimuthal angle (θ) is determined using the arctangent of y/x

$$\theta = \arctan\left(\frac{y}{x}\right)$$

3) The polar angle (ϕ) is computed using the arccos of z divided by ρ

$$\phi = \arccos\left(\frac{z}{\rho}\right)$$

The panorama is rendered at 1920×1080 Full HD resolution. A scaling factor transforms the θ and ϕ values into Cartesian coordinates, with ρ maintained at 1.0 to place the image plane at unit distance from the observation point. The horizontal coordinates are scaled by $\frac{1920}{2\pi}$, while vertical coordinates use $\frac{960}{\pi}$. Due to Python's np.arctan2 function returning values between -π and π, an additional π is added to the longitudes for proper alignment. These transformations result in the final panoramic image projection.

IV. Results

A. *Orientation Tracking*

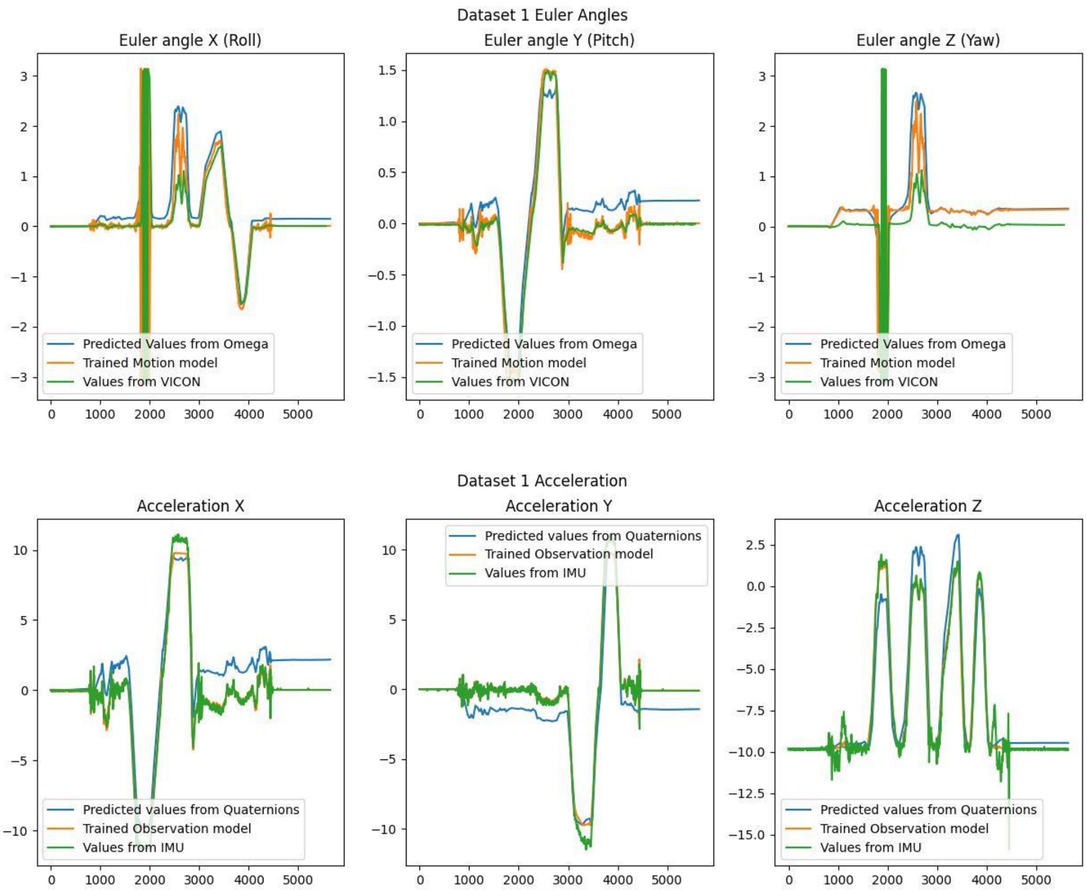

Fig. 2. Orientation & Acceleration of Dataset 1 (train set)

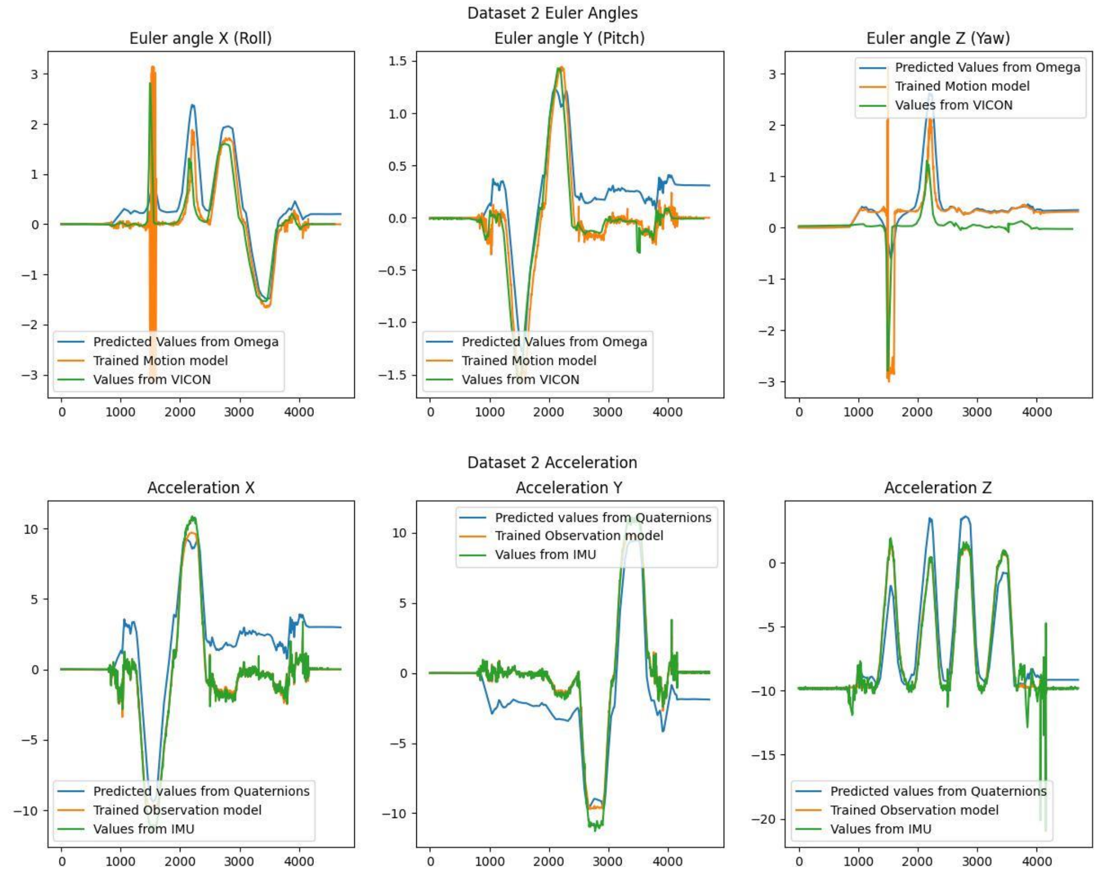

Fig. 3. Orientation & Acceleration of Dataset 2 (train set)

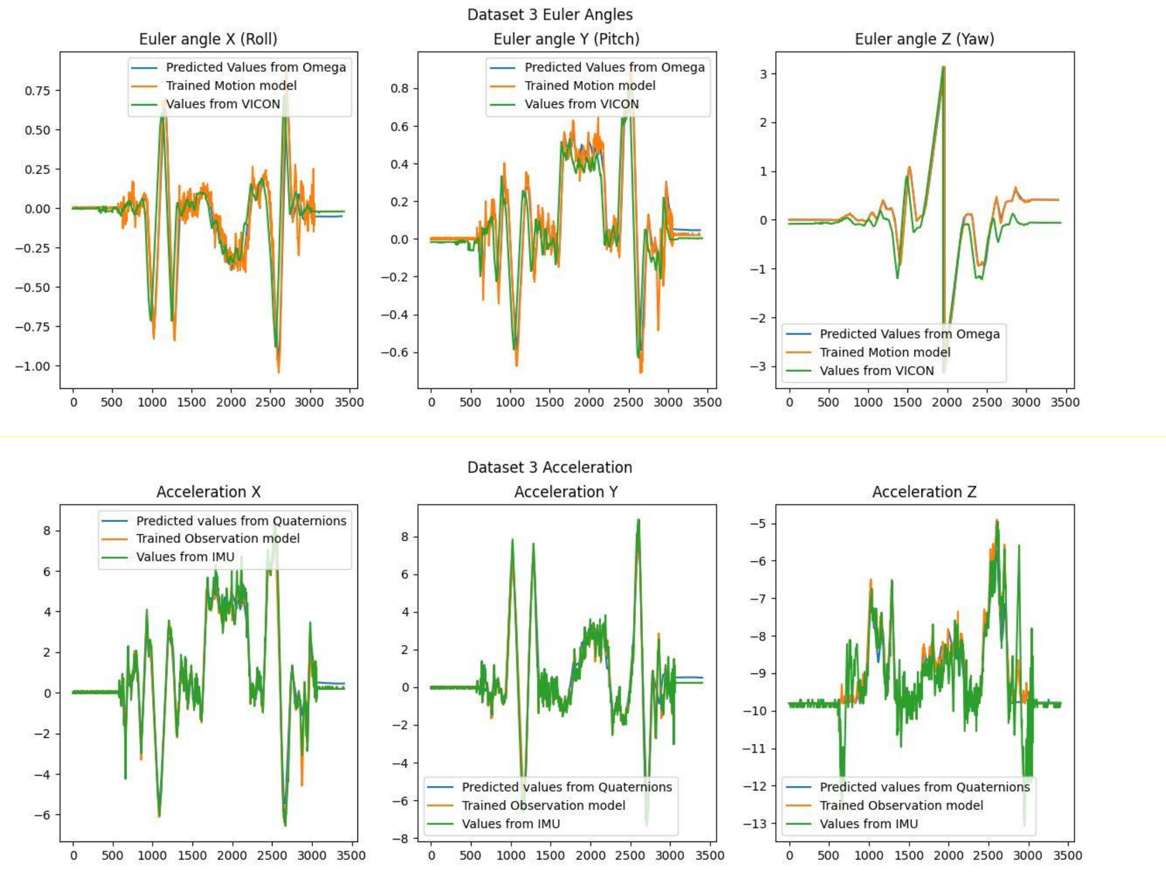

Fig. 4. Orientation & Acceleration of Dataset 3 (train set)

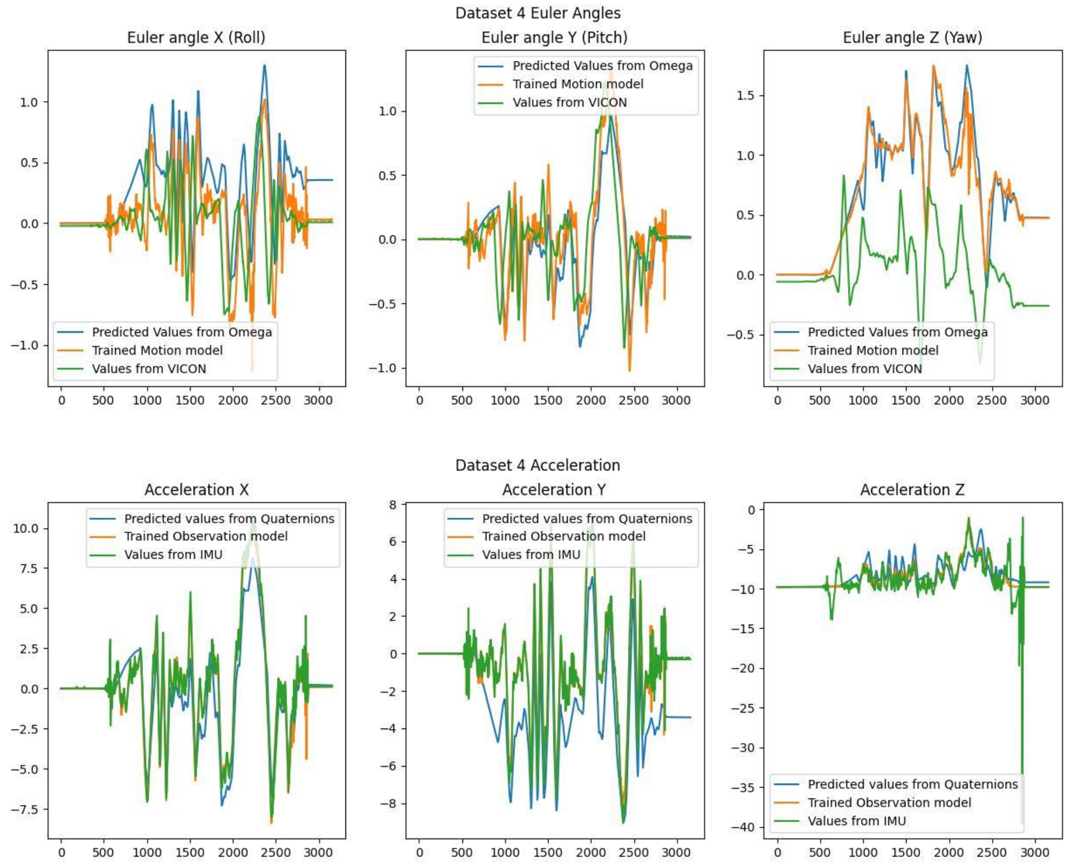

Fig. 5. Orientation & Acceleration of Dataset 4 (train set)

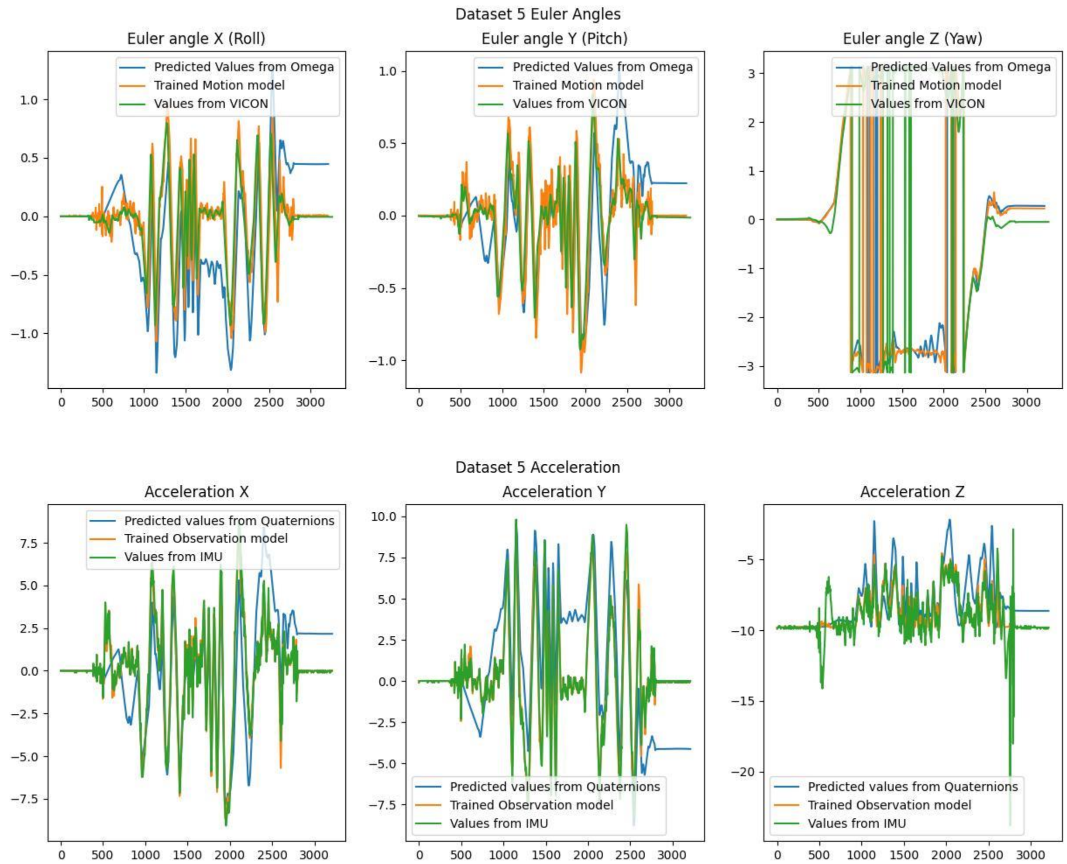

Fig. 6. Orientation & Acceleration of Dataset 5 (train set)

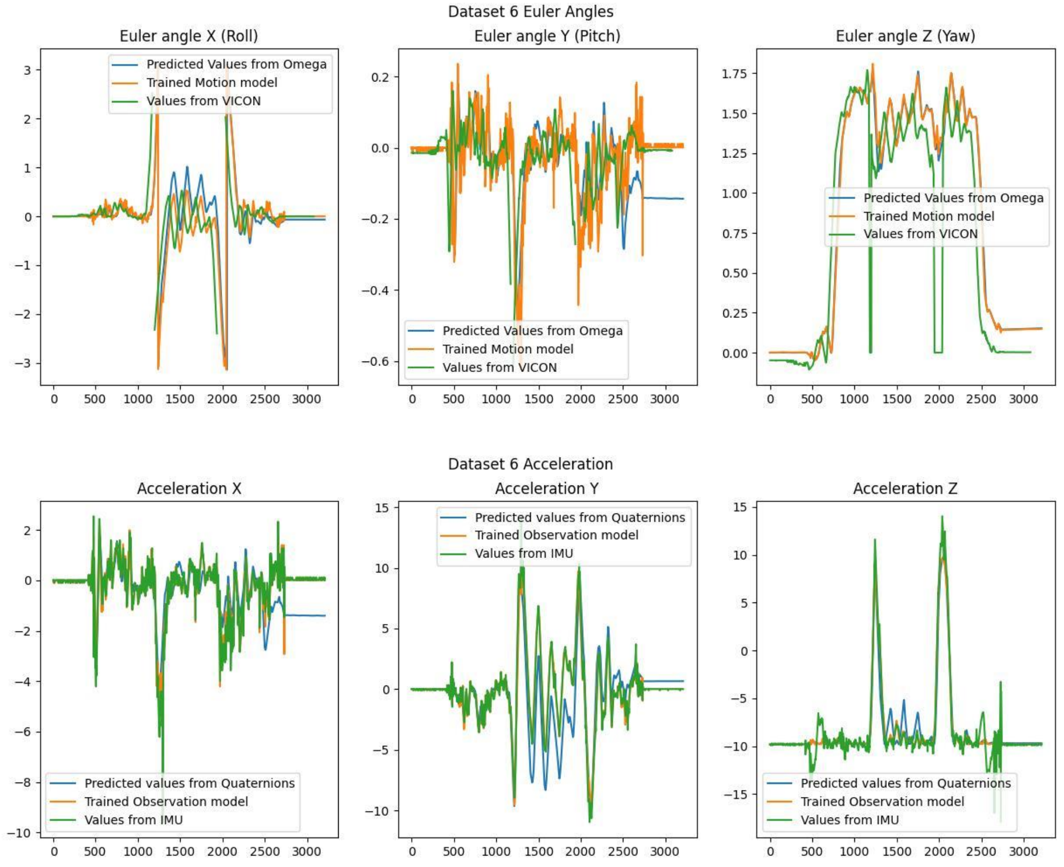

Fig. 7. Orientation & Acceleration of Dataset 6 (train set)

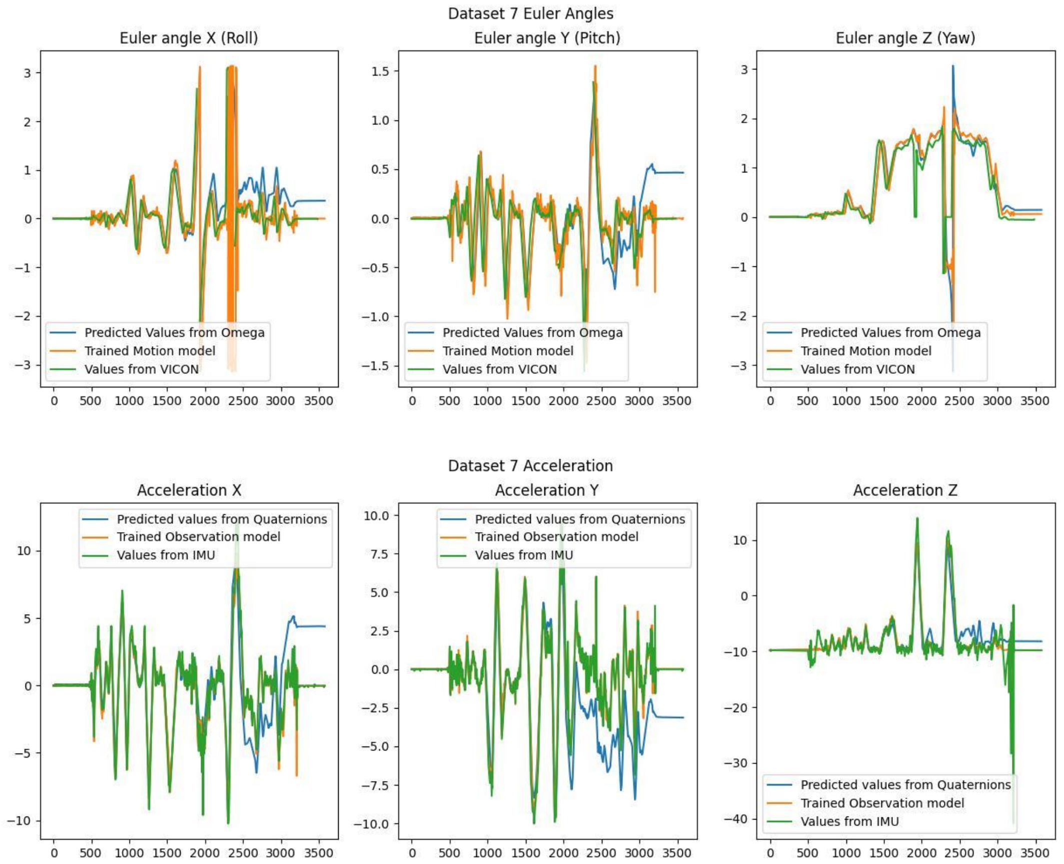

Fig. 8. Orientation & Acceleration of Dataset 7 (train set)

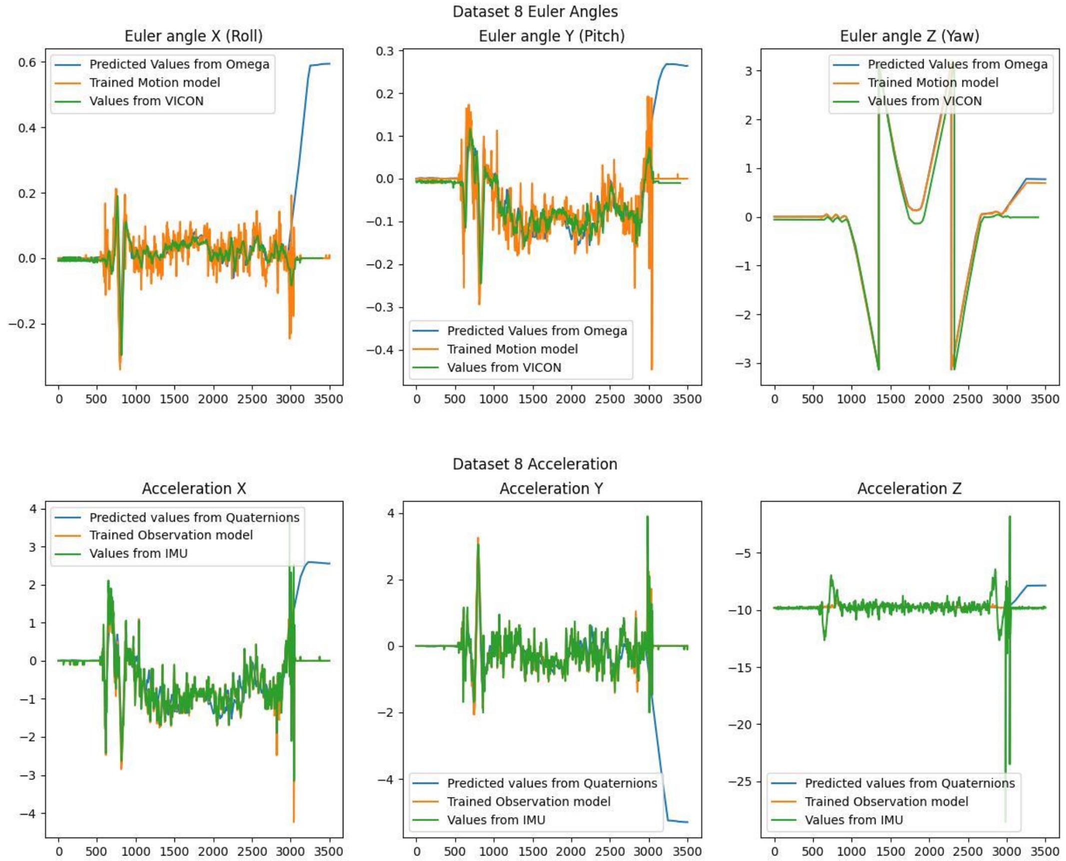

Fig. 9. Orientation & Acceleration of Dataset 8 (train set)

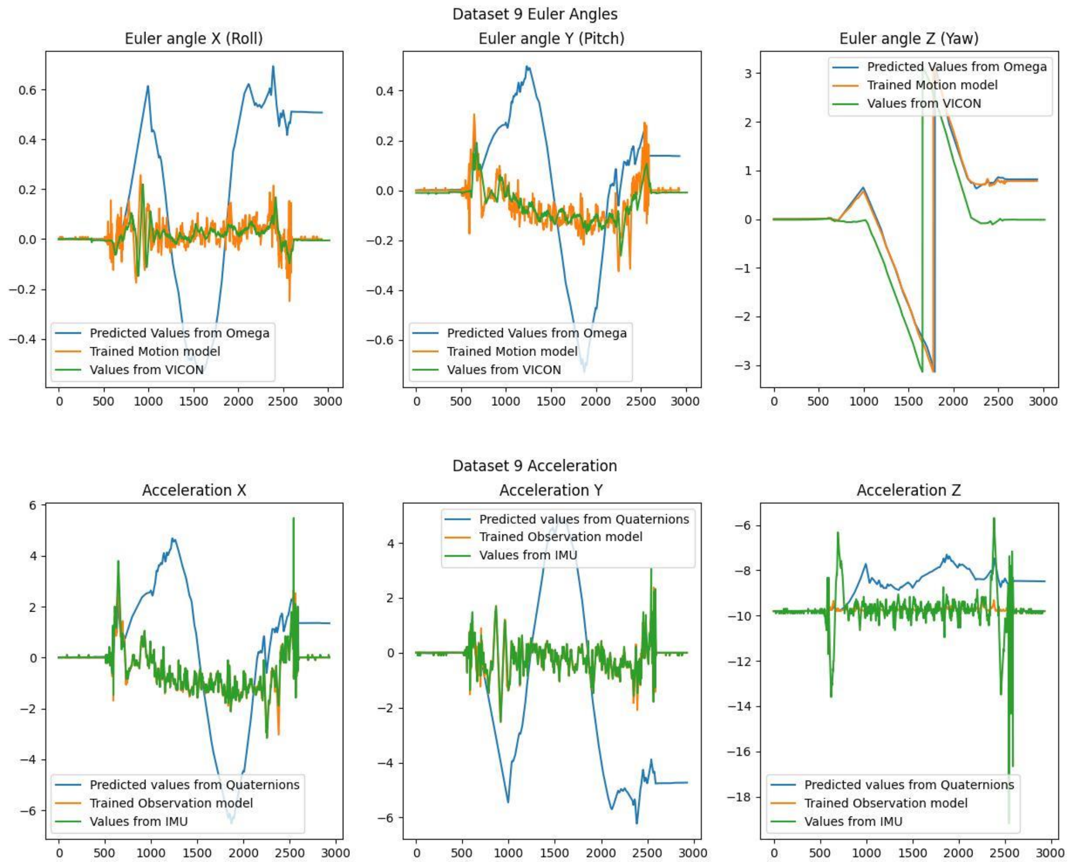

Fig. 10. Orientation & Acceleration of Dataset 9 (train set)

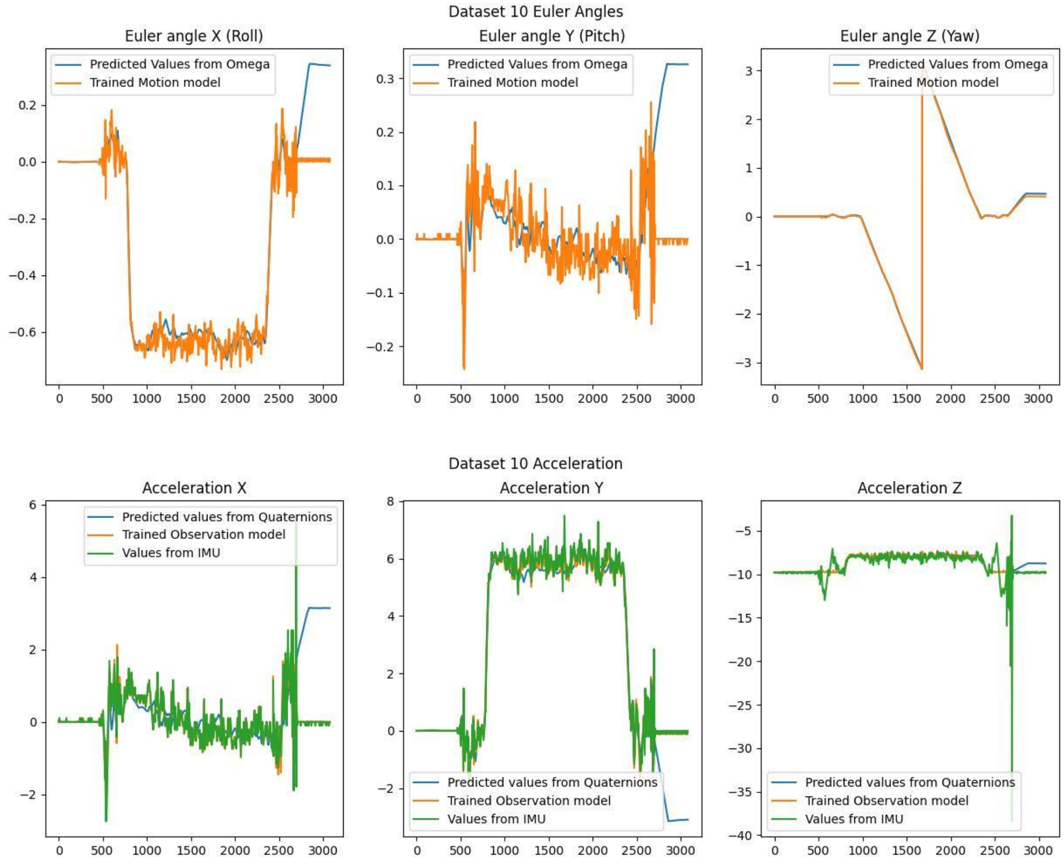

Fig. 11. Orientation & Acceleration of Dataset 10 (test set)

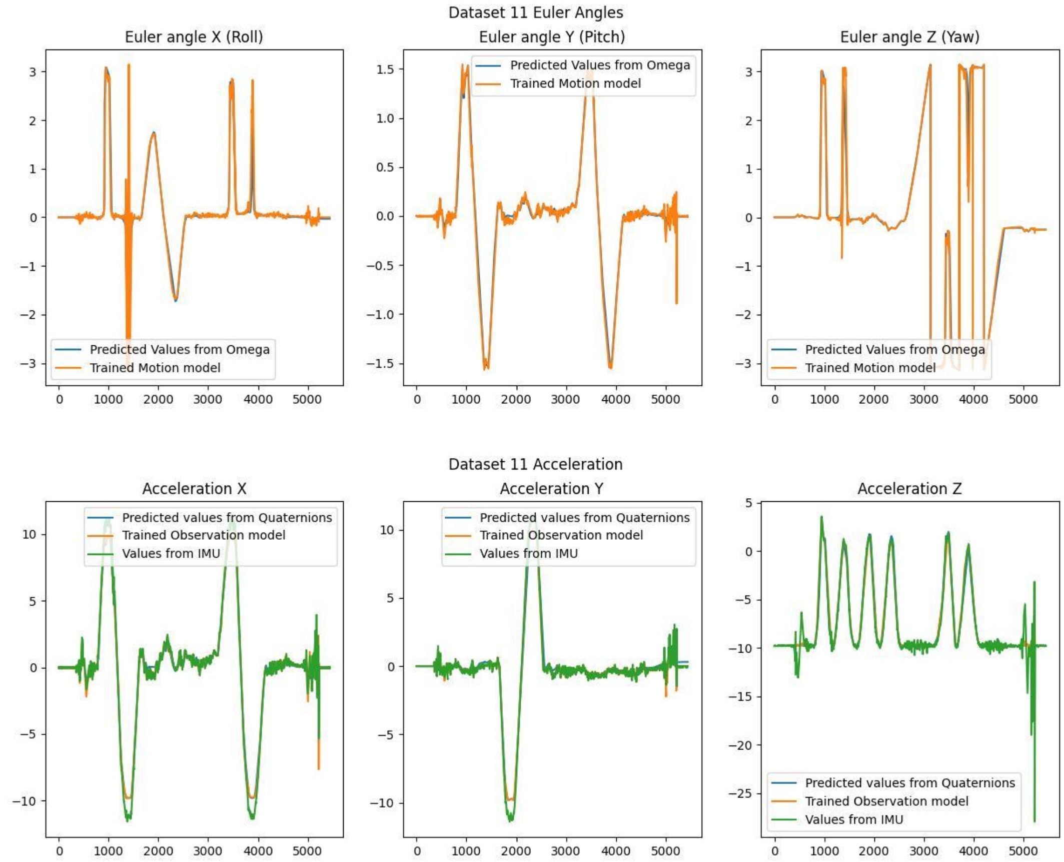

Fig. 12. Orientation & Acceleration of Dataset 11 (test set)

B. Panorama

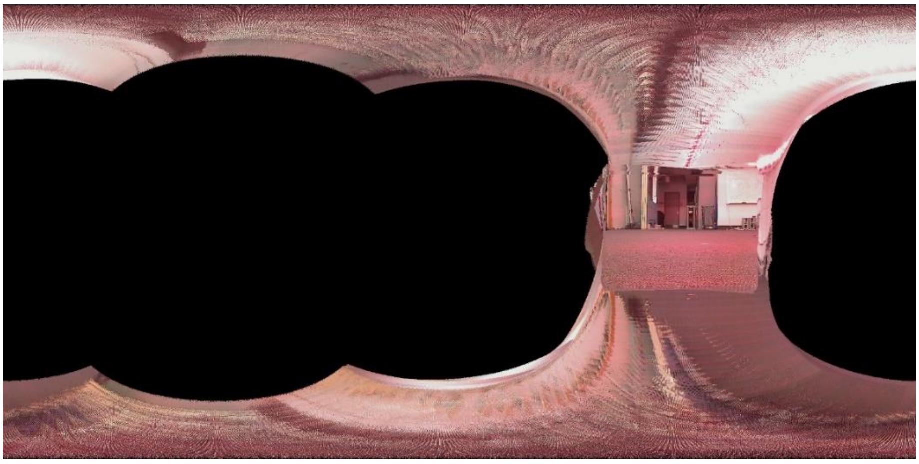

Fig. 13. Panorama of Dataset 1 (train set)

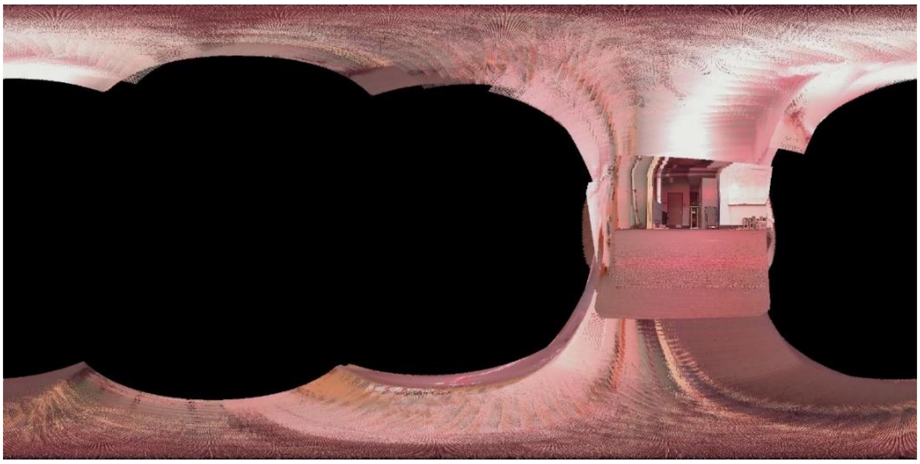

Fig. 14. Panorama of Dataset 2 (train set)

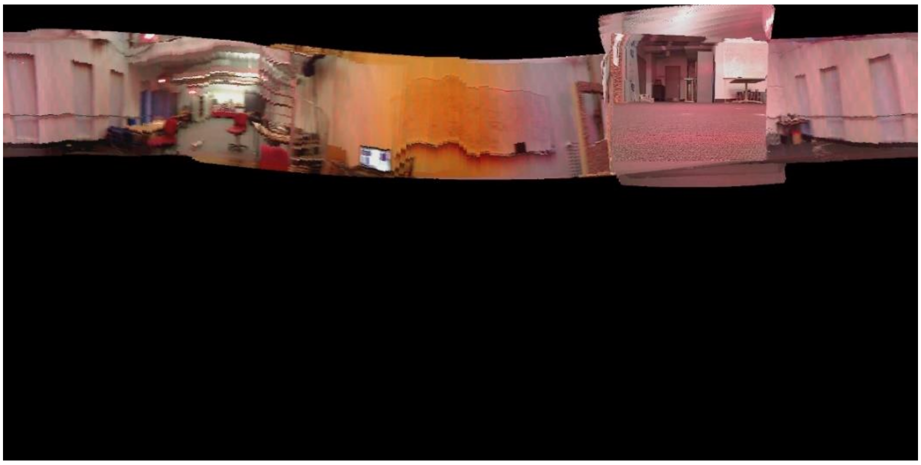

Fig. 15. Panorama of Dataset 8 (train set)

Fig. 16. Panorama of Dataset 9 (train set)

Fig. 17. Panorama of Dataset 10 (test set)

Fig. 18. Panorama of Dataset 11 (test set)

C. Result Analysis

Quaternion and Euler Angle Performance

The initial quaternion calculations produced Euler angles that followed the general trend of the VICON ground truth data, though with significant deviations. The acceleration measurements showed better accuracy even before optimization, with dataset 9 being an exception. After applying gradient descent optimization, which incorporated both IMU acceleration data and subsequent timestamp quaternions, the Euler angles demonstrated marked improvement, aligning much closer to the VICON reference data.

Optimization Performance

The gradient descent optimization proved particularly effective at addressing two types of errors: 1) Large and rapid deviations from ground truth, 2) Cumulative errors that build up over time. However, the algorithm showed limitations when handling high-frequency variations with small amplitudes, requiring increased iterations for accurate orientation estimation.

Future Improvements

While gradient descent provides effective orientation estimation and optimization, its post-analysis nature requires future state data, limiting real-time applications. Implementation of real-time filtering methods, such as Kalman filtering, could enable optimization during live operation, enhancing the system's practical utility.

References

[1] N. Atanasov, UCSD ECE276A: Sensing & Estimation in Robotics (Winter 2025), https://natanaso.github.io/ece276a/schedule.html.