Orientation Tracking and Panorama Stitching

Varun Sankar Moparthi
University of California San Diego
vmoparthi@ucsd.edu

Nikolay Antasnov
University of California San Diego
natanasov@ucsd.edu

***Abstract*—Accurate estimation of sensor orientation is critical for various applications in robotics, navigation, and computer vision. This problem focuses on leveraging data from an Inertial Measurement Unit (IMU) and cameras to refine angular velocity measurements and improve orientation estimation of a sensor setup. By fusing IMU and visual data, a rotation matrix is computed, representing the sensor's orientation. This rotation matrix is then mapped into the world coordinate frame using projections, ensuring accurate spatial representation. The projected images are subsequently stitched together to create a composite image that aligns with the calculated pixel values, preserving geometric consistency. The proposed methodology enhances the precision of orientation estimation and image reconstruction, contributing to improved sensor fusion techniques for real-world applications such as SLAM, augmented reality, and autonomous navigation.**

I. Introduction

Estimating orientation and position from sensor data is a fundamental problem in robotics, navigation, and computer vision. In this paper, we explore a approach that integrates data from an Inertial Measurement Unit (IMU) and a camera to enhance motion tracking and panoramic image reconstruction. The IMU, equipped with an accelerometer and a gyroscope, provides measurements of linear acceleration and angular velocity. However, these raw readings are affected by noise, bias, and calibration errors, which must be addressed to obtain accurate motion estimates. To mitigate these issues, we pre-process the IMU data by removing biases and converting the raw signals into meaningful physical units, allowing precise estimation of the orientation and trajectory of the system.

In addition to IMU-based motion tracking, the system captures visual data through an onboard camera, while a VICON motion capture system provides high-accuracy ground-truth orientation measurements. Synchronizing these sensor modalities is essential due to their varying measurement rates. The first part of the paper focuses on estimating the system's 3D orientation using IMU data alone. This involves correcting measurement biases and applying a quaternion-based motion model to track orientation over time. To improve accuracy, we formulate an optimization problem that minimizes the error between predicted and actual orientation trajectories, with validation performed against VICON ground-truth data.

The second part of the project leverages the camera data to generate a panoramic image by aligning multiple RGB frames using the estimated body orientations. The images are initially mapped onto a spherical surface to determine their spatial positions in 3D space. These coordinates are then transformed from the sensor's local frame to the world frame using rotation matrices derived from VICON data. To create a seamless panorama, the transformed images are projected onto a cylindrical surface and subsequently unwrapped into a final composite image. This approach ensures accurate spatial alignment while preserving visual continuity across multiple frames.

Identify applicable funding agency here. If none, delete this.

II. Problem Formulation

1) Orientation Tracking: The goal of this project is to track the 3D orientation of a body over time and estimate the rotations (in terms of rotation matrices or quaternions) the body undergoes at each time step. We collect data from an IMU sensor, which provides linear acceleration and rotational velocity at each time instance. This data is used to estimate the orientation trajectory of the body, from which we can build a panoramic model. The problem is defined by the following cost function:

We define a cost function $c(q_1, q_2, \ldots, q_T)$ that takes into account both the motion model and the observation model. The cost function is formulated as:

$$c(q_1, q_2, \ldots, q_T) = \frac{1}{2} \sum_{t=0}^{T-1} \left\| 2 \log \left(q_{t+1}^{-1} \circ f(q_t, \tau_t, \omega_t) \right) \right\|_2^2 + \frac{1}{2} \sum_{t=1}^{T} \| a_t - h(q_t) \|_2^2$$

Where:

- $f(q_t, \tau_t, \omega_t)$ models the orientation kinematics and predicts the orientation quaternion at the next time step $t+1$.
- $h(q_t)$ represents the observation model that predicts the linear acceleration based on the current orientation q_t.
- q_t represents the quaternion at time t, representing the orientation of the body with respect to the world frame.

The motion model error is defined in the first term of the cost function, where the predicted orientation $f(q_t, \tau_t, \omega_t)$ is compared to the estimated orientation q_{t+1}. The second term accounts for observation model error.

To estimate the optimal orientation trajectory, we minimize the following cost function with respect to the quaternions $q_1, q_2, \ldots, q_T$:

$$\min_{q_1, q_2, \ldots, q_T} c(q_1, q_2, \ldots, q_T)$$

Additionally, the following constraint on quaternions must satisfy $\|q_t\|_2 = 1$ for all $t \in \{1, 2, \ldots, T\}$, ensuring that the orientations remain valid throughout the process.

A. Panaroma Stiching

Given the orientation estimates $q_1, q_2, \ldots, q_T$ optimized from above approach and the sequence of RGB camera images $c_1, c_2, \ldots, c_N$, we aim to stitch the images together into a single panoramic image C_{panorama}. The problem is formulated as follows:

The goal is to find the nearest transformations T_t for each image c_t such that the images are aligned and stitched into a single panoramic image. The transformation T_t is defined as:

$$T_t = [R(q_t)]$$

where $R(q_t)$ is the rotation matrix corresponding to the quaternion q_t.

III. Technical Approach

A. Orientation Tracking

The IMU provides raw measurements of acceleration and angular velocity, but these measurements contain biases that must be corrected.

To remove bias from the sensor readings, we compute the average of the first M samples while assuming the system is at rest:

$$\tilde{a} = \frac{1}{M}\sum_{j=1}^{M} a_j, \quad \tilde{\omega} = \frac{1}{M}\sum_{j=1}^{M} \omega_j. \tag{1}$$

Here, $\tilde{a}$ and $\tilde{\omega}$ represent the estimated bias for acceleration and angular velocity, respectively.

Since the IMU measures acceleration in the presence of gravity, the vertical component must be adjusted.

$$a_{\text{corrected},z} = a_z - g. \tag{2}$$

The IMU outputs signals in voltage, which need to be converted into physical units using a scaling factor:

$$S_{\text{accel}} = \frac{V_{\text{ref, accel}}}{\text{Resolution} \times \text{Sensitivity}_{\text{accel}}}, \tag{3}$$

$$S_{\text{gyro}} = \frac{V_{\text{ref, gyro}}}{\text{Resolution} \times \text{Sensitivity}_{\text{gyro}}}. \tag{4}$$

where V_{ref} is the reference voltage, and Sensitivity is given in mV per unit (e.g., mV/g for accelerometer, mV/°/s for gyroscope). Here resolution is 1023 bits. $V_{\text{ref, accel}}$ and $V_{\text{ref, gyro}}$ can be found in datasheet.

Using the computed biases and scaling factors, the final corrected values are:

$$a_{\text{final}} = (a_{\text{measured}} - \bar{a}) \cdot S_{\text{accel}}, \quad \omega_{\text{final}} = (\omega_{\text{measured}} - \tilde{\omega}) \cdot S_{\text{gyro}}. \tag{5}$$

These corrected values are then used for further computations. Quaternions are computed using a motion model to verify the correctness of the calibrated IMU data. The quaternion at the next time step is given by:

$$q_{t+1} = f(q_t, \tau_t, \omega_t), \tag{6}$$

$$q_{t+1} = q_t \circ \exp\left(\frac{1}{2}[0, \tau_t \omega_t]\right). \tag{7}$$

where the initial body orientation is set as:

$$q_0 = [1, 0, 0, 0]. \tag{8}$$

Here, $\circ$ represents quaternion multiplication, and $\exp$ denotes the exponential map used for quaternion updates.

Quaternions $q_{1:T}$ computed from IMU data are trasnformed into euler angles using the `transforms3D` library to compare the estimated quaternions with ground truth data of VICON.

Quaternions $q_{1:T}$ are used to estimate the acceleration using the observation model. Since the body undergoes pure rotation, its acceleration in the world frame should be approximately:

$$[0, 0, -g], \tag{9}$$

where g represents gravitational acceleration.
using the following observation model:

$$[0, a_t] = h(q_t) \tag{10}$$

$$[0, a_t] = q_t^{-1} \circ [0, 0, 0, -g] \circ q_t. \tag{11}$$

Here, q_t^{-1} represents the inverse of the quaternion q_t.

To refine the estimated orientation, we use gradient-based optimization.

The cost function is minimized by computing the gradient with respect to the quaternion variables:

$$q_{\text{new}} = q_{\text{old}} - \alpha \frac{\partial C}{\partial q}. \tag{12}$$

where α is the learning rate Since quaternions must remain unit-norm for valid rotations, we normalize the updated quaternion:

$$q_{\text{final}} = \frac{q_{\text{new}}}{\|q_{\text{new}}\|}. \tag{13}$$

This ensures numerical stability and valid orientation tracking.

Parameters are set to the following in the optimisation and `autograd` library has been used for the optimisation:

$$num\_iterations = 600 \tag{14}$$

$$\alpha = 0.001 \tag{15}$$

By applying bias correction, quaternion-based motion modeling, and projected gradient-based optimization, we achieve robust orientation tracking for the IMU-based system.

B. Panaroma stitching

Panorama stitching involves merging multiple images taken from different orientations into a single seamless wide-angle image. The approach relies on estimating the transformation between consecutive images using sensor data and mathematical projections.

Given a sequence of images captured by a camera with a known field of view, we transform the pixel coordinates into spherical coordinates. For a given image of size $h \times w$, the spherical coordinates $(\lambda, \phi, 1)$ can be computed as:

$$\lambda = -\frac{(v - w/2)}{w} \times HFOV, \tag{16}$$

$$\phi = \frac{(u - h/2)}{h} \times VFOV \tag{17}$$

where λ represents longitude and ϕ represents latitude. HFOV and VFOV stands for horizontal field of view and vertical field of view respectively. This transformation aligns the image to a unit-radius sphere centered at the origin.

The above spherical coordinates are transformed into car etesian plane using the following transformation

$$x = r\cos\phi\cos\lambda, \tag{18}$$

$$y = r\cos\phi\sin\lambda, \tag{19}$$

$$z = -r\sin\phi, \tag{20}$$

where $r = 1$. Rotate the Cartesian coordinates to the world frame using rotation matrix R.

$$(x, y, z)_{world} = R(x, y, z) \tag{21}$$

Convert the Cartesian coordinates in world frame back into spherical frame.

$$\lambda_{world} = arcsin(-\frac{z_{world}}{r}) \tag{22}$$

$$\phi_{world} = arctan(\frac{y_{world}}{x_{world}}) \tag{23}$$

Inscribing the sphere in a cylinder so that a point $(\lambda, \phi, 1)$ on the sphere has height ϕ on the cylinder and longitude λ along the cylinder circumference Unwrap the cylinder surface to a rectangular image of size (final_height, final_width) with width 2π radians and height π radians. Once the images are projected, we map their pixels to a final panorama image of resolution 900×1080. The mapping follows:

Each pixel coordinate of rectangular image(u, v) is given by the following.

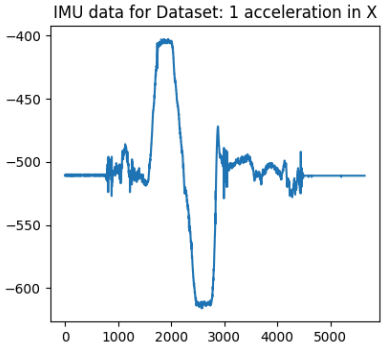
$$u = \frac{\phi_{world} + 0.5\pi}{\pi} \times final\_height, \tag{24}$$

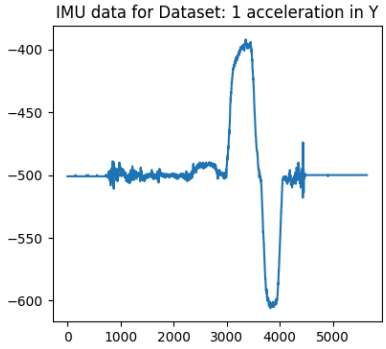
$$v = \frac{\phi_{world} - \lambda}{2\pi} \times final\_width. \tag{25}$$

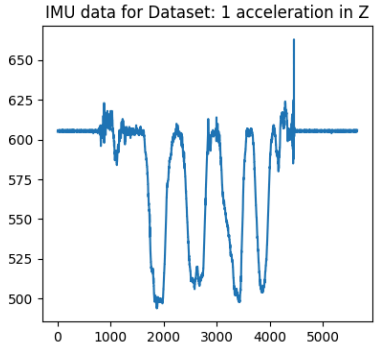
IV. Results

Dataset - 1

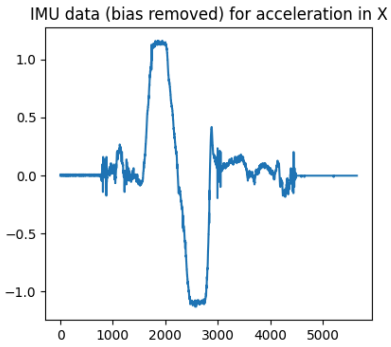
IMU raw data :

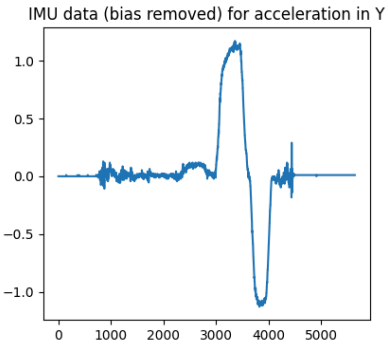


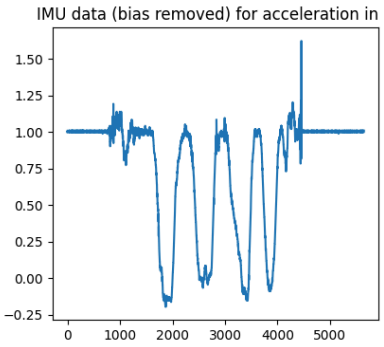


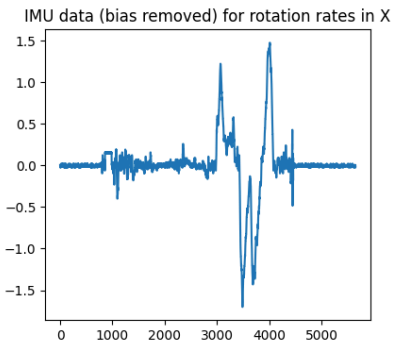


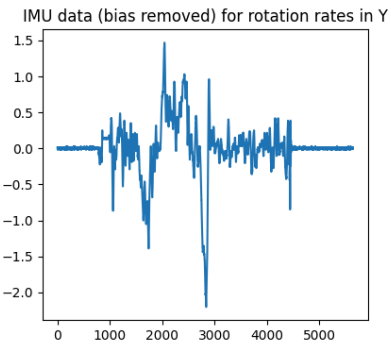
IMU data (after bias removal) :

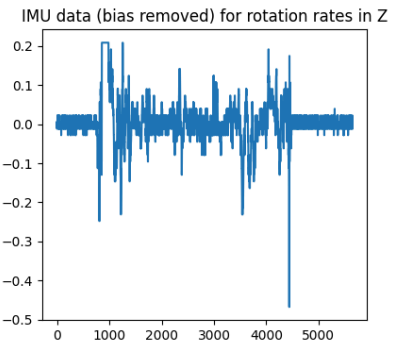




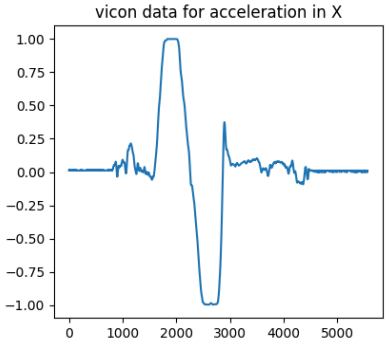


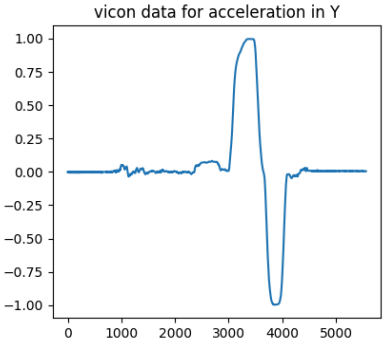


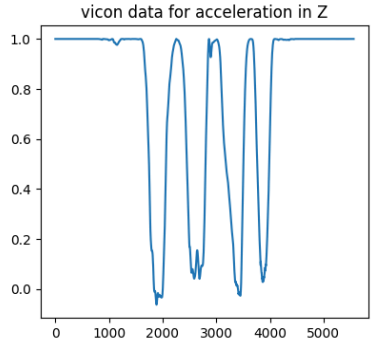


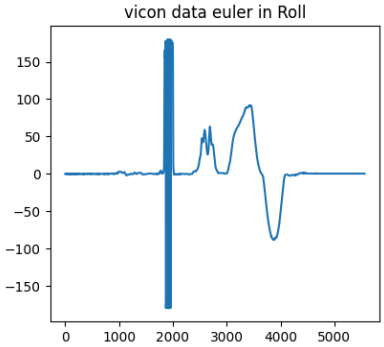


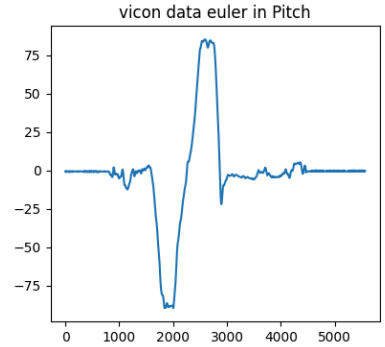
Vicon Data:

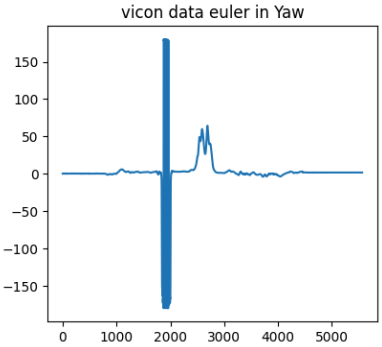




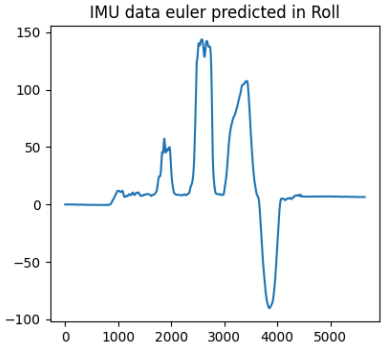


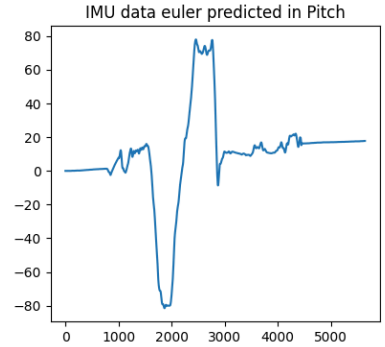


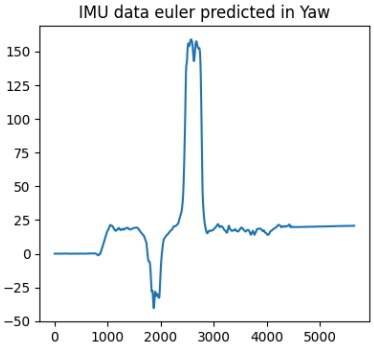




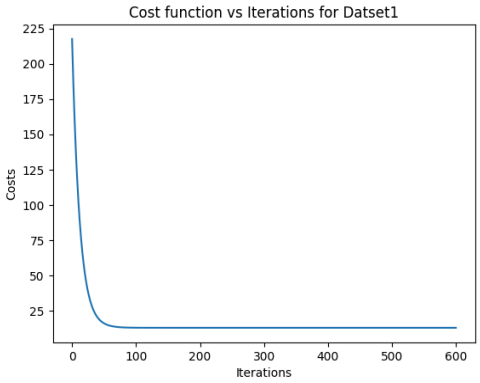
IMU estimated euler:



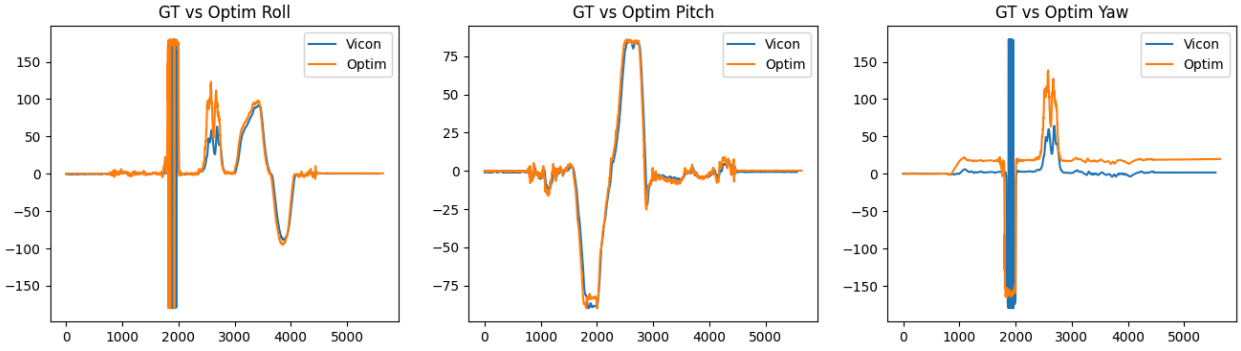




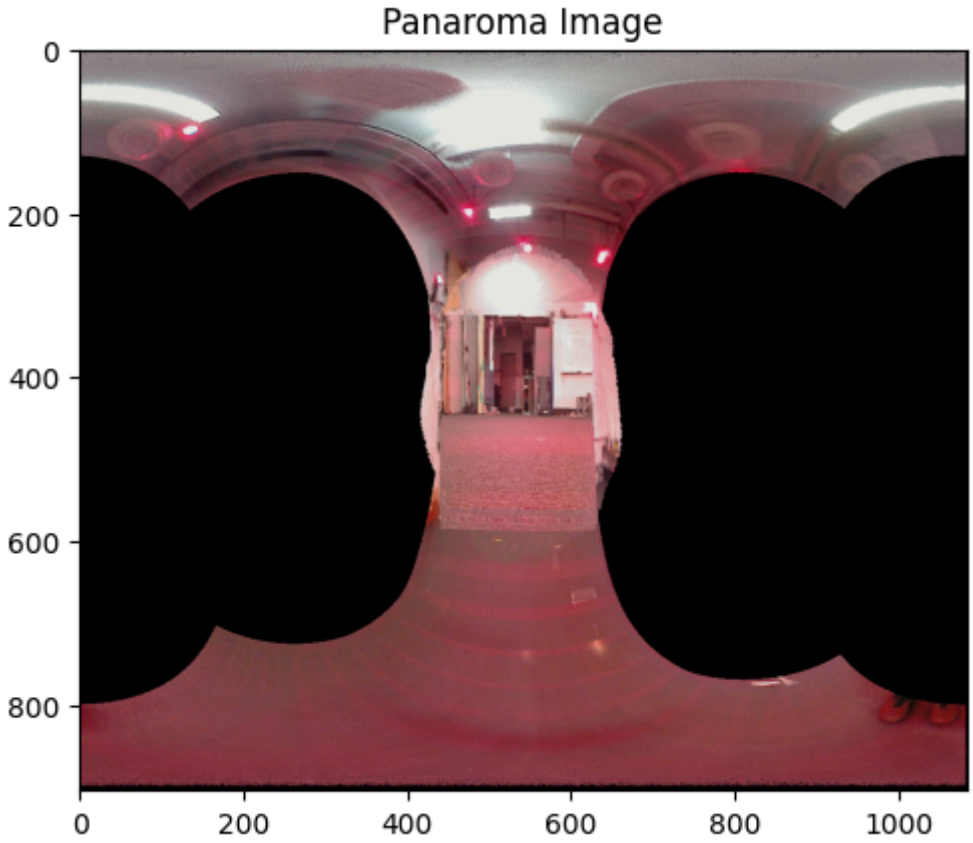
Cost function vs iterations:



IMU optimised euler:

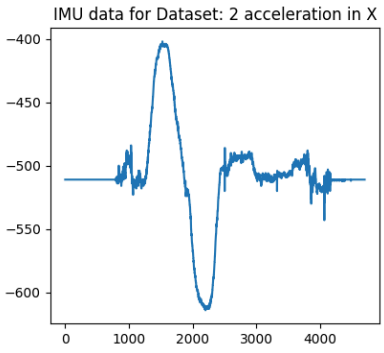


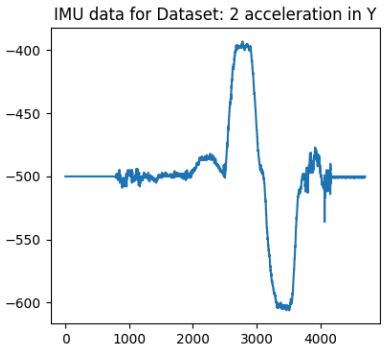
Panorama Image:

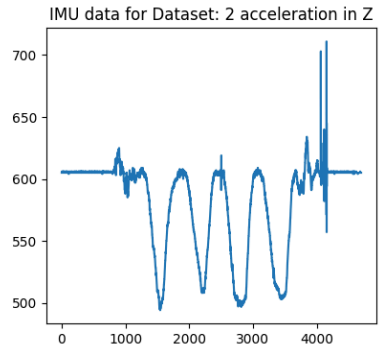


Dataset - 2

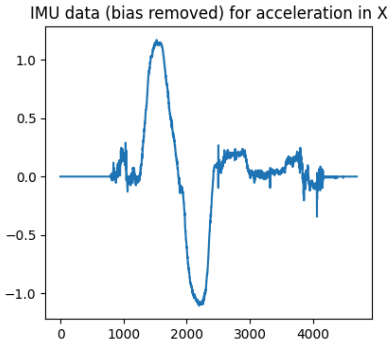
IMU raw data :

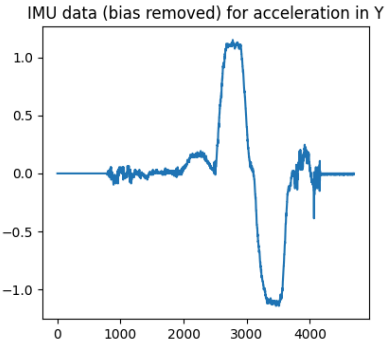


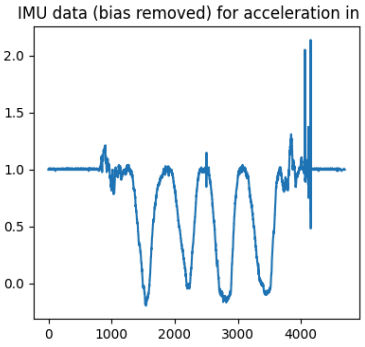


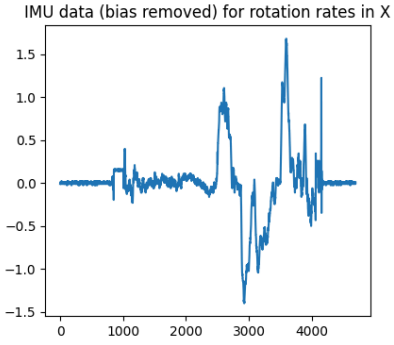


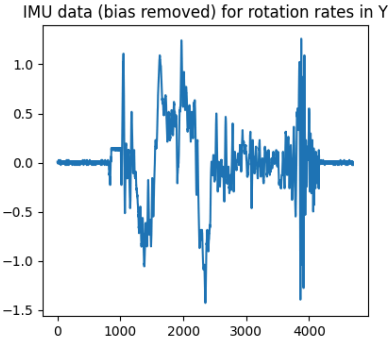
IMU data (after bias removal) :

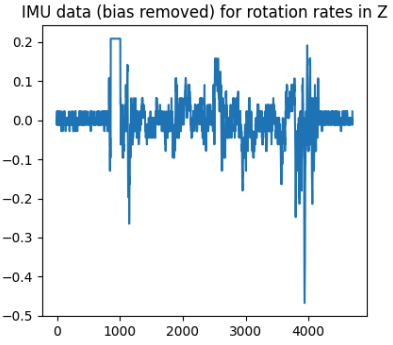




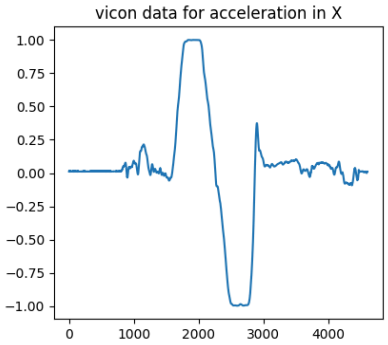


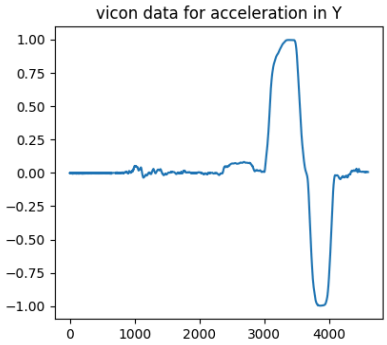


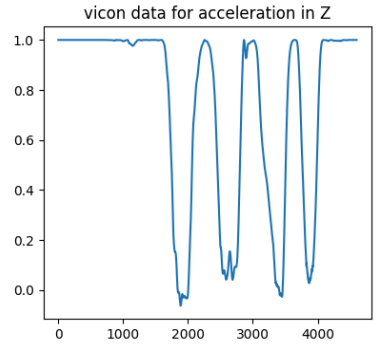




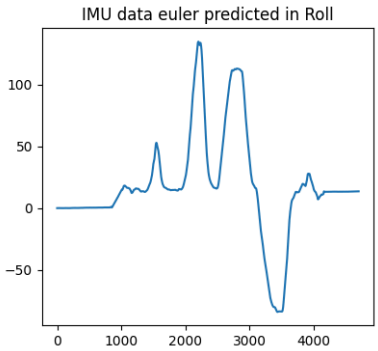
Vicon Data:

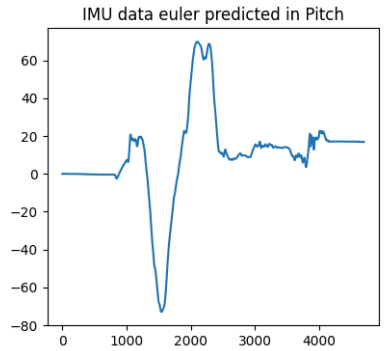


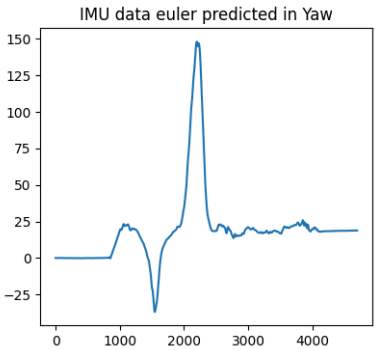




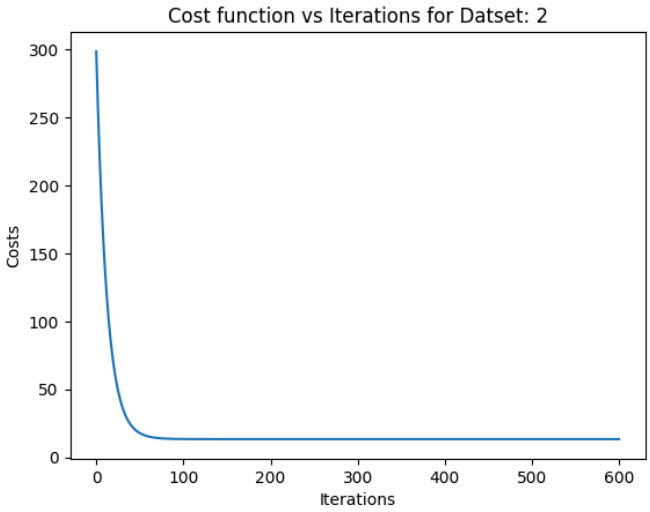
IMU estimated euler:



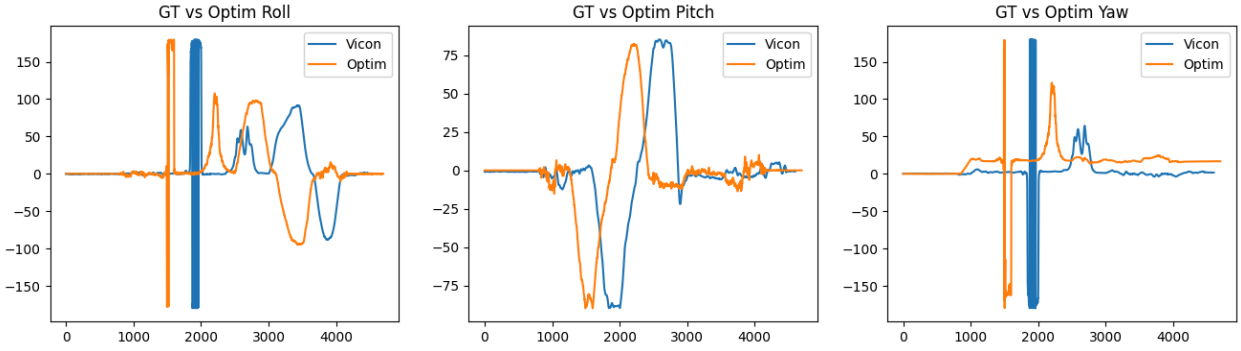




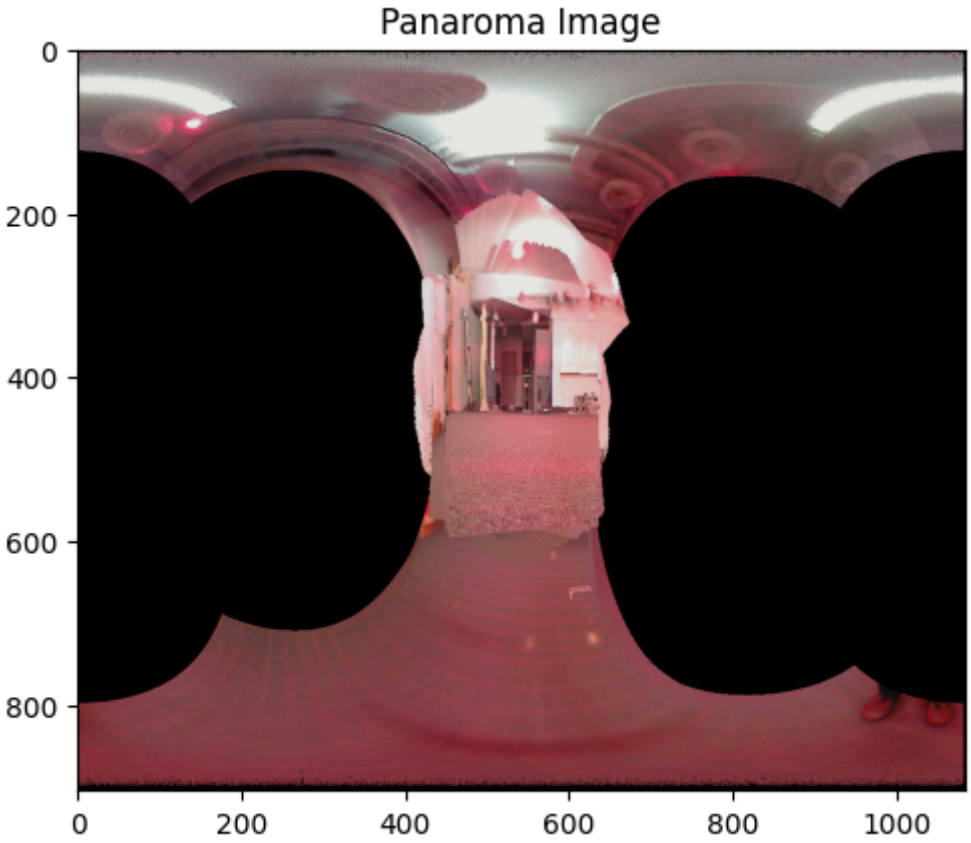
Cost function vs iterations:



IMU optimised euler:

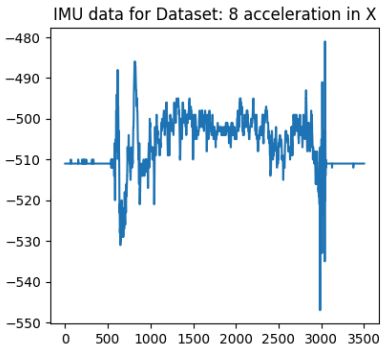


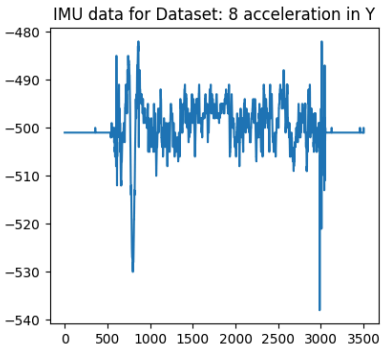
Panorama Image:

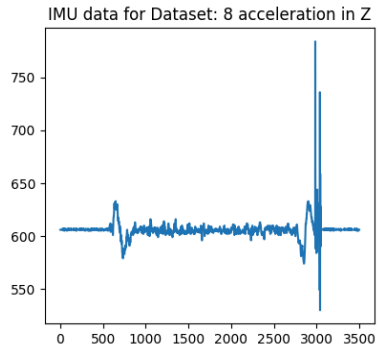


Dataset - 8

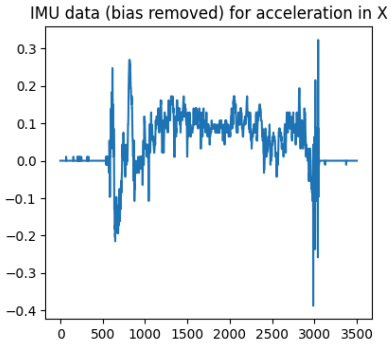
IMU raw data :

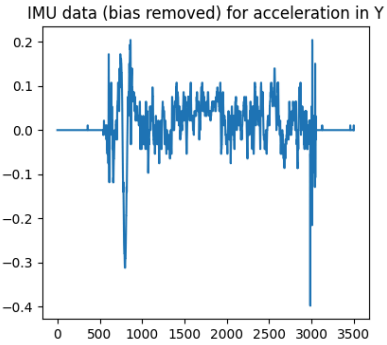


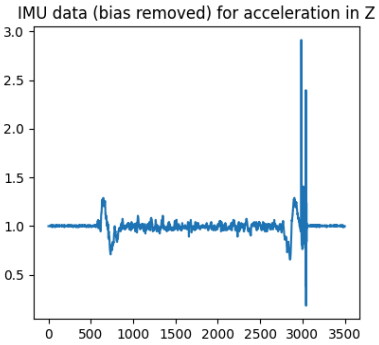


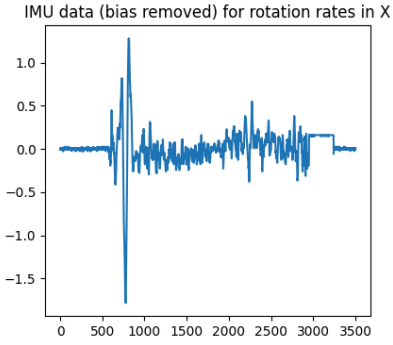


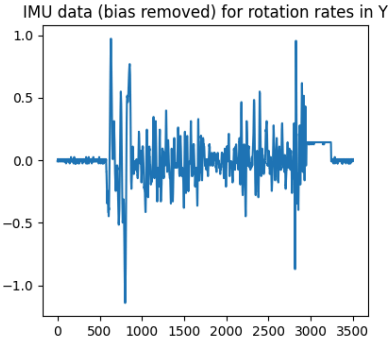
IMU data (after bias removal) :

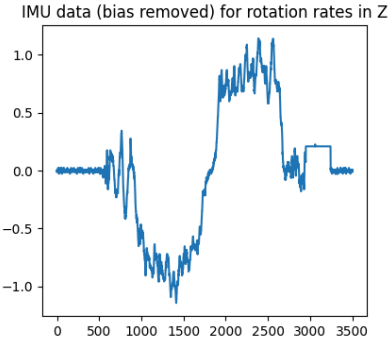




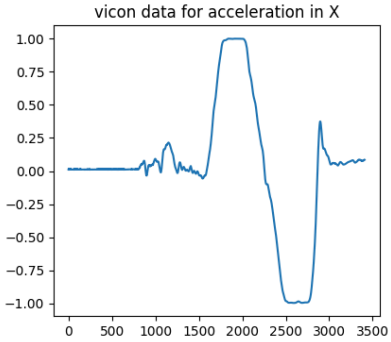


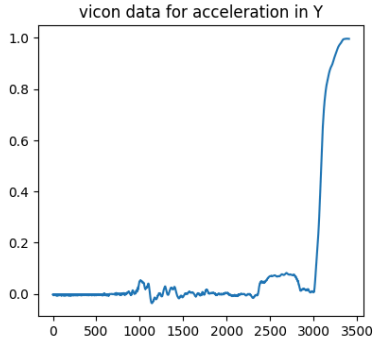


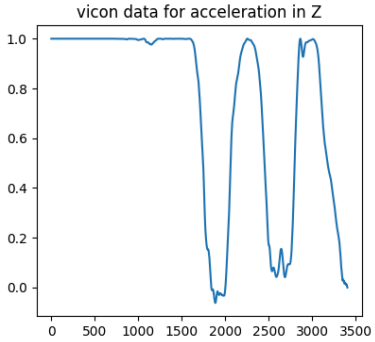




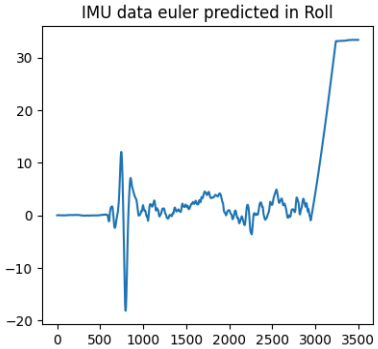
Vicon Data:

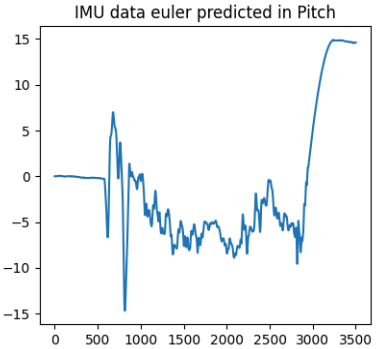


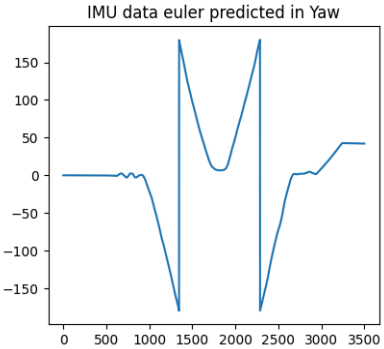




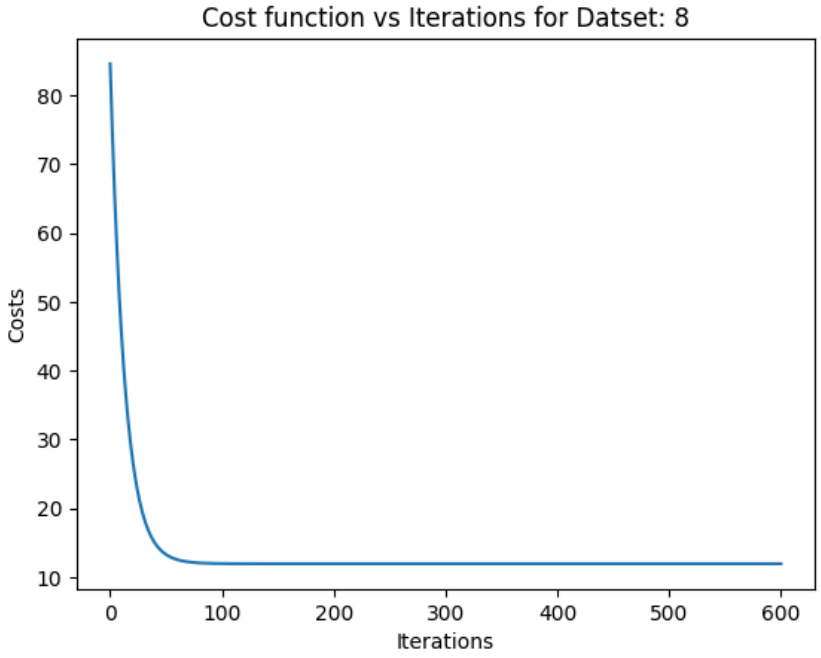
IMU estimated euler:



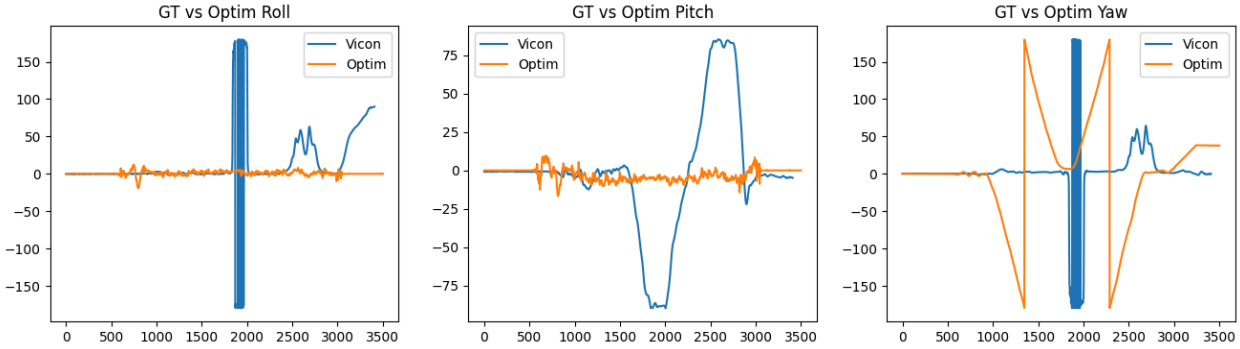




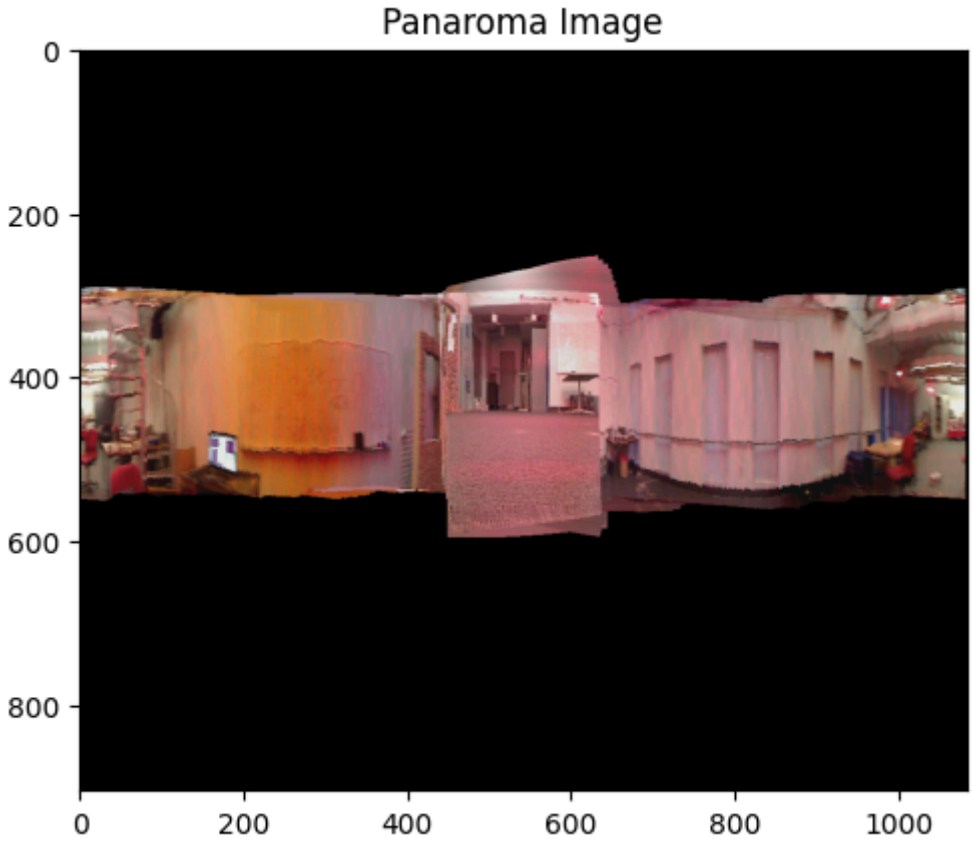
Cost function vs iterations:



IMU optimised euler:

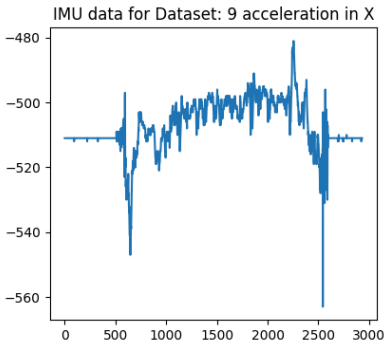


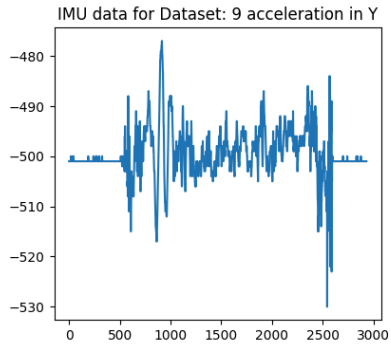
Panorama Image:

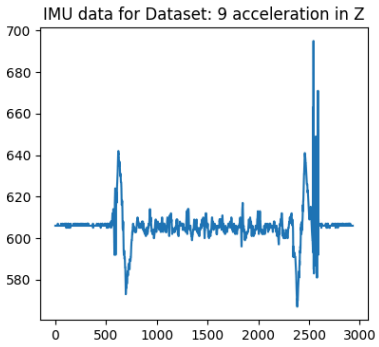


Dataset - 9

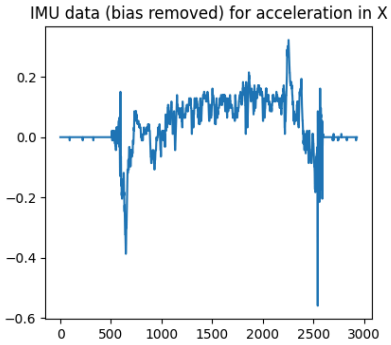
IMU raw data :

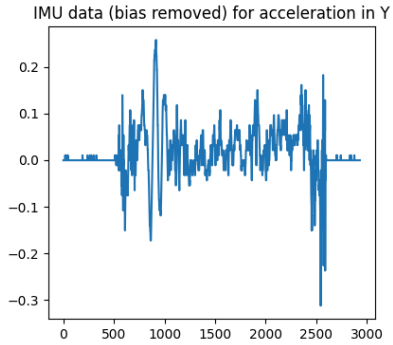


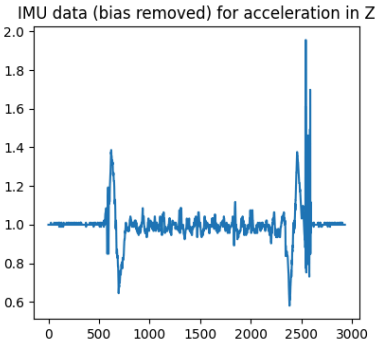


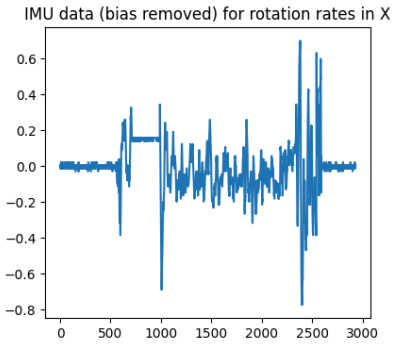


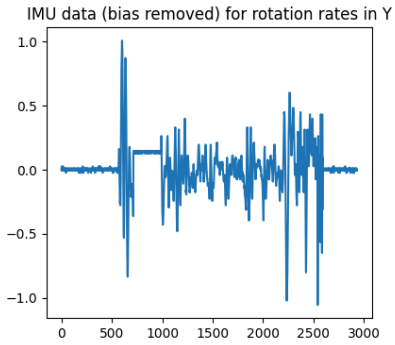
IMU data (after bias removal) :

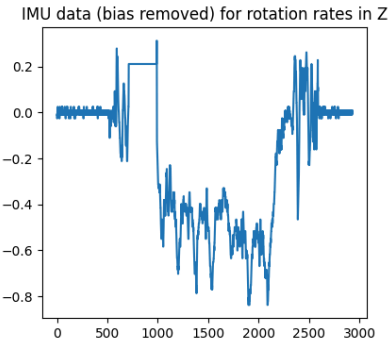




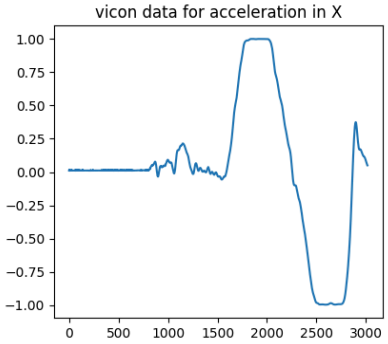


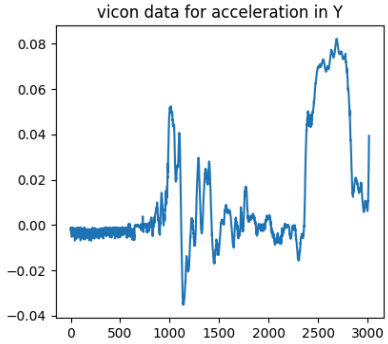


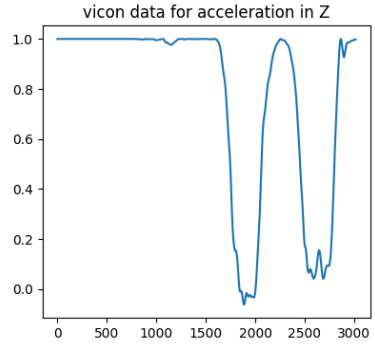




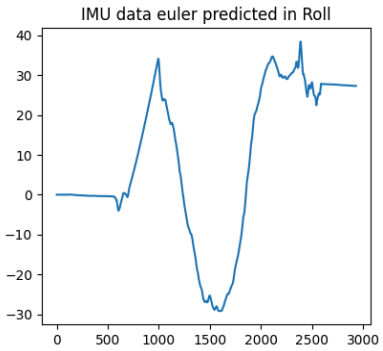
Vicon Data:

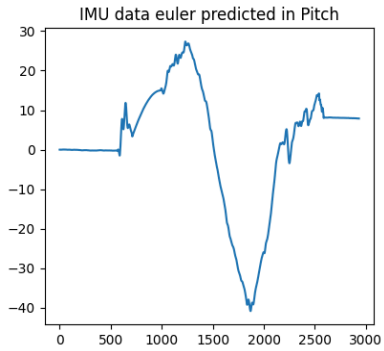


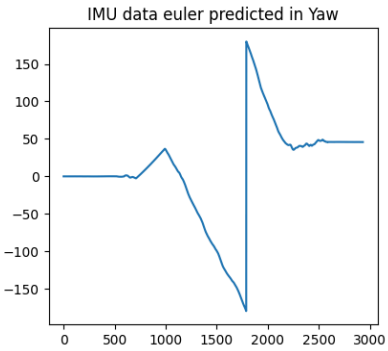




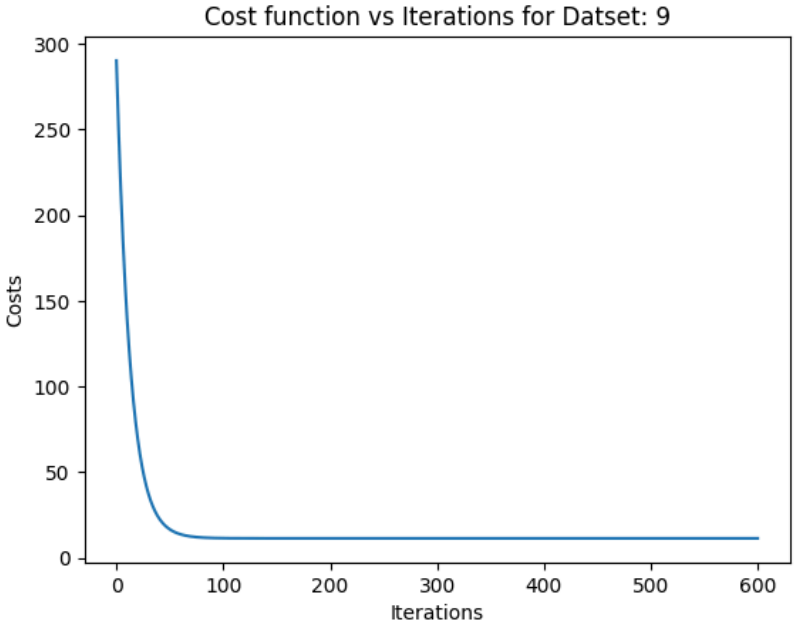
IMU estimated euler:



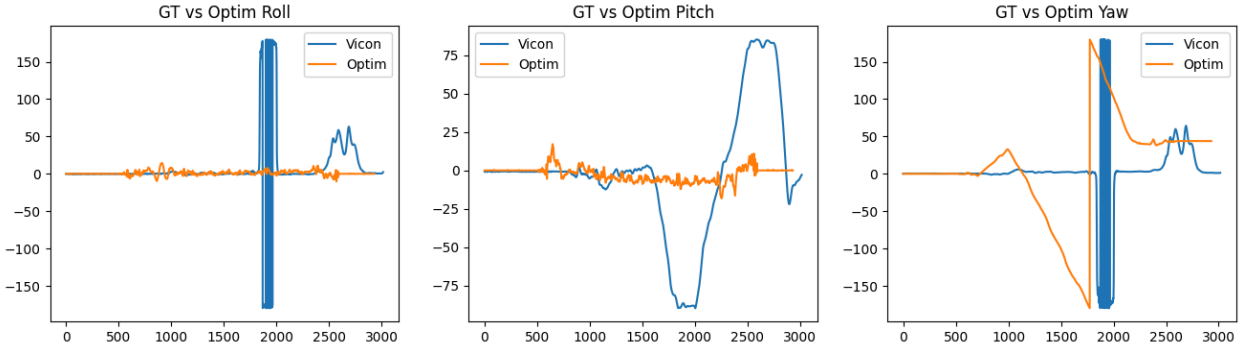




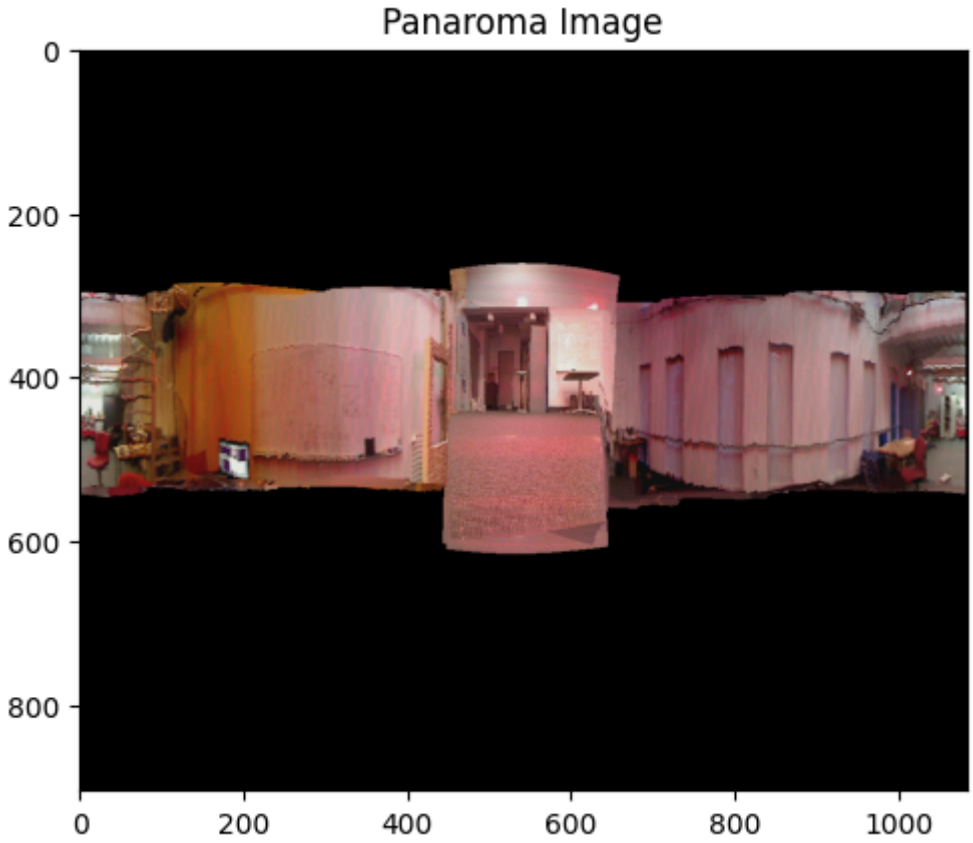
Cost function vs iterations:



IMU optimised euler:

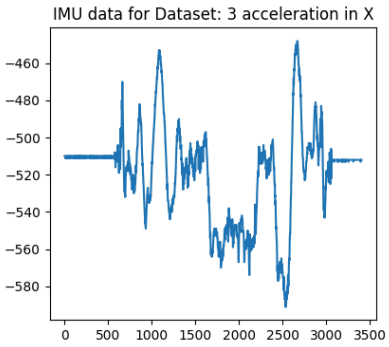


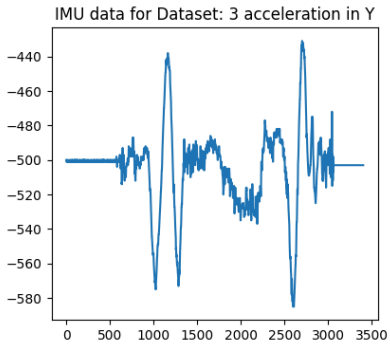
Panorama Image:

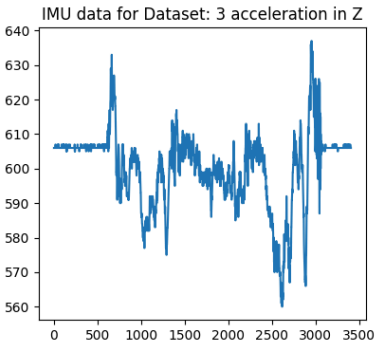


Dataset - 3

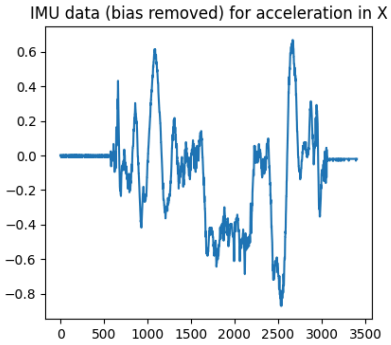
IMU raw data :

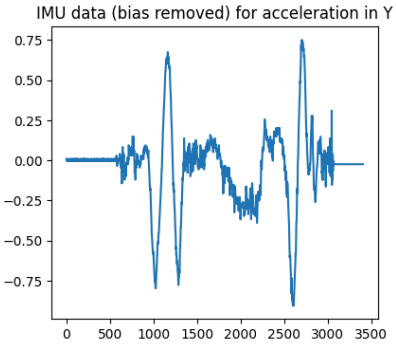


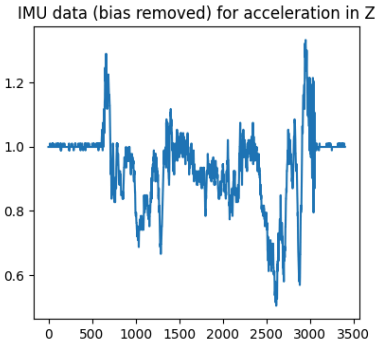


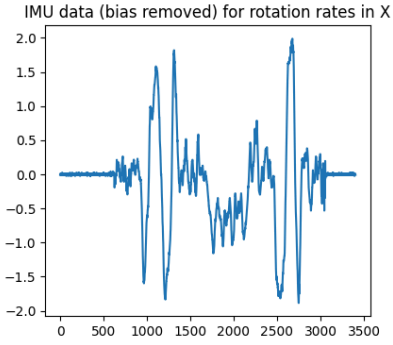


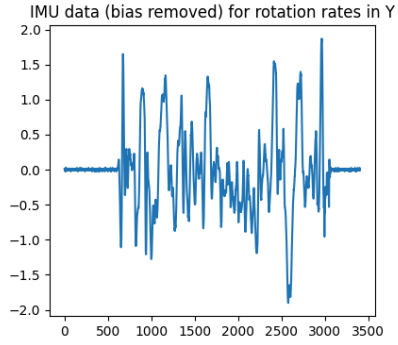
IMU data (after bias removal) :

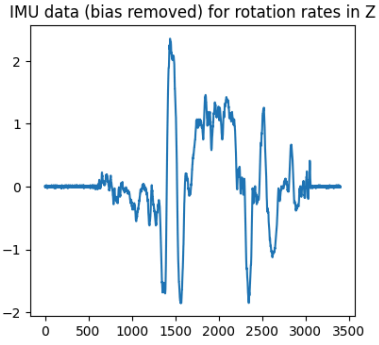




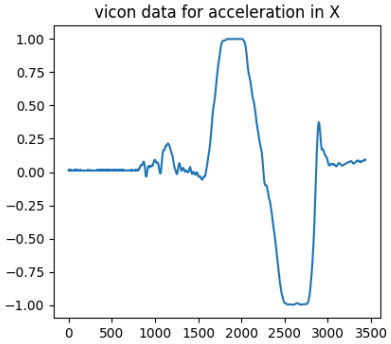


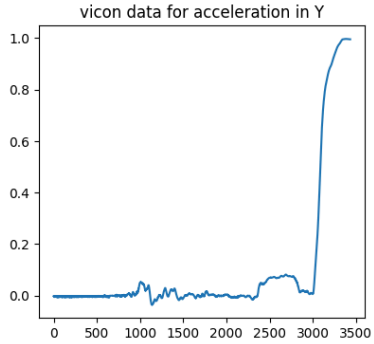


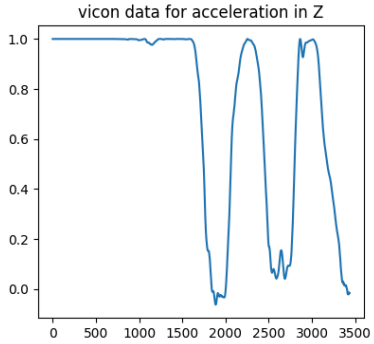




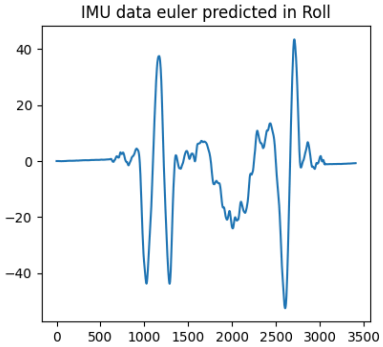
Vicon Data:

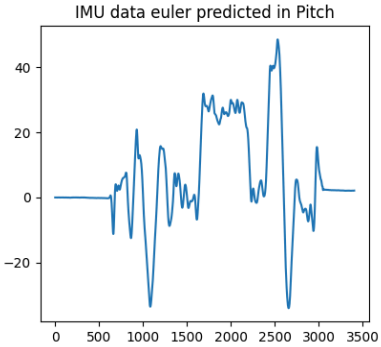


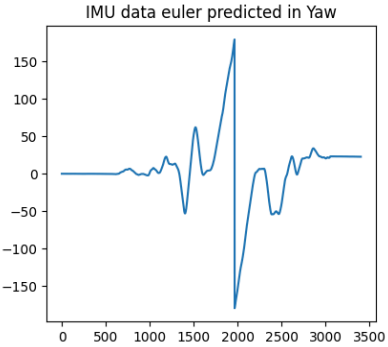




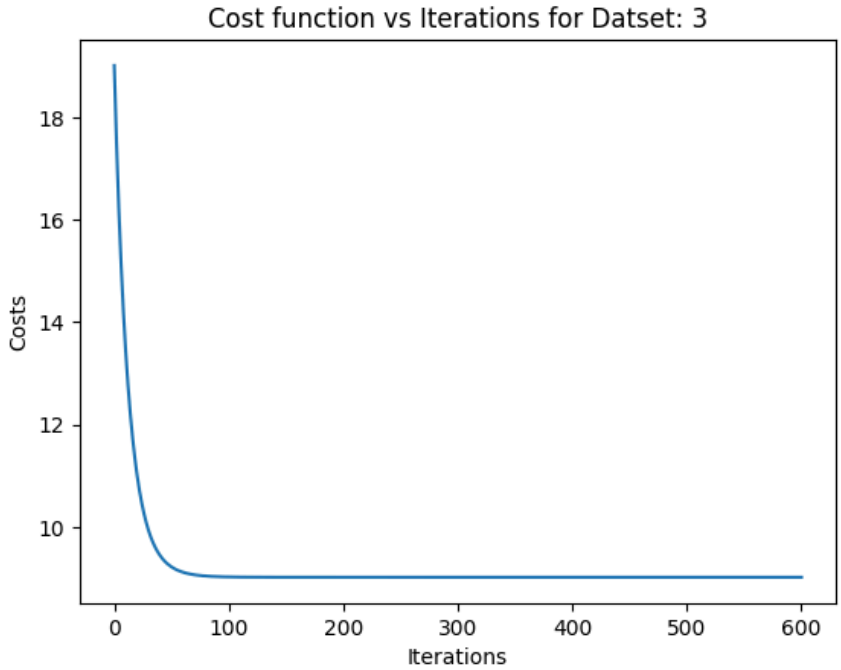
IMU estimated euler:



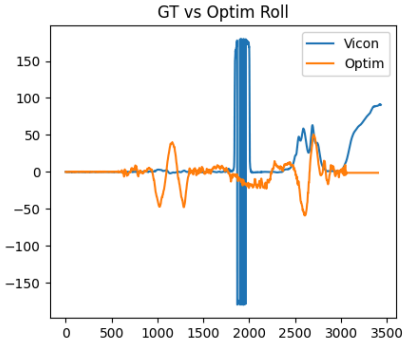


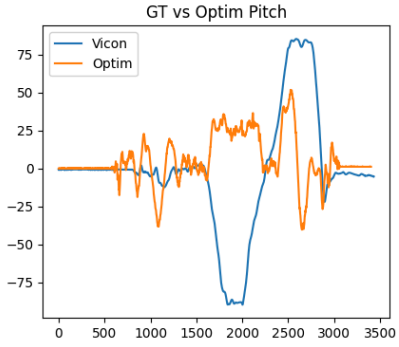


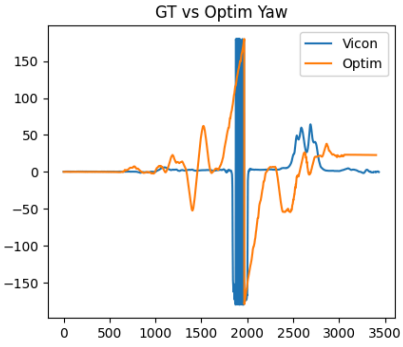
Cost function vs iterations:



IMU optimised euler:

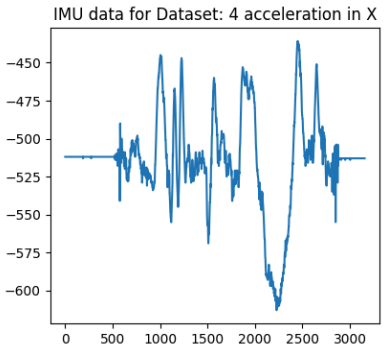


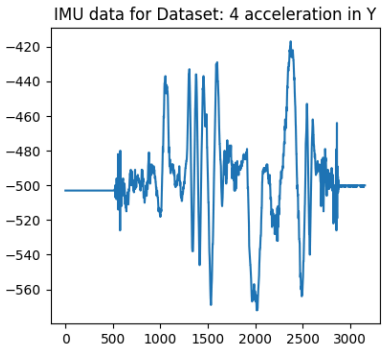


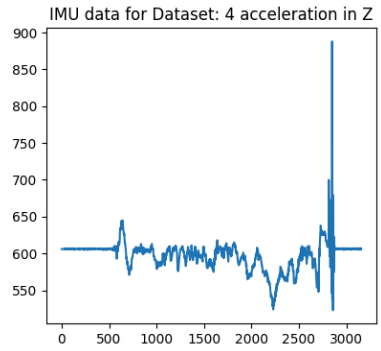


Dataset - 4

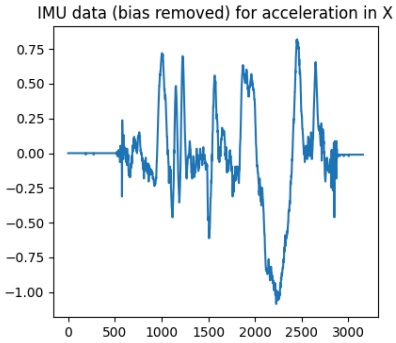
IMU raw data :

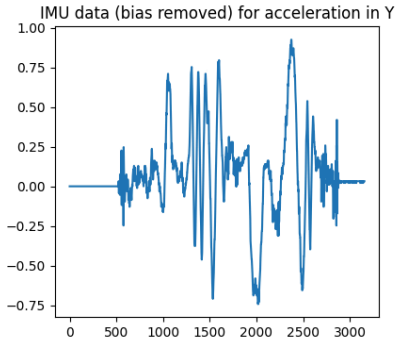


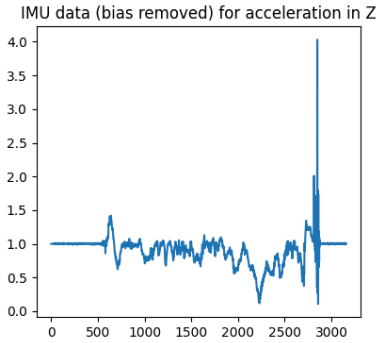


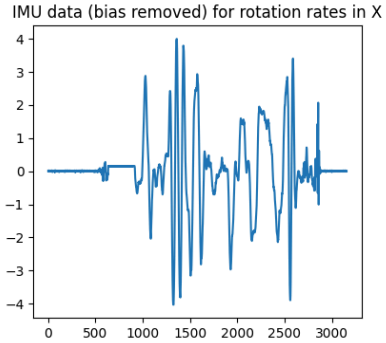


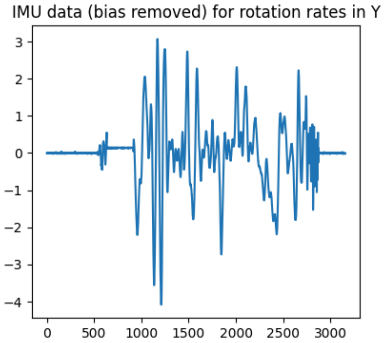
IMU data (after bias removal) :

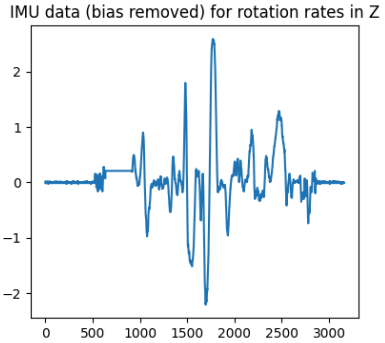




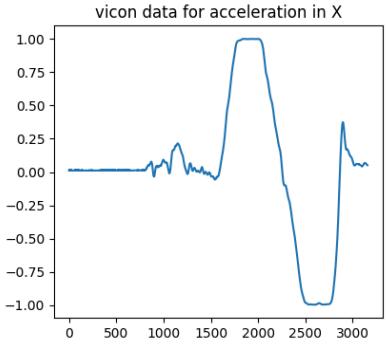


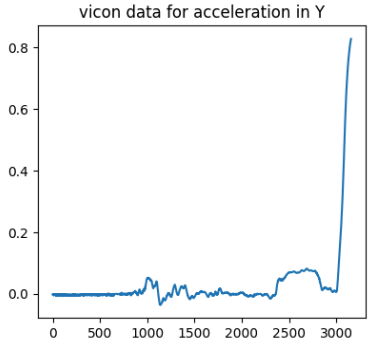


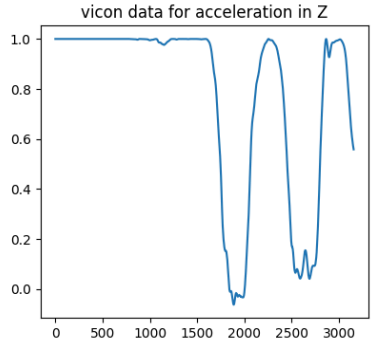




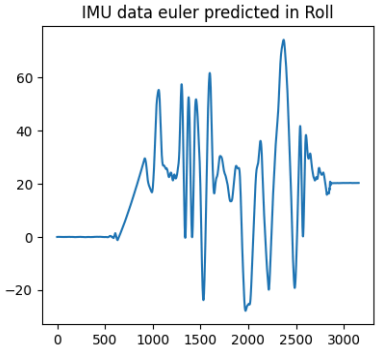
Vicon Data:

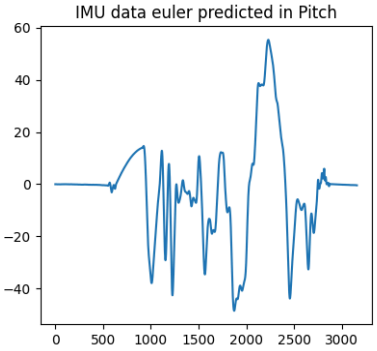


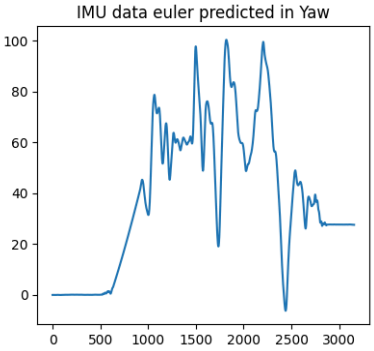




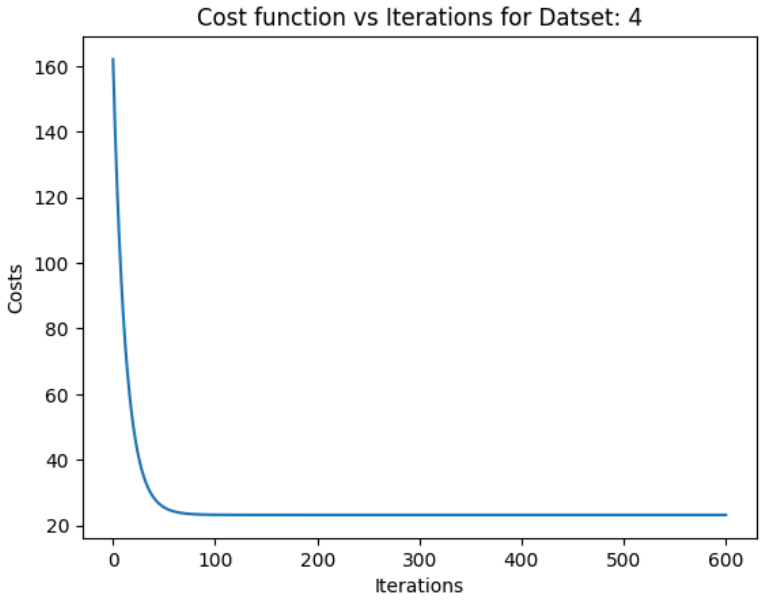
IMU estimated euler:



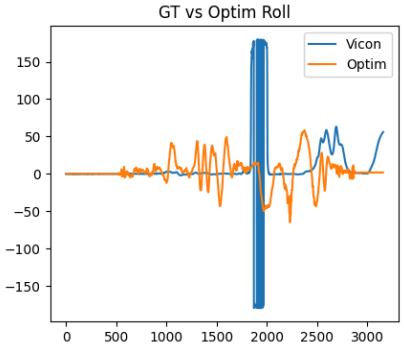


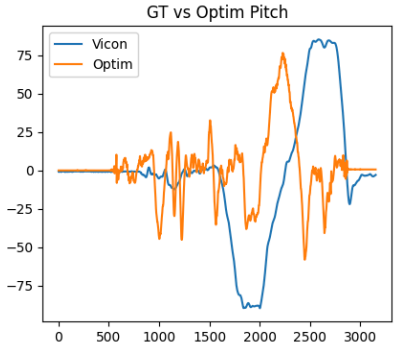


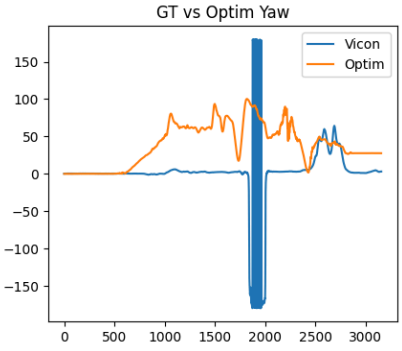
Cost function vs iterations:



IMU optimised euler:

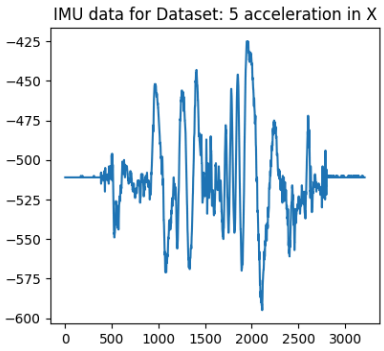


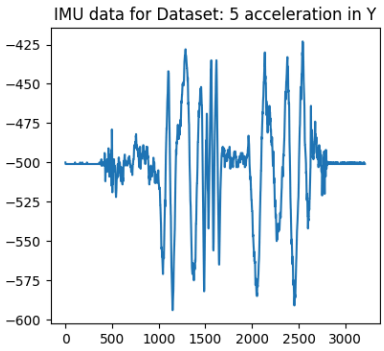


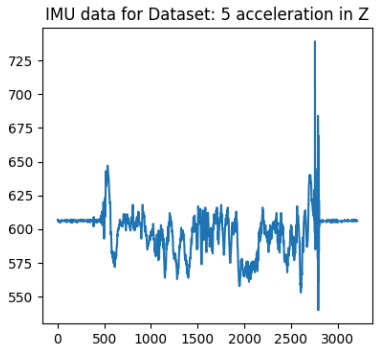


Dataset - 5

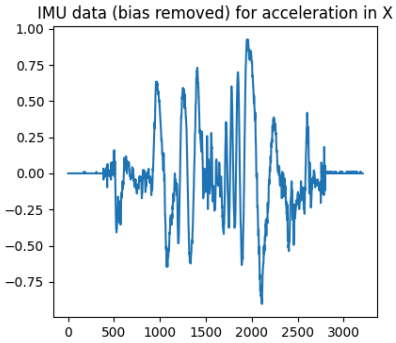
IMU raw data :

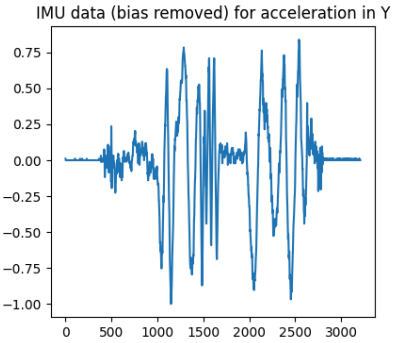


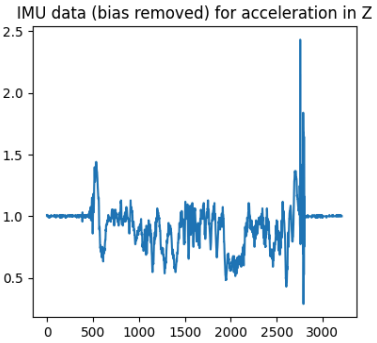


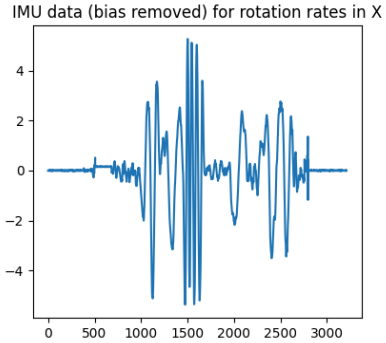


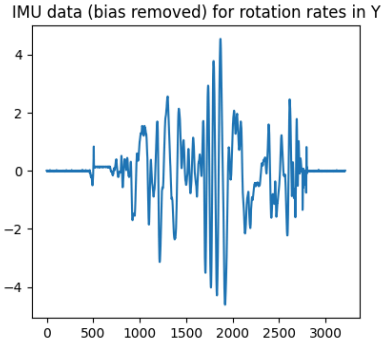
IMU data (after bias removal) :

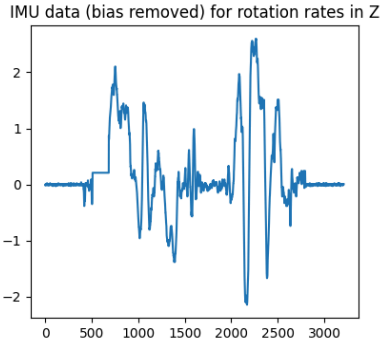




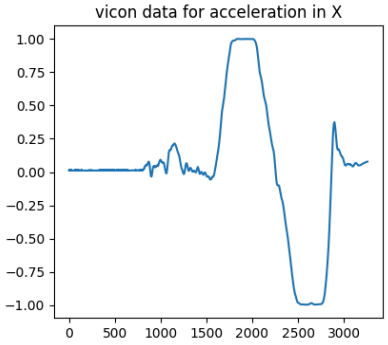


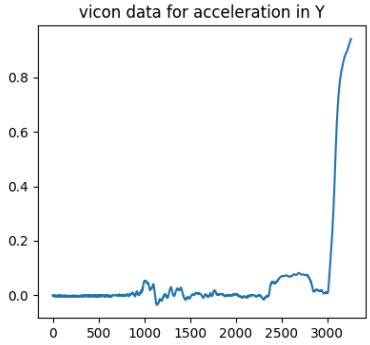


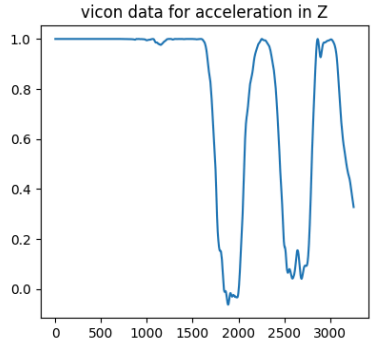




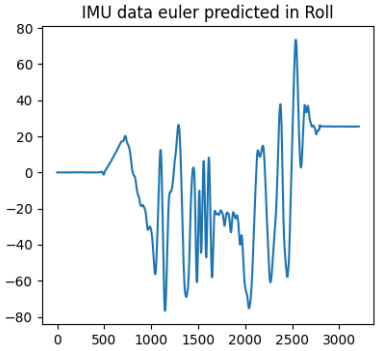
Vicon Data:

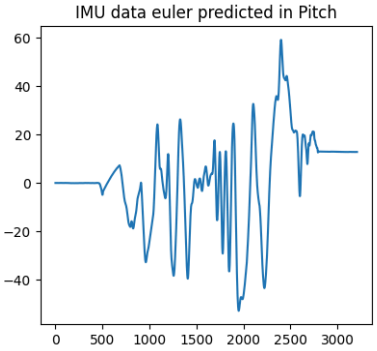


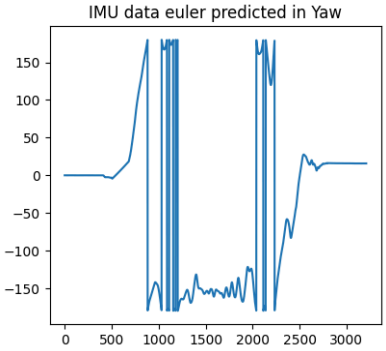




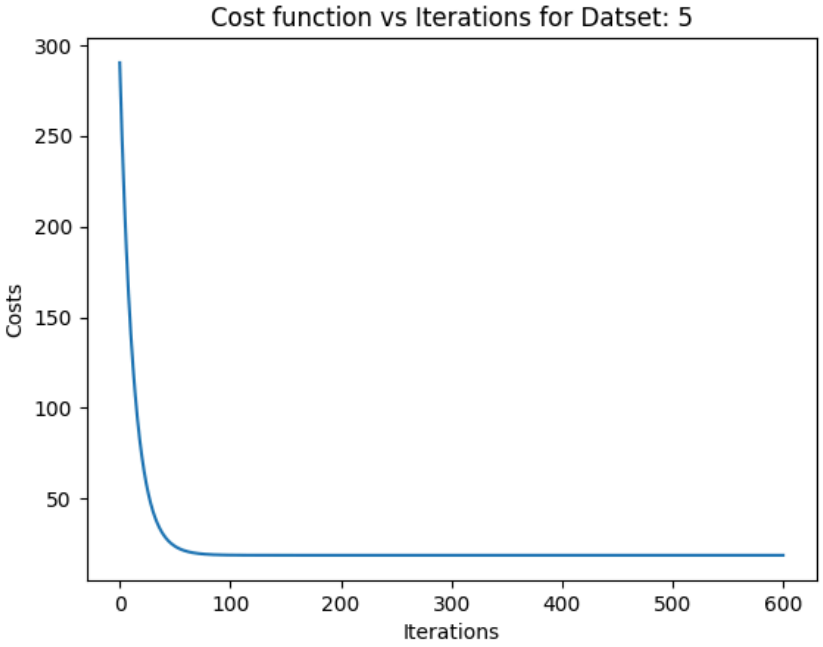
IMU estimated euler:



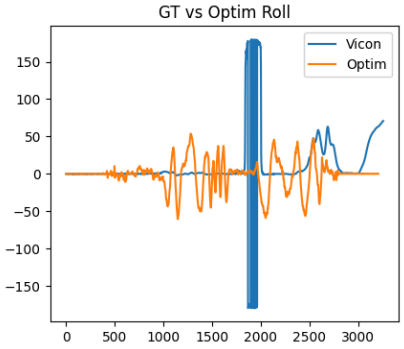


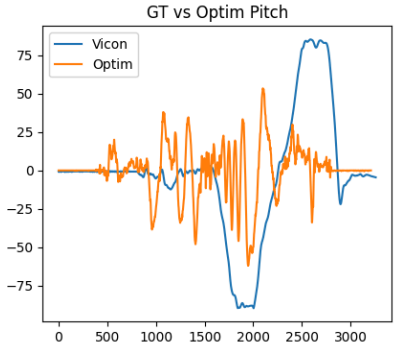


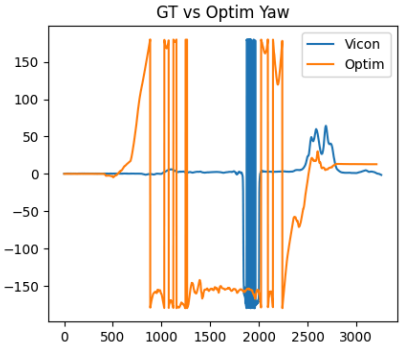
Cost function vs iterations:



IMU optimised euler:

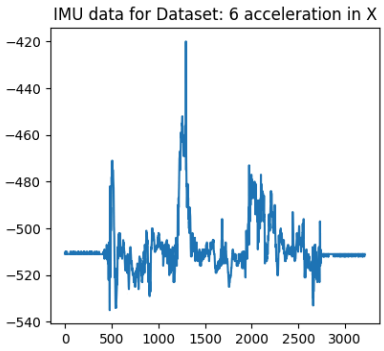


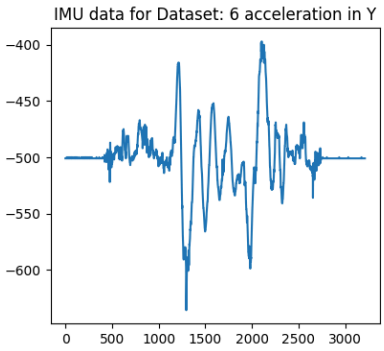


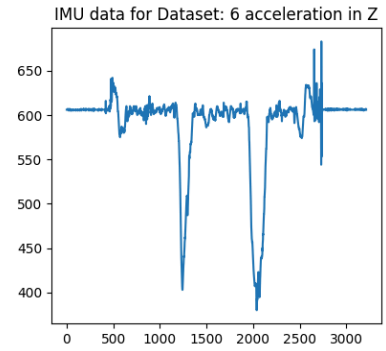


Dataset - 6

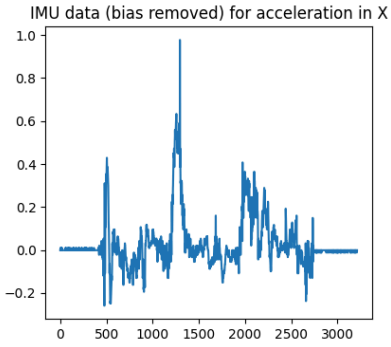
IMU raw data :

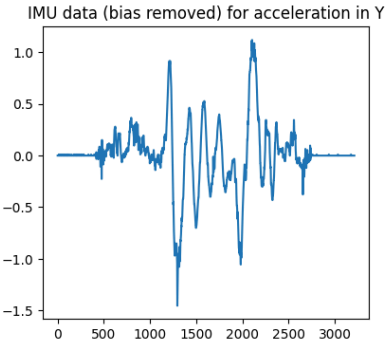


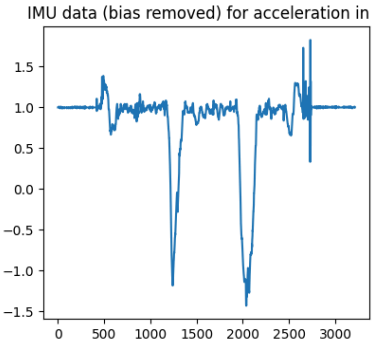


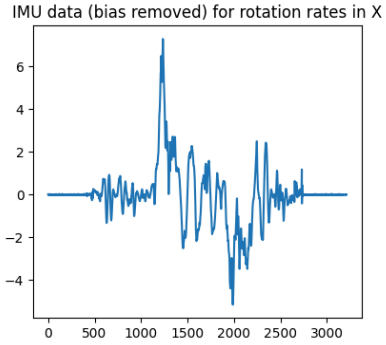


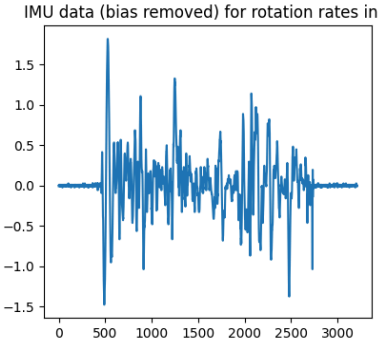
IMU data (after bias removal) :

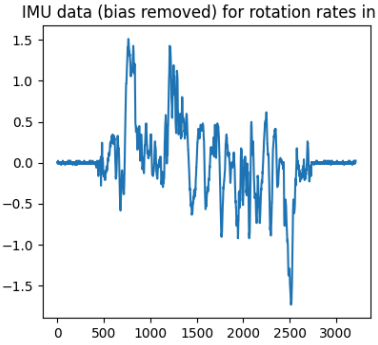




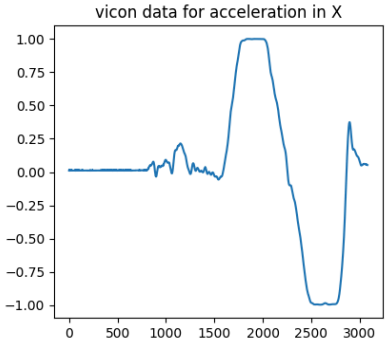


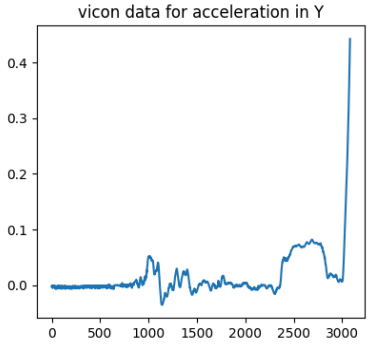


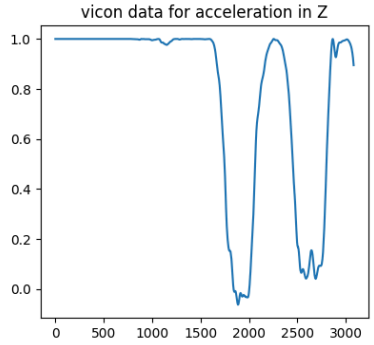




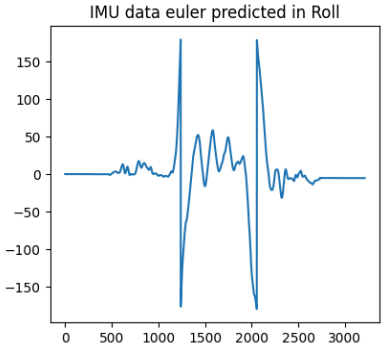
Vicon Data:

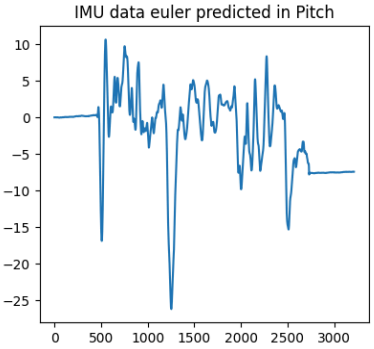


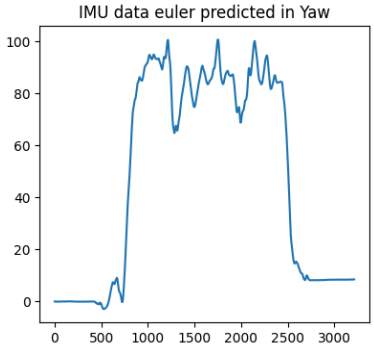




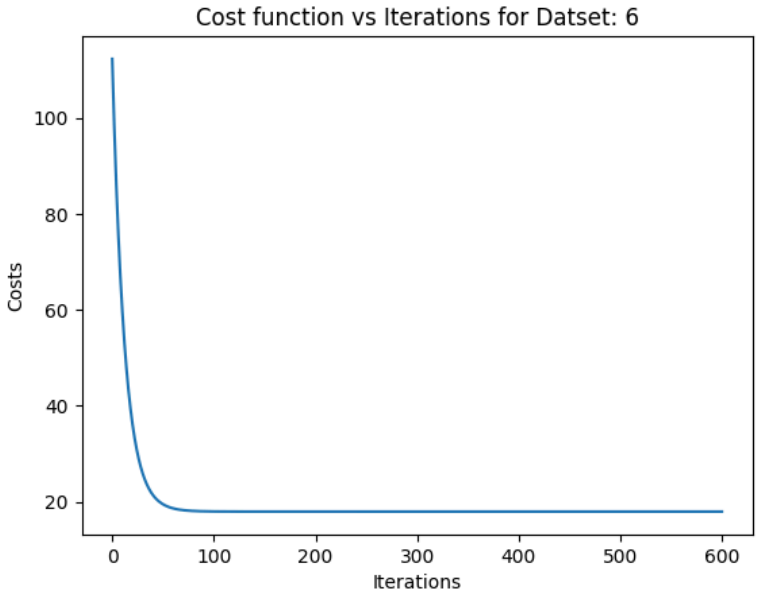
IMU estimated euler:



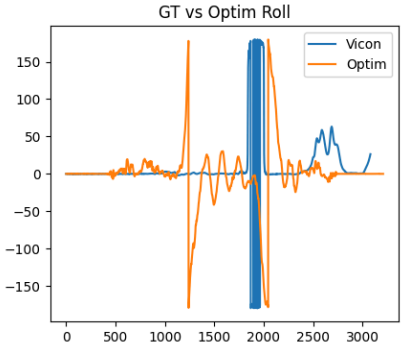


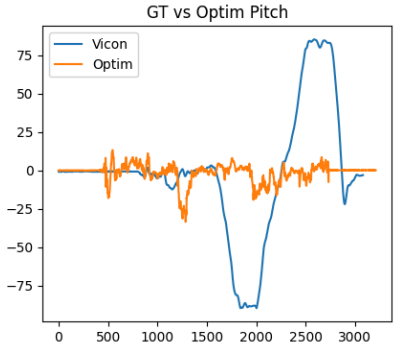


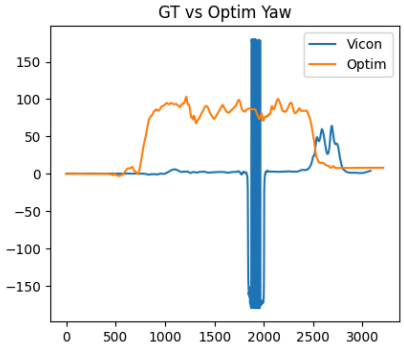
Cost function vs iterations:



IMU optimised euler:

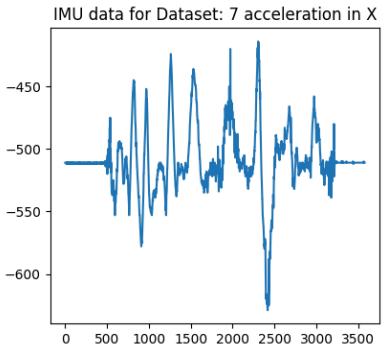


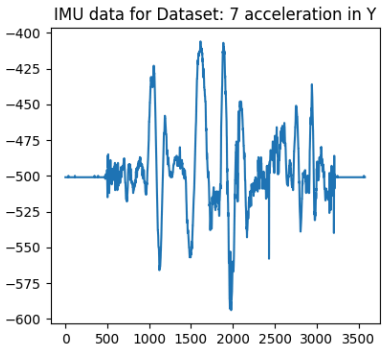


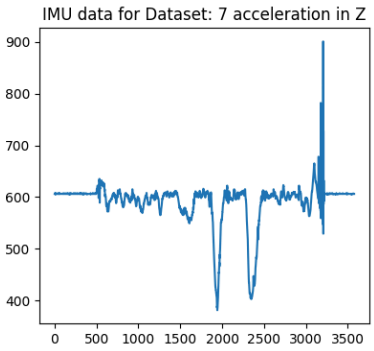


Dataset - 7

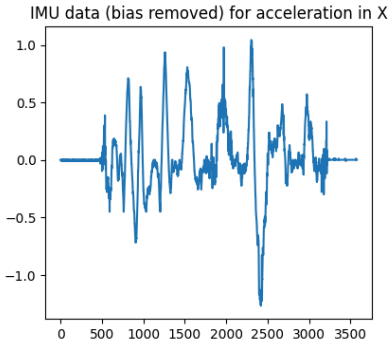
IMU raw data :

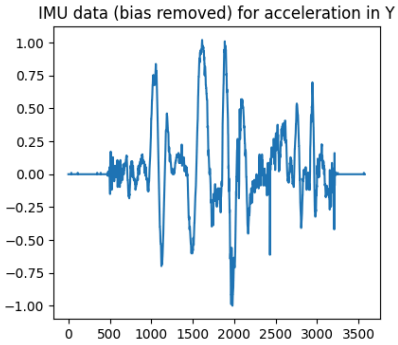


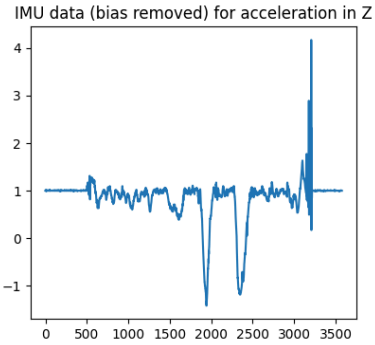


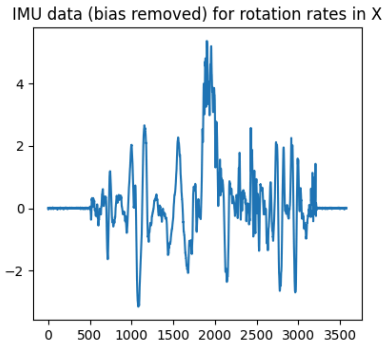


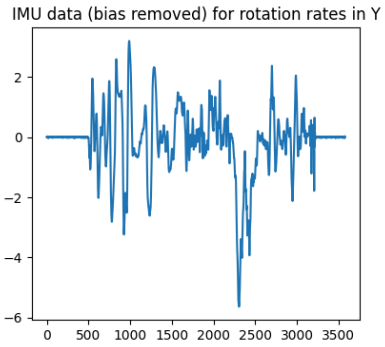
IMU data (after bias removal) :

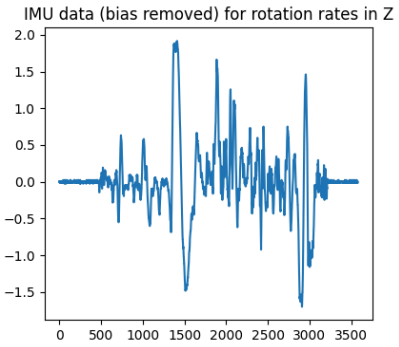




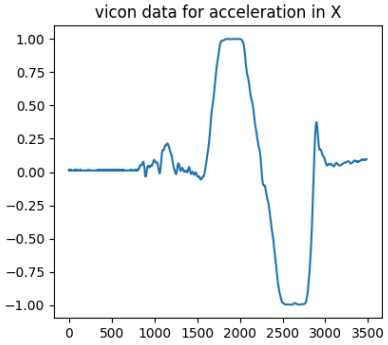


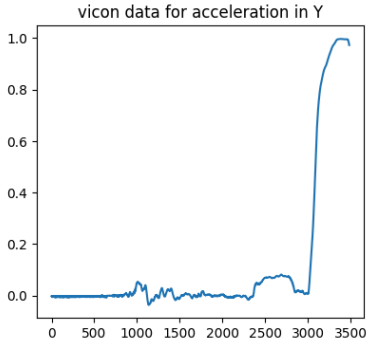


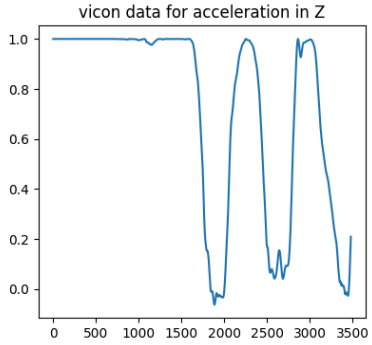




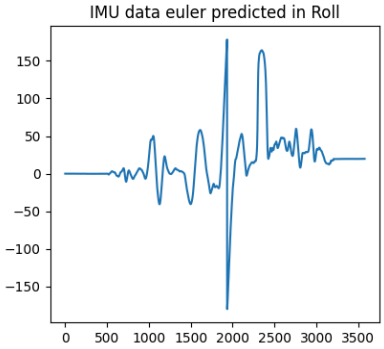
Vicon Data:

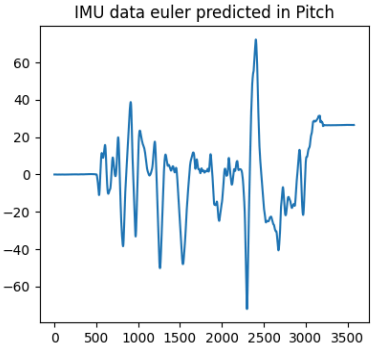


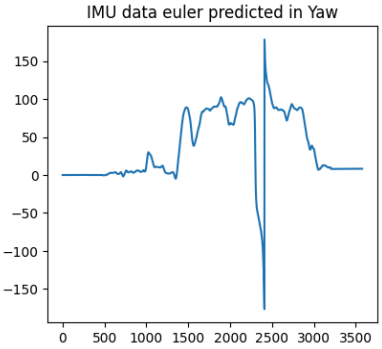




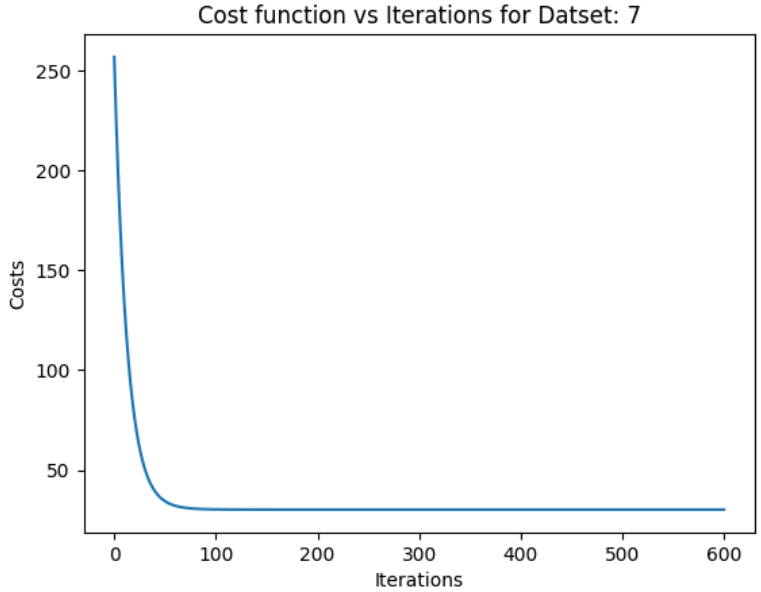
IMU estimated euler:







Cost function vs iterations:



IMU optimised euler: