Project 1: Orientation Tracking

Behrad Rabiei
Dept. of Electrical and Computer Engineering
University of California San Diego
Email: brabiei@ucsd.edu

***Abstract*—This paper is a report for Project 1 of ECE 276A (Sensing and Estimation in Robotics). The objective of the project is to track the orientation of a rigid body given sensor measurements by performing projected gradient descent. Additionally, we aim to generate a panorama by linking camera readings with rotations based on timestamps.**

I. Introduction

Localization, planning, mapping, and control are often referred to as the core components of robotic autonomy. This stack represents the fundamental capabilities required for a robot to navigate, interact with its environment, and perform tasks autonomously. Localization involves determining the robot's position and orientation relative to its environment. This typically utilizes sensors such as GPS, IMUs (Inertial Measurement Units), wheel encoders, and visual odometry to accurately estimate the robot's pose. In this project, we present an approach to perform localization for a camera equipped with an IMU by tracking its orientation over time. For the sake of validating our approach, we are provided with VICON (Motion Capture) data to serve as ground truth. As a bonus, given the camera images with corresponding timestamps, we aim to create a panorama that represents the images the camera recorded in 3D space. This can act as an additional measure to see if our approach works.

II. Problem Formulation

A. Orientation Tracking

We aim to perform localization for a camera equipped with an IMU by tracking its orientation over time. An image of the setup is provided in the Appendix. The readings from the IMU give us linear acceleration **(Ax, Ay, Az)** and angular velocity **(Wx, Wy, Wz)** in 3D space each with a corresponding timestamp. Let the linear acceleration data at time t be $\mathbf{a}_t = [Ax_t, Ay_t, Az_t]$, the angular velocity at time t $\omega_t = [Wx_t, Wy_t, Wz_t]$ and let $\mathbf{q}_t \in \mathbf{H}^*$ denote the quaternion representing the body-frame orientation at time t. Using the IMU angular velocity measurements ω_t and the differences between consecutive timestamps τ_t, we can predict the quaternion at the next step $\mathbf{q}_{t+1}$ using the quaternion kinematics **motion model**:

$$\mathbf{q}_{t+1} = f(\mathbf{q}_t, \tau_t\omega_t) := \mathbf{q}_t \circ exp([0, \tau_t\omega_t/2]). \tag{1}$$

Note that exp(·) is the exponential function for quaternions defined in Lecture 3.

Since the body is undergoing pure rotation, the acceleration of the body should be approximately $[0, 0, -g]$, in the world frame of reference, where g is the acceleration due to gravity. This is the case because gravity is always in play. Hence, the measured acceleration $\mathbf{a}_t$ in the IMU frame should agree with gravity acceleration after it is transformed to the IMU frame using the orientation $\mathbf{q}_t$, leading to the following **observation model**:

$$\mathbf{a}_t = h(\mathbf{q}_t) := \mathbf{q}_t^{-1} \circ [0, 0, 0, -g] \circ \mathbf{q}_t. \tag{2}$$

At this point, having both a model for our kinematics and our observation, we aim to estimate the orientation trajectory $\mathbf{q}_{1:T} := \mathbf{q}_1, \mathbf{q}_2, ..., \mathbf{q}_T$.

B. Panorama

After obtaining an estimate for our orientation trajectory, our objective is to generate a panoramic representation that encapsulates the visual data captured by the camera in 3D space. This entails seamlessly stitching together all the images, each captured at its unique orientation. The camera data comprises 320x240 RGB images, each paired with its corresponding timestamp. Our aim is to assemble these images into a cohesive larger frame, ensuring that their relative orientations are preserved throughout the panorama.

III. Technical Approach

A. Sensor Calibration

Note: The following is for dataset one; you can find the images for the other datasets in the Appendix.
Before we can get started, we first need to ensure that the data we have at hand is accurate and representative. We can first see a visualization for our IMU data in figure 1:

In the notes, it is mentioned that the IMU Ax and Ay directions are flipped (due to device design), so positive acceleration in the body frame will result in negative acceleration reported by the IMU. To correct this, we need to multiply Ax and Ay by -1. We also discover that the order of the angular velocity is [Wz, Wx, Wy], so we need to rearrange the angular velocity data. Additionally, the biases and scale factors of the accelerometers and gyroscopes are unknown in our case, so it is important to find those values. We need to look at the datasheets for the IMU's sensors and use the VICON data as ground truth to validate our data. The equations for the scale factor of the accelerometer and gyroscope are the following, respectively:

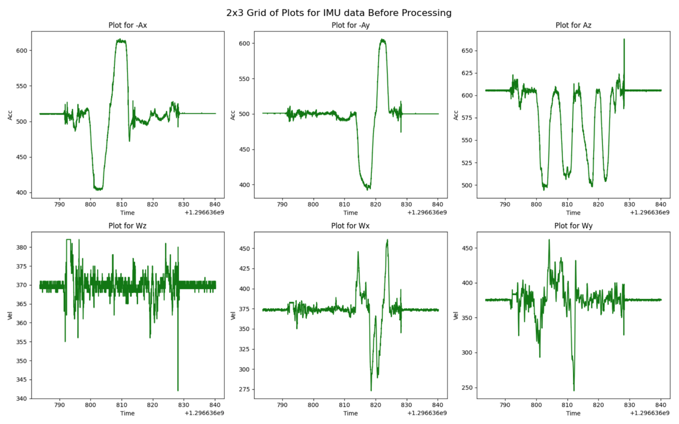


Fig. 1. IMU Data before Processing.

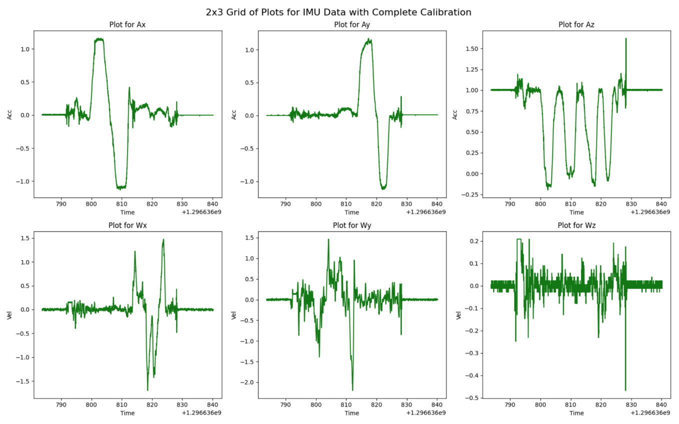


Fig. 2. IMU Data after Processing.

$$sfa = V_{ref}/1023/Sensitivity \tag{3}$$

$$sfg = V_{ref}/1023/Sensitivity * (\pi/180) \tag{4}$$

After investigating the datasheets, we find the V_{ref} is 3300 mV for both the accelerometer and gyroscope, and the sensitivities are 300 mV/g and 3.33 mV/deg/s, respectively. For the gyroscope, we need to multiply by $\pi/180$ to convert our unit to radians. In every dataset, the first few seconds are static, i.e., there is no rotation. We can use this static data portion to calibrate the IMU bias, e.g., the acceleration measured by the accelerometer in the first few seconds should be $[0, 0, 1]^T$ in gravity units. For my implementation, I compute the mean of the first 300 samples to get the bias for all six IMU readings. Now that we have the bias and the scale factor, we can compute the actual value of the IMU by using the following equation:

$$value = (raw - bias) \cdot sf \tag{5}$$

We need to adjust all 6 readings by their corresponding biases and scale factors. Finally, we need to remember to shift our acceleration in the z-axis by adding 1. This is to account for the constant acceleration caused by gravity. *Known glitch*: Some IMU datasets contain a small glitch in the beginning, where all 3 gyro values jump to a fixed value and then come back to normal operation (typically several seconds after starting). This problem occurred during data collection when a reset pin fired, locking the gyros into the nominal zero, rather than the true zero bias gyro level. From my analysis, this occurred in datasets 2, 4, and 9. Please refer to the Appendix for a visual representation. Our final calibrated IMU data can be seen in Figure 2.

Now that our IMU data is in order, we would like to visualize our VICON data for the sake of gaining intuition. The VICON data stores rotation matrices mapping the orientation of the body frame to the world frame. To plot the VICON data, we will get the Euler angles from the rotation matrices and plot rotation in the 3 axes independently. The angles in the x, y, and z axes over time are shown in Figure 3. *Note: The sudden 'noise' in the x and z axes are normal. This is because* $-180\,\deg$ *and* $180\,\deg$ *are the same.*

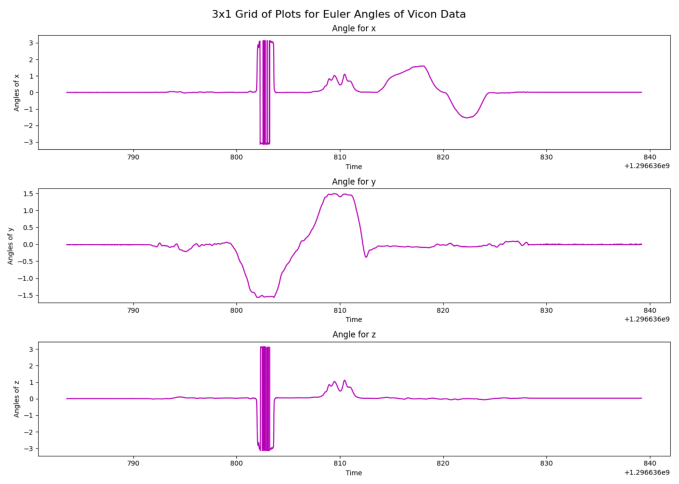


Fig. 3. Euler Angles of VICON.

Small programming note, the IMU data is stored as 'uint16'. This means that it cannot store negative or floating-point values. However, once we scale and shift the data, we need to be able to store both negative and floating-point values. To account for this, we need to convert the IMU data to 'float64'.

B. Orientation Tracking

Our objective for orientation tracking is to estimate the orientation of the body over time using the IMU angular velocity ω_t and linear acceleration $\mathbf{a}_t$ measurements. We can construct a cost function that formulates an optimization problem to estimate the orientation trajectory $\mathbf{q}_{1:T} := \mathbf{q}_1, \mathbf{q}_2, ..., \mathbf{q}_T$ based on the motion model in (1) and observation model in (2). The cost function for the optimization problem can be defined as:

$$c(\mathbf{q}_{1:T}) := \frac{1}{2}\sum_{t=0}^{T-1}\|2\log(\mathbf{q}_{t+1}^{-1}\circ f(\mathbf{q}_t,\tau_t\omega_t))\|_2^2 + \frac{1}{2}\sum_{t=1}^{T}\|\mathbf{a}_t - h(\mathbf{q}_t)\|_2^2 \tag{6}$$

Note that $\log(\cdot)$ is the logarithm function for quaternions defined in Lecture 3.

The first term evaluates the disparity between the estimated orientation and the prediction from the motion model, while the second term assesses the deviation between the acceleration measurements and the forecast from the observation model. The error in the motion model stems from the relative rotation $q_t^{-1}\circ f(q_t,\tau_t\omega_t)$ between the anticipated orientation $f(q_t,\tau_t\omega_t)$ and the estimated orientation q_{t+1}. This error is characterized by the axis-angle parametrization of the relative rotation error using the quaternion $\log(\cdot)$ function, measuring the rotation angle as the norm of the axis-angle vector. Throughout the optimization, it's imperative to maintain the constraint that the quaternions q_t retain unit norm, $q_t \in H^*$. Thus, the optimization problem is inherently constrained:

$$\min_{\mathbf{q}_{1:T}} c(\mathbf{q}_{1:T}) \tag{7}$$

$$\text{s.t. } \|\mathbf{q}_t\|_2 = 1 \quad \forall t\in\{1,2,\ldots,T\} \tag{8}$$

Initializing with $\mathbf{q}_0$ = [1, 0, 0, 0], we will implement a projected gradient descent algorithm to optimize the quaternion trajectory $\mathbf{q}_t$. With our initial $\mathbf{q}_0$, we will recursively feed it into the motion model (1) to get our initial trajectory estimate. Figure 4 shows a comparison of our initial trajectory against the VICON data.

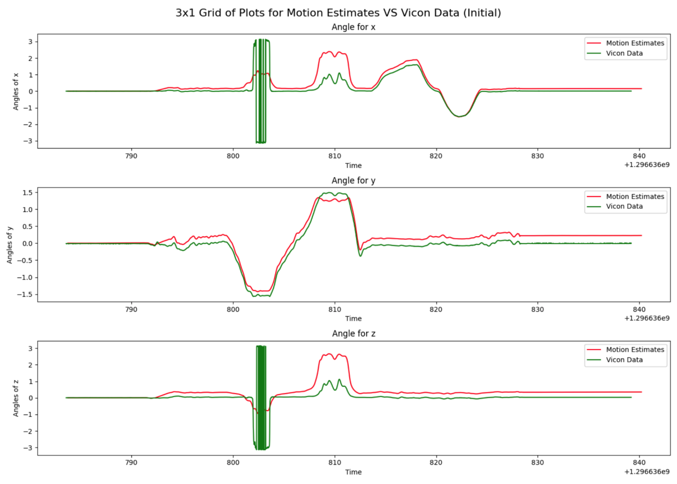


Fig. 4. Motion Estimate Vs. VICON (Initial).

We can see that our calibration is correct, and we are roughly estimating the true motion. Now with our trajectory initialized, we would like to test our observation model (2). We will feed our trajectory into the observation model and compare its output to the true acceleration data. This comparison can be seen in Figure 5.

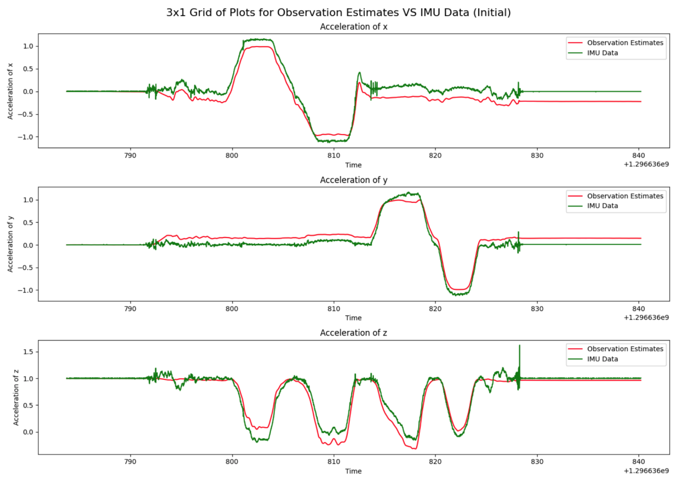


Fig. 5. Observation Estimate Vs. Accelerometer (Initial).

This again reinforces that our calibration is correct and that our observation model is roughly estimating the accelerometer data.

C. Panorama

Now that we have our trajectory estimates, our objective is to construct a panoramic image by stitching the RGB camera images over time based on the body orientation $\mathbf{q}_{1:T}$. To accomplish this, we must first find the longitude λ and latitude ϕ of each pixel using the number of rows and columns and the horizontal (60 degrees) and vertical (45 degrees) fields of view. This can be accomplished using the following equations:

$$\lambda = (\pi/180)\cdot((45/240)\cdot y - 22.5) \tag{9}$$

$$\phi = (\pi/180)\cdot((60/320)\cdot y - 30). \tag{10}$$

After finding these angles, we need to represent our angles in spherical coordinates by assuming the depth is 1 (i.e., $(\lambda,\phi,1)$). Then we will convert our spherical coordinates to Cartesian coordinates using the following equations:

$$x = -\sin(\phi)\cdot\cos(\lambda)$$
$$y = \cos(\phi)\cdot\sin(\lambda)$$
$$z = \sin(\phi)$$

At this point, we must rotate our Cartesian coordinate. We can accomplish this by multiplying our point with R^T where R is the rotation matrix representation of our estimated quaternion $\mathbf{q}_t$. Now that we have rotated our pixels, we now need to go back to spherical coordinates. This can be accomplished using the following equations:

$$r = 1$$
$$\phi = \arcsin\left(\frac{z}{r}\right)$$
$$\lambda = \arctan 2(y, -x)$$

Now we must map these angles back to some arbitrary pixel coordinate. For my implementation, I have constructed an image that is larger than the original images that we were given. This image is where the panorama will be projected. The dimensions are 480x960. The equations for finding the new pixel coordinates are:

$$\text{row} = \left(\frac{479}{\pi}\right)\left(\frac{\pi}{2} + \phi\right)$$
$$\text{col} = \left(\frac{959}{2\pi}\right)(\pi + \lambda)$$

Finally, with the new pixel coordinates at our disposal, we can copy the pixel values from the original image to the panorama.

IV. Results

V. Orientation Tracking

Note: The following is for dataset one; you can find the images for the other datasets in the Appendix.
After performing the constraint gradient descent, we attain a much more accurate estimate for both the trajectory and observation estimates. Figures 6 and 7 show our final estimates after training, and Figure 8 shows the loss curve during training. Note, for datasets 2, 4 and 9, where the reset button is triggered, we get a much worse initial estimate but drastically improve our estimates after training. With that said, there is an offset that appears in the yaw angle.

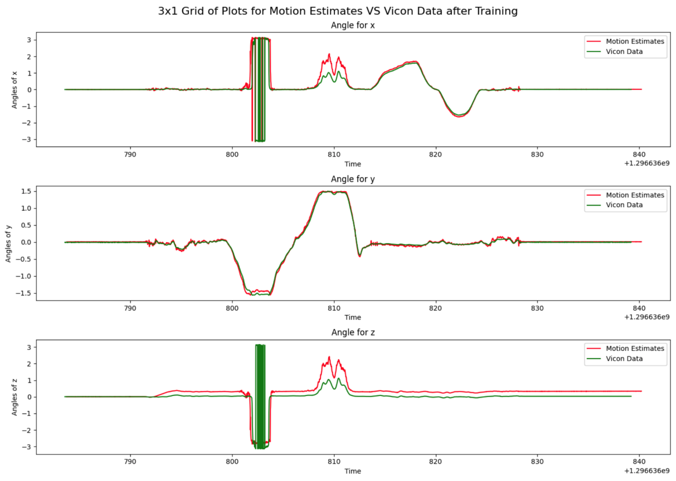


Fig. 6. Motion Estimate Vs. VICON (Final).

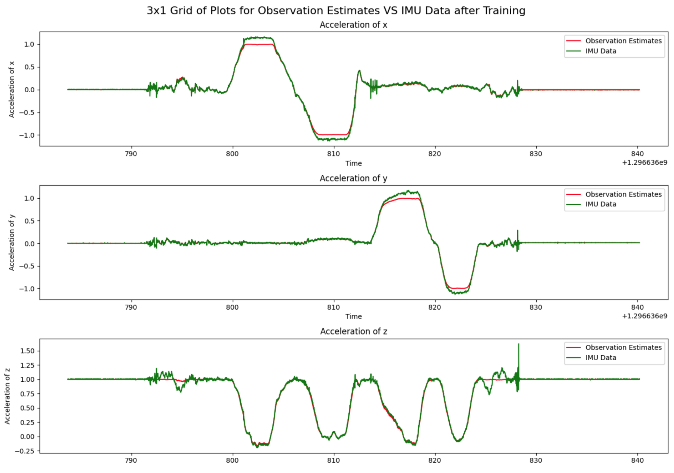


Fig. 7. Observation Estimate Vs. Accelerometer (Final).

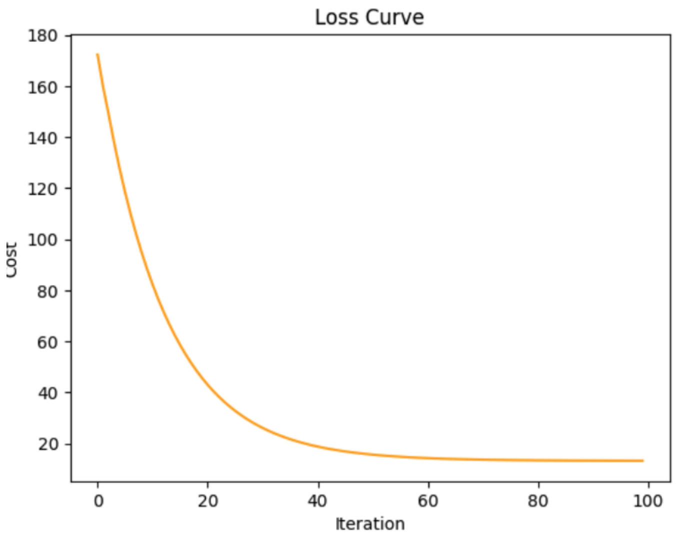


Fig. 8. Loss Curve.

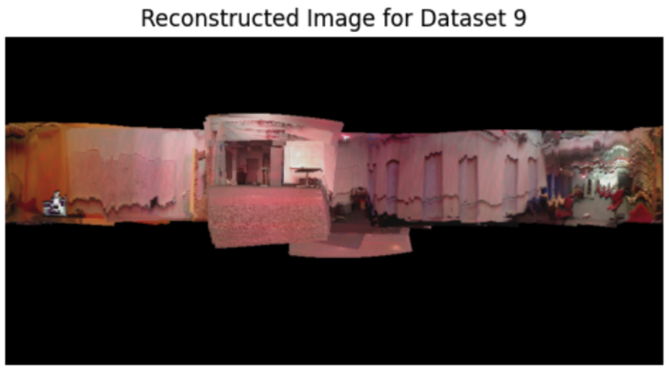


Fig. 9. Final Panoramic.

VI. Panorama

Note: The following is for dataset 9; you can find the images for the other datasets in the Appendix.

After following the steps we mentioned, we get a rough panoramic that relates the relative positions of the images. Please look at Figure 9 for the final result. The final result looks fairly well for this dataset. However, certain datasets look quite distorted. I assume that this is caused by the camera tilting too high or too low. Please refer to the appendix for other dataset results.

References

[1] https://en.wikipedia.org/wiki/Spherical coordinate system
[2] https://jax.readthedocs.io/en/latest/
[3] https://natanaso.github.io/ece276a/

APPENDIX

A. Camera Setup

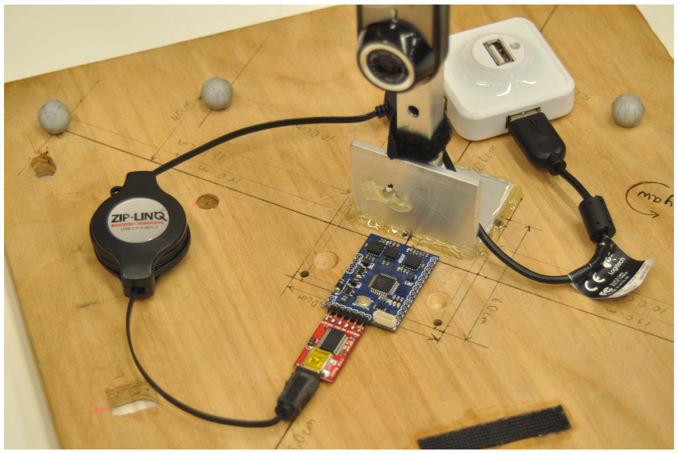


Fig. 10. Camera Setup.

B. Panoramics

Dataset 1: Dataset 2: Dataset 8: Dataset 9: Dataset 10:

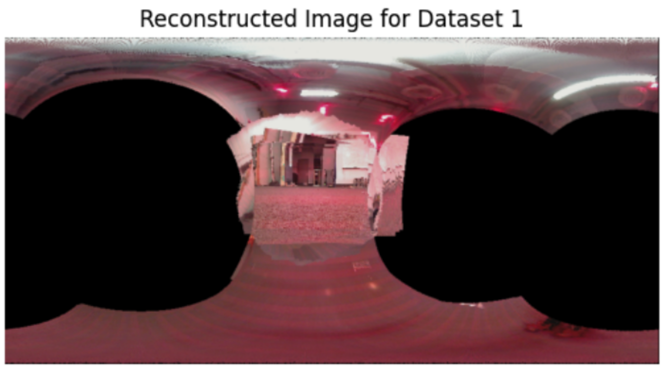


Fig. 11. Panoramic for Dataset 1.

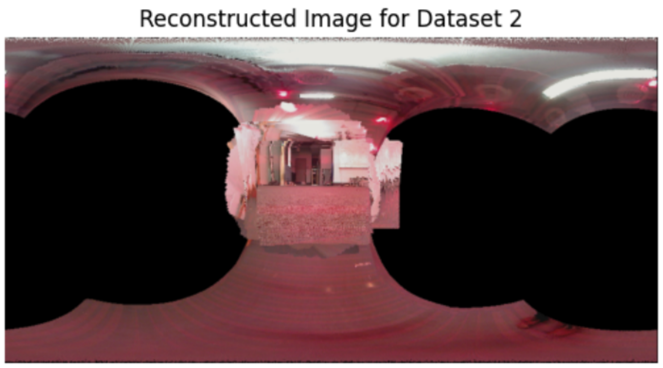


Fig. 12. Panoramic for Dataset 2.

Dataset 11:

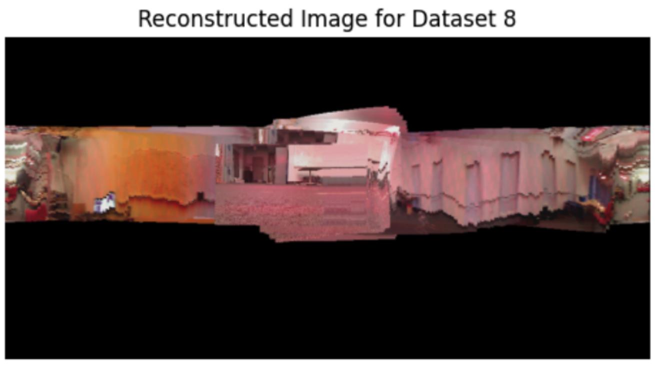


Fig. 13. Panoramic for Dataset 8.

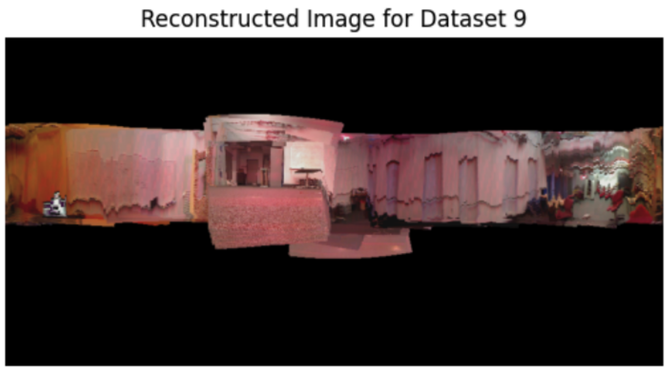


Fig. 14. Panoramic for Dataset 9.

C. Motion Estimate VS VICON (Initial)

Dataset 2: Dataset 3: Dataset 4: Dataset 5: Dataset 6: Dataset 7: Dataset 8: Dataset 9: Dataset 10: Dataset 11:

D. Motion Estimate VS VICON (Final)

Dataset 2: Dataset 3: Dataset 4: Dataset 5: Dataset 6: Dataset 7: Dataset 8: Dataset 9: Dataset 10: Dataset 11:

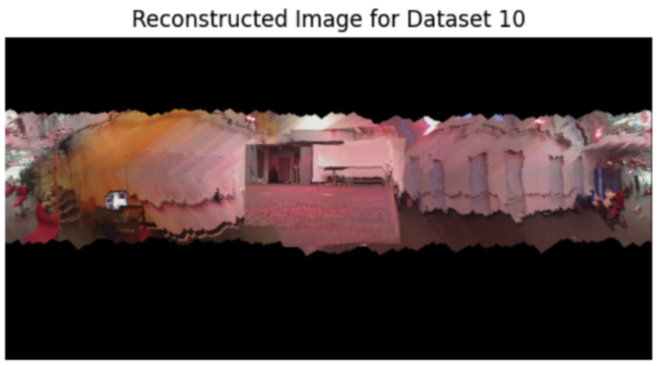
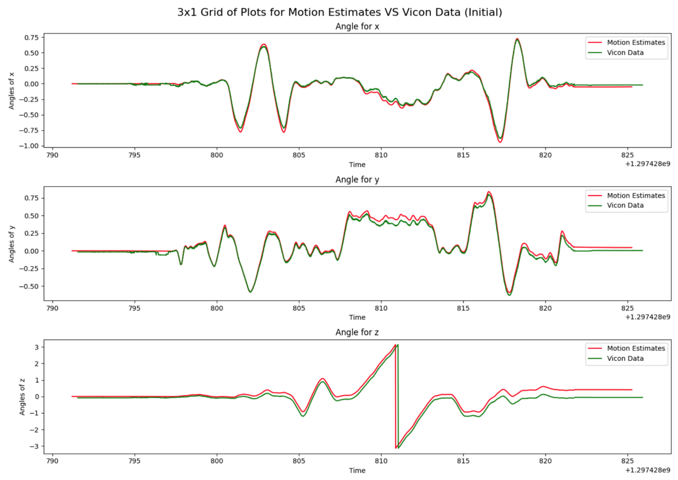


Fig. 15. Panoramic for Dataset 10.



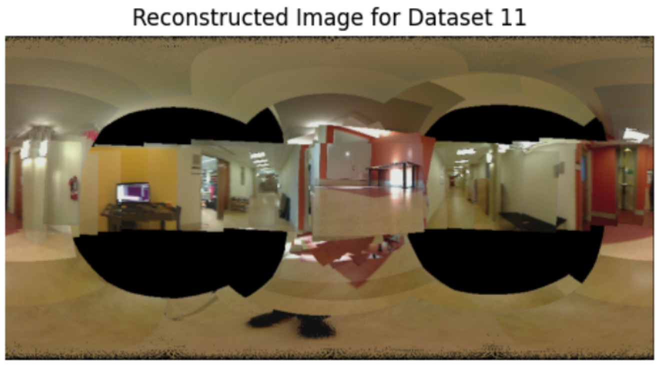
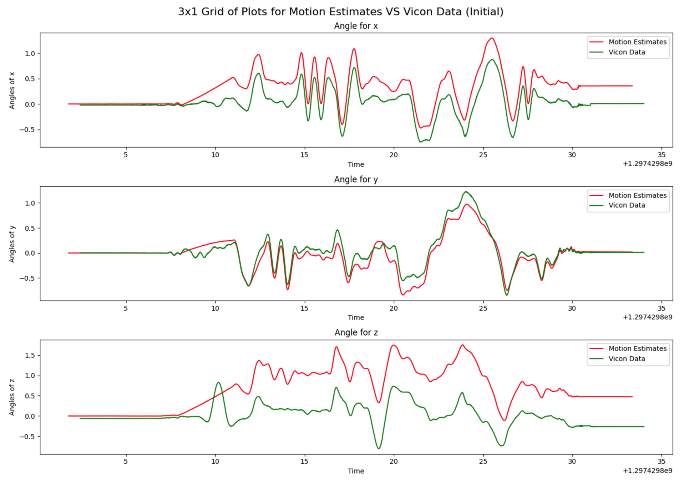
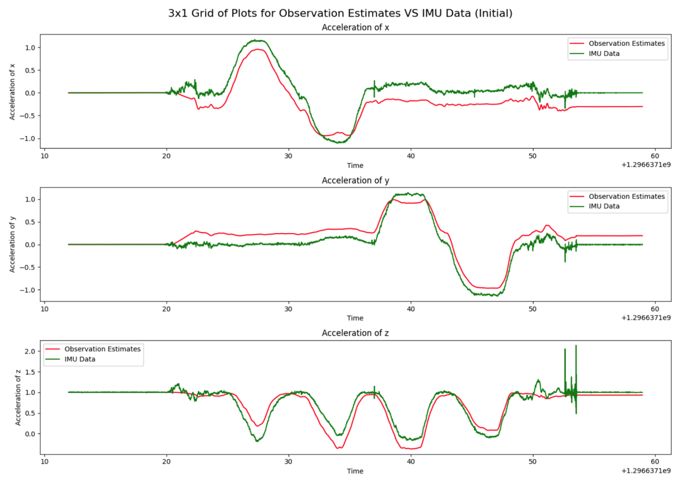
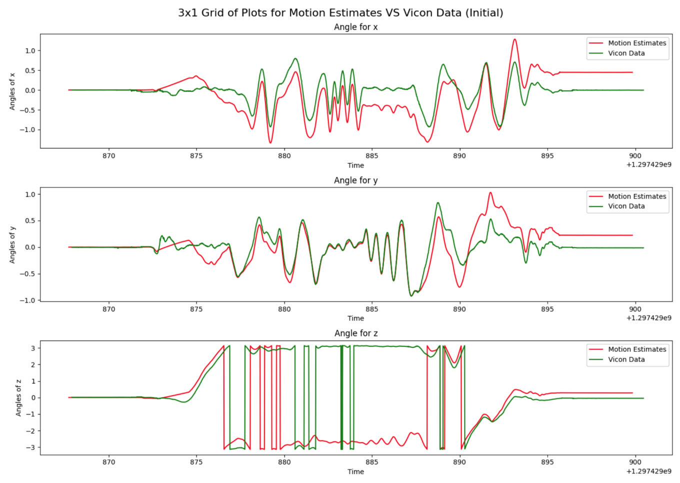
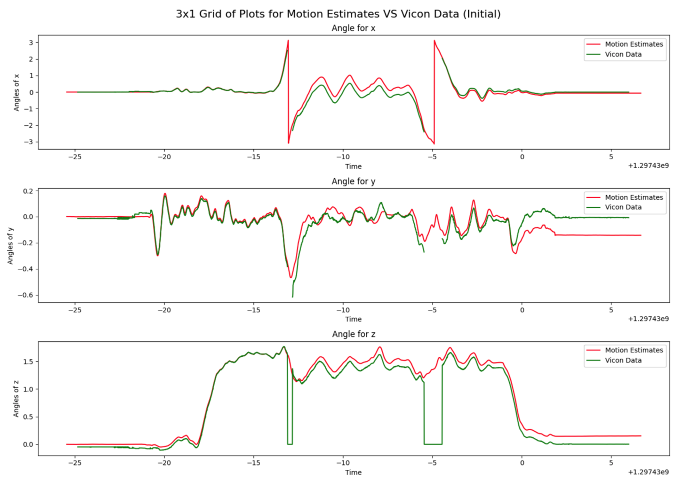


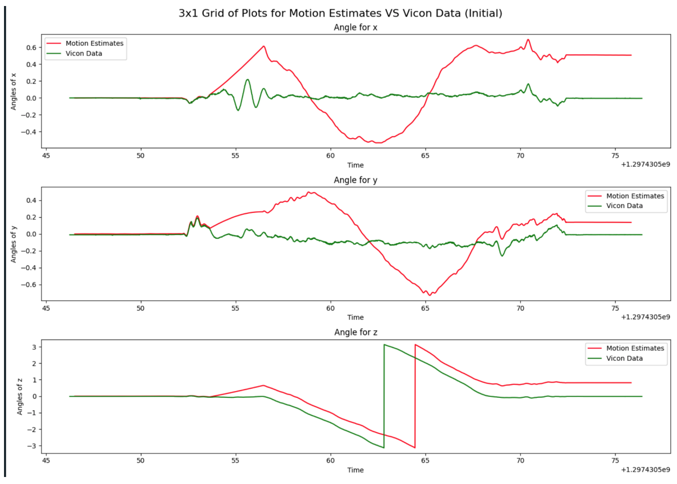
Fig. 16. Panoramic for Dataset 11.

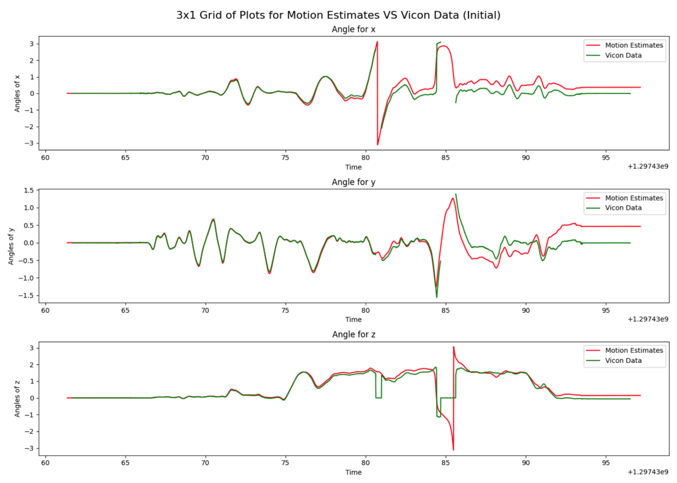


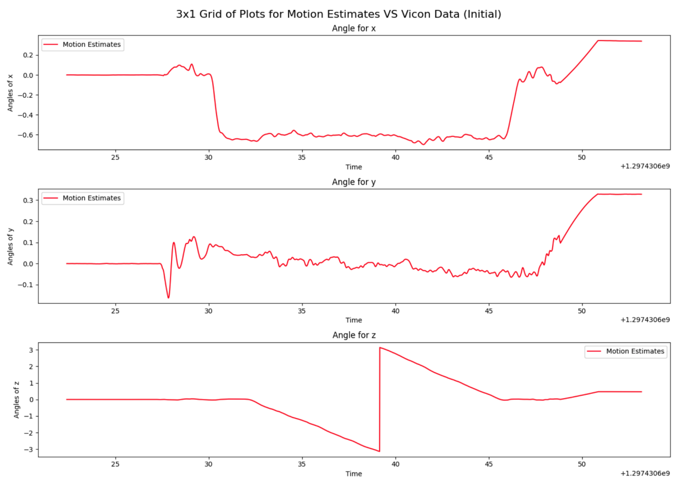


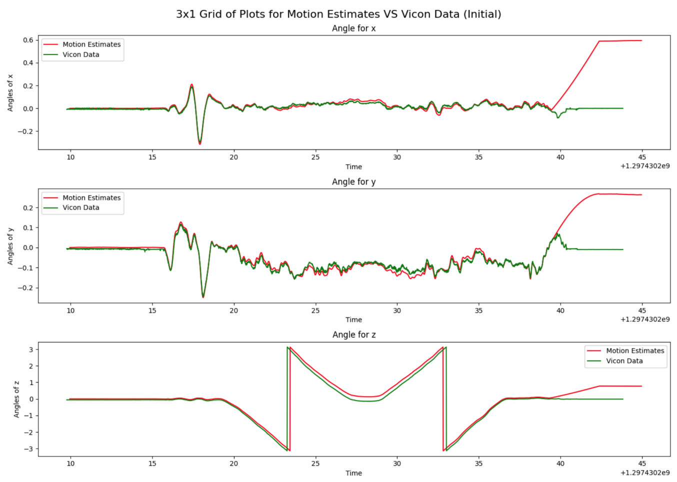


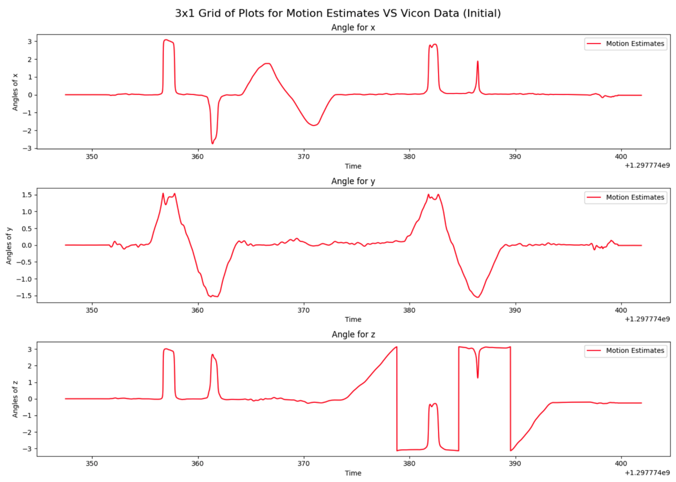

3x1 Grid of Plots for Motion Estimates VS Vicon Data (Initial)
Angle for x
Angle for y
Angle for z
Angles of x
Angles of y
Angles of z
Time
Motion Estimates
Vicon Data
+1.29743e9

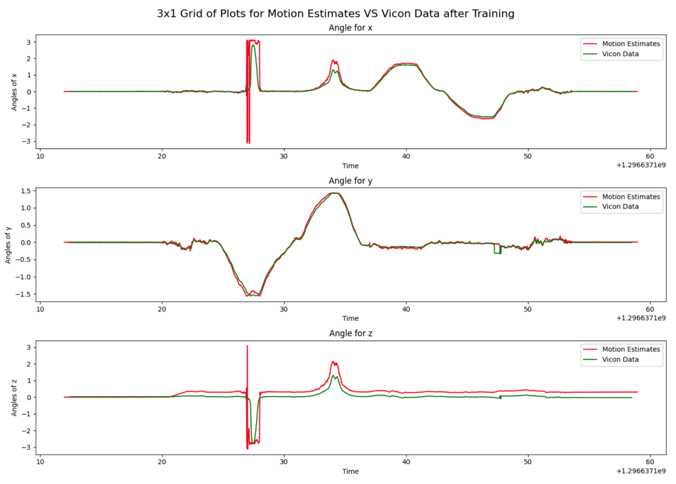

3x1 Grid of Plots for Motion Estimates VS Vicon Data (Initial)
Angle for x
Angle for y
Angle for z
Angles of x
Angles of y
Angles of z
Time
Motion Estimates
Vicon Data
+1.2974305e9

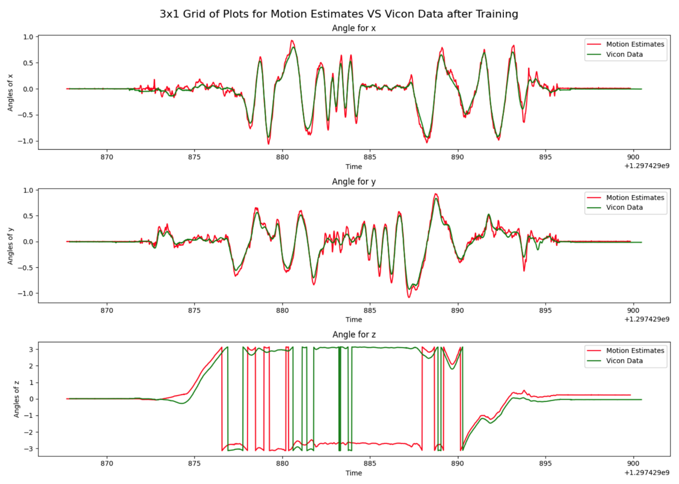

3x1 Grid of Plots for Motion Estimates VS Vicon Data (Initial)
Angle for x
Angle for y
Angle for z
Angles of x
Angles of y
Angles of z
Time
Motion Estimates
Vicon Data
+1.29743e9

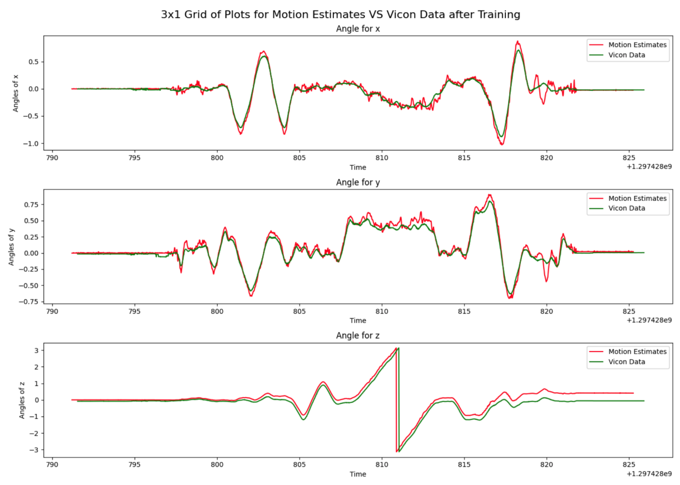

3x1 Grid of Plots for Motion Estimates VS Vicon Data (Initial)
Angle for x
Angle for y
Angle for z
Angles of x
Angles of y
Angles of z
Time
Motion Estimates
+1.2974306e9

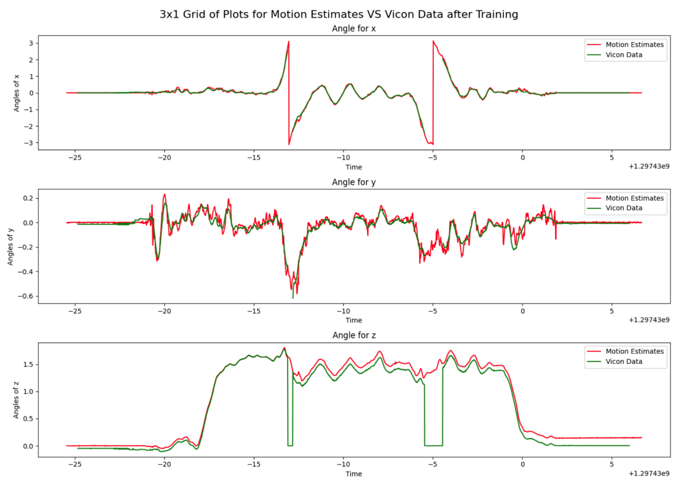

3x1 Grid of Plots for Motion Estimates VS Vicon Data (Initial)
Angle for x
Angle for y
Angle for z
Angles of x
Angles of y
Angles of z
Time
Motion Estimates
Vicon Data
+1.2974302e9

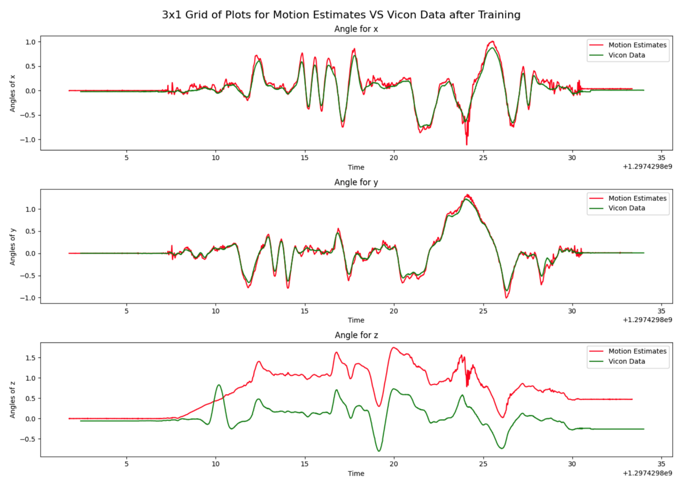

3x1 Grid of Plots for Motion Estimates VS Vicon Data (Initial)
Angle for x
Angle for y
Angle for z
Angles of x
Angles of y
Angles of z
Time
Motion Estimates
+1.297774e9

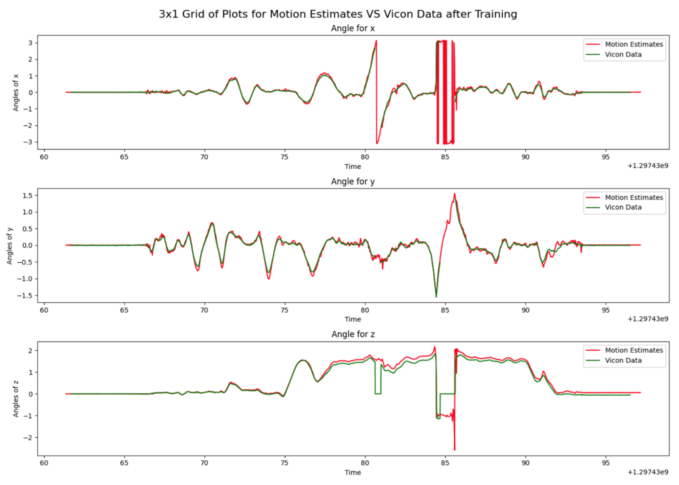

3x1 Grid of Plots for Motion Estimates VS Vicon Data after Training
Angle for x
Angle for y
Angle for z
Angles of x
Angles of y
Angles of z
Time
+1.2966371e9
Motion Estimates
Vicon Data

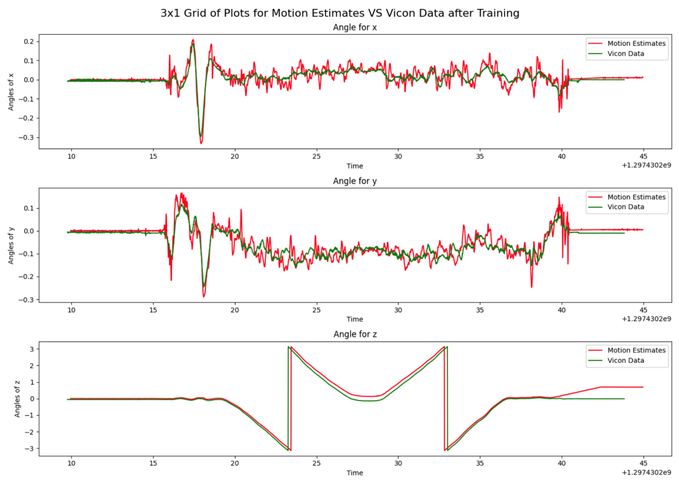

3x1 Grid of Plots for Motion Estimates VS Vicon Data after Training
Angle for x
Angle for y
Angle for z
Angles of x
Angles of y
Angles of z
Time
+1.297429e9
Motion Estimates
Vicon Data

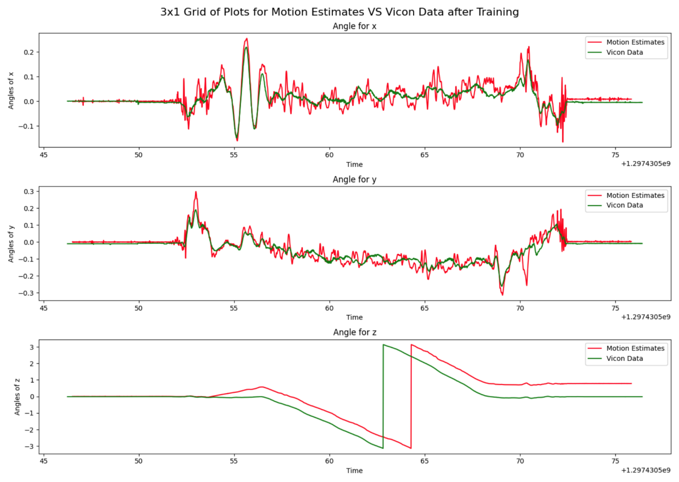

3x1 Grid of Plots for Motion Estimates VS Vicon Data after Training
Angle for x
Angle for y
Angle for z
Angles of x
Angles of y
Angles of z
Time
+1.297428e9
Motion Estimates
Vicon Data

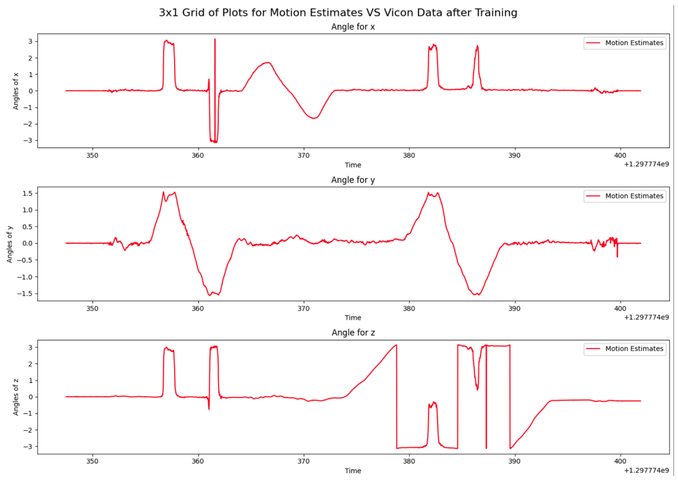

3x1 Grid of Plots for Motion Estimates VS Vicon Data after Training
Angle for x
Angle for y
Angle for z
Angles of x
Angles of y
Angles of z
Time
+1.29743e9
Motion Estimates
Vicon Data

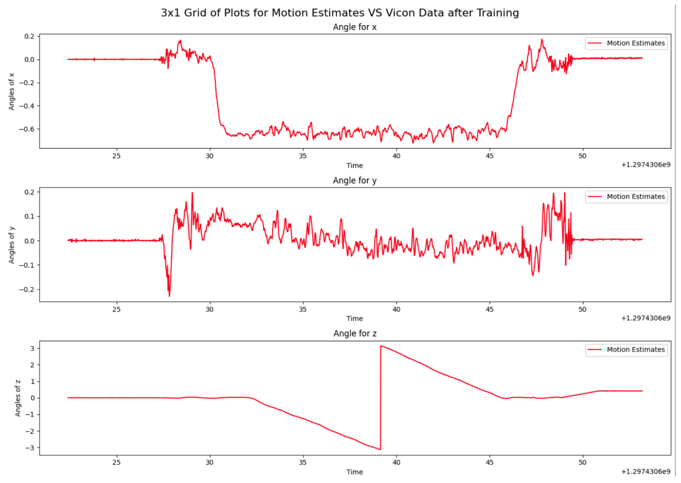

3x1 Grid of Plots for Motion Estimates VS Vicon Data after Training
Angle for x
Angle for y
Angle for z
Angles of x
Angles of y
Angles of z
Time
+1.2974298e9
Motion Estimates
Vicon Data


3x1 Grid of Plots for Motion Estimates VS Vicon Data after Training
Angle for x
Angle for y
Angle for z
Angles of x
Angles of y
Angles of z
Time
+1.29743e9
Motion Estimates
Vicon Data


3x1 Grid of Plots for Motion Estimates VS Vicon Data after Training
Angle for x
Angles of x
Angle for y
Angles of y
Angle for z
Angles of z
Time
+1.2974302e9
Motion Estimates
Vicon Data


3x1 Grid of Plots for Motion Estimates VS Vicon Data after Training
Angle for x
Angles of x
Angle for y
Angles of y
Angle for z
Angles of z
Time
+1.2974305e9
Motion Estimates
Vicon Data


3x1 Grid of Plots for Motion Estimates VS Vicon Data after Training
Angle for x
Angles of x
Angle for y
Angles of y
Angle for z
Angles of z
Time
+1.297774e9
Motion Estimates


3x1 Grid of Plots for Motion Estimates VS Vicon Data after Training
Angle for x
Angles of x
Angle for y
Angles of y
Angle for z
Angles of z
Time
+1.2974306e9
Motion Estimates

Contents