ECE276A HW1

1st Armon Barakchi
abarakchi@ucsd.edu
A16950466

***Abstract*—In robotics, state estimation is an integral task that determines the success of all downstream actions. In this project, we consider a rigid body that undergoes pure rotational motion. The unit quaternion provides a valid parameterization of three dimensional rotation. I present my work on estimating unit quaternions from IMU measurements via a gradient descent method. This estimated orientation trajectory is then used, in conjunction with an onboard camera, to create a visual representation of the surroundings through panoramic construction.**

***Index Terms*—quaternion, orientation tracking, panorama construction**

I. Introduction

The field of autonomous robotics concerns itself with two interdependent problems: state estimation (observation) and motion planning. Clearly, the latter is impossible without the former – the next planned action is directly dependent on the accuracy of the estimated state.

In this context, state estimation seeks to estimate the pose and motion from noisy measurements. An essential first step in that process is elucidating the robot's orientation relative to the world. The orientation determines how the robot's sensor observations relate to the world frame $\{\mathbf{W}\}$. Errors in orientation estimation can propagate to downstream tasks such as motion planning, leading to unsafe/unexpected behaviors.

Accurate orientation tracking can also be used, in conjunction with cameras, to construct visual representations, or *maps*, of the surrounding environment. This enhances the robot's capabilities to understand the surrounding scenes – an important task in today's world of autonomous robots working with humans. Consequently, stable and reliable orientation tracking is an important capability for systems working in real-world environments.

II. Problem Formulation

A. Orientation Tracking

We model a robot to be a rigid body. This means that the distance between all points on the robot's structure remains constant over all time. Consider the robot to be moving in a fixed world reference frame $\{\mathbf{W}\}$. It is sufficient to characterize the motion of one point $\mathbf{p}(t) \in \mathbb{R}^3$ and three coordinate axes $\mathbf{r}_1(t), \mathbf{r}_2(t), \mathbf{r}_3(t)$ attached to the point. A point $\mathbf{s}$ on the rigid body has fixed coordinates $\mathbf{s}_B \in \mathbb{R}^3$ in the body frame $\{\mathbf{B}\}$, but time-varying coordinates $\mathbf{s}_W(t) \in \mathbb{R}^3$ in the world frame $\{\mathbf{W}\}$. The primary task is finding the rotation and translation of $\{\mathbf{B}\}$ relative to $\{\mathbf{W}\}$.

In three dimensions, the motion of a rigid body consists of three translational and three rotational degrees of freedom. This is captured by the **Special Euclidean Group** $\mathrm{SE}(3)$**:** the set of functions $g : \mathbb{R}^3 \to \mathbb{R}^3$ that preserve the norm and the cross product of any two vectors:

- **Norm preservation:**

$$\|g^*(\mathbf{u}) - g^*(\mathbf{v})\| = \|\mathbf{v} - \mathbf{u}\|, \quad \forall\, \mathbf{u}, \mathbf{v} \in \mathbb{R}^3.$$

- **Cross-product preservation:**

$$g^*(\mathbf{u}) \times g^*(\mathbf{v}) = g^*(\mathbf{u} \times \mathbf{v}), \quad \forall\, \mathbf{u}, \mathbf{v} \in \mathbb{R}^3.$$

where

$$g^*(\mathbf{x}) := g(\mathbf{x}) - g(\mathbf{0}).$$

This characterization captures the functions that map translational and rotational movements from the body frame to the world frame. However, pure rotational motion is a special case of rigid body motion which can be captured in the following way:

$${}_{\{W\}}R_{\{B\}} = \begin{bmatrix}\mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3\end{bmatrix} \in \mathbb{R}^{3\times 3}.$$

Which describes the orientation of $\{\mathbf{B}\}$ in $\{\mathbf{W}\}$ by three orthogonal unit vectors:

$$\mathbf{r}_1 = g(\mathbf{e}_1), \quad \mathbf{r}_2 = g(\mathbf{e}_2), \quad \mathbf{r}_3 = g(\mathbf{e}_3),$$

where $g \in \mathrm{SE}(3)$.
Clearly, the matrix ${}_{\{\mathbf{W}\}}\mathbf{R}_{\{\mathbf{B}\}}$ is in the following group:

$$SO(3) := \left\{ R \in \mathbb{R}^{3\times 3} \mid R^\top R = R R^\top = I, \det(R) = 1 \right\}.$$

Members of this group preserve distance and don't allow reflections making them a representation of rotations. We will now explore quaternions as a way of obtaining and tracking ${}_{\{\mathbf{W}\}}\mathbf{R}_{\{\mathbf{B}\}}$.
Quaternions extend complex numbers to four dimensions and can be used to describe three dimensional rotations.

$$\mathbb{H} = \mathbb{C} \oplus \mathbb{C}j,$$

A quaternion $\mathbf{q} \in \mathbb{H}$ can be expressed as

$$\mathbf{q} = q_s + q_1 i + q_2 j + q_3 k = (q_s + q_1 i) + (q_2 + q_3 i)j = [\, q_s,\ \mathbf{q}_v \,],$$

where $q_s \in \mathbb{R}$ is the scalar part and $\mathbf{q}_v \in \mathbb{R}^3$ is the vector part.

Three-dimensional rotations can be represented as members of the set of unit quaternions

$$\mathbb{H}^* := \left\{ \mathbf{q} \in \mathbb{H} \,\middle|\, q_s^2 + \mathbf{q}_v^\top \mathbf{q}_v = 1 \right\}.$$

Given a unit quaternion $\mathbf{q} \in \mathbb{H}^*$, the corresponding rotation matrix $R(\mathbf{q}) \in \mathrm{SO}(3)$ is given by

$$R(\mathbf{q}) = E(\mathbf{q})\, G(\mathbf{q})^\top,$$

where

$$E(\mathbf{q}) = \begin{bmatrix} -\mathbf{q}_v & q_s I + \widehat{\mathbf{q}}_v \end{bmatrix}, \qquad G(\mathbf{q}) = \begin{bmatrix} -\mathbf{q}_v & q_s I - \widehat{\mathbf{q}}_v \end{bmatrix}.$$

Note that every rotation matrix $R \in \mathrm{SO}(3)$ can be obtained from two different $q \in \mathbb{H}^*$.

In this problem, as previously mentioned, the robot is modeled as a rigid body equipped with an inertial measurement unit (IMU). As established, the orientation of the body frame relative to the world frame is fully captured by

$${}_{\{W\}}R_{\{B\}}(t) \in \mathrm{SO}(3),$$

or equivalently by a unit quaternion

$$\mathbf{q}(t) \in \mathbb{H}^*,$$

The IMU provides time-stamped measurements of the angular velocity and linear acceleration expressed in the IMU frame, which, for simplicity, will be assumed to be the body frame. However, these measurements are corrupted by noise and cannot be used directly. The goal of this project is to estimate the orientation of the rigid body by recovering a sequence of unit quaternions

$$\{\mathbf{q}_t\}_{t=1}^{T} \subset \mathbb{H}^*$$

that is consistent with the available IMU measurements. Thus, the central problem addressed in this section is the estimation of an orientation trajectory evolving on the manifold $\mathrm{SO}(3)$, or equivalently on the unit-quaternion manifold $\mathbb{H}^*$, from noisy measurements. As discussed previously, estimating $\mathbf{q}(t)$ is equivalent to estimating the orientation of $\{\mathbf{B}\}$ in $\{\mathbf{W}\}$ so this formulation of the problem is sufficient to answer the orientation tracking problem.

B. Panorama Construction

For this section, consider that the robot has an RGB camera attached. Let $\{\mathbf{O}\}$ denote the optical frame and $\{\mathbf{W}\}$ the fixed world frame. The camera pose at time t is fully determined by its orientation in the absence of translation

$${}_{\{W\}}R_{\{O\}}(t) \in \mathrm{SO}(3),$$

which is assumed known from the orientation estimation problem described above. The camera position is assumed fixed as described in the problem description.

At discrete times $\{t_k^{\mathrm{cam}}\}_{k=1}^{K}$, the camera acquires an image

$$\{I_k\}, \qquad I_k \in \mathbb{R}^{H \times W \times 3}.$$

where the timestamps are provided by the camera.

Each pixel (u, v) in image I_k corresponds to a bearing direction in the optical frame. A bearing direction is the unit vector representing the direction of a camera ray corresponding to a pixel. Under the pinhole camera model with calibration matrix

$$K \in \mathbb{R}^{3\times 3},$$

the bearing vector associated with pixel (u, v) is given by

$$\mathbf{x}_k(u,v) = \frac{K^{-1}\begin{bmatrix} u \\ v \\ 1 \end{bmatrix}}{\left\| K^{-1}\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \right\|} \in \mathbb{S}^2.$$

This bearing direction expressed in the world frame is obtained via the rotation

$$\mathbf{y}_k(u,v) = {}_{\{W\}}R_{\{O\}}(t_k)\, \mathbf{x}_k(u,v) \in \mathbb{S}^2.$$

With all that established, the panorama is an image that parameterizes directions on the unit sphere, or equivalently a cylindrical projection of $\mathbb{S}^2$. Let

$$\pi : \mathbb{S}^2 \to \Omega_P$$

denote the chosen spherical or cylindrical projection mapping world-frame directions to pixel coordinates in the panorama domain Ω_P.

The panorama construction problem is to determine an image

$$P : \Omega_P \to \mathbb{R}^3$$

such that, for all camera images and pixels, the color observed at (u, v) in image I_k is assigned to the panorama location corresponding to the rotated viewing direction:

$$P(\pi(\mathbf{y}_k(u,v))) \leftarrow I_k(u,v), \quad \forall\, k,\ \forall\, (u,v).$$

When multiple image pixels map to the same panorama location, an aggregation rule is applied.

Note: The information in this section was gathered from [1] and [2]

III. Technical Approach

A. Orientation Tracking

As previously mentioned, the goal of the orientation tracking component is to estimate the time-varying orientation of the body frame $\{\mathbf{B}\}$ relative to the world frame $\{\mathbf{W}\}$ using measurements from the onboard IMU.

1) IMU Calibration: To begin this task, we must consider that the IMU may not be faithful to the true state of the robot due to sensor biases. To that end, we must perform a calibration step to estimate the bias of the IMU and then remove it.

In this project, calibration is performed using an initial static time interval where the true values of angular velocity and linear acceleration are known to be zero and just the gravitational force, respectively, followed by a validation procedure that compares gyro-only orientation integration against VICON ground-truth rotations.

The raw IMU data are provided as ADC counts and are corrupted by sensor biases. Let the raw measurements at time index t be

$$\boldsymbol{\omega}_t^{\mathrm{raw}} \in \mathbb{R}^3, \qquad \mathbf{a}_t^{\mathrm{raw}} \in \mathbb{R}^3,$$

corresponding to the gyroscope and accelerometer, respectively.

We assume that the rigid body is stationary during an initial time window

$$t \in [t_0,\, t_0 + T_{\text{static}}],$$

In my approach, I have used $t_0 = 0$ and $T_{\text{static}} = 5$. Under this assumption, the sensor models are

$$\boldsymbol{\omega}_t^{\text{raw}} = \boldsymbol{\omega}_t + \mathbf{b}_\omega + \boldsymbol{\eta}_{\omega,t}, \qquad \mathbf{a}_t^{\text{raw}} = \mathbf{a}_t + \mathbf{b}_a + \boldsymbol{\eta}_{a,t},$$

where $\mathbf{b}_\omega$ and $\mathbf{b}_a$ denote constant gyroscope and accelerometer biases and $\boldsymbol{\eta}_{w,t}$ and $\boldsymbol{\eta}_{a,t}$ are measurement noise. The noise will be ignored for this problem as we have not learned about kalman filtering yet.

a) Gyroscope Bias Estimation: During the static interval, $\boldsymbol{\omega}_t \approx \mathbf{0}$. The gyroscope bias is therefore estimated as the mean raw measurement:

$$\mathbf{b}_\omega \approx \frac{1}{T_{\text{static}}} \sum_{t \in \mathcal{S}} \boldsymbol{\omega}_t^{\text{raw}},$$

where $\mathcal{S}$ denotes the set of indices in the static window. The bias-corrected gyroscope measurements are then

$$\boldsymbol{\omega}_t^{\text{cal}} = \boldsymbol{\omega}_t^{\text{raw}} - \mathbf{b}_\omega.$$

b) Accelerometer Bias Estimation: When static, the accelerometer should measure gravity with unit magnitude. We assume the gravity vector expressed in the body frame is

$$\mathbf{g}_B = [0,\, 0,\, 1]^\top \text{ (g-units)}.$$

The accelerometer bias is estimated as

$$\mathbf{b}_a \approx \frac{1}{T_{\text{static}}} \sum_{t \in \mathcal{S}} \mathbf{a}_t^{(g)} - \mathbf{g}_B,$$

and the calibrated accelerometer measurements are

$$\mathbf{a}_t^{\text{cal}} = \mathbf{a}_t^{(g)} - \mathbf{b}_a.$$

To validate the effectiveness of the calibration, we compare a gyro-only orientation estimate against ground-truth orientations obtained from a VICON motion capture system.

c) Conversion to Physical Units: Raw ADC counts are converted to physical units using the ADC reference voltage and datasheet sensitivities. Let V_{ref} denote the ADC reference voltage and N_{ADC} the ADC resolution. The voltage-per-count scale is

$$V_{\text{count}} = \frac{V_{\text{ref}}}{N_{\text{ADC}}}.$$

Using the accelerometer sensitivity S_a (V/g) and gyroscope sensitivity S_ω (V/(deg/s)), the scaled measurements are

$$\mathbf{a}_t^{(g)} = V_{\text{count}}\, \mathbf{a}_t^{\text{cal}} / S_a, \qquad \boldsymbol{\omega}_t^{(\text{deg}/s)} = V_{\text{count}}\, \boldsymbol{\omega}_t^{\text{cal}} / S_\omega.$$

The gyroscope measurements are then converted to radians per second.

d) Gyro-Only Orientation Integration: Starting from the identity orientation

$$\mathbf{q}_0 = [1,\, 0,\, 0,\, 0]^\top,$$

the calibrated angular velocity measurements $\boldsymbol{\omega}_k^{\text{cal}}$ are integrated using unit quaternions. Over a time interval $\Delta t_t = t_{t+1} - t_t$, the incremental rotation is

$$\Delta \mathbf{q}_t = \exp\big([0,\ \tfrac{1}{2}\boldsymbol{\omega}_t^{\text{cal}} \Delta t_t]\big),$$

and the orientation is propagated as

$$\mathbf{q}_{t+1} = \mathbf{q}_t \circ \Delta \mathbf{q}_t,$$

followed by normalization to enforce $\|\mathbf{q}_{t+1}\| = 1$. Each quaternion is converted to a rotation matrix ${}_{\{W\}}R_{\{B\}}^{\text{imu}}(t_t) \in \text{SO}(3)$.

e) Time Alignment with VICON: Because the IMU and VICON systems operate at different sampling rates, VICON rotations are aligned to IMU timestamps using a nearest-past heuristic. For each IMU time t_t, the corresponding VICON rotation ${}_{\{W\}}R_{\{B\}}^{\text{vic}}(t_t)$ is selected as the most recent VICON sample prior to t_t.

f) $\text{SO}(3)$ Rotation-Angle Error Metric: Orientation error is evaluated using a metric defined directly on $\text{SO}(3)$. The relative rotation between the IMU estimate and VICON ground truth is

$$R_{\text{err}}(t_t) = \Big({}_{\{W\}}R_{\{B\}}^{\text{vic}}(t_t)\Big)^\top {}_{\{W\}}R_{\{B\}}^{\text{imu}}(t_t).$$

The magnitude of this error is quantified by the rotation angle

$$\theta_t = \cos^{-1}\left(\frac{\text{tr}(R_{\text{err}}(t_t)) - 1}{2}\right), \qquad \theta_t \in [0, \pi].$$

This metric avoids Euler-angle singularities and provides a physically meaningful measure of orientation discrepancy [3]. The overall calibration quality is summarized using the RMS rotation-angle error across the trajectory.

2) Orientation Tracking with Calibrated IMU: Now that the IMU biases have been accounted for, the orientation of the robot can be properly estimated.

Since the motion considered in this project is purely rotational, the system state at time k is represented by a unit quaternion

$$\mathbf{q}_t \in \mathbb{H}^*,$$

which parameterizes the rotation ${}_{\{W\}}R_{\{B\}}(t_t) \in \text{SO}(3)$.

a) Motion Model: The IMU provides measurements of angular velocity expressed in the body frame. Given the calibrated gyroscope measurement $\boldsymbol{\omega}_t \in \mathbb{R}^3$ (rad/s) over a time interval $\Delta t_t = t_{t+1} - t_t$, the rigid-body rotational kinematics evolve in the following form:

$$\mathbf{q}_{t+1} = f(\mathbf{q}_t, \tau_t \boldsymbol{\omega}_t) = \mathbf{q}_t \circ \exp\big([0,\ \tfrac{1}{2}\boldsymbol{\omega}_t \Delta t_t]\big),$$

where $\circ$ denotes quaternion multiplication and $\exp(\cdot)$ is the quaternion exponential map.

3) Observation Model: In addition to angular velocity, the IMU provides measurements of linear acceleration. Since there is no translational motion, the accelerometer should only measure gravity expressed in the world frame. Let

$$\mathbf{g}_W = [0,\, 0,\, 1]^\top$$

denote the gravity vector expressed in the world frame in gravity units. Accordingly, the measured acceleration $\mathbf{a}_t$ in the IMU frame should agree with the gravity direction transformed into the body frame via the orientation $\mathbf{q}_t$, leading to the observation model

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

where $\mathbf{q}_t$ is obtained from the current quaternion estimate. This observation model constrains the orientation so that the estimated gravity direction aligns with the measured accelerometer data.

4) Optimization Problem: Orientation estimation is formulated as a trajectory optimization problem over the sequence of unit quaternions $\{\mathbf{q}_t\}_{t=1}^T$. The objective is to find an orientation trajectory that is consistent with both the motion and observation model.

This is achieved by minimizing a cost function of the form

$$\begin{aligned} c(\mathbf{q}_{1:T}) := \frac{1}{2} \sum_{t=0}^{T-1} \left\| 2 \log\Big(\mathbf{q}_{t+1}^{-1} \circ f(\mathbf{q}_t, \tau_t \boldsymbol{\omega}_t) \Big) \right\|_2^2 \\ + \frac{1}{2} \sum_{t=1}^{T} \| [0, \mathbf{a}_t] - h(\mathbf{q}_t) \|_2^2 . \end{aligned} \tag{3}$$

subject to the unit-norm constraints

$$\|\mathbf{q}_t\| = 1, \quad \forall\, t = 1, ..., T.$$

The first term penalizes deviations from the gyroscope motion model, while the second term penalizes discrepancies between the predicted and measured gravity directions.

5) Solution via Projected Gradient Descent: Let the objective be the constrained optimization problem

$$\min_{\mathbf{q}_{1:T}} c(\mathbf{q}_{1:T}) \quad \text{s.t.} \quad \|\mathbf{q}_t\|_2 = 1,\ t = 1, \ldots, T,$$

where $c(\mathbf{q}_{1:T})$ is defined in equation (3) Projected gradient descent (PGD) solves this problem by alternating an unconstrained gradient step with a projection onto the set of unit quaternions.

a) Gradient step.: At iteration k, denote the current quaternion trajectory by

$$\mathbf{q}_{1:T}^{(k)} := \{\mathbf{q}_1^{(k)}, \ldots, \mathbf{q}_T^{(k)}\}.$$

Compute the gradient of the cost with respect to the entire trajectory using the *PyTorch* **AUTOGRAD** function

$$\nabla c\Big(\mathbf{q}_{1:T}^{(k)}\Big) = \left[\left(\frac{\partial c}{\partial \mathbf{q}_1} \right)^\top \cdots \left(\frac{\partial c}{\partial \mathbf{q}_T} \right)^\top \right]^\top,$$

where each $\frac{\partial c}{\partial \mathbf{q}_t} \in \mathbb{R}^4$. The unconstrained gradient descent update is

$$\tilde{\mathbf{q}}_{1:T}^{(k+1)} = \mathbf{q}_{1:T}^{(k)} - \alpha_k \, \nabla c\Big(\mathbf{q}_{1:T}^{(k)}\Big),$$

equivalently, component-wise for $t = 1, \ldots, T$,

$$\tilde{\mathbf{q}}_t^{(k+1)} = \mathbf{q}_t^{(k)} - \alpha_k \, \frac{\partial c}{\partial \mathbf{q}_t}\Big(\mathbf{q}_{1:T}^{(k)}\Big),$$

where $\alpha_k > 0$ is the chosen learning rate.

b) Projection step.: Because the gradient step may violate the unit-norm constraint, each updated quaternion is projected back onto the unit-quaternion manifold $\mathbb{H}^*$ via normalization:

$$\mathbf{q}_t^{(k+1)} = \Pi_{\mathbb{H}^*}\Big(\tilde{\mathbf{q}}_t^{(k+1)}\Big) := \frac{\tilde{\mathbf{q}}_t^{(k+1)}}{\Big\|\tilde{\mathbf{q}}_t^{(k+1)}\Big\|_2}, \qquad t = 1, \ldots, T.$$

Thus, the PGD iteration is summarized by

$$\mathbf{q}_{1:T}^{(k+1)} = \Pi_{\mathbb{H}^*}\Big(\mathbf{q}_{1:T}^{(k)} - \alpha_k \, \nabla c\Big(\mathbf{q}_{1:T}^{(k)}\Big)\Big),$$

where $\Pi_{\mathbb{H}^*}$ is applied element-wise across time indices.

c) Stopping criterion.: The iterations are run until either a fixed maximum number of iterations is reached or the decrease in objective value satisfies a convergence criterion, e.g.,

$$\Big| c\Big(\mathbf{q}_{1:T}^{(k+1)}\Big) - c\Big(\mathbf{q}_{1:T}^{(k)}\Big) \Big| < \varepsilon.$$

In this project, I always used a maximum number of iterations of 1000.

d) Output.: The final iterate $\mathbf{q}_{1:T}^{(K)}$ yields a feasible orientation trajectory with $\|\mathbf{q}_t^{(K)}\|_2 = 1$ for all t, which is then evaluated against VICON ground truth using the aforementioned $\mathrm{SO}(3)$ rotation-angle error metric and visualized via roll–pitch–yaw angle trajectories.

B. Panorama Construction

Given time-stamped camera images $\{(I_k, t_k^{\mathrm{cam}})\}_{k=1}^K$ and an estimated orientation trajectory from the IMU found in III-A, the panorama is constructed by mapping each image pixel to a global viewing direction and then rasterizing these directions into a 2D panorama.

a) Time alignment via closest-in-the-past index: The IMU-derived rotations are available at IMU timestamps $\{t_i^{\mathrm{imu}}\}_{i=1}^N$, while images are captured at camera timestamps $\{t_k^{\mathrm{cam}}\}_{k=1}^K$. For each image time t_k^{cam}, we associate the most recent available IMU orientation using a closest-in-the-past rule,

$$i_k = \max\{\, i \;:\; t_i^{\mathrm{imu}} \le t_k^{\mathrm{cam}} \,\}.$$

In code, this is implemented using a binary search (`searchsorted`) followed by clamping to valid indices. This choice avoids extrapolation of the orientation trajectory and matches the assignment requirement of using the closest previous state estimate at each timestamp.

b) Camera intrinsic approximation and bearing precomputation: Since no explicit camera calibration parameters are provided with the dataset, I approximate the intrinsic matrix using standard pinhole assumptions [2]. Let (H, W) denote the image height and width. We assume the principal point lies at the image center,

$$c_x = \frac{W-1}{2}, \qquad c_y = \frac{H-1}{2},$$

and we choose a focal length proportional to the image size,

$$f = 0.4 \max(W, H).$$

This choice yields a field of view consistent with a wide-angle camera and was found empirically to produce stable and visually coherent panorama reconstructions.

The resulting intrinsic matrix is

$$K = \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix}.$$

Using K, each pixel (u, v) is mapped to a unit bearing direction in the optical frame $\{O\}$:

$$\mathbf{x}(u, v) = \frac{K^{-1}[u \;\; v \;\; 1]^\top}{\left\| K^{-1}[u \;\; v \;\; 1]^\top \right\|} \in \mathbb{S}^2.$$

Since the camera resolution is fixed, all bearings $\mathbf{x}(u, v)$ are precomputed once and reused for all frames to reduce computation.

c) Fixed camera–IMU extrinsic rotation R_{BO}: The orientation estimator produces rotations for the IMU/body frame $\{B\}$, but rays are expressed in the camera optical frame $\{O\}$. We therefore apply a fixed extrinsic rotation ${}_B R_O \in \mathrm{SO}(3)$ that relates the optical axes to the body axes. Following the dataset description (camera axis alignment relative to the IMU) and the standard optical-frame convention (the optical z axis points forward), I define $R_B R_O$ as a constant axis-permutation mapping between frames and use its transpose when needed to ensure the correct direction of transformation. In the implementation, the world-from-optical rotation is formed as

$${}_W R_O(t_k) \;=\; {}_W R_B(t_{i_k})\, {}_B R_O,$$

where ${}_W R_B(t_{i_k})$ is the IMU-derived rotation selected at the closest-in-the-past timestamp.

d) World-frame viewing directions and cylindrical projection: For each frame k and each pixel (u, v), we rotate the precomputed bearing into the world frame:

$$\mathbf{y}_k(u, v) = {}_W R_O(t_k)\, \mathbf{x}(u, v) \in \mathbb{S}^2, \qquad \mathbf{y}_k = (X, Y, Z).$$

We parameterize these directions using a cylindrical panorama mapping. The azimuth is computed as

$$\theta = \mathrm{atan2}(X, Z),$$

and is wrapped to $[0, 2\pi)$ to avoid negative angles. For the vertical coordinate, we use the world vertical component directly,

$$v = Y,$$

which is numerically stable and sufficient for direction-based panorama construction.

e) (5) Temporal azimuth continuity using a reference ray: To reduce discontinuities introduced by wrapping and to maintain consistency across frames, we compute a reference azimuth per image using the center pixel ray $\mathbf{x}_0 = \mathbf{x}(\lfloor W/2 \rfloor, \lfloor H/2 \rfloor)$. For each frame k we compute

$$\theta_k^{\mathrm{ref}} = \mathrm{atan2}\big(({}_W R_O(t_k)\mathbf{x}_0)_x,\; ({}_W R_O(t_k)\mathbf{x}_0)_z \big),$$

and unwrap $\{\theta_k^{\mathrm{ref}}\}$ over time. In the implementation, this reference is used to shift the per-pixel azimuth values of each frame to encourage consistent placement within the panorama coordinate system.

f) Panorama bounds, resolution selection, and rasterization: After projecting all frames, the panorama bounds are chosen from the minimum and maximum projected coordinates (θ, v) (with a small margin). The panorama resolution (H_P, W_P) is selected proportional to the angular span using a pixels-per-radian parameter. Each pixel is then mapped to panorama coordinates (U, V) via linear scaling:

$$U = \frac{\theta - \theta_{\min}}{\theta_{\max} - \theta_{\min}}(W_P - 1), \qquad V = \frac{v - v_{\min}}{v_{\max} - v_{\min}}(H_P - 1).$$

Finally, colors are rasterized into the panorama using nearest-neighbor splatting (rounding (U, V) to integer indices) with overwrite assignment.

This pipeline produces a single panoramic image that parameterizes observed RGB data by global viewing direction, consistent with the estimated rotation trajectory and the camera imaging model.

IV. Training Results

A. Calibration

For the IMU calibration, the results of calibration on dataset 1 is shared. Below, there is a table detailing the calibration process and results:

TABLE I
IMU Calibration Summary (Static Window)

Parameter	Value
Static window duration T_{static} (s)	5.0
Number of static samples N_{static}	500
Gyroscope bias (raw counts)	$\begin{bmatrix} 373.568 & 375.356 & 369.680 \end{bmatrix}$
Accelerometer bias (g)	$\begin{bmatrix} -5.4925 & -5.3870 & 5.5071 \end{bmatrix}$
Accelerometer scale (g/count)	1.0752×10^{-2}
Gyroscope scale (rad/s/count)	1.6690×10^{-2}
Mean gyro magnitude (static) (rad/s)	1.6654×10^{-2}
Mean accel magnitude (static) (g)	1.0000

In the above table, note that the second section for the accelerometer and gyroscope include the resting output of the two devices. The conversion to true bias is done below for thoroughness:

a) Gyroscope Conversion:

- Reference voltage: $V_{\text{ref}} = 3.3$ V
- ADC resolution: 10-bit, maximum code $= 1023$
- Gyroscope zero-rate level: $V_{0\omega} = 1.23$ V
- Gyroscope scale factor: $\alpha_\omega = 0.01665493$ rad/s per count

At rest, the expected ADC code corresponding to the zero-rate voltage is given by

$$c_{0\omega} = \frac{V_{0\omega}}{V_{\text{ref}}} \cdot 1023 = \frac{1.23}{3.3} \cdot 1023 \approx 381.30. \tag{1}$$

Therefore to get the true bias in raw ADC counts, let $\bar{c}_\omega$ denote the mean gyroscope measurement over the static window. From the data, this value is

$$\bar{c}_\omega = \begin{bmatrix} 373.60 & 375.36 & 369.68 \end{bmatrix}. \tag{2}$$

The gyroscope bias in raw ADC counts is therefore

$$\mathbf{b}_{\omega,\text{counts}} = \bar{c}_\omega - c_{0\omega} \approx \begin{bmatrix} -7.70 & -6.02 & -11.60 \end{bmatrix}. \tag{3}$$

Using the calibrated scale factor, the bias expressed in radians per second is

$$\mathbf{b}_{\omega,\text{rad/s}} = \mathbf{b}_{\omega,\text{counts}} \cdot \alpha_\omega \approx \begin{bmatrix} -0.130 & -0.102 & -0.196 \end{bmatrix} \text{ rad/s}. \tag{4}$$

Equivalently, this corresponds to approximately

$$\begin{bmatrix} -7.5 & -5.8 & -11.2 \end{bmatrix} \text{ deg/s}. \tag{5}$$

These values represent the true gyroscope bias after accounting for the datasheet-specified zero-rate output and seem relatively reasonable.

b) Accelerometer: Unlike the gyroscope, the accelerometer measures gravity even at rest. As a result, its static mean cannot be interpreted as a pure sensor bias without knowledge of the body orientation although this is what is displayed in the table. The apparent accelerometer "bias" therefore absorbs both orientation and gravity expressed in the body frame. Therefore, it is the magnitude of this vector at rest that is important and we can see in the last section of the table that this is 1g.

B. Calibration Validation via Gyroscope-Only Orientation Tracking

To validate the IMU calibration, we compare a gyroscope-only orientation estimate against ground-truth motion capture data from the VICON system. The IMU angular velocity measurements are first calibrated then integrated forward in time using the quaternion kinematics described before.

1) SO(3) Rotation-Angle Error Metric: Just to restate, at each time step t, the relative rotation between the IMU estimate $R_{\text{IMU}}(t)$ and the VICON ground truth $R_{\text{VICON}}(t)$ is computed as

$$R_{\text{err}}(t) = R_{\text{VICON}}(t)^\top R_{\text{IMU}}(t).$$

The magnitude of the orientation error is then defined as the corresponding rotation angle

$$\theta(t) = \cos^{-1}\left(\frac{\text{tr}(R_{\text{err}}(t)) - 1}{2}\right), \quad \theta(t) \in [0, \pi].$$

This metric provides a coordinate-free and physically meaningful measure of orientation error.

For Dataset 1, the resulting RMS rotation-angle error over the full trajectory is

$$\theta_{\text{RMS}} = 0.333 \text{ rad} \approx 19.08^\circ.$$

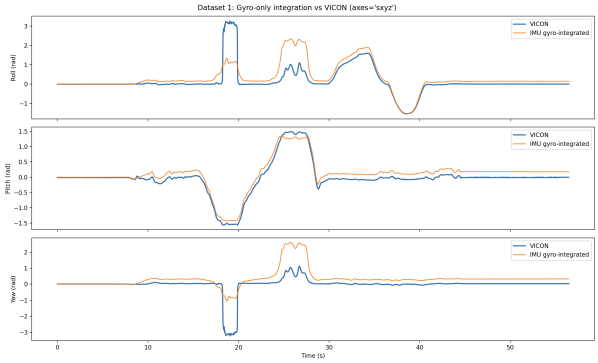

Fig. 1. SO(3) Euler angle comparison between gyroscope-only IMU orientation and VICON ground truth.

figure 1 shows a fairly faithful approximation of the ground truth data. The observed rotation-angle error demonstrates that the calibrated gyroscope measurements are physically reasonable but insufficient for long-term orientation tracking when used alone. This discrepancy motivates the inclusion of observation-based corrections in the following section. These results demonstrate that the calibration was successful.

C. Orientation Tracking

Below, the optimized trajectories for dataset 1 and 2 are displayed along with the VICON ground truth data in 2 and 3

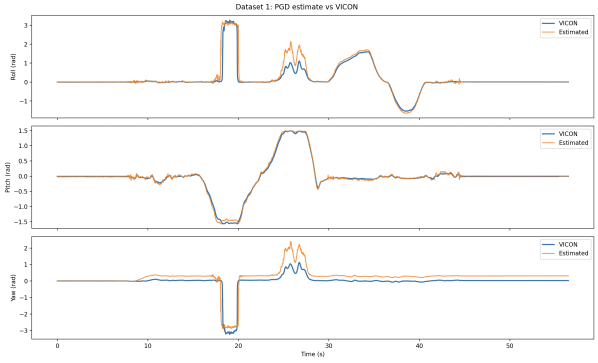

Fig. 2. Optimized angle trajectories for dataset 1.

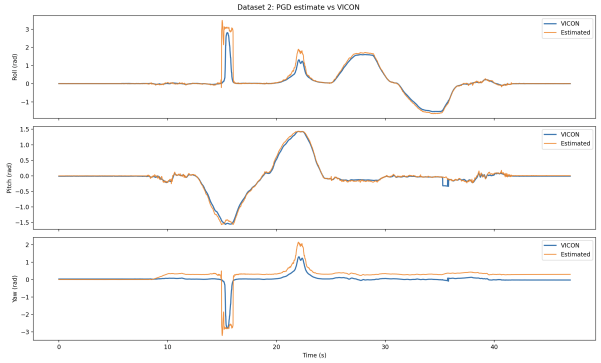

Fig. 3. Optimized angle trajectories for dataset 2. Note: IMU calibration was also done on dataset 2 for this result.

Below, a table shares the θ_{RMS} before and after optimization for each dataset; the graphs are omitted here for brevity, but included in appendix A.

TABLE II
SO(3) RMS ROTATION ERROR BEFORE AND AFTER OPTIMIZATION.

Dataset	θ_{RMS} (Before PGD) (deg)	θ_{RMS} (After PGD)
1	19.08	14.68
2	24.95	15.26
3	16.52	16.94
4	46.77	43.64
5	35.78	26.98
6	–	–
7	–	–
8	19.16	15.43
9	42.57	34.20

I was unable to get values for dataset 6 and 7 because the yaw keeps measuring after the roll and pitch have stopped in this dataset. I was unable to remedy this issue in time. Graphs were still produced in A, but values came out as **NaN** here.

As can be observed, all datasets except dataset 3 observe a decrease in the overall error using this method.

Qualitatively, the trajectory for dataset 1 is much more faithful after the optimization. The other gyro-only orientation tracking graphs are provided in A alongside their respective optimized graphs so that the reader can judge the results. Note that all optimizations were done over 1000 iterations with a learning rate of 1.0^{-3}.

D. Panorama Construction

The panoramas for Data Sets 1 and 2 are shown in 4 and 5. All optimizations were performed using the aforementioned iteration count and the learning rate. In addition, the pixels per radian variable was chosen to be 300 as this worked well empirically.

It would appear that these two datasets contained pretty similar images and trajectories. Slight differences are seen around the the middle of the screen when the camera rotates 90 degrees clockwise. I suspeect that in dataset 2 the rotation was much slower because there are many more frames at this part of the image. It seems that the panorama works pretty faithfully.

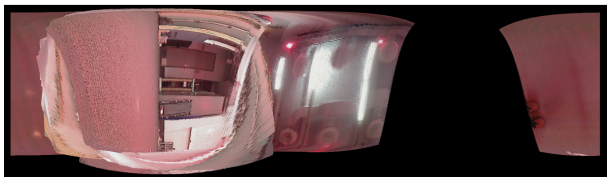

Fig. 4. Panorama on a cylindrical projection for dataset 1.

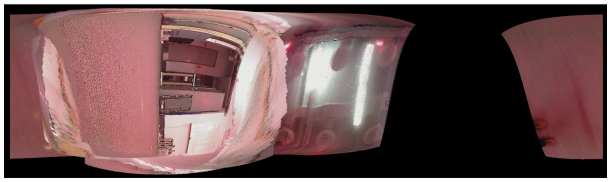

Fig. 5. Panorama on a cylindrical projection for dataset 2.

V. TEST RESULTS

A. IMU Calibration

The calibration results shared here are focused on sharing the estimated trajectories using gyro-only orientation tracking. The biases are similar to IV-A so that treatment is not replicated here. Below, the trajectories are shared for the reader's reference in 6 and 7.

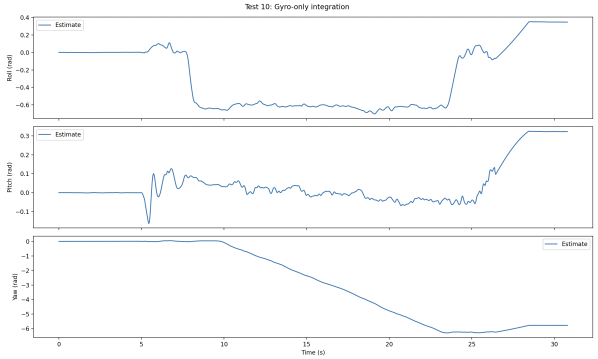

Fig. 6. Gyro-only integration for dataset 10

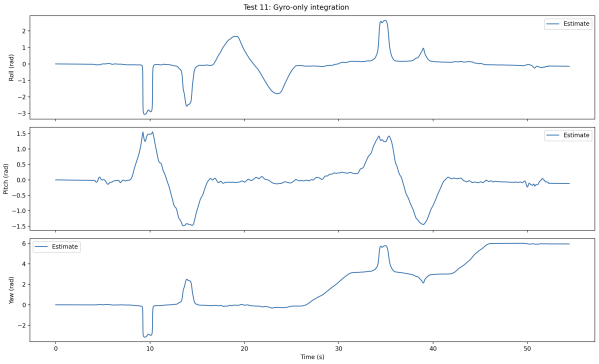

Fig. 7. Gyro-only integration for dataset 11

These graphs are depicting the pre-optimized trajectory for the test datasets. They seem in line with what has been produced in the training results section, so we will continue under the assumption that these graphs are a fairly good approximation of the ground truth.

B. Orientation Tracking

Now, the discussed PGD algorithm is used on the test dataset to produce the optimized trajectories for datasets 10 and 11. The results are shown in 8 and 9

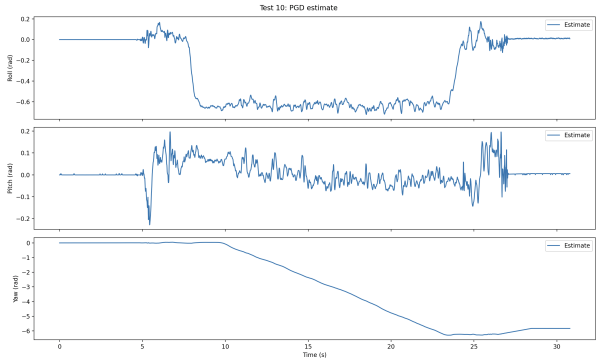

Fig. 8. Optimized trajectories using PGD for dataset 10.

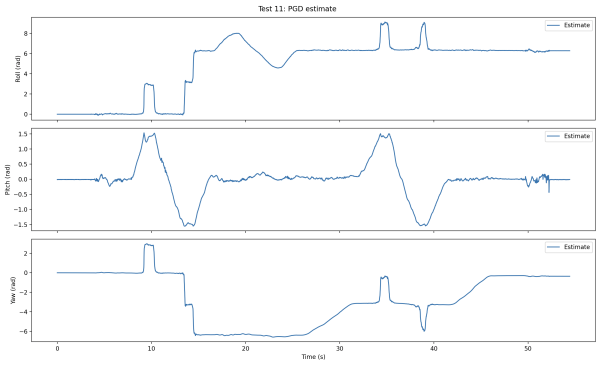

Fig. 9. Optimized trajectories using PGD for dataset 11.

The optimized trajectory for dataset 10 seems to follow a pretty similar path to the gyro-only integration shown in 6. This elucidates that the state estimation did a good job here. However, the same cannot be said when comparing 9 and 7. This is pretty odd, as it is the only dataset in which I found that to be the case. Note that these results used the same maximum iterations of 1000 and the same learning rate of 1^{-3}.

C. Panorama

Now, the panoramas for the test sets are included below in 10 and 11. The same parameters were used as in IV-D.

As expected, the panorama for dataset 10 looks better than for dataset 11. This is because the optimized trajectory for dataset 11 did not work as well. As an experiment, the panorama for dataset 11 is included using the gyro-only integration in 12.

Fig. 10. Panorama for dataset 10.

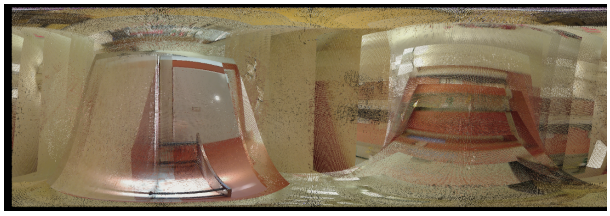

Fig. 11. Panorama for dataset 11.

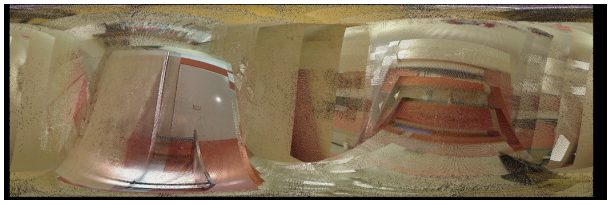

Fig. 12. Panorama for dataset 11 using gyro-only integration.

However, when comparing 11 and 12, they look virtually identical. This elucidates that the optimized trajectory was fairly faithful.

VI. Conclusion

In conclusion, in this project, I provided a robust mathematical formulation of the orientation tracking and panorama construction problems. In addition, results were produced on a training data set as well as a test data set. The results elucidate that a PGD algorithm performs fairly well in approximating a purely rotational trajectory. [1]

References

[1] R. Szeliski, "Image alignment and stitching: A tutorial," *Foundations and Trends in Computer Graphics and Vision*, 2006.

[2] R. Szeliski, *Computer Vision: Algorithms and Applications*. Springer, 2 ed., 2022.

[3] T. D. Barfoot, *State Estimation for Robotics*. Cambridge University Press, 2017.

Appendix

Below, the optimized orientation graph of each omitted dataset is displayed followed by its unoptimized graph for the reader's reference.

[1] ChatGPT (OpenAI) was used as a clarification tool during this project as well as for help with LaTeX formatting issues.

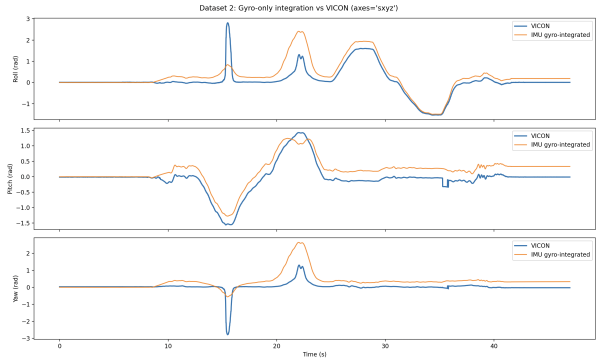

Fig. 13. Unoptimized angle trajectories for dataset 2.

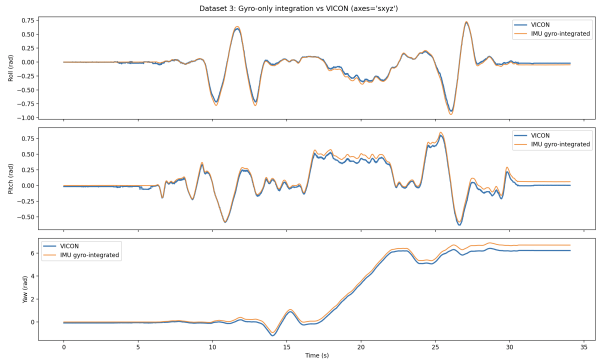

Fig. 14. Unoptimized angle trajectories for dataset 3.

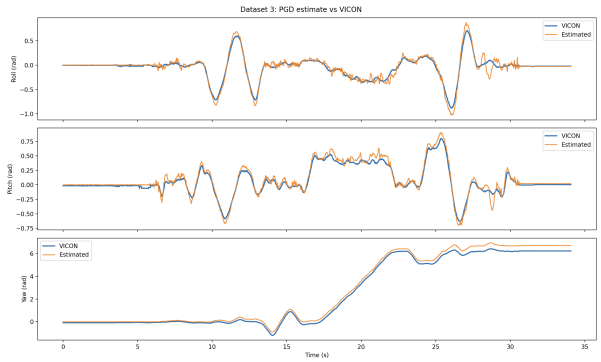

Fig. 15. Optimized angle trajectories for dataset 3.

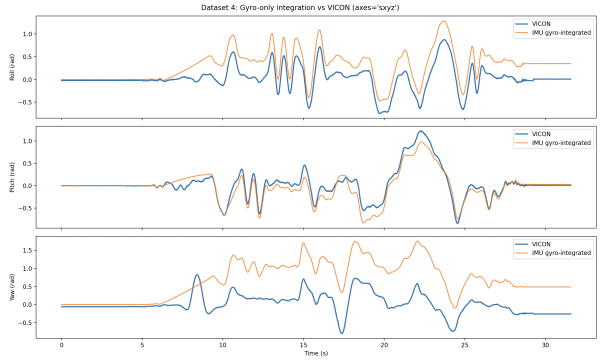

Fig. 16. Unoptimized angle trajectories for dataset 4.

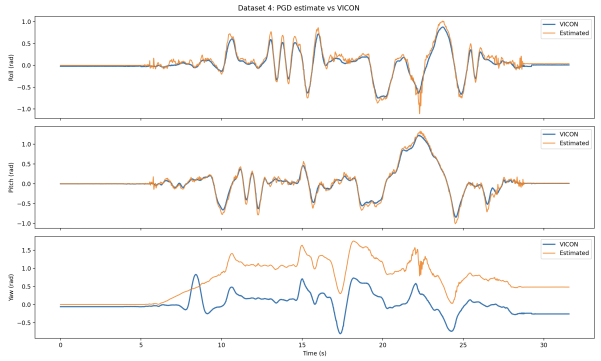

Fig. 17. Optimized angle trajectories for dataset 4.

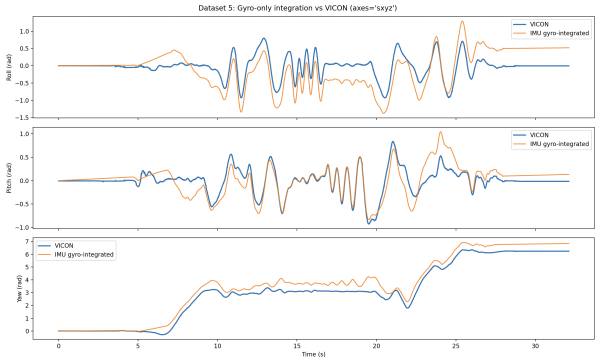

Fig. 18. Unoptimized angle trajectories for dataset 5.

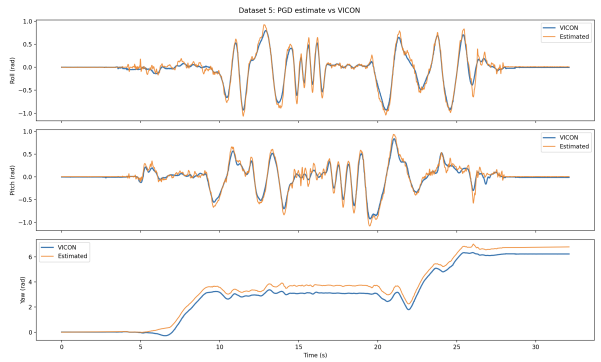

Fig. 19. Optimized angle trajectories for dataset 5.

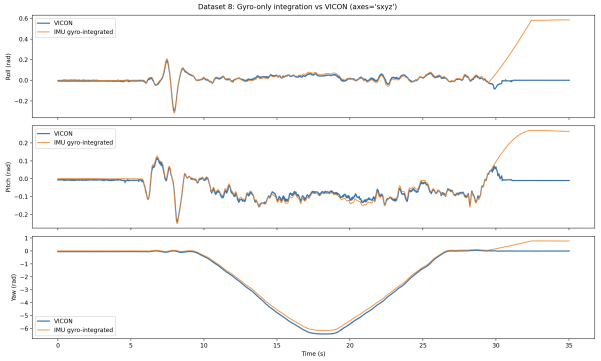

Fig. 20. Unoptimized angle trajectories for dataset 8.

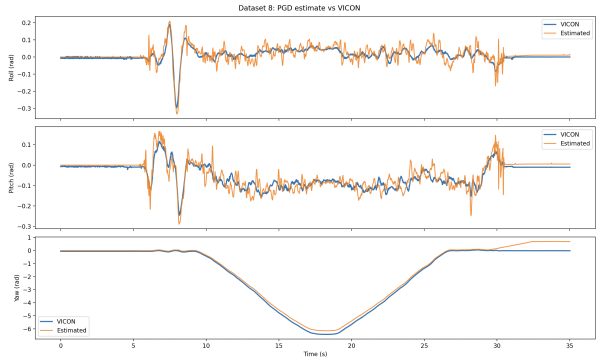

Fig. 21. Optimized angle trajectories for dataset 8.

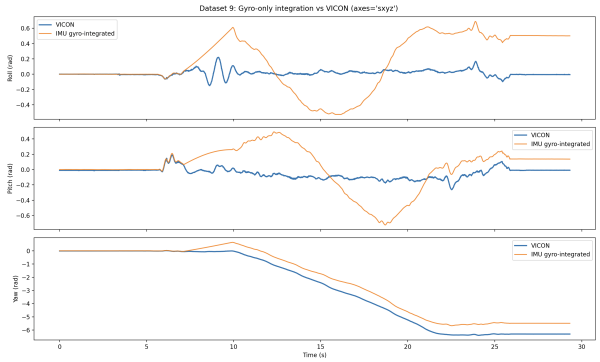

Fig. 22. Unoptimized angle trajectories for dataset 9.

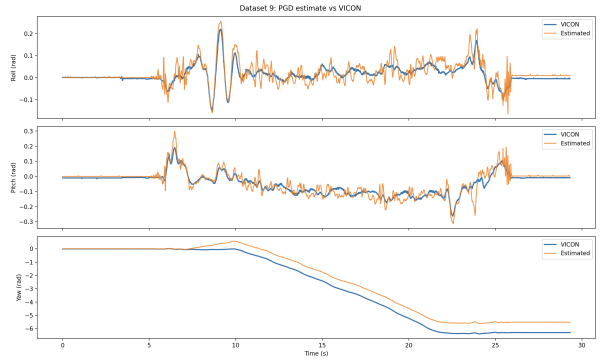

Fig. 23. Optimized angle trajectories for dataset 9.