Project1 Report: Orientation Tracking

Cheng Han Yu
University of California, San Diego
La Jolla, California
chy084@ucsd.edu

I. Introduction

Panoramas can be created by rotating a camera and stitching images captured over time. However, in real-world settings, accurately estimating the camera' s orientation is challenging because inertial measurement unit (IMU) data are inevitably corrupted by sensor noise and biases, which can lead to drift in orientation estimates over time.

The objective of this project is to recover accurate roll, pitch, and yaw angles from IMU measurements by fusing complementary sensor information. I formulate the orientation estimation problem as a gradient-descent optimization and evaluate its performance by comparing the estimated orientations against the provided Vicon ground-truth data. Using the estimated camera orientations, I then construct a panorama by warping and stitching RGB images captured over time. [1]

II. Problem Formulation

The measurement setup consists of a camera and an IMU (Fig. 1), and a Vicon motion-capture system is used to provide ground-truth orientations for evaluation. My goal is to construct an accurate panorama from the RGB images using the estimated camera orientations. To accomplish this, the project consists of two parts: (1) orientation tracking and (2) panorama reconstruction. In orientation tracking, I estimate the orientation trajectory (roll, pitch, and yaw) and compare it against the Vicon ground truth. I then use the estimated orientations to warp and stitch the RGB images over time into a spherical panorama, which is subsequently projected onto a cylindrical image to obtain the final panorama.

A. Orientation Estimation

For precise estimation for camera orientatioin, we need to first construct motion model and observation model. Motion model describes rotational kinematics and observation model describes mapping from system state to expected sensor measurement. From the IMU, we obtain accelerometer measurements $[A_x, A_y, A_z]$ and gyroscope measurements $[\omega_x, \omega_y, \omega_z]$.

1) Motion model (gyroscope integration): I use the gyroscope readings to define a motion model via quaternion integration, and I use the accelerometer readings (assuming negligible linear acceleration) to define an observation model based on the gravity direction. I estimate

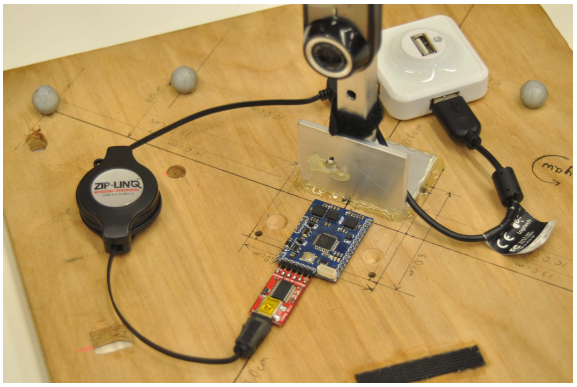


Fig. 1. IMU–camera sensor setup.

orientation by integrating the gyroscope measurements in quaternion form, where q_t denotes the attitude at time step t. I represent the quaternion as $q_t = [w_t, x_t, y_t, z_t]^\top$.

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

This yields a sequence of quaternions that captures the motion estimated from the angular velocity measurements.

2) Observation model (accelerometer/gravity): To map from system state to expected sensor measurement, we introduce accelerometer. Assuming the platform undergoes pure rotation (i.e., negligible linear acceleration), the accelerometer primarily measures gravity. In the world frame, the specific force is expected to be $[0, 0, -g]$ $(\mathrm{m/s^2})$ in world frame.

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

Here, a_t is the 3D accelerometer measurement at time t expressed in the body frame, and $[0, a_t]$ denotes the corresponding pure-imaginary quaternion. The quaternion q_t represents the attitude (world-to-body), and $\circ$ denotes quaternion multiplication. The quaternion $[0, 0, 0, 1]$ corresponds to the gravity direction in the world frame (unit vector). Therefore, $h(q_t)$ is the predicted accelerometer measurement obtained by rotating gravity into the body frame.

3) Optimization: Given the motion model and observation model, I estimate the entire trajectory $q_{1:T}$ by

minimizing a cost function using gradient descent.

$$\mathcal{C}(q_{1:T}) := \frac{1}{2}\sum_{t=0}^{T-1}\left\|2\,\log\Big(q_{t+1}^{-1}\circ f\big(q_t,\,\tau_t\omega_t\big)\Big)\right\|_2^2 + \frac{1}{2}\sum_{t=1}^{T}\|[0,\,a_t]-h(q_t)\|_2^2\,. \quad (3)$$

Finally, I compare the estimated orientation trajectory against the Vicon ground truth to evaluate the accuracy of the proposed method.

B. Panorama Reconstruction

In panorama reconstruction, firstly, I use the estimated orientations to warp and stitch the RGB images over time. Given the estimated IMU orientation trajectory $q_{1:T}$, I compute the IMU pose in the world frame s and transform it to the camera frame c using the (fixed) IMU–camera extrinsics. Let $T_{sb,t} \in SE(3)$ denote the homogeneous transform from IMU frame b to world frame s at time t (with rotation $R_{sb,t}$ obtained from q_t), and let $T_{bc} \in SE(3)$ denote the constant transform from camera frame c to IMU frame b. Then the camera pose in the world frame is

$$\begin{aligned} T_{sc,t} &= T_{sb,t}\,T_{bc},\\ T_{bc} &= \begin{bmatrix} R_{bc} & t_{bc}\\ 0 & 1\end{bmatrix}, \qquad t_{bc} = [0,\,0,\,0.1]^\top \text{ m},\\ R_{bc} &= \begin{bmatrix} 0 & 0 & 1\\ -1 & 0 & 0\\ 0 & -1 & 0\end{bmatrix}. \end{aligned} \quad (4)$$

thus, we get T_{bc}:

$$T_{bc} = \begin{bmatrix} 0 & 0 & 1 & 0\\ -1 & 0 & 0 & 0\\ 0 & -1 & 0 & 0.1\\ 0 & 0 & 0 & 1\end{bmatrix}. \quad (5)$$

For each pixel in the RGB image at time t, I back-project it to a unit ray on the camera sphere, transform it to the world frame using $T_{sc,t} = T_{sb,t}T_{bc}$, and accumulate the result in a spherical panorama. Finally, I apply a cylindrical projection to obtain the final panorama.

III. Technical Approach

A. Preprocessing

For IMU calibration, I first convert raw ADC readings to physical units using

$$\begin{aligned} \text{value} &= (\text{raw} - \text{bias}) \times \text{scale},\\ \text{scale} &= \frac{V_{\text{ref}}}{1023 \cdot \text{sensitivity}}. \end{aligned} \quad (6)$$

The ADC is 10-bit, so the maximum count is 1023. During data collection, the reference voltage is $V_{\text{ref}} = 3.3\,\text{V} = 3300\,\text{mV}$. The accelerometer sensitivity is $330\,\text{mV}/g$, and the gyroscope sensitivity is $3.33\,\text{mV}/(\text{deg/s}) = 3.33 \cdot 180/\pi\,\text{mV}/(\text{rad/s})$.

Next, I estimate sensor biases using an initial static segment: I subtract the mean of the first 100 samples from the entire dataset and align the measurements so that for the static portion of the dataset $[A_x, A_y, A_z] = [0, 0, 1]$ [g] and $[\omega_x, \omega_y, \omega_z] = [0, 0, 0]$ [rad/sec]. I convert the Vicon ground-truth rotation matrix $R \in SO(3)$ into ZYX Euler angles (yaw–pitch–roll), i.e., (ψ, θ, ϕ), where ψ is yaw, θ is pitch, and ϕ is roll. Given

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13}\\ r_{21} & r_{22} & r_{23}\\ r_{31} & r_{32} & r_{33}\end{bmatrix}, \quad (7)$$

I compute

$$\begin{aligned} \theta &= \arcsin(-r_{31}),\\ \phi &= \arctan(r_{32},\,r_{33}),\\ \psi &= \arctan(r_{21},\,r_{11}). \end{aligned} \quad (8)$$

When $|r_{31}| \approx 1$ (gimbal lock), the yaw and roll angles become coupled; in this case we keep θ from $\arcsin(-r_{31})$ and compute the remaining angle using a consistent convention.

B. Orientation Estimation

1) Motion model calibration: Using the motion model in (1), we integrate the gyroscope measurements to obtain an initial quaternion trajectory. For each time step, I compute the time increment τ_t from timestamps and form the incremental rotation as a pure-imaginary quaternion $\delta_t = [0,\,\tau_t\omega_t/2]$. We then propagate the orientation by

$$q_{t+1} = q_t \circ \exp(\delta_t), \quad (9)$$

followed by the unit-quaternion projection in (11) to prevent numerical drift. For evaluation, we convert both the integrated quaternion trajectory and compare it to ground truth data and plot the resulting roll, pitch, and yaw curves for comparison.

2) Optimization: For optimization, we minimize the cost in (3) using gradient descent over the quaternion trajectory $q_{1:T}$. At each iteration k, we compute the gradient $\nabla\mathcal{C}(q_{1:T}^{(k)})$ using automatic differentiation (`torch.autograd.grad`) and perform the update with step size $\alpha = 0.01$.

$$q_{1:T}^{(k+1)} = q_{1:T}^{(k)} - \alpha\,\nabla\mathcal{C}(q_{1:T}^{(k)}), \quad (10)$$

In (3), the first term penalizes the relative rotation error between the predicted orientation $f(q_t, \tau_t\omega_t)$ and the next state q_{t+1}. I use the quaternion logarithm to map the relative rotation to a 3D rotation-vector (Lie algebra) representation; the factor $2\log(\cdot)$ converts the unit-quaternion error into an angle-axis vector whose norm equals the rotation angle. The second term enforces consistency with the accelerometer observation by encouraging $h(q_t)$ to match the measured $[0, a_t]$. After each gradient descent update, I project the quaternion back onto the unit-quaternion manifold $\mathbb{H}^*$ by normalization:

$$\Pi_{\mathbb{H}^*}(q) = \frac{q}{\|q\|_2}. \quad (11)$$

Apply the projection to each q_t. We terminate when the change in the objective satisfies $|\mathcal{C}^{(k+1)} - \mathcal{C}^{(k)}| < 10^{-6}$. After convergence, we convert the optimized quaternions to Euler angles and compare them against the Vicon ground truth.

C. Panorama Reconstruction

From orientation tracking, we obtain the quaternion trajectory $q_{1:T}$ (IMU/world frame). Using the extrinsic calibration, we transform the orientation to the camera frame and obtain the camera rotation R_t at each time step.

Given an RGB image of size $H \times W$ (here $H = 240, W = 320$) with horizontal and vertical fields of view fov_h and fov_v (in radians; e.g., $\text{fov}_h = 60^\circ = \pi/3$ and $\text{fov}_v = 45^\circ = \pi/4$), we map each pixel (i, m) with $i \in \{0, \dots, W-1\}$ and $m \in \{0, \dots, H-1\}$ to spherical angles (θ, ϕ), where θ is the horizontal (azimuth) angle and ϕ is the vertical (elevation) angle:

$$\theta(i) = \left(\frac{i}{W-1} - \frac{1}{2}\right)\frac{\pi}{3}, \qquad \phi(m) = \left(\frac{m}{H-1} - \frac{1}{2}\right)\frac{\pi}{4}. \tag{12}$$

We then convert (θ, ϕ) to a unit ray in the camera frame (note that $\phi \in [-\pi/6, \pi/6]$ and $\theta \in [-\pi/8, \pi/8]$ under the above FoVs),

$$\ell_C = \begin{bmatrix} x_C \\ y_C \\ z_C \end{bmatrix} = \begin{bmatrix} \cos\phi \, \sin\theta \\ \sin\phi \\ \cos\phi \, \cos\theta \end{bmatrix}, \tag{13}$$

rotate the Cartesian coordinate into the world frame via $[\ell_W^\top, 1]^\top \propto T_{SC}\,[\ell_C^\top, 1]^\top$, and normalize to obtain the corresponding point on the unit sphere in the world frame. Then, we map spherical coordinates to a cylindrical panorama by computing the longitude $\lambda = -\arctan(y_C/x_C)$ and latitude $\varphi = \arcsin(z_C)$. The panorama is thus unwrapped such that the horizontal axis spans $\lambda \in [-\pi, \pi]$ and the vertical axis spans $\varphi \in [-\pi/2, \pi/2]$. The resulting spherical coordinates are then accumulated and projected to a cylindrical panorama for stitching.

IV. Results

A. Orientation Optimization

Before orientation optimization, the estimated angles do not match the ground truth well. This discrepancy is expected because:

- The measurements are discrete and are integrated over time; thus, small gyroscope noise and bias can accumulate and lead to drift, which introduces errors in the motion model. This effect can be observed in datasets 1 and 2, where brief periods of rapid motion increase the integration error and lead to a larger mismatch with the ground truth.
- Datasets 4, 5, and 9 contain missing or corrupted segments (Fig. **??**); as a result, the measured angular velocities can be inconsistent with the true motion, which reduces estimation accuracy.
- Without an external heading measurement, the yaw angle can drift over time, which also affects the overall alignment with the ground truth.

After optimization, the roll and pitch estimates improve significantly because the second term in the cost function in (3) realigns the gravity direction (the world z-axis), which helps correct gyroscope drift. However, some discrepancies still remain for the following reasons:

- The assumption of "pure rotation" is occasionally violated due to linear accelerations, so the accelerometer no longer measures gravity only. This mismatch introduces errors in the observation model. Observe can be make by the first few
- For datasets 4, 5, and 9, the roll and pitch angle match well after opimization, but because yaw is by the accelerometer term, yaw angle still exist discrepancy.

Possible further improvements:

- Weight the two cost terms differently (motion-model vs. accelerometer) to better balance drift and noise.
- Add a magnetometer and introduce a third cost term to algin the heading (yaw).

B. Panorama Visualizations

The results show a good match with the ground-truth panorama; however, some misalignment and discontinuities remain for the following reasons:

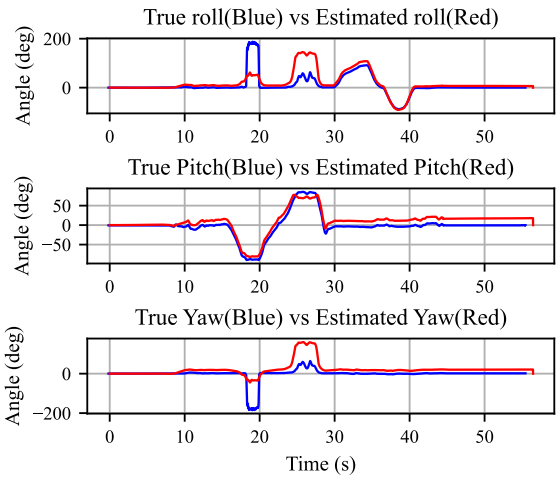
- A yaw bias in the estimated orientation leads to a longitude misalignment in the panorama.
- Residual roll and pitch errors introduce small discontinuities between frames.
- Imperfect time synchronization in IMU and camera data can further degrade alignment with the ground truth.

Possible further improvements:

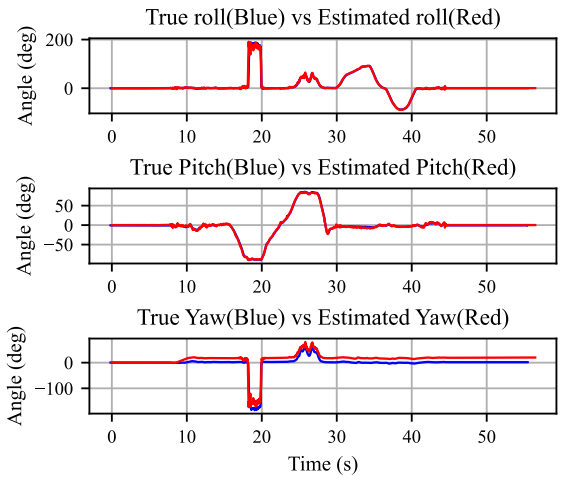
- Include feature tracking to refine image alignment in overlapping regions.
- Enforcing a constraint that aligns features between the start and end frames would distribute the accumulated yaw error across the entire trajectory, eliminating the seam at the panorama boundary.

References

[1] N. Atanasov, "Ece 276a: Sensing & estimation in robotics project 1: Orientation tracking," Course handout / website, 2026, accessed: 2026-02-02.

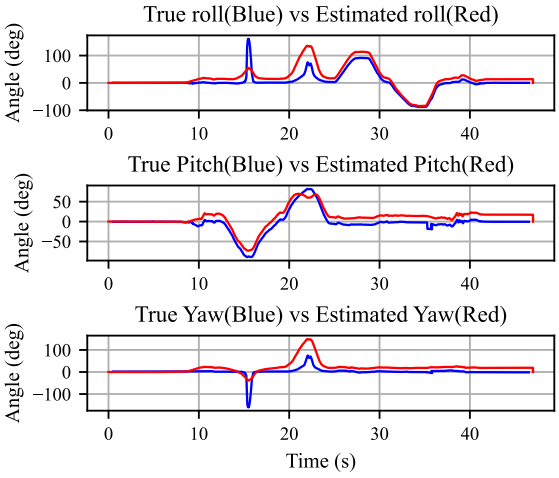


Dataset 1 (before)

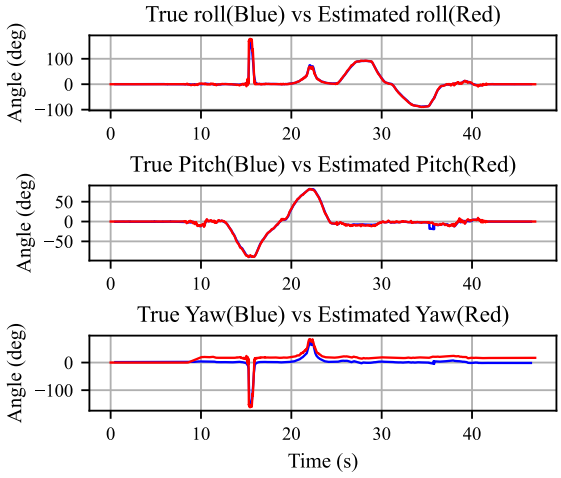


Dataset 1 (after)

Fig. 2. Training set: dataset 1 before/after optimization.

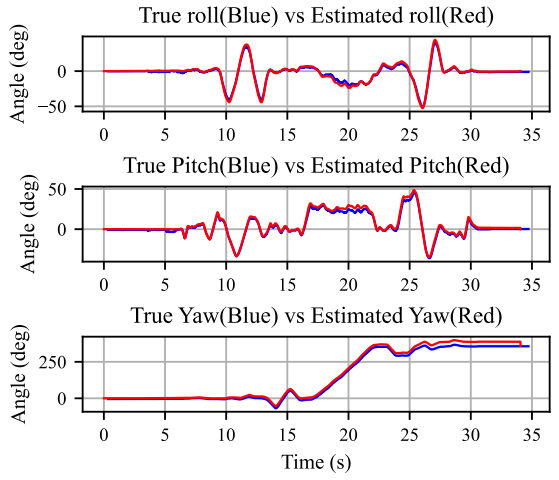


Dataset 2 (before)

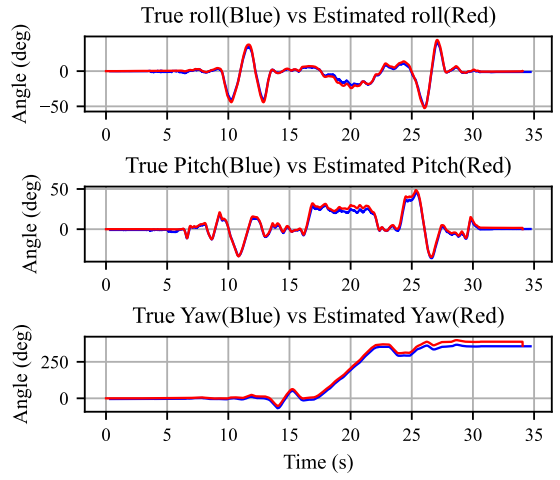


Dataset 2 (after)

Fig. 3. Training set: dataset 2 before/after optimization.

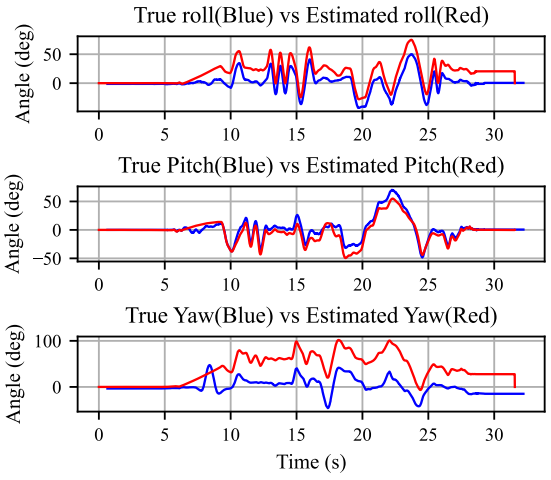


Dataset 3 (before)

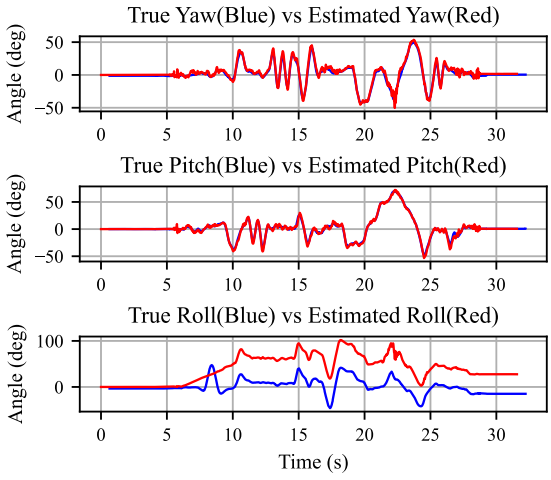


Dataset 3 (after)

Fig. 4. Training set: dataset 3 before/after optimization.

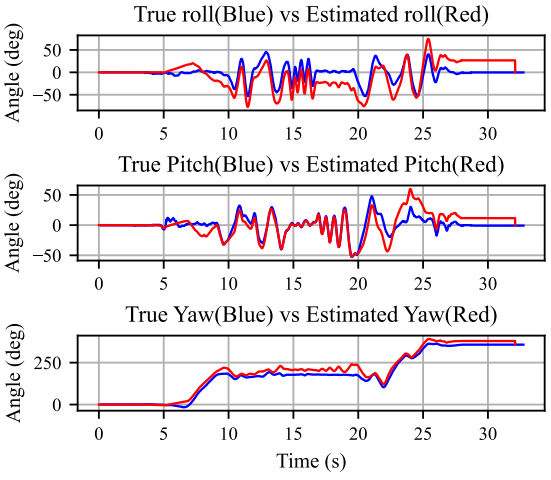


Dataset 4 (before)

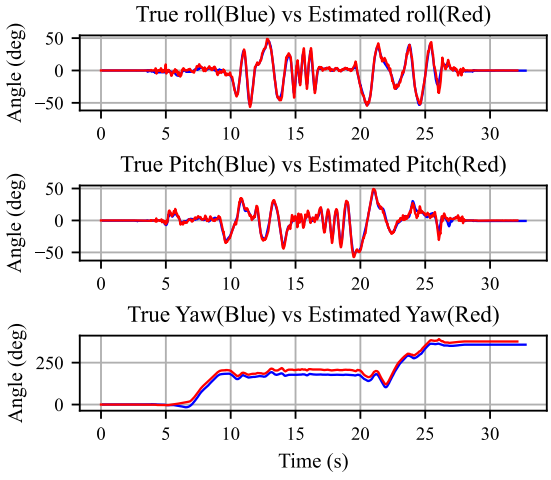


Dataset 4 (after)

Fig. 5. Training set: dataset 4 before/after optimization.

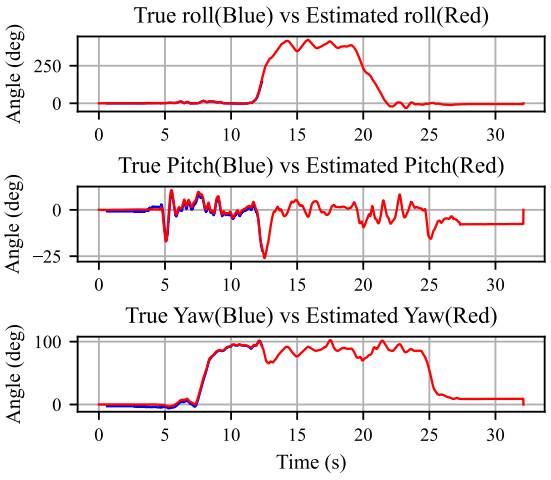


Dataset 5 (before)

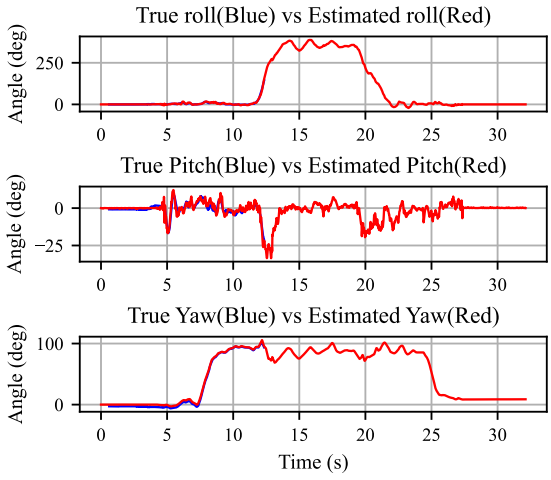


Dataset 5 (after)

Fig. 6. Training set: dataset 5 before/after optimization.

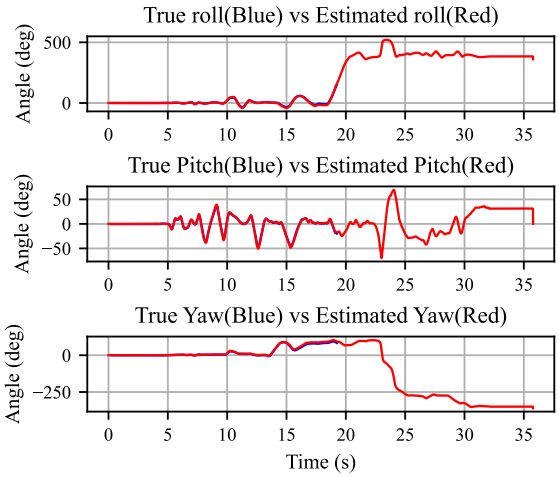


Dataset 6 (before)

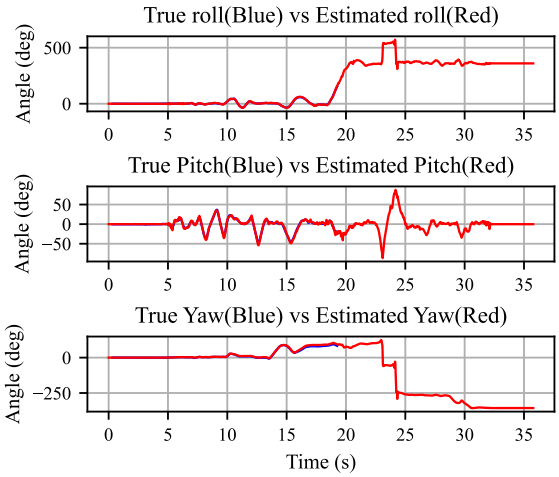


Dataset 6 (after)

Fig. 7. Training set: dataset 6 before/after optimization.

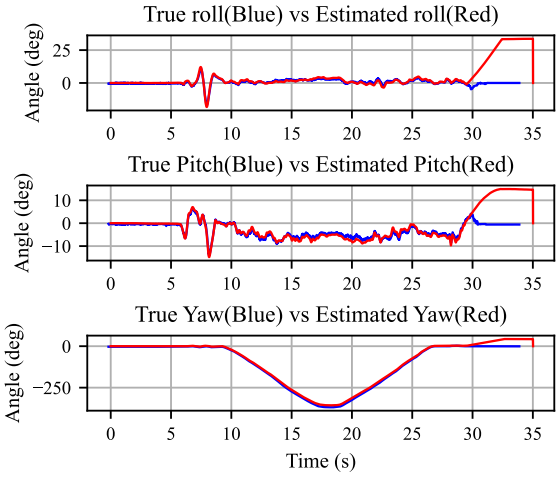


Dataset 7 (before)

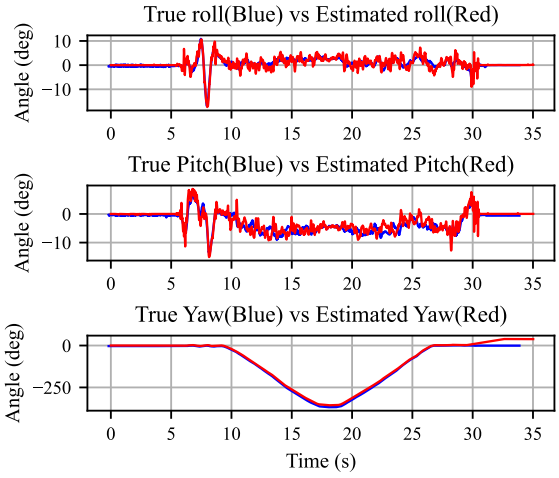


Dataset 7 (after)

Fig. 8. Training set: dataset 7 before/after optimization.

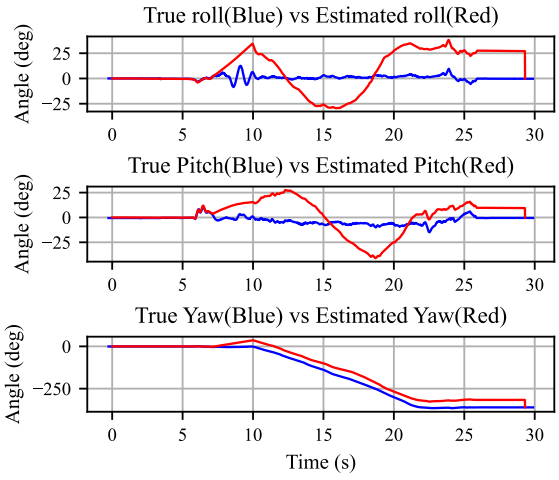


Dataset 8 (before)

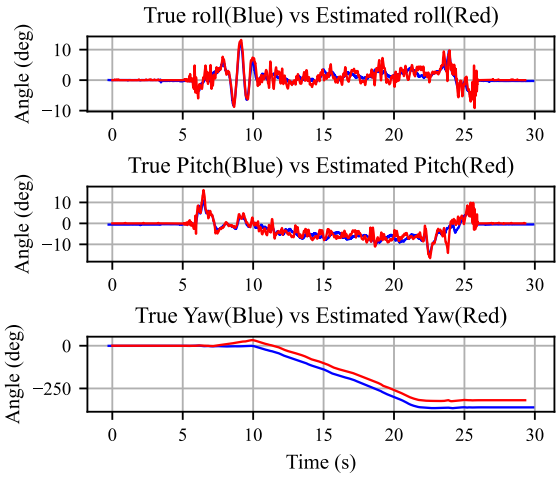


Dataset 8 (after)

Fig. 9. Training set: dataset 8 before/after optimization.

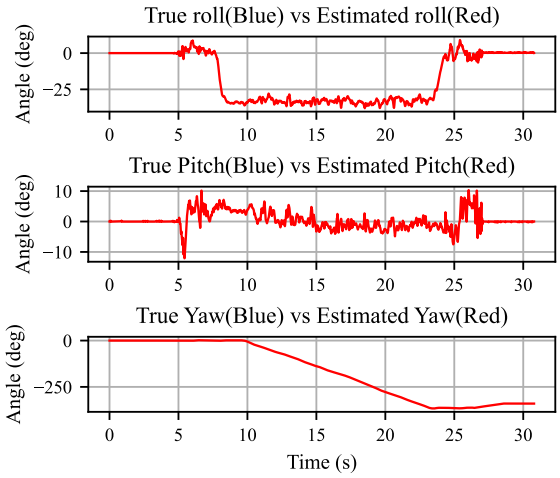


Dataset 9 (before)

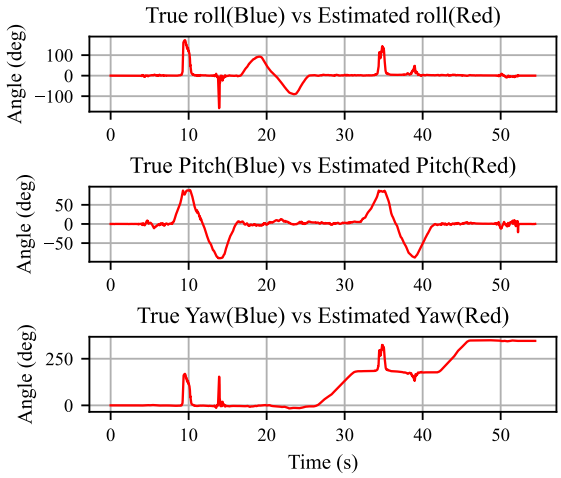


Dataset 9 (after)

Fig. 10. Training set: dataset 9 before/after optimization.

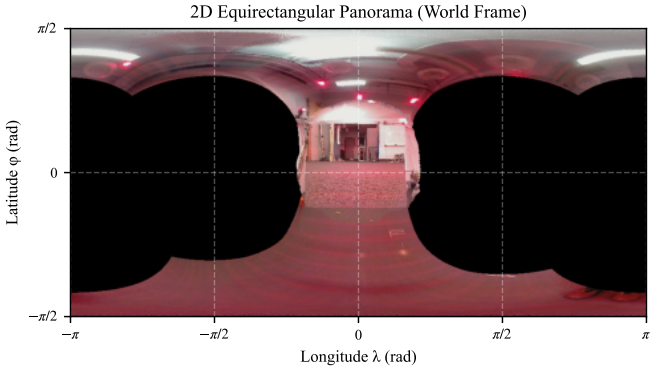


Dataset 10 (after)

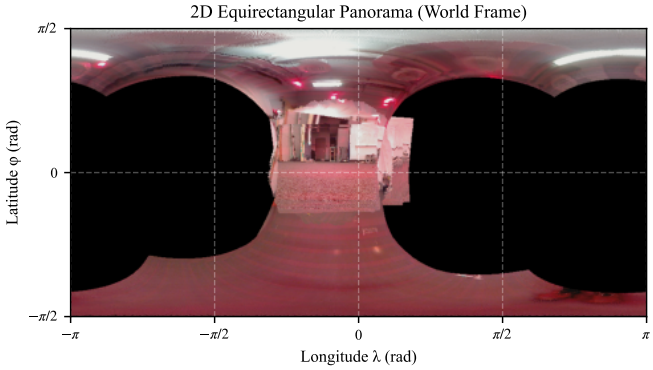


Dataset 11 (after)

Fig. 11. Test set: datasets 10 and 11 (after optimization).

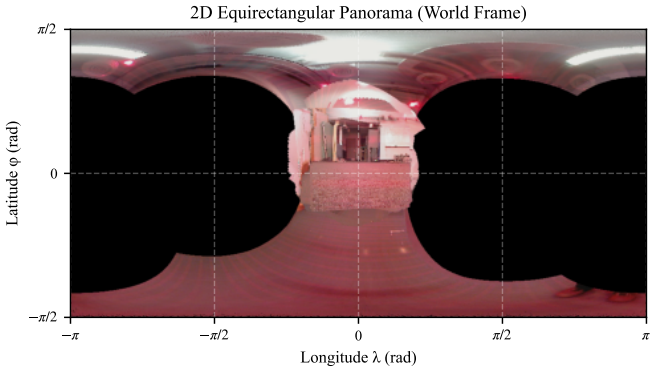


Ground-truth panorama (dataset 1)

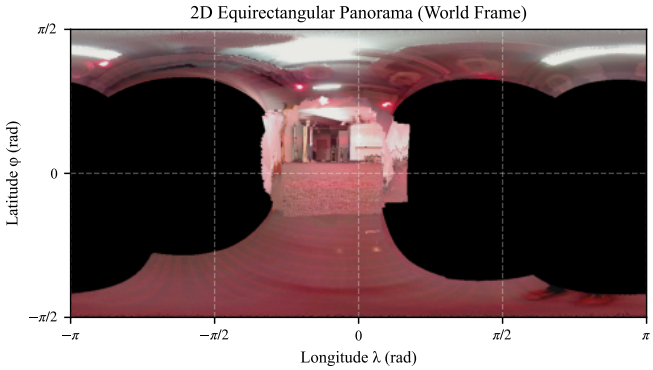


Estimated panorama (dataset 1)

Fig. 12. Panorama visualization for dataset 1.

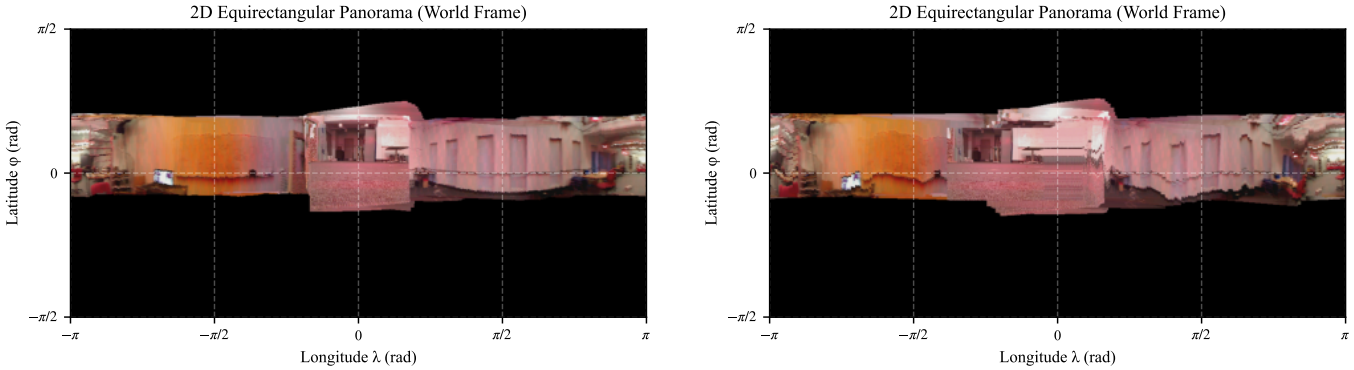


Ground-truth panorama (dataset 2)



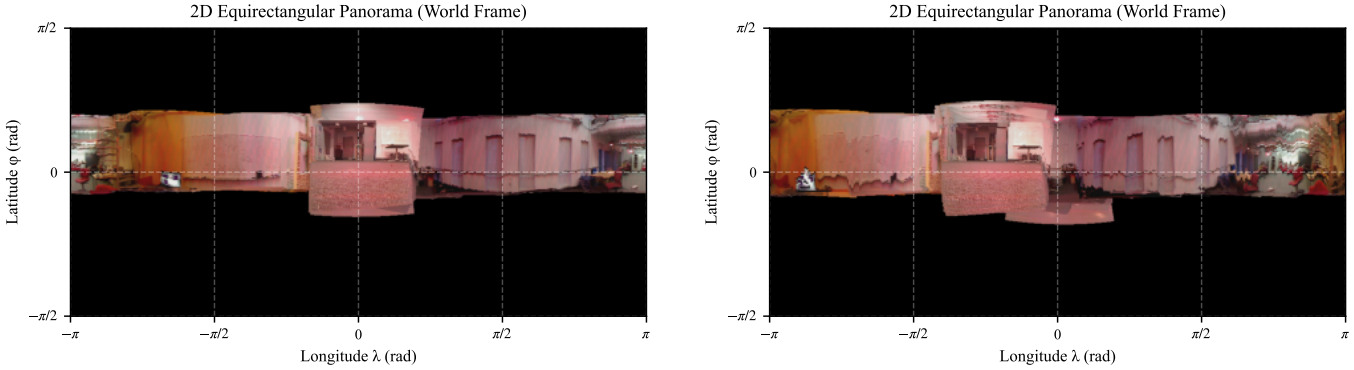
Estimated panorama (dataset 2)

Fig. 13. Panorama visualization for dataset 2.



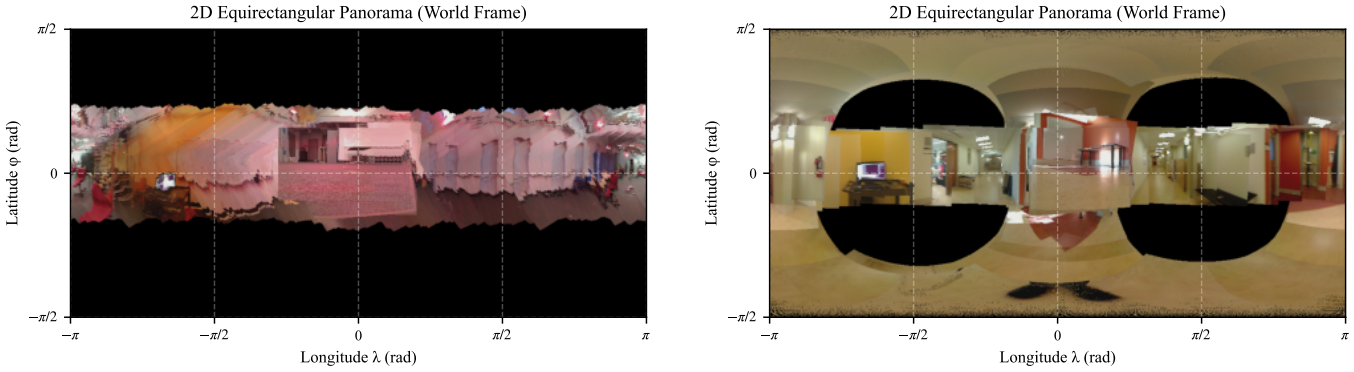
Ground-truth panorama (dataset 8) Estimated panorama (dataset 8)

Fig. 14. Panorama visualization for dataset 8.



Ground-truth panorama (dataset 9) Estimated panorama (dataset 9)

Fig. 15. Panorama visualization for dataset 9.



Estimated panorama (dataset 10) Estimated panorama (dataset 11)

Fig. 16. Estimated panoramas for datasets 10 and 11.