DRaM-LHM: A Quaternion Framework for Iterative Camera Pose Estimation

Supplementary Material

7. Mathematical details

For detailed derivations and a comprehensive breakdown of the equations, we refer the reader to the foundational work by [9], Here, we present the complete formulation of the $\mathbf{S}_{\mathcal{ORTHO}}(q_{ij}; x, y, z; u, v)$ as follows:

$$
\begin{aligned}
\mathbf{S}_{\mathcal{ORTHO}}&(q_{ij}; x, y, z; u, v) = \\
& {q_{00}}^2\, xx + {q_{11}}^2 xx + {q_{22}}^2 xx + {q_{33}}^2 xx + {q_{00}}^2\, yy + {q_{11}}^2 yy \\
& + {q_{22}}^2 yy + {q_{33}}^2 yy - 4q_{00}q_{01}\, yz + 4q_{00}q_{02}\, xz \\
& + 2q_{00}q_{11}\, xx - 2q_{00}q_{11}\, yy + 8q_{00}q_{12}\, xy + 4q_{00}q_{13}\, xz \\
& - 2q_{00}q_{22}\, xx + 2q_{00}q_{22}\, yy + 4q_{00}q_{23}\, yz - 2q_{00}q_{33}\, xx \\
& - 2q_{00}q_{33}\, yy - 2q_{00}\, ux - 2q_{00}\, vy + 4{q_{01}}^2 zz \\
& - 8q_{01}q_{03}\, xz + 4q_{01}q_{11}\, yz - 8q_{01}q_{12}\, xz \\
& - 4q_{01}q_{22}\, yz - 8q_{01}q_{23}\, zz + 4q_{01}q_{33}\, yz + 4q_{01}\, vz \\
& + 4{q_{02}}^2 zz - 8q_{02}q_{03}\, yz + 4q_{02}q_{11}\, xz + 8q_{02}q_{12}\, yz \\
& + 8q_{02}q_{13}\, zz - 4q_{02}q_{22}\, xz - 4q_{02}q_{33}\, xz - 4q_{02}\, uz \\
& + 4{q_{03}}^2 xx + 4{q_{03}}^2 yy - 8q_{03}q_{11}\, xy + 8q_{03}q_{12}\, xx \\
& - 8q_{03}q_{12}\, yy - 8q_{03}q_{13}\, yz + 8q_{03}q_{22}\, xy + 8q_{03}q_{23}\, xz \\
& + 4q_{03}\, uy - 4q_{03}\, vx + 4q_{11}q_{13}\, xz - 2q_{11}q_{22}\, xx \\
& - 2q_{11}q_{22}\, yy - 4q_{11}q_{23}\, yz - 2q_{11}q_{33}\, xx + 2q_{11}q_{33}\, yy \\
& - 2q_{11}\, ux + 2q_{11}\, vy + 4{q_{12}}^2 xx + 4{q_{12}}^2 yy \\
& + 8q_{12}q_{13}\, yz + 8q_{12}q_{23}\, xz - 8q_{12}q_{33}\, xy - 4q_{12}\, uy \\
& - 4q_{12}\, vx + 4{q_{13}}^2 zz - 4q_{13}q_{22}\, xz - 4q_{13}q_{33}\, xz \\
& - 4q_{13}\, uz + 4q_{22}q_{23}\, yz + 2q_{22}q_{33}\, xx - 2q_{22}q_{33}\, yy \\
& + 2q_{22}\, ux - 2q_{22}\, vy + 4{q_{23}}^2 zz - 4q_{23}q_{33}\, yz \\
& - 4q_{23}\, vz + 2q_{33}\, ux + 2q_{33}\, vy + uu + vv\,.
\end{aligned}
$$

and all 10 subdeterminants:

$$
\begin{aligned}
& d_{xyz} \to det\left(\begin{bmatrix} xx & xy & xz \\ xy & yy & yz \\ xz & yz & zz \end{bmatrix}\right), \quad d_{yzu} \to det\left(\begin{bmatrix} xy & xz & ux \\ yy & yz & uy \\ zy & zz & uz \end{bmatrix}\right), \\
& d_{zyu} \to det\left(\begin{bmatrix} xz & xx & ux \\ yz & yx & uy \\ zz & zx & uz \end{bmatrix}\right), \quad d_{xyu} \to det\left(\begin{bmatrix} xx & xy & ux \\ yx & yy & uy \\ zx & zy & uz \end{bmatrix}\right), \\
& d_{yzv} \to det\left(\begin{bmatrix} xy & xz & vx \\ yy & yz & vy \\ zy & zz & vz \end{bmatrix}\right), \quad d_{zxv} \to det\left(\begin{bmatrix} xz & xx & vx \\ yz & yx & vy \\ zz & zx & vz \end{bmatrix}\right), \\
& d_{xyv} \to det\left(\begin{bmatrix} xx & xy & vx \\ yx & yy & vy \\ zx & zy & vz \end{bmatrix}\right), \quad d_{xuv} \to det\left(\begin{bmatrix} xx & ux & vx \\ yx & uy & vy \\ zx & uz & vz \end{bmatrix}\right), \\
& d_{yuv} \to det\left(\begin{bmatrix} xy & ux & vx \\ yy & uy & vy \\ zy & uz & vz \end{bmatrix}\right), \quad d_{zyv} \to det\left(\begin{bmatrix} xz & ux & vx \\ yz & uy & vy \\ zz & uz & vz \end{bmatrix}\right).
\end{aligned}
\tag{14}
$$

The last three subdeterminants, d_{xuv}, d_{yuv}, and d_{zuv}, are not featured in the orthographic projection least square solution referenced in Equation 10. However, their ratios relative to the self-covariance correspond to the third row of the optimized rotation matrix in noise-free conditions. As previously mentioned, this row is obtained through the cross product of the first two rows.

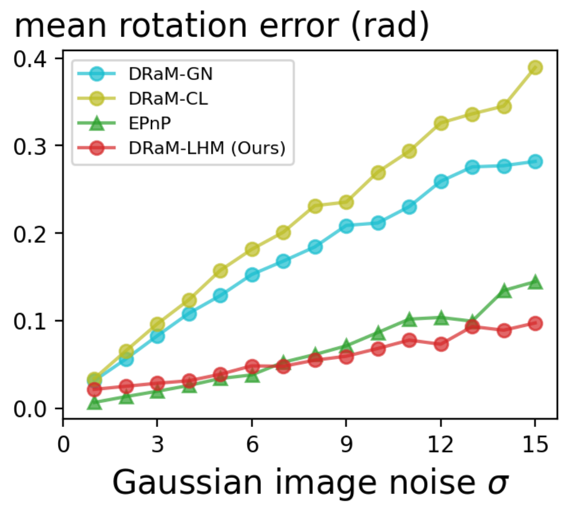

Figure 8. Comparative analysis of the mean rotation error θ across various iterative methods, using EPnP as a benchmark, with Gaussian image noise σ varied and the number of control points fixed at 8.

8. Sanity check on different iterative method

There are several ways to refine the pose after initialization. Based on our sanity checks, LHM's Orthogonal Iterative method proves to be the most effective, outperforming direct numerical least squares optimization via the Gauss-Newton method in minimizing both reprojection error (DRaM-GN) and collinearity error (DRaM-CL), even when implemented with the Ceres solver [1]. The key reason for this superiority lies in the presence of noise, when noise is introduced, the gradient descent path in raw pixel space or along its corresponding ray tends to deviate from the true pose, making direct optimization less reliable.

9. DRaM-LHM algorithm

The algorithmic description of DRaM-LHM requires specifying two key thresholds: a collinearity error threshold and a maximum number of iterations. After initialization using DRaM, the camera poses are refined by minimizing the collinearity error through orthogonal iterations.

Algorithm 1 DRaM-LHM: Solve PnP

Require: Number of points, $n \geq 4$
3D points $\mathbf{X}_i \in \mathbb{R}^3, i = 1, ..., n$
2D projections $\mathbf{u}_i = (x_i, y_i, 1)^\top, i = 1, ..., n$

$\mathbf{V} \leftarrow \{\mathbf{u}_i\}_{i=1}^n$ ▷ Initialize line-of-sight matrix
Maximum number of iterations, T_{max}
Collinearity error threshold, ϵ

$\bar{\mathbf{X}}, \bar{\mathbf{u}} \leftarrow \frac{1}{n}\sum_{i=1}^n \mathbf{X}_i, \frac{1}{n}\sum_{i=1}^n \mathbf{u}_i$ ▷ Compute centroids
$\mathbf{X}_i \leftarrow \mathbf{X}_i - \bar{\mathbf{X}}, \quad \mathbf{u}_i \leftarrow \mathbf{u}_i - \bar{\mathbf{u}}$ ▷ Centralize points
$\mathbf{t}_0 \leftarrow \begin{bmatrix} \bar{\mathbf{X}}_{xy} - \bar{\mathbf{u}}_{xy} \\ \bar{\mathbf{X}}_z - 0 \end{bmatrix}$
$\mathbf{R}_0 \leftarrow$ DRaM+BI$(\{\mathbf{X}_i - \bar{\mathbf{X}}, \mathbf{u}_i - \bar{\mathbf{u}}\}_{i=1}^n)$ Equation 10
$\mathbf{X}_i \leftarrow \mathbf{R}_0\mathbf{X}_i + \mathbf{t}_0, \quad \forall i \in \{1, ..., n\}$
$\epsilon_0 \leftarrow$ collinearity error$(\mathbf{X}_i, \mathbf{V})$ ▷ Compute initial collinearity error
while step $< T_{\text{max}} \wedge$ collinearity error $> \epsilon_{t+1}$ **do**
  $\mathbf{R}_{t+1}, \mathbf{t}_{t+1} \leftarrow$ Update$(\mathbf{X}_i, \mathbf{V})$ ▷ Update using Equation 12 and SVD
  $\mathbf{X}_i \leftarrow \mathbf{R}_{t+1}\mathbf{X}_i + \mathbf{t}_{t+1}, \quad \forall i \in \{1, ..., n\}$ ▷ Update 3D points
  $\epsilon_{t+1} \leftarrow$ collinearity error$(\mathbf{X}_i, \mathbf{V})$ ▷ Compute new collinearity error
  step $\leftarrow$ step $+ 1$
end while
return $(\mathbf{R}, \mathbf{t})$

10. Minimal solver evaluation in RANSAC framework

Following the setup in section 4, we conducted experiments using 30 random points, introducing varying levels of outlier by randomly replacing valid correspondences. As shown in Figure 9, non-minimal solvers (EPnP, DRaM) closely match the accuracy of a minimal-solver (P3P) within (LO)RANSAC. Similar trends were observed for inlier count and success rate. While DRaM-LHM is inherently a non-minimal solver—requiring at least four non-coplanar points to avoid numerical degeneracies—it incurs negligible computational overhead for larger sample size, making it well-suited for robust estimation under challenging conditions. We will discuss this aspect in the final version, as it is a relevant application consideration.

11. Configuration settings in ORB-SLAM2

We used the *rgbd_tum.cc* and *stereo_euroc.cc* executables to perform visual odometry on the TUM RGBD and EuRoC MAV datasets, respectively. This involved calling *TrackRGBD* and *TrackStereo* from the System class, leveraging depth maps or stereo disparity to initialize each frame. To extract 2D-3D correspondences and 6-DoF poses, we implemented a custom function that processes keyframes from ORB-SLAM2's visual odometry pipeline.

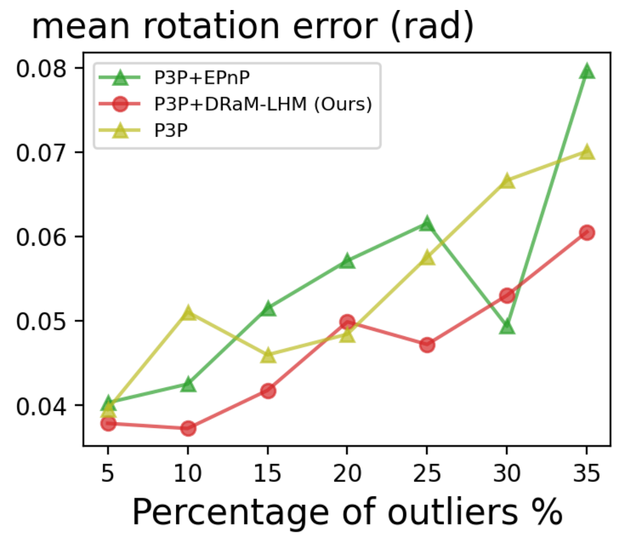

Figure 9. Rotation error θ comparison for P3P, EPnP, DRaM (with (LO)RANSAC) under varying outlier ratios (5%–35%) at fixed noise level σ = 2.0.

12. AUC computation for real data sequences

To evaluate pose estimation accuracy, we compute the **Area Under Curve (AUC)** by measuring the fraction of frames where both rotation and translation errors fall within predefined thresholds. Given a set of cosine differences for rotation and percentage translation errors, the method follows these steps:

1. **Convert rotation errors (in radian) to angular errors (in degrees)** using:

$$\theta = \cos^{-1}(\theta_{rad}) \times \frac{180}{\pi} \tag{15}$$

2. **Compute accuracy at multiple predefined thresholds**, where a frame is considered correct if:

$$\theta \leq \theta_{\text{thresh}} \quad \text{and} \quad t_{\text{error}} \leq t_{\text{thresh}} \tag{16}$$

The thresholds are defined as:
- Rotation error thresholds: $[2°, 4°, 6°, 8°, 10°]$
- Translation error thresholds (percentage of scene scale): $[1.25\%, 3.75\%, 5\%, 10\%]$

3. Integrate the accuracy curve using the **trapezoidal rule**, normalizing over the threshold range to obtain the final AUC score:

$$\text{AUC} = \frac{1}{\theta_{\max} - \theta_{\min}} \int_{\theta_{\min}}^{\theta_{\max}} \text{accuracy}(\theta) d\theta \tag{17}$$

This AUC metric provides a compact evaluation of pose accuracy across varying error tolerances, enabling direct comparison between methods.

13. Additional results on real-world data

Alongside Figure 7, we report the AUC across varying numbers of points for the EuRoC MAV MH01 sequence.

Methods/Number of Points	10	20	30	40	50
SQPnP	0.5682	0.5730	0.5739	0.5742	0.5755
EPnP	0.5008	0.5190	0.5239	0.5301	0.5286
DRaM (8 iters)	0.4944	0.5414	0.5597	0.5634	0.5672
DRaM (10 iters)	0.4705	0.5545	0.5642	0.5703	0.5708

Table 3. AUC across varying numbers of 2D-3D pairs (10–50) for three methods on Euroc MAV's MH01.