7. Supplementary Material: InCaRPose

7.1. Supplementary Tables

Table 6. Detailed inference runtime measurements on a single NVIDIA RTX 4090 GPU. We report average per-frame latency (ms), frames per second (FPS), and relative speedup with respect to the FP32 baseline at the corresponding backbone and resolution.

Backbone	Res.	Config	Latency (ms)	FPS	Speedup
Small	224	Baseline (FP32)	14.65	68.28	1.00
		FP16	14.57	68.64	1.01
		`torch.compile`	14.00	71.41	1.05
		FP16 + `torch.compile`	14.48	69.06	1.01
Small	512	Baseline (FP32)	24.75	40.40	1.00
		FP16	16.49	60.65	1.50
		`torch.compile`	24.38	41.03	1.02
		FP16 + `torch.compile`	14.76	67.75	1.68
Base	224	Baseline (FP32)	14.91	67.06	1.00
		FP16	14.93	66.97	1.00
		`torch.compile`	15.36	65.12	0.97
		FP16 + `torch.compile`	14.46	69.16	1.03
Base	512	Baseline (FP32)	31.18	32.07	1.00
		FP16	17.62	56.75	1.77
		`torch.compile`	30.79	32.48	1.01
		FP16 + `torch.compile`	17.62	56.74	1.77
Large	224	Baseline (FP32)	21.51	46.48	1.00
		FP16	21.73	46.02	0.99
		`torch.compile`	17.04	58.69	1.26
		FP16 + `torch.compile`	14.62	68.38	1.47
Large	512	Baseline (FP32)	62.49	16.00	1.00
		FP16	30.44	32.85	2.05
		`torch.compile`	60.97	16.40	1.02
		FP16 + `torch.compile`	24.70	40.49	2.53

Table 7. Backbone ablation on COLMAP ground truth (distorted images and 224 resolution).

Backbone	Metric	Rot. Err (°)	Dir. Err (°)
Dune-Base	Mean	6.26	52.11
	Median	3.88	50.90
DUST3R-Large	Mean	6.98	81.52
	Median	5.29	73.93
DINOv2-Base	Mean	8.11	45.60
	Median	5.30	39.58
DINOv3-Base	Mean	8.36	48.74
	Median	6.14	41.23

7.2. Rotation Representation

We investigate different rotation representations and show how each can be mapped back to a uniform rotation matrix for the Universal Loss as described below:

1. **Rotation Vector** ($\mathbb{R}^3$): The rotation is represented by a compact axis-angle vector ω. The vector's direction specifies the rotation axis $\mathbf{u}$, and its magnitude represents the rotation angle $\theta = \|\omega\|_2$ in radians. The mappingto a rotation matrix R is given by Rodrigues' [48] formula:
$$R = I + \frac{\sin\theta}{\theta}[\omega]_\times + \frac{1-\cos\theta}{\theta^2}[\omega]_\times^2 \tag{3}$$
where $[\omega]_\times$ is the skew-symmetric matrix of ω. The final output is $\mathbf{y} = [\omega^\top, \mathbf{t}^\top]^\top$.
2. **Euler Angles: Intrinsic Rotation** ($\mathbb{R}^3$): We support intrinsic rotations (moving axes) using the standard ZYX convention. Given angles (α, β, γ), the final rotation matrix is computed by successive rotations around the transformed axes:
$$R_{\text{int}} = R_z(\alpha)R_{y'}(\beta)R_{x''}(\gamma) \tag{4}$$
The final output is $\mathbf{y} = [\alpha, \beta, \gamma, t_x, t_y, t_z]^\top$.
3. **Euler Angles: Extrinsic Rotation** ($\mathbb{R}^3$): Extrinsic rotations are performed around the fixed, global axes (X, Y, Z). For a sequence (γ, β, α), the resulting matrix is:
$$R_{\text{ext}} = R_z(\alpha)R_y(\beta)R_x(\gamma) \tag{5}$$
The final output is $\mathbf{y} = [\gamma, \beta, \alpha, t_x, t_y, t_z]^\top$.
4. **Quaternions** ($\mathbb{R}^4$): The rotation is represented by a unit quaternion $\mathbf{q} = [w, x, y, z]^\top$, where $\|\mathbf{q}\|_2 = 1$. The mapping to R is defined as:
$$R = \begin{bmatrix} 1-2(y^2+z^2) & 2(xy-wz) & 2(xz+wy) \\ 2(xy+wz) & 1-2(x^2+z^2) & 2(yz-wx) \\ 2(xz-wy) & 2(yz+wx) & 1-2(x^2+y^2) \end{bmatrix} \tag{6}$$
The final output is $\mathbf{y} = [\mathbf{q}^\top, \mathbf{t}^\top]^\top$.
5. **Rotation Matrix** ($\mathbb{R}^9$): The rotation is represented directly by the flattened elements of $R \in \mathbb{R}^{3\times3}$. The matrix must satisfy the constraints of the Special Orthogonal group:
$$\text{SO}(3) = \{R \in \mathbb{R}^{3\times3} : R^\top R = I, \det(R) = +1\} \tag{7}$$
The final output is the flattened nine elements of R followed by $\mathbf{t}$, resulting in $\mathbf{y} = [r_{11}, r_{12}, \ldots, r_{33}, t_x, t_y, t_z]^\top$.

If the rotation is described as rotation matrix we can describe the full relative transformation as follows:

$$T_{\text{rel}} = T_{\text{view1}}^{-1} T_{\text{view2}}, \qquad T_{\text{view1}} \equiv T_{\text{ref}}$$

$$T_{\text{view}} = \begin{pmatrix} R_{\text{view}} & t_{\text{view}} \\ 0 & 1 \end{pmatrix}, \qquad T_{\text{rel}} = \begin{pmatrix} R_{\text{rel}} & t_{\text{rel}} \\ 0 & 1 \end{pmatrix}$$

$$T_{\text{view1}}^{-1} = \begin{pmatrix} R_{\text{view1}}^\top & -R_{\text{view1}}^\top t_{\text{view1}} \\ 0 & 1 \end{pmatrix}$$

$$R_{\text{rel}} = R_{\text{view1}}^\top R_{\text{view2}}, \tag{8}$$
$$t_{\text{rel}} = R_{\text{view1}}^\top \big(t_{\text{view2}} - t_{\text{view1}}\big). \tag{9}$$

7.3. Error Metrics and Loss Functions

The introduction of various transformation representations necessitates a systematic investigation into the loss functions and error metrics tailored to each output format. We observed that the choice of representation significantly impacts optimization behavior and that final model performance is highly sensitive to the specific objective function employed.

7.3.1. Individual Error Metrics

Geodesic Distance:. Measures the minimum rotation angle required to align the estimated rotation matrix $\mathbf{R}_{\text{est}}$ with the ground truth $\mathbf{R}_{\text{gt}}$:

$$e_{\text{rot\_geo}} = \arccos\left(\frac{\text{Tr}(\mathbf{R}_{\text{est}}^{\top}\mathbf{R}_{\text{gt}}) - 1}{2}\right) \tag{10}$$

Quaternion Error:. Minimizes the angular distance on the hypersphere between unit quaternions $\mathbf{q}_{\text{est}}$ and $\mathbf{q}_{\text{gt}}$, accounting for the double-cover property of $SO(3)$:

$$e_{\text{rot\_quat}} = 2\arccos\left(\left|\mathbf{q}_{\text{est}}^{\top}\mathbf{q}_{\text{gt}}\right|\right) \tag{11}$$

Euclidean Distance:. Measures the absolute metric distance between translation vectors in meters:

$$e_{\text{trans\_eucl}} = \left\|\mathbf{t}_{\text{est}} - \mathbf{t}_{\text{gt}}\right\|_2 \tag{12}$$

Translation Direction Error:. Measures the angular difference between predicted and ground truth translation vectors, providing a scale-invariant metric:

$$e_{\text{trans\_dir}} = \arccos\left(\frac{\mathbf{t}_{\text{est}}^{\top}\mathbf{t}_{\text{gt}}}{\|\mathbf{t}_{\text{est}}\| \cdot \|\mathbf{t}_{\text{gt}}\|}\right) \tag{13}$$

7.3.2. Composite Loss Functions

During training, these metrics are combined into the following loss formulations:

Universal Transformation Loss. This loss handles full transformation matrices $\mathbf{T} \in SE(3)$ (rotation and translation). We decompose the matrices back into rotation and translation components and apply a weighted sum of geodesic and Euclidean errors:

$$\mathcal{L}_{\text{universal}} = \mathbb{E}\left[e_{\text{rot\_geo}}\right] + \alpha \cdot \mathbb{E}\left[e_{\text{trans\_eucl}}\right] \tag{14}$$

where α is a weighting factor to balance the different units.

Reloc3r Loss. Inspired by [17], this loss is designed for the estimation of relative poses, where translation is predicted as a unit vector. It combines geodesic rotation error with translation direction error:

$$\mathcal{L}_{\text{Reloc3r}} = \mathbb{E}\left[e_{\text{rot\_geo}} + \alpha \cdot e_{\text{trans\_dir}}\right] \tag{15}$$

Mean Squared Error (MSE) Loss. A standard baseline applied to raw output vectors $\mathbf{p} \in \mathbb{R}^d$. This is utilized for different output representations, such as Euler angles ($d = 6$) or Quaternions ($d = 7$):

$$\mathcal{L}_{\text{MSE}} = \frac{1}{d}\left\|\mathbf{p}_{\text{est}} - \mathbf{p}_{\text{gt}}\right\|_2^2 \tag{16}$$

Quaternion-based Pose Loss:. This is our primary loss for the metric estimation. It utilizes the quaternion error for orientation and either Euclidean distance (for metric pose) or direction error (for scale-invariant pose) for translation:

$$\mathcal{L}_{\text{quat}} = \mathbb{E}\left[e_{\text{rot\_quat}}\right] + \alpha \cdot \mathbb{E}\left[e_{\text{trans}}\right] \tag{17}$$

The choice of e_{trans} ($e_{\text{trans\_eucl}}$ or $e_{\text{trans\_dir}}$) allows the model to trade off between absolute metric accuracy and directional consistency.

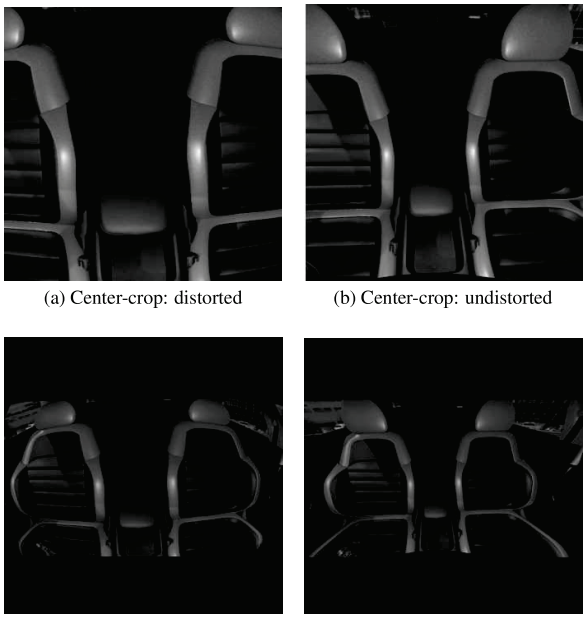
7.4. Datasets

We show two different datasets in Fig. 6. The center-crop dataset crops the image, resulting in data loss, while the zero-padded dataset does not cut the image. Instead, it pads the image to make it square. In both pipelines, images are converted to 8-bit RGB and, optionally, undistorted when intrinsics are available. The center-crop dataset resizes and center-crops to the target resolution, then applies standard ImageNet normalization. The zero-padded dataset rescales while preserving the aspect ratio, pads to a square canvas, and applies the same per-channel normalization. The two datasets also demonstrate the trade-off between higher detail per resolution (center-crop dataset) and more border information (zero-padded dataset) while producing a predefined, fixed image size.

7.5. Ground Truth Discussion

To assess the reliability of ground truth pose estimation in confined automotive interiors, we compare ArUco-based tracking with trajectories reconstructed using COLMAP. Both translations are normalized to a uniform scale for a consistent comparison. All sequences are expressed in a common local coordinate system defined by a single reference view within the vehicle cabin. This reference view is selected to maximize feature overlap across all other views. This is important to maximize the likelihood that matching ArUco markers are visible across all scenes. All camera poses are represented as relative transformations with respect to this origin.

Table 8 summarizes the observed discrepancies between COLMAP and ArUco-based estimates. Rotation error is measured as the angular deviation between the COLMAP-estimated orientation and the ArUco-based ground truth. Since COLMAP recovers translation only in unknown scale, translation error is evaluated exclusively in terms of direction,



(a) Center-crop: distorted (b) Center-crop: undistorted

(c) Zero-padded: distorted (d) Zero-padded: undistorted

Figure 6. Comparison of preprocessing methods. (a) and (b): images are cropped to the center. (c) and (d): images are zero-padded to a square aspect ratio. This adjustment is necessary to handle varying input resolutions within the dataset while maintaining a consistent model input.

Table 8. Rotation and translation direction errors. Translation direction error is only evaluated when the relative ArUco ground truth translation exceeds 0.1 m.

Metric	Value
Max rotation error	19.86°
Mean rotation error	3.08°
Median rotation error	2.27°
Max translation direction error	55.83°
Mean translation direction error	5.75°
Median translation direction error	4.60°

computed as the angular difference between the estimated and ground-truth translation vectors. To avoid degenerate cases, translation direction error is reported only for frames where the relative Euclidean displacement of the ArUco ground truth exceeds 0.1 m. We apply this threshold to ensure that only frames with meaningful translation contribute to this metric.

As illustrated in Fig. 7 and Fig. 9, discrepancies between the two methods arise in corner views. In particular, COLMAP occasionally fails to recover vertical displacement (translation along the y-axis) or rotation around the z-axis, resulting in poses that are inconsistent with the physical camera placement. Such failure cases are characteristic of

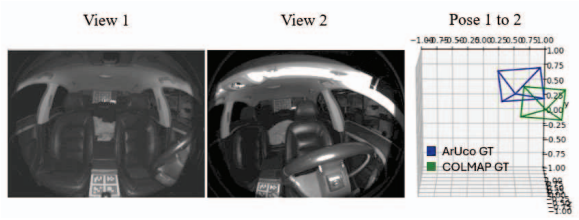


Figure 7. COLMAP fails to estimate translation along the y-axis. In the first view the camera is to the side of the steering wheel. In the second view the camera moved upwards (y) above the steering wheel. Translation is normalized.

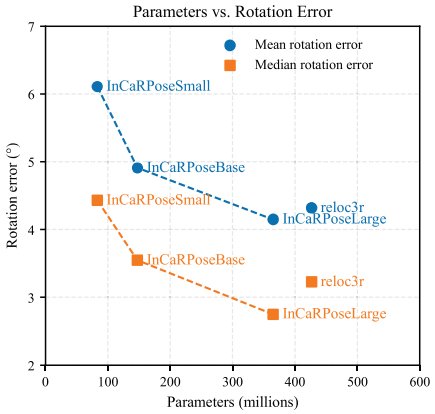


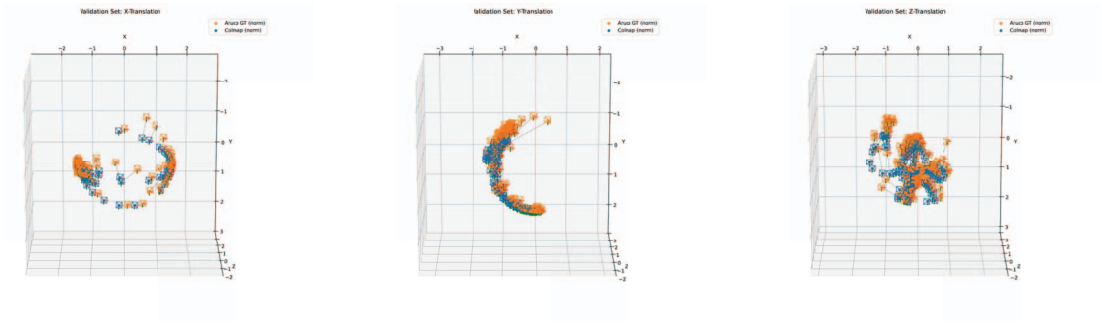
Figure 8. Rotation error in degrees versus the number of parameters. Evaluated on image resolution of 224 on the In-Cabin-Pose dataset.

confined interior environments, where limited baseline, weak texture, and reflective surfaces can lead to ill-conditioned structure-from-motion reconstructions.

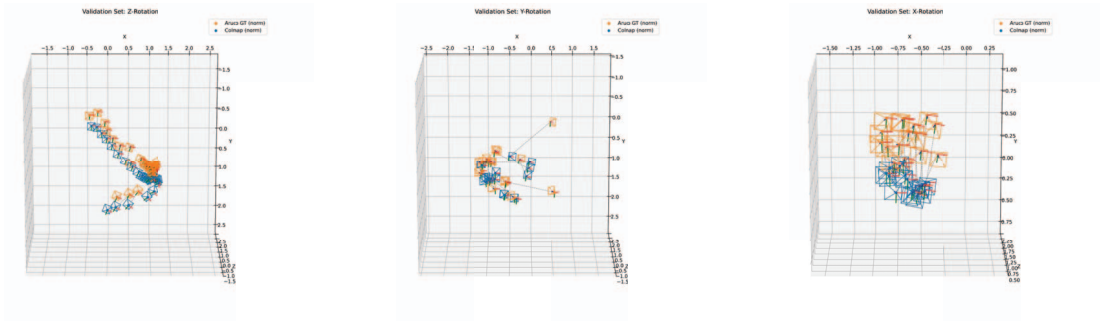
Given that ArUco markers provide metric-scale translation and yield more physically plausible poses in these edge cases, we adopt the ArUco-based estimates as the primary ground truth for all quantitative evaluations in this work. For completeness and reproducibility, we additionally release the corresponding COLMAP-based trajectories.

7.6. ArUco-free Inference

We also provide several samples with occluded ArUco markers, shown in Fig. 10. These examples demonstrate that the model does not rely on the presence of ArUco markers in the real-world test data to make its predictions. For this data collection, we first captured frames in which the ArUco markers were fully visible. We then physically occluded the markers and recorded additional frames. Finally, we assigned the ground-truth poses from the visible-marker frames to the corresponding occluded-marker frames.

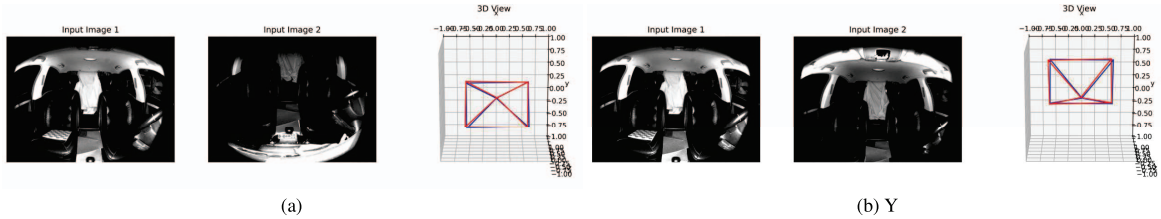


(a) X-Translation (b) Y-Translation (c) Z-Translation

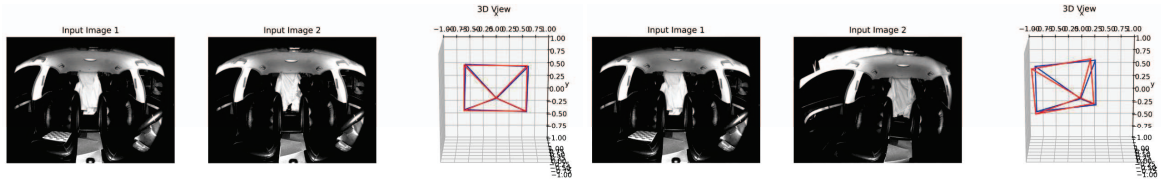


(d) Z-Rotation (e) Y-Rotation (f) X-Rotation

Figure 9. ArUco (orange) vs. COLMAP (blue) camera trajectories for intervals focused on specific transformations. Each sequence predominantly captures motion along the target axis, though residual degrees-of-freedom are also present.



(a) (b) Y



(c) Z (d) Y

Figure 10. Inference on frames with physically occluded ArUco markers. We also changed the reference image, in which a different object (a high-contrast checkerboard on the left seat) is now visible in only one image to additionally challenge the model.