The 8-Point Algorithm as an Inductive Bias for Relative Pose Prediction by ViTs

Chris Rockwell, Justin Johnson, David F. Fouhey
University of Michigan

Our supplement presents the following.

Additional experimental details for the paper's main results. These consist of detailed network architectures (§1), descriptions of datasets and data pre-processing (§2), and additional analysis that would not fit in the main paper (§3).

Discussion of the Essential Matrix Module. We present additional discussion and exposition of the Essential Matrix Module. This consists of a more detailed derivation of the fact that the unique entries of $\mathbf{U}^\top\mathbf{U}$ can be written by $\mathbf{\Phi}^\top\mathbf{A}\mathbf{\Phi}$ as described in the main paper (§4), how accurately the network can compute $\mathbf{\Phi}^\top\mathbf{A}\mathbf{\Phi}$ in practice (§5), as well as an explicit writing-out of one of the key steps of this (§7). It also includes a detailed description of the experiment done with synthetic data which appears in the method section (§6).

1. Detailed Network Architectures

Our architecture is outlined below and detailed in the following two-page Table 1.

Backbone. Our backbone consists of a vanilla encoder of Residual Modules and one Block, followed by a vanilla ViT. The model ends with a three-layer MLP.

Essential Matrix Module. Recall from the paper, the Essential Matrix Module is closely built off a standard Cross-Attention Block. The changes we make are carefully chosen to study their contribution to performance. Changes can be seen in the "Essential Matrix Cross-Attn" block of the architecture. First, the final dimension of query, key and values is increased by a size of 6; positional encodings fill these 6 new spaces for value matrices. Second, Softmax is computed over both the last and second to last dimension of affinities, and multiplied elementwise. Finally, Bi-Directional attention is computed, $V.T@A@V$. Note this results in output of different shape than standard Cross-Attention.

Baselines. Baselines use generally the same components as our entire architecture, with very minor changes. This helps us study our proposed contributions.

CNN Pose Regressor. CNN Pose Regressor follows our model architecture, with the exception it only uses encoder, and one pooling layer, before the MLP. The pooling layer consists of two 1x1 Convs with Batchnorm (and ReLU before the second Conv). Feature size is reduced to 96, then 43; resulting in MLP input of $24768 = 43 * 24 * 24$. We do this so the input to the MLP is comparable to our method ($29400 = 2 * 3 * (64 + 6) * (64 + 6)$), and so this method is less prone to poor overfitting. Early experiments with bigger input size to MLP hurt performance. The CNN and MLP are otherwise identical to ours.

+ViT. Starting from the CNN Pose Regressor, we simply add the ViT Layers back from our architecture. This is followed by a vanilla Cross-Attention block. This uses the same pooling layer as the CNN. The Cross Attention-Block can be distinguished from our Essential Matrix Module by changing bidirectional attention, dual softmax and positional encoding. Looking in the table, the differences can be found in Essential Matrix Cross-Attn. First, affinities are calculated using the standard $(A@V).T$. Second, Softmax is not applied over $dim = -2$. Third, values do not get positional encodings.

+Bilinear Attention. This method uses our architecture as-is, with the exception of dual softmax and positional encoding. Looking in the table, the differences can be found in Essential Matrix Cross-Attn. First, Softmax is not applied over $dim = -2$. Second, values will not get positional encodings.

+Dual Softmax. This method uses our architecture as-is, with the exception of positional encoding. Looking in the table, the difference can be found in Essential Matrix Cross-Attn – values will no longer get positional encodings.

2. Dataset Information

Matterport3D. Matterport3D is a collection of scanned indoor scenes. We use the image pairs Jin *et al.* collected using the Habitat simulator. The number of image pairs in the train, val and test set are 31932, 4707, and 7996, respectively. Images are originally 480x640, but are downsampled to 256x256 for models. Images are collected using a camera at random height of 1.5-1.6m, with a downward tilt of 11 degrees to simulate human perspective. Candidate pairs are randomly sampled cameras within each room. Next, Jin *et al.* detect planes, and select pairs such that at least 3

Table 1. **Model Architecture**. Detailed model architecture, broken down into sub-components. Please note, some components have more complicated structure, so we define operations at the beginning and forward pass at the bottom. For instance, in the Residual Module we define the two branches, followed by the forward pass calling each branch. We use $H = 24$, $D = 192$, $N_h = 3$ in accordance with standard ViT-Tiny. We do not define ResNet Blocks below, as we use the standard implementation available publicly.

Overview

Operation	Output Shape
Input Image	$2 \times 3 \times 256 \times 256$
Encoder	$2 \times H \times H \times D$
ViT Layer (x5)	$2 \times H \times H \times D$
Essential Matrix Module	$2 \times N_h \times (D/N_h + 6) \times (D/N_h + 6)$
MLP	7

Encoder

Operation	Output Shape
ResNet-18 Block 1	$2 \times 56 \times 56 \times 64$
ResNet-18 Block 2	$2 \times H \times 28 \times 128$
Residual Module	$2 \times H \times H \times D$

MLP

Operation	Output Shape
Linear & ReLU (x2)	512
Linear & ReLU	7

Residual Module

Operation	Output Shape
Branch A	
2D Conv k=3 s=1, BN, ReLU	$2 \times 28 \times 28 \times D$
2D Conv k=5 s=1, BN, ReLU	$2 \times H \times H \times D$
Branch B	
2D Conv k=5 s=1, BN	$2 \times H \times H \times D$
Forward Pass	
ReLU(Branch A + Branch B)	$2 \times H \times H \times D$

planes are shared between images, and at least 3 planes are unique to each image. The average rotation is 53 degrees, translation 2.3m, and overlap 21%.

InteriorNet. InteriorNet consists of 10,050 indoor panoramic views across 112 synthetic houses. 82 houses are used for training and 30 are used for testing. We sample paired images from the panoramas using the same procedure as Cai *et al.* We also follow their image selection process, which samples images over a uniform distribution of angles (yaw in [-180, 180]; pitch in [-30,30]) within panoramas. Their process samples 100 images per panorama, filters images too close to walls, and does not apply roll; arguing this does not affect performance. Images are 256x256 and have 90° FOV. For full details see the original paper.

For the InteriorNet-T dataset, pairs are selected from different panoramas, resulting in translation; for InteriorNet, pairs are selected from the same panorama, resulting in no translation. Translations in InteriorNet-T are selected to be less than 3m. The full set of extracted image pairs on InteriorNet is 250K (610K for InteriorNet-T). Note these train set sizes are smaller than reported in Cai *et al.* (roughly 1M and 700k) but were supplied by the authors directly and via their repo; these sets replicate their reported paper results. Both have a test set of 1K pairs. We consider only overlapping pairs, making the train set smaller: InteriorNet: 150K, InteriorNet-T: 350K. The test set sizes are also reduced after filtering for overlap. Pairs are further broken down into large or small overlap, thresholded using rotation of 45 degrees. InteriorNet test set: 695 pairs (403 large overlap, 292 small overlap). InteriorNet-T test set: 659 pairs (335 large overlap, 324 small overlap).

Table 2. **Model Architecture: Essential Matrix Module.** Details of essential matrix module and ViT layer.

ViT Layer	
Operation	Output Shape
Attn	
Q,K,V = Linear	$2 \times N_h \times (H \times H) \times (D/N_h)$
A = Softmax(Q @ K.T, dim=-1)	$2 \times N_h \times (H \times H) \times (H \times H)$
(A @ V).T	$2 \times H \times H \times D$
Linear	$2 \times H \times H \times D$
MLP	
Linear & GeLU	$2 \times H \times H \times (D \times 4)$
Linear	$2 \times H \times H \times D$
Forward Pass	
Attn(LayerNorm) + Residual	$2 \times H \times H \times D$
MLP(LayerNorm) + Residual	$2 \times H \times H \times D$

Essential Matrix Module	
Operation	Output Shape
MLP	
Linear & GeLU	$2 \times H \times H \times (D \times 4)$
Linear	$2 \times H \times H \times D$
Forward Pass	
Essential Matrix Cross-Attn(LayerNorm) + Residual	$2 \times N_h \times (D/N_h + 6) \times (D/N_h + 6)$
MLP(LayerNorm) + Residual	$2 \times N_h \times (D/N_h + 6) \times (D/N_h + 6)$

Essential Matrix Cross-Attn	
Operation	Output Shape
Attn Branch	
A = Softmax(Q @ K.T, dim=-1) $\times$ Softmax(Q @ K.T, dim=-2)	$N_h \times (H \times H) \times (H \times H)$
(V.T @ A @ V).T	$N_h \times (D/N_h + 6) \times (D/N_h + 6)$
Linear	$N_h \times (D/N_h + 6) \times (D/N_h + 6)$
Forward Pass	
Q_1, K_1, V_1 = Linear(Input[0])	$N_h \times (H \times H) \times (D/N_h + 6)$
Q_2, K_2, V_2 = Linear(Input[1])	$N_h \times (H \times H) \times (D/N_h + 6)$
$V_1[..., -6:], V_2[..., -6:]$ = Pos Encoding	$N_h \times (H \times H) \times (D/N_h + 6)$
Concat(Attn Branch(Q_1, K_2, V_2), Attn Branch(Q_2, K_1, V_1))	$2 \times N_h \times (D/N_h + 6) \times (D/N_h + 6)$

StreetLearn. StreetLearn consists of 143K panoramic outdoor views in New York City and Pittsburgh; we follow the setup of Cai *et al.* which uses 56K panoramas from Manhattan, with 1K randomly allocated for testing. We also follow their image selection process, which samples images over a uniform distribution of angles (yaw in [-180, 180]; pitch in [-45,45]) within panoramas. Their process samples 100 images per panorama, and does not apply roll; arguing this does not affect performance. Images are 256x256 and have 90° FOV. For full details see the original paper. For the StreetLearn-T dataset, pairs are selected from different panoramas, resulting in translation; for StreetLearn, pairs are selected from the same panorama, resulting in no translation. Translations in StreetLearn-T are selected to be less than 10m. The full set of extracted image pairs on StreetLearn is 1.1M (670K for StreetLearn-T). Both have a test set of 1K pairs. We consider only overlapping pairs, making the train set smaller: StreetLearn: 460K, StreetLearn-T: 260K. The test set sizes are also reduced after filtering for overlap. Pairs are further broken down

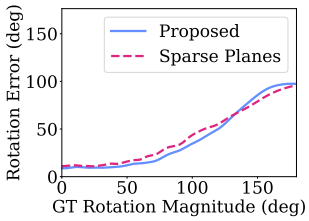

Figure 1. **Mean Error as a Function of Magnitude: Rotation.**

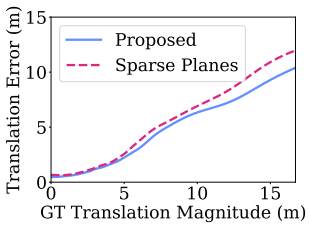

Figure 2. **Mean Error as a Function of Magnitude: Translation.**

into large or small overlap, thresholded using rotation of 45 degrees. StreetLearn test set: 471 pairs (170 large overlap, 301 small overlap). StreetLearn-T test set: 469 pairs (152 large overlap, 317 small overlap).

StreetLearn panoramas were captured based on Google Street View, and therefore contain real people. The authors of the dataset blurred all faces and license plates, and manually reviewed images for privacy. The dataset is distributed only upon request. If individuals request for a panorama to be taken down or blurred, the dataset is updated to reflect the request.

3. Additional Analysis

Error against GT. In Figures 1 and 2, we report error vs. magnitude for rotation and translation, respectively; for our method and baseline Sparse Planes. Lines are fit by applying adaptive kernel density estimation. For both rotation and translation, both methods show similar general trends, increasing error as magnitude increases. In the case of very small rotations (i.e. $< 30°$) or translations (i.e. $< 3m$), Sparse Planes is quite competitive with the proposed method. However, beyond this magnitude, our method tends to be more robust, outperforming until the very most extreme rotations. This is consistent with the qualitative results and CDFs plotted in the paper.

Generalization Across Datasets. A good measure of inductive bias is to evaluate methods when trained on one dataset and tested without fine-tuning on another. We evaluate our method against the most competitive baselines across datasets in Table 3. We tend to generalize better, especially in average error, though indoor $\leftrightarrow$ outdoor does cause a large drop in performance.

Is ViT Backbone Needed? Our proposed method appends an Essential Matrix Module layer to the end of 5 ViT layers. In Table 4, we study the effect of replacing the 5 ViT layers with a comparable CNN-based backbone, a modified PWC-Net [1]. PWC-Net is a good choice as it predicts optical flow, a task closely related to correspondence estimation and therefore useful for downstream pose estimation; TartanVO [2] uses this extractor before predicting relative pose. We use PWC-Net out of the box except for modifications to feature and stride size so that it takes extracted features at the same resolution (24x24) and feature size (192) as the ViT, and produces output features at this same resolution and feature size. The CNN-based backbone performs similarly to the ViT backbone – better on StreetLearn-T but worse on InteriorNet-T. This indicates the Essential Matrix Module may be flexible for use with more generic models. Recall from Tables 3, 5 and 6 in the main paper, the ViT backbone is not the primary reason for our success – the Essential Matrix Module is a helpful inductive bias, particularly in the case of limited data.

Table 3. **Generalization Across Datasets**, training Cai *et al.* and Ours on the opposite of IN-T and SL-T. SuperGlue is trained on ScanNet; the only public version. In format "Large / Small" Overlap.

	InteriorNet-T			StreetLearn-T		
Method	Avg (° ↓)	Med. (° ↓)	10 (% ↑)	Avg (° ↓)	Med. (° ↓)	10 (% ↑)
SuperGlue	35.9 / 85.4	35.9 / 81.6	5.4 / 0.0	48.9 / 85.5	42.9 / 82.4	2.0 / 0.0
Cai *et al.*	31.2 / 67.4	**9.3** / 84.4	**52.0 / 23.8**	43.0 / 71.4	21.8 / 72.5	32.4 / 9.9
Ours	**18.7 / 58.6**	11.1 / **66.4**	45.1 / 10.5	**28.6 / 40.7**	**14.3 / 24.0**	**39.5 / 27.8**

Table 4. **Model Backbone Ablations.** A ViT backbone is not necessary for competitive performance; a CNN replacement performs similarly if we keep the Essential Matrix Module.

		InteriorNet-T			StreetLearn-T		
Overlap	Method	Avg (° ↓)	Med. (° ↓)	10 (% ↑)	Avg (° ↓)	Med. (° ↓)	10 (% ↑)
Large	CNN	5.36	3.62	94.03	**3.74**	**2.29**	**94.74**
	ViT (Ours)	**2.90**	**1.83**	**97.91**	4.08	2.43	90.13
Small	CNN	8.31	4.58	86.42	**7.30**	**3.03**	**90.54**
	ViT (Ours)	**4.48**	**2.38**	**96.30**	9.19	3.25	87.70

Can one Construct the Essential Matrix Module from Conv Kernels? One can construct the essential matrix from $\mathbf{\Phi}^\top \mathbf{A} \mathbf{\Phi}$, which is similar to the attention operation performed in ViTs but not to any operation in Conv layers.

Can Essential Matrix Module Output be used for the 8-Point Algorithm? No, it is an inductive bias used for pose prediction. An analogy is convolution layers, which have the capacity to detect edges but which do not necessarily just detect edges when learned. Likewise, the EM Module has the capacity to represent information like $\mathbf{U}^\top \mathbf{U}$, from which one could compute the Essential matrix. Our results show that this inductive bias helps learned pose estimation. We also note that some of the EM Module's output (the upper $D \times D$ submatrix) are image features that cannot be used in the 8-Point algorithm.

How closely does EMM's $\sim \mathbf{U}^\top \mathbf{U}$ match true $\mathbf{U}^\top \mathbf{U}$? We save EMM $\sim \mathbf{U}^\top \mathbf{U}$ output for the Matterport test set and compare to true $\mathbf{U}^\top \mathbf{U}$, computed by projecting 100k

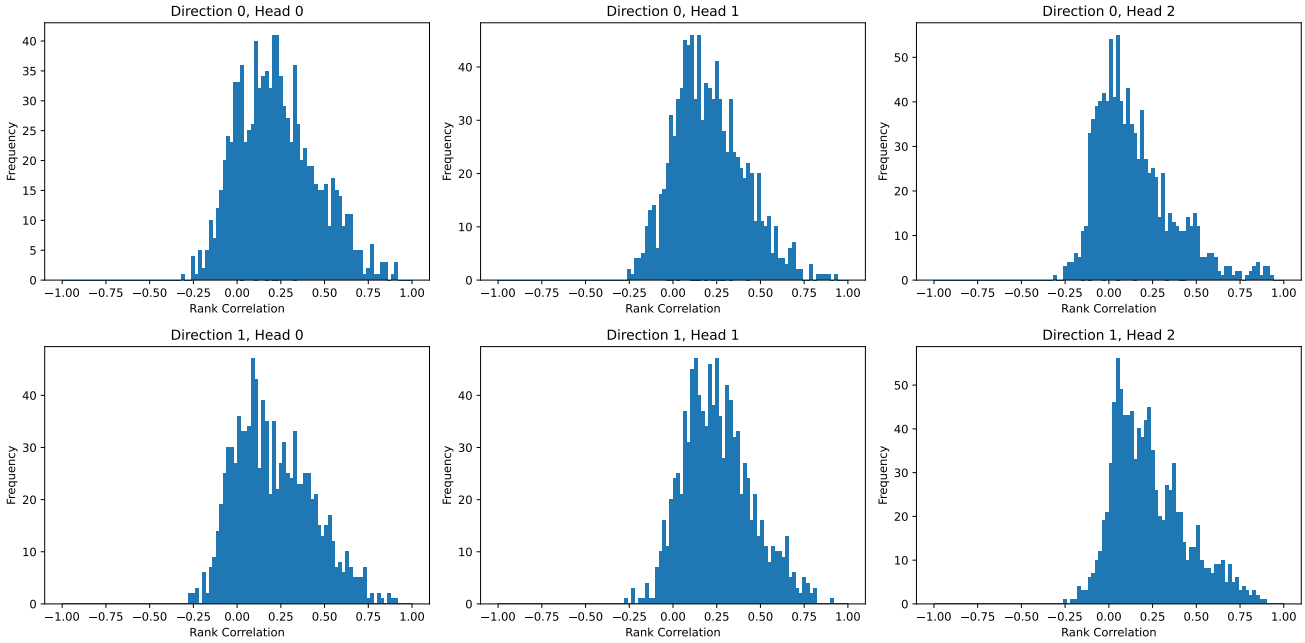

Figure 3. Distribution of rank correlations between ground-truth $\mathbf{U}^\top\mathbf{U}$ and the $\mathbf{U}^\top\mathbf{U}$ as computed by the transformer. We re-arrange the 6×6 bottom sub-matrix to be 9×9 and then compute the Spearman rank correlation per-image (which ranges from -1 to 1). We plot histograms of these rank correlations for each direction (i.e., image 1 or image 2) and each of the three transformer heads. Even though the matrices computed by the transformers are not used in the same way as in the 8-point algorithm, we find substantial rank correlation between the two.

points in a 10x10x10 box into both images using ground truth pose. Note the EMM computes cross-attention in both img1 $\rightarrow$ img2 and img2 $\rightarrow$ img1, and has attention 3 heads for each direction. We therefore have $6 \sim \mathbf{U}^\top\mathbf{U}$ to compare to the ground truth. We measure rank correlation between entries of $\sim \mathbf{U}^\top\mathbf{U}$ and $\mathbf{U}^\top\mathbf{U}$.

As we see in Figure 3 (total correlation values), there is a nontrivial rank correlation between EMM $\sim \mathbf{U}^\top\mathbf{U}$ and the true $\mathbf{U}^\top\mathbf{U}$. This correlation varies some on a per-head and per-direction basis, which is not surprising given directions and heads are concatenated before pose regression, and may be used for differing purposes and extents. Beyond concatenating heads, one important reason correlation is not higher is that $\sim \mathbf{U}^\top\mathbf{U}$ is followed by linear projection and normalization layers before pose regression. Since there is no constraint on $\sim \mathbf{U}^\top\mathbf{U}$ to match the true $\mathbf{U}^\top\mathbf{U}$, the end-to-end system is instead encouraged to learn pose optimally, with this structure used as an inductive bias. Nevertheless, the positive correlation between $\sim \mathbf{U}^\top\mathbf{U}$ and $\mathbf{U}^\top\mathbf{U}$ is consistent with our expectations the EMM is in fact computing similar positional structure to the eight point algorithm.

We visualize $\sim \mathbf{U}^\top\mathbf{U}$ output of individual image pairs in Figure 3. Here we see clear examples of the varying, but mostly positive and meaningful, correlation between actual $\sim \mathbf{U}^\top\mathbf{U}$ and output from different heads and directions. In addition, comparison of individual outputs in the $9x9 \sim \mathbf{U}^\top\mathbf{U}$ tends to show similar cells of interest across heads, regardless of actual correlation. We emphasize we should not expect $\sim \mathbf{U}^\top\mathbf{U}$ to precisely match $\mathbf{U}^\top\mathbf{U}$.

Why not use 8-Point Coefficients as $\mathbf{\Phi}$? Using 8-point coefficients as $\mathbf{\Phi}$ (Eqn. 10) actually leads to the same attention output entries as ours (Eqn. 8), only ours has the advantage of being more compact, as it doesn't duplicate entries. This is detailed in Sec. 7.

How is Scale Ambiguity Handled? The model is trained end-to-end using translation loss with scale, leaving the network free to learn to overcome the scale ambiguity via recognition. We believe this is due to a mix of a distribution over likely relative poses inside scenes as well as familiar objects. Essential Matrix Module output mixes learned appearance features $\mathbf{V}$ (which can encode familiar objects), positional encodings $\mathbf{\Phi}$, and their interaction. Since $\mathbf{\Phi}$ uses intrinsics (L476), features can model varying cameras.

Additional Qualitative Results: Matterport. In Figure 5, we see the proposed method generates Epipolar lines generally similar to true view changes. This is consistent with random rotation and translation errors in Figure 6, which are generally small. Not surprisingly, random results in this challenging setting are also sometimes poor, e.g. the top left Epipolar example, or the top left rotation error example. Sorting by error in Figure 7, we see the general trend

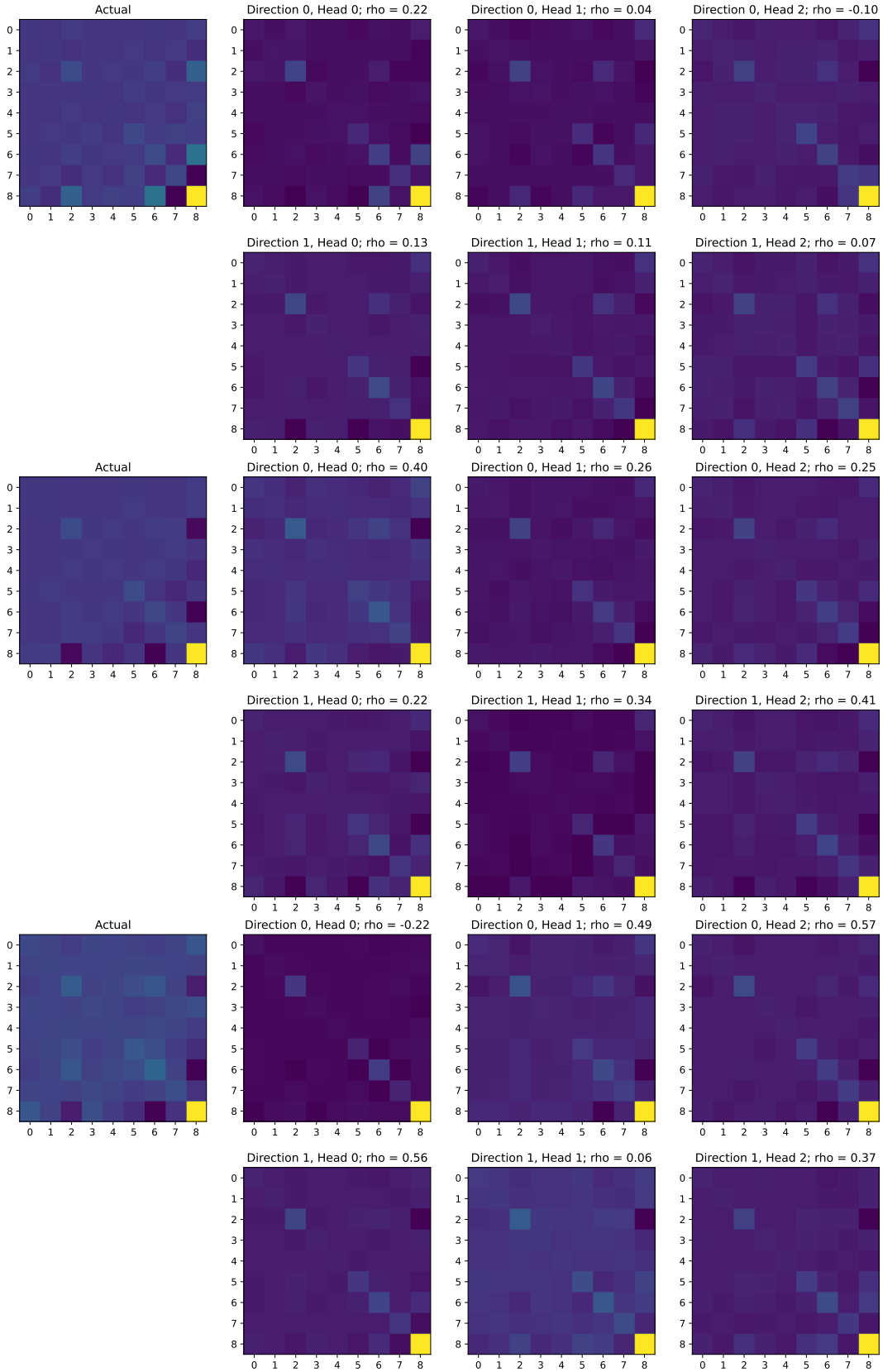

Figure 4. Selected examples of the actual and transformer $\mathbf{U}^\top\mathbf{U}$ matrices for three examples. Top left: actual $\mathbf{U}^\top\mathbf{U}$. Each row shows a different direction (i.e., image 1 or image 2) and each column shows a different transformer head (i.e., one of the three heads). The transformer matrix is actually 6×6, but we re-arrange it to be the 9×9 $\mathbf{U}^\top\mathbf{U}$ matrix.

of increased view change being associated with increased error, which is consistent with our Error against GT study.

Additional Qualitative Results: InteriorNet and StreetLearn. In Figure 8, we see random rotation and translation errors, which are generally quite small. Random results in these datasets are not often poor, but have some weaker examples – e.g. the bottom row of StreetLearn-T has most errors above 1°. Sorting by error in Figure 9, we again see the trend of increased view change being associated with increased error. Yet, results are often quite good even in very high rotations (e.g. InteriorNet all examples).

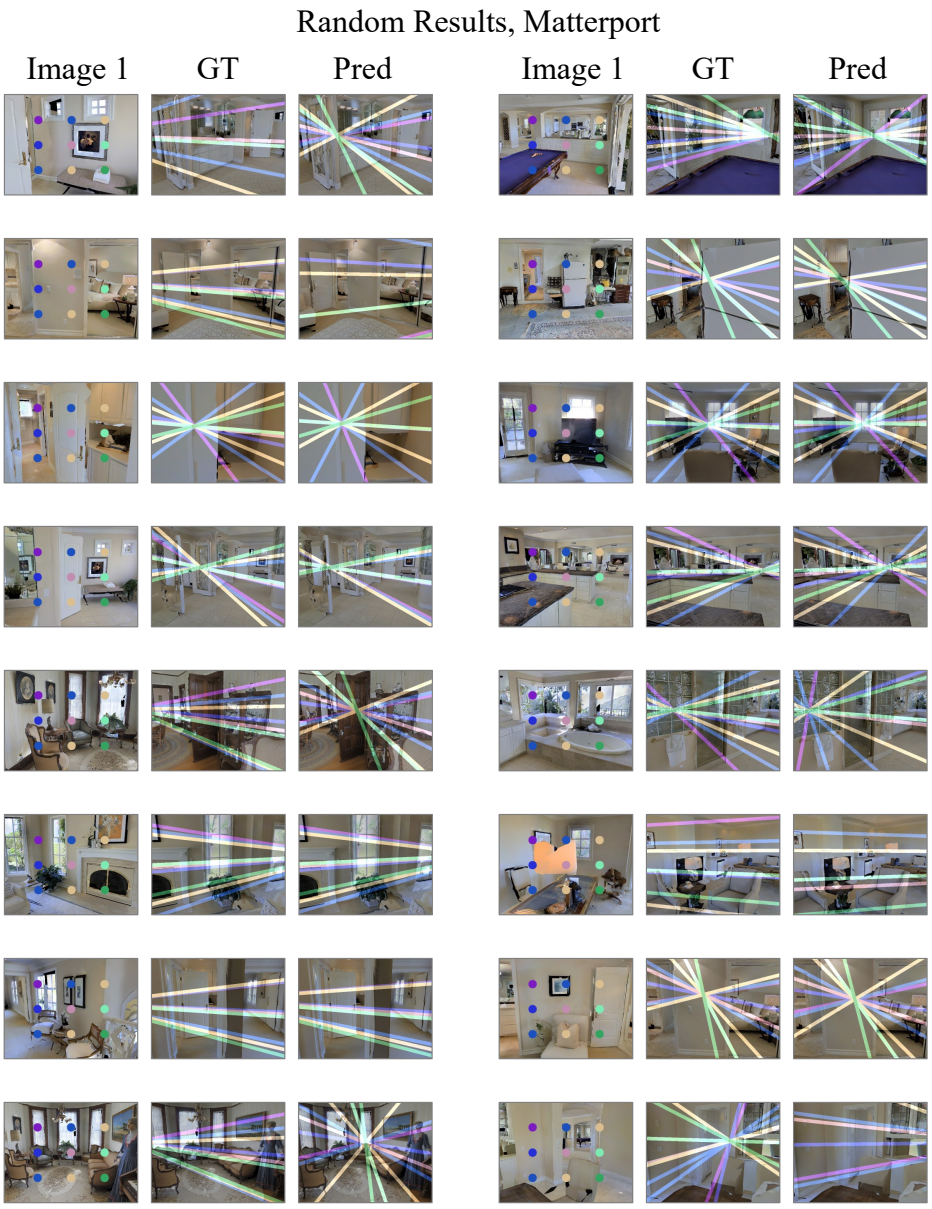

Figure 5. **Random Results on Matterport: Epipolar Lines.**

Figure 6. **Random Results on Matterport.**

Figure 7. **Results by error on Matterport.**

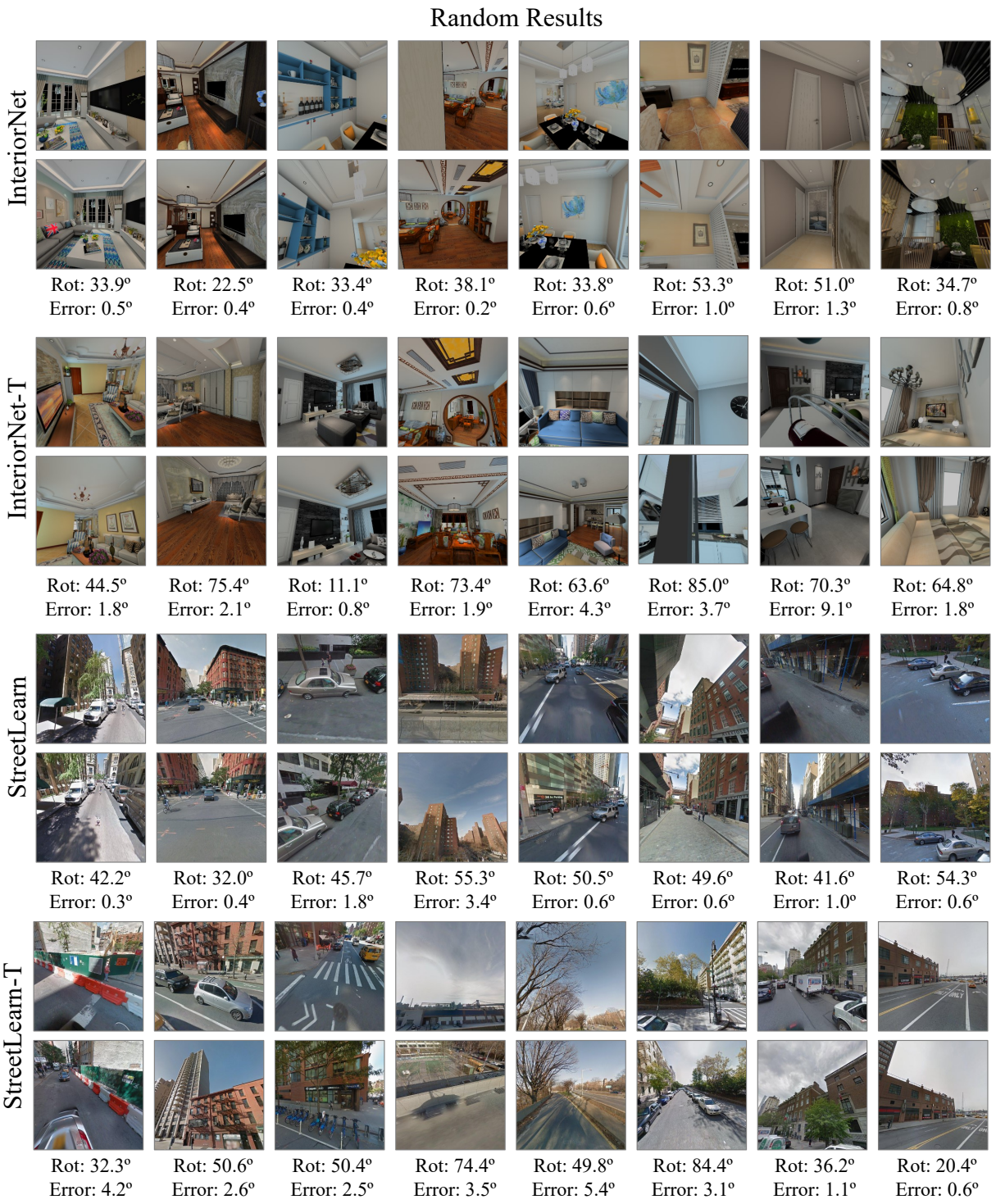

Figure 8. **Random Results on InteriorNet and StreetLearn.**

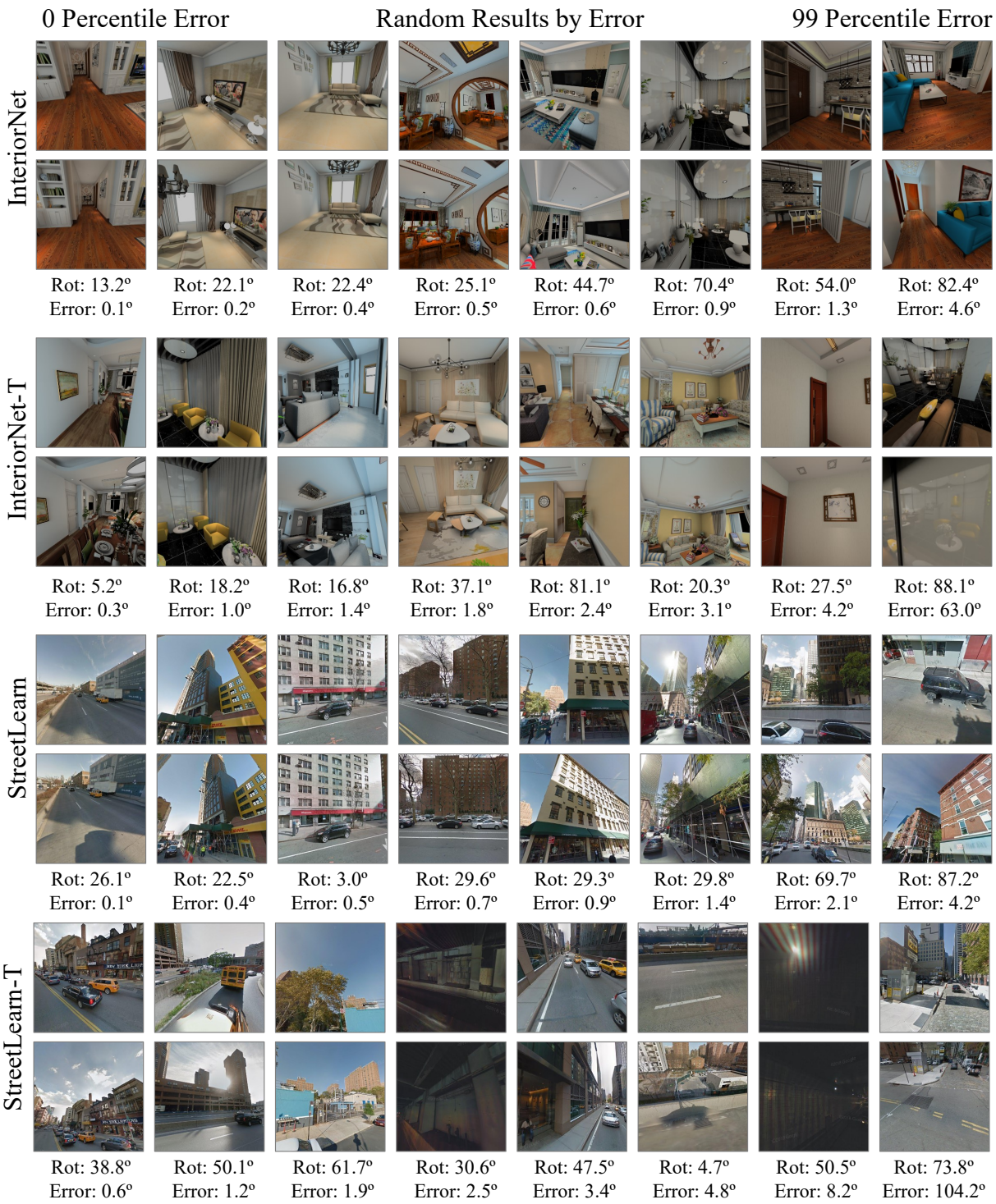

Figure 9. **Results by error on InteriorNet and StreetLearn.**

4. Derivation of the Unique Entries of $\mathbf{U}^\top\mathbf{U}$

Our goal is to show that the unique entries of $\mathbf{U}^\top\mathbf{U}$ that is used in the Eight Point Algorithm can be computed as $\mathbf{\Phi}^\top\mathbf{A}\mathbf{\Phi}$ for an attention matrix $\mathbf{A} \in \{0,1\}^{P\times P}$ and $\mathbf{\Phi} \in \mathbb{R}^{P\times 6}$ as defined in the main paper.

Setup. Given N correspondences, the eight-point algorithm constructs a matrix $\mathbf{U} \in \mathbb{R}^{N\times 9}$ row-wise via the Kronecker products of the homogeneous coordinates of the correspondences involved. Define $\mathbf{x}_i = [u_i, v_i, 1]$ and $\mathbf{x}' = [u'_i, v'_i, 1]$. Then the ith row of $\mathbf{U}$ is

$$\mathbf{U}_{i,:} = \begin{bmatrix} u_iu'_i & u_iv'_i & u_i & v_iu'_i & v_iv'_i & v_i & u'_i & v'_i & 1 \end{bmatrix} \tag{1}$$

or more compactly,

$$\mathbf{U}_{i,:} = (\mathbf{x}_i \otimes \mathbf{x}'_i)^\top. \tag{2}$$

Note that when estimating the Essential matrix, one uses $\mathbf{x}_i \equiv \mathbf{K}^{-1}[u_i, v_i, 1]^\top$ and $\mathbf{x}'_i \equiv \mathbf{K}^{-1}[u'_i, v'_i, 1]^\top$. Since these coordinates can be rescaled by any arbitrary non-zero scalar, we assume that the last coordinate is 1.

Usual approach. Given correct correspondences, the eigenvector of $\mathbf{U}^\top\mathbf{U} \in \mathbb{R}^{9\times 9}$ that corresponds to the smallest eigenvector is the Essential or Fundamental matrix. Usually, the matrix is not rank-deficient, and so one reshapes the eigenvector and then performs rank-reduction.

Alternate approach. We will now show that the unique entries of $\mathbf{U}^\top\mathbf{U}$ can be computed in an alternate fashion using a setup that is amenable to computation via a transformer. We'll start with the following basic substitutions and cleaning up:

$$\mathbf{U}^\top\mathbf{U} = \sum_{i=1}^{N} \mathbf{U}_{i,:}^\top\mathbf{U}_{i,:} = \sum_{i=1}^{N} (\mathbf{x}_i \otimes \mathbf{x}'_i)(\mathbf{x}_i \otimes \mathbf{x}'_i)^\top. \tag{3}$$

We'll first rewrite the interior of the sum, and then the sum itself.

Rewriting $\mathbf{U}_{i,:}^\top\mathbf{U}_{i,:}$ with a basis expansion. We'll tackle the interior of the sum first. While $\mathbf{U}_{i,:}^\top\mathbf{U}_{i,:} = (\mathbf{x}_i \otimes \mathbf{x}'_i)(\mathbf{x}_i \otimes \mathbf{x}'_i)^\top \in \mathbb{R}^{9\times 9}$ and thus has 81 entries, there are only 36 unique entries. The smaller number of entries can be seen mechanically via direct expansion (see §7 to see this). It can also be reasoned out by distributing transposes and using the mixed product property to rewrite it as

$$(\mathbf{x}_i \otimes \mathbf{x}'_i)(\mathbf{x}_i \otimes \mathbf{x}'_i)^\top = (\mathbf{x}_i\mathbf{x}_i^\top) \otimes (\mathbf{x}'_i{\mathbf{x}'_i}^\top). \tag{4}$$

Note that while $\mathbf{x}_i\mathbf{x}_i^\top$ has 9 entries, it only has 6 *unique* entries (1, u, v, uv, u^2, v^2). Likewise, $\mathbf{x}'_i{\mathbf{x}'_i}^\top$ has 6 unique entries (1, u', v', $u'v'$, u'^2, v'^2). Therefore, their Kronecker product $(\mathbf{x}_i\mathbf{x}_i^\top) \otimes (\mathbf{x}'_i{\mathbf{x}'_i}^\top)$ has only 36 unique entries.

We can create a 6×6 matrix containing the unique entries of $\mathbf{U}_{i,:}^\top\mathbf{U}_{i,:}$ by applying a basis expansion to the coordinates. Let us define $\phi([u, v, 1]) = [1, u, v, uv, u^2, v^2]$. Then the unique entries of $\mathbf{U}_{i,:}^\top\mathbf{U}_{i,:} \in \mathbb{R}^{9\times 9}$ can be written as $\phi(\mathbf{x}_i)\phi(\mathbf{x}'_i)^\top \in \mathbb{R}^{6\times 6}$. This means that the unique entries of $\mathbf{U}^\top\mathbf{U}$ are given by

$$\sum_{i=1}^{N} \phi(\mathbf{x}_i)\phi(\mathbf{x}'_i)^\top. \tag{5}$$

As an additional benefit, this factorization separates the terms involving each image into separate components.

Making the sum implicit. We next rewrite the sum implicitly by assuming each correspondence lies on a fixed grid. Given a grid of P patches in each image, we assume that $\mathbf{p}_j$ is the jth patch's location. Then, rather than have N correspondences, we can define the correspondences implicitly via an indicator matrix $\mathbf{A} \in \{0,1\}^{P\times P}$ such that $\mathbf{A}_{j,k} = 1$ if and only if points $\mathbf{p}_k$ and $\mathbf{p}_j$ are in correspondence and 0 otherwise. If each correspondence is on each patch, then we can rewrite

$$\sum_{i=1}^{N} \phi(\mathbf{x}_i)\phi(\mathbf{x}'_i)^\top = \sum_{j=1}^{P}\sum_{k=1}^{P} \phi(\mathbf{p}_k)\mathbf{A}_{j,k}\phi(\mathbf{p}'_j)^\top. \tag{6}$$

This can be further simplified by gathering the basis expanded coordinates of the grid in a matrix $\mathbf{\Phi} \in \mathbb{R}^{P\times 6}$ such that $\mathbf{\Phi}_{j,:} = \phi(\mathbf{p}_j)^\top$. Then $\phi(\mathbf{p}_k)\mathbf{A}_{j,k}\phi(\mathbf{p}'_j)^\top = \mathbf{\Phi}_{k,:}^\top\mathbf{A}_{j,k}\mathbf{\Phi}_{j,:}$, and so Equation 6 can be rewritten as

$$\sum_{j=1}^{P}\sum_{k=1}^{P} \mathbf{\Phi}_{k,:}^\top\mathbf{A}_{j,k}\mathbf{\Phi}_{j,:} = \mathbf{\Phi}^\top\mathbf{A}\mathbf{\Phi}, \tag{7}$$

and therefore the unique entries of $\mathbf{U}^\top\mathbf{U}$ can be compactly written as $\mathbf{\Phi}^\top\mathbf{A}\mathbf{\Phi}$.

5. Discussion of Limitations

The $\mathbf{\Phi}^\top\mathbf{A}\mathbf{\Phi}$ expression is exact when: (1) every correspondence can be represented as one of the P patches; and (2) the attention matrix $\mathbf{A}$ produced by the ViT represents correspondence and is binary. Without an explicit binarization of attention and infinitely small patches, the Essential Matrix Module can, at best, compute an approximation. We now discuss how close this approximation can get.

The closeness of these approximation depends in part on the network architecture and field of view. Throughout, we use patches that are arrayed in a 24×24 grid.

Representing Each Correspondence as a Patch. We replace each correspondence with its equivalent patch, effectively quantizing the correspondence locations. With patches that are the size of pixels, this has close to no impact on the accuracy of estimating pose; if one represents the image with a handful of patches, this clearly ought to have a large impact on the accuracy. We now analyze the impact empirically.

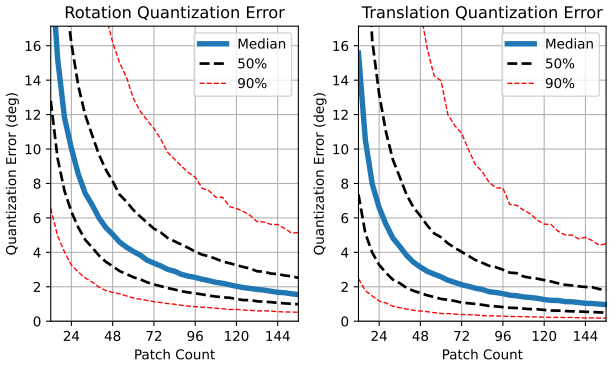

Figure 10. Correspondence quantization error as a function of the number of patches for rotation and translation. We use a patch count of 24×24, which has a moderate quantization error. However, since transformers can contain sub-patch information implicitly, this quantization may be considerably lower.

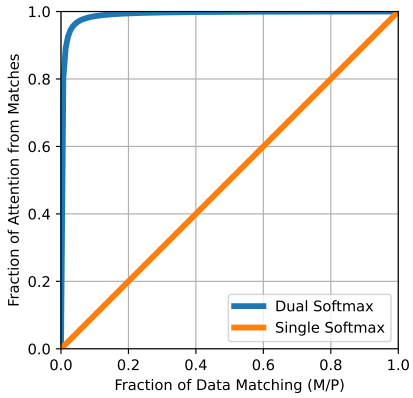

Figure 11. Fraction of attention contributed by matches as a fraction of prevalence of matches. Dual softmax enables matches to rapidly dominate the attention matrix's entries.

Defining Quantization Error. For a given quantization level q (i.e., number of patches that uniformly divide the image along each axis), we generated 10,000 instances of synthetic correspondence by: sampling a relative camera pose with uniform Euler angles, and translation $\sim$Unif$(-1, 1)$, as well as a set of 3D points $\sim$Unif$(-1, 1)$. We render these points to the images using the Matterport3D intrinsics producing a set of correspondences $(x_i, y_i) \leftrightarrow (x'_i, y'_i)$. We then compute the relative pose two ways: first, we do this with the original correspondences, yielding $\mathbf{R}_o$ and $\mathbf{t}_o$; second, we do it with the correspondences uniformly quantized to q levels, which yields estimates $\mathbf{R}_q$ and $\mathbf{t}_q$. We then define the quantization error as the rotation geodesic between $\mathbf{R}_o$ and $\mathbf{R}_q$ as well as the angle between $\mathbf{t}_o$ and $\mathbf{t}_q$.

We then plot the median quantization error per quantization level, along with 50% and 90% intervals in Figure 10. The quantization error rapidly decreases as patch size increases. We use a patch count of 24×24 in this work, which corresponds to a moderate quantization error (d$(\mathbf{R}_o, \mathbf{R}_q) \approx 10°$, d$(\mathbf{t}_o, \mathbf{t}_q) \approx 7°$). Transformer tokens can represent sub-patch information, and once the patch count reaches 96×96, the errors become quite small (d$(\mathbf{R}_o, \mathbf{R}_q) \approx 2.5°$, d$(\mathbf{t}_o, \mathbf{t}_q) \approx 1.6°$).

Producing A. We now discuss how closely a transformer can its attention $\mathbf{A} = \text{norm}(\mathbf{QK}^\top)$ match the binarized matrix that our setup uses. We divide this into two cases: patches that have correspondence and patches without correspondence. We refer to the total contribution as the total size of the weights for a patch j, or $\sum_{k=1}^{P} \mathbf{A}_{j,k}$.

Patches with correspondence. If patch j has a correspondence with patch k, then we would like $\mathbf{A}_{j,k} = 1$ and $\mathbf{A}_{j,k'} = 0$ for all $k' \neq k$. Standard attention cannot exactly reach this, but can get arbitrarily close by making its dot product $(\mathbf{QK}^\top)_{j,k}$ as high as possible. Thus the total contribution of a matching patch j $\sum_{k=1}^{P} \mathbf{A}_{j,k} \approx 1$.

Patches without correspondence. If patch j has no correspondence, then we would like $\mathbf{A}_{j,k'} = 0$ for all k'. This is impossible under standard attention. We can get this to be as close to 0 as possible, by the following: make $(\mathbf{QK}^\top)_{j,:}$ and $(\mathbf{QK}^\top)_{:,j}$ all equal, which in turn makes the resulting softmax distributions uniform. In turn this makes softmax$(\mathbf{QK}^\top)_{j,k'}$ = softmax$(\mathbf{QK}^\top)_{k',j} = 1/P$. Under dual-softmax, then $\mathbf{A}_{j,k'} = 1/P^2$ for all k'. Thus, the total contribution of a patch j is $\sum_{k'=1}^{P} 1/P^2 = 1/P$.

Together, this means that with dual softmax, the vast majority of the attention matrix's energy comes from matches. We do a simple experiment, assuming that $(\mathbf{QK}^\top)_{j,k} = 100$ if patches j and k match and 1 otherwise. We can quantify the fraction of the attention that is from matches by examining the fraction of the resulting matrix $\mathbf{A} = \text{norm}(\mathbf{QK}^\top)$ that corresponds to matches. We plot this as a function of the prevalence of matches in Fig. 11, comparing regular and dual softmax. With dual softmax, a handful of matches dominate the attention, whereas for regular softmax, attention increases linearly.

6. Synthetic Experimental Details

Datasets. For each dataset, we generate an instance consisting of a scene and relative camera pose. These can be used to derive features that are suitable for training. *Scenes:* Each scene consists of a points drawn uniformly inside a sphere with center $\mathbf{c}$ with its coordinates independently and identically sampled from Unif$(-\frac{1}{2}, \frac{1}{2})$ and radius $r \sim$Unif$(\frac{1}{2}, \frac{3}{2})$. This sampling is done with rejection sampling. *Relative Camera Pose:* We generate Euler angles $(\theta_x, \theta_y, \theta_z)$ for the three axes, and a translation vector $\mathbf{t}$. In all cases, we reject samples with $||\mathbf{t}|| \leq \frac{1}{2}$.

1. **3D** (General 3D Motion): $\theta_x, \theta_y, \theta_z \sim$ Unif$(0, 360°)$;

$\mathbf{t}_x, \mathbf{t}_y, \mathbf{t}_z \sim \text{Unif}(-1, 1)$.

2. **2D Large** (Large-Rotation Motion In XZ plane): $\theta_y \sim \text{Normal}(0, 25^\circ)$, $\theta_x, \theta_z \sim \text{Normal}(0, 1.25^\circ)$; $\mathbf{t}_x, \mathbf{t}_z \sim \text{Normal}(0, \frac{1}{3})$, $\mathbf{t}_y \sim \text{Normal}(0, \frac{1}{60})$.

3. **2D Medium** (Medium-Rotation Motion In XZ plane): $\theta_y \sim \text{Normal}(0, 5^\circ)$, $\theta_x, \theta_z \sim \text{Normal}(0, 0.25^\circ)$; $\mathbf{t}_x, \mathbf{t}_z \sim \text{Normal}(0, \frac{1}{3})$, $\mathbf{t}_y \sim \text{Normal}(0, \frac{1}{60})$.

4. **2D Small** (Small-Rotation Motion In XZ plane): $\theta_y \sim \text{Normal}(0, 1^\circ)$, $\theta_x, \theta_z \sim \text{Normal}(0, 0.05^\circ)$; $\mathbf{t}_x, \mathbf{t}_z \sim \text{Normal}(0, \frac{1}{3})$, $\mathbf{t}_y \sim \text{Normal}(0, \frac{1}{60})$.

Given a scene and relative camera pose, we project the points onto a virtual camera with height and width 800 units, a focal length of 800 units and principal point of 400 units. We record the point if it is in front of the camera, and on the virtual sensor. We reject the image pair and scene if fewer than 100 points out of 10K random points are valid for both cameras.

Input Feature. Given a 3D point, we denote its projection into image 1 as $\mathbf{x}$ and its projection into image 2 as $\mathbf{x}'$. Given the set of valid points, we compute the explicit form of $\mathbf{U}^\top\mathbf{U}$ with two modifications. First, for numerical stability we divide each coordinate by the width of the image and subtract $1/2$, which centers the data. Second, to make the feature independent of the number of correspondences, we normalize by the number of points to obtain $\frac{1}{N}\mathbf{U}^\top\mathbf{U}$ rather than $\mathbf{U}^\top\mathbf{U}$. These are identical from the perspective of eigenvectors, but normalizing makes the feature independent of the number of points.

Method. For each task, we train a multilayer perceptron consisting of 3 hidden layers with 4096 units each. Each hidden layer is capped with a leaky ReLU. We predict a normalized vector (3D for translation direction, 4D for rotation).

7. Verifying that $\phi(\mathbf{x})\phi(\mathbf{x}')$ contains all the terms needed for $\mathbf{U}^\top\mathbf{U}$

To enable visually verifying Equation 5, we'll show a visual expansion. To avoid notational clutter, we will drop the ith index and deal with $\mathbf{x} = [u, v, 1]$ and $\mathbf{x}' = [u', v', 1]$. Our goal is to show that $(\mathbf{x} \otimes \mathbf{x}')^\top(\mathbf{x} \otimes \mathbf{x}')$ has the same entries as $\phi(\mathbf{x})\phi(\mathbf{x}')^\top$. We'll first expand out $\phi(\mathbf{x})\phi(\mathbf{x}')^\top$. When factored out,

$$\phi(\mathbf{x})\phi(\mathbf{x}')^\top = \begin{bmatrix} 1 & u' & v' & u'v' & u'^2 & v'^2 \\ u & uu' & uv' & uu'v' & uu'^2 & uv'^2 \\ v & vu' & vv' & vu'v' & vu'^2 & vv'^2 \\ uv & uvu' & uvv' & uvu'v' & uvu'^2 & uvv'^2 \\ u^2 & u^2u' & u^2v' & u^2u'v' & u^2u'^2 & u^2v'^2 \\ v^2 & v^2u' & v^2v' & v^2u'v' & v^2u'^2 & v^2v'^2 \end{bmatrix}. \tag{8}$$

We color the terms according to which row they appear in $\phi(\mathbf{x})\phi(\mathbf{x}')^\top$. The matrix created for each correspondence is $(\mathbf{x} \otimes \mathbf{x}')^\top(\mathbf{x} \otimes \mathbf{x}')$. We'll first define $(\mathbf{x} \otimes \mathbf{x}')$:

$$(\mathbf{x} \otimes \mathbf{x}') = \begin{bmatrix} uu' & uv' & u & vu' & vv' & v & u' & v' & 1 \end{bmatrix}. \tag{9}$$

We can then compute the outer product $(\mathbf{x} \otimes \mathbf{x}')^\top(\mathbf{x} \otimes \mathbf{x}')$. This is highly redundant – note that the ith row and ith column are identical. More specifically,

$$(\mathbf{x} \otimes \mathbf{x}')^\top(\mathbf{x} \otimes \mathbf{x}') = \begin{bmatrix} u^2u'^2 & u^2u'v' & u^2u' & uvu'^2 & uvu'v' & uvu' & uu'^2 & uu'v' & uu' \\ u^2u'v' & u^2v'^2 & u^2v' & uvu'v' & uvv'^2 & uvv' & uu'v' & uv'^2 & uv' \\ u^2u' & u^2v' & u^2 & uvu' & uvv' & uv & uu' & uv' & u \\ uvu'^2 & uvu'v' & uvu' & v^2u'^2 & v^2u'v' & v^2u' & vu'^2 & vu'v' & vu' \\ uvu'v' & uvv'^2 & uvv' & v^2u'v' & v^2v'^2 & v^2v' & vu'v' & vv'^2 & vv' \\ uvu' & uvv' & uv & v^2u' & v^2v' & v^2 & vu' & vv' & v \\ uu'^2 & uu'v' & uu' & vu'^2 & vu'v' & vu' & u'^2 & u'v' & u' \\ uu'v' & uv'^2 & uv' & vu'v' & vv'^2 & vv' & u'v' & v'^2 & v' \\ uu' & uv' & u & vu' & vv' & v & u' & v' & 1 \end{bmatrix} \tag{10}$$

Note that the rows of $\phi(\mathbf{x})\phi(\mathbf{x}')^\top$ appear in 3×3 blocks inside $(\mathbf{x} \otimes \mathbf{x}')^\top(\mathbf{x} \otimes \mathbf{x}')$. In particular, inside each 3×3 block, the columns of the $\phi(\mathbf{x})\phi(\mathbf{x}')^\top$ appear in the following order:

$$\begin{bmatrix} 5 & 4 & 2 \\ 4 & 6 & 3 \\ 2 & 3 & 1 \end{bmatrix}. \tag{11}$$

References

[1] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *CVPR*, 2018. 4

[2] Wenshan Wang, Yaoyu Hu, and Sebastian Scherer. Tartanvo: A generalizable learning-based vo. In *CoRL*, 2021. 4