ArchSym: Detecting 3D-Grounded Architectural Symmetries in the Wild

Supplementary Material

1. Implementation details

Our implementation builds upon the official MASt3R [1] and VGGT [10] codebases.

1.1. Training details

We use a base learning rate of 1e-4 and a cosine decay learning rate schedule, with an effective batch size of 48. For data augmentation, we perform random center crop, random horizontal flip, and color jitter. Training is performed on 4 A6000 GPUs for 2 days.

1.2. Network architecture

Our model consists of the pre-trained VGGT [10] model and our symmetry prediction head. The VGGT backbone and point prediction head are frozen with weights from the officially released checkpoints. We find no significant difference in performance between using the point head and using the depth and camera heads for point map prediction.

The architecture of our symmetry prediction head is similar to the VGGT point prediction head, with the only difference being the additional FiLM conditioning [4] before each fusion block. We use eight instance queries to identify up to eight reflectional symmetries. They are passed through a three-layer transformer decoder with an embedding dimension of 256. The instance queries attend to the final layer features of the VGGT backbone, which are projected to the same embedding dimension. The refined instance queries are passed through a two-layer MLP to obtain four pairs of FiLM conditioning parameters for each of the four fusion blocks.

Classification logits are predicted by a two-layer MLP that takes in the final upsampled feature maps after global average pooling. We find no significant difference in performance between regressing classification logits directly from the refined instance queries and from the final upsampled feature maps.

2. Evaluation details

Detailed per-scene evaluation results on 19 test scenes are reported in Table 1. For each scene, we report the *median* geodesic distance and dense symmetry error, which are robust against outlier images within a scene (e.g., images with extreme viewpoints or severe occlusions). The mean statistics are then computed as the average across all per-scene statistics.

2.1. Visibility filtering heuristic

Although our dataset curation pipeline identifies ground truth symmetries at the scene level, for training and evaluation purposes, it is necessary to determine which of these symmetries are actually visible in each image. A symmetry plane is considered visible if the image captures sufficient geometric structure (e.g. 5% of pixels with valid depth) on both sides of the plane. Additionally, we filter out images where the landmark structure is not the primary subject to ensure a clean training signal. The details of our filtering heuristic are presented in Algorithm 1.

2.2. Asymmetric F-score calculation

We present our F-score calculation scheme in Algorithm 2, which is based on the evaluation code from REFLECT3D [2]. We modify the code to take into account our asymmetric evaluation scheme—all ground truth planes are used in evaluating *exactness*, but only visible ground truth planes are used in evaluating *completeness*. In particular, bipartite matching is performed between predicted planes and *all* ground truth planes instead of only the visible ground truth planes. Predicted planes that are within the angular error threshold of some non-visible ground truth plane are *not* considered as false positives during F-score calculation,

Algorithm 1 Visibility filtering heuristic

Require: Depth map D, camera parameters $(\boldsymbol{K}, [\boldsymbol{R} \mid \boldsymbol{t}])$, plane parameters $\pi = (\boldsymbol{n}, d)$
Ensure: Boolean indicating if the plane π is visible.

Crop D to its central 80% region, yielding D_c.
Let $\mathcal{X}_c$ be the set of pixel coordinates with valid depth.
$N_{\text{valid}} \leftarrow |\mathcal{X}_c|$
if $N_{\text{valid}} < 1000$ **then** ▷ not enough valid pixels
  return False
end if
Initialize $N_{\text{pos}} \leftarrow 0$, $N_{\text{neg}} \leftarrow 0$
for all pixel $x_k \in \mathcal{X}_c$ **do**
  $\boldsymbol{p}_k \leftarrow \mathsf{unproject}(D_c(x_k), \boldsymbol{K}, [\boldsymbol{R} \mid \boldsymbol{t}])$.
  $s_k \leftarrow \boldsymbol{n}^\top \boldsymbol{p}_k + d$.
  if $s_k > 0$ **then** ▷ positive signed distance
    $N_{\text{pos}} \leftarrow N_{\text{pos}} + 1$
  else if $s_k < 0$ **then** ▷ negative signed distance
    $N_{\text{neg}} \leftarrow N_{\text{neg}} + 1$
  end if
end for
$\text{prop}_{\text{pos}} \leftarrow N_{\text{pos}} / N_{\text{valid}}$ ▷ filter by proportion
$\text{prop}_{\text{neg}} \leftarrow N_{\text{neg}} / N_{\text{valid}}$
if $\text{prop}_{\text{pos}} < 0.05$ **or** $\text{prop}_{\text{neg}} < 0.05$ **then**
  return False
else
  return True
end if

Scene	Geo ↓			F@1° ↑			F@5° ↑			F@15° ↑			$\mathcal{E}_{\text{dense}}$ ↓	
	R3D	DIR	OURS	R3D	DIR	OURS	R3D	DIR	OURS	R3D	DIR	OURS	DIR	OURS
Arc de Triomphe	8.19	4.29	**1.44**	0.06	0.12	**0.36**	0.34	0.64	**0.79**	0.62	0.89	**0.90**	0.24	**0.12**
Arch of Hadrian	23.17	3.22	**2.00**	0.02	0.13	**0.33**	0.34	0.73	**0.78**	0.54	**0.92**	0.90	0.19	**0.11**
Basilica of Bom Jesus	6.93	1.75	**1.48**	0.06	0.22	**0.31**	0.30	**0.79**	0.77	0.56	0.84	**0.86**	0.07	**0.04**
Bath Abbey	5.30	3.10	**1.93**	0.08	0.18	**0.32**	0.35	0.61	**0.64**	0.51	**0.81**	0.79	0.11	**0.08**
Cathedral of Saint Paul	8.84	**7.54**	7.78	0.01	0.08	**0.08**	0.25	**0.47**	0.37	0.50	0.81	**0.85**	0.25	**0.19**
Charlottenburg Palace	15.43	**1.48**	1.86	0.12	**0.29**	0.17	0.25	**0.88**	0.83	0.34	**0.94**	0.85	**0.05**	**0.05**
Frauenkirche (Dresden)	13.05	12.79	**9.18**	0.01	0.02	**0.06**	0.15	0.28	**0.55**	0.45	0.64	**0.74**	0.38	**0.27**
Gateway of India	8.35	5.70	**4.89**	0.06	0.11	**0.21**	0.35	0.60	**0.61**	0.56	**0.84**	0.79	0.20	**0.17**
Illinois State Capitol	7.67	**1.82**	2.06	0.13	**0.30**	0.27	0.43	0.77	**0.84**	0.58	0.84	**0.92**	0.10	**0.08**
Isa Khan Niyazi's tomb	10.48	**8.24**	8.44	0.03	0.04	**0.05**	0.23	**0.38**	0.36	**0.68**	0.64	0.62	0.29	**0.25**
Montmartre	3.66	1.77	**0.92**	0.16	0.28	**0.52**	0.62	0.80	**0.86**	0.76	0.88	**0.91**	0.07	**0.04**
Notre-Dame Basilica	7.79	2.27	**1.79**	0.10	0.20	**0.31**	0.40	**0.75**	0.73	0.61	0.88	**0.90**	0.08	**0.05**
Panthéon de Paris	7.88	1.89	**1.34**	0.21	0.29	**0.45**	0.50	**0.88**	0.85	0.63	**0.93**	0.91	0.09	**0.05**
Royal Liver Building	8.32	4.99	**2.59**	0.05	0.05	**0.21**	0.34	0.54	**0.73**	0.60	0.85	**0.87**	0.22	**0.11**
Saints Peter and Paul Church	6.38	**1.85**	1.92	0.05	0.27	**0.28**	0.46	**0.88**	0.87	0.66	0.92	**0.96**	0.07	**0.06**
Torre de Belém	14.55	12.54	**7.17**	0.01	0.04	**0.09**	0.16	0.33	**0.64**	0.39	0.65	**0.79**	0.35	**0.20**
Town Hall Tower in Kraków	16.14	14.83	**9.73**	0.01	0.03	**0.12**	0.13	0.24	**0.60**	0.34	0.54	**0.74**	0.44	**0.28**
Victoria Memorial	7.86	**2.20**	2.50	0.07	0.21	**0.24**	0.40	**0.80**	0.63	0.55	**0.89**	0.84	**0.08**	0.10
Westminster Abbey	8.70	1.91	**1.38**	0.07	0.23	**0.37**	0.38	0.78	**0.80**	0.56	0.87	**0.88**	0.09	**0.07**
Mean	10.46	5.06	**3.71**	0.07	0.16	**0.25**	0.34	0.64	**0.70**	0.55	0.81	**0.84**	0.18	**0.13**

Table 1. **Detailed per-scene comparison across three methods.** For each scene, we report the median geodesic distance and dense symmetry error, which are robust against outlier images. REFLECT3D [2] and DIRECT are abbreviated as R3D and DIR. Geo: geodesic distance (↓, degrees). F@x°: F-score at x° threshold (↑). $\mathcal{E}_{\text{dense}}$: dense symmetry error (↓).

whereas in a standard F-score calculation scheme they would be. Intuitively, this means we penalize the model for not predicting visible symmetries, but we do not penalize the model for *accurately* predicting non-visible symmetries when they apparant from other indirect cues.

Algorithm 2 Visibility-aware F-score calculation

Require: Predicted normals $\widehat{\mathcal{N}}$, full GT normals $\mathcal{N}$, visible GT normals $\mathcal{N}_{\text{vis}}$, threshold x°
Ensure: F-score at threshold x°

```
Find optimal matching M between N̂ and N based on geodesic distance.
Initialize tp ← 0, fp ← 0, fn ← 0, nv ← 0
for all pair (n̂_i, n_j) with distance d_ij in M do
    if d_ij < x° then                          ▷ distance within threshold
        if n_j ∈ N_vis then                    ▷ matched to visible GT
            tp ← tp + 1
        else                                   ▷ matched to non-visible GT
            nv ← nv + 1
        end if
    end if
end for
fp ← |N̂| − tp − nv
fn ← |N_vis| − tp
return (2·tp)/(2·tp + fp + fn)
```

3. Ablation studies

We validate our design choices by comparing the full model against two baselines. First, we remove the instance-specific FiLM conditioning [4] (w/o FiLM) to predict multiple symmetry planes directly from shared DPT features [5]. Second, we supervise the model using ground truth signed distance maps (w/ GT) derived from the ground truth geometry instead of the pseudo-ground truth (derived from the model's predicted point maps) that is consistent with the predicted geometry. As shown in Table 2, we observe that both the orientation and alignment of the predicted planes degrade severely in both cases, highlighting the effectiveness of our two-stage model architecture and the importance of self-consistent predictions.

Table 2. **Ablation studies on model architecture and loss supervision.** Plane prediction quality severely degrades if we remove FiLM conditioning [4] (w/o FiLM) or use ground truth signed distance map supervision (w/ GT) that is inconsistent with the predicted geometry.

Method	Normal-only				Full-plane
	Geo ↓	F@1° ↑	F@5° ↑	F@15° ↑	$\mathcal{E}_{\text{dense}}$ ↓
Full	**3.71**	**0.25**	**0.70**	**0.84**	**0.13**
w/o FiLM	6.72	0.16	0.51	0.73	0.20
w/ GT	6.99	0.13	0.48	0.73	0.19

Figure 1. **Generalization to object-centric scenes.** We demonstrate accurate symmetry annotation on real (CO3D [7], left/center) and synthetic (NeRF-Synthetic [3], right) objects. We show sample input images and annotated symmetry planes overlaid on COLMAP MVS [8] point clouds.

4. Additional results

4.1. Generalization to object-centric data

Our paper focuses on architectural scenes since the MegaScenes dataset [9] already provides real-world variability (e.g., illumination, viewing angles) necessary to benchmark this task. However, our symmetry annotation pipeline and signed-distance formulation are general-purpose. As shown in Figure 1, our automated annotation pipeline can correctly produce 3D symmetries when directly applied to non-architectural object-centric scenes. We run our annotation pipeline on three scenes sampled from the CO3D [6] and NeRF-Synthetic [3] datasets. The detected reflectional symmetry planes are overlaid on dense point clouds from COLMAP MVS [8] for visualization.

4.2. Additional qualitative comparisons

Figure 2 presents additional qualitative comparisons on images sampled from 16 test scenes. These examples further illustrate that our signed-distance parameterization allows OURS to consistently predict planes that are better aligned with the underlying scene geometry.

4.3. Per-query predictions

To provide insight into the behavior of our multi-instance detection head, we visualize the symmetry plane predictions associated with each of the eight learnable instance queries in Figure 3. Subplots with highlighted frames correspond to valid symmetry planes (i.e., those with logits above the extraction threshold). We observe that prediction slots corresponding to different instance queries learn to specialize in extracting different types of symmetries. For example, the first slot often detects a front-to-back reflection, while the sixth slot often detects a reflection across the main facade. Notably, this specialization can be observed even when the corresponding symmetry is not present in the specific scene (resulting in suppressed predictions). This highlights the effectiveness of our two-stage architecture and set prediction formulation in handling scenes with varying numbers and types of symmetry planes.

References

[1] Bardienus Duisterhof, Lojze Zust, Philippe Weinzaepfel, Vincent Leroy, Yohann Cabon, and Jerome Revaud. Mast3r-sfm: a fully-integrated solution for unconstrained structure-from-motion. *arXiv preprint*, 2024. 1

[2] Xiang Li, Zixuan Huang, Anh Thai, and James M Rehg. Symmetry strikes back: From single-image symmetry detection to 3d generation. In *CVPR*, 2025. 1, 2, 4

[3] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 3

[4] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *AAAI*, 2018. 1, 2

[5] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *ICCV*, pages 12179–12188, 2021. 2

[6] Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotny. Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10901–10911, 2021. 3

[7] Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotny. Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction. In *International Conference on Computer Vision*, 2021. 3

[8] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016. 3

[9] Joseph Tung, Gene Chou, Ruojin Cai, Guandao Yang, Kai Zhang, Gordon Wetzstein, Bharath Hariharan, and Noah Snavely. Megascenes: Scene-level view synthesis at scale. In *ECCV*, 2024. 3

[10] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *CVPR*, 2025. 1

Figure 2. **Additional qualitative comparisons of single-view symmetry detection results.** Input images are sampled from 16 different test scenes. REFLECT3D [2] often misses partially visible symmetries and produces redundant detections, while DIRECT often predicts planes that are misaligned with the scene geometry. We encourage zooming into the figure to see differences in plane orientation and alignment in detail.

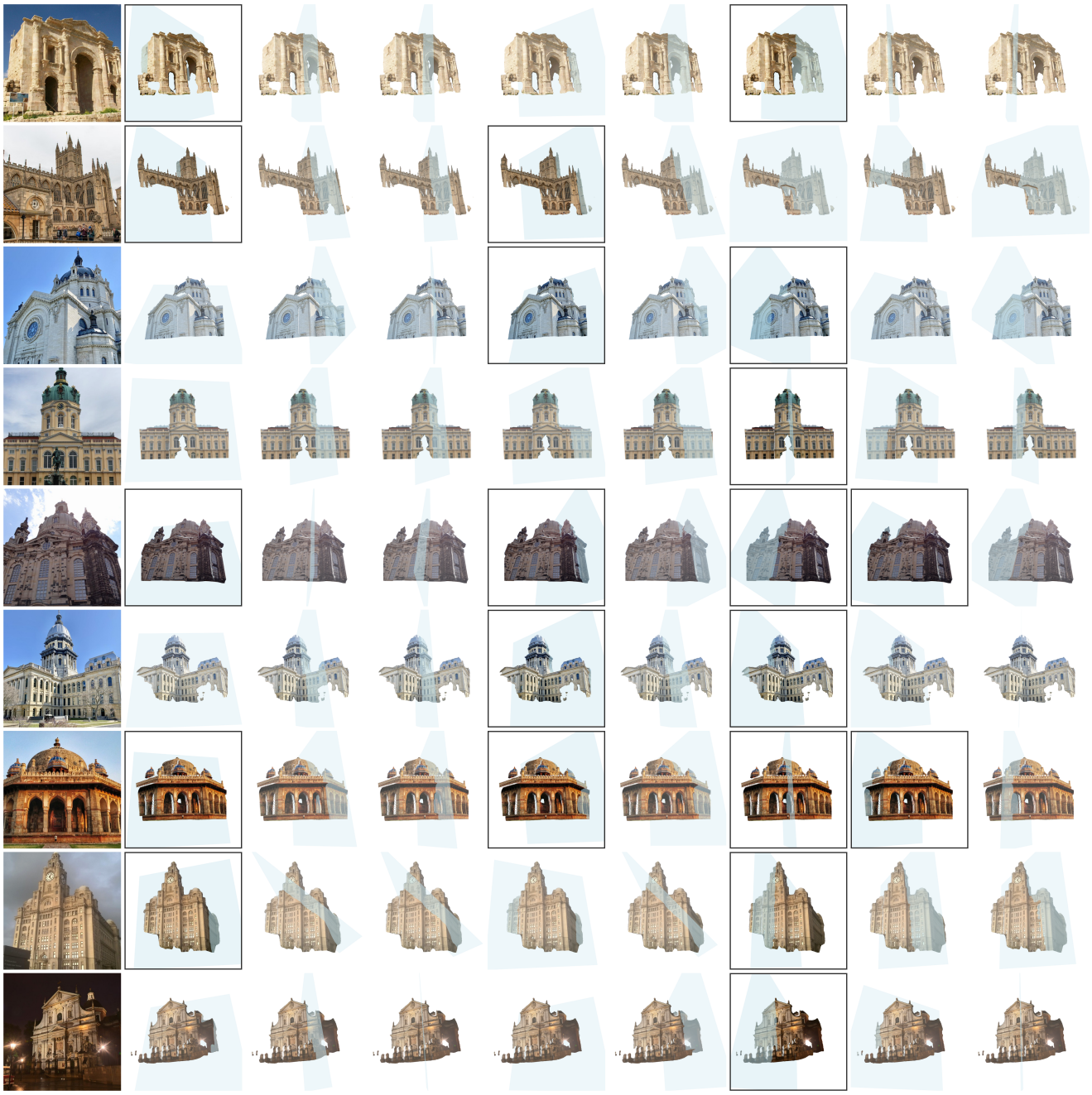

Figure 3. **Visualization of symmetry plane predictions from individual instance queries.** Each row shows an input image alongside the symmetry planes predicted from each of the eight instance queries. Highlighted frames indicate valid planes with predicted logits above the extraction threshold. This visualization demonstrates how different prediction slots specialize to capture specific types of symmetries within the scene.