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

Hanyu Chen[1] Ruojin Cai[2] Steve Marschner[1] Noah Snavely[1]
[1]Cornell University [2] Kempner Institute, Harvard University

Figure 1. **Our method robustly detects 3D-grounded symmetries in challenging, in-the-wild images.** From a single RGB image (left in each pair), our model recovers dominant 3D symmetry planes (right) even when they are partially occluded or not directly visible. To train our model, we introduce a novel pipeline to automatically curate ArchSym, a large-scale dataset of landmark symmetries. The results above are on images from landmarks unseen during training and highlight our model's strong generalization: it performs consistently on structures from different eras (first column); it is robust to extreme viewpoints and varying illumination (second column); and while trained primarily on Western landmarks, it generalizes effectively to diverse architectural styles (third column).

Abstract

Symmetry detection is a fundamental problem in computer vision, and symmetries serve as powerful priors for downstream tasks. However, existing learning-based methods for detecting 3D symmetries from single images have been almost exclusively trained and evaluated on object-centric or synthetic datasets, and thus fail to generalize to real-world scenes. Furthermore, due to the inherent scale ambiguity of monocular inputs, which makes localizing the 3D plane an ill-posed problem, many existing works only predict the plane's orientation. In this paper, we address these limitations by presenting the first framework for detecting 3D-grounded reflectional symmetries *from single, in-the-wild RGB images, focusing on architectural landmarks. We introduce two key innovations: (1) a scalable data annotation pipeline to automatically curate a large-scale dataset of architectural symmetries, ArchSym, from SfM reconstructions by leveraging cross-view image matching; and building on the dataset, (2) a single-view symmetry detector that accurately localizes symmetries in 3D by parameterizing them as signed distance maps defined relative to predicted scene geometry. We validate our symmetry annotation pipeline against geometry-based alternatives and demonstrate that our symmetry detector significantly outperforms state-of-the-art baselines on our new benchmark.*

1. Introduction

Symmetry is a fundamental principle in nature and in human design, and it serves as a powerful prior for various computer vision tasks. Symmetry is particularly useful for reasoning about occluded or partially observed geometry in 3D reconstruction and generation problems [19, 45, 46, 48]. Symmetries also provide a canonical orientation for pose estimation [36, 51] and help resolve ambiguities [23, 50]. Detecting symmetries in visual data is a long-standing problem with foundational work dating back several decades [1, 44]. Early approaches relied on geometric heuristics and handcrafted features to identify symmetric patterns in images or 3D models. While effective in controlled settings, these classical methods struggle with real-world scenarios. Recent learning-based methods [19, 35, 51] have achieved impressive results that generalize better than their handcrafted predecessors.

Despite this progress, existing methods for single-view

3D symmetry detection are almost exclusively trained and evaluated on object-centric datasets, such as ShapeNet [4] and Objaverse [6], which contain clean, pre-segmented objects without complex backgrounds. Consequently, their performance degrades significantly when applied to in-the-wild scenes featuring complex environments, varying illumination, and occlusions, leaving the problem of 3D symmetry detection in real-world environments largely unsolved.

In this work, we present a novel pipeline for predicting scene geometry and 3D reflectional symmetries grounded in the predicted geometry from single images of *in-the-wild* scenes. We focus specifically on architectural scenes, as their man-made designs often contain symmetrical structures. We first introduce a scalable method for curating a dataset of architectural images labeled with 3D reflectional symmetries. Our method is inspired by the "doppelganger" problem in 3D reconstruction [2, 47], where image matchers fail to distinguish physically distinct, but visually similar structures. We leverage this property of feature matchers to automatically annotate symmetries from structure-from-motion (SfM) reconstructions. Building on this data, we introduce a novel single-view 3D symmetry detector that parameterizes reflectional symmetries as *signed distance maps* defined relative to the predicted scene geometry. This parameterization resolves the scale ambiguity inherent to single-view symmetry detection methods, enabling accurate 3D localization of symmetry planes.

The main contributions of our work are as follows:

- We introduce a new scalable pipeline for automatically curating 3D symmetry annotations, yielding ArchSym, a large-scale dataset of in-the-wild landmark symmetries.
- We present the first end-to-end model that detects 3D-grounded symmetries from real-world images by parameterizing them relative to the predicted scene geometry.
- We establish a new benchmark for this task and demonstrate that our model significantly outperforms state-of-the-art methods, even when they are finetuned on ArchSym.

2. Related work

Symmetry detection. Prior work in symmetry detection can be categorized along two primary axes: the input modality (e.g., RGB images, RGB-D scans, or 3D models) and the output space (2D vs. 3D). Here, we review key methods that are most relevant to our work. For comprehensive surveys, we refer the reader to Mitra et al. [25] for classical methods and Funk et al. [11] for learning-based methods.

Many early methods for symmetry detection from images focus on identifying symmetries in front-facing objects, where 3D reflectional planes reduce to 2D lines of symmetry in the image plane [5, 8, 14, 22, 26, 39]. Subsequent work extends this idea to dense heatmaps that highlight centers of rotation and lines of reflection for unconstrained, real-world images [10]. While effective for 2D tasks, these approaches inherently lack the ability to localize symmetries in 3D.

More recently, machine learning has led to significant progress in 3D symmetry detection. Such approaches can be broadly classified by input modality: RGB images [19, 21, 51], RGB-D scans [35–37], or 3D models and point clouds [12, 13, 18, 49]. Most related to our work are NeRD/NeRD++ [21, 51] and Reflect3D [19], which detect reflectional symmetries from single-view RGB images. The former proposes an iterative refinement strategy that reflects image features in 3D to identify symmetry planes; the latter leverages foundation models to regress plane parameters. However, a key limitation of these methods is that they are trained almost exclusively on object-centric data, and their performance degrades significantly in complex, in-the-wild scenes.

Symmetry for 3D reconstruction. Symmetry is a powerful prior for reconstructing 3D geometry, particularly from limited data. Early work like that of Sawada et al. [32] derives geometric conditions under which 2D object contours imply a 3D mirror symmetry to constrain the ill-posed problem of monocular shape recovery. Köser et al. [15] propose detecting a 3D symmetry plane and performing dense reconstruction by matching features between an image and its reflected counterpart. In particular, this idea suggests that image matching can be a useful tool for symmetry extraction.

More recently, symmetry has also been leveraged as a cue for learning-based 3D reconstruction and generation models. Wu et al. [45], for example, use a symmetry prior to disentangle depth, albedo, and viewpoint in single-view 3D reconstruction. Li et al. [19] detect and aggregate symmetry planes from multiple views to guide mesh generation, ensuring global consistency from partial observations.

Symmetry ambiguities in 3D reconstruction. Conversely, symmetry and other forms of structural repetition are often problematic in structure-from-motion (SfM) [33]. The visual similarity between distinct but symmetric parts of a scene leads image matchers to produce incorrect correspondences. These illusory matches, or "doppelgangers" [2, 47], can introduce significant errors in 3D reconstruction, particularly for large-scale architectural scenes where such ambiguities are common. However, this phenomenon also suggests that these incorrect, yet *geometrically consistent*, matches could be a valuable signal for discovering landmark symmetries.

Automated symmetry annotation. Learning-based methods have proven to excel in symmetry detection, but curating a dataset of ground truth 3D symmetry annotations, especially for real-world scenes, is challenging. Prior methods that use synthetic and object-centric data have relied on pre-aligned 3D models [51], iterative closest point algorithms [19], and manual labeling [35, 37]. Several unsupervised [12, 18] and training-free [13] methods have also been proposed to automatically generate symmetry annotations from point clouds. However, these geometry-based methods often struggle with the noisy and incomplete point clouds produced by SfM on in-the-wild scenes. By operating solely on geometry, they also

discard visual information in the original images. This motivates the need for alternative data annotation methods that can robustly identify symmetries in real-world reconstructions.

3. Dataset: ArchSym

The goal of our work is to predict the plane parameters of *global reflectional symmetries* from a single input image. Formally, given an image I, we aim to learn a mapping from the image to a set of symmetry planes, $\Pi = \{\pi_k\}_{k=1}^{N}$, represented in the camera coordinate system. Each plane is defined by a unit normal $\boldsymbol{n} \in S^2$ and its signed distance to the origin, or offset, $d \in \mathbb{R}$. We define global reflectional symmetries as planes of reflection that apply to an entire landmark structure or to a complete facade. As photographs often capture single, symmetric facades of otherwise asymmetric buildings, treating these as global symmetries reflects this common real-world scenario and provides a well-defined target for our model.

A large-scale dataset of real-world 3D symmetries is essential for training a symmetry detector for in-the-wild images. However, defining real-world symmetries is inherently subjective, as landmarks inevitably contain minor imperfections that break perfect symmetry. To resolve this, we ground our definition in *human perception*: a structure is considered symmetric if its distinct surfaces are visually indistinguishable, a phenomenon known as *perceptual aliasing*. This ensures we capture dominant structural symmetries while naturally ignoring local architectural variations. Guided by this perceptual definition, we introduce an automated pipeline that extracts symmetry annotations directly from structure-from-motion (SfM) reconstructions of landmark scenes.

Our pipeline is motivated by two observations. The first is a classical technique in 3D reconstruction: matching an image to its own reflected counterpart is an effective way to extract symmetries visible from the image [15] (*within-view* matching). The second is an insight from the "doppelganger" problem: image matchers often find incorrect, yet geometrically consistent, matches between visually similar but physically distinct structures [2, 47]. We find that within-view matching tends to only discover dominant symmetries, such as a reflection across a main facade. Therefore, we leverage the insight from the doppelganger problem and additionally match each image against the reflected versions of *other*, visually similar images from the scene (*cross-view* matching). This allows us to recover symmetries that are not apparent or fully visible from any single viewpoint. For example, as visualized in Figure 2, within-view matching on the Arc de Triomphe extracts the reflection across its main facade, while cross-view matching recovers the front-to-back reflection, which is less obvious from a single image.

3.1. Data and preprocessing

To create ArchSym, we select 93 landmark scenes from MegaScenes, a large dataset of in-the-wild landmark image collections [40]. For each scene, given its set of images $\mathcal{I} = \{I_i\}$, we produce a structure-from-motion (SfM) reconstruction of the scene by running MASt3R-SfM [7], along with Doppelganger++ [47] to correct reconstruction errors. The reconstruction yields an intrinsic camera matrix, extrinsic pose, and estimated depth map for each image. We also produce a set of horizontally reflected images $\mathcal{I}' = \{I'_i\}$.

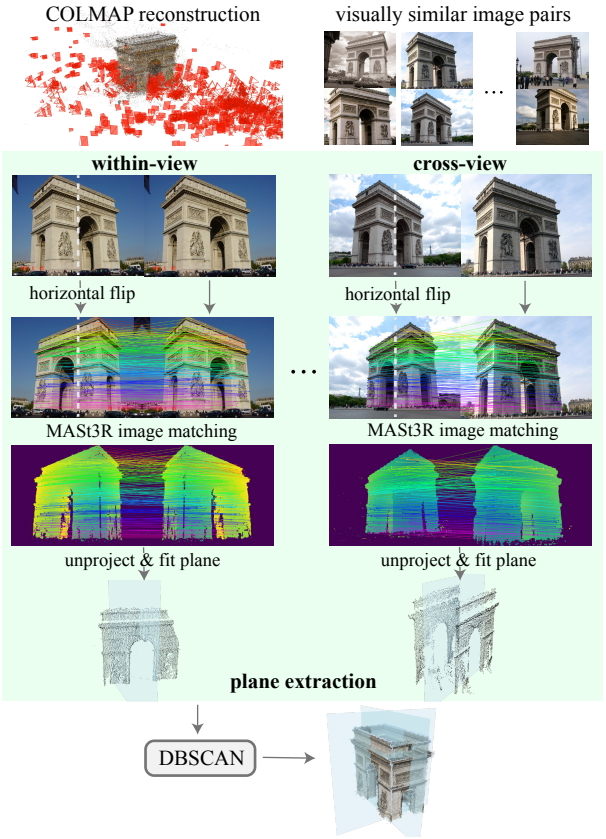

Figure 2. **Overview of our automated pipeline for extracting symmetry annotations.** We visualize *within-view* (left) and *cross-view* (right) matching on image pairs sampled from an SfM reconstruction. For each pair, we horizontally flip one image, find dense matches with the other image via MASt3R [7], unproject matched pixels to 3D points using depth maps, and fit a plane to the resulting point pairs. The final symmetry planes annotations are then determined by clustering candidate planes with DBSCAN [9].

3.2. Symmetry plane annotation

As shown in Figure 2, the process for generating a single symmetry annotation from a pair of images is as follows:

1. **Image pair selection.** For each image I_i, in addition to its own reflection I'_i, we sample a subset of visually similar reflected images $\mathcal{J}_i \subset \mathcal{I}'$ based on ASMK similarity [38].
2. **Image matching.** Pairing image I_i with flipped image $I'_j \in \mathcal{J}_i \cup \{I'_i\}$, we run an image matcher, MASt3R [17], to find 2D correspondences $\mathcal{M} = \{(x_i^k, x_j'^k)\}$.
3. **3D point unprojection.** For each 2D correspondence $(x_i^k, x_j'^k)$, pixel x_i^k is unprojected to a 3D point $\boldsymbol{p}_i^k$ using

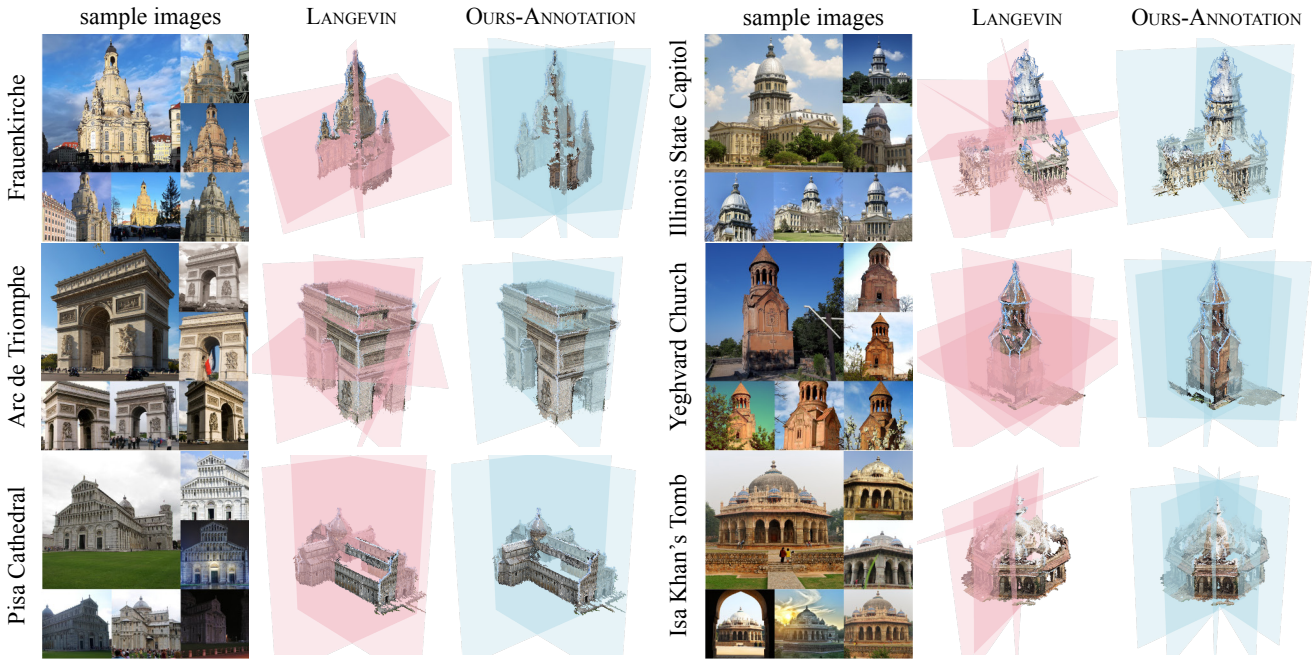

Figure 3. **Visualization of automated symmetry plane annotations.** We run LANGEVIN [13] and our symmetry extraction pipeline on six scenes from MegaScenes. Extracted planes are visualized with a dense point cloud from COLMAP [33, 34]. For LANGEVIN, the dense point cloud from COLMAP is used as input. For OURS-ANNOTATION, sampled pairs of input images and depth maps are used as input. We observe that LANGEVIN, as a purely geometry-based method, is highly sensitive to incomplete point clouds (e.g. Frauenkirche, Isa Khan's Tomb) and detects architecturally implausible planes (e.g., horizontal plane on the Arc de Triomphe, misaligned plane on the Pisa Cathedral). In contrast, OURS-ANNOTATION extracts semantically-correct symmetries. All planes are visualized without manual filtering or postprocessing.

its depth and camera parameters from the reconstruction. The corresponding 3D point $\boldsymbol{p}_j^k$ is similarly computed by unprojecting the original, un-flipped pixel coordinate of $x_j'^k$. This yields two sets of corresponding 3D points, $\mathcal{P}_i$ and $\mathcal{P}_j$, approximately related by a reflectional symmetry.

4. **Symmetry plane fitting.** We estimate the parameters of a candidate symmetry plane $\pi^* = (\boldsymbol{n}^*, d^*)$ from the 3D point correspondences by minimizing the sum of squared distances between points in $\mathcal{P}_i$ and reflections of corresponding points in $\mathcal{P}_j$:

$$(\boldsymbol{n}^*, d^*) = \underset{\|\boldsymbol{n}\|=1, d}{\operatorname{argmin}} \sum_k \|\boldsymbol{p}_i^k - \mathcal{R}_{\boldsymbol{n},d}(\boldsymbol{p}_j^k)\|^2. \quad (1)$$

where $\mathcal{R}_{\boldsymbol{n},d}$ is the reflection operator for a symmetry plane with normal $\boldsymbol{n}$ and offset d.

3.3. Plane aggregation and verification

Each image pair yields a potentially noisy candidate plane. We aggregate thousands of planes from each scene and use DBSCAN [9] to cluster these candidate planes. The centers of large clusters are kept as high-confidence symmetry annotations. For the final dataset, we manually inspect these high-confidence planes and filter out any incorrect or local symmetries and add global symmetries that our automated process may have missed. The resulting planes are treated as ground truths for training our symmetry detector.

3.4. Symmetry annotation results

Dataset statistics. Our final ArchSym dataset consists of 93 landmark scenes and a total of 34,177 images. The distributions of images and annotated symmetries per scene are shown in Figure 4. Most scenes contain one or two symmetries, while a few contain four (e.g., clock towers) or up to eight (e.g., octagonal buildings) symmetries. The images contribute a wide range of challenges, including partial views, occlusions, and varying camera parameters and illumination. The dataset provides a rigorous benchmark for evaluating symmetry detection across a wide range of real-world image conditions.

Qualitative comparisons. To validate the effectiveness of the symmetry extraction method used to build ArchSym, we compare the quality of our extracted symmetry annotations to those produced by a recent state-of-the-art geometry-based method that uses Riemannian Langevin dynamics to detect reflectional symmetries [13], or LANGEVIN for short. As ground truth is subjective for this task, the comparison is primarily qualitative (Figure 3). We provide dense SfM point clouds as input to LANGEVIN. All visualized planes are direct outputs of the two methods without manually filtering.

For each scene, we visualize sample images and the dense point cloud from COLMAP overlaid with the symmetry planes detected by LANGEVIN and our method. We note several failure modes of the geometry-based approach. LANGEVIN is highly sensitive to incomplete reconstructions

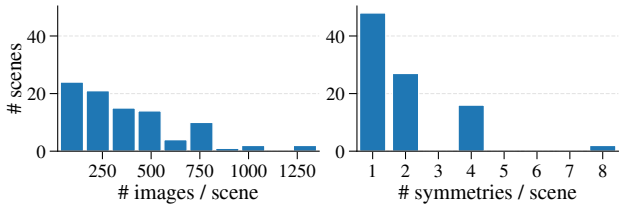

Figure 4. **Statistics of the ArchSym dataset,** showing the distribution of the number of images available (left) and the number of global symmetries annotated (right) in each scene.

and fails to identify symmetries where points to one side of the symmetry plane are largely missing (e.g., Isa Khan's Tomb, Frauenkirche). It often prioritizes the coarse shape of the point cloud over the underlying architectural semantics. For example, on the Pisa Cathedral, LANGEVIN incorrectly places a symmetry plane halfway along the building's length, and on cuboid-like silhouettes (e.g., Arc de Triomphe), it detects horizontal planes that are architecturally implausible.

In contrast, our method consistently extracts semantically correct two-way and four-way symmetries for landmarks across a wide range of architectural styles. For more challenging landmarks such as Isa Khan's Tomb, which has an eight-way symmetry, our method successfully extracts five of these reflectional symmetries via image matching, and the remaining three symmetries can be robustly estimated from the shared intersection line of the extracted symmetries during postprocessing. Moreover, for scenes like the Pisa Cathedral, our method correctly identifies the multiple reflectional symmetries at the center of the distinct facades.

4. Single-view symmetry detector

Using our newly curated ArchSym dataset, we train a model to predict 3D symmetry planes from a single RGB image. We choose to finetune a 3D foundation model, VGGT [42], adding a symmetry plane prediction head to its backbone, which is pretrained on diverse 3D geometry tasks. We elaborate on the model architecture in Section 4.2.

A key challenge in single-view 3D symmetry detection is *scale ambiguity*, which makes grounding symmetry planes in 3D an ill-posed task. Existing methods [19, 35, 37, 51] address scale ambiguity only partially—either by only predicting plane normals and relying on downstream optimization to localize them in 3D, or by using RGB-D inputs that provide a reference scale. Our choice of finetuning VGGT, which already predicts scene geometry, circumvents this issue: the predicted point map provides a natural, scale-consistent coordinate frame for grounding 3D symmetry.

However, a straightforward two-stage baseline of running a geometry-based symmetry detector on this point map output often fails in practice, since VGGT predicts geometry for only the *visible* parts of the scene, resulting in incomplete point clouds that often include irrelevant geometry (e.g., other buildings). Therefore, we instead allow our new symmetry head to implicitly reason about symmetries by operating directly on the frozen VGGT backbone features. As we describe next, we parameterize reflectional symmetries in relation to this predicted geometry, ensuring that our predictions are both 3D-grounded and consistent with the underlying scene geometry.

4.1. Symmetry prediction as a signed distance map

To provide a scale-consistent supervision signal, rather than directly regressing plane parameters, we formulate the problem as a dense prediction task: for each pixel, the model predicts the signed distance from that pixel's corresponding *predicted* 3D point to the ground truth symmetry plane.

Given a predicted point map $\widehat{\mathcal{P}} = \{\widehat{\boldsymbol{p}}^k\}$ from the frozen point head and a ground truth point map $\mathcal{P} = \{\boldsymbol{p}^k\}$, we first compute a similarity transformation $T : \mathbb{R}^3 \to \mathbb{R}^3$ that aligns $\mathcal{P}$ to $\widehat{\mathcal{P}}$. We also apply T to the ground truth symmetry plane to align it to the predicted point map. Using the aligned plane $\pi = (\boldsymbol{n}, d)$, we compute a signed distance map $\mathcal{S} = \{s^k\}$ as

$$s^k = \boldsymbol{n}^\top \widehat{\boldsymbol{p}}^k + d. \tag{2}$$

This signed distance map is treated as the *ground truth* for training purposes[1]. We note that these signed distance maps remain constant throughout training, since they correspond to the fixed 3D pointmaps predicted by the fixed VGGT head. This provides a stable, scale-consistent supervision signal that naturally resolves inherent scale ambiguities. We visualize examples of signed distance maps in Figure 5.

4.2. Multiple symmetry plane prediction head

Detecting multiple symmetries within a single image requires both global reasoning to identify potential symmetry planes and pixel-level precision to accurately localize each plane in 3D. Accordingly, our symmetry plane prediction head is designed as a two-stage architecture. In the first stage, a transformer-based [41] module identifies a set of potential symmetry planes, while in the second stage, a dense prediction head regresses the actual signed distance maps. An overview of the architecture is shown in Figure 5.

Symmetry identification module. The first stage of the network performs high-level reasoning on the *final-layer* feature map $\mathbf{F}^{\text{final}}$ from the frozen VGGT backbone. A set of M learnable instance queries $(\boldsymbol{q}_1, \ldots, \boldsymbol{q}_M)$ are refined by a lightweight transformer decoder that allows them to attend to frozen backbone features:

$$(\boldsymbol{q}'_1, \ldots, \boldsymbol{q}'_M) = \mathsf{Decoder}((\boldsymbol{q}_1, \ldots, \boldsymbol{q}_M), \mathbf{F}^{\text{final}}). \tag{3}$$

Intuitively, each refined instance feature vector encodes information about a potential symmetry plane in the scene.

[1] Crucially, this signed distance map is derived from the network's own *predicted* geometry as opposed to the ground truth geometry, which allows the model to make *self-consistent* predictions.

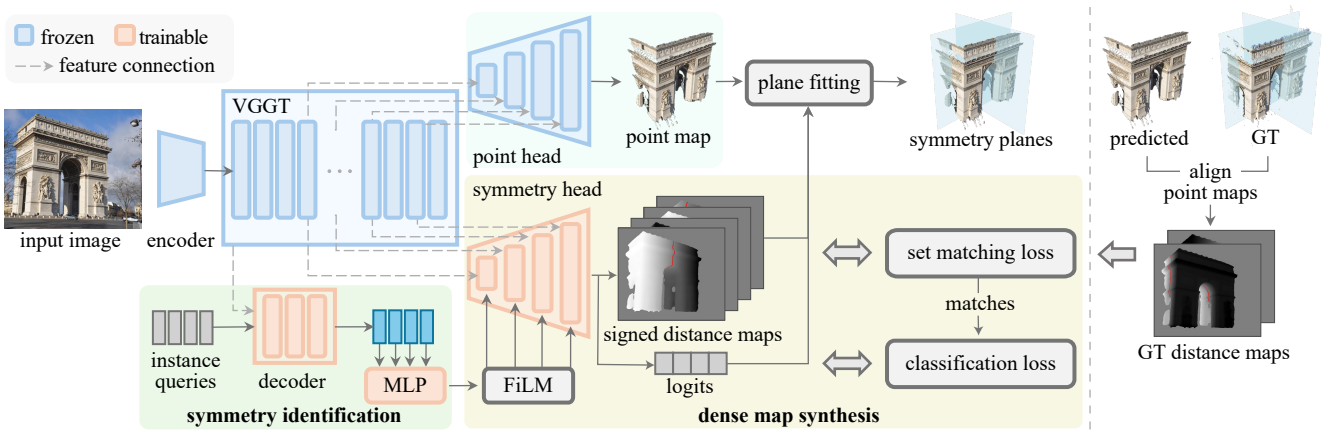

Figure 5. **Overview of our single-view symmetry detector architecture.** A frozen VGGT [42] backbone first extracts features from a single input image. The features are processed by a transformer decoder with learnable instance queries to identify symmetries. A lightweight MLP generates conditioning parameters from the resulting instance features. Then, a DPT-style [29] prediction head fuses multi-layer features to generate signed distance maps, with conditioning parameters injected via FiLM layers [28]. The model also predicts classification logits for extracting valid symmetries at inference time. The final 3D-grounded symmetry planes are recovered by solving an optimization problem over the predicted signed distance maps and the point map from VGGT's frozen point head. In the visualized signed distance maps, lighter pixels indicate positive values and darker pixels indicate negative values. Pixels with values close to zero are highlighted in red.

Dense map synthesis module. The second stage synthesizes the final dense maps using a dense prediction transformer (DPT) head [29]. We propose a lightweight extension to the DPT head that allows it to predict multiple instances of symmetry planes: Before each of the four feature fusion blocks, we inject instance-specific information using a feature-wise linear modulation (FiLM) layer [28].

Scale and shift $(\boldsymbol{\gamma}_i^l, \boldsymbol{\beta}_i^l)$ vectors used for the FiLM layers are regressed from refined instance features $\boldsymbol{q}_i'$ from the previous stage through a small MLP. We follow Peebles and Xie [27] and apply adaptive layer normalization (adaLN) before each conditioning layer. Each fusion stage takes the current fused feature $\mathbf{G}_i^l$ and the feature map $\mathbf{F}^l$ from the VGGT backbone and combines them via a RefineNet-based feature fusion block [20, 29]:

$$\tilde{\mathbf{G}}_i^l = (1+\boldsymbol{\gamma}_i^l)\cdot \mathsf{adaLN}(\mathbf{G}_i^l) + \boldsymbol{\beta}_i^l, \tag{4}$$

$$\mathbf{G}_i^{l+1} = \mathsf{FusionBlock}(\tilde{\mathbf{G}}_i^l, \mathbf{F}^l). \tag{5}$$

The output of the final fusion block is passed through a lightweight convolutional head to produce a set of predicted signed distance maps and confidence maps:

$$(\widehat{\mathcal{S}}_i, \mathcal{C}_i) = \mathsf{ConvHead}(\mathbf{G}_i^{\text{final}}). \tag{6}$$

Bipartite matching loss. Given ground truth distance maps $\{\mathcal{S}_1, \ldots, \mathcal{S}_N\}$, predicted distance maps $\{\widehat{\mathcal{S}}_1, \ldots, \widehat{\mathcal{S}}_M\}$, and confidence maps $\{\mathcal{C}_1, \ldots, \mathcal{C}_M\}$, we compute pairwise matching costs between all ground truths and predictions and find an optimal matching using the Hungarian matching algorithm [16], following prior work (e.g. DETR [3]). Similar to the confidence-aware point prediction loss proposed by Wang et al. [43], the pairwise matching cost is computed as

$$\mathcal{L}_{ij} = \sum_k c_i^k \cdot \ell^1(\hat{s}_i^k, s_j^k) - \alpha \log c_i^k, \tag{7}$$

where α is a hyperparameter that controls the regularization strength. Given an optimal matching $\mathcal{M}$, the final loss is computed as the mean cost over matched pairs:

$$\mathcal{L} = \frac{1}{|\mathcal{M}|} \sum_{(i,j)\in\mathcal{M}} \mathcal{L}_{ij}. \tag{8}$$

Classification loss. To filter predicted symmetry planes, we pass the final feature maps through a small MLP to produce classification logits. Predicted planes matched to ground truth planes receive a positive label and unmatched ones receive a negative label. At inference time, we threshold on the predicted logits to output only valid planes.

4.3. Symmetry plane fitting

Our model is trained on dense signed distance map predictions without extracting explicit symmetry planes. Only at inference time do we use the predicted point maps and signed distance maps to extract the actual planes.

We first filter symmetry predictions by thresholding on the predicted classification logits. For each valid plane instance, we select a high-confidence set of 3D points by thresholding on predicted confidence maps. This filtering produces a final set of predicted 3D points $\widehat{\mathcal{P}}'$ and their corresponding signed distances $\widehat{\mathcal{S}}'$ for each plane. We fit an optimal plane $\widehat{\pi} = (\widehat{\boldsymbol{n}}, \widehat{d})$ by solving a constrained least-squares optimization problem:

$$(\widehat{\boldsymbol{n}}, \widehat{d}) = \underset{|\boldsymbol{n}|=1, d}{\operatorname{argmin}} \sum_{\boldsymbol{p}\in\widehat{\mathcal{P}}', s\in\widehat{\mathcal{S}}'} \left((\boldsymbol{n}^\top \boldsymbol{p} + d) - s\right)^2, \tag{9}$$

which yields the final set of predicted planes.

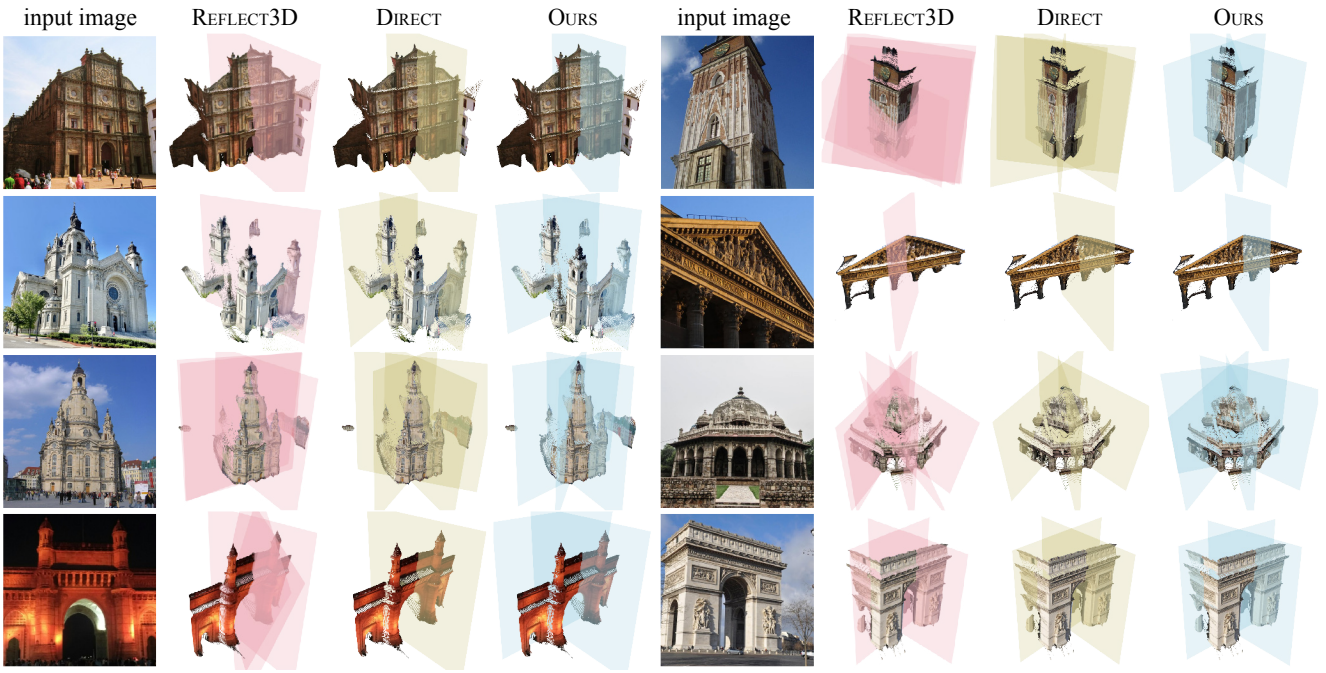

Figure 6. **Qualitative comparison of single-view symmetry detection results.** Input images are sampled from eight different test scenes. Since REFLECT3D [19] does not predict plane offsets, for visualization purposes, we use the point closest to the center of the landmark on the corresponding ground truth plane as an anchor point for REFLECT3D's predicted normals. Point clouds shown are predicted by VGGT [42]. We observe that REFLECT3D often misses partially visible symmetries and produces redundant detections, DIRECT predicts planes with more accurate normals but are often misaligned with the scene geometry, while OURS consistently predicts planes with accurate orientation and alignment. We encourage zooming into the figure to see differences in plane orientation and alignment in detail.

5. Results

We compare our method to a recent state-of-the-art single-view symmetry detector, REFLECT3D [19], as well as a simple baseline, DIRECT, where we use features from the frozen VGGT backbone to directly regress symmetry plane parameters. We empirically find that parameterizing each symmetry plane by its normal direction and a point on the plane (instead of its offset) leads to more stable training and more accurate predictions. Since REFLECT3D was originally trained on object-level datasets without complex backgrounds (e.g., ShapeNet and Objaverse), we finetune their model on the ArchSym dataset to ensure a fair comparison.

Experimental setup. We randomly split the 93 scenes from the ArchSym dataset into a set of 74 training scenes and a set of 19 test scenes. We split by scene rather than by image to ensure that there is no scene overlap between the training and test sets. This is crucial for preventing data leakage and for accurately evaluating generalizability to unseen structures.

Evaluation metrics. Since REFLECT3D predicts symmetry plane normals while OURS and DIRECT predict full plane parameters, we use two different metrics for comparison.

Normal-only. Following Li et al. [19], we use two evaluation metrics based on the geodesic distance, or angular error, of the predicted normals. For a given input image I, we denote the set of ground truth symmetry plane normals as $\mathcal{N}$, the set of predicted plane normals as $\widehat{\mathcal{N}}$, and the set of ground truth symmetries that are *visible* from the input image as $\mathcal{N}_{\text{vis}} \subseteq \mathcal{N}$. Specifically, a symmetry plane is considered visible in an image if at least 5% of all visible points lie on each side of the plane. This filtering is necessary as a zoomed-in view of one facade, for example, provides no visual evidence for symmetries on other distinct facades of the building. We refer to the supplementary material for details on visibility filtering.

The geodesic distance is computed as the average of two metrics: *exactness*, defined as the mean angular error from each predicted $\widehat{\boldsymbol{n}} \in \widehat{\mathcal{N}}$ to the closest ground truth $\boldsymbol{n} \in \mathcal{N}$, and *completeness*, defined as the mean angular error from each *visible* ground truth $\boldsymbol{n} \in \mathcal{N}_{\text{vis}}$ to the closest predicted $\widehat{\boldsymbol{n}} \in \widehat{\mathcal{N}}$. Exactness is assessed against all valid symmetries while completeness is assessed only against symmetries visible in the image. We also report the F-score at various angular thresholds (F@x°), which reflects the proportion of correct predictions within the threshold. We refer to the supplementary material for details on F-score calculation.

Full-plane. We follow Shi et al. [35] and report the dense symmetry error defined with respect to the ground truth point cloud $\mathcal{P}$. We denote the reflection transformation of a plane π as $\mathcal{R}_\pi$. Then, the dense symmetry error is computed as

$$\mathcal{E}_{\text{dense}}(\widehat{\pi},\pi) = \frac{1}{|\mathcal{P}|} \sum_{\boldsymbol{p}\in\mathcal{P}} \frac{\|\mathcal{R}_{\widehat{\pi}}(\boldsymbol{p}) - \mathcal{R}_\pi(\boldsymbol{p})\|}{\rho(\mathcal{P},\pi)}, \qquad (10)$$

Table 1. **Quantitative evaluation of single-view symmetry detection.** Evaluation metrics are averaged across all test scenes. REFLECT3D [19] 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 (↓).

Method	Normal-only				Full-plane
	Geo ↓	F@1° ↑	F@5° ↑	F@15° ↑	$\mathcal{E}_{\text{dense}}$ ↓
R3D	10.46	0.07	0.34	0.55	—
DIR	5.06	0.16	0.64	0.81	0.18
OURS	**3.71**	**0.25**	**0.70**	**0.84**	**0.13**

where $\rho(\mathcal{P}, \pi)$ is a normalization constant equal to the maximum distance from any point in $\mathcal{P}$ to the ground truth plane π. Using the pairwise error as our distance measure, the final error is computed by averaging the exactness and completeness components, analogous to the geodesic distance metric in the normal-only evaluation.

Results. We summarize our single-view symmetry detection evaluation results, averaged across 19 test scenes, in Table 1. As shown in the table, OURS outperforms both REFLECT3D and DIRECT in normal-only prediction and outperforms DIRECT in full-plane prediction. We refer to the supplementary material for detailed per-scene evaluation results.

Qualitative comparisons. We show qualitative comparisons on eight test scene images in Figure 6. While REFLECT3D can often identify the orientation of the most dominant symmetry plane, it struggles with partially visible symmetries and produces redundant detections. Meanwhile, DIRECT benefits from the VGGT backbone features and predicts more accurate normals, but often predicts planes misaligned with the scene geometry, highlighting the challenge of directly regressing plane parameters. In contrast, our signed-distance-based parameterization allows OURS to consistently predict planes that are well-aligned with the scene geometry.

Applications. We demonstrate the accuracy of our symmetry detector via a simple downstream application in single-view point cloud completion in Figure 7. Since geometric foundation models like VGGT [42] only predict geometry for visible pixels in the image, output point clouds are inherently incomplete (e.g., missing the back of a building). By simply reflecting the predicted points across our detected symmetry planes, we can accurately "hallucinate" the occluded geometry.

6. Limitations

Our approach to curating ArchSym relies on the availability of accurate SfM reconstructions, whose quality directly influences the extracted symmetries. While our current pipeline focuses exclusively on reflectional symmetries, it can be adapted to extract rotational symmetries by matching images against *non-reflected* versions of other views. We did not prioritize this as pure rotational symmetries are less prevalent in architectural landmarks and can often be derived from the composition of reflectional symmetries. Our pipeline focuses on discovering global symmetries by aggregating correspondences from full-image matching. It is not designed to detect partial symmetries (e.g., individual windows), which is an interesting direction for future work.

Figure 7. Single-view completion using detected symmetries. We visualize the input image (top left), a top-down view of the completed point cloud (bottom left), and frontal view comparisons of the original VGGT (top right) and our completed results (bottom right).

The performance of our symmetry detector is linked to the geometry predicted by the VGGT point head. Our signed-distance-based approach prioritizes geometric consistency and may produce less accurate planes when the predicted geometry is highly ambiguous, e.g., due to severe occlusion or blurring. While methods that directly regress plane parameters may produce plausible orientation estimates in such cases, it is unclear whether they can be meaningfully localized in 3D.

7. Conclusion

We present a complete pipeline for tackling the previously unaddressed problem of detecting 3D-grounded symmetries from single RGB images of in-the-wild scenes. Due to the lack of existing training data, we first introduce a novel automated pipeline that leverages cross-image reflection matching within SfM reconstructions to curate ArchSym, a large-scale dataset of architectural images labeled with symmetries. Building on the dataset, we propose a novel detection model that parameterizes symmetry planes as signed distance maps relative to the model's own predicted geometry, naturally resolving the scale ambiguity issue inherent in monocular reconstruction. Our experiments demonstrate that our approach significantly outperforms existing alternatives. We believe our ArchSym dataset and method provide a strong foundation for future research, allowing 3D symmetries to be used as a robust geometric prior for in-the-wild 3D reconstruction and pose estimation.

Acknowledgments This work was funded in part by the National Science Foundation (IIS-2212084) and by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korean Government (MSIT) (No. RS-2024-00457882, National AI Research Lab Project). We thank the authors of VGGT [42], REFLECT3D [19], and LANGEVIN [13] for assistance with model finetuning and running baselines.

References

[1] Mikhail J. Atallah. On symmetry detection. *IEEE Transactions on Computers*, 100(7):663–666, 1985. 1

[2] Ruojin Cai, Joseph Tung, Qianqian Wang, Hadar Averbuch-Elor, Bharath Hariharan, and Noah Snavely. Doppelgangers: Learning to disambiguate images of similar structures. In *ICCV*, 2023. 2, 3

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 6

[4] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint*, 2015. 2

[5] Marcelo Cicconet, David GC Hildebrand, and Hunter Elliott. Finding mirror symmetry via registration and optimal symmetric pairwise assignment of curves: Algorithm and results. In *ICCV Workshops*, pages 1759–1763, 2017. 2

[6] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *CVPR*, 2023. 2

[7] 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. 3, 11

[8] Mohamed Elawady, Christophe Ducottet, Olivier Alata, Cécile Barat, and Philippe Colantoni. Wavelet-based reflection symmetry detection via textural and color histograms. In *ICCV Workshops*, pages 1725–1733, 2017. 2

[9] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proc. KDD*, page 226–231. AAAI Press, 1996. 3, 4

[10] Christopher Funk and Yanxi Liu. Beyond planar symmetry: Modeling human perception of reflection and rotation symmetries in the wild. In *ICCV*, 2017. 2

[11] Christopher Funk, Seungkyu Lee, Martin R Oswald, Stavros Tsogkas, Wei Shen, Andrea Cohen, Sven Dickinson, and Yanxi Liu. 2017 iccv challenge: Detecting symmetry in the wild. In *ICCV Workshops*, 2017. 2

[12] Lin Gao, Ling-Xiao Zhang, Hsien-Yu Meng, Yi-Hui Ren, Yu-Kun Lai, and Leif Kobbelt. Prs-net: Planar reflective symmetry detection net for 3d models. *IEEE TVCG*, 27(6): 3007–3018, 2020. 2

[13] Jihyeon Je, Jiayi Liu, Guandao Yang, Boyang Deng, Shengqu Cai, Gordon Wetzstein, Or Litany, and Leonidas Guibas. Robust symmetry detection via riemannian langevin dynamics. In *SIGGRAPH Asia 2024 Conference Papers*, pages 1–11, 2024. 2, 4, 8

[14] Nahum Kiryati and Yossi Gofman. Detecting symmetry in grey level images: The global optimization approach. *International Journal of Computer Vision*, 29(1):29–45, 1998. 2

[15] Kevin Köser, Christopher Zach, and Marc Pollefeys. Dense 3d reconstruction of symmetric scenes from a single image. In *Joint Pattern Recognition Symposium*, pages 266–275. Springer, 2011. 2, 3

[16] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955. 6

[17] Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with mast3r. In *ECCV*, 2024. 3

[18] Ren-Wu Li, Ling-Xiao Zhang, Chunpeng Li, Yu-Kun Lai, and Lin Gao. E3sym: Leveraging e (3) invariance for unsupervised 3d planar reflective symmetry detection. In *ICCV*, 2023. 2

[19] 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, 5, 7, 8, 11, 12, 14

[20] Guosheng Lin, Anton Milan, Chunhua Shen, and Ian Reid. Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. In *CVPR*, 2017. 6

[21] Yancong Lin, Silvia-Laura Pintea, and Jan van Gemert. Nerd++: Improved 3d-mirror symmetry learning from a single image. *arXiv preprint*, 2021. 2

[22] Gareth Loy and Jan-Olof Eklundh. Detecting symmetry and symmetric constellations of features. In *ECCV*, 2006. 2

[23] Nathaniel Merrill, Yuliang Guo, Xingxing Zuo, Xinyu Huang, Stefan Leutenegger, Xi Peng, Liu Ren, and Guoquan Huang. Symmetry and uncertainty-aware object slam for 6dof object pose estimation. In *CVPR*, 2022. 1

[24] 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. 13

[25] Niloy J Mitra, Mark Pauly, Michael Wand, and Duygu Ceylan. Symmetry in 3d geometry: Extraction and applications. In *Comput. Graph. Forum*, pages 1–23. Wiley Online Library, 2013. 2

[26] Rajendra Nagar and Shanmuganathan Raman. Symmmap: Estimation of the 2-d reflection symmetry map and its applications. In *ICCV Workshops*, pages 1715–1724, 2017. 2

[27] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *ICCV*, pages 4195–4205, 2023. 6

[28] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *AAAI*, 2018. 6, 11, 12

[29] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *ICCV*, pages 12179–12188, 2021. 6, 12

[30] 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. 13

[31] 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. 13

[32] Tadamasa Sawada, Yunfeng Li, and Zygmunt Pizlo. Detecting 3-d mirror symmetry in a 2-d camera image for 3-d shape recovery. *Proceedings of the IEEE*, 102(10):1588–1606, 2014. 2

[33] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 2, 4

[34] 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. 4, 13

[35] Yifei Shi, Junwen Huang, Hongjia Zhang, Xin Xu, Szymon Rusinkiewicz, and Kai Xu. Symmetrynet: Learning to predict reflectional and rotational symmetries of 3d shapes from single-view rgb-d images. *ACM TOG*, 39(6):1–14, 2020. 1, 2, 5, 7

[36] Yifei Shi, Zixin Tang, Xiangting Cai, Hongjia Zhang, Dewen Hu, and Xin Xu. Symmetrygrasp: Symmetry-aware antipodal grasp detection from single-view rgb-d images. *RA-L*, 7(4): 12235–12242, 2022. 1

[37] Yifei Shi, Xin Xu, Junhua Xi, Xiaochang Hu, Dewen Hu, and Kai Xu. Learning to detect 3d symmetry from single-view rgb-d images with weak supervision. *IEEE TPAMI*, 45(4): 4882–4896, 2022. 2, 5

[38] Giorgos Tolias, Yannis Avrithis, and Hervé Jégou. To aggregate or not to aggregate: Selective match kernels for image search. In *ICCV*, pages 1401–1408, 2013. 3

[39] Stavros Tsogkas and Iasonas Kokkinos. Learning-based symmetry detection in natural images. In *ECCV*, 2012. 2

[40] 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, 13

[41] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 30, 2017. 5

[42] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *CVPR*, 2025. 5, 6, 7, 8, 11

[43] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *CVPR*, pages 20697–20709, 2024. 6

[44] Jan D Wolter, Tony C Woo, and Richard A Volz. Optimal algorithms for symmetry detection in two and three dimensions. *The Visual Computer*, 1(1):37–48, 1985. 1

[45] Shangzhe Wu, Christian Rupprecht, and Andrea Vedaldi. Unsupervised learning of probably symmetric deformable 3d objects from images in the wild. In *CVPR*, 2020. 1, 2

[46] Shangzhe Wu, Ameesh Makadia, Jiajun Wu, Noah Snavely, Richard Tucker, and Angjoo Kanazawa. De-rendering the world's revolutionary artefacts. In *CVPR*, 2021. 1

[47] Yuanbo Xiangli, Ruojin Cai, Hanyu Chen, Jeffrey Byrne, and Noah Snavely. Doppelgangers++: Improved visual disambiguation with geometric 3d features. In *CVPR*, 2025. 2, 3

[48] Yuan Yao, Nico Schertler, Enrique Rosales, Helge Rhodin, Leonid Sigal, and Alla Sheffer. Front2back: Single view 3d shape reconstruction via front to back prediction. In *CVPR*, 2020. 1

[49] Zhaoxuan Zhang, Bo Dong, Tong Li, Felix Heide, Pieter Peers, Baocai Yin, and Xin Yang. Single depth-image 3d reflection symmetry and shape prediction. In *ICCV*, 2023. 2

[50] Heng Zhao, Shenxing Wei, Dahu Shi, Wenming Tan, Zheyang Li, Ye Ren, Xing Wei, Yi Yang, and Shiliang Pu. Learning symmetry-aware geometry correspondences for 6d object pose estimation. In *ICCV*, 2023. 1

[51] Yichao Zhou, Shichen Liu, and Yi Ma. Nerd: Neural 3d reflection symmetry detector. In *CVPR*, 2021. 1, 2, 5

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

Supplementary Material

A. Implementation details

Our implementation builds upon the official MASt3R [7] and VGGT [42] codebases.

A.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.

A.2. Network architecture

Our model consists of the pre-trained VGGT [42] 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 [28] 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.

B. Evaluation details

Detailed per-scene evaluation results on 19 test scenes are reported in Table A1. 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.

B.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.

B.2. Asymmetric F-score calculation

We present our F-score calculation scheme in Algorithm 2, which is based on the evaluation code from REFLECT3D [19]. 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*

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 A1. **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 [19] 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 (↓).

considered as false positives during F-score calculation, 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)
```

C. Ablation studies

We validate our design choices by comparing the full model against two baselines. First, we remove the instance-specific FiLM conditioning [28] (w/o FiLM) to predict multiple symmetry planes directly from shared DPT features [29]. 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 A2, we observe that both the orientation and alignment of the predicted planes

Table A2. **Ablation studies on model architecture and loss supervision.** Plane prediction quality severely degrades if we remove FiLM conditioning [28] (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 A1. **Generalization to object-centric scenes.** We demonstrate accurate symmetry annotation on real (CO3D [31], left/center) and synthetic (NeRF-Synthetic [24], right) objects. We show sample input images and annotated symmetry planes overlaid on COLMAP MVS [34] point clouds.

degrade severely in both cases, highlighting the effectiveness of our two-stage model architecture and the importance of self-consistent predictions.

D. Additional results

D.1. Generalization to object-centric data

Our paper focuses on architectural scenes since the MegaScenes dataset [40] 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 A1, 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 [30] and NeRF-Synthetic [24] datasets. The detected reflectional symmetry planes are overlaid on dense point clouds from COLMAP MVS [34] for visualization.

D.2. Additional qualitative comparisons

Figure A2 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.

D.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 A3. 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.

Figure A2. **Additional qualitative comparisons of single-view symmetry detection results.** Input images are sampled from 16 different test scenes. REFLECT3D [19] 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.

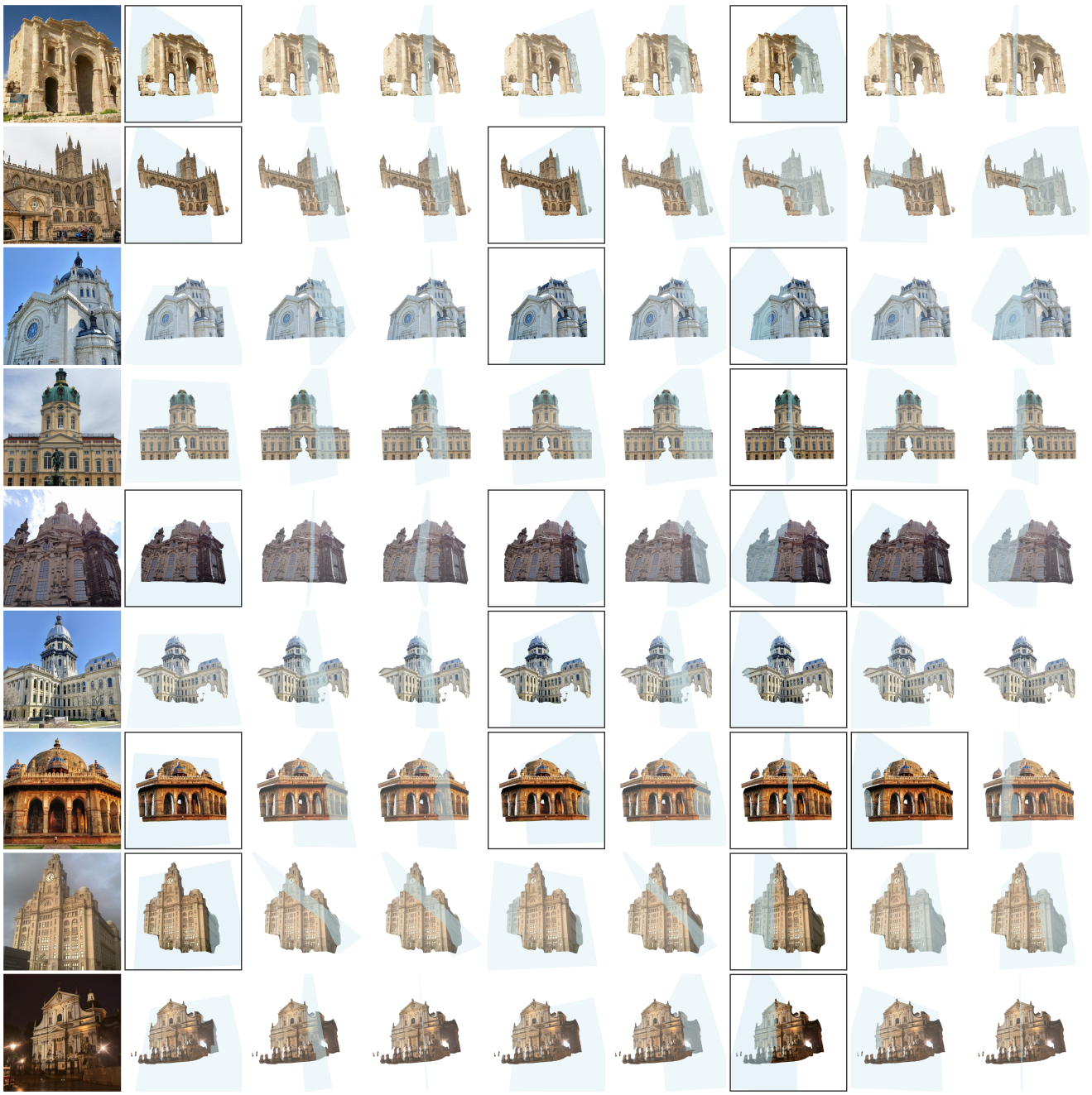

Figure A3. **Visualization of symmetry plane predictions from individual instance queries.** Each row shows an input image alongside the symmetry planes predicted from each of the eight instance queries. Highlighted frames indicate valid planes with predicted logits above the extraction threshold. This visualization demonstrates how different prediction slots specialize to capture specific types of symmetries within the scene.