NEAT: Distilling 3D Wireframes from Neural Attraction Fields

Nan Xue[1] Bin Tan[1,2] Yuxi Xiao[1,3] Liang Dong[4] Gui-Song Xia[2] Tianfu Wu[5*] Yujun Shen[1]

[1]Ant Group [2]Wuhan University [3]Zhejiang University [4]Google Inc. [5]NC State University

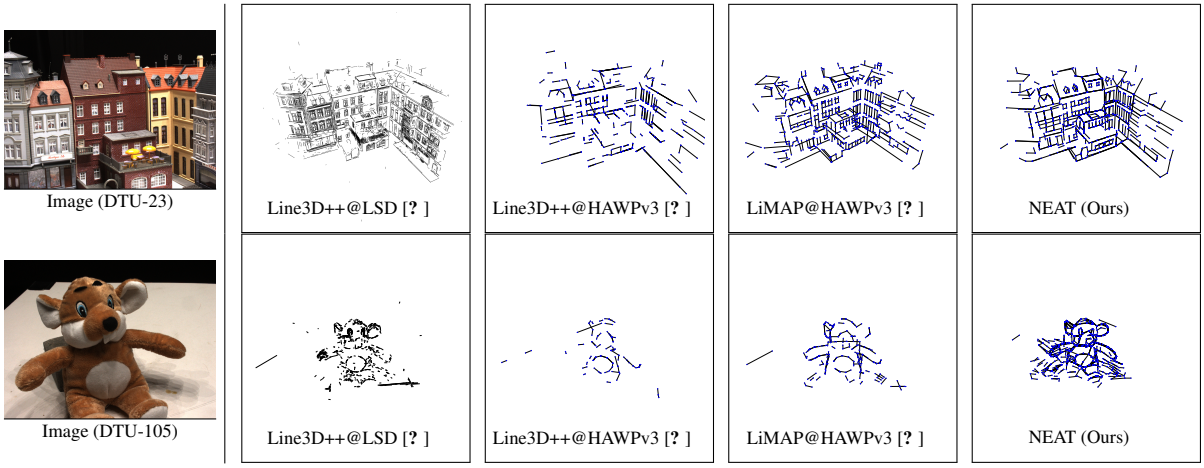


Figure 1. **Showcasing the evolution of 3D wireframe reconstruction**: The top reveals the transformative steps from a straight-line dominated urban landscape to an abstract wireframe, contrasting various methodologies. Below, the intricate transition from a curve-rich stuffed animal to its skeletal representation is depicted. While Line3D++ [**?**] and LiMAP [**?**] utilize line-matching techniques, our novel NEAT approach forgoes matching, resulting in superior reconstruction fidelity with our proposed rendering-distilling formulation.

Abstract

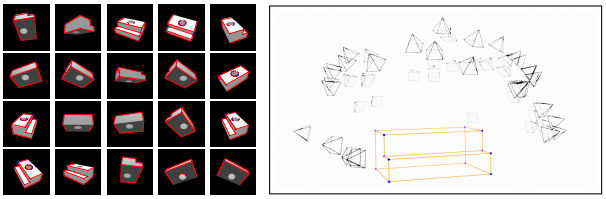
This paper studies the problem of structured 3D reconstruction using wireframes that consist of line segments and junctions, focusing on the computation of structured boundary geometries of scenes. Instead of leveraging matching-based solutions from 2D wireframes (or line segments) for 3D wireframe reconstruction as done in prior arts, we present NEAT, a ***rendering-distilling*** *formulation using neural fields to represent 3D line segments with 2D observations, and bipartite matching for perceiving and distilling of a sparse set of 3D global junctions. The proposed NEAT enjoys the joint optimization of the neural fields and the global junctions from scratch, using view-dependent 2D observations without precomputed cross-view feature matching. Comprehensive experiments on the DTU and BlendedMVS datasets demonstrate our NEAT's superiority over state-of-the-art alternatives for 3D wireframe reconstruction. Moreover, the distilled 3D global junctions by NEAT, are a better initialization than SfM points, for the recently-emerged 3D Gaussian Splatting for high-fidelity novel view synthesis using about 20 times fewer initial 3D points. Project page:* https://xuenan.net/neat.

*Corresponding author.

1. Introduction

In this paper, we explore the field of multi-view 3D reconstruction, drawing inspiration from the paradgim of the primal sketch proposed by D. Marr [**?**]. Our objective is to develop a concise yet precise representation of 3D scenes, derived from multi-view images with known camera poses. Specifically, our focus is on wireframe representations [**? ? ? ?**], which define the boundary geometry of scene images through line segments and junctions as the 2D wireframe representation. We dedicate our efforts to advancing the reconstruction of 3D wireframes based on their 2D counterparts detected in multi-view images, as shown in **??** and **??**.

The challenge of *multi-view 3D wireframe reconstruction* has been previously explored within the realm of line-based 3D reconstruction [**? ? ?**], primarily following the feature triangulation pipelines [**?**], which heavily rely on the accuracy of multi-view feature correspondences. Various methods have been developed to enhance this accuracy [**? ? ?**]. However, a significant challenge arises from view-dependent occlusions of line features: when projecting a 3D line segment onto 2D images, the endpoints of the line segment may be truncated in the 2D projections by chance. Such discrepancies can severely impact the



(a) Imagess & 2D Wireframes (b) 3D Wireframe from Posed Images

Figure 2. **Illustrative Overview** of the problem of 3D wireframe reconstruction. Given a set of posed images and the corresponding 2D wireframe detection results in (**??**), the proposed NEAT estimates the 3D wireframe representation of the scene in (**??**).

accuracy of the reconstruction, as the matching process relies on these endpoints to accurately represent the 3D geometry. These matching-based methods often result in incomplete 3D line models or suffer from fragmentation and noise, depending on the choice of 2D detectors [**? ? ? ? ? ?**] and matchers [**? ?**] of line segments, as in **??**.

Dense Fields of Sparse Geometries. We challenge the explicit matching pipeline of 3D wireframe reconstruction from the perspective of dense field representation. We draw inspiration from the "**implicit matching**" capacity [**?**] of the emerging neural implicit fields [**? ? ?**] for 3D dense representations (*e.g.*, density fields and signed distance functions), and propose to *render* 3D line segments from multi-view 2D observations. Such a basic idea roughly works by leveraging a coordinate MLP to render 3D line segments from 2D observations, but remains problematic due to the entailed view-by-view rendering of 3D line segments in two-fold: (1) the 2D line segments of a detected wireframe often undergo localization errors, resulting in erroneous 3D line segment predictions via view-by-view rendering, and (2) simply stacking the rendered 3D line segments from all views leads to a very large amount of 3D line segments, requiring non-trivial merging/fusion to form a 3D wireframe representation of the scene.

Line-to-Point Attraction in Neural Fields. We tackle the above issues by leveraging the line-to-point attraction that inherently persists in the wireframe representation, in which *every endpoint of a 3D line segment should be in the set of 3D junctions of the underlying scene*. Based on this, we formulate the two types of entities of 3D wireframes, the 3D line segments and junctions, in a novel *rendering-distilling* formulation, where the sparse set of 3D line segments are represented in a dense neural field while the junctions play the role of distilling a sparse wireframe structure from the fields. Our work is entitled as *NEural Attraction* (NEAT) for 3D wireframe reconstruction, mainly because of the neural design of the 3D line segments and junctions, and of leveraging the line-to-point attraction to enable joint optimization of the neural networks from multi-view images and its 2D wireframe detection results. To the best of our knowledge, we accomplish the first matching-free solution of 3D wireframe/line reconstruction by learning and optimizing from random initializations without any 3D scene information required.

In experiments, we showcase that our matching-free NEAT solution significantly outperforms all the matching-based approaches with accurate yet complete 3D wireframe reconstruction results on both the DTU [**?**] and BlendedMVS [**?**] datasets, working well in both straight-line dominated scenes and curve-based (or polygonal line segment dominated) scenes that challenges the traditional matching-based approaches, paving a way towards learning 3D primal sketch in a more general way. Furthermore, we show that the neurally perceived 3D junctions is applicable to the recently proposed 3D Gaussian Splatting [**?**] as better initialization than the COLMAP [**?**] with about 20 times fewer points, showing case the potential of structured and compact 3D reconstruction.

2. Related Work

Structured 3D Reconstruction in Geometric Primitives. Because of the inherent structural regularities for scene representation conveyed by line structures [**? ? ? ? ?**] and planar structures [**? ?**], there has been a vast body of literature on line-based multiview 3D reconstruction tasks including single-view 3D reconstruction [**? ?**], line-based SfM [**? ?**], SLAM [**? ?**], and multi-view stereo [**? ? ?**] based on the theory of multi-view geometry [**?**]. Due to the challenge of line segment detection and matching in 2D images, most of those studies expected the 2D line segments detected from input images to be redundant and small-length to maximize the possibility of line segment matching. As for the estimation of scene geometry and camera poses, the keypoint correspondences (even including the 3D point clouds) are usually required. For example in Line3D++ [**?**], given the known camera poses by keypoint-based SfM systems [**? ? ? ?**], it is still challenging though to establish reliable correspondences for the pursuit of structural regularity for 3D line reconstruction. For our goal of 3D wireframe reconstruction, because 2D wireframe parsers aim at producing parsimonious representations with a small number of 2D junctions and long-length line segments, those correspondence-based solutions pose a challenging scenario for cross-view wireframe matching, thus leading to inferior results than the ones using redundant and small-length 2D line segments detected by the LSD [**?**]. To this end, we present a correspondence-free formulation based on coordinate MLPs, which provides a novel perspective to accomplish the goal of 3D wireframe reconstruction from the parsed 2D wireframes.

Neural Rendering for Geometric Primitives. In recent years, the emergence of neural implicit representations [**? ? ? ?**] have greatly renown the 3D vision community.

By using coordinate MLPs to implicitly learn the scene geometry from multi-view inputs without knowing either the cross-view correspondences or the 3D priors, it has largely facilitated many 3D vision tasks including novel view synthesis, multi-view stereo, surface reconstruction, *etc*. Some recent studies further exploited the neural implicit representations by (explicitly and implicitly) taking the geometric primitives such as 2D segmentation masks into account to lift the 2D detection results into 3D space for scene understanding and interpretation [**?** **?** **?** **?**]. Most recently, nerf2nerf [**?**] exploited a geometric 3D representation, surface fields as a drop-in replacement for point clouds and polygonal meshes, and takes the keypoint correspondences to register two NeRF MLPs. Our study can be categorized as the exploration of geometric primitives in neural implicit representation, but we focus on computing a parsimonious representation by using the most fundamental geometric primitives, the junction (points) and line segments, to provide a compact and explicit representation from coordinate MLPs.

3. NEAT of 3D Wireframe Reconstruction

In this section, we formulate the problem of 3D wireframe reconstruction, lying on the high-level idea of approaching the goal of using volume rendering instead of the explicit line segment matching to build a unified 3D computational representation of line segments and junctions from the 2D detected wireframes.

Problem Statement. For the problem illustrated in **??**, we present our approach for 3D wireframe reconstruction from n-view posed images, $\{\mathcal{I}_i\}_{i=1}^{n}$. Each image $\mathcal{I}_i$ is characterized by intrinsic and extrinsic matrices. We use the HAWPv3 model [**?**] to detect 2D wireframes in these images, represented as undirected graphs $G_i = (V_i, E_i)$. The goal is to construct a 3D wireframe graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, translating these 2D wireframes into a 3D representation with $\mathcal{V}$ as 3D junctions and $\mathcal{E}$ as the 3D line segments.

Method Overview. Our NEAT method is built on the VolSDF framework [**?**] with two primary neural components: (1) a Neural Attraction Field for 3D line segments, and (2) a Global 3D Junction Perceiver (GJP). These components work jointly to create NEAT 3D wireframe models from the 2D wireframe observations. We start by learning a dense representation of 3D line segments from 2D wireframes using the Neural Attraction Field, as visualized in Figure **??**. This is followed by the Global 3D Junction Perceiver, which identifies a set of 3D junctions. As a final step of the wireframe reconstruction, the perceived 3D junctions play in a distillation role to clean up the optimized NEAT field. In implementation, we adopt a simple design for the MLPs used in the SDF and radiance field, aligned with VolSDF specifications. For the NEAT field, a 4-layer MLP renders the 3D line segments.

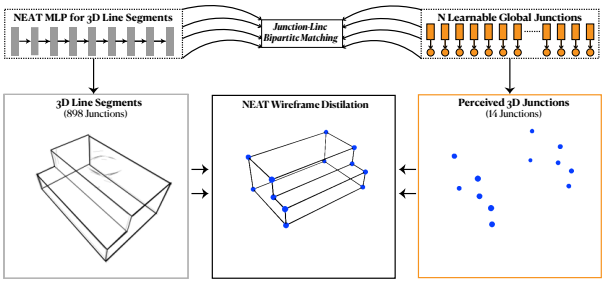


Figure 3. The proposed NEAT field learning framework for 3D wireframe reconstruction. In the top, the neural design of NEAT MLP and the predefined N global junctions are illustrated, these two components are "attracted" by the junction-to-line bipartite matching, resulting a rendering-yet-distillation formulation to render 3D line segments in NEAT MLP as a dense representation of 3D line segments, and then distilled by the learned 3D global junctions for wireframe reconstruction.

Additional implementation details and hyperparameters are outlined in the **??**.

3.1. Rendering 3D Line Segments from 2D

We propose to leverage the power of "implicit matching" ability of neural fields to obtain 3D line segments. Our method is built on the basic formulation of VolSDF [**?**] that renders a ray $\mathbf{x}_t = \mathbf{c} + t \cdot \mathbf{v}$ emanating from the camera location $\mathbf{c} \in \mathbb{R}^3$ with the (unit) view direction $v \in \mathbb{R}^3$ to estimate the image appearance by,

$$\hat{I}(c, v) = \int_0^\infty T(t) \cdot \sigma(x_t) \cdot \mathbf{r}(x_t, v, \mathbf{n}(x_t), z(x_t)) dt, \quad (1)$$

where $\mathbf{r}(\cdot)$ is the radiance of the ray x_t, and $T(t)$ is the transmittance $T(t) = \exp - \int_0^t \sigma\left(x(s)\right) ds$ along the ray from camera center to t, the density field $\sigma(\cdot)$ is transformed by the signed distance function $d_\Omega(\mathbf{x})$ of an implicit field using,

$$\sigma(\mathbf{x}) = \frac{1}{\beta} \Psi_\beta(-d_\Omega(\mathbf{x})), \quad (2)$$

with the learnable scaling factor β. As for the optimization of SDF and radiance fields, the image loss $\mathcal{L}_{\text{img}}$ between the rendered image $\hat{I}$ and its corresponding ground-truth $\mathcal{I}$, and Eikonal loss $\mathcal{L}_{\text{eik}}$ for SDF network are used.

Neural Attraction Fields. In our NEAT method, we adapt volume rendering, typically used for optimizing dense 3D representations like density fields and SDFs, to focus on 3D line segments and junctions. Our approach is inspired by the dense attraction field representations used in 2D line segment detection and wireframe parsing, as extensively researched in previous studies [**?** **?**]. As illustrated in **??** using a synthetic example, we utilize the attracted

pixels of 2D line segments in each image to define the rays for 3D rendering. For each segment, its attracted pixels are projected perpendicularly onto the 2D segment. This projection is confined within the endpoints of the segment with respect to a predefined distance threshold, τ_{ray}. Each pixel is associated with its nearest line segment, ensuring a dense coverage of supporting areas for the segments. This approach facilitates the volume rendering of 3D line segments by providing a robust underlying structure.

In our approach, we model a 3D line segment at any point $\mathbf{x}_t$ along a ray. The endpoint displacements $(\Delta\mathbf{x}_t^1, \Delta\mathbf{x}_t^2)$ relative to $\mathbf{x}_t$ are computed as,

$$(\Delta\mathbf{x}_t^1, \Delta\mathbf{x}_t^2) = L(\mathbf{x}_t) \in \mathbb{R}^{2\times 3}, \tag{3}$$

yielding the two endpoints of the segment by $(\mathbf{x}_t + \Delta\mathbf{x}_t^1, \mathbf{x}_t + \Delta\mathbf{x}_t^2)$. The mapping function $L(\cdot)$ is parameterized by a 4-layer coordinate MLP. It incorporates the view direction $\mathbf{v}$, the surface normal $\mathbf{n}(\cdot)$ from the SDF gradient, and a 128-dimensional feature vector $\mathbf{z}(\mathbf{x}_t)$ from the SDF network, reflecting the view-dependent nature of 2D line segments. For rendering a 3D line segment, we apply the equation,

$$(\mathbf{x}^s, \mathbf{x}^t) = \int_0^\infty T(t)\sigma(t)\left(L(\mathbf{x}_t) + \mathbf{x}_t\right) dt. \tag{4}$$

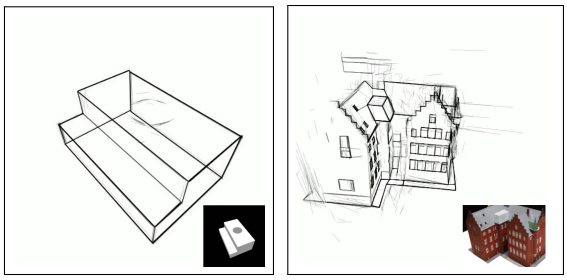
Here, $\mathbf{x}^s$ and $\mathbf{x}^t$ are the 3D endpoints for the attraction pixel $\mathbf{x}$ of a 2D line segment $\ddot{l} = (\jmath_1, \jmath_2) \in V_i \times V_i$ of the i-th view, calculated along its ray $\mathbf{x}_t$.

According to the pixel-to-line relationship defined by 2D attraction field representations, the rendered 3D line segment $(\mathbf{x}^s, \mathbf{x}^t)$ of a ray $\mathbf{x}_t$ should be consistent with $\ddot{l} = (\jmath_1, \jmath_2)$, thus resulting in a loss function between the projected 2D endpoints by viewpoint projection $\Pi(\cdot)$ and $\ddot{l}$ in,

$$\mathcal{L}_{\text{neat}} = \|\Pi(\mathbf{x}^s) - \jmath_1\|^2 + \left\|\Pi(\mathbf{x}^t) - \jmath_2\right\|^2. \tag{5}$$

The proposed Neural Attraction Fields of 3D line segments is optimized together with SDF and the radiance field by minimizing the loss functions stated above, forming a querable and dense representation of 3D line segments.

Minimizing the loss functions $\mathcal{L}_{\text{neat}}$, $\mathcal{L}_{\text{img}}$, and $\mathcal{L}_{\text{eik}}$ allows us to derive a geometrically meaningful but noisy 3D line cloud from multi-view images, as demonstrated in **??** using both a synthetic example and a real case from the DTU-24 scene [**?**]. The absence of explicit line matching across multiple views leads to duplication of the same 3D line segments, each with its own view-dependent prediction errors. In the following section, we discuss how this redundancy and noise, while initially seeming detrimental, actually provide a strong inductive bias towards achieving the goal of 3D wireframe reconstruction.



(a) 898 Lines (100 Views) (b) 6731 Lines (49 Views)

Figure 4. Two cases of learned noisy and redundant 3D line segments by line segment rendering. The case (a) takes the images and line segments introduced in **??**, and the case (b) is a real-world case of DTU-24 scene.

3.2. Neural 3D Junction Perceiver

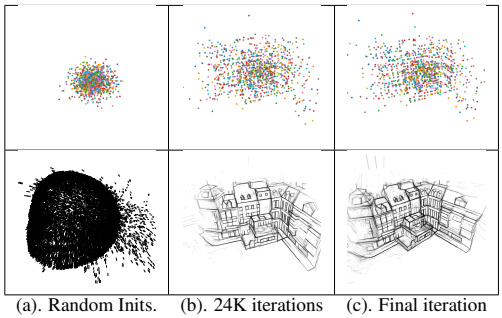
This section introduces our method to "clean up" the noisy and redundant 3D line cloud created by Neural Attraction Fields. Leveraging the relationship between 3D junctions and line segments in wireframes, we propose a neural and joint optimization approach, central to our NEAT method. Using the 3D line cloud, denoted by $\mathbf{L}_{\text{neat}}$, a query-based learning method is designed for perceiving 3D junctions (**??**) via junction-line attraction, which plays the role of distillation for 3D wireframe reconstruction.

Global 3D Junction Percieving. Our 3D line segment rendering inherits the dense representation as the density field and the radiance field. To achieve parsimonious wireframes, we propose a novel query-based design to holistically perceive a predefined sparse set of N 3D junctions by

$$Q_{N\times C} \xrightarrow{\text{MLP}} J_{N\times 3}, \tag{6}$$

where $Q_{N\times C}$ are C-dim latent queries (randomly initialized in learning). Surprisingly, as we shall show in experiments, the underlying 3D scene geometry induced synergies between $J_{N\times 3}$ and the above 3D line segment rendering integral enable us to learn a very meaningful global 3D junction perceiver.

In the absence of well-defined ground-truth for learning 3D junctions, we use the endpoints of redundant rendered 3D line segments (**??**) as noisy labels. By reshaping the line cloud $\mathbf{L}_{\text{neat}}$ into $\mathbf{J}_{\text{neat}} \in \mathbb{R}^{2M\times 3}$, our process involves two steps: (1) clustering $\mathbf{J}_{2M\times 3}$ using DBScan to yield pseudo 3D junctions $\mathbf{J}_{\text{cls}} \in \mathbb{R}^{m\times 3}$ with $m < 2M$ clusters; (2) applying bipartite set-to-set matching between the perceived junctions $J_{N\times 3}$ (**??**) and $\mathbf{J}_{\text{cls}}$ using the Hungarian algorithm. The matching cost is based on the ℓ_2 norm between 3D points. We define $\mathcal{J} = \{(J_k, \mathbf{J}_{i_k}^{\text{cls}}) | k = 1, \ldots, K\}$ as the set of matched junctions, where $K = \min(N, m)$, and i_k is the index of the k-th matched pseudo label $\mathbf{J}_{i_k}^{\text{cls}}$. Then, our goal is to minimize the distance



(a). Random Inits. (b). 24K iterations (c). Final iteration

Figure 5. **Optimization Process** of 3D Junction Perceiving (top) from the noisy 3D line cloud (bottom) on the DTU-23 scene.

between matched junctions and their corresponding pseudo labels using

$$\mathcal{L}_{jc}(J_k, \mathbf{J}_k) = \|J_k - \mathbf{J}_k\|_1 + \lambda \cdot \|\Pi(J_k) - \Pi(\mathbf{J}_k)\|_1 , \quad (7)$$

where $\Pi()$ is the 3D-to-2D projection, and λ the trade-off parameters (e.g. 0.01 in our experiments).

Joint Optimization. In our final implementation, we refine our approach by jointly optimizing the NEAT field and the 3D junction perceiver. This optimization involves minimizing all aforementioned loss functions in a weighted sum, which allows for dynamic distillation of 3D junctions from the noisy 3D line cloud generated by the NEAT field. The total loss function, $\mathcal{L}_{\text{total}}$, is expressed as:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{img}} + \lambda_e \mathcal{L}_{\text{eik}} + \lambda_n \mathcal{L}_{\text{neat}} + \lambda_j \mathcal{L}_{jc}, \quad (8)$$

where $\mathcal{L}_{\text{img}}$ and $\mathcal{L}_{\text{eik}}$ are as defined in [**?**]. The weights λ_n, λ_e, and λ_j are all set to 0.01. As depicted in Figure **??**, this optimization process continually refines the global 3D junctions by extracting them from the 3D line cloud of NEAT field at each iteration, all trained from scratch.

3.3. NEAT Wireframe Distillation using Junctions

After training, we acquire N 3D junctions $J_{N\times 3}$ and M 3D line segments $\mathbf{L}_{\text{neat}} \in \mathbb{R}^{M\times 2\times 3}$. The line segments are indexed by 3D junctions based on their spatial relationship, assigning each segment $\mathbf{L}^i_{\text{neat}}$ a global ID within $(u, v) \in \{0, \dots, N-1\} \times \{0, \dots, N-1\}$, with $u < v$. Indexing is informed by endpoint distances. Segments with angular distances over 10 degrees or perpendicular distances above 0.01 units in 3D space are deemed "too far" and removed, ensuring alignment with the 3D junctions. Further details are available in **??**.

Endpoint indexing significantly reduces the number of 3D line segments. Segments like $(\mathbf{x}^s_i, \mathbf{x}^t_i)$ and $(\mathbf{x}^s_j, \mathbf{x}^t_j)$ sharing the same junction IDs $(u_i, v_i) = (u_j, v_j) = (u, v)$ are grouped under one global line segment defined by (u, v). We represent these grouped segments as $\mathbf{L}_{u,v} = \{\mathbf{l}^1_{u,v}, \dots, \mathbf{l}^T_{u,v}\} \in \mathbb{R}^{T\times 2\times 3}$, where $T = T_{u,v}$ indicates the count of segments in $\mathbf{L}_{u,v}$. For convenience, the global line segment for index (u, v) is denoted as $\mathbf{l}^0_{u,v} = (J_u, J_v)$. Junctions not indexed by more than one line segment in $\mathbf{L}_{\text{neat}}$ are marked as inactive.

The 3D Wireframe. After indexing the 3D line segments $\mathbf{L}_{\text{neat}}$ with global junctions, we form the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ composed of active global junctions and their index pairs. To refine this graph, we remove isolated junctions and line segments, resulting in the final 3D wireframe $\mathcal{G}$, where $\mathcal{V} \subset \mathbb{R}^3$ represents the vertices and $\mathcal{E} \subset \mathbb{Z}^2$ the edges.

Least Square Optimization of 3D Junctions. Given that 3D junctions are derived from a noisy 3D line cloud, we optimize them by leveraging their relationships with global line segments (J_u, J_v) and corresponding 3D line segments $\mathbf{L}(u, v)$. This alignment aims to match junctions with their supporting 3D line segments. The optimization is framed as a non-linear least squares problem with the cost function $\mathcal{L}(J)$, defined as:

$$\mathcal{L}(J) = \sum_{(u,v)} \sum_{i=1}^{T_{u,v}} d_{\text{ang}}(\mathbf{l}^0_{u,v}, \mathbf{l}^i_{u,v})^2 + d_{\text{perp}}(\mathbf{l}^0_{u,v}, \mathbf{l}^i_{u,v})^2, \quad (9)$$

where d_{ang} and d_{perp} represent the angular and perpendicular distances between two 3D line segments, respectively. The optimization details are provided in **??**.

The Final Wireframe. After leveraging the least square optimization to adjust the position 3D junctions, we further remove the isolated junctions and the isolated line segments in $\mathcal{G}$ of which their projection to 2D space are not supported by any line segment of the 2D wireframe observations. Here, the criterion of the support is defined by the minimum angular distance and the perpendicular distance between the projected 3D line segment and the 2D line segment is not more than 10 degree and 5 pixels, respectively. After the filtering, we adjust the actived 3D junctions by querying SDF, see **??** for details.

4. Experiments

In experiments, we mainly testify our NEAT on two datasets (*i.e.*, the DTU dataset [**?**] and the BMVS dataset [**?**]) for real-scene multiview images with known camera poses. In addition to those two datasets, in **??**, the experiments on the ABC dataset [**?**] evaluated by using the 3D wireframe annotations further verified our proposed NEAT approach for the 3D wireframe representation.

4.1. Baselines, Datasets and Evaluation Metrics

We take the well-engineered Line3D++ [**?**] and the recently-proposed LiMAP [**?**] as the baselines to make quantative and qualitative comparisons, all of which are mainly designed for line-based 3D reconstruction based on two-view line matching results. Because our target is

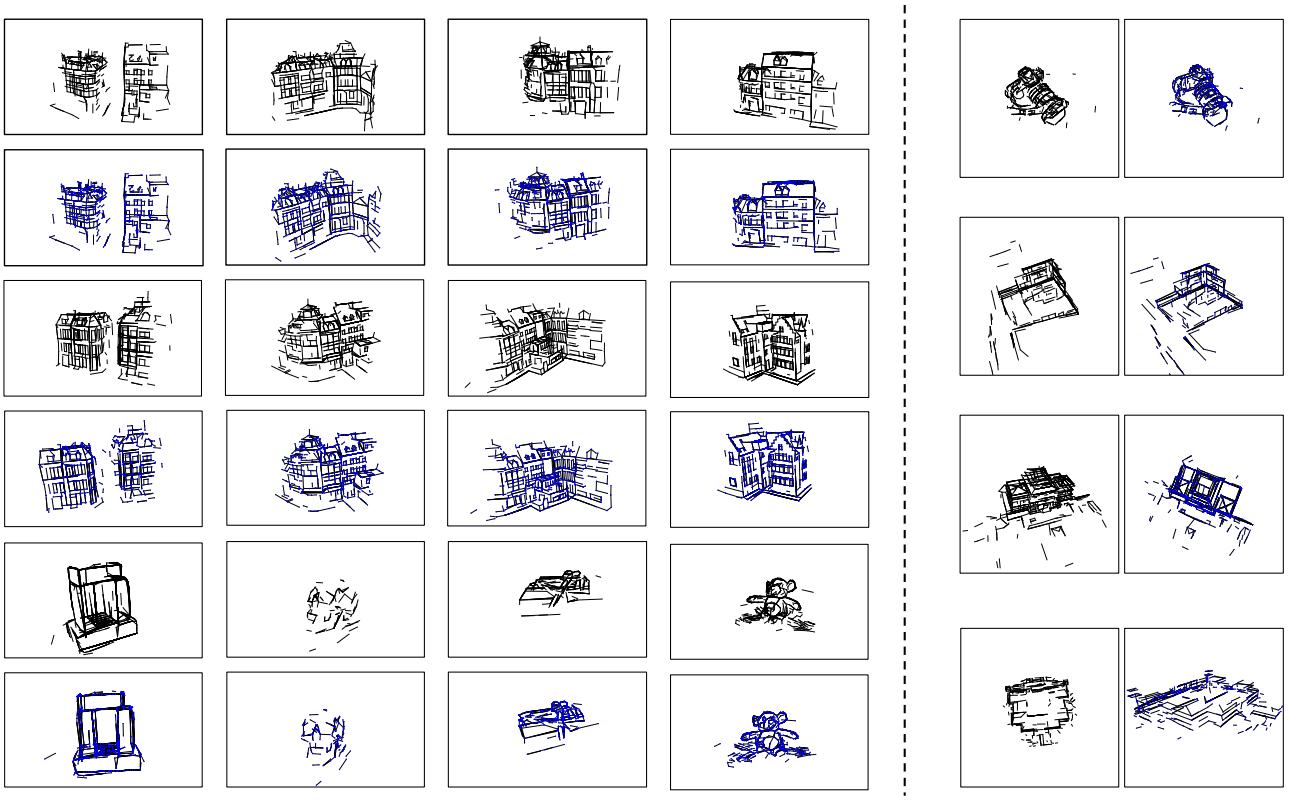


Figure 6. **Visualization of 3D Wireframe Reconstruction** on the 12 scenes from the DTU dataset [**?**] and the 4 scenes from the BlendedMVS dataset [**?**]. For each scene, we show its line segment view (by hiding the junctions) in black, and the wireframe view by coloring the junctions in blue. For the comparison, please see our video.

Table 1. Evaluation Results on the DTU and BlendedMVS datasets for the reconstructed 3D wireframes. ACC-J and ACC-L are the evaluation for junctions and line segments. For Line3D++@HAWP, LiMAP and ELSR, all the endpoints of line segments are treated as junctions.

	NEAT (Ours)					LiMAP [**?**]				Line3D++@HAWP			
Scan	ACC-J ↓	ACC-L	COMP-L ↓	#Lines ↑	#Junctions	ACC-J ↓	ACC-L	COMP-L ↓	#Lines ↑	ACC-J ↓	ACC-L ↓	COMP-L ↓	#Lines ↑
DTU Dataset [**?**]													
Avg.	**0.7718**	**0.8002**	**6.1064**	624	503	1.0944	0.8547	7.7756	231	0.9019	0.8133	8.5086	249
16	0.8263	0.7879	5.4135	729	554	1.0385	0.7898	6.0420	335	0.7957	0.6992	6.9052	388
17	0.7754	0.6695	5.0498	738	546	1.1015	0.8804	5.8212	388	0.8816	0.7778	7.6257	395
18	0.6429	0.6868	5.3796	701	596	0.9950	0.8253	7.0154	287	0.7894	0.7528	7.7082	305
19	0.6989	0.6923	4.6529	809	510	0.7689	0.7110	7.9461	160	0.6815	0.7953	6.9776	330
21	0.9042	0.6923	4.6529	809	571	1.1011	0.8884	5.9821	319	0.9064	0.7953	6.9776	330
22	0.6343	0.6910	5.0871	758	596	0.8998	0.7353	6.8567	281	0.7494	0.7079	7.8014	328
23	0.5882	0.6193	5.5992	771	597	1.0561	0.8293	6.5078	377	0.8005	0.7356	8.2679	320
24	0.6386	0.5944	5.9104	860	549	1.0314	0.8293	6.5078	377	0.7940	0.6807	7.6886	366
37	1.4815	1.0856	7.5362	420	405	1.2721	1.2352	8.6413	120	1.1796	1.0287	10.2244	60
40	0.6298	1.0354	8.7825	137	469	1.2108	0.8327	9.9988	41	0.8486	0.6877	10.1206	83
65	0.7212	1.0354	8.7825	137	171	1.0469	0.5071	11.1936	7	1.1008	1.0697	11.1519	23
105	0.7204	1.0127	6.4296	621	478	1.6108	1.1929	10.7943	90	1.2957	1.0286	10.6539	61
BlendedMVS Dataset [**?**]													
Avg.	**0.1949**	**0.1802**	**6.4621**	602	514	0.3712	0.3169	6.9415	313	0.3743	0.3545	6.8760	724
1	0.0365	0.0404	3.7253	653	565	0.0488	0.0651	5.0457	226	0.0682	0.0650	5.3625	691
2	0.1715	0.1585	8.2943	328	343	0.3478	0.2817	8.7663	195	0.4327	0.4174	8.8864	396
3	0.2564	0.2165	7.5600	931	664	0.3796	0.3162	7.5366	467	0.3795	0.3582	7.3192	931
4	0.3153	0.3055	6.2686	509	483	0.7086	0.6045	6.4174	365	0.6171	0.5774	5.9359	876

3D wireframe reconstruction instead of 3D line segment reconstruction, for fair comparisons, we use HAWPv3 [**?**] as the alternative for 2D detection in the use of Line3D++ and LiMAP. For those baselines, we use their official implementation for 3D line segments reconstruction.

DTU [?] and BlendedMVS [?] Datasets. These two

datasets were mainly designed for multiview stereo (MVS), but they are applicable to 3D wireframe reconstruction as they provided high-quality 3D point clouds as annotations. For our experiments, we run our method on 12 scenes from DTU datasets and 4 scenes from BlendedMVS datasets. For the quantitative evaluation, we first convert the reconstructed wireframe model by NEAT (or the 3D line segment model by baselines) into the point cloud by sampling 32 points on each line segment and computing the ACC metric to make comparisons. Because the reconstructed 3D wireframes (and line segments) are rather sparse than the dense surfaces, the COMP metric used for comparison would be less informative than ACC. Therefore, we additionally use the number of reconstructed 3D line segments and junctions as the reference of completeness.

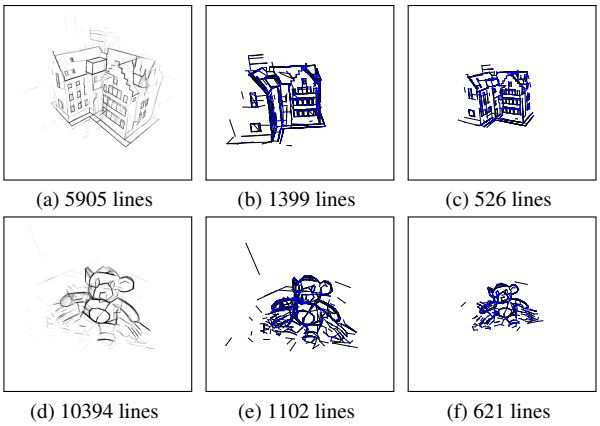
4.2. Main Comparisons

We compare our NEAT approach with three baselines on the scenes from DTU and BlendedMVS datasets, which include both the straight-line dominant scenes and some curve-based ones. In Tab. **??**, we quantitatively report the ACCs for both 3D line segments and their junctions (or endpoints), as well as the number of geometric primitives. Compared to the baseline *Line3D++@HAWP* that takes the same 2D wireframes as input, our NEAT significantly outperforms it in all metrics, which indicates that NEAT is able to yield more accurate and complete 3D reconstruction results than L3D++ for HAWP inputs. Fig. **??** visualizes the reconstructed 3D wireframes for the evaluated scenes on the DTU and BlendedMVS datasets.

4.3. Ablation Studies

In our ablation study, two scenes (*i.e.*, DTU-24 and DTU-105) are used as representative cases to discuss our NEAT approach. In the first, we qualitatively show the NEAT lines (*i.e.*, raw output of 3D line segments by querying the NEAT field), the initial reconstruction by binding the queried NEAT lines to global junctions, and the final reconstruction results by the visibility checking. Then, we discuss our NEAT approach in the following two aspects: (1) the parameterization of NEAT Fields and (2) the view dependency issue for junction perceiving. For more ablation studies for the hyperparameter setting, especially for the number of global junctions, please refer to **??**.

The Process of Wireframe Reconstruction. Fig. **??** shows the three components for wireframe reconstruction. In the first component, we query all possible 3D line segments from the optimized NEAT field. In the second component, the queried 3D line segments are binding to the global junctions. In the third step, by leveraging the non-linear optimization and a relaxed visibility checking, the unstable 3D line segments are removed from the initial wireframe models. Benefitting from the proposed novel mechanism of learning global 3D junctions, we largely simplified the way of removing duplicated and unreliable line segments without using either the known 3D points or the complicated line segment matching.



(a) 5905 lines (b) 1399 lines (c) 526 lines

(d) 10394 lines (e) 1102 lines (f) 621 lines

Figure 7. Left: NEAT lines (by coordinate MLP); Middle: initial wireframes (without visibility checking); Right: the final wireframes (with visibility checking) in the right.

Table 2. Quantatively evaluation results for ablation studies on the DTU-24 and DTU-105 scenes.

	View Dir.	Clustering	ACC (J)↓	ACC (L)↓	# Lines	# Junctions
DTU-24	No	No	0.925	0.847	744	531
	Yes	No	0.796	0.678	827	475
	Yes	Yes	**0.639**	**0.594**	**860**	**549**
DTU-105	No	No	0.822	1.209	607	499
	Yes	No	0.749	1.154	557	408
	Yes	Yes	**0.720**	**1.013**	**621**	478

Parameterization of NEAT Fields. We found that the parameterization of NEAT Fields learning is playing in a vital role in the wireframe reconstruction. Even though our NEAT field aims at representing 3D line segments by the displacement vectors of the 3D points, the localization error in the detected 2D wireframes will possibly lead to some 3D line segments that cannot be well supported by high-quality 2D detection results missing. The information on view direction is a key factor to avoid this issue and yield more complete results. According to Tab. **??**, the parameterization without the viewing directions will result in a coarser reconstruction with larger ACC errors for both 3D junctions and line segments while having fewer line segments although the number of global junctions is similar to the final model.

Clustering in Junction Perceiving. The DBScan [**?**] clustering is a key factor in accurately perceiving global junctions from the view-dependent coordinate MLP of the NEAT field. To verify this factor, we ablated the DBScan clustering to optimize MLPs on DTU-24 and DTU-105. Quantitatively reported in Tab. **??**, although the parameterization of viewing direction largely reduced the ACC errors for both reconstructed junctions and line

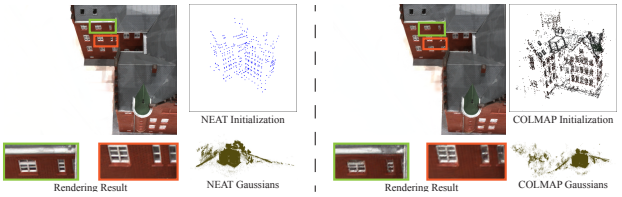


Figure 8. NEAT is applicable to 3D Gaussian Splatting framework to obtain more meaningful 3D Gaussian ellipsoids for better rendering results using 20 times fewer initial 3D points.

segments, the number of 3D junctions and line segments is also significantly reduced. When we enable the clustering during optimization, the lower-quality 3D local junctions (from the NEAT field) can be filtered, thus leading to an easy-to-optimize mode to yield more 3D junctions and line segments with fewer reconstruction errors.

4.4. NEAT for 3D Gaussian Splatting

Recently, 3D Gaussian Splatting [**?**] has become popular in neural rendering, owing to its computational efficiency and high-quality rendering. Our proposed NEAT method effectively represents 3D scenes using a limited number of junctions and line segments in wireframe format. We explored whether these reconstructed 3D junctions and line segments enhance novel view synthesis in 3D Gaussian Splatting [**?**] and found positive results. As demonstrated in **??**, the final 3D Gaussian ellipsoids, optimized using different initialization (i.e., SfM Points and NEAT junctions), show that using only 549 points from the 3D junctions can yield more accurate geometry of Gaussian ellipsoids, thus improving rendering quality. Due to space constraints, further rendering experiments using NEAT's output are detailed in the **??**.

4.5. Failure Mode and Limitations

Volume Rendering of NEAT Fields. Our method, based on VolSDF [**?**], faces inherent difficulties in inside-out scenes for neural surface rendering, similar to recent studies [**?**]. Overcoming these challenges, though possible with techniques like pre-trained monocular depth and normal maps [**?**], is beyond this paper's scope and reserved for future work.

2D Detection Results are Critical. Another critical issue is the quality of 2D wireframe detection. Failures in the HAWP model [**?**] directly impact our 3D wireframe reconstruction and parsing goals. Fig.**??** illustrates a failure case from the ScanNet[**?**] dataset, highlighting issues like motion blur affecting wireframe detection and leading to inaccuracies in 3D line segments. Despite these challenges, our global junctions (Fig. **??**) show potential in learning from blurry 2D wireframes, suggesting new insights into the relationship between junctions and line segments in wireframe representation.

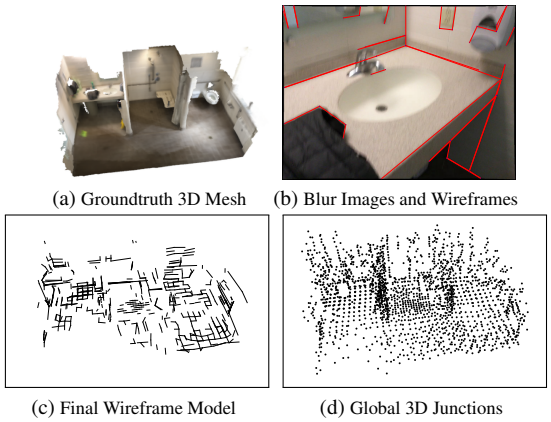


Figure 9. A Representative Failure Mode on ScanNet.

The Scalability Issue. Our proposed method is currently limited by the predefined number of 3D global junctions (*e.g.* 1024 junctions), which would be challenged in large-scale scenes that apparently contain much more 3D junctions. Though this limitation can be alleviated by leveraging a divide-and-conquer strategy like Block-NeRF [**?**], the number of junctions should be scene-dependent and be automatically determined instead of being treated as a predefined hyperparameter in the future work.

5. Conclusion

This paper studied the problem of multi-view 3D wireframe parsing (reconstruction) to provide a novel viewpoint for compact 3D scene representation. Building on the basis of the volumetric rendering formulation, we propose a novel NEAT solution that simultaneously learns the coordinate MLPs for the implicit representation of the 3D line segments, and the global junction perceiving (GJP) to explicitly learn global junctions from the randomly-initialized latent arrays in a self-supervised paradigm. Based on new findings, we finally achieve our goal of computing a parsimonious 3D wireframe representation from 2D images and wireframes without considering any heuristic correspondence search for 2D wireframes. To our knowledge, we are the first to achieve multi-view 3D wireframe reconstruction with volumetric rendering. Our proposed novel junction perceiving module opens a door to characterize the scene geometry from 2D supervision in structured point-level 3D representation.

Acknowledgment. N. Xue was partially supported by the NSFC under Grant 62101390. T. Wu was supported in part by NSF IIS-1909644. We would like to thank anonymous reviewers for their constructive suggestions. The views presented in this paper are those of the authors and should not be interpreted as representing any funding agencies.

NEAT: Distilling 3D Wireframes from Neural Attraction Fields

Supplementary Material

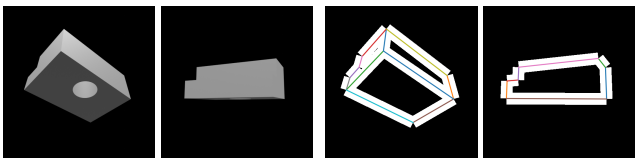
The supplementary document is summarized as follows:

- Appx. **??** gives a summary of the supplementary video.
- Appx. **??** elaborates on the technical details (*introduced in Sec. 3.2 of the main paper*) of NEAT optimization.
- Appx. **??** supplies the details for the final step of distillation for 3D wireframe reconstruction (*introduced in Sec. 3.3* of the main paper).
- Appx. **??** presents the additional experiments on the ABC dataset [**?**] to discuss the performance given the ground-truth annotations of 3D wireframes.
- Appx. **??** quantitatively reports the potential of NEAT for view synthesis with 3D Gaussian Splatting on the DTU dataset.
- Appx. **??** shows the miscellaneous stuff.

A. Video

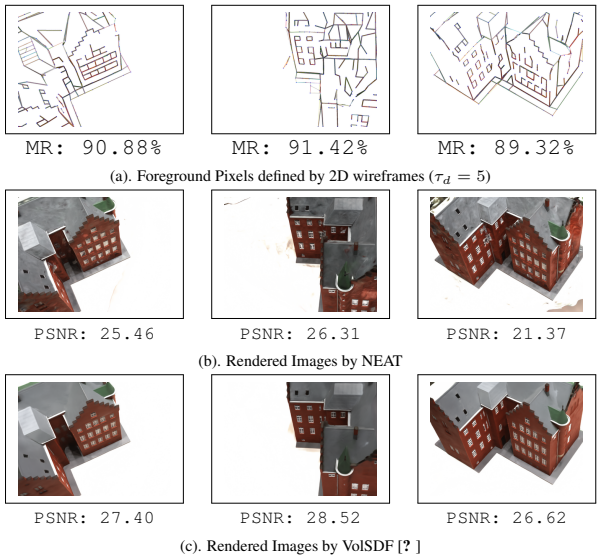
In our supplementary video, we begin by demonstrating the core concepts of our research. Using a basic object from the ABC dataset as an illustrative example, we showcase the 3D line segments learned through the NEAT field, the functionality of the global junction perceiving module, and the construction of the final 3D wireframe model. Following this, the video highlights the learning of redundant 3D line segments and the optimization process for global junctions, using the DTU-24 dataset as a case study. The video concludes with qualitative evaluations on both the DTU and BlendedMVS datasets, providing visual support to the quantitative analyses of the main paper.

B. Optimization of NEAT



(a) Input images (b) 2D Wireframes & rendering pixels

Figure 10. A toy example on the ABC dataset [**?**] for the foreground pixels defined by the detected 2D wireframes.



(a). Foreground Pixels defined by 2D wireframes ($\tau_d = 5$)

(b). Rendered Images by NEAT

(c). Rendered Images by VolSDF [**?**]

Figure 11. A comparison for volumetric rendering learned from wireframe-related rays (pixels) *vs*. the vanilla ray sampling. In (a), we show the 2D line segments detected by HAWPv3 [**?**] and the used foreground pixels in each view. "MR" denotes the mask ratio (the number of foreground pixels among all the pixels). In (b), we show the corresponding views rendered by NEAT that are learned by the foreground pixels in (a). In the bottom (c), we show the rendered images by VolSDF [**?**] as the reference. In (b) and (c), the PSNR values are marked at the bottom for each view.

B.1. Details on Line Segment Rendering

Our method renders 3D line segments based on the detected 2D wireframes in each view, distinguishing itself from conventional volume rendering approaches that utilize all pixels (rays) for rendering. As demonstrated in Fig. **??** with a toy example from the ABC dataset, only pixels with "white" colors are engaged in the rendering process of 3D line segments. This technique is inspired by the attraction field representations [**? ? ? ? ?**], where the involved pixels are determined by the perpendicular distance between a point and a line segment. We set a threshold, τ_{ray} (as mentioned in Sec. 3.1 of our main paper), to differentiate the rendering pixels as foreground while disregarding the non-rendering pixels as background. Practically, τ_{ray} is usually set to 5 for training/optimization, and reduced to 1 to minimize computational costs. We refer to this approach as *wireframe-driven ray sampling*.

To demonstrate the effectiveness of wireframe-driven ray sampling, we conducted a series of experiments on `scene 24 from the DTU dataset` [**?**]. Fig. **??** illustrates

Table 3. The influence of wireframe reconstruction results from different distance thresholds. The larger τ_d value is, the more line segments are involved in the optimization/learning.

	ACC-J↓	ACC-L↓	COMP-L↓	#Lines	#Junctions	MR	PSNR
$\tau_d = 1$	0.853	0.764	6.137	785	540	97.49%	17.79
$\tau_d = 5$	0.639	0.594	5.910	860	528	89.70%	21.55
$\tau_d = 20$	0.578	0.596	6.158	694	508	66.10%	24.68

the feasibility of optimizing coordinate MLPs using this sampling technique. As depicted in Fig. **??**(a), by masking over 80% of the pixels (using a distance threshold of 5 pixels), we can still effectively optimize coordinate MLPs, leading to the reasonable outcomes shown in Fig. **??**(b).

In addition to rendering results, we observed that increasing the distance threshold leads to a reduction in the number of line segments and junctions. As detailed in Tab. **??**, setting the distance threshold to $\tau_d = 20$ results in fewer 3D lines and junctions. Although the ACC errors are marginally reduced, there is an increase in completeness. Conversely, when the distance threshold τ_d is set to 1, a performance degradation is noted across all metrics due to insufficient supervision signals.

B.2. The Number of Global Junctions

The number of global junctions is determined heuristically to encompass all potential 3D junctions. Based on observations from both the DTU and BlendedMVS datasets, where the detected 2D line segments are in the hundreds, we set the estimated number of 3D junctions to 1024. In **??**, we present experiments conducted on the DTU-24 scene with varying numbers of junctions, denoted as N, to assess performance differences. The results indicate that increasing the number of possible global 3D junctions to a larger value (e.g., $N = 2048$) yields only a marginal increase in the count of learned 3D line segments and junctions in the final wireframe models. Conversely, a smaller N tends to result in incomplete 3D wireframe models.

N	# 2D Juncs.	# 3D Junctions	# 3D Lines	ACC-J	ACC-L	COMP-L
1024 (default)	212 (min)	549	860	0.639	0.549	5.910
$N = 128$	297 (max)	99	93	0.422	0.440	8.541
$N = 512$	258.2 (avg)	397	641	0.526	0.574	6.302
$N = 2048$		624	983	0.656	0.599	5.849

Table 4. The performance influence of wireframe reconstruction from different configuration of the number of 3D junctions during optimization.

B.3. Additional Implementation Details

Network Architecture. The coordinate MLPs used in our NEAT approach are derived from VolSDF [**?**], which contains three coordinate MLPs for SDF, the radiance field, and the NEAT field. For the MLP of SDF, it contains 8 layers with hidden layers of width 256 and a skip connection from the input to the 4th layer. The radiance field and the NEAT field share the same architecture with 4 layers with hidden layers of width 256 without skip connections. The proposed global junction perceiving (GJP) module contains two hidden layers and one decoding layer as described in the code snippets of Sec. 1 in our main paper.

Hyperparameters. The distance threshold τ_d about the foreground pixel (ray) generation is set to 5 by default.For the number of global junctions (*i.e.*, the size of the latent), we set it to 1024 on the DTU and BlendedMVS datasets. When the scene scale is larger (*e.g.*, a scene from ScanNet mentioned in Fig. 5 of the main paper), the number of global junctions is set to 2048. For DBScan [**?**], we use the implementation from `sklearn` package, set the epsilon (for the maximum distance between two samples) to 0.01 and the number of samples (in a neighborhood for a point to be considered as a core point) to 2.

C. The Final Distillation Step of NEAT

This section elaborates on the final distillation step required in our NEAT methodology for 3D wireframe reconstruction, with a particular focus on the extensive use of global junctions. We aim to provide a detailed insight into this crucial phase of the NEAT process.

To begin with, let us consider the challenge inherent in the junction-driven finalization of NEAT. As depicted in Fig. **??**, using a toy ABC scene as an example, we observe that a considerable number of 3D line segments are rendered and aggregated across different views. Concurrently, 3D junctions are dynamically distilled from the NEAT fields. While a simple approach to combine these 3D junctions with the redundant 3D line segments might seem viable, it is critical to address the potential misalignments between the junctions and line segments. To resolve this issue, we employ a least squares optimization combined with an SDF-based refinement scheme. This approach is designed to precisely adjust the position of 3D junctions, thereby ensuring an accurate and coherent reconstruction of the 3D wireframe.

C.1. Least Square Optimization

To be convenient for readers, we copy Eq. (9) in our main paper to **??**,

$$\mathcal{L}(J) = \sum_{(u,v)} \sum_{i=1}^{T_{u,v}} d_{\text{ang}}(\mathbf{l}^0_{u,v}, \mathbf{l}^i_{u,v})^2 + d_{\text{perp}}(\mathbf{l}^0_{u,v}, \mathbf{l}^i_{u,v})^2, \tag{10}$$

which is the main objective function to adjust the junction positions according to the observation from the op-

Table 5. An Ablation study of the SDF-based 3D Junction Refinement on the DTU dataset for the reconstructed 3D wireframes. ACC-J and ACC-L are the evaluation for junctions and line segments.

Scan	NEAT (Final)				NEAT (w/o Non-Linear Optimization)				NEAT (w/o SDF-based Refinement)			
	ACC-J ↓	ACC-L ↓	#Lines	#Junctions	ACC-J ↓	ACC-L ↓	#Lines	#Junctions	ACC-J ↓	ACC-L ↓	#Lines	#Junctions
Avg.	0.772	0.800	624.2	503.5	1.145	0.872	907.7	589.7	1.275	1.044	729.1	514.3
16	0.826	0.788	729	554	0.834	0.829	852	566	1.190	1.045	751	570
17	0.775	0.670	738	546	0.982	0.765	991	651	1.047	0.836	753	557
18	0.643	0.687	701	596	0.930	0.759	993	689	1.040	0.927	821	609
19	0.699	0.692	809	510	0.956	0.703	994	656	1.051	0.863	714	518
21	0.904	0.692	809	571	0.960	0.725	981	654	1.119	0.848	816	581
22	0.634	0.691	758	596	0.896	0.748	939	684	0.976	0.897	769	603
23	0.588	0.619	771	597	0.840	0.703	933	670	0.926	0.821	774	602
24	0.639	0.594	860	549	0.818	0.620	1008	618	0.872	0.748	866	556
37	1.482	1.086	420	405	1.804	1.477	636	565	2.014	1.860	440	425
40	0.630	1.035	137	469	1.342	0.808	1672	591	1.382	0.983	1241	475
65	0.721	1.035	137	171	1.582	1.178	191	221	1.631	1.340	147	185
105	0.720	1.013	621	478	1.793	1.143	702	511	2.053	1.360	657	490

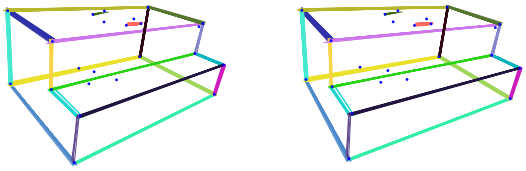


Figure 12. Two different views of the reconstruction of 3D wireframe on the toy scene of ABC dataset before the final distillation step.

timized/learned NEAT field. Here, we mathematically define the alignment cost between the junction-driven 3D line segments $\mathbf{l}^0_{u,v} = (J_u, J_v)$ and its i-th NEAT-field observation $\mathbf{l}^i_{u,v} = (\mathbf{x}^i_u, \mathbf{x}^i_v)$ by the angular cost and the perpendicular cost as follow

$$\begin{aligned} d_{\text{ang}}(\mathbf{l}^0_{u,v}, \mathbf{l}^i_{u,v}) &= 1 - |\langle \frac{J_u - J_v}{\|J_u - J_v\|}, \frac{\mathbf{x}^i_u - \mathbf{x}^i_v}{\|\mathbf{x}^i_u - \mathbf{x}^i_v\|} \rangle|, \\ d_{\text{perp}}(\mathbf{l}^0_{u,v}, \mathbf{l}^i_{u,v}) &= \left\| J_u - \text{proj}(\mathbf{l}^i_{u,v}; J_u) \right\| \\ &\quad + \left\| J_v - \text{proj}(\mathbf{l}^i_{u,v}; J_v) \right\|, \end{aligned} \tag{11}$$

where $\langle \cdot, \cdot \rangle$ is the inner product between two 3D vectors, and the function $\text{proj}(\mathbf{l}^i_{u,v}; J_v)$ projects the point J_v onto the infinite 3D line passing through the line segment $\mathbf{l}^i_{u,v}$. In Tab. **??**, we report the performance changes by disabling the non-linear optimization on the DTU dataset, which will result in inferior 3D wireframes with larger ACC errors for both junctions and line segments.

C.2. SDF-based 3D Junction Refinement

Following the non-linear optimization, we employ an SDF-based refinement scheme to further enhance the localization accuracy of junctions. Specifically, for an initial 3D junction $J_i \in \mathbb{R}^3$ and an optimized SDF $d_\Omega(\cdot)$, we refine the location of J_i using the following equation:

$$J^{\text{refined}}_i = J_i - d_\Omega(J_i) \cdot \nabla d_\Omega(J_i), \tag{12}$$

where ∇d_Ω represents the normal direction of the surface at the point J_i.

To assess the impact of this SDF-based refinement on junctions, we conducted an ablation study comparing 3D wireframe models with and without the SDF refinement. The results, presented in Tab. **??**, clearly demonstrate the necessity of this refinement step for achieving significantly improved results.

Table 6. The performance change w.r.t. the visibility threshold on the DTU dataset.

Vis	Metric	16	17	18	19	21	22	23	24	37	40	65	105	Avg.
1	ACC.↓	0.788	0.670	0.687	0.692	0.692	0.691	0.619	0.594	1.086	1.035	1.035	1.013	0.800
	COMP.↓	5.414	5.050	5.380	4.653	4.653	5.087	5.599	5.910	7.536	8.783	8.783	6.430	6.106
	Avg. Len.	22.3	23.6	26.7	27.4	27.4	22.8	26.9	27.0	27.9	23.2	23.2	27.5	25.5
	#Lines	729.0	738.0	701.0	809.0	809.0	758.0	771.0	860.0	420.0	137.0	137.0	621.0	624.2
2	ACC.↓	0.770	0.669	0.650	0.642	0.686	0.678	0.604	0.585	1.251	0.755	1.005	1.011	0.776
	COMP.↓	5.493	5.067	5.043	5.562	4.742	5.208	5.670	6.032	7.517	7.027	9.131	6.643	6.095
	Avg. Len.	22.3	23.6	24.4	27.0	27.6	22.8	26.9	27.1	27.4	49.8	22.8	27.0	27.4
	#Lines	711.0	729.0	789.0	667.0	784.0	737.0	756.0	840.0	391.0	1140.0	124.0	572.0	686.7
3	ACC.↓	0.729	0.642	0.640	0.629	0.652	0.639	0.590	0.575	1.188	0.748	0.909	0.981	0.743
	COMP.↓	5.551	5.095	5.117	5.742	4.843	5.357	5.720	6.113	7.473	7.182	9.076	6.785	6.171
	Avg. Len.	22.5	23.7	24.5	27.2	27.8	22.7	26.9	27.2	27.7	49.9	22.8	26.9	27.5
	#Lines	689.0	708.0	765.0	642.0	751.0	708.0	748.0	826.0	371.0	1091.0	112.0	544.0	662.9
4	ACC.↓	0.704	0.619	0.623	0.617	0.607	0.632	0.583	0.556	1.118	0.735	0.891	0.945	0.719
	COMP.↓	5.572	5.256	5.222	5.838	5.021	5.458	5.825	6.168	7.612	7.164	9.220	7.004	6.280
	Avg. Len.	22.5	23.8	24.8	27.5	28.0	22.9	27.0	27.3	27.7	50.5	22.8	26.3	27.6
	#Lines	672.0	679.0	737.0	617.0	723.0	683.0	721.0	806.0	347.0	1052.0	97.0	501.0	636.3

C.3. Visibility Checking

As detailed in Sec. 3.3 of the main paper, we evaluate the reconstructed 3D line segments by projecting them onto 2D images from each view. This process involves computing both the angular and perpendicular distances between the projected 3D line segments and the detected 2D line segments. A 3D line segment is considered to be supported by a 2D detection if it aligns within an angular distance of 10 degrees and a perpendicular distance of 5 pixels, with a minimum overlap ratio of 50%. This

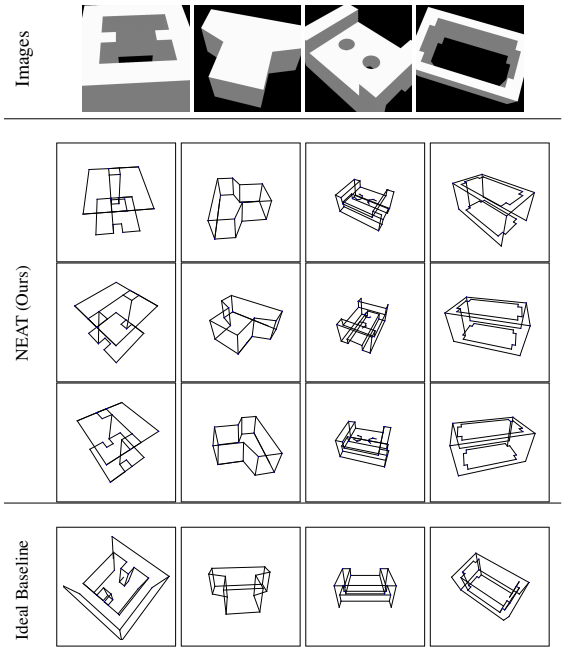


Figure 13. Qualitative Comparisons on ABC objects.

methodology allows us to determine the visibility of each 3D line segment and to filter out those that are invisible as false alarms.

In our standard approach, the visibility threshold for each line segment is set to 1, aiming to achieve a more complete reconstruction. Moreover, we explore the impact of varying this visibility threshold from 1 to 4 on the DTU dataset. The findings, as summarized in **??**, indicate that increasing the visibility threshold results in an improvement in the ACC metric, while the COMP metric increases.

D. Experiments on the ABC Dataset

Because the 3D wireframe annotations are very difficult to obtain for real scene images, to better discuss the problem of 3D wireframe reconstruction and analyze our proposed NEAT approach, we conduct experiments on objects from ABC Datasets as it provides 3D wireframe annotations.

Data Preparation. We use Blender [**?**] to render 4 objects from the ABC dataset. The object IDs are mentioned in Tab. **??**. For each object, we first resize it into a unit cube by dividing the size of the longest side and then moving it to the origin center. Then, we randomly generate 100 camera locations, each of which is distant from the origin by $\sqrt{1.5^2 + 1.5^2} \approx 2.1213$ units. The setting of the distance, $\sqrt{1.5^2 + 1.5^2}$, is from our early-stage development for the rendering, in which we set a camera at $(0, 1.5, 1.5)$ location. By setting the cameras to look at the origin $(0, 0, 0)$, we obtain 100 camera poses. Considering the fact that the ABC dataset is relatively simple, we set the focal length to 60.00mm to ensure the object is slightly occluded for rendering images. The sensor width and height of the camera in Blender are all set to 32mm. The ground truth annotations of the 3D wireframe are from the corresponding `STEP` files. For the simplicity of evaluation, we only keep the straight-line structures and ignore the curvature structures to obtain the ground truth annotations. The rendered images are with the size of 512×512.

Baseline Configuration. Fig. **??** illustrates the rendered input images for the used four objects. Because the rendered images are textureless and with planar objects, the dependency of those baselines on the correspondence-based sparse reconstruction by SfM systems [**?**] is hardly satisfied to produce reliable line segment matches for 3D line reconstruction. Accordingly, we set up an ideal baseline instead of using Line3D++ [**?**] and LiMAP [**?**] for comparison. Specifically, we first detect the 2D wireframes for the rendered input images and then project the junctions and line segments of the ground-truth 3D wireframe models onto the 2D image plane. For the 2D junctions, if a projected ground-truth junction can be supported by a detected one within 5 pixels in any view, we keep the ground-truth junction as the reconstructed one in the ideal case. For the 2D line segments, we compute the minimal value for the distance of the two endpoints of a detected line segment to check if it can support a ground-truth 3D line. The threshold is also set to 5 pixels. Then, we count the number of reconstructed 3D line segments and junctions in such an ideal case.

Evaluation Metrics. For our method, we compute the precision and recall for the reconstructed 3D junctions and line segments under the given thresholds. Because the objects (and the ground-truth wireframes) are normalized in a unit cube, we set the matching thresholds to $\{0.01, 0.02, 0.05\}$ for evaluation. For the matching distance of line segments, we use the maximal value of the matching distance between two endpoints to identify if a line segment is successfully reconstructed under the specific distance threshold. For the ideal baseline, we report the number of ground-truth primitives (junctions or line segments), the number of reconstructed primitives, and the reconstruction rate.

Results and Discussion. Tab. **??** quantitatively summarizes the evaluation results and the statistics on the used scenes. As it is reported, our NEAT approach could accurately reconstruct the wireframes from posed multiview images. The main performance bottleneck of our method comes from the 2D detection results. As shown in the ideal baseline, by projecting the 3D junctions and line segments

ID		Evaluation Results						Ideal Baseline		
		$P_{0.01}$	$P_{0.02}$	$P_{0.05}$	$R_{0.01}$	$R_{0.02}$	$R_{0.05}$	#GT	# Reconstructed	Recon. Rate
4981	J	0.706	0.765	0.882	0.750	0.812	0.938	32	28	0.875
	L	0.758	0.758	0.758	0.521	0.521	0.521	48	41	0.854
13166	J	0.889	0.889	0.889	1.000	1.000	1.000	16	16	1.000
	L	1.000	1.000	1.000	1.000	1.000	1.000	24	24	1.000
17078	J	0.400	0.629	0.686	0.583	0.917	1.000	24	23	0.958
	L	0.408	0.653	0.714	0.556	0.889	0.972	36	32	0.889
19674	J	0.969	1.000	1.000	0.969	1.000	1.000	32	32	1.000
	L	0.969	1.000	1.000	0.969	1.000	1.000	48	40	0.833

Table 7. Evaluation Results and some Statistics on ABC objects. In each object, we evaluate the precision and recall rates for junctions (J) and line segments (L). For the ideal baseline, we count the number of ground-truth primitives, the number of reconstructed 3D primitives, and the reconstruction rate in the ideal baseline.

into the image planes to obtain the ideal 2D detection results, the 2D detection results by HAWPv3 [**?**] did not perfectly hit all ground-truth annotations. Furthermore, suppose we use the hit (localization error is less than 5 pixels) ground truth for 3D wireframe reconstruction, there is a chance to miss some 3D junctions and more 3D line segments. In this sense, given a relaxed threshold of the reconstruction error for precision and recall computation, our NEAT approach is comparable with the performance of the ideal solution. For the first object (ID 4981), because of the severe self-occlusion, some line segments are not successfully reconstructed for both the ideal baseline and our approach. For object 17078, our NEAT approach reconstructed some parts of the two circles that are excluded from the ground truth, which leads to a relatively low precision rate. Fig. **??** also supported our results.

E. 3D Gaussians with NEAT Junctions

In this section, we extend the application of our NEAT framework to 3D Gaussian Splatting, as proposed by Kerbl et al. [**?**], by substituting the initial point cloud derived from Structure-from-Motion (SfM) with the junctions identified by NEAT. This experiment is designed to showcase the efficacy of NEAT junctions as a compact initialization method for 3D Gaussian Splatting. Using only a few hundred points, our NEAT junctions demonstrate an enhanced fitting ability on the DTU dataset, as evidenced by improved metrics in both Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM).

The experimental results on 12 scenes from the DTU dataset are detailed in **??**. It is observed that by initializing the 3D Gaussians with NEAT junctions, there is a notable improvement in performance: PSNR increases by 0.38 dB and SSIM improves by 0.0003 points. This finding underscores the effectiveness of NEAT junctions in providing a more precise and compact starting point for 3D Gaussian Splatting.

Table 8. Quantitative comparison between the NEAT junctions and SfM points for the initialization of 3D Gaussian Splatting on the DTU dataset.

Scene ID	NEAT Junctions					SfM Points (by COLMAP [?])				
	PSNR ↑	SSIM ↑	#Points (init)	#Points (7k)	#Points (30k)	PSNR ↑	SSIM ↑	#Points (init)	#Points (7k)	#Points (30k)
DTU-16	**28.7** (+0.7)	**0.889** (+0.006)	554	603k	1,496k	28.0	0.883	22k	558k	1,048k
DTU-17	**29.2** (+0.5)	**0.898** (+0.005)	546	903k	2,279k	28.7	0.893	24k	893k	1,305k
DTU-18	**29.3** (+0.4)	**0.901** (+0.004)	596	629k	1,234k	28.9	0.897	18k	581k	1,078k
DTU-19	**29.6** (+0.4)	0.893 (-0.001)	510	475k	1,140k	29.2	**0.894**	19k	561k	756k
DTU-21	**28.7** (+0.2)	**0.898** (+0.004)	571	725k	1,657k	28.5	0.894	19k	698k	1,528k
DTU-22	**29.1** (+0.2)	**0.892** (+0.005)	596	641k	1,455k	28.9	0.887	21k	615k	1,113k
DTU-23	**28.4** (+0.4)	**0.886** (+0.006)	597	974k	2,243k	28.0	0.880	25k	850k	1,667k
DTU-24	**31.1** (+0.9)	**0.909** (+0.008)	549	587k	1,181k	30.2	0.901	13k	528k	852k
DTU-37	**28.2** (+0.5)	**0.875** (+0.000)	405	420k	1,180k	27.7	**0.875**	27k	409k	713k
DTU-40	**30.6** (+0.2)	**0.862** (+0.002)	422	520k	1,403k	30.4	0.860	32k	515k	1,070k
DTU-65	**32.4** (+0.2)	0.855 (-0.001)	171	139k	294k	32.2	**0.856**	11k	150k	208k
DTU-105	30.8 (-0.1)	0.852 (-0.001)	478	165k	238k	**30.9**	**0.853**	23k	169k	216k
Avg.	**29.68** (+0.38)	**0.884** (+0.003)	499.58	565k	1,317k	29.30	0.881	21k	544k	963k

F. Miscellaneous

F.1. Evaluation Metrics

The Definition of ACC and COMP Metrics. We follow the official evaluation protocol of the DTU dataset [**?**] to compute the reconstruction accuracy (ACC) and completeness (COMP), which is defined to

$$\mathrm{ACC} = \underset{\mathbf{p}\in P}{\mathrm{mean}} \left(\min_{\mathbf{p}^*\in P^*} \|\mathbf{p}-\mathbf{p}^*\| \right), \tag{13}$$

and

$$\mathrm{COMP} = \underset{\mathbf{p}^*\in P^*}{\mathrm{mean}} \left(\min_{\mathbf{p}\in P} \|\mathbf{p}-\mathbf{p}^*\| \right), \tag{14}$$

where P and P^* are the point clouds sampled from the predictions and the ground truth mesh.

F.2. Information of Used BlendedMVS Scenes

The scene IDs and their MD5 code of the BlendedMVS scenes are:

- Scene-01: `5c34300a73a8df509add216d`
- Scene-02: `5b6e716d67b396324c2d77cb`
- Scene-03: `5b6eff8b67b396324c5b2672`
- Scene-04: `5af28cea59bc705737003253`