END-TO-END NEURON TREE TRACING WITH HISTORICAL TRAJECTORIES

Anonymous authors
Paper under double-blind review

ABSTRACT

Reconstructing neuronal trees from 3-D microscopy requires recovering branching connectivity from weak and ambiguous local image evidence. A false junction or missed branch can corrupt an entire downstream arbor. We introduce TreeTracer, an end-to-end autoregressive model that expands a point-and-direction prompt into a neuronal subtree. An elegant end-to-end design enables the joint optimization of neuronal tree geometry and topology predictions through gradient-based learning. A causal decoder encodes historical trajectory features for more accurate topology prediction and employs local–global cross-attention to extract visual evidence more efficiently. On block-level subtree reconstruction, TreeTracer achieved Fiber F1 scores of 58.4% on CWMBS and 66.4% on CORAL, compared with 51.8% and 61.0% for the strongest competing methods, respectively. On brain-wide neuron reconstruction, it reached 41.6%, compared with 35.1% for the strongest competing method.

1 INTRODUCTION

Reconstructing neuronal trees from 3-D microscopy images provides the morphological basis for studying neuronal diversity and circuit organization (Manubens-Gil et al., 2023). The desired output is a spatially embedded tree: its usefulness depends on both where neurites extend and how they connect. A false junction between nearby fibers can reassign an entire downstream arbor, corrupting many root-to-leaf paths despite affecting only a small image region. Recognizing thin, faint, and densely interwoven neurites therefore requires integrating local image evidence with long-range trajectory and visual context.

A common approach first predicts neuronal foreground and then recovers connectivity with a separate reconstruction algorithm (Li & Shen, 2020; Liu et al., 2024b). This separation propagates missed or spurious foreground into broken paths or false connections. Connectivity-aware segmentation objectives improve tubular continuity (Shit et al., 2021; Qi et al., 2023), but extracting an explicit neuronal tree still requires decisions about which points belong to the same path and where paths branch. Point-based methods make the output sparse, yet also separate point prediction from connectivity construction or refinement (Wang et al., 2024; Zhao et al., 2023). Hand-designed connection rules remain vulnerable where weak signals and closely apposed fibers make spatial proximity an unreliable guide to parent–child relationships.

Learning-based tracers instead follow neurites sequentially. Representative methods learn local progression from image evidence (Chen et al., 2022; Liu et al., 2024a); NETracer further predicts connective edges to guide successor selection (Liu et al., 2025). These methods learn to predict only geometric information,, such as the forward direction or the next point's location, while still relying on hand-crafted rules to identify topology-related events such as branching and termination. As a result, they do not fully address the challenge of accurately identifying local topology.

To address these challenges, we introduce **TreeTracer**, an end-to-end neuron tree tracing model that learns tree-expansion decisions for branching neuronal structures in 3-D microscopy. Given an image crop and a point-and-direction prompt, the model directly generates the corresponding local subtree. An image encoder provides visual features, while trajectory tokens encode previously generated positions, directions, and node types. A causal decoder jointly predicts a positional offset, outgoing directions, and a node type specifying continuation, branching, or termination. At a branch, child trajectories inherit the preceding history and share the generated junction. Thus, parent–child

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

relationships arise from learned expansion decisions during tracing. This yields end-to-end image-to-subtree prediction within each crop; larger reconstructions use sliding-window strategy.

This formulation couples two requirements: efficient encoding and evidence selection in dense 3D images, and correction of positional errors that accumulate during tracing. Thin neurites demand precise local features, whereas their extended trajectories require context beyond a small neighborhood. We combine local cross-attention around the proposed next position with global cross-attention to read compact context tokens. These context tokens summarize the current crop through attention-based aggregation, inspired by latent bottleneck architectures (Jaegle et al., 2021). The decoder thereby accesses detailed spatial evidence and broader context without repeatedly attending to the full dense voxel grid.

Discrete directions and fixed-length advances provide coarse proposals for tracing, but introduce geometric errors that can accumulate along a trajectory and displace subsequent image queries. TreeTracer refines each proposal with a continuous positional offset conditioned on image evidence and trajectory context. We train this refinement using perturbed trajectory positions and clean next-position targets, enabling correction of both coarse-step errors and existing positional drift.

Our contributions are:

- An elegant end-to-end model jointly predicts geometric and topological information for the next point, enabling a single prompted trace to expand into a branching subtree.
- A causal decoder fuses historical trajectory features with high-resolution local and compact global image features to better capture key local topological cues.
- On CWMBS and CORAL, our method outperforms state-of-the-art methods in local subtree prediction and brain-wide reconstruction of complete neurons.

2 Related Work

2.1 Traditional Non-learning-based Reconstruction Methods

Classical neuron reconstruction combines image-dependent path costs with geometric constraints and pruning rules. APP constructs an over-complete tree from seed-rooted geodesic paths and removes redundant components (Peng et al., 2011). APP2 introduces a gray-weighted distance transform, fast-marching tree construction, and hierarchical pruning (Xiao & Peng, 2013). neuTube combines seed-based branch tracing with shortest-path connection and interactive editing within an SWC-based framework (Feng et al., 2015). Kimimaro extracts skeletons from labeled 3D segmentations using a TEASAR-derived algorithm that iteratively traces penalized shortest paths and invalidates nearby voxels (Silversmith et al., 2021). These reconstruction procedures require no training but rely on hand-crafted image-processing and linking rules.

2.2 Two-stage Reconstruction Methods

Two-stage reconstruction first segments neuronal foreground, then extracts an explicit tree. V-Net provides a 3D segmentation architecture trained with a Dice-based objective to address foreground–background imbalance (Milletari et al., 2016). For neuron segmentation, ADTL-Net selects decoder parameters according to image appearance to accommodate heterogeneous signal and noise (Liu et al., 2024b), while GBP-Net fuses broader spatial context with high-resolution features through a dual U-Net architecture (Yan et al., 2026). Other methods specifically optimize tubular structure segmentation: clDice measures skeleton–mask overlap and provides a differentiable soft-clDice loss to encourage connectivity (Shit et al., 2021); DSCNet combines dynamic snake convolutions, multi-view feature fusion, and a persistent-homology-based continuity loss to capture thin, tortuous structures (Qi et al., 2023). These advances improve dense predictions, but using them for neuron reconstruction still requires a separate step to recover explicit nodes and connections.

Point-set methods follow a related two-stage paradigm, predicting sparse skeletal points before constructing connectivity. PointNeuron initializes connections using tracers such as APP2 and refines them with a graph auto-encoder (Zhao et al., 2023). Wang et al. (2024) predicts points with a Transformer, then constructs connections using a separate rule-based module. Thus, replacing dense masks with sparse points does not remove the dependence on subsequent connectivity inference.

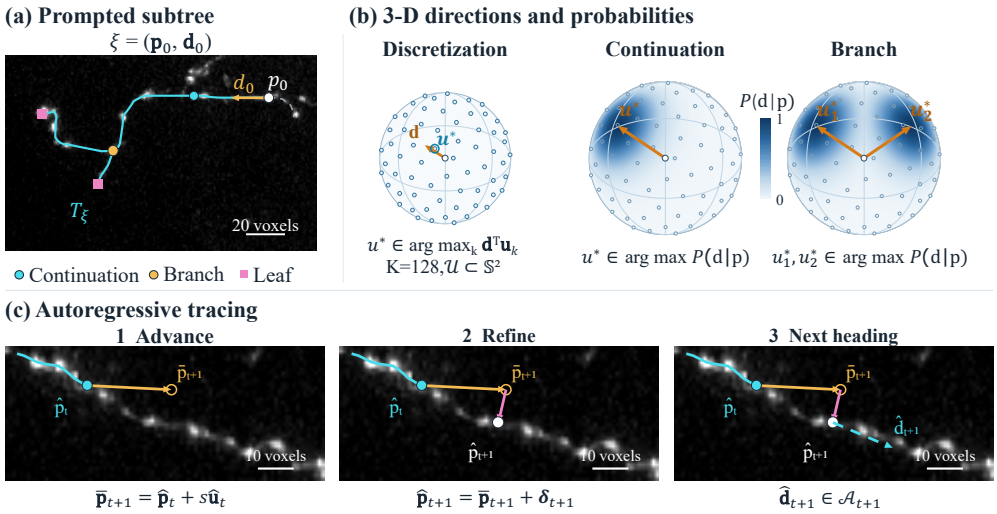

Figure 1: **Neuron tree tracing. (a)** A point-and-direction prompt $(\mathbf{p}_0, \mathbf{d}_0)$ selects the target subtree (cyan). **(b)** A spherical direction codebook supports unimodal continuation and multimodal branch predictions. **(c)** Each step advances along a quantized direction, refines the position with a continuous offset, and predicts outgoing directions for subsequent tracing.

2.3 Learning-based Tracing Methods

Existing learning-based tracers advance one path at a time without explicitly detecting branch points to expand multiple child paths. The deep reinforcement learning (DRL) tracker of Dai et al. (2019) requires manually specified start and end points. SPE-DNR traces from multiple seeds and directly connects paths when they meet (Chen et al., 2022). DeepBranchTracer and NETracer also trace from multiple seeds but stop upon reaching existing paths (Liu et al., 2024a; 2025). Both rely on predicted centerline maps for off-centerline stopping decisions, making them sensitive to errors in those maps and prone to premature termination.

3 Method

We first formulate neuron reconstruction as subtree generation from a 3-D image and a point-and-direction prompt (Section 3.1). We then introduce our end-to-end tree tracing framework, which jointly predicts the geometric and topological attributes of the next points (Section 3.2). Section 3.3 describes the model to implement this framework. Finally, Section 3.4 presents the training strategy and objective functions.

3.1 Problem Formulation

Point-and-direction prompted subtree reconstruction. We formulate neuron reconstruction as autoregressive generation of a rooted tree conditioned on a 3-D image and an oriented starting point. Let $I \in \mathbb{R}^{1\times Z\times Y\times X}$ be a microscopy volume and $\mathcal{T} = (\mathcal{V}, \mathcal{E}, \mathbf{x})$ a rooted neuronal tree, where $\mathbf{x}_v \in \mathbb{R}^3$ denotes the position of node v. An oriented root edge $\xi = (r \to r^+)$, from the root r to a direct child r^+, defines the prompt

$$\xi \equiv (\mathbf{p}_0, \mathbf{d}_0), \qquad \mathbf{p}_0 = \mathbf{x}_r, \qquad \mathbf{d}_0 = \frac{\mathbf{x}_{r^+} - \mathbf{x}_r}{\|\mathbf{x}_{r^+} - \mathbf{x}_r\|_2}. \tag{1}$$

As Figure 1a shows, the point specifies where tracing begins, while the direction determines which root branch is reconstructed. The target is the induced descendant subtree

$$\mathcal{T}_\xi = \mathcal{T}\left[\{r\} \cup \{v \in \mathcal{V} : r^+ \preceq v\}\right], \tag{2}$$

where $\preceq$ is the ancestor relation, including equality. Given only I and $(\mathbf{p}_0, \mathbf{d}_0)$ at inference, the goal is to recover the subtree's geometry and connectivity.

3.2 TRAJECTORY- AND TOPOLOGY-AWARE AUTOREGRESSIVE TRACING

Node type. For a node v, $\mathrm{ch}(v)$ denotes the set of its direct children. Its topology-derived type is

$$c(v) = \begin{cases} \text{C (continuation)}, & |\mathrm{ch}(v)| = 1, \\ \text{B (branch)}, & |\mathrm{ch}(v)| \geq 2, \\ \text{L (leaf)}, & |\mathrm{ch}(v)| = 0. \end{cases} \tag{3}$$

Discrete representation of 3-D directions. We represent outgoing directions from a node to its children using an approximately uniform Fibonacci-sphere codebook $\mathcal{U} = \{\mathbf{u}_k\}_{k=1}^{K} \subset \mathbb{S}^2$ (Figure 1b), constructed as detailed in Appendix A.1. A continuous direction $\mathbf{d}$ maps to its nearest codeword through $q(\mathbf{d}) = \arg\max_k \mathbf{d}^\top \mathbf{u}_k$.

Causal path state and prediction. At local step t, each active path carries an ordered history

$$\mathcal{H}_t = \big((\widehat{\mathbf{p}}_j, \widehat{\mathbf{u}}_j, \widehat{c}_j)\big)_{j=0}^{t}, \qquad \mathcal{H}_0 = \big((\mathbf{p}_0, \mathbf{u}_0, \mathrm{START})\big). \tag{4}$$

Here, $\widehat{\mathbf{u}}_j$ is the outgoing codeword assigned to this path at node j, with $\mathbf{u}_0 = \mathbf{u}_{q(\mathbf{d}_0)}$. The history contains the states from the root to the current node. Conditioned on I and $\mathcal{H}_t$, the tracer predicts

$$\big(\widehat{\boldsymbol{\pi}}_{t+1}^{d}, \widehat{\boldsymbol{\pi}}_{t+1}^{c}, \widetilde{\mathbf{o}}_{t+1}\big) = f_\theta(I, \mathcal{H}_t), \tag{5}$$

where $\widehat{\boldsymbol{\pi}}_{t+1}^{d} \in \Delta^{K-1}$ is a distribution over outgoing directions at the new node. As illustrated in Figure 1b, it is unimodal when the point at position $t+1$ is a continuation point and multimodal when it is a branch point. $\widehat{\boldsymbol{\pi}}_{t+1}^{c} \in \Delta^2$ is a distribution over $\{\mathrm{C}, \mathrm{B}, \mathrm{L}\}$, and $\widetilde{\mathbf{o}}_{t+1} \in \mathbb{R}^3$ parameterizes a positional residual. We select the node type as $\widehat{c}_{t+1} = \arg\max_c \widehat{\pi}_{t+1,c}^{c}$.

Advance and refine. Figure 1c illustrates a single iteration in the iterative tracking process. The current path's outgoing direction first defines a coarse next position; a bounded residual then adjusts it to accommodate direction quantization and local localization errors:

$$\overline{\mathbf{p}}_{t+1} = \widehat{\mathbf{p}}_t + s\widehat{\mathbf{u}}_t, \qquad \boldsymbol{\delta}_{t+1} = s\rho \tanh(\widetilde{\mathbf{o}}_{t+1}), \qquad \widehat{\mathbf{p}}_{t+1} = \overline{\mathbf{p}}_{t+1} + \boldsymbol{\delta}_{t+1}, \tag{6}$$

where $s > 0$ is the fixed proposal step size and $\rho \geq 0$ controls the correction range. The offset uses trajectory context and image evidence around $\overline{\mathbf{p}}_{t+1}$ to correct coarse-step errors and existing positional drift before the refined position enters the history. The current $\widehat{\mathbf{u}}_t$ determines this advance; the newly predicted $\widehat{\boldsymbol{\pi}}_{t+1}^{d}$ supplies outgoing directions for subsequent transitions.

Type-conditioned path expansion. After placing the new node, its predicted type determines whether to continue, branch, or stop. Let σ_{t+1} sort the direction probabilities in descending order. For a cumulative probability threshold $\alpha \in (0, 1]$, define

$$m_{t+1} = \min\left\{ m : \sum_{j=1}^{m} \widehat{\pi}_{t+1,\sigma_{t+1}(j)}^{d} \geq \alpha \right\}, \qquad \mathcal{D}_{t+1}^{(\alpha)} = \left\{ \mathbf{u}_{\sigma_{t+1}(j)} : 1 \leq j \leq m_{t+1} \right\}. \tag{7}$$

The decoding rule converts these predictions into a set of outgoing actions:

$$\mathcal{A}_{t+1} = \begin{cases} \{\mathbf{u}_{\arg\max_k \widehat{\pi}_{t+1,k}^{d}}\}, & \widehat{c}_{t+1} = \mathrm{C}, \\ \mathcal{D}_{t+1}^{(\alpha)}, & \widehat{c}_{t+1} = \mathrm{B}, \\ \varnothing, & \widehat{c}_{t+1} = \mathrm{L}. \end{cases} \tag{8}$$

During inference, angular peak suppression and a small path cap can remove redundant near-collinear branch candidates. For every $\mathbf{u}_k \in \mathcal{A}_{t+1}$, we create a new active state

$$\mathcal{H}_{t+1}^{(k)} = \mathrm{concat}\left(\mathcal{H}_t, (\widehat{\mathbf{p}}_{t+1}, \mathbf{u}_k, \widehat{c}_{t+1})\right). \tag{9}$$

Child paths share the preceding history and junction $\widehat{\mathbf{p}}_{t+1}$, with distinct outgoing directions.

Reconstructed subtree. Recursion ends when all paths terminate or reach an image boundary or step budget. Let $\mathcal{P}_\xi$ index the resulting paths, with node sequences $(v_0^\ell, \ldots, v_{L_\ell}^\ell)$ rooted at the shared prompt node at $\mathbf{p}_0$. Inherited prefix nodes retain their identities across paths. Their union defines the reconstructed subtree:

$$\widehat{\mathcal{V}}_\xi = \bigcup_{\ell \in \mathcal{P}_\xi} \{v_t^\ell\}_{t=0}^{L_\ell}, \quad \widehat{\mathcal{E}}_\xi = \bigcup_{\ell \in \mathcal{P}_\xi} \{(v_t^\ell, v_{t+1}^\ell)\}_{t=0}^{L_\ell - 1}, \quad \widehat{\mathcal{T}}_\xi = (\widehat{\mathcal{V}}_\xi, \widehat{\mathcal{E}}_\xi, \widehat{\mathbf{x}}), \qquad \widehat{\mathbf{x}}_{v_t^\ell} = \widehat{\mathbf{p}}_t^{\,\ell}. \tag{10}$$

The output $\widehat{\mathcal{T}}_\xi$ estimates the prompted subtree $\mathcal{T}_\xi$ in Eq. (2), with geometry given by $\widehat{\mathbf{x}}$ and connectivity established during expansion.

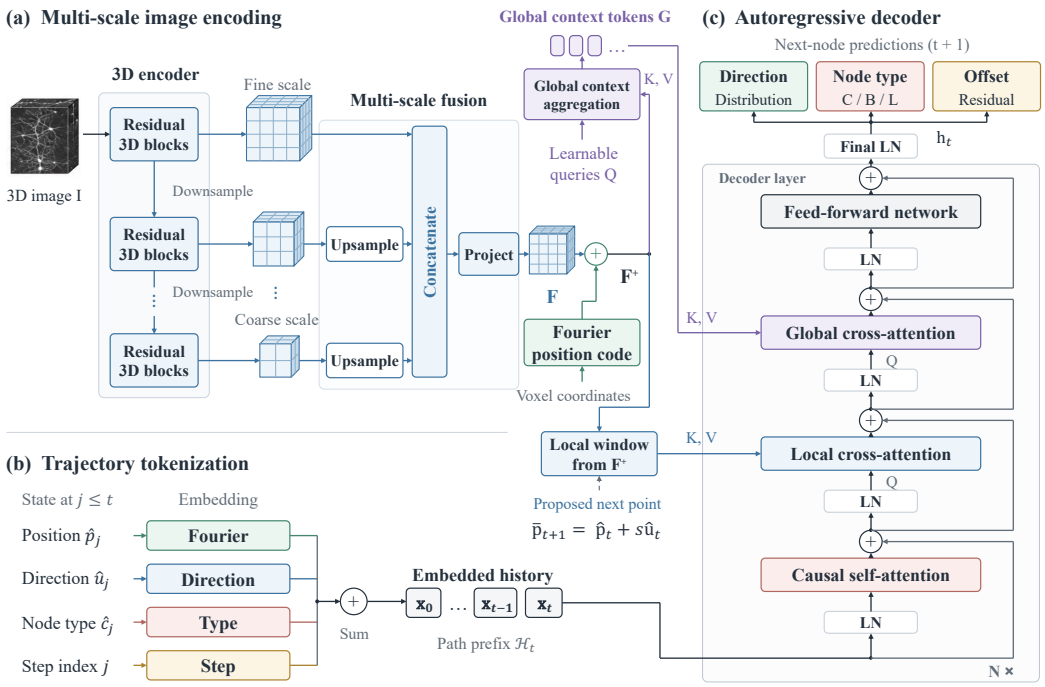

Figure 2: **TreeTracer architecture.** **(a)** Position-encoded multi-scale image features provide local detail and compact global context. **(b)** Position, outgoing direction, node type, and step embeddings form trajectory tokens. **(c)** A causal local–global decoder predicts outgoing directions, node type, and a positional offset for autoregressive tracing.

3.3 Model Architecture

TreeTracer implements f_θ with a multi-scale image encoder and a causal Transformer decoder (Figure 2). To retain spatial detail without attending to the full voxel grid at every step, the decoder reads local feature windows and compact global context tokens together with the trajectory history.

Multi-scale image encoding. Residual 3-D convolutional blocks extract features at multiple resolutions (Figure 2a). Coarse features are upsampled to the finest grid, concatenated with fine-resolution features, and projected to a dense map $\mathbf{F}$, combining neurite detail with broader surroundings. To distinguish nearby spatial locations, we add a multi-frequency Fourier embedding (Mildenhall et al., 2020; Tancik et al., 2020) of each voxel coordinate $\mathbf{v}$:

$$\mathbf{F}^+(\mathbf{v}) = \mathbf{F}(\mathbf{v}) + \phi_{\text{pos}}(\mathbf{v}). \tag{11}$$

The decoder reads local windows from $\mathbf{F}^+$. For global context, learnable queries attend to all locations in $\mathbf{F}^+$ as keys and values, producing compact global features $\mathbf{G}$.

Trajectory tokenization. Each state in $\mathcal{H}_t$ is embedded by summing its position, outgoing direction, node type, and step representations (Figure 2b):

$$\mathbf{x}_j = \phi_{\text{pos}}(\widehat{\mathbf{p}}_j) + \mathbf{E}_d[k_j] + \mathbf{E}_c[\widehat{c}_j] + \phi_{\text{step}}(j), \qquad 0 \le j \le t, \tag{12}$$

where k_j indexes $\widehat{\mathbf{u}}_j = \mathbf{u}_{k_j}$, $\mathbf{E}_d$ and $\mathbf{E}_c$ are learned direction and type embeddings, and ϕ_{step} encodes path order. Sharing ϕ_{pos} with the image grid places trajectory and image coordinates in a common representation. At a shared junction, direction embeddings distinguish the continuations assigned to child paths.

Causal local–global decoding. Each decoder layer applies causal self-attention, local cross-attention, global cross-attention, and a feed-forward network in sequence, with pre-layer normalization and residual connections (Figure 2c). Causal self-attention attends to historical trajectory features up to j. Local cross-attention uses the normalized trajectory state as its query and features sampled from $\mathbf{F}^+$ around the coarse proposal $\overline{\mathbf{p}}_{j+1}$ in Eq. (6) as keys and values. Global cross-attention then uses the updated, normalized state to query $\mathbf{G}$, incorporating evidence beyond the local window within the input image.

Geometry and topology prediction. After the decoder stack, a final layer normalization produces the shared current-step representation $\mathbf{h}_t$. Three task-specific heads predict the next node's attributes as defined in Eq. (5):

$$\widehat{\boldsymbol{\pi}}^{d}_{t+1} = \mathrm{softmax}(g_d(\mathbf{h}_t)), \qquad \widehat{\boldsymbol{\pi}}^{c}_{t+1} = \mathrm{softmax}(g_c(\mathbf{h}_t)), \qquad \widetilde{\mathbf{o}}_{t+1} = g_o(\mathbf{h}_t). \tag{13}$$

3.4 Training Strategy

Event-aware sampling. We sample continuation, branch, and leaf nodes with prescribed proportions to increase supervision for rare branching and termination decisions while retaining continuation examples. Sampling proportions and training hyperparameters are given in Appendix A.2.

Training positional refinement. Training uses annotated path prefixes with positions $\mathbf{p}^*_t$, node types c^*_t, and path-assigned outgoing codewords $\mathbf{u}^*_t \in \mathcal{U}$. To learn correction from locally displaced states, we perturb valid input positions in voxel coordinates, preserving the initial prompt and START token:

$$\widetilde{\mathbf{p}}_t = \mathrm{clip}_{\Omega}(\mathbf{p}^*_t + b_t\boldsymbol{\epsilon}_t), \qquad b_t \sim \mathrm{Bernoulli}(p_{\mathrm{noise}}), \qquad \boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \sigma^2_{\mathrm{noise}}\mathbf{I}_3), \tag{14}$$

where clip_{Ω} clips to the valid image region. Direction and type inputs retain their annotated values. The perturbed positions are used in trajectory tokens and coarse proposals, while the target positions remain unchanged. Consistent with Eq. (6), the residual target is

$$\overline{\mathbf{p}}_{t+1} = \widetilde{\mathbf{p}}_t + s\mathbf{u}^*_t, \qquad \boldsymbol{\delta}^*_{t+1} = \mathbf{p}^*_{t+1} - \overline{\mathbf{p}}_{t+1}. \tag{15}$$

This target supervises the bounded residual in Eq. (6), combining coarse-step correction with recovery from positional perturbations.

Topology-aware soft direction targets. Direction supervision accounts for both angular proximity in $\mathcal{U}$ and multiple outgoing branches. At a non-leaf ground-truth node v corresponding to step $t+1$, each child $w \in \mathrm{ch}(v)$ defines a direction $\mathbf{d}_{v,w} = (\mathbf{x}_w - \mathbf{x}_v)/\|\mathbf{x}_w - \mathbf{x}_v\|_2$. We form an angular soft target around its nearest codeword:

$$a_k(\mathbf{d}) = \frac{\exp\left(\mathbf{u}^{\top}_{q(\mathbf{d})}\mathbf{u}_k/\tau\right)}{\sum_{j=1}^{K}\exp\left(\mathbf{u}^{\top}_{q(\mathbf{d})}\mathbf{u}_j/\tau\right)}, \qquad k = 1, \ldots, K, \tag{16}$$

where $\tau > 0$ controls the angular spread. Taking the elementwise maximum over children and normalizing gives a target for both continuation and branch nodes:

$$\pi^{d,*}_{t+1,k} = \frac{\max_{w\in\mathrm{ch}(v)} a_k(\mathbf{d}_{v,w})}{\sum_{j=1}^{K}\max_{w\in\mathrm{ch}(v)} a_j(\mathbf{d}_{v,w})}. \tag{17}$$

The target supervises all outgoing branches and reduces to a single angular distribution for continuation nodes (Figure 1b). Leaf nodes are excluded from direction supervision.

Multi-task objective. We use focal cross-entropy for node type ($\mathcal{L}_{\mathrm{type}}$), soft-label focal cross-entropy for direction ($\mathcal{L}_{\mathrm{dir}}$), and Smooth L1 for the positional offset ($\mathcal{L}_{\mathrm{off}}$). Each loss is averaged over its valid supervised steps; full definitions are given in Appendix A.3. The joint objective is

$$\mathcal{L} = \frac{\lambda_{\mathrm{dir}}\mathcal{L}_{\mathrm{dir}} + \lambda_{\mathrm{type}}\mathcal{L}_{\mathrm{type}} + \lambda_{\mathrm{off}}\mathcal{L}_{\mathrm{off}}}{\lambda_{\mathrm{dir}} + \lambda_{\mathrm{type}} + \lambda_{\mathrm{off}}}. \tag{18}$$

4 Experiment

4.1 Datasets and Metrics

We selected two public mouse-brain light-microscopy datasets, **CWMBS** (Liu et al., 2024b) and **CORAL** (Yang et al., 2026), because both provide image blocks cropped from whole-brain volumes, allowing local subtree evaluation in the imaging setting of brain-wide tracing. CORAL further enables evaluation of how local tracing translates into complete neuron reconstruction. CWMBS contains cluttered backgrounds and thin, weak-signal neuronal structures; we evaluated 483 held-out subtrees. On CORAL, we used a fixed subset of 500 test subtrees and 14 test neurons for local

and brain-wide evaluation, respectively. Source blocks were disjoint across training, validation, and testing. Appendix B details image acquisition, subtree preparation, data splits, and length statistics.

Fiber metrics. The structure-aware Fiber metrics (Yang et al., 2026) measure recovery of root-to-leaf trajectories and their connectivity. We report micro-averaged precision (P), recall (R), and F1 as percentages, with local results grouped by whether the reference subtree branches. Subscripts denote bootstrap standard deviations (SDs), also in percentage points. We report dendrite F1, axon F1, and overall F1 for brain-wide reconstruction. Appendix C describes more detail.

4.2 Comparison with SOTA Methods

Block-level subtree reconstruction. We first evaluated whether learned tree expansion preserves trajectory continuity and connectivity in local image blocks. For two-stage pipelines, we compared VNet (Milletari et al., 2016) and SegMamba (Xing et al., 2025) as general-purpose 3D segmentation networks, DSCNet (Qi et al., 2023) for tubular structures, and GBP-Net (Yan et al., 2026) for neuron segmentation. We paired each segmentation model with neuTube (Feng et al., 2015), APP2 (Xiao & Peng, 2013), and Kimimaro (Silversmith et al., 2021). We also compared learning-based tracers DRL (Dai et al., 2019), SPE-DNR (Chen et al., 2022), and NETracer (Liu et al., 2025). TreeTracer (vanilla) directly reconstructs the subtree selected by the supplied point-and-direction prompt. TreeTracer (sandwich) augments this prompt with image-derived seeds, applies the same trained tracer, and fuses and refines the resulting trajectories before extracting the target subtree. Its model parameters remain fixed (Appendix A.4).

Table 1: **Quantitative comparison on CWMBS.** "raw" denotes no segmentation preprocessing. **Bold** and underlined values indicate the best and second-best results in each column, respectively.

Method		**No-branch** ($n = 335$)			**Branch** ($n = 148$)			**All** ($n = 483$)		
		P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)
Segmentation + reconstruction										
neuTube	raw	$5.5_{2.2}$	$33.4_{3.5}$	$9.4_{2.8}$	$12.9_{3.2}$	$46.3_{5.0}$	$20.1_{3.7}$	$9.2_{2.7}$	$41.5_{3.7}$	$15.0_{3.4}$
	V-Net	$15.7_{3.7}$	$36.1_{2.4}$	$21.9_{3.6}$	$35.1_{4.7}$	$\underline{61.2}_{3.2}$	$44.6_{3.6}$	$26.6_{3.5}$	$51.8_{2.7}$	$35.1_{3.0}$
	DSCNet	$14.0_{3.0}$	$43.9_{3.5}$	$21.3_{3.5}$	$27.5_{3.7}$	$\mathbf{63.3}_{3.7}$	$38.3_{3.3}$	$21.5_{2.8}$	$\underline{56.1}_{3.1}$	$31.1_{2.9}$
	SegMamba	$15.1_{3.5}$	$38.5_{2.8}$	$21.7_{3.6}$	$28.9_{4.3}$	$57.6_{4.3}$	$38.5_{3.5}$	$23.0_{3.3}$	$50.5_{3.0}$	$31.6_{3.0}$
	GBP-Net	$22.5_{3.4}$	$36.1_{3.8}$	$27.8_{2.8}$	$34.3_{4.2}$	$55.7_{4.3}$	$42.4_{2.7}$	$29.9_{2.8}$	$48.4_{3.5}$	$37.0_{1.9}$
APP2	V-Net	$21.2_{4.2}$	$49.0_{4.0}$	$29.6_{4.1}$	$28.3_{5.2}$	$37.8_{4.3}$	$32.3_{3.8}$	$24.7_{4.0}$	$41.9_{3.6}$	$31.1_{3.3}$
	DSCNet	$18.7_{2.9}$	$51.6_{3.8}$	$27.4_{3.1}$	$26.0_{5.9}$	$40.8_{4.6}$	$31.8_{4.4}$	$22.3_{3.4}$	$44.8_{3.8}$	$29.8_{3.1}$
	SegMamba	$20.7_{3.5}$	$50.1_{4.1}$	$29.3_{3.7}$	$26.5_{5.0}$	$36.0_{4.5}$	$30.5_{3.6}$	$23.5_{3.4}$	$41.3_{3.7}$	$29.9_{3.0}$
	GBP-Net	$27.9_{7.6}$	$34.0_{3.4}$	$30.7_{4.5}$	$45.5_{8.4}$	$28.0_{4.0}$	$34.7_{4.2}$	$36.0_{8.1}$	$30.3_{3.0}$	$32.9_{3.8}$
Kimimaro	V-Net	$48.8_{4.2}$	$47.8_{4.2}$	$48.3_{3.6}$	$55.8_{4.1}$	$33.3_{3.9}$	$41.7_{3.7}$	$52.3_{3.5}$	$38.7_{3.3}$	$44.5_{3.1}$
	DSCNet	$44.3_{5.5}$	$58.5_{3.5}$	$50.5_{4.0}$	$\underline{58.0}_{4.2}$	$44.9_{3.2}$	$\underline{50.6}_{2.1}$	$51.1_{4.3}$	$49.9_{2.8}$	$50.5_{2.3}$
	SegMamba	$37.1_{5.2}$	$54.6_{4.4}$	$44.2_{4.3}$	$42.0_{6.6}$	$41.3_{4.4}$	$41.6_{3.5}$	$39.7_{5.0}$	$46.3_{3.7}$	$42.7_{3.4}$
	GBP-Net	$53.8_{4.5}$	$57.0_{3.4}$	$55.4_{3.2}$	$55.6_{5.0}$	$44.3_{3.0}$	$49.3_{2.6}$	$54.8_{3.5}$	$49.1_{2.4}$	$51.8_{2.3}$
Tracing										
DRL		$4.7_{1.5}$	$3.6_{1.1}$	$4.1_{1.3}$	$0.8_{0.8}$	$0.2_{0.2}$	$0.3_{0.3}$	$3.4_{0.9}$	$1.4_{0.4}$	$2.0_{0.5}$
SPE-DNR		$9.1_{1.7}$	$19.4_{2.5}$	$12.3_{1.6}$	$13.8_{3.4}$	$13.7_{3.9}$	$13.7_{3.5}$	$11.1_{1.8}$	$15.8_{2.9}$	$13.1_{1.9}$
NETracer		$42.5_{3.3}$	$33.1_{2.8}$	$37.2_{2.9}$	$35.5_{5.8}$	$7.6_{1.5}$	$12.6_{2.3}$	$40.3_{3.0}$	$17.1_{1.7}$	$24.0_{2.1}$
TreeTracer (vanilla)		$\mathbf{67.3}_{4.3}$	$\underline{73.1}_{3.7}$	$\mathbf{70.1}_{3.8}$	$\mathbf{61.4}_{3.5}$	$27.3_{2.0}$	$37.8_{2.4}$	$\mathbf{64.9}_{3.4}$	$44.4_{2.2}$	$\underline{52.7}_{2.4}$
TreeTracer (sandwich)		$\underline{57.9}_{4.1}$	$\mathbf{81.8}_{2.4}$	$\underline{67.8}_{3.2}$	$56.9_{2.8}$	$46.1_{3.3}$	$\mathbf{50.9}_{2.7}$	$\underline{57.4}_{2.7}$	$\mathbf{59.4}_{2.7}$	$\mathbf{58.4}_{2.2}$

TreeTracer (sandwich) achieves the highest overall Fiber F1 on both CWMBS and CORAL, reaching 58.4% and 66.4%, respectively, compared with 51.8% and 61.0% for the strongest baseline, Kimimaro (GBP-Net), corresponding to absolute improvements of 6.6 and 5.4 percentage points. Vanilla also exceeds this baseline on both datasets. Vanilla achieves the highest no-branch F1, exceeding Kimimaro (GBP-Net) by 14.7 and 9.8 percentage points. This advantage is consistent with using trajectory history and local and global image context to maintain coherent paths through ambiguous regions. Figure 3 illustrates this behavior under weak signals and closely apposed fibers.

The branching results support learning geometry and topology jointly: vanilla reaches F1 of 37.8% and 46.0%, compared with at most 13.7% and 16.3% among DRL, SPE-DNR, and NETracer. Its recall of 27.3% and 34.0% still leaves many paths unrecovered. Additional image-derived seeds

Table 2: **Quantitative comparison on CORAL.** "raw" denotes no segmentation preprocessing. **Bold** and underlined values indicate the best and second-best results in each column, respectively.

Method		No-branch ($n = 359$)			Branch ($n = 141$)			All ($n = 500$)		
		P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)
Segmentation + reconstruction										
neuTube	raw	$8.3_{4.8}$	$49.6_{2.6}$	$14.2_{6.4}$	$14.5_{2.3}$	$59.3_{3.6}$	$23.3_{2.9}$	$11.1_{3.2}$	$54.8_{2.4}$	$18.4_{4.2}$
	V-Net	$37.3_{12.7}$	$54.6_{2.7}$	$44.3_{8.5}$	$54.0_{9.2}$	$61.5_{3.7}$	$57.5_{6.1}$	$45.3_{8.7}$	$58.3_{2.4}$	$51.0_{5.6}$
	DSCNet	$53.8_{5.1}$	$58.8_{2.6}$	$56.2_{3.2}$	$48.8_{9.6}$	$64.4_{3.3}$	$55.5_{6.2}$	$50.9_{6.2}$	$61.8_{2.3}$	$55.8_{3.9}$
	SegMamba	$37.3_{11.7}$	$55.4_{2.9}$	$44.6_{7.9}$	$59.7_{8.6}$	$\mathbf{66.7}_{3.4}$	$\mathbf{63.1}_{5.2}$	$47.8_{10.4}$	$61.5_{2.5}$	$53.8_{6.4}$
	GBP-Net	$39.7_{8.3}$	$57.7_{2.8}$	$47.0_{5.7}$	$49.5_{7.2}$	$\underline{66.3}_{3.3}$	$56.6_{5.2}$	$44.8_{6.2}$	$\underline{62.3}_{2.3}$	$52.1_{4.3}$
APP2	V-Net	$46.4_{11.8}$	$39.0_{2.8}$	$42.4_{5.1}$	$47.5_{8.5}$	$33.5_{4.3}$	$39.3_{4.0}$	$46.9_{8.8}$	$36.0_{2.6}$	$40.8_{3.4}$
	DSCNet	$44.7_{14.7}$	$38.7_{2.6}$	$41.5_{6.4}$	$53.4_{9.5}$	$34.2_{3.7}$	$41.7_{4.2}$	$48.7_{9.2}$	$36.3_{2.3}$	$41.6_{3.8}$
	SegMamba	$55.2_{8.1}$	$40.1_{2.7}$	$46.5_{3.5}$	$52.0_{7.5}$	$34.7_{4.2}$	$41.6_{3.7}$	$53.5_{6.3}$	$37.2_{2.5}$	$43.9_{2.6}$
	GBP-Net	$21.3_{11.2}$	$24.5_{2.4}$	$22.8_{5.4}$	$41.9_{5.0}$	$26.6_{3.9}$	$32.5_{3.2}$	$29.4_{8.6}$	$25.6_{2.5}$	$27.4_{3.5}$
Kimimaro	V-Net	$73.9_{3.5}$	$54.3_{2.7}$	$62.6_{2.6}$	$\underline{73.9}_{4.7}$	$31.1_{2.7}$	$43.8_{3.1}$	$73.9_{3.0}$	$41.8_{2.2}$	$53.4_{2.3}$
	DSCNet	$\mathbf{76.3}_{3.2}$	$56.5_{2.8}$	$65.0_{2.6}$	$\mathbf{75.7}_{4.2}$	$38.8_{3.2}$	$51.3_{3.2}$	$\mathbf{76.0}_{2.9}$	$47.0_{2.3}$	$58.1_{2.3}$
	SegMamba	$73.0_{3.1}$	$54.3_{2.7}$	$62.3_{2.5}$	$73.1_{4.6}$	$38.3_{3.0}$	$50.2_{3.2}$	$73.0_{2.8}$	$45.7_{2.2}$	$56.2_{2.2}$
	GBP-Net	$61.5_{3.6}$	$70.8_{2.4}$	$65.8_{2.5}$	$63.2_{4.6}$	$50.5_{3.1}$	$56.1_{3.0}$	$62.2_{2.9}$	$59.8_{2.1}$	$61.0_{2.0}$
Tracing										
DRL		$1.4_{0.7}$	$1.1_{0.6}$	$1.2_{0.6}$	$0.0_{0.0}$	$0.0_{0.0}$	$0.0_{0.0}$	$1.0_{0.5}$	$0.5_{0.3}$	$0.7_{0.3}$
SPE-DNR		$15.5_{1.6}$	$18.9_{2.1}$	$17.1_{1.8}$	$18.8_{2.7}$	$14.4_{2.3}$	$16.3_{2.4}$	$16.9_{1.5}$	$16.5_{1.5}$	$16.7_{1.4}$
NETracer		$22.0_{2.5}$	$16.4_{1.9}$	$18.8_{2.1}$	$29.3_{4.5}$	$8.1_{1.4}$	$12.7_{2.1}$	$24.2_{2.2}$	$12.0_{1.2}$	$16.0_{1.5}$
TreeTracer (vanilla)		$\underline{75.7}_{2.7}$	$\underline{75.5}_{2.3}$	$\mathbf{75.6}_{2.3}$	$71.0_{5.3}$	$34.0_{2.3}$	$46.0_{2.8}$	$\underline{74.0}_{2.6}$	$53.2_{2.1}$	$\underline{61.9}_{2.0}$
TreeTracer (sandwich)		$65.9_{2.8}$	$\mathbf{84.1}_{1.9}$	$\underline{73.9}_{2.2}$	$59.3_{4.2}$	$58.9_{3.1}$	$\underline{59.1}_{2.9}$	$62.8_{2.6}$	$\mathbf{70.5}_{2.0}$	$\mathbf{66.4}_{1.9}$

provide alternative entry points. Sandwich raises branch recall to 46.1% and 58.9% and F1 to 50.9% and 59.1%. TreeTracer (vanilla) also achieves 130.39 voxel/s on CWMBS, 14.6% faster than Kimimaro (GBP-Net) (see Appendix D for timing details).

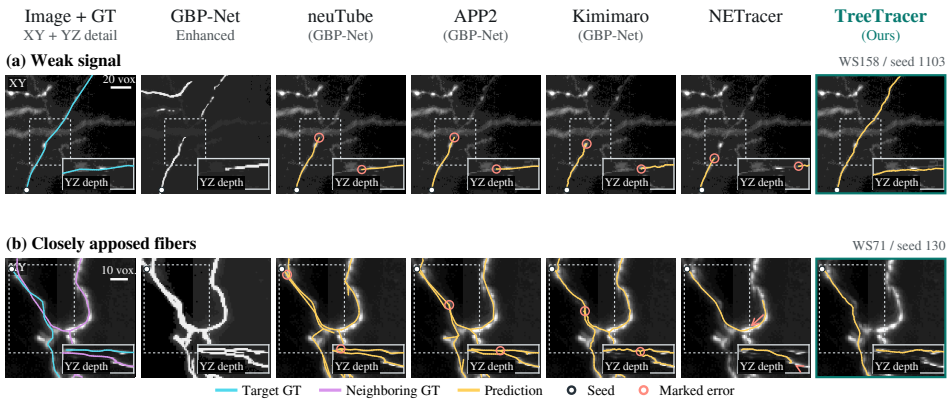

Figure 3: **Tracing weak signals and closely apposed fibers.** XY maximum-intensity projections with YZ insets of dashed regions. Circles denote premature endpoints or false branches, arrows fiber switches, and white dots start roots. neuTube, APP2 and Kimimaro used GBP-Net-enhanced inputs.

Brain-wide neuron reconstruction. We reconstructed CORAL's 14 test neurons from soma initializations. Each baseline uses its best segmentation model by overall brain-wide Fiber F1 (Yang et al., 2026): SwinUNETR (Hatamizadeh et al., 2021) for APP2, MedNeXt (Roy et al., 2023) for neuTube, and SegMamba (Xing et al., 2025) for Kimimaro. Table 3 shows three main findings. (1) TreeTracer achieves state-of-the-art overall Fiber F1, exceeding the strongest baseline, Kimimaro (SegMamba), by 6.5 percentage points. (2) TreeTracer achieves the highest Axon F1. This gain is consistent with more reliable local topology prediction, which helps reduce drift and premature

Table 3: **Brain-wide reconstruction results on CORAL's 14 test neurons.** Best and second-best values are **bold** and underlined, respectively.

Method	Dendrite F1(%)	Axon F1(%)	Fiber F1(%)
Kimimaro (SegMamba)	**64.8**	24.1	<u>35.1</u>
APP2 (SwinUNETR)	52.3	<u>24.4</u>	32.8
neuTube (MedNeXt)	45.2	23.2	30.8
TreeTracer	<u>62.2</u>	**33.4**	**41.6**

termination during repeated tree expansion and sustain long-range axon tracing. (3) Kimimaro's conservative connections favor precision and yield the highest Dendrite F1. The same conservatism can cause premature termination on long axons, consistent with its lower overall recall (23.3% vs. 43.6% for TreeTracer) and Axon F1. TreeTracer substantially improves axon reconstruction while remaining competitive on dendrites.

Qualitative Study. Figure 3 shows two CWMBS cases. All four baselines terminated near the weak-signal segment, whereas TreeTracer followed the GT trajectory through it. For closely apposed fibers overlapping in XY but separated in YZ, neuTube, APP2 and Kimimaro (using GBP-Net-enhanced inputs) introduced spurious branches, while NETracer switched fibers. TreeTracer maintained the target trajectory without a false junction.

4.3 Ablation Study

Trajectory content. Table 4 evaluates direction, position, and node-type information in the trajectory representation. Removing direction history caused the largest overall F1 drop, followed by shared positional encoding and node-type history. Node types mainly aid branching reconstruction: their removal reduced branch F1 from 37.8% to 33.7%, while leaving no-branch F1 nearly unchanged.

Table 4: **Trajectory-content ablation on CWMBS.** Columns report Fiber F1 (%); **bold** marks the best score per column.

Variant	No-branch ($n = 335$)	Branch ($n = 148$)	All ($n = 483$)
TreeTracer (vanilla)	$70.1_{3.8}$	$\mathbf{37.8}_{2.4}$	$\mathbf{52.7}_{2.4}$
w/o direction	$48.6_{6.7}$	$28.5_{2.7}$	$38.1_{4.0}$
w/o shared position	$65.7_{3.1}$	$35.3_{2.4}$	$49.7_{2.1}$
w/o node type	$\mathbf{70.2}_{3.4}$	$33.7_{2.2}$	$50.5_{2.3}$

Decoder attention. Table 5 separates trajectory self-attention (S) from local (L) and global (G) image cross-attention. Removing L reduced overall F1 from 52.7% to 2.4%, whereas L alone reached 51.1%, identifying local image evidence as the main source of reconstruction accuracy. Adding S or G to L raised precision from 55.4% to 58.5% or 64.5%, respectively. The full decoder reached 64.9%, with lower recall than L alone (44.4% vs. 47.5%), indicating a precision–recall trade-off. The incremental precision gain from S over L+G was small (0.4 percentage points). Appendix E reports all combinations and paired uncertainty estimates.

Positional refinement. Table 6 ablates continuous offset correction and position perturbation during training. Offset correction alone improved fiber F1 from 27.5% to 34.8%, while combining it with position perturbation further raised F1 to 52.7%. Perturbation without offset correction yielded 26.6%. These results support learning positional corrections from displaced trajectories.

Table 5: **Decoder attention ablation on CWMBS.** S: trajectory self-attention; L/G: local/global image cross-attention. The first row is the full decoder; bold marks column best.

S	L	G	P(%)	R(%)	F1(%)
✓	✓	✓	$\mathbf{64.9}_{3.4}$	$44.4_{2.2}$	$\mathbf{52.7}_{2.4}$
✓	—	✓	$2.8_{0.8}$	$2.1_{0.5}$	$2.4_{0.6}$
✓	✓	—	$58.5_{5.1}$	$45.4_{2.3}$	$51.1_{2.6}$
—	✓	✓	$64.5_{3.2}$	$44.3_{2.2}$	$52.5_{2.3}$
—	✓	—	$55.4_{4.0}$	$\mathbf{47.5}_{2.0}$	$51.1_{2.2}$

Table 6: **Positional refinement on CWMBS.** Position noise was applied during training; offsets refined positions during tracing.

Position noise	Offset	F1(%)
—	—	$27.5_{2.3}$
✓	—	$26.6_{2.0}$
—	✓	$34.8_{2.5}$
✓	✓	$\mathbf{52.7}_{2.4}$

5 Conclusion

We presented TreeTracer, an autoregressive model that expands a point-and-direction prompt into a neuronal subtree. Jointly predicting geometric and topological information at future positions enables end-to-end learning of tree-structured tracing. A causal decoder fuses historical trajectory features, local high-resolution image features, and compact global image features, substantially improving the accuracy of local topology prediction. Experiments on CWMBS and CORAL showed that the TreeTracer achieved higher overall Fiber F1 than the evaluated baselines and also achieved the highest overall and axon F1 in brain-wide reconstruction.

AI USE STATEMENT

We used generative AI tools to improve the clarity and readability of the manuscript and to assist with code editing. We reviewed and revised all AI-assisted manuscript edits to ensure that they preserved the intended technical meaning. We take full responsibility for the final content of this work, including its text, code, claims, and citations.

REPRODUCIBILITY STATEMENT

We provide a detailed description of our implementation in Appendix A. Our code will also be released upon acceptance of the paper.

REFERENCES

Weixun Chen, Min Liu, Hao Du, Miroslav Radojevic, Yaonan Wang, and Erik Meijering. Deep-learning-based automated neuron reconstruction from 3D microscopy images using synthetic training images. *IEEE Transactions on Medical Imaging*, 41(5):1031–1042, 2022. doi: 10.1109/TMI.2021.3130934.

Tianhong Dai, Magda Dubois, Kai Arulkumaran, Jonathan Campbell, Cher Bass, Benjamin Billot, Fatmatulzehra Uslu, Vincenzo de Paola, Claudia Clopath, and Anil Anthony Bharath. Deep reinforcement learning for subpixel neural tracking. In *Proceedings of the 2nd International Conference on Medical Imaging with Deep Learning*, volume 102 of *Proceedings of Machine Learning Research*, pp. 130–150. PMLR, 2019.

Linqing Feng, Ting Zhao, and Jinhyun Kim. neuTube 1.0: A new design for efficient neuron reconstruction software based on the SWC format. *eNeuro*, 2(1):ENEURO.0049–14.2014, 2015. doi: 10.1523/ENEURO.0049-14.2014.

Ali Hatamizadeh, Vishwesh Nath, Yucheng Tang, Dong Yang, Holger R Roth, and Daguang Xu. Swin unetr: Swin transformers for semantic segmentation of brain tumors in mri images. In *International MICCAI brainlesion workshop*, pp. 272–284. Springer, 2021.

Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. Perceiver: General perception with iterative attention. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 4651–4664. PMLR, 2021. URL https://proceedings.mlr.press/v139/jaegle21a.html.

Qiufu Li and Linlin Shen. 3D neuron reconstruction in tangled neuronal image with deep networks. *IEEE Transactions on Medical Imaging*, 39(2):425–435, 2020. doi: 10.1109/TMI.2019.2926568.

Chao Liu, Ting Zhao, and Nenggan Zheng. DeepBranchTracer: A generally-applicable approach to curvilinear structure reconstruction using multi-feature learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 3548–3557, 2024a. doi: 10.1609/aaai.v38i4.28143.

Chao Liu, Yangbo Jiang, and Nenggan Zheng. NETracer: A topology-aware iterative tracing approach for tubular structure extraction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 20593–20602, 2025.

Min Liu, Shuhan Wu, Runze Chen, Zhuangdian Lin, Yaonan Wang, and Erik Meijering. Brain image segmentation for ultrascale neuron reconstruction via an adaptive dual-task learning network. *IEEE Transactions on Medical Imaging*, 43(7):2574–2586, 2024b. doi: 10.1109/TMI.2024.3367384.

Linus Manubens-Gil, Zhi Zhou, Hanbo Chen, et al. BigNeuron: A resource to benchmark and predict performance of algorithms for automated tracing of neurons in light microscopy datasets. *Nature Methods*, 20(6):824–835, 2023. doi: 10.1038/s41592-023-01848-5.

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 *Computer Vision – ECCV 2020*, 2020.

Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 fourth international conference on 3D vision (3DV)*, pp. 565–571. Ieee, 2016.

Hanchuan Peng, Fuhui Long, and Gene Myers. Automatic 3D neuron tracing using all-path pruning. *Bioinformatics*, 27(13):i239–i247, 2011. doi: 10.1093/bioinformatics/btr237.

Yaolei Qi, Yuting He, Xiaoming Qi, Yuan Zhang, and Guanyu Yang. Dynamic snake convolution based on topological geometric constraints for tubular structure segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6070–6079, 2023.

Saikat Roy, Gregor Koehler, Constantin Ulrich, Michael Baumgartner, Jens Petersen, Fabian Isensee, Paul F Jaeger, and Klaus H Maier-Hein. Mednext: transformer-driven scaling of convnets for medical image segmentation. In *International conference on medical image computing and computer-assisted intervention*, pp. 405–415. Springer, 2023.

Suprosanna Shit, Johannes C. Paetzold, Anjany Sekuboyina, Ivan Ezhov, Alexander Unger, Andrey Zhylka, Josien P. W. Pluim, Ulrich Bauer, and Bjoern H. Menze. clDice – a novel topology-preserving loss function for tubular structure segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16560–16569, 2021.

William Silversmith, J. Alexander Bae, Peter H. Li, and A. M. Wilson. Kimimaro: Skeletonize densely labeled 3d image segmentations, September 2021. URL `https://doi.org/10.5281/zenodo.5539913`.

Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Yijun Wang, Rui Lang, Rui Li, and Junsong Zhang. NRTR: Neuron reconstruction with transformer from 3D optical microscopy images. *IEEE Transactions on Medical Imaging*, 43(2):886–898, 2024. doi: 10.1109/TMI.2023.3323466.

Hang Xiao and Hanchuan Peng. APP2: Automatic tracing of 3D neuron morphology based on hierarchical pruning of a gray-weighted image distance-tree. *Bioinformatics*, 29(11):1448–1454, 2013. doi: 10.1093/bioinformatics/btt170.

Zhaohu Xing, Tian Ye, Yijun Yang, Du Cai, Baowen Gai, Xiao-Jian Wu, Feng Gao, and Lei Zhu. Segmamba-v2: Long-range sequential modeling mamba for general 3d medical image segmentation. *IEEE Transactions on Medical Imaging*, 2025.

Haiyang Yan, Yanchao Zhang, Zhenchen Li, Jinyue Guo, Hao Zhai, Jiazheng Liu, Yongwei Zhong, Jingbin Yuan, Lijun Shen, Linlin Li, Xufei Du, and Hua Han. Glancing beyond patch: Spatial contextual cues for 3D neuron segmentation. *IEEE Transactions on Medical Imaging*, 45(2): 596–607, 2026. doi: 10.1109/TMI.2025.3602097.

Zekang Yang, Jiamin Li, Zhenghua Li, Jiaqi Fan, Zengcai Guo, and Xiaolin Hu. CORAL: A benchmark for structure-aware and brain-wide neuron reconstruction in light microscopy. *arXiv preprint arXiv:2608.30768*, 2026.

Runkai Zhao, Heng Wang, Chaoyi Zhang, and Weidong Cai. PointNeuron: 3D neuron reconstruction via geometry and topology learning of point clouds. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 5787–5797, 2023.

APPENDIX CONTENTS

A Implementation Details **13**

A.1 Direction codebook construction 13

A.2 Training hyperparameters 13

A.3 Definitions of the loss functions 13

A.4 TreeTracer (sandwich): Multi-seed Inference and Structural Refinement 14

A.5 Model hyperparameters 17

A.6 Hyperparameter search 18

A.6.1 Event-sampling probabilities 18

A.6.2 Loss functions 18

A.6.3 Loss weights 19

A.6.4 Direction codebook size 20

A.7 Large-volume inference 21

B Dataset Preparation and Statistics **22**

B.1 Source data and subtree construction 22

B.2 Subtree lengths by branching class 23

B.3 CORAL brain-wide reconstruction data 25

C Fiber Evaluation Protocol **25**

D Tracing Speed **26**

E Decoder Attention Ablations **26**

F 3-D Positional Encoding Ablation **27**

G Limitations **29**

A Implementation Details

A.1 Direction codebook construction

We construct the Fibonacci-sphere codebook $\mathcal{U} = \{\mathbf{u}_k\}_{k=1}^{K}$ using the golden angle $\gamma = \pi(3 - \sqrt{5})$. For $k = 1, \ldots, K$, we set

$$z_k = 1 - \frac{2(k - \frac{1}{2})}{K}, \qquad \phi_k = (k-1)\gamma, \qquad \mathbf{u}_k = \left[\sqrt{1 - z_k^2}\cos\phi_k \quad \sqrt{1 - z_k^2}\sin\phi_k \quad z_k\right]^\top. \tag{19}$$

This places the K candidate directions approximately uniformly on the unit sphere. The quantizer q in Section 3.2 assigns each continuous direction to its nearest codeword.

A.2 Training hyperparameters

Each experiment was trained on two NVIDIA GeForce RTX 3090 GPUs using input image crops of $32 \times 96 \times 96$ voxels in depth–height–width order and a batch size of 4 per GPU (8 in total). We used the AdamW optimizer with a learning rate of 10^{-4} and weight decay of 5×10^{-4}. Training ran for at most 100 epochs, with 5,000 training samples per epoch. Early stopping terminated training when the validation loss did not improve for 5 consecutive epochs, with the minimum improvement threshold set to 0.

We used $K = 192$ direction codewords, which gave the highest observed overall validation F1 among the seven sizes in Appendix A.6.4. The event-aware sampler used continuation, branch, and leaf proportions of 0.2, 0.6, and 0.2, respectively (Appendix A.6.1). For direction supervision, we adopted soft focal CE with $\eta_{\mathrm{dir}} = 2$, and used focal CE for node type. For loss weighting, we used a direction–type–offset weight ratio of (1:0.5:1), as determined in Appendix A.6.3. The validation searches in Appendices A.6.2 and A.6.3 compared the loss functions and their weights. Position perturbation for refinement training used $p_{\mathrm{noise}} = 0.75$ and $\sigma_{\mathrm{noise}} = 1$ voxel, with the initial prompt kept unchanged.

A.3 Definitions of the loss functions

Notation and valid targets. We retain the notation of Section 3.4: $\widehat{\boldsymbol{\pi}}^{d}_{t+1}$ and $\widehat{\boldsymbol{\pi}}^{c}_{t+1}$ are the predicted direction and node-type distributions; $\boldsymbol{\pi}^{d,*}_{t+1}$ is the soft direction target in Eq. (17), and $c^*_{t+1} \in \{\mathrm{C}, \mathrm{B}, \mathrm{L}\}$ is the target node type. The nonnegative focal exponents are η_{dir} and η_{type}. Each $\mathcal{S}_h$ contains the valid supervised steps for head h, excluding padding and any masked targets. In particular, $\mathcal{S}_{\mathrm{dir}}$ requires at least one valid outgoing direction. We express each head's loss as the mean of its per-node loss over valid steps:

$$\mathcal{L}^a_h = \frac{1}{|\mathcal{S}_h|} \sum_{t \in \mathcal{S}_h} \ell^a_{h,t}, \qquad h \in \{\mathrm{dir}, \mathrm{type}, \mathrm{off}\}, \tag{20}$$

where a denotes the loss variant.

Direction-head losses. At the target node v corresponding to step $t+1$, let $\mathcal{C}_{t+1} \subseteq \mathrm{ch}(v)$ contain the children with valid direction labels, and define $k^*_{t+1,w} = q(\mathbf{d}_{v,w})$ using the codebook quantizer

in Section 3.2. For $t \in \mathcal{S}_{\mathrm{dir}}$, the six per-node losses in Table 8a are

$$\ell_{\mathrm{dir},t}^{\text{hard CE}} = -\frac{1}{|\mathcal{C}_{t+1}|} \sum_{w \in \mathcal{C}_{t+1}} \log \widehat{\pi}_{t+1,k^*_{t+1,w}}^{d}, \tag{21}$$

$$\ell_{\mathrm{dir},t}^{\text{hard focal CE}} = -\frac{1}{|\mathcal{C}_{t+1}|} \sum_{w \in \mathcal{C}_{t+1}} \left(1 - \widehat{\pi}_{t+1,k^*_{t+1,w}}^{d}\right)^{\eta_{\mathrm{dir}}} \log \widehat{\pi}_{t+1,k^*_{t+1,w}}^{d}, \tag{22}$$

$$\ell_{\mathrm{dir},t}^{\text{soft CE}} = -\sum_{k=1}^{K} \pi_{t+1,k}^{d,*} \log \widehat{\pi}_{t+1,k}^{d}, \tag{23}$$

$$\ell_{\mathrm{dir},t}^{\text{soft focal CE}} = -\sum_{k=1}^{K} \pi_{t+1,k}^{d,*} \left(1 - \widehat{\pi}_{t+1,k}^{d}\right)^{\eta_{\mathrm{dir}}} \log \widehat{\pi}_{t+1,k}^{d}, \tag{24}$$

$$\ell_{\mathrm{dir},t}^{\text{soft KL}} = \sum_{k=1}^{K} \pi_{t+1,k}^{d,*} \log \frac{\pi_{t+1,k}^{d,*}}{\widehat{\pi}_{t+1,k}^{d}}, \tag{25}$$

$$\ell_{\mathrm{dir},t}^{\text{soft focal KL}} = \sum_{k=1}^{K} \pi_{t+1,k}^{d,*} \left(1 - \widehat{\pi}_{t+1,k}^{d}\right)^{\eta_{\mathrm{dir}}} \log \frac{\pi_{t+1,k}^{d,*}}{\widehat{\pi}_{t+1,k}^{d}}. \tag{26}$$

Hard targets supervise every valid child, averaging their losses within each node before averaging across nodes. Children quantized to the same codeword still contribute separately. Soft targets instead use the normalized angular distributions defined in the main text. Terms with $\pi_{t+1,k}^{d,*} = 0$ contribute zero. In soft focal KL, the focal factor multiplies each codeword's KL term before summation; the resulting objective need not be a nonnegative divergence.

Relation between CE and KL. With fixed soft targets, soft CE equals soft KL plus the target entropy $-\sum_k \pi_{t+1,k}^{d,*} \log \pi_{t+1,k}^{d,*}$. They therefore have the same gradients with respect to the direction logits in exact arithmetic; Table 8 compares separately trained checkpoints. This equivalence does not extend to the focal variants, because their codeword weights depend on the predicted probabilities. Setting $\eta_{\mathrm{dir}} = 0$ recovers the corresponding non-focal objective.

Node-type losses. For the hard node-type targets described in the main text, the two per-node objectives in Table 8b are

$$\ell_{\mathrm{type},t}^{\mathrm{CE}} = -\log \widehat{\pi}_{t+1,c^*_{t+1}}^{c}, \tag{27}$$

$$\ell_{\mathrm{type},t}^{\text{focal CE}} = -\left(1 - \widehat{\pi}_{t+1,c^*_{t+1}}^{c}\right)^{\eta_{\mathrm{type}}} \log \widehat{\pi}_{t+1,c^*_{t+1}}^{c}. \tag{28}$$

The focal factor is computed from the predicted probability of the correct node type. Setting $\eta_{\mathrm{type}} = 0$ recovers CE.

Offset loss and the joint objective. Using the voxel-coordinate residuals $\boldsymbol{\delta}_{t+1} = s\rho \tanh(\widetilde{\mathbf{o}}_{t+1})$ and $\boldsymbol{\delta}^*_{t+1} = \mathbf{p}^*_{t+1} - \overline{\mathbf{p}}_{t+1}$, the adopted Smooth L1 offset loss is

$$\mathcal{L}_{\mathrm{off}} = \frac{1}{|\mathcal{S}_{\mathrm{off}}|} \sum_{t \in \mathcal{S}_{\mathrm{off}}} \ell_{\mathrm{SL1}}\left(\boldsymbol{\delta}_{t+1} - \boldsymbol{\delta}^*_{t+1}\right), \tag{29}$$

$$\ell_{\mathrm{SL1}}(\mathbf{r}) = \frac{1}{3}\sum_{j=1}^{3} h(r_j), \qquad h(r) = \begin{cases} \frac{1}{2}r^2, & |r| < 1, \\ |r| - \frac{1}{2}, & |r| \geq 1. \end{cases} \tag{30}$$

The penalty averages over coordinates, with the quadratic-to-linear transition at one voxel. With all three heads enabled, the selected ratio $1 : 0.5 : 1$ gives

$$\mathcal{L} = 0.4\mathcal{L}_{\mathrm{dir}} + 0.2\mathcal{L}_{\mathrm{type}} + 0.4\mathcal{L}_{\mathrm{off}}. \tag{31}$$

A.4 TreeTracer (sandwich): Multi-seed Inference and Structural Refinement

Motivation. TreeTracer (vanilla) reconstructs a directed neuronal subtree from a supplied seed and an initial direction. A missed bifurcation can prevent the tracer from entering an outgoing branch,

Algorithm 1 TreeTracer (sandwich): independent directed-subtree inference

Require: Image I, seed pair (s_i, q_i), pretrained tracer θ
Require: Automatic-seed budget $B = 32$, suppression radius $r_c = 5$
Ensure: Predicted directed subtree $\widehat{T}_i$

$J \leftarrow$ PREPAREIMAGE(I)
$M \leftarrow$ DETECTSOMATA(J)
$\mathcal{A} \leftarrow$ PROPOSESEEDS(J) ▷ Ordered image-only candidates and directions
$d_i \leftarrow (q_i - s_i)/\|q_i - s_i\|_2$
$T_0 \leftarrow$ TRACESEED($J, s_i, \{d_i\}, \theta$)
$\mathcal{L}_i \leftarrow [T_0]$ with empty traces omitted
$V_i \leftarrow \mathrm{nodes}(T_0)$; $b_i \leftarrow 0$
for each (a, D_a) in $\mathcal{A}$, in proposal order **do**
  if $b_i = B$ **then**
    break
  end if
  if $\|a - s_i\|_2 \le r_c$ or $\mathrm{dist}(a, V_i) \le r_c$ **then**
    continue
  end if
  $T_a \leftarrow$ TRACESEED(J, a, D_a, θ) ▷ May reuse a cached automatic trace
  if $T_a \neq \varnothing$ **then**
    Append T_a to $\mathcal{L}_i$
    $V_i \leftarrow V_i \cup \mathrm{nodes}(T_a)$; $b_i \leftarrow b_i + 1$
  end if
end for
$F_i \leftarrow$ FUSEOVERLAPS($\mathcal{L}_i$)
$F_i \leftarrow$ PRUNESHORTTERMINALS(F_i)
$F_i \leftarrow$ RESAMPLEANDPRUNETINYTERMINALS(F_i)
$F_i \leftarrow$ CONNECTENDPOINTS(F_i, I)
$F_i \leftarrow$ DEDUPLICATETERMINALS(F_i)
$F_i \leftarrow$ REFINE(F_i, M) ▷ Uses the current forest roots
$\widehat{T}_i \leftarrow$ EXTRACTDIRECTEDSUBTREE(F_i, s_i, q_i)
$\widehat{T}_i \leftarrow$ REFINE($\widehat{T}_i, M$) ▷ Uses the sample root after rerooting
return $\widehat{T}_i$

leaving its downstream arbor unrecovered. TreeTracer (sandwich) addresses this failure mode by surrounding the pretrained tracer with image-based seed augmentation and structural postprocessing. Additional seeds provide alternative entry points into visible neurites, including branches missed by the initial trace. The resulting trajectories are fused, connected where supported by the image, and refined before returning the requested subtree. The objective is to improve branch recall while controlling the redundant traces and erroneous connections introduced by additional tracing. The model parameters remain fixed; the procedure requires no additional training.

Inputs and independent-sample protocol. For sample i, the inputs are a three-dimensional image I, pretrained parameters θ, and a directed seed pair (s_i, q_i), with $d_i = (q_i - s_i)/\|q_i - s_i\|_2$. In the independent-sample setting (`batch-mode 3`), each sample uses its own supplied seed together with automatically proposed image seeds. Coverage, automatic-seed acceptance, fusion, and extraction are recomputed for each sample. Thus, samples from the same image can produce different intermediate forests. The supplied pair is read from the sample specification; in our implementation, it corresponds to the `seed_id`/`next_id` coordinates in the referenced annotation. No additional reference branches are used for tracing, fusion, or refinement. Algorithm 1 describes one sample. PREPAREIMAGE loads and intensity-normalizes the image. Spatial quantities below use voxel coordinates; the default configuration uses unit image rescaling.

Image-based seed proposals. PROPOSESEEDS selects bright voxels at or above the image's 99.7th intensity percentile, excludes a boundary margin, and retains at most 20,000 candidate voxels. Candidates are processed by decreasing intensity, with deterministic spatial tie-breaking. A

candidate is retained only if it is more than 18 voxels from previously retained proposals and passes a local foreground test. Within a radius-6 neighborhood, this test requires at least six foreground voxels and at least three voxels in the 26-connected foreground component containing the candidate. Foreground is thresholded at the larger of the global threshold and the local 90th percentile. At most 256 proposals are retained. An intensity-weighted local principal component analysis (PCA), using a radius-3 patch, supplies the leading axis v_a and the two initial directions $D_a = \{v_a, -v_a\}$. A fixed axis is used if PCA cannot be estimated. Consequently, the seed augmentation is deterministic and image-guided.

Tracing and coverage-based scheduling. TRACESEED runs the pretrained TreeTracer separately for each supplied direction. It uses moving image windows with ancestor context, stitches continuations across window boundaries, and applies overlap merging and terminal pruning after assembling each direction's trace. Directions producing fewer than two nodes after this cleanup are discarded; the remaining directional traces are combined by identifying their common seed root. If no direction survives, the seed returns an empty trace. The supplied sample seed is processed first and does not consume the automatic-seed budget. Automatic proposals are then suppressed near the supplied seed or near any node already traced for this sample, where $\mathrm{dist}(a, V) = \min_{v \in V} \|a - v\|_2$ and $\mathrm{dist}(a, \varnothing) = +\infty$. Only nonempty automatic traces consume the budget B; rejected proposals and empty traces do not. Coverage is updated before processing the next proposal. It controls seed scheduling, not termination of an ongoing trace.

Overlap fusion and endpoint connection. FUSEOVERLAPS combines the accepted traces in order, identifies geometrically overlapping paths, and coalesces their shared portions while retaining divergent continuations. With the default settings, a path match requires an overlap length of at least 5 and either an overlap fraction of at least 0.1 for one of the paths or a distance-based fiber IoU of at least 0.1; the matching distance is 3. The forest is then pruned by terminal length, resampled, and pruned by terminal node count. CONNECTENDPOINTS considers terminal-node pairs from distinct connected components. A pair is connected only when its distance is at most 5, its outward tangents are approximately opposite ($\angle(u_a, -u_b) \leq 10^\circ$), and the intervening image signal passes both support and contrast tests. These tests require a supported fraction of at least 0.75 and a median local contrast of at least 10 under the implementation's 8-bit support-image convention. Local contrast subtracts the 35th percentile of a radius-3 in-plane neighborhood; a sampled gap location is supported when its contrast exceeds $\max\{2, 0.1Q_{0.9}(|c|)\}$, where c denotes the gap contrasts. Component membership is checked again before each connection to prevent cycles, and one component is rerooted to install the connecting edge.

Terminal deduplication. DEDUPLICATETERMINALS operates on the forest after endpoint connection. It compares terminal segments of length at least 5 and removes the shorter segment only when it is largely explained by the longer one: at least 90% of its sampled points lie within distance 3, the 95th percentile of nearest-point distances is at most 3, the mean absolute tangent cosine over matched samples is at least 0.9, and both endpoints lie within 3.75 of the longer segment. The branch or root anchor is preserved when the redundant segment is removed.

Soma and turning refinement. REFINE first applies soma stopping and then removes paths with excessive turns near junctions. The soma mask is obtained from the image using background-subtracted intensity, distance-transform cores, shape checks, and persistence under a stronger intensity threshold. Along each directed path, soma stopping retains the first node entering a detected soma and removes its descendants. A root initially inside a soma may exit that initial soma once; subsequent soma entry terminates the path. Turning refinement examines root-to-leaf paths and rejects paths whose maximum local forward turn near a junction exceeds 100°. The angle uses incoming and outgoing secants over an arc-length offset of 4, evaluated at spacing 1 within distance 1.5 of a junction. Nodes shared with any retained path are preserved. In the default configuration, soma stopping is applied during postprocessing; it neither excludes soma seeds nor stops the neural decoder during tracing.

Directed subtree extraction. EXTRACTDIRECTEDSUBTREE resamples the refined forest and locates the nearest node to s_i. If no node lies within distance 4, it returns an empty prediction. Otherwise, the corresponding component is rerooted at that node, whose coordinates are anchored exactly

Table 7: Principal settings of TreeTracer (sandwich). Distances are in voxel coordinates under unit rescaling. Per-direction cleanup occurs after window stitching; cross-seed cleanup occurs after trace fusion.

Stage	Parameter	Value
Seed proposals	Maximum proposals / accepted automatic seeds	$256/32$
	Proposal spacing / coverage radius	$18/5$
Tracing	Maximum windows per direction / ancestor context	$128/12$
	Maximum decoding steps per path	256
	Maximum search paths per window	128
Per-direction cleanup	Overlap distance / minimum overlap length	$3/3$
	Overlap fraction / fiber IoU thresholds	$0.3/0.4$
	Minimum terminal length / removable-node count	$3/2$
Cross-seed cleanup	Overlap distance / minimum overlap length	$3/5$
	Overlap fraction / fiber IoU thresholds	$0.1/0.1$
	Minimum terminal length / removable-node count	$3/3$
	Resampling spacing	2
Extraction	Maximum root-snap distance / resampling spacing	$4/2$

at s_i. Among the new root's children, only the branch with the smallest angle to $q_i - s_i$ is retained, together with all its descendants. The extracted subtree is resampled at spacing 2. Soma and turning refinement are applied again because extraction changes the root and the directions of the paths being evaluated. This order is part of the algorithm: forest refinement precedes extraction, and subtree refinement follows it. An auxiliary trace contributes to the output only when it becomes part of the selected directed subtree and survives refinement.

Inference settings and reuse. The tracer uses deterministic branch search, with argmax decisions for ordinary directions and node types. Branch candidate selection uses a cumulative probability threshold of 0.5 and retains at most two directions after angular deduplication. Other principal settings are listed in Table 7. For samples sharing an image, the implementation caches the prepared image, soma mask, unfiltered proposal list, and automatic traces before cross-seed fusion. Cached traces are copied and accessed only after the current sample's suppression and budget checks. The supplied-seed trace, accepted trace list, coverage, budget, and merged forest remain sample-specific. This reuse memoizes fixed-input tracing computations without sharing sample-dependent inference decisions.

A.5 Model hyperparameters

We used a hidden feature dimension of 192 for the fused image features, global context tokens, and trajectory representations. The image encoder comprised three residual 3-D convolutional stages with channel widths $(16, 32, 64, 192)$ and strides $(1, 1, 1)$, $(1, 2, 2)$, $(1, 2, 2)$, and $(1, 2, 2)$, respectively, in depth–height–width order. These strides preserve depth resolution while downsampling the two in-plane axes. Multi-scale features were fused on the finest grid and projected to 192 channels. The global attention pooling module used 128 learnable queries to produce 128 compact feature tokens, each of dimension 192, giving $\mathbf{G} \in \mathbb{R}^{128\times192}$ for each input crop.

The causal Transformer decoder contained four layers, with four attention heads (48 dimensions per head) and a dropout rate of 0.1. Each layer applied local cross-attention followed by global cross-attention, as described in Section 3.3. Local feature sampling used a radius of 3 on the dense feature grid, corresponding to a $7 \times 7 \times 7$ window around the coarse next-position proposal. The positional offset head was enabled with correction ratio $\rho = 1$ in Eq. (6).

Point positions used Fourier embeddings with 32 frequency bands, shared with the image positional embeddings. Trajectory tokens also included learned direction and node-type embeddings and sinusoidal step embeddings. Training sequences were capped at 256 steps.

A.6 HYPERPARAMETER SEARCH

We searched event-sampling probabilities, loss functions, relative loss weights, and direction-codebook size on the CWMBS validation set. Each search used the fiber evaluation protocol in Appendix C, reporting overall performance and the no-branch and branch groups. Error bars and subscripts denote standard deviations from 1,000 source-image bootstrap resamples for one checkpoint per setting.

A.6.1 EVENT-SAMPLING PROBABILITIES

Setup. Branching and termination decisions determine tree connectivity and path endpoints. To study their sampling balance, we fixed the leaf probability at $p_{\mathrm{L}} = 0.2$, varied the branch probability $p_{\mathrm{B}} \in \{0, 0.2, 0.4, 0.6, 0.8\}$, and set the continuation probability to $p_{\mathrm{C}} = 0.8 - p_{\mathrm{B}}$. Natural sampling follows approximately $(p_{\mathrm{C}}, p_{\mathrm{B}}, p_{\mathrm{L}}) = (0.66, 0.17, 0.17)$. All six runs used the full decoder with $K = 256$ direction codewords and were evaluated on the CWMBS validation set under the same protocol.

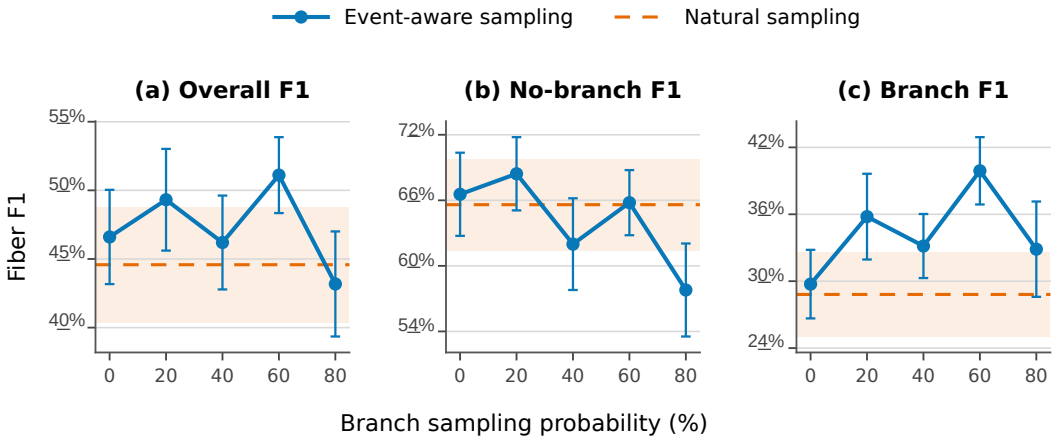

Figure 4: **Event-sampling search on the CWMBS validation set.** Fiber F1 was pooled over all validation subtrees or within the no-branch and branch groups. Blue curves vary the branch sampling probability with the leaf probability fixed at 20%; orange dashed lines show natural sampling.

Results. Figure 4 shows that $p_{\mathrm{B}} = 0.6$ gave the highest overall and branch F1 (approximately 51.1% and 39.9%), compared with 44.6% and 28.8% under natural sampling. No-branch F1 peaked at $p_{\mathrm{B}} = 0.2$ (approximately 68.4%). Increasing the branch probability from 0.6 to 0.8 lowered F1 in all three groups. We therefore adopted $(p_{\mathrm{C}}, p_{\mathrm{B}}, p_{\mathrm{L}}) = \mathbf{(0.2, 0.6, 0.2)}$ for its highest observed overall and branch F1 on the validation set.

A.6.2 LOSS FUNCTIONS

Setup. We compared candidate loss functions for the direction and node-type heads. All runs used the full decoder with $K = 256$ direction codewords. The checkpoints shared the same architecture and inference configuration. We evaluated the same 445 CWMBS validation subtrees from 37 source images using the protocol in Appendix C. Table 8 compares six direction losses with focal cross-entropy (CE) for node type, then compares CE and focal CE for node type with soft focal CE for direction. KL denotes Kullback–Leibler divergence. Appendix A.3 gives the formulas and reductions for the compared losses.

Direction-head loss. Table 8a shows that focal weighting improved CE with either hard or soft targets: overall F1 rose from 44.0% to 49.7% for hard targets and from 44.2% to 51.4% for soft targets. Soft focal CE achieved the highest overall F1 (51.4%), no-branch F1 (68.9%), and branch-group F1 (38.0%). It also had the highest overall precision (58.8%) and recall (45.6%). Hard focal CE attained overall F1 of 49.7%, with no-branch and branch F1 of 65.1% and 37.2%. Soft KL and

Table 8: **Loss-function search on the CWMBS validation set.** Fiber micro-averaged P, R, and F1 (↑), grouped by reference topology. Subscripts denote bootstrap SDs. Bold scores mark column best within each panel, ranked before rounding. The last row of each panel uses the selected loss functions; both rows report the same checkpoint.

Loss function	**No-branch** ($n = 294$)			**Branch** ($n = 151$)			**All** ($n = 445$)		
	P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)
(a) Direction-head loss (type head: focal CE)									
Hard CE	$58.0_{5.2}$	$66.3_{3.6}$	$61.9_{4.2}$	$36.5_{11.8}$	$27.7_{2.7}$	$31.5_{5.0}$	$46.5_{7.8}$	$41.8_{3.2}$	$44.0_{4.7}$
Hard focal CE	$60.8_{5.3}$	$70.1_{4.1}$	$65.1_{4.6}$	$\mathbf{55.0}_{6.3}$	$28.1_{2.6}$	$37.2_{3.4}$	$58.2_{4.9}$	$43.4_{3.6}$	$49.7_{3.8}$
Soft CE	$58.0_{3.3}$	$62.6_{3.4}$	$60.2_{3.1}$	$44.3_{6.8}$	$25.0_{2.3}$	$31.9_{3.3}$	$51.5_{4.5}$	$38.7_{2.8}$	$44.2_{3.3}$
Soft KL	$56.0_{5.4}$	$65.0_{3.7}$	$60.2_{4.4}$	$49.8_{6.2}$	$28.8_{2.0}$	$36.5_{2.6}$	$53.1_{4.7}$	$42.0_{2.6}$	$46.9_{3.0}$
Soft focal KL	$56.3_{6.8}$	$71.4_{3.4}$	$63.0_{5.2}$	$37.8_{8.0}$	$27.1_{3.0}$	$31.6_{4.4}$	$47.1_{7.1}$	$43.2_{3.7}$	$45.1_{5.0}$
Soft focal CE	$\mathbf{65.7}_{5.2}$	$\mathbf{72.4}_{3.5}$	$\mathbf{68.9}_{4.2}$	$51.3_{8.3}$	$\mathbf{30.2}_{2.5}$	$\mathbf{38.0}_{3.5}$	$\mathbf{58.8}_{5.4}$	$\mathbf{45.6}_{3.1}$	$\mathbf{51.4}_{3.6}$
(b) Type-head loss (direction head: soft focal CE)									
CE	$46.1_{6.4}$	$67.0_{3.6}$	$54.6_{5.2}$	$39.7_{7.2}$	$\mathbf{32.0}_{2.5}$	$35.4_{3.8}$	$43.0_{5.8}$	$44.7_{2.9}$	$43.8_{4.0}$
Focal CE	$\mathbf{65.7}_{5.2}$	$\mathbf{72.4}_{3.5}$	$\mathbf{68.9}_{4.2}$	$\mathbf{51.3}_{8.3}$	$30.2_{2.5}$	$\mathbf{38.0}_{3.5}$	$\mathbf{58.8}_{5.4}$	$\mathbf{45.6}_{3.1}$	$\mathbf{51.4}_{3.6}$

soft focal KL had lower overall F1 (46.9% and 45.1%, respectively). We adopted **soft focal CE**, defined in Eq. (24), for its best observed overall and branch F1. Its gains over hard focal CE were 1.6 and 0.9 percentage points, respectively, computed before rounding.

Node-type loss. Using focal CE for the type head improved branch-group F1 from 35.4% to 38.0% and overall F1 from 43.8% to 51.4% (Table 8b). The branch-group gain is driven primarily by precision (39.7% to 51.3%), despite lower recall (32.0% to 30.2%): false-positive fibers decreased from 249 to 147, while true positives decreased from 164 to 155. No-branch F1 also improved from 54.6% to 68.9%. These results are consistent with more reliable branching decisions from focal weighting of difficult node-type predictions, although the reported fiber metrics do not directly measure branch-point classification accuracy. We therefore adopted **focal CE** for node type, as defined in Eq. (28).

A.6.3 Loss weights

Setup. We compared seven ratios of the direction, node-type, and offset loss weights in Eq. (18), always written in the order $\lambda_{\text{dir}} : \lambda_{\text{type}} : \lambda_{\text{off}}$. The loss functions were soft focal CE for direction, focal CE for node type, and Smooth L1 for offset. All runs used the full decoder with $K = 256$, continuation/branch/leaf sampling proportions of $(0.4, 0.4, 0.2)$, and the same CWMBS validation samples and evaluation settings as the loss-function study above. The joint objective is normalized by the sum of the three weights, so the ratios specify their relative contributions.

Effect of relative weighting. Table 9 shows that $1 : 0.5 : 0.5$ achieved the highest overall F1 (51.4%), no-branch F1 (68.9%), and branch-group F1 (38.0%) among the seven ratios on the validation set. Compared with equal weights ($1 : 1 : 1$), it improved overall and branch F1 by 7.9 and 5.6 percentage points, respectively, computed before rounding, and raised no-branch F1 from 59.6% to 68.9%. Overall precision rose from 43.4% to 58.8% and recall from 43.5% to 45.6%. Increasing the type or offset weight to 2 yielded overall F1 of 43.6% or 45.1%, while halving only the direction or offset weight yielded 40.4% or 41.8%, respectively.

Comparison with the adopted ratio. The adopted $\mathbf{1 : 0.5 : 1}$ ratio ranked second in overall validation F1 (50.3%), with no-branch and branch F1 of 66.6% and 37.2%. It achieved the highest overall precision (61.7%) and branch precision (58.3%). Compared with $1 : 0.5 : 0.5$, increasing the offset weight from 0.5 to 1 lowered overall and branch recall from 45.6% to 42.5% and from

Table 9: **Loss-weight search on the CWMBS validation set.** Weights are ordered as direction:type:offset. Fiber micro-averaged P, R, and F1 (↑) use the same topology groups and bootstrap standard-deviation subscripts as Table 8. Bold scores mark column best before rounding; the last row gives the adopted ratio.

Loss weights	**No-branch** ($n = 294$)			**Branch** ($n = 151$)			**All** ($n = 445$)		
	P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)
1 : 1 : 1	$53.8_{4.9}$	$66.7_{3.4}$	$59.6_{4.1}$	$34.9_{7.8}$	$\mathbf{30.2}_{2.8}$	$32.4_{3.8}$	$43.4_{6.0}$	$43.5_{3.0}$	$43.5_{3.9}$
1 : 1 : 2	$57.8_{4.4}$	$65.3_{3.6}$	$61.3_{3.8}$	$47.9_{9.2}$	$24.0_{2.0}$	$31.9_{3.1}$	$53.5_{4.9}$	$39.0_{2.9}$	$45.1_{3.2}$
1 : 2 : 1	$56.5_{3.6}$	$66.3_{3.7}$	$61.0_{3.5}$	$35.0_{7.9}$	$29.2_{2.4}$	$31.8_{3.9}$	$44.6_{5.8}$	$42.8_{3.0}$	$43.6_{3.8}$
0.5 : 1 : 1	$49.7_{6.2}$	$63.6_{4.3}$	$55.8_{5.1}$	$32.8_{9.5}$	$26.5_{2.5}$	$29.3_{4.2}$	$40.8_{6.5}$	$40.0_{3.3}$	$40.4_{4.3}$
1 : 1 : 0.5	$50.3_{5.6}$	$66.3_{3.5}$	$57.2_{4.7}$	$33.0_{6.4}$	$29.0_{1.7}$	$30.9_{3.2}$	$41.0_{5.0}$	$42.6_{2.6}$	$41.8_{3.6}$
1 : 0.5 : 0.5	$\mathbf{65.7}_{5.2}$	$\mathbf{72.4}_{3.5}$	$\mathbf{68.9}_{4.2}$	$51.3_{8.3}$	$\mathbf{30.2}_{2.5}$	$\mathbf{38.0}_{3.5}$	$58.8_{5.4}$	$\mathbf{45.6}_{3.1}$	$\mathbf{51.4}_{3.6}$
1 : 0.5 : 1	$64.2_{6.0}$	$69.0_{3.3}$	$66.6_{4.4}$	$\mathbf{58.3}_{5.5}$	$27.3_{1.9}$	$37.2_{2.4}$	$\mathbf{61.7}_{4.2}$	$42.5_{2.7}$	$50.3_{3.2}$

30.2% to 27.3%, respectively. Overall and branch F1 decreased by 1.0 and 0.9 percentage points, computed before rounding. As in the loss-function study, the reported uncertainty reflects validation-image resampling for one checkpoint per ratio.

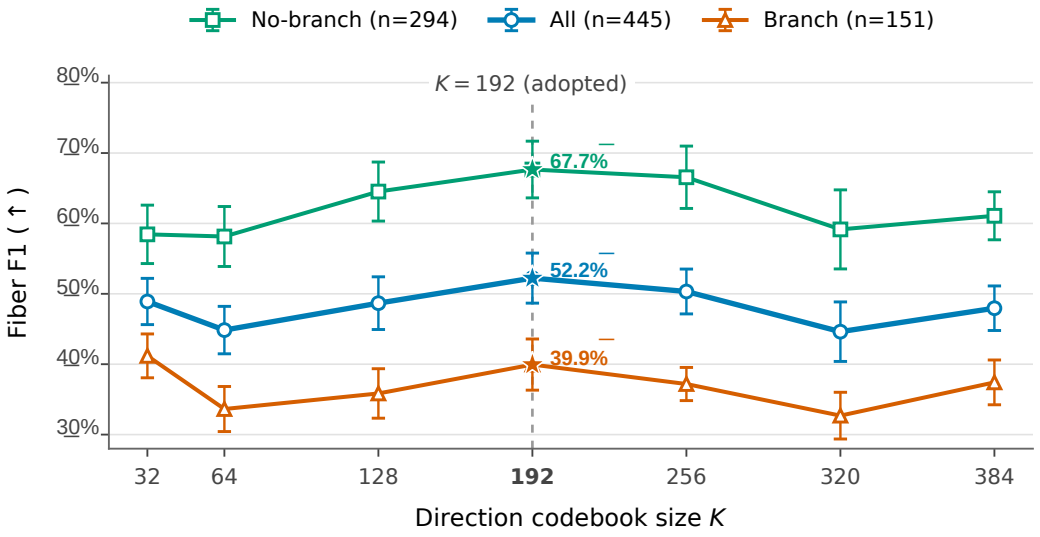

Figure 5: **Direction-codebook size search on the CWMBS validation set.** Curves show fiber micro F1 for all validation subtrees and for the no-branch and branch groups, using a shared vertical scale. Stars mark the adopted $K = 192$. The highest observed overall and no-branch F1 occurred at $K = 192$; branch F1 was highest at $K = 32$.

A.6.4 Direction codebook size

Setup. We compared seven Fibonacci-sphere direction codebooks with $K = 32$, 64, 128, 192, 256, 320, and 384 to examine the effect of angular discretization. All seven configurations used the full decoder with hidden dimension 192, and the same inference settings. We evaluated the same 445 CWMBS validation subtrees from 37 source images using the protocol in Appendix C: 294 have no reference branch nodes and 151 contain at least one. The adopted model uses $K = 192$.

Results. Figure 5 shows that increasing codebook size did not monotonically improve reconstruction. Overall F1 was 48.9%, 44.8%, 48.7%, 52.2%, 50.3%, 44.6%, and 48.0% for $K = 32$, 64, 128, 192, 256, 320, and 384, respectively. The highest overall F1 occurred at $K = 192$, exceeding the next-best size, $K = 256$, by 1.9 percentage points (computed before rounding). The preferred size differed across topology groups: $K = 192$ had the highest no-branch F1 (67.7%), whereas $K = 32$ had the highest branch F1 (41.2%). The adopted $K = 192$ reached branch F1 of 39.9%. Relative to $K = 32$, $K = 192$ had higher overall precision (63.8% versus 54.3%) with nearly unchanged recall (44.2% versus 44.5%), yielding the highest overall F1 among the tested sizes.

Angular resolution and supervision noise. Codebook size controls both angular quantization and the granularity of direction supervision. A small codebook offers coarse directions, whose residual angular errors can accumulate over successive tracing steps. A larger codebook reduces quantization error but introduces more direction classes with smaller angular separation. For a fixed level of positional noise in the annotations, direction uncertainty can span more neighboring classes as the codebook becomes finer, potentially making supervision less separable. These are possible mechanisms rather than effects directly measured by the reconstruction metrics. The validation results show no consistent benefit from increasing K, and the best size differed between the two topology groups. We adopted $\mathbf{K = 192}$ for its highest observed overall and no-branch validation F1. The error bars quantify validation-image sampling variability for one checkpoint per size.

Algorithm 2 Large-volume inference by boundary continuation

Require: Volume I; oriented seed P_0; local tracer $\mathcal{F}_\theta$; crop shape $\mathbf{w}$
Require: History limit h; tail exclusion e; boundary margin δ_b;
node threshold $n_{\min}$; window budget B_{win}
Ensure: Reconstructed subtree $\widehat{\mathcal{T}}$ in global coordinates

$\mathcal{T} \leftarrow \varnothing$; $\mathcal{Q} \leftarrow [(P_0, \perp)]$; $\mathcal{D} \leftarrow \varnothing$; $n \leftarrow 0$
while $\mathcal{Q} \neq \varnothing$ **and** $n < B_{\text{win}}$ **do**
  $q \leftarrow \text{PopFront}(\mathcal{Q})$
  if $\neg\, \text{Admit}(q; \mathcal{D})$ **then**
    continue
  end if
  $(W, P) \leftarrow \text{PlaceWindow}(I, q, \mathbf{w})$
  $\mathcal{S} \leftarrow \mathcal{F}_\theta(I_W, \text{ToLocal}(P, W))$; $n \leftarrow n + 1$
  if $\mathcal{S} = \varnothing$ **then**
    continue
  end if
  $\mathcal{S} \leftarrow \text{ToGlobal}(\mathcal{S}, W)$
  $(\mathcal{T}, \phi) \leftarrow \text{Stitch}(\mathcal{T}, \mathcal{S}, q.u)$
  if $\neg\, \text{CanContinue}(\mathcal{S}, W, n_{\min})$ **then**
    continue
  end if
  $\mathcal{C} \leftarrow [\,]$
  for all $v \in \text{BoundaryLeaves}(\mathcal{S}, W, I, \delta_b)$ **do**
    $P_v \leftarrow \text{ContinuationPrompt}(\mathcal{S}, v, h, e)$
    APPEND($\mathcal{C}$, $(P_v, \phi(v))$)
  end for
  ENQUEUEFRONTIERS($\mathcal{Q}, \mathcal{C}, \mathcal{D}$)
end while
return $\widehat{\mathcal{T}} \leftarrow \text{GlobalPostprocess}(\mathcal{T})$

A.7 LARGE-VOLUME INFERENCE

TreeTracer reconstructs volumes larger than its input crop by carrying predicted paths across crop boundaries. A first-in, first-out (FIFO) queue schedules these continuations, while a global tree stores their geometry and connectivity. This procedure extends the local tracing policy in Section 3.2; the model's global image context remains confined to the current crop. We describe the default moving-window subtree evaluation, with optional branch recovery disabled.

Local inference and frontier state. Let $\mathcal{F}_\theta(I_W, P)$ denote local subtree inference on crop I_W, conditioned on $P = (H, \mathbf{a}, \mathbf{d})$: an ordered history H of point positions and node types, a tracing anchor $\mathbf{a}$, and its outgoing direction $\mathbf{d}$. The returned subtree starts at the anchor and is then merged into the global tree. Initially, $H_0 = \varnothing$ and the anchor and direction are given by the oriented seed $(\mathbf{p}_0, \mathbf{d}_0)$ in Eq. (1). Each queue record $q = (P, u)$ also stores the global attachment node u; $u = \perp$ denotes initialization. Algorithm 2 summarizes the procedure.

Boundary continuation. BOUNDARYLEAVES selects structural leaves, i.e., nodes with no children, that remain inside the full image domain Ω_I and reach a movable crop face within margin δ_b. A face is movable when image content exists beyond it along the outward axis. Thus, a crop boundary can trigger continuation even when the local type head predicts a leaf; reaching the full-volume boundary does not launch an outward continuation. CANCONTINUE permits expansion if the local tree has more than $n_{\min}$ nodes or contains a point outside the crop box. The second condition preserves boundary handoffs represented by out-of-crop points. Trees failing this test are still stitched into the reconstruction, but do not schedule further windows.

History transfer and crop placement. For each eligible endpoint v, we follow the local parent chain ending at v, retain up to h most recent points, and remove the last e to form H_v. The anchor remains $\mathbf{x}_v$, and its direction follows the displacement from its immediate parent, computed before truncation. Hence, excluding the history tail does not move the attachment point. The initial crop samples a valid seed placement using the training margin and a fixed random seed. A continuation crop is centered on the newest retained history point, or the anchor if the history is empty, subject to crop and image bounds. Placement retains the newest history suffix that fits together with the anchor. The same coordinate transform is applied to the prompt and image crop, preserving their spatial correspondence.

Stitching and duplicate suppression. After converting a local tree to global coordinates, STITCH identifies its root with the existing attachment node u, removes the duplicate root, and attaches its children to u. For the initial prediction, the entire local tree initializes $\mathcal{T}$. The map ϕ records the resulting global identifiers, including the shared root. ADMIT checks and updates the visited state $\mathcal{D}$ using rounded endpoint positions, while preserving any pending type-confirmation state. ENQUEUEFRONTIERS retains distinct headings for direct terminal siblings from the same local prediction that share a rounded endpoint; their provenance is recorded for subsequent admission checks. This exception prevents position-based deduplication from discarding distinct sibling continuations at a shared location.

Termination and post-processing. Expansion stops when the queue is empty or B_{win} local calls have been made; an empty local prediction also consumes one call. Budget exhaustion returns the reconstruction accumulated so far. Global post-processing then merges overlapping branches and prunes short terminal branches. Coordinate normalization and any local post-processing are encapsulated in $\mathcal{F}_\theta$. Every call uses the same crop shape, so the model's image input size is independent of the full volume size; the stored global tree and frontier queue grow with the reconstruction.

Evaluation settings. The CWMBS and CORAL vanilla runs used the shared settings in Table 10. These were applied to the moving-window subtree evaluations in Tables 1 and 2. The history limit h is separate from the direction-codebook size K. All spatial thresholds are expressed in native voxel coordinates.

B DATASET PREPARATION AND STATISTICS

B.1 SOURCE DATA AND SUBTREE CONSTRUCTION

CWMBS. The Complex Whole Mouse Brain Sub-Image dataset (CWMBS) contains 245 expertly annotated blocks cropped from a whole mouse-brain image, comprising 83 blocks with cluttered background noise and 162 blocks with thin filamentary structures (Liu et al., 2024b). At its native resolution of $0.2 \times 0.2 \times 1\,\mu\text{m}$, a 256^3-voxel block covers $51.2 \times 51.2 \times 256\,\mu\text{m}$.

Table 10: **Moving-window settings for vanilla subtree evaluation.** The history limit is applied before tail exclusion and crop fitting.

Parameter	Value
Crop shape $\mathbf{w}$, in (Z, Y, X) order	$32 \times 96 \times 96$ voxels
History limit h / excluded tail e	12 / 4 points
Boundary margin δ_b	3 voxels
Continuation threshold $n_{\min}$	4 nodes
Window budget B_{win}	128 local calls
Random seed	42
Frontier deduplication	Rounded position, with sibling exception
Global overlap merging / terminal pruning	Enabled / enabled

CORAL. CORAL provides whole-brain fluorescence micro-optical sectioning tomography (fMOST) images from a mouse brain, with 32 neurons annotated through initial tracing, independent review, and a third annotator's global cross-validation (Yang et al., 2026). Its local blocks are cropped from the whole-brain volume. Each 300^3-voxel block covers $105 \times 105 \times 300\,\mu\text{m}$ at $0.35 \times 0.35 \times 1\,\mu\text{m}$ resolution.

Subtree preparation. We constructed rooted subtree targets from the source SWC annotations. The preparation used a resampling interval of 2 voxels, a soma-proximal trimming distance of 50 voxels for soma-rooted components, and a minimum retained tree length of 5 voxels. These distances are measured in the native SWC coordinate system. Each sample stores its source block, root position, and outgoing root edge, from which the point-and-direction prompt is obtained. Multiple samples may originate from the same block; all such samples remained in the same training, validation, or test partition. We classified each reference subtree by the number of nodes with at least two children: no branch, one branch, or multiple branches. The last two classes are combined into the *Branch* group in the main comparisons.

Partitions and the evaluation subset. Table 11 reports counts from the split manifests used in this work. CWMBS has 171/37/37 source blocks and 2,378/445/483 subtrees for training/validation/testing. CORAL's prepared version has 767 training blocks and 140 validation blocks, with 3,169 and 551 subtrees, respectively. For local evaluation, we used only the fixed subset of 500 subtrees from CORAL's test set, drawn from 349 blocks and specified by the test manifest. This subset contains 359 no-branch and 141 branching subtrees, with a branching proportion of 28.2%, compared with 30.6% for CWMBS. All CORAL test statistics and local reconstruction results in this paper refer to this 500-subtree subset. Source-block identities are disjoint across the training, validation, and evaluated test samples.

Table 11: **Prepared subtree datasets.** Blocks are unique source images; subtrees are reconstruction samples. B denotes the number of reference branch nodes. Fiber counts exclude root-to-leaf paths shorter than 5 voxels. CORAL Test (500) denotes the 500-subtree subset used for local evaluation.

Dataset	Split	Blocks	Subtrees	$B = 0$	$B = 1$	$B \geq 2$	Fibers
CWMBS	Train	171	2,378	1,630	380	368	4,420
CWMBS	Validation	37	445	294	86	65	807
CWMBS	Test	37	483	335	64	84	899
CORAL	Train	767	3,169	2,256	562	351	5,164
CORAL	Validation	140	551	392	92	67	868
CORAL	Test (500)	349	500	359	81	60	777

B.2 Subtree lengths by branching class

Length definition and grouping. We computed lengths directly from the ground-truth subtree SWCs referenced by the test manifests, using native voxel coordinates without physical scaling.

Table 12: **Test-set native-coordinate lengths by branching class.** Each subtree is one observation. No-branch length is the single root-to-leaf path length; branch length sums all subtree edges, counting each edge once. Lengths are in voxels. P90 denotes the 90th percentile. Only the evaluated test samples are included: 483 CWMBS subtrees and the 500-subtree CORAL subset.

Dataset	Group	n	Mean	Median	P90
CWMBS	No-branch	335	132.5	105.2	279.6
CWMBS	Branch	148	557.6	448.8	1037.1
CORAL	No-branch	359	163.5	131.5	359.8
CORAL	Branch	141	428.9	411.8	688.2

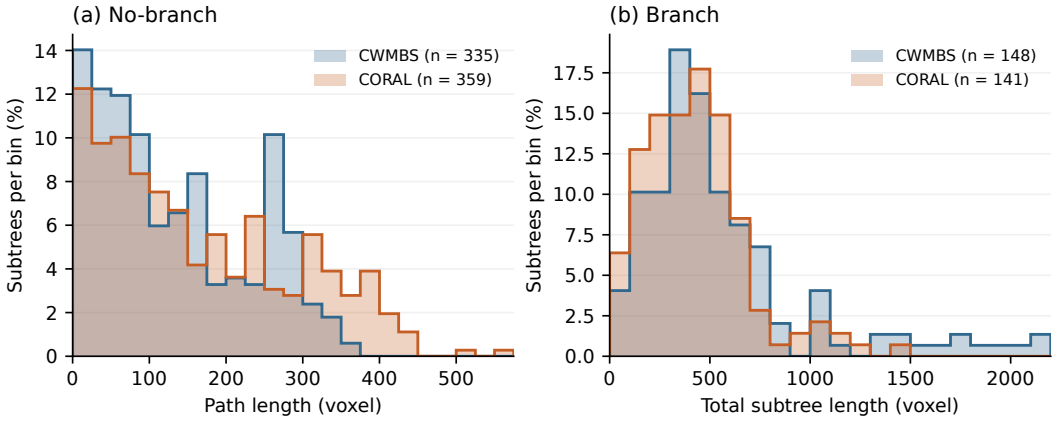

Figure 6: **Test-set length distributions by branching class.** (a) No-branch targets: the length of the single root-to-leaf path. (b) Branching targets: the total length of the entire subtree, counting each edge once. All lengths use native voxel coordinates. Both datasets share bin edges within each panel, with widths of 25 and 100 voxels, respectively. Heights show the percentage of subtrees within each dataset and group; n is the number of subtrees. CORAL uses only the 500-subtree test subset (359 no-branch and 141 branching targets).

Each subtree contributes one observation. For a target $\mathcal{T} = (\mathcal{V}, \mathcal{E})$, its length is

$$L_{\text{vox}}(\mathcal{T}) = \sum_{(u,v)\in\mathcal{E}} \|\mathbf{x}_v - \mathbf{x}_u\|_2. \tag{32}$$

For a no-branch subtree, this is the arc length of its single root-to-leaf path. For a branching subtree, it is the total length of the entire subtree, with every edge, including shared trunk edges, counted exactly once. We included all edges of each prepared target; the minimum-fiber-length filter used for Fiber evaluation did not remove edges from this statistic. The two groups follow the reference branch labels used in the main result tables. Both Table 12 and Figure 6 include only the evaluated test samples: 483 CWMBS subtrees and the 500-subtree CORAL test subset.

Comparison between datasets. Table 12 and Figure 6 show different length profiles for the two groups. Among no-branch test targets, CORAL has a median path length of 131.5 voxels, compared with 105.2 voxels for CWMBS ($1.25\times$); their 90th percentiles are 359.8 and 279.6 voxels, respectively. Among branching targets, the median total subtree length is 411.8 voxels for CORAL and 448.8 voxels for CWMBS. CWMBS has a heavier upper tail in this group, with a 90th percentile of 1,037.1 voxels versus 688.2 voxels for CORAL. Thus, CORAL's no-branch targets require longer paths, while CWMBS includes branching targets with greater total reconstruction length. Reporting the two groups separately distinguishes path continuation length from the amount of arbor that must be reconstructed across multiple branches.

B.3 CORAL BRAIN-WIDE RECONSTRUCTION DATA

CORAL additionally supports reconstruction beyond an individual image block. Its published protocol uses left-hemisphere data for training and reserves right-hemisphere data for testing (Yang et al., 2026). Of the 17 neurons with somata in the right hemisphere, three extend into the training hemisphere and are excluded from the brain-wide test set, leaving 14 neurons. The benchmark reports target reconstruction lengths from 1,404 to $86,680\,\mu\mathrm{m}$ across these neurons, with 746 dendritic and 2,785 axonal root-to-leaf fibers in total. These are properties of CORAL's brain-wide targets, distinct from the cropped subtree statistics above. CORAL's whole-brain tracing framework supplies iterative local reconstruction and integration into a global tree from soma initializations, allowing methods to be compared on long-range continuity and complete neuron recovery.

C FIBER EVALUATION PROTOCOL

Fiber overlap and assignment. Following Yang et al. (2026), we decompose the predicted and reference trees for sample i into root-to-leaf fiber sets $\mathcal{F}_{p,i}$ and $\mathcal{F}_{g,i}$. Fibers are resampled along arc length with interval h_f. For a resampled fiber f, let Z_f be its ordered point sequence, c_j the midpoint of segment j, and δ_j its length. Directed overlap counts the length of segments whose midpoints lie within distance τ_f of the other fiber's resampled points. We symmetrize this overlap and compute the fiber intersection-over-union (FIoU) using fiber lengths $L(f)$ and $L(g)$:

$$O(f \to g) = \sum_j \delta_j \, \mathbb{I}\left[\min_{z \in Z_g} \|c_j - z\|_2 \le \tau_f\right], \tag{33}$$

$$O(f, g) = \tfrac{1}{2}\big(O(f \to g) + O(g \to f)\big), \tag{34}$$

$$\mathrm{FIoU}(f, g) = \frac{O(f, g)}{L(f) + L(g) - O(f, g) + \epsilon_{\mathrm{IoU}}}. \tag{35}$$

The Hungarian algorithm determines a one-to-one assignment $\mathcal{M}_i$ maximizing total FIoU. An assigned pair is a true positive only if its FIoU is at least η:

$$\mathrm{TP}_i = \sum_{(f,g) \in \mathcal{M}_i} \mathbb{I}[\mathrm{FIoU}(f, g) \ge \eta], \qquad \mathrm{FP}_i = |\mathcal{F}_{p,i}| - \mathrm{TP}_i, \qquad \mathrm{FN}_i = |\mathcal{F}_{g,i}| - \mathrm{TP}_i. \tag{36}$$

Thus, unmatched fibers and assigned pairs below the threshold are penalized. A fragmented trajectory or a path that switches to a neighboring neurite receives credit only when a single predicted root-to-leaf path preserves sufficient overlap with the reference.

Subtree settings. The local evaluation used an initial tree resampling interval of 2 voxels, $h_f = 1$ voxel, $\tau_f = 5$ voxels, a minimum fiber length of 5 voxels, and $\eta = 0.8$. Both datasets used coordinate scale $(1, 1, 1)$ for evaluation: FIoU was therefore computed in native voxel coordinates. The minimum-fiber-length filter excluded one very short root-to-leaf path in each evaluated test set. The subtree-length statistics in Appendix B.2 count each complete target once. We used $\epsilon_{\mathrm{IoU}} = 10^{-7}$ as in CORAL. The brain-wide comparison used $\eta = 0.75$, $h_f = 1$ voxel, $\tau_f = 5$ voxels, a minimum fiber length of 5 voxels, and coordinate scale $(1, 1, 1)$. Trees were initially resampled at 2-voxel intervals. Axon and dendrite F1 were computed by matching fibers within each anatomical type; the overall Fiber F1 pools the full set of fibers.

Aggregation and uncertainty. We compute micro-averaged precision, recall, and F1 by pooling fiber counts across evaluated samples:

$$P = \frac{\sum_i \mathrm{TP}_i}{\sum_i (\mathrm{TP}_i + \mathrm{FP}_i)}, \qquad R = \frac{\sum_i \mathrm{TP}_i}{\sum_i (\mathrm{TP}_i + \mathrm{FN}_i)}, \qquad F1 = \frac{2PR}{P + R + 10^{-12}}. \tag{37}$$

Precision or recall is set to zero when its denominator is zero. Reported P, R, and F1 values are these fractions multiplied by 100; their bootstrap SDs and absolute differences use percentage points. The same computation is performed separately for the no-branch and branch groups, using the reference topology to define group membership. For uncertainty in the local comparisons, we resampled source images with replacement and retained all evaluated subtrees from each sampled image, including multiplicities. We recomputed the pooled metrics for each of 1,000 replicates, using random

seed 42. Subscripts in the result tables denote the sample standard deviation (SD) across replicates. The resampling units were the 37 CWMBS test images and the 349 images represented in the CORAL test subset; the hyperparameter searches in Appendix A.6 resampled CWMBS validation images instead.

D TRACING SPEED

Evaluation and metric. We measured tracing speed on the same 483 CWMBS test prompts from 37 source images for TreeTracer (vanilla) and six two-stage pipelines combining SegMamba or GBP-Net with APP2, Kimimaro, or neuTube. All runs completed all 483 prompts. We report predicted path length per second,

$$v_{\mathrm{trace}} = \frac{\sum_j L_j}{\sum_j t_j} \quad (\mathrm{voxel/s}), \tag{38}$$

where L_j is the total length of the returned subtrees for source image j, computed by applying Eq. 32 to each predicted tree, and t_j is its accumulated tracing time. Lengths use native voxel coordinates, with overlapping predictions from different prompts counted separately. The ratio pools lengths and times across images; it measures generated path length, with reconstruction accuracy evaluated separately in Table 1.

Timing scope and hardware. For TreeTracer, timing covers CUDA-synchronized subtree tracing, including window cropping, data transfers, model inference, merging, and postprocessing. For two-stage pipelines, timing includes one segmentation pass per image and the sum of reconstruction-request durations for that image. Each request includes server queueing, image reading, reconstruction, subtree extraction, and response I/O. Model loading, initial image loading, local output writing, and metric evaluation are excluded from these timed stages. All neural inference used NVIDIA GeForce RTX 3090 GPUs.

Table 13: **Tracing speed on CWMBS.** Predicted length and accumulated time are totals over 483 test prompts from 37 images. Speed is their ratio under the timing scope above; **bold** marks the highest value.

Method	Length (voxels)	Time (s)	Speed (voxel/s) ↑
APP2 (SegMamba)	46,749.15	467.40	100.02
Kimimaro (SegMamba)	22,201.58	240.43	92.34
neuTube (SegMamba)	77,776.60	751.38	103.51
APP2 (GBP-Net)	76,693.72	1,396.51	54.92
Kimimaro (GBP-Net)	108,605.81	954.42	113.79
neuTube (GBP-Net)	187,528.03	2,038.37	92.00
TreeTracer (vanilla)	82,151.61	630.06	**130.39**

Results. Table 13 shows that TreeTracer (vanilla) achieved 130.39 voxel/s, the highest recorded rate among the seven methods. Kimimaro (GBP-Net), the strongest baseline by overall Fiber F1 in Table 1, reached 113.79 voxel/s, giving TreeTracer a 14.6% higher rate ($1.15\times$). Across the six baselines, the observed throughput ratios ranged from $1.15\times$ to $2.37\times$ under these timing definitions.

E DECODER ATTENTION ABLATIONS

Configurations and evaluation. We enabled or disabled causal self-attention (S), local cross-attention (L), and global cross-attention (G) in each of the four decoder layers. S aggregates the trajectory prefix, L reads image features near the proposed next position, and G accesses image context across the input patch. Disabling S retained the trajectory-token embeddings; the ablation removed attention across history tokens, not their per-step content. All seven configurations used $K = 192$ direction codewords, hidden dimension 192, the same 483 CWMBS test subtrees from 37 source images, and the evaluation settings in Appendix C.

Table 14: **Complete decoder attention ablations on CWMBS.** Fiber P, R, and F1 (↑), grouped by reference topology. S/L/G indicate the enabled attention sublayers. Subscripts denote bootstrap SDs; bold marks column best.

Attention			**No-branch** ($n = 335$)			**Branch** ($n = 148$)			**All** ($n = 483$)		
S	L	G	P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)
✓	✓	✓	$\mathbf{67.3}_{4.3}$	$73.1_{3.7}$	$\mathbf{70.1}_{3.8}$	$61.4_{3.5}$	$27.3_{2.0}$	$37.8_{2.4}$	$\mathbf{64.9}_{3.4}$	$44.4_{2.2}$	$\mathbf{52.7}_{2.4}$
✓	—	—	$3.2_{1.2}$	$6.6_{1.4}$	$4.3_{1.3}$	$0.2_{0.2}$	$0.2_{0.2}$	$0.2_{0.2}$	$1.7_{0.4}$	$2.6_{0.5}$	$2.1_{0.4}$
—	—	✓	$5.0_{1.4}$	$6.6_{1.4}$	$5.7_{1.3}$	$0.5_{0.5}$	$0.2_{0.2}$	$0.3_{0.3}$	$3.6_{0.8}$	$2.6_{0.5}$	$3.0_{0.6}$
✓	—	✓	$4.4_{1.3}$	$5.4_{1.4}$	$4.8_{1.3}$	$0.4_{0.5}$	$0.2_{0.2}$	$0.2_{0.3}$	$2.8_{0.8}$	$2.1_{0.5}$	$2.4_{0.6}$
—	✓	—	$57.7_{6.0}$	$\mathbf{73.4}_{2.8}$	$64.7_{4.4}$	$52.5_{4.1}$	$\mathbf{32.1}_{2.5}$	$\mathbf{39.8}_{2.2}$	$55.4_{4.0}$	$\mathbf{47.5}_{2.0}$	$51.1_{2.2}$
✓	✓	—	$62.4_{4.7}$	$72.8_{2.8}$	$67.2_{3.6}$	$53.4_{6.2}$	$29.1_{2.2}$	$37.7_{2.2}$	$58.5_{5.1}$	$45.4_{2.3}$	$51.1_{2.6}$
—	✓	✓	$66.6_{3.9}$	$72.5_{2.9}$	$69.4_{3.2}$	$\mathbf{61.5}_{4.8}$	$27.5_{1.9}$	$38.0_{2.3}$	$64.5_{3.2}$	$44.3_{2.2}$	$52.5_{2.3}$

Local evidence and contextual refinement. Every configuration without L had overall F1 at most 3.0% (Table 14), while L alone achieved 51.1%. Adding trajectory history and global context increased precision, but did not uniformly improve recall or branch-subset F1: L alone had the highest branch recall (32.1%) and F1 (39.8%), whereas the full decoder reached 27.3% and 37.8%, respectively. The full decoder attained the highest overall precision and F1, reflecting a different balance between recovering reference fibers and avoiding false positives.

Paired uncertainty. For each contrast, we drew the same 1,000 bootstrap samples of the 37 source images for both configurations, using random seed 42. Each draw retained all subtrees of each sampled image, with multiplicities; we recomputed pooled precision and took its difference between configurations. Table 15 reports the 2.5th and 97.5th percentiles of these paired differences. The joint precision gain from adding S and G to L was 9.5 percentage points, with a 95% interval of $[0.8, 18.2]$ percentage points. The intervals for adding S or G individually include zero, so these data do not establish statistically significant gains for either addition individually. These intervals quantify test-image sampling uncertainty for one trained checkpoint per configuration, rather than variability across training runs.

Table 15: **Paired precision gains from trajectory and global context.** Differences were computed before rounding; intervals use paired source-image bootstrap resampling.

Contrast	**ΔP (pp)**	**95% interval (pp)**
Add S to L	3.1	$[-3.8, 10.6]$
Add G to L	9.1	$[-1.0, 18.2]$
Add S and G to L	9.5	$[0.8, 18.2]$
Add S to L+G	0.4	$[-5.4, 6.5]$

F 3-D Positional Encoding Ablation

Setup. We ablated the positional representation by comparing multi-frequency Fourier encoding with a learned multilayer perceptron (MLP) applied directly to 3-D coordinates. In both variants, the positional embedding ϕ_{pos} was shared between trajectory tokens and image-grid features. The Fourier variant used 32 frequency bands. Both configurations used the full decoder with hidden dimension 192, $K = 192$ direction codewords, the same stored tracing step of approximately 1.978 voxels, and identical inference settings. We evaluated the same 483 CWMBS test subtrees from

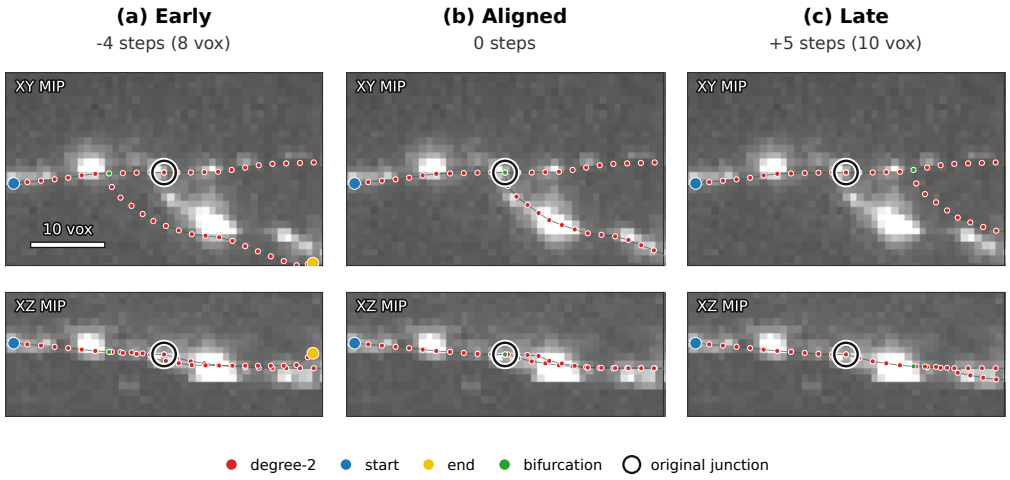

Figure 7: **Sensitivity to bifurcation-label location.** The same training image is shown in XY and XZ maximum-intensity projections with SWC nodes resampled at 2-voxel path intervals. Red, blue, yellow, and green nodes denote degree-2, start, end, and annotated bifurcation nodes, respectively; the leftmost visible node is shown as the blue start marker. The black ring marks the original SWC junction. Green bifurcation nodes use the same size and marker style as red degree-2 nodes. In (a) and (c), one daughter subtree is translated rigidly to an attachment four steps upstream or five steps downstream. The image and the other daughter remain fixed. These controlled perturbations illustrate how annotation noise can shift the apparent bifurcation location; they are not independently verified annotation errors.

37 source images using the protocol in Appendix C. The Fourier result is the TreeTracer (vanilla) checkpoint used in the main CWMBS comparison.

Table 16: **3-D positional encoding ablation on CWMBS.** Fiber micro-averaged P, R, and F1 (↑), grouped by reference topology. Subscripts denote standard deviations from 1,000 source-image bootstrap resamples. Bold scores mark column best before rounding; Fourier is the encoding used in TreeTracer.

Encoding	**No-branch** ($n = 335$)			**Branch** ($n = 148$)			**All** ($n = 483$)		
	P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)
MLP	$65.7_{3.1}$	$68.7_{3.0}$	$67.2_{2.9}$	$55.8_{3.3}$	$22.9_{1.8}$	$32.5_{2.1}$	$61.8_{2.6}$	$39.9_{2.1}$	$48.5_{2.2}$
Fourier	$\mathbf{67.3}_{4.3}$	$\mathbf{73.1}_{3.7}$	$\mathbf{70.1}_{3.8}$	$\mathbf{61.4}_{3.5}$	$\mathbf{27.3}_{2.0}$	$\mathbf{37.8}_{2.4}$	$\mathbf{64.9}_{3.4}$	$\mathbf{44.4}_{2.2}$	$\mathbf{52.7}_{2.4}$

Ablation results. Table 16 shows that Fourier encoding achieved higher precision, recall, and F1 in all three groups. Overall F1 rose from 48.5% with the MLP to 52.7% with Fourier features, an absolute gain of 4.2 percentage points. No-branch F1 increased from 67.2% to 70.1%, while branch-group F1 increased from 32.5% to 37.8%, with gains of 2.9 and 5.3 percentage points, respectively, computed before rounding. Overall precision improved from 61.8% to 64.9% and recall from 39.9% to 44.4%: Fourier encoding recovered 399 reference fibers compared with 359 for the MLP, while reducing false-positive fibers from 222 to 216. The reported standard deviations quantify test-image sampling variability for one checkpoint per encoding, rather than variation across independent training runs.

G LIMITATIONS

The relatively low recall of bifurcation detection is mainly attributable to annotation noise near junctions. In the current SWC-derived supervision, a bifurcation is represented as a single discrete node, although the annotated point may occur upstream of the first stable topological split or downstream after the two daughter branches have separated. As illustrated in Fig. 7, even a displacement of a few 2-voxel resampling steps changes whether the target lies before, at, or after the physical split. The model therefore has to learn both event detection and an uncertain event location, causing strict recall to underestimate its ability to recognize visible bifurcations. Future work will address this limitation at two levels: improving annotation quality through a reproducible branch-point definition, independent 3-D annotations, uncertainty intervals, and adjudication; and using uncertainty-aware training with soft or interval targets, separate event presence and location prediction, tolerance-aware losses, and tolerance-based evaluation.