THESIS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

Centerline Extraction for Tubular Trees in Medical Images

Methods for Structured Image Analysis and Localized Prediction

Roman Naeem

Department of Electrical Engineering
Chalmers University of Technology
Gothenburg, Sweden, 2026

Centerline Extraction for Tubular Trees in Medical Images

Methods for Structured Image Analysis and Localized Prediction

Roman Naeem

ISBN 978-91-8103-409-7

Acknowledgements, dedications, and similar personal statements in this thesis reflect the author's own views.

Doktorsavhandlingar vid Chalmers tekniska högskola
Ny serie nr 5866
ISSN 0346-718X

Department of Electrical Engineering
Chalmers University of Technology
SE-412 96 Gothenburg, Sweden
Phone: +46 (0)31 772 1000

Cover:
A London plane in the Botanical Garden of Visby, Gotland, photographed by the author.

Printed by Chalmers Digital Printing
Gothenburg, Sweden, May 2026

Dedicated to my family.

Centerline Extraction for Tubular Trees in Medical Images

Methods for Structured Image Analysis and Localized Prediction

Roman Naeem
Department of Electrical Engineering
Chalmers University of Technology

Abstract

Tubular tree structures such as blood vessels and airways are central to medical imaging from diagnosis to follow-up. A useful representation is the centerline graph, which captures both the medial course and the branching topology of the structure. Classical tracking and segmentation-based pipelines often produce graphs with topological errors such as disconnected components or cycles, along with missing or duplicate branches, reducing downstream usefulness. Recent learning-based image-to-graph methods address some of these issues but remain limited in topology preservation, 3D scalability, and clinical applicability.

This thesis develops recurrent Transformer-based image-to-graph methods that produce topologically valid centerline trees by construction. Trexplorer (Paper A) formulates centerline tracking as recurrent structured prediction, ensuring topological validity through sequential traversal. Trexplorer Super (Paper B) improves robustness through expanded trajectory training and focused higher-resolution features, with broader evaluation across synthetic and real CT data. RefTr (Paper C) advances this line with recurrent refinement of branch trajectories, duplicate suppression, and radius-aware evaluation.

Beyond extraction, centerline graphs also serve as effective inputs for localized clinical prediction. In follow-up after endovascular aneurysm repair, CEVAR (Paper D) uses point-wise centerline embeddings to predict protocol-driven measurements such as vessel diameters and seal lengths, replacing a post-hoc geometric step. The resulting automated pipeline outperforms a commercial semi-automatic workflow on a clinical cohort. Finally, ARTA (Paper E) is a mixed-resolution token allocation method that improves the trade-off between spatial detail and computational cost in dense feature extraction, with direct relevance to the sparse fine-structure analysis central to this thesis.

Keywords: Centerline extraction, tree topology, clinical translation, efficient feature extraction.

List of Publications

This thesis is based on the following publications:

[A] **Roman Naeem**, David Hagerman, Lennart Svensson, Fredrik Kahl, "Trexplorer: Recurrent DETR for Topologically Correct Tree Centerline Tracking". MICCAI 2024.

[B] **Roman Naeem**, David Hagerman, Jennifer Alvén, Lennart Svensson, Fredrik Kahl, "Trexplorer Super: Topologically Correct Centerline Tree Tracking of Tubular Objects in CT Volumes". MICCAI 2025.

[C] **Roman Naeem**, David Hagerman, Jennifer Alvén, Fredrik Kahl, "RefTr: Recurrent Refinement of Confluent Trajectories for 3D Tubular Tree Centerlines". Under Review.

[D] **Roman Naeem**, Timo Niiniskorpi, Naman Desai, Charlotte Sandström, Anders Jeppsson, Ida Häggström, Fredrik Kahl, Håkan Roos, Jennifer Alvén, "CEVAR: Centerline Embedding Extraction for Endovascular Aneurysm Repair". Under Review.

[E] David Hagerman*, **Roman Naeem***, Erik Brorsson, Fredrik Kahl, Lennart Svensson, "ARTA: Adaptive Mixed-Resolution Token Allocation for Efficient Dense Feature Extraction". Under Review.

Other publications by the author, not included in this thesis, are:

[F] D. Hagerman, **R. Naeem**, J. Lindqvist, C. Lindström, F. Kahl, and L. Svensson, "SwInception - Local Attention Meets Convolutions". *Proc. International Conference on Pattern Recognition and Artificial Intelligence*, pp. 3–17, Springer, 2024.

[G] D. Hagerman, A. Johnning, **R. Naeem**, F. Kahl, E. Kristiansson, and L. Svensson, "Optimizing gene-based testing for antibiotic resistance prediction". *Proc. AAAI Conference on Artificial Intelligence*, vol. 39, no. 27, pp. 28033–28041, 2025.

*Shared first authorship.

Contents

Acknowledgements

I would like to express my sincere gratitude to all those who have supported me during my PhD journey.

First and foremost, I would like to thank my supervisors, Fredrik Kahl, Jennifer Alvén, and Lennart Svensson, for their guidance, encouragement, and generous support throughout this work, and David Hagerman for the close collaboration throughout my PhD. I am grateful to all of them for the many discussions, ideas, and constructive feedback that have shaped this thesis.

I am also grateful to the past and current members of the computer vision group for making the 7th floor a pleasant place to work. Thanks to Josef, Victor, Sofie, Richard, both Davids, Vilgot, Tianyu, Bernardo, Jorge, Yara, Xixi, Kunal, Rasmus, Georg, Lucas, José, Ida, Ji, Huu, Erik, and Sophia for the many discussions, shared experiences, and the supportive and enjoyable atmosphere they helped create. I am grateful to be part of such a stimulating and friendly research environment.

A special thanks goes to my clinical collaborators, Håkan Roos, Charlotte Sandström, Timo Niiniskorpi, and Anders Jeppsson, whose data, annotations, and clinical perspective made the EVAR work possible and helped connect the methodological research to clinically relevant problems. I am especially grateful to Håkan for many discussions that deepened my understanding of the underlying medical problem. Thanks also to Naman Desai, whose master's thesis work contributed to the automated pipeline in Paper D.

Finally, I would like to thank my family and friends for their patience, encouragement, and unwavering support throughout these years. Their belief in me has meant more than I can express.

Gothenburg, 2026
Roman Naeem

This project was funded by MedTech West in a joint collaboration between Chalmers University of Technology and Sahlgrenska University Hospital.

Acronyms

AI:	Artificial Intelligence
AP:	Average Precision
AR:	Average Recall
ARTA:	Adaptive Mixed-Resolution Token Allocation
ATM:	Airway Tree Modeling Challenge 2022
CEVAR:	Centerline Embedding Extraction for EVAR
CNN:	Convolutional Neural Network
CPR:	Curved Planar Reconstruction
CT:	Computed Tomography
DETR:	DEtection TRansformer
EVAR:	Endovascular Aneurysm Repair
EVAR4C:	EVAR Four Components
F1:	F1 score
FFN:	Feed-Forward Network
FLOPs:	Floating Point Operations
MAE:	Mean Absolute Error
MLP:	Multi-Layer Perceptron
MRI:	Magnetic Resonance Imaging
NMS:	Non-Maximum Suppression
PARSE:	Pulmonary Artery Segmentation Challenge 2022
rAP:	Radius-aware Average Precision

rAR:	Radius-aware Average Recall
rF1:	Radius-aware F1 score
rBAP:	Radius-aware Branch Average Precision
rBAR:	Radius-aware Branch Average Recall
rBF1:	Radius-aware Branch F1 score
SOTA:	State of the Art
SVT:	Synthetic Vascular Toolkit
SwinUNETR:	Shifted Window U-Net Transformer
TNMS:	Tree Non-Maximum Suppression
ViT:	Vision Transformer
VMTK:	Vascular Modeling Toolkit

Part I

Overview

CHAPTER 1

Introduction

Tubular tree structures are abundant in human anatomy. Blood vessels, airways, biliary ducts, and related branching systems form networks that are essential for transport and function. Their morphology is closely linked to disease, and changes such as stenosis, aneurysmal dilation, occlusion, or branch loss can carry important diagnostic and prognostic information [1], [2], [3]. Accurate computational analysis of such structures is therefore an important problem in medical image analysis.

Computed tomography (CT) plays a central role in this setting. It provides high-resolution volumetric images and is widely used in vascular and thoracic imaging for diagnosis, treatment planning, intervention, and follow-up [4], [5], [6]. At the same time, manual analysis of tubular trees in CT remains demanding. These structures are long, thin, highly branched, and often weakly contrasted against surrounding tissue. Small distal branches may be difficult to identify, bifurcations may be ambiguous, and artifacts or pathology may interrupt apparent continuity [7], [8]. Recovering reliable structural descriptions from volumetric data is therefore challenging.

A particularly useful representation of a tubular tree is its centerline graph as illustrated in Figure 1.1. The centerline captures the medial course of the

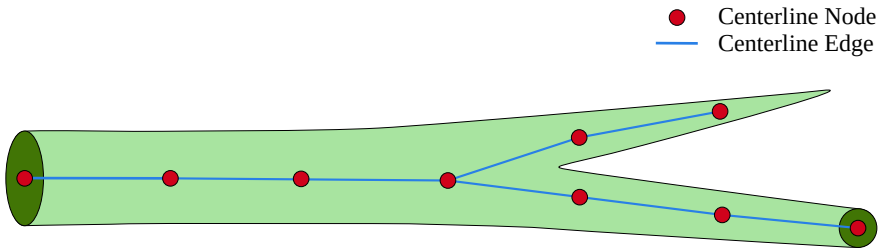


Figure 1.1: Illustration of a centerline graph for a segment of a tubular tree.

structure, while the graph encodes its branching organization. A tree here refers to a connected acyclic graph: there is exactly one path between any two nodes, and no loops. Each tree has a single root corresponding to the entry point of the structure (for example, the aortic root for an aorto-iliac tree, or the trachea for the airway tree). Leaf nodes (or endpoints) terminate the tree at distal branches, and bifurcation nodes are points where a branch splits into two or more child branches. The connected sequence of nodes between any two of these special points is called a branch.

This representation is attractive both methodologically and clinically. Methodologically, it turns a dense 3D image analysis problem into a sparse structured prediction problem. Clinically, it supports length measurements, orthogonal diameter estimation, anatomical labeling, navigation, and localized downstream analysis along anatomically meaningful paths. Centerline graphs are also compact, interpretable, and more naturally aligned with tree topology than dense masks.

Obtaining accurate centerline graphs from medical images is difficult. The challenge is not only to detect tubular structures, but also to recover their correct topology. In many downstream tasks, a geometrically plausible but topologically incorrect centerline is of limited value. Missing branches, disconnected components, duplicate branches, or spurious cycles can invalidate measurements and reduce trust in the output. Existing approaches include model-based tracking, segmentation followed by skeletonization, direct graph prediction, iterative tracking and iterative refinement. Each offers advantages, but also important limitations. Model-based tracking methods are interpretable and require no training data, but rely on handcrafted assumptions about tubu-

lar structures' appearance and geometry that often fail to generalize across datasets, scanners, and pathological variation. Segmentation-based approaches benefit from mature dense prediction architectures and large-scale training data, but rely on costly voxel-level labels and produce centerlines only indirectly through post-processing, where small segmentation errors can introduce broken or spurious branches. Direct graph prediction aligns more closely with the desired output, but does not necessarily guarantee a valid tree structure. Iterative tracking methods follow the branching process more naturally, but must handle error accumulation, bifurcations, and class imbalance. Iterative refinement methods can improve branch recall, but depend on the quality and coverage of the initial trajectory or branch hypotheses.

In parallel, medical image analysis has been transformed by deep learning. Convolutional neural networks (CNNs) enabled strong learned image representations for classification, segmentation, and detection. More recently, Transformer-based models have improved long-range interaction modeling and introduced query-based prediction paradigms that are well suited to structured outputs. These developments are particularly relevant to tubular tree extraction, where local information alone is often insufficient and global structural reasoning is important.

This thesis builds on that perspective. It studies how centerline extraction can be formulated as a structured image-to-graph problem, how valid topology can be preserved during prediction, how such methods should be evaluated, and how centerline representations can support downstream clinical analysis. It also considers efficient feature extraction for sparse fine structures, which is closely related to the challenges of tubular anatomy in 3D images.

More broadly, the thesis is motivated by the following question: how can structured, topologically correct, and computationally efficient representations of fine anatomical trees be extracted from medical images and used for localized analysis? The central argument is that this requires combining three perspectives: structured centerline graph prediction, explicit treatment of topology, and feature extraction methods that preserve fine detail while remaining efficient on large volumetric data.

1.1 Research Objectives

The overall objective of this thesis is to develop methods for extracting centerline graphs of tubular tree structures from medical images, with particular focus on CT volumes, and to investigate how such graph representations and their learned embeddings can support localized prediction and downstream image analysis.

More specifically, the thesis addresses the following research questions:

1. How can tubular tree centerlines be extracted from 3D medical images while preserving correct tree topology?

2. How can iterative and Transformer-based image-to-graph models improve the recall, precision, and robustness of centerline extraction?

3. How should such methods be evaluated across synthetic and real datasets, especially when topology and branch recovery are central concerns?

4. How can centerline graphs and point-wise centerline embeddings be used for localized clinical prediction tasks?

5. How can dense feature extraction be made more adaptive and efficient for tasks involving sparse, fine-scale anatomical structures?

1.2 Contributions

The thesis includes five papers that contribute to these questions. Trexplorer (Paper A) introduces a recurrent DETR-based [9] framework for topologically correct tree centerline tracking in 3D volumes. Trexplorer Super (Paper B) extends this framework with an improved model architecture and training strategy, and evaluation on both synthetic and real datasets. RefTr (Paper C) advances this line of work further through recurrent refinement of confluent trajectories, together with duplicate suppression and radius-aware evaluation. CEVAR (Paper D) demonstrates how learned centerline representations can support protocol-driven clinical analysis in EVAR follow-up CT. Finally, ARTA (Paper E) contributes a coarse-to-fine mixed-resolution Transformer architecture for efficient dense feature extraction, providing a methodological perspective that is highly relevant to sparse anatomical analysis.

Table 1.1: Mapping of research questions to the included papers.

Research Question	A	B	C	D	E
RQ1: Topology-preserving centerline extraction	•	•	•		
RQ2: Recurrent / Transformer image-to-graph models	•	•	•		
RQ3: Evaluation across synthetic and real datasets		•	•	•	
RQ4: Centerline-based localized clinical prediction				•	
RQ5: Adaptive and efficient feature extraction					•

Table 1.1 summarizes how each paper contributes to the research questions outlined in Section 1.1.

1.3 Thesis Outline

Part I provides the context, background, and synthesis of the research contributions. Chapter 1 introduces the problem setting, research objectives, and main contributions. Chapter 2 presents the methodological background, covering medical image analysis, deep learning, CNNs, Transformers, object detection, and structured prediction. Chapter 3 reviews prior work on centerline graph extraction, including model-based, segmentation-based, graph-based, and iterative approaches. Chapter 4 discusses the clinical translation of centerline extraction, with emphasis on workflow relevance and EVAR follow-up. Chapter 5 reviews image feature extraction methods relevant to fine-structure analysis, including CNN-based, Transformer-based, and adaptive coarse-to-fine approaches. Chapter 6 summarizes the included papers and clarifies the author's individual contributions. Chapter 7 concludes the thesis with a discussion of the main findings, limitations, and future work directions. Part II contains the five papers that form the core contributions of the thesis.

1.4 Notation

This thesis uses standard notation from graph theory, deep learning, and medical image analysis. A 3D medical image volume is denoted by I, defined on a discrete voxel grid. A centerline tree is represented as a graph $G = (V, E)$,

where V is the set of nodes and E is the set of edges. Each node $v \in V$ corresponds to a centerline point and may be parameterized by spatial position and radius, for example $v = [x, y, z, r]$. An edge $e \in E$ denotes connectivity between two nodes. A *branch* refers to a connected sequence of nodes between two special points, such as the root, a bifurcation, or an endpoint.

When relevant, a centerline tree may also be represented as a set of trajectories $T = \{T_i\}_{i=1}^{n}$, where each trajectory T_i is an ordered sequence of centerline points. In confluent settings, multiple trajectories may overlap before diverging into distinct branches.

Learned representations are referred to as features, tokens, queries, or embeddings, depending on context. In CNN-based models, intermediate representations are denoted as feature maps. In Transformer-based models, token embeddings denote vector representations associated with image patches, trajectories, or object queries. More generally, a learned feature vector is denoted by $f \in \mathbb{R}^d$, where d is the feature dimension.

Throughout this thesis, tubular tree structure (or simply tubular tree) refers to the general class of branching anatomical networks that includes blood vessels and airways. Vessel is used specifically for blood vessels (including the synthetic vessel datasets, the pulmonary arteries in PARSE 2022, and the aortic data used in the EVAR follow-up work), and vascular is its adjectival form. Airway refers specifically to the pulmonary airway tree, as in the ATM'22 dataset. A branch denotes a structural element of a tree, namely the connected sequence of nodes between two special points (root, bifurcation, or endpoint), rather than a tubular structure as a whole.

Additional notation specific to individual chapters or papers is introduced locally where needed.

1.5 Chapter Summary

This chapter has introduced the motivation for the thesis and positioned it at the intersection of centerline graph extraction, topology-preserving structured prediction, localized analysis, and efficient feature extraction. The next chapter presents the methodological background for the included work.

CHAPTER 2

Background

Medical image analysis seeks to extract clinically meaningful information from medical images for diagnosis, treatment planning, intervention, and follow-up. Over time, the field has evolved from handcrafted image processing pipelines to data-driven learning systems [4]. This shift has been especially important for volumetric modalities such as computed tomography (CT), where the size and complexity of the data make manual analysis difficult.

The problems studied in this thesis lie at the intersection of several areas in medical image analysis. Centerline graph extraction from tubular trees requires methods that can reason about continuity, branching topology, and sparse elongated anatomy in 3D images, while feature extraction must preserve fine detail and broader anatomical context. This chapter summarizes the methodological background needed for the remainder of the thesis.

2.1 Medical Image Analysis

Medical image analysis concerns computational interpretation of images acquired from modalities such as CT, magnetic resonance imaging (MRI), ultrasound, positron emission tomography, and X-ray. Depending on the

application, this may involve segmentation, detection, registration, classification, reconstruction, or extraction of compact structural representations such as landmarks, graphs, or centerlines.

CT is the main modality considered in this thesis. It provides high-resolution volumetric data and is widely used in vascular and thoracic imaging. CT is well suited for visualizing extended tubular anatomy such as arteries and airways, but it also presents important challenges. Volumes are large and memory-intensive, relevant structures may occupy only a small fraction of the image, contrast varies across patients and protocols, and thin distal branches may be difficult to distinguish from surrounding tissue. These factors place strong demands on the models used.

A recurring theme in medical image analysis is the design of representations that balance anatomical faithfulness, computational tractability, and clinical usefulness. Dense voxel-wise representations preserve rich spatial information but can be costly to annotate and process. Sparse structured representations, such as centerline graphs, are more compact and often align more directly with downstream measurement tasks. Figure 2.1 shows an example from the PARSE 2022 dataset [10], together with the corresponding segmentation mask and centerline graph of the pulmonary artery tree. Refer to Section 3.6 for more details about the datasets used in this work.

2.2 Classical and Deep Learning Approaches

Before deep learning, medical image analysis relied primarily on handcrafted methods based on image processing, geometry, optimization, and statistical modeling. Examples include thresholding, region growing, deformable models, atlas-based segmentation, vesselness filtering, minimal path methods, and graph-based optimization. For tubular structures, such methods [11], [12], [13] typically enhance elongated image patterns and then recover likely centerlines by tracing or optimization. Their main strength is interpretability, since they encode explicit assumptions about image appearance and geometry. However, they are often sensitive to parameter choices and struggle to generalize across datasets, scanners, and pathological variation. In the remainder of this thesis, classical centerline extraction pipelines based on handcrafted image operators and explicit optimization criteria are referred to as model-based methods.

Deep learning transformed medical image analysis by replacing handcrafted

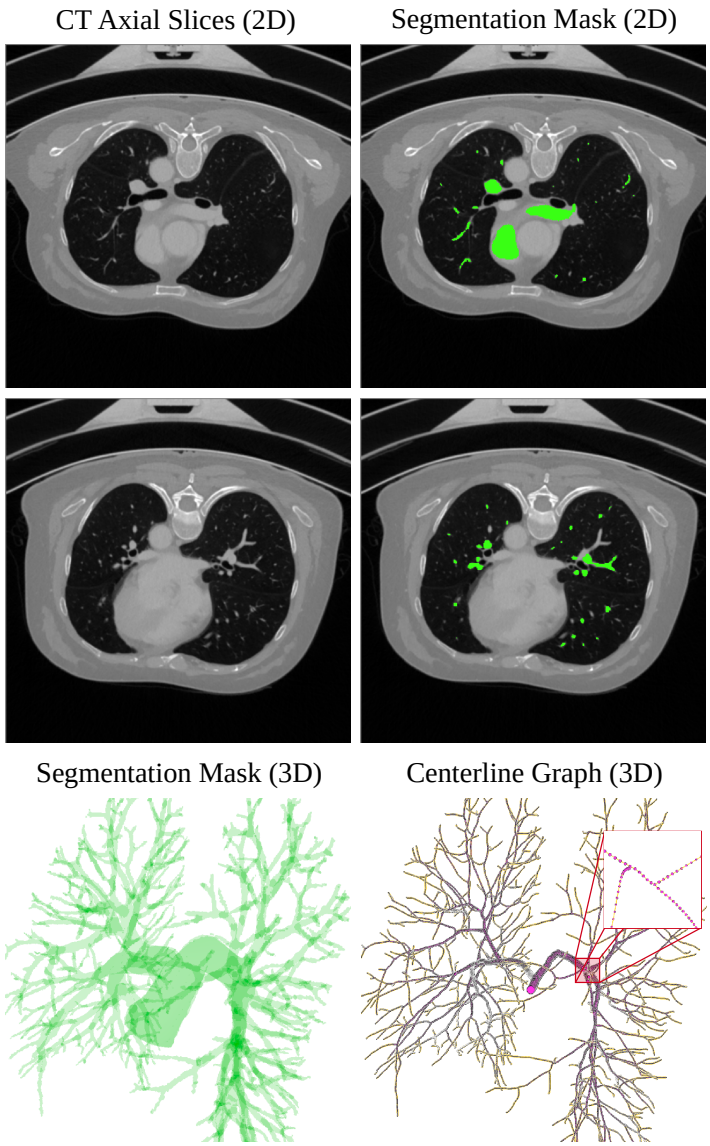


Figure 2.1: Axial slices of a CT volume from the PARSE 2022 dataset [10], together with the corresponding binary segmentation map and centerline graph of the pulmonary artery tree.

feature design with data-driven representation learning. Neural networks learn hierarchical features directly from annotated data and have improved performance on tasks such as segmentation, detection, classification, and registration [14], [15]. In medical imaging, this has been especially valuable because anatomy and pathology can vary substantially in size, shape, and appearance. At the same time, deep learning introduces challenges including limited annotated data, class imbalance, domain shift across scanners and protocols, and the computational cost of 3D imaging.

Centerline extraction often requires structured prediction, meaning inferring paths or graphs rather than only voxel-wise labels. Classical methods achieved this through handcrafted feature extraction coupled with minimal paths or graph optimization. Deep learning enables learned image-to-structure models that predict such structured outputs directly from images, reducing reliance on fixed pipelines. For the problems studied in this thesis, deep learning is therefore important not only for improved image representations, but also for enabling end-to-end structured prediction beyond voxel-wise labeling.

2.3 Convolutional Neural Networks

Convolutional neural networks (CNNs) [16] were the first deep models to dominate image analysis. By learning local filters that slide across the image, CNNs build hierarchical representations: shallow layers capture low-level appearance such as edges and textures, while deeper layers capture more abstract semantics. Figure 2.2 illustrates the basic convolution operation, in which a small filter is applied at each position of the input to produce a corresponding pixel in the output feature map.

In medical imaging, CNNs became standard for dense tasks such as segmentation and detection. Encoder-decoder models such as U-Net [14] were especially influential because they combine high-level semantic features with fine spatial detail through skip connections. CNNs remain attractive for tubular structure analysis because they are efficient, well-established, and effective at capturing local appearance and multiscale structure.

Their main limitation in this context is that long-range dependencies are modeled only indirectly: receptive fields grow gradually through stacked layers, so capturing context across an entire elongated structure requires deep networks. For tubular trees, this can make continuation and connectivity difficult to

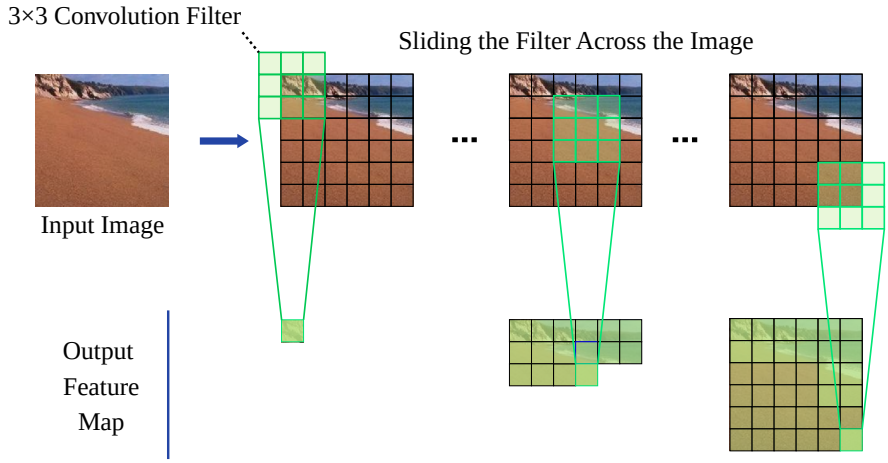


Figure 2.2: Illustration of a 2D convolution operation. A 3×3 filter slides across the input image, and at each position computes a weighted sum of local pixel values to produce one pixel in the output feature map. The feature map is built up progressively as the filter visits every valid position. By learning many such filters and stacking convolutional layers, CNNs build hierarchical image representations.

resolve over larger spatial ranges, which has motivated interest in architectures that model long-range interactions more directly.

2.4 Transformers and Structured Prediction

Transformers [17] introduced a different approach to representation learning using self-attention, allowing tokens to interact directly across long spatial ranges. In vision, this has led to models that can integrate global context more flexibly than CNNs. In medical imaging, Transformers have become increasingly important for both 2D and 3D tasks [18], especially when long-range anatomical dependencies matter.

Tokens and Self-Attention

A Transformer does not operate on raw text or raw pixels directly. Instead, the input is first split into smaller pieces called tokens, and each token is

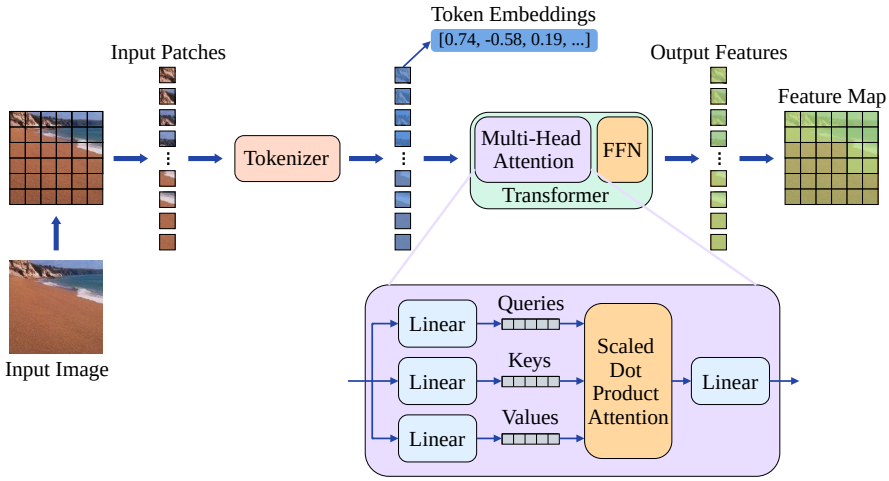


Figure 2.3: Overview of a Transformer for vision. An input image is partitioned into patches and converted by a tokenizer into a sequence of token embeddings. The Transformer block processes these tokens using multi-head attention and a feed-forward network (FFN), producing a sequence of output features that can be reshaped into a feature map. The bottom panel shows the internal structure of a single attention head: input tokens are linearly projected into queries, keys, and values, which are combined through scaled dot-product attention and projected to the output.

converted into a vector of numbers (an embedding) that the model can process. In language models, a token is typically a word or subword. In vision Transformers [19], an image is partitioned into a grid of non-overlapping patches (for example 6×6 pixels), and each patch is flattened and projected linearly into a d-dimensional embedding, as illustrated in Figure 2.3. A learned position embedding is added so that the model can distinguish tokens by their location in the original input. The result is a sequence of N token embeddings $\{x_1, \ldots, x_N\}$, $x_i \in \mathbb{R}^d$, that serves as input to the Transformer.

The core operation is self-attention, which allows each token to aggregate information from every other token. For each token, three vectors are computed by linear projection: a query q_i, a key k_i, and a value v_i (Figure 2.3, bottom). The attention output for token i is a weighted sum of all value vectors, where

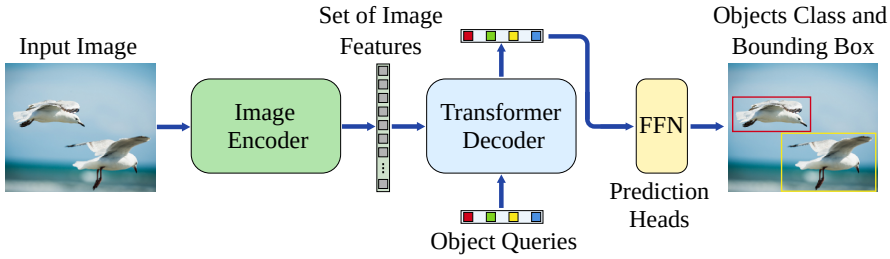


Figure 2.4: Overview of the DETR pipeline. A set of object queries attends to image features and is then used by the prediction heads to predict object classes and bounding boxes.

the weights are determined by the similarity between q_i and each k_j:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V, \tag{2.1}$$

where Q, K, and V stack the queries, keys, and values for all tokens, and d_k is the key dimension. Intuitively, each token asks (via its query) which other tokens are relevant to it, matches this against the keys of all tokens, and then aggregates their values accordingly. Because every token can attend to every other token in a single step, long-range dependencies are modeled directly rather than through repeated local operations as in CNNs.

In practice, self-attention is applied in parallel through multiple attention heads, each with its own projections, so that the model can attend to different relational patterns simultaneously. A Transformer block stacks multi-head self-attention with a feed-forward network and residual connections, and the full model stacks many such blocks. Their main strength is flexibility in modeling token interactions; their main drawback is computational cost, which grows quadratically with the number of tokens. This is especially limiting in 3D volumes and has motivated hierarchical, sparse, and windowed variants [20].

Set Prediction with Object Queries

Beyond encoding image features, Transformers fit naturally with structured prediction settings through the use of object queries. An object query is a learned embedding that, unlike image tokens, does not correspond to a

specific image region. Instead, each query attends to image features through cross-attention and produces one element of the output set, such as a detected object, a node, a branch, or a trajectory.

DETR [9] reformulated object detection along these lines as direct set prediction, as illustrated in Figure 2.4. A fixed number of learned object queries are processed by a Transformer decoder that alternates self-attention among queries (so that they coordinate) with cross-attention to image features (so that they ground in the image). Each query then produces a class label and a bounding box through a small prediction head. Because the model outputs a set rather than an ordered sequence, predictions are matched to ground-truth objects using bipartite matching computed with the Hungarian algorithm [21], and a set-based loss is applied to the matched pairs. Predictions not matched to any ground-truth object are supervised to predict a "no-object" class.

DETR's importance extends beyond detection itself. It provides a general framework for predicting sparse structured outputs as sets of entities, which makes it highly relevant to image-to-graph problems.

2.5 Graph Prediction from Images

Many real-world structures are more naturally represented as graphs than as dense masks. Roads, lane networks, river systems, vascular trees, and airway trees are all examples. This has motivated growing interest in image-to-graph prediction, where the goal is to infer sparse graph structure, including nodes, edges, and their attributes, directly from images, rather than recovering it indirectly from dense voxel- or pixel-wise predictions.

Graph prediction differs from dense prediction in two important ways. First, the model must infer both the locations of relevant entities and their connectivity, rather than only assigning labels to a fixed grid. Second, it often requires explicit structural validity constraints, since the predicted graph should satisfy properties of the target anatomy or scene, such as connectivity, acyclicity, or specific branching patterns. Standard dense prediction losses do not enforce such constraints, which is one reason image-to-graph prediction has emerged as a distinct line of work. Closely related problems appear in remote sensing and autonomous driving, where vectorized map prediction and road graph extraction [22], [23] share important similarities with centerline extraction.

The query-based view introduced in the previous section connects naturally to this setting, and especially to sequential prediction: queries can act as persistent latent states that are carried across steps and updated with new image evidence, as in multi-object tracking. Centerline extraction fits this pattern, since a tubular tree is typically recovered progressively by following branches, handling bifurcations, and predicting continuation or termination. This perspective underlies the recurrent centerline extraction models developed in Papers A, B, and C, where learned queries represent centerline nodes (Trexplorer, Trexplorer Super) or branch trajectories (RefTr) that are matched to ground-truth structures via bipartite assignment.

2.6 Chapter Summary

This chapter outlined the methodological background of the thesis. CNNs and Transformers provide complementary feature extraction paradigms, with Transformers and DETR-style set prediction motivating the structured image-to-graph modeling used throughout the thesis. The next chapter reviews prior work on centerline graph extraction.

CHAPTER 3

Centerline Graph Extraction

Centerline graph extraction from tubular tree structures is a central problem in medical image analysis. In vascular and pulmonary imaging, centerlines provide compact representations that support measurement, navigation, morphometric analysis, and downstream prediction [24], [25]. A centerline graph captures both the geometric course of a structure and its branching organization, making it a natural representation for vessels, airways, and related tubular tree-shaped anatomy.

Extracting such graphs from medical images remains challenging. Tubular structures are thin, elongated, and often weakly contrasted against surrounding tissue. Small distal branches are easily missed, bifurcations may be difficult to localize, and artifacts or pathology may interrupt apparent continuity. The problem is therefore not only to detect where a structure exists, but also to recover a connected and anatomically plausible graph.

A central difficulty is topology preservation. In many applications, geometric plausibility alone is not enough: disconnected components, spurious cycles, missing branches, and duplicate branches all reduce downstream usefulness [26], [27]. This motivates treating centerline extraction as a structured prediction problem.

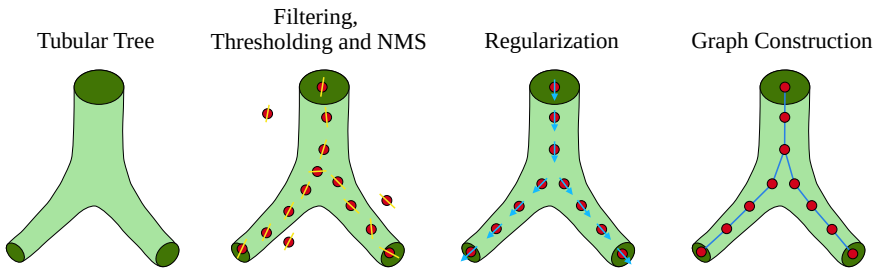


Figure 3.1: Example of a classical model-based centerline extraction pipeline. Vesselness filtering, thresholding, and non-maximum suppression (NMS) yield candidate centerline points along the medial axis with local orientation estimates (yellow). Regularization refines positions and orientations (cyan) and removes spurious candidates. A graph-construction step (e.g., minimum spanning tree) connects the cleaned candidates into a tree-structured centerline graph.

In this thesis, the output of centerline extraction is treated as a graph representation of a tubular tree. A centerline graph can be written as $G = (V, E)$, where nodes V correspond to sampled centerline points in 3D space and edges E represent their connectivity. Depending on the method, nodes may also carry local attributes such as an estimated radius, and the graph may be represented explicitly through nodes and edges or implicitly through ordered branch trajectories that can later be converted into a graph. These representation choices affect both how topology constraints are imposed and how extraction quality is evaluated.

The literature can be grouped into five broad families: model-based methods, segmentation-based methods followed by skeletonization, direct image-to-graph methods, iterative tracking methods, and iterative refinement methods. This chapter reviews these families, summarizes the datasets and evaluation metrics used to assess centerline extraction quality, and positions the contributions of this thesis within that development.

3.1 Model-based Methods

Early centerline extraction methods were largely model-based, often formulated as optimization problems. These approaches encoded assumptions about

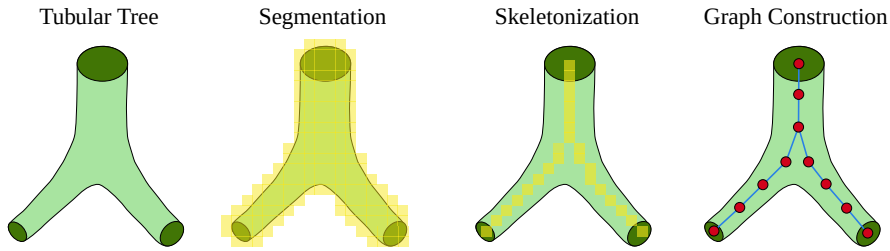


Figure 3.2: Segmentation-based centerline extraction with skeletonization: a tubular structure is first segmented to obtain a binary mask, which is then skeletonized (thinned) to a 1-voxel-wide centerline and converted into a graph by identifying endpoints/bifurcations and connecting skeletal paths.

tubular appearance and geometry directly into the algorithm. A common strategy was to enhance likely vessels or airways using handcrafted image operators and then recover the centerline by tracing or optimization. Vesselness filtering [11], template-based tracking [12], and energy-based regularization with geometric priors [28] were especially influential. An example of a vesselness-based pipeline, following the formulation in [28], is shown in Figure 3.1.

The main strength of such methods was interpretability, since anatomical priors and optimization criteria were made explicit. However, they were often sensitive to image quality, parameter choices, and variation across datasets or pathologies. Their limitations motivated the shift toward learning-based methods.

3.2 Segmentation-based Methods with Skeletonization

A common learning-based strategy [29], [30] is to first predict a dense segmentation of the tubular structure and then derive a centerline through skeletonization, thinning, medial-axis extraction, or graph post-processing. This is attractive because segmentation is a mature problem with strong architectures and straightforward voxel-wise supervision.

The main limitation is that the final target is still a graph, not the segmentation itself. Skeletonization is sensitive to discontinuities and local errors, and small segmentation mistakes can lead to broken or spurious branches.

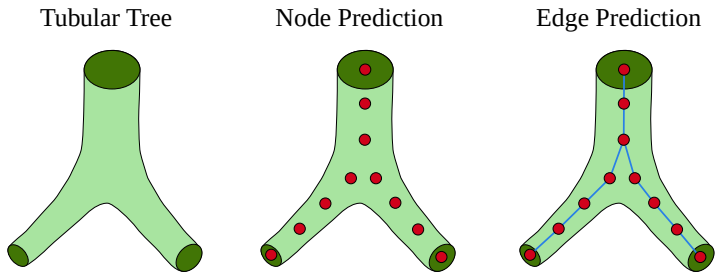


Figure 3.3: Direct image-to-graph model pipeline: a tubular tree segment (left), predicted centerline nodes (middle), edge prediction (right).

As a result, these approaches often require substantial post-processing, and voxel-wise training objectives do not necessarily align with graph-level correctness. An example of such a pipeline is shown in Figure 3.2.

3.3 Direct Image-to-Graph Methods

A more direct alternative [31], [32] is to formulate centerline extraction as an image-to-graph problem, where the model predicts graph elements such as nodes, edges, or branch segments directly from image features (Figure 3.3). This mitigates common segmentation-to-skeleton failure modes in which small voxel-level errors are amplified by thinning. For example, small gaps in a segmentation mask can yield disconnected skeletons, while boundary noise or partial-volume artifacts can introduce spurious branches that require heuristic pruning. By predicting sparse graph structure directly, image-to-graph methods reduce dependence on brittle skeletonization and post-processing, and better align learning with the desired output representation.

However, direct graph prediction introduces its own challenges. The model must infer both geometry and connectivity under variable graph size and branching complexity, and neither connectivity nor a valid tree structure is guaranteed automatically. In practice, predictions may contain disconnected components or spurious cycles, motivating explicit structural constraints, such as enforcing connectivity and acyclicity by restricting outputs to tree graphs.

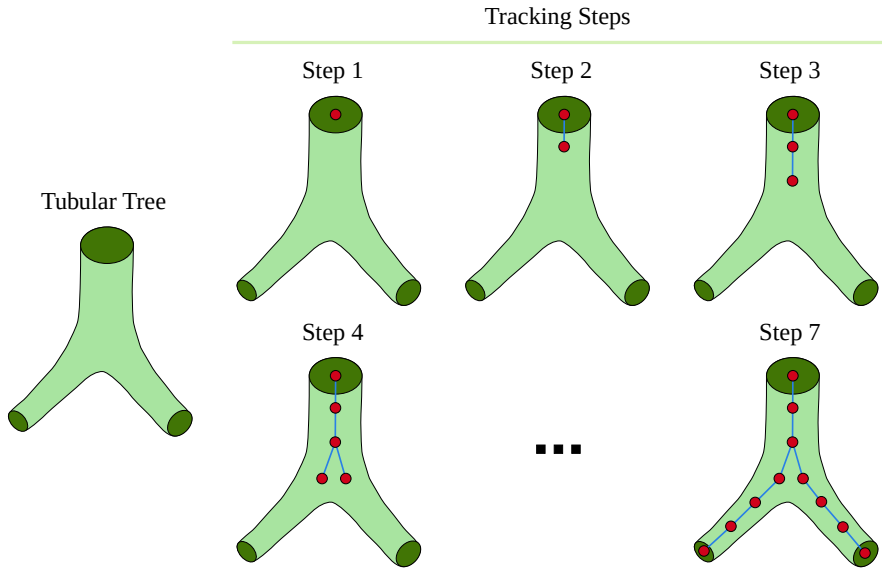


Figure 3.4: Illustration of iterative centerline tracking. Starting from a root point, the predicted centerline graph is extended over multiple tracking steps by predicting continuation and bifurcations until the full tree is recovered.

3.4 Iterative Tracking Methods

Another family of methods approaches centerline extraction as an iterative traversal process. Starting from a root or seed point, the model progressively follows the tree, predicting continuation, branching, and termination as it moves through the image. This formulation is well aligned with the branching structure of tubular trees.

Earlier iterative methods used CNN-based orientation classifiers [33] or reinforcement learning [34], [35], [36]. More recent approaches have adopted recurrent and Transformer-based formulations. In this thesis, Trexplorer (Paper A) and Trexplorer Super (Paper B) belong to this family. They frame centerline extraction as structured recurrent prediction and explicitly emphasize topology preservation and branch tracking. An example of this process is shown in Figure 3.4.

Many iterative tracking approaches still struggle with missed branches and

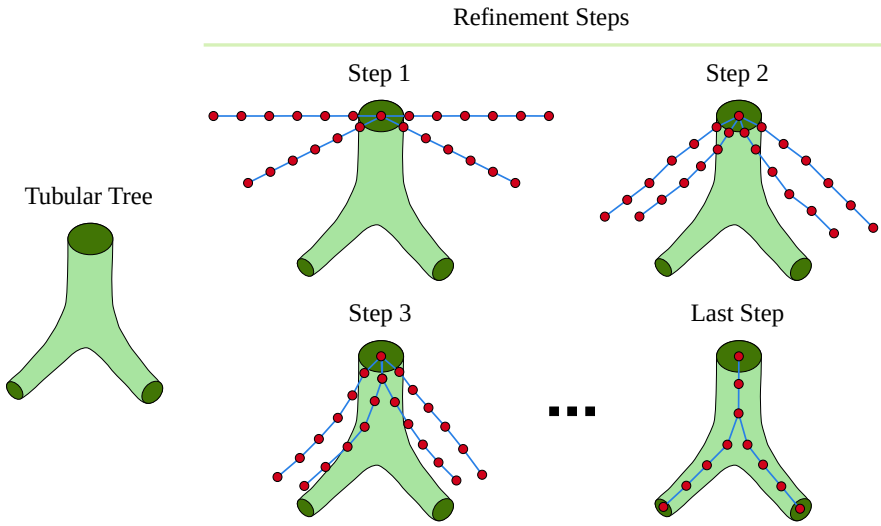


Figure 3.5: Conceptual illustration of iterative refinement. Starting from an overcomplete set of candidate trajectories, the model refines them toward anatomically plausible branches. Duplicates are suppressed, and divergence between refined trajectories defines bifurcations in the resulting centerline graph.

premature termination. One contributing factor is extreme class imbalance, since bifurcation and endpoint decisions are rare compared to continuation steps. In addition, strictly sequential designs can make error recovery difficult, because early mistakes may propagate to later tracking steps.

3.5 Iterative Refinement Methods

Recent work moves beyond point-wise tracking toward larger structured units, such as branch trajectories. Rather than predicting only the next local step, these approaches generate an overcomplete set of trajectory hypotheses and progressively refine them into anatomically plausible branches. In this thesis, RefTr (Paper C) follows this direction by introducing a refinement-based image-to-graph framework built on structured branch-level trajectory representations, with mechanisms for suppressing duplicate trajectories and recovering consistent bifurcations. A conceptual overview is shown in Figure 3.5.

While iterative refinement has been explored only sparingly for centerline tree extraction, similar paradigms are common for other structured outputs in medical image analysis, particularly surface and mesh reconstruction. Voxel2Mesh [37] starts from an initial mesh and iteratively deforms it to better align with image evidence, while TopoFit [38] enables rapid reconstruction of topologically correct cortical surfaces. Although centerline extraction outputs a sparse tree graph rather than a dense surface, these works motivate refinement-based formulations in which structured hypotheses are progressively corrected while preserving structural validity.

3.6 Datasets and Ground Truth Generation

The dataset source and annotation procedure shape what conclusions can be drawn from centerline extraction evaluation. Synthetic datasets provide clean and fully specified ground truth but may omit important anatomical and imaging complexity, while real CT datasets better reflect clinical variability yet often rely on centerlines derived from segmentation masks. Clinically curated centerlines are typically the most realistic, but are costly to obtain and therefore limited in scale. For this reason, the papers in this thesis evaluate models across a progression from public synthetic vessel trees (Paper A), to synthetic and public real CT datasets with derived ground truth (Papers B and C), and finally to a retrospective EVAR cohort with manually adjusted centerlines for protocol-driven downstream assessment (Paper D).

For controlled development, Paper A evaluates Trexplorer on the synthetic vessel centerline dataset introduced with DeepVesselNet [29], using established splits and evaluation protocols from prior work [32]. To obtain more realistic synthetic trees, Paper B generates a new synthetic dataset using the Synthetic Vascular Toolkit (SVT) [39], which incorporates collision avoidance and reduces unrealistic self-intersections between vessel branches while still providing procedural ground truth. An example is shown in Figure 3.6.

To evaluate on real CT, Paper B derives centerline graphs from two public tubular tree segmentation datasets: ATM'22 (airways) [40] and PARSE 2022 (pulmonary arteries) [10]. Volumes are resampled to isotropic resolution, root points are estimated using the Vascular Modeling Toolkit (VMTK) [41], [42], and centerlines are traced from the segmentation masks using Kimimaro [43]. These datasets introduce greater anatomical variability and imaging artifacts

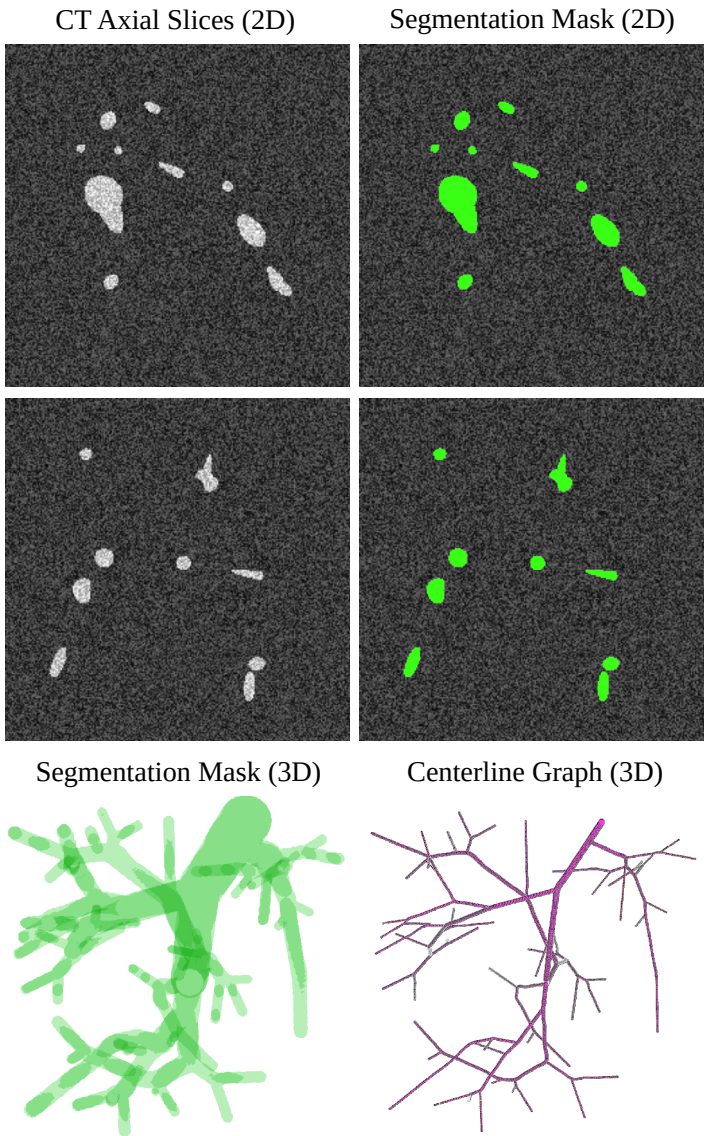


Figure 3.6: Example from the new synthetic dataset (Paper B): axial slices with segmentation masks (top), 3D mask (bottom left), and centerline graph (bottom right). Node size in the graph is proportional to the local radius.

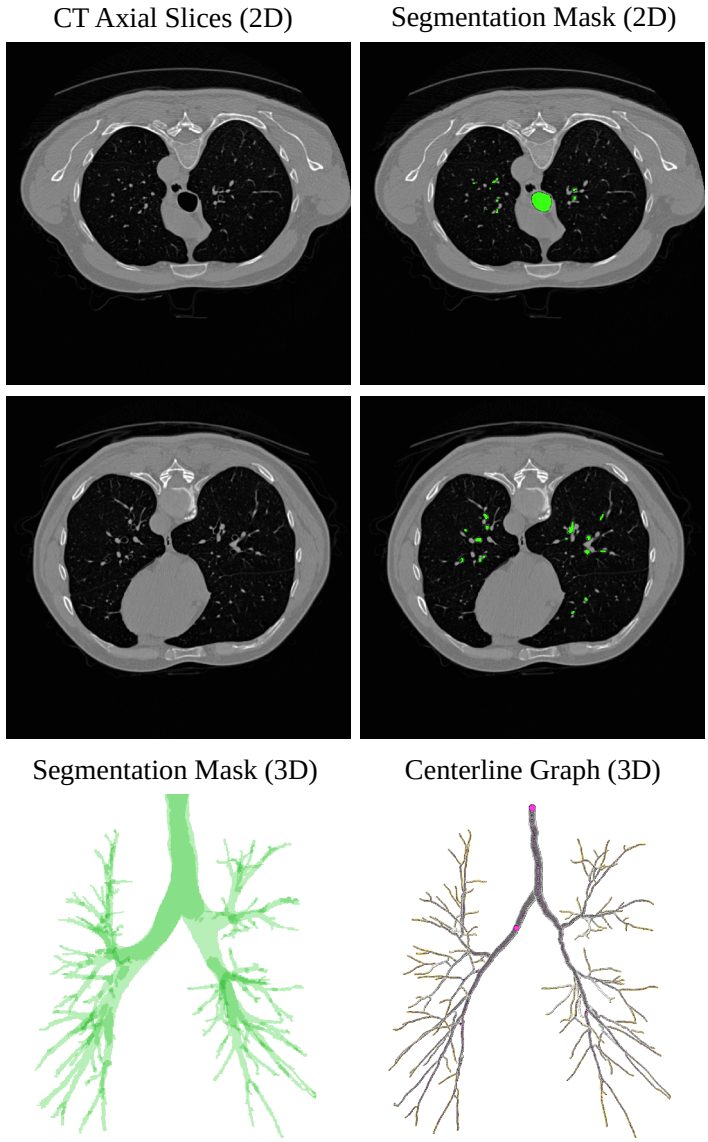


Figure 3.7: Example from the ATM'22 dataset (Paper B): axial slices with segmentation masks (top), 3D mask (bottom left), and centerline graph (bottom right). Node size in the graph is proportional to the local radius.

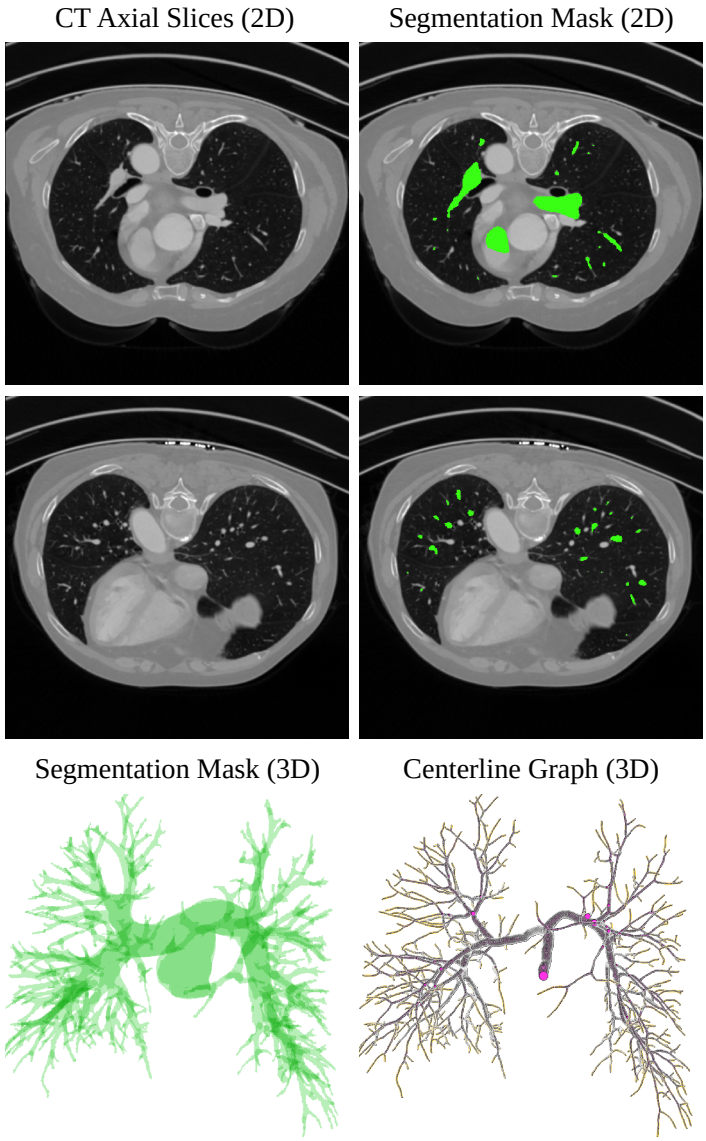


Figure 3.8: Example from the PARSE 2022 dataset (Paper B): axial slices with segmentation masks (top), 3D mask (bottom left), and centerline graph (bottom right). Node size in the graph is proportional to the local radius.

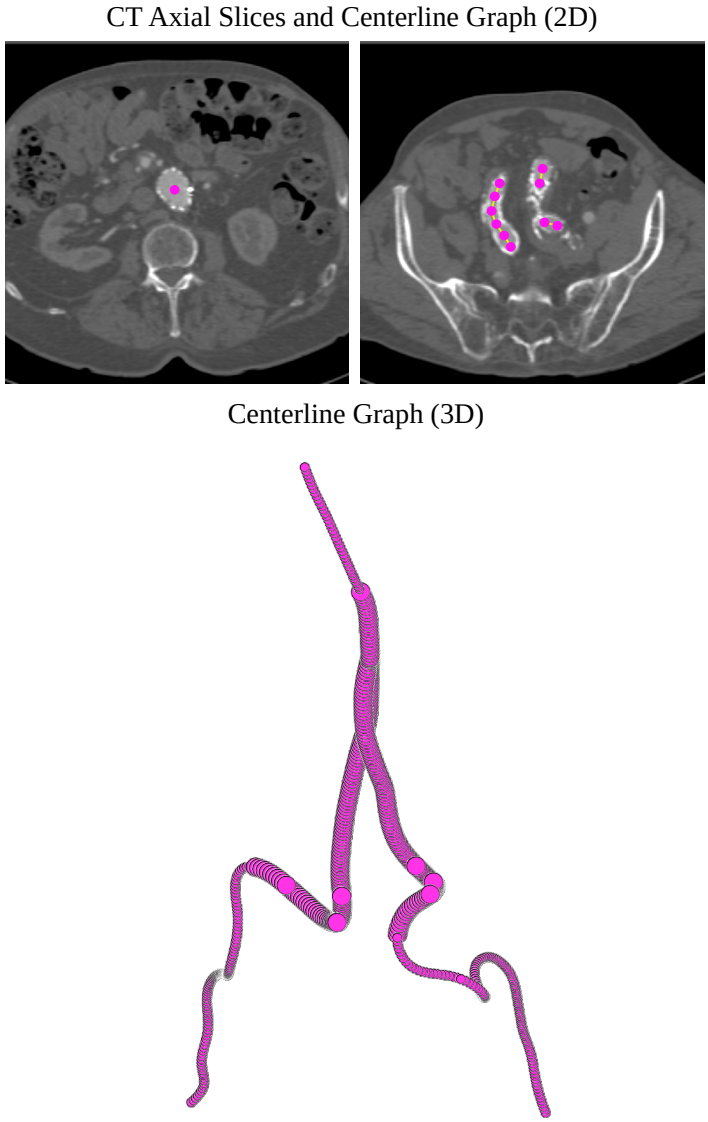


Figure 3.9: An example from the clinically curated dataset (Paper D): axial slices with centerline points (top), and corresponding 3D centerline graph (bottom). The large nodes in the centerline graph represent the points in the stent region.

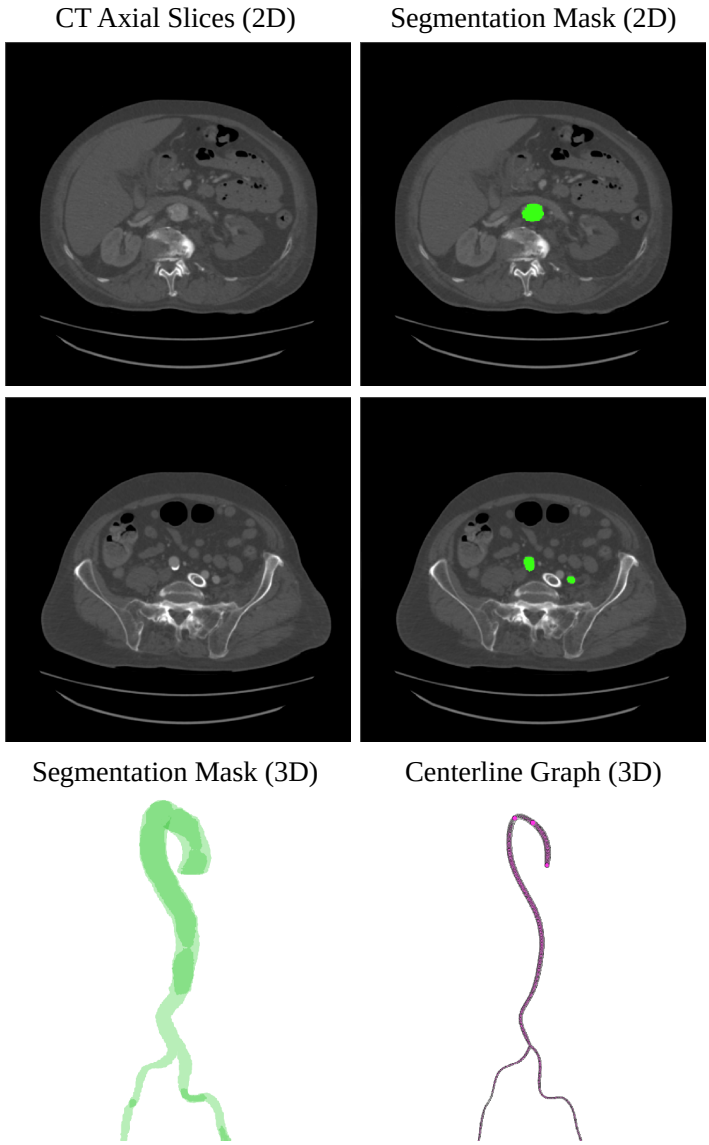


Figure 3.10: Example from the Aorta24 dataset (Paper D): axial slices with segmentation masks (top), 3D mask (bottom left), and centerline graph (bottom right). Node size in the graph is proportional to the local radius.

than synthetic data, making them stricter tests of robustness and topology preservation. Examples are shown in Figures 3.7 and 3.8. Segmentation maps are shown in these figures for visualization only; the centerline models use only the compact centerline graph annotations for training and evaluation.

Deriving centerlines from segmentations enables scalable evaluation on public CT data, but it also introduces dependence on segmentation quality and on the extraction pipeline. For this reason, the thesis complements derived ground truth with clinically curated annotations.

Paper D uses a retrospective EVAR follow-up cohort from two hospitals, including both contrast and non-contrast CT, where centerlines are generated and then manually adjusted by a physician and measurements are recorded according to the EVAR4C protocol. Unlike the other datasets, only centerline graph annotations are available; no segmentation masks are provided. Examples are shown in Figure 3.9. To support pretraining on aortic anatomy before fine-tuning on the clinical cohort, Paper D also derives centerlines from the public Aorta24 [44] dataset by extracting centerlines from aorta segmentation masks using VMTK. An example is shown in Figure 3.10.

Overall, the datasets used in this thesis span a progression from synthetic benchmarks, to public real CT with derived centerlines, to clinically curated centerlines for protocol-driven assessment, supporting both methodological evaluation and downstream clinical relevance. Further dataset details are provided in the corresponding papers in Part II.

3.7 Evaluation Metrics

Evaluating centerline graph extraction is nontrivial because accuracy must be assessed beyond point-wise geometry. Earlier benchmarks for coronary centerline extraction [45] introduced standardized point-level overlap and accuracy measures, but did not address tree-level topology directly. A predicted set of points may be geometrically close to the ground truth, yet form a graph that is disconnected, incomplete, or contains duplicate branches. Such errors can invalidate downstream measurements even when point-level distances are small. The papers in this thesis therefore evaluate predictions at three complementary levels: point, branch, and graph. This section summarizes the metrics used; further details are given in the corresponding papers in Part II.

A common matching principle underlies the point- and branch-level metrics.

Predicted elements (nodes or branches) are matched to ground-truth elements one-to-one, and unmatched predictions and ground-truth elements are counted as false positives and false negatives, respectively. Average precision (AP), average recall (AR), and F1-score follow the standard definitions over the matched pairs. To reduce sensitivity to a single matching threshold, metrics are averaged over a range of thresholds.

Point-Level Metrics

At the point level, predicted centerline nodes are matched to ground-truth nodes based on spatial proximity. A predicted node is counted as a true positive if it lies within a distance threshold of a previously unmatched target node; otherwise, it is a false positive. Unmatched ground-truth nodes are false negatives.

A fixed distance threshold treats branches with large and small radii equally, even though acceptable localization error scales with branch radius. To address this, RefTr (Paper C) introduces *radius-aware* matching, in which the threshold scales with the ground-truth branch radius: a predicted node is accepted as a true positive if it lies within $\max(1.5, \tau^{\text{rad}} \cdot r)$ of an unmatched target node, where r is the ground-truth radius and τ^{rad} is a threshold parameter. Reported metrics (rAP, rAR, rF1) are averaged over $\tau^{\text{rad}} = 0.25{:}0.05{:}0.75$ to reduce sensitivity to the choice of τ^{rad}. In addition to spatial accuracy, the mean absolute error (MAE) of the predicted radius is reported.

Point-level metrics capture local geometric accuracy but say little about whether branches are recovered as complete units or whether the overall structure is connected.

Branch-Level Metrics

Branch-level metrics treat individual branches as objects and assess whether each ground-truth branch is recovered as a coherent whole. A predicted branch is counted as a true positive if it overlaps with at least a fraction τ^{match} of nodes from a previously unmatched ground-truth branch, where overlap is defined using the same radius-aware proximity rule as at the point level. Unmatched predicted and ground-truth branches are false positives and false negatives, respectively.

The corresponding metrics (rBAP, rBAR, rBF1) are averaged over $\tau^{\text{match}} =$

0.4:0.05:0.8 with τ^{rad} fixed at 0.5. Branch-level metrics are more sensitive than point-level metrics to two failure modes: missed distal branches (branches that are not recovered at all) and duplicate branches (multiple predictions covering the same target branch).

Graph-Level Metrics

Graph-level metrics assess overall topological correctness. The thesis uses the mean absolute error of the Betti numbers β_0 and β_1, which count connected components and independent cycles, respectively. For a centerline tree, the ground-truth values are $\beta_0 = 1$ (a single connected tree) and $\beta_1 = 0$ (no cycles). Deviations indicate disconnected components or spurious loops in the predicted graph. Betti-error metrics are insensitive to small geometric inaccuracies and instead penalize structural failures that point- and branch-level metrics may miss.

Together, the point-, branch-, and graph-level metrics provide a more complete picture than any single measure, but they remain proxies for clinical utility. They assess the quality of the predicted centerline graph itself, rather than the accuracy of clinical measurements derived from it (such as seal lengths in EVAR follow-up CT) or whether errors occur in clinically critical regions of the anatomy. Closing this gap requires task-specific evaluation, which is the perspective taken in Chapter 4 and developed in Paper D.

3.8 Comparative Perspective and Open Challenges

Taken together, the literature shows a progression from indirect and handcrafted pipelines toward increasingly direct and structured image-to-graph methods. Model-based approaches introduced continuity and geometry priors, but were limited by handcrafted assumptions. Segmentation-based methods benefited from strong dense predictors, but remained dependent on post-processing. Direct graph prediction aligned the model more closely with the desired output, but raised challenges in graph validity and scaling. Iterative tracking and refinement methods are particularly attractive because they align the prediction process with the branching structure itself. Table 3.1 summarizes the methodological trade-offs across the five families, including topology guarantees, required training signal, and typical failure modes.

Table 3.1: Comparison of method families for centerline graph extraction from medical images. The right-most column gives representative examples, with papers included in this thesis shown in bold.

Family	Topology guarantees	Training signal	Typical failure modes	Representative methods
Model-based	No; tracing depends on handcrafted rules	None (unsupervised)	Sensitivity to hyperparameters; poor generalization across datasets and pathology	Multiple hypothesis template tracking [12], Divergence prior model [28]
Segmentation + skeletonization	No; topology depends on mask quality and post-processing	Voxel-wise segmentation labels	Broken or spurious branches from small segmentation errors; costly voxel-level annotation	DeepVesselNet [29], Deep Open Snake Tracker [30]
Direct image-to-graph	Partial; requires explicit constraints for validity	Graph annotations (nodes, edges)	Disconnected components; spurious cycles; scaling with graph size	Relationformer [31], Vesselformer [32]
Iterative tracking	Yes, by construction (sequential traversal)	Graph annotations along trajectories	Missed branches; premature termination; error accumulation	**Trexplorer (A)**, **Trexplorer Super (B)**
Iterative refinement	Yes, via structured trajectory units	Graph annotations with branch/trajectory grouping	Missed branches if initial trajectory proposals do not cover them	**RefTr (C)**

Several challenges remain central across the literature. Topology preservation is essential because disconnected predictions, duplicate branches, or cycles reduce downstream usefulness. Recovery of small distal branches remains difficult because such structures are thin, weakly contrasted, and easily confused with noise or artifacts. Robustness across datasets and acquisition conditions is also important, especially when moving from synthetic to real clinical data. Evaluation remains nontrivial, since point-level accuracy alone is insufficient for graph extraction. Finally, extraction quality must ultimately be judged in light of downstream use. These challenges motivate the structured, topology-aware, and evaluation-focused approaches developed in the papers included in this thesis.

3.9 Chapter Summary

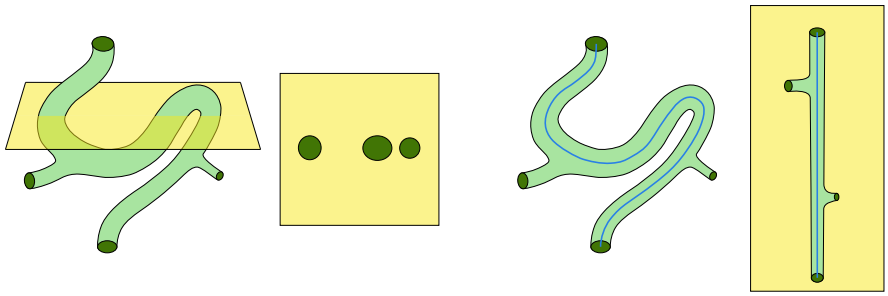
This chapter has reviewed the main methodological families for centerline graph extraction. Model-based methods established important geometric ideas, segmentation-based methods leveraged strong dense prediction, direct graph prediction moved closer to the desired output representation, and iterative tracking and refinement brought stronger structural reasoning. Within this progression, the work in this thesis contributes to the move toward end-to-end structured image-to-graph prediction. The next chapter turns from methodology to application and considers clinical translation.

CHAPTER 4

Clinical Translation

Centerline graph extraction is not only a technical problem in medical image analysis, but also one of clinical translation. In practice, centerlines are valuable because they support measurements, decisions, and workflows. This is particularly true in vascular imaging, where centerlines are used for diameter measurement, length estimation, anatomical localization, intervention planning, and follow-up [46], [47]. Their clinical usefulness lies less in the graph itself than in what it enables.

Centerlines provide a sparse but anatomically meaningful description of tubular structures. Unlike dense segmentations, they reduce the anatomy to its medial paths and branching organization. This aligns well with many clinical tasks, where measurements and decisions are naturally expressed along a vessel or airway path. In this role, the centerline acts as a coordinate system. Length is measured along it, orthogonal diameters are defined relative to it, and focal findings can be localized by position along the tree. This same representation also enables curved planar reconstruction (CPR) as shown in Figure 4.1, where image data are resampled along the centerline to produce a straightened view of the structure, simplifying visualization and assessment. Centerlines are also interpretable, since their topology is explicit and can be inspected directly. For



(a) Axial Sections

(b) Curved Planar Reconstruction

Figure 4.1: Illustration of axial sections versus curved planar reconstruction (CPR) for a tortuous tubular structure. (a) Conventional axial slices intersect the anatomy at fixed planes, resulting in fragmented cross-sectional views. (b) Using the centerline as a reference, CPR resamples the image along the path of the structure, generating a straightened view that preserves continuity and simplifies analysis.

these reasons, they are often a natural intermediate representation between the image and the downstream clinical task.

At the same time, the broader medical imaging literature has repeatedly noted a gap between promising algorithmic performance and successful deployment in routine care. Trust and adoption depend not only on accuracy, but also on transparent reporting, external testing, workflow integration, and alignment with real clinical endpoints [48], [49].These issues are directly relevant to centerline extraction.

This chapter discusses clinical translation from that perspective. It first outlines the main factors that shape the practical utility of centerline methods, and then turns to automated EVAR follow-up as the application that most directly grounds this thesis.

4.1 Clinical Translation and Practical Utility

For centerline extraction, the translation gap appears in several ways. Many methods are developed and evaluated on curated or synthetic datasets, often using local geometric metrics that do not fully capture downstream utility. Clinical use, however, depends on whether the resulting graph is topologically reliable, robust under difficult acquisition conditions, and directly usable in

measurement workflows.

Generalizability is a major issue. Models must remain robust across scanners, protocols, patient populations, and disease presentations. Workflow integration is another. Even a strong model may have limited value if its output cannot be incorporated into the existing review process without substantial manual correction. The relevance of the evaluation endpoint also matters. Point-level overlap or distance alone may say little about whether branches are missing, whether a graph is connected, or whether protocol-based measurements are reliable [50], [51].

These observations suggest a few general principles. Centerline extraction should be evaluated at the level of structural utility rather than only geometric proximity. Methods should be developed and tested on difficult clinical conditions rather than only idealized data. It is also valuable to move from centerline extraction as an isolated endpoint toward centerline-enabled downstream prediction. Finally, clinical usefulness depends strongly on automation of the full analysis chain, including initialization and measurement.

The work in this thesis is positioned directly within this perspective. Trexplorer Super emphasizes the difference between synthetic and real-data performance, RefTr strengthens evaluation through graph-level and radius-aware comparison, and CEVAR (Paper D) places centerline extraction inside a concrete clinical workflow with protocol-based measurement.

4.2 Automated EVAR Follow-up

The application that most directly grounds this thesis in clinical translation is follow-up after endovascular aneurysm repair (EVAR). In this setting, centerlines are central to structured review because seal lengths and orthogonal diameters are measured along the vessel centerline according to EVAR4C [52] protocol. This makes EVAR follow-up a clear example of why centerline extraction matters clinically. Figure 4.2 illustrates the recurring clinical workflow that an EVAR patient enters after intervention.

Current workflows often require manual centerline editing by expert operators, particularly in post-EVAR scans with altered anatomy, metal artifacts, and low or absent contrast. CEVAR (Paper D) addresses this by combining Transformer-based centerline extraction with embedding-based geometric measurement prediction in a protocol-driven framework. Within

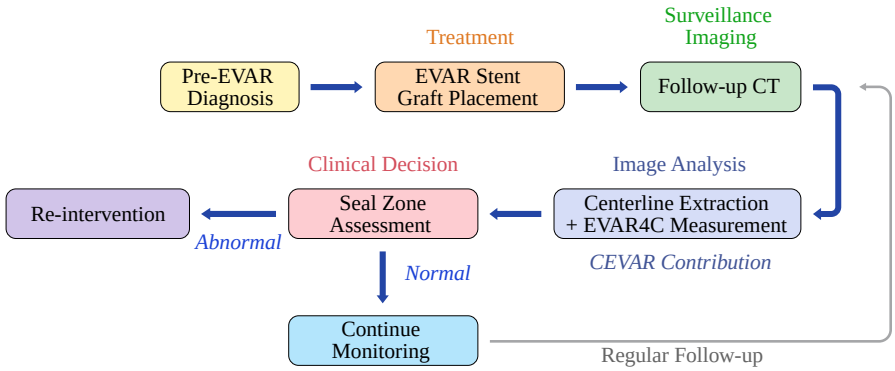


Figure 4.2: The EVAR clinical workflow. After pre-treatment diagnosis and stent graft placement, patients enter lifelong surveillance. Each follow-up CT is reviewed by extracting an aorto-iliac centerline and computing measurements according to the EVAR4C protocol, after which a seal zone assessment determines whether the patient continues regular follow-up or undergoes re-intervention. CEVAR (Paper D) contributes an automated centerline extraction and measurement pipeline that targets the image analysis step.

Part I, this work illustrates the thesis perspective that centerline graphs can support localized feature prediction in clinically meaningful workflows; the methodological details are presented in Part II.

4.3 Chapter Summary

This chapter has argued that centerline graph extraction is well-suited to clinical translation because centerlines provide compact, interpretable, and anatomically meaningful representations for tubular structures. Their value lies in supporting localized measurements and structured reasoning. At the same time, clinical translation requires more than technical accuracy. Generalizability, workflow integration, clinically relevant evaluation, and automation are all essential if centerline methods are to move beyond benchmark performance. The next chapter returns to the methodological side and discusses image feature extraction in greater detail.

CHAPTER 5

Feature Extraction

The extraction of centerline graphs from tubular tree structures depends not only on the output formulation, but also on how image features are computed. In volumetric CT, the structures of interest are often thin, elongated, sparsely distributed, and embedded in heterogeneous surrounding anatomy. A model must therefore preserve fine local detail while also capturing enough context to reason about continuity, bifurcations, and anatomical location.

Tubular structures place unusual demands on image representations. They exhibit large-scale variation, may be only a voxel wide, and often occupy only a small fraction of the full image volume [10], [40]. Reliable analysis therefore requires fine spatial detail, long-range context, robustness to strong class imbalance, and computational efficiency. From this perspective, feature extraction is not merely a preprocessing step before graph prediction. It is one of the central methodological bottlenecks in sparse anatomical analysis.

In the centerline extraction setting studied in this thesis, this bottleneck appears as a practical trade-off between spatial resolution and field of view. High-resolution input is needed to resolve small distal branches, but a limited input patch may not provide enough anatomical context for tracking through large branches or bifurcations. Increasing the patch size can address this only

at substantial computational cost. This makes efficient feature extraction especially important for centerline tracking in 3D medical images.

This chapter reviews the main ideas in image feature extraction that are relevant to the thesis. It first discusses CNN- and Transformer-based dense feature extractors, then turns to adaptive computation and coarse-to-fine processing, and finally places ARTA (Paper E) in that broader context.

5.1 Dense Feature Extraction

CNNs [16] were the first deep architectures to dominate dense image analysis and remain foundational in medical imaging. Through convolution, pooling, and multiscale processing, they build hierarchical representations that preserve spatial structure while increasing semantic abstraction. U-Net [14] and related encoder-decoder architectures have been especially influential in biomedical segmentation because they combine high-level features with fine spatial detail through skip connections. For tubular structures, CNNs remain attractive because they are efficient, mature, and effective at capturing local appearance and multiscale context. Their main limitation in this thesis setting is that long-range interaction is modeled only indirectly. For elongated tree structures, this can make continuity and connectivity difficult to resolve over larger spatial ranges.

While CNNs have long been the dominant backbone for dense medical image analysis, Transformers have introduced a complementary approach with stronger long-range interaction. Transformer-based [17], [19] models replace fixed local convolution with attention-based interaction among tokens. This makes them attractive for dense prediction tasks where long-range dependencies matter. In medical imaging, hierarchical Transformer backbones have become increasingly important for segmentation and related dense tasks, particularly in 3D settings. For sparse anatomical structures, Transformers offer improved long-range context modeling and interface naturally with query-based and structured prediction frameworks. Their main challenge is computational cost, especially in large volumetric data, which has motivated hierarchical, sparse, and windowed designs [18].

Taken together, CNNs and Transformers offer complementary strengths for tubular structure analysis. CNNs provide efficient local and multiscale feature extraction, while Transformers offer more flexible long-range interaction. The

choice between them, or their combination, depends on how local detail, global context, and computational cost are balanced for the task at hand.

5.2 Adaptive Computation and Coarse-to-Fine Processing

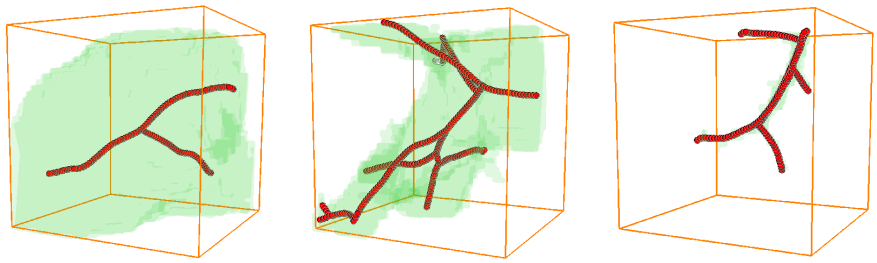
A key observation in dense prediction is that not all parts of an image are equally informative. Large regions may be semantically homogeneous, while thin structures, boundaries, and small objects require more precise representation. This is especially true in medical images, where vessels or airways occupy only a small fraction of the image.

This motivates adaptive computation and content-aware processing. Rather than applying the same computation everywhere, the model attempts to allocate more capacity where it is most needed. This idea is highly relevant to centerline extraction because the target structures are sparse, fine-scale, and topologically organized.

Most dense vision models still follow a fine-to-coarse strategy, starting from dense high-resolution input and progressively reducing spatial resolution. Although natural, this means that all regions are processed initially at high resolution, even when much of the image is semantically simple. Several recent methods aim to make this strategy more efficient by reducing redundancy after dense tokenization, for example through token sharing, pruning, or merging [53], [54], [55]. These approaches can improve efficiency, but they still begin from a uniformly fine representation and only later compress it.

A coarse-to-fine strategy reverses this logic. It starts with coarse representations that provide global coverage and allocates finer resolution only where the image content suggests that it is needed. For sparse fine-structure analysis, this is attractive because it treats high-resolution as a selective resource rather than a default. In this sense, coarse-to-fine processing complements token-merging and token-pruning approaches by addressing inefficiency at an earlier stage of the representation pipeline.

The centerline tracking models developed in this thesis encounter this trade-off directly at the patch level. Trexplorer (Paper A), Trexplorer Super (Paper B), and RefTr (Paper C) operate on fixed-size 3D patches, which must simultaneously provide enough field of view to resolve large proximal branches and enough fine detail to recover small distal branches. As shown earlier in

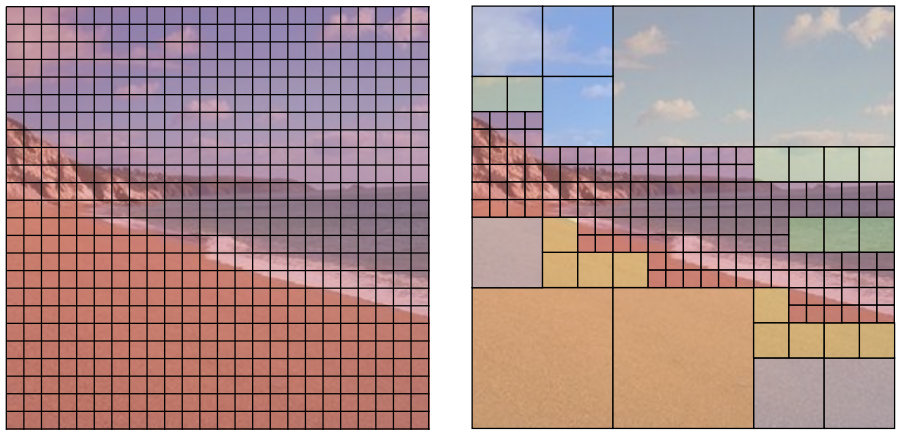


(a) Large vessels **(b)** Medium vessels **(c)** Small vessels

Figure 5.1: Trade-off between field of view and spatial detail in fixed-resolution patch-based processing of PARSE 2022. All three panels show the same 64×64×64 patch (orange box) with vessel segmentation (green) and centerline (red). (a) A large proximal vessel can fill nearly the entire patch, leaving little surrounding context to infer tracking direction or detect the next bifurcation. (b) Medium-sized vessels fit comfortably within the patch with adequate context. (c) Small distal branches occupy only a tiny fraction of the patch volume, so most of the high-resolution capacity is spent on empty space. Adaptive feature extraction methods aim to allocate fine resolution selectively to the regions where it is most useful.

Section 3.6, branch radius varies substantially within the datasets used, and in the most extreme cases the largest branch radius exceeds the spatial extent of the input patch (Figure 5.1). When this happens, the branch can fill the entire patch and leave too little surrounding context for the model to infer the tracking direction or detect upcoming bifurcations, leading to missed branches.

Practical workarounds during development, such as downsampling to a lower resolution to gain field of view or increasing the patch size for finer detail, helped in some cases but introduced their own trade-offs in spatial detail or training cost. These experiences highlight a central limitation of fixed-resolution patch-based processing: a single representation cannot efficiently serve the full range of radius scales encountered in 3D tubular tree analysis. This tension is one of the main motivations for exploring adaptive feature extraction methods that allocate computational resources where they are most needed.



(a) Uniform fine tokenization **(b)** Mixed-resolution token allocation

Figure 5.2: Comparison of token allocation strategies for dense feature extraction. (a) Conventional vision Transformers tokenize the input image into a uniform grid of fine patches, producing many tokens regardless of local content. Homogeneous regions such as the sky and beach receive the same token density as the more informative coastline. (b) ARTA instead allocates tokens adaptively, using coarse tokens in homogeneous regions and finer tokens near semantically complex areas such as object boundaries. This concentrates computation where higher resolution is most useful and substantially reduces the total number of tokens.

5.3 Adaptive Mixed-Resolution Token Allocation (ARTA)

ARTA (Paper E) implements this coarse-to-fine idea as a mixed-resolution Transformer architecture. Conventional vision Transformers tokenize the input into a uniform fine grid (Figure 5.2a), spending equal computation on every region of the image regardless of content. ARTA instead begins with coarse tokens and uses a lightweight allocator to introduce finer tokens only in regions that appear semantically complex (Figure 5.2b), concentrating computation near boundaries while keeping homogeneous regions coarse. The resulting mixed-resolution token set is processed jointly through attention.

For the purposes of this thesis, ARTA is relevant less for its original application domain than for its computational principle: representing the image

Table 5.1: Point-level centerline extraction performance on the PARSE 2022 dataset using ARTA-3D as the feature extractor. Results are reported for Trexplorer Super and RefTr with radius-aware point metrics and radius mean absolute error (MAE).

Model	Point-level@ $\tau^{\text{rad}} = [0.25{:}0.05{:}0.75]$			
	rAP(%)↑	rAR(%)↑	rF1(%)↑	Radius (MAE)↓
RefTr (SwinUNETR)	52.68	**37.80**	**42.63**	0.69
RefTr (ARTA-3D)	**58.12**	34.90	42.55	**0.64**

Table 5.2: Comparison of centerline extraction methods on the PARSE 2022 dataset using branch-level accuracy and graph-level topology error. Branch performance is reported with radius-aware metrics, while graph quality is assessed using Betti-0 and Betti-1 mean absolute error (MAE).

Model	Branch-level@ $\tau^{\text{match}} = [0.4{:}0.05{:}0.8]$			Graph-level (MAE)	
	rBAP(%)↑	rBAR(%)↑	rBF1(%)↑	Betti-0↓	Betti-1↓
RefTr (SwinUNETR)	46.41	**33.30**	**37.29**	**0.00**	**0.00**
RefTr (ARTA-3D)	**51.00**	30.78	**37.29**	**0.00**	**0.00**

with a non-uniform token distribution that follows content complexity. In centerline extraction, proximal branches with large radii require a sufficient field of view to resolve direction and bifurcation structure, whereas small distal branches require fine spatial detail. Fixed-resolution patch-based processing makes these demands difficult to satisfy simultaneously at a reasonable cost. Tubular trees therefore provide a natural setting for mixed-resolution feature extraction, where coarse tokens capture broader context and finer tokens focus on boundaries, bifurcations, and small branches.

A preliminary 3D extension (ARTA-3D), explored in this thesis but not part of the included paper, applies the mixed-resolution principle to volumetric medical imaging. Both ARTA and ARTA-3D use a lightweight allocator to produce a mixed-resolution token set, which is then processed by a heavier refiner. The refiner jointly attends over all active tokens at each layer, and progressively ejects one resolution at a time as the network deepens, so that the active token set shrinks toward the deepest layers. In ARTA, the finest tokens are ejected first, leaving only coarse tokens in the deepest layers, while in ARTA-3D the order is reversed: coarse tokens are ejected first, leaving only

Table 5.3: Computational efficiency of RefTr with SwinUNETR and ARTA-3D as feature extractors on the PARSE 2022 dataset. We report the number of encoder parameters, average inference runtime per patch, average training time per iteration, and peak GPU memory usage.

Model	Enc. Params (M)↓	Runtime (ms/patch)↓	Training Time (sec/it)↓	GPU Mem. (GB)↓
RefTr (SwinUNETR)	23.13	82	1.117	50.02
RefTr (ARTA-3D)	**17.20**	**53**	**1.115**	**46.05**

fine tokens in the deepest layers. In both cases the refiner only ever processes a sparse mixed-resolution token set, never a dense uniformly fine one, so the usual cost concern with starting from fine tokens does not apply. The essential contribution is the adaptive mixed-resolution token allocation itself, with the layer-wise ejection order chosen empirically per task.

Preliminary results using ARTA-3D as the feature extractor for centerline extraction in PARSE 2022 are reported in Tables 5.1 and 5.2 for extraction quality, and in Table 5.3 for computational efficiency. ARTA-3D matches SwinUNETR on overall F1 at both the point level and the branch level, with consistently higher precision (rAP and rBAP) traded for slightly lower recall (rAR and rBAR). ARTA-3D also produces more accurate radius estimates, and both backbones yield topologically valid graphs with Betti-0 and Betti-1 errors of zero. At the same time, ARTA-3D uses approximately 25% fewer encoder parameters, runs 35% faster at inference, and consumes roughly 8% less peak GPU memory than SwinUNETR. Hyperparameters for ARTA-3D were not tuned in these experiments, so the small recall gap may close with proper tuning. Taken together, these preliminary numbers suggest that adaptive mixed-resolution tokenization can match a strong fixed-resolution baseline at substantially lower computational cost.

The current implementation is a first integration of ARTA-3D as a feature extractor and is not optimized for memory or runtime. Further optimization is expected to reduce training and inference time, lower memory consumption, and enable larger input patch sizes and broader training configurations.

5.4 Chapter Summary

This chapter reviewed the role of image feature extraction in tubular tree analysis. CNNs and Transformers provide complementary approaches to dense feature extraction, while adaptive computation addresses the imbalance between image size, field of view, and target sparsity. ARTA illustrates a coarse-to-fine strategy that is especially relevant to sparse anatomical structures and to the practical trade-off between context and detail in 3D centerline tracking. The next chapter summarizes the included papers and explains how each contributes to the overall thesis argument.

CHAPTER 6

Summary of Included Papers

This chapter provides a summary of the included papers.

6.1 Paper A

Roman Naeem, David Hagerman, Lennart Svensson, Fredrik Kahl
Trexplorer: Recurrent DETR for Topologically Correct Tree Centerline Tracking
Published in Proceedings of International Conference on Medical Image Computing and Computer-Assisted Intervention
pp. 744–754, Springer, 2024

Summary: This paper introduces Trexplorer, a recurrent DETR-based [9] framework for extracting tubular tree centerlines from 3D medical images while explicitly preserving valid tree topology. The method formulates centerline extraction as a structured breadth-first tracking problem, in which object queries recurrently predict child centerline nodes from local image features. In contrast to segmentation-based pipelines that require subsequent

skeletonization and graph repair, Trexplorer generates the centerline tree directly and guarantees a topologically correct structure without post-processing. Experiments on a synthetic vessel dataset show that the method preserves correct tree topology while maintaining strong detection performance.

Contributions: Roman contributed to idea generation, model implementation, experimentation, result analysis, creating the illustrations, and writing the paper. David, Lennart, and Fredrik contributed to idea generation, result analysis, and writing the paper.

6.2 Paper B

Roman Naeem, David Hagerman, Jennifer Alvén, Lennart Svensson, Fredrik Kahl
Trexplorer Super: Topologically Correct Centerline Tree Tracking of Tubular Objects in CT Volumes
Published in Proceedings of International Conference on Medical Image Computing and Computer-Assisted Intervention,
pp. 595–605, Springer, 2025

Summary: This paper extends Trexplorer into a more robust and practically effective framework for centerline tree extraction in CT data. Trexplorer Super addresses two important limitations of the original model, namely duplicate branch prediction and premature tracking termination, through several methodological improvements, including Super Trajectory Training, Focal Cross Attention, and Target Augmentation. The paper also contributes one synthetic and two real centerline datasets of increasing difficulty, together with point-, branch-, and graph-level evaluation metrics, enabling broader and more realistic benchmark evaluation. Across these datasets, Trexplorer Super consistently outperforms previous state-of-the-art methods and further shows that strong results on synthetic data do not necessarily transfer to real clinical scans.

Contributions: Roman contributed to idea generation, dataset and benchmark creation, model implementation, experimentation, result analysis, creating the illustrations, and writing the paper. David, Jennifer, Lennart, and Fredrik contributed to idea generation, result analysis, and writing the paper.

6.3 Paper C

Roman Naeem, David Hagerman, Jennifer Alvén, Fredrik Kahl
RefTr: Recurrent Refinement of Confluent Trajectories for 3D Tubular Tree Centerlines
Under Review

Summary: This paper presents RefTr, a Transformer-based [17] image-to-graph framework for tubular tree centerline extraction in 3D medical images that shifts from point-wise tracking to recurrent refinement of whole branch trajectories. The method represents the tree using confluent trajectories, which share a common path until a bifurcation and then diverge, and uses a Producer–Refiner architecture in which candidate trajectories are iteratively refined toward target branches. This design improves both recall and precision while reducing decoder complexity compared with previous recurrent tracking methods. The paper also introduces Tree Non-Maximum Suppression for duplicate branch suppression and radius-aware evaluation metrics for anatomically informed comparison, demonstrating strong performance across multiple public datasets.

Contributions: Roman contributed to idea generation, model implementation, experimentation, result analysis, evaluation metrics, creating the illustrations, and writing the paper. David, Jennifer, and Fredrik contributed to idea generation, result analysis, and writing the paper.

6.4 Paper D

Roman Naeem, Timo Niiniskorpi, Naman Desai, Charlotte Sandström, Anders Jeppsson, Ida Häggström, Fredrik Kahl, Håkan Roos, Jennifer Alvén
CEVAR: Centerline Embedding Extraction for Endovascular Aneurysm Repair
Under Review

Summary: This paper investigates how learned centerline representations can support clinically meaningful downstream prediction in follow-up

after endovascular aneurysm repair (EVAR). The proposed framework combines automated 3D aorto-iliac centerline extraction with point-wise centerline embeddings used to predict protocol-relevant geometric quantities, including stent position, vessel diameters, and seal lengths according to the EVAR4C [52] protocol. The paper evaluates two state-of-the-art image-to-graph models on a private longitudinal clinical cohort of 374 scans from 142 EVAR patients, including both contrast and non-contrast follow-up CT, with manual expert-refined centerlines and annotations. It shows that a fully automatic learning-based workflow can outperform a commercial semi-automatic centerline analysis pipeline. The work demonstrates that centerline graphs are not only useful structural outputs, but also effective substrates for localized clinical feature prediction.

Contributions: Roman contributed to idea generation, development of the public and private datasets, model implementation, experimentation, result analysis, creating the illustrations, and writing the paper. Timo and Anders contributed to idea generation and development of the private dataset. Naman contributed to idea generation and implementation of the automatic seed point generation model. Ida and Fredrik contributed to idea generation. Charlotte, Håkan and Jennifer contributed to idea generation, development of the private dataset, result analysis and writing the paper.

6.5 Paper E

David Hagerman*, **Roman Naeem***, Erik Brorsson, Fredrik Kahl, Lennart Svensson
ARTA: Adaptive Mixed-Resolution Token Allocation for Efficient Dense Feature Extraction
Under Review

Summary: This paper presents ARTA, a coarse-to-fine mixed-resolution vision Transformer for efficient dense feature extraction. Unlike conventional dense prediction models that begin with uniformly high-resolution tokens, ARTA starts with coarse tokens and uses a lightweight allocator to identify regions that require finer representation, concentrating computation near

*Shared first authorship.

semantic boundaries while keeping homogeneous regions coarse. The resulting mixed-resolution token set is processed jointly through attention, enabling efficient interaction between coarse and fine features. Experiments on semantic segmentation benchmarks show that ARTA achieves state-of-the-art or competitive accuracy with substantially lower computational cost, providing an important methodological contribution to adaptive feature extraction for sparse and fine-grained visual structure.

Contributions: David and Roman contributed to idea generation, model implementation, result analysis, preparation of figures, and paper writing. David conducted the experiments. Erik, Fredrik, and Lennart contributed to idea generation, result analysis, and paper writing.

CHAPTER 7

Concluding Remarks and Future Work

7.1 Main Findings

This thesis has examined methods for centerline graph extraction of tubular tree structures in medical images and their role in structured image analysis and localized prediction, with particular emphasis on CT volumes. Across the thesis, centerline graphs have been treated as compact and interpretable representations that support measurement, reasoning, and downstream prediction. The contributions can be summarized in relation to the research questions posed in Chapter 1.

RQ1 & RQ2: Topology-preserving extraction with recurrent Transformer methods

Papers A–C contribute recurrent Transformer-based image-to-graph methods that produce topologically valid centerline trees by construction. Trexplorer (Paper A) showed that a recurrent DETR-based formulation can extract centerline trees with controlled topology by tracking branches sequentially through 3D patches. Trexplorer Super (Paper B) improved over Trexplorer with

architectural and training refinements, and demonstrated that the approach generalizes from synthetic data to real anatomical structures by extending evaluation to two public CT datasets (ATM'22 and PARSE 2022). RefTr (Paper C) advanced the line further by predicting trajectories rather than individual nodes, introducing duplicate suppression through tree non-maximum suppression and radius-aware evaluation. Across the three papers, results show consistent improvements in branch recovery and topological correctness over comparable methods that do not enforce tree structure during prediction, supporting the conclusion that topology-aware design is not only methodologically appealing but empirically beneficial for tubular tree extraction.

RQ3: Evaluation across synthetic and real datasets

Papers B and C extend evaluation from synthetic benchmarks to public real CT data and develop metrics that assess accuracy at the point, branch, and graph levels. Paper D adds a clinically curated cohort, completing a dataset progression from controlled synthetic data to real clinical follow-up data. The metrics introduced and adopted across these papers, especially radius-aware matching and Betti-error evaluation, expose failure modes such as missed distal branches and spurious cycles that point-level metrics alone do not capture.

RQ4: Centerline-based localized clinical prediction

CEVAR (Paper D) demonstrates that centerlines and point-wise centerline embeddings can support protocol-driven clinical analysis in follow-up CT after EVAR. The proposed pipeline predicts stent position, vessel diameters, and seal lengths directly from a learned centerline representation, replacing a separate post-hoc geometric step. It outperforms a commercial semi-automatic workflow on both contrast and non-contrast clinical CT. This shows that centerline graphs are not only structural outputs but also useful representations for downstream prediction tasks in clinical workflows.

RQ5: Adaptive and efficient feature extraction

ARTA (Paper E) contributes an adaptive mixed-resolution Transformer architecture for dense feature extraction. While the published paper is developed and evaluated on natural images, the underlying principle of representing the

image with a non-uniform token distribution that follows content complexity is directly relevant to sparse anatomical analysis, where vessel and airway structures occupy only a small fraction of large 3D volumes. A preliminary 3D extension (ARTA-3D), reported in Chapter 5, provides early evidence that this principle transfers to volumetric medical imaging.

Taken together, the work supports a structured, topology-aware, and clinically grounded view of medical centerline extraction. The methodological framework developed here, comprising recurrent set prediction for sequentially structured outputs and adaptive token allocation for sparse dense prediction, is also relevant to broader computer vision tasks that share these structural properties.

7.2 Limitations

Although the work presented in this thesis advances centerline extraction substantially, several limitations remain. Generalization across scanners, protocols, institutions, anatomical targets, and modalities beyond CT still requires further evaluation. Although the broadened evaluation in Trexplorer Super is an important step, more work is needed to capture the full variability encountered in clinical imaging.

Reliable recovery of very small distal branches also remains difficult because such branches are thin, weakly contrasted, and easily confused with background or artifacts. Although RefTr improves recall, this problem is not fully resolved.

In addition, the centerline methods are not yet fully unified. Some rely on manual initialization, and in the automated EVAR pipeline the seed is generated by a separate model. This leaves room for more integrated automation.

Clinical validation is also still limited in scope. CEVAR provides strong evidence of translational relevance, but it remains one application with a specific protocol and cohort. Broader validation would require larger multi-center studies and prospective assessment of workflow impact.

The contribution of ARTA in this thesis is primarily methodological. While the preliminary ARTA-3D extension presented in Chapter 5 provides initial evidence of relevance to volumetric medical imaging, a thorough evaluation across anatomies and clinical conditions remains future work.

7.3 Future Work

Several natural directions for future research follow from these limitations and from the broader findings of the thesis. One is the development of more robust centerline extraction models across anatomies, modalities, and acquisition settings. Another is uncertainty-aware graph extraction, in which the model predicts not only the centerline but also confidence in branches, bifurcations, or downstream measurements.

Longitudinal modeling is also important, particularly in follow-up settings such as EVAR, where clinically relevant questions concern anatomical change over time. On the methodological side, trajectory-based refinement remains a rich design space, and stronger integration between image encoders and graph refinement mechanisms is likely to be beneficial.

Further progress in evaluation is also needed. Point-, branch-, and graph-level metrics with radius-aware thresholds are important, but future work could relate graph quality more directly to downstream measurement error and clinical decision support.

Adaptive mixed-resolution feature extraction for 3D medical imaging is a particularly promising direction. As discussed in Chapter 5, fixed-size high-resolution patches create a practical trade-off between spatial detail and field of view in centerline tracking. Content-adaptive token allocation addresses this trade-off directly by allowing the model to use coarse representations for large proximal branches and fine representations for small distal branches within the same patch, making it a natural fit for vessels, airways, and other sparse fine structures.

The methodological contributions of this thesis also extend beyond medical imaging. The recurrent DETR-based formulation underlying Trexplorer and Trexplorer Super provides a general framework for sequential set prediction in tasks where outputs are themselves structured trajectories rather than independent objects. Closely related applications include road and lane graph extraction in aerial and street-level imagery, river network mapping in remote sensing, and multi-object tracking, all of which share the requirement of producing topologically valid, sequentially structured outputs from images. The trajectory-based refinement and duplicate suppression introduced in RefTr are similarly general: they apply wherever an overcomplete set of candidate trajectories must be reduced to a coherent set of structured paths. Exploring these cross-domain transfers would help validate the generality of the methods

and could motivate further architectural refinements.

In summary, the work presented here establishes a foundation for structured tubular tree analysis in medical images while opening several directions for future research. Although the focus has been on medical centerline extraction, the methodological contributions of recurrent structured set prediction and adaptive mixed-resolution feature extraction are also relevant to broader computer vision tasks that share these structural properties.

References

[1] A. Ismail et al., "Carotid artery stenosis: A look into the diagnostic and management strategies, and related complications," *Cureus*, vol. 15, no. 5, e38794, 2023.

[2] A. A. Diaz, R. S. J. Estépar, and G. R. Washko, "Computed tomographic airway morphology in chronic obstructive pulmonary disease. remodeling or innate anatomy?" *Annals of the American Thoracic Society*, vol. 13, no. 1, pp. 4–9, 2016.

[3] D. Barnes et al., "Central airway pathology: Clinic features, CT findings with pathologic and virtual endoscopy correlation," *Insights into Imaging*, vol. 8, no. 2, pp. 255–270, 2017.

[4] S. Moccia, E. De Momi, S. El Hadji, and L. S. Mattos, "Blood vessel segmentation algorithms – review of methods, datasets and evaluation metrics," *Computer Methods and Programs in Biomedicine*, vol. 158, pp. 71–91, 2018.

[5] D. Huang, W. Tang, Y. Ding, T. Wan, and Y. Chen, "An interactive 3D preoperative planning and training system for minimally invasive vascular surgery," in *2011 12th International Conference on Computer-Aided Design and Computer Graphics*, IEEE, 2011, pp. 443–449.

[6] A. D. Choi et al., "CT evaluation by artificial intelligence for atherosclerosis, stenosis and vascular morphology (CLARIFY): A multi-center, international study," *Journal of Cardiovascular Computed Tomography*, vol. 15, no. 6, pp. 470–476, 2021.

[7] G. Mistelbauer et al., "Semi-automatic vessel detection for challenging cases of peripheral arterial disease," *Computers in Biology and Medicine*, vol. 133, p. 104 344, 2021.

[8] D. Jia and X. Zhuang, "Learning-based algorithms for vessel tracking: A review," *Computerized Medical Imaging and Graphics*, vol. 89, p. 101 840, 2021.

[9] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European Conference on Computer Vision*, Springer, 2020, pp. 213–229.

[10] G. Luo et al., "Efficient automatic segmentation for multi-level pulmonary arteries: The PARSE challenge," *arXiv:2304.03708*, 2023.

[11] A. F. Frangi, W. J. Niessen, K. L. Vincken, and M. A. Viergever, "Multiscale vessel enhancement filtering," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 1998, pp. 130–137.

[12] O. Friman, M. Hindennach, C. Kühnel, and H.-O. Peitgen, "Multiple hypothesis template tracking of small 3D vessel structures," *Medical Image Analysis*, vol. 14, no. 2, pp. 160–171, 2010.

[13] Z. Zhang, D. Marin, M. Drangova, and Y. Boykov, "Confluent vessel trees with accurate bifurcations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 9573–9582.

[14] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2015, pp. 234–241.

[15] G. Balakrishnan, A. Zhao, M. R. Sabuncu, J. Guttag, and A. V. Dalca, "VoxelMorph: A learning framework for deformable medical image registration," *IEEE Transactions on Medical Imaging*, vol. 38, no. 8, pp. 1788–1800, 2019.

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 25, 2012.

[17] A. Vaswani et al., "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[18] A. Hatamizadeh, V. Nath, Y. Tang, D. Yang, H. R. Roth, and D. Xu, "Swin UNETR: Swin transformers for semantic segmentation of brain tumors in MRI images," in *International MICCAI Brainlesion Workshop*, Springer, 2021, pp. 272–284.

[19] A. Dosovitskiy et al., "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2020.

[20] Z. Liu et al., "Swin Transformer: Hierarchical vision transformer using shifted windows," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, Montreal, QC, Canada: IEEE, Oct. 2021, pp. 9992–10 002, ISBN: 978-1-6654-2812-5.

[21] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[22] B. Liao et al., "MapTR: Structured modeling and learning for online vectorized HD map construction," *arXiv preprint arXiv:2208.14437*, 2022.

[23] Z. Xu, Y. Liu, Y. Sun, M. Liu, and L. Wang, "CenterlineDet: Centerline graph detection for road lanes with vehicle-mounted sensors by transformer for HD map generation," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2023, pp. 3553–3559.

[24] J. Pu et al., "CT based computerized identification and analysis of human airways: A review," *Medical Physics*, vol. 39, no. 5, pp. 2603–2616, 2012.

[25] O. Miraucourt, S. Salmon, M. Szopos, and M. Thiriet, "Blood flow in the cerebral venous system: Modeling and simulation," *Computer Methods in Biomechanics and Biomedical Engineering*, vol. 20, no. 5, pp. 471–482, 2017.

[26] S. Shit et al., "clDice – a novel topology-preserving loss function for tubular structure segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 16 560–16 569.

[27] R. Naeem, D. Hagerman, J. Alvén, L. Svensson, and F. Kahl, "Trexplorer Super: Topologically correct centerline tree tracking of tubular objects in CT volumes," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2025, pp. 595–605.

[28] Z. Zhang, D. Marin, E. Chesakov, M. M. Maza, M. Drangova, and Y. Boykov, "Divergence prior and vessel-tree reconstruction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 10 216–10 224.

[29] G. Tetteh et al., "DeepVesselNet: Vessel segmentation, centerline prediction, and bifurcation detection in 3-D angiographic volumes," *Frontiers in Neuroscience*, vol. 14, p. 1285, 2020.

[30] L. Chen et al., "Deep open snake tracker for vessel tracing," in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2021*, Springer, 2021, pp. 579–589.

[31] S. Shit et al., "Relationformer: A unified framework for image-to-graph generation," in *European Conference on Computer Vision*, Springer, 2022, pp. 422–439.

[32] C. Prabhakar et al., "Vesselformer: Towards complete 3D vessel graph generation from images," in *Medical Imaging with Deep Learning*, PMLR, 2024, pp. 320–331.

[33] J. M. Wolterink, R. W. van Hamersvelt, M. A. Viergever, T. Leiner, and I. Išgum, "Coronary artery centerline extraction in cardiac CT angiography using a CNN-based orientation classifier," *Medical Image Analysis*, vol. 51, pp. 46–60, 2019.

[34] P. Zhang, F. Wang, and Y. Zheng, "Deep reinforcement learning for vessel centerline tracing in multi-modality 3D volumes," in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2018*, Springer, 2018, pp. 755–763.

[35] Y. Zhang, G. Luo, W. Wang, and K. Wang, "Branch-aware double DQN for centerline extraction in coronary CT angiography," in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2020*, Springer, 2020, pp. 35–44.

[36] Z. Li, Q. Xia, Z. Hu, W. Wang, L. Xu, and S. Zhang, "A deep reinforced tree-traversal agent for coronary artery centerline extraction," in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2021*, Springer, 2021, pp. 418–428.

[37] U. Wickramasinghe, E. Remelli, G. Knott, and P. Fua, "Voxel2Mesh: 3D mesh model generation from volumetric data," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2020, pp. 299–308.

[38] A. Hoopes, J. E. Iglesias, B. Fischl, D. Greve, and A. V. Dalca, "TopoFit: Rapid reconstruction of topologically-correct cortical surfaces," *Proceedings of Machine Learning Research*, vol. 172, p. 508, 2022.

[39] Z. A. Sexton, *Synthetic vascular toolkit*, Accessed: 2025-02-24, 2023.

[40] M. Zhang et al., "Multi-site, multi-domain airway tree modeling," *Medical Image Analysis*, vol. 90, p. 102 957, 2023.

[41] L. Antiga and D. A. Steinman, *VMTK: The vascular modeling toolkit*, Accessed: 2025-02-24.

[42] Slicer Development Community, *3D Slicer*, Accessed: 2025-02-24, 2025.

[43] W. Silversmith, J. A. Bae, P. H. Li, and A. Wilson, *Kimimaro: Skeletonize densely labeled 3D image segmentations*, version 3.0.0, Accessed: 2021-09-29, 2021.

[44] M. Imran, J. R. Krebs, M. A. Cooper, J. Ma, Y. Zhou, and W. Shao, "Multi-class segmentation of the aorta," in *AortaSeg 2024 Challenge, Held in Conjunction with MICCAI 2024*, Cham, Switzerland: Springer, 2024.

[45] M. Schaap et al., "Standardized evaluation methodology and reference database for evaluating coronary artery centerline extraction algorithms," *Medical Image Analysis*, vol. 13, no. 5, pp. 701–714, 2009.

[46] F. Rengier, T. F. Weber, F. L. Giesel, D. Böckler, H.-U. Kauczor, and H. von Tengg-Kobligk, "Centerline analysis of aortic CT angiographic examinations: Benefits and limitations," *American Journal of Roentgenology*, vol. 192, no. 5, W255–W263, 2009.

[47] S. M. Han et al., "Comparison of intravascular ultrasound- and centerline computed tomography-determined aortic diameters during thoracic endovascular aortic repair," *Journal of Vascular Surgery*, vol. 66, no. 4, pp. 1184–1191, 2017.

[48] A. S. Tejani et al., "Checklist for artificial intelligence in medical imaging (CLAIM): 2024 update," *Radiology: Artificial Intelligence*, vol. 6, no. 4, e240300, 2024.

[49] A. S. Tejani, T. S. Cook, M. Hussain, T. Sippel Schmidt, and K. P. O'Donnell, "Integrating and adopting AI in the radiology workflow: A primer for standards and integrating the healthcare enterprise (IHE) profiles," *Radiology*, vol. 311, no. 3, e232653, 2024.

[50] J. H. Ketola et al., "Testing process for artificial intelligence applications in radiology practice," *Physica Medica*, vol. 128, p. 104 842, 2024.

[51] K. Wenderott, J. Krups, F. Zaruchas, and M. Weigl, "Effects of artificial intelligence implementation on efficiency in medical imaging – a systematic literature review and meta-analysis," *NPJ Digital Medicine*, vol. 7, no. 1, p. 265, 2024.

[52] C. Sandström et al., "Sealing zone failure decreases the long term durability of endovascular aneurysm repair," *European Journal of Vascular and Endovascular Surgery*, vol. 69, no. 2, pp. 238–247, 2025, ISSN: 1078-5884.

[53] C. Lu, D. de Geus, and G. Dubbelman, "Content-aware token sharing for efficient semantic segmentation with vision transformers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 23 631–23 640.

[54] Q. Tang, B. Zhang, J. Liu, F. Liu, and Y. Liu, "Dynamic token pruning in plain vision transformers for semantic segmentation," in *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2023, pp. 777–786.

[55] C. Ziwen et al., "AutoFocusFormer: Image segmentation off the grid," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 18 227–18 236.

Part II

Papers

PAPER A

Trexplorer: Recurrent DETR for Topologically Correct Tree Centerline Tracking

Roman Naeem, David Hagerman, Lennart Svensson, Fredrik Kahl

Published in Proceedings of International Conference on Medical Image Computing and Computer-Assisted Intervention
pp. 744–754, Springer, 2024

The layout has been revised.

Abstract

Tubular structures with tree topology such as blood vessels, lung airways, and more are abundant in human anatomy. Tracking these structures with correct topology is crucial for many downstream tasks that help in early detection of conditions such as vascular and pulmonary diseases. Current methods for centerline tracking suffer from predicting topologically incorrect centerlines and complex model pipelines. To mitigate these issues we propose Trexplorer, a recurrent DETR based model that tracks topologically correct centerlines of tubular tree objects in 3D volumes using a simple model pipeline. We demonstrate the model's performance on a publicly available synthetic vessel centerline dataset and show that our model outperforms the state-of-the-art on centerline topology and graph-related metrics, and performs well on detection metrics. The code is available at https://github.com/RomStriker/Trexplorer.

Keywords: Centerline, tracking, tree topology.

1 Introduction

Tubular structures with tree topologies are ubiquitous in human anatomy and can be found in the vascular system (arteries, veins, and capillaries), lung airways, renal tubules, and more. These structures are associated with many disease groups such as cardiovascular, pulmonary, and ophthalmological diseases, and tracking them in medical images aids early diagnosis and treatment planning [1], [2]. Producing an accurate topology of the tracked structure is critical for downstream tasks such as hemodynamics and blood flow modeling [3], interventional/preoperative planning [4], vascular morphometry [5] and assessment of vascular diseases such as atherosclerosis and stenosis [6]. There are many ways of representing tree structures but a centerline graph is generally preferred as it provides a concise and semantically rich representation. Tools capable of generating topologically correct centerlines are, therefore, of great interest to the medical community.

Manual extraction of these centerlines is too time-consuming, and model-

based methods [7], [8] suffer from poor generalizability and performance. Deep learning-based approaches have recently seen increased popularity and success. One common approach is to post-process a predicted semantic segmentation mask using thinning and skeletonization algorithms [9], [10] to produce the centerlines. Segmentation models generally aggregate image features locally to handle 3D medical images such as CT and MRI scans. Due to local aggregation, the long-range dependencies of these trees are difficult to capture, leading to disconnectivity issues in the segmentation mask. Obtaining centerlines from such segmentation masks could, therefore, result in incorrect topology, as a fully connected tree cannot be guaranteed. Some segmentation models [11], [12] utilize topology information by using topology-aware losses or graph priors. This leads to better connectivity but does not ensure a tree topology. CorSegRec [13] employs a complex three-stage pipeline to join the disconnected segments to the closest largest connected component using a regularized walk algorithm. However, it can make incorrect connections, especially when dealing with multiple disconnected components or more complex trees.

Another set of models iteratively tracks the centerline by leveraging the fact that the entire tree can be reached from its root. This procedure ensures that the resulting tree has the correct topology. One deep reinforcement learning (RL) based method [14] trains an agent to find centerline points, one action at a time. However, it cannot deal with bifurcations and does not predict important information such as the radius. Subsequent RL models [15], [16] mitigate these problems by utilizing an additional detector model. However, the first approach struggles to find termination points, while the second requires specific techniques to prevent backtracking and the repetitive tracing of identical branches, resulting in a more complex model pipeline.

Centerline tracking can be framed as a combination of an object detection problem where we predict centerline points as objects and an edge prediction problem where we predict edges between all possible point pairs. Recently, Relationformer [17] and Vesselformer [18] have utilized this framework to perform simultaneous prediction of vertices and edges of a centerline graph in a 3D volume. These models are based on DETR [19], an end-to-end transformer [20] based object detector, and utilize object queries to detect centerline points. In contrast to iterative models, these models have a simple pipeline and utilize object queries to detect multiple branches simultaneously. However, they do not enforce the correct topology, leading to disconnected centerline components.

Furthermore, for a complex tree with tortuous morphology, a large number of object queries are required to detect all the centerline points, significantly increasing model complexity. Lastly, Relationformer can only perform centerline detection on a small patch of the full volume. Vesselformer does the same but glues the output graph patches of the full volume together using custom heuristics in post-processing, which may introduce further topology errors.

Inspired by TrackFormer [21], a video multi-object tracker, we present Trexplorer, a novel recurrent DETR model with a simple pipeline that tracks the centerline graph of a tree structure in a complete volume, while also attributing important information like radius and class for each point of the centerline. In contrast to many existing methods, Trexplorer is guaranteed to generate a tree topology and does not require any post-processing steps to estimate the centerline tree. Trexplorer combines the simplicity of transformer-based models and the dynamic programming approach of breaking down the centerline tracking into simpler sub-problems of the iterative models. Given a 3D volume and the root position of a tree in the volume, our model detects centerline points level by level, while also taking care of any new branches that might start due to a bifurcation. We train Trexplorer in an end-to-end fashion and use it to estimate the full centerline tree graph.

Our contributions are summarized as follows. **(1)** We present a novel method for centerline tracking, which is guaranteed to yield a tree topology without any post-processing. **(2)** We successfully modify a video object tracking method to generate tree-structured graphs in medical volumes. **(3)** We evaluate Trexplorer on the publicly available synthetic vessel dataset [9] and demonstrate that it yields state-of-the-art performance.

2 Method

Our goal is to estimate a centerline tree. A tree is formally a graph (V, E) with nodes V and edges E. In our context, each node $\mathbf{v} \in V$ is a vector $\mathbf{v} = [\mathbf{x}, r]$ where $\mathbf{x} = [x, y, z]$ is the position of the node in our volume and r is the radius of the vessel at that point. To estimate the tree, we are given a 3D image, such as a CT or MRI scan, that contains the entire tree.

The basic idea in Trexplorer is to track every branch of the tree from the root node to the end of each branch, which means that we estimate the number of branches as well as the sequence of nodes along each branch. In each step

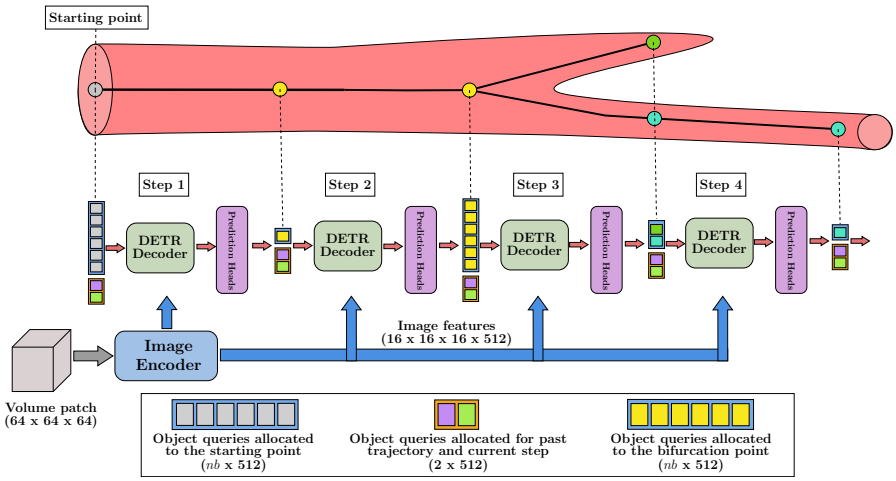


Figure 1: Trexplorer architecture unrolled over four steps. Starting and bifurcation points use the same number of object queries *nb*. The prediction heads act as a filter for background and end-points while allowing intermediate points through for further tracking. For a bifurcation point, *nb* copies of the object query are created for tracking new branches.

of the algorithm, we estimate the next position of each branch (approximately one voxel away) and classify each node as either an end node, an intermediate node, or a bifurcation node. If the predicted node is an end node we stop tracking that branch, if it is an intermediate node we continue tracking it, and, if it is a bifurcation point, we start tracking all the branches leaving the bifurcation point. Trexplorer is a Transformer-based model designed to solve this tracking problem in 3D volumes, see Fig. 1 for an illustration.

2.1 The Trexplorer Architecture

The Trexplorer architecture builds on the DETR model [19] and is inspired by the TrackFormer model [21]. The DETR model contains an encoder block that extracts features from the input image and a decoder block that outputs object detections in terms of class probabilities and an estimated bounding box for each potential object. The Trexplorer architecture is a modified version of the DETR model, which outputs class probabilities for the classes *end,*

intermediate, bifurcation, background, and estimates of the position of the next point and the radius of the vessel at that point.

Given the increased challenge of attention's quadratic complexity in a 3D space, both the image encoder and the transformer encoder in DETR have been replaced by a modified SwinUNETR [22] model. It utilizes windowed attention to efficiently create rich feature representations of an input volume. These features along with a set of object queries are used to compute cross-attention in the DETR decoder. To further reduce the number of tokens used in the cross-attention operation, the image features are extracted at $\frac{1}{4}$ of the initial resolution. Three MLPs are used as prediction heads to predict the position, radius, and class of each object query. See supplementary material for detailed architecture figures.

2.2 Object Queries and Bifurcations

The input object queries used in Trexplorer represent the previous state of branches that are currently tracked. Once updated through cross-attention with the image features, the updated object queries will represent the next state of those branches. As only a single object query is responsible for tracking a branch, it allows our model to predict very dense centerline trees with only a few object queries. The decision on how these updated queries are used is contingent on the classification head's output. The outputs of all object queries that are classified as intermediate points will be added as inputs to the next step. However, object queries marked as either an end-point or background are discarded, with the key distinction that the end-points are added to the global graph, marking where branches stop. If an object query is classified as a bifurcation, a fixed number nb of its copies are added to the next step's input, each with its own learned positional embedding. The positional embedding corresponds to the nb possible directions from a voxel in a 3D grid. Intuitively, it can be seen as if the bifurcating vessel ends, up to nb new vessels could begin from that location, each with its own direction. The value of nb should be set to at least the maximum expected bifurcation degree in the data to allow the model to track all new possible branches. The object queries that do not correspond to new vessels will be classified as background in the next step and are therefore tracked for only one step and never added to the global graph. The bifurcation object queries attend to each other through self-attention and are penalized for tracking the same branch in DETR's Hungarian loss, which

discourages overlapping centerlines.

2.3 Efficient Tracking Using Patches

As vessel tracking requires voxel-level image features, the number of tokens grows exponentially with the resolution of the input volume. This is an issue even considering the linear scaling of cross-attention and windowed attention. The most common strategy, also used in this work, is to train on patches taken from the complete volume. While this reduces the compute requirement significantly, it also introduces several new issues: Firstly, inference cannot simply be performed in a naive sliding-window fashion as this would create disconnected sub-trees in each patch, secondly, the vessel tracking in a patch does not have any information regarding the tree topology in the surrounding patches, and finally, the image features used as context for our object queries are now restricted to the patch.

In Trexplorer, we define a patch as a 64x64x64 cube, and vessels are tracked for ten steps starting from the central voxel. During inference, once tracking has finished in a patch, new starting points are created from the endpoints of the centerline graph of the current patch, and new patches centered around these endpoints are created. The model then tracks the centerline in each of the new patches independently. This ensures connectivity between patches and as the number of steps tracked is much smaller than the actual size of the patch, the context will always contain the previous tracked vessel. To further improve the performance when changing patches, a past trajectory token is added to the object queries. This token is obtained by embedding the past trajectory vector using an MLP. The vector is a concatenation of the position and radius from the past trajectory nodes in the previous patch. The model also uses a step token to get information about the current step which is a linear embedding of the current step number.

2.4 Loss Functions

Let $(\mathbf{p}, \hat{\mathbf{x}}, \hat{r})$ denote the predicted class probabilities, position, and radius for a single node, whereas c is the node class and $\mathbf{x}$ and r denote the position of and radius at the node if the class is not *background*. For the class probabilities,

we use a class-balanced multi-class focal loss [23],

$$\mathrm{FL}(\mathbf{p}, c) = -(1 - w_c)(1 - \mathbf{p}_c)^{\gamma} \log(\mathbf{p}_c), \tag{A.1}$$

where $\mathbf{p}_c$ is the c^{th} element in $\mathbf{p}$, $(1 - w_c)$ is a class balancing parameter and w_c represents the frequency of class c in our training data. The total loss for predicting $(\mathbf{p}, \hat{\mathbf{x}}, \hat{r})$ is

$$L = \frac{w_{cls}}{Q}\mathrm{FL}(\mathbf{p}, c) + \frac{I(c)}{N}\left(w_{\text{pos}}\|x - \hat{\mathbf{x}}\|_1 + w_{rad}|r - \hat{r}|\right), \tag{A.2}$$

where Q is number of allocated queries in the step, N is number of target points in the step, and w_{cls}, w_{pos}, w_{rad} are the weighting coefficients of the different losses and $I(c)$ is an indicator function that takes the value 1 unless $c =$background for which $I(c) = 0$.

After a bifurcation point, we will make multiple (Q) predictions of multiple (N) ground truth nodes. The association between predictions and ground truth nodes is generally unknown. To compute the loss function we follow the DETR model's Hungarian loss and solve an assignment problem where we minimize the total loss of all assignments. A prediction not assigned to a ground truth node is assumed to be background.

3 Experiments and Results

3.1 Dataset

We evaluate Trexplorer on the synthetic vessel dataset [9], which is the only publicly available vessel centerline dataset to the best of our knowledge. The synthetic vessel dataset contains 136 3D volumes of size (325 x 204 x 600). Each volume contains $\{min : 11, max : 21, mean : 16.1\}$ vessel trees. Each tree has a width of $\{min : 1, max : 97, mean : 19.6\}$ and the depth is $\{min : 34, max : 1319, mean : 438.7\}$. The bifurcation degree is always 2. The max radius of each vessel tree is $\{min : 11, max : 21, mean : 16.1\}$ voxels. The same training and test splits as reported by Vesselformer [18] are used, i.e. the first 40 volumes for training and validation, and the next 10 volumes for testing.

3.2 Experiments

Trexplorer is trained on the synthetic dataset from scratch. We train the model for 240,000 iterations with a batch size of 8. The number of tokens allocated to a bifurcation point nb is set to 2, while the max number of concurrently tracking tokens mq is set to 10. The models have been implemented using the open-source libraries Pytorch and MONAI, and are trained using 4 A100 GPUs on a single node with mixed precision enabled.

3.3 Results

We report the same metrics as reported by Vesselformer for a fair comparison. The reported metrics include the Street Mover Distance (SMD) which is the graph Wasserstein distance, the relative error in % of Betti-0 (the number of connected components) and Betti-1 (the number of cycles), mean average precision (mAP) and mean average recall (mAR) for both nodes and edges, and the mean absolute error for the predicted radius. We also include the results obtained by Voreen [24], an open-source framework for the analysis of multi-modal volumetric data. The results for Voreen and Vesselformer are taken from results reported by Vesselformer's authors.

As shown in Table 1, Trexplorer has the lowest SMD score which is a graph similarity score, showcasing our model's great performance on whole graph tracking. Both Betti-0 and Betti-1 topology errors are zero for Trexplorer, due to its constrained topology output. Our model has lower node and edge mAP compared to Vesselformer. However, Vesselformer is trained and tested on a pruned version of the centerlines, where nodes of degree 2 with neighboring edges forming angles larger than 160 degrees are removed, and the neighboring nodes are connected. In contrast, Trexplorer is trained on dense centerlines with node spacing of around 1 voxel. This results in a predicted centerline graph with many nodes for a single volume. To be able to compute matching-based mAP and mAR scores in a reasonable time, we prune our predicted and ground truth vessel centerlines in the same manner as described above. This may result in the pruning of true positives, leading to lower scores. Another reason for a lower mAP is that in the synthetic dataset, some end-points of a vessel tree can be connected or be very close to a different vessel tree, and upon reaching those end-points, the model starts tracking this connected tree, resulting in extra false positives.

Table 1: Comparison of Trexplorer with Vesselformer and Voreen. Some of the results are missing for Voreen due to instability in metric computation.

Model	SMD ↓	%-Betti Error ↓		Node ↑		Edge ↑		MAE ↓
		Betti-0	Betti-1	mAP	mAR	mAP	mAR	(radius)
Voreen	0.03071	0.2955	0.2766	36.17	43.35	*	*	1.79
Vesselformer	0.01381	0.2188	0.2054	**72.32**	**80.11**	**72.19**	76.24	**0.52**
Trexplorer	**0.0075**	**0.0000**	**0.0000**	60.88	77.88	59.74	**82.45**	0.74

We compare the outputs of Trexplorer and Vesselformer with the ground truth centerline graph in Figure 2. The examples show that Trexplorer can predict complicated vessel trees accurately with correct topology while Vesselformer struggles to get the topology right and even misses some branches altogether. Vesselformer predicts sparse centerline graphs and for the sake of this comparison, we add extra nodes between the connected edges to make it dense.

3.4 Ablations

In this section, we ablate two components that are added to the DETR model to help it with the recurrent flow of information, namely the past trajectory token and the step token. For the past trajectory, we change the number of previous positions and the radius. For this ablation study, we use a patch-wise mean average precision (mAP) and bipartite matched F1-score (BP-F1), evaluated on 640 validation patches. The results are shown in Table 2. Using the past trajectory token results in a significant performance boost as it provides the decoder with past context and helps it determine which way to go next. Adding more past positions, corresponding radii, and the step token leads to small improvements.

Table 2: Ablation experiments for examining the past trajectory and step token.

Index	Past trajectory token		Step Token	BP-F1	mAP
	Num. Prev. Pos.	Radius			
1	0	✗	✓	0.900	0.851
2	5	✗	✓	0.944	0.900
3	5	✗	✗	0.941	0.900
4	10	✓	✓	**0.946**	**0.901**

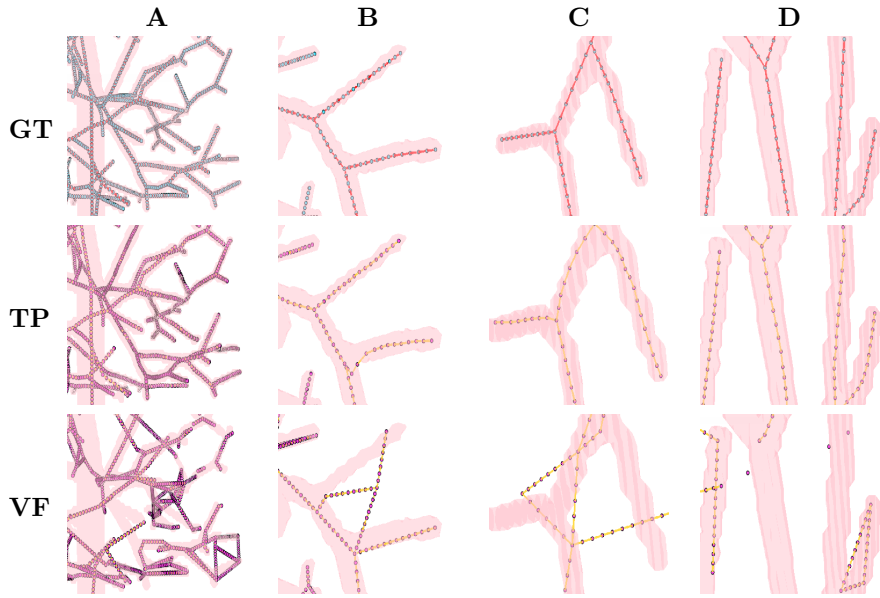


Figure 2: Visual comparison between ground truth (GT), Trexplorer (TP), and Vesselformer (VF) centerlines using four examples patches (A, B, C, D) from the synthetic vessel dataset.

4 Conclusion

We propose a novel recurrent DETR model and demonstrate that our model can effectively track centerline graphs with correct topology using a simple pipeline. Trexplorer does not require post-processing and can produce a vessel tree for huge volumes by processing only relevant patches. The results show that it performs significantly better than the state-of-the-art model on the graph and topology-related metrics while performing comparably on other detection-based metrics. Although the results are impressive, our model has limitations such as possible premature tracking termination and duplicate tracking. Future works can potentially address these limitations by utilizing DETR variants with stronger priors and advanced tree-matching algorithms.

Acknowledgements. Compute and storage resources were provided by NAISS, partially funded by the Swedish Research Council (grant 2022-06725).

Supplementary Material

A Detailed Architecture

A.1 Image Encoder

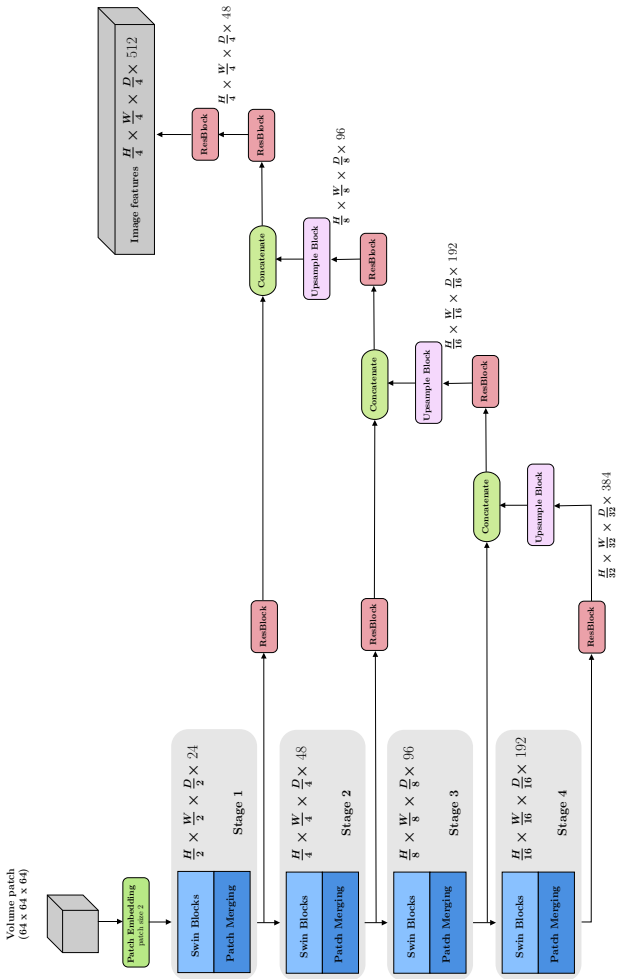


Figure 3: Detailed architecture of the image encoder.

A.2 DETR Decoder

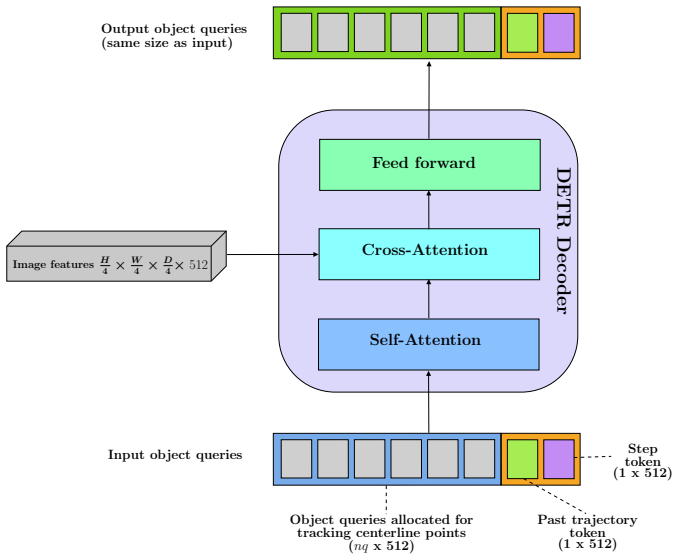


Figure 4: Detailed architecture of the DETR decoder. Here nq is the number of object queries which is set dynamically depending on the number of branches being tracked.

A.3 Prediction Heads

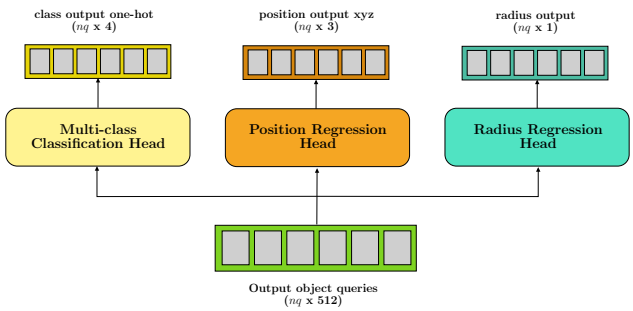


Figure 5: Detailed architecture of the prediction heads.

References

[1] S. Moccia, E. De Momi, S. El Hadji, and L. S. Mattos, "Blood vessel segmentation algorithms – review of methods, datasets and evaluation metrics," *Computer Methods and Programs in Biomedicine*, vol. 158, pp. 71–91, 2018.

[2] G. Cheng, X. Wu, W. Xiang, C. Guo, H. Ji, and L. He, "Segmentation of the airway tree from chest CT using tiny atrous convolutional network," *IEEE Access*, vol. 9, pp. 33 583–33 594, 2021.

[3] O. Miraucourt, S. Salmon, M. Szopos, and M. Thiriet, "Blood flow in the cerebral venous system: Modeling and simulation," *Computer Methods in Biomechanics and Biomedical Engineering*, vol. 20, no. 5, pp. 471–482, 2017.

[4] D. Huang, W. Tang, Y. Ding, T. Wan, and Y. Chen, "An interactive 3D preoperative planning and training system for minimally invasive vascular surgery," in *2011 12th International Conference on Computer-Aided Design and Computer Graphics*, IEEE, 2011, pp. 443–449.

[5] Z. Khan et al., "Three-dimensional morphometric analysis of the renal vasculature," *American Journal of Physiology – Renal Physiology*, vol. 314, no. 5, F715–F725, 2018.

[6] A. D. Choi et al., "CT evaluation by artificial intelligence for atherosclerosis, stenosis and vascular morphology (CLARIFY): A multi-center, international study," *Journal of Cardiovascular Computed Tomography*, vol. 15, no. 6, pp. 470–476, 2021.

[7] Z. Zhang, D. Marin, E. Chesakov, M. M. Maza, M. Drangova, and Y. Boykov, "Divergence prior and vessel-tree reconstruction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 10 216–10 224.

[8] Z. Zhang, D. Marin, M. Drangova, and Y. Boykov, "Confluent vessel trees with accurate bifurcations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 9573–9582.

[9] G. Tetteh et al., "DeepVesselNet: Vessel segmentation, centerline prediction, and bifurcation detection in 3-D angiographic volumes," *Frontiers in Neuroscience*, vol. 14, p. 1285, 2020.

[10] L. Chen et al., "Deep open snake tracker for vessel tracing," in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2021*, Springer, 2021, pp. 579–589.

[11] D. Keshwani, Y. Kitamura, S. Ihara, S. Iizuka, and E. Simo-Serra, "TopNet: Topology preserving metric learning for vessel tree reconstruction and labelling," in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2020*, Springer, 2020, pp. 14–23.

[12] Z. Tan, J. Feng, and J. Zhou, "SGNet: Structure-aware graph-based network for airway semantic segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2021, pp. 153–163.

[13] Y. Qiu et al., "CorSegRec: A topology-preserving scheme for extracting fully-connected coronary arteries from CT angiography," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2023, pp. 670–680.

[14] P. Zhang, F. Wang, and Y. Zheng, "Deep reinforcement learning for vessel centerline tracing in multi-modality 3D volumes," in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2018*, Springer, 2018, pp. 755–763.

[15] Y. Zhang, G. Luo, W. Wang, and K. Wang, "Branch-aware double DQN for centerline extraction in coronary CT angiography," in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2020*, Springer, 2020, pp. 35–44.

[16] Z. Li, Q. Xia, Z. Hu, W. Wang, L. Xu, and S. Zhang, "A deep reinforced tree-traversal agent for coronary artery centerline extraction," in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2021*, Springer, 2021, pp. 418–428.

[17] S. Shit et al., "Relationformer: A unified framework for image-to-graph generation," in *European Conference on Computer Vision*, Springer, 2022, pp. 422–439.

[18] C. Prabhakar et al., "Vesselformer: Towards complete 3D vessel graph generation from images," in *Medical Imaging with Deep Learning*, PMLR, 2024, pp. 320–331.

[19] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European Conference on Computer Vision*, Springer, 2020, pp. 213–229.

[20] A. Vaswani et al., "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[21] T. Meinhardt, A. Kirillov, L. Leal-Taixe, and C. Feichtenhofer, "Trackformer: Multi-object tracking with transformers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 8844–8854.

[22] Y. Tang et al., "Self-supervised pre-training of Swin transformers for 3D medical image analysis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 20 730–20 740.

[23] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2980–2988.

[24] J. Meyer-Spradow, T. Ropinski, J. Mensmann, and K. Hinrichs, "Voreen: A rapid-prototyping environment for ray-casting-based volume visualizations," *IEEE Computer Graphics and Applications*, vol. 29, no. 6, pp. 6–13, 2009.

PAPER B

Trexplorer Super: Topologically Correct Centerline Tree Tracking of Tubular Objects in CT Volumes

Roman Naeem, David Hagerman, Jennifer Alvén, Lennart Svensson, Fredrik Kahl

Published in Proceedings of International Conference on Medical Image Computing and Computer-Assisted Intervention, pp. 595–605, Springer, 2025

The layout has been revised.

Abstract

Tubular tree structures, such as blood vessels and airways, are essential in human anatomy, and accurately tracking them while preserving their topology is crucial for various downstream tasks. Trexplorer is a recurrent model designed for centerline tracking in 3D medical images, but it is prone to predicting duplicate branches and terminating tracking prematurely. To address these issues, we present Trexplorer Super, an enhanced version that substantially improves performance through several novel advancements. Evaluating centerline tracking models is challenging due to the lack of public benchmark datasets. To enable thorough evaluation, we develop three centerline datasets, one synthetic and two real, each with increasing difficulty. Using these datasets, we perform a comprehensive comparison of existing state-of-the-art (SOTA) models with our approach. Trexplorer Super outperforms previous SOTA models on every dataset. Our results also highlight that strong performance on synthetic data does not necessarily translate to real datasets. The code and datasets are available at https://github.com/RomStriker/Trexplorer-Super

Keywords: Centerline tracking, tubular structures, tree topology.

1 Introduction

Tubular tree structures in the vascular and respiratory systems play a critical role in transporting essential substances throughout the body. Accurately tracking the centerlines of these structures in medical images is fundamental for early diagnosis, treatment, and various downstream tasks [1], [2], [3], [4]. In this paper, we introduce a new method for centerline tree tracking and propose a comprehensive framework for its evaluation.

Several existing approaches address the challenge of centerline extraction, but each comes with its own limitations. A common method segments the image and then applies skeletonization [5], but such models struggle with

long-range dependencies, leading to connectivity issues. Other models [6], [7] detect centerline nodes and edges in a two-step process but also suffer from connectivity errors. Recurrent models, such as reinforcement learning-based methods [8], [9], iteratively track centerlines but rely on complex pipelines. Trexplorer [10] simplifies this with a DETR-based transformer [11] that uses breadth-first tracking to ensure correct topology. However, it struggles with duplicate branch detections and premature tracking terminations.

To overcome the limitations of existing centerline tracking methods, we propose Trexplorer Super, which builds on the Trexplorer model with several key enhancements to improve accuracy, robustness, and completeness. Our method reduces premature terminations and duplicate branches while improving new branch detection and preserving fine spatial details in image features. To ensure more consistent centerline extraction, we introduce Super Trajectory Training, a strategy that retains and reuses tracking information across multiple steps. We also refine feature representation with Focal Cross Attention, which selectively attends to a focal region in high-resolution image features while maintaining broader contextual awareness. To further enhance robustness, we employ Target Augmentation, a strategy that improves bifurcation and new branch detection while minimizing duplicate branches. These advancements contribute to a more reliable and comprehensive centerline tracking framework.

Evaluating centerline tracking in 3D medical images is challenging due to the lack of publicly available real datasets. Existing synthetic datasets have topological limitations, and strong performance on these does not generalize well to real data. To address this, we create one synthetic and two real datasets and establish a comprehensive baseline by evaluating prior SOTA models alongside our approach using point-, branch-, and tree-level metrics.

Our key contributions include: **(1)** enhancing the Trexplorer framework with novel techniques, namely Super Trajectory Training, Focal Cross Attention, and Target Augmentation; and **(2)** creating three datasets and thoroughly evaluating the previous SOTA models alongside our method.

2 Method

Our goal is to estimate the centerline tree from a given CT volume and a starting root point. The centerline tree is represented as a graph (V, E) with V nodes and E edges. Each node $\mathbf{v} \in V$ is defined as a vector $\mathbf{v} = [x, y, z, r]$,

representing the 3D position and radius of a centerline point, while an edge $\mathbf{e} \in E$ represents a connection between two nodes.

Trexplorer Super is a DETR-based model that uses object queries to track branches. It begins at the root point and tracks each branch to its endpoint. The model estimates the total number of branches and the number of nodes within each branch. Tracking follows a sequential, breadth-first approach in which, at each step, all child nodes at the next graph level are predicted. Each child node is approximately one voxel away from its parent and is classified into one of three categories: end node, intermediate node, or bifurcation node. The model stops tracking a branch when it predicts an end node. If the node is intermediate, it continues tracking subsequent nodes. For bifurcation nodes, it halts tracking of the current branch and assigns a new set of 26 object queries, some of which start tracking the new branches, while the rest are discarded. Trexplorer Super builds on Trexplorer by incorporating the following novel key components. The model architecture is shown in Figure 1.

2.1 Super Trajectory Training

Trexplorer generates a centerline graph over 9 steps, starting from the center of a volume patch. New patches are created at the last step nodes of the tracked graph, which are used by the model, along with the past trajectory, to continue tracking until all endpoints of the tree are reached. The past trajectory token, an embedding of up to 10 previous node positions, is the only source of past information when tracking in a new patch. This simplifies the formation of training batches, as each batch is independent. However, it results in the loss of the learned past trajectory embedding stored in the object queries, causing premature branch terminations and missed branches, leading to a lower recall.

To better utilize past trajectory information, we propose Super Trajectory Training (STT). In STT, each sample consists of a super trajectory of size 54, divided into six sub-trajectories of size 10 with shared link nodes. Each sub-trajectory is paired with a volume patch centered at its starting node, as shown in Figure 1(a). Secondary branches may appear within a sub-trajectory, but they are only tracked within the corresponding patch during training.

During training, object query outputs from the previous sub-trajectory are used as inputs for the next sub-trajectory, along with new volume patches. This preserves the valuable past trajectory information embedded in the object queries. In contrast, Trexplorer's training strategy is analogous to training

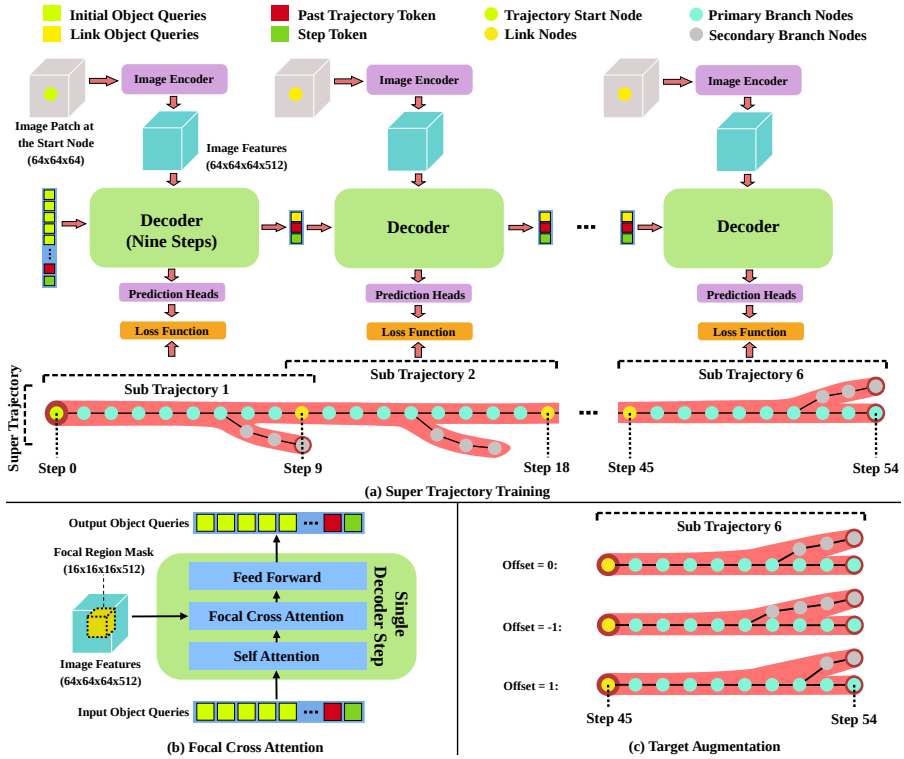


Figure 1: Overview of the Trexplorer Super architecture, illustrating: (a) Super Trajectory Training, (b) Focal Cross Attention, and (c) Target Augmentation.

on a single sub-trajectory. The first sub-trajectory still relies on the past trajectory token. During inference, tracking begins at the root point, and for subsequent patches, the object query outputs from previous patches are used. This forms a continuous chain from the root to all endpoints, enabling more effective tracking.

2.2 Focal Cross Attention

In Trexplorer, object queries track branches by aggregating information from image features using the transformer's cross-attention module [12]. For tubular structures like vessel trees, capturing long-range dependencies and

fine-grained spatial details is crucial for accurately locating thin, elongated branches. However, using large, high-resolution features quickly becomes computationally infeasible for 3D medical images. Some methods address this by using learned sparse attention [13], [14], [15], but they rely on object queries to determine which features to attend to. This conflicts with Trexplorer's use of object queries to store branch tracking history, leading to poor performance.

Trexplorer Super introduces Focal Cross Attention (FCA), which extracts high-resolution features over a large region but restricts cross-attention to the small focal region where branches are being tracked, as shown in Figure 1(b). By training the model end-to-end, the responsibility for aggregating long-range dependencies while maintaining fine-grained spatial details is delegated to the feature extractor. This design allows the decoder object queries to focus on retaining tracking history while cross-attending to a smaller, more relevant set of features.

2.3 Target Augmentation

As the radius of a bifurcation node increases, the area of viable positions for the bifurcation also increases, rather than being a single fixed point. Trexplorer Super is trained to account for this positional ambiguity using Target Augmentation. During target augmentation, for each bifurcation point in the primary branch of a super trajectory, an offset is sampled from a Laplace distribution with mean $\mu = 0$ and scale b proportional to the bifurcation radius. This offset shifts secondary branches up or down along the primary branch, effectively generating viable augmented targets as shown in Figure 1(c). The branches around these new bifurcation points are smoothed to preserve natural trajectories.

Training with augmented targets encourages the model to consider more nodes as potential bifurcations, improving the detection of new branches. This strategy also reduces duplicates, leading to fewer volume patches to process and eliminating the need for post-processing. The reduction in duplicates can be attributed to the combined effect of Super Trajectory Training and Target Augmentation, which greatly enhances the effect of the Hungarian loss and self-attention between object queries, two key components used for preventing duplicates in Trexplorer. Additionally, a small amount of Gaussian noise ($\mu = 0, \sigma = 0.025$) is added to all points to further improve robustness.

Table 1: Statistics of the ground truth centerline graphs for the provided datasets.

Dataset	Samples			Max Node	Average			Radius		
	train	val	test	Degree	points	depth	width	max	mean	min
Synthetic Dataset	368	32	100	4	2 763	343	20	17	5.37	2
ATM'22	220	16	60	4	7 398	504	55	23	2.47	1
Parse 2022	72	8	20	4	19 644	498	126	38	2.55	1

3 Experiments and Results

3.1 Datasets

To our knowledge, the only publicly available 3D centerline tree dataset is the synthetic vessel tree dataset [5]. However, these trees were generated without collision avoidance, resulting in unrealistic vessel intersections. To address this issue, we use the Synthetic Vascular Toolkit (SVT) [16], [17] to generate a new synthetic tree dataset with collision avoidance. We follow the same procedure as [5] to create the corresponding images. This dataset serves as a useful toy example for model research and development.

For a comprehensive evaluation, we also generate centerline ground truth from two publicly available real tubular tree segmentation datasets: ATM'22 dataset [18], [19], [20], [21], [22] (airway segmentation), licensed under CC BY-NC and Parse 2022 dataset [23] (pulmonary artery segmentation), licensed under CC BY-NC-ND 6.0. The use of these datasets complies with the terms set by the dataset owners. Before extracting the ground truth, we resample the data volumes to 0.5 mm isotropic resolution. The Vascular Modeling Toolkit (VMTK) [24], [25] is used to extract root points from segmentation masks, which Kimimaro [26] then uses to trace the tubular tree centerlines. Further dataset statistics are provided in Table 1.

3.2 Evaluation Metrics

We evaluate predictions at the point, branch, and tree levels to capture different aspects of accuracy. At the point level, we use Precision, Recall, and F1-score. A predicted node is considered a True Positive (TP) if a ground truth node exists within its 1.5-voxel radius that has not been matched to another prediction; otherwise, it is a False Positive (FP). An unmatched ground truth

node is considered a False Negative (FN). We also assess radius accuracy using Mean Absolute Error (MAE). Given the large number of nodes, we avoid metrics that require solving the Linear Assignment Problem.

At the branch level, we again use Precision, Recall, and F1-score while treating branches as objects. A predicted branch is considered a TP if it correctly matches at least 80% of the points in a ground truth branch within a 1.5-voxels radius, provided that the ground truth branch has not already been matched. Otherwise, it is an FP. Unmatched ground truth branches are considered FNs. To evaluate the overall graph structure, we use topological metrics, specifically, the MAE of Betti-0 (connected components) and Betti-1 (cycles).

3.3 Experiments

We evaluate two previous state-of-the-art (SOTA) models, Vesselformer and Trexplorer, alongside our proposed method, Trexplorer Super, on three datasets. All models were trained on a single node with four A100 GPUs, with Trexplorer and Trexplorer Super using mixed precision for improved efficiency. To ensure a fair comparison focused on model improvements, Trexplorer and Trexplorer Super share nearly identical hyperparameters: allocating 26 tokens for a bifurcation node, and a maximum of 196 tokens. Both models were trained for approximately 2 million iterations. Vesselformer produced the best results for author-optimized hyperparameters, with 80 object tokens and approximately 12 million training iterations. Each model was trained five times per dataset, and we report the mean and standard deviation for each metric.

3.4 Results

Tables 2 and 3 summarize the performance of the evaluated models on the proposed datasets, while Figure 2 provides a visual comparison using one sample from each dataset. On the synthetic dataset, Vesselformer achieves a higher F1-score than Trexplorer. Although Trexplorer has better recall, it generates many duplicate branches, leading to low precision. Trexplorer Super improves recall over Vesselformer, though still lower than Trexplorer, while drastically reducing duplicates, resulting in the highest overall F1-score. On ATM'22, Vesselformer retains some centerline tracking ability but misses

Table 2: Comparison of different models using point-level metrics on the Synthetic, ATM'22, and Parse 2022 datasets.

Model	Point-level			
	Precision(%)↑	Recall(%)↑	F1(%)↑	Radius (MAE)↓
Synthetic Dataset				
Vesselformer	44.53 ± 7.87	61.52 ± 1.14	48.18 ± 5.62	0.4244 ± 0.0134
Trexplorer	30.91 ± 9.45	**78.21 ± 4.13**	39.40 ± 8.62	0.2263 ± 0.0323
Trexplorer Super	**91.91 ± 3.28**	70.44 ± 3.02	**77.83 ± 1.89**	**0.0955 ± 0.0061**
ATM'22 Dataset				
Vesselformer	22.32 ± 1.35	34.37 ± 0.94	26.77 ± 1.27	0.7908 ± 0.0095
Trexplorer	3.20 ± 0.73	4.33 ± 0.63	3.34 ± 0.30	0.9744 ± 0.0844
Trexplorer Super	**67.51 ± 1.35**	**60.65 ± 2.01**	**60.45 ± 1.03**	**0.3925 ± 0.0241**
Parse 2022 Dataset				
Vesselformer	18.49 ± 1.84	15.28 ± 0.83	16.43 ± 0.78	1.1144 ± 0.0269
Trexplorer	9.87 ± 3.76	12.01 ± 7.46	10.01 ± 4.98	1.2108 ± 0.3042
Trexplorer Super	**55.27 ± 3.00**	**33.99 ± 3.34**	**39.46 ± 1.93**	**0.5627 ± 0.0141**

Table 3: Comparison of different models using branch-level and graph-level metrics on the Synthetic, ATM'22, and Parse 2022 datasets.

Model	Branch-level			Graph-level (MAE)	
	Precision(%)↑	Recall(%)↑	F1(%)↑	Betti-0↓	Betti-1↓
Synthetic Dataset					
Vesselformer	12.51 ± 0.64	27.08 ± 2.85	15.95 ± 0.36	81.7 ± 16.8	653.5 ± 138.7
Trexplorer	19.20 ± 7.31	64.91 ± 3.68	26.26 ± 7.18	**0.000 ± 0.0**	**0.000 ± 0.0**
Trexplorer Super	**96.01 ± 4.42**	**67.52 ± 2.94**	**77.12 ± 1.59**	**0.000 ± 0.0**	**0.000 ± 0.0**
ATM'22 Dataset					
Vesselformer	1.35 ± 0.19	3.67 ± 0.41	1.95 ± 0.25	312.5 ± 25.1	180.4 ± 35.9
Trexplorer	0.03 ± 0.01	0.14 ± 0.04	0.05 ± 0.02	**0.00 ± 0.0**	**0.00 ± 0.0**
Trexplorer Super	**45.15 ± 2.39**	**42.23 ± 1.54**	**41.15 ± 1.28**	**0.00 ± 0.0**	**0.00 ± 0.0**
Parse 2022 Dataset					
Vesselformer	2.32 ± 0.20	1.89 ± 0.18	1.99 ± 0.16	410.1 ± 23.9	246.7 ± 78.1
Trexplorer	3.40 ± 1.41	5.36 ± 3.50	3.71 ± 1.91	**0.00 ± 0.0**	**0.00 ± 0.0**
Trexplorer Super	**35.45 ± 2.89**	**20.09 ± 1.86**	**23.46 ± 1.09**	**0.00 ± 0.0**	**0.00 ± 0.0**

branches and predicts duplicates, resulting in a low overall score. Trexplorer performs worse, barely tracking any branches due to limited access to past

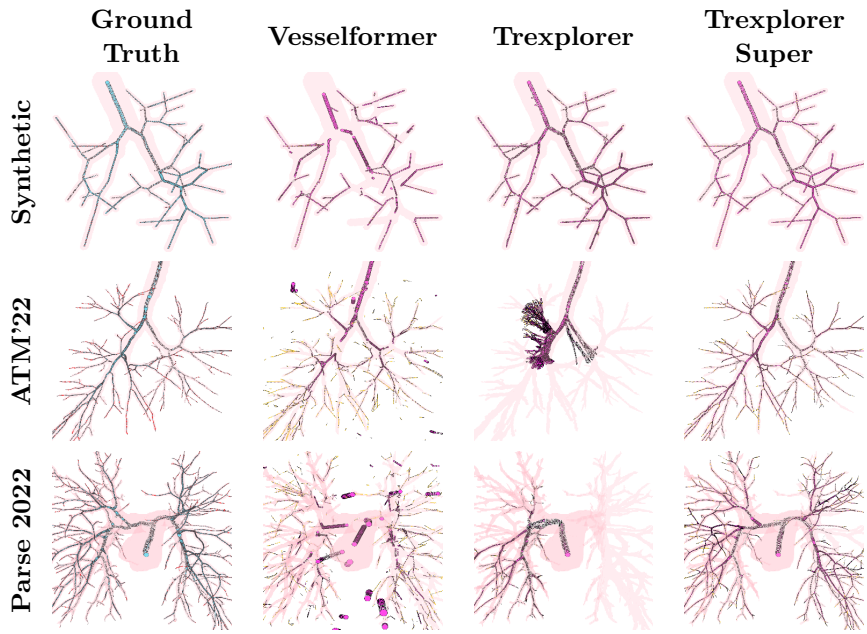


Figure 2: Visual comparison between the ground truth and the predictions from Vesselformer, Trexplorer, and Trexplorer Super on one sample from each dataset. Centerline marker size is proportional to the radius at each node.

trajectory information. Trexplorer Super, with its enhancements, delivers strong improvements across all metrics. Parse 2022 presents a greater challenge due to denser trees and weaker vessel signals. Both Vesselformer and Trexplorer struggle, but Trexplorer Super clearly outperforms them.

Trexplorer Super achieves the lowest radius MAE for all datasets. At branch level, it also performs the best across all metrics. Both Trexplorer and Trexplorer Super ensure topological correctness with zero Betti-0 and Betti-1 errors, whereas Vesselformer struggles, predicting multiple disconnected components and cycles.

The performance of Trexplorer Super is impacted by certain cases where it fails to track the centerline, likely due to unseen intensity variations in the input volumes. Applying image augmentation could help address this issue and further improve performance.

Table 4: Evaluation of the novel key components, Super Trajectory Training (STT), Focal Cross Attention (FCA), and Target Augmentation (TA) incorporated in the Trexplorer framework by Trexplorer Super for the ATM'22 Dataset.

Index	Novel Key Components			Point-level		
	STT	FCA	TA	Precision(%)↑	Recall(%)↑	F1(%)↑
1				3.07 ± 0.08	3.59 ± 0.21	3.17 ± 0.11
2		✓		36.57 ± 54.94	3.66 ± 3.20	2.56 ± 2.19
3	✓			44.87 ± 10.20	27.43 ± 7.69	33.01 ± 8.70
4	✓	✓		61.76 ± 2.23	54.72 ± 7.45	53.91 ± 5.05
5	✓	✓	✓	**67.82 ± 1.04**	**61.20 ± 2.50**	**60.66 ± 0.64**

3.5 Ablations

We conduct an ablation study on the ATM'22 dataset to evaluate the impact of our key modifications to Trexplorer: Super Trajectory Training, Focal Cross Attention, and Target Augmentation. Table 4 reports the mean and standard deviation of point-level metrics, including precision, recall, and F1-score, averaged over three runs for each ablation. The results highlight Super Trajectory Training as the most critical improvement, allowing Trexplorer to perform effectively on real data. Focal Cross Attention further enhances performance by enabling the feature extractor to condense relevant information in the focal region. Target Augmentation improves both recall and precision by reducing duplicate predictions and enhancing the detection of new branches, leading to more complete branch reconstructions.

4 Conclusion

We present major improvements to Trexplorer, enhancing its accuracy, robustness, and completeness. Additionally, we introduce three new datasets and conduct a comprehensive evaluation, demonstrating that our model outperforms two state-of-the-art baselines. While the results are promising, challenges remain, particularly on the Parse 2022 dataset. Future work includes leveraging a larger pretrained feature extractor and integrating advanced DETR variants or recurrent architectures, such as LSTMs, for further refinement.

Acknowledgements. Compute and storage resources were provided by NAISS

(grant 2022-06725, Swedish Research Council) and the Berzelius system at the National Supercomputer Centre, funded by the Knut and Alice Wallenberg Foundation.

References

[1] O. Miraucourt, S. Salmon, M. Szopos, and M. Thiriet, "Blood flow in the cerebral venous system: Modeling and simulation," *Computer Methods in Biomechanics and Biomedical Engineering*, vol. 20, no. 5, pp. 471–482, 2017.

[2] D. Huang, W. Tang, Y. Ding, T. Wan, and Y. Chen, "An interactive 3D preoperative planning and training system for minimally invasive vascular surgery," in *2011 12th International Conference on Computer-Aided Design and Computer Graphics*, IEEE, 2011, pp. 443–449.

[3] Z. Khan et al., "Three-dimensional morphometric analysis of the renal vasculature," *American Journal of Physiology – Renal Physiology*, vol. 314, no. 5, F715–F725, 2018.

[4] A. D. Choi et al., "CT evaluation by artificial intelligence for atherosclerosis, stenosis and vascular morphology (CLARIFY): A multi-center, international study," *Journal of Cardiovascular Computed Tomography*, vol. 15, no. 6, pp. 470–476, 2021.

[5] G. Tetteh et al., "DeepVesselNet: Vessel segmentation, centerline prediction, and bifurcation detection in 3-D angiographic volumes," *Frontiers in Neuroscience*, vol. 14, p. 1285, 2020.

[6] C. Prabhakar et al., "Vesselformer: Towards complete 3D vessel graph generation from images," in *Medical Imaging with Deep Learning*, PMLR, 2024, pp. 320–331.

[7] S. Shit et al., "Relationformer: A unified framework for image-to-graph generation," in *European Conference on Computer Vision*, Springer, 2022, pp. 422–439.

[8] Y. Zhang, G. Luo, W. Wang, and K. Wang, "Branch-aware double DQN for centerline extraction in coronary CT angiography," in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2020*, Springer, 2020, pp. 35–44.

[9] Z. Li, Q. Xia, Z. Hu, W. Wang, L. Xu, and S. Zhang, "A deep reinforced tree-traversal agent for coronary artery centerline extraction," in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2021*, Springer, 2021, pp. 418–428.

[10] R. Naeem, D. Hagerman, L. Svensson, and F. Kahl, "Trexplorer: Recurrent DETR for topologically correct tree centerline tracking," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2024, pp. 744–754.

[11] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European Conference on Computer Vision*, Springer, 2020, pp. 213–229.

[12] A. Vaswani et al., "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[13] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "Deformable DETR: Deformable transformers for end-to-end object detection," *arXiv preprint arXiv:2010.04159*, 2020.

[14] B. Roh, J. Shin, W. Shin, and S. Kim, "Sparse DETR: Efficient end-to-end object detection with learnable sparsity," *arXiv preprint arXiv:2111.14330*, 2021.

[15] D. Zheng, W. Dong, H. Hu, X. Chen, and Y. Wang, *Less is more: Focus attention for efficient DETR*, 2023.

[16] Z. A. Sexton et al., "Rapid model-guided design of organ-scale synthetic vasculature for biomanufacturing," *ArXiv*, arXiv–2308, 2023.

[17] Z. A. Sexton, *Synthetic vascular toolkit*, Accessed: 2025-02-24, 2023.

[18] M. Zhang et al., "Multi-site, multi-domain airway tree modeling," *Medical Image Analysis*, vol. 90, p. 102 957, 2023.

[19] M. Zhang, H. Zhang, G.-Z. Yang, and Y. Gu, "CFDA: Collaborative feature disentanglement and augmentation for pulmonary airway tree modeling of COVID-19 CTs," in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2022*, Springer, 2022, pp. 506–516.

[20] H. Zheng et al., "Alleviating class-wise gradient imbalance for pulmonary airway segmentation," *IEEE Transactions on Medical Imaging*, vol. 40, no. 9, pp. 2452–2462, 2021.

[21] W. Yu, H. Zheng, M. Zhang, H. Zhang, J. Sun, and J. Yang, "BREAK: Bronchi reconstruction by geodesic transformation and skeleton embedding," in *2022 IEEE 19th International Symposium on Biomedical Imaging (ISBI)*, IEEE, 2022, pp. 1–5.

[22] Y. Qin et al., "AirwayNet: A voxel-connectivity aware approach for accurate airway segmentation using convolutional neural networks," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2019, pp. 212–220.

[23] G. Luo et al., "Efficient automatic segmentation for multi-level pulmonary arteries: The PARSE challenge," *arXiv:2304.03708*, 2023.

[24] L. Antiga and D. A. Steinman, *VMTK: The vascular modeling toolkit*, Accessed: 2025-02-24.

[25] Slicer Development Community, *3D Slicer*, Accessed: 2025-02-24, 2025.

[26] W. Silversmith, J. A. Bae, P. H. Li, and A. Wilson, *Kimimaro: Skeletonize densely labeled 3D image segmentations*, version 3.0.0, Accessed: 2021-09-29, 2021.

PAPER C

RefTr: Recurrent Refinement of Confluent Trajectories for 3D Tubular Tree Centerlines

Roman Naeem, David Hagerman, Jennifer Alvén, Fredrik Kahl

Under Review

The layout has been revised.

Abstract

Tubular tree structures such as blood vessels and lung airways are central to many clinical tasks, including diagnosis, treatment planning, and surgical navigation. Accurate centerline extraction with correct topology is essential, as missing small branches can lead to incomplete assessments or overlooked abnormalities. We propose RefTr, a 3D image-to-graph framework that generates tubular tree centerlines via recurrent refinement of confluent trajectories. RefTr adopts a Transformer-based Producer–Refiner architecture in which the Producer predicts candidate trajectories and a shared Refiner iteratively aligns with target branches. The confluent trajectory representation enables whole-branch refinement while explicitly enforcing valid topology. This recurrent scheme improves precision and reduces decoder parameters by 2.4× compared to the state-of-the-art. We further introduce an efficient non-maximum suppression algorithm for spatial tree graphs to merge duplicate branches and extend evaluation metrics to be radius-aware for robust comparison. Experiments on multiple public datasets demonstrate higher recall and precision, faster inference, and substantially fewer parameters, highlighting the effectiveness of RefTr for 3D tubular tree analysis.

Keywords: Image-to-graph, centerline Extraction, tree topology.

1 Introduction

Tubular trees such as blood vessels and lung airways are essential for material transport in the human body, and abnormalities in these networks are linked to many diseases. Accurate, topologically correct centerline extraction is essential for diagnosis and treatment planning [1], [2], surgical navigation [3], hemodynamic analysis [4], and vascular morphometry [5], [6]. Missing small branches can lead to incomplete assessments and clinical risk, making high recall crucial. Centerline graphs provide a compact, interpretable representation,

easier to annotate than dense masks and amenable to enforcing valid tree topology.

Segmentation-based methods [7], [8] require costly voxel-level labels, post-processing skeletonization, and lack end-to-end graph prediction. Graph-based methods [9], [10] predict nodes and edges but do not guarantee valid tree topology and scale poorly. Sequential tracking methods [11], [12] iteratively generate centerlines but suffer from low recall due to imbalanced bifurcation/termination classification and cannot correct early errors.

To address these limitations, we propose RefTr, a parameter-efficient Transformer based [13], [14] 3D image-to-graph model that generates centerline trees via recurrent refinement of confluent trajectories. RefTr follows a Producer–Refiner architecture where the Producer predicts multiple trajectory candidates rooted at the input patch center and a shared Refiner recurrently aligns them with the target, enhancing precision and reducing decoder parameters by 2.4× compared to the state-of-the-art Trexplorer Super [12]. The confluent trajectory representation allows refinement of complete trajectories while explicitly encoding tree topology through predicted divergence and end positions, as illustrated in Figure 1a.

RefTr maximizes recall and improves precision by predicting multiple candidates per target via many-to-one matching, creating an ensembling effect. This intentional overprediction introduces duplicate trajectories, which are suppressed at the patch level using predicted divergence positions between trajectory pairs, since duplicates do not diverge. Remaining duplicates at the global tree level are removed using Tree Non-Maximum Suppression (TNMS), a novel, efficient algorithm that merges overlapping branches in spatial tree graphs while preserving topology. For robust comparison, we extend the evaluation framework of Trexplorer Super [12] with radius-aware thresholds that scale with branch radius, and report the average across multiple thresholds.

Our contributions include: (1) RefTr, a parameter-efficient 3D image-to-graph framework that predicts confluent trajectories and achieves the highest recall and precision among compared methods; (2) Tree Non-Maximum Suppression, an efficient algorithm for removing duplicate branches while preserving tree topology; and (3) radius-aware evaluation metrics for robust comparison across datasets with varying branch radii.

2 Method

RefTr generates centerline graphs of tubular trees from CT volumes using a Transformer-based Producer–Refiner architecture and a confluent trajectory representation. The model architecture is shown in Figure 1.

2.1 Problem Formulation and Inference

The goal is to infer a centerline tree graph $G = (V, E)$ from a 3D image patch. Each node $\mathbf{v} \in V$ represents a spatial position and the branch radius at that position. RefTr represents the tree as a set of confluent trajectories $\mathcal{T} = \{T_i\}_{i=1}^{n}$, as shown in Figure 1a, where each trajectory is a sequence of nodes originating from the patch center.

Confluence is modeled by overlapping trajectory segments until divergence positions, after which trajectories separate to form branches. During inference, predicted trajectories are converted into a tree by introducing bifurcations at divergence points and terminations at predicted end positions. Branches extending beyond trajectory length are explored recursively using new patches.

2.2 RefTr architecture

RefTr consists of an image encoder and a customized Transformer decoder with two modules, a Producer and a Refiner. The encoder extracts feature representations from the input patch. For a fair comparison, we adopt the same encoder as Trexplorer Super [12], a modified SwinUNETR [15]. The Producer generates n initial trajectory proposals from learned token sets, and the Refiner recurrently updates them over $S-1$ steps using shared parameters. Both modules use a single decoder block. This recurrent process improves alignment with the m target trajectories ($n > m$) while keeping the model parameter efficient. The full pipeline is shown in Figure 1c.

Each trajectory token set includes L node tokens that predict node positions and radii, an end token that predicts the termination position, and a divergence token that is pairwise concatenated with those of other trajectories to predict branching positions between them. The end and divergence positions are defined along the trajectory and range from 0 to L.

Self-attention models interactions among trajectories, enabling effective many-to-one matching in which predictions are distributed approximately

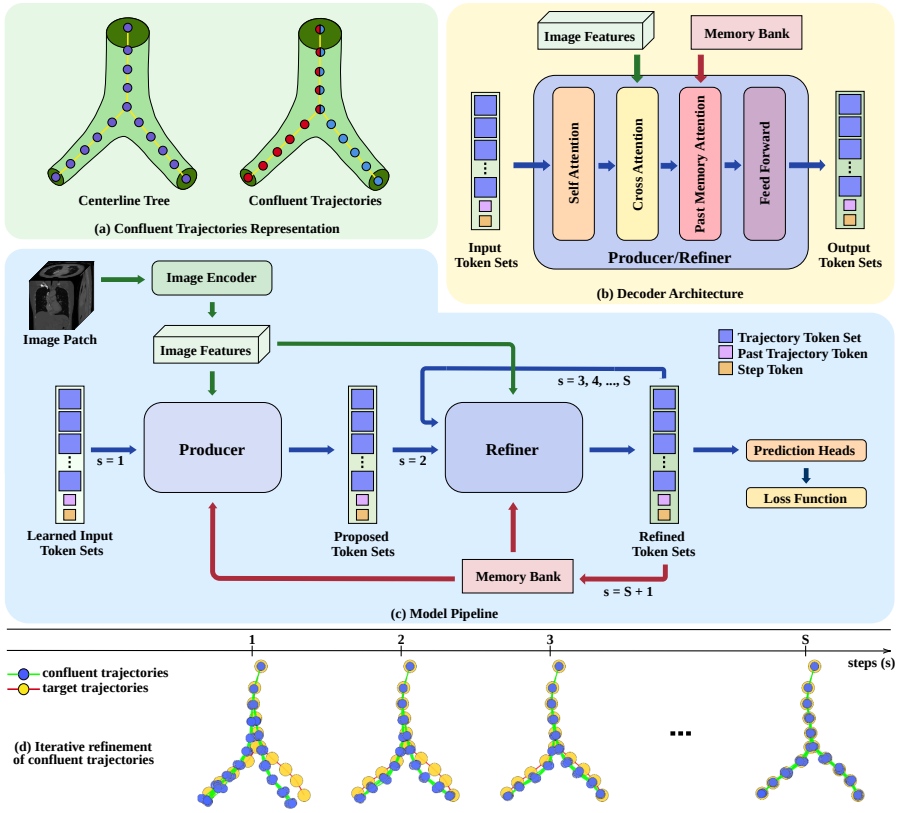


Figure 1: Overview of the RefTr architecture. **(a)** A tubular tree centerline graph with corresponding confluent trajectories and divergence position at the fifth node. **(b)** Producer/Refiner architecture with Self-, Cross-, and Past-Memory attention. **(c)** Model pipeline including Image Encoder, Producer, Refiner, and prediction heads. The Producer uses learned token sets and image features to propose multiple trajectory embeddings, which the Refiner iteratively refines over S steps. Loss is computed at every step, including the Producer step (omitted for clarity). One refined token set per continuing branch is stored in the Memory Bank for the next patch. **(d)** Recurrent refinement of multiple confluent trajectories toward two target trajectories over S steps.

evenly across targets, yielding an ensembling effect that improves recall and

precision. Cross-attention aggregates image features. Past-Memory attention incorporates context from a memory bank that stores token sets of ancestor trajectories from up to five previously processed patches. This context helps maintain the correct trajectory direction, especially for branches with large radii where a single patch has limited field of view, and reduces duplicate predictions across patches. As in Trexplorer Super, each training sample contains six patches along a single trajectory, so the memory bank holds up to five ancestor token sets. The detailed decoder architecture is shown in Figure 1b.

In addition to trajectory token sets, we use a past trajectory token that encodes up to the last ten positions and radii from the patch center. This token helps determine the centerline direction, especially when the memory bank is empty. We also include a step token that indicates the current refinement step and regulates the magnitude of updates made by the Refiner.

2.3 Matching and Loss

Predicted trajectories are matched to targets using a combined L1 (Manhattan) cost over position and radius differences. To allow multiple predictions per branch, we employ many-to-one matching by replicating targets and solving a bipartite assignment with the Hungarian algorithm [16].

All loss components use L1 loss. The total loss aggregates trajectory position and radius errors across refinement steps, and includes end-position and divergence losses at the final step, where trajectories are most refined:

$$\mathcal{L} = \sum_{s} (\alpha_{\text{pos}} \mathcal{L}_{\text{pos}}^{(s)} + \alpha_{\text{rad}} \mathcal{L}_{\text{rad}}^{(s)}) + \alpha_{\text{end}} \mathcal{L}_{\text{end}} + \alpha_{\text{div}} \mathcal{L}_{\text{div}}. \tag{C.1}$$

To ensure consistent supervision, the matching is computed only once using the predictions from the first step. This fixed assignment is reused for all remaining $S - 1$ refinement steps during loss computation.

2.4 Tree Non-Maximum Suppression

RefTr predicts multiple trajectories per target using many-to-one matching, creating an ensembling effect that maximizes recall and improves precision. Duplicate predictions are filtered at the patch level using the pairwise divergence position. To remove duplicates globally, we introduce Tree Non-Maximum

Suppression (TNMS), an efficient algorithm that merges overlapping branches while preserving tree topology.

Given a predicted tree with N nodes, TNMS performs a pre-order traversal and identifies duplicate nodes using a radius-adaptive spatial threshold. Branches with a high fraction of duplicates are merged, and any cycles are resolved by retaining edges closest to the root. Using KD-tree queries, the algorithm runs in $\mathcal{O}(N \log N)$ time, processing graphs with thousands of nodes in under one second on a CPU.

3 Experiments and Results

3.1 Datasets

We evaluate on the benchmark centerline datasets [17] introduced in prior work [12], including a synthetic dataset, the ATM'22 airway dataset [18], and the Parse 2022 pulmonary artery dataset [19], covering diverse structures.

3.2 Evaluation metrics

For consistency with prior work Trexplorer Super [12], we report node-level, branch-level, and graph-level metrics. To enable a more robust evaluation, we introduce radius-aware matching thresholds that scale with the ground-truth branch radius. A predicted node is considered a true positive if it lies within $\max(1.5, \tau^{\text{rad}} \cdot r)$ of a previously unmatched target node. Duplicate predictions are treated as false positives.

At the node-level, we report radius-aware Average Precision (rAP), Average Recall (rAR), and Average F1-score (rF1), averaged over $\tau^{\text{rad}} = 0.25{:}0.05{:}0.75$ to reduce sensitivity to threshold selection. For branch-level evaluation, a predicted branch is counted as a true positive if it overlaps with at least a fraction τ^{match} of nodes from an unmatched target branch. We report rBAP, rBAR, and rBF1, the branch-level counterparts of the node-level metrics. These are averaged over $\tau^{\text{match}} = 0.4{:}0.05{:}0.8$, with τ^{rad} fixed at 0.5. For graph-level evaluation, we report topological metrics, specifically the mean absolute error (MAE) of Betti-0 (connected components) and Betti-1 (cycles), following Trexplorer Super.

Table 1: Comparison of various models using point-level metrics.

Model	Point-level@ τ^{rad} = [0.25:0.05:0.75]			
	rAP(%)↑	rAR(%)↑	rF1(%)↑	Radius (MAE)↓
Synthetic Dataset				
Vesselformer	48.53 ± 10.35	70.05 ± 2.76	53.15 ± 8.19	0.43 ± 0.011
Trexplorer	31.10 ± 9.52	78.64 ± 4.10	39.64 ± 8.68	0.23 ± 0.032
Trexplorer Super	92.10 ± 3.29	70.58 ± 3.03	77.98 ± 1.90	**0.10 ± 0.006**
RefTr (ours)	**94.88 ± 0.77**	**79.39 ± 1.03**	**85.69 ± 1.30**	0.45 ± 0.093
ATM'22 Dataset				
Vesselformer	24.85 ± 1.47	38.48 ± 1.00	29.84 ± 1.37	0.79 ± 0.020
Trexplorer	4.54 ± 1.44	5.90 ± 0.70	4.49 ± 0.29	0.98 ± 0.086
Trexplorer Super	71.32 ± 1.22	60.84 ± 3.38	61.45 ± 1.23	**0.40 ± 0.021**
RefTr (ours)	**74.92 ± 1.17**	**66.73 ± 1.48**	**68.15 ± 0.59**	0.60 ± 0.036
Parse 2022 Dataset				
Vesselformer	21.93 ± 2.06	18.21 ± 1.00	19.54 ± 2.69	1.11 ± 0.027
Trexplorer	11.37 ± 3.73	13.42 ± 7.80	11.36 ± 5.17	1.22 ± 0.304
Trexplorer Super	53.65 ± 5.91	35.91 ± 3.22	40.01 ± 3.86	**0.58 ± 0.025**
RefTr (ours)	**53.68 ± 1.09**	**36.69 ± 1.12**	**42.37 ± 1.13**	0.75 ± 0.054

3.3 Implementation details

Experiments were conducted on a single node with four NVIDIA A100 GPUs using mixed precision. RefTr was trained for 1.9 million iterations with $S = 8$ refinement steps, trajectory length $L = 10$, and $n = 20$ predicted branches. To ensure fair comparison with Trexplorer and Trexplorer Super [11], [12], we used the same dataset and train–test split, adopting the authors' reported hyperparameters (26 tokens per bifurcation node, maximum 196 tokens) and trained for 2 million iterations. For Vesselformer [9], we report the best performance obtained using the published hyperparameters, with 80 object tokens and 12 million training iterations.

3.4 Results

Table 1 and Table 2 report the mean and standard deviation over five runs for point-level, branch-level, and graph-level metrics. At the point level, RefTr achieves the highest recall and precision across all datasets with sub-voxel radius error. On Parse 2022, improvements are smaller due to missing small branches

Table 2: Comparison of various models using branch-level and graph-level metrics.

Model	Branch-level@ $\tau^{\text{match}} = [0.4{:}0.05{:}0.8]$			Graph-level (MAE)	
	rBAP(%)↑	rBAR(%)↑	rBF1(%)↑	Betti-0↓	Betti-1↓
Synthetic Dataset					
Vesselformer	20.43 ± 4.22	43.63 ± 4.72	25.43 ± 3.75	81.7 ± 16.8	653.5 ± 138.7
Trexplorer	30.58 ± 8.66	77.47 ± 4.11	39.40 ± 7.82	**0.00 ± 0.0**	**0.00 ± 0.0**
Trexplorer Super	92.35 ± 3.54	70.82 ± 3.06	78.22 ± 1.78	**0.00 ± 0.0**	**0.00 ± 0.0**
RefTr (ours)	**96.75 ± 0.72**	**79.96 ± 1.15**	**86.76 ± 1.38**	**0.00 ± 0.0**	**0.00 ± 0.0**
ATM'22 Dataset					
Vesselformer	11.70 ± 0.73	16.77 ± 1.07	13.46 ± 0.81	312.5 ± 25.1	180.4 ± 35.9
Trexplorer	4.02 ± 1.71	2.06 ± 0.46	2.33 ± 0.31	**0.00 ± 0.0**	**0.00 ± 0.0**
Trexplorer Super	68.25 ± 1.75	60.91 ± 3.64	59.87 ± 1.39	**0.00 ± 0.0**	**0.00 ± 0.0**
RefTr (ours)	**68.40 ± 1.97**	**66.41 ± 1.53**	**64.23 ± 0.89**	**0.00 ± 0.0**	**0.00 ± 0.0**
Parse 2022 Dataset					
Vesselformer	9.23 ± 0.80	6.31 ± 0.85	7.30 ± 0.45	410.1 ± 23.9	246.7 ± 78.1
Trexplorer	11.32 ± 3.56	11.77 ± 7.22	10.45 ± 4.98	**0.00 ± 0.0**	**0.00 ± 0.0**
Trexplorer Super	**51.42 ± 5.58**	**34.41 ± 1.09**	**38.17 ± 3.63**	**0.00 ± 0.0**	**0.00 ± 0.0**
RefTr (ours)	47.17 ± 1.19	32.31 ± 1.13	37.09 ± 1.16	**0.00 ± 0.0**	**0.00 ± 0.0**

in the ground truth (qualitative examples in supplementary material), which RefTr correctly predicts (as verified by a radiologist review) but is penalized for, and encoder limitations from limited training data (72 training examples). We used the same encoder as Trexplorer Super for a fair comparison.

RefTr achieves superior branch-level metrics on the Synthetic and ATM'22 datasets. On Parse 2022, performance is comparable. However, the small branches that are missing in the ground truth have a larger negative impact at this level because small and large branches are weighted equally. RefTr also predicts topologically correct centerline trees, as reflected by the graph-level metrics.

Table 3 shows that, compared to the Trexplorer Super, RefTr reduces the number of decoder parameters by 2.4×, achieves faster inference, and requires substantially less training time and memory, while delivering superior performance. Figure 2 provides a qualitative comparison.

Table 3: Number of parameters, runtime, training time, and memory usage for the compared models. Note: Vesselformer has a much higher effective runtime due to a multi-hour post-processing. Experiments conducted on four Nvidia A100 80GB GPUs.

Model	Parameters (M)			Runtime	Training	GPU Mem.
	Encoder	Decoder	Total	(ms/patch)	(hrs)	(GBs)
Vesselformer	82.83	13.59	98.73	23*	37.4	**13.8**
Trexplorer	23.31	25.22	55.46	222	18.6	51.34
Trexplorer Super	23.12	25.22	55.27	189	37.2	73.9
RefTr (ours)	**23.12**	**10.50**	**42.32**	**156**	**18.2**	39.5

Table 4: Ablation study evaluating the impact of the key components and design choices for the RefTr model on the ATM'22 dataset.

Description	Point-level@ $\tau^{\text{rad}} = [0.25{:}0.05{:}0.75]$		
	rAP(%)↑	rAR(%)↑	rF1(%)↑
Key High-level Components			
No Recurrent Refinement ($S = 2$)	61.96 ± 2.17	58.08 ± 1.17	56.70 ± 1.66
No Many-to-one Matching	66.70 ± 1.27	63.25 ± 3.49	61.34 ± 1.68
No Tree Non-max Suppression	37.98 ± 2.42	**69.96 ± 1.12**	37.98 ± 3.87
Recurrent Refinement Steps			
Refinement Steps: $S = 4$	65.10 ± 6.32	55.72 ± 2.83	58.17 ± 3.42
Refinement Steps: $S = 6$	64.39 ± 5.17	60.93 ± 2.57	59.42 ± 3.19
Refinement Steps: $S = 10$	71.43 ± 3.12	65.16 ± 1.58	65.60 ± 1.91
Past Trajectory and Memory Attention			
No Past Trajectory Token	**78.85 ± 3.24**	37.07 ± 1.21	48.34 ± 2.78
No Past-Memory Attention	61.27 ± 2.12	66.38 ± 1.87	60.46 ± 2.24
Producer and Refiner Design Configurations			
Shared Producer and Refiner	65.18 ± 1.44	61.76 ± 1.32	60.27 ± 1.78
Increase Producer Blocks: 2	68.80 ± 2.24	62.88 ± 1.11	63.19 ± 1.83
Increase Refiner Blocks: 2	71.81 ± 2.07	64.67 ± 1.57	64.46 ± 1.98
Proposed Model	74.16 ± 1.45	67.72 ± 1.49	**68.35 ± 1.27**

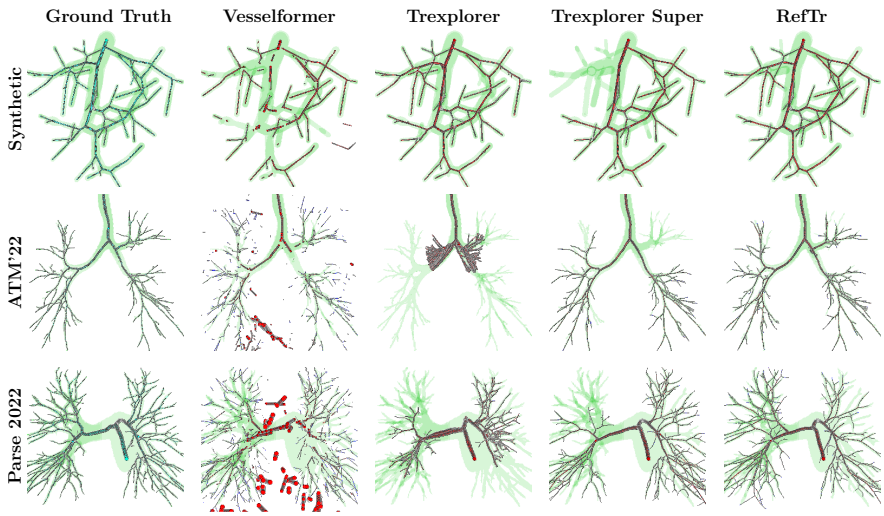


Figure 2: Visual comparison of ground truth and predictions from RefTr and baseline methods on one sample per dataset. Marker sizes are proportional to branch radii at node positions. The tubular structure mask (green) is for visualization only and not used during training or evaluation.

3.5 Ablation Study

Table 4 presents an extensive ablation study of our method's key components, including Recurrent Refinement, Many-to-One matching, Tree Non-Maximum Suppression, Past-Attention, and various Producer and Refiner design choices. We report the mean and standard deviation over three runs.

The study shows that Recurrent Refinement, Many-to-One matching, and Past-Memory attention are essential for achieving high recall, while Tree Non-Maximum Suppression is crucial for removing duplicates from the intentional overprediction.

Using the same decoder block for both the Producer and Refiner is suboptimal because the shared block must process two different input types: learned token sets and already refined token sets. Increasing the number of Producer blocks makes the initial proposals overly rigid, which limits the Refiner's ability to correct early errors. Conversely, increasing the number of Refiner blocks causes each refinement step to make larger updates, which can push predictions

off track and make subsequent steps less able to recover. Overall, the best performance is achieved with separate Producer and Refiner modules, each using a single decoder block and $S = 8$ refinement steps.

4 Conclusion

We present RefTr, a parameter-efficient framework for generating topologically accurate centerline graphs of tubular trees from 3D medical images. By predicting confluent trajectories and refining them recurrently, RefTr delivers state-of-the-art performance with substantially improved efficiency. Results across multiple datasets show faster inference and strong potential for clinical analysis. Future work will focus on improving the image encoder to further enhance feature quality and performance on challenging cases.

Acknowledgements. Compute and storage resources were provided by NAISS (grant 2022-06725, Swedish Research Council) and the Berzelius system at the National Supercomputer Centre, funded by the Knut and Alice Wallenberg Foundation.

Supplementary Material

This supplementary material is organized as follows. Section A describes how we reconstruct a centerline tree from the predicted confluent trajectories. Section B summarizes the experimental setup, including the architecture and training hyperparameters. Section C details the matching procedure and training losses. Section D describes Tree Non-Max Suppression (TNMS). Section E presents additional qualitative examples and discusses annotation ambiguities and representative failure cases.

A From Confluent Trajectories to a Centerline Tree

RefTr predicts a set of confluent trajectories, together with an end position for each trajectory and a divergence position for each pair of trajectories. These positions are represented as continuous values in $[0, 1]$. We first unnormalize them to the trajectory index range $[0, L - 1]$ and then discretize them by

rounding to the nearest integer, yielding a trajectory node index in $\{0, 1, \ldots, L-1\}$ at which the end or divergence occurs.

We construct the centerline tree from the predicted confluent trajectories as follows:

1. Group trajectories by their predicted divergence positions, such that trajectory pairs with divergence position $L-1$ are assigned to the same group. A divergence position of $L-1$ indicates that the two trajectories remain confluent until the final node.

2. Merge the trajectories within each group into a representative trajectory by averaging their node positions and node radii, and by averaging their end positions.

3. For the representative trajectories, compute pairwise divergence positions by averaging the corresponding predicted divergence positions.

4. Starting from the shared root, construct the tree level by level in a breadth-first manner.

5. At each level, cluster the representative trajectories according to their divergence positions. When multiple clusters are present, create a new branch for each cluster. These clusters may split again at later levels, giving rise to further branches.

6. Terminate branches once their predicted end positions are reached.

This procedure produces a single connected, acyclic tree and therefore preserves topological correctness.

B Experimental Setup

We provide the key architecture and training hyperparameters in Tables 5 and 6, respectively.

Table 5: Architecture hyperparameters.

Hyperparameter	Value
3D patch size	$64 \times 64 \times 64$
Producer blocks	1
Refiner blocks	1
Decoder heads	16
Token dimension	512
Decoder FF dim	2048
Position head dim	[512,512,512]
Radius head dim	[512,512,512]
Divergence head dim	[1024,512,256,128]
End head dim	[512,256,128,64]
Encoder feat. dim	24
Encoder depths	[2,2,2,2]
Encoder heads	[3,6,12,24]
Encoder patch size	2
Encoder window size	7

Table 6: Training hyperparameters.

Hyperparameter	Value
Training iterations	1.92M
Sequence length L	10
Refinement steps S	8
Predicted branches n	20
Position loss weight α_{pos}	4.2
Position cost weight λ_{pos}	3
Radius loss weight α_{rad}	1.15
Radius cost weight λ_{rad}	1
End loss weight α_{end}	0.94
Divergence loss weight α_{div}	0.3

C Matching and Loss Function

C.1 Matching cost

Given n predicted trajectories $\hat{\mathcal{T}} = \{\hat{T}_i\}_{i=1}^{n}$ and m target trajectories $\mathcal{T} = \{T_j\}_{j=1}^{m}$, we compute a matching from predictions to targets using a cost matrix $\mathbf{C} \in \mathbb{R}^{n \times m}$. Each entry $\mathbf{C}_{i,j}$ is defined as a weighted sum of position and radius costs:

$$\mathbf{C}_{i,j} = \lambda_{\text{pos}} \cdot \mathcal{C}_{\text{pos}}(\hat{T}_i, T_j) + \lambda_{\text{rad}} \cdot \mathcal{C}_{\text{rad}}(\hat{T}_i, T_j), \tag{C.2}$$

where

$$\mathcal{C}_{\text{pos}}(\hat{T}_i, T_j) = \frac{1}{L} \sum_{l=1}^{L} \left\| \hat{\mathbf{x}}_i^{(l)} - \mathbf{x}_j^{(l)} \right\|_1, \tag{C.3}$$

$$\mathcal{C}_{\text{rad}}(\hat{T}_i, T_j) = \frac{1}{L} \sum_{l=1}^{L} \left| \hat{r}_i^{(l)} - r_j^{(l)} \right|. \tag{C.4}$$

Here, $\lambda_\cdot$ denotes the scalar weight of each cost component.

To ensure consistent supervision across S refinement steps, the matching is computed only once using the predictions from the first decoding step. This

fixed assignment is then reused for the remaining $S-1$ steps during loss computation.

C.2 Many-to-one matching

We choose $n > m$ and enable many-to-one matching by replicating the target set to form a square cost matrix $\tilde{\mathbf{C}} \in \mathbb{R}^{n \times n}$. This allows us to apply the Hungarian algorithm to perform bipartite matching between the n predicted trajectories and the n replicated target trajectories:

$$\sigma^* = \arg\min_{\sigma \in \mathfrak{S}_n} \sum_{i=1}^{n} \tilde{\mathbf{C}}_{\sigma(i),i}, \tag{C.5}$$

where $\mathfrak{S}_n$ denotes the set of all permutations over n elements and σ^* is the permutation that minimizes the total cost.

Since the target trajectories are replicated, we map the matched replicated index back to the original target index set $\{1, \ldots, m\}$:

$$\hat{a}_i = 1 + ((\sigma^*(i) - 1) \bmod m), \tag{C.6}$$

where $\hat{a}_i$ denotes the target trajectory assigned to the i-th predicted trajectory.

C.3 Loss components

Let $\hat{a}_i$ denote the target trajectory assigned to predicted trajectory i. For each decoder step $s = 1, \ldots, S$, we define the position and radius losses as

$$\mathcal{L}_{\text{pos}}^{(s)} = \frac{1}{nL} \sum_{i=1}^{n} \sum_{l=1}^{L} \left\| \hat{\mathbf{x}}_i^{(l,s)} - \mathbf{x}_{\hat{a}_i}^{(l)} \right\|_1, \tag{C.7}$$

$$\mathcal{L}_{\text{rad}}^{(s)} = \frac{1}{nL} \sum_{i=1}^{n} \sum_{l=1}^{L} \left| \hat{r}_i^{(l,s)} - r_{\hat{a}_i}^{(l)} \right|. \tag{C.8}$$

The end loss and divergence loss are computed only at the final decoding step $s = S$, where the predicted trajectories are most refined. The end loss is

$$\mathcal{L}_{\text{end}} = \frac{1}{n} \sum_{i=1}^{n} \left| \hat{e}_i^{(S)} - e_{\hat{a}_i} \right|. \tag{C.9}$$

Let $\mathbf{D} \in \mathbb{R}^{m \times m}$ denote the target divergence matrix, where $\mathbf{D}_{j,\ell}$ is the divergence position between target trajectories T_j and T_ℓ. Given the assignment $\hat{a}_i$ for each predicted trajectory i, we construct a prediction-aligned target divergence matrix $\mathbf{D}' \in \mathbb{R}^{n \times n}$ by

$$\mathbf{D}'_{i,k} = \mathbf{D}_{\hat{a}_i, \hat{a}_k}, \quad \forall i \neq k. \tag{C.10}$$

That is, the target divergence supervising the pair of predictions (i, k) is taken from the divergence between their assigned target trajectories.

The divergence loss is then defined as

$$\mathcal{L}_{\text{div}} = \frac{1}{n(n-1)} \sum_{\substack{i,k=1 \\ i \neq k}}^{n} \left| \hat{\mathbf{D}}^{(S)}_{i,k} - \mathbf{D}'_{i,k} \right|. \tag{C.11}$$

C.4 Total loss

The overall loss aggregates the position and radius losses across all steps and adds the end and divergence losses from the final step:

$$\mathcal{L} = \sum_{s=1}^{S} \left(\alpha_{\text{pos}} \mathcal{L}^{(s)}_{\text{pos}} + \alpha_{\text{rad}} \mathcal{L}^{(s)}_{\text{rad}} \right) + \alpha_{\text{end}} \mathcal{L}_{\text{end}} + \alpha_{\text{div}} \mathcal{L}_{\text{div}}, \tag{C.12}$$

where $\alpha_\cdot$ are scalar weights that balance the contributions of the different loss terms.

D Tree Non-Max Suppression

Tree Non-Max Suppression (TNMS) is an efficient post-processing algorithm that merges overlapping branches in predicted trees while preserving topological consistency. Although the divergence head resolves many duplicates, TNMS removes any remaining overlaps.

Given a directed tree $G = (V, E)$ with 3D node positions and radii, TNMS performs a top-down (pre-order) traversal from the root. It maintains a list of visited nodes and uses a KD-tree for fast spatial queries. The spatial matching threshold is adaptive: for a node v with radius r_v, we use

$$\tau = \max(\tau_{\min}, \tau_{\text{pos}} \cdot r_v),$$

with $\tau_{\text{pos}} = 0.3$ and $\tau_{\text{min}} = 2.0$. The radius-aware threshold allows TNMS to operate robustly across vascular trees of different scales. Nodes within this threshold of previously visited nodes are flagged as duplicates.

Branches with a sufficiently large fraction of flagged nodes (above $\rho = 0.2$) are marked for merging. The hyperparameters τ_{pos}, τ_{min}, and ρ were selected by simple grid search. Duplicate nodes are grouped and replaced by a single node, and cycles are removed by retaining only the incoming edge whose parent is closest to the root. The full procedure is summarized in Algorithm 1.

TNMS is fast and scalable. It requires a single traversal, uses sublinear KD-tree lookups, and compares each node only to previously visited nodes, yielding $\mathcal{O}(N \log N)$ complexity for N nodes. Merging and cycle resolution are lightweight, allowing TNMS to process graphs with thousands of nodes in under a second on CPU.

E Failure Cases and Annotation Ambiguities

In this section, we present qualitative cases from the PARSE 2022 dataset that illustrate annotation ambiguities affecting evaluation, as well as representative prediction failures. The goal is to distinguish genuine failure cases from disagreements that arise from incomplete or ambiguous reference annotations.

E.1 Missing ground truth annotations

We present qualitative examples of branches missing from the ground truth annotations. As shown in examples in Figure 3, a bifurcation occurs along the selected vessel in the CT slice sequence, and a new branch emerges, but the ground truth does not follow this branch, whereas RefTr does. Because these correctly tracked vessels are counted as false positives in both point-level metrics (rAP, rF1) and branch-level metrics (rBAP, rBF1), they reduce the reported precision and overall F1 score.

Figure 4 shows examples of early termination in the ground truth annotations. In each case, the vessel is initially tracked by both the ground truth and RefTr, but the ground truth annotation stops after a few slices, even though the CT data clearly shows the vessel continuing. RefTr correctly continues to follow the vessel. However, these valid RefTr predictions beyond the prematurely terminated ground truth path are also treated as false positives in both point-

Algorithm 1 Tree Non-Max Suppression

Require: Graph $G = (V, E)$, radius scaling threshold τ_{pos}, minimum distance threshold $\tau_{\min}$, duplicate ratio ρ
Ensure: Deduplicated graph G
Initialize visited nodes $\mathcal{V} \leftarrow []$, positions $\mathcal{P} \leftarrow []$
Initialize branch nodes $\mathcal{B}_v \leftarrow []$, positions $\mathcal{B}_p \leftarrow []$
Set root node v_{root}
for each node v in pre-order from v_{root} **do**
  if v starts a new branch **then**
    Append $\mathcal{B}_v, \mathcal{B}_p$ to $\mathcal{V}, \mathcal{P}$; rebuild KD-tree
    Reset $\mathcal{B}_v \leftarrow []$, $\mathcal{B}_p \leftarrow []$
  end if
  Append v and its position to $\mathcal{B}_v, \mathcal{B}_p$
  Query KD-tree for nearest neighbor u within $\tau = \max(\tau_{\min}, \tau_{\text{pos}} \cdot r_v)$
  if match found **then**
    Mark v as duplicate of u
  end if
end for
Identify branches with duplicate ratio $\geq \rho$
Group duplicate pairs into sets and merge each set into a single node
for each node v with in-degree > 1 **do**
  Keep only the edge from the parent closest to the root
end for
return G

level and branch-level metrics, further lowering the reported precision and F1 score.

These instances of missing ground truth annotations have been confirmed by an expert radiologist. The authors of PARSE 2022 do not specify any criteria for excluding certain vessels from annotation. The selection may be based on criteria of clinical importance defined by the expert annotating clinicians. While many datasets omit vessels below a certain diameter, this does not appear to be the case for PARSE 2022, since the authors emphasize that thinner pulmonary arteries are both more challenging to detect and more clinically important. Depending on the task, the predicted centerline graphs can be readily post-processed to filter branches using length or radius thresholds.

E.2 Earlier predicted bifurcations

In Figure 5, we show several examples of vessel bifurcations in CT image patches. RefTr often bifurcates earlier than the ground truth annotation, likely to avoid missing a branch that could be overlooked if divergence is predicted too late. Both the predicted and ground truth centerlines remain spatially and topologically valid, since there is inherent ambiguity in the exact positions of centerline nodes and bifurcations. This ambiguity increases as vessel size increases.

The presence of this ambiguity can reduce branch-level metrics, as correctly predicted branches may be counted as false positives and ground truth branches as false negatives, lowering both precision and recall and resulting in a lower overall F1 score, especially for higher branch matching thresholds ($\tau^{\text{match}} > 0.7$).

E.3 Failure cases

In Figure 6, we show cases where another vessel in the CT image comes very close to the currently tracked vessel, either touching or nearly touching, before diverging. At the divergence point, RefTr interprets the nearby vessel as a new branch originating from the tracked vessel, predicts a bifurcation, and begins tracking it. This newly predicted branch is counted as a false positive, reducing both point-level and branch-level precision.

Figure 7 shows additional failure cases where RefTr fails to predict a bifurcation and misses a new branch. In these examples, the bifurcation is often ambiguous. In the first and second examples, the bifurcating vessels remain partially connected across several slices before fully separating. In such cases, RefTr treats the diverging vessel as an anomaly and does not predict it as a new branch, likely due to limited examples of such cases in the training dataset. In the third example, RefTr misses a relatively thin new branch compared to the original tracked vessel. This is uncommon, as vessels typically bifurcate into child branches with radii that are more similar and smaller than the parent branch. These cases result in lower point-level and branch-level recall.

These types of failures could be mitigated by increasing the diversity and amount of training data, applying hard-mining strategies for difficult vessels, or using data augmentation techniques. Exploring these approaches could improve performance in future work.

Figure 8 illustrates another failure mode, where RefTr prematurely terminates a currently tracked branch. In some cases, RefTr is uncertain whether a branch continues past an ambiguous region or represents a different structure, as in the first example. In others, RefTr appears to stop based on an internal estimate of where the ground-truth branch should end. In the second example, the ground-truth lower-leg branch ends at a later bifurcation, but RefTr terminates at an earlier bifurcation. These cases also result in lower recall. Failures of the first type could be addressed using the techniques mentioned for previous failure cases, while the second type may benefit from explicitly enforcing specific branch exclusion criteria.

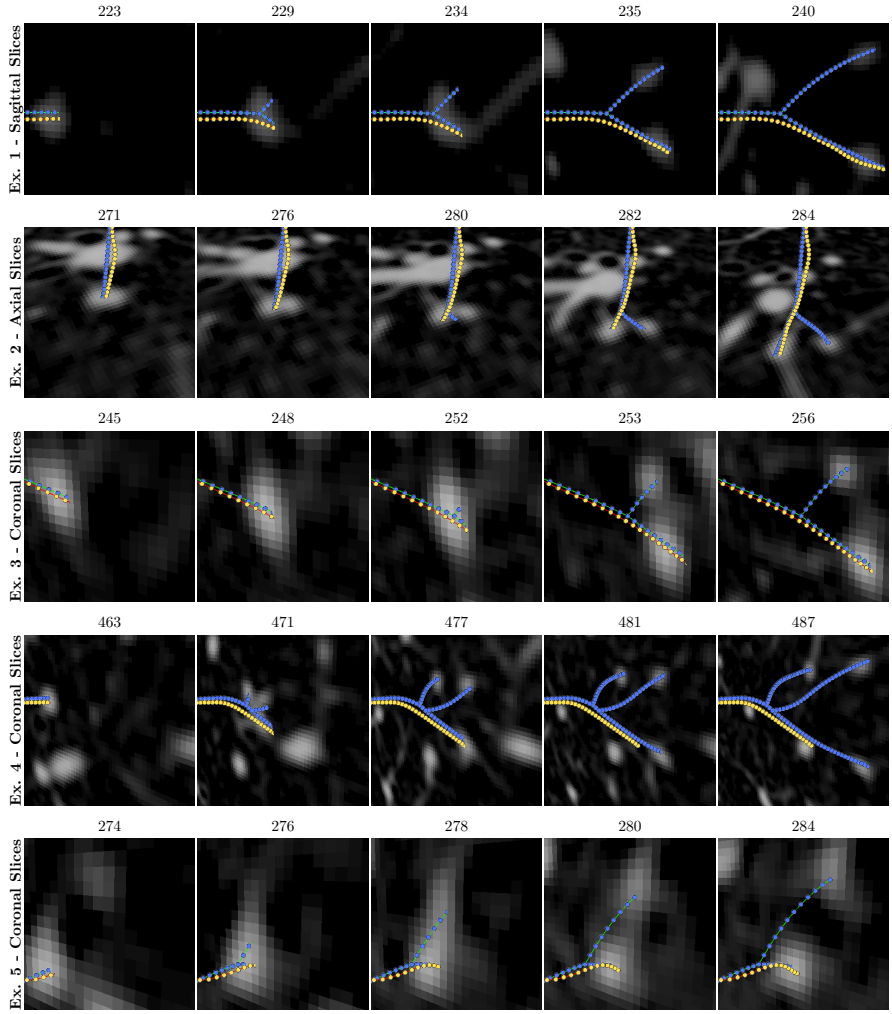


Figure 3: Qualitative examples of vessel bifurcations missing in the ground truth (yellow) but correctly predicted by RefTr (blue). Each row shows one example across a sequence of CT image slices. Only vessels branching from the selected vessel in the first slice are shown.

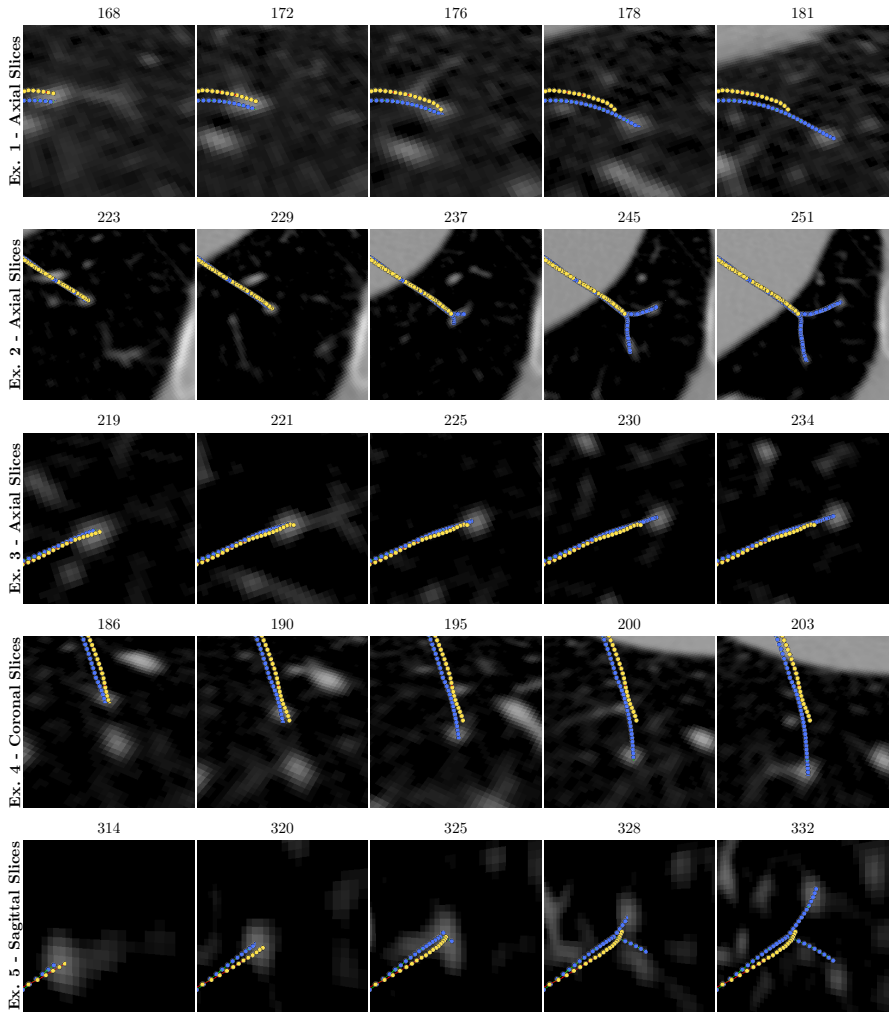


Figure 4: Qualitative examples of vessels terminated early in the ground truth (yellow) but correctly predicted by RefTr (blue). Each row shows one example across a sequence of CT image slices. Only vessels branching from the selected vessel in the first slice are shown.

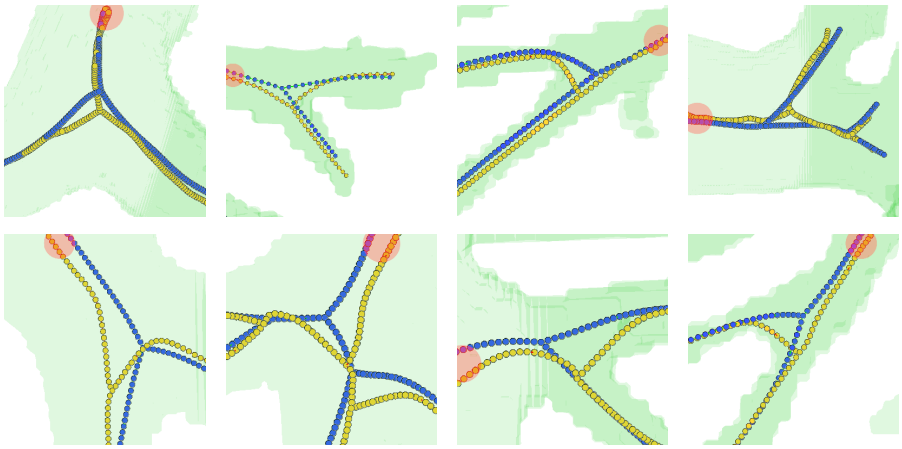


Figure 5: Qualitative examples of differing bifurcation locations between the ground truth (yellow) and RefTr predictions (blue). The red marker denotes the starting point of the highlighted vessel within each patch. Each grid cell shows a separate case. RefTr bifurcates earlier to avoid missing a branch.

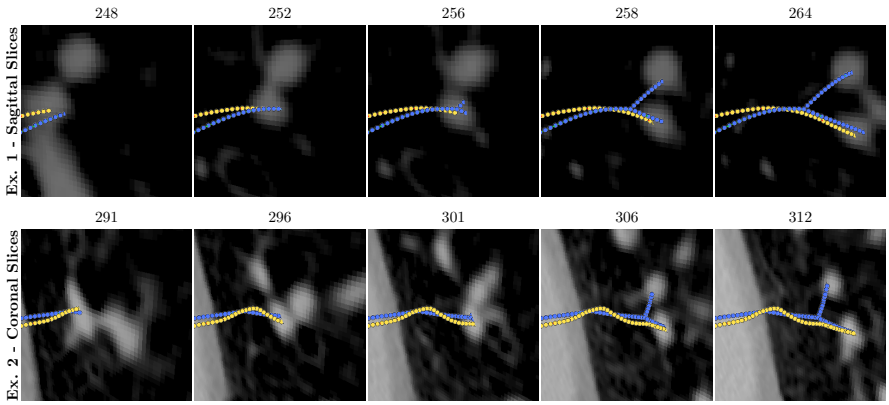


Figure 6: Qualitative examples of failure cases where two nearby branches run close together before diverging. At the divergence point, RefTr incorrectly begins to track the diverging branch as a new branch.

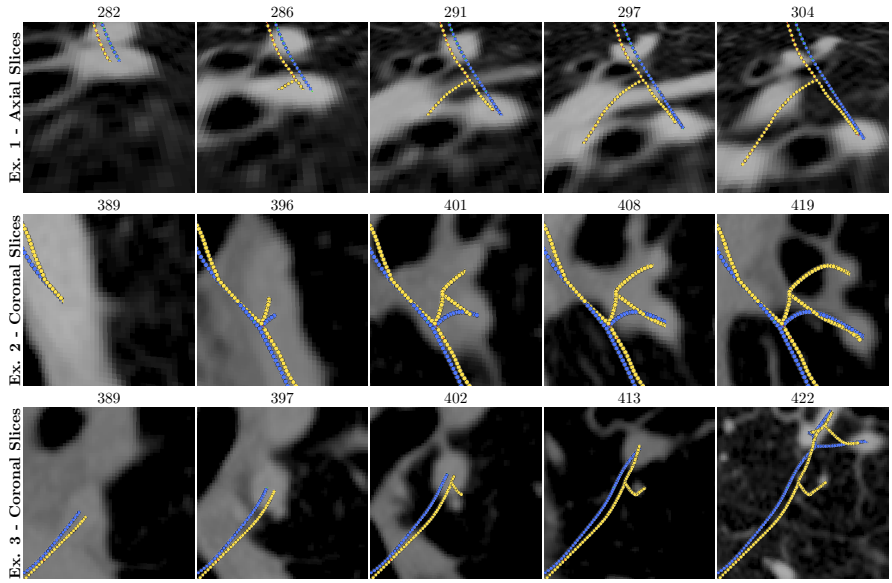


Figure 7: Qualitative examples of failure cases where RefTr (blue) misses a vessel bifurcation that leads to a new ground truth vessel (yellow).

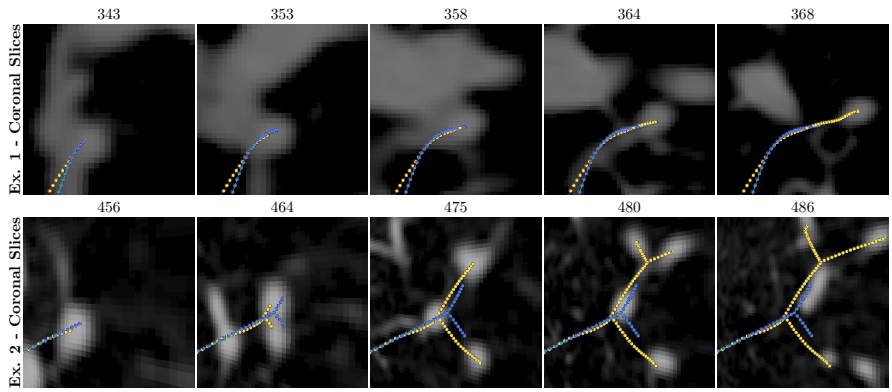


Figure 8: Qualitative examples of failure cases where RefTr (blue) terminates a branch earlier than the ground truth (yellow).

References

[1] H. Li, Z. Tang, Y. Nan, and G. Yang, "Human treelike tubular structure segmentation: A comprehensive review and future perspectives," *Computers in Biology and Medicine*, vol. 151, p. 106 241, 2022.

[2] S. Moccia, E. De Momi, S. El Hadji, and L. S. Mattos, "Blood vessel segmentation algorithms – review of methods, datasets and evaluation metrics," *Computer Methods and Programs in Biomedicine*, vol. 158, pp. 71–91, 2018.

[3] D. Huang, W. Tang, Y. Ding, T. Wan, and Y. Chen, "An interactive 3D preoperative planning and training system for minimally invasive vascular surgery," in *2011 12th International Conference on Computer-Aided Design and Computer Graphics*, IEEE, 2011, pp. 443–449.

[4] O. Miraucourt, S. Salmon, M. Szopos, and M. Thiriet, "Blood flow in the cerebral venous system: Modeling and simulation," *Computer Methods in Biomechanics and Biomedical Engineering*, vol. 20, no. 5, pp. 471–482, 2017.

[5] A. D. Choi et al., "CT evaluation by artificial intelligence for atherosclerosis, stenosis and vascular morphology (CLARIFY): A multi-center, international study," *Journal of Cardiovascular Computed Tomography*, vol. 15, no. 6, pp. 470–476, 2021.

[6] Z. Khan et al., "Three-dimensional morphometric analysis of the renal vasculature," *American Journal of Physiology – Renal Physiology*, vol. 314, no. 5, F715–F725, 2018.

[7] B. Wittmann, Y. Wattenberg, T. Amiranashvili, S. Shit, and B. Menze, "vesselFM: A foundation model for universal 3D blood vessel segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025, pp. 20 874–20 884.

[8] G. Tetteh et al., "DeepVesselNet: Vessel segmentation, centerline prediction, and bifurcation detection in 3-D angiographic volumes," *Frontiers in Neuroscience*, vol. 14, p. 1285, 2020.

[9] C. Prabhakar et al., "Vesselformer: Towards complete 3D vessel graph generation from images," in *Medical Imaging with Deep Learning*, PMLR, 2024, pp. 320–331.

[10] S. Shit et al., "Relationformer: A unified framework for image-to-graph generation," in *European Conference on Computer Vision*, Springer, 2022, pp. 422–439.

[11] R. Naeem, D. Hagerman, L. Svensson, and F. Kahl, "Trexplorer: Recurrent DETR for topologically correct tree centerline tracking," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2024, pp. 744–754.

[12] R. Naeem, D. Hagerman, J. Alvén, L. Svensson, and F. Kahl, "Trexplorer Super: Topologically correct centerline tree tracking of tubular objects in CT volumes," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2025, pp. 595–605.

[13] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European Conference on Computer Vision*, Springer, 2020, pp. 213–229.

[14] A. Vaswani et al., "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[15] A. Hatamizadeh, V. Nath, Y. Tang, D. Yang, H. R. Roth, and D. Xu, "Swin UNETR: Swin transformers for semantic segmentation of brain tumors in MRI images," in *International MICCAI Brainlesion Workshop*, Springer, 2021, pp. 272–284.

[16] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[17] R. Naeem, *Trexplorer Super datasets (1.0)*, `https://doi.org/10.5281/zenodo.15888958`, 2025.

[18] M. Zhang et al., "Multi-site, multi-domain airway tree modeling," *Medical Image Analysis*, vol. 90, p. 102 957, 2023.

[19] G. Luo et al., "Efficient automatic segmentation for multi-level pulmonary arteries: The PARSE challenge," *arXiv:2304.03708*, 2023.

PAPER D

CEVAR: Centerline Embedding Extraction for Endovascular Aneurysm Repair

Roman Naeem, Timo Niiniskorpi, Naman Desai, Charlotte Sandström, Anders Jeppsson, Ida Häggström, Fredrik Kahl, Håkan Roos, Jennifer Alvén

Under Review

The layout has been revised.

Abstract

Long-term mortality rates after endovascular aneurysm repair (EVAR) remain elevated due to post-EVAR rupture caused by loss of seal in stent graft sealing zones. Structured CT review using centerline measurements improves detection, but current workflows require manual centerline editing and expert operators. We propose a transformer framework for automated, protocol-driven sealing zone assessment that combines 3D centerline tracking with embedding-based geometric prediction. Two state-of-the-art image-to-graph models are evaluated for aorto-iliac centerline extraction from follow-up CT and for measurement of stent position, vessel diameters, and seal lengths according to EVAR4C protocol. Across the full test set and a challenging non-contrast subset, the proposed fully automatic method outperforms the commercial semi-automatic workflow.

Keywords: Endovascular aneurysm repair (EVAR), vascular centerline extraction, 3D image-to-graph.

1 Introduction

Abdominal aortic aneurysm is a dilation of the abdominal aorta exceeding 50% of normal diameter [1]. Enlargement increases rupture risk and treatment is recommended beyond a threshold size [2]. Endovascular Aneurysm Repair (EVAR) is now the predominant treatment due to favorable short term outcomes, yet long term survival remains limited due to late post-EVAR rupture [3].

Post-EVAR rupture is strongly associated with progressive loss of seal in proximal or distal stent graft sealing zones [4], [5]. Structured follow-up protocols detect loss of seal in up to 40% of patients within five years and predict failure more reliably than aneurysm sac expansion or endoleaks [6].

The EVAR4C protocol [6] evaluates sealing zones using vessel centerlines as reference for length and orthogonal diameter measurements. It more than doubles the detection of seal loss compared to conventional review, but requires manual centerline editing by highly trained operators [6], [7].

Accurate centerline generation is a major bottleneck. Existing tools are designed for contrast-enhanced preoperative CT and require extensive manual adjustment [8], [9] in post-EVAR scans affected by metal artifacts, altered anatomy, and low contrast.

Full automation of the EVAR4C protocol would enable standardized, objective sealing-zone evaluation in routine clinical follow-up, reducing operator dependency and manual workload. Achieving this requires reliable centerline extraction and protocol-specific measurements. We evaluate Transformer-based image-to-graph methods for centerline prediction in post-EVAR CT and investigate whether learned embeddings enable estimation of clinically relevant metrics.

Our key contributions are:

- A systematic assessment of Transformer-based 3D centerline extraction in post-EVAR CT, addressing challenges of metal artifacts and suboptimal contrast.
- A curated clinical dataset featuring manually refined centerlines and expert annotations for stent edges and sealing-zone measurements.
- An embedding-based framework that leverages learned point-wise representations for multi-task prediction of stent classification, vessel radii, and seal-zone length.
- An automated pipeline for protocol-driven centerline and geometric quantification, and a first step toward automated sealing-zone analysis in post-EVAR follow-up.

2 Related Work

Deep learning methods for EVAR imaging address screening [10], endoleak detection [11], morphological change [12], migration [13], and risk stratification [14]. These approaches detect late manifestations of failure and do not quantify sealing zone integrity. Prior work [15] on centerline extraction produces segmentations rather than protocol measurements. To our knowledge, no previous study has used learned centerline embeddings for automated sealing zone quantification in post-EVAR CT.

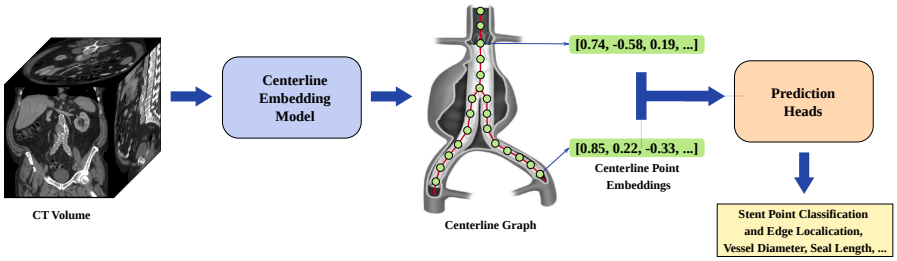


Figure 1: Proposed pipeline for automated post-EVAR analysis from CT. A CT volume is processed by a Transformer-based 3D centerline model that outputs the vessel centerline as point-wise embeddings. These embeddings are fed into a feature-prediction network to estimate clinically relevant measurements, including stent-region labels, vessel radii, and sealing-zone lengths, enabling automated EVAR4C protocol assessment.

3 Method

Postoperative CT examinations after EVAR were evaluated according to the EVAR4C protocol for CT review. The protocol specifies centerline-based measurements along each stent-graft limb, including sealing-zone lengths and orthogonal diameters at predefined locations. Vessel and stent-graft diameters were measured at (i) the stent-graft edges, (ii) 10 and 15 mm from each stent-graft edge, and (iii) the sealing-zone margin closest to the aneurysm. Sealing lengths were measured along the centerline and annotated for each graft limb.

3.1 Datasets

Private Dataset

The dataset was collected retrospectively from a clinical cohort of patients treated with standard bifurcated EVAR at two hospitals during 2010-2012. Ethical approval was granted (508-14). For each patient, the preoperative CT, the postoperative CT at one month, the postoperative CT at one year, and the last available follow-up CT were reviewed. For each CT examination, centerlines were generated and subsequently manually adjusted by the reviewing physician, using the commercially avaliable software Terarecon Intuition [16]. Markers indicating the stent-graft edges were placed manually.

All measurements and annotations were performed according to the EVAR4C protocol.

To train the centerline extraction models, we utilized 374 scans from 142 subjects, partitioned by subject into training, validation, and test sets. The training set contains 267 scans (102 subjects; 207 contrast/60 non-contrast), the validation set 32 scans (12 subjects; 30 contrast/2 non-contrast), and the test set 75 scans (28 subjects; 59 contrast/16 non-contrast).

Aorta24 Centerline Dataset

We derived a centerline dataset from the public Aorta24 challenge [17], which consists of 100 CT scans with corresponding multiclass aorta segmentation masks. Centerlines were automatically extracted from these masks using the Vascular Modeling Toolkit (VMTK) [18], [19]. This processed dataset served as the foundation for pretraining the centerline models.

3.2 Centerline Extraction Methods

We compare the commercial postprocessing workflow with state-of-the-art deep learning-based image-to-graph centerline extraction methods.

TeraRecon Intuition

TeraRecon Intuition [16] is a standard commercial platform for radiological post-processing. Its dedicated EVAR workflow uses an intensity-guided algorithm that assumes sufficient contrast enhancement for vessel tracking. In this study, we evaluate the software using three distinct initialization modes to establish a comprehensive baseline.

The automatic (SemAuto) mode generates multiple candidate paths, requiring the user to manually select the correct left and right aortic centerlines. In single-seed (2Seeds) mode, a seed is placed at the proximal aorta to generate a centerline for one branch, and a second seed is placed at the distal end of the other iliac branch to connect it to the first centerline. Finally, the two-click (4Seeds) mode requires the user to mark the vessel above and below the region of interest for both aorto-iliac branches. In all configurations, the final centerline is computed by linking these user-defined markers based on local voxel intensities.

Trexplorer Super

Trexplorer Super [20] is an image-to-graph framework using a Transformer architecture [21] to extract vascular centerline graphs from CT volumes via sequential tracking. From a localized seed point, the model expands the centerline graph breadth-first, ensuring a topologically consistent tree without disconnected components or cycles. Its core is a recursive tracking module that autoregressively predicts the next point embedding from the current point, capturing both local image features and long-range spatial dependencies along the vessel lumen.

Tracking is guided by a multi-class classification head labeling embeddings as intermediate, bifurcation, end, or background. Tracking proceeds along intermediate points and stops at end or background points. Bifurcations spawn multiple embedding copies to initialize potential new branches; some continue tracking while others are discarded based on subsequent classifications. This mechanism enables the model to dynamically follow complex, multi-scale vascular hierarchies while preserving global connectivity.

RefTr

RefTr [22] is a parameter-efficient 3D image-to-graph model representing centerline trees as confluent trajectories that share a root and follow the same path until bifurcation. Using a Transformer-based Producer–Refiner architecture, the Producer generates candidate trajectories from the input patch root at its center, and a shared Refiner iteratively aligns them with the target branches. This approach refines entire trajectories while enforcing topologically valid tree structures, improving precision and making the model lighter and faster than Trexplorer Super.

To boost accuracy, RefTr predicts multiple candidates per target, creating an ensembling effect resolved via a novel Tree Non-Maximum Suppression (TNMS) algorithm that merges redundant branches while preserving global topology. It also extends Trexplorer Super's evaluation framework, supporting point-, branch-, and graph-level metrics for centerline assessment, which we adopt in this work.

Prediction Heads

Due to RefTr's superior efficiency and performance, we extend it to predict EVAR4C protocol measurements. Beyond the standard centerline embedding heads, we add:

1. **Stent point classification head:** A binary head that predicts whether a point is inside or outside the stent, used to localize proximal and distal stent edges for diameter and seal length measurements.

2. **Stent edge regression head:** For stent edge points, a learned token is introduced, which is processed to estimate four values: edge diameter, diameters at 10mm and 15mm inward, and seal length.

Automatic Seed Localization

To enable a fully automated post-EVAR analysis pipeline, we replace manual initialization with a deep-learning-based seed localization module. A pretrained SwinUNETR [23] was fine-tuned for 80,000 iterations to regress a heatmap centered on the stent's proximal edge. The seed point is then extracted by thresholding the heatmap and selecting its peak coordinates, which serve as the starting point for both Trexplorer Super and RefTr.

Training Strategy

Trexplorer Super and RefTr were pretrained on the Aorta24 centerline dataset for 2 million iterations. These pretrained weights initialized all subsequent fine-tuning experiments on the private dataset, with each model fine-tuned for 200,000 iterations. Training used a cosine annealing learning rate scheduler with a single 20,000-iteration warm-up at the start and a base learning rate of 7×10^{-4}, on a single NVIDIA RTX A6000 GPU (48GB VRAM) with a batch size of 8. Pretraining required 190 GPU hours for Trexplorer Super and 124 GPU hours for RefTr, while fine-tuning took 21 GPU hours for Trexplorer Super and 14 GPU hours for RefTr.

4 Experiments and Results

4.1 Evaluation Metrics

For centerline evaluation, we adopt the hierarchical metrics proposed in RefTr [22]. Node-level performance is measured using radius-aware Average Precision (rAP), Recall (rAR), and F1-score ($rF1$). A predicted node is considered a true positive (TP) if it lies within a radius τ^{rad} of an unmatched target node. These metrics are averaged over $\tau^{rad} \in [3.0, 15.0]$ mm with a 1.5 mm step. Branch-level metrics ($rBAP$, $rBAR$, $rBF1$) evaluate structural connectivity, where a branch is a TP if it covers at least a fraction τ^{match} of nodes from an unmatched target branch. These are averaged over $\tau^{match} \in [0.4, 0.8]$ with a 0.05 step, using $\tau^{rad} = 6.0$ mm. Predictions outside the stent region are discarded, as the ground truth centerline is not well-defined in these areas. Graph-level topology is assessed using the Mean Absolute Error (MAE) of Betti-0 (connected components) and Betti-1 (cycles) numbers. We also report MAE for vessel diameter and seal length.

For stent node classification, we report mean Average Precision (mAP), mean Recall (mRec), and mean maximum F1-score (mF1), averaged across all τ^{rad} thresholds. All results reflect the mean and standard deviation across three independent runs.

4.2 Results

Table 1 summarizes point- and branch-level performance on the full test set. Commercial semi-automatic extraction using TeraRecon (TR) required multiple seeds to achieve reasonable accuracy but remained below the performance of learning-based methods. Among single-seed approaches, RefTr achieved the best overall performance, with the highest point-level recall and F1, as well as branch F1. Applying Tree Non-max Suppression to Trexplorer Super (TrexTNMS) improved branch precision, but it still lagged behind RefTr in recall.

In the fully automatic setting, all learning-based methods outperformed the commercial baseline. TrexTNMS achieved the highest precision, while RefTr provided the best precision–recall balance, yielding the highest branch F1 and recall. This indicates superior recovery of complete vascular trees without manual initialization.

Table 1: Comparison of TeraRecon (TR), Trexplorer Super (Trex), Trexplorer Super with Tree Non-maximum Suppression (TrexTNMS), and RefTr under different seeding strategies (SemAuto, 2Seeds, 4Seeds, 1Seed, Auto), evaluated using point-level and branch-level metrics for the **full test set**. Best scores are underlined.

Model	Point-level@ $\tau^{\text{rad}} = [3.0{:}1.5{:}15.0]mm$			Branch-level@ $\tau^{\text{match}} = [0.4{:}0.05{:}0.8]$		
	rAP(%)↑	rAR(%)↑	rF1(%)↑	rBAP(%)↑	rBAR(%)↑	rBF1(%)↑
Commercial Semi-automatic Methods						
TR:SemAuto	2.01	2.27	2.11	2.67	2.27	2.37
TR:2Seeds	53.48	58.56	55.19	71.83	60.59	63.90
TR:4Seeds	60.29	66.06	62.64	76.77	66.27	68.91
Centerline Semi-automatic Methods (1Seed)						
Trex	68.61 ± 1.07	72.38 ± 1.07	69.12 ± 0.23	89.76 ± 0.95	80.64 ± 2.52	83.27 ± 1.18
TrexTNMS	71.24 ± 0.95	72.99 ± 1.77	70.78 ± 0.30	94.29 ± 1.31	79.95 ± 2.00	84.57 ± 0.39
RefTr	70.49 ± 0.22	**78.53 ± 0.59**	73.27 ± 0.23	**94.52 ± 0.91**	**85.50 ± 0.94**	**88.47 ± 0.38**
Centerline Fully Automatic Methods (Auto)						
Trex	74.83 ± 1.67	71.50 ± 0.91	71.33 ± 0.68	89.11 ± 2.44	59.36 ± 0.66	68.58 ± 0.27
TrexTNMS	**78.47 ± 1.39**	72.49 ± 1.99	73.61 ± 0.35	94.49 ± 2.25	60.15 ± 1.20	70.26 ± 0.74
RefTr	76.12 ± 0.29	75.76 ± 0.92	**74.59 ± 0.65**	93.75 ± 0.80	66.19 ± 1.10	75.23 ± 0.66

Table 2: Comparison of TR, Trex, TrexTNMS, and RefTr across different seeding strategies (SemAuto, 2Seeds, 4Seeds, 1Seed, Auto), evaluated with point-level and branch-level metrics on the **non-contrast test set**. Best scores are underlined.

Model	Point-level@ $\tau^{\text{rad}} = [3.0{:}1.5{:}15.0]mm$			Branch-level@ $\tau^{\text{match}} = [0.4{:}0.05{:}0.8]$		
	rAP(%)↑	rAR(%)↑	rF1(%)↑	rBAP(%)↑	rBAR(%)↑	rBF1(%)↑
Commercial Semi-automatic Methods						
TR:SemAuto	0.00	0.00	0.00	0.00	0.00	0.00
TR:2Seeds	4.90	4.07	4.08	7.41	4.17	4.91
TR:4Seeds	15.20	13.56	14.20	25.69	13.19	13.21
Centerline Semi-automatic Methods (1Seed)						
Trex	66.94 ± 1.89	60.29 ± 4.50	62.31 ± 2.73	90.28 ± 3.64	66.44 ± 7.74	73.24 ± 7.23
TrexTNMS	68.47 ± 1.37	60.83 ± 4.84	63.35 ± 3.04	**91.20 ± 4.07**	66.67 ± 7.50	**73.77 ± 7.28**
RefTr	58.42 ± 2.24	59.87 ± 1.22	58.10 ± 0.60	83.02 ± 1.36	**69.21 ± 2.73**	73.00 ± 1.60
Centerline Fully Automatic Methods (Auto)						
Trex	76.33 ± 0.60	61.00 ± 1.76	66.61 ± 0.74	88.66 ± 2.82	46.53 ± 0.93	58.18 ± 1.02
TrexTNMS	**77.84 ± 0.49**	**61.96 ± 2.32**	**67.84 ± 1.18**	90.82 ± 1.67	47.69 ± 1.22	59.92 ± 0.84
RefTr	68.69 ± 2.65	60.99 ± 1.77	63.38 ± 0.58	85.26 ± 5.08	53.01 ± 1.01	62.59 ± 1.11

Results on the non-contrast subset (Table 2) reveal severe degradation for the commercial method, with near-zero scores even with additional seeds.

Table 3: Comparison of two approaches for diameter prediction: (1) fine-tuning the pretrained radius prediction head, and (2) introducing an additional token with a new prediction head. Diameters are evaluated at the three stent edges, as well as at 10 mm and 15 mm below the proximal edge and above the left and right distal edges. Stent seal length cannot be predicted using the pretrained head and is therefore reported only for the additional-token method.

Model	Proxal Edge Diameter (MAE)↓			Right Edge Diameter (MAE)↓		
	Edge	@10mm	@15mm	Edge	@10mm	@15mm
Radius Finetune	**1.27 ± 0.10**	**1.80 ± 0.06**	**2.07 ± 0.13**	**2.56 ± 0.11**	**1.81 ± 0.10**	**1.94 ± 0.05**
Additional Token	1.59 ± 0.15	2.43 ± 0.09	2.72 ± 0.05	3.49 ± 0.40	3.25 ± 0.59	3.22 ± 0.65
Model	Left Edge Diameter (MAE)↓			Stent Seal Length (MAE)↓		
	Edge	@10mm	@15mm	Proxal	Right	Left
Radius Finetune	**2.14 ± 0.08**	**1.80 ± 0.01**	**2.20 ± 0.13**	-	-	-
Additional Token	3.05 ± 0.19	2.90 ± 0.11	3.05 ± 0.15	**9.10 ± 0.38**	**13.12 ± 0.54**	**11.90 ± 0.23**

Learning-based methods remained robust, with TrexTNMS achieving the highest fully automatic point-level F1, and RefTr showing competitive branch recall, suggesting improved branch recovery under low-contrast conditions.

On the full test set, commercial extraction showed decreasing connectivity errors with more seeds (Betti-0 MAE: 0.960 → 0.347 → 0.027), whereas all learning-based methods achieved perfect topology (Betti-0 = Betti-1 = 0). On non-contrast scans, the commercial method degraded noticeably (Betti-0 MAE: 0.938 → 0.812 → 0.063), while learning-based approaches again produced topologically correct trees in all modes.

Vessel diameter measurements across proximal and distal stent edges ranged from approximately 7.6 to 56.5 mm, reflecting substantial anatomical variability. Seal lengths exhibited an even wider distribution, from 0 to 77.6 mm across sealing-zones. Table 3 compares diameter and seal length measurement accuracy. Fine-tuning the pretrained radius head achieved lower diameter errors than the additional-token approach, whereas the additional token enabled seal-length prediction at with larger error. Stent-region classification showed strong performance across models (mAP = 0.860 ± 0.003, mRec = 0.763 ± 0.008, mF1 = 0.826 ± 0.005,). Qualitative examples of these results are shown in Figure 2.

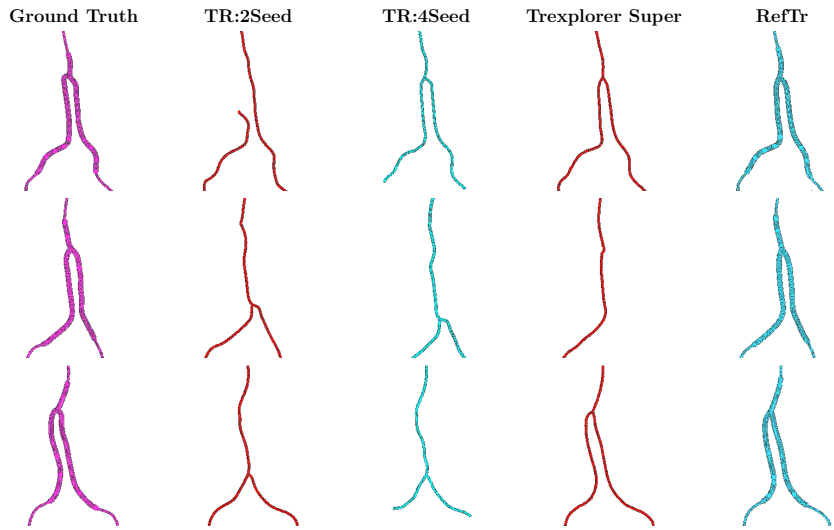


Figure 2: Visual comparison of ground truth and predicted centerlines. Larger points indicate stent nodes for the ground truth and RefTr, which uses a stent classification head. TR:AutoSem is omitted due to poor performance.

5 Discussion and Conclusion

In this work, we propose a Transformer-based framework for automated and protocol-driven sealing zone assessment after EVAR, combining 3D centerline tracking with embedding-based geometric prediction. Compared to commercially available semi-automatic software, our approach demonstrates substantially improved geometric accuracy and topological consistency, including in challenging non-contrast CT examinations.

Accurate centerline extraction is a prerequisite for reliable sealing zone evaluation within the EVAR4C protocol. The results demonstrate that learning-based centerline tracking can reduce dependence on manual adjustment while preserving the geometric consistency required for downstream measurements. In particular, the learned point-wise centerline embeddings enable reliable identification of stent graft edges and accurate diameter measurements at predefined protocol locations. Seal length prediction remains challenging, likely reflecting the difficulty of modeling vessel–stent interactions, especially

where separation between vessel wall and stent graft is subtle, and improved modeling likely requires larger training cohorts and richer geometric supervision. Ongoing work therefore focuses on expanding the cohort and incorporating more comprehensive pixel-wise annotations of sealing zone geometry.

Importantly, the proposed framework enables the use of existing large-scale longitudinal follow-up data consisting of routine CT examinations and associated tabular measurements, which opens the possibility of developing and validating fully automated EVAR follow-up pipelines at substantially larger scale. Overall, the results indicate that Transformer-based centerline tracking combined with embedding-based geometric prediction can provide a robust foundation for automated, protocol-driven post-EVAR evaluation.

Acknowledgements. The compute resources were provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS), partially funded by the Swedish Research Council through grant agreement no. 2022-06725.

References

[1] K. W. Johnston, R. B. Rutherford, M. D. Tilson, D. M. Shah, L. Hollier, J. C. Stanley, et al., "Suggested standards for reporting on arterial aneurysms," *Journal of Vascular Surgery*, vol. 13, no. 3, pp. 452–458, 1991.

[2] A. Wanhainen et al., "Editor's choice – European society for vascular surgery (ESVS) 2024 clinical practice guidelines on the management of abdominal aorto-iliac artery aneurysms," *European Journal of Vascular and Endovascular Surgery*, vol. 67, no. 2, pp. 192–331, Feb. 2024, ISSN: 1078-5884.

[3] J. T. Powell et al., "Meta-analysis of individual-patient data from EVAR-1, DREAM, OVER and ACE trials comparing outcomes of endovascular or open repair for abdominal aortic aneurysm over 5 years," *Journal of British Surgery*, vol. 104, no. 3, pp. 166–178, 2017.

[4] N. Diehm, F. Dick, B. T. Katzen, J. Schmidli, C. Kalka, and I. Baumgartner, "Aortic neck dilatation after endovascular abdominal aortic aneurysm repair: A word of caution," *Journal of Vascular Surgery*, vol. 47, no. 4, pp. 886–892, 2008.

[5] F. B. Gonçalves et al., "Iliac seal zone dynamics and clinical consequences after endovascular aneurysm repair," *European Journal of Vascular and Endovascular Surgery*, vol. 53, no. 2, pp. 185–192, 2017.

[6] C. Sandström et al., "Sealing zone failure decreases the long term durability of endovascular aneurysm repair," *European Journal of Vascular and Endovascular Surgery*, vol. 69, no. 2, pp. 238–247, 2025, ISSN: 1078-5884.

[7] M. Andersson et al., "Structured computed tomography analysis can identify the majority of patients at risk of post-endovascular aortic repair rupture," *Journal of Vascular Surgery*, vol. 76, no. 5, p. 1424, 2022.

[8] C. Adam et al., "Pre-surgical and post-surgical aortic aneurysm maximum diameter measurement: Full automation by artificial intelligence," *European Journal of Vascular and Endovascular Surgery*, vol. 62, no. 6, pp. 869–877, 2021.

[9] M. A. Corriere, A. Islam, T. E. Craven, T. D. Conlee, J. B. Hurie, and M. S. Edwards, "Influence of computed tomography angiography reconstruction software on anatomic measurements and endograft component selection for endovascular abdominal aortic aneurysm repair," *Journal of Vascular Surgery*, vol. 59, no. 5, pp. 1224–1231, 2014.

[10] G. Spinella et al., "Artificial intelligence application to screen abdominal aortic aneurysm using computed tomography angiography," *Journal of Digital Imaging*, vol. 36, no. 5, pp. 2125–2137, 2023.

[11] Q. Yang et al., "Detection of endoleak after endovascular aortic repair through deep learning based on non-contrast CT," *CardioVascular and Interventional Radiology*, vol. 47, no. 9, pp. 1267–1275, 2024.

[12] F. Lareyre et al., "Imaging analysis using artificial intelligence to predict outcomes after endovascular aortic aneurysm repair: Protocol for a retrospective cohort study," *BMJ Open*, vol. 15, no. 7, e098724, 2025.

[13] U. Asenbaum et al., "Stent-graft surface movement after endovascular aneurysm repair: Baseline parameters for prediction, and association with migration and stent-graft-related endoleaks," *European Radiology*, vol. 29, no. 12, pp. 6385–6395, 2019.

[14] B. Long, D. L. Cremat, E. Serpa, S. Qian, and J. Blebea, "Applying artificial intelligence to predict complications after endovascular aneurysm repair," *Vascular and Endovascular Surgery*, vol. 58, no. 1, pp. 65–75, 2024.

[15] B. Sabrowsky-Hirsch, S. Thumfart, W. Fenz, R. Hofer, P. Schmit, and F. Fellner, "Automatic segmentation of the abdominal aorta and stent-grafts," *Journal of Image and Graphics*, vol. 9, no. 3, 2021.

[16] TeraRecon, Inc., *Intuition advanced visualization platform*, Computer software, Durham, NC, USA, 2025.

[17] M. Imran, J. R. Krebs, M. A. Cooper, J. Ma, Y. Zhou, and W. Shao, "Multi-class segmentation of the aorta," in *AortaSeg 2024 Challenge, Held in Conjunction with MICCAI 2024*, Cham, Switzerland: Springer, 2024.

[18] L. Antiga and D. A. Steinman, *VMTK: The vascular modeling toolkit*, Accessed: 2025-02-24.

[19] Slicer Development Community, *3D Slicer*, Accessed: 2025-02-24, 2025.

[20] R. Naeem, D. Hagerman, J. Alvén, L. Svensson, and F. Kahl, "Trexplorer Super: Topologically correct centerline tree tracking of tubular objects in CT volumes," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2025, pp. 595–605.

[21] A. Vaswani et al., "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[22] R. Naeem, D. Hagerman, J. Alvén, and F. Kahl, *RefTr: Recurrent refinement of confluent trajectories for 3D vascular tree centerlines*, Under review, 2026.

[23] A. Hatamizadeh, V. Nath, Y. Tang, D. Yang, H. R. Roth, and D. Xu, "Swin UNETR: Swin transformers for semantic segmentation of brain tumors in MRI images," in *International MICCAI Brainlesion Workshop*, Springer, 2021, pp. 272–284.

PAPER E

ARTA: Adaptive Mixed-Resolution Token Allocation for Efficient Dense Feature Extraction

David Hagerman*, **Roman Naeem***, Erik Brorsson, Fredrik Kahl, Lennart Svensson

Under Review

*Shared first authorship.

The layout has been revised.

Abstract

We present ARTA, a mixed-resolution coarse-to-fine vision transformer for efficient dense feature extraction. Unlike models that begin with dense high-resolution (fine) tokens, ARTA starts with low-resolution (coarse) tokens and uses a lightweight allocator to predict which regions require more fine tokens. The allocator iteratively predicts a semantic (class) boundary score and allocates additional tokens to patches above a low threshold, concentrating token density near boundaries while maintaining high sensitivity to weak boundary evidence. This targeted allocation encourages tokens to represent a single semantic class rather than a mixture of classes. Mixed-resolution attention enables interaction between coarse and fine tokens, focusing computation on semantically complex areas while avoiding redundant processing in homogeneous regions. Experiments demonstrate that ARTA achieves state-of-the-art results on ADE20K and COCO-Stuff with substantially fewer FLOPs, and delivers competitive performance on Cityscapes at markedly lower compute. For example, ARTA-Base attains 54.6 mIoU on ADE20K in the ∼100M-parameter class while using fewer FLOPs and less memory than comparable backbones.

Keywords: Segmentation, adaptive token allocation, mixed-resolution.

1 Introduction

In semantic segmentation tasks, not all pixels are equally important. In a typical scene, large regions are often homogeneous (e.g., sky, road, or walls) and exhibit little spatial or semantic variation, making dense per-pixel computation unnecessary. In contrast, regions with high object density or class boundaries require higher spatial resolution for accurate predictions. This uneven distribution of semantic content motivates architectures that adaptively allocate higher spatial resolution to more informative regions of the image.

Most computer vision architectures process images using a uniform grid of

square patches, assigning equal computational cost to each patch regardless of its semantic content. For example, in Vision Transformers [1], it is common to divide the input image into fixed-size patches (e.g., 16×16), each of which is embedded into a token, applying the same representation capacity to both simple and complex patches. Convolutional neural networks such as ConvNext [2] similarly apply filters over regularly spaced patches, implicitly treating all areas of the image with equal importance. While some recent models [3], [4], [5] introduce mechanisms for content-aware processing or adaptive computation, the majority of widely used architectures still rely on uniform spatial partitioning, which does not align with the uneven distribution of information in real-world images.

While uniform spatial partitioning is inefficient, most non-uniform approaches still follow a high-to-low resolution processing pipeline. In these architectures, the input image is initially represented at high resolution, and information is gradually compressed through pooling, striding, or token merging. As a result, all image patches, regardless of their semantic importance, are initially processed at the finest spatial granularity. This leads to unnecessary computation in homogeneous or uninformative patches and limits the potential efficiency gains of token pruning or merging in later stages. An ideal approach would avoid assigning high-resolution capacity to patches that do not require it in the first place, allocating computational resources only where semantic complexity demands it.

Beyond where computation is spent, there is also the question of how representational capacity is used within each token. When a token aggregates pixels from multiple semantic classes, its feature vector must encode the class mixture (which classes are present and in what proportions), the spatial layout and extent of each class within the patch, and the patch's overall position. By contrast, if a token predominantly corresponds to a single class, its representation can devote more dimensions to modeling intra-class variation and higher-order semantics rather than resolving class boundaries. Constraining tokens to be largely class-specific improves the efficiency of the representation, allowing models with modest width to rival the semantic expressiveness of wider baselines.

To address these computational and representational inefficiencies, we propose ARTA (**A**daptive Mixed-**R**esolution **T**oken **A**llocation), a two-stage encoder. In Stage 1, a lightweight allocator adaptively assigns token density

across the image through three hierarchical allocation rounds. Starting from coarse tokens, it predicts a semantic class-boundary score for each token and allocates additional finer-grained tokens to patches whose scores exceed a low threshold, increasing resolution only where boundary evidence is present. In subsequent rounds, scoring is re-applied only to the newly allocated finest-resolution tokens, reducing allocator compute by avoiding repeated dense evaluation over the full token set. Repeating this process yields a mixed-resolution token set that concentrates spatial detail near class boundaries and avoids unnecessary allocation in uniform regions, ensuring that no patch is processed at higher resolution than needed. In Stage 2, we refine the resulting mixed-resolution tokens with a deeper hierarchical encoder to capture higher-level semantics. To summarize, our main contributions are:

- A lightweight, class-boundary scoring allocator for adaptive mixed resolution token allocation. Starting from coarse tokens, it iteratively re-scores and allocates finer-grained tokens to patches containing class-boundaries, increasing token density in semantically rich regions while leaving uniform areas at coarse granularity.
- ARTA: A two-stage encoder that couples the proposed mixed-resolution token allocation with a deeper hierarchical refinement network. Mixed resolution attention enables interaction between coarse and fine tokens, concentrating computation on semantically complex patches.
- We validate ARTA on ADE20K, COCO-Stuff, and Cityscapes, achieving state-of-the-art results on ADE20K and COCO-Stuff with substantially fewer FLOPs, and competitive performance on Cityscapes at lower compute.

2 Related Work

2.1 Token dropping and merging

Many methods improve transformer efficiency by removing tokens deemed uninformative or by aggregating nearby/similar tokens [6], [7], [8], [9], [10], [11], [12], [13]. These fine-to-coarse strategies typically start from a dense high-resolution grid of tokens and progressively reduce the number of tokens by pruning or merging similar tokens to gain efficiency. This yields a

content-adaptive token density, but only through token reduction. The spatial resolution never exceeds that of the initial grid, and training still processes an initially dense token set. In contrast, ARTA follows a coarse-to-fine strategy that starts from coarse tokens and allocates additional tokens only where finer detail is needed, increasing resolution on demand rather than uniformly.

2.2 Adaptive downsampling

A complementary direction makes tokenization or routing content-aware, for example via saliency-driven multi-resolution grids, attention-guided sampling, or per-input policy decisions [14], [15], [16]. For dense prediction, AutoFocusFormer (AFF) [3] proposes point-based local attention with balanced clustering and a learnable neighborhood-merging module to support segmentation heads. Other works merge/share or prune tokens based on semantics or difficulty [4], [5]. While these approaches introduce adaptivity, most still start from a dense tokenization and then select, share, or prune tokens thereafter. As a result, they generally do not increase spatial token density on demand, and training compute remains dominated by the initial high-resolution token set.

2.3 Learned upsampling operators

Another line of work studies learned upsampling operators that incorporate local context. Dynamic upsampling operators such as DySample [17] and frequency-aware fusion modules such as FreqFusion [18] operate on dense feature maps. DySample learns sampling locations for each output position, while FreqFusion applies adaptive low-/high-pass filtering and offsets to boundary sharpness during upsampling. These approaches focus on how to upsample by improving the upsampling operator itself, and they typically upsample densely across spatial locations. In contrast, ARTA addresses what to upsample by starting from a coarse tokenization and allocating additional tokens only when required, producing a mixed-resolution representation that keeps homogeneous patches compact. Therefore, learned upsampling operators are orthogonal to ARTA.

2.4 Coarse-to-fine architectures

Several coarse-to-fine architectures follow an overview-to-detail schedule, redistributing capacity toward coarse scales [19], [20]. OverLoCK [20] begins with a coarse global context derived from low-resolution features, and later refines the representation using fine-grained attention. These architectures alter how capacity is distributed across scales but generally keep resolution schedules fixed and input-agnostic rather than content-driven within an image.

Beyond encoder design, boundary-refinement post-processing methods such as SegFix [21] improve boundary quality by redirecting boundary pixels toward more reliable interior predictions using learned offsets. In contrast, ARTA incorporates boundary awareness directly into the encoder through class-boundary scoring and hierarchical token allocation. SegFix operates as a post-processing step on model outputs, whereas ARTA controls where spatial resolution is allocated during feature extraction.

Overall, most prior work either reduces token count after a dense beginning, improves upsampling or fusion on dense feature maps, or redistributes capacity with fixed multi-scale plans. A gap remains for architectures that start from coarse tokens and adaptively allocate higher token density only where semantic complexity warrants it, while maintaining mixed-resolution representations that interact across scales during dense feature extraction.

3 Method

We present ARTA, a two-stage hierarchical encoder that constructs mixed-resolution feature representations by adaptively allocating token density to semantically complex image patches. ARTA is designed to encourage tokens to correspond to a single semantic class, rather than mixing multiple classes within the same token. This is achieved by identifying tokens likely to overlap semantic (class) boundaries and selectively allocating additional tokens to only those patches, while keeping homogeneous patches compact.

ARTA differs from token-pruning or token-merging approaches that begin with a dense high-resolution grid and later reduce token count. Instead, ARTA starts from coarse tokens and increases spatial resolution only where needed. This avoids ever allocating high-resolution tokens to semantically simple patches and enables a principled and efficient use of computation.

3.1 Overview

ARTA follows a two-stage design. In Stage 1, an adaptive mixed-resolution token allocator iteratively predicts a class-boundary score for tokens and hierarchically allocates additional high-resolution tokens to the image patches containing class-boundaries over multiple allocation rounds, producing a mixed-resolution token set. In Stage 2, a deeper hierarchical encoder refines token features over multiple refinement rounds to capture higher-level semantics using mixed-resolution attention, allowing coarse and fine tokens to interact while concentrating compute on semantically rich patches. The overall architecture is illustrated in Figure 1 and configurations details can be found in Table 1.

3.2 Adaptive Mixed-Resolution Token Allocation

Coarse initialization

Given an input image, ARTA begins with a coarse patch embedding using a 32×32 kernel to form an initial low-resolution token grid. During a pre-allocation round, these tokens are processed by a lightweight Vision Transformer to produce boundary-aware coarse features, enabling reliable class-boundary scoring during the allocation rounds.

Hierarchical allocation rounds

Stage 1 allocates tokens through R allocation rounds (we use $R = 3$). At allocation round r, the allocator predicts a class-boundary score $c_i^{(r)}$ for the candidate patch represented by token i at the current finest resolution under consideration. Patches with scores above a low threshold are selected for finer allocation, and additional tokens are introduced by partitioning each selected region into finer sub-patches (in our implementation, a 2×2 partition yielding four tokens per selected patch).

The resulting mixed-resolution token set is then processed by a cluster attention block from AFF [3], which computes attention over localized clusters that are recomputed each forward pass under the current allocation. While AFF adaptively downsamples by merging less important tokens into important ones, yielding non-overlapping tokens, ARTA can contain overlapping tokens at multiple resolutions covering the same region. We use the same clustering mechanism to enable efficient cross-resolution interaction, so that fine tokens

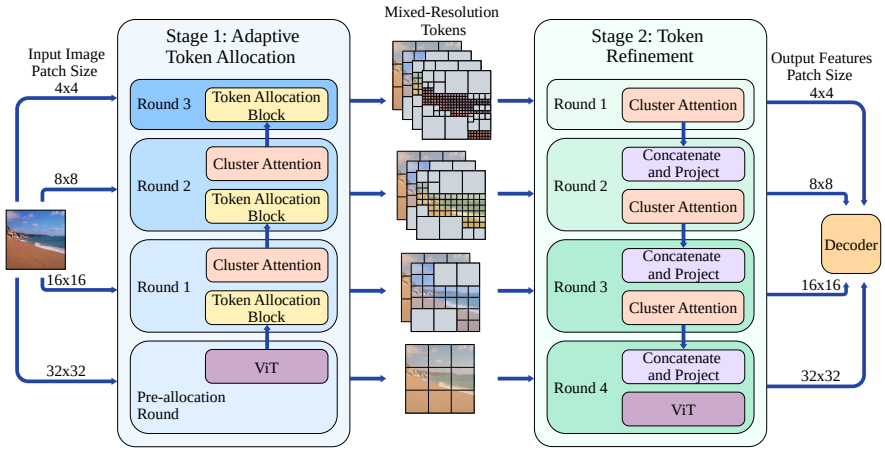


Figure 1: ARTA overview. ARTA has two stages: Adaptive Token Allocation (Stage 1) and Token Refinement (Stage 2). Stage 1 proceeds bottom-up from coarse 32×32 tokens. A pre-allocation ViT produces boundary-aware features, after which each allocation round applies a token allocation block (rounds 1–2 also use cluster attention). The allocation block scores the finest tokens and allocates finer-grained tokens to the corresponding patches containing class-boundaries (Figure 2), progressively building mixed-resolution sets up to $[32\times32, 16\times16, 8\times8, 4\times4]$. Stage 2 proceeds top-down from the final set and refines tokens using cluster attention, with a ViT in the final round. At each refinement round, the finest tokens are output, and the remaining tokens continue. Lateral Stage 1 outputs are fused into Stage 2 by concatenation and projection before attention. The decoder uses these multi-scale features for dense prediction.

can attend to nearby coarse tokens for high-level context, while coarse tokens absorb local detail carried by fine tokens that inject newly extracted sub-patch features.

At the next round, the allocator is re-applied only to the finest-resolution tokens produced by the previous allocation round, rather than re-scoring the entire token set, reducing allocation overhead by avoiding repeated dense evaluation.

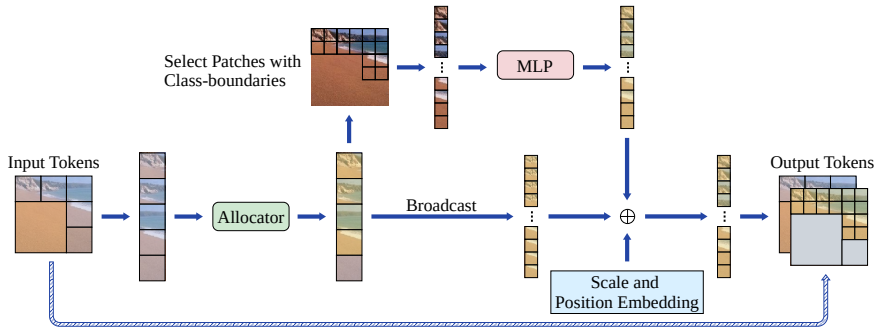


Figure 2: Token Allocation Block. The allocator scores the current finest-resolution tokens and selects the corresponding patches containing class-boundaries. Selected patches are split into 2×2 sub-patches to allocate finer tokens. Each new token is initialized from sub-patch image features (MLP) and combined with a broadcast residual from the parent token. Finally, scale and position embeddings are added to get output tokens.

Mixed-resolution token set

After R rounds, Stage 1 produces a mixed-resolution representation consisting of coarse tokens for homogeneous regions and finer tokens concentrated around predicted semantic boundaries. This ensures that no region is represented at a higher resolution than necessary.

3.3 Token Allocation Block

The token allocation block assigns a scalar class-boundary score to each token at the finest resolution in allocation round r. For an input token i, the allocator outputs a score $c_i^{(r)}$ estimating the amount of semantic (class) boundary content within the token patch.

Class-boundary score

The allocator is trained to predict a class-boundary score for each token, indicating how important it is to allocate additional tokens to the corresponding image patch. Ground-truth scores are derived from the segmentation labels by first computing a binary map of class-boundary pixels, where a pixel is

marked as a boundary (1) if any of its neighboring pixels has a different class label, and as non-boundary (0) otherwise. For each token patch, we sum the boundary pixels within the patch and normalize the result to obtain the target score. The allocator MLP is trained to regress this target using mean squared error (MSE).

This scoring choice encourages each token to represent a single semantic class, simplifying downstream reasoning. A token that spans multiple classes must encode both class identities and their spatial arrangement, increasing representational burden. In contrast, single-class tokens allow the encoder to focus on semantic meaning rather than intra-token class composition.

Threshold-based allocation

We allocate additional tokens only for regions whose predicted scores exceed a round-specific threshold τ_r. This induces a variable token budget per sample. Let N_r denote the number of candidate tokens for a given sample at round r, and let

$$K_r = \sum_{i=1}^{N_r} \mathbf{1}\Big(c_i^{(r)} > \tau_r\Big) \tag{E.1}$$

denote the number of selected tokens. The threshold τ_r is set slightly above zero to tolerate small prediction noise and to ensure that only regions with sufficient predicted boundary complexity are selected for further allocation.

Because K_r varies across samples, token counts differ within a mini-batch. During training, we pad each sample's token set (per allocation round) to the maximum token count in the batch and mask the padded tokens so they do not participate in attention or loss computation. This enables mini-batch training while ensuring each sample uses only the number of tokens required by its content.

Token allocation

For each selected token, the corresponding image patch is partitioned into 2×2 sub-patches, and allocate four finer-grained tokens, one per sub-patch. To initialize a new token, we extract the raw pixel values of its sub-patch, flatten them into a vector, and project this vector to the token embedding dimension using a learned linear layer. The projected vector is then passed through an MLP to produce an image feature. To reuse the selected token's

Table 1: Encoder architecture hyperparameters for ARTA-Tiny, ARTA-Small, and ARTA-Base. For each round, we report the token embedding dimensions and the number of attention blocks. Entries are listed in execution order, following ARTA's coarse-to-fine-to-coarse layout.

Encoder	Stage 1 Patch Size: $[32^2, 16^2, 8^2, 4^2]$		Stage 2 Patch Size: $[4^2, 8^2, 16^2, 32^2]$	
	Dims	Blocks	Dims	Blocks
ARTA-Tiny	[512, 256, 128, 64]	[1, 1, 1, 0]	[64, 128, 256, 512]	[4, 4, 16, 4]
ARTA-Small	[512, 256, 128, 64]	[2, 2, 2, 0]	[64, 128, 256, 512]	[4, 6, 24, 3]
ARTA-Base	[768, 384, 192, 96]	[2, 2, 2, 0]	[96, 192, 384, 768]	[8, 6, 18, 4]

boundary-aware coarse representation, we add it as a residual to each new token. Each token further receives (i) a learned scale embedding and (ii) a learned sub-patch position embedding to disambiguate within-patch spatial identity.

3.4 Mixed-Resolution Token Refinement

Stage 2 takes the mixed-resolution token set produced by the final allocation round in Stage 1 and refines token features in a top-down manner to capture higher-level semantics. Stage 2 is organized into multiple refinement rounds. In each round, we apply cluster attention [3] over localized token clusters, which is computationally efficient for sparse mixed-resolution token sets and enables cross-resolution interactions. Fine tokens gain broader semantic context by attending to nearby coarse tokens, while coarse tokens are enriched with local structure information carried by fine tokens.

Across refinement rounds, Stage 2 progressively outputs the finest-resolution tokens to form multi-scale features for dense prediction. Starting from mixed-resolution tokens at all four scales, each refinement round outputs the current finest scale (e.g., 4×4, then 8×8, then 16×16, then 32×32), while the remaining coarser tokens are passed to the next round. The final refinement round uses a ViT block to strengthen high-level semantic representations at the coarsest scale.

Cross-stage residual connections

To preserve low-level and boundary-aware information from the allocation stage, Stage 2 incorporates lateral connections from Stage 1. Before the attention block in each refinement round, we fuse the current Stage 2 tokens with a Stage 1 lateral output at the same resolution via concatenation followed by linear projection. Concretely, refinement rounds 2–4 use lateral outputs from allocation round 2, allocation round 1, and the pre-allocation round, respectively. This pre-attention fusion injects Stage 1 features into Stage 2 refinement while maintaining the mixed-resolution structure. The output finest-scale tokens from each refinement round are forwarded to the decoder as multi-scale features.

3.5 Decoder strategy

We use the point-based Mask2Former decoder [22] as in AFF [3]. A pixel decoder first processes and aligns features across scales, followed by masked cross-attention that iteratively refines a set of class queries using multi-scale token features.

Before masked cross-attention, we densify only the highest-resolution feature map. At each spatial location, we select the finest token available, replicate its feature over the corresponding region, and add learned positional embeddings for spatial differentiation. Lower-resolution feature maps remain sparse. This preserves sparse computation in early decoding while enabling dense, fine-grained updates in the final masked attention stage.

4 Experiments

4.1 Datasets

We evaluate our method on three publicly available semantic segmentation benchmark datasets: ADE20K [23], Cityscapes [24], and COCO-Stuff [25], [26]. ADE20K is a scene parsing dataset containing 20,210 images annotated with 150 fine-grained semantic classes, covering a diverse range of indoor and outdoor environments. Cityscapes is a street-level driving dataset consisting of 5,000 high-resolution images with fine annotations for 19 semantic categories. COCO-Stuff extends the COCO dataset with dense pixel-level annotations,

Table 2: Comparisons with state-of-the-art methods on ADE20K val.

Model	Params	FLOPs ↓	mIoU (SS) ↑	mIoU (MS) ↑
ViT-CoMer-T [27]	38.7M	-	43.0	44.3
SegFormer-B3 [28]	47.3M	79G	49.4	50.0
SegNeXt-L [29]	48.9M	70G	51.0	52.1
SegMAN-B [30]	51.8M	58G	**52.6**	-
Mask2Former-Swin-T [22]	46.5M	74G	47.7	49.6
AFF-Tiny-1/5 [3]	46.5M	51G	50.0	-
ARTA-Tiny	48.5M	**44±7G**	51.5	**52.6**
ViT-CoMer-S [27]	61.4M	-	46.5	47.7
VMamba-T [31]	62.0M	-	47.9	48.8
ViT-Adapter-S [32]	57.6M	-	46.2	47.1
OverLoCK-Tiny [20]	63.0M	-	50.3	-
HRFormer-B [33]	56.2M	-	48.7	50.0
SegFormer-B4 [28]	64.1M	96G	50.3	51.1
LRFormer-B [34]	69M	75G	51.0	-
Mask2Former-Swin-S [22]	66.5M	98G	51.3	52.4
AFF-Small-1/5 [3]	62.1M	67G	51.9	-
ARTA-Small	61.7M	**60.7±9.3G**	**52.1**	**53.9**
ViT-CoMer-B [27]	144.7M	-	48.8	49.4
ViT-Adapter-B [32]	133.9M	-	48.8	49.7
VMamba-B [31]	122.0M	-	51.0	51.6
ConvNeXt V2-B [2]	122.0M	-	52.1	-
OverLoCK-Base [20]	124M	-	51.7	-
SegFormer-B5 [28]	84.7M	183G	51.0	51.8
LRFormer-L [34]	113M	183G	52.6	-
SegMAN-L [30]	92.4M	97G	53.2	-
Mask2Former-Swin-B [22]	106.5M	222.7G	52.4	53.7
ARTA-Base	111.5M	**82.4±14.0G**	**53.5**	**54.6**

providing 172 semantic labels across 164,000 images.

4.2 Experimental Setup

We build on Detectron2 with components from AFF and Mask2Former, and train on NVIDIA A100 GPUs. For complete details of the experimental protocol and additional settings, see the supplementary material.

For pre-training, we use ImageNet classification. During this stage the token allocation blocks are disabled and tokens are allocated at random according to a fixed ratio schedule (no content-based selection). ImageNet classification results can be found in the supplement.

For fine-tuning, we follow AutoFocusFormer and Mask2Former for data augmentation and general hyperparameters. Models were optimized using AdamW with a learning rate of 4×10^{-5}. Crop sizes are fixed for all reported methods: 512×512 for ADE20K and COCO-Stuff, and 1024×1024 for Cityscapes. Training was conducted for 80k iterations with a batch-size of 32 on ADE20K and COCO-Stuff, and for 90k iterations on Cityscapes with a batch-size of 16. The threshold τ_r is set to [0.005, 0.01, 0.02] for the three allocation rounds.

For evaluation on ADE20K and COCO-Stuff, we resize the short side to 512 with preserved aspect ratio; for Cityscapes, we use overlapping 1024×1024 sliding-window inference. Performance is reported using mean Intersection over Union (mIoU)

FLOPs are measured for the full network at fixed input sizes: 512×512 (ADE20K/COCO-Stuff) and 1024×2048 (Cityscapes), reported as mean ± std per image over the validation set. We compute FLOPs for ARTA using this protocol, while FLOPs for other methods are taken from their respective papers. For fairness, we report FLOPs only for methods evaluated at the same input resolution.

ADE20K

We compare ARTA at three model scales against state-of-the-art baselines on ADE20K val (Table 2). At the smallest scale, SegMAN-B attains higher single-scale mIoU than ARTA-Tiny, but ARTA-Tiny is more compute-efficient. As model capacity increases, ARTA scales more favorably and at the large scale ARTA-Base surpasses SegMAN-L while using fewer FLOPs. When comparing to other encoders using a Mask2Former [22] decoder such as AFF [3] and Mask2Former with Swin backbones, we can observe that ARTA has higher mIoU and lower FLOPs over all model sizes. Across all ARTA variants, multi-scale testing further boosts performance, with ARTA-Base reaching 54.6 mIoU.

Table 3: Comparisons with state-of-the-art methods on the validation sets of COCO-Stuff and Cityscapes. "†" indicates that ImageNet22k was used for pre-training.

Model	Params	COCO-Stuff		Cityscapes	
		FLOPs ↓	mIoU ↑ (SS/MS)	FLOPs ↓	mIoU ↑ (SS/MS)
SegFormer-B3 [28]	47.3M	79G	45.5 / -	963G	81.7 / 83.3
SegNext-L [29]	48.9M	70G	46.5 / 47.2	578G	83.2 / **83.9**
SegMAN-B [30]	51.8M	58G	**48.4**/ -	479G	**83.8** / -
Mask2Former-Swin-T [22]	46.5M	-	- / -	537G	82.1 / 83.0
ARTA-Tiny	49.5M	**45±9G**	47.2 / 47.7	**286±20G**	81.8 / 82.9
HRFormer-B [33]	56.2M	280G	42.4 / 43.3	2224G	81.9 / 82.6
SegFormer-B4 [28]	64.1M	96G	46.5 / -	1241G	82.3 / **83.9**
LRFormer-B [34]	67M	75G	47.2 / -	555G	**83.0** / -
Mask2Former-Swin-S [22]	66.5M	-	- / -	732G	82.6 / 83.6
ARTA-Small	61.7M	**57±12G**	**47.6** / **48.3**	**355±25G**	81.9 / 83.2
SegFormer-B5 [28]	84.7M	112G	46.7 / -	1460G	82.4 / 84.0
LRFormer-L [34]	111.0M	122G	47.9 / -	908G	83.2 / -
SegMAN-L [30]	92.4M	97G	48.8 / -	796G	**84.2** / -
Mask2Former-Swin-B† [22]	106.5M	-	- / -	1050G	83.3 / **84.5**
ARTA-Base	111.5M	**87±19G**	**49.0** / **49.4**	**590±40G**	82.6 / 83.3

COCO-Stuff

Table 3 mirrors the trend observed on ADE20K: SegMAN is strongest at the smallest scale, whereas ARTA improves more with capacity. In the ∼50M-parameter regime, SegMAN-B achieves higher single-scale mIoU than ARTA-Tiny, but ARTA-Tiny is notably more compute-efficient. As model size increases, ARTA gains substantially more mIoU from Tiny to Base than SegMAN does from B to L. At the large scale, ARTA-Base surpasses SegMAN-L in single-scale mIoU while using fewer FLOPs on average, and further improves with multi-scale testing (Table 3).

Cityscapes

On Cityscapes, ARTA delivers competitive accuracy while using a fraction of the compute of comparable baselines (Table 3). Across model scales, ARTA attains mIoU close to the state of the art, but with substantially lower FLOPs (e.g., **ARTA-Tiny** at 286 GFLOPs vs. 479–963 GFLOPs for similar-size baselines), providing a favorable accuracy–efficiency trade-off.

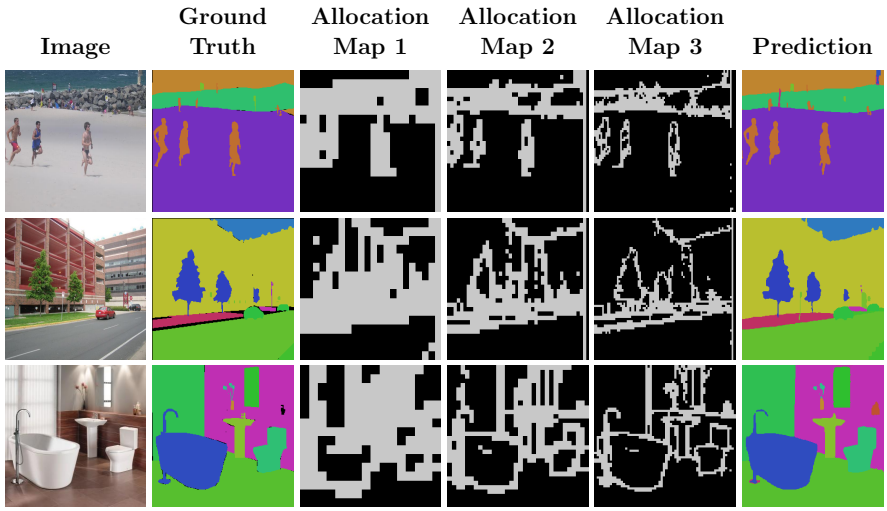


Figure 3: From left to right: original image, ground truth, 32×32 patches selected for allocation, 16×16 patches selected for allocation, 8×8 patches selected for allocation, and prediction. Black in the ground truth means that the pixel was not labeled.

Speed Analysis

We benchmark inference speed and memory usage on Cityscapes with a single NVIDIA A100 GPU. Following SegMAN [30], we report FPS averaged over 128 inference steps with batch size 2, using full-resolution inputs (1024×2048) in FP32. We report peak allocated GPU memory over the same run using PyTorch's CUDA memory statistics. As shown in Table 4, ARTA's reduced FLOPs translate to higher throughput and lower memory usage across model scales.

Qualitative Examples

Figure 3 shows qualitative results and the tokens selected for refinement at each scale. The model keeps homogeneous regions (e.g., sky, road, walls) coarse while allocating finer tokens near boundaries and small/fine structures, aligning with the goal of spending resolution where semantic complexity is high.

Table 4: FPS and peak allocated memory during inference on Cityscapes with a single NVIDIA A100.

Model	Params	FLOPs	FPS	Peak mem (GB)
SegFormer-B3 [28]	47.3M	963G	6.6	21.9
SegNeXt-L [29]	48.8M	554G	**7.3**	16.9
SegMAN-B [30]	51.8M	479G	5.6	23.8
AFF-Tiny [3]	46.5M	463G	3.8	34.4
ARTA-Tiny	49.5M	**286±20G**	6.6	**14.6**
SegFormer-B4 [28]	64.1M	1241G	4.7	29.5
AFF-Small [3]	62.1M	639G	3.6	38.7
ARTA-Small	61.7M	**355±25G**	**6.0**	**19.8**
SegFormer-B5 [28]	84.7M	1460G	4.0	34.9
SegMAN-L [30]	92.4M	796G	3.9	31.6
ARTA-Base	111.5M	**590±40G**	**4.9**	**25.4**

4.3 Ablation Studies

Without adaptive token allocation

We ablate adaptive token allocation by forcing dense tokenization: during ImageNet pre-training, every selected coarse token is always allocated finer tokens, producing dense token grids at all scales. We then fine-tune this ARTA-Tiny variant on ADE20K and compare it to the adaptive baseline under the same training setup. Removing adaptive allocation reduces accuracy and increases compute, achieving 50.4 mIoU at 74G FLOPs versus 51.5 mIoU at 44±7G for the baseline.

Choice of encoder blocks

We ablate the block type used at different locations in ARTA-Tiny, trained from scratch on ADE20K. The pre-allocation and final refinement rounds operate on coarse and dense token grids, where standard dense backbones are applicable. In contrast, all other rounds process sparse mixed-resolution token sets with substantially higher token counts, requiring layers that can handle multi-scale sparse inputs while keeping compute low. A vanilla ViT supports sparse inputs in principle, but becomes infeasible in the intermediate rounds due to its quadratic attention cost. We therefore consider two compatible

Table 5: Block-type ablation for ARTA-Tiny on ADE20K. "OOM" indicates out of memory.

Initial/Final block type	Intermediate block type	mIoU
Initial/Final Block Ablation		
Cluster Attention [3]	Cluster Attention [3]	42.1
ConvNeXt V2 [2]		41.8
Swin [35]		42.2
ViT [1]		**42.4**
Intermediate Block Ablation		
ViT [1]	ViT [1]	OOM
	MSDeformAttn [3]	42.0
	Cluster Attention [3]	**42.4**

alternatives that support sparse mixed-resolution processing with compute constraints: cluster attention and (point-based) multi-scale deformable attention (MSDeformAttn) [3]. Table 5 shows that ViT is best for the coarse dense initial/final rounds, while cluster attention performs best for intermediate mixed-resolution rounds.

Ablation: Oracle Token Allocation Scores.

We test whether injecting ground-truth class-boundary scores during training improves segmentation. Models are fine-tuned on ADE20K; the allocation blocks are always trained and, at inference, allocation uses predicted scores. An oracle rate of $x\%$ means we randomly use oracle scores for $x\%$ of training batches (and predicted scores otherwise). Results on validation set are shown in Table 6.

Oracle scores do not improve validation mIoU and higher oracle rates degrade performance. While training losses (Dice/mask) decrease when using oracle scores, we hypothesize this reflects a train–test mismatch: oracle scores may act as a privileged boundary cue during training, encouraging the model to rely on boundary information that is not available at test time. When allocation reverts to predicted scores at inference, this reliance may not transfer and performance drops.

Table 6: Oracle token allocation scores on ADE20K with ARTA-Tiny.

Oracle rate	mIoU ↑
100%	35.3
50%	48.9
10%	50.8
0%	**51.5**

Table 7: Ablation of Stage 2 and Stage 1 token initialization for ARTA-Tiny on ADE20K.

Method	mIoU ↑
Baseline (Stage 1 + Stage 2)	**42.4**
Stage 1 only	41.6
w/o aux image init	41.0
w/o feature residual	41.3

Ablation: Stage design and token initialization.

We ablate Stage 2 mixed-resolution token refinement and two token initialization choices in Stage 1 allocation, training all variants from scratch on ADE20K for 80k iterations. The baseline uses Stage 1 allocation followed by Stage 2 refinement. *Stage 1 only* removes Stage 2, reallocates depth to Stage 1 to match parameters, and adds a final cluster-attention layer (as in Stage 2) to refine newly allocated high-resolution tokens. *No aux image data* disables adding sub-patch image features when initializing newly allocated tokens. *No feature residual* removes the residual copy of the selected token feature, initializing new tokens solely from sub-patch image features (plus embeddings/MLP). As shown in Table 7, removing Stage 2 or either initialization component reduces mIoU.

5 Conclusion

We presented ARTA, a coarse-to-fine segmentation backbone that predicts semantic boundary density early and allocates additional tokens to regions needing higher spatial detail. Starting from coarse tokens and allocating finer tokens adaptively, ARTA preserves compute by avoiding fine processing in homogeneous areas while maintaining interacting mixed-resolution features across scales. ARTA achieves state-of-the-art accuracy on ADE20K and COCO-Stuff with substantially fewer FLOPs, and remains competitive on Cityscapes at a fraction of the compute of comparable baselines. These results show that content-adaptive resolution is a strong approach for accurate, efficient dense prediction. Scalability to much larger backbones and longer pre-training remains untested. Future work includes extending ARTA to 3D segmentation.

Acknowledgements. Compute and storage resources were provided by NAISS, partially funded by the Swedish Research Council (grant 2022-06725).

Supplementary Material

A Hardware

ImageNet pre-training was performed using 16 NVIDIA A100 GPUs; fine-tuning used 4 A100 GPUs, mixed-precision was enabled throughout.

B Pre-training

The model was pre-trained on the ImageNet classification task. During this phase, the token allocation blocks were disabled, and tokens were allocated at random rather than based on predicted importance. A fixed allocation ratio was applied uniformly across all spatial scales, starting at 100% (i.e., all tokens allocated) and gradually decaying to 50% by the end of training. This strategy exposed the model to dense supervision early in training while encouraging sparser representations as learning progressed, facilitating better alignment with the adaptive token allocation used during fine-tuning. For details regarding this choice, see Section F.

C Fine-tuning

Fine-tuning was performed on downstream semantic segmentation datasets using random resizing with a scale ratio between 0.5 and 2.0, random horizontal flipping, and random cropping on all datasets. Crop sizes were set to 512×512 for ADE20K, 1024×1024 for Cityscapes, and 512×512 for COCO-Stuff.

Optimization uses AdamW with backbone learning rate 4×10^{-5}, a 10× multiplier for the decoder and upsampling blocks, and weight decay 0.05. We adopt a WarmupPolyLR schedule with 2500 warmup iterations (warmup factor 0.00001).

For Cityscapes, we follow SegFormer and use three overlapping 1024×1024 windows for sliding-window inference. When multi-scale testing is used, we follow Mask2Former and resize the short side to $[256, 384, 512, 640, 768, 896]$

Table 8: Additional model hyperparameters

Parameter	ARTA-Tiny	ARTA-Small	ARTA-Base
Drop-path-rate	0.3	0.3	0.3
Drop-rate	0.0	0.0	0.0
Attn-Drop-rate	0.0	0.0	0.0
MLP-ratio	3.0	4.0	4.0
Head-Dim	32	32	32
Cluster-size	8	8	8
Neighborhood-size	48	48	48
LayerScale	0.0	1e-5	1e-5
TokenAllocationLossWeight	10	10	10
Mask2Former-MaskDim	256	256	256
Mask2Former-MLP-Ratio	8.0	8.0	8.0
Mask2Former-Head-Dim	32	32	32
Mask2Former-Layers	9	9	9
PixelDec-Dim	256	256	256
PixelDec-Layers	6	6	6

for ADE20K/COCO-Stuff and $[512, 768, 1024, 1280, 1536, 1792]$ for Cityscapes (with aspect ratio preserved) and include horizontal flips at each scale; predictions are aggregated to produce the final result.

D Additional Model Hyperparameters

In Table 8, additional hyperparameters can be found that was used to train ARTA. Note that the majority of these hyperparameters are default Mask2Former and AutoFocusFormer parameters with no changes made.

E Additional Qualitative Results

In Figure 4, additional qualitative results can be found.

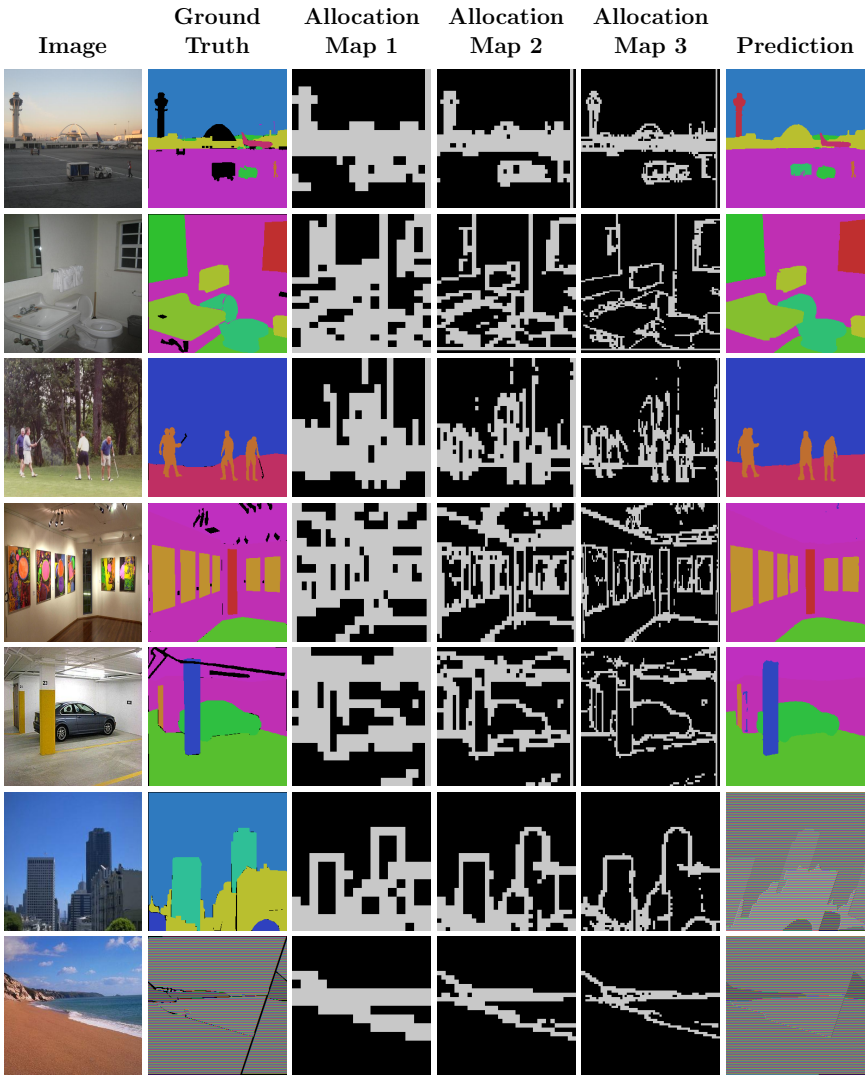


Figure 4: From left to right: original image, ground truth, 32×32 patches selected for allocation, 16×16 patches selected for allocation, 8×8 patches selected for allocation, and prediction. Black in the ground truth means that the pixel was not labeled.

Table 9: ImageNet pre-training ablation: full vs. random upsampling. Top-1 accuracy is reported on ImageNet; mIoU is evaluated after fine-tuning.

Method	**Up-ratio**	**ImageNet Acc@1**	**ADE20K mIoU**	**Cityscapes mIoU**
ARTA-Tiny	100%	81.2	50.8	80.6
ARTA-Tiny	50%	78.9	51.5	81.5

F ImageNet pre-training.

We investigate the effects of pre-training with random upsampling versus full upsampling and evaluate the final results after finetuning on ADE20K and Cityscapes. In the results in Table 9 we can see that even if the model using full upsampling performed better on imagenet (which is to be expected), by forcing the model to handle sparse tokens, downstream performance was improved.

We compare ImageNet pre-training with full upsampling versus random upsampling with upsampling blocks disabled and fixed ratio schedule as described in Appendix B. We then evaluate downstream performance after fine-tuning on ADE20K and Cityscapes. As shown in Table 9, the full-upsampling model achieves higher ImageNet accuracy, as expected since it is trained with dense high-resolution tokens throughout. However, the random-upsampling model improves downstream segmentation mIoU after fine-tuning, suggesting better robustness to sparse mixed-resolution inputs.

G Upsampling score loss ablation.

We investigate the effect of using mean-squared error (MSE) versus mean absolute error (MAE) as the loss function for the upsampling score prediction blocks. The model used is ARTA-Tiny, pretrained on ADE20K for 80K iterations. As shown in Table 10, supervising the upsampling score with MSE yields clearly better performance than MAE. This is intuitive: small deviations in the predicted score are often negligible, as they typically do not change whether a token is selected for upsampling, whereas larger errors can flip the upsampling decision. Squaring the error places more emphasis on these larger

mistakes, which leads to improved mIoU.

Table 10: Ablation study on the effect of loss function on the upsampling score.

Method	Loss Function	mIoU
ARTA-Tiny	MAE	49.2
ARTA-Tiny	MSE	51.5

References

[1] A. Dosovitskiy et al., "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2020.

[2] S. Woo et al., "ConvNeXt V2: Co-designing and scaling ConvNets with masked autoencoders," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 16 133–16 142.

[3] C. Ziwen et al., "AutoFocusFormer: Image segmentation off the grid," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 18 227–18 236.

[4] C. Lu, D. de Geus, and G. Dubbelman, "Content-aware token sharing for efficient semantic segmentation with vision transformers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 23 631–23 640.

[5] Q. Tang, B. Zhang, J. Liu, F. Liu, and Y. Liu, "Dynamic token pruning in plain vision transformers for semantic segmentation," in *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2023, pp. 777–786.

[6] Y. Rao, W. Zhao, B. Liu, J. Lu, J. Zhou, and C.-J. Hsieh, "DynamicViT: Efficient vision transformers with dynamic token sparsification," in *Advances in Neural Information Processing Systems*, vol. 34, Curran Associates, Inc., 2021, pp. 13 937–13 949.

[7] Z. Pan, B. Zhuang, J. Liu, H. He, and J. Cai, "Scalable vision transformers with hierarchical pooling," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2021, pp. 367–376.

[8] D. Bolya, C.-Y. Fu, X. Dai, P. Zhang, C. Feichtenhofer, and J. Hoffman, "Token merging: Your ViT but faster," in *The Eleventh International Conference on Learning Representations*, 2022.

[9] D. Marin, J.-H. R. Chang, A. Ranjan, A. Prabhu, M. Rastegari, and O. Tuzel, "Token pooling in vision transformers for image classification," in *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2023, pp. 12–21.

[10] S. Long, Z. Zhao, J. Pi, S. Wang, and J. Wang, "Beyond attentive tokens: Incorporating token importance and diversity for efficient vision transformers," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 10 334–10 343.

[11] S. Wei, T. Ye, S. Zhang, Y. Tang, and J. Liang, "Joint token pruning and squeezing towards more aggressive compression of vision transformers," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2023, pp. 2092–2101.

[12] M. Kim, S. Gao, Y.-C. Hsu, Y. Shen, and H. Jin, "Token fusion: Bridging the gap between token pruning and token merging," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2024, pp. 1383–1392.

[13] S. Chang et al., "Making vision transformers efficient from a token sparsification view," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 6195–6205.

[14] T. Ronen, O. Levy, and A. Golbert, "Vision Transformers With Mixed-Resolution Tokenization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 4613–4622.

[15] M. Fayyaz et al., "Adaptive token sampling for efficient vision transformers," in *Computer Vision – ECCV 2022*, S. Avidan, G. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, Eds., Cham: Springer Nature Switzerland, 2022, pp. 396–414, ISBN: 978-3-031-20083-0.

[16] L. Meng et al., "AdaViT: Adaptive vision transformers for efficient image recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12 309–12 318.

[17] W. Liu, H. Lu, H. Fu, and Z. Cao, "Learning to upsample by learning to sample," in *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023, pp. 6027–6037.

[18] L. Chen, Y. Fu, L. Gu, C. Yan, T. Harada, and G. Huang, "Frequency-aware feature fusion for dense image prediction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 12, pp. 10 763–10 780, 2024.

[19] X. Zhu et al., "Parameter-inverted image pyramid networks," in *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, Nov. 2024.

[20] M. Lou and Y. Yu, "OverLoCK: An overview-first-look-closely-next ConvNet with context-mixing dynamic kernels," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025, pp. 128–138.

[21] Y. Yuan, J. Xie, X. Chen, and J. Wang, "SegFix: Model-agnostic boundary refinement for segmentation," in *Computer Vision – ECCV 2020*, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds., Cham: Springer International Publishing, 2020, pp. 489–506, ISBN: 978-3-030-58610-2.

[22] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar, "Masked-attention mask transformer for universal image segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 1290–1299.

[23] B. Zhou et al., "Semantic understanding of scenes through the ADE20K dataset," *International Journal of Computer Vision*, vol. 127, no. 3, pp. 302–321, Mar. 2019, ISSN: 1573-1405.

[24] M. Cordts et al., "The Cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 3213–3223.

[25] T.-Y. Lin et al., "Microsoft COCO: Common objects in context," in *European Conference on Computer Vision*, Springer, 2014, pp. 740–755, ISBN: 978-3-319-10602-1.

[26] H. Caesar, J. Uijlings, and V. Ferrari, "COCO-stuff: Thing and stuff classes in context," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1209–1218.

[27] C. Xia, X. Wang, F. Lv, X. Hao, and Y. Shi, "ViT-CoMer: Vision transformer with convolutional multi-scale feature interaction for dense predictions," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 5493–5502.

[28] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "SegFormer: Simple and efficient design for semantic segmentation with transformers," in *Advances in Neural Information Processing Systems*, Nov. 2021.

[29] M.-H. Guo, C.-Z. Lu, Q. Hou, Z.-N. Liu, M.-M. Cheng, and S.-m. Hu, "SegNeXt: Rethinking convolutional attention design for semantic segmentation," in *Advances in Neural Information Processing Systems*, 2022.

[30] Y. Fu, M. Lou, and Y. Yu, "SegMAN: Omni-scale context modeling with state space models and local attention for semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025, pp. 19 077–19 087.

[31] Y. Liu et al., "VMamba: Visual state space model," *Advances in Neural Information Processing Systems*, vol. 37, pp. 103 031–103 063, Dec. 2024.

[32] Z. Chen et al., "Vision transformer adapter for dense predictions," in *The Eleventh International Conference on Learning Representations*, 2022.

[33] Y. Yuan et al., "HRFormer: High-resolution vision transformer for dense predict," in *Advances in Neural Information Processing Systems*, Nov. 2021.

[34] Y.-H. Wu et al., "Low-resolution self-attention for semantic segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 47, no. 9, pp. 8180–8192, Sep. 2025, ISSN: 1939-3539.

[35] Z. Liu et al., "Swin Transformer: Hierarchical vision transformer using shifted windows," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, Montreal, QC, Canada: IEEE, Oct. 2021, pp. 9992–10 002, ISBN: 978-1-6654-2812-5.