ROUTE-STRUCTURED PERCEPTION FOR RAILWAY SCENE UNDERSTANDING USING FORWARD-LOOKING CAMERA DATA

MOHAMMADJAVAD GHORBANALIVAKILI

A DISSERTATION SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILMENT OF THE REQUIREMENTS
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

GRADUATE PROGRAM IN THE DEPARTMENT OF EARTH AND SPACE SCIENCE
AND ENGINEERING, LASSONDE SCHOOL OF ENGINEERING

YORK UNIVERSITY
TORONTO, ONTARIO

SEPTEMBER 2026

© Mohammadjavad Ghorbanalivakili, SEPTEMBER 2026

Abstract

Railway perception is not only a problem of detecting obstacles or infrastructure elements. For safe railway automation, the important question is how scene elements relate to the route that a train can occupy. An object near the railway corridor may have different safety implications depending on whether it lies outside all reachable tracks, intersects a possible rail path, or lies on the valid ego-path. Since trains cannot steer around hazards, perception systems must represent the route structure that constrains future motion. This thesis therefore studies railway scene understanding as a structured perception problem centered on ego-path interpretation.

The thesis develops this idea through a sequence of methods for forward-looking imagery. First, path perception is formulated as a triplet-based extraction problem, where left rail, right rail, and centerline information are used to generate possible ego-path proposals in multi-track scenes. This representation connects low-level rail detection to route-level interpretation by preserving which tracks form plausible paths ahead of the train. The thesis then advances this formulation through learned ego-path instance tracing, using an attraction-field representation to reduce reliance on heuristics and to support route continuity through complex layouts.

The thesis further addresses the distinction between possible paths and the valid route. In switch scenes, several paths may be visible, but only one may correspond to the route the train is set to follow. This problem is investigated through reasoning-guided segmentation with a railway-adapted vision–language model, which is prompted to identify the valid ego-path and may generate an auxiliary rationale intended to describe switch geometry. Joint semantic and reasoning-segmentation finetuning produces higher ego-path overlap than semantic-only finetuning, with larger observed gains in switch-dependent scenes. This stage moves the

thesis from route proposal generation toward route validity interpretation.

Overall, this thesis contributes an integrated account of railway scene understanding in which ego-paths serve as enabling representations for downstream safety reasoning. The thesis identifies remaining challenges in occlusion, long-range ambiguity, switch interpretation, topology-aware evaluation, explanation quality, and dataset coverage.

Acknowledgements

Completing this PhD has been one of the most challenging and meaningful journeys of my life. It would not have been possible without the guidance, encouragement, and support of many people and organizations, to whom I am deeply grateful.

First and foremost, I would like to express my sincere gratitude to my supervisor, Professor Gunho Sohn, for giving me the opportunity to pursue my doctoral studies at York University and to become a member of the Augmented Urban Space Modelling Laboratory. His guidance, trust, and high academic standards have played an essential role in shaping both this thesis and my development as a researcher. I am particularly grateful for the freedom he gave me to explore new research directions, as well as for his continued encouragement during the most demanding stages of this work.

I would also like to thank the members of my supervisory and examination committees for their valuable time, thoughtful feedback, and constructive recommendations. Their perspectives helped strengthen this research and encouraged me to examine my work with greater depth and clarity.

My sincere appreciation goes to my colleagues and friends in the Augmented Urban Space Modelling Laboratory. The discussions, technical support, shared experiences, and moments of friendship made this journey both intellectually rewarding and personally memorable. I am especially thankful to those who contributed to data preparation, model development, experimental evaluation, and the many discussions that helped me overcome research challenges.

I gratefully acknowledge York University, the Department of Earth and Space Science and Engineering, and the Lassonde School of Engineering for providing the academic environment,

facilities, and financial support that enabled this research.

I would further like to express my gratitude to the Smart Mobility Laboratory at Hanyang University for welcoming me as a visiting researcher. This experience broadened my perspective on autonomous mobility and allowed me to explore how computer vision and intelligent perception can address practical challenges beyond the railway environment.

I am deeply thankful to all the friends I met during my years in Canada. Starting a new life in another country while pursuing a PhD was not always easy. Their companionship, kindness, and encouragement gave me a sense of belonging and helped me through many difficult moments. I am equally grateful to my longtime friends in Iran, whose support remained constant despite the distance.

I would like to express my deepest appreciation to my fiancé, Samira Dabbagh, and to my closest friends, Mohammad Moein Sheikholeslami, Seyed Mostafa Ahmadi, and Jungwon Kang. Their friendship, patience, encouragement, and constant presence have been an invaluable source of strength throughout this journey. They shared in both the difficult and joyful moments, listened when the path became uncertain, and supported me in ways that extended far beyond my academic life. I feel truly fortunate to have had them beside me during this important chapter of my life.

Most importantly, I would like to thank my family. To my parents, thank you for your unconditional love, sacrifices, patience, and unwavering belief in me. Although we were separated by great distance throughout much of this journey, your support was always present in every step I took. Everything I have achieved rests upon the opportunities, values, and strength you gave me. I also extend my heartfelt gratitude to the rest of my family for their encouragement and for always reminding me that I was never alone.

This thesis represents far more than the completion of a research project. It reflects years of learning, uncertainty, persistence, personal growth, and the collective support of everyone who accompanied me along the way. To all those who contributed to this journey, directly or indirectly, I offer my deepest and most sincere thanks.

Table of Contents

Abstract **ii**

Acknowledgements **iv**

Table of Contents **vi**

List of Tables **xii**

List of Figures **xv**

1 Introduction **1**

1.1 Background and Motivation 2

1.2 Significance of Rail-Path and Ego-Path Perception 3

1.3 Railway Scene Understanding as a Perception Problem 6

1.4 Challenges in Rail-Path and Scene Interpretation 6

1.5 From Geometric Perception to Reasoning-Oriented Perception 8

1.6 Research Gap and Thesis Objectives 9

1.7 Thesis Contributions 10

1.8 Thesis Organization 10

2 Literature Review **12**

2.1 Railway Scene Understanding as Structured Perception 12

2.2 Camera-Based Rail Detection, Segmentation, and Obstacle Perception 14

2.3 Rail Path, Ego-Path, and Instance-Level Route Extraction 16

2.4 Learning Route Continuity and Generalization 18
2.5 Vision-Language and Reasoning-Guided Segmentation for Route Validity . . 19
2.6 Datasets, Annotations, Metrics, and Evaluation Gaps 21
2.7 Research Gap and Thesis Positioning 23

3 TPE-Net for Multiple-Track Rail Path Association 25
3.1 Chapter Overview 25
3.2 Problem Formulation 26
3.3 Rail Path Representation and Methodology 27
3.3.1 System Overview 27
3.3.2 Triplet-based Path Representation 28
3.3.3 Triple-Location Regression 29
3.3.4 Location-Radius Regression 29
3.3.5 TPE-Net Architecture 32
3.3.6 Track Segment Generation 33
3.3.7 All Possible Ego-path Generation 34
3.3.8 Post-Processing and Visualization 35
3.3.9 Loss Function 37
3.4 Datasets and Experimental Setup 38
3.4.1 RailSem19 38
3.4.2 RailSem19+ 39
3.4.3 YDHR Track Image Dataset 39
3.4.4 Training Setup 40
3.4.5 Evaluation Protocol 40
3.5 Results 42
3.5.1 Regression Design and Post-processing Ablation 42
3.5.2 RailSem19 and RailSem19+ Baseline Comparison 44
3.5.3 YDHR Evaluation 45
3.5.4 Comparison with Prior Work 47

3.6 Limitations, Future Directions, and Discussion 48
3.6.1 Scope and Methodological Limitations 48
3.6.2 Future Directions 48
3.6.3 Role in the Thesis 49

4 TRIT-Net for Railway Ego-path Candidate Tracing 50
4.1 Chapter Overview 50
4.2 Motivation Beyond Chapter 3 51
4.3 TRIT-Net Framework 54
4.3.1 Triplet Representation and Overall Framework 54
4.3.2 Centerline Regression Map 55
4.3.3 Attraction Field Map 55
4.3.4 Ego-path Candidate Construction 57
4.4 Network Architecture and Training Strategy 62
4.4.1 TRIT-Net Architecture 62
4.4.2 Self-Supervised Pretraining 65
4.4.3 Fine-Tuning for Ego-Path Extraction 67
4.5 Datasets, Metrics, and Training Setup 68
4.5.1 Evaluation Metrics 68
4.5.2 Datasets 70
4.5.3 Training Setup 71
4.6 Results and Analysis 73
4.6.1 Cross-Backbone Comparison on RailSem19 73
4.6.2 AFM-Based Tracing Versus Spatial Clustering 74
4.6.3 Multi-Dataset Evaluation 77
4.6.4 Zero-Shot Generalization 79
4.6.5 Attraction-Intensity Ablation 80
4.6.6 Qualitative Analysis and Failure Cases 81
4.6.7 Result Synthesis 82

4.7 Limitations and Chapter Discussion 83
4.7.1 Severe occlusion 83
4.7.2 Extreme long-range ambiguity 83
4.7.3 Atypical switch geometries 84
4.7.4 Decoder topology and output growth 84
4.7.5 Private-dataset scope and reproducibility 84
4.7.6 Evaluation-protocol scope 85
4.7.7 Deployment evidence 85
4.7.8 Future directions 86

5 Reasoning-Guided Ego-path Segmentation with Vision-Language Models 87
5.1 Chapter Overview 87
5.2 Motivation Beyond Earlier Camera-Based Chapters 88
5.3 Positioning in Prior Work 89
5.3.1 Railway Scene Understanding and Ego-path Perception 89
5.3.2 Vision-Language Models for Language-Conditioned Perception 89
5.4 Reasoning-Guided Segmentation Framework 90
5.4.1 Task Formulation 90
5.4.2 Railway Adaptation of LISA 91
5.4.3 Railway-Specific Prompting and Supervision 94
5.5 Dataset Construction and Finetuning Strategy 95
5.5.1 Railway Reasoning Dataset 96
5.5.2 Semantic Segmentation Stream 99
5.5.3 Image Preprocessing for the CLIP and SAM Branches 99
5.5.4 Finetuning Setup 100
5.5.5 Validation Protocol 102
5.5.6 Inference Workflow 103
5.6 Experimental Results 104
5.6.1 Evaluation Setup 104

5.6.2 Qualitative Results . . . 106
5.6.3 Quantitative Results . . . 106
5.7 Limitations and Future Directions . . . 113
5.7.1 Limitations of the Current Study . . . 113
5.7.2 Future Research Directions . . . 115
5.8 Chapter Conclusions . . . 116

6 Discussions **118**
6.1 Overview of Thesis-Level Findings . . . 118
6.2 From Camera-Based Perception to Reasoning Perception . . . 119
6.3 Role of Vision–Language Reasoning in Addressing Earlier Limitations . . . 120
6.4 Cross-Chapter Comparison of Assumptions, Methods, Outputs, and Limitations 121
6.5 Practical Implications for Railway Scene Understanding . . . 122
6.6 Overall Scientific Contribution of the Thesis . . . 123
6.7 Remaining Limitations and Open Problems . . . 124

7 Conclusion and Future Work **126**
7.1 Restatement of the Thesis Problem . . . 126
7.2 Final Answers to the Thesis Research Questions . . . 127
7.3 Summary of Main Contributions . . . 128
7.4 Final Reflection on Camera-Based Rail Perception . . . 128
7.5 Limitations of the Thesis . . . 129
7.6 Future Work . . . 129

References **131**

Appendices **147**

A List of Publications and Manuscripts **148**
A.1 Published Papers . . . 148

A.2 Submitted Manuscript 149
A.3 Manuscript Under Review 149

List of Tables

3.1 Performance of HarDNet-based TPE-Net designs on 1,500 randomly selected RailSem19 test images, separated into single-track and multi-track scenes. A check mark indicates that the segmentation output is used during post-processing. 42

3.2 Overall performance of four TPE-Net baselines on 1,500 randomly selected RailSem19 test images. Each baseline uses 3-class segmentation and location-radius regression and is trained on either RailSem19 or RailSem19+. 44

3.3 Performance of four TPE-Net baselines on 1,500 randomly selected RailSem19 test images, separated into single-track and multi-track scenes. Each baseline uses 3-class segmentation and location-radius regression and is trained on RailSem19 or RailSem19+. 45

3.4 Overall performance on 1,316 YDHR images using TPE-Net baselines trained on RailSem19+ with 3-class segmentation and location-radius regression. . . 45

3.5 Performance on 1,316 YDHR images separated into single-track and multi-track scenes. All TPE-Net baselines use 3-class segmentation and location-radius regression and are trained on RailSem19+. 47

4.1 Layer and stage configurations used in the TRIT-Net encoder–decoder architecture. 65

4.2 Public datasets used for ego-path candidate extraction, including their image counts, annotation types, and principal forms of scene diversity. 70

4.3 Cross-backbone comparison on RailSem19. Each model includes the centerline-regression and AFM branches and is trained and evaluated on RailSem19. . . 73

4.4 Desktop-workstation inference efficiency. GPU and CPU times are separately measured stage means. Their arithmetic sum and reciprocal throughput are compute-stage estimates rather than synchronized end-to-end measurements. Loading, preprocessing, CPU–GPU transfers, and visualization are excluded. Memory is the highest whole-process value displayed by Mission Center during an approximately one-minute activity window and includes more than the post-processor alone. 74
4.5 Within-dataset ego-path extraction results on RailSem19, RailDB, RailSet, and the Private Track Image Dataset. The dataset column identifies both the training and evaluation dataset. TPE-Net denotes centerline regression followed by spatial-clustering-based path construction. 77
4.6 Paired image-level uncertainty analysis for SSL-pretrained TRIT-Net with AFM relative to TPE-Net with spatial clustering on the 113 RailSem19 multi-candidate test images. Differences and intervals are reported in percentage points; positive values favour TRIT-Net. Intervals are based on 10,000 paired percentile-bootstrap resamples. 78
4.7 Zero-shot evaluation on RailSet. All methods are trained exclusively on RailSem19 and evaluated on RailSet without target-domain fine-tuning. TPE-Net denotes centerline regression followed by spatial-clustering-based path construction. 79
4.8 Attraction-intensity ablation on RailSem19. The Attraction Intensity Map is replaced by the constant value 1. Entries are performance differences relative to the complete model; negative values denote degradation. 80

5.1 RailSem19 data composition and encoder-specific image preprocessing. . . . 96
5.2 Distribution of the 242 unique reasoning-training images by switch topology, active-route side, and view type, before stochastic counterfactual flipping. . . 97

5.3 Ego-path overlap under the reasoning-oriented ego-route prompt and the generic track-bed prompt, separated by whether route determination depends on a downstream switch. All values are percentages computed against the ground-truth ego-path mask. 107
5.4 Paired image-level uncertainty analysis for joint semantic and reasoning finetuning relative to semantic-only finetuning under the reasoning-oriented ego-route prompt. Differences and intervals are reported in percentage points; positive values favour joint finetuning. Intervals are based on 10,000 paired percentile-bootstrap resamples. Differences are computed from unrounded scores and may differ slightly from subtraction of the rounded scores in Table 5.3. . . . 109
5.5 Aggregate mask and route-logic rationale assessment on the balanced 30-image switch-dependent test subset. 112

6.1 Thesis-level comparison of the technical chapters. 122

List of Figures

1.1 Thesis research progression. Chapters 3–5 form a camera-based sequence from possible ego-path extraction to learned instance tracing and reasoning-guided valid-route segmentation. 2

3.1 Railway scene captured by a forward-facing train-mounted camera. Green trajectories indicate possible ego-paths, while red trajectories denote rail paths that are not reachable by the train. Yellow boxes identify example rail-scene objects for context. Object detection is not part of the task addressed in this chapter. 26

3.2 Overview of the TPE-Net rail path extraction pipeline. The network first detects and associates left, right, and center rail pixels. Track segments are then formed inside strips, after which a path tree is constructed to encode all potential ego-paths. The final stage discards invalid or redundant paths and fits polynomials to the retained rail pixels. 27

3.3 Triplet-based rail path representation. In sampled rows of the image, each color group denotes one triplet consisting of right, center, and left rail pixels. At this stage, no inter-row association has yet been performed. 28

3.4 Conceptual difference between object-level keypoint triplets and the TPE-Net row-wise rail path representation. In the latter, each rail path is represented by one triplet per overlapping image row rather than by one triplet for the whole object. 28

3.5 Ground-truth illustration for triple-location regression, showing a sample rail scene together with its center, right-distance, and left-distance regression targets and the corresponding row-wise value trends. 30
3.6 Ambiguity of triple-location regression in overlapping rail regions. A pixel in a shared switch region may correspond to multiple rail distances, which cannot be represented faithfully by one scalar value in a single-channel distance map. 30
3.7 Ground-truth illustration for location-radius regression. The figure shows the single regression map and its value profile along a sample row, where peak locations indicate center track points and peak values encode equal left-right rail distance. 31
3.8 TPE-Net architecture. The network supports two regression-branch designs, one for triple-location regression and one for location-radius regression, alongside the shared segmentation branch. 32
3.9 Examples of shortcomings in the original RailSem19 labels: incorrect left-right association for a path instance and inconsistency in the start and end coordinates of the paired rails. 39
3.10 Comparison of location-radius regression (green) and triple-location regression (blue) using the HarDNet baseline on the RailSem19 test set. The configurations use (a) 3-class segmentation with segmentation-assisted post-processing, (b) 3-class segmentation without segmentation-assisted post-processing, (c) 19-class segmentation with segmentation-assisted post-processing, and (d) 19-class segmentation without segmentation-assisted post-processing. 43
3.11 Qualitative rail path extraction results on RailSem19 and YDHR using the HarDNet baseline trained on RailSem19+ with location-radius regression, 3-class segmentation, and segmentation-assisted post-processing. Each example includes the input image, regression output, segmentation output, and final extracted paths under varied weather, illumination, and switch conditions. . 46

4.1 The earlier approach can generate invalid track-point candidates in overlapping switch regions and relies on spatial clustering to construct ego-path candidates. The AFM instead represents shared track-bed regions explicitly and supports learned row-wise tracing. 52
4.2 Overview of the TRIT-Net framework. The network predicts a Centerline Regression Map (R), an Attraction Intensity Map (S), and an Attraction Direction Map (D), which are then used to recover ego-path candidates. . . . 53
4.3 Conceptual illustration of track-point tracing. The yellow point denotes the starting point and the red points denote traced track points in successive rows. 56
4.4 Attraction Field Map visualization in a sample turnout region. The learned field guides pixels toward path centerlines while supporting branching behavior in overlapping regions. 58
4.5 Overall TRIT-Net architecture. The encoder–decoder backbone combines convolutional processing, transformer-style Encoder Stages, and Harmonic Dense Blocks before separate task heads predict R, D, and S. The numbers below the convolutional layers indicate their output-channel counts. 62
4.6 Architecture of a TRIT-Net Encoder Stage. A convolutional patch-embedding layer with kernel size p and stride s is followed by layer normalization and repeated Encoder Blocks. 63
4.7 Architecture of a TRIT-Net Encoder Block, comprising Efficient Multi-Head Attention and a Mix Feed-Forward Network with pre-normalization and residual connections. 64
4.8 Architecture of a Harmonic Dense Block. The block uses Convolution–Batch Normalization–ReLU layers with harmonic channel growth and logarithmic feature reuse. 64
4.9 VICReg-based self-supervised pretraining architecture. The projection-head parameter is set to $f = 4096$, while d_{in} and d_{out} denote the input and output dimensions of a fully connected layer, respectively. 65

4.10 Representative raw input images from the Private Track Image Dataset, showing variations in track layout, switch regions, and vehicle-route viewpoints at the collection site. No annotation or model-prediction overlays are displayed. 71

4.11 AFM-based tracing versus spatial-clustering-based path construction on RailSem19 for TPE-Net and SSL-pretrained TRIT-Net. M1–M4 report TP-track pixel-level performance, M5–M8 report all-track pixel-level performance, and M9–M12 report track-level performance. Within each four-metric group, odd-numbered metrics correspond to single-candidate scenes and even-numbered metrics to multi-candidate scenes; the first two metrics report AP and the final two report AR. 76

4.12 Qualitative comparison comprising 20 model-output panels. Rows show, from top to bottom, RailSem19, RailDB, RailSet, and the Private Track Image Dataset. Columns show, from left to right: SSL-pretrained TRIT-Net with AFM-based tracing; SSL-pretrained TRIT-Net with spatial-clustering post-processing; TRIT-Net trained from random initialization with AFM-based tracing; SSL-pretrained TRIT-Net with the Attraction Intensity Map removed; and TPE-Net with spatial clustering. Each row holds the input scene fixed across the five configurations. True-positive predicted ego-path candidates are shown as distinct coloured rail boundaries, and the ground-truth track bed is shown in blue. The visualizations display raw extracted rail pixels before moving-average smoothing. 81

5.1 Original LISA architecture for reasoning segmentation, reproduced from [65]. In the illustrated LISA configuration, the SAM image encoder and LLaVA vision backbone are frozen, while LoRA is applied to the language-model attention projections [64, 63, 66]. The railway adaptation retains this architecture but uses the selective training strategy described in the accompanying text. . . . 92

5.2 Flowchart for multimodal finetuning. The figure illustrates how semantic segmentation and railway reasoning-segmentation samples are mixed during training, together with the role of the semantic-sampling probability P_s and the explanatory-sampling probability P_e. Counterfactual image–language augmentation, which precedes the illustrated encoder paths, is described in the text. 93
5.3 Sample images from the railway reasoning-segmentation dataset, illustrating switch configurations under different fields of view. 98
5.4 Native-resolution distribution of the 242 reasoning-training images before branch-specific preprocessing. Each point represents one image; points with identical dimensions coincide, including 116 samples at 1024×1024. 100
5.5 Mask-overlap validation trajectories for joint semantic and reasoning-segmentation finetuning. Validation is performed every 50 optimizer updates on the 500-image held-out validation set; the horizontal axis reports cumulative optimizer-update steps. 103
5.6 Qualitative results of the rail-finetuned LISA model on 1024×1024 horizontally centered, bottom-aligned RailSem19 test crops using the query "By examining rail continuity and switch geometry, segment the active ego-route and explain why." Each example presents the predicted ego-route mask together with the generated rationale, illustrating both coherent predictions and remaining mask or route-logic limitations. 105

Chapter 1

Introduction

Railway environments contain a distinctive form of scene structure. Unlike general road scenes, where many objects may move freely within a shared drivable region, train movement is constrained by rails, switches, and the topology of the track network. A perception system for railway applications must therefore identify not only visible scene components, but also the route structure that makes those components meaningful for train movement. In multi-track scenes, this requires understanding which rails form a track, which tracks form possible ego-paths ahead of the train, and which route is valid when switch geometry creates alternatives.

This thesis studies that problem through a sequence of camera-based railway perception contributions using forward-looking RGB imagery captured from train-mounted or rail-facing viewpoints. It begins with extraction of possible ego-paths from images, continues with learned tracing of ego-path instances, and then moves toward reasoning-guided valid-route segmentation using vision–language models. The thesis therefore develops railway scene understanding as an image-domain ego-path interpretation problem that progresses from structured extraction to route-validity reasoning. Figure 1.1 summarizes this thesis-level

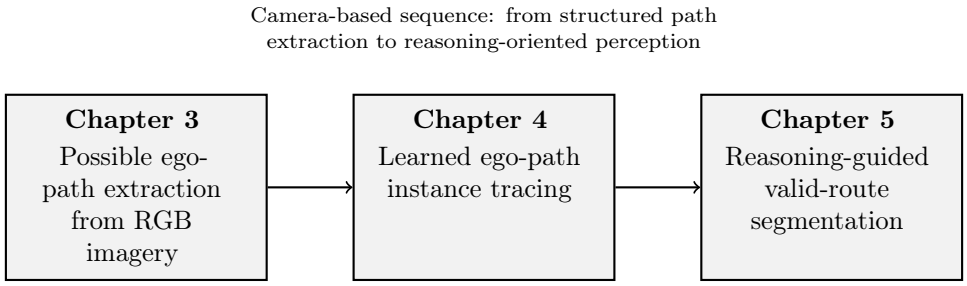


Figure 1.1 Thesis research progression. Chapters 3–5 form a camera-based sequence from possible ego-path extraction to learned instance tracing and reasoning-guided valid-route segmentation.

progression.

1.1 Background and Motivation

Railway perception is relevant to safety, automation, inspection, and operational decision support because downstream railway decisions depend on the correct interpretation of track geometry, route availability, and the objects or infrastructure elements that lie along the train's possible path [1]. For an automated or decision-support system, detecting rails as image regions is not sufficient if the system cannot determine how those rails combine into route instances. Similarly, recognizing a switch as a scene element is incomplete if the perception output does not preserve the route alternatives created by that switch [2].

The completed chapters of this thesis focus on complex rail scenes in which multiple tracks, turnouts, merging routes, or switch regions make simple semantic labeling inadequate. In these settings, a rail-pixel mask may show where rails are located, but it may not indicate which left and right rails belong together or which path the train can plausibly follow. The thesis therefore treats railway scene understanding as a structured perception problem. The relevant output is not only a collection of detected pixels or components, but a representation of rail pairing, route continuity, path candidates, and, in the reasoning-oriented chapter, route validity.

The primary intended application is a forward-looking perception module within an autonomous, semi-autonomous, or driver-assistance railway system. In complex multi-track

and switch scenes, this module provides the route structure needed by downstream object detection, risk assessment, and operational decision-making: Chapters 3 and 4 identify the possible paths available ahead of the train, while Chapter 5 considers which visible route is valid under the observed switch geometry. This information is needed because rail or obstacle detections alone cannot determine whether an object lies on the train's route, on an alternative branch, or on a parallel track.

Camera-based perception is a natural starting point for this problem because forward-looking imagery is directly aligned with the train's direction of travel and can capture the visual context of the route ahead. Existing railway vision studies include rail segmentation, rail detection, obstacle detection, and topology-aware track detection [3, 4, 5, 6, 7, 8]. However, camera-based perception also has important limitations. The chapters in this thesis repeatedly identify sensitivity to occlusion, long-range ambiguity, overlapping track structures, unusual switch geometries, and dataset-specific evaluation constraints. These limitations motivate the thesis progression from geometric and instance-level path extraction toward reasoning-oriented perception.

1.2 Significance of Rail-Path and Ego-Path Perception

The focus of this thesis on rail paths and ego-paths is motivated by the safety role that route structure plays in railway perception. In a railway scene, the most safety-relevant question is not only whether rails, objects, or obstacles are visible, but whether they are geometrically and topologically related to a route that the train can actually occupy. A detected object near the railway corridor may be harmless if it lies outside all reachable rail paths, operationally relevant if it intersects a possible path, and immediately critical if it lies on the valid ego-path of the train. Therefore, a perception system that only produces rail pixels or generic obstacle detections remains incomplete for downstream decision-making unless it also identifies the route structure that determines where the train can move [9].

This distinction is especially important because train motion is constrained by the rail

infrastructure. Unlike road vehicles, trains cannot freely steer around an obstacle once a conflict is detected. The available motion of the train is defined by the connected rail route ahead, including the current track, possible diverging or merging tracks, and the state of switches. As a result, collision-risk assessment depends on knowing which physical space belongs to the train's future route. If the perception system confuses a parallel track with the ego-path, it may either generate false alarms for objects that are not on the train's route or fail to identify an actual hazard on the route that the train will follow. The perception output must therefore support a route-aware interpretation of the scene rather than a purely semantic description of rail components.

The notion of a path proposal provides a useful way to connect low-level perception with higher-level autonomy and driver-assistance functions. In autonomous driving, path proposals have been defined as routes that a driver or vehicle would be expected to take through a given traffic configuration, and they have been used as an intermediate representation between image understanding and planning [10]. Related work on ego-vehicle path prediction also shows that predicting possible future paths can support driver-assistance tasks by providing structured hypotheses about where the vehicle may move next [11, 12]. The same principle is even more constrained in railway environments, because motion is restricted to rails, path proposals should correspond to physically plausible rail routes rather than arbitrary drivable regions.

In railway vision, this route-aware representation is necessary because many challenging scenes contain more than one visible track. Multi-track corridors, turnouts, crossings, and merging regions can contain several rail pairs within the same image. A conventional rail segmentation mask may correctly identify the rail pixels while still failing to answer three essential questions: which left and right rails form the same track, which track instances form possible routes ahead of the train, and which route is valid under the current switch configuration. Recent work on train ego-path detection has explicitly argued that railway perception should identify the immediate track that the train is set to follow, rather than treating all visible rails uniformly [13]. Similarly, the rail path proposal formulation in

TPE-Net emphasizes the need to extract topological information about possible rail routes ahead of the train so that obstacles can be interpreted with respect to the train's potential movement [14].

Rail-path proposals are therefore not an end goal by themselves. They are an enabling representation for downstream safety reasoning. Once possible ego-paths are extracted, a collision-risk module can evaluate whether a detected object, pedestrian, vehicle, or infrastructure element lies on or near a route that the train may occupy. This enables risk to be assessed in route coordinates rather than only image coordinates. In such a formulation, the perception system provides candidate rail routes, and the downstream decision system can combine these route hypotheses with object locations, train speed, braking constraints, signalling information, and operational rules to determine whether a warning, braking action, or automation intervention is required. Prediction-based collision-risk studies in intelligent transportation similarly show that future path or trajectory hypotheses are central to estimating time-to-collision, spatial overlap, and collision probability over a prediction horizon [15, 16].

This thesis therefore treats ego-path extraction as a structured perception problem positioned between semantic scene understanding and safety-critical decision-making. The camera-based chapters first address the generation of possible rail path proposals, because downstream systems must know the set of route hypotheses before they can reason about hazards along them. The later reasoning-guided chapter then addresses a more selective question: when several path proposals are visible, which one is the valid route for the train? This progression reflects the thesis-level argument that railway perception must move beyond detecting rails as visual objects and toward representing the route structure that makes collision-risk assessment and timely intervention possible.

1.3 Railway Scene Understanding as a Perception Problem

The central perception problem considered in this thesis is route-aware interpretation of railway scenes. In the camera-based chapters, the input is a forward-looking RGB image and the desired interpretation concerns the rail path or paths ahead of the train. In Chapter 3, the task is to extract all possible ego-paths and associate the left and right rail boundaries belonging to each path. In Chapter 4, the same broad task is reformulated as learned instance tracing through an Attraction Field Map. In Chapter 5, the task changes from candidate-path extraction to valid-route segmentation conditioned by a natural-language query.

Although camera calibration and train position and orientation would be useful in a deployed railway perception system, this thesis deliberately studies whether forward-looking RGB imagery alone contains sufficient visual evidence for ego-path extraction and route interpretation. Accordingly, the methods in Chapters 3–5 do not use camera intrinsics, calibrated pose, or train position and orientation during inference, and their reported performance reflects this RGB-based visual-input setting.

This progression reflects a distinction that is important throughout the thesis. Rail detection, rail segmentation, path proposal generation, and valid-route reasoning are related but not identical tasks. A method may correctly identify rail pixels while failing to associate them into tracks. It may also recover all plausible paths while leaving unresolved which path is physically valid through a switch. The thesis therefore uses the term *railway scene understanding* to refer to structured interpretation of rail scenes, especially where multi-track geometry and switch states create ambiguity.

1.4 Challenges in Rail-Path and Scene Interpretation

Several challenges recur across the completed chapters. The first is left–right rail association. A track is not a single rail, but a paired structure. In a scene with several visible rails, the

perception system must determine which rail boundaries form the same path. This issue motivates the triplet-based representation used in Chapter 3, where left rail, center track point, and right rail information are predicted and then assembled into possible ego-path proposals.

The second challenge is route continuity through switches and overlapping track regions. Switch scenes can contain shared track-bed regions, diverging or merging rails, and candidate paths that are visually close to one another. Chapter 4 addresses this issue by replacing strip-wise spatial clustering with learned triplet tracing. The Attraction Field Map is used to guide the local continuation of path instances, so that route construction depends less on hand-crafted post-processing and more on learned path-continuation cues.

The third challenge is valid-route ambiguity. The early camera-based chapters intentionally extract all possible ego-paths rather than selecting a unique route by explicit switch-state reasoning. This is a useful and well-defined perception boundary, but it leaves open a higher-level question: when several paths are visible, which route is valid for the train? Chapter 5 addresses this unresolved ambiguity by adapting a vision–language segmentation model to predict a valid ego-path mask and, when prompted, generate an auxiliary rationale intended to describe visible switch geometry.

The fourth challenge is robustness under varying scene and imaging conditions. Camera images provide rich appearance information but may be affected by occlusion, viewpoint, illumination, and long-range ambiguity. The reasoning-guided formulation also introduces sensitivity to prompt design and the possibility of mask–rationale inconsistency. These limitations require the reported findings to be interpreted within the datasets, task definitions, and evaluation protocols used by the individual chapters.

1.5 From Geometric Perception to Reasoning-Oriented Perception

The intellectual arc of the thesis moves from geometric and structural extraction toward reasoning-oriented perception. Chapter 3 establishes the first complete camera-based formulation in the thesis. It treats railway perception as extraction of all plausible ego-paths from forward-looking RGB imagery and introduces a triplet-based formulation for associating left and right rails into route-level path proposals.

Chapter 4 keeps the same candidate-path boundary but changes how path instances are constructed. Instead of using network prediction followed by strip-wise clustering and path-tree generation, it formulates route construction as learned tracing through an Attraction Field Map. This chapter therefore advances the thesis from detecting ingredients of rail paths toward learning the local continuation behavior that supports coherent path instances.

Chapter 5 changes the question again. It recognizes that candidate-path extraction, even when improved, does not determine the valid route through a switch. The chapter therefore introduces reasoning-guided segmentation with a vision–language model. The motivation is not that a vision–language model replaces geometric perception, but that valid-route selection involves switch evidence such as visible route continuity, blade–stock contact, and rail gaps. The VLM formulation allows the route-validity question to be posed explicitly through language and associates the predicted mask with an optional auxiliary rationale. A controlled comparison with semantic-only finetuning evaluates prompt-conditioned ego-path overlap separately in switch-independent and switch-dependent scenes.

This transition also introduces new limitations. A targeted 30-image audit shows that mask and rationale correctness may not be reliably coupled. Ordinary overlap metrics may not fully capture wrong-route errors in overlapping track scenes, and topology is not explicitly enforced by the training losses. The thesis therefore treats reasoning-oriented perception as an important extension of the camera-based sequence, not as a final solution to railway route understanding.

1.6 Research Gap and Thesis Objectives

The research gap addressed by this thesis is the lack of a sufficiently structured treatment of railway perception in multi-track and switch-rich scenes. Generic rail segmentation or component detection alone does not preserve the route information needed to interpret possible train movement. Earlier camera-based path extraction must also confront the limitations of heuristic post-processing, route ambiguity, occlusion, long-range uncertainty, and dataset constraints. Reasoning-capable models create a possible route toward valid-route interpretation, but the completed work shows that rationale grounding, topology, and route-specific evaluation remain unresolved.

The overall objective of the thesis is therefore to develop and analyze camera-based railway perception methods that make route structure increasingly explicit, from possible-path extraction and learned instance tracing to reasoning-guided valid-route interpretation. This objective is expressed through the following research questions:

1. How can forward-looking camera imagery be used to extract possible railway ego-paths while preserving left–right rail association and route-level structure?

2. How can camera-based ego-path extraction reduce dependence on heuristic route assembly and instead learn path-instance continuity in complex rail scenes?

3. How can route-validity interpretation in switch scenes be formulated as a reasoning-guided vision–language perception problem?

These questions distinguish candidate-path extraction, learned path tracing, and route-validity reasoning as related but separate stages of the camera-based research progression. Together, they support a coherent thesis-level argument while preserving the distinct task definition addressed by each completed study.

1.7 Thesis Contributions

The thesis makes three main contributions.

1. It formulates camera-based railway ego-path extraction as a route-structured perception problem and develops a triplet-based representation for associating left and right rails into possible path proposals. This contribution is developed in Chapter 3.

2. It advances camera-based ego-path extraction through learned instance tracing with an Attraction Field Map, reducing reliance on heuristic route assembly and expanding evaluation across multiple datasets and training settings. This contribution is developed in Chapter 4.

3. It introduces reasoning-guided valid-route segmentation for railway switch scenes by adapting a vision–language segmentation model to produce route masks and optional auxiliary rationales conditioned by natural-language queries. The contribution is evaluated through a controlled semantic-only comparison, switch-dependent stratification, and a targeted mask–rationale consistency audit in Chapter 5.

Taken together, these contributions develop a thesis-level account of railway scene understanding that moves from structured visual path extraction and learned instance tracing to reasoning-guided route interpretation.

1.8 Thesis Organization

Chapter 1 introduces the motivation, research gap, objectives, contributions, and structure of the thesis. Chapter 2 synthesizes relevant work on railway perception, camera-based rail-path understanding, and vision–language reasoning.

Chapter 3 presents the TPE-Net contribution. It establishes the first camera-based problem formulation in the thesis by extracting all possible ego-paths from forward-looking RGB imagery and associating left and right rails into route-level path proposals.

Chapter 4 presents the TRIT-Net contribution. It builds on the TPE-Net formulation by replacing heuristic route assembly with learned triplet tracing through an Attraction Field Map and by broadening the empirical evaluation across datasets and training settings.

Chapter 5 presents the reasoning-guided vision–language contribution. It shifts the camera-based storyline from candidate-path extraction toward valid-route segmentation in switch scenes and examines the relationship between predicted masks and auxiliary textual rationales.

Chapter 6 synthesizes the three technical chapters into a thesis-level argument, comparing their assumptions, methods, outputs, and remaining limitations. Chapter 7 concludes the thesis by restating the research problem, summarizing the contributions, and identifying future work in stronger VLM reasoning, uncertainty estimation, dataset development, and real-world validation.

Chapter 2

Literature Review

This chapter reviews the literature relevant to railway scene understanding and positions the thesis within the broader development of structured railway perception. The discussion is organized thematically, progressing from camera-based rail detection, segmentation, and obstacle perception to instance-level route extraction, learned route continuity, and reasoning-guided route interpretation. It then considers available datasets, evaluation practices, and the research gaps addressed by the thesis.

2.1 Railway Scene Understanding as Structured Perception

Railway perception differs from general scene perception because the relevant visual and geometric evidence is constrained by track infrastructure. A rail scene contains visible components such as rails, track beds, switches, wayside objects, and possible obstacles, but the operational meaning of these components depends on how they define routes for train motion. This dependence is reflected in methods that reconstruct track topology to determine the relevant obstacle-detection region and in systems that combine track and switch detection to estimate the railway ego track [17, 18]. For this thesis, the central distinction is therefore between detecting rail-related components and preserving the route structure that gives those components operational meaning.

Several terms recur across the thesis. A *track* is a paired structure consisting of a left

rail and a right rail. A *track point* is a center point between the paired rails, and the set of track points forms a path or route centerline. A *track bed* or rail area is the ground region bounded by the left and right rails. A *switch region* enables transition between tracks. Turnout and merging configurations are cases where multiple route candidates can coexist. These definitions motivate a structured interpretation problem: a perception system must determine not only where rails are visible, but also which rails form a track and which track or tracks can be followed.

This problem becomes more important as railway automation moves from centrally controlled automatic operation toward systems that require local situational awareness. Railway-automation literature describes Grade of Automation Level 4 operation, Automatic Train Control, and autonomous railway operation as settings in which obstacle detection, movement authority, and local perception are tightly connected [19, 20, 21, 22]. The completed thesis does not propose a full train-control architecture, but it studies a perception layer that could support route-aware interpretation of the scene ahead.

The railway computer-vision literature includes several task formulations: obstacle detection, rail or track-bed segmentation, rail line detection, topology-guided track association, and ego-path detection [23, 24, 8, 13]. Public resources increasingly reflect this breadth. RAIL-BENCH organizes railway perception into separate challenges for rail-track detection, object detection, vegetation segmentation, multi-object tracking, and visual odometry, with task-specific evaluation protocols [25]. These tasks are related, but they are not interchangeable. A rail segmentation mask may identify rail pixels without separating route instances, while an obstacle detector may identify wayside objects without determining whether those objects lie along a valid route. Chapter 1 therefore frames railway scene understanding as route-aware interpretation rather than as generic component detection.

2.2 Camera-Based Rail Detection, Segmentation, and Obstacle Perception

Camera-based railway perception has commonly been motivated by the ability of forward-looking imagery to capture the appearance of the track area and surrounding scene. A review of vision-based onboard railway perception categorizes the literature into traditional computer vision and artificial-intelligence-based methods, and examines rail-track detection, obstacle detection, and obstacle-distance estimation as related functions [26]. Earlier image-based rail extraction studies used traditional computer vision methods, including Hough-transform-based rail or track-bed extraction, line extraction, sequential line detection, Kalman filtering, Gabor-wavelet and morphological processing, and template matching. The review further notes that traditional approaches commonly exploit rail geometry or image features such as edges, gradients, and track continuity, but can be sensitive to changing illumination, weather, backgrounds, and camera vibration [26].

Deep learning shifted much of the railway vision literature toward detection and semantic segmentation. Convolutional neural networks have been used for rail-track segmentation, railroad detection, rail area detection, obstacle detection, and near-miss or anomaly recognition [3, 4, 5, 27, 6, 28, 29, 9, 7]. More recent work extends this line to railway intrusion detection, lightweight obstacle detection, track-region segmentation, transformer-based rail segmentation, and resource-constrained rail segmentation [24, 30, 31, 32, 33, 34, 35, 36].

These works provide important component-level perception capabilities. Obstacle and intrusion detection address the presence of objects that may create operational risk [2, 24]. Recent multimodal onboard work further treats foreign-object intrusion as an open-set anomaly-detection problem, using camera and LiDAR measurements to identify candidate anomalous regions before downstream recognition [37]. Camera–LiDAR fusion has also been used for category-agnostic foreign-object segmentation, with normal-scene modeling and simulated anomalous samples supporting pixel-level masks for intruding objects in open railway environments [38]. A related video-surveillance approach formulates railway intrusion

detection as category-agnostic moving-object segmentation for real-time edge deployment [1]. Rail and track-bed segmentation identify regions of the scene that correspond to railway infrastructure [3, 4, 7, 32, 33]. Multitask railway perception has also begun to combine intrusion detection, track drivable-area segmentation, and track-line segmentation within one real-time framework [39]. Lightweight and real-time methods address computational constraints that are relevant to onboard perception [29, 34, 40]. The onboard obstacle-detection review makes the connection between these tasks explicit: rail-track detection is used to define the region in which potentially hazardous objects should be sought, while distance estimation is needed to determine whether detection occurs early enough for a train to stop [26]. However, these component-level formulations do not necessarily separate the visible infrastructure into individual route instances.

This limitation is central to the thesis because a semantic mask identifies rail or track-bed regions without necessarily encoding which rail boundaries form the same track or preserving the identities of different routes. Ziegler et al. address this distinction by detecting individual rails and then clustering them geometrically into the ego track and neighboring tracks [41]. Drahý et al. similarly show that a frame-wise segmentation mask can include both the main and side tracks, and introduce connected-component filtering and temporal context to isolate the main rails [42]. Together with topology-guided rail-track association [8], these studies motivate a transition from component-level segmentation toward instance-level and topology-aware perception. In multi-track scenes, particularly near switches, this transition requires methods that preserve rail pairing and route continuity rather than treating all visible rail pixels as one undifferentiated class.

Conventional semantic segmentation assigns a semantic class to each pixel, such as rail, track bed, or background. Route-aware railway perception adds relational and operational requirements: the system must associate left and right rails into track instances, preserve route continuity through switches, distinguish parallel and branching paths, and, where required, identify the route that is physically valid for the train. The added complexity is therefore primarily task-level rather than a claim about computational cost. Pixel-level

overlap alone cannot establish whether the predicted route is correct, so route-level structure and route-specific evaluation are also required.

2.3 Rail Path, Ego-Path, and Instance-Level Route Extraction

Several methods move beyond component-level perception by addressing rail pairing, route instances, or ego-path detection. Topology-guided rail-track detection associates left and right rails after segmentation and uses structural information to form track instances [8]. This line is important because it recognizes that rail perception cannot stop at pixel classification. It also shows a recurring methodological pattern: segmentation is often followed by a post-processing stage that imposes geometric or topological assumptions.

TPE-Net addresses this instance-level gap by representing each rail path through row-wise triplets composed of a left rail pixel, a center track point, and a right rail pixel [14]. This formulation makes left-right association part of the network output and supports extraction of possible ego-path proposals in multi-track scenes. It also distinguishes between extracting all possible ego-paths and selecting one unique route through switch-state reasoning. This boundary is important because the early camera-based contributions intentionally recover candidate route structure without claiming to solve the final route-decision problem.

Switch detection and classification provide a complementary source of route information. Jahan et al. use onboard camera images to distinguish switches from straight rail and to classify switch configurations as left or right according to the gap between the switch blade and stock rail [43]. The paper identifies this classification as relevant to predicting the driven path and compensates for limited railway training data through transfer learning and anchor-box optimization. Its output is nevertheless a detected switch and its configuration rather than a complete set of associated rail-path instances. This distinction illustrates why switch-state recognition and path-instance extraction are related but separate perception tasks.

Other methods approach path geometry using anchor lines or keypoints. Anchor-adaptive and anchor-line methods provide compact geometric representations for rail detection, while low-light keypoint extraction addresses rail landmark detection in image grids [44, 45, 46, 47, 48]. Recent depth-aware work extends this movement into 3-D rail-track detection by using long- and short-focal-length camera images to estimate depth and regress paired 3-D rail-track points over long ranges [49]. These methods are relevant to Chapter 4 because they show a broader movement from dense pixel classification toward structured rail geometry. However, in multi-track scenes, converting rail cues into consistent left-right associations and distinct route instances may still require explicit association rules or fixed constraints.

Recent work has increasingly targeted the train's ego track rather than all visible rail geometry. TEP-Net, currently available as a preprint, formulates train ego-path detection as an end-to-end learning problem for the immediate track ahead [13]. Alonso et al. instead combine track estimation with switch detection and classification, using the inferred switch configuration to determine which branch belongs to the ego track [50]. Jass et al. similarly combine track- and switch-detection networks for ego-track estimation, while Track2Net uses keypoint alignment, track anchors, and turnout-specific post-processing to refine the detected track segment occupied by the train [18, 51]. These formulations differ from paired-track detection methods such as KPENet, which represents a rail track as a pair and addresses assignment ambiguity at turnouts but remains focused on detecting track instances [52]. They also differ from standalone switch classifiers, which provide local switch information without constructing a complete route [43]. The literature therefore distinguishes rail and track-instance detection, candidate-path extraction, switch classification, and ego-track estimation as related but non-equivalent tasks. In this thesis, valid-route reasoning refers specifically to selecting a route supported by visible switch geometry. It should not be interpreted as establishing movement authority or operational safety.

2.4 Learning Route Continuity and Generalization

A recurring issue in railway perception is that learning-based feature extraction does not by itself provide a structured route representation or demonstrate cross-domain generalization. Some methods stop at semantic masks. For example, ERTNet and the study by Ooppakaew et al. evaluate track-region segmentation on their respective datasets but do not construct route instances or perform cross-dataset testing [32, 53]. Other methods retain explicit geometric stages after segmentation. RailNet applies inverse-perspective transformation, sliding-window feature extraction, and polynomial fitting to convert its binary segmentation output into rail lines [29]. Tang et al. similarly combine camera-based segmentation with LiDAR, inverse projection, and 3D fitting to recover track trajectories [54]. These studies show that learned segmentation and geometric post-processing remain complementary in several railway systems. They do not establish that a segmentation model alone preserves route instances.

The TRIT-Net line addresses this limitation by moving from post-hoc spatial clustering toward learned route continuity. Instead of predicting only candidate triplets and then assembling them through strip-wise clustering, TRIT-Net predicts a Centerline Regression Map, an Attraction Direction Map, and an Attraction Intensity Map. This Attraction Field Map represents local continuation cues that guide bottom-up tracing of ego-path instances. The underlying motivation is supported by prior work on topology-guided methods, anchor/keypoint rail representations, and TPE-Net's triplet formulation [8, 14, 44, 48].

The TRIT-Net network design also reflects broader segmentation architecture developments. HarDNet is used as a low-memory convolutional component, while SegFormer provides a transformer-based semantic segmentation reference for global context modeling [55, 56]. These generic architectures do not solve railway route extraction by themselves. Rather, they provide architectural context for a task-specific output representation that models path continuation and branching.

Transfer and pretraining offer another approach to railway-domain generalization, although

the available studies generally address narrower tasks than route-instance tracing. Lee et al. evaluate pretrained segmentation models with frozen weights for railway track detection, demonstrating that representations learned from large-scale datasets can be reused for a specialized railway task without target-domain retraining [57]. Jahan et al. use transfer learning to address limited railway-switch training data, but their output remains switch detection and classification rather than a continuous route instance [43]. Drahý et al. train on synthetic railway imagery and evaluate on real onboard video while propagating temporal context between frames [42]. This constitutes synthetic-to-real evaluation, but the study targets main-track segmentation and uses an unsupervised evaluation protocol. It therefore does not establish cross-domain generalization for multi-route tracing. These studies support the reuse of pretrained representations and temporal context, but do not directly test whether learned path-continuation representations generalize better than carefully tuned association or fitting procedures.

Against this background, Chapter 4 evaluates TRIT-Net across RailSem19, RailDB, RailSet, and a private dataset, includes zero-shot transfer from RailSem19 to RailSet, and investigates self-supervised encoder pretraining with VICReg [58, 59, 60, 61]. These experiments broaden the evidence beyond within-dataset evaluation and test whether learned route-continuation cues are less dependent on dataset-specific clustering heuristics. Their scope nevertheless remains candidate-path extraction under the datasets and protocols considered. The reviewed literature does not establish that learned continuity alone guarantees domain-independent performance, valid-route selection, or deployment robustness.

2.5 Vision-Language and Reasoning-Guided Segmentation for Route Validity

The camera-based extraction literature creates a natural next question: if several path candidates are visible, which route is valid for the train? TPE-Net and TRIT-Net distinguish this question from candidate-path extraction by focusing on possible ego-paths rather than

valid-route interpretation [14, 62]. Chapter 5 instead formulates valid-route segmentation as a language-conditioned reasoning task, where the model receives a railway image and a natural-language query and outputs the train's valid route mask together with an optional auxiliary rationale intended to describe visible switch geometry.

This formulation builds on vision-language and promptable segmentation models. LLaVA established a visual instruction tuning approach that connects image features to a large language model [63]. SAM introduced a promptable segmentation model that can generate masks conditioned by prompts [64]. LISA combines a multimodal language model with SAM-style mask generation and introduces a special segmentation token for reasoning segmentation [65]. LoRA supports parameter-efficient adaptation of large language models, which is relevant when the task-specific railway dataset is small [66]. Railway-specific VLM adaptation has also been studied for obstacle detection, with TVA-Rock combining track-region segmentation, patch-wise rock classification, and sequential textual–visual prompt tuning under limited railway obstacle data [67].

Relevant reasoning-segmentation and segmentation-oriented multimodal models include LISA++, which extends the reasoning-segmentation baseline. GSVA addresses generalized segmentation through multimodal large language models. See, Say, and Segment studies false-premise queries in segmentation-capable multimodal systems [68, 69, 70]. Related work has also moved toward dense grounded conversation, multi-target pixel reasoning, and multi-step reasoning segmentation: GLaMM connects grounded language responses with segmentation masks, PixelLM addresses pixel-level reasoning over multiple open-world targets, and RSVP combines visual prompting with multimodal chain-of-thought and mask refinement [71, 72, 73]. These developments position Chapter 5 within a broader movement from category-level segmentation toward language-conditioned segmentation and reasoning.

The railway-specific gap is that existing reasoning-segmentation work is not designed for switch interpretation. In switch scenes, the valid route is not merely an object category. It is a context-dependent structural decision supported by route continuity, rail gaps, and blade–stock contact. Chapter 5 therefore extends the thesis from structured extraction to

reasoning-guided route interpretation. Its evaluation compares joint semantic and reasoning finetuning with semantic-only finetuning, separates switch-independent and switch-dependent scenes, and audits mask–rationale consistency on a targeted subset. The results also expose limitations: rationale correctness is not reliably coupled to mask correctness, standard overlap metrics may not capture wrong-route errors in overlapping track beds, and the training objective does not explicitly enforce topology. These concerns are consistent with broader findings that grounding objectives do not automatically eliminate hallucination and that visual-text alignment and explanation faithfulness remain open problems in large vision-language models [74, 75]. They are also consistent with topology-aware segmentation literature, which treats connectedness and branch structure as properties that region-overlap measures alone may not preserve [76, 77, 8].

2.6 Datasets, Annotations, Metrics, and Evaluation Gaps

Dataset availability is a recurring constraint across the thesis. RailSem19 is the central public benchmark for the camera-based chapters and provides semantic rail-scene labels together with annotations for all visible rail-path instances in these images [58]. It supports segmentation and rail-path experiments, supplies the intended and competing route masks used in the Chapter 5 audit, and serves as the basis for that chapter's reasoning-segmentation dataset. Chapter 3 additionally considers corrected left-right rail associations through RailSem19+ as part of the experimental evidence for route-level extraction.

Chapter 4 expands evaluation to additional railway datasets. RailDB provides rail polylines and scenario diversity, while RailSet includes railway imagery with rail and track-bed related annotations and anomaly context [59, 60]. These datasets make it possible to discuss cross-dataset behavior and zero-shot transfer. Data-centric work such as Labels4Rails further emphasizes the importance of annotation tools and switch-aware labeling for autonomous rail perception [78]. More recently, RailMFOD extends the dataset landscape toward multimodal foreign-object segmentation by pairing camera and LiDAR data for open railway intrusion

perception [38].

Evaluation metrics must also match the task. Pixel-level segmentation metrics are useful for rail masks and track-bed masks, but they do not fully measure route instance correctness. The TPE-Net and TRIT-Net evaluations therefore report route-level or track-level measures in addition to true-positive-pixel and all-pixel evaluations [14, 62]. This distinction reflects the need to evaluate whether predicted rail pixels belong to the correct path instance rather than only whether they overlap rail regions.

The VLM chapter raises a further evaluation gap. CIoU and GIoU measure mask overlap, but a wrong-route mask in a switch scene may still overlap substantially with the correct track-bed region. Chapter 5 partly addresses this issue through switch-dependent stratification and a branch-exclusive mask criterion on the 30-image audit subset, but corresponding route-selection assessment is not available for all 178 switch-dependent test images. Railway-specific evidence also cautions against using IoU alone: Jaß and Thomas argue that centerline error is needed to expose safety-relevant rail-track detection failures that overlap scores may obscure [79]. Topology-aware methods provide complementary information. clDice evaluates agreement between masks and their centerlines, Betti matching compares spatially matched topological features, and Topograph enforces graph-based topology preservation [76, 80, 77]. OpenLane-V2 further illustrates how centerline geometry and graph connectivity can be evaluated separately in a transportation network [81]. These methods are informative analogues rather than validated railway-switch metrics: Betti-based measures capture components and loops but do not necessarily identify an incorrect branch in a tree-shaped route graph, for which explicit route-graph annotations would be required.

Uncertainty evaluation introduces a separate requirement. Calibration measures test whether confidence values correspond to observed correctness, whereas risk–coverage analysis tests whether abstaining on low-confidence predictions reduces error [82, 83]. Neither property follows from segmentation accuracy alone, and neither has been established for the route masks in this thesis. Rationale evaluation is similarly distinct from linguistic fluency. CHAIR measures hallucinated object mentions using caption annotations, while

CLIPScore measures broad image–text compatibility without requiring reference captions [84, 85]. Their direct transfer to railway rationales would require a railway-specific vocabulary, grounding annotations, or human assessment. Therefore, a high compatibility score would not by itself establish factual faithfulness. Chapter 5 manually assesses switch type and active-route direction on 30 switch-dependent images, but this targeted audit does not evaluate the complete rationale's factual correctness or visual grounding. Existing railway datasets and the reasoning dataset used in this thesis do not provide one unified set of route-graph, calibrated-confidence, and rationale-grounding annotations. Joint evaluation of route topology, uncertainty, and mask–rationale consistency therefore remains an open problem.

2.7 Research Gap and Thesis Positioning

The literature reviewed in this chapter supports a coherent thesis gap. First, component-level camera perception has advanced through obstacle detection, rail detection, and semantic segmentation, but these outputs often do not preserve route instances [3, 5, 7, 24, 33]. Second, topology-guided, anchor-based, keypoint-based, and ego-path methods move toward structured railway perception, but they may rely on post-processing, fixed association logic, or task definitions that differ from all-candidate path extraction [8, 44, 46, 48, 13]. Third, reasoning-capable segmentation models make it possible to ask explicitly for a valid route, but rationale grounding, topology, prompt robustness, and hallucination risk remain open issues [65, 68, 69, 70].

This thesis is positioned at the intersection of these gaps. Chapter 3 addresses camera-based possible ego-path extraction with explicit left-right rail association. Chapter 4 advances candidate-path extraction by learning route continuity through an Attraction Field Map. Chapter 5 shifts the task toward reasoning-guided valid-route segmentation. The thesis does not claim that VLM reasoning resolves all switch interpretation problems. Instead, it develops a sequence of route-structured perception contributions and uses the literature to identify

why each step is needed.

Safety-oriented validation constitutes a related but distinct gap. N-version agreement has been investigated for railway segmentation, while scenario-based validation using generative models has been proposed to broaden testing across operating conditions [79, 86]. These studies provide mechanisms for evaluating failures and scenario coverage, but they do not establish system-level safety approval for the route-perception tasks considered here. Calibrated uncertainty, broader scenario coverage, and system-level safety assurance remain directions for future work rather than demonstrated outcomes of this thesis.

Chapter 3

TPE-Net for Multiple-Track Rail Path Association

This chapter presents TPE-Net [14] as a unified framework for extracting and associating multiple possible ego-paths from forward-facing railway imagery. It also draws on the following submitted manuscript:

- M. Ghorbanalivakili, J. Kang, and G. Sohn, "TPE-Net: Location-Radius Regression for Ego-Path Candidate Enumeration in Railway Switch Scenes," submitted to *IEEE Open Journal of Intelligent Transportation Systems*, 2026.

3.1 Chapter Overview

The chapter addresses railway scene understanding for autonomous trains in complex multi-track environments, especially merging and diverging rail scenes where route ambiguity is common. The central task is to identify all possible ego-paths ahead of the train and associate the left and right rail boundaries belonging to each path instance.

The method makes three principal contributions. First, it formulates triplet-based rail-boundary regression within a multi-task fully convolutional network and investigates *triple-location regression* and *location-radius regression*. Second, it constructs a path tree through the Track Segment Generation (TSG) and All Possible Ego-path Generation (APEG) stages to organize predicted triplets into candidate ego-paths. Third, the experimental

evaluation covers RailSem19, the corrected RailSem19+ annotations, and the YDHR dataset through ablation, baseline, and cross-dataset experiments.

3.2 Problem Formulation

An autonomous train must reason about the routes on which it can possibly move before it can interpret obstacle locations with respect to the rail corridor ahead. In multi-track scenes, this requires more than classifying rail pixels as one semantic region. The system must instead recover route instances and preserve their topology.

In this chapter, a *track* is defined as a pair consisting of one left rail and one right rail. *Track points* are centerline points located midway between the paired rails, and the *rail area* is the region enclosed by those rails. The objective is to recover all plausible ego-paths ahead of the train rather than to determine one unique route by explicitly inferring switch states.

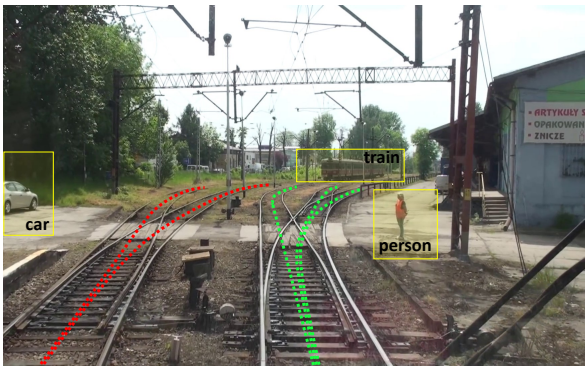


Figure 3.1 Railway scene captured by a forward-facing train-mounted camera. Green trajectories indicate possible ego-paths, while red trajectories denote rail paths that are not reachable by the train. Yellow boxes identify example rail-scene objects for context. Object detection is not part of the task addressed in this chapter.

Figure 3.1 illustrates the intended output: the method retains rail paths that the train can potentially follow and rejects visually present rail paths that are not reachable from its current position. The highlighted car, train, and person provide scene context only. This formulation makes the association of left and right rails into route instances central to the chapter.

3.3 Rail Path Representation and Methodology

3.3.1 System Overview

TPE-Net follows a three-stage bottom-up process [87], shown in Figure 3.2. First, the network extracts pixel-level triplets consisting of left rail, center track point, and right rail information. Second, the extracted triplets are clustered within local strips to form track segments. Third, the segments are linked across neighboring strips to generate the path tree that contains all detected possible ego-paths.

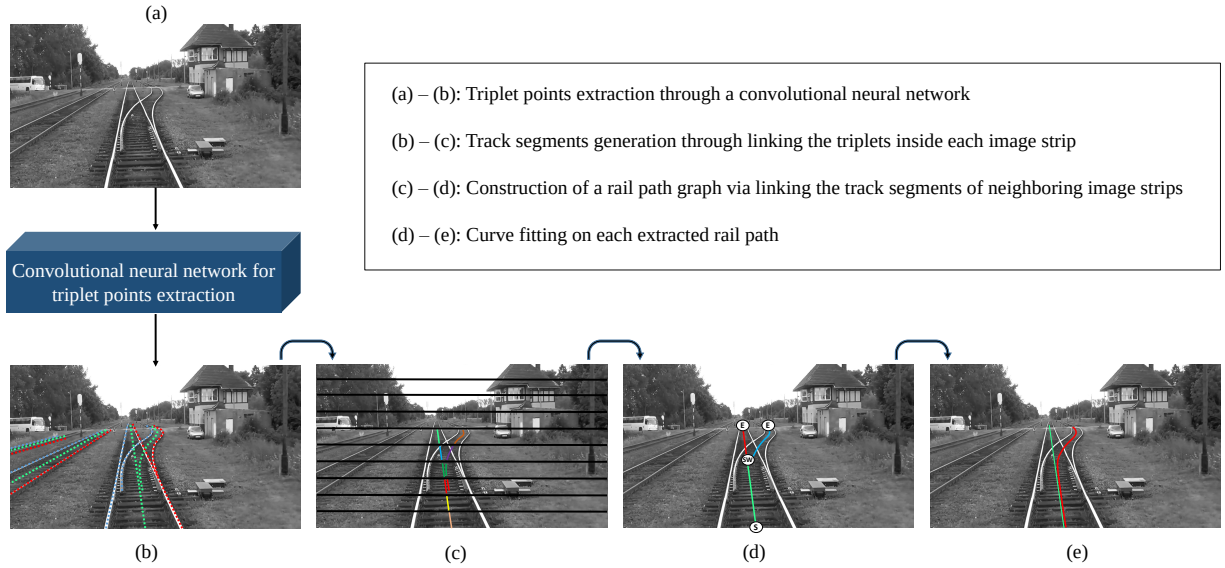


Figure 3.2 Overview of the TPE-Net rail path extraction pipeline. The network first detects and associates left, right, and center rail pixels. Track segments are then formed inside strips, after which a path tree is constructed to encode all potential ego-paths. The final stage discards invalid or redundant paths and fits polynomials to the retained rail pixels.

Together, these stages combine learned within-row triplet association with strip-wise spatial clustering and path-tree construction across rows.

3.3.2 Triplet-based Path Representation

A rail path is represented as a sequence of *triplets*. Each triplet contains one left rail pixel, one center track point, and one right rail pixel in an image row. Figure 3.3 shows this row-wise triplet representation.

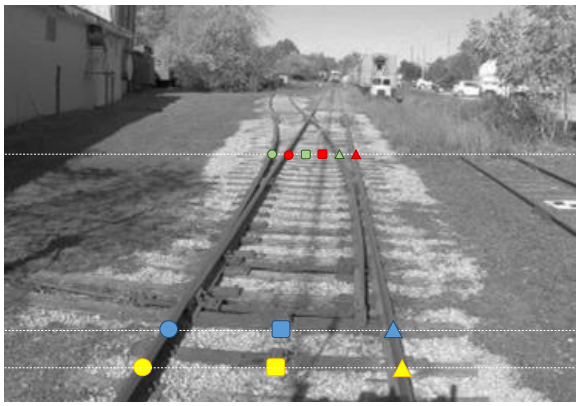


Figure 3.3 Triplet-based rail path representation. In sampled rows of the image, each color group denotes one triplet consisting of right, center, and left rail pixels. At this stage, no inter-row association has yet been performed.

This design differs from the object-level keypoint triplets used in CenterNet [88]. There, a single triplet may define a whole object, whereas in TPE-Net one triplet is assigned to each image row that overlaps a rail path. Figure 3.4 illustrates this conceptual distinction.

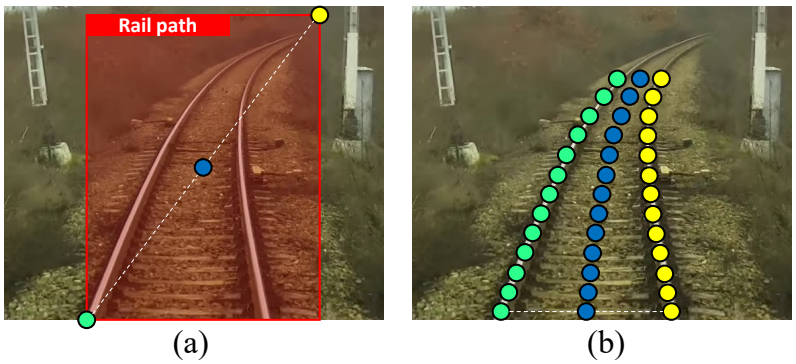


Figure 3.4 Conceptual difference between object-level keypoint triplets and the TPE-Net row-wise rail path representation. In the latter, each rail path is represented by one triplet per overlapping image row rather than by one triplet for the whole object.

3.3.3 Triple-Location Regression

The first regression design is termed *triple-location regression*. Under this design, the network predicts three heatmaps at full image resolution: the probability of being a center track point, the horizontal distance to the left rail, and the horizontal distance to the right rail.

Using the notation adopted throughout this chapter, the ground-truth center probability map is

$$C_{GT}(x,y) = \max\left(\frac{\min(l_1^{(x,y)}, r_1^{(x,y)})}{W_1^x}, \ldots, \frac{\min(l_n^{(x,y)}, r_n^{(x,y)})}{W_n^x}\right), \tag{3.1}$$

where $l_n^{(x,y)}$ and $r_n^{(x,y)}$ denote the horizontal distances from pixel (x,y) to the left and right rails of the nth rail area instance, and W_n^x is half of the corresponding track width at image row x. Pixels outside rail areas are assigned zero.

The distance-to-right and distance-to-left targets are

$$R_{GT}(x,y) = \max(r_1^{(x,y)}, \ldots, r_n^{(x,y)}) \tag{3.2}$$

and

$$L_{GT}(x,y) = \max(l_1^{(x,y)}, \ldots, l_n^{(x,y)}). \tag{3.3}$$

Figure 3.5 illustrates these regression targets. Nonmaximum suppression is applied row-wise to the center heatmap to recover track-point locations, after which the sampled left-distance and right-distance predictions yield the associated rail pixels.

3.3.4 Location-Radius Regression

Triple-location regression is unreliable in switch regions because a pixel inside a shared rail area can correspond to more than one left or right rail distance. This ambiguity is illustrated in Figure 3.6.

To address this issue, the second design uses *location-radius regression* [62]. Here, the network predicts a single heatmap $\mathbb{R}$ that simultaneously encodes the degree of being a track

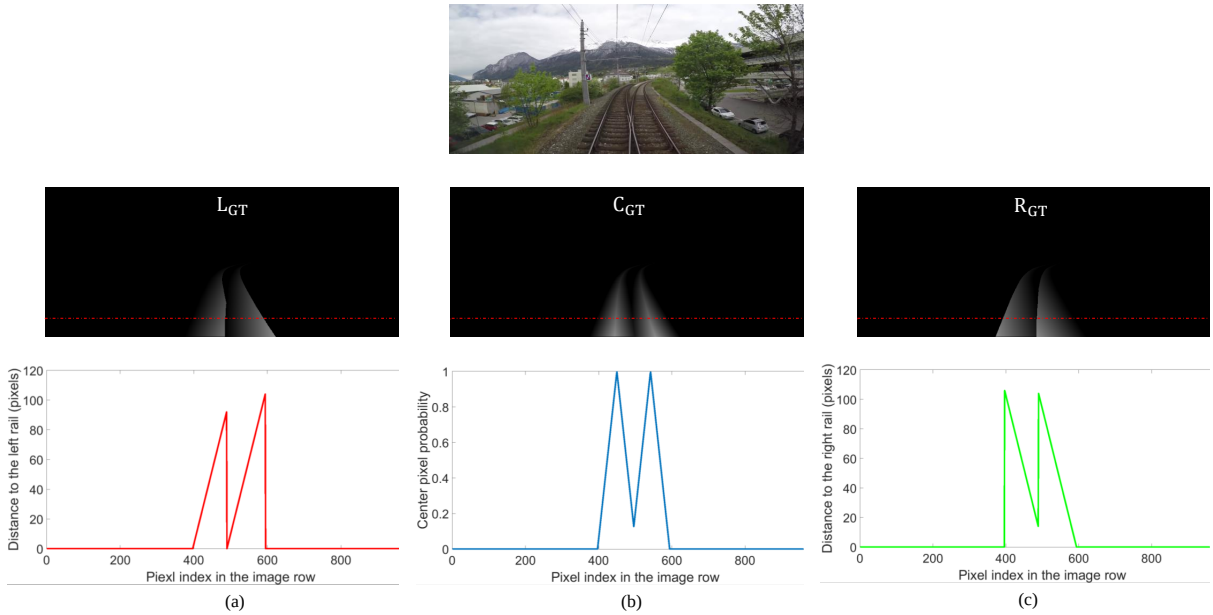


Figure 3.5 Ground-truth illustration for triple-location regression, showing a sample rail scene together with its center, right-distance, and left-distance regression targets and the corresponding row-wise value trends.

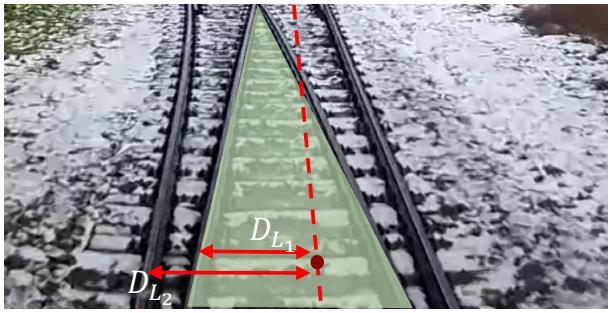


Figure 3.6 Ambiguity of triple-location regression in overlapping rail regions. A pixel in a shared switch region may correspond to multiple rail distances, which cannot be represented faithfully by one scalar value in a single-channel distance map.

point and the equal horizontal distance from that point to the left and right rails. The target map is

$$\mathbb{R}_{GT}(x,y) = \max\left(\min(l_1^{(x,y)}, r_1^{(x,y)}), \ldots, \min(l_n^{(x,y)}, r_n^{(x,y)})\right). \tag{3.4}$$

Pixels outside rail areas are again set to zero. The peaks of $\mathbb{R}$ provide track-point coordinates, while the peak value gives the equal distance to the left and right rails. For a detected center point (x_0, y_0), the triplet coordinates are recovered as

$$[(x_0, y_0) - \mathbb{R}(x_0, y_0),\ (x_0, y_0),\ (x_0, y_0) + \mathbb{R}(x_0, y_0)]. \tag{3.5}$$

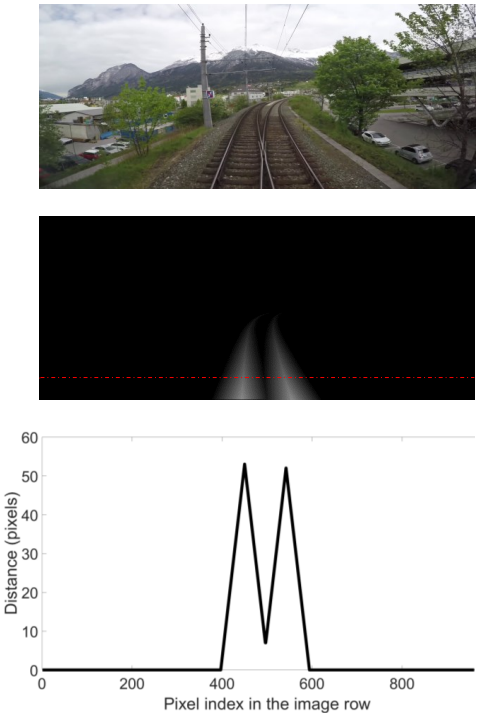


Figure 3.7 Ground-truth illustration for location-radius regression. The figure shows the single regression map and its value profile along a sample row, where peak locations indicate center track points and peak values encode equal left-right rail distance.

Figure 3.7 illustrates the resulting target map. This formulation is more reliable in switch regions because a track point belongs to only one track instance even when rail areas overlap.

3.3.5 TPE-Net Architecture

The TPE-Net architecture is a multi-task fully convolutional encoder-decoder network that combines semantic segmentation with rail-path regression [89]. The segmentation branch follows a U-Net-style encoder-decoder design [90] and outputs per-pixel semantic labels, while the regression branch predicts either the triple-location outputs or the location-radius output. The regression branch fuses the final two segmentation features in order to leverage semantic information when predicting rail-path geometry [91].

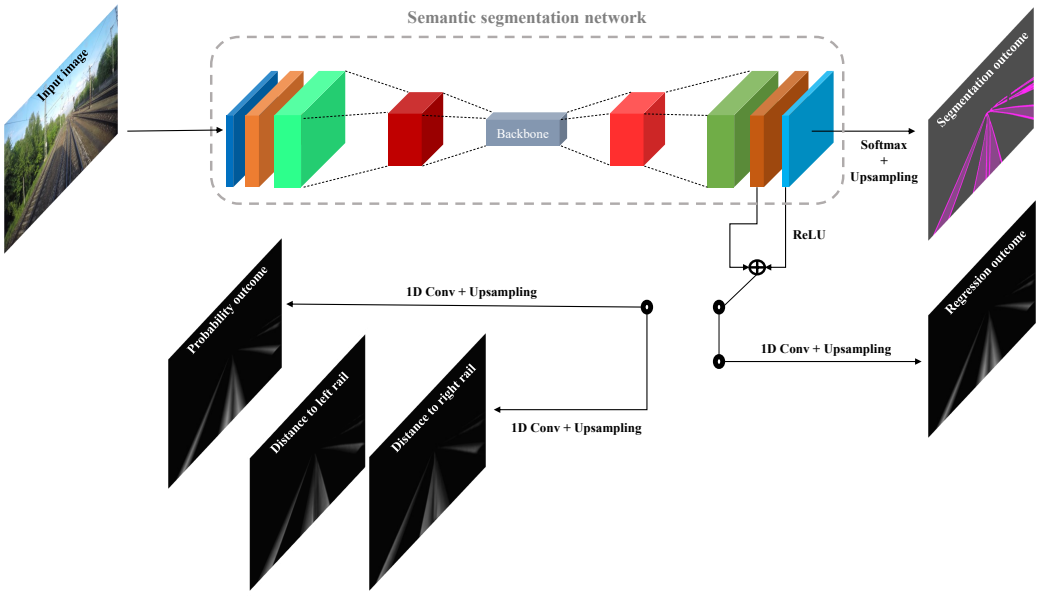


Figure 3.8 TPE-Net architecture. The network supports two regression-branch designs, one for triple-location regression and one for location-radius regression, alongside the shared segmentation branch.

Figure 3.8 shows the complete architecture. The framework can incorporate a suitable semantic segmentation backbone. The evaluated baselines are HarDNet [55], D-LinkNet34 [92], BiSeNet V2 [93], and ERFNet [94].

3.3.6 Track Segment Generation

The network extracts at most one triplet in each image row for a given rail path, but it does not establish associations between triplets in adjacent rows. The Track Segment Generation (TSG) block performs this first association stage. For an input image of height H and width W, the image-coordinate space is divided into $\lfloor H/h \rfloor$ non-overlapping horizontal strips of height h and width W. The strips are processed from the bottom of the image to the top, corresponding to the near-to-far direction in the forward-facing camera view.

Algorithm 1 Track Segment Generation (TSG) block

for $j \in \{1, 2, \ldots, \lfloor H/h \rfloor\}$ **do**
  $strip_clusters \leftarrow$ empty list
  $num_total_clusters \leftarrow 0$
  for $r_k^j \in \{1, 2, \ldots, h\}$ **do**
    for $c_{i,k}^j \in \{1, 2, \ldots, n_k\}$ **do**
      if $num_total_clusters = 0$ **then**
        $c_{i,k}^j$ generates a cluster s_1^j
        Append s_1^j to $strip_clusters$
        $num_total_clusters \leftarrow 1$
      else
        for $s_q^j \in \{1, 2, \ldots, n_j\}$ **do**
          Compute D_q, the distance between $c_{i,k}^j$ and $\mathbf{C}_{n_q,q}^j$
        end for
        $D_{q'} \leftarrow \min(D_1, \ldots, D_{n_j})$
        if $D_{q'} \leq thresh_0$ **then**
          Append $c_{i,k}^j$ to $s_{q'}^j$
        else
          $c_{i,k}^j$ generates a cluster $s_{n_j+1}^j$
          Append $s_{n_j+1}^j$ to $strip_clusters$
        end if
      end if
    end for
  end for
end for

Within strip j, the kth row is denoted by

$$r_k^j = \{c_{1,k}^j, c_{2,k}^j, \ldots, c_{n_k,k}^j\}, \tag{3.6}$$

where $c_{i,k}^j$ is the center coordinate of the ith extracted triplet in that row and n_k is the

number of triplets in the row. The set of track segments formed in the strip is

$$S_j = \{s_1^j, s_2^j, \ldots, s_{n_j}^j\}, \tag{3.7}$$

where n_j is the number of segments and

$$s_q^j = \{\mathbf{C}_{1,q}^j, \mathbf{C}_{2,q}^j, \ldots, \mathbf{C}_{n_q,q}^j\} \tag{3.8}$$

contains the n_q triplets assigned to segment q in bottom-to-top order.

The first available triplet initializes a segment. For each subsequent triplet, the Euclidean distance D_q is measured between its center point and the center point of the last triplet $\mathbf{C}_{n_q,q}^j$ in every existing segment. If the minimum distance does not exceed the association threshold $thresh_0$, the complete triplet is appended to the nearest segment. Otherwise, it initializes a new segment. Algorithm 1 formalizes this strip-wise clustering procedure.

TSG is applied independently within each strip. Consequently, its output consists of local track segments, with no association yet established between segments in neighboring strips.

3.3.7 All Possible Ego-path Generation

The All Possible Ego-path Generation (APEG) block performs inter-strip association. It links the track segments produced by TSG while discarding segments that do not contribute to a path reachable from the train. The output is a path tree

$$T = \{E_1, E_2, \ldots, E_{n_E}\}, \tag{3.9}$$

where each edge E_e stores a start node, an end node, one or more switch nodes, and a list of subedges. Each subedge is a track segment inherited from the TSG stage. The tree therefore retains both the two-dimensional geometry of the associated triplets and the branching structure of the candidate ego-paths.

Three node types are used. The unique start node represents the common origin of the

possible ego-paths in front of the train. A switch node denotes a location at which two or more paths begin to separate, and an end node is the final detected point of a possible ego-path. To initialize the tree, the method searches the bottom strip for the segment whose first triplet has the smallest Euclidean distance D_q to the bottom-center image location $(H, W/2)$. If the bottom strip contains no segment, the same search proceeds through subsequent strips until a root is found. The selected triplet becomes the start node, and its segment becomes the first subedge of E_1. The other segments in that initialization strip are excluded from the tree.

The remaining strips are processed from bottom to top. For a segment s_q^j in the current strip, the distance $d_{q,e}^j$ is measured between its first triplet and the terminal triplet of the uppermost subedge in each existing edge E_e. A segment within threshold $thresh_1$ extends the corresponding edge. When two or more segments in the same strip associate with one edge, the previous terminal point becomes a switch node and new outgoing edges are created for the branches. Algorithm 2 formalizes this path-tree construction procedure.

After all strips have been processed, an edge is classified as start-to-end, start-to-switch, switch-to-switch, or switch-to-end according to the node fields it contains. Consecutive edges are then assembled according to these node connections. Each possible ego-path is obtained by traversing the resulting tree from its start node to an end node.

3.3.8 Post-Processing and Visualization

TPE-Net uses the segmentation output to reduce regression errors during post-processing. If a regressed left or right rail pixel does not lie within an allowable horizontal distance of a rail-class pixel, the estimate is shifted to the nearest rail-class pixel with highest probability in that image row. Finally, least-squares polynomial fitting is applied to the left and right rail points of each retained path in order to generate coherent path visualizations.

Algorithm 2 All Possible Ego-path Generation (APEG) block

$path_tree \leftarrow$ empty list
$root_found \leftarrow$ False
for $S_j \in \{1, 2, \ldots, \lfloor H/h \rfloor\}$ **do**
    if $j = 1$ **or** $root_found =$ False **then**
        $E_1 \leftarrow \{$start node : $\emptyset$, end node : $\emptyset$, switch node : $\emptyset$, subedges : $\emptyset\}$
        for $s_q^j \in \{1, 2, \ldots, n_j\}$ **do**
            Compute D_q
        end for
        $D_l \leftarrow \min(D_1, D_2, \ldots, D_{n_j})$
        $E_1 \leftarrow \{$start node : $\mathbf{C}_{1,l}^j$, end node : $\mathbf{C}_{n_l,l}^j$, switch node : $\emptyset$, subedges : $[s_l^j]\}$
        Append E_1 to $path_tree$
        $root_found \leftarrow$ True
    else
        for $s_q^j \in \{1, 2, \ldots, n_j\}$ **do**
            $linked_edges \leftarrow$ empty list
            for $E_e \in \{1, 2, \ldots, n_E\}$ **do**
                Compute $d_{q,e}^j$
            end for
            for $d_{q,e}^j \in \{1, 2, \ldots, n_j\}$ **do**
                if $d_{q,e}^j \leq thresh_1$ **then**
                    Append s_q^j to $linked_edges$
                end if
            end for
            if $linked_edges$ has one element, which is s_q^j **then**
                E_e[end node] $\leftarrow \mathbf{C}_{n_q,q}^j$
                Append s_q^j to E_e[subedges]
            else if $linked_edges$ has more than one element, s_q^j and s_w^j **then**
                E_e[end node] $\leftarrow \emptyset$
                Append $\mathbf{C}_{n_u,u}^e \mid u = n_e$ to E_e[switch node]
                $E_\alpha \leftarrow \{$start node : $\emptyset$, end node : $\mathbf{C}_{n_q,q}^j$, switch node : $\mathbf{C}_{n_u,u}^e \mid u = n_e$, subedges : $[s_q^j]\}$
                $E_\beta \leftarrow \{$start node : $\emptyset$, end node : $\mathbf{C}_{n_w,w}^j$, switch node : $\mathbf{C}_{n_u,u}^e \mid u = n_e$, subedges : $[s_w^j]\}$
                Append E_α and E_β to $path_tree$
            end if
        end for
    end if
end for

3.3.9 Loss Function

The training objective separates the loss into regression loss, segmentation loss, and total multi-task loss. For a batch of regression outputs, the regression loss is

$$Loss_{reg} = \frac{1}{BS \times W \times H} \sum_{i=1}^{BS} \sum_{y=1}^{W} \sum_{x=1}^{H} \left| I_G^i(x,y) - I_{est}^i(x,y) \right|, \tag{3.10}$$

where BS is batch size, W and H are image width and height, I_G is the ground-truth image, and I_{est} is the predicted regression image.

For semantic segmentation, TPE-Net uses bootstrapped cross-entropy [95]. The original ground-truth masks can be reduced from 19 classes to 3 classes representing rails, rail areas, and other objects as background. The segmentation loss is

$$Loss_{seg} = \frac{1}{K} \sum_{i=1}^{BS} \sum_{y=1}^{W} \sum_{x=1}^{H} ind\big(CE^i(x,y) > t_K\big) \times CE^i(x,y), \tag{3.11}$$

where t_K is the cross-entropy threshold and K is the normalization count. For c segmentation classes, the per-pixel cross-entropy is

$$CE(x,y) = -\sum_{n=1}^{c} L_n \log(P_n(x,y)), \tag{3.12}$$

where L_n is the one-hot ground-truth label for class n and $P_n(x,y)$ is the corresponding estimated Softmax probability. Given a minimum retained-pixel count K_0, the value of K is determined by

$$K = \max \left(K_0, \sum_{i=1}^{BS} \sum_{y=1}^{W} \sum_{x=1}^{H} \begin{cases} 1, & \text{if } CE^i(x,y) > t_K \\ 0, & \text{if } CE^i(x,y) \le t_K \end{cases} \right). \tag{3.13}$$

Thus, only pixels whose cross-entropy exceeds t_K contribute to the sum, while the normalization count is lower-bounded by K_0.

By retaining only hard pixels, bootstrapped cross-entropy reduces the contribution of easily classified pixels, including much of the background, which would otherwise dominate the segmentation loss. No explicit class-frequency weighting is applied.

The total multi-task loss is defined separately for the two regression designs. For location-radius regression, the loss contains the single regression output $\mathbb{R}$:

$$Loss_{location\text{-}radius} = \alpha \times Loss_{reg}^{\mathbb{R}} + Loss_{seg}, \tag{3.14}$$

where α is its regression-loss weight. For triple-location regression, the loss combines the left-distance, right-distance, and center-probability regression losses:

$$\begin{aligned} Loss_{triple\text{-}location} = \alpha_L \times Loss_{reg}^{L} + \alpha_R \times Loss_{reg}^{R} \\ + \alpha_P \times Loss_{reg}^{P} + Loss_{seg} \end{aligned}. \tag{3.15}$$

Here, α_L, α_R, and α_P are the respective regression-loss weights, and each component is calculated using Equation 3.10.

3.4 Datasets and Experimental Setup

3.4.1 RailSem19

RailSem19 [58] is the primary public benchmark used to train and evaluate TPE-Net. It contains 8,500 railway images collected under varied environmental and scene conditions, together with semantic labels and left-right rail coordinates for each track instance. These labels make it possible to derive both segmentation masks and the regression targets used by TPE-Net.

3.4.2 RailSem19+

RailSem19+ is a corrected version of RailSem19 created to address shortcomings in the original left-right rail associations. The relabeling process was applied to all 8,500 original images. A switch path is defined as a turnout path or a merge-in path. Of the 7,135 detected switch paths in the original data, 5,019 had correct left-right associations under the mIoU criterion, 2,025 had incorrect associations, and 91 switch paths were completely missed. The resulting RailSem19+ dataset therefore contains 7,226 switch paths with corrected associations and aligned start/end vertical coordinates for the paired rails.



(a) (b)

Figure 3.9 Examples of shortcomings in the original RailSem19 labels: incorrect left-right association for a path instance and inconsistency in the start and end coordinates of the paired rails.

Figure 3.9 illustrates the types of annotation issues addressed by RailSem19+.

3.4.3 YDHR Track Image Dataset

The YDHR track image dataset provides an external evaluation setting. Its images were collected on the York-Durham Heritage Railway in Ontario, Canada, using a forward-facing camera installed on a vehicle. The data cover track sections of approximately 500 meters containing three major tracks and several switches, with 10 image sets collected from different route traversals.

Because fully manual annotation of the collected images was impractical, the YDHR labels were generated through a semi-automatic three-stage workflow:

1. **Generation of 3D track points:** Dense survey point clouds were acquired with a terrestrial LiDAR from multiple locations along the target tracks and merged into a unified georeferenced point cloud. An existing track-extraction tool was then used to obtain the 3D track points.

2. **Estimation of camera poses:** A prism mounted on the vehicle was tracked by a robotic total station to provide the georeferenced (X, Y, Z) camera positions. The camera pose for each image was estimated from the resulting trajectory.

3. **Projection and label generation:** The 3D track points were projected into the images using the estimated camera poses and converted into track labels.

This procedure produces pixelwise left and right rail boundaries for every rail path represented in an image. The later quantitative evaluation uses 1,316 YDHR images.

3.4.4 Training Setup

Four TPE-Net baselines are trained using either RailSem19 or RailSem19+, with triple-location or location-radius regression selected according to the experiment. For RailSem19 and RailSem19+, the train-validation-test split is 6,000/1,000/1,500. All images are resized from 1080×1920 to 540×960. The training setup uses stochastic gradient descent with learning rate 0.001, polynomial decay scheduling, momentum 0.9, weight decay 0.0005, batch size 8, and 100 epochs. The segmentation loss parameters are set to $t_K = 0.3$ and $K_0 = 8192$. Training uses an NVIDIA GeForce RTX 3090 GPU with Ubuntu 20.04 LTS and CUDA 11.3 and takes approximately 48 hours.

3.4.5 Evaluation Protocol

The segmentation branch is evaluated on the RailSem19 test set using mean Intersection over Union (mIoU). Evaluating the path proposals requires an additional ground-truth preparation step because the original RailSem19 and RailSem19+ annotations contain distinct rail routes,

whereas the method returns only paths that are reachable from the train. For each test image, the annotated triplets from all routes are first stacked to remove their original route associations. This representation is consistent with the unassociated triplets obtained after applying nonmaximum suppression to the regression output. The stacked ground-truth triplets are then processed by the same TSG and APEG logic used for the predictions, producing a ground-truth path tree that contains only possible ego-paths.

Following the evaluation procedure in [8], correspondence between an estimated path and a ground-truth path is measured using an F1 matching rate. For each ground-truth rail pixel, a local window is centered on its image coordinates. A detected rail pixel within that window is associated with the ground-truth pixel and counted as a true-positive (TP) pixel. If no detected pixel lies within the window, the ground-truth pixel is counted as a false negative (FN). Detected pixels that remain unmatched after all ground-truth pixels have been examined are counted as false positives (FP). The numerical window size and F1 threshold are not specified.

If the resulting F1 score exceeds the matching threshold, the estimated and ground-truth paths form a true-positive route pair. An estimated path without a matched ground-truth path is a false-positive route, while an unmatched ground-truth path is a false-negative route. Route-level precision and recall are calculated for each image from these route counts. Average Precision (AP) and Average Recall (AR) are then obtained by averaging the per-image precision and recall values over the test set.

The path-extraction evaluation operates at three levels:

- **Route level:** precision and recall for complete path instances using the F1-based route matching described above.
- **Pixel level on true-positive routes:** pixelwise precision and recall are calculated only for matched route pairs. False-positive estimated routes and false-negative ground-truth routes are excluded.
- **Pixel level on all detected and missed routes:** the same pixelwise comparison is

applied to all paths, with pixels from false-positive routes contributing to the FP count and pixels from false-negative routes contributing to the FN count.

The evaluation also distinguishes scene complexity. A single-track scene contains one possible ego-path, whereas a multi-track scene contains multiple possible ego-paths ahead of the train.

3.5 Results

In multi-track scenes, TPE-Net achieves up to 0.98 route-level recall and 0.94 pixel-level average precision. The following analysis examines this performance through ablation, dataset correction, baseline comparison, and cross-dataset evaluation.

3.5.1 Regression Design and Post-processing Ablation

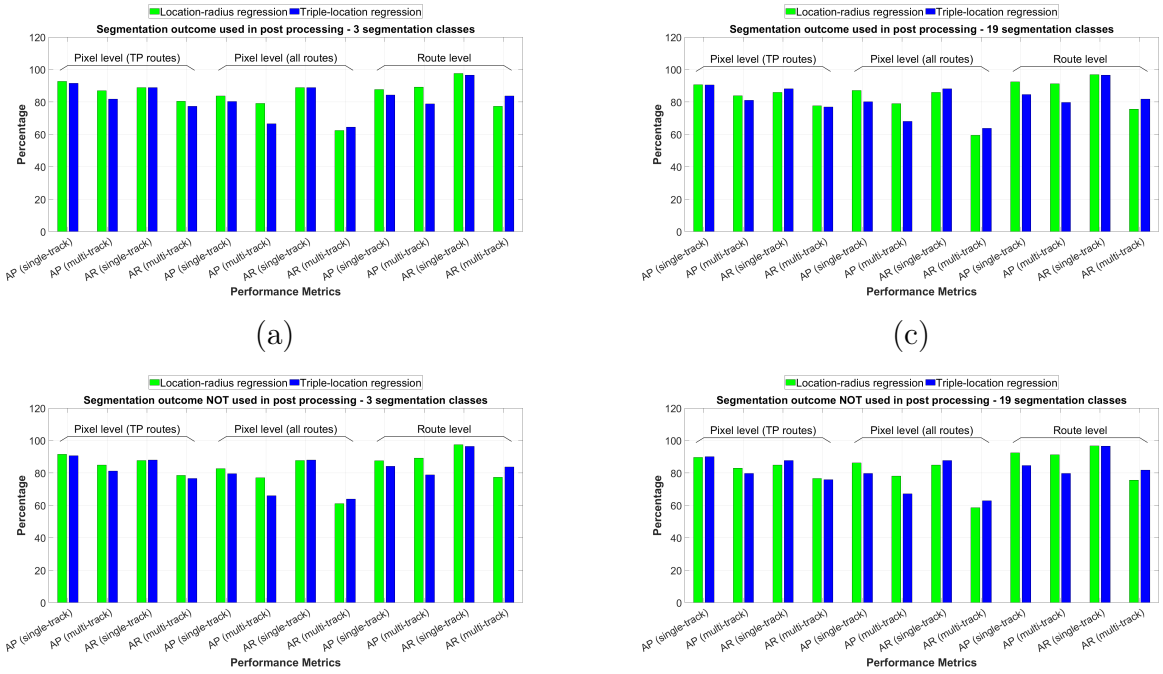
Table 3.1 compares the two regression designs, the 3-class and 19-class segmentation settings, and the use of segmentation-assisted post-processing. The evaluation is separated into single-track and multi-track scenes at the true-positive-route pixel, all-route pixel, and route levels.

Table 3.1 Performance of HarDNet-based TPE-Net designs on 1,500 randomly selected RailSem19 test images, separated into single-track and multi-track scenes. A check mark indicates that the segmentation output is used during post-processing.

Test	Regression type	Segmentation classes	Segmentation used in post-processing	Pixel level (TP routes)				Pixel level (all routes)				Route level			
				AP		AR		AP		AR		AP		AR	
				single	multi	single	multi	single	multi	single	multi	single	multi	single	multi
1	Location-radius	3	✓	**0.9277**	**0.8696**	0.8890	**0.8047**	0.8374	**0.7909**	0.8890	0.6238	0.8770	0.8924	**0.9754**	0.7740
2				0.9162	0.8488	0.8774	0.7857	0.8272	0.7711	0.8774	0.6104	0.8763	0.8924	0.9747	0.7740
3		19	✓	0.9063	0.8389	0.8589	0.7769	**0.8713**	0.7902	0.8589	0.5944	**0.9250**	**0.9124**	0.9682	0.7556
4				0.8964	0.8291	0.8486	0.7666	0.8622	0.7812	0.8486	0.5857	0.9243	0.9124	0.9674	0.7556
5	Triple-location	3	✓	0.9162	0.8184	**0.8892**	0.7733	0.8033	0.6656	**0.8892**	**0.6457**	0.8432	0.7882	0.9660	**0.8376**
6				0.9068	0.8116	0.8800	0.7658	0.7952	0.6593	0.8800	0.6398	0.8410	0.7882	0.9638	0.8376
7		19	✓	0.9058	0.8100	0.8817	0.7695	0.8021	0.6809	0.8817	0.6379	0.8459	0.7979	0.9653	0.8178
8				0.9009	0.7978	0.8771	0.7587	0.7977	0.6719	0.8771	0.6291	0.8466	0.7979	0.9660	0.8178

Figure 3.10 provides a visual comparison of the same four combinations of segmentation classes and post-processing choices.

The quantitative and visual comparisons show that location-radius regression performs better than triple-location regression across multiple precision and recall measures. Reducing the segmentation output from 19 classes to 3 classes and using it during post-processing also improve performance in most of the evaluated settings. Based on this ablation, the subsequent experiments use location-radius regression, 3-class segmentation, and segmentation-assisted post-processing.



(a) (c)

(b) (d)

Figure 3.10 Comparison of location-radius regression (green) and triple-location regression (blue) using the HarDNet baseline on the RailSem19 test set. The configurations use (a) 3-class segmentation with segmentation-assisted post-processing, (b) 3-class segmentation without segmentation-assisted post-processing, (c) 19-class segmentation with segmentation-assisted post-processing, and (d) 19-class segmentation without segmentation-assisted post-processing.

3.5.2 RailSem19 and RailSem19+ Baseline Comparison

Overall Results

Table 3.2 compares the four baselines using the original and corrected training annotations. Training on RailSem19+ generally improves true-positive-route pixel AP/AR, all-route pixel recall, and route recall, although not every metric increases for every baseline. For HarDNet, true-positive-route pixel AP/AR increase from 0.9207/0.8721 to 0.9344/0.9087, while route-level AP/AR increase from 0.9275/0.9491 to 0.9448/0.9643. The mixed changes in segmentation mIoU indicate that the path-extraction gains are not equivalent to uniform improvements in semantic segmentation.

Table 3.2 Overall performance of four TPE-Net baselines on 1,500 randomly selected RailSem19 test images. Each baseline uses 3-class segmentation and location-radius regression and is trained on either RailSem19 or RailSem19+.

Test	TPE-Net baseline	Training data	Pixel level (TP routes)		Pixel level (all routes)		Route level		mIoU	
			AP	AR	AP	AR	AP	AR	Rail	Track
1	HarDNet	RailSem19	0.9207	0.8721	0.8843	0.8537	0.9275	0.9491	**0.7007**	**0.8974**
		RailSem19+	**0.9344**	**0.9087**	**0.8997**	**0.8903**	**0.9448**	**0.9643**	0.7006	0.8949
2	D-LinkNet34	RailSem19	0.9104	0.8664	**0.7665**	0.8568	**0.8100**	0.9622	**0.6424**	0.7054
		RailSem19+	**0.9145**	**0.8758**	0.7663	**0.8664**	0.8080	**0.9662**	0.5660	**0.7064**
3	BiSeNet V2	RailSem19	0.8939	0.8509	**0.8433**	0.8323	0.9084	0.9455	0.6636	**0.8777**
		RailSem19+	**0.9004**	**0.8741**	0.8424	**0.8569**	**0.9140**	**0.9600**	**0.6652**	0.8775
4	ERFNet	RailSem19	0.9290	0.8864	**0.8451**	0.8756	0.8818	0.9701	**0.7307**	**0.9044**
		RailSem19+	**0.9318**	**0.9039**	0.8417	**0.8950**	**0.8838**	**0.9790**	0.7166	0.8963

Single-track and Multi-track Results

Table 3.3 separates the same comparison by scene type. RailSem19+ training improves the true-positive-route pixel AP and AR for every baseline in both single-track and multi-track scenes. The effects on all-route precision and route precision are less uniform, showing that corrected associations do not improve every evaluation dimension equally.

Table 3.3 Performance of four TPE-Net baselines on 1,500 randomly selected RailSem19 test images, separated into single-track and multi-track scenes. Each baseline uses 3-class segmentation and location-radius regression and is trained on RailSem19 or RailSem19+.

Test	TPE-Net baseline	Training data	Pixel level (TP routes)				Pixel level (all routes)				Route level			
			AP		AR		AP		AR		AP		AR	
			single	multi	single	multi	single	multi	single	multi	single	multi	single	multi
1	HarDNet	RailSem19	0.9277	0.8696	0.8890	0.8047	0.8374	0.7909	0.8890	**0.6238**	0.8770	0.8924	0.9754	**0.7740**
		RailSem19+	**0.9383**	**0.8891**	**0.9140**	**0.8481**	**0.9040**	**0.8457**	**0.9140**	0.6136	**0.9449**	**0.9400**	**0.9848**	0.7246
2	D-LinkNet34	RailSem19	0.9155	0.8486	0.8728	0.7920	0.7738	**0.6870**	0.8728	0.6701	**0.8126**	**0.7774**	0.9740	0.8164
		RailSem19+	**0.9195**	**0.8563**	**0.8817**	**0.8075**	**0.7739**	0.6781	**0.8817**	**0.6878**	0.8109	0.7753	**0.9761**	**0.8503**
3	BiSeNet V2	RailSem19	0.9037	0.8385	0.8646	0.7833	**0.8528**	**0.7995**	0.8646	0.5773	0.9105	0.9223	0.9696	0.7302
		RailSem19+	**0.9048**	**0.8502**	**0.8795**	**0.8122**	0.8458	0.7980	**0.8795**	**0.5932**	**0.9121**	**0.9299**	**0.9790**	**0.7373**
4	ERFNet	RailSem19	0.9341	0.8658	0.8931	0.8094	**0.8533**	0.7414	0.8931	0.6719	0.8878	0.8081	0.9848	0.7980
		RailSem19+	**0.9363**	**0.8803**	**0.9097**	**0.8355**	0.8497	**0.7464**	**0.9097**	**0.7222**	**0.8883**	**0.8284**	**0.9891**	**0.8602**

3.5.3 YDHR Evaluation

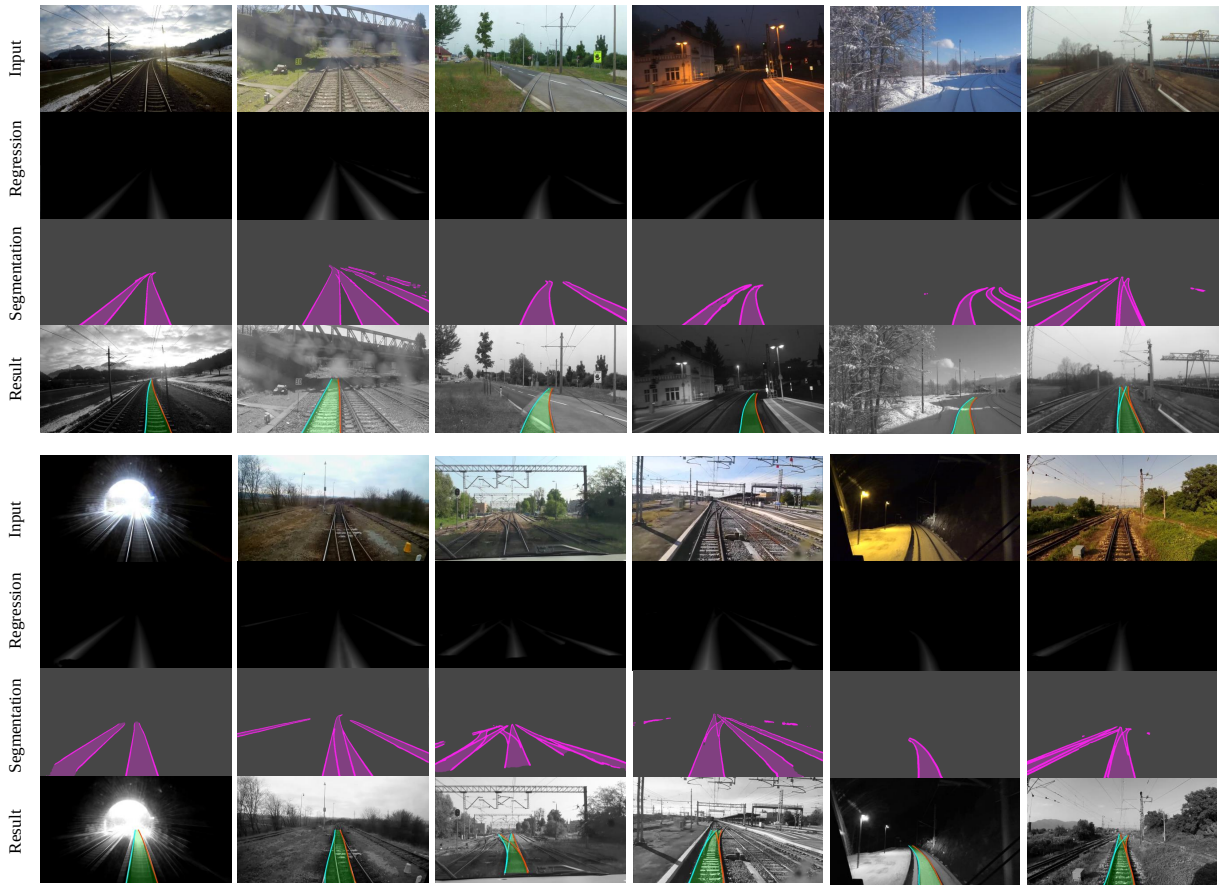
Overall Results

Table 3.4 presents the external YDHR evaluation. ERFNet achieves the highest true-positive-route pixel AP/AR at 0.8694/0.8144 and the highest route-level AR at 0.9867, while BiSeNet V2 achieves the highest route-level AP at 0.9043.

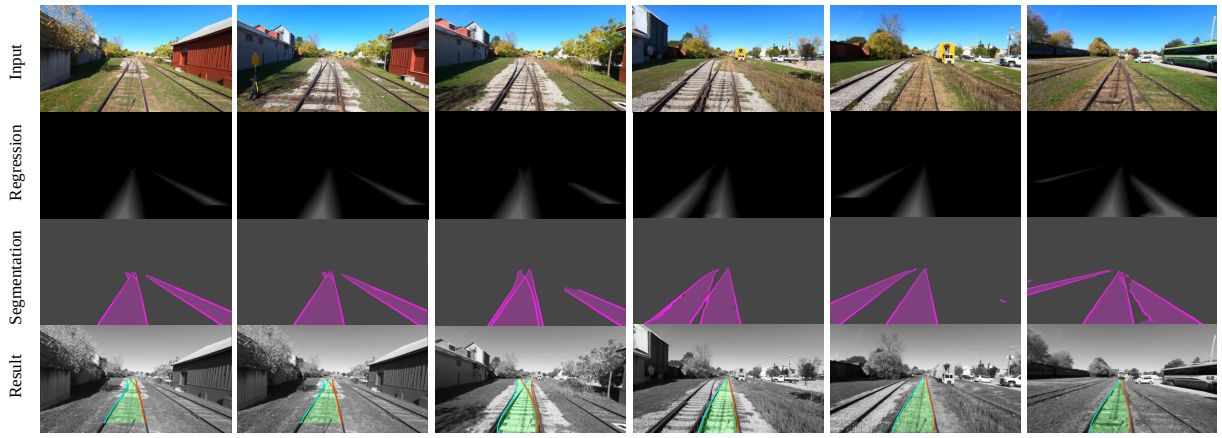
Table 3.4 Overall performance on 1,316 YDHR images using TPE-Net baselines trained on RailSem19+ with 3-class segmentation and location-radius regression.

Test	TPE-Net baseline	Pixel level (TP routes)		Pixel level (all routes)		Route level	
		AP	AR	AP	AR	AP	AR
1	HarDNet	0.8328	0.7881	0.7455	0.7747	0.8975	0.9840
2	D-LinkNet34	0.8214	0.7451	0.6280	0.7441	0.7469	0.9863
3	BiSeNet V2	0.8376	0.7889	0.7658	0.7709	0.9043	0.9737
4	ERFNet	0.8694	0.8144	0.7291	0.8097	0.8263	0.9867

The lower overall performance on YDHR is attributed to two dataset characteristics. First, YDHR contains more overlapping tracks and is therefore more challenging. Second, the forward-looking camera is positioned much closer to the ground than in RailSem19, which introduces more ground-level noise and more difficult track-alignment variation.



(a)



(b)

Figure 3.11 Qualitative rail path extraction results on RailSem19 and YDHR using the HarDNet baseline trained on RailSem19+ with location-radius regression, 3-class segmentation, and segmentation-assisted post-processing. Each example includes the input image, regression output, segmentation output, and final extracted paths under varied weather, illumination, and switch conditions.

Table 3.5 Performance on 1,316 YDHR images separated into single-track and multi-track scenes. All TPE-Net baselines use 3-class segmentation and location-radius regression and are trained on RailSem19+.

Test	TPE-Net baseline	Pixel level (TP routes)				Pixel level (all detected and missed routes)				Route level			
		AP		AR		AP		AR		AP		AR	
		single	multi	single	multi	single	multi	single	multi	single	multi	single	multi
1	HarDNet	0.8229	0.8925	0.7797	0.8389	0.7444	0.7527	0.7797	0.7450	0.9059	0.8475	0.9991	0.8936
2	D-LinkNet34	0.8150	0.8604	0.7416	0.7670	0.6420	0.5441	0.7416	0.7600	0.7681	0.6197	0.9902	0.9628
3	BiSeNet V2	0.8337	0.8611	0.7882	0.7935	0.7533	0.8414	0.7882	0.6674	0.8921	0.9778	0.9956	0.8431
4	ERFNet	0.8669	0.8852	0.8127	0.8244	0.7376	0.6788	0.8127	0.7917	0.8375	0.7592	0.9911	0.9601

Single-track and Multi-track Results

Table 3.5 shows that the relative behavior of the baselines differs across metrics and scene types. All four baselines report higher true-positive-route pixel AP in multi-track scenes than in single-track scenes, while route-level precision and recall vary across the two scene categories.

Qualitative Results and Runtime

Figure 3.11 illustrates the progression from the input image and network outputs to the final rail path extraction result. In the final visualizations, the ground-truth track bed is shown in green, detected right-rail polynomials are orange, and detected left-rail polynomials are cyan.

For this configuration, the network GPU time is 0.0156 seconds and the post-processing CPU time is 0.0727 seconds per image. The CPU measurement was obtained using an AMD Ryzen 9 5900X 12-core processor.

3.5.4 Comparison with Prior Work

The topology-guided rail-track detection method of Yang et al. [8] is one of the few prior approaches evaluated using performance measures comparable to those used for TPE-Net. This method achieves 95.4% true-positive rail-pixel accuracy, 94.87% all-pixel accuracy, and 98.18% route-level average precision. These values exceed the corresponding TPE-Net results.

Two differences limit direct interpretation of this numerical comparison. First, the prior

method is evaluated on a private dataset, whereas TPE-Net is evaluated on the publicly available RailSem19 benchmark. Second, the prior dataset was captured using one camera mounted on a specific train. Its known camera parameters support inverse perspective transformation and the derivation of three-dimensional information for rail detection and association. RailSem19 instead contains images collected in different regions without consistent camera parameters. The values therefore reflect different data and sensing conditions and should not be interpreted as a controlled comparison of the two algorithms.

3.6 Limitations, Future Directions, and Discussion

3.6.1 Scope and Methodological Limitations

The method focuses on extracting all potential ego-paths rather than identifying the unique route that the train will follow. It does not explicitly model switch states, and its output is therefore a set of plausible path proposals rather than a final route decision.

TPE-Net learns the left-center-right association within each predicted triplet, but association between triplets remains dependent on the TSG and APEG post-processing blocks. These stages use thresholded spatial clustering to assemble track segments and the path tree. Consequently, the complete pipeline is not end-to-end trainable with respect to inter-row triplet association.

Comparison with other methods is also constrained by differences in task definition, dataset availability, and camera information. As illustrated by the comparison above, similar route- and pixel-level measures do not by themselves establish a controlled benchmark when the underlying data and geometric inputs differ.

3.6.2 Future Directions

One future direction is learning-based triplet association. In this formulation, the neural network would learn robust features for associating triplets, allowing the proposed post-

processing blocks to be removed.

A second direction is explicit modeling of switch structure to identify one unique ego-path. The objective is an end-to-end trainable framework that integrates candidate path extraction with ego-path identification while reducing reliance on manual intervention and hard-coded association rules.

3.6.3 Role in the Thesis

Within the thesis, this chapter establishes the initial camera-based formulation of route-instance extraction and its empirical performance across the evaluation datasets. Its reliance on spatial-clustering post-processing provides the transition to the subsequent camera-based work on learned triplet association, while unique route identification remains outside the scope of TPE-Net.

Chapter 4

TRIT-Net for Railway Ego-path Candidate Tracing

Reliable railway perception requires more than the localization of individual rail pixels. This task also requires the preservation of left–right rail associations and path continuity in scenes containing multiple tracks, turnouts, and merges. Building on the triplet-based rail representation introduced in Chapter 3, TRIT-Net replaces heuristic spatial clustering with a trainable candidate-tracing framework in which path geometry is recovered through centerline regression and a learned Attraction Field Map (AFM). This formulation is designed to reduce reliance on dataset-specific post-processing while extracting the set of visible ego-path candidates connected to the train's vicinity under the tracing assumptions described below. The chapter draws on the published TRIT-Net paper [62] and the following submitted manuscript:

- M. Ghorbanalivakili, A. Varghese, and G. Sohn, "Triplet-based Railway Ego-path Candidate Tracing Using a Learned Attraction Field Representation," under review in *Engineering Applications of Artificial Intelligence*, 2026.

4.1 Chapter Overview

Given a forward-looking RGB image, the task is to extract the visible ego-path candidates connected to the train's vicinity. It does not determine the single commanded route through

explicit interpretation of switch states. *TRIT-Net* instead separates and traces these candidates while preserving their associated left and right rails. The framework retains the Centerline Regression Map R introduced with TPE-Net and adds an Attraction Field Map composed of an Attraction Direction Map D and an Attraction Intensity Map S. A task-specific three-head encoder–decoder jointly predicts R, D, and S, which guide bottom-up path construction from the near field towards the horizon, including controlled branching in shared track-bed regions.

The methodological development begins with the triplet-based tracing formulation and the TRIT-Net architecture, followed by supervised fine-tuning and self-supervised pretraining on unlabeled railway imagery. Evaluation on RailSem19, RailDB, RailSet, and the Private Track Image Dataset includes cross-backbone comparisons, direct comparison with spatial-clustering-based post-processing, zero-shot transfer, and component-level ablation. Qualitative analysis and failure-case examination then characterize the remaining limitations and the role of learned path tracing within the broader progression of camera-based railway perception developed in this thesis.

4.2 Motivation Beyond Chapter 3

The TPE-Net formulation in Chapter 3 established that a triplet-based rail representation can support the extraction of multiple ego-path candidates. However, those candidates were still assembled through explicit post-processing after dense prediction, creating two related limitations. First, track-point candidates derived primarily from rail saliency can fall between adjacent rails in overlapping switch regions without corresponding to a valid ego-path candidate. Second, grouping these points through two-dimensional spatial clustering requires hard-coded association rules that may depend on the characteristics of the training dataset. These limitations motivate a trainable representation that encodes both path association in shared regions and continuity across image rows.

Figure 4.1 relates these limitations to the two components of the Attraction Field Map.

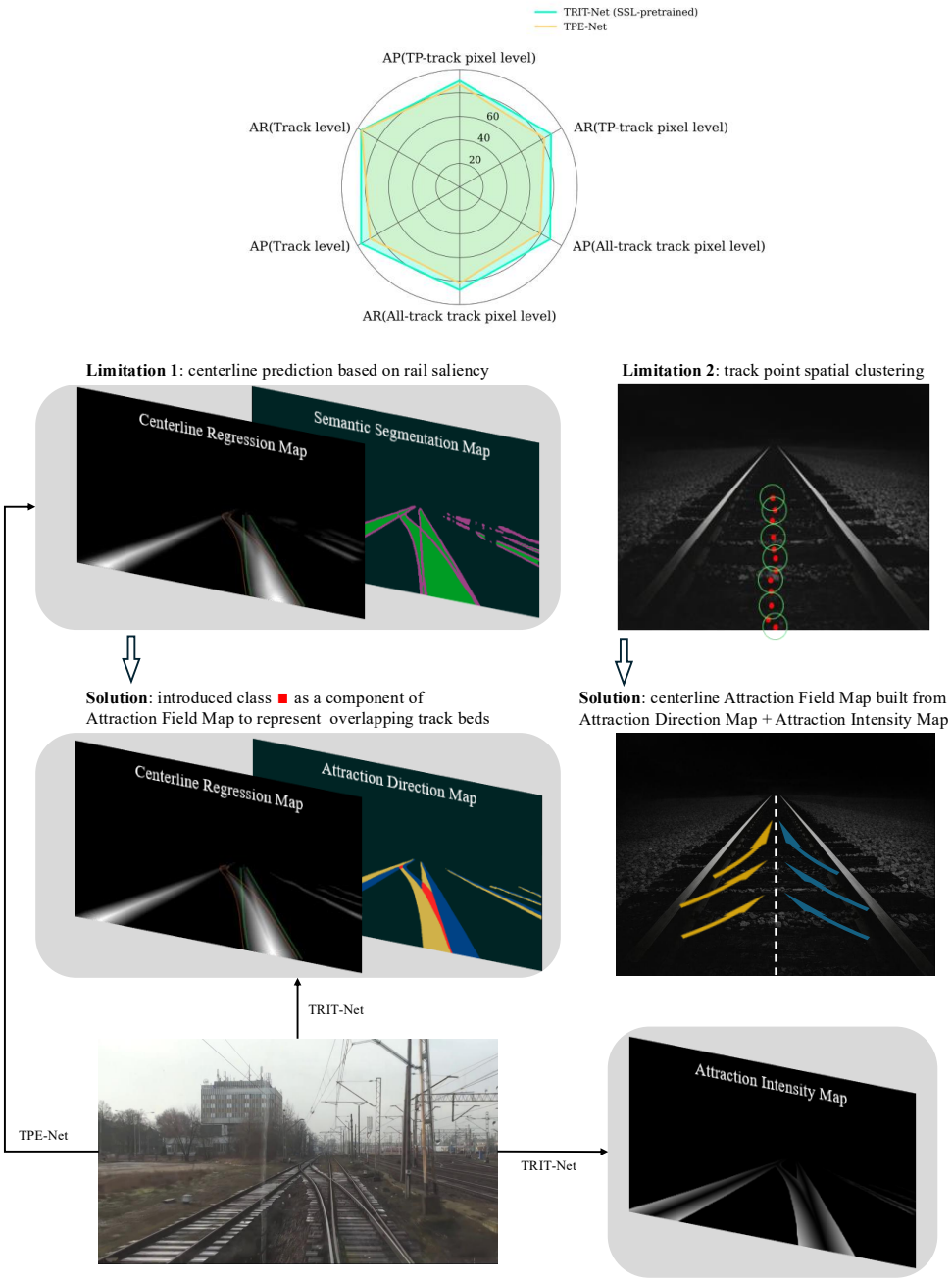


Figure 4.1 The earlier approach can generate invalid track-point candidates in overlapping switch regions and relies on spatial clustering to construct ego-path candidates. The AFM instead represents shared track-bed regions explicitly and supports learned row-wise tracing.

The Attraction Direction Map D assigns Class 3 to shared track-bed regions enclosed by overlapping path centerlines. By making this ambiguity explicit, D allows such locations to support more than one path continuation and enables controlled branching during tracing. The Attraction Intensity Map S provides the magnitude of the displacement between successive track points. Together, D and S constitute the AFM and supply the direction and magnitude required to trace ego-path candidates sequentially from the near field towards the horizon, reducing reliance on dataset-specific spatial-clustering rules.

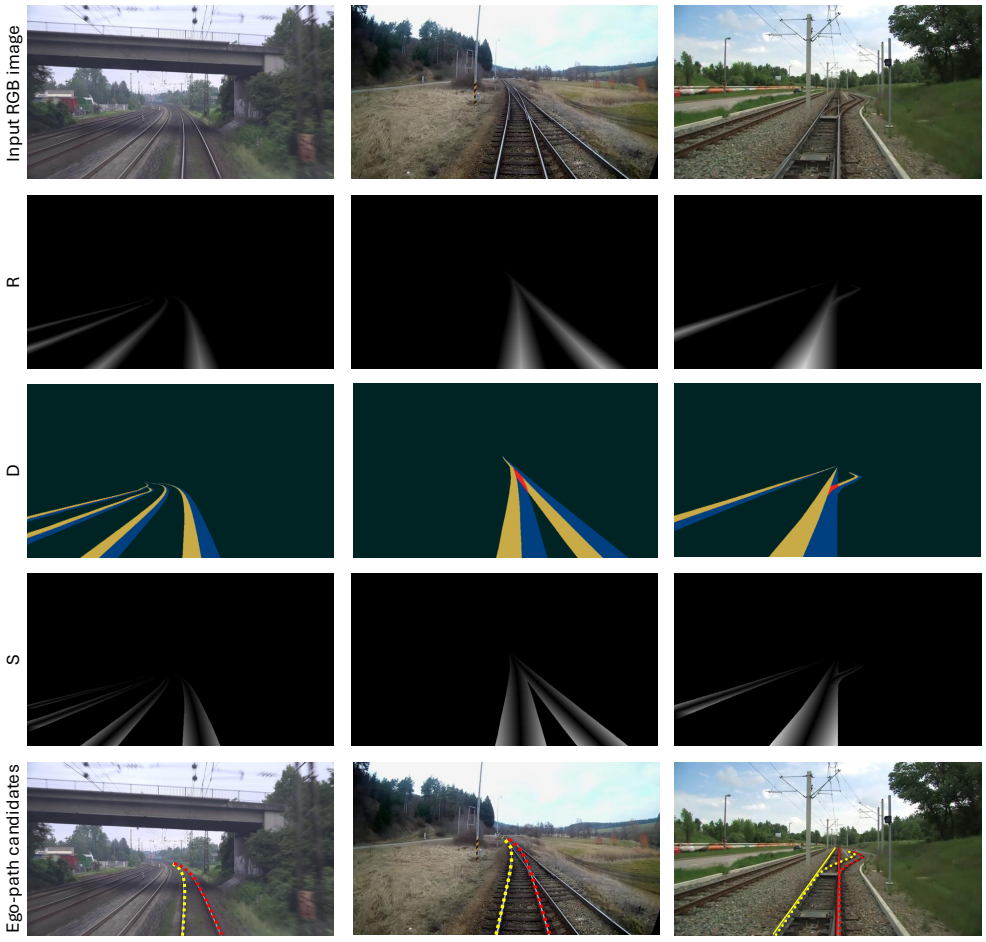


Figure 4.2 Overview of the TRIT-Net framework. The network predicts a Centerline Regression Map (R), an Attraction Intensity Map (S), and an Attraction Direction Map (D), which are then used to recover ego-path candidates.

4.3 TRIT-Net Framework

4.3.1 Triplet Representation and Overall Framework

A *track*, or *rail path*, is bounded by a left rail and a right rail. A visible track connected to the train's vicinity is an *ego-path candidate*. A *track point* is the geometric midpoint between the paired rails at a given image row, and the ordered sequence of track points forms the *track line*, also referred to as the path *centerline*. The *track bed* is the region bounded by the paired rails. A *switch region* permits transitions between paths: a *turnout* branches one path into another, whereas a *merge* combines two paths into one. These definitions distinguish the ego-path candidates to be traced from the individual rail pixels and other visible tracks in the image.

TRIT-Net retains the triplet representation described above, in which each ego-path candidate is modeled as an ordered set of triplet keypoints [14]. At most one triplet is defined per image row. Each triplet contains a left-rail pixel, a right-rail pixel, and a center track point located at the geometric midpoint between them. This representation preserves left–right rail association while remaining directly derivable from dataset annotations without requiring camera calibration or scene-specific geometric assumptions.

Figure 4.2 shows the overall TRIT-Net framework. Given one RGB input image, the network jointly predicts three outputs: the inherited Centerline Regression Map R and the two proposed AFM components, the Attraction Direction Map D and Attraction Intensity Map S. The map R provides local triplet and rail-offset evidence, whereas D and S provide the association cues used for bottom-up tracing of ego-path candidates from the near field toward the far field.

Formally, let $I \in \mathbb{R}^{H\times W\times 3}$ denote a forward-facing RGB image. The goal is to learn a mapping

$$f_\theta : I \rightarrow \{\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_Q\}, \tag{4.1}$$

where $\mathcal{P}_q$ denotes the qth ego-path candidate represented by its triplet keypoints, and Q

is the number of near-field-connected candidates returned under the decoder assumptions stated below. Physically visible tracks that do not connect to the train's vicinity are outside this ego-path candidate definition.

4.3.2 Centerline Regression Map

The Centerline Regression Map $R \in \mathbb{R}^{H \times W}$, inherited from TPE-Net [14], localizes track points and estimates their horizontal distance to the associated left and right rails. Its ground-truth target is defined as

$$\mathrm{R}_{\mathrm{GT}}(x, y) = \max\Big(\min(|y - y^L_{1,x}|, |y - y^R_{1,x}|), \ldots, \min(|y - y^L_{n,x}|, |y - y^R_{n,x}|) \Big), \tag{4.2}$$

where $y^L_{i,x}$ and $y^R_{i,x}$ denote the horizontal coordinates of the left and right rail pixels, respectively, for the ith ground-truth ego-path candidate at image row x. In Equation (4.2), n is the number of ground-truth ego-path candidates whose track beds contain pixel (x, y). This definition accounts for locations that belong to overlapping track beds in switch regions. At inference time, local maxima of R along each row correspond to candidate track-point locations, and the magnitude at each peak represents the horizontal distance from the track point to its associated rails.

4.3.3 Attraction Field Map

While R provides candidate track points, it does not describe how those points should be linked across rows. The proposed AFM consists specifically of D and S, which supply this associative information and guide tracing from the bottom of the image toward the horizon.

Starting from the track point closest to the bottom-center region of the image, the next predicted track point at row $x + 1$ is computed as

$$y_{x+1,n} = y_{x,n} + \Delta y_{x,n}, \tag{4.3}$$

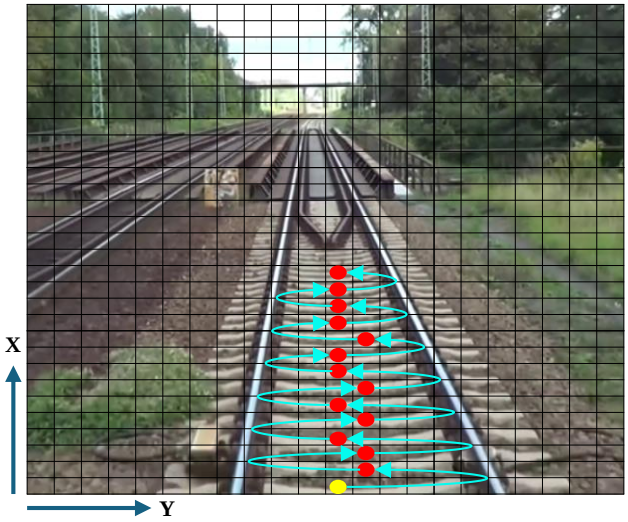


Figure 4.3 Conceptual illustration of track-point tracing. The yellow point denotes the starting point and the red points denote traced track points in successive rows.

where $\Delta y_{x,n}$ is the signed horizontal displacement along ego-path candidate n. Rather than regressing $\Delta y_{x,n}$ directly as one signed scalar, the method decomposes it into two learned outputs:

- **Attraction Direction Map** $D \in \{0, 1, 2, 3\}^{H \times W}$**:** a four-class semantic segmentation output encoding background, leftward motion, rightward motion, and shared track-bed regions in switch areas.
- **Attraction Intensity Map** $S \in \mathbb{R}^{H \times W}$**:** a dense map encoding the magnitude of the displacement.

The ground-truth intensity map is defined as

$$\mathrm{S}_{\mathrm{GT}}(x, y) = \min\left(\left|y - \frac{y^L_{1,x+1} + y^R_{1,x+1}}{2}\right|, \ldots, \left|y - \frac{y^L_{n,x+1} + y^R_{n,x+1}}{2}\right|\right). \tag{4.4}$$

The AFM training targets are generated automatically from the annotated left and right rail coordinates rather than through separate manual vector annotation. For each ego-path candidate, the track-point coordinate at row $x + 1$ is calculated as the midpoint of its

annotated left and right rails. Each supported pixel (x, y) is assigned a displacement toward the closest valid track point in the next row: the sign of this displacement determines its direction class in D_{GT}, while its absolute value determines the intensity in S_{GT}. Background pixels are assigned to Class 0, negative and positive horizontal displacements are assigned to Classes 1 and 2, and pixels in shared track-bed regions that can continue toward both centerlines are assigned to Class 3.

The learned displacement is horizontal by design, but the complete tracing step is not purely horizontal. The row coordinate advances deterministically from x to $x + 1$, while the AFM predicts only the horizontal displacement Δy. Thus, the full image-plane transition has a fixed one-row vertical component and a learned horizontal component. This formulation follows the assumptions that each path contains at most one triplet per image row and that tracing proceeds monotonically toward the horizon.

The maps R, D, and S are instance-agnostic: the network predicts one shared set of maps for an input image rather than a separate set for each candidate. Distinct ego-path candidates are subsequently recovered by tracing and branching within these shared outputs.

Class 3 in D is especially important in turnout regions because it identifies shared track-bed pixels that can legitimately branch into more than one path continuation. This is the mechanism by which the tracing process models multiple ego-path candidates without reverting to explicit clustering heuristics.

Figure 4.4 illustrates the AFM in a turnout region. Vectors farther from the centerline have larger magnitudes, while vectors near the centerline appear smaller. Decomposing the displacement into direction and intensity is necessary because a single signed scalar cannot represent the bidirectional behavior that occurs in shared track-bed regions.

4.3.4 Ego-path Candidate Construction

Each active trajectory maintains a state $m \in \{\mathsf{outside}, \mathsf{left}, \mathsf{right}\}$ for the current contiguous Class 3 region. The state $\mathsf{outside}$ means that the trajectory is not presently committed to a direction within such a region and is therefore eligible to branch on entry. When an

outside trajectory encounters Class 3, it is duplicated into left and right continuations. Each continuation retains its selected direction while it remains in that Class 3 region, preventing repeated duplication at every shared-region pixel. The state resets to outside after exit, allowing either descendant to branch again upon entering a later Class 3 region. The queue thereby enumerates sequential binary divergences recursively.

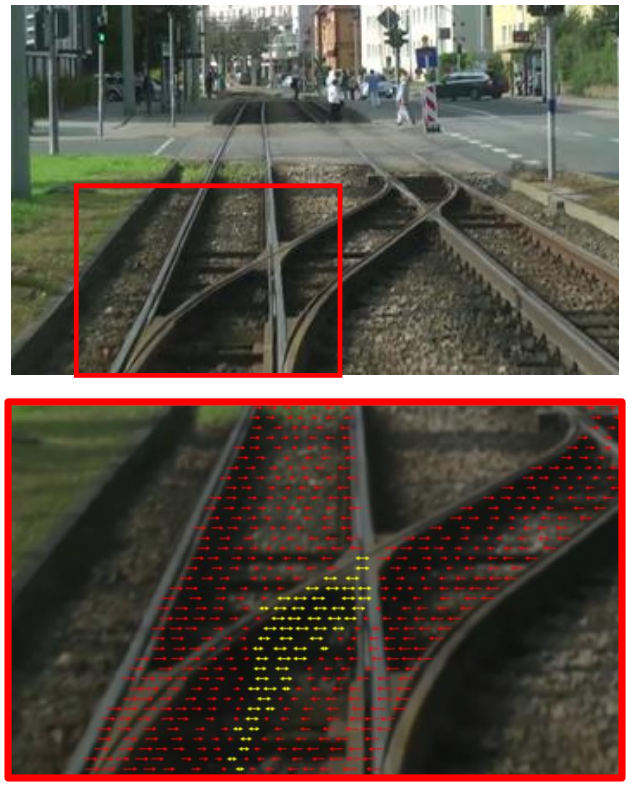


Figure 4.4 Attraction Field Map visualization in a sample turnout region. The learned field guides pixels toward path centerlines while supporting branching behavior in overlapping regions.

The path-extraction process can be summarized as follows:

- Extract the starting point from R as the track point closest to the bottom-center region of the image.
- Use D and S to predict the next track point row by row.

- When a trajectory in the outside state enters a new Class 3 region, duplicate it into state-labelled left and right branches, retain those states within the region, and reset each descendant to outside after exit.

- Terminate tracing when D indicates background or R no longer provides a valid track-point response.

- Recover the left and right rail pixels using the regressed value of R at each traced track point.

- Apply endpoint-distance non-maximum suppression: for each path pair whose endpoint distance is below 5 pixels, suppress the trajectory with higher horizontal total variation. Smooth each retained trajectory using a moving average with a fixed window of 10 points.

This construction retains several image-plane assumptions. Tracing starts from the detected track point nearest the bottom-center pixel, so the returned ego-path candidates are limited to those connected to this near-field seed. Each candidate contains at most one triplet per image row, and tracing proceeds monotonically from the near field toward the horizon. Each newly encountered Class 3 region is treated as a binary left/right divergence, allowing the queue to enumerate candidates through successive binary splits. Within this scope, the procedure is not a general graph-search decoder: it does not explicitly model merge recombination, non-binary junctions, looped connectivity, or arbitrary crossings. The AFM therefore reduces reliance on dataset-tuned spatial-clustering heuristics without eliminating these geometric and topological assumptions.

The effect of a seed error depends on whether the selected point remains within the support of a valid track bed. If the seed has a lateral offset from the centerline but remains inside the track bed, the predicted direction and intensity can move the next point toward the closest centerline, allowing the tracing process to correct the offset. If the seed lies outside the track-bed support, the expected direction is background and no valid continuation is available, so tracing terminates. The decoder does not search for an alternative seed or

reinitialize elsewhere in the image. False-positive responses in R and D outside the track bed may instead produce a spurious trajectory. The method can therefore tolerate some within-track-bed initialization error but remains sensitive to seed errors outside the valid AFM support.

Algorithm 3 Ego-path candidate construction using R, D, and S under the bottom-center seed and row-wise tracing assumptions

Extract starting point (x_0, y_0) as the track point closest to the bottom-center pixel
Initialize queue $\mathcal{Q}$ with $(\mathcal{T}, m)$, where $\mathcal{T} = [(x_0, y_0)]$ and $m = \mathsf{outside}$ ▷ $m \in \{\mathsf{outside}, \mathsf{left}, \mathsf{right}\}$ is the state within the current Class 3 region
while $\mathcal{Q}$ is not empty **do**
    Pop $(\mathcal{T}, m)$ from $\mathcal{Q}$
    Set $(x, y) \leftarrow$ the last point in $\mathcal{T}$
    while $D(x, y) \neq 0 \wedge R(x, y) \neq 0$ **do**
        if $D(x, y) = 3$ **then**
            if $m = \mathsf{outside}$ **then**
                Duplicate $\mathcal{T}$ as $\mathcal{T}_{\mathsf{left}}$ and $\mathcal{T}_{\mathsf{right}}$
                Push $(\mathcal{T}_{\mathsf{right}}, \mathsf{right})$ into $\mathcal{Q}$
                Set $(\mathcal{T}, m) \leftarrow (\mathcal{T}_{\mathsf{left}}, \mathsf{left})$
            end if
            if $m = \mathsf{left}$ **then**
                Set $D_{\text{eff}}(x, y) \leftarrow 1$
            else if $m = \mathsf{right}$ **then**
                Set $D_{\text{eff}}(x, y) \leftarrow 2$
            end if
        else
            Set $D_{\text{eff}}(x, y) \leftarrow D(x, y)$
            Set $m \leftarrow \mathsf{outside}$ ▷ Enable branching at a later Class 3 region
        end if
        Compute Δy using $S(x, y)$ and $D_{\text{eff}}(x, y)$
        Set the next point $(x + 1, y') \leftarrow (x + 1, y + \Delta y)$
        Append $(x + 1, y')$ to $\mathcal{T}$
        Update $(x, y) \leftarrow (x + 1, y')$
    end while
    Recover left and right rails using $(x_k, y_k \pm R(x_k, y_k))$
    Store the completed trajectory
end while
For each path pair with endpoint distance below 5 pixels, suppress the path with higher horizontal total variation
Smooth each retained trajectory using a moving average with window size 10

With the computational complexity, let N denote the number of triplet peaks extracted

from R, and let P denote the number of AFM-traced candidates before endpoint suppression. The peak-extraction step shared by both post-processing approaches scans the row-wise responses in $\mathcal{O}(HW)$ time and stores $\mathcal{O}(N)$ peaks. Once an AFM trajectory is initialized, each row-wise update reads D, S, and R at the current coordinate and performs a constant amount of computation. Tracing, rail recovery, and fixed-window smoothing therefore require $\mathcal{O}(PH)$ time, while pairwise endpoint-distance suppression adds $\mathcal{O}(P^2)$. Excluding the already resident dense output maps, the complete AFM post-processing stage consequently requires $\mathcal{O}(HW + PH + P^2)$ time and $\mathcal{O}(N + PH)$ auxiliary storage. The value of P is determined separately for each image rather than fixed as a hyperparameter or network-output dimension. Let B denote the maximum number of sequential Class 3 divergence regions encountered along a traced trajectory. Because each region creates a binary left–right split, $P \leq 2^B$: no divergence gives $P = 1$, one divergence gives at most $P = 2$, and two sequential divergences give at most $P = 4$. The implementation imposes no additional fixed cap on P. After endpoint suppression, the number of returned candidates Q therefore satisfies $Q \leq P$. The experiments did not record the empirical distribution of P, so no dataset-level mean or maximum is reported.

The TPE-Net Rail Path Graph post-processor instead performs explicit distance-based association. Let K be the maximum number of segments already formed in a horizontal strip when a new triplet is processed. Direct triplet-to-segment association requires $\mathcal{O}(NK)$ time. For inter-strip assembly, let m_j denote the number of segments in strip j and e_{j-1} the number of path-tree edges before that strip is processed; the nested segment-to-edge comparisons add $\mathcal{O}(\sum_j m_j e_{j-1})$. If L is the total number of vertices across the enumerated paths, path enumeration and fixed-degree polynomial fitting require $\mathcal{O}(L)$ time. Including peak extraction, the explicit time bound is therefore

$$\mathcal{O}\left(HW + NK + \sum_j m_j e_{j-1} + L\right), \tag{4.5}$$

with $\mathcal{O}(HW + N + E + L)$ storage, where E is the final number of path-tree edges. Bounding

the association stages by the structures generated from the N extracted points yields the conservative worst-case time bound $\mathcal{O}(HW + N^2 + L)$.

These bounds do not establish strict asymptotic dominance for either post-processor in every scene. AFM obtains each continuation through direct learned-map lookups, whereas the Rail Path Graph repeatedly evaluates association distances. However, when the candidate counts are constant and only one path is traced, both procedures are linear in the number of traversed rows after peak extraction. The practical difference is therefore most relevant in multi-candidate regions and is evaluated empirically in the controlled comparison shown in Figure 4.11.

4.4 Network Architecture and Training Strategy

4.4.1 TRIT-Net Architecture

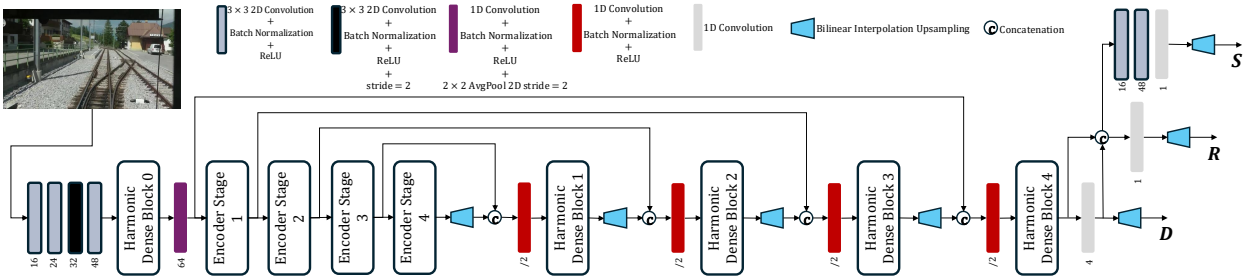


Figure 4.5 Overall TRIT-Net architecture. The encoder–decoder backbone combines convolutional processing, transformer-style Encoder Stages, and Harmonic Dense Blocks before separate task heads predict R, D, and S. The numbers below the convolutional layers indicate their output-channel counts.

As shown in Figure 4.5, TRIT-Net uses a shared encoder–decoder backbone to predict the three maps required for candidate tracing. Its architectural contribution is the task-specific integration of a common feature extractor and decoder with three lightweight heads that jointly predict the inherited map R and the proposed AFM components D and S. The constituent transformer and dense-block mechanisms retain their established designs and cited

provenance. Convolutional operations provide local spatial processing, while transformer-style attention provides broader contextual modeling. Four Encoder Stages progressively reduce the feature-map resolution, and the decoder uses bilinear upsampling, skip-feature concatenation, and Harmonic Dense Blocks (HDBs) to recover spatial detail before the three task heads produce their respective outputs.

An Encoder Stage receives a feature map $F \in \mathbb{R}^{h \times w \times c}$ and applies a convolutional patch-embedding layer to form overlapping patches. In Figure 4.6, p denotes the patch-embedding kernel size, s its stride, and $\bar{c}$ the number of output channels. Layer normalization is applied before the resulting representation is processed by a sequence of Encoder Blocks. The parameter *depth* specifies the number of Encoder Blocks repeated within a stage.

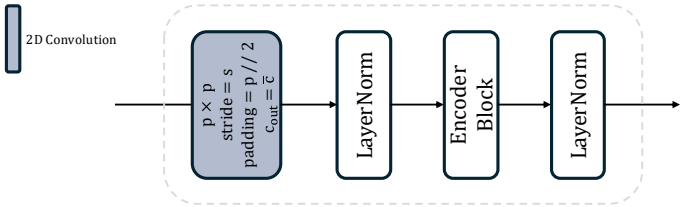


Figure 4.6 Architecture of a TRIT-Net Encoder Stage. A convolutional patch-embedding layer with kernel size p and stride s is followed by layer normalization and repeated Encoder Blocks.

Each Encoder Block contains an Efficient Multi-Head Attention (EMHA) module and a Mix Feed-Forward Network (Mix-FFN), as shown in Figure 4.7. Both components use pre-normalization and residual connections. EMHA reduces the cost of attention on high-resolution feature maps by applying a strided convolution to spatially downsample the keys and values before attention is computed, while the queries remain at the original resolution. The parameter r denotes this spatial-reduction factor, and n_{heads} denotes the number of attention heads. The Mix-FFN combines pointwise and depthwise convolutions with a nonlinear activation, allowing local spatial refinement after attention-based contextual processing. Here, e denotes its intermediate channel-expansion factor [56].

An HDB consists of densely connected Convolution–Batch Normalization–ReLU layers whose channel capacities increase according to a harmonic growth pattern. Its logarithmic

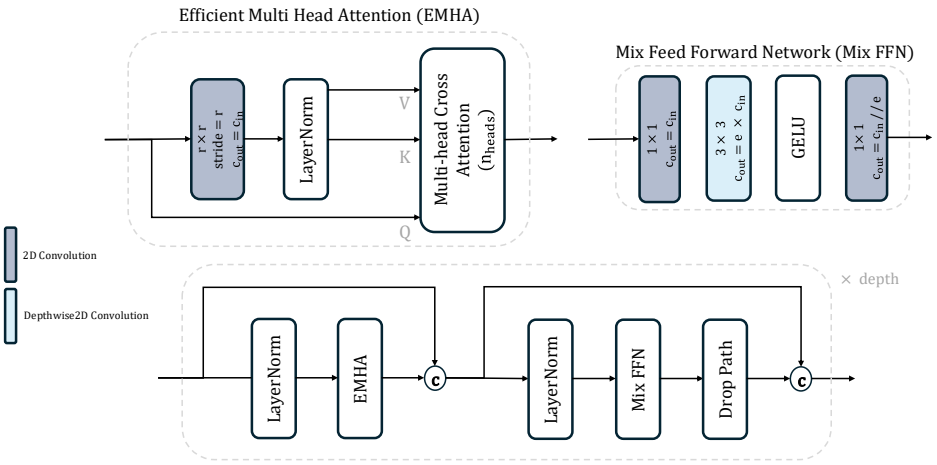


Figure 4.7 Architecture of a TRIT-Net Encoder Block, comprising Efficient Multi-Head Attention and a Mix Feed-Forward Network with pre-normalization and residual connections.

feature-reuse scheme reuses selected intermediate features to support fine-grained spatial processing [55]. Figure 4.8 shows the internal HDB structure. The parameter k is the base channel-growth rate, m is the channel-growth multiplier, and n_{layers} is the number of internal convolutional layers.

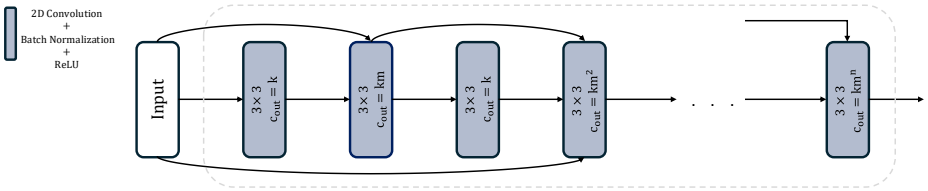


Figure 4.8 Architecture of a Harmonic Dense Block. The block uses Convolution–Batch Normalization–ReLU layers with harmonic channel growth and logarithmic feature reuse.

Within the decoder, HDBs aggregate low-level spatial cues from the encoder with higher-level semantic features while progressively recovering the resolution required for thin rail structures. The final task heads share this decoded representation but produce separate outputs for centerline regression, attraction intensity, and attraction direction.

Table 4.1 specifies the configuration of each stage and block. For the Encoder Stages, *depth*, p, s, r, e, $\bar{c}$, and n_{heads} denote, respectively, the number of Encoder Blocks, patch-embedding kernel size, patch-embedding stride, attention spatial-reduction factor, Mix-FFN

channel-expansion factor, output-channel count, and number of attention heads. For the HDBs, k, m, and n_{layers} denote the base channel-growth rate, channel-growth multiplier, and number of internal layers.

Table 4.1 Layer and stage configurations used in the TRIT-Net encoder–decoder architecture.

Layer	**Design parameters**
Harmonic Dense Block 0	$k = 10,\ m = 1.7,\ n_{layers} = 4$
Encoder Stage 1	depth= 3, $p = 7,\ s = 2,\ r = 8,\ e = 4,\ \bar{c} = 64,\ n_{heads} = 1$
Encoder Stage 2	depth= 4, $p = 3,\ s = 2,\ r = 4,\ e = 4,\ \bar{c} = 128,\ n_{heads} = 2$
Encoder Stage 3	depth= 6, $p = 3,\ s = 2,\ r = 2,\ e = 4,\ \bar{c} = 256,\ n_{heads} = 4$
Encoder Stage 4	depth= 3, $p = 3,\ s = 2,\ r = 1,\ e = 4,\ \bar{c} = 512,\ n_{heads} = 8$
Harmonic Dense Block 1	$k = 24,\ m = 1.7,\ n_{layers} = 8$
Harmonic Dense Block 2	$k = 18,\ m = 1.7,\ n_{layers} = 8$
Harmonic Dense Block 3	$k = 16,\ m = 1.7,\ n_{layers} = 4$
Harmonic Dense Block 4	$k = 16,\ m = 1.7,\ n_{layers} = 4$

Together, these configurations define the shared feature extractor and decoder used by all three prediction tasks.

4.4.2 Self-Supervised Pretraining

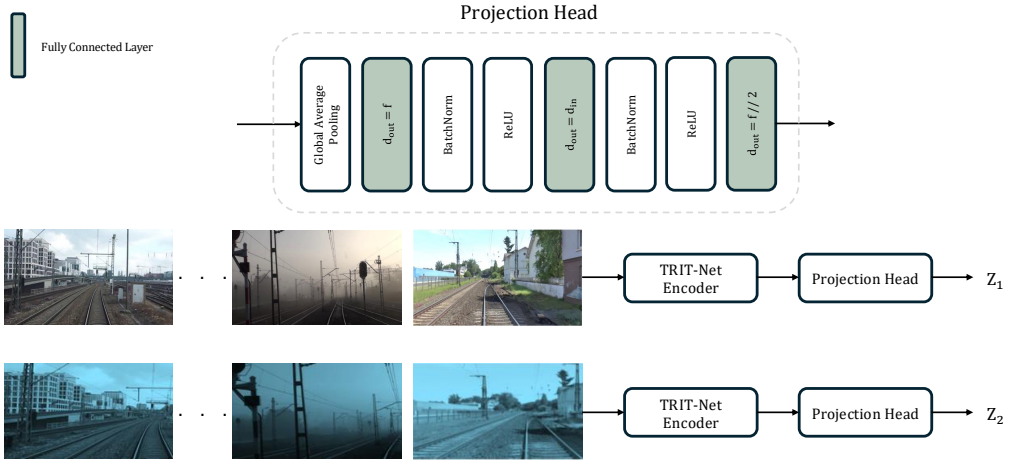


Figure 4.9 VICReg-based self-supervised pretraining architecture. The projection-head parameter is set to $f = 4096$, while d_{in} and d_{out} denote the input and output dimensions of a fully connected layer, respectively.

The TRIT-Net encoder is pretrained on 23,924 unlabeled rail-scene images captured from a North American metropolitan railway network. These images span different weather, illumination, railway environments, and track configurations. As shown in Figure 4.9, the pretraining procedure follows the Variance–Invariance–Covariance Regularization (VICReg) framework [61]. Two augmented views of each image are processed by the encoder and projection head to obtain a pair of representations. The objective encourages agreement between the paired representations, maintains variation across embedding dimensions, and reduces redundancy between feature channels.

Let $Z_1, Z_2 \in \mathbb{R}^{b\times d}$ denote the projection-head outputs for a batch of b samples with d-dimensional embeddings. The total pretraining loss is

$$\mathcal{L}_{\text{VICReg}} = \lambda\mathcal{L}_{\text{inv}} + \mu\mathcal{L}_{\text{var}} + \nu\mathcal{L}_{\text{cov}}, \tag{4.6}$$

where λ, μ, and ν weight the invariance, variance, and covariance terms, respectively. The invariance term minimizes the mean-squared difference between paired embeddings:

$$\mathcal{L}_{\text{inv}} = \frac{1}{b\times d}\sum_{i=1}^{b}\sum_{j=1}^{d}\left(Z_{1,ij} - Z_{2,ij}\right)^2. \tag{4.7}$$

The variance term applies a hinge penalty when the standard deviation of an embedding dimension falls below one:

$$\mathcal{L}_{\text{var}} = \frac{1}{d}\sum_{j=1}^{d}\left[\max(0, 1-\sigma_{1,j}) + \max(0, 1-\sigma_{2,j})\right], \tag{4.8}$$

where

$$\sigma_{k,j} = \sqrt{\frac{1}{b}\sum_{i=1}^{b}\left(Z_{k,ij} - \bar{Z}_{k,j}\right)^2 + \epsilon}, \qquad k \in \{1,2\}, \tag{4.9}$$

$\bar{Z}_{k,j}$ is the batch mean of dimension j in representation k, and ϵ is a small constant used for numerical stability.

The covariance term penalizes squared off-diagonal entries of the covariance matrices:

$$\mathcal{L}_{\text{cov}} = \frac{1}{d} \sum_{k \in \{1,2\}} \sum_{i \neq j} C_k(i,j)^2, \tag{4.10}$$

with

$$\begin{aligned} C_k &= \frac{1}{b-1} \widetilde{Z}_k^\top \widetilde{Z}_k \in \mathbb{R}^{d \times d}, \\ \widetilde{Z}_{k,ij} &= Z_{k,ij} - \bar{Z}_{k,j}. \end{aligned} \tag{4.11}$$

Here, $\widetilde{Z}_k$ denotes the mean-centred representation. During pretraining, only the encoder and projection head contribute to this objective. The decoder and task-specific heads are excluded and are initialized randomly for downstream fine-tuning.

4.4.3 Fine-Tuning for Ego-Path Extraction

After self-supervised pretraining, the learned encoder weights initialize TRIT-Net, and the complete network is fine-tuned end-to-end to predict R, S, and D. The regression maps R and S are supervised using mean absolute error. For a batch of b samples, their losses are

$$\begin{aligned} \mathcal{L}_R &= \frac{1}{b} \sum_{i=1}^{b} \left| R^i - R^i_{\text{GT}} \right|, \\ \mathcal{L}_S &= \frac{1}{b} \sum_{i=1}^{b} \left| S^i - S^i_{\text{GT}} \right|. \end{aligned} \tag{4.12}$$

The Attraction Direction Map is trained using bootstrapped cross-entropy [95] so that optimization focuses on pixels whose cross-entropy exceeds a threshold t_0. Its loss is

$$\mathcal{L}_D = \frac{1}{\tilde{k}} \sum_{i=1}^{b} \sum_{y=1}^{W} \sum_{x=1}^{H} \mathbf{1}\big(\text{CE}^i(x,y) > t_0\big)\, \text{CE}^i(x,y), \tag{4.13}$$

where $\text{CE}^i(x,y)$ is the cross-entropy at pixel (x,y) in sample i, and $\mathbf{1}(\cdot)$ is the indicator

function. The normalization count $\tilde{k}$ is determined by

$$\tilde{k} = \max\left(\tilde{k}_0, \sum_{i=1}^{b}\sum_{y=1}^{W}\sum_{x=1}^{H}\begin{cases}1, & \mathrm{CE}^i(x,y) > t_0,\\ 0, & \mathrm{CE}^i(x,y) \le t_0,\end{cases}\right), \tag{4.14}$$

where $\tilde{k}_0$ provides a lower bound for the normalization count.

This hard-pixel selection reduces the influence of easily classified pixels, including background regions that can dominate dense segmentation losses. No explicit class-frequency weighting is applied.

The three task losses are combined as

$$\mathcal{L} = \alpha\mathcal{L}_R + \beta\mathcal{L}_S + \gamma\mathcal{L}_D. \tag{4.15}$$

The experimental values of the bootstrapping parameters and task weights are specified in Section 4.5.3.

4.5 Datasets, Metrics, and Training Setup

4.5.1 Evaluation Metrics

Ego-path extraction is evaluated at three complementary levels:

- **Track level:** detection of complete ego-path candidates.
- **True-positive-track pixel level:** localization accuracy restricted to correctly matched ego-path candidates.
- **All-track pixel level:** localization accuracy across matched, missed, and false-positive ego-path candidates.

For track-level evaluation, each predicted ego-path candidate is compared with the ground-truth candidates using an F1-based correspondence score computed within a local spatial

tolerance. A correspondence that exceeds the predefined matching threshold is counted as a true positive (TP). An unmatched predicted candidate is counted as a false positive (FP), whereas an unmatched ground-truth candidate is counted as a false negative (FN). Track-level precision and recall are computed for each image from these assignments.

Pixel-level evaluation is performed after candidate matching. For the true-positive-track setting, only pixels belonging to matched predicted and ground-truth candidates are evaluated. Pixel correspondence follows the local-neighbourhood protocol of Yang et al. [8], for which the neighbourhood dimensions were not specified. TPE-Net subsequently used a 12×12-pixel window [14]. The present evaluation fixes Δ to a stricter 8×8-pixel square to reduce the spatial deviation permitted for a match. This window is neither optimized as a model parameter nor claimed to be optimal. Because absolute pixel-level scores depend on the permitted tolerance, every proposed and comparison method is reevaluated using the same 8×8 criterion. Pixel-level precision and recall are then computed. The all-track setting uses the same correspondence rule but additionally includes pixels belonging to FP and FN candidates, thereby accounting for both localization quality and candidate-level detection errors.

For all three evaluation levels, precision and recall are computed per image and averaged over the test set to obtain Average Precision (AP) and Average Recall (AR). Results are also separated by scene type. A *single-candidate scene* contains exactly one annotated ground-truth ego-path candidate, whereas a *multi-candidate scene* contains multiple annotated ground-truth ego-path candidates. This classification depends on the number of annotated ego-path candidates rather than the total number of physical tracks visible in the image. Consequently, an image containing several visible tracks remains a single-candidate scene when only one of them is annotated as a valid ego-path candidate.

Uncertainty for the principal RailSem19 multi-candidate comparison is quantified using a paired nonparametric percentile bootstrap over the 113 test images in that stratum. For each image, the score difference between SSL-pretrained TRIT-Net with AFM tracing and TPE-Net with spatial clustering is computed for four measures: TP-track pixel AP and

AR, and all-track pixel AP and AR. Pairing between the methods is retained throughout. The 113 paired image records are then resampled with replacement 10,000 times using random seed 20260719. For each resample, the mean difference is calculated, and its 2.5th and 97.5th percentiles define the paired-bootstrap 95% confidence interval. These intervals quantify image-level sampling uncertainty conditional on the evaluated checkpoints and test set. They do not estimate variability arising from model retraining. The ablation differences are consequently retained as descriptive effect sizes rather than assigned inferential intervals.

4.5.2 Datasets

The evaluation uses three public railway perception datasets with complementary annotation types and scene characteristics, as summarized in Table 4.2.

Table 4.2 Public datasets used for ego-path candidate extraction, including their image counts, annotation types, and principal forms of scene diversity.

Dataset	#Images	Annotation type	Scene diversity
RailSem19 [58]	8,500	Pixel-wise semantic masks, bounding-box annotations for wayside objects, and left–right rail splines	Various environments worldwide with crossings, switches, and tram scenes
RailDB [59]	7,432	Rail polylines with ordered indices	Sun, rain, night, curves, slopes, crossings, and near/far tracks under nine predefined scenarios
RailSet [60]	7,700	Left–right rail polylines, track-bed regions, and anomaly masks	Diverse weather, lighting, long-range tracks, and synthetic defects captured from multiple real-world scenes

A fourth dataset, referred to in this thesis as the *Private Track Image Dataset*, contains 1,316 forward-facing RGB images and pixel-wise rail-boundary annotations acquired along approximately 500 m of a heritage railway in North America. A forward-looking camera mounted on the vehicle captured the images. The surveyed section contains three main tracks and several switches. Because different switch states permit different vehicle routes through the site, the acquisition is organized into 10 image sets corresponding to these routes.

The dataset is divided into 630 training, 200 validation, and 486 test images. Images were sampled to preserve the distribution of switch-region types across the three splits, thereby supporting evaluation across different switch layouts within the same surveyed site. Because

the route sets share this geographically restricted section and may contain correlated views, the test results represent same-site evaluation under route and switch-state variation rather than fully route-disjoint or sequence-disjoint generalization.

The pixel-wise left and right rail boundaries were generated through a three-stage semi-automatic workflow. First, terrestrial LiDAR scans collected from multiple locations along the target tracks were merged into a unified geo-referenced point cloud. An existing track-point extraction tool was then used to obtain 3D points for the target rails. Second, a prism mounted on the vehicle was tracked by a robotic total station to obtain the geo-referenced (X, Y, Z) camera trajectory, from which the camera pose was estimated for each image. Third, the extracted 3D rail points were projected into each image and converted into pixel-wise left and right rail boundaries for every visible path. This procedure provides the path-specific boundary annotations required to evaluate centerline tracing and left–right rail association in multi-candidate scenes.

Figure 4.10 presents representative raw acquisition images without annotation or model-prediction overlays. The panels illustrate the site appearance, multiple track layouts, switch regions, and vehicle-route viewpoints represented in the dataset.



Figure 4.10 Representative raw input images from the Private Track Image Dataset, showing variations in track layout, switch regions, and vehicle-route viewpoints at the collection site. No annotation or model-prediction overlays are displayed.

4.5.3 Training Setup

To support fair comparisons, all supervised path-extraction baselines use the same training configuration, irrespective of whether ego-path candidates are constructed through AFM-based tracing or spatial clustering. The input size is $H = 540$ and $W = 960$, with batch size

$b = 8$. The hard-pixel parameters are $t_0 = 0.3$ and $\tilde{k}_0 = 8192$, as defined in Equations (4.13) and (4.14), and the task-loss weights are $\alpha = \beta = \gamma = 1$. All compared methods use the fixed dataset partitions unchanged. The RailSem19 partition matches that used for TPE-Net, with 6,000 training, 1,000 validation, and 1,500 test images, thereby preserving a direct split-matched comparison. RailDB is randomly shuffled and divided into 6,000 training/validation images and 1,432 test images while preserving the distribution of switch-region types between the two subsets. For RailSet, only 10% of the complete dataset is publicly available, and the split supplied by the dataset authors is used without modification. The Private Track Image Dataset split described above likewise preserves the distribution of switch-region types across its training, validation, and test subsets.

The evaluation consequently follows a fixed held-out protocol rather than k-fold cross-validation. Repartitioning RailSem19 would prevent a split-matched comparison with TPE-Net, while ordinary random image-level folds would not necessarily improve independence for video-derived data because correlated frames could be assigned to different folds. Robustness is instead examined through fixed held-out evaluation on three public datasets and the complementary same-site Private Track Image Dataset, together with zero-shot transfer from RailSem19 to RailSet. The paired image-level bootstrap described above further quantifies test-image sampling uncertainty conditional on the fixed checkpoints and test set. This protocol is not presented as a substitute for grouped cross-validation.

Supervised optimization uses stochastic gradient descent with an initial learning rate of 0.001, momentum 0.9, a polynomial learning-rate decay schedule, and L_2 weight regularization. Each model is trained for 90,000 iterations on an NVIDIA GeForce RTX 3090 GPU under Ubuntu 20.04 LTS with CUDA 11.6.

VICReg pretraining uses four NVIDIA L40 GPUs, input size 224×224, per-GPU batch size 64, and gradient accumulation over eight steps. The augmented image pairs are generated using the augmentation procedure defined for VICReg [61]. Optimization uses Layer-wise Adaptive Rate Scaling with an initial learning rate of 1.6, cosine annealing, momentum 0.9, L_2 weight regularization, 6,000 warmup iterations, and 90,000 total iterations. The weights

λ, μ, and ν in Equation (4.6) are set to 5, 5, and 0.01 for the invariance, variance, and covariance terms, respectively.

4.6 Results and Analysis

4.6.1 Cross-Backbone Comparison on RailSem19

Table 4.3 compares five encoder–decoder backbones after each is equipped with the centerline-regression and AFM branches. All configurations are trained and evaluated on RailSem19 under the common setup described in Section 4.5.3.

Table 4.3 Cross-backbone comparison on RailSem19. Each model includes the centerline-regression and AFM branches and is trained and evaluated on RailSem19.

Baseline	Latency (ms)	TP-track pixel		All-track pixel		Track level		mIoU			
		AP	AR	AP	AR	AP	AR	Class 0	Class 1	Class 2	Class 3
SegFormer	14.23	0.8183	0.8043	0.8104	0.7832	0.9701	0.9560	0.8408	0.9882	0.8386	0.4741
BiSeNet-V2	6.40	0.8769	0.8723	0.8618	0.8544	0.9603	0.9623	0.8728	0.9912	0.8716	0.5729
TPE-Net	10.35	0.8755	0.8677	0.8659	0.8477	**0.9721**	0.9641	0.8667	0.9908	0.8666	0.5211
ERFNet	6.75	0.8964	0.8880	0.8770	0.8707	0.9565	**0.9653**	**0.8838**	**0.9921**	**0.8828**	**0.5754**
TRIT-Net (SSL-pretrained)	17.74	**0.9032**	**0.8962**	**0.8880**	**0.8764**	0.9644	0.9638	0.8663	0.9902	0.8629	0.5612

The comparison demonstrates that AFM-based tracing is not tied to one backbone, but it does not establish a single architecture as best under every criterion. The SSL-pretrained TRIT-Net leads all four pixel-level AP/AR measures. Relative to the strongest non-TRIT result for each measure, its margins are 0.68 percentage points for TP-track pixel AP, 0.82 points for TP-track pixel AR, 1.10 points for all-track pixel AP, and 0.57 points for all-track pixel AR. These absolute gains are modest but consistent across the four localization measures.

The remaining criteria show different leaders. TPE-Net attains the highest track-level AP, ERFNet attains the highest track-level AR and the strongest direction-map IoU for all four classes, and BiSeNet-V2 has the lowest listed latency. Direction-map IoU measures an intermediate output, whereas final path extraction depends on the interaction of R, D, and S during tracing. ERFNet's higher D-map IoU therefore does not imply higher rail-pixel localization accuracy. The SSL-pretrained TRIT-Net also has the highest listed latency.

Its advantage is consequently concentrated in pixel-level localization rather than universal dominance in semantic segmentation, track-level detection, or computational cost.

4.6.2 AFM-Based Tracing Versus Spatial Clustering

Figure 4.11 isolates the path-construction strategy by evaluating each backbone with either centerline regression followed by spatial clustering or centerline regression combined with the AFM branches. Here, spatial clustering denotes the TPE-Net Rail Path Graph post-processor, which groups extracted track points using geometric association rules and applies degree-2 polynomial fitting to the resulting paths. AFM-based tracing does not use polynomial fitting. It applies the endpoint-distance suppression and fixed-window moving-average smoothing specified in Algorithm 3. AFM-based tracing improves pixel-level localization across both TPE-Net and SSL-pretrained TRIT-Net, with particularly noticeable gains in multi-candidate cases. The track-level results are more mixed: AFM improves the reported track-level precision measures, whereas spatial clustering retains higher multi-candidate track-level recall (M12) for both backbones. The evidence therefore supports AFM primarily as a more accurate pixel-level association and tracing mechanism, rather than as a method that improves every high-level detection metric.

Table 4.4 summarizes the complete desktop-workstation measurements while keeping the GPU network stage separate from CPU post-processing.

Table 4.4 Desktop-workstation inference efficiency. GPU and CPU times are separately measured stage means. Their arithmetic sum and reciprocal throughput are compute-stage estimates rather than synchronized end-to-end measurements. Loading, preprocessing, CPU–GPU transfers, and visualization are excluded. Memory is the highest whole-process value displayed by Mission Center during an approximately one-minute activity window and includes more than the post-processor alone.

Inference pipeline	GPU network (ms/frame)	CPU post-processing (ms/frame)	Sum of stage means (ms/frame)	Approximate throughput (frames/s)	Highest observed memory (GiB)
TPE-Net + Rail Path Graph clustering	10.35	72.7	83.05	12.0	2.24
TRIT-Net + AFM tracing	17.74	40.8	58.54	17.1	1.86

The latency column in Table 4.3 reports only the GPU network forward pass, whereas the CPU post-processing column in Table 4.4 isolates the subsequent path-construction stage.

Measurements were obtained on a desktop workstation equipped with an NVIDIA GeForce RTX 3090 GPU and an AMD Ryzen 9 5900X CPU under Ubuntu 20.04 LTS and CUDA 11.6. Both stages use 540×960-pixel inputs and batch size 1, and the network forward pass uses FP32 arithmetic. Ten warm-up runs precede measurement, after which mean per-image stage times are calculated over the corresponding test-set evaluation. Image loading and preprocessing, CPU–GPU transfers, and visualization are excluded. Application-level CPU thread counts are not fixed. PyTorch, OpenCV, and MKL therefore use their configured backend defaults.

AFM tracing requires 40.8 ms per frame of CPU post-processing, compared with 72.7 ms for Rail Path Graph clustering, corresponding to a 43.9% reduction. Although the TRIT-Net network forward pass is slower, the arithmetic sums of the separately measured stage means are 58.54 ms per frame for TRIT-Net with AFM and 83.05 ms per frame for TPE-Net with Rail Path Graph clustering. These values correspond to approximate compute-stage throughput estimates of 17.1 and 12.0 frames per second, respectively, and a 29.5% lower combined-stage estimate for the TRIT-Net–AFM pipeline.

The GPU and CPU stages were timed separately rather than as a synchronized wall-clock pipeline. Their sums and reciprocal throughput values are therefore derived compute-stage estimates, not measured end-to-end latency or throughput. The highest observed whole-process memory values are 1.86 GiB for TRIT-Net with AFM and 2.24 GiB for TPE-Net with Rail Path Graph clustering, a difference of 0.38 GiB. These values are observations from Mission Center over an approximately one-minute activity window. They are neither profiler-derived peak allocations nor isolated measurements of post-processor memory.

The workstation was not optimized or qualified for on-board deployment. Accordingly, these measurements establish only a comparative compute-stage result under the stated configuration. They do not demonstrate real-time performance or characterize latency, memory, thermal behaviour, or sustained throughput on an embedded or railway-certified platform. Target-device optimization and synchronized end-to-end profiling remain necessary before deployment readiness can be assessed.

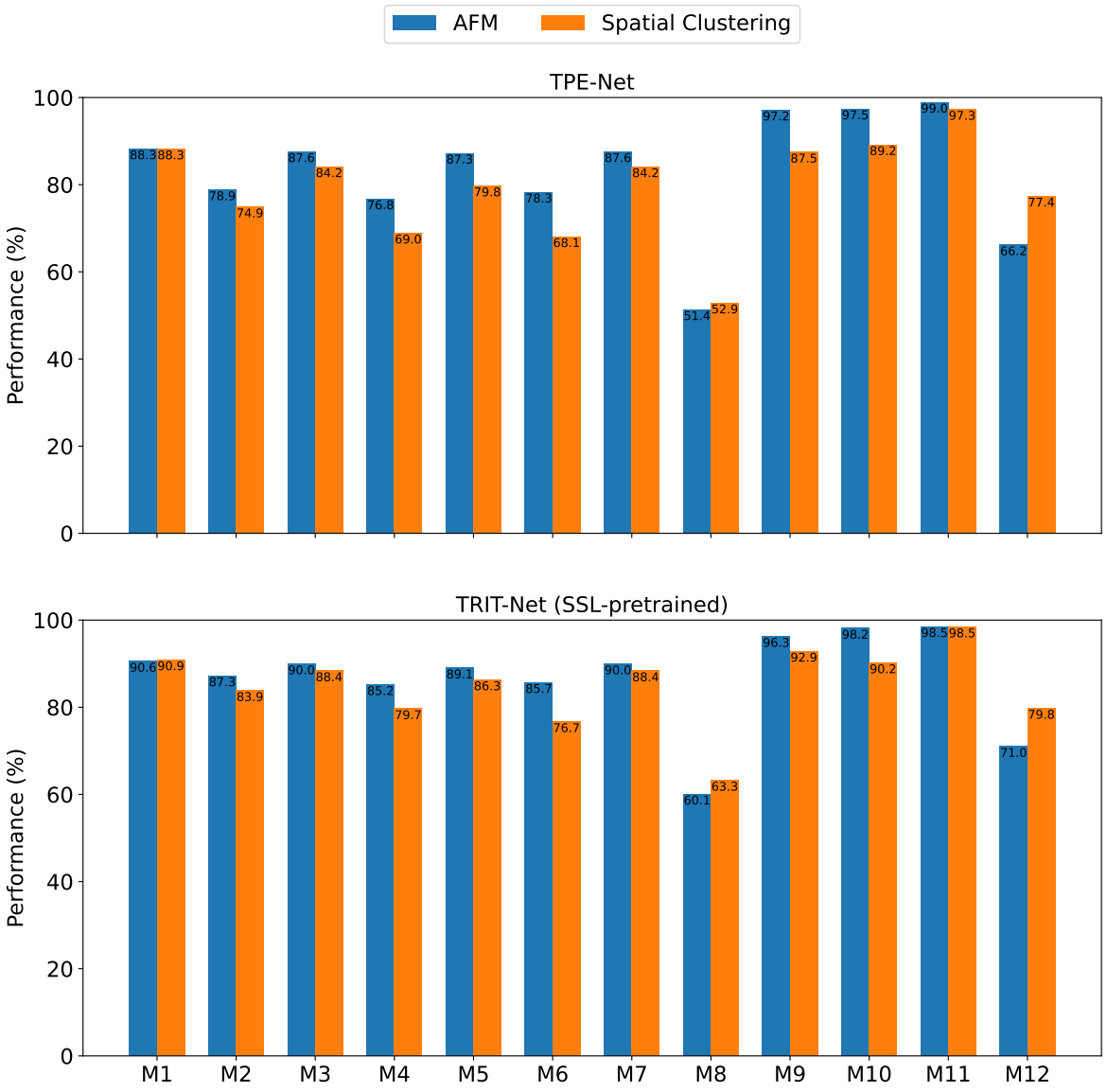


Figure 4.11 AFM-based tracing versus spatial-clustering-based path construction on RailSem19 for TPE-Net and SSL-pretrained TRIT-Net. M1–M4 report TP-track pixel-level performance, M5–M8 report all-track pixel-level performance, and M9–M12 report track-level performance. Within each four-metric group, odd-numbered metrics correspond to single-candidate scenes and even-numbered metrics to multi-candidate scenes; the first two metrics report AP and the final two report AR.

4.6.3 Multi-Dataset Evaluation

Table 4.5 reports within-dataset evaluation on RailSem19, RailDB, RailSet, and the Private Track Image Dataset. For each dataset block, the listed methods are trained and evaluated on that dataset.

Table 4.5 Within-dataset ego-path extraction results on RailSem19, RailDB, RailSet, and the Private Track Image Dataset. The dataset column identifies both the training and evaluation dataset. TPE-Net denotes centerline regression followed by spatial-clustering-based path construction.

Dataset	Method	TP-track pixel level				All-track pixel level				Track level			
		AP		AR		AP		AR		AP		AR	
		single	multi	single	multi	single	multi	single	multi	single	multi	single	multi
RailSem19	TPE-Net	0.8826	0.7493	0.8416	0.6895	0.7977	0.6813	0.8416	0.5290	0.8751	0.8924	0.9732	**0.7740**
	TRIT-Net	0.8844	0.8111	0.8769	0.7951	0.8697	0.8063	0.8769	0.5557	0.9614	0.9774	0.9840	0.6991
	TRIT-Net (SSL-pretrained)	**0.9059**	**0.8726**	**0.8999**	**0.8523**	**0.8906**	**0.8573**	**0.8999**	**0.6012**	**0.9629**	**0.9816**	**0.9855**	0.7104
RailDB	TPE-Net	0.9702	**0.9449**	0.8121	0.7939	0.9645	0.9051	0.8121	0.7075	0.9892	0.9557	0.9953	0.8877
	TRIT-Net	0.9660	0.9059	0.9202	0.7972	0.9637	0.9001	0.9202	0.7613	0.9946	**0.9931**	**0.9976**	**0.9591**
	TRIT-Net (SSL-pretrained)	**0.9752**	0.9238	**0.9265**	**0.8039**	**0.9732**	**0.9083**	**0.9265**	**0.7641**	**0.9949**	0.9829	**0.9976**	0.9540
RailSet	TPE-Net	0.9008	0.9441	0.8223	0.8515	0.8947	0.9441	0.8223	0.4661	0.9709	**1.0000**	0.9776	0.5476
	TRIT-Net	0.9027	0.9301	0.8388	0.8604	0.8989	0.9301	0.8388	0.5353	**0.9866**	**1.0000**	**0.9910**	**0.6309**
	TRIT-Net (SSL-pretrained)	**0.9224**	**0.9689**	**0.8546**	**0.9024**	**0.9207**	**0.9464**	**0.8546**	**0.5666**	0.9799	0.9761	0.9866	**0.6309**
Private	TPE-Net	0.7110	0.7072	0.6698	0.6578	0.6285	0.7072	0.6698	0.5252	0.8874	**1.0000**	**1.0000**	0.8031
	TRIT-Net	0.7365	0.7718	0.7039	**0.7348**	0.7279	0.7649	0.7039	**0.7327**	0.9482	0.9946	0.9671	**0.9973**
	TRIT-Net (SSL-pretrained)	**0.8341**	**0.7721**	**0.8003**	0.7341	**0.8339**	**0.7721**	**0.8003**	0.7179	**0.9991**	**1.0000**	**1.0000**	0.9760

Across the four datasets, both TRIT-Net variants generally report higher pixel-level precision and recall than TPE-Net. The SSL-pretrained variant attains the highest number of best-performing scores and leads most pixel-level measures. Because TPE-Net and TRIT-Net differ in both architecture and path-construction strategy, Table 4.5 should be interpreted as a whole-system comparison rather than as an isolated measurement of the AFM contribution. The controlled comparison in Figure 4.11 isolates the path-construction strategy, whereas the comparison between the two TRIT-Net variants in Table 4.5 indicates the additional effect associated with VICReg initialization.

The clearest margins occur in RailSem19 multi-candidate scenes. Relative to TPE-Net, the SSL-pretrained TRIT-Net improves TP-track pixel AP by 12.33 percentage points, TP-track pixel AR by 16.28 points, all-track pixel AP by 17.60 points, and all-track pixel AR by 7.22 points.

Table 4.6 reports the paired image-level uncertainty analysis for the principal RailSem19 multi-candidate comparison using the procedure defined in the evaluation protocol.

Table 4.6 Paired image-level uncertainty analysis for SSL-pretrained TRIT-Net with AFM relative to TPE-Net with spatial clustering on the 113 RailSem19 multi-candidate test images. Differences and intervals are reported in percentage points; positive values favour TRIT-Net. Intervals are based on 10,000 paired percentile-bootstrap resamples.

Metric	Mean difference (percentage points)	Paired-bootstrap 95% CI
TP-track pixel AP	12.33	$[10.72, 13.89]$
TP-track pixel AR	16.28	$[14.96, 17.56]$
All-track pixel AP	17.60	$[15.92, 19.26]$
All-track pixel AR	7.22	$[5.87, 8.54]$

All four confidence intervals in Table 4.6 remain above zero. The fixed SSL-pretrained TRIT-Net checkpoint scores higher than the fixed TPE-Net checkpoint on 104 of 113 images for TP-track pixel AP, 112 for TP-track pixel AR, 109 for all-track pixel AP, and 95 for all-track pixel AR. The intervals therefore support positive mean image-level localization differences for this fixed-checkpoint, fixed-test-set comparison. They do not establish universal per-image superiority or quantify variation across independent training runs.

The gains are not uniform across metrics or datasets. TPE-Net retains the highest RailSem19 multi-candidate track-level recall and the highest RailDB multi-candidate TP-track pixel AP. On RailDB, the TRIT-Net variants lead most recall-oriented and track-level measures, but the best values are divided between the randomly initialized and SSL-pretrained variants. On RailSet, the SSL-pretrained model leads the multi-candidate TP-track pixel AP and AR and all-track pixel AR, while TPE-Net and non-pretrained TRIT-Net share the highest multi-candidate track-level AP. On the Private Track Image Dataset, the SSL-pretrained model gives the strongest single-candidate pixel-level results, whereas non-pretrained TRIT-Net leads the multi-candidate TP-track, all-track, and track-level recall measures. Self-supervised pretraining therefore most consistently improves pixel localization, but does not guarantee the highest path-level score in every dataset or scene category.

The absolute results also remain domain-dependent: RailDB yields comparatively higher scores, whereas the Private Track Image Dataset is more challenging because it contains a large proportion of multi-track segments and complex switch regions. Because every method in Table 4.5 is retrained on its corresponding dataset, these results assess cross-dataset

consistency rather than zero-shot transfer. The private-dataset results remain same-site evidence rather than external-site generalization evidence.

The reported values are aggregates over the available test images and should not be interpreted as condition-stratified evidence. Although RailDB provides coarse scenario groupings and switch-region categories were used when forming the RailDB and private-data partitions, the evaluated datasets do not share an evaluation-ready set of per-image labels for ego-path switch topology, weather, illumination severity, or rail occlusion. Reliable subgroup analysis would require a separate annotation protocol, ambiguity resolution, quality control, and adequate sample counts for each condition. Subjective post-hoc grouping is therefore avoided, and no quantified robustness claim is made for these operating-condition subgroups.

4.6.4 Zero-Shot Generalization

Table 4.7 evaluates transfer from RailSem19 to RailSet without target-domain fine-tuning.

Table 4.7 Zero-shot evaluation on RailSet. All methods are trained exclusively on RailSem19 and evaluated on RailSet without target-domain fine-tuning. TPE-Net denotes centerline regression followed by spatial-clustering-based path construction.

Method	TP-track pixel level				All-track pixel level				Track level			
	AP		AR		AP		AR		AP		AR	
	single	multi	single	multi	single	multi	single	multi	single	multi	single	multi
TPE-Net	0.8852	**0.9401**	0.8017	0.8294	0.8389	**0.9401**	0.8017	0.4733	0.9263	**1.0000**	0.9776	0.5714
TRIT-Net	0.8724	0.9008	0.8168	0.8415	0.8724	0.8784	0.8168	**0.5974**	**0.9821**	0.9761	**0.9811**	**0.7023**
TRIT-Net (SSL-pretrained)	**0.8935**	0.9154	**0.8271**	**0.8552**	**0.8884**	0.8778	**0.8271**	0.5689	0.9657	0.9523	0.9732	0.6666

The zero-shot results are mixed. TPE-Net retains the highest multi-candidate AP at the TP-track pixel, all-track pixel, and track levels, showing that the spatial-clustering baseline preserves higher precision for some target-domain candidates. The TRIT-Net variants instead lead most recall-oriented measures. In particular, non-pretrained TRIT-Net raises multi-candidate all-track pixel AR from 0.4733 to 0.5974 and multi-candidate track-level AR from 0.5714 to 0.7023 relative to TPE-Net. The SSL-pretrained variant leads several other pixel-level measures, including single-candidate precision and both single- and multi-candidate TP-track recall. The distribution of best values between the two TRIT-Net variants also shows that the observed transfer cannot be attributed exclusively to self-supervised pretraining.

This pattern reflects a precision–recall tradeoff rather than overall superiority of TPE-Net in multi-candidate scenes. One plausible explanation is that its fixed spatial-association rules behave conservatively under the RailSem19-to-RailSet domain shift: uncertain branches are more likely to be rejected, which can limit false-positive candidates while also increasing missed paths. The learned attraction field instead recovers a larger proportion of the candidate-path evidence, as indicated by its higher recall, but target-domain errors in the predicted direction or intensity maps can introduce incorrect associations and reduce precision. The tradeoff is more visible in multi-candidate scenes because paths share nearby rail and track-bed evidence before separating farther from the train. This interpretation is consistent with the measured precision and recall, although the whole-system comparison changes both the network architecture and the path-construction mechanism and therefore does not isolate the cause experimentally.

Table 4.7 therefore provides evidence of partial transfer rather than uniform cross-domain superiority. Learned row-wise association reduces some dependence on dataset-specific Rail Path Graph clustering thresholds, particularly for candidate recall and continuity, but the remaining precision gaps show that the learned representation still depends on the RailSem19 training-domain distribution. Broader cross-domain validation and condition-stratified evaluation are required before making a more general robustness claim.

4.6.5 Attraction-Intensity Ablation

Table 4.8 isolates the contribution of the Attraction Intensity Map by replacing S with the constant value 1 while retaining the remaining model components.

Table 4.8 Attraction-intensity ablation on RailSem19. The Attraction Intensity Map is replaced by the constant value 1. Entries are performance differences relative to the complete model; negative values denote degradation.

Ablation variant	TP-track pixel level		All-track pixel level		Track level	
	ΔAP (%)	ΔAR (%)	ΔAP (%)	ΔAR (%)	ΔAP (%)	ΔAR (%)
TRIT-Net	-2.04	-2.73	-2.09	-2.72	-0.20	-0.02
TRIT-Net (SSL-pretrained)	-2.88	-2.57	-2.80	-2.55	-0.17	-0.16

Across both TRIT-Net variants, removing the learned intensity values produces a pixel-

level degradation of approximately 2–3%, while the track-level changes are comparatively small. Coarse path detection can remain largely intact, but fine-grained localization and row-to-row propagation deteriorate without learned displacement magnitudes.

4.6.6 Qualitative Analysis and Failure Cases

The four scenes in Figure 4.12 were deliberately selected because differences among the configurations are visually discernible. They illustrate the mechanisms underlying the aggregate trends rather than constituting statistically representative evidence. The full-test-set results in Tables 4.5 and 4.7 remain the primary evidence.

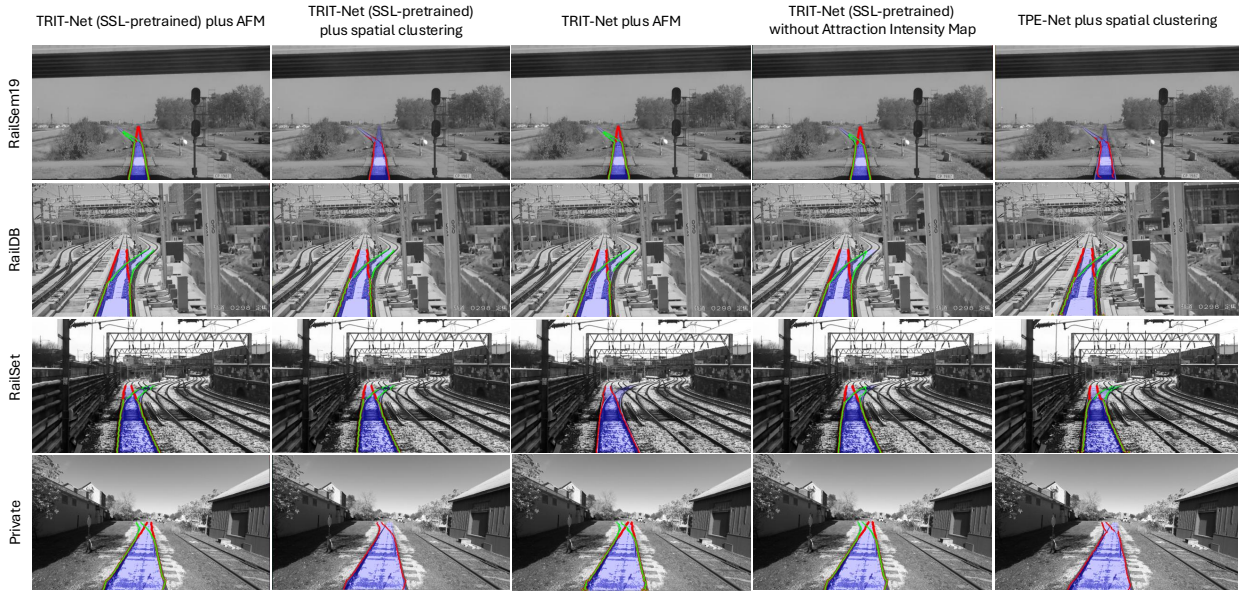


Figure 4.12 Qualitative comparison comprising 20 model-output panels. Rows show, from top to bottom, RailSem19, RailDB, RailSet, and the Private Track Image Dataset. Columns show, from left to right: SSL-pretrained TRIT-Net with AFM-based tracing; SSL-pretrained TRIT-Net with spatial-clustering post-processing; TRIT-Net trained from random initialization with AFM-based tracing; SSL-pretrained TRIT-Net with the Attraction Intensity Map removed; and TPE-Net with spatial clustering. Each row holds the input scene fixed across the five configurations. True-positive predicted ego-path candidates are shown as distinct coloured rail boundaries, and the ground-truth track bed is shown in blue. The visualizations display raw extracted rail pixels before moving-average smoothing.

The figure supports three direct comparisons. The first and second columns isolate AFM-

based tracing from spatial clustering under the same SSL-pretrained TRIT-Net backbone. AFM tracing produces more complete trajectories in densely overlapping multi-candidate scenes. The first and third columns compare SSL pretraining with random initialization and show improved discrimination between closely spaced paths. The first and fourth columns isolate the Attraction Intensity Map, whose removal weakens tracing particularly along paths with stronger lateral curvature. This effect is less pronounced in the Private Track Image Dataset scene, where the track geometry is comparatively milder. The fifth column provides the TPE-Net spatial-clustering baseline. Because the displayed outputs precede moving-average smoothing, the examples expose differences in the underlying tracing and localization behavior rather than differences introduced by that smoothing stage.

4.6.7 Result Synthesis

Taken together, the experiments support five bounded conclusions. First, AFM can be integrated with multiple encoder–decoder backbones, and the SSL-pretrained TRIT-Net provides modest but consistent pixel-level margins over the strongest non-TRIT results. Other backbones nevertheless lead semantic, track-level, or latency criteria. Second, replacing Rail Path Graph clustering with AFM most consistently benefits pixel-level association and CPU post-processing time, while clustering retains selected track-level recall advantages. Third, the RailSem19 multi-candidate bootstrap supports positive mean pixel-level differences for the evaluated fixed checkpoints, but the per-image outcomes are not uniform and the other datasets retain metric-specific exceptions. Fourth, VICReg pretraining generally strengthens pixel-level performance across datasets but does not dominate every track-level or zero-shot metric. Fifth, the intensity ablation shows that S primarily contributes to fine-grained localization and stable propagation rather than to coarse candidate detection. The evidence therefore supports attraction-field tracing as a trainable path-construction strategy that reduces reliance on heuristic clustering across the evaluated settings, without implying superiority under every metric, dataset, training run, or operating condition.

4.7 Limitations and Chapter Discussion

The experiments show improvements in several path-association and tracing measures, but the method still depends on visual evidence from a single forward-facing RGB image. The observed failure cases arise when the image does not provide sufficiently clear track-point or branching cues, or when the visible infrastructure differs substantially from the configurations represented during training.

4.7.1 Severe occlusion

Large foreground objects, vegetation, or prolonged obstruction of the camera view can conceal the rails required to localize and associate track points. Under these conditions, R, D, and S may not provide reliable continuation cues. The tracing procedure can consequently terminate before reaching the far field or select an incorrect branch in a shared track-bed region. Premature termination removes the remaining candidate continuation and therefore primarily reduces recall. This limitation applies even to the SSL-pretrained TRIT-Net.

4.7.2 Extreme long-range ambiguity

Near the vanishing point, adjacent rails can converge to separations that provide insufficient visual evidence for distinguishing closely spaced ego-path candidates. AFM tracing may continue through some such regions, but candidate separation can degrade in long-range multi-candidate scenes, particularly when images are captured using short-focal-length cameras. A candidate may therefore remain detectable at the track level even when its far-field rail pixels cannot be localized accurately. This is an image-evidence limitation rather than one that can be resolved solely through a different row-to-row association rule.

4.7.3 Atypical switch geometries

Switch layouts that differ substantially from those observed during training, including double-slip and diamond crossings, can produce incorrect branching decisions. Self-supervised pretraining improves resilience relative to random initialization in the reported qualitative cases, but it does not remove this failure mode. The result identifies a limitation of inferring switch-region structure from appearance alone without explicit switch-state modeling.

4.7.4 Decoder topology and output growth

The candidate-construction procedure is restricted to the near-field-connected paths represented by a bottom-center seed, at most one triplet per image row, monotonic tracing toward the horizon, and successive binary Class 3 divergences. It is not a general graph-search decoder for explicit merge recombination, non-binary junctions, looped connectivity, or arbitrary crossings. Moreover, the number P of candidates before suppression is output-sensitive and can grow exponentially with the number of sequential binary divergence regions in an unconstrained worst case. These restrictions limit the railway topologies represented explicitly, while output-sensitive branching can increase decoder cost sharply in complex divergence sequences.

4.7.5 Private-dataset scope and reproducibility

The Private Track Image Dataset covers a single, geographically restricted railway section of approximately 500 m. Its three-track layout and switch configurations do not represent the structural diversity of wider railway networks, and the route-based image sets may contain correlated views from the same site. The associated results should therefore be interpreted as same-site evaluation under route and switch-state variation rather than as route-disjoint, sequence-disjoint, or external-site generalization evidence.

The semi-automatic annotations also inherit uncertainty from point-cloud registration and track-point extraction, prism-based camera-pose estimation, and projection of the 3D

rail geometry into the image plane. These effects can be more pronounced for distant or partially occluded rails. The projected boundaries are therefore treated as semi-automatic annotations rather than error-free ground truth, and no measured pixel-level uncertainty bounds are claimed. Because the complete dataset is not publicly available, independent reproduction of its dataset-specific results is limited. The public benchmarks, particularly RailSem19, consequently provide the primary external-dataset evidence in this chapter.

4.7.6 Evaluation-protocol scope

The fixed held-out partitions preserve direct split-matched comparison with TPE-Net, but they are not a substitute for grouped cross-validation and do not measure variability across independent training runs. Evaluation across several held-out datasets, the same-site private domain, and RailSem19-to-RailSet transfer broadens the evidence, while the paired bootstrap quantifies only test-image sampling uncertainty for fixed checkpoints. In addition, the datasets do not share evaluation-ready per-image labels for switch topology, weather, illumination severity, or rail occlusion. The aggregate results therefore do not establish condition-specific robustness.

4.7.7 Deployment evidence

The efficiency measurements separate the GPU network and CPU post-processing stages on a desktop workstation and exclude loading, preprocessing, transfers, and visualization. They are not synchronized end-to-end measurements, and the sampled whole-process memory values are not profiler-derived post-processor allocations. The results consequently do not establish real-time performance, embedded-platform latency, thermal behaviour, sustained throughput, or suitability for a railway-certified on-board system.

4.7.8 Future directions

The resulting research directions are broader modeling of complex railway topologies and switch semantics, temporal tracking, multimodal sensor fusion, uncertainty estimation, broader field validation, and efficient implementation and profiling on on-board hardware. None of these directions is evaluated in this chapter, and they should not be interpreted as demonstrated remedies for the observed failures.

Relative to the route-assembly procedure in Chapter 3, TRIT-Net replaces heuristic clustering with learned path tracing and is evaluated across a broader range of architectures, datasets, and training regimes. Image-only tracing nevertheless remains one perception component rather than a complete autonomy solution. Its output is the set of near-field-connected ego-path candidates returned under its tracing assumptions rather than a unique commanded route. Determining which candidate is operationally valid requires information beyond path continuity alone, particularly explicit interpretation of switch configuration. This distinction motivates the later camera-based treatment of route understanding and reasoning.

Chapter 5

Reasoning-Guided Ego-path Segmentation with Vision-Language Models

Chapters 3 and 4 advanced camera-based ego-path perception from bottom-up proposal generation to learned path tracing. Both approaches recover plausible visible paths but leave unresolved which branch a switch enables. This chapter addresses that route-selection problem using switch geometry and language-conditioned segmentation. It is based on a published paper [96].

5.1 Chapter Overview

This chapter adapts the Large Language Instructed Segmentation Assistant (LISA) to railway ego-path perception and formulates the task as *reasoning-guided segmentation*: given a forward-facing railway image and a natural-language query, the model predicts the valid ego-path mask and, when prompted, generates a textual rationale intended to describe visible switch geometry [65]. Here, *reasoning-guided* refers to the use of route-logic prompts and explanatory supervision. It does not imply that the generated text is a faithful account of the internal process that produces the mask.

The contributions of this chapter are the formulation of railway switch understanding as reasoning-guided ego-path segmentation; railway-specific adaptation of LISA through switch-aware prompts, polygon ego-path masks, and explanatory supervision; and an ini-

tial RailSem19-based experimental protocol for evaluating segmentation performance, task-conditioned prompt sensitivity, and route-logic consistency between masks and generated rationales. The implementation code and reasoning-segmentation data are available at `https://github.com/mvakili96/Railway_Perception_FoundationModel`.

5.2 Motivation Beyond Earlier Camera-Based Chapters

Reliable ego-path perception is required when train automation is extended beyond constrained settings toward more open and infrastructure-light railway environments [62]. In switch regions, this requirement is difficult because several rails may be visible even though only one route remains physically continuous for the train.

When several candidate paths remain visible, route validity depends on structured cues that are not equivalent to generic rail segmentation alone. The physically valid route depends on switch evidence such as which blade is closed against its stock rail, which branch preserves rail continuity, and which alternative path is blocked by an open gap between the rails. Route understanding in switch regions is therefore not only a segmentation problem but also a reasoning problem over visible track geometry.

A vision-language formulation allows the query to request identification of the train's valid forward route. The primary aim is to condition route segmentation on switch-aware language, while also investigating whether the mask can be accompanied by an interpretable textual rationale.

The intended use case is switch-aware route interpretation within a forward-looking railway perception pipeline. Given a railway image and a route-focused query, the model produces a mask of the route that appears physically valid and can provide an auxiliary rationale referring to visible switch geometry. The language-model component is needed because the target depends on an instruction about route validity and visual relationships, including blade closure and rail continuity, rather than a fixed semantic class alone. It conditions the visual segmentation on this route-specific task and enables a textual output that can be

reviewed alongside the predicted mask. The method does not make train-control decisions, and its reported results do not establish operational safety.

5.3 Positioning in Prior Work

5.3.1 Railway Scene Understanding and Ego-path Perception

Public railway datasets remain limited, and rail environments have domain-specific structures [23]. RailSem19 provides a public benchmark for semantic railway scene understanding from ego-perspective train and tram imagery [58]. Existing railway vision pipelines predict rails, track centers, or ego-track regions from monocular or multi-camera images and may use geometric constraints, post-processing, or task-specific decoders to recover a route. Subsequent methods move from semantic labels toward route-level perception. TPE-Net associates left and right rail pixels into route instances, while TEP-Net formulates train ego-path detection as an end-to-end task for identifying the immediate track ahead [14, 13]. Track2Net similarly uses keypoint alignment and track-anchor identification for railway-line tracking [51].

Annotation and validation present additional constraints. Labels4Rails addresses railway-image annotation and emphasizes the scarcity of switch-aware labels, while N-version railway segmentation studies architectural redundancy as a means of improving robustness [78, 79]. These studies demonstrate progress in rail and ego-path perception, but their primary objectives remain visual prediction and segmentation accuracy. They do not generate a textual route rationale that can be inspected alongside the prediction. The availability of such text does not itself establish operator trust or safety assurance. Its consistency with the visual evidence and predicted mask must be evaluated separately.

5.3.2 Vision-Language Models for Language-Conditioned Perception

The reasoning-guided formulation builds on three developments in multimodal perception. LLaVA connects visual features to a large language model through visual instruction tuning,

SAM provides prompt-conditioned mask generation, and LISA combines a multimodal language model with SAM by decoding the hidden representation of a dedicated `[SEG]` token into a segmentation mask [63, 64, 65].

LISA belongs to a developing family of segmentation-oriented multimodal models. LISA++ strengthens the baseline through additional training data and visual chain-of-thought supervision, GSVA extends the formulation to multiple or absent referents, and See, Say, and Segment studies false-premise queries and language-based rejection of invalid segmentation requests [68, 69, 70]. These methods are primarily evaluated on generic referring-expression and reasoning-segmentation benchmarks. At the time of this study, reasoning segmentation had not previously been adapted to railway switch interpretation, where the target is a physically valid route and the generated rationale is intended to refer to blade closure, rail continuity, and switch topology.

5.4 Reasoning-Guided Segmentation Framework

5.4.1 Task Formulation

In this chapter, a *track* is defined as one left rail paired with one right rail, and the *track bed* is the region bounded by those rails. A railway *path* or *route* is the track together with its corresponding track bed. A *switch region* is a railway section that enables transitions between paths. A turnout branches one path into another, whereas a merge combines two paths into one. An *ego-path candidate* is a route that extends forward from the track currently occupied by the camera-carrying train. In the forward view, this upstream ego reference is associated with the near-field track entering the lower central image region. The *active ego-route* is the ego-path candidate that constitutes the train's physically continuous forward connection. When multiple candidates occur at a switch, it is the candidate enabled by the observed switch configuration.

Switch-dependent selection is the central focus of this task. The model predicts the active route directly rather than explicitly reconstructing the complete candidate set as an

intermediate output.

The chapter formulates reasoning-guided railway ego-path segmentation as follows. Given a forward-looking RGB image $I \in \mathbb{R}^{H \times W \times 3}$ captured by a train-mounted camera and a natural-language query Q, the model predicts a binary mask $M \in \mathbb{R}^{H \times W}$ corresponding to the active ego-route. When the query requests justification, the model additionally produces a free-form rationale A intended to describe visible switch geometry. Unlike ordinary rail or track-bed segmentation, the target is not simply "rails" or "track," but the track bed of the candidate that remains physically valid for the train based on route continuity and, where applicable, switch state.

In switch scenes, the supervised rationale is expected to identify which blade is closed, which branch remains continuous, and which upstream path connects into the downstream mainline. The mask and rationale are paired outputs of the same route-interpretation formulation, but no explicit mechanism enforces agreement between them.

5.4.2 Railway Adaptation of LISA

The method builds on LISA, which combines a LLaVA-style multimodal language backbone with the promptable mask-generation capability of Segment Anything Model (SAM) [65, 64, 63]. Figure 5.1 presents the original LISA architecture that provides the starting point for the railway adaptation. LISA treats segmentation as a language-conditioned generation problem: the model processes an image–text conversation and emits a special `[SEG]` token when a grounded mask should be produced.

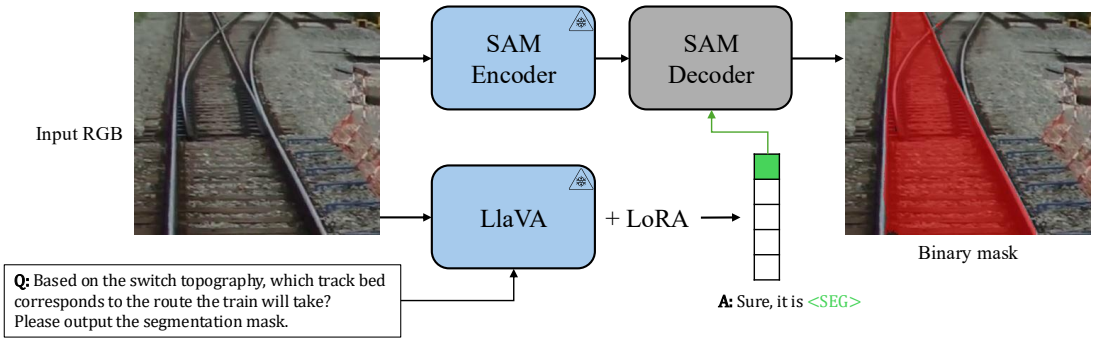


Figure 5.1 Original LISA architecture for reasoning segmentation, reproduced from [65]. In the illustrated LISA configuration, the SAM image encoder and LLaVA vision backbone are frozen, while LoRA is applied to the language-model attention projections [64, 63, 66]. The railway adaptation retains this architecture but uses the selective training strategy described in the accompanying text.

The `[SEG]` token has two roles. First, it acts as a mask carrier: the hidden state at the token position is projected into a prompt embedding for the SAM prompt encoder and mask decoder. Second, it acts as a gate, because a segmentation mask is generated only when the model actually emits `[SEG]`. If the token is absent, the system remains in text-only mode. This mechanism is retained for the railway domain so that route-specific prompts can trigger valid ego-path segmentation together with an optional textual rationale.

This gating behavior reflects the original LISA training mixture. The base model is jointly optimized on semantic segmentation, referring segmentation, visual question answering (VQA), and reasoning segmentation. The reasoning-segmentation portion uses 239 sample images to teach implicit queries, world knowledge, explanatory answers, and reasoning-intensive segmentation requests. The same decoder can therefore support both language-only responses and responses that include a mask.

The railway finetuning used in this chapter does not retain the VQA stream, and most railway prompts explicitly request ego-path segmentation. The adaptation consequently focuses on learning which route should be masked rather than primarily learning whether a mask should be emitted. The `[SEG]` gating mechanism nevertheless remains part of the architecture and supports both text-only explanatory supervision and mask-producing

responses.

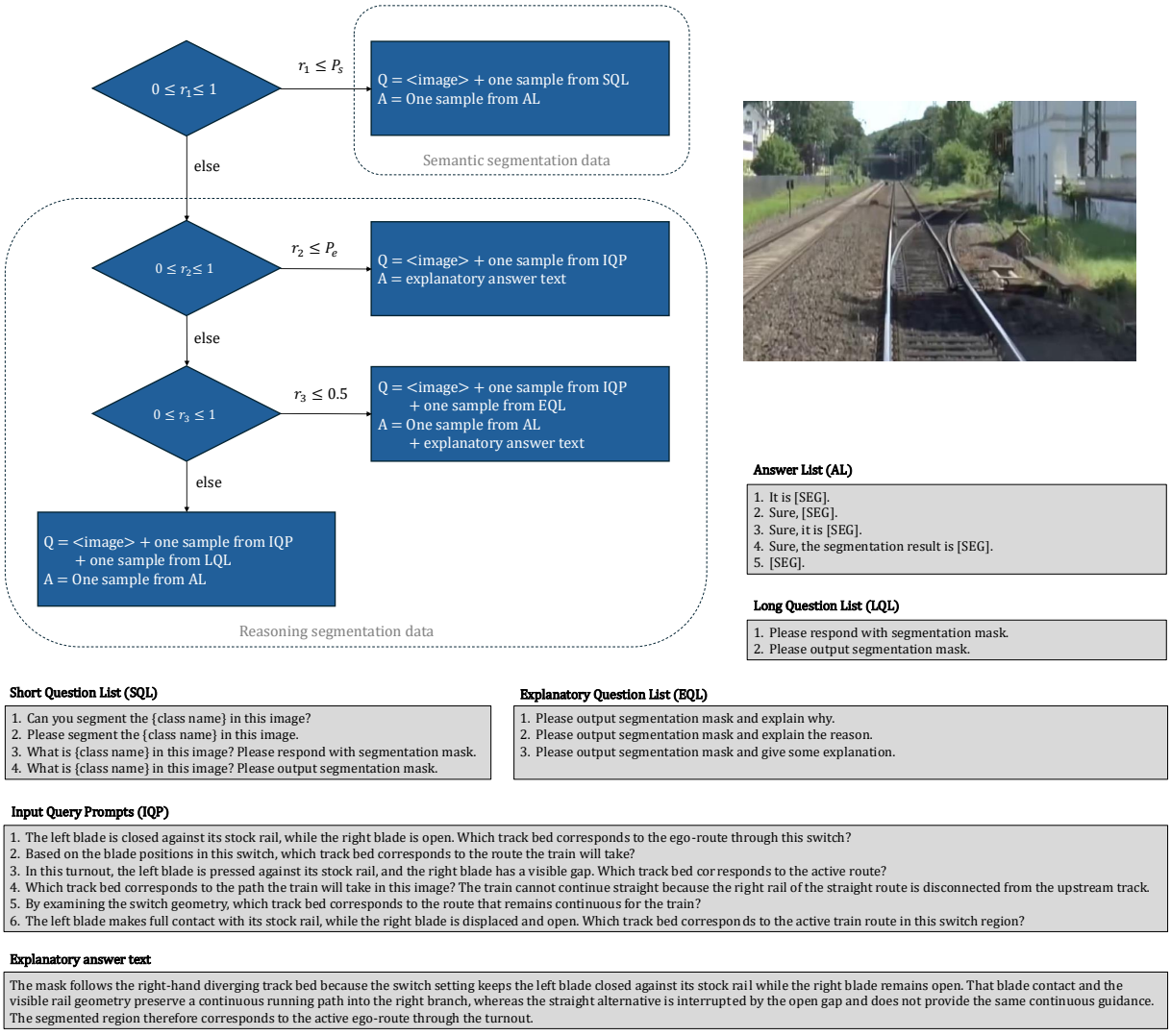


Figure 5.2 Flowchart for multimodal finetuning. The figure illustrates how semantic segmentation and railway reasoning-segmentation samples are mixed during training, together with the role of the semantic-sampling probability P_s and the explanatory-sampling probability P_e. Counterfactual image–language augmentation, which precedes the illustrated encoder paths, is described in the text.

The railway adaptation retains the LISA architecture but changes the original freezing strategy. Its trainable parameter groups comprise LoRA adapters in the query and value projections of the language-model attention layers together with the language token em-

beddings and output head; the language-to-SAM projection, SAM prompt encoder, SAM mask decoder, and final 16 transformer blocks of the SAM image encoder; and the LLaVA multimodal projector and final eight CLIP transformer blocks that contribute to the visual feature selected by LLaVA. The remaining pretrained parameters, including the earlier visual blocks and the SAM patch embedding and neck, remain frozen. The resulting strategy is selective multimodal adaptation rather than either full-model finetuning or a frozen-vision, LoRA-only configuration [66].

5.4.3 Railway-Specific Prompting and Supervision

The railway adaptation extends LISA with switch-aware prompt templates, polygon ego-path masks, and explanatory supervision aligned with turnout and merging logic. Figure 5.2 summarizes the multimodal finetuning process.

An important design choice is that the semantic segmentation stream is still expressed in language-conditioned form rather than as a conventional multi-class pixel classifier. The model does not predict one multi-class mask through a single softmax over all categories. Instead, the data loader samples a class name, poses a corresponding segmentation prompt, and converts the semantic label map into a binary one-vs-all target for the queried class. This keeps the semantic supervision aligned with the later reasoning stage, because both are expressed as text-conditioned binary mask prediction problems.

As illustrated in Figure 5.2, random variables r_1–r_3 determine which training objective and response form are sampled. The probability P_s controls selection of semantic segmentation rather than railway reasoning segmentation, while P_e controls whether an eligible reasoning sample is used for explanation without mask prediction. This text-only mode supervises free-form route-rationale generation when the query emphasizes explanation, while the remaining reasoning samples teach responses that include `[SEG]` and an associated mask.

Before the CLIP and SAM preprocessing paths, each railway reasoning sample is independently subjected to counterfactual horizontal flipping with probability 0.5. When a flip is applied, the image and ego-path mask are flipped together, while left–right references

and the corresponding blade-state descriptions in the language supervision are transformed consistently. This augmentation is intended to reduce dependence on a fixed image-side convention while preserving agreement among the image, target mask, query, and explanation. It is applied stochastically during loading and therefore does not change the 242 unique-image counts reported in Table 5.2.

The training objective combines an autoregressive language cross-entropy loss with binary cross-entropy and Dice mask losses:

$$\mathcal{L} = \mathcal{L}_{\text{CE}} + 2\mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{Dice}}. \tag{5.1}$$

The language loss teaches the model to produce the expected response format, including use of `[SEG]` and, when requested, the corresponding rationale text. Within $\mathcal{L}_{\text{CE}}$, target tokens identifying the switch type and the side of the open blade are each assigned a weight of 60, while all other target tokens retain a weight of 1. This weighting emphasizes the sparse route-critical attributes that could otherwise be diluted by the more numerous ordinary response tokens. The two mask losses supervise the predicted ego-path mask.

For the binary ego-path masks, the Dice term complements binary cross-entropy by directly optimizing foreground-mask overlap, reducing the tendency of the more numerous background pixels to dominate the mask supervision.

5.5 Dataset Construction and Finetuning Strategy

Table 5.1 consolidates the composition, provenance, native dimensions, and encoder-specific preprocessing of the data used for training, validation, and testing. The 1024×1024 reasoning samples are categorized as wider-context views, while the remaining variable-resolution samples are categorized as close-up switch-region views.

Table 5.1 RailSem19 data composition and encoder-specific image preprocessing.

Data type (n)	**RailSem19 source and selection**	**Input dimensions**	**Encoder-specific preprocessing**
Semantic training (6,000)	First 6,000 images; all included.	Unscaled 1024×1024 crops; horizontally centered, bottom-aligned	**CLIP:** shortest side to 224; 224×224 center crop; normalize. **SAM:** normalize; no resizing or padding.
Reasoning: close-up (126)	Qualifying switch scenes from images 1–4,000; annotated ego-path traverses a switch region whose configuration is visually discernible; per-image ROI retains the switch mechanism and relevant route geometry.	Width: 170–1266; height: 102–659 pixels	**CLIP:** shortest side to 224; 224×224 center crop; normalize. **SAM:** if needed, resize the longest side down to 1024 while preserving aspect ratio; normalize; zero-pad to 1024×1024. Images are never upscaled.
Reasoning: wider context (116)	Qualifying switch scenes from images 1–4,000 under the same visible-switch criterion; surrounding context retained.	1024×1024	**CLIP:** shortest side to 224; 224×224 center crop; normalize. **SAM:** normalize; no resizing or padding.
Validation (500)	RailSem19 images 8,001–8,500; held out from training.	Unscaled 1024×1024 crops; horizontally centered, bottom-aligned	**CLIP:** shortest side to 224; 224×224 center crop; normalize. **SAM:** normalize; no resizing or padding.
Test (2,000)	RailSem19 images 6,001–8,000; held out from training and validation.	Unscaled 1024×1024 crops; horizontally centered, bottom-aligned	**CLIP:** shortest side to 224; 224×224 center crop; normalize. **SAM:** normalize; no resizing or padding.

5.5.1 Railway Reasoning Dataset

To adapt LISA to railway ego-path reasoning, a specialized reasoning-segmentation dataset is constructed from the first 4,000 RailSem19 images [58]. This source range is inspected manually, and every image for which the annotated ego-path traverses a switch region and the switch configuration is visually discernible to a human annotator is selected. Close-up views are produced by cropping the switch region of interest so that varied switch configurations and local blade–stock geometry occupy a larger portion of the input to the CLIP vision encoder.

Crop boundaries and dimensions are adjusted manually for each image without resizing the source pixels. Therefore, no fixed minimum apparent size or pixel threshold is imposed. Each candidate crop is checked after applying CLIP's shortest-side resize and subsequent 224×224 center-crop geometry. If the visible switch mechanism or the route geometry required by the supervision falls outside the final CLIP crop, a revised crop is taken directly from the raw image and checked again. Rejected intermediate crops are not included as dataset samples. This procedure continues until the task-relevant switch region is retained, so all 126 final close-up samples (100%) satisfy the CLIP-retention criterion by construction. The criterion concerns survival of the switch evidence required for the local reasoning task rather than preservation of a fixed fraction of surrounding image area. The total retained-area fraction is therefore not used as a dataset-quality statistic: it would measure discarded peripheral context rather than whether the task-defining switch evidence reaches the CLIP encoder.

For each close-up sample, the ego reference is inherited from the original full forward-looking scene: the train-occupied approach and associated target-route identity are established before region-of-interest extraction and retained when defining the cropped annotation, rather than inferred anew from the crop alone. Wider-context views retain one or multiple switches together with the surrounding forward-looking scene and preserve the camera-centric upstream reference, which is generally associated with the track bed nearest the bottom-center region of the image. The two view types therefore provide complementary local and scene-level supervision rather than representing separate task classes.

Table 5.2 Distribution of the 242 unique reasoning-training images by switch topology, active-route side, and view type, before stochastic counterfactual flipping.

Switch topology	**Active ego-route**	**Close-up**	**Wider context**	**Total**
Turnout	Right	42	33	75
	Left	33	37	70
Merge	Right	28	25	53
	Left	23	21	44
Total		**126**	**116**	**242**

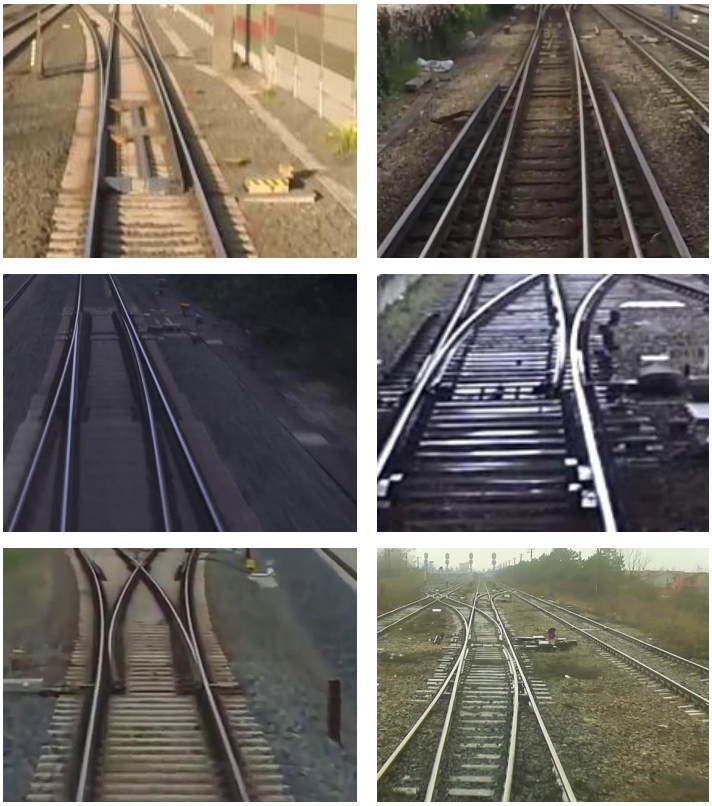


Figure 5.3 Sample images from the railway reasoning-segmentation dataset, illustrating switch configurations under different fields of view.

The reasoning and semantic streams are not disjoint at the source-image level. The resulting reasoning set provides structured coverage rather than equal class frequencies. Table 5.2 shows that all four combinations of switch topology and active-route side occur in both close-up and wider-context views. This composition exposes the model to both turnout and merge logic and to active routes on either side, while the two view types provide complementary evidence at different spatial scales. Straight or curved appearance is not treated as a separate class. Examples in any of the four groups may appear straight or curved according to the visible track geometry. All 242 reasoning images are retained for training so that the already limited coverage of switch topology, active-route side, and view type is not reduced by a further split. Consequently, no dedicated holdout is drawn from this subset. Generalization is instead assessed on the disjoint validation and test source ranges summarized in Table 5.1. Figure 5.3 shows representative reasoning images.

Each annotated image has six segmentation-oriented prompt paraphrases that preserve the same route logic while varying the wording. Explanatory supervision is also included: each image is paired with a query asking why the segmented route is correct and an answer describing cues such as blade contact, visible rail gaps, or rail continuity. These design choices are intended to reduce overfitting to one question form and to specialize the reasoning behavior to railway switch semantics.

5.5.2 Semantic Segmentation Stream

The semantic stream complements the switch-specific reasoning data with broader railway-scene supervision. Its labels are grouped into three scene elements: *rail*, *track bed*, and *background*. This additional supervision is needed because the reasoning dataset alone is too small and too switch-specific to teach robust railway context from scratch.

5.5.3 Image Preprocessing for the CLIP and SAM Branches

Each image follows two independent preprocessing paths because LISA uses the LLaVA vision tower for multimodal language reasoning and the SAM image encoder for mask generation. Both paths start from the same RGB sample before encoder-specific processing. Neither branch receives an image that has already been transformed for the other branch. The LLaVA branch uses the `CLIPImageProcessor` associated with `openai/clip-vit-large-patch14`. This processor resizes the image so that its shortest side is 224 pixels, applies a center crop of 224×224 pixels, and normalizes the result using the CLIP channel statistics. For a square input, the resize already produces a 224×224 image, whereas for a non-square input the center crop removes peripheral content along the longer resized dimension. The close-up acceptance procedure described during reasoning-dataset construction accounts for this geometry.

The SAM path applies an aspect-ratio-preserving longest-side transformation with a 1024-pixel limit and with upscaling disabled. An image is downscaled only when its longest side exceeds 1024 pixels. Otherwise, its dimensions are retained. The image is then normalized using the SAM channel statistics and zero-padded along its right and bottom edges to form

a 1024×1024 tensor. Nonzero padding is therefore introduced only for the 126 variable-resolution close-up reasoning samples. Semantic-training, wider-context reasoning, validation, and test images are already 1024×1024 and require no padding. Figure 5.4 reports the native-resolution distribution independently of these branch-specific transformations.

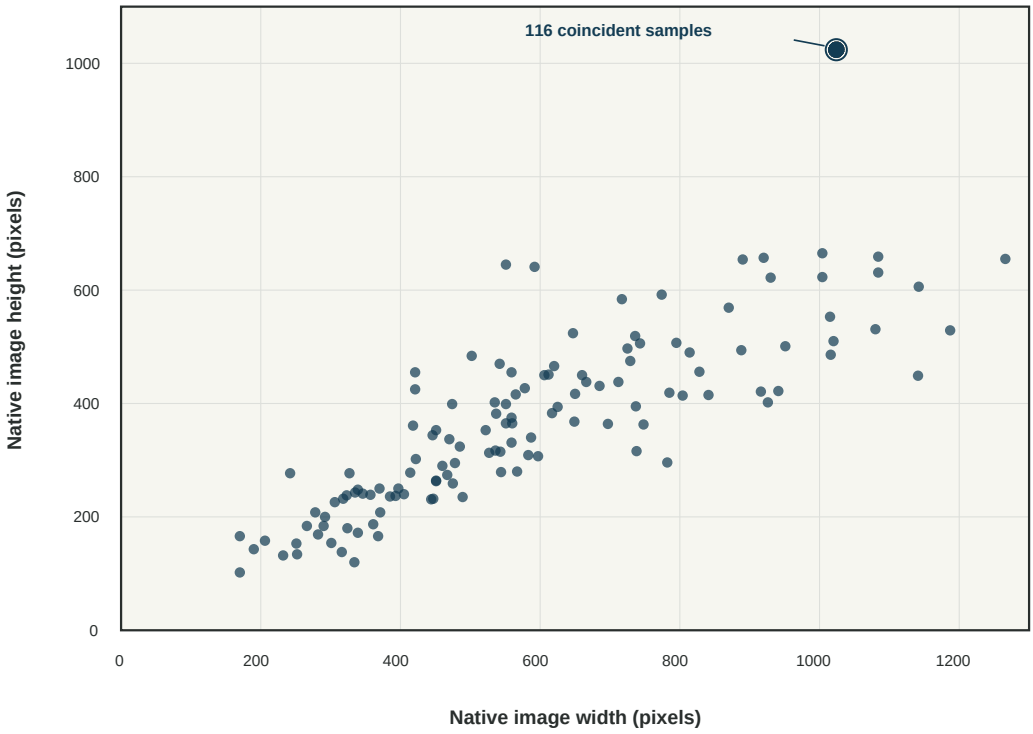


Figure 5.4 Native-resolution distribution of the 242 reasoning-training images before branch-specific preprocessing. Each point represents one image; points with identical dimensions coincide, including 116 samples at 1024×1024.

5.5.4 Finetuning Setup

The railway finetuning starts from the LISA-7B-v1 demo checkpoint released with LISA [65]. This checkpoint is pretrained on LISA's original mixture of semantic segmentation, referring segmentation, VQA, and generic reasoning segmentation. The "7B" designation refers to the 7-billion-parameter LLaMA-based LLaVA language backbone used by the smallest LISA variant. The checkpoint is loaded in its merged Hugging Face format. LoRA uses rank 8,

scaling parameter $\alpha = 16$, and dropout 0.05 for the query and value attention projections. The selectively trained multimodal components are listed in Section 5.4.2.

Finetuning is carried out in bfloat16 precision on an HPC cluster using 8 NVIDIA L40 GPUs distributed over two compute nodes. A per-GPU batch size of 2 is used with a gradient-accumulation factor of 1, corresponding to an effective batch size of 16 samples per optimizer update. Training is organized into 20 intervals of 50 iterations, with one optimizer update per iteration, for a total of 1,000 optimizer updates. The implementation labels each interval as an epoch because validation is performed at its end. In this chapter, this label denotes the validation cadence rather than a complete pass through either training dataset. At each data-loader call, a training stream is sampled according to the configured weights and an image is selected stochastically from that stream. Image selection is performed with replacement, so the same image may be selected more than once within or across training intervals.

For joint training, semantic and railway reasoning-segmentation samples are assigned sampling weights of 10 and 1, respectively. Normalization gives $P_s = 10/11 \approx 0.909$ for the semantic stream and $1/11 \approx 0.091$ for the reasoning stream. The explanatory probability is $P_e = 0.5$: among reasoning-stream draws, 50% provide text-only explanatory supervision. The other 50% produce a mask and are divided equally between mask-only and mask-plus-explanation responses, giving expected conditional proportions of 50%, 25%, and 25%, respectively.

Across 1,000 optimizer updates with an effective batch size of 16, the schedule produces 16,000 sample exposures. At the reasoning-stream probability of 1/11, approximately 1,455 exposures are expected from that stream, corresponding to about 6 exposures per unique reasoning image on average. These are aggregate expectations under the weighted stream sampling rather than a fixed number of complete dataset passes or deterministic repetitions: prompt paraphrasing, response-form sampling, and counterfactual horizontal flipping vary the supervision presented when an image is encountered again in a later iteration.

Optimization uses a base learning rate of 1×10^{-4} for the trainable parameter groups

except the unfrozen CLIP blocks, for which the learning rate is 1×10^{-5}. AdamW uses $\beta_1 = 0.9$, $\beta_2 = 0.95$, and zero weight decay. The schedule applies 100 linear warm-up steps followed by learning-rate decay over the remaining updates. Gradients are clipped to a maximum norm of 1.0, and distributed optimization uses DeepSpeed ZeRO Stage 2.

5.5.5 Validation Protocol

For each image in the validation range, comprising RailSem19 images 8,001–8,500 as summarized in Table 5.1, one of the following prompts is selected randomly:

1. "Based on the blade positions in this switch, which route corresponds to the route the train takes? Please respond with segmentation mask."

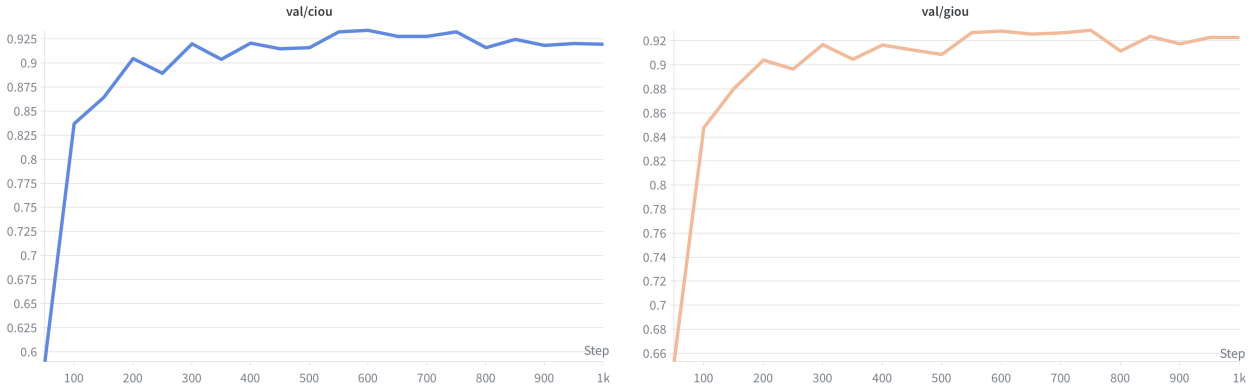
2. "Which path corresponds to the ego-path if the rails of the ego-path are continuously connected? Please respond with segmentation mask."

3. "By examining the switch geometry, which route corresponds to the route that remains continuous for the train? Please respond with segmentation mask."

This held-out set supports monitoring and checkpoint selection under varied switch-aware prompts. Validation is performed after each 50-iteration interval, giving 20 validation assessments over the predetermined budget of 1,000 optimizer updates. As defined above, the corresponding epoch labels mark validation intervals rather than complete traversals of the training data. Training continues for the complete predetermined budget. For each finetuning run, the checkpoint attaining the highest validation GIoU among the 20 assessments is selected for test-set evaluation. Because each validation prompt requests a segmentation mask and the available validation labels are ego-path masks, these measurements assess mask overlap. Manually verified rationale-reference annotations were not produced for the complete validation and test partitions because annotation effort was prioritized toward constructing the reasoning-training dataset. Rationale correctness is therefore evaluated

separately through the targeted audit of 30 switch-dependent test images for which switch-type and active-route-direction reference labels were manually established. As detailed in Table 5.5, this audit does not assess the complete rationale's visual grounding.

Figure 5.5 shows the validation trajectories for the model jointly finetuned with semantic and reasoning-segmentation data. Both overlap measures improve rapidly during the early training intervals and subsequently fluctuate within a comparatively narrow range. The GIoU trajectory determines checkpoint selection, while CIoU provides a complementary cumulative-overlap view of validation-mask quality.



(a) Validation CIoU.

(b) Validation GIoU.

Figure 5.5 Mask-overlap validation trajectories for joint semantic and reasoning-segmentation finetuning. Validation is performed every 50 optimizer updates on the 500-image held-out validation set; the horizontal axis reports cumulative optimizer-update steps.

5.5.6 Inference Workflow

At inference time, the input consists of a raw railway image and a natural-language prompt. Three representative prompt forms are considered: a direct request to segment the train's valid forward path; a question asking which route is active in a switch together with a mask request; and a request for both the ego-route mask and a rationale for why it is valid.

The generated text may include `[SEG]`, whose hidden state is decoded into the predicted ego-path mask. When a justification is requested, the language decoder also generates a rationale. The route-logic rationale is intended to identify cues such as which blade is pressed

against its stock rail, which competing branch is interrupted by an open gap, and which approach connects continuously into the shared mainline.

5.6 Experimental Results

5.6.1 Evaluation Setup

The final 2,500 RailSem19 images are partitioned by source order: images 6,001–8,000 form the 2,000-image test set, and images 8,001–8,500 form the 500-image validation set. Each 1024×1024 input is cropped directly from its raw 1920×1080-pixel image (width $\times$ height), without scaling during crop construction. The crop is horizontally centered and bottom-aligned: 448 pixels are removed from each lateral side and 56 pixels are removed from the top. The ego-path ground-truth mask is cropped using exactly the same coordinates, so prediction and ground truth are evaluated within the same retained image region. Ego-path annotations are obtained from the annotations provided with TEP-Net [13].

Cropping is used instead of resizing the complete raw frame so that native pixel spacing is preserved within the retained region before branch-specific preprocessing. This avoids compressing fine rail and switch geometry through an additional full-frame resize. Given the forward-looking camera configuration and the crop dimensions, the protocol assumes that the ego-path relevant to the train remains within the lower central field of view rather than near the discarded lateral boundaries. After crop construction, the resulting square image follows the separate CLIP and SAM transformations described in Section 5.5.3. In particular, the subsequent CLIP resize and center crop remain part of the model input pipeline.

The complete 2,000-image test set is inspected manually to identify scenes in which ego-path determination requires interpretation of a downstream switch. An image is designated *switch-dependent* when its annotated ego-path traverses a switch region and the corresponding switch configuration is visually discernible to a human annotator. All 178 images satisfying these conditions form the switch-dependent subset. The remaining 1,822 images do not contain such a qualifying visible switch along the annotated ego-path and are designated

switch-independent. These terms describe the route-selection condition in this evaluation and do not refer to the number of ground-truth masks.

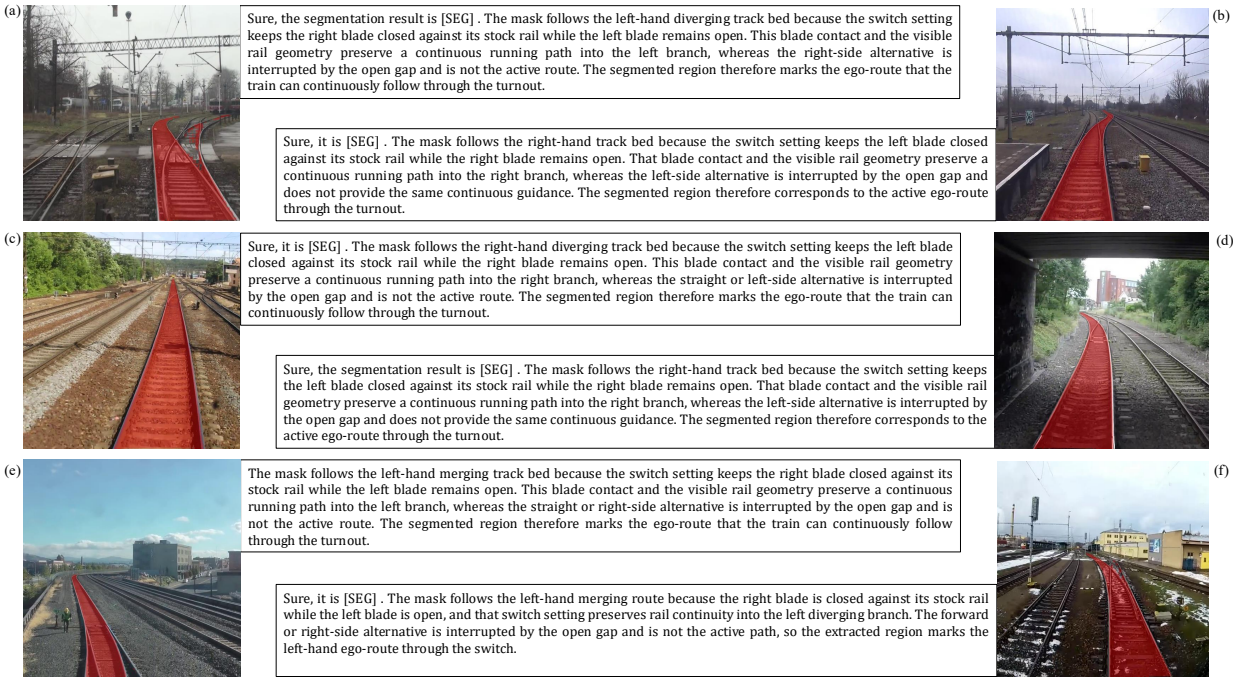


Figure 5.6 Qualitative results of the rail-finetuned LISA model on 1024×1024 horizontally centered, bottom-aligned RailSem19 test crops using the query "By examining rail continuity and switch geometry, segment the active ego-route and explain why." Each example presents the predicted ego-route mask together with the generated rationale, illustrating both coherent predictions and remaining mask or route-logic limitations.

For every test image and prompt, mask overlap is measured against the ground-truth ego-path. CIoU aggregates the pixel intersections and unions over all N images in a scene type, whereas GIoU averages the image-wise IoU values:

$$\mathrm{CIoU} = \frac{\sum_{i=1}^{N} |P_i \cap Y_i|}{\sum_{i=1}^{N} |P_i \cup Y_i|}, \qquad \mathrm{GIoU} = \frac{1}{N} \sum_{i=1}^{N} \frac{|P_i \cap Y_i|}{|P_i \cup Y_i|}, \tag{5.2}$$

where P_i and Y_i denote the predicted and ground-truth ego-path masks, respectively.

The qualitative analysis examines whether the predicted masks and generated rationales remain mutually consistent in switch scenes.

5.6.2 Qualitative Results

Figure 5.6 presents six predictions produced under the horizontally centered, bottom-aligned crop protocol. The masks are generally concentrated on a single candidate track-bed region and show coherent spatial localization, although residual noise and spatial diffusion remain in some examples. The rationales refer to rail continuity, blade–stock contact, and gaps between competing routes, but they are not always consistent with the predicted mask or visible switch geometry. In particular, rationales that identify merging routes also use turnout or diverging-branch terminology. These examples show that route selection and verbal rationale generation are related outputs, but are not yet reliably aligned in all cases.

5.6.3 Quantitative Results

The quantitative comparison uses two prompt formulations: the reasoning-oriented prompt "By examining rail continuity and switch geometry, segment the active ego-route the train is following in this image." and the generic instruction "Segment the track bed in this image." The former requests selection of the active ego-route, whereas the latter requests all track-bed regions regardless of the ego-path annotation. Because both outputs are evaluated against the ground-truth ego-path, the generic-prompt results measure how much the requested track-bed mask overlaps the ego-path rather than directly measuring all-track-bed segmentation accuracy. This prompt is included to determine whether the model changes its segmentation behavior according to the requested task.

Three checkpoints are evaluated. The first is the original LISA demo checkpoint. The other two are the best-validation checkpoints selected by validation GIoU from rail-finetuning runs that use the same initialization, selective adaptation strategy, optimization settings, and checkpoint-selection rule. They differ only in their training data: one uses the semantic segmentation stream alone, while the other jointly uses the semantic and reasoning-segmentation streams.

Table 5.3 Ego-path overlap under the reasoning-oriented ego-route prompt and the generic track-bed prompt, separated by whether route determination depends on a downstream switch. All values are percentages computed against the ground-truth ego-path mask.

Model	Reasoning-oriented ego-route prompt				Generic track-bed prompt			
	Switch-independent (n = 1,822)		Switch-dependent (n = 178)		Switch-independent (n = 1,822)		Switch-dependent (n = 178)	
	CIoU	**GIoU**	**CIoU**	**GIoU**	**CIoU**	**GIoU**	**CIoU**	**GIoU**
LISA [65]	30.07	32.39	36.59	39.73	8.25	7.70	4.87	4.57
Rail-finetuned LISA *semantic segmentation data only*	66.23	66.15	56.95	57.97	65.92	66.03	57.21	58.18
Rail-finetuned LISA *joint semantic and reasoning segmentation data*	89.00	88.34	90.49	90.33	65.58	65.94	57.87	58.89

Table 5.3 shows that the original LISA checkpoint responds to the prompt wording but remains weakly aligned with the railway task. Replacing the generic prompt with the reasoning-oriented prompt raises its CIoU and GIoU by 21.82 and 24.69 percentage points, respectively, in switch-independent scenes, and by 31.72 and 35.16 points in switch-dependent scenes. Nevertheless, its reasoning-oriented results remain between 30.07% and 39.73% across the two metrics and scene types. A counterintuitive pattern is that these results are 6.52 CIoU and 7.34 GIoU points higher in switch-dependent scenes than in switch-independent scenes. One possible explanation is that the explicit reference to switch geometry is more closely aligned with the visual content of switch-dependent scenes. Another is that the amount of shared track-bed area or other differences in subset composition make mask overlap easier to obtain in that stratum. The aggregate metrics do not distinguish between these explanations, and the pattern should not be interpreted as evidence that the original checkpoint understands railway switch configurations.

Finetuning with semantic data alone provides strong railway-domain adaptation but little task-specific prompt sensitivity. Under the reasoning-oriented prompt, this variant obtains a CIoU and GIoU of 66.23% and 66.15% in switch-independent scenes and 56.95% and 57.97% in switch-dependent scenes. Its corresponding generic-prompt results differ by

no more than 0.31 percentage points. The approximately eight- to nine-point reduction from switch-independent to switch-dependent scenes under either prompt further indicates that semantic supervision alone does not resolve the additional route-selection requirement introduced by a downstream switch.

Adding the reasoning-segmentation data changes this behavior. Relative to semantic-only finetuning under the reasoning-oriented prompt, joint finetuning improves CIoU and GIoU by 22.77 and 22.19 percentage points in switch-independent scenes and by 33.53 and 32.35 points in switch-dependent scenes. The jointly trained model consequently reaches 89.00% CIoU and 88.34% GIoU in switch-independent scenes and 90.49% CIoU and 90.33% GIoU in switch-dependent scenes. The two scene types therefore have similar aggregate mask-overlap scores, differing by 1.49 CIoU and 1.99 GIoU points. Because the 178 switch-dependent images belong to a disjoint image-index range and require switch-based route determination, the result demonstrates transfer beyond the exact 242 reasoning-training images. It does not establish route-, site-, or sequence-disjoint generalization or rule out reliance on recurring scene characteristics. Moreover, because alternative routes can share substantial track-bed area, similar aggregate scores do not rule out branch-selection errors or establish equally accurate route selection in switch-dependent scenes.

Under the generic prompt, joint and semantic-only finetuning remain closely matched: their differences are -0.34 CIoU and -0.09 GIoU points in switch-independent scenes, and $+0.66$ and $+0.71$ points in switch-dependent scenes. Within the jointly trained model, however, the reasoning-oriented prompt increases ego-path overlap by 23.42 CIoU and 22.40 GIoU points in switch-independent scenes, and by 32.62 and 31.44 points in switch-dependent scenes. This is the intended form of prompt sensitivity: the same model distinguishes a request for the active ego-route from a request for generic track-bed segmentation. The lower ego-path overlap under the generic prompt is consistent with producing track-bed regions beyond the single annotated ego-path, particularly in switch-dependent scenes. CIoU and GIoU exhibit the same overall patterns, so these conclusions are consistent across the two aggregation methods.

Table 5.4 reports the uncertainty in the central reasoning-oriented comparison between semantic-only and joint finetuning. Images are resampled as paired units so that both checkpoints are evaluated on the same bootstrap sample. The analysis uses 10,000 paired image-level percentile-bootstrap resamples, a 95% confidence level, and base seed 2026.

Table 5.4 Paired image-level uncertainty analysis for joint semantic and reasoning finetuning relative to semantic-only finetuning under the reasoning-oriented ego-route prompt. Differences and intervals are reported in percentage points; positive values favour joint finetuning. Intervals are based on 10,000 paired percentile-bootstrap resamples. Differences are computed from unrounded scores and may differ slightly from subtraction of the rounded scores in Table 5.3.

Scene type	Metric	Difference (percentage points)	Paired-bootstrap 95% CI
Switch-independent	CIoU	22.77	[22.00, 23.53]
($n = 1{,}822$)	GIoU	22.19	[21.50, 22.89]
Switch-dependent	CIoU	33.53	[31.10, 36.07]
($n = 178$)	GIoU	32.35	[30.10, 34.68]

All four confidence intervals in Table 5.4 remain above zero. For the fixed checkpoints and test set, the paired analysis therefore supports higher ego-path overlap from joint finetuning under the reasoning-oriented prompt in both scene types. The observed gains are larger in switch-dependent scenes: their lower confidence bounds exceed 30 percentage points for both metrics, compared with approximately 22 points for switch-independent scenes. This pattern is consistent with reasoning-segmentation supervision being especially useful when downstream switch geometry determines the target. However, the reported intervals quantify each within-stratum comparison and do not directly test the difference in gains between the two strata. They quantify image-level sampling uncertainty for these fixed checkpoints, not variation across independent training runs.

In this chapter, the reliability of the explanatory output is evaluated through a limited route-logic consistency audit rather than as a general measure of model interpretability. The generated rationale is compared with manually verified labels for switch type and active-route direction, and it is considered jointly correct only when both attributes match. This assessment is paired with the strict mask criterion to determine whether a correct route-logic description accompanies a correct segmentation result. The protocol measures factual

consistency for these two attributes, but it does not establish that the rationale faithfully represents the internal process that produced the mask or that the complete explanation is visually grounded.

A targeted manual audit is conducted on 30 switch-dependent images from the 2,000-image test set using the same validation-selected joint-finetuning checkpoint evaluated in Table 5.3. Ground-truth ego-path masks are available for the complete test set and support the full-set CIoU and GIoU evaluation. Assessing the generated rationales, however, additionally requires manually verified reference labels for switch type and active-route direction. These rationale-reference labels are available for 30 switch-dependent images. The mask assessment uses the same 30 images so that mask and rationale correctness can be paired image by image. The audit size is therefore determined by rationale-label availability rather than segmentation-mask availability. Within this constraint, the subset is balanced separately by switch type and active-route direction: it contains 15 turnout and 15 merge scenes, together with 15 left-active and 15 right-active routes. These are marginal balances for the two assessed attributes rather than equal counts for all four type–direction combinations. The results are therefore descriptive of this balanced subset rather than prevalence-weighted estimates for the complete test set.

The RailSem19 annotations used in the preceding camera-based chapters provide instance masks for all rail paths in these scenes. The intended-route and competing-route masks used in this audit are obtained from these rail-path-instance annotations. The mask assessment separates route selection from mask quality. Let P denote the predicted binary mask, Y_t the reference mask of the intended route, $\mathcal{C}$ the index set of competing route candidates, and Y_j the reference mask of competitor $j \in \mathcal{C}$. The union of the competing masks, Y_c, and the target- and competitor-exclusive regions, E_t and E_c, are defined as

$$\begin{aligned} &Y_c = \bigcup_{j \in \mathcal{C}} Y_j, \qquad E_t = Y_t \setminus Y_c, \qquad E_c = Y_c \setminus Y_t, \\ &d_{\text{excl}} = |P \cap E_t| + |P \cap E_c|, \qquad r_{\text{comp}} = \frac{|P \cap E_c|}{d_{\text{excl}}}, \qquad d_{\text{excl}} > 0. \end{aligned} \tag{5.3}$$

Here, d_{excl} is the number of predicted pixels in either type of route-exclusive region, r_{comp} is the proportion of those pixels assigned to competing routes, and $|\cdot|$ denotes pixel cardinality. Pixels in the shared portions of the candidate routes are excluded from both E_t and E_c, so isolated competing-route pixels have only a proportional effect rather than causing an automatic failure. If $d_{\text{excl}} = 0$, the prediction provides no evidence in any route-exclusive region and is counted as incorrect.

With $\text{IoU}(P, Y_k) = |P \cap Y_k|/|P \cup Y_k|$, the strict mask criterion is

$$\text{Correct}(P) \iff \begin{cases} \text{IoU}(P, Y_t) = \max\limits_{k \in \{t\} \cup \mathcal{C}} \text{IoU}(P, Y_k) \\ \text{IoU}(P, Y_t) > 0.90 \\ d_{\text{excl}} > 0 \\ r_{\text{comp}} < 0.10 \end{cases} \tag{5.4}$$

The 90% intended-route IoU and 10% competing-route coverage cutoffs are design thresholds for this audit. A mask is therefore reported as *Incorrect* in Table 5.5 whenever at least one condition in Equation 5.4 is not satisfied. Among these failures, *wrong route* means that a competing candidate has the highest IoU, whereas *overlapping routes* means that the intended candidate retains the highest IoU but $r_{\text{comp}} \geq 0.10$. Any remaining failure for which no competitor has a higher IoU is an *other mask-quality failure*. This includes insufficient intended-route IoU without substantial competing-route coverage and the absence of predicted route-exclusive pixels.

The generated rationale is assessed only on whether it identifies the correct switch type (turnout or merge) and active-route direction (left or right). For the rationale rows of Table 5.5, *Incorrect* means that the named attribute does not match its reference label. The joint row is correct only when both attributes match. This audit does not assess the fluency, detailed geometric correctness, or full visual grounding of the remaining text.

Table 5.5 Aggregate mask and route-logic rationale assessment on the balanced 30-image switch-dependent test subset.

(a) Output-level correctness

Assessment	**Correct**	**Incorrect**
Strict mask criterion	24 (80.0%)	6 (20.0%)
Switch type in rationale	26 (86.7%)	4 (13.3%)
Active-route direction in rationale	21 (70.0%)	9 (30.0%)
Switch type and direction jointly	18 (60.0%)	12 (40.0%)

(b) Paired mask–rationale outcomes

Mask assessment	**Type and direction correct**	**At least one incorrect**	**Total**
Correct	13	11	24
Incorrect	5	1	6
Total	**18**	**12**	**30**

Panel (a) of Table 5.5 shows that switch type is identified more consistently than active-route direction: 26 of 30 rationales contain the correct type, while 21 contain the correct direction and 18 contain both. Under the strict mask criterion, 24 masks are correct. Among the six failures, two overlap multiple candidate routes, two select the wrong route, and two fail the intended-route IoU threshold. Panel (b) shows that 13 samples have both a correct mask and a jointly correct rationale. Of the 24 correct masks, 11 (45.8%) are accompanied by an error in switch type, route direction, or both. Conversely, five of the six incorrect masks are accompanied by a jointly correct type-and-direction description. The two outputs are therefore not reliably coupled in this audit, and the generated text should be treated as an auxiliary rationale rather than a verified explanation of the mask-generation process. As switch-type and active-route-direction rationale labels are added to further switch-dependent test images, the same protocol can support a broader assessment.

5.7 Limitations and Future Directions

5.7.1 Limitations of the Current Study

Several limitations are important for the thesis narrative. First, the training objective relies on conventional semantic segmentation losses and therefore does not explicitly enforce structural continuity or topology of railway tracks. This limitation is consistent with the residual noise and spatial diffusion observed in some qualitative examples.

Second, standard IoU-based metrics can overstate correctness in switch scenes because substantial overlap between alternative track beds may allow an incorrect route prediction to exceed 50% IoU. Separating the aggregate results by switch dependence makes this difficulty visible but does not by itself determine whether an overlapping incorrect route was selected. The branch-exclusive criterion in the 30-image audit provides this distinction for the audited subset, but not for all 178 switch-dependent test images. In particular, the similar aggregate scores obtained by the jointly trained model across the two scene types can coexist with branch-selection errors if the shared track-bed region contributes most of the intersection and union. In addition, the generic-prompt outputs are evaluated against the ego-path mask rather than an all-track-bed ground truth. Those columns therefore diagnose task-dependent output behavior but do not directly measure the correctness of generic track-bed segmentation.

Third, the textual output remains an auxiliary rationale rather than a verified explanation. Neither the architecture nor the training objective explicitly requires the generated rationale to agree with the predicted mask or the relevant visual evidence. The paired audit contains both correct masks with incorrect route-logic descriptions and incorrect masks with correct descriptions, demonstrating this limitation directly. Moreover, the 30-image audit evaluates only switch type and active-route direction. It does not evaluate fluency, detailed factual correctness, blade–stock interaction, rail-continuity reasoning, train-traversal prediction, hallucination risk, or full grounding in the visual evidence and predicted mask. The rationales are therefore not validated for operator-facing interpretation or safety assurance.

Fourth, the controlled semantic-only and joint comparison isolates the contribution of adding the reasoning-segmentation stream as a complete training configuration, but it does not separate segmentation-oriented reasoning supervision from explanatory supervision, switch-geometry coverage, prompt paraphrasing, the 10:1 stream-sampling ratio, or counterfactual flipping. It also does not isolate selective visual unfreezing from a frozen-vision, LoRA-only alternative. It therefore remains uncertain which individual components produce the observed route-selection and prompt-conditioning behavior. Fifth, although the close-up acceptance criterion ensures that task-relevant switch information survives the CLIP center crop, the experiments do not isolate the effects of mixing close-up and wider reasoning views or variation in the SAM content-to-padding ratio on mask quality, reasoning quality, or training stability. Because nonzero SAM padding occurs only for close-up reasoning samples, this systematic input pattern could act as a partial stream or task cue during joint training. The present experiments do not determine whether the model exploits this cue instead of, or in addition to, the intended switch geometry and language supervision.

Two broader limitations concern the dataset and evaluation design. One is the absence of a direct comparison with existing state-of-the-art methods, because the proposed dataset pairs segmentation with reasoning supervision rather than reusing a standard benchmark protocol. The other is data scarcity: although the model is exposed to switch scenes and switch-related textual supervision, the training data do not provide explicit component-level localization labels, such as switch masks or bounding boxes. The model must therefore learn the spatial relationships among switches, stock rails, switch blades, rail frogs, diamond crossings, and upstream–downstream path structure without dedicated component targets, potentially limiting robustness. Scenes captured beyond the switch region remain inherently ambiguous once blade–stock interaction is no longer visible, which can lead to hallucinated ego-path masks and uninformative or inconsistent rationales.

5.7.2 Future Research Directions

These limitations motivate incorporating structural-continuity and topology constraints into the training objective, extending task-specific route-selection assessment beyond the 30-image audit, and creating dedicated switch-region labels for public benchmarks. Such assessment should complement whole-mask IoU with branch-exclusive coverage or explicit active-route correctness, so that overlap along the shared approach does not obscure an incorrect route choice. Evaluation protocols are also needed to extend the present type-and-direction audit to detailed alignment among the generated rationale, visual evidence, and predicted mask rather than evaluating linguistic plausibility alone.

Rationale faithfulness must therefore be treated as a separate research objective. As manually verified switch-type and active-route-direction rationale labels become available for additional switch-dependent test images, the balanced audit should be expanded to provide broader estimates of mask–rationale consistency. Future work should also assess the factual correctness of the complete generated text and investigate explicit consistency objectives or verification mechanisms that relate stated switch type, route direction, and geometric cues to both the predicted mask and the corresponding visual evidence. These developments are required before generated rationales can be considered for operator-facing use.

Further work includes ablations that isolate explanatory supervision, switch geometry, prompt paraphrasing, stream-sampling ratios, counterfactual flipping, and selective visual unfreezing; direct comparisons under standardized protocols; and richer supervision for railway topology. For scenes in which the switch mechanism is no longer visible, temporal information from consecutive frames is a necessary direction for reducing ambiguity and hallucination.

The method also forms part of a longer-term research direction. A reliable ego-path mask can define a region of interest aligned with the train's valid trajectory. This region can support future work on context-aware behavior prediction and collision-risk estimation, with visual perception and common-sense knowledge used to interpret dynamic scenes and anticipate hazards. These directions are not evaluated in the present chapter.

5.8 Chapter Conclusions

This chapter introduces reasoning-guided ego-path segmentation as a formulation for railway switch understanding in which a model predicts the valid route and, when prompted, generates an auxiliary textual rationale. The railway adaptation of LISA combines rail-specific prompting, semantic and reasoning supervision, selective multimodal finetuning, and language-conditioned mask generation. Relative to both the original LISA checkpoint and semantic-only railway finetuning, the controlled comparison shows that reasoning supervision substantially improves ego-path segmentation under the reasoning-oriented prompt, with the largest gains in switch-dependent scenes. Under the generic prompt, its measured ego-path overlap remains closely aligned with the semantic-only model, indicating that the added reasoning supervision primarily changes the response to the route-selection instruction. The balanced 30-image audit shows that switch type is identified more consistently than active-route direction, while the paired outcomes demonstrate that rationale correctness is not reliably coupled to mask correctness. The generated text is therefore an exploratory output rather than a validated explanation of the model's route-selection process.

This work is an initial step rather than a complete solution for safety-critical railway perception. Broader evaluation, stronger reasoning verification, more comprehensive ablations, and richer supervision over railway topology are required before the reliability of the formulation can be established.

The present study does not measure end-to-end inference latency, throughput, memory use, or real-time operation, and therefore makes no real-time performance claim. A potential deployment path is to use the rail-finetuned LISA model as a high-capacity teacher and distill its route-aware mask predictions into a smaller student model designed for on-board inference. Reduced-precision inference, including quantization, and deployment-specific runtime optimization could further reduce latency, but their effect on route-selection and segmentation accuracy must be validated. A tiered system could use a lightweight visual model for routine track segments and invoke the vision-language model for ambiguous or

switch-dependent scenes. Hardware-specific benchmarking and safety evaluation are required before any real-time capability can be claimed.

Within the thesis, this chapter advances the camera-based storyline by exploring whether route validity can be addressed through reasoning-capable vision-language models rather than through better candidate-path extraction alone. At the same time, it does not replace the earlier visual extraction chapters. Instead, it exposes new issues related to rationale quality, grounding, topology, route ambiguity, and evaluation. These findings complete the technical progression developed across Chapters 3–5 and prepare their thesis-level synthesis in Chapter 6.

Chapter 6

Discussions

The preceding technical chapters develop three stages of camera-based railway perception: triplet-based path extraction, learned instance tracing, and reasoning-guided perception with vision–language models. This chapter synthesizes those contributions at thesis level. Rather than repeating the chapter summaries, it identifies the common problem structure, the intellectual progression across the thesis, and the limitations that remain after the individual studies are considered together.

6.1 Overview of Thesis-Level Findings

The central thesis-level finding is that railway scene understanding in complex multi-track environments requires more structure than generic rail segmentation. Across the three technical chapters, the relevant output progresses from paired left–right rail paths and candidate ego-path instances to a language-conditioned valid-route mask. The shared problem is therefore not only detection of railway components, but interpretation of how rail geometry constrains possible train movement.

The first camera-based stage, Chapter 3, establishes this structure by treating railway perception as extraction of all possible ego-paths from forward-looking RGB imagery. Its contribution is not merely that rails are detected, but that left and right rail boundaries are associated into route-level path proposals. This provides the first complete thesis formulation of the problem: in switch scenes, a perception method must preserve route topology and rail

pairing rather than returning an unstructured rail mask.

Chapter 4 keeps the same candidate-path boundary but changes the mechanism by which path instances are assembled. Instead of relying on strip-wise spatial clustering and path-tree construction after dense prediction, it formulates path construction as learned triplet tracing through the Attraction Field Map. This stage shows that the thesis problem is not only one of representation, but also one of how much route continuity should be learned directly rather than imposed through hand-crafted post-processing.

Chapter 5 then changes the question from candidate-path extraction to route-validity interpretation. Earlier camera-based methods deliberately output all plausible ego-paths rather than a unique valid route. The vision–language chapter addresses this unresolved ambiguity by conditioning segmentation on a natural-language query and by associating the predicted mask with an optional auxiliary rationale intended to describe switch geometry. This does not eliminate all ambiguity, but it marks the thesis transition from structured extraction toward language-conditioned route interpretation.

6.2 From Camera-Based Perception to Reasoning Perception

The camera-based sequence progresses through three conceptual stages. The first stage establishes route-aware visual perception. In Chapter 3, the problem is framed around extracting all possible ego-path candidates visible from an onboard camera. This stage is important because it rejects a purely semantic interpretation of rails as one class of pixels. Instead, railway perception is formulated around path instances and left–right association.

The second stage strengthens the instance-construction mechanism. Chapter 4 retains the triplet representation but replaces heuristic route assembly with learned tracing. The Attraction Field Map provides row-wise direction and intensity cues that guide the construction of ego-path instances through turnout regions. The intellectual shift is therefore from detecting rail-path ingredients and assembling them through explicit rules toward learning the local

continuation behavior that makes a path coherent.

The third stage introduces reasoning as part of the perception task. Chapter 5 recognizes that even a well-traced set of candidate routes does not answer which route is physically valid through a switch. The route-validity problem depends on cues such as blade–stock contact, rail gaps, and visible route continuity. By adapting a vision–language segmentation model, the thesis moves from producing candidate path geometry toward asking for an interpreted route mask and, when prompted, an auxiliary textual rationale.

This progression should not be read as a simple replacement of one chapter by the next. The earlier camera-based chapters provide structured visual extraction mechanisms that remain relevant when the goal is to recover possible paths. The reasoning-guided chapter addresses a different layer of the problem: selecting the valid route when candidate paths coexist and assessing the accompanying rationale separately. Thus, the thesis camera sequence develops from extraction, to learned instance tracing, to route-validity interpretation.

6.3 Role of Vision–Language Reasoning in Addressing Earlier Limitations

The vision–language chapter responds most directly to a limitation shared by Chapters 3 and 4: those chapters intentionally extract possible ego-paths rather than infer a unique valid route through switch-state reasoning. In railway switch scenes, this distinction is central. Multiple routes may be visible, but only one may be physically valid for the train depending on the switch configuration. A method that outputs all plausible paths can support downstream interpretation, but it does not itself decide the route.

The reasoning-guided formulation partially addresses this gap by making route validity explicit in the query and conditioning segmentation on language that refers to visible switch evidence. This changes the role of language from a user interface detail into a task-conditioning mechanism. Under the reasoning-oriented prompt, the joint semantic and reasoning configuration produces higher ego-path overlap than semantic-only finetuning in

both scene strata, with larger observed gains in switch-dependent scenes and paired-bootstrap intervals above zero. Under the generic track-bed prompt, the two finetuned configurations remain closely matched, which is consistent with task-conditioned output behavior.

At the same time, Chapter 5 exposes limitations that were less visible in the earlier extraction chapters. In the balanced 30-image audit, 24 masks satisfy the strict criterion and 18 rationales identify both switch type and active-route direction correctly, but only 13 samples satisfy both assessments. Within this audited subset, rationale correctness is therefore not reliably coupled to mask correctness. Standard IoU-based metrics can also overstate correctness when overlapping track beds make wrong-route masks appear spatially plausible, and the segmentation losses do not explicitly enforce rail topology or structural continuity. Language-conditioned segmentation therefore does not automatically solve rationale grounding, topology, or route-specific evaluation.

For the thesis as a whole, this is an important qualification. Vision–language reasoning is not presented as a complete replacement for geometric rail modeling. It is better understood as a complementary layer that targets route-validity ambiguity left open by candidate extraction. Its scientific role is to demonstrate that valid-route perception can be formulated as a reasoning-guided segmentation problem, while its limitations identify the need for stronger rationale grounding, topology-aware objectives, and evaluation beyond the present type-and-direction audit.

6.4 Cross-Chapter Comparison of Assumptions, Methods, Outputs, and Limitations

Table 6.1 compares the technical chapters at the level of thesis assumptions rather than at the level of detailed experimental results. The comparison highlights the main continuity across the thesis: all three chapters address structured railway perception in settings where multiple rails or switches make simple semantic labeling insufficient.

Several cross-chapter patterns follow from this comparison. First, the output representation

Table 6.1 Thesis-level comparison of the technical chapters.

Chapter	Input	Methodological emphasis	Primary output	Main limitations retained at thesis level
Chapter 3	Forward-looking RGB image	Triplet extraction, segmentation-assisted regression, bottom-up path construction	All possible ego-path proposals with left–right rail association	No unique route selection; sensitivity to camera occlusion; limited extrapolation from short-range camera settings; benchmarking difficulty across task definitions
Chapter 4	Forward-looking RGB image	Learned triplet tracing through the Attraction Field Map, with self-supervised encoder pretraining	Candidate ego-path instances traced through rail scenes	Still outputs possible paths rather than switch-state route decisions; severe occlusion; long-range ambiguity; atypical switch geometries
Chapter 5	RGB image plus natural-language query	Railway-adapted reasoning-guided segmentation with vision–language prompting	Valid ego-path mask and optional auxiliary rationale	No explicit topology enforcement; 30-image rationale audit limited to switch type and route direction; limited reasoning data; hallucination risk in ambiguous scenes; no direct comparison with state-of-the-art methods

becomes increasingly task-specific across the camera sequence: from candidate paths, to traced path instances, to a valid-route mask. Second, the unresolved issue shifts from whether rail paths can be extracted to whether route validity, topology, and uncertainty can be evaluated reliably.

6.5 Practical Implications for Railway Scene Understanding

The practical implication of the thesis is that railway scene understanding should be treated as a structured interpretation problem rather than as isolated component detection. A

perception system for railway environments must preserve which rails form a track, which track instances remain plausible through switches, and which route is valid for the train under visible switch geometry. The thesis chapters address these requirements at different levels of abstraction.

Within an autonomous or decision-support setting, these outputs can supply the route context required for downstream object-risk assessment and operational decisions.

For camera-based railway perception, the early chapters imply that route-aware representations are necessary in multi-track scenes. A rail-pixel mask alone is insufficient when left and right rail boundaries must be paired into path instances. The TRIT-Net chapter further suggests that learned tracing can reduce dependence on hand-crafted route assembly, although the remaining limitations show that learned tracing still needs stronger robustness under occlusion, long-range ambiguity, and unusual switch geometry.

For reasoning-oriented perception, the VLM chapter shows that valid-route segmentation can be framed as a language-conditioned task. The formulation can additionally generate an auxiliary rationale intended to describe why one branch is valid and another is not. However, these rationales should not be treated as verified safety evidence: the audit assesses only switch type and active-route direction, and mask correctness does not guarantee rationale correctness.

6.6 Overall Scientific Contribution of the Thesis

The overall scientific contribution of the thesis is an integrated account of railway perception that moves from structured visual path extraction toward reasoning-guided route interpretation. The thesis contributes a sequence of methods and analyses that collectively address multi-track and switch-rich railway scenes, where the central challenge is not simply detecting rails but preserving route structure.

At the camera-based level, the thesis first contributes a triplet-based formulation for extracting possible ego-paths with left–right rail association. It then contributes a learned

tracing formulation that reduces reliance on heuristic post-processing and evaluates path extraction across a broader range of data settings. These two stages establish the methodological basis for structured ego-path extraction.

At the reasoning level, the thesis contributes an initial formulation of valid ego-path segmentation as a vision–language problem. This shifts attention from candidate enumeration to route validity and auxiliary rationale generation. The contribution is not that language solves all railway reasoning problems, but that route interpretation can be explicitly queried and conditioned on language referring to visible switch geometry.

6.7 Remaining Limitations and Open Problems

Several limitations remain visible across the thesis. The first is route decision under uncertainty. The TPE-Net and TRIT-Net chapters intentionally output possible ego-paths, while the VLM chapter targets valid-route segmentation but still faces rationale-grounding, topology, and route-specific evaluation limitations. A complete railway perception system would need a more explicit treatment of uncertainty and switch-state evidence than is provided by any single chapter.

The second limitation is robustness under difficult visibility. Occlusion appears repeatedly in the camera-based chapters, while long-range ambiguity near the vanishing point remains difficult for path extraction and route reasoning. Across the camera-based sequence, the common open problem is how to maintain reliable track and switch interpretation when the relevant structure is incomplete, distant, occluded, or ambiguous.

Camera calibration and train position and orientation would be useful in deployment because they could provide geometric constraints for TPE-Net path association and TRIT-Net path tracing, particularly when camera-mount variation, occlusion, distance, or rail geometry make image evidence ambiguous. These inputs were not incorporated or evaluated in the thesis, so the magnitude of any performance improvement cannot be inferred from the reported experiments.

The third limitation is topology. The earlier camera-based chapters encode path structure, but they do not determine a unique commanded route. The VLM chapter introduces route-validity reasoning, but its loss functions and evaluation metrics do not explicitly guarantee structural continuity or topology correctness. Future work therefore needs stronger representations of railway topology, switch state, and route continuity in camera-based railway perception.

The fourth limitation concerns datasets and evaluation protocols. Several chapters note difficulty comparing with prior work because of private datasets, different task definitions, or missing annotation types. The VLM chapter is further constrained by limited reasoning data. Its 30-image rationale audit quantifies switch-type and route-direction correctness but does not evaluate the complete rationale's factual correctness or visual grounding. These constraints limit how broadly the reported findings can be generalized.

Taken together, these limitations clarify the thesis contribution. The thesis advances railway perception by making route structure increasingly explicit: first through path proposal generation, then through learned instance tracing, and then through reasoning-guided valid-route segmentation. The remaining challenge is to combine structural extraction, verified rationale grounding, and uncertainty estimation in railway perception systems that can be evaluated under realistic operational conditions.

Chapter 7

Conclusion and Future Work

This chapter concludes the thesis by restating the overall research problem, summarizing the main contributions, and identifying future research directions that follow from the limitations discussed in the technical chapters and in Chapter 6. The chapter is intentionally concise. Its purpose is not to repeat the discussion, but to close the thesis-level argument.

7.1 Restatement of the Thesis Problem

The thesis addressed railway scene understanding in complex multi-track environments, with particular attention to rail paths, switches, and the ambiguity that arises when several routes are visible at once. Across the thesis, the central problem was not only to detect railway components, but to preserve the structure that makes those components meaningful for train movement: which rails form a track, which tracks form possible ego-paths, and which route is valid when switch geometry creates alternatives.

This problem was studied through forward-looking railway imagery. The technical chapters examined how visual perception can move from rail-path proposal generation to learned instance tracing and then to reasoning-guided valid-route segmentation.

7.2 Final Answers to the Thesis Research Questions

The first research question asked how forward-looking camera imagery can support possible ego-path extraction while preserving left–right rail association and route-level structure. Chapter 3 addressed this question through a triplet-based representation of left rail, center track point, and right rail information, followed by association into possible ego-path proposals. This formulation represents rail pairing and path structure rather than returning only an undifferentiated rail mask.

The second research question asked how ego-path extraction can reduce dependence on heuristic route assembly and learn path-instance continuity. Chapter 4 addressed this question through Attraction Field Map based triplet tracing, replacing the earlier strip-wise clustering and path-tree construction with learned local continuation cues. The chapter also broadened the evaluation across datasets and training settings, while retaining the task boundary of candidate-path extraction rather than unique route selection.

The third research question asked how valid-route interpretation in switch scenes can be formulated as a reasoning-guided vision–language perception problem. Chapter 5 addressed this question by conditioning valid ego-path segmentation on natural-language queries and associating the predicted mask with an optional auxiliary rationale intended to describe visible switch geometry. Under the reasoning-oriented prompt, the joint semantic and reasoning configuration produces higher ego-path overlap than semantic-only finetuning in both scene strata, with larger observed gains in switch-dependent scenes. The two configurations remain closely matched under the generic track-bed prompt. In the balanced 30-image audit, 24 masks satisfy the strict criterion and 18 rationales identify both assessed route attributes correctly, but only 13 samples satisfy both assessments. The formulation is therefore established, while rationale grounding, topology, and broader route-specific evaluation remain unresolved.

7.3 Summary of Main Contributions

The first main contribution of the thesis is a camera-based formulation of possible ego-path extraction from forward-looking RGB imagery. Chapter 3 established triplet-based rail path representation, associated left and right rail boundaries into path proposals, and framed multi-track railway perception as a route-structured problem rather than a pixel-only segmentation task.

The second main contribution is a learned tracing formulation for railway ego-path instances. Chapter 4 replaced the earlier dependence on strip-wise clustering and path-tree construction with Attraction Field Map based triplet tracing. This chapter advanced the camera-based sequence by learning local path continuation behavior and by broadening the empirical evaluation across datasets and training settings.

The third main contribution is the introduction of reasoning-guided valid-route segmentation for railway scenes. Chapter 5 adapted a vision–language segmentation model so that route interpretation could be conditioned by natural-language prompts and, when requested, accompanied by an auxiliary rationale about switch geometry. This moved the thesis from extracting candidate paths toward reasoning about route validity.

7.4 Final Reflection on Camera-Based Rail Perception

The completed thesis develops a progression from camera-based perception to reasoning-based perception. The early camera chapters focused on recovering possible path structure from images. The TRIT-Net chapter strengthened that structure by learning how path instances continue through complex scenes. The VLM chapter then addressed a different level of interpretation: when several candidate routes are visible, the perception problem becomes one of valid-route reasoning rather than candidate extraction alone.

7.5 Limitations of the Thesis

Several limitations remain across the thesis. The early camera-based chapters intentionally extract possible ego-paths rather than selecting a unique valid route through switch-state reasoning. The reasoning-guided chapter addresses valid-route segmentation, but it does not provide explicit topology enforcement. Its 30-image audit quantifies switch-type and active-route-direction correctness but shows that mask and rationale correctness are not reliably coupled and does not assess the complete rationale's factual correctness or visual grounding.

Robustness also remains incomplete. The camera-based chapters identify sensitivity to occlusion, long-range ambiguity, and unusual switch geometries. These issues show that reliable railway perception requires stronger treatment of incomplete or ambiguous visual evidence.

The thesis is also limited by dataset and evaluation constraints. Some chapters note difficulty comparing with prior work because of private datasets, different task definitions, or missing annotation types. The VLM chapter is constrained by limited railway reasoning data, a branch-exclusive mask assessment restricted to 30 of the 178 switch-dependent test images, and no direct comparison with existing methods under a common protocol.

7.6 Future Work

Future work should first strengthen reasoning-capable vision–language models for railway scenes. This includes grounding auxiliary rationales in visible switch geometry, extending branch-exclusive and rationale assessment beyond the 30-image subset, and testing whether the predicted route and rationale are mutually consistent. Such work should also consider topology-aware objectives or evaluation measures, because ordinary overlap metrics may not fully capture wrong-route errors in overlapping track scenes.

A second direction is uncertainty and confidence estimation. The thesis repeatedly

encounters ambiguity from occlusion, distance, and switch structure. Future systems should therefore estimate confidence in rail-path extraction, route validity, switch interpretation, and generated rationales, rather than returning only a mask or path set.

A third direction is the development of larger and more diverse railway datasets. The completed chapters point to the need for broader coverage of switch types, long-range scenes, occlusion conditions, weather and illumination variation, and detailed annotations for topology, switch state, left–right rail association, valid routes, and rationale grounding. Such datasets would also make fairer comparison across methods possible.

Finally, future research should move toward real-world deployment and safety validation. The thesis contributes methods for structured path perception and reasoning-guided segmentation, but it does not establish operational safety. Closing that gap would require systematic validation under realistic railway operating conditions, careful failure analysis, and explicit assessment of how perception uncertainty affects downstream decision-making.

References

[1] Yongbo Wu, Ziyang Feng, Xinyu Zhu, Ping Tan, Huoping Yi, Jin Ding, Zhen Li, and Jien Ma. "RT-RIDMOS: Real-Time Railway Intrusion Detection Method Based on Moving Object Segmentation". In: *IEEE Transactions on Intelligent Transportation Systems* 27.9 (2026), pp. 11461–11473. DOI: 10.1109/TITS.2026.3694886.

[2] Xin-Yue Xu, Su-Mei Wang, Wen-Qiang Liu, and Yi-Qing Ni. "Advancements in Obstacle Intrusion Detection Methods for Rail Transit: A Comprehensive Review". In: *IEEE Transactions on Instrumentation and Measurement* 74 (2025), pp. 1–34. DOI: 10.1109/TIM.2025.3612624.

[3] Haoran Li, Qichao Zhang, Dongbin Zhao, and Yaran Chen. "RailNet: An Information Aggregation Network for Rail Track Segmentation". In: *2020 International Joint Conference on Neural Networks (IJCNN)*. 2020, pp. 1–7. DOI: 10.1109/IJCNN48605.2020.9206967.

[4] Yin Wang, Lide Wang, Yu Hen Hu, and Ji Qiu. "RailNet: A Segmentation Network for Railroad Detection". In: *IEEE Access* 7 (2019), pp. 143772–143779. DOI: 10.1109/ACCESS.2019.2945633.

[5] Deqiang He, Yefeng Qiu, Jian Miao, Zhiheng Zou, Kai Li, Chonghui Ren, and Guoqiang Shen. "Improved Mask R-CNN for Obstacle Detection of Rail Transit". In: *Measurement* 190, 110728 (2022). DOI: 10.1016/j.measurement.2022.110728.

[6] Deqiang He, Kai Li, Yanjun Chen, Jian Miao, Xianwang Li, Sheng Shan, and Ruochen Ren. "Obstacle Detection in Dangerous Railway Track Areas by a Convolutional

Neural Network". In: *Measurement Science and Technology* 32.10, 105401 (2021). DOI: 10.1088/1361-6501/abfdde.

[7] Zhangyu Wang, Xinkai Wu, Guizhen Yu, and Mingxing Li. "Efficient Rail Area Detection using Convolutional Neural Network". In: *IEEE Access* 6 (2018), pp. 77656–77664. DOI: 10.1109/ACCESS.2018.2883704.

[8] Songyue Yang, Guizhen Yu, Zhangyu Wang, Bin Zhou, Peng Chen, and Qiang Zhang. "A Topology Guided Method for Rail-Track Detection". In: *IEEE Transactions on Vehicular Technology* 71.2 (2022), pp. 1426–1438. DOI: 10.1109/TVT.2021.3133327.

[9] Amartuvshin Dagvasumberel, Bilguunmaa Myagmardulam, Byambasuren Myagmar, Byambasuren Luvsankhuu, and Tadachika Nakayama. "Railroad Near-Miss Occurrence Detection and Risk Estimation System with Data from Camera Using Deep Learning". In: *2021 5th International Conference on Imaging, Signal Processing and Communications (ICISPC)*. 2021, pp. 83–87. DOI: 10.1109/ICISPC53419.2021.00023.

[10] Dan Barnes, Will Maddern, and Ingmar Posner. "Find Your Own Way: Weakly-Supervised Segmentation of Path Proposals for Urban Autonomy". In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. 2017, pp. 203–210. DOI: 10.1109/ICRA.2017.7989025.

[11] Ulrich Baumann, Claudius Gläser, Michael Herman, and J. Marius Zöllner. "Predicting Ego-Vehicle Paths from Environmental Observations with a Deep Neural Network". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. 2018, pp. 4709–4716. DOI: 10.1109/ICRA.2018.8460704.

[12] Alex Zyner, Stewart Worrall, and Eduardo M. Nebot. "Naturalistic Driver Intention and Path Prediction Using Recurrent Neural Networks". In: *IEEE Transactions on Intelligent Transportation Systems* 21.4 (2020), pp. 1584–1594. DOI: 10.1109/TITS.2019.2913166.

[13] Thomas Laurent. *Train Ego-Path Detection on Railway Tracks Using End-to-End Deep Learning*. 2024. DOI: 10.48550/arXiv.2403.13094. arXiv: 2403.13094 [cs.CV]. URL: https://arxiv.org/abs/2403.13094.

[14] Mohammadjavad Ghorbanalivakili, Jungwon Kang, Gunho Sohn, David Beach, and Veronica Marin. "TPE-Net: Track Point Extraction and Association Network for Rail Path Proposal Generation". In: *2023 IEEE 19th International Conference on Automation Science and Engineering (CASE)*. 2023, pp. 1–7. DOI: 10.1109/CASE56687.2023.10260541.

[15] John C. Hayward. "Near-Miss Determination Through Use of a Scale of Danger". In: *Highway Research Record* 384 (1972), pp. 24–34. URL: https://onlinepubs.trb.org/Onlinepubs/hrr/1972/384/384-004.pdf.

[16] Xinwei Wang, Zirui Li, Javier Alonso-Mora, and Meng Wang. "Prediction-Based Reachability Analysis for Collision Risk Assessment on Highways". In: *2022 IEEE Intelligent Vehicles Symposium (IV)*. 2022, pp. 504–510. DOI: 10.1109/IV51971.2022.9827304.

[17] Jinghao Cao, Yang Li, and Sidan Du. "Robust Artificial Intelligence-Aided Multimodal Rail-Obstacle Detection Method by Rail Track Topology Reconstruction". In: *Applied Sciences* 14.7, 2795 (2024). DOI: 10.3390/app14072795.

[18] Philipp Jaß, Carsten Thomas, Tina Hiebert, and Gustav Plettig. "Towards Safe Obstacle Detection for Autonomous Train Operation: Combining Track and Switch Detection Neural Networks for Robust Railway Ego Track Detection". In: *Proceedings of ERTS 2024*. 2024. URL: https://hal.science/hal-04634672.

[19] Lorenzo Barruffo, Bianca Caiazzo, Alberto Petrillo, and Stefania Santini. "A GoA4 Control Architecture for the Autonomous Driving of High-Speed Trains over ETCS: Design and Experimental Validation". In: *IEEE Transactions on Intelligent Transportation Systems* 25.6 (2024), pp. 5096–5111. DOI: 10.1109/TITS.2023.3338295.

[20] Mario Gleirscher, Anne E. Haxthausen, and Jan Peleska. "Probabilistic Risk Assessment of an Obstacle Detection System for GoA4 Freight Trains". In: *Proceedings of the 9th ACM SIGPLAN International Workshop on Formal Techniques for Safety-Critical Systems*. 2023, pp. 26–36. DOI: 10.1145/3623503.3623533.

[21] Jiateng Yin, Tao Tang, Lixing Yang, Jing Xun, Yeran Huang, and Ziyou Gao. "Research and Development of Automatic Train Operation for Railway Transportation Systems: A Survey". In: *Transportation Research Part C: Emerging Technologies* 85 (2017), pp. 548–572. DOI: 10.1016/j.trc.2017.09.009.

[22] Patrick Urassa. "Exploring Train Driving Automation and Signaling Interaction". In: *Transportation Engineering* 21, 100361 (2025). DOI: 10.1016/j.treng.2025.100361.

[23] Bryan Olivier, Feng Guo, Yu Qian, and David Connolly. "A Review of Computer Vision for Railways". In: *IEEE Transactions on Intelligent Transportation Systems* 26.7 (2025), pp. 11034–11065. DOI: 10.1109/TITS.2025.3552011.

[24] Zhiwei Cao, Yong Qin, Limin Jia, Zhengyu Xie, Yang Gao, Yaguan Wang, Ping Li, and Zujun Yu. "Railway Intrusion Detection Based on Machine Vision: A Survey, Challenges, and Perspectives". In: *IEEE Transactions on Intelligent Transportation Systems* 25.7 (2024), pp. 6427–6448. DOI: 10.1109/TITS.2024.3412170.

[25] Annika Bätz, Pavel Klasek, Seo-Young Ham, Philipp Neumaier, Martin Köppel, and Martin Lauer. *Railway Artificial Intelligence Learning Benchmark (RAIL-BENCH): A Benchmark Suite for Perception in the Railway Domain*. 2026. DOI: 10.48550/arXiv.2604.22507. arXiv: 2604.22507 [cs.CV]. URL: https://arxiv.org/abs/2604.22507.

[26] Danijela Ristić-Durrant, Marten Franke, and Kai Michels. "A Review of Vision-Based On-Board Obstacle Detection and Distance Estimation in Railways". In: *Sensors* 21.10, 3452 (2021). DOI: 10.3390/s21103452.

[27] Zan Chen, Weilong Niu, Cheng Wu, Lijun Zhang, and Yiming Wang. "Near Real-time Situation Awareness and Anomaly Detection for Complex Railway Environment". In: *2021 IEEE Conference on Cognitive and Computational Aspects of Situation Management (CogSIMA)*. 2021, pp. 1–8. DOI: 10.1109/CogSIMA51574.2021.9475947.

[28] Tao Ye, Xi Zhang, Yi Zhang, and Jie Liu. "Railway Traffic Object Detection Using Differential Feature Fusion Convolution Neural Network". In: *IEEE Transactions on Intelligent Transportation Systems* 22.3 (2021), pp. 1375–1387. DOI: 10.1109/TITS.2020.2969993.

[29] Zhen Tao, Shiwei Ren, Yueting Shi, Xiaohua Wang, and Weijiang Wang. "Accurate and Lightweight RailNet for Real-Time Rail Line Detection". In: *Electronics* 10.16, 2038 (2021). DOI: 10.3390/electronics10162038.

[30] Zongyang Zhao, Jiehu Kang, Zefeng Sun, Tao Ye, and Bin Wu. "A Real-Time and High-Accuracy Railway Obstacle Detection Method Using Lightweight CNN and Improved Transformer". In: *Measurement* 238, 115380 (2024). DOI: 10.1016/j.measurement.2024.115380.

[31] Tingyi Hu, Feng Gao, and Fuqiang Zhou. "Railway Obstacle Intrusion Detection and Risk Assessment Based on MSIA-YOLOv8 and DALNet". In: *Expert Systems with Applications* 299, 130132 (2026). DOI: 10.1016/j.eswa.2025.130132.

[32] Zhichao Chen, Jie Yang, Lifang Chen, Zhicheng Feng, and Limin Jia. "Efficient Railway Track Region Segmentation Algorithm Based on Lightweight Neural Network and Cross-Fusion Decoder". In: *Automation in Construction* 155, 105069 (2023). DOI: 10.1016/j.autcon.2023.105069.

[33] Zhichao Chen, Jie Yang, and Fazhu Zhou. "RailSegVITNet: A Lightweight ViT-Based Real-Time Track Surface Segmentation Network for Improving Railroad Safety". In: *Journal of King Saud University - Computer and Information Sciences* 36.1, 101929 (2024). DOI: 10.1016/j.jksuci.2024.101929.

[34] Chenglin Chen, Fei Wang, Min Yang, Yong Qin, and Yun Bai. "An Efficient Lightweight Railway Track Segmentation Network for Resource-Constrained Platforms with TensorRT". In: *Intelligent Transportation Infrastructure* 3, liae009 (2024). DOI: 10.1093/iti/liae009.

[35] Deguang Wang and Shujie Cheng. "Railway Image Semantic Segmentation Based on an Improved DeepLabV3+ Model". In: *Proceedings of the 2025 4th International Conference on Intelligent Systems, Communications and Computer Networks*. 2025, pp. 381–387. DOI: 10.1145/3732945.3733001.

[36] Jiawei Peng, Liqiang Zhu, Jingyu Hu, Kexun Wang, Zhihao Huang, and Baoqing Guo. "GQ-Former: Edge-Guided Query Initialization for Robust Railway Scene Segmentation". In: *The Journal of Supercomputing* 82.10, 559 (2026). DOI: 10.1007/s11227-026-08701-4.

[37] Xiying Song, Haifeng Song, Shaoqing Liu, Yu Zhang, and Hairong Dong. "An Onboard Multimodal Sparse Perception Framework for Railway Intrusion Detection". In: *IEEE Sensors Journal* 26.16 (2026), pp. 24478–24490. DOI: 10.1109/JSEN.2026.3708154.

[38] Tao Sun, Baoqing Guo, Zujun Yu, Hang Yu, Tao Ruan, Zhihao Huang, and Binbin Yuan. "A Camera-LiDAR Fusion Network With Asymmetric Attention for Generalizable Railway Foreign Object Segmentation". In: *IEEE Sensors Journal* 26.14 (2026), pp. 21778–21789. DOI: 10.1109/JSEN.2026.3701111.

[39] Lu Yang, Xinyi Wang, Guodong Zhu, Mengpan Xiao, Yiwen Bai, and Chongke Bi. "RailMTL: Real-Time Efficient Perception Generator for Railway Multitask Learning". In: *Journal of Real-Time Image Processing* 23.4, 129 (2026). DOI: 10.1007/s11554-026-01930-z.

[40] Tao Sun, Baoqing Guo, Tao Ruan, Xingfang Zhou, and Dingyuan Bai. "RailEINet: A Novel Scene Segmentation Network for Automatic Train Operation Based on Feature Alignment". In: *Engineering Applications of Artificial Intelligence* 138, 109295 (2024). DOI: 10.1016/j.engappai.2024.109295.

[41] Markus Ziegler, Vishal Mhasawade, Martin Köppel, Philipp Neumaier, and Volker Eiselein. "A Comprehensive Framework for Evaluating Vision-Based On-Board Rail Track Detection". In: *2023 IEEE Intelligent Vehicles Symposium (IV)*. 2023, pp. 1–8. DOI: 10.1109/IV55152.2023.10186659.

[42] Vojtěch Drahý, Radek Mařík, and Heikki Kälviäinen. "Context-Aware Railway Video Segmentation with Unsupervised Evaluation". In: *Pattern Analysis and Applications* 29.2, 78 (2026). DOI: 10.1007/s10044-026-01647-6.

[43] Kanwal Jahan, Joshua Niemeijer, Nils Kornfeld, and Michael Roth. "Deep Neural Networks for Railway Switch Detection and Classification Using Onboard Camera Images". In: *2021 IEEE Symposium Series on Computational Intelligence*. 2021, pp. 1–7. DOI: 10.1109/SSCI50451.2021.9659983.

[44] Lei Tong, Limin Jia, Yixuan Geng, Keyan Liu, Yong Qin, and Zhipeng Wang. "Anchor-Adaptive Railway Track Detection from Unmanned Aerial Vehicle Images". In: *Computer-Aided Civil and Infrastructure Engineering* 38.18 (2023), pp. 2666–2684. DOI: 10.1111/mice.13004.

[45] Bowei Zhou, Guanyu Xing, and Yanli Liu. "Rail Line Detection Algorithm Based on Improved CLRNet". In: *Journal of Shanghai Jiaotong University (Science)* 30.5 (2025), pp. 923–934. DOI: 10.1007/s12204-023-2683-y.

[46] Zichen Yu, Quanli Liu, Wei Wang, Liyong Zhang, and Xiaoguang Zhao. "DALNet: A Rail Detection Network Based on Dynamic Anchor Line". In: *IEEE Transactions on Instrumentation and Measurement* 73 (2024), pp. 1–14. DOI: 10.1109/TIM.2024.3387496.

[47] Zhiming Fang, Ning Sun, Jixin Liu, Lei Chai, and Cong Wu. "LASA: A Line Anchor Self-Assigned Network for Rail Detection". In: *Proceedings of the 16th International Conference on Wireless Communications and Signal Processing*. 2024, pp. 1133–1138. DOI: 10.1109/WCSP62071.2024.10827490.

[48] Jing Wang, Yu Ren, Zhijian Ye, and Zhao Tang. "LRE-Net: A Vision-Based Real-Time Rail Extraction Method Under Metro Tunnel Low-Light Scene". In: *Measurement* 221, 113567 (2023). DOI: 10.1016/j.measurement.2023.113567.

[49] Wentao Liu, Zhangyu Wang, Guizhen Yu, Bin Zhou, and Songyue Yang. "3DRailNet: A Multifocal Cameras Fusion Network for Long-Range 3-D Rail-Track Detection". In: *IEEE Transactions on Industrial Informatics* 22.8 (2026), pp. 7553–7564. DOI: 10.1109/TII.2026.3687135.

[50] Pablo Alonso, Maider Larrazabal, Aitor Iglesias, Nerea Aranjuelo, Mikel Labayen, and Marcos Nieto. "Real-Time Multi-Camera System for On-Board Train Path Estimation". In: *2024 IEEE 12th International Conference on Intelligent Systems (IS)*. 2024, pp. 1–8. DOI: 10.1109/IS61756.2024.10705203.

[51] Haipeng Guo, Xiukun Wei, Yang Ji, Chengyu Ge, Qingfeng Tang, and Kunlin Cai. "Track^{2}Net: A Fast Lightweight Model With Keypoint Alignment and Track Anchors Identification for Railway Line Tracking". In: *IEEE Transactions on Intelligent Transportation Systems* 27.1 (2026), pp. 433–447. DOI: 10.1109/TITS.2025.3628655.

[52] Songyue Yang, Zhangyu Wang, Guizhen Yu, and Wentao Liu. "Key Point Estimate Network for Rail-Track Detection". In: *IEEE Transactions on Intelligent Transportation Systems* 25.5 (2024), pp. 4077–4088. DOI: 10.1109/TITS.2023.3327996.

[53] Wichian Ooppakaew, Jakkrit Onshaunjit, Saifun Khrueakhrai, and Jakkree Srinonchat. "Improving Railway Track Detection With a Mixed-Modality Deep Learning Approach". In: *IEEE Access* 13 (2025), pp. 134541–134551. DOI: 10.1109/ACCESS.2025.3593629.

[54] Tiejian Tang, Jinghao Cao, Xiong Yang, Sheng Liu, Dongsheng Zhu, Sidan Du, and Yang Li. "A Real-Time Method for Railway Track Detection and 3D Fitting Based on Camera and LiDAR Fusion Sensing". In: *Remote Sensing* 16.8, 1441 (2024). DOI: 10.3390/rs16081441.

[55] Ping Chao, Chao-Yang Kao, Yu-Shan Ruan, Chien-Hsiang Huang, and Youn-Long Lin. "HarDNet: A Low Memory Traffic Network". In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, pp. 3551–3560. DOI: 10.1109/ICCV.2019.00365.

[56] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, José M. Alvarez, and Ping Luo. "SegFormer: Simple and Efficient Design for Semantic Segmentation with Transformers". In: *Advances in Neural Information Processing Systems*. Vol. 34. 2021, pp. 12077–12090. URL: https://proceedings.neurips.cc/paper/2021/hash/64f1f27bf1b4ec22924fd0acb550c235-Abstract.html.

[57] Seungmin Lee, Beomseong Kim, and Heesung Lee. "Semantic Segmentation Models with Frozen Weights for Railway Track Detection". In: *Science Progress* 107.4, 00368504241304204 (2024). DOI: 10.1177/00368504241304204.

[58] Oliver Zendel, Markus Murschitz, Marcel Zeilinger, Daniel Steininger, Sara Abbasi, and Csaba Beleznai. "RailSem19: A Dataset for Semantic Rail Scene Understanding". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2019, pp. 1221–1229. DOI: 10.1109/CVPRW.2019.00161.

[59] Xinpeng Li and Xiaojiang Peng. "Rail Detection: An Efficient Row-based Network and a New Benchmark". In: *Proceedings of the 30th ACM International Conference on Multimedia*. 2022, pp. 6455–6463. DOI: 10.1145/3503161.3548050.

[60] Arij Zouaoui, Ankur Mahtani, Mohamed Hadded, Sebastien Ambellouis, Jacques Boonaert, and Hazem Wannous. "RailSet: A Unique Dataset for Railway Anomaly Detection". In: *Proceedings of the 5th IEEE International Conference on Image Processing, Applications and Systems*. 2022, pp. 1–6. DOI: 10.1109/IPAS55744.2022.10052883.

[61] Adrien Bardes, Jean Ponce, and Yann LeCun. "VICReg: Variance-Invariance-Covariance Regularization for Self-Supervised Learning". In: *Proceedings of the 10th International Conference on Learning Representations*. 2022. URL: https://openreview.net/forum?id=xm6YD62D1Ub.

[62] Mohammadjavad Ghorbanalivakili and Gunho Sohn. “TRIT-Net: Triplet-based Railway Instance Tracing Network Using Attraction Field Representation”. In: *Proceedings of the 22nd Conference on Robots and Vision*. 2025. DOI: 10.21428/d82e957c.2ae359db. URL: https://crv.pubpub.org/pub/h6d3dccv.

[63] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. “Visual Instruction Tuning”. In: *Advances in Neural Information Processing Systems*. Vol. 36. 2023, pp. 34892–34916. DOI: 10.52202/075280-1516. URL: https://proceedings.neurips.cc/paper_files/paper/2023/hash/6dcf277ea32ce3288914faf369fe6de0-Abstract-Conference.html.

[64] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. “Segment Anything”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 3992–4003. DOI: 10.1109/ICCV51070.2023.00371.

[65] Xin Lai, Zhuotao Tian, Yukang Chen, Yanwei Li, Yuhui Yuan, Shu Liu, and Jiaya Jia. “LISA: Reasoning Segmentation via Large Language Model”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 9579–9589. DOI: 10.1109/CVPR52733.2024.00915.

[66] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. *LoRA: Low-Rank Adaptation of Large Language Models*. 2021. DOI: 10.48550/arXiv.2106.09685. arXiv: 2106.09685 [cs.CL]. URL: https://arxiv.org/abs/2106.09685.

[67] Vincenzo Carletti, Antonio Greco, Alessia Saggese, Camilla Spingola, and Bruno Vento. “Improving Rock Detection Accuracy by Using Vision-Language Models With Textual–Visual Prompt Alignment”. In: *IEEE Access* 14 (2026), pp. 77944–77957. DOI: 10.1109/ACCESS.2026.3695395.

[68] Senqiao Yang, Tianyuan Qu, Xin Lai, Zhuotao Tian, Bohao Peng, Shu Liu, and Jiaya Jia. *LISA++: An Improved Baseline for Reasoning Segmentation with Large Language Model*. 2023. DOI: 10.48550/arXiv.2312.17240. arXiv: 2312.17240 [cs.CV]. URL: https://arxiv.org/abs/2312.17240.

[69] Zhuofan Xia, Dongchen Han, Yizeng Han, Xuran Pan, Shiji Song, and Gao Huang. "GSVA: Generalized Segmentation via Multimodal Large Language Models". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 3858–3869. DOI: 10.1109/CVPR52733.2024.00370.

[70] Tsung-Han Wu, Giscard Biamby, David Chan, Lisa Dunlap, Ritwik Gupta, Xudong Wang, Joseph Gonzalez, and Trevor Darrell. "See, Say, and Segment: Teaching LMMs to Overcome False Premises". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 13459–13469. DOI: 10.1109/CVPR52733.2024.01278.

[71] Hanoona Rasheed, Muhammad Maaz, Sahal Shaji, Abdelrahman Shaker, Salman Khan, Hisham Cholakkal, Rao M. Anwer, Eric Xing, Ming-Hsuan Yang, and Fahad S. Khan. "GLaMM: Pixel Grounding Large Multimodal Model". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 13009–13018. DOI: 10.1109/CVPR52733.2024.01236. URL: https://openaccess.thecvf.com/content/CVPR2024/html/Rasheed_GLaMM_Pixel_Grounding_Large_Multimodal_Model_CVPR_2024_paper.html.

[72] Zhongwei Ren, Zhicheng Huang, Yunchao Wei, Yao Zhao, Dongmei Fu, Jiashi Feng, and Xiaojie Jin. "PixelLM: Pixel Reasoning with Large Multimodal Model". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 26364–26373. DOI: 10.1109/CVPR52733.2024.02491. URL: https://openaccess.thecvf.com/content/CVPR2024/html/Ren_PixelLM_Pixel_Reasoning_with_Large_Multimodal_Model_CVPR_2024_paper.html.

[73] Yi Lu, Jiawang Cao, Yongliang Wu, Bozheng Li, Licheng Tang, Yangguang Ji, Chong Wu, Jay Wu, and Wenbo Zhu. "RSVP: Reasoning Segmentation via Visual Prompting and Multi-modal Chain-of-Thought". In: *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vienna, Austria: Association for Computational Linguistics, 2025, pp. 14699–14716. DOI: 10.18653/v1/2025.acl-long.715. URL: https://aclanthology.org/2025.acl-long.715/.

[74] Gregor Geigle, Radu Timofte, and Goran Glavaš. "Does Object Grounding Really Reduce Hallucination of Large Vision-Language Models?" In: *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Miami, Florida, USA: Association for Computational Linguistics, 2024, pp. 2728–2742. DOI: 10.18653/v1/2024.emnlp-main.159. URL: https://aclanthology.org/2024.emnlp-main.159/.

[75] Dong Shu, Haiyan Zhao, Jingyu Hu, Weiru Liu, Ali Payani, Lu Cheng, and Mengnan Du. "Large Vision-Language Model Alignment and Misalignment: A Survey Through the Lens of Explainability". In: *Findings of the Association for Computational Linguistics: EMNLP 2025*. Suzhou, China: Association for Computational Linguistics, 2025, pp. 1713–1735. DOI: 10.18653/v1/2025.findings-emnlp.90. URL: https://aclanthology.org/2025.findings-emnlp.90/.

[76] Suprosanna Shit, Johannes C. Paetzold, Anjany Sekuboyina, Ivan Ezhov, Alexander Unger, Andrey Zhylka, Josien P. W. Pluim, Ulrich Bauer, and Bjoern H. Menze. "clDice - A Novel Topology-Preserving Loss Function for Tubular Structure Segmentation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 16555–16564. DOI: 10.1109/CVPR46437.2021.01629. URL: https://openaccess.thecvf.com/content/CVPR2021/html/Shit_clDice_-_A_Novel_Topology-Preserving_Loss_Function_for_Tubular_Structure_CVPR_2021_paper.html.

[77] Laurin Lux, Alexander H. Berger, Alexander Weers, Nico Stucki, Daniel Rueckert, Ulrich Bauer, and Johannes C. Paetzold. "Topograph: An Efficient Graph-Based Framework

for Strictly Topology Preserving Image Segmentation". In: *International Conference on Learning Representations*. 2025. URL: https://openreview.net/forum?id=QOzmmNNePz.

[78] Tina Hiebert, Florian Hofstetter, Carsten Thomas, Savera Mushtaq, Eero Kaan, and Biranavan Parameswaran. "Labels4Rails: A Railway Image Annotation Tool and Associated Reference Dataset". In: *Data* 10.12, 210 (2025). DOI: 10.3390/data10120210.

[79] Philipp Jaß and Carsten Thomas. "Using N-Version Architectures for Railway Segmentation with Deep Neural Networks". In: *Machine Learning and Knowledge Extraction* 7.2, 49 (2025). DOI: 10.3390/make7020049.

[80] Nico Stucki, Johannes C. Paetzold, Suprosanna Shit, Bjoern Menze, and Ulrich Bauer. "Topologically Faithful Image Segmentation via Induced Matching of Persistence Barcodes". In: *Proceedings of the 40th International Conference on Machine Learning*. Vol. 202. Proceedings of Machine Learning Research. 2023, pp. 32698–32727. URL: https://proceedings.mlr.press/v202/stucki23a.html.

[81] Huijie Wang, Tianyu Li, Yang Li, Li Chen, Chonghao Sima, Zhenbo Liu, Bangjun Wang, Peijin Jia, Yuting Wang, Shengyin Jiang, Feng Wen, Hang Xu, Ping Luo, Junchi Yan, Wei Zhang, and Hongyang Li. "OpenLane-V2: A Topology Reasoning Benchmark for Unified 3D HD Mapping". In: *Advances in Neural Information Processing Systems*. Vol. 36. 2023, pp. 18873–18884. DOI: 10.52202/075280-0827. URL: https://openreview.net/forum?id=OMOOO3ls6g.

[82] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. "On Calibration of Modern Neural Networks". In: *Proceedings of the 34th International Conference on Machine Learning*. Vol. 70. Proceedings of Machine Learning Research. 2017, pp. 1321–1330. URL: https://proceedings.mlr.press/v70/guo17a.html.

[83] Yukun Ding, Jinglan Liu, Jinjun Xiong, and Yiyu Shi. "Revisiting the Evaluation of Uncertainty Estimation and Its Application to Explore Model Complexity–Uncertainty Trade-Off". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and*

Pattern Recognition Workshops. 2020, pp. 22–31. DOI: 10.1109/CVPRW50498.2020.00010. URL: https://openaccess.thecvf.com/content_CVPRW_2020/html/w1/Ding_Revisiting_the_Evaluation_of_Uncertainty_Estimation_and_Its_Application_to_CVPRW_2020_paper.html.

[84] Anna Rohrbach, Lisa Anne Hendricks, Kaylee Burns, Trevor Darrell, and Kate Saenko. "Object Hallucination in Image Captioning". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 2018, pp. 4035–4045. DOI: 10.18653/v1/D18-1437.

[85] Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. "CLIPScore: A Reference-Free Evaluation Metric for Image Captioning". In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. 2021, pp. 7514–7528. DOI: 10.18653/v1/2021.emnlp-main.595.

[86] Thomas Decker, Ananta R. Bhattarai, and Michael Lebacher. "Towards Scenario-Based Safety Validation for Autonomous Trains with Deep Generative Models". In: *Computer Safety, Reliability, and Security*. Vol. 14181. Lecture Notes in Computer Science. Springer, 2023, pp. 273–281. DOI: 10.1007/978-3-031-40923-3_20.

[87] Matthijs H. Zwemer, Dennis W. J. M. van de Wouw, Egbert G. T. Jaspers, Sveta Zinger, and Peter H. N. de With. "A Vision-based Approach for Tramway Rail Extraction". In: *Video Surveillance and Transportation Imaging Applications 2015*. Vol. 9407. 2015, 94070R. DOI: 10.1117/12.2075641.

[88] Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. "CenterNet: Keypoint Triplets for Object Detection". In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, pp. 6568–6577. DOI: 10.1109/ICCV.2019.00667.

[89] Benjamin Bischke, Patrick Helber, Joachim Folz, Damian Borth, and Andreas Dengel. "Multi-Task Learning for Segmentation of Building Footprints with Deep Neural

Networks". In: *2019 IEEE International Conference on Image Processing (ICIP)*. 2019, pp. 1480–1484. DOI: 10.1109/ICIP.2019.8803050.

[90] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*. Vol. 9351. Lecture Notes in Computer Science. 2015, pp. 234–241. DOI: 10.1007/978-3-319-24574-4_28.

[91] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. *Objects as Points*. 2019. DOI: 10.48550/arXiv.1904.07850. arXiv: 1904.07850 [cs.CV]. URL: https://arxiv.org/abs/1904.07850.

[92] Lichen Zhou, Chuang Zhang, and Ming Wu. "D-LinkNet: LinkNet with Pretrained Encoder and Dilated Convolution for High Resolution Satellite Imagery Road Extraction". In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2018, pp. 182–186. DOI: 10.1109/CVPRW.2018.00034.

[93] Changqian Yu, Changxin Gao, Jingbo Wang, Gang Yu, Chunhua Shen, and Nong Sang. "BiSeNet V2: Bilateral Network with Guided Aggregation for Real-Time Semantic Segmentation". In: *International Journal of Computer Vision* 129.11 (2021), pp. 3051–3068. DOI: 10.1007/s11263-021-01515-2.

[94] Eduardo Romera, José M. Alvarez, Luis M. Bergasa, and Roberto Arroyo. "ERFNet: Efficient Residual Factorized ConvNet for Real-Time Semantic Segmentation". In: *IEEE Transactions on Intelligent Transportation Systems* 19.1 (2018), pp. 263–272. DOI: 10.1109/TITS.2017.2750080.

[95] Sibaji Gaj, Daniel Ontaneda, and Kunio Nakamura. "Automatic Segmentation of Gadolinium-Enhancing Lesions in Multiple Sclerosis Using Deep Learning from Clinical MRI". In: *PLoS ONE* 16.9, e0255939 (2021). DOI: 10.1371/journal.pone.0255939.

[96] Mohammadjavad Ghorbanalivakili, Ashley Varghese, and Gunho Sohn. "Reasoning-guided Ego-path Segmentation for Autonomous Trains using Vision-language Models". In: *The International Archives of the Photogrammetry, Remote Sensing and Spatial In-*

formation Sciences XLIX-B3-2026 (2026), pp. 89–96. DOI: 10.5194/isprs-archives-XLIX-B3-2026-89-2026.

Appendices

Appendix A

List of Publications and Manuscripts

This appendix lists the published papers and active manuscripts on which Chapters 3–5 are based.

A.1 Published Papers

1. Mohammadjavad Ghorbanalivakili et al. "TPE-Net: Track Point Extraction and Association Network for Rail Path Proposal Generation". In: *2023 IEEE 19th International Conference on Automation Science and Engineering (CASE)*. 2023, pp. 1–7. DOI: 10.1109/CASE56687.2023.10260541

2. Mohammadjavad Ghorbanalivakili and Gunho Sohn. "TRIT-Net: Triplet-based Railway Instance Tracing Network Using Attraction Field Representation". In: *Proceedings of the 22nd Conference on Robots and Vision*. 2025. DOI: 10.21428/d82e957c.2ae359db. URL: https://crv.pubpub.org/pub/h6d3dccv

3. Mohammadjavad Ghorbanalivakili, Ashley Varghese, and Gunho Sohn. "Reasoning-guided Ego-path Segmentation for Autonomous Trains using Vision-language Models". In: *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* XLIX-B3-2026 (2026), pp. 89–96. DOI: 10.5194/isprs-archives-XLIX-B3-2026-89-2026

A.2 Submitted Manuscript

1. M. Ghorbanalivakili, J. Kang, and G. Sohn, "TPE-Net: Location-Radius Regression for Ego-Path Candidate Enumeration in Railway Switch Scenes," submitted to *IEEE Open Journal of Intelligent Transportation Systems*, 2026.

A.3 Manuscript Under Review

1. M. Ghorbanalivakili, A. Varghese, and G. Sohn, "Triplet-based Railway Ego-path Candidate Tracing Using a Learned Attraction Field Representation," under review in *Engineering Applications of Artificial Intelligence*, 2026.