Computer Modeling in Engineering & Sciences

Tech Science Press

https://doi.org/10.32604/cmes.2026.089903

ARTICLE

An Actuator-Aware Hierarchical LiDAR Control Framework for Autonomous Articulated Vehicles in Underground Tunnels

Ya Liu[1], **Pingan Peng**[1,*], **Liguan Wang**[1], **Ru Lei**[2] **and Shuangwei Gong**[1]

[1]School of Resources and Safety Engineering, Central South University, Changsha, China
[2]Zijin Engineering Technology Co., Ltd., Changsha, China

*Corresponding Author: Pingan Peng. Email: ping_an@csu.edu.cn

Received: 27 July 2026; Accepted: 24 September 2026; Published: 21 October 2026

ABSTRACT: Autonomous load–haul–dump (LHD) vehicles operating in underground tunnels must cope with degraded visual sensing, branch-level route ambiguity, and the steering constraints of articulated vehicles. This study proposes an actuator-aware hierarchical light detection and ranging (LiDAR) control framework, termed LiDAR-LHD-Net, that separates learned local geometric steering from sparse topological route guidance. The local policy is trained by behavioral cloning from a nonlinear model predictive control expert and processes raw three-dimensional point clouds using a multi-branch PointNet-style encoder, residual feature enhancement, spatial self-attention, and a bidirectional long short-term memory (Bi-LSTM)-based feature-processing block. In the evaluated implementation, the Bi-LSTM receives only the current synchronized LiDAR feature and measured articulation state as a sequence of length one. Sparse waypoint bearing provides branch-level route intent, and distance-scheduled arbitration adjusts its contribution near topological decision regions before the final articulation command is applied. In Gazebo simulation, the local policy achieved a mean absolute error of approximately 0.036 rad and coefficient-of-determination (R^2) values above 0.996 across three tunnel environments. Same-modality ablations indicate that spatial attention and the Bi-LSTM-based feature-processing block improve open-loop prediction under the reported protocol. Representative closed-loop cases demonstrate the integration of local steering and sparse route guidance in straight, right-angle-turning, and multi-branch layouts. The results demonstrate effective hierarchical coordination between LiDAR-based local steering and sparse route guidance across the evaluated underground simulation scenarios.

KEYWORDS: Hierarchical control; LiDAR point clouds; articulated vehicles; load–haul–dump; autonomous navigation

1 Introduction

Learning-based autonomy has advanced rapidly in road transport and mobile robotics, but underground load–haul–dump (LHD) vehicles (Fig. 1) operate in narrow, poorly illuminated roadways with repetitive geometry, limited clearance, and articulation and hydraulic-actuation constraints [1]. These conditions make closed-loop navigation more demanding than surface-road steering.

Conventional LHD automation is commonly organized as a modular pipeline comprising perception [2,3], localization [4], planning [5,6], and control [7]. This separation provides interpretable intermediate states and permits specialized optimization. Its system performance, however, depends on the compatibility of module interfaces: uncertainty introduced by one stage propagates downstream, and locally accurate modules do not necessarily produce a robust closed loop. These interface effects become consequential in confined tunnels, where small geometric or localization errors can lead to large control consequences.

Copyright © 2026 The Authors. Published by Tech Science Press. This work is licensed under a Creative Commons Attribution 4.0 International License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.



(a) (b)

Figure 1: Representative underground LHD vehicle and corresponding digital model. (**a**) Underground physical LHD. (**b**) LHD simulation model.

End-to-end learning offers a complementary strategy by estimating control commands directly from sensor observations [8,9]. It can learn relationships between environmental geometry and articulated motion that are difficult to encode with hand-designed rules. A purely reactive end-to-end policy, however, has two structural limitations. First, locally similar observations at a T- or Y-junction can correspond to different valid routes, so geometry alone does not determine global intent. Second, an unconstrained regression output need not respect articulation magnitude or rate limits [10].

These limitations motivate a hierarchical rather than a strictly monolithic formulation. In the proposed framework, a learned local policy estimates steering from the observable tunnel geometry, sparse topological waypoints specify branch-level route intent, and actuator-aware command handling limits the final articulation command to the admissible range. These functions are coordinated through distance-scheduled arbitration while retaining explicit interfaces between local perception, route intent, and command generation.

Sensor modality is also central to the problem. Most end-to-end driving studies concern surface passenger vehicles and use camera images, sometimes together with route commands or planner displays [11]. Underground tunnels provide little ambient illumination, weak and repetitive visual texture, airborne dust, water mist, and strong headlamp reflections. These factors limit the direct application of camera-based methods to LHD operations and have led to the adoption of active 3D ranging technology.

Light detection and ranging (LiDAR) provides direct range measurements and has been widely used for localization and navigation of underground mining vehicles, including LHDs [12,13]. However, airborne dust and water droplets can attenuate laser returns, generate spurious measurements, and reduce point-cloud fidelity [14,15]. Raw LiDAR point clouds provide geometric information about tunnel-wall profiles, roadway dimensions, and turning boundaries. In the present study, each point cloud is normalized before being provided to the local policy. Consequently, the learned input representation emphasizes the relative geometric arrangement of the tunnel boundaries and vehicle surroundings, while absolute scale is not retained in the normalized network input and is handled outside the learned representation. The present work therefore uses LiDAR as the primary exteroceptive input and treats the camera-based PilotNet model as a cross-modal reference rather than a component-matched architectural baseline.

The unresolved systems question is therefore not simply how to regress a steering angle from a point cloud. It is how to coordinate local geometric reaction, discrete route intent, and constrained articulated actuation under partial observation. We investigate this question in a Gazebo-based LHD simulation and evaluate both open-loop steering prediction and closed-loop navigation behavior.

Existing end-to-end driving studies have established conditional imitation learning, trajectory-guided control prediction, multimodal fusion, and LiDAR-based steering. The problem considered here differs in the coordination required by a centrally articulated underground LHD: local tunnel geometry determines an immediate steering response but does not specify mission-level branch choice, while the resulting articulation command must remain compatible with vehicle actuation. The proposed framework addresses these coupled requirements through a LiDAR-based local steering policy, sparse route guidance, and distance-scheduled command arbitration.

The main contributions of this study are as follows:

(1) Hierarchical coordination for underground articulated navigation. Underground LHD navigation is formulated as a coordination problem involving learned local geometric steering, sparse mission-level route intent, and actuator-aware command generation. The proposed framework explicitly separates these functions and coordinates them through distance-scheduled arbitration near topological decision regions.

(2) LiDAR-based local steering from raw point sets. A learning-based local policy estimates the articulation-steering response directly from three-dimensional LiDAR point clouds and the measured articulation state. A multi-branch PointNet-style encoder, residual feature enhancement, spatial self-attention, and a Bi-LSTM-based feature-processing block are integrated to form a compact representation of tunnel geometry for articulated steering.

(3) Integration of local steering and sparse route guidance. A deterministic target-bearing term supplies branch-level route intent, while a distance-dependent arbitration weight adjusts its influence on the learned local steering estimate. The resulting command is limited before being supplied to the simulated LHD. Open-loop experiments evaluate the learned local policy, and representative closed-loop cases demonstrate the functional integration of local steering, waypoint guidance, and command arbitration.

The remainder of this paper is organized as follows. Section 2 reviews relevant automation, end-to-end driving, and point-cloud methods. Section 3 formulates the control problem and presents the proposed framework and training strategy. Section 4 describes the simulation protocol and results. Section 5 discusses the transferable systems contribution, evidence boundaries, and limitations. Section 6 concludes the paper.

2 Related Work

2.1 Automation for Mining and Heavy Machinery

Traditional LHD automation uses explicit perception [16,17], decision-making [18], planning [19,20], and control [21] modules. Such architectures expose intermediate representations and facilitate fault localization [22,23]. Their limitation is not modularity itself, but system-level sensitivity to interface uncertainty and timing: an error or delay in one stage can alter all downstream decisions, while independent module objectives may not optimize the resulting closed loop [24].

For confined industrial vehicles, robustness must therefore be assessed at the closed-loop level rather than inferred solely from the accuracy of individual components. A useful architecture must also accommodate vehicle-specific actuation limits without making every perception and planning component dependent on a detailed dynamic model.

Learning-based policies reduce manually specified interfaces and have been applied widely in end-to-end road driving and agricultural machinery. Nevertheless, underground LHDs combine articulated kinematics, narrow clearances, repetitive geometry, and degraded sensing in a way that remains comparatively underexplored.

While effective in structured environments, modular systems often lack the adaptability of learning-based approaches. With recent advances in convolutional neural networks (CNNs), end-to-end learning methods have been extensively investigated for autonomous driving control [25–27]. Nevertheless, the combination of articulated kinematics, confined roadways, and degraded sensing encountered by underground LHDs remains comparatively underexplored.

Existing underground LHD research has predominantly adopted modular autonomy architectures in which LiDAR supports localization, mapping, path planning, or model-based path tracking rather than direct learning of articulation steering from raw three-dimensional point sets. Recent underground mining studies have also investigated solid-state-LiDAR re-localization and advanced path-tracking control, demonstrating substantial progress in individual navigation and control components. However, these methods differ from the present task in both control interface and sensing-to-command formulation. Consequently, a directly matched benchmark that maps raw 3D LiDAR observations to articulation steering for an underground LHD is not readily available within the experimental formulation considered here.

2.2 Camera-Based Methods and their Limitations

Camera-based end-to-end driving was popularized by influential behavioral-cloning models such as NVIDIA PilotNet, which directly maps forward images to steering commands [28]. PilotNet established a representative sensor-to-steering paradigm and remains a useful historical reference for evaluating direct control learning. Subsequent conditional imitation-learning approaches further introduced high-level navigation commands so that visually similar observations could produce different control actions according to route intent [9]. These developments demonstrated that direct control learning can be augmented with mission-level information rather than relying exclusively on reactive perception.

Nevertheless, these approaches were developed primarily for surface-road vehicles and camera-based sensing. Underground tunnels present unstable illumination, dust, water mist, repetitive rock texture, and strong headlamp reflections, which can substantially degrade camera observations [29]. Moreover, conventional road-vehicle steering differs from the articulated steering mechanism of an LHD.

In the present study, PilotNet is included as a representative camera-based behavioral-cloning architecture that established the direct sensor-to-steering paradigm. In the present study, it provides a cross-modal reference for steering prediction in the same simulated tunnel environment, while the same-modality architectural analysis is performed separately with LiDAR-based ablations.

2.3 LiDAR and Point Cloud Processing

PointNet introduced permutation-invariant processing of unordered point sets, and PointNet++ extended this idea through hierarchical neighborhood aggregation [30,31]. The encoder used here is PointNet-style rather than a standard PointNet++ backbone: three parallel one-dimensional convolutional branches extract representations of different depths, which are subsequently fused and spatially reweighted.

2.4 LiDAR-Based End-to-End Driving

LiDAR-based end-to-end driving has been investigated using bird's-eye-view (BEV) representations, LiDAR-derived images, and fusion with global navigation satellite system (GNSS) information [11,32–34]. These approaches demonstrate that LiDAR can support direct driving decisions while avoiding dependence on ambient illumination. However, projection-based representations may discard part of the original three-dimensional point-set structure, and GNSS-dependent formulations are difficult to apply directly in underground tunnels.

Conditional imitation-learning methods introduce high-level route information so that similar local observations can generate different actions according to navigation intent. Trajectory-guided Control Prediction (TCP) [35] combines trajectory prediction with direct control, while TransFuser [36] fuses camera and LiDAR features within an end-to-end architecture. Reinforcement learning (RL) and deep reinforcement learning (DRL) have also been studied for articulated-vehicle tracking and underground loader control [37–39]. These studies demonstrate the potential of learning-based control but generally address surface-road driving, trajectory tracking, maneuvering, or loading rather than direct articulation steering from raw three-dimensional LiDAR point sets in underground tunnel navigation.

The present study therefore adopts a different learning formulation. The local policy is trained by behavioral cloning from a nonlinear model predictive control (NMPC) expert rather than through reward-driven trial-and-error interaction, while sparse waypoint guidance explicitly provides branch-level route intent.

Underground LHD autonomy has traditionally retained more explicit localization, planning, and control components. Representative studies have combined LiDAR localization with topological navigation and model predictive control (MPC), or developed re-localization and hybrid navigation methods for underground machines [40,41]. These approaches provide strong modular solutions but differ from the sensing-to-command formulation considered here.

The present study therefore adopts behavioral cloning from a nonlinear model predictive control (NMPC) expert for local steering and uses sparse waypoint guidance to provide branch-level intent. The proposed framework occupies an intermediate position between a monolithic end-to-end policy and a fully modular localization–planning–tracking architecture. NVIDIA PilotNet is retained only as a historical cross-modal reference, whereas the PointNet–multilayer perceptron (MLP) configuration and module-removal experiments provide the available same-modality evidence. Accordingly, the experimental design combines a historical cross-modal reference with same-modality LiDAR ablations to evaluate steering prediction and the contribution of the local-policy components.

3 System Formulation and Methodology

Let o_t denote the LiDAR point cloud available at control step t, $s_t = \phi_t$, the measured central-articulation angle, $p_t = \left(x_t^{veh}, y_t^{veh}, \psi_t^{veh}\right)$, the vehicle pose used for waypoint guidance, and $g_t = \left(x_t^g, y_t^g\right)$, the next sparse topological waypoint. The learned local steering policy is written as $\delta_t^{net} = \pi\theta\left(o_t, s_t\right)$, where θ denotes the learned network parameters. The deterministic waypoint-guidance module produces $\delta_t^{nav} = h\left(p_t, g_t\right)$, and the arbitration stage generates the final commanded articulation angle δ_t^{final}.

3.1 Behavioral Cloning and System Assumptions

The local steering policy is trained by supervised behavioral cloning [42] to reproduce steering behavior generated by an NMPC-based path-tracking controller, which serves as the expert policy during data collection. The expert controller receives the simulated LHD state and the prescribed reference path and computes the central articulation steering action required for trajectory tracking. The corresponding articulation-steering commands are recorded together with the LiDAR observations and are subsequently used as the supervised targets of the learned local policy.

The NMPC expert incorporates state and control constraints during finite-horizon optimization, yielding articulation-steering demonstrations within the prescribed operating range of the simulated LHD. The learned policy reproduces these demonstrations, while the hierarchical controller subsequently applies command arbitration and limiting before actuation. The present study considers static tunnel geometry, constant low longitudinal speed, synchronized sensing and state feedback, and no dynamic obstacles.

3.2 Dataset Construction and Preprocessing

Data were generated in a self-constructed LHD simulation environment using Ubuntu 20.04, Gazebo 11, and Robot Operating System (ROS 1). Velodyne LiDAR point clouds, forward-camera images, vehicle articulation states, and articulation-control records were collected during the same simulated LHD runs. The camera and LiDAR datasets therefore correspond to the same tunnel environment and vehicle trajectories rather than to independently generated routes. However, the two sensors operated at different sampling frequencies, resulting in different numbers of recorded frames. After dataset construction and preprocessing, the LiDAR dataset contained 14,729 samples, whereas the camera dataset contained 26,139 samples. The camera data were used only for the historical PilotNet reference, whereas the proposed local policy was trained using the LiDAR data and vehicle-state information. Fig. 2 shows the sensor arrangement.

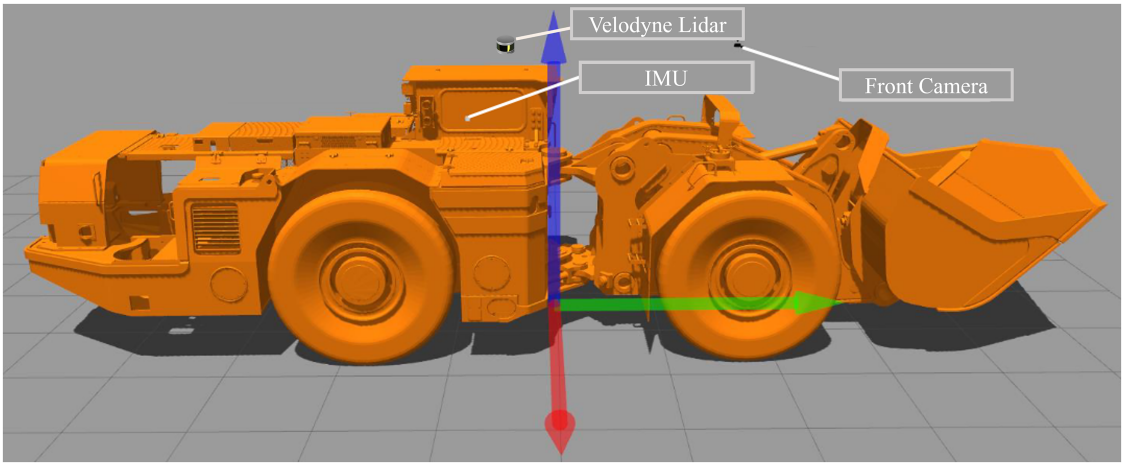


Figure 2: Sensor arrangement on the articulated LHD model.

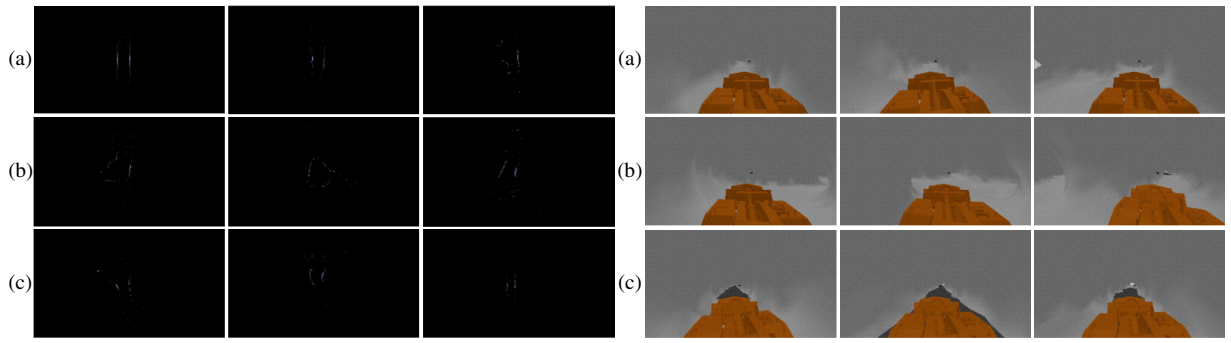
LiDAR frames and steering records were stored with ROS timestamps and paired through timestamp-based synchronization during dataset construction. The steering reference represents the central articulation-joint steering quantity generated during NMPC-controlled motion rather than a steering-wheel angle. The development data were divided using an 80/20 frame-level training/validation split. Two additional tunnel environments were reserved for external simulation evaluation. Because the development split was performed at frame level within the same tunnel, it is not trajectory-disjoint, as discussed in Section 5.4.

Representative paired observations are shown in Fig. 3 for straight driving, entry to a turn, and post-turn travel.

Raw point clouds are unordered and exhibit non-uniform density. A consistent preprocessing and augmentation pipeline was therefore used to form fixed-size network inputs.

(1) Data Preprocessing

After voxel-grid downsampling at 0.10 m resolution, each point cloud is converted to a fixed-size input of $N = 4096$ points by random subsampling or zero padding. Per-cloud centering and scale normalization reduce variation in point density and coordinate range, yielding a consistent input representation for network training. Under this normalization, the learned representation emphasizes relative tunnel geometry rather than absolute geometric scale.



(A) LiDAR point clouds

(B) Camera observations synchronized with the LiDAR examples in Fig.3

Figure 3: Representative synchronized sensor observations: (**A**) LiDAR point clouds and (**B**) Camera images during (**a**) Straight driving, (**b**) Turning, and (**c**) Post-turn travel.

(2) Data Augmentation

Online training augmentation included lateral reflection with corresponding articulation-label sign inversion, in-plane rotation within ±30°, translation in the x–y plane, scaling from 0.95 to 1.05, and random removal of up to 20% of the points. The reflection operation was label-coupled because left–right mirroring reverses the sign of the corresponding articulation command. The remaining geometric transformations were introduced as generic representation-level regularization to increase variation in the normalized point-set input, as shown in Fig. 4.

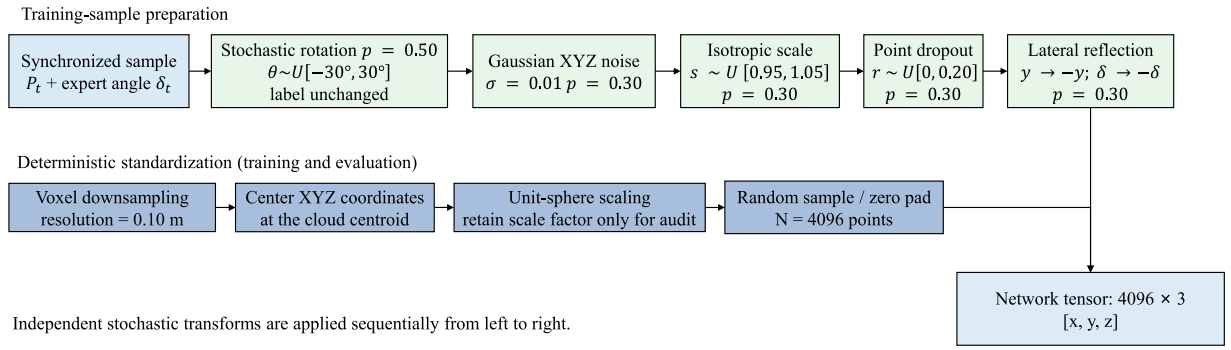


Figure 4: Point-cloud preprocessing and augmentation workflow. Each processed point cloud contains N = 4096 points represented by cartesian coordinates (x, y, z), resulting in a 4096 × 3 single-frame network input.

The in-plane rotation is used as a representation-level perturbation of the normalized point set. A physical change in LHD pose, by contrast, involves a coupled change in vehicle position or heading, LiDAR geometry, vehicle state, and the corresponding articulation target. The ±30° interval therefore specifies the augmentation range applied to the point-cloud representation rather than a distribution of physical vehicle-orientation errors.

3.3 Actuator-Aware Hierarchical Architecture

LiDAR-LHD-Net estimates a local articulation-steering command from the current LiDAR observation and measured central-articulation state, as shown in Fig. 5. The learned policy is embedded in a hierarchical closed-loop controller that supplements local geometric steering with sparse route intent near topological

decision regions and limits the resulting command before actuation. The measured central-articulation angle ϕ_t represents the actual relative orientation between the front and rear vehicle bodies and is distinct from the commanded articulation quantities δ_t^{net}, δ_t^{nav}, and δ_t^{final}.

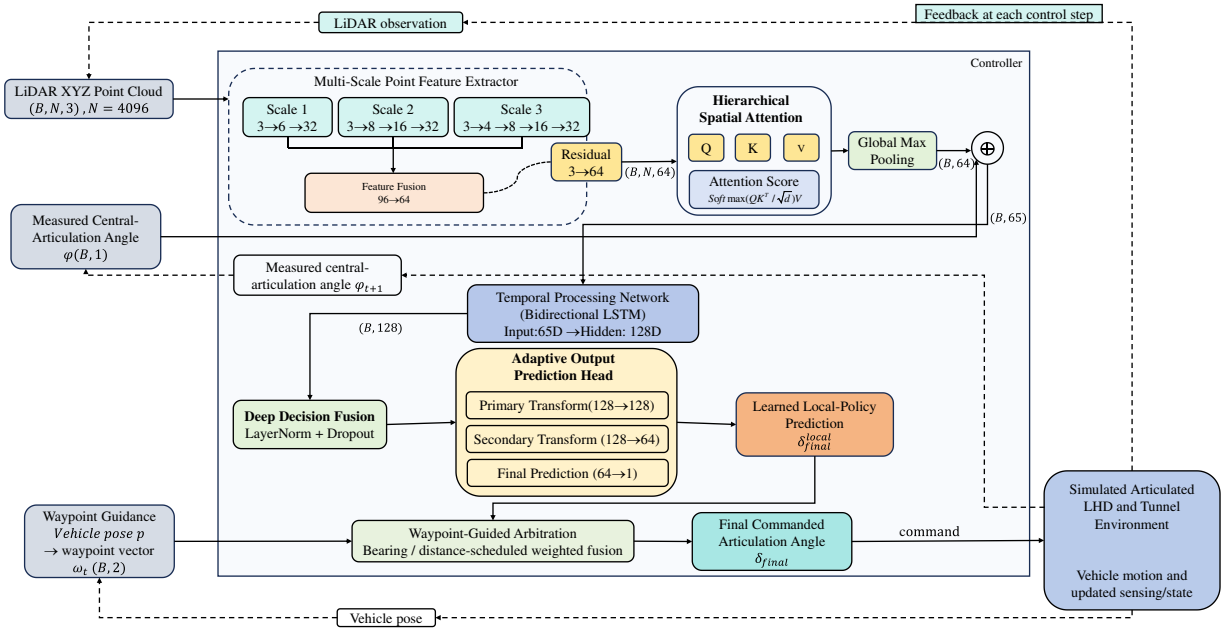


Figure 5: Closed-loop structure of the proposed hierarchical controller. The current LiDAR observation and measured central-articulation angle feed the learned local steering policy; sparse waypoints provide route intent; distance-scheduled arbitration and command limiting generate the final articulation command.

The local policy comprises a multi-branch point-cloud encoder with residual feature enhancement, a spatial self-attention module, a Bi-LSTM-based feature-processing block, and a regression head. The residual connection is applied before spatial attention. In the evaluated implementation, the Bi-LSTM receives only the fused feature associated with the current synchronized observation as a sequence of length one; no multi-frame LiDAR history or future observation is supplied to the network.

The controller operates in closed loop: the final commanded articulation angle is applied to the simulated LHD, and the resulting vehicle motion updates the LiDAR observation, measured articulation state, and vehicle pose used at the next control step. This feedback path is shown explicitly in the Fig. 5.

(1) Multi-branch PointNet-style encoder

The encoder receives a fixed-size XYZ point set of $N = 4096$ points, represented as $B \times N \times 3$ for mini-batch processing. Three parallel shared point-wise one-dimensional convolutional branches then transform the three-coordinate input into point-wise feature representations of different depths. Their outputs are concatenated and projected to a 64-dimensional fused representation for subsequent residual enhancement and spatial self-attention, as illustrated in Fig. 6A. By combining parallel branches with different transformation depths, the encoder captures complementary point-wise representations without explicit neighborhood grouping, farthest-point sampling, or graph construction, thereby retaining a compact network structure.

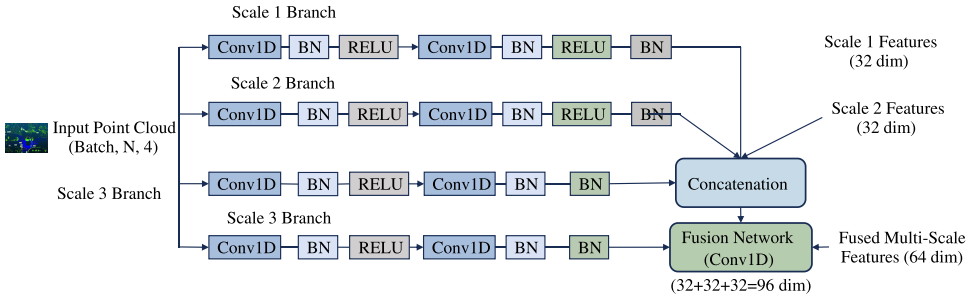
In Fig. 6A, Conv1D denotes a shared point-wise one-dimensional convolution, BN denotes batch normalization, and ReLU denotes the rectified linear unit. The three parallel branches operate on the same point-cloud feature input but use different transformation depths. Their outputs are concatenated along the

feature-channel dimension and subsequently projected to the 64-dimensional fused representation used by the following modules.

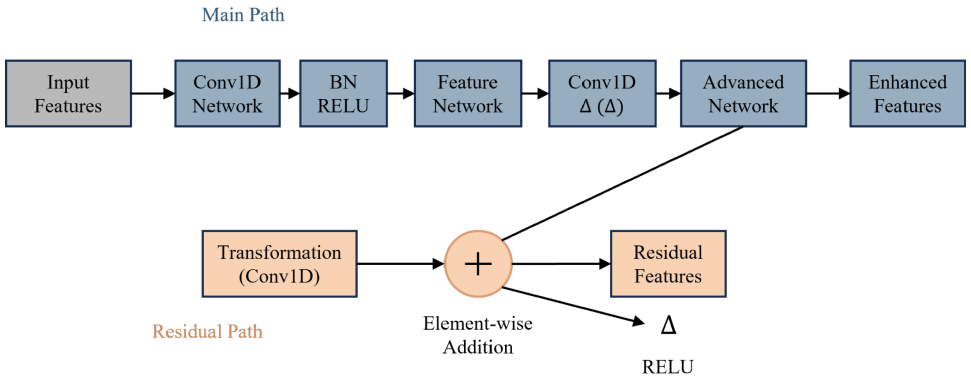
A residual feature path is incorporated after multi-branch feature fusion. The 64-dimensional fused representation forms the main feature path, while a point-wise projection provides a dimension-matched residual representation. The two representations are added before the spatial self-attention module. This residual connection facilitates gradient propagation and preserves transformed input information without introducing a separate high-level decision module. It is therefore treated as an implementation mechanism within the point-cloud encoder rather than as a standalone methodological contribution, as illustrated in Fig. 6B.

(2) Spatial self-attention module

After residual feature enhancement, spatial self-attention is applied to the point-wise representation, as shown in Fig. 6C. Not all observed points contribute equally to a steering decision. Scaled dot-product self-attention computes correlations among the learned point features and reweights their representation so that steering-relevant geometric structures can contribute more strongly than less informative observations. In the tunnel scenarios considered here, these structures may include wall boundaries, curvature transitions, and candidate branch regions. The ablation study in Section 4.4 evaluates the aggregate predictive contribution of this module.

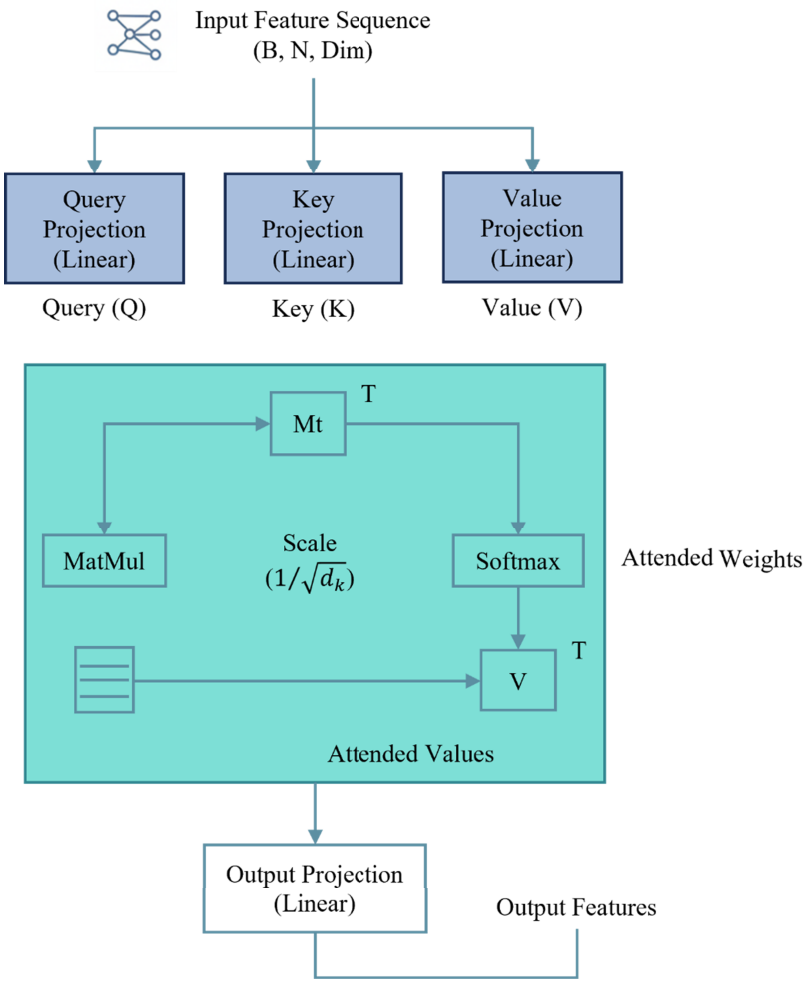


(A) Multi-branch PointNet-style feature encoder



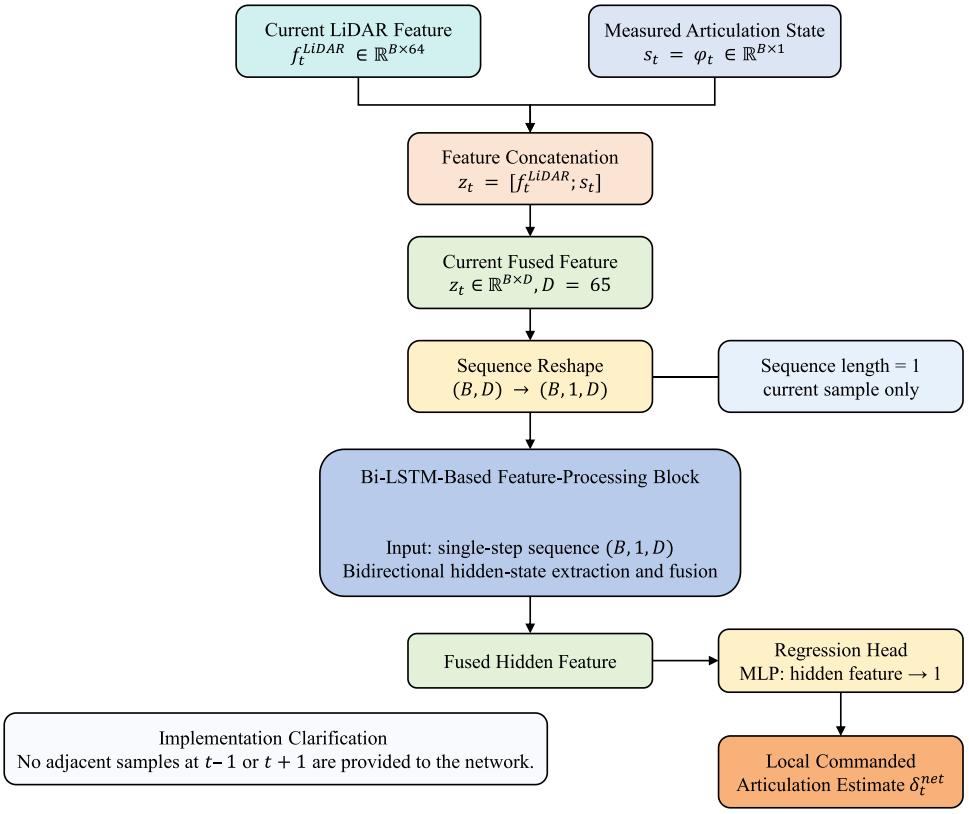
(B) Residual feature connection used in the local policy

Figure 6: (Continued)



(C) Scaled dot-product spatial self-attention used to reweight point features

Figure 6: (Continued)



(D) Bi-LSTM-based processing of the current fused point-cloud and articulation-state representation

Figure 6: Consolidated local steering-policy architecture: (**A**) Multi-branch point-feature encoder; (**B**) Residual enhancement; (**C**) Spatial self-attention; and (**D**) Bi-LSTM-based feature processing of the current synchronized feature with sequence length one.

Let Q, K, and V denote the query, key, and value matrices obtained by linear projection of the point-wise input features. The scaled dot-product attention is computed as $A = Softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)$, $F_{att} = AV$, where d_k denotes the feature dimension of each query/key vector, and the factor $\frac{1}{d_k}$ prevents the magnitude of the dot product from increasing excessively with feature dimension. The superscript $(\cdot)^T$ denotes matrix transpose. A is the attention-weight matrix and F_{att} is the resulting attended point-feature representation.

(3) Bi-LSTM-based feature-processing block

After spatial feature reweighting and global aggregation, the point-cloud descriptor is concatenated with the synchronized articulation-state scalar and supplied to the Bi-LSTM-based feature-processing block, as shown in Fig. 6D. In the evaluated implementation, this fused feature is represented as a sequence of length one; therefore, no multi-frame LiDAR history or future observation is used. With a sequence length of one, the Bi-LSTM operates here as a gated bidirectional transformation of the current fused representation; no multi-frame temporal context enters the recurrent block.

(4) Local steering estimation and command generation

The steering quantity estimated by the learned local policy is the articulation-angle variable used for lateral steering of the simulated LHD. After point-cloud feature extraction and spatial reweighting, global max pooling converts the point-wise representation into a fixed-dimensional geometric descriptor.

This descriptor is concatenated with the synchronized articulation-state scalar and passed through the Bi-LSTM-based feature-processing block and the subsequent regression head. The regression head consists of fully connected transformations followed by a single linear output neuron, producing the scalar local steering estimate.

Accordingly, δ_{net} should be interpreted as the learned local steering estimate associated with the currently observed tunnel geometry and articulation state. It is not itself the final actuator input near a topological decision region. Outside such regions, the local estimate provides the dominant steering contribution. Near an intersection or branch, it is combined with the waypoint-derived directional term δ_{nav} through the distance-scheduled arbitration described in the following subsection. The resulting δ_{final}, after command limiting, is the steering signal supplied to the simulated LHD actuator.

(5) Sparse-waypoint route guidance and arbitration

A local end-to-end policy cannot determine route intent when similar observations admit several geometrically feasible branches. At a T- or Y-junction, similar local LiDAR geometries may correspond to different mission-dependent branches. Route selection therefore requires information beyond the instantaneous local geometry.

The proposed controller therefore combines learned local steering with sparse waypoint guidance near topological decision regions.

In this simulation study, several sparsely distributed waypoints are predefined as local goals before each closed-loop run to prevent the vehicle from driving blindly. These waypoints are neither derived from the LiDAR point cloud nor generated by LiDAR-LHD-Net. This separation is intentional: the learned local strategy estimates steering based on the currently observable tunnel geometry, while the sequence of waypoints specifies which branch to take when multiple locally feasible routes exist. Therefore, W_{target} represents the next sparse waypoint associated with the predefined route.

The final actuator command δ_{final} is determined from two complementary steering components.

Local geometric steering: LiDAR-LHD-Net produces the scalar local steering estimate δ_{net} from the point-cloud representation and articulation-state input. This term represents the learned steering response associated with the observable local tunnel geometry.

Topology-guided steering: the waypoint module deterministically computes a directional guidance term δ_{nav} from the relative bearing between the vehicle forward heading and the next sparse waypoint W_{target}. This term supplies branch-level route intent and does not replace the learned response to local tunnel geometry.

To make the target-bearing computation explicit, let the current planar vehicle position be:

$$P_{veh} = (x_v, y_v) \tag{1}$$

and let the next sparse route waypoint be:

$$W_{t\arg et} = (x_\omega, y_\omega) \tag{2}$$

Both positions are expressed in the same global coordinate frame. The absolute bearing from the current vehicle position to the target waypoint is calculated using the two-argument arctangent:

$$\psi_{t\arg et} = a\tan 2\left(y_\omega - y_v, x_\omega - x_v\right) \tag{3}$$

Here, ψ_{target} is the absolute direction from the vehicle to the selected waypoint.

Let ψ_{veh} denote the current forward-body heading of the vehicle in the same global coordinate frame. The relative target-bearing error is then calculated as:

$$e_{\psi} = wrap_{[-\pi,\pi]}\left(\psi_{target} - \psi_{veh}\right) \tag{4}$$

The wrapping operation constrains the directional error to the interval from $-\pi$ to π and prevents artificial discontinuities when the heading crosses the $\pm\pi$ boundary.

The waypoint-guidance steering term is obtained by limiting the relative bearing error to the admissible steering range:

$$\delta_{nav} = Clip\left(e_{\psi}, \delta_{\min}, \delta_{\max}\right) \tag{5}$$

The two-argument arctangent preserves the correct target quadrant, while angular wrapping avoids discontinuities near $\pm\pi$. The waypoint term supplies branch-level route intent rather than local wall-clearance information; the latter remains primarily represented by the LiDAR-based local steering estimate.

For example, in the right-angle turning case, the selected waypoint on the outgoing branch generates a positive relative bearing before the vehicle reaches the front wall. Under the articulation-sign convention used in the simulator, this produces a positive waypoint-guidance command corresponding to leftward central articulation. As the vehicle approaches the decision region, the increasing arbitration weight causes this directional term to contribute more strongly to the final command, thereby initiating the required turn before reaching the front wall.

The learned local steering estimate and waypoint-guidance term are combined through the distance-scheduled arbitration law:

$$\delta_{final} = (1-\lambda)\cdot\delta_{net} + \lambda\cdot Clip\left(\delta_{nav}, \delta_{\min}, \delta_{\max}\right) \tag{6}$$

where: δ_{final} is the final commanded articulation angle supplied to the simulated articulation controller; δ_{net} is the LiDAR-LHD-Net local steering estimate; δ_{nav} is the waypoint-derived steering guidance; λ is the navigation-guidance weight; $\delta_{\min}$ and $\delta_{\max}$ define the admissible steering-command range.

The arbitration weight is scheduled according to the vehicle's distance from the upcoming topological decision region. Let W_{node} denote the center of that decision region. The distance between the current vehicle position and the decision-region center is $d = \|P_{veh} - W_{node}\|_2$.

For an influence radius R, the navigation weight is defined as:

$$\lambda(d) = \begin{cases} \lambda_{\min}, & if\ d > R \\ \lambda_{\min} + (\lambda_{\max} - \lambda_{\min})\cdot\left(1 - \dfrac{d}{R}\right), & if\ 0 \le d \le R \end{cases} \tag{7}$$

For the simulated tunnel geometry, the arbitration parameters were set empirically to $R = 15.0$ m, and $\lambda_{\min} = 0.1$, $\lambda_{\max} = 0.6$. These settings maintain local-policy dominance away from decision regions while progressively increasing waypoint influence near an intersection. The lower weight keeps the learned local steering dominant away from decision regions, whereas the upper weight limits waypoint influence so that local geometric steering remains active even near an intersection. The radius R determines the spatial extent over which this transition occurs.

Outside the influence region, where $d > R$, the navigation weight remains at $\lambda = 0.1$, so the learned local steering estimate remains dominant.

As the vehicle enters the influence region and approaches the decision-region center, the distance d decreases and the navigation weight increases linearly. Consequently, the sparse waypoint contributes progressively more strongly to the final steering command.

At the center of the decision region, $d = 0$, and therefore $\lambda = \lambda_{\max} = 0.6$. At this point, waypoint guidance reaches its maximum predefined influence, while the LiDAR-based local term retains a weight of 0.4.

This design separates two functions that cannot be obtained reliably from the same local observation alone. The LiDAR-based term provides a reaction to observable tunnel geometry, whereas the target-bearing term provides mission-dependent route intent. Their distance-scheduled combination allows global guidance to become stronger near intersections without allowing waypoint direction to dominate local steering throughout the entire tunnel.

3.4 Training Objective and Optimization

The local policy is trained by supervised behavioral cloning using a scene-weighted regression objective. Mean squared error (MSE) and mean absolute error (MAE) are combined to balance sensitivity to large prediction errors and robustness to individual deviations.

(1) Base regression loss

$$L_{base} = \alpha \left(y - \hat{y}\right)^2 + \beta \left|y - \hat{y}\right| \tag{8}$$

where, y denotes the expert articulation-steering target and $\hat{y}$ the network prediction. The weighting coefficients used in the reported training configuration are $\alpha = 0.7$ and $\beta = 0.3$.

(2) Scene-importance weighting

To increase the contribution of samples containing larger articulation commands, a scene-dependent sample weight ω_i is applied:

$$\omega_i = \begin{cases} \gamma, & if\ |y_i| > |\overline{y}| \\ 1, & otherwise \end{cases} \tag{9}$$

where, $|\overline{y}|$ is the mean absolute steering magnitude computed from the training set and $\gamma = 2.0$ is the weighting factor used in the reported configuration. The batch training objective is:

$$L = \frac{1}{N} \sum_{i=1}^{N} \omega_i L_{base,i} \tag{10}$$

This loss is used to train the local steering estimator. Actuator-aware command handling is implemented at the controller stage through waypoint arbitration and command limiting rather than through an additional actuator-dynamics model in the local-policy training loss.

(3) Optimization settings

The model is trained using the AdamW optimizer with an initial learning rate of 1×10^{-4} and a weight decay of 1×10^{-5}.

4 Simulation Experiments and Analysis

The experiments were conducted entirely in the Gazebo-based LHD simulation environment. After preprocessing, the LiDAR and camera datasets contained 14,729 and 26,139 samples, respectively. Both

modalities were recorded along the same simulated vehicle trajectories; the difference in sample count results primarily from their different sampling frequencies. The camera data are used only for the historical PilotNet reference and are therefore not interpreted as a sample-count-matched modality comparison. The evaluation covers nominal simulated conditions and does not explicitly model mine-specific sensing degradation, localization drift, vehicle-parameter uncertainty, dynamic obstacles, or actuator disturbances.

All experiments were conducted on a desktop computer equipped with an Intel Core i9-9900K processor, 32 GB of random-access memory (RAM), and an NVIDIA GeForce RTX 2080 graphics processing unit (GPU). The software environment used Ubuntu 20.04, PyTorch 1.10.1, CUDA 11.3, and cuDNN 8.2.0.

4.1 Evaluation Metrics

Open-loop steering prediction was evaluated using MSE, MAE, root mean squared error (RMSE), and the coefficient of determination (R^2). Mean absolute angular error in degrees is additionally reported to express the physical magnitude of steering deviation.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \widehat{y_i}\right)^2 \tag{11}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - \widehat{y_i}\right| \tag{12}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}\left(y_i - \widehat{y_i}\right)^2}{\sum_{i=1}^{n}\left(y_i - \overline{y_i}\right)^2} \tag{13}$$

MSE and RMSE emphasize larger deviations, MAE describes typical absolute prediction error, and R^2 measures explained variation relative to the mean target. These metrics characterize open-loop steering prediction; closed-loop behavior is examined separately in Section 4.5.

4.2 Historical Cross-Modal End-to-End Reference

NVIDIA PilotNet [28] provides the historical camera-based end-to-end reference. Because the two pipelines differ in sensing modality, preprocessing, network architecture, state information, and output scaling, this comparison is used to characterize their respective prediction behavior. Architectural contributions are evaluated with the same-modality LiDAR configurations in Section 4.4.

4.3 Open-Loop Prediction Results

LiDAR-LHD-Net was evaluated on held-out simulated tunnel data in three environments: a straight roadway, a right-angle turning roadway, and a multi-branch roadway (Fig. 7). The open-loop results reported in Table 1 characterize the steering-estimation performance of the proposed LiDAR-based local policy under these simulated conditions. The historical camera-based PilotNet implementation was evaluated separately as a cross-modal end-to-end reference, as described in Section 4.2, and is not treated as a component-matched benchmark.

The environments increase in topological complexity from a predominantly straight roadway to a right-angle turn and then a multi-branch network. The first two also include irregular wall protrusions, whereas the third introduces multiple feasible branches that require explicit route intent.

Over 100 epochs, the training and validation losses decreased and stabilized after approximately 60 epochs (Fig. 8a), while the validation MAE converged to approximately 0.04 rad (Fig. 8b). These trends indicate optimization convergence under the selected frame-level development split. Because sequential

observations from the same simulated trajectory are strongly correlated and the 80/20 validation partition is not trajectory-disjoint, the close training–validation behavior should not be interpreted as evidence of independent temporal or route-level generalization.

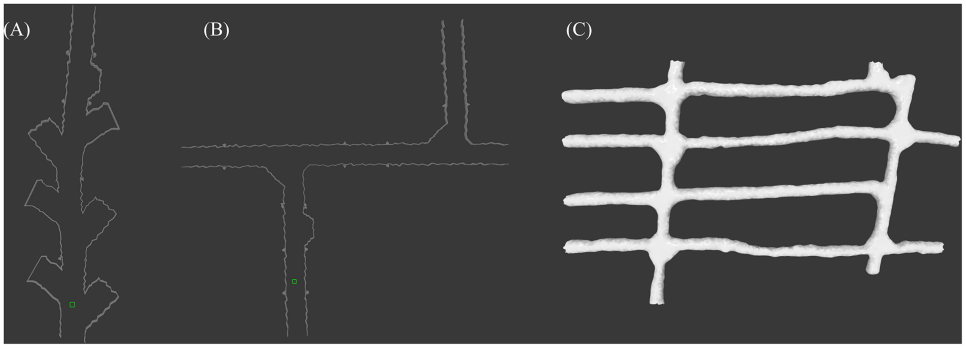
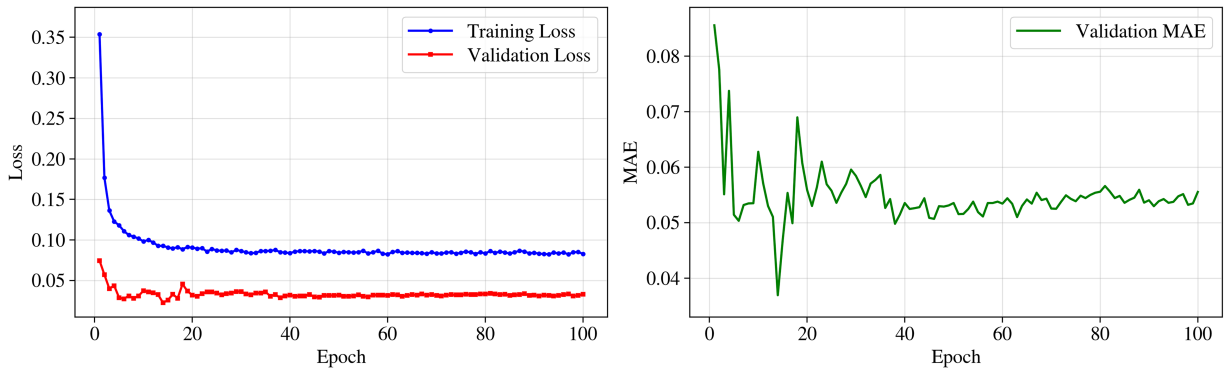


Figure 7: Simulated test environments: (**A**) Straight roadway, (**B**) Right-angle turning roadway, and (**C**) Multi-branch roadway.

Table 1: Open-loop steering-estimation results of LiDAR-LHD-Net in the three simulated tunnel environments.

Metric	Single Tunnel Scenario	Turning Lane Scenario	Multi-Branch Tunnel Scenario
MAE (rad)	0.035636	0.036647	0.036078
R^2	0.997688	0.997612	0.996760
MSE (rad^2)	0.002312	0.002388	0.003240
Mean absolute angular error (°)	2.04	2.10	2.07
RMSE (rad)	0.048085	0.048868	0.056922



(a) Loss comparison: training vs. validation

(b) Verifying the MAE

Figure 8: Training history: (**a**) Training and validation loss and (**b**) Validation MAE.

The reported training curves and open-loop prediction metrics correspond to the evaluated training run. Multiple independent training runs with different random seeds were not conducted in the present study; therefore, standard deviations or confidence intervals associated with training stochasticity are not reported. The numerical results should consequently be interpreted as point estimates under the stated simulation and data-split protocol rather than as evidence of seed-independent statistical robustness.

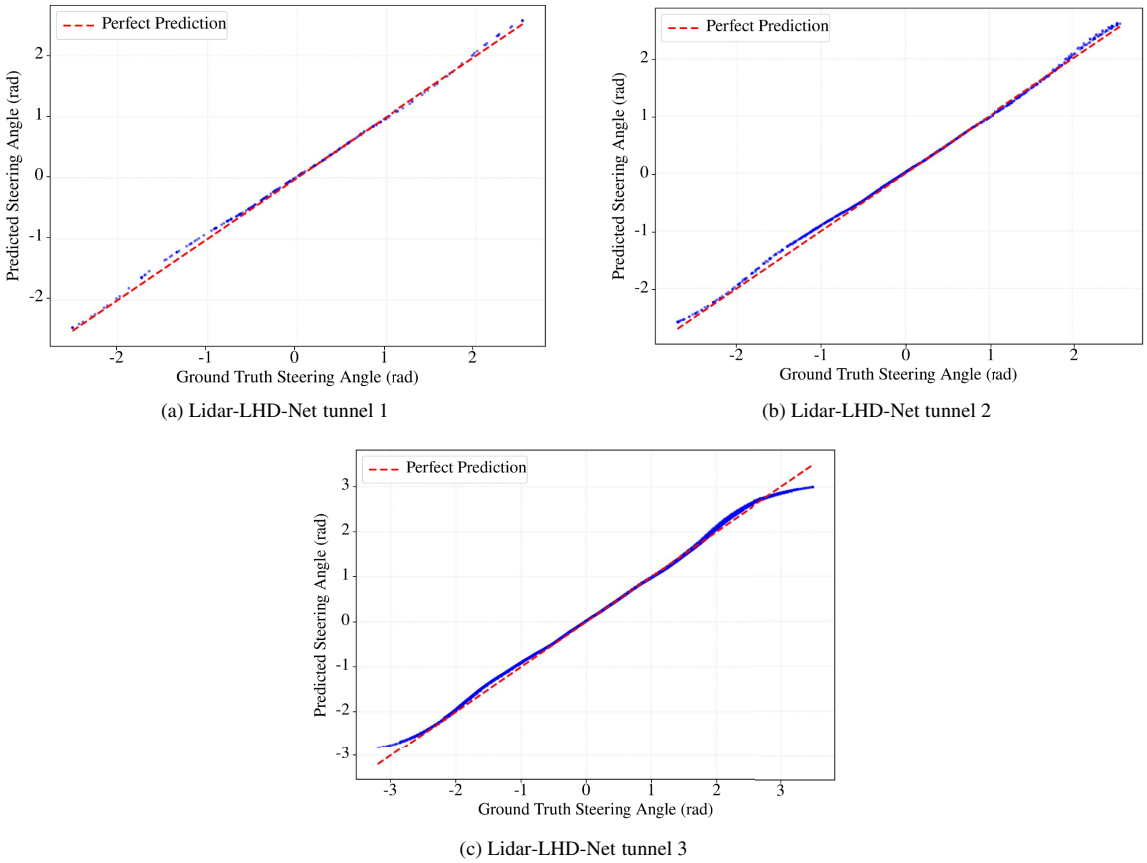
Across the three simulated test environments, LiDAR-LHD-Net achieved MAE values of 0.0356–0.0366 rad, R^2 values of 0.9968–0.9977, and mean absolute angular errors of 2.04°–2.10° (Table 1). These results characterize steering-prediction fit under the reported simulation and data-split protocol.

The historical PilotNet reference produced R^2 values of 0.8524–0.9697 and mean absolute angular errors of 6.52°–15.53°, with the largest angular deviation occurring in the right-angle turning environment (Table 2).

Table 2: Descriptive open-loop results of the historical camera-based PilotNet reference under the selected simulation protocol.

Metric	Single Tunnel Scenario	Turning Lane Scenario	Multi-Branch Tunnel Scenario
R^2	0.969708	0.902294	0.852430
Mean absolute angular error (°)	6.524321	15.533215	10.465023

Figs. 9 and 10 show the predicted-vs.-target relationships for LiDAR-LHD-Net and PilotNet, respectively. LiDAR-LHD-Net predictions remain closely distributed around the reference relationship across all three tunnel environments, consistent with the errors reported in Table 1. The PilotNet distributions show larger deviations, particularly in the right-angle turning environment, in agreement with Table 2.

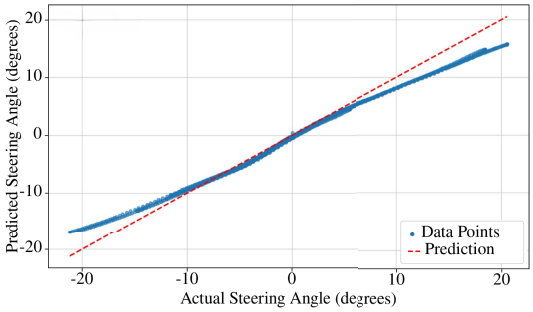


(a) Lidar-LHD-Net tunnel 1

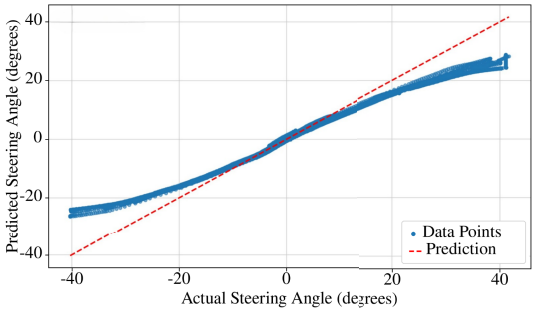
(b) Lidar-LHD-Net tunnel 2

(c) Lidar-LHD-Net tunnel 3

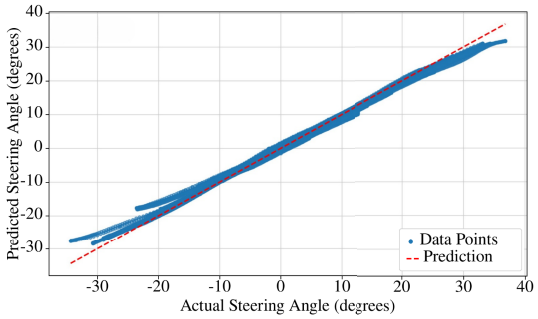
Figure 9: LiDAR-LHD-Net predicted vs. target steering for (**a**) Straight, (**b**) Right-angle, and (**c**) Multi-branch tunnel tests. The diagonal denotes ideal agreement.



(a) NVIDIA PilotNet tunnel 1



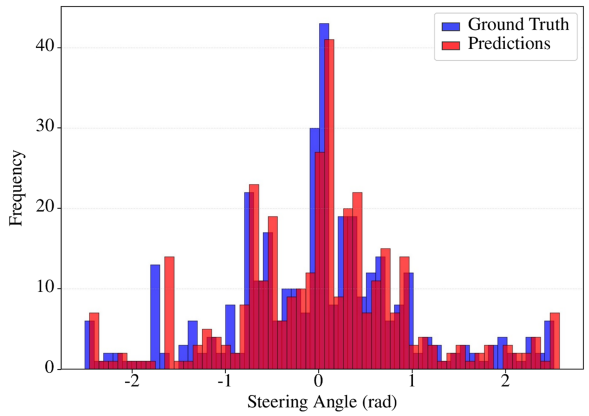
(b) NVIDIA PilotNet tunnel 2



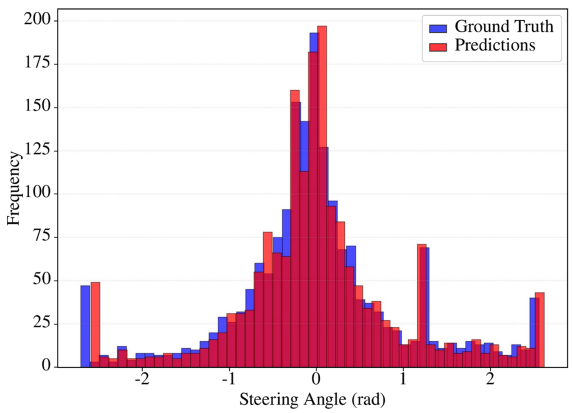
(c) NVIDIA PilotNet tunnel 3

Figure 10: PilotNet predicted vs. target steering for (**a**) Straight, (**b**) Right-angle, and (**c**) Multi-branch tunnel tests. The diagonal denotes ideal agreement.

Figs. 11 and 12 provide distribution-level comparisons of the target and predicted steering angles for LiDAR-LHD-Net and PilotNet, respectively. Fig. 11 shows the frequency distributions of the LiDAR-LHD-Net predictions and their corresponding targets, while Fig. 12 presents the corresponding smoothed distributions for PilotNet. The degree of overlap between the predicted and target distributions provides a complementary view of steering-prediction behavior across the straight, right-angle, and multi-branch tunnel environments. Together with the pointwise predicted-vs.-target relationships in Figs. 9 and 10 and the quantitative results in Tables 1 and 2, these plots characterize the open-loop prediction performance from both pointwise and distributional perspectives.

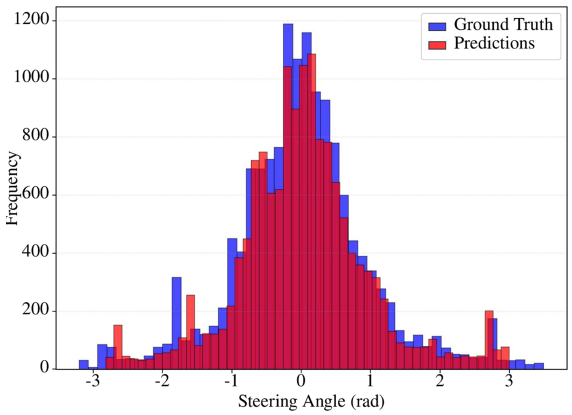


(a) Comparison of turning angle distributions in Tunnel 1 using LiDAR-LHD-Net



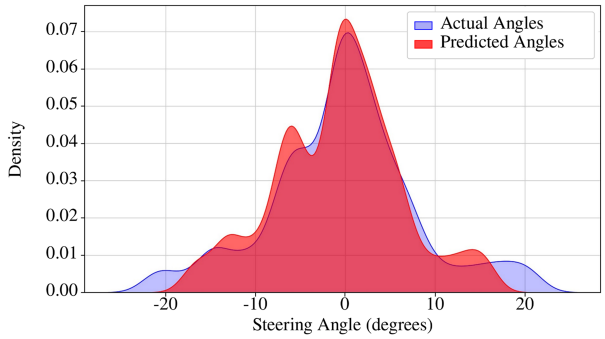
(b) Comparison of turning angle distributions in Tunnel 2 using LiDAR-LHD-Net

Figure 11: (Continued)

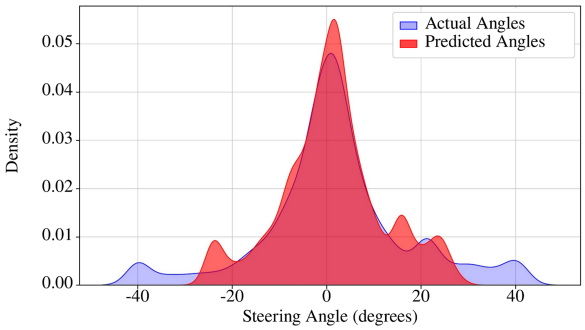


(c) Comparison of turning angle distributions in Tunnel 3 using LiDAR-LHD-Net

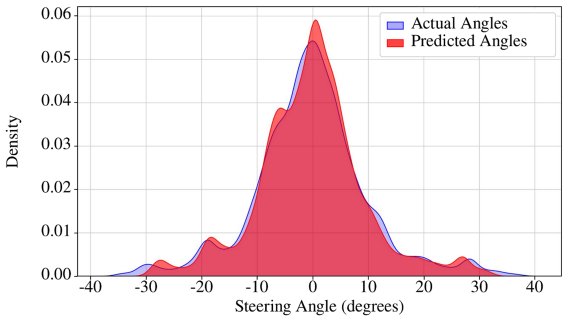
Figure 11: Histograms of the LiDAR-LHD-Net's superimposed target and predicted steering angle distributions in (**a**) Straight-line, (**b**) Right-angle, and (**c**) Multi-branch tunnel tests.



(a) Comparison of turning angle distributions in Tunnel 1 using NVIDIA PilotNet



(b) Comparison of turning angle distributions in Tunnel 2 using NVIDIA PilotNet



(c) Comparison of turning angle distributions in Tunnel 3 using NVIDIA PilotNet

Figure 12: Smoothed distributions of target and predicted steering angles for PilotNet in (**a**) Straight, (**b**) Right-angle, and (**c**) Multi-branch tunnel tests.

The distribution plots complement the aggregate metrics by showing how the predicted steering-angle range and concentration correspond to the target distributions in each tunnel environment. For LiDAR-LHD-Net, the strong overlap between predicted and target distributions is consistent with the low errors reported in Table 1. The PilotNet distributions provide the corresponding cross-modal reference and are

consistent with the larger angular errors reported in Table 2. Together with Figs. 9 and 10, these distribution-level results provide a complementary characterization of the open-loop steering-prediction behavior across the three simulated tunnel environments.

4.4 Local-Policy Network Ablation

The ablation study focuses on the learned local steering policy and examines the contributions of the point-cloud encoder, spatial self-attention, and Bi-LSTM-based feature-processing block under a common LiDAR dataset and training protocol. The complete local model is compared with a PointNet–MLP baseline and variants without spatial attention or without Bi-LSTM-based feature processing. Because these configurations use the same sensing modality and steering target, they provide the principal internal evidence for the contribution of the additional feature-processing modules. The results are shown in Fig. 13 and Table 3.

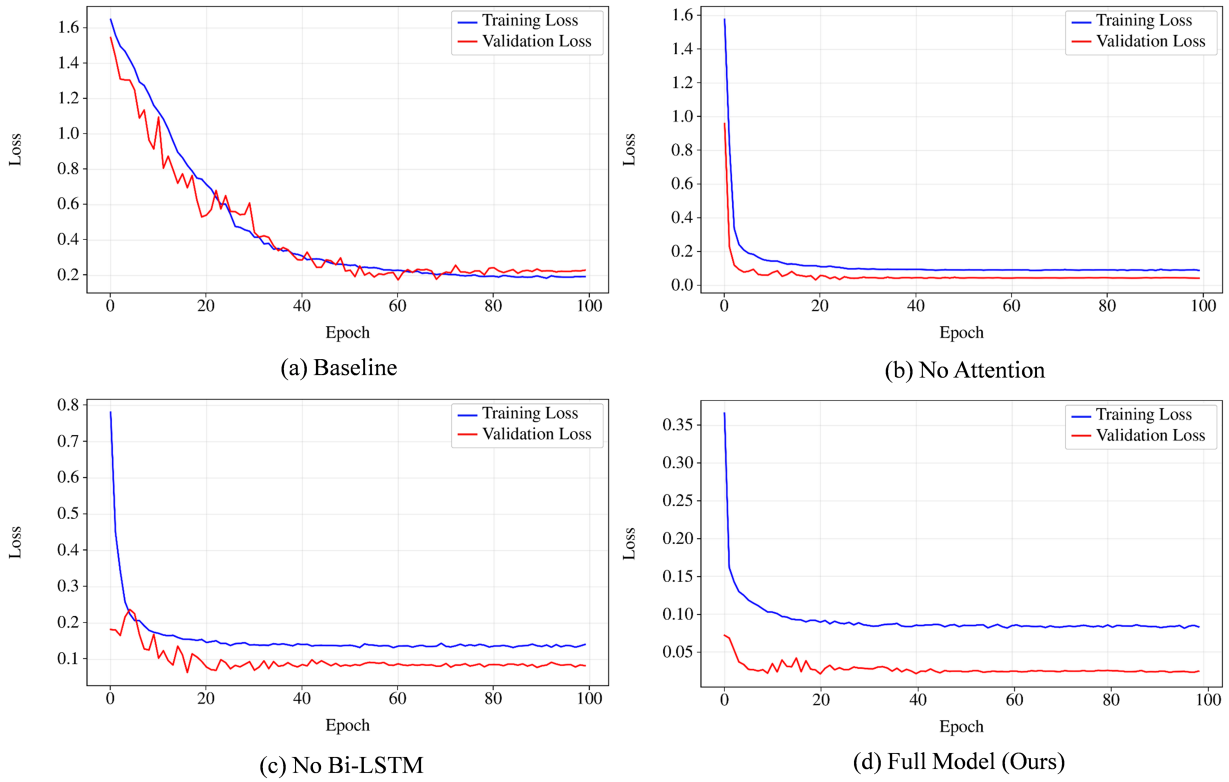


Figure 13: Training and validation loss for the local-policy network ablations: (**a**) PointNet–MLP baseline, (**b**) Without spatial attention, (**c**) Without Bi-LSTM processing, and (**d**) Complete local model.

Table 3: Accuracy and structural-complexity comparison of the learned local steering-policy configurations.

Model Variant	Architecture Description	Trainable Parameters	Best Validation Loss↓	Best Validation MAE (rad)↓
a. PointNet–MLP baseline	Basic PointNet-style encoder and MLP prediction head	24,191	0.1744	0.2410

(Continued)

Table 3 (continued)

Model Variant	Architecture Description	Trainable Parameters	Best Validation Loss↓	Best Validation MAE (rad)↓
b. Without attention	Spatial self-attention removed	109,977	0.0315	0.0512
c. Without Bi-LSTM	Bi-LSTM-based feature-processing block removed	40,575	0.0634	0.0913
d. Complete model	Multi-branch encoder, attention, and Bi-LSTM	126,361	0.0206	0.0349

Note: ↓ indicates that lower values are better.

The sparse-waypoint bearing and distance-scheduled arbitration operate outside the regression network and are examined through the closed-loop cases in Section 4.5.

The complete model contains 126,361 trainable parameters, with the additional complexity associated primarily with the attention and Bi-LSTM-based feature-processing modules.

4.5 Closed-Loop Simulation Cases

The trained local policy was integrated with sparse-waypoint guidance, distance-scheduled arbitration, and command limiting and deployed in three closed-loop tunnel scenarios. The straight roadway examines continuous wall-relative steering, the right-angle case introduces an abrupt directional transition, and the multi-branch environment additionally requires route selection at topological decision regions. Figs. 14–19 show representative closed-loop trajectories for LiDAR-LHD-Net and the PilotNet reference.

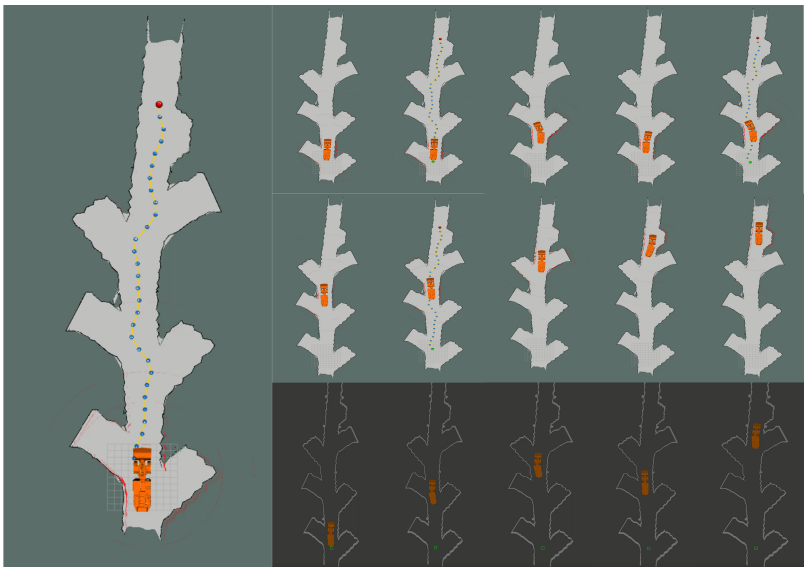


Figure 14: Representative closed-loop states in the straight-roadway case using LiDAR-LHD-Net.

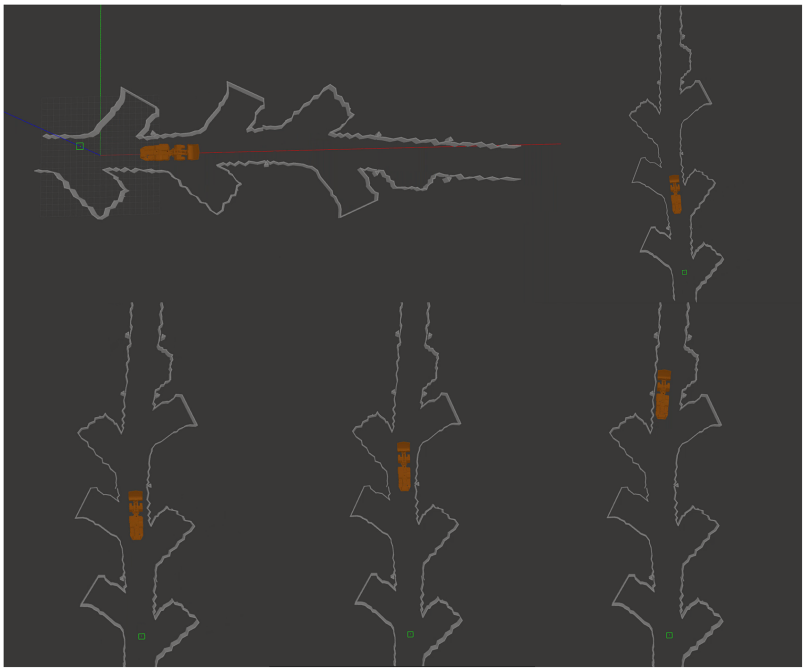


Figure 15: Representative closed-loop states in the straight-roadway case using PilotNet.

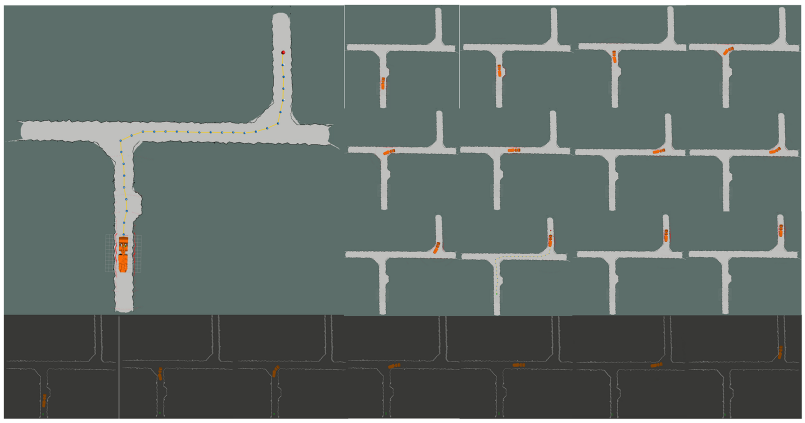


Figure 16: Representative closed-loop states in the right-angle turning case using LiDAR-LHD-Net.

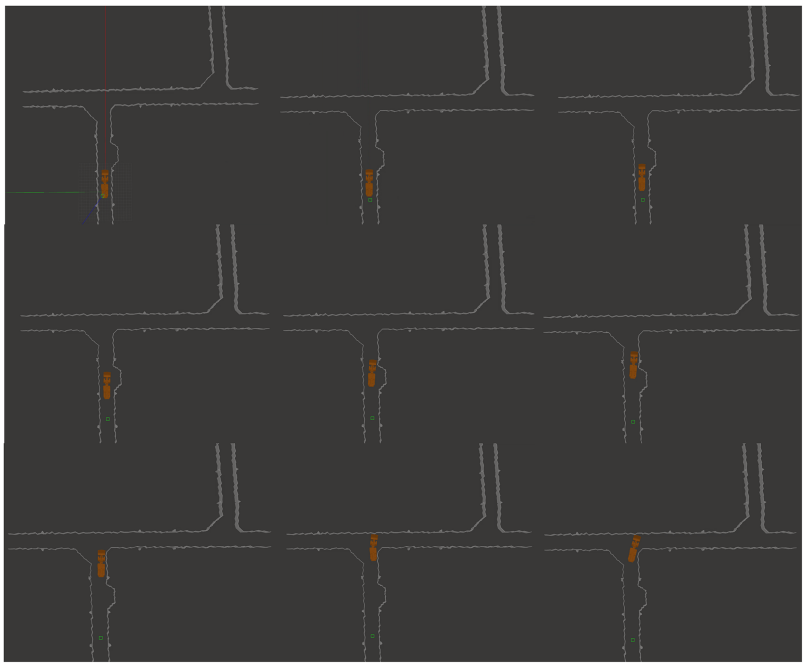


Figure 17: Representative closed-loop states in the right-angle turning case using PilotNet.

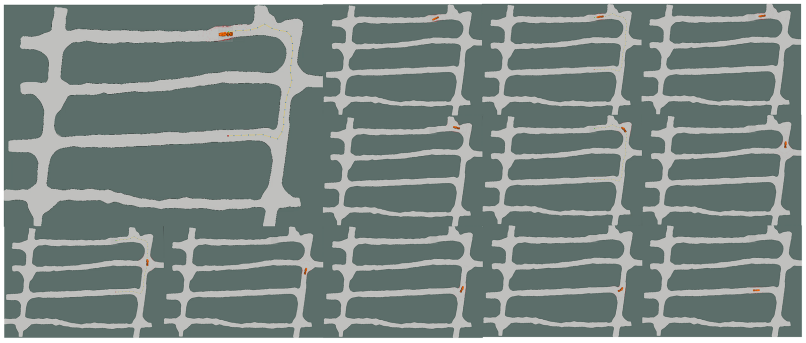


Figure 18: Representative closed-loop states in the multi-branch case using LiDAR-LHD-Net with waypoint arbitration.

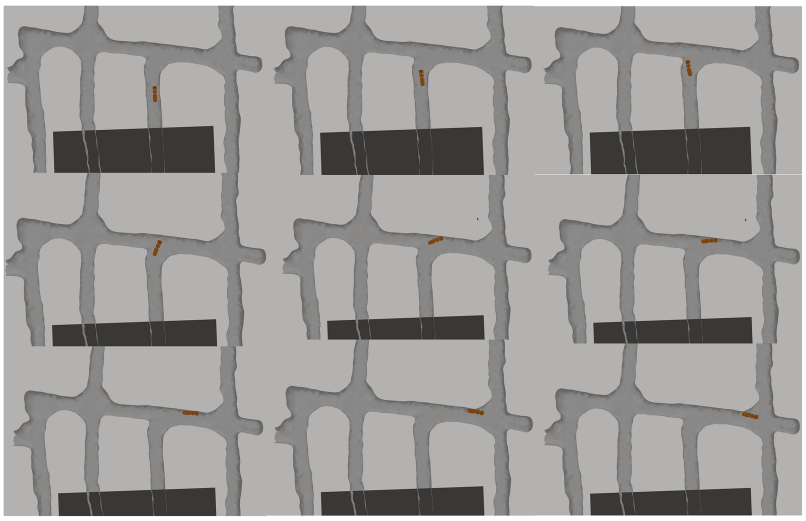


Figure 19: Representative closed-loop states in the multi-branch case using PilotNet.

(a) Straight-roadway case

The straight-roadway case contains a long main tunnel and several non-selected branches. The vehicle travelled at a constant 1.0 m/s. Figs. 14 and 15 show representative time points for LiDAR-LHD-Net and PilotNet, respectively.

In the illustrated LiDAR-LHD-Net run, the waypoint sequence maintains the intended route while the local policy adjusts steering relative to the walls. In the PilotNet run, steering varies less and the vehicle travels closer to the tunnel boundary. These observations are qualitative and apply to the depicted runs.

(b) Right-angle turning case

The turning case contains a T-shaped decision region and tests both steering anticipation and waypoint-guided branch selection (Figs. 16 and 17).

In the PilotNet run, insufficient articulation is generated before the front wall, and the vehicle becomes immobilized after contact. LiDAR-LHD-Net initiates articulation earlier and completes the right-angle turn.

(c) Multi-branch case

The multi-branch case introduces several feasible routes and therefore requires explicit topological intent in addition to local steering.

Figs. 18 and 19 compare representative runs. In the PilotNet case, the vehicle approaches the inner wall after turning, reducing clearance and complicating realignment.

In the LiDAR-LHD-Net run, the hierarchical controller selects the waypoint-directed branch and then realigns the vehicle with the local tunnel geometry. The sequence illustrates the coordinated action of local LiDAR steering, target-bearing guidance, and distance-scheduled arbitration through a multi-branch decision region.

5 Discussion

5.1 System-Level Interpretation

The system-level contribution of the proposed framework is the explicit coordination of local geometric steering, mission-dependent route intent, and articulation-command generation. Raw LiDAR geometry

drives the learned local response, while sparse waypoints provide branch information that is unavailable from locally similar tunnel observations. Distance-scheduled arbitration links these two information sources before actuation, yielding a control structure that lies between monolithic end-to-end driving and conventional localization–planning–tracking architectures.

This decomposition is particularly suited to confined underground networks, where topological ambiguity and local geometric control arise at different spatial scales. The learned policy can remain responsive to immediate tunnel boundaries, while route decisions enter through a sparse, interpretable guidance signal. As a result, mission-level branch choice can be introduced without embedding a complete global planner into the point-cloud regression network.

5.2 Computational Characteristics

The complete local policy contains 126,361 trainable parameters and relies primarily on point-wise convolutions, a 64-dimensional fused representation, spatial attention, and one Bi-LSTM-based feature-processing stage. It operated online throughout the Gazebo closed-loop experiments on the reported desktop platform. Deployment-oriented characterization will require end-to-end timing measurements covering point-cloud preprocessing, inference, controller execution, memory use, and sensing-to-actuation latency on vehicle-grade hardware.

5.3 Limitations and Simulation-to-Real Considerations

Transfer to an operational underground mine introduces sensing, localization, and vehicle-dynamics effects that are absent from the nominal Gazebo environment. LiDAR measurements may be altered by dust and water mist, missing returns, wet-surface reflections, vibration-induced calibration changes, and partial occlusion by equipment or personnel. In this context, the point-cloud augmentation used during training provides geometric regularization but does not reproduce a calibrated mine-specific sensor-degradation model.

Vehicle-level uncertainty introduces a separate set of challenges. Position and heading errors modify waypoint bearing and decision-region activation, while payload, roadway slope, wheel slip, hydraulic response, actuator delay, saturation, and tracking error alter the relationship between commanded and realized articulation. Evaluation under these disturbances is therefore a necessary step when transferring the controller from nominal simulation to an operating LHD.

The current data protocol uses an 80/20 frame-level development split, supplemented by two environment-disjoint tunnel evaluations. This setup characterizes prediction across the reported environments, whereas trajectory-disjoint testing and repeated independent training runs would provide a stronger basis for quantifying statistical variability. The closed-loop experiments likewise demonstrate integrated operation of the local policy, waypoint guidance, and arbitration mechanism; repeated matched trials and component-level closed-loop ablations are required for statistical evaluation of navigation reliability and the isolated contribution of the guidance layer.

6 Conclusions

An actuator-aware hierarchical LiDAR control framework was developed for centrally articulated underground LHD navigation. The framework couples a learned local steering policy with sparse waypoint guidance through distance-scheduled arbitration and command limiting. Raw three-dimensional LiDAR

point sets and the measured articulation state are processed by a compact network comprising a multi-branch PointNet-style encoder, residual feature enhancement, spatial self-attention, and a Bi-LSTM-based feature-processing block.

Across three simulated tunnel environments, LiDAR-LHD-Net achieved a mean absolute steering error of approximately 0.036 rad and (R^2) values above 0.996 under the reported evaluation protocol. Under the same LiDAR training protocol, the complete model produced lower validation error than the PointNet–MLP, no-attention, and no-Bi-LSTM configurations. Representative closed-loop cases further illustrated the coordinated operation of local geometric steering and sparse waypoint guidance in straight, right-angle-turning, and multi-branch tunnel layouts.

The resulting control structure separates local geometric reaction from mission-level branch intent while preserving an explicit articulation-command interface. The present results establish the simulation-level feasibility of this hierarchical coordination strategy under the tested conditions. Future work will extend the framework toward trajectory-disjoint evaluation, repeated closed-loop trials, mine-specific LiDAR degradation, localization and actuator disturbances, embedded computational profiling, and physical underground validation.

Acknowledgement: None.

Funding Statement: This work was supported by the National Natural Science Foundation of China under Grant 52374168, the Science and Technology Innovation Program of Hunan Province under Grant 2023RC3069, and the National Key Research and Development Program of China under grants 2022YFC2904105 and 2023YFC2907403.

Author Contributions: Ya Liu: methodology, investigation, software, validation, writing—original draft. Pingan Peng: conceptualization, funding acquisition, writing—review & editing. Liguan Wang: data curation, writing—review & editing. Ru Lei: software, validation. Shuangwei Gong: formal analysis, visualization. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The data that support the findings of this study are available from the Corresponding Author Pingan Peng upon reasonable request.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Liu Y, Peng PA, Wang LG, Wu JX, Lei MY, Zhang CW, et al. PSO-NMPC control strategy based path tracking control of mining LHD (scraper). Sci Rep. 2024;14(1):28516. doi:10.1038/s41598-024-79248-8.
2. Song R, Ai Y, Tian B, Chen L, Zhu F, Yao F. MSFANet: a light weight object detector based on context aggregation and attention mechanism for autonomous mining truck. IEEE Trans Intell Veh. 2023;8(3):2285–95. doi:10.1109/TIV.2022.3221767.
3. Zhu M, Gong Y, Tian C, Zhu Z. A systematic survey of transformer-based 3D object detection for autonomous driving: methods, challenges and trends. Drones. 2024;8(8):412. doi:10.3390/drones8080412.
4. Wang J, Wang L, Jiang Y, Peng P, Wu J, Liu Y. A novel global re-localization method for underground mining vehicles in haulage roadways: a case study of solid-state LiDAR-equipped load-haul-dump vehicles. Tunn Undergr Space Technol. 2025;156:106270. doi:10.1016/j.tust.2024.106270.
5. Yang B, Fan F, Ni R, Wang H, Jafaripournimchahi A, Hu H. A multi-task learning network with a collision-aware graph transformer for traffic-agents trajectory prediction. IEEE Trans Intell Transp Syst. 2024;25(7):6677–90. doi:10.1109/TITS.2023.3345296.

6. Jiang Y, Peng P, Wang L, Wang J, Wu J, Liu Y. LiDAR-based local path planning method for reactive navigation in underground mines. Remote Sens. 2023;15(2):309. doi:10.3390/rs15020309.
7. Daoud MA, Mehrez MW, Rayside D, Melek WW. Simultaneous feasible local planning and path-following control for autonomous driving. IEEE Trans Intell Transp Syst. 2022;23(9):16358–70. doi:10.1109/TITS.2022.3149986.
8. Chen L, Wu P, Chitta K, Jaeger B, Geiger A, Li H. End-to-end autonomous driving: challenges and frontiers. IEEE Trans Pattern Anal Mach Intell. 2024;46(12):10164–83. doi:10.1109/TPAMI.2024.3435937.
9. Codevilla F, Müller M, López A, Koltun V, Dosovitskiy A. End-to-end driving via conditional imitation learning. In: Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA); 2018 May 21–25; Brisbane, QLD, Australia. p. 4693–700. doi:10.1109/ICRA.2018.8460487.
10. Ly AO, Akhloufi M. Learning to drive by imitation: an overview of deep behavior cloning methods. IEEE Trans Intell Veh. 2021;6(2):195–209. doi:10.1109/TIV.2020.3002505.
11. Tampuu A, Aidla R, van Gent JA, Matiisen T. LiDAR-as-camera for end-to-end driving. Sensors. 2023;23(5):2845. doi:10.3390/s23052845.
12. Inostroza F, Parra-Tsunekawa I, Ruiz-Del-Solar J. Robust localization for underground mining vehicles: an application in a room and pillar mine. Sensors. 2023;23(19):8059. doi:10.3390/s23198059.
13. Jacobson A, Zeng F, Smith D, Boswell N, Peynot T, Milford M. What localizes beneath: a metric multisensor localization and mapping system for autonomous underground mining vehicles. J Field Robot. 2021;38(1):5–27. doi:10.1002/rob.21978.
14. Phillips TG, Guenther N, McAree PR. When the dust settles: the four behaviors of LiDAR in the presence of fine airborne particulates. J Field Robot. 2017;34(5):985–1009. doi:10.1002/rob.21701.
15. Montalban K, Reymann C, Atchuthan D, Dupouy PE, Riviere N, Lacroix S. A quantitative analysis of point clouds from automotive lidars exposed to artificial rain and fog. Atmosphere. 2021;12(6):738. doi:10.3390/atmos12060738.
16. Tian Y, Li X, Zhang H, Zhao C, Li B, Wang X, et al. VistaGPT: generative parallel transformers for vehicles with intelligent systems for transport automation. IEEE Trans Intell Veh. 2023;8(9):4198–207. doi:10.1109/TIV.2023.3307012.
17. Xia X, Bhatt NP, Khajepour A, Hashemi E. Integrated inertial-LiDAR-based map matching localization for varying environments. IEEE Trans Intell Veh. 2023;8(10):4307–18. doi:10.1109/TIV.2023.3298892.
18. Hughes AJ, Bull LA, Gardner P, Dervilis N, Worden K. On robust risk-based active-learning algorithms for enhanced decision support. Mech Syst Signal Process. 2022;181:109502. doi:10.1016/j.ymssp.2022.109502.
19. Hu X, Chen L, Tang B, Cao D, He H. Dynamic path planning for autonomous driving on various roads with avoidance of static and moving obstacles. Mech Syst Signal Process. 2018;100:482–500. doi:10.1016/j.ymssp.2017.07.019.
20. Zhu H, Ouyang H, Xi H. Neural network-based time optimal trajectory planning method for rotary cranes with obstacle avoidance. Mech Syst Signal Process. 2023;185:109777. doi:10.1016/j.ymssp.2022.109777.
21. Qiu ZC, Chen GH, Zhang XM. Trajectory planning and vibration control of translation flexible hinged plate based on optimization and reinforcement learning algorithm. Mech Syst Signal Process. 2022;179:109362. doi:10.1016/j.ymssp.2022.109362.
22. Zhang X, Guo A, Ai Y, Tian B, Chen L. Real-time scheduling of autonomous mining trucks via flow allocation-accelerated tabu search. IEEE Trans Intell Veh. 2022;7(3):466–79. doi:10.1109/TIV.2022.3166564.
23. Li B, Ouyang Y, Li X, Cao D, Zhang T, Wang Y. Mixed-integer and conditional trajectory planning for an autonomous mining truck in loading/dumping scenarios: a global optimization approach. IEEE Trans Intell Veh. 2023;8(2):1512–22. doi:10.1109/TIV.2022.3214777.
24. Teng S, Hu X, Deng P, Li B, Li Y, Ai Y, et al. Motion planning for autonomous driving: the state of the art and future perspectives. IEEE Trans Intell Veh. 2023;8(6):3692–711. doi:10.1109/TIV.2023.3274536.
25. Xu H, Gao Y, Yu F, Darrell T. End-to-end learning of driving models from large-scale video datasets. In: Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2017 Jul 21–26; Honolulu, HI, USA. p. 3530–8. doi:10.1109/cvpr.2017.376.

26. Kim J, Canny J. Interpretable learning for self-driving cars by visualizing causal attention. In: Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV); 2017 Oct 22–29; Venice, Italy. p. 2961–9. doi:10.1109/iccv.2017.320.
27. Yang L, Liang X, Wang T, Xing E. Real-to-virtual domain unification for end-to-end autonomous driving. In: Proceedings of the 15th European Conference on Computer Vision (ECCV); 2018 Sep 8–14; Munich, Germany. doi:10.1007/978-3-030-01225-0_33.
28. Bojarski M, Del Testa D, Dworakowski D, Firner B, Flepp B, Goyal P, et al. End to end learning for self-driving cars. arXiv:1604.07316. 2016.
29. Li Q, Chen L, Li M, Shaw SL, Nüchter A. A sensor-fusion drivable-region and lane-detection system for autonomous vehicle navigation in challenging road scenarios. IEEE Trans Veh Technol. 2014;63(2):540–55. doi:10.1109/TVT.2013.2281199.
30. Charles RQ, Hao S, Mo K, Guibas LJ. PointNet: deep learning on point sets for 3D classification and segmentation. In: Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2017 Jul 21–26; Honolulu, HI, USA. p. 77–85. doi:10.1109/cvpr.2017.16.
31. Qi CR, Yi L, Su H, Guibas LJ. PointNet++: deep hierarchical feature learning on point sets in a metric space. In: Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17); 2017 Dec 4–9; Long Beach, CA, USA. p. 5105–14.
32. Feng Y. A study on explainable end-to-end autonomous driving [dissertation]. Hong Kong, China: The Hong Kong Polytechnic University; 2025.
33. Ganesan M, Kandhasamy S, Chokkalingam B, Mihet-Popa L. A comprehensive review on deep learning-based motion planning and end-to-end learning for self-driving vehicle. IEEE Access. 2024;12:66031–67. doi:10.1109/ACCESS.2024.3394869.
34. Zhao R, Chen Z, Fan Y, Gao F, Men Y. BETAV: a unified BEV-transformer and Bézier optimization framework for jointly optimized end-to-end autonomous driving. Sensors. 2025;25(11):3336. doi:10.3390/s25113336.
35. Wu P, Jia X, Chen L, Yan J, Li H, Qiao Y. Trajectory-guided control prediction for end-to-end autonomous driving: a simple yet strong baseline. In: Advances in Neural Information Processing Systems 35; 2022 Nov 28–Dec 9; New Orleans, LA, USA. p. 6119–32. doi:10.52202/068431-0443.
36. Prakash A, Chitta K, Geiger A. Multi-modal fusion transformer for end-to-end autonomous driving. In: Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2021 Jun 20–25; Nashville, TN, USA. p. 7073–83. doi:10.1109/CVPR46437.2021.00700.
37. Kang Q, Hartmannsgruber A, Tan SH, Zhang X, Chew CM. Deep reinforcement learning based tractor-trailer tracking control. In: Proceedings of the 2024 IEEE 27th International Conference on Intelligent Transportation Systems (ITSC); 2024 Sep 24–27; Edmonton, AB, Canada. p. 3147–53. doi:10.1109/ITSC58415.2024.10919887.
38. Backman S, Lindmark D, Bodin K, Servin M, Mörk J, Löfgren H. Continuous control of an underground loader using deep reinforcement learning. Machines. 2021;9(10):216. doi:10.3390/machines9100216.
39. Salas R, Leiva F, Ruiz-del-Solar J. Autonomous loading of ore piles with Load-Haul-Dump machines using deep reinforcement learning. Expert Syst Appl. 2025;264:125770. doi:10.1016/j.eswa.2024.125770.
40. Mascaró M, Parra-Tsunekawa I, Tampier C, Ruiz-del-Solar J. Topological navigation and localization in tunnels—application to autonomous load-haul-dump vehicles operating in underground mines. Appl Sci. 2021;11(14):6547. doi:10.3390/app11146547.
41. Jiang Y, Peng P, Huo X, Wang J, Wang L. Intelligent autonomy: a novel hybrid navigation system for autonomous load-haul-dump vehicles. J Field Robot. 2026;43(6):3794–817. doi:10.1002/rob.70214.
42. Pomerleau DA. ALVINN: an autonomous land vehicle in a neural network. In: Proceedings of the 2nd International Conference on Neural Information Processing Systems (NIPS'88); 1988 Nov 28–Dec 1; Denver, CO, USA. p. 305–13.