applied sciences

MDPI

Article

Compressing Transformer-Based Sensor Fusion Models for Autonomous Driving Using Curriculum-Based Multi-Task Knowledge Distillation

Dhieddine Barhoumi [1,2], Stefan Hensel [1] and Marin B. Marinov [3,*]

[1] Institute for Machine Learning and Analytics (IMLA), University of Applied Sciences Offenburg, 77652 Offenburg, Germany; dhieddine.barhoumi@hs-offenburg.de or dhieddine.barhoumi@insat.ucar.tn (D.B.); stefan.hensel@hs-offenburg.de (S.H.)
[2] National Institute of Applied Sciences and Technology, University of Carthage, Tunis 1080, Tunisia
[3] Department of Electronics, Technical University of Sofia, 1756 Sofia, Bulgaria
* Correspondence: mbm@tu-sofia.bg

Abstract

Multimodal sensor fusion architectures such as TransFuser achieve strong waypoint-based driving performance by fusing RGB panoramas and LiDAR bird's-eye-view (BEV) representations through transformer attention, but their dual backbones and quadratic-cost fusion blocks preclude deployment on resource-constrained platforms. This work proposes LightFuser, a hybrid Performer–Transformer fusion architecture that retains multimodal reasoning at substantially reduced cost. LightFuser halves backbone parameters and FLOPs, applies Performer attention with FAVOR+ at the early, high-resolution fusion stage to obtain linear time and memory complexity, and reserves exact Transformer attention for the mid and late stages, where token counts are small and semantics are richer. A multi-task knowledge distillation framework transfers the decision quality of a high-capacity TransFuser teacher to the lightweight student, organized as a task-ordered curriculum: supervision starts with waypoint control and successively adds depth, semantic segmentation, and BEV objectives. On the CARLA Leaderboard, LightFuser attains a Driving Score of 42.4 (teacher: 47.3) while reducing network latency from 128.4 ms to 67.6 ms per frame. Latency is measured over network modules only and excludes simulator stepping, sensor acquisition, preprocessing, and I/O overhead; the resulting 14.8 FPS sustains a 10 Hz control cycle, meaning near-real-time operation throughout.

Keywords: autonomous driving; sensor fusion; LightFuser; Transformer; Performer; knowledge distillation; curriculum learning; LiDAR; RGB; CARLA; real-time inference

Academic Editor: Stefano Quer

Received: 14 July 2026
Revised: 9 September 2026
Accepted: 15 September 2026
Published: 20 September 2026

1. Introduction

Recent transformer-based architectures for end-to-end autonomous driving, such as TransFuser, demonstrate strong performance in multimodal perception and waypoint prediction within simulated environments. By integrating panoramic RGB images and LiDAR BEV representations via multi-scale attention, TransFuser captures long-range dependencies and cross-modal interactions critical for navigating dense urban scenes. Nevertheless, several limitations affect its scalability, efficiency, and real-world applicability [1,2].

TransFuser relies on dual RegNetY-3.2GF backbones and multiple transformer-based fusion blocks whose self-attention scales quadratically with the number of tokens, making high-resolution feature fusion particularly expensive [3]. The model has roughly

165.8 million parameters, performs 34.9 G multiply–accumulate operations, and consumes about 2.8 GB of GPU memory during inference [4]. Beyond raw compute, it requires strict temporal alignment of the camera and LiDAR streams. It fuses individual frames without temporal integration, reducing robustness in dynamic scenes where temporal context helps handle occlusions, sensor noise, and rapidly changing traffic [2,5,6].

These limitations expose the trade-off at the core of TransFuser's design: rich scene understanding through hierarchical fusion comes at the cost of computational complexity and operational fragility.

Building on this diagnosis, the present work introduces LightFuser, a hybrid Performer–Transformer fusion architecture designed to overcome the performance constraints observed in TransFuser. Four main principles guide the design. First, both RGB and LiDAR streams are processed by lighter backbones, reducing model complexity and enabling faster inference than the baseline's heavier networks. Specifically, RegNetY-3.2GF encoders are replaced with RegNetY-1.6GF encoders, halving the parameter count and per-stream computational cost [7].

Second, multi-scale fusion integrates modality-specific features at the early, mid, and late stages of the backbone. Because standard attention is prohibitively expensive on high-resolution feature maps, the early stage uses Performer attention, whose FAVOR+ kernel approximation reduces self-attention's quadratic complexity to linear time and memory [3,8]. Exploratory experiments with a Performer-only pipeline, however, showed that the approximation sacrifices the fine-grained semantic–geometric correspondences needed for downstream control; full Transformer attention is therefore retained at the mid and late stages, where token counts are small enough for exact attention to be affordable.

Third, independent decoders for control, perception, and detection ensure task modularity. Finally, a comprehensive multi-task knowledge distillation framework serves as the central design principle: a high-capacity TransFuser teacher supervises the student at both the feature and task levels, with distillation weights increased progressively during the early epochs to ensure stable convergence [9,10].

The study is guided by two research questions (RQ) that structure both the design of the architecture and its evaluation:

- RQ1: Can a transformer-based multimodal fusion model be compressed to near-real-time performance without catastrophic loss of driving competence?
- RQ2: What training regime makes lightweight multi-task student networks converge stably?

The first question targets the architectural side of the accuracy–efficiency trade-off: whether the components responsible for TransFuser's driving competence can be retained while its dominant cost centers—heavy dual backbones and quadratic attention at high resolution—are removed or replaced. The second question addresses the optimization side: students trained on many simultaneous objectives with reduced capacity are prone to non-convergence or oscillation, so the schedule for introducing supervised and distillation signals becomes a first-class design decision rather than an implementation detail. The curriculum-based knowledge distillation regime developed in response to this question constitutes the core contribution of this article.

The main contributions of this work can be summarized as follows:

- A component-level computational analysis of the TransFuser baseline that identifies the dual RegNetY-3.2GF encoders and high-resolution self-attention as the dominant latency and memory bottlenecks;
- LightFuser, a compact dual-stream architecture that combines RegNetY-1.6GF encoders with a scale-aware hybrid Performer–Transformer fusion scheme;

- A curriculum-based multi-task knowledge distillation regime that introduces teacher supervision in a task-ordered, progressively weighted schedule, which the ablation shows to be necessary rather than merely helpful for the convergence of the compact student when set against no distillation and against undifferentiated distillation from the first epoch;
- A closed-loop evaluation on the CARLA Leaderboard, complemented by component-level latency profiling and an ablation of distillation regimes, that quantifies the accuracy–efficiency trade-off.

The remainder of this article is organized as follows. Section 2 reviews related work on multimodal fusion, efficient attention, and knowledge distillation. Section 3 analyzes the computational bottlenecks of the TransFuser baseline and presents the LightFuser architecture. Section 4 develops the curriculum knowledge distillation framework. Section 5 describes the dataset, training setup, and evaluation methodology, and Section 6 reports driving-performance, distillation, and efficiency results. Sections 7–9 discuss the findings with respect to the research questions, state the limitations, and conclude.

Taken together, the principal novelty of this work lies neither in the hybrid fusion scheme nor in the distillation curriculum in isolation, but in their integration: the scale-aware fusion design is what makes the compression affordable, and the task-ordered curriculum is what makes the compressed student trainable at all. Of the two, the curriculum is the load-bearing element, since Section 6.2 shows that the reduced architecture does not reach a usable driving policy without it.

2. Related Work

2.1. Multimodal Sensor Fusion for End-to-End Driving

End-to-end autonomous driving has increasingly shifted toward multimodal fusion architectures that jointly exploit complementary sensor streams, such as RGB panoramas and LiDAR bird's-eye-view (BEV) representations. Among these architectures, TransFuser has emerged as a strong baseline for closed-loop control in simulated urban environments, thanks to its ability to integrate image and LiDAR features through transformer-based attention mechanisms [2,11].

TransFuser employs dual RegNetY-3.2GF encoders, one per modality. It combines their multi-scale feature hierarchies via a stack of transformer fusion blocks, capturing long-range dependencies and rich cross-modal relationships that yield robust waypoint prediction. The price is computational: the heavy backbones and repeated attention operations restrict deployment on real-time or resource-constrained platforms and make large-scale training time-consuming on commodity hardware [2,7]. The model further requires strict temporal synchronization of its sensor streams and fuses single frames without explicit temporal modeling, which limits robustness in highly dynamic scenes [5,6].

TransFuser is not the only representative of learned RGB–LiDAR fusion. Continuous fusion approaches, such as the ContFuse project, fuse image features into the bird's-eye-view plane via continuous convolutions [12]. Meanwhile, BEVFusion unifies camera and LiDAR streams into a shared BEV representation that supports multiple perception tasks [13]. In end-to-end driving, privileged-agent frameworks such as Learning by Cheating decouple perception from control by first training an expert with access to ground-truth state and then imitating it with a sensorimotor student [14]. These lines of work consistently show that attention-guided or BEV-centric interaction between visual semantics and LiDAR geometry improves driving-relevant perception; equally consistently, they rely on deep backbones and repeated cross-modal operations whose cost grows with feature resolution. Efficiency-oriented redesign of such fusion architectures, the focus of this article, has received comparatively little attention.

Driving performance on the CARLA benchmark has advanced considerably since TransFuser was introduced. InterFuser places an interpretable safety layer over multi-view fusion [15], ReasonNet augments fusion with temporal and global reasoning [16], TCP shows that a carefully regularized trajectory-guided controller is a strong baseline in its own right [17], and TransFuser++ demonstrates that a substantial part of the remaining headroom lies in training-time biases rather than in architecture [18]. Planning-oriented and vectorized designs such as UniAD [19] and VAD [20] go further and restructure the entire perception–prediction–planning stack. This article deliberately does not compete with those systems on absolute Driving Score. Its subject is the accuracy–efficiency trade-off itself, and TransFuser is used precisely because it is a well-documented, openly available, and widely reproduced fusion baseline: that makes the compression result interpretable as a controlled teacher–student comparison rather than as a position in a benchmark ranking. The two mechanisms studied here—resolution-dependent attention budgeting and task-ordered distillation—are properties of multi-scale fusion stacks in general, and the more recent architectures cited above share that structure.

2.2. Efficient Attention Mechanisms

These constraints have motivated research on more efficient attention mechanisms and fusion strategies. Standard multi-head self-attention, as used in Transformers, scales quadratically with the number of tokens, making it expensive for high-resolution feature maps typical of early backbone stages. To address this, researchers have proposed efficient attention variants such as Performer. Performer replaces exact softmax attention with a kernel-based approximation, FAVOR+, that re-expresses attention as a series of inner products in a randomized feature space. This reformulation reduces time and memory complexity from quadratic to linear in sequence length, enabling attention over large token sets without exhausting computational resources. In the context of multimodal fusion, Performer-style attention is particularly attractive at early stages, when spatial feature maps are dense and token counts are high [3,8].

However, empirical studies suggest that efficiency alone is not sufficient. While Performer-based fusion excels at reducing cost, a pure Performer pipeline can struggle to capture fine-grained semantic–geometric correspondences, particularly in the mid and late-backbone stages, where representations are more abstract yet decision-critical. In these deeper layers, the number of tokens is smaller, and the computational cost of standard Transformer attention becomes more manageable. In contrast, its exact attention formulation often yields more precise cross-modal reasoning. This has led to hybrid designs that combine Performer attention at early, high-resolution stages with full Transformer attention at later, semantically richer stages [3,8]. Beyond attention approximation, the broader literature on efficient deep learning offers complementary levers, e.g., compact network design spaces [7], structured pruning, quantization, and operator-level optimization [21], and systematic surveys of efficient transformer variants document the accuracy–cost trade-offs of linearized attention in detail [22]. LightFuser draws on this body of work by treating attention complexity as a resolution-dependent budget rather than a global architectural constant.

2.3. Knowledge Distillation and Curriculum Learning

In parallel, knowledge distillation has become a central technique for compressing large models: a smaller student is trained not only on ground-truth labels but also to mimic the intermediate representations and final predictions of a high-capacity teacher, inheriting much of its representational richness at a fraction of the inference cost [9]. In autonomous driving, such supervision can act at the feature level, aligning intermediate activations across scales and modalities, and at the task level, distilling predictions for waypoints,

occupancy, semantics, and depth. This enables competitive end-to-end driving models under real-time and memory constraints [9,10].

Two further strands of the distillation literature are directly relevant to this work. First, feature-level distillation in the tradition of FitNets shows that aligning intermediate representations—not only output logits—substantially improves the trainability of thin students, because the teacher's hidden layers act as hints that shape the student's internal feature hierarchy [23]; comprehensive surveys categorize such response-, feature-, and relation-based transfer schemes and their scheduling variants [24]. Second, the difficulty of balancing many simultaneous objectives is well documented in multi-task learning, where naive uniform weighting can let one task dominate the gradient signal; principled remedies range from uncertainty-based loss weighting [25] to curriculum learning, which orders training signals from simple to complex to improve convergence and the quality of the reached optima [10]. What has remained largely unexplored is the combination of these ideas in the multimodal end-to-end driving setting: a reduced-capacity student that must simultaneously absorb control, semantic, geometric, and occupancy knowledge from a large fusion teacher. The curriculum knowledge distillation framework presented in Section 4 precisely addresses this gap, treating the temporal ordering of distillation signals as the decisive stabilizing mechanism.

3. Methodology

3.1. Computational Bottleneck Analysis of the TransFuser Baseline

The design of LightFuser is profiling-driven: before any architectural decision was made, the TransFuser baseline was instrumented, and its inference pipeline was decomposed into an image backbone, a LiDAR backbone, fusion, a decoder, and an output head. On the reference platform used throughout this study (NVIDIA RTX 2080, 8 GB; NVIDIA Corporation, Santa Clara, CA, USA), the baseline spends a median of 46.0 ms per frame in the image backbone, 44.9 ms in the LiDAR backbone, and 34.1 ms in the attention-based fusion stage, for a total network latency of 128.4 ms per frame—approximately 7.8 frames per second. The decoder and output head together contribute less than 3 ms. Three observations follow directly from this profile.

First, the two modality-specific encoders dominate the budget: the dual RegNetY-3.2GF backbones account for roughly 71% of network latency and for 38.8 million of the model's approximately 165.8 million parameters. Any compression strategy that leaves the encoders untouched can therefore recover at most a modest fraction of the total cost. Second, the fusion stage—although smaller in absolute terms—scales worst: its self-attention operations grow quadratically with token count, so the early, high-resolution fusion blocks are disproportionately expensive relative to the semantic value they add. Third, the remaining components are already negligible, meaning decoder-side optimizations would be a waste of effort. This analysis fixes the two levers exploited in the remainder of this section—lighter encoders and resolution-aware attention—and explains why compression must be paired with a knowledge-transfer mechanism: the encoders that must shrink are precisely the components that carry the representational capacity responsible for TransFuser's driving competence.

3.2. Architecture Overview

LightFuser follows the dual-stream layout of TransFuser, with separate branches for RGB panoramas and LiDAR BEV representations, but replaces the computationally intensive components with more efficient alternatives [1,2]. Synchronized inputs are encoded by two parallel RegNetY-1.6GF backbones that produce multi-scale feature pyramids. At selected depths, hybrid fusion modules integrate the two streams: Performer attention

at the high-resolution early stage and standard Transformer attention at the mid and late stages, where spatial resolution is reduced and semantics are richer [3,8]. The fused representation is routed to task-specific decoders—convolutional decoders for semantic segmentation, depth estimation, and BEV map reconstruction, and a control decoder that combines multi-layer perceptrons (MLPs) with a gated recurrent unit (GRU) to generate future waypoints conditioned on the navigation goal [26].

During training, LightFuser acts as a student supervised by a high-capacity TransFuser teacher whose intermediate features and predictions are aligned with the student through a structured set of distillation losses (Section 4). Every component thus targets one of the bottlenecks identified in Section 3.1: lighter encoders reduce feature-extraction cost, resolution-aware attention limits fusion overhead, modular decoders allow targeted optimization for each task, and distillation bridges the capacity gap to the teacher model. The overall architecture is illustrated in Figure 1.

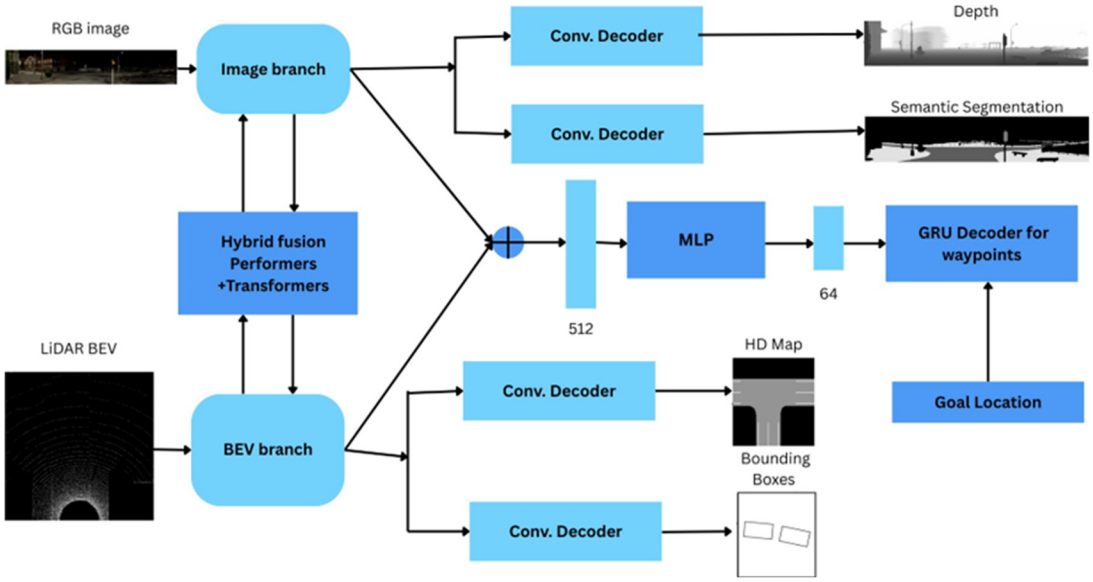

Figure 1. LightFuser architecture at inference time. The diagram shows the deployed student network only. In a later Section, Figure 3 shows the training-time teacher–student supervision paths, including where each distillation signal is injected.

3.3. Backbone Design

A key design decision in LightFuser is to use RegNetY-1.6GF as the backbone for both RGB and LiDAR streams. In contrast to the RegNetY-3.2GF backbones used in TransFuser, RegNetY-1.6GF models are significantly lighter yet retain strong feature-extraction capabilities [7].

Quantitatively, each RegNetY-3.2GF encoder contains around 19.4 million parameters and requires approximately 3.2 GFLOPs per forward pass. With two such backbones (one per modality), the encoder portion of TransFuser has about 38.8 million parameters and requires 6.4 GFLOPs. By comparison, RegNetY-1.6GF contains approximately 11.2 million parameters and 1.6 GFLOPs per stream. Using two of these reduces the total backbone parameters to about 22.4 million, a 42% decrease, and halves the per-stream computational cost [2,7].

These savings free up computational and memory headroom for the hybrid fusion modules and the multiple task-specific decoders, translating directly into lower inference latency and reduced GPU memory usage. Despite the reduction, RegNetY-1.6GF retains sufficient representational capacity for the downstream tasks, particularly when reinforced by knowledge distillation from a stronger teacher. Table 1 summarizes the comparison.

Table 1. Comparison of backbone configurations used in TransFuser and LightFuser.

Backbone	Parameters per Stream	Total Parameters (2 Streams)	Approx. FLOPs	Relative Memory Usage
RegNetY-3.2GF	19.4 M	38.8 M	3.2 GF	High
RegNetY-1.6GF	11.2 M	22.4 M	1.6 GF	Moderate

3.4. Hybrid Fusion Modules

The fusion mechanism balances computational efficiency against cross-modal expressiveness by adapting the attention type to the spatial scale. At the early stage, feature maps are large and token counts high, so exact self-attention would be prohibitively expensive; LightFuser instead applies Performer attention, whose FAVOR+ approximation keeps time and memory linear in the sequence length [3]. At the mid and late stages, resolution has decreased sufficiently for standard multi-head attention to be affordable. Its exact computation provides the more precise cross-modal reasoning required for object interaction, lane-level navigation, and the control and BEV heads that consume the deepest, semantically richest features [8].

Each fusion module follows the same pattern: modality-specific features are projected into a shared embedding space, tokenized and concatenated, processed by the stage-specific attention mechanism, and reshaped back into spatial maps, which are reinjected into both streams through residual connections.

Empirically, a pure Transformer pipeline exceeds acceptable compute and memory limits at the early stage, while a pure Performer pipeline loses the most nuanced relationships at deeper levels; the hybrid configuration is faster than the former and more accurate than the latter. The multi-scale fusion workflow is summarized in Figure 2.

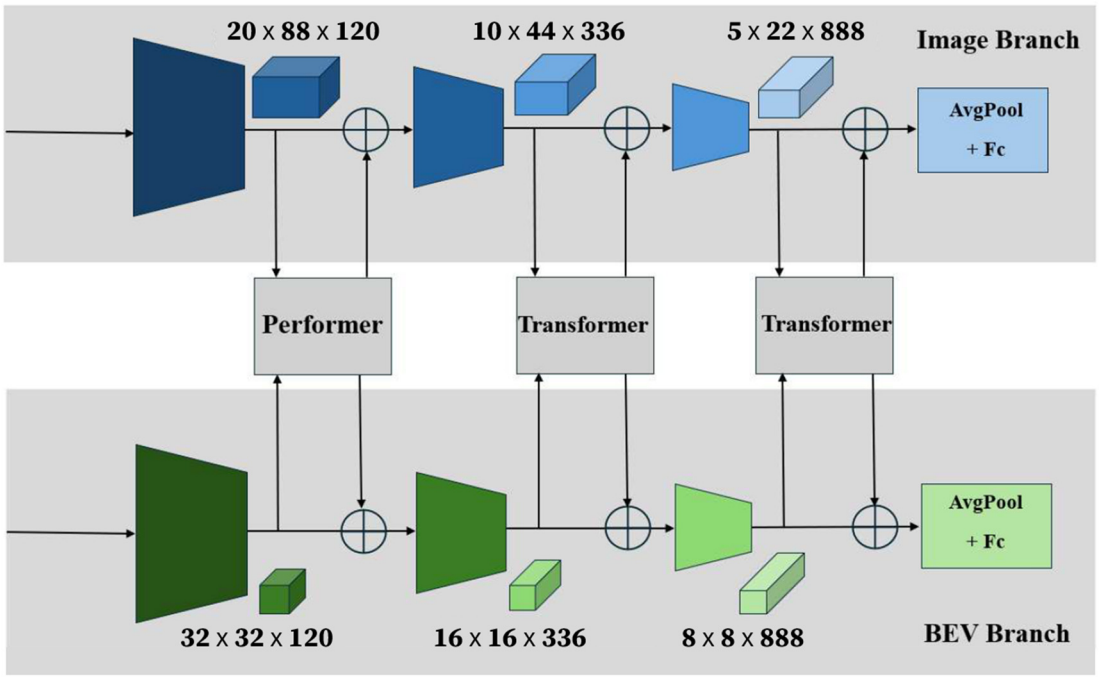

Figure 2. Multi-scale hybrid fusion flow.

Figure 2 annotates each fusion stage with the shape of the feature maps entering it, written as height × width × channels. The three fusion points correspond to the second, third, and fourth stages of the RegNetY-1.6GF encoders, whose channel widths are 120, 336, and 888. For the 160 × 704 RGB panorama the image branch therefore contributes maps

of 20 × 88 × 120, 10 × 44 × 336, and 5 × 22 × 888; for the 256 × 256 BEV grid the LiDAR branch contributes 32 × 32 × 120, 16 × 16 × 336, and 8 × 8 × 888. The token count per modality accordingly falls from 1760 to 110 in the image branch and from 1024 to 64 in the BEV branch across the three stages. This is precisely why exact attention is affordable at the mid and late stages but not at the early one: the quadratic term is evaluated on 110 and 64 tokens rather than on 1760 and 1024.

3.5. Input Representations

LightFuser retains TransFuser's input interface, so that performance differences are attributable to the architecture rather than the data. Three synchronized camera views are stitched and resized into 160 × 704-pixel RGB panoramas, and LiDAR sweeps are projected into 256 × 256 BEV grids encoding occupancy and height, with increased vertical resolution in the height channel to better separate obstacles of similar height. Photometric augmentation is restricted to brightness and contrast jitter and is applied at a reduced magnitude relative to the TransFuser default, since strong illumination shifts were observed to disrupt early-stage cross-modal attention alignment. Geometric transformations are applied identically to every modality of a frame, so that RGB, depth, semantics, and BEV remain registered, and the normalization statistics of both modalities are recalibrated to the expectations of the RegNetY-1.6GF backbones.

4. Curriculum Knowledge Distillation

This section develops the article's core contribution: a curriculum-based multi-task knowledge distillation framework that transfers the predictive power of the high-capacity TransFuser teacher to the lighter student [9]. The framework is motivated by RQ2. The compression described in Section 3 removes so much capacity that, as the ablation in Section 6.2 shows, the student cannot recover a usable driving policy from ground-truth supervision alone. At the same time, naively distilling all teacher signals from the first epoch destabilizes optimization. The proposed approach has three components: distillation at two representational levels (Section 4.1), a task-ordered curriculum that governs when each distillation signal is introduced (Section 4.2), and implementation measures that make teacher supervision affordable in memory (Section 4.3).

4.1. Feature-Level and Task-Level Distillation

In LightFuser, knowledge distillation is treated as a primary design tool rather than an optional regularizer. The teacher is a TransFuser model with RegNetY-3.2GF backbones; the student is LightFuser with RegNetY-1.6GF encoders and hybrid fusion. Following the standard paradigm, the student learns not only from ground-truth labels but also by matching the teacher's intermediate representations and final predictions [9], thereby enabling supervision at two complementary levels.

At the feature level, intermediate feature maps of teacher and student are aligned across multiple scales, encouraging the student's backbone to encode RGB and LiDAR inputs into a latent space that mirrors the teacher's semantic and geometric structure, in the spirit of hint-based distillation [23]. Formally, the feature-alignment loss is

$$L_{\text{feat}} = \sum_{s \in S} \| \varphi_s\left(F_s^{\text{S}}\right) - F_s^{\text{T}} \|^2 \tag{1}$$

where S is the set of distillation scales (selected stages of the RGB and LiDAR backbones together with the fused latent representation), F_s^{S} and F_s^{T} denote the student and teacher feature maps at scale s, and φ_s is a lightweight projection (a 1 × 1 convolution with, where necessary, spatial interpolation) that maps student features to the teacher's chan-

nel and spatial dimensions. The projections absorb the architectural mismatch between RegNetY-1.6GF and RegNetY-3.2GF feature hierarchies, so that the alignment penalizes representational—not dimensional—disagreement.

At the task level, the student matches the teacher's output heads: future trajectory waypoints for anticipatory control, BEV occupancy maps capturing road, lane, and obstacle layout, semantic segmentation, and dense depth estimates. In addition, the fused multi-modal representation obtained by combining RGB and LiDAR features is aligned across the teacher and student models, encouraging a similar cross-modal reasoning process that bridges perception and decision-making.

4.2. Task-Ordered Curriculum Schedule

To maintain training stability, especially given the multi-task nature of distillation, LightFuser controls not only how strongly each distillation signal acts on the student but also when it does so. Writing T = {waypoints, segmentation, depth, BEV} for the task set, the total objective at training epoch e is

$$L_{\text{total}} = \sum_{t \in T} \lambda_t L_t^{\text{sup}} + \sum_{t \in T} \gamma_t(e) L_t^{\text{KD}} + \gamma_{\text{feat}}(e) L_{\text{feat}} \tag{2}$$

where λ_t are fixed task weights, the supervised losses L_t^{sup} follow the task-specific choices detailed in Section 5.2 (Smooth L1 for waypoints and depth, weighted cross-entropy for segmentation, and a CenterNet-style composite loss for BEV detection), and the distillation losses L_t^{KD} use regression penalties for continuous outputs (waypoints, depth) and divergence-based penalties for probabilistic outputs (segmentation, BEV occupancy). The decisive elements are the epoch-dependent coefficients $\gamma_t(e)$, which implement the curriculum as a task-ordered linear ramp:

$$\gamma_t(e) = \overline{\gamma}_t \cdot \min\left(1, \max\left(0, \frac{e - e_t^0}{T_{\text{r}}}\right)\right) \tag{3}$$

where $\overline{\gamma}_t$ is the target weight of task t, e_t^0 is its activation epoch, and T_{r} is the ramp length. In all experiments T_{r} spans ten epochs of a thirty-epoch budget, and the activation epochs are $e_t^0 = 0$ for waypoints, 10 for depth, and 20 for semantic segmentation and BEV occupancy, so that each signal reaches its target weight before the next one is introduced; the two transitions are also visualized and marked in the results Figure, which is shown in Section 6. The same schedule governs $\gamma_{\text{feat}}(e)$. Ground-truth supervision is active from the first epoch throughout.

The task ordering encoded in the activation epochs e_t^0 is deliberate. Distillation begins with the waypoint head alone: the control trajectory is the lowest-dimensional and least-noisy target, and it aligns directly with the primary supervised objective, so early teacher guidance on waypoints accelerates rather than perturbs the optimization. The dense auxiliary signals are introduced progressively in later epochs, beginning with depth—a single-channel, continuous target whose gradients are comparatively well conditioned—and followed by the categorical and class-imbalanced semantic segmentation and BEV-occupancy objectives, because dense prediction targets are highly sensitive to the noisy, poorly formed student features of early training and, if imposed too early, dominate the gradient signal at the expense of the driving objective. This simple-to-complex ordering follows the rationale of curriculum learning [10] and mirrors the observation in multi-task learning that unbalanced loss magnitudes allow individual tasks to dominate the optimization [25].

The schedule follows from this ordering principle rather than from a search over activation epochs and target weights: the ablation in Section 6.2 compares it against its two natural alternatives—no distillation, and undifferentiated distillation from the first

epoch—rather than against a grid of schedules. A systematic sensitivity analysis over the curriculum hyperparameters was outside the scope of this study and is identified as future work in Section 9.

The resulting schedule allows the student to first establish a stable feature-extraction pipeline under direct supervision, then absorb the teacher's control behavior, and only afterwards refine its dense perceptual representations under full teacher guidance; Section 6.2 compares this schedule with its two natural alternatives.

4.3. Implementation and Memory Optimization

Distillation doubles the number of networks that must reside on the GPU during training, so its memory footprint must be engineered accordingly. The teacher remains frozen throughout; all teacher–student comparisons are performed only after ensuring spatial and dimensional compatibility between the corresponding feature maps and outputs, using the projections in Equation (1). Memory usage during distillation is further reduced by storing teacher activations in half-precision while accumulating student weights in full precision, which nearly halves memory consumption without compromising convergence—a direct application of mixed-precision training practice [27]. Finally, all architectural, fusion, and distillation hyperparameters, including the activation epochs and target weights in Equation (3), are consolidated into a single configuration file, making the curriculum reproducible and simplifying systematic tuning. The overall teacher–student distillation workflow is shown in Figure 3.

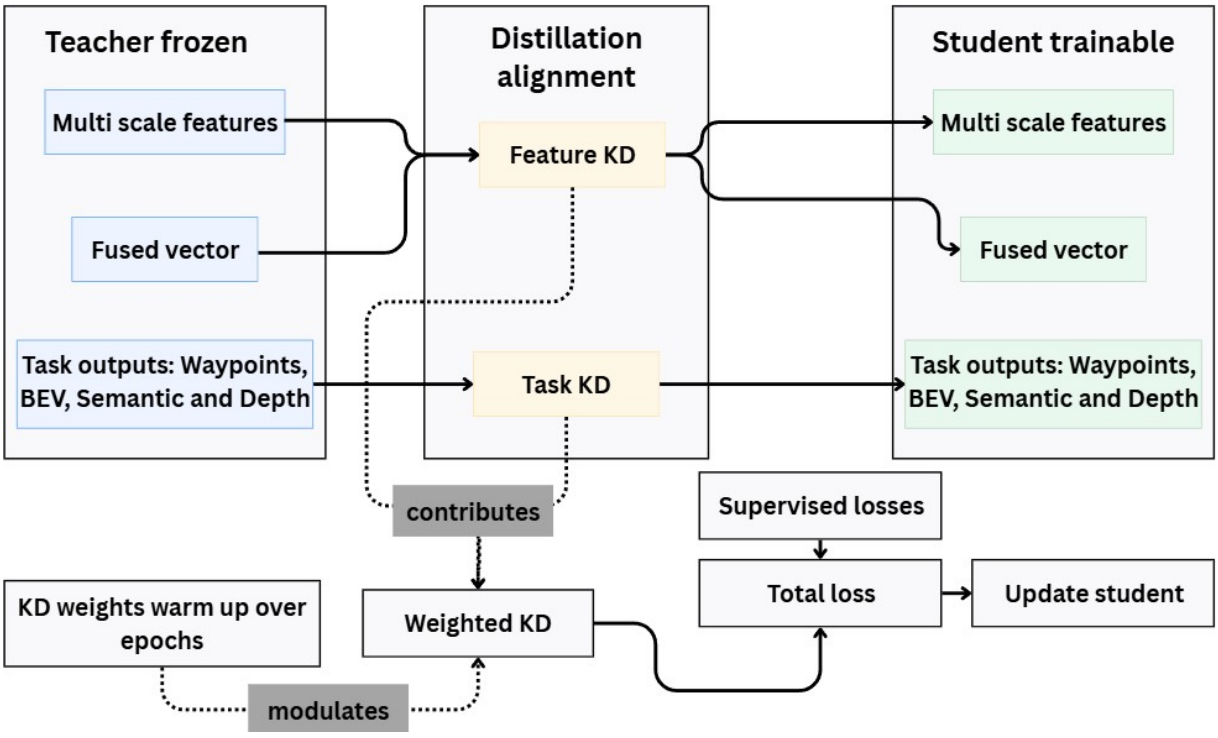

Figure 3. Knowledge-distillation workflow between the TransFuser teacher and the LightFuser student. Dashed lines represent intermediate representations, solid lines final predictions (*KD = Knowledge Distillation; BEV = Bird's Eye View; GRU = Gated Recurrent Unit).*

5. Experiments

5.1. Dataset and Data Generation

The experimental evaluation of the proposed hybrid Performer–Transformer fusion model is based on a dataset generated in the CARLA Leaderboard framework [5,6]. Data

collection is performed with a privileged agent, similar in spirit to the "cheating" policy introduced in prior work, which has full access to ground-truth information about the map layout and all dynamic actors. This agent is used only for supervision and data generation, not at test time, ensuring that training labels are accurate and reproducible [6,28].

Data collection covers towns 1–6 and uses a set of routes that is disjoint from the Longest6 routes on which the models are evaluated (Section 5.3): no evaluation route is driven during data generation. The towns are shared between the two sets, as is standard practice for the Longest6 benchmark, so the evaluation measures generalization to unseen routes, traffic configurations, and weather within seen towns rather than to unseen towns. The collected routes are of comparable length, approximately 1.5 km each, and expose the model to a wide range of road geometries and traffic patterns. Weather and illumination are randomized over clear, rainy, foggy, daytime, and nighttime conditions, mirroring the diversity in the official benchmark and encouraging robustness to environmental variability [6].

To simplify learning while retaining task-relevant structure, the original 28 semantic labels available in CARLA are compressed into a 7-class semantic map. Static, non-critical categories (e.g., buildings, fences, and vegetation) are grouped into a generic background class. In contrast, key driving elements—road surface, sidewalks, lane markings, pedestrians, vehicles, and traffic lights—are preserved as separate classes. This mapping reduces class imbalance and highlights decision-relevant entities [5].

The mapping from the original 28 CARLA semantic classes to the seven task-oriented training categories is summarized in Table 2. This reduced taxonomy preserves the classes most relevant to autonomous-driving decisions while limiting class imbalance and unnecessary semantic complexity.

Table 2. Mapping of CARLA semantic classes into the seven training categories.

ID	Class	CARLA Labels	Rationale
0	Unlabeled	Building, fence, vegetation, wall, etc.	Reduces class imbalance by removing non-critical entities.
1	Vehicle	Car, truck, bus, motorcycle, bicycle	Primary dynamic obstacles for collision avoidance.
2	Road	Road	Defines the primary drivable surface.
3	Red light	Traffic light (red/yellow phase)	Critical control signals requiring a stop.
4	Pedestrian	Pedestrian	High-priority vulnerable road users.
5	Road line	Lane marking, stop line	Guides lateral control and lane discipline.
6	Sidewalk	Sidewalk	Defines non-drivable boundaries.

At each step along a route, the privileged agent executes a safe control command based on full environment knowledge while triggering synchronized logging of all sensor streams (RGB, LiDAR, depth) and recording ground-truth labels for semantics, depth, occupancy, and future waypoints. Every frame is thus paired with consistent annotations across modalities, and the modular pipeline accommodates new towns, routes, or sensors with minimal changes.

A preprocessing stage converts the raw simulator outputs into standardized tensors: large modalities, such as LiDAR sweeps, are serialized into compressed arrays for efficient storage and loading; RGB images are normalized; and depth values are clipped to mitigate outliers. Geometric transformations are applied consistently across all modalities within a frame to preserve alignment among RGB, depth, semantics, and BEV. Temporally syn-

chronized samples are then grouped into mini-batches for GPU training. The overall data generation process is illustrated in Figure 4.

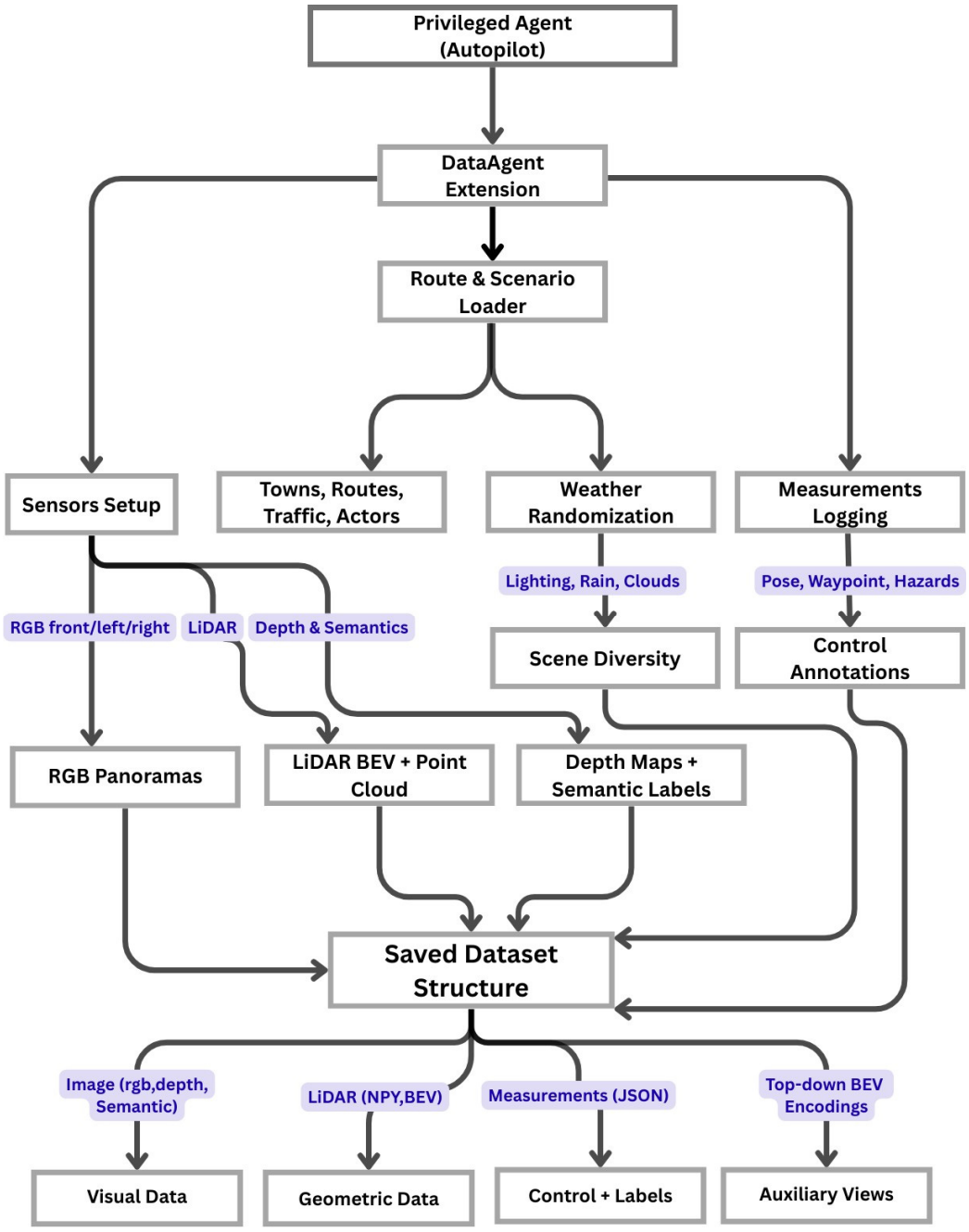

Figure 4. Overview of the data generation process in CARLA.

Table 3 summarizes the sensor modalities, preprocessing steps, and corresponding supervisory roles. Combining RGB, depth, semantic, LiDAR/BEV, and control signals enables the model to jointly learn visual understanding, geometric reasoning, and end-to-end driving behavior.

Table 3. Dataset modalities and their respective roles in training.

Modality	Data Type	Role in Training
RGB Panoramas	Visual frames	Scene understanding, traffic light state, lane perception.

Table 3. *Cont.*

Modality	Data Type	Role in Training
Depth Maps	Per-pixel distance	Geometric structure, obstacle distance estimation.
Semantic Segmentation	7-class label maps	Pixel-level scene parsing for context.
LiDAR Point Clouds	3D geometric sweeps	Direct 3D structure input for detection.
LiDAR BEV	2D occupancy maps	Road layout and free-space estimation.
Waypoints	Navigation targets	Supervision for control prediction.
Vehicle Measurements	Velocity, steering	Ground truth for imitation learning dynamics.

5.2. Training Setup

The training procedure for LightFuser combines multi-task supervised learning with knowledge distillation, under an optimization scheme designed for stability and efficiency.

The overall training objective is the weighted sum of task-specific losses given in Equation (2), encouraging the model to learn control, scene understanding, and spatial reasoning jointly. Waypoint prediction uses a Smooth L1 (Huber) loss against the privileged agent's trajectories, penalizing deviations while remaining robust to outliers; the same loss family also supervises dense depth estimation. Semantic segmentation uses a weighted cross-entropy loss whose per-class weights prioritize safety-critical categories—pedestrians (2.5) and vehicles (2.0)—with 0.5 for the background class. BEV supervision follows a CenterNet-style composite objective [29] that locates object centers in the BEV grid and regresses their size, orientation, and velocity, together with the vehicle's future braking intent [29,30]. The distillation terms of Section 4 complement this ground-truth supervision across all heads.

Training uses the Adam optimizer, which is well suited to multi-task settings where gradient magnitudes vary across heads. The base learning rate is set to 1×10^{-4} and follows a cosine annealing schedule, gradually decaying over the 30 training epochs to promote smooth convergence [31].

Gradient clipping with a maximum norm of 5 protects against occasional large updates from outliers or strong distillation signals. Training runs in mixed precision (FP16/FP32) with dynamic loss scaling, thereby reducing the memory footprint and accelerating computation without sacrificing numerical stability [27].

A curriculum schedule controls the influence of knowledge distillation, following Equation (3) in Section 4.2: distillation weights start near zero, are activated first for the waypoint head, and increase linearly over the first 10 of 30 epochs before saturating, so that the student establishes a stable supervised solution before absorbing the teacher's guidance [10].

In each iteration, the frozen teacher processes the same synchronized mini-batch as the student, the composite objective of Equation (2) is evaluated with the current curriculum weights, and backpropagation updates only the student. Performance on a held-out validation set is monitored periodically, and the checkpoint with the best driving metrics is retained for testing.

5.3. Evaluation Methodology

All experiments are conducted in the CARLA Leaderboard environment, whose routes span multiple towns with diverse layouts and are evaluated under varying weather and illumination, from clear daytime to rain, fog, and night. For a fair comparison, all agents use the same sensor suite (panoramic RGB and LiDAR BEV) and run on a single workstation with an NVIDIA RTX 2080 (8 GB) under a controlled software environment, with fixed random seeds and deterministic simulation settings to minimize variance.

Driving quality is quantified using the official CARLA Leaderboard metrics [6]: Route Completion (RC), the percentage of the route distance actually traversed; the Infraction Score (IS), a multiplicative penalty that starts at 1.0 and is reduced for each collision or traffic violation; and the Driving Score (DS), the product of RC and IS per route averaged across routes, which jointly rewards progress and safety. Infractions are additionally summarized as counts per kilometer,

$$I/\mathrm{km} = \frac{\sum_i \mathrm{Infractions}_i}{\sum_i k_i},$$

where k_i is the distance driven in kilometers along route i. This normalizes events such as collisions, off-road episodes, and red-light violations by distance, yielding a fair comparison even when route lengths differ.

Complementary statistics (e.g., lane-invasion rate, off-road percentage, timeouts) are reported to highlight specific error modes that aggregate metrics might obscure.

Evaluation uses the Longest6 benchmark of the CARLA Leaderboard: for each of towns 1–6 the six longest predefined routes are selected, giving $N = 36$ routes of approximately 1.5 km each. Each route is driven once per agent under fixed random seeds and deterministic simulator stepping, and the aggregate metrics are means over routes. The $\pm$ values reported in Table 4 are accordingly standard deviations over these 36 routes for a single trained model per method. They quantify route-to-route variability, that is, the spread between easier and harder routes, and not variability across random seeds or across independent training runs; neither the training nor the evaluation was repeated for this study. The intervals should therefore not be read as confidence intervals on the difference between LightFuser and its teacher, and no claim of statistical significance is attached to that difference. Establishing one would require several independent trainings per configuration, which exceeded the compute budget available here and is noted in Section 8. The Late Fusion, Geometric Fusion, and TransFuser rows of Table 4 are reproduced from [2,6] with the dispersion published there; we have not verified that the underlying convention matches ours, so those intervals are comparable in magnitude but not necessarily identical in definition to the LightFuser row.

Table 4. Driving performance comparison on the CARLA Leaderboard benchmark.

Method	DS↑	RC↑	IS↑	Ped↓	Veh↓	Red↓
Late Fusion (LF) [2]	22.47 ± 3.71	83.30 ± 3.04	0.27 ± 0.04	0.05	4.63	0.11
Geometric Fusion (GF) [2]	27.32 ± 0.80	91.13 ± 0.95	0.30 ± 0.01	0.06	4.64	0.13
TransFuser [2,6]	47.30 ± 5.72	93.38 ± 1.20	0.50 ± 0.06	0.03	2.45	0.16
LightFuser (Proposed)	42.40 ± 5.10	90.90 ± 1.30	0.45 ± 0.07	0.04	2.58	0.17

Note: DS = Driving Score; RC = Route Completion; IS = Infraction Score; Ped = pedestrian infractions; Veh = vehicle infractions; Red = red-light infractions. Arrows indicate whether higher (↑) or lower (↓) values are preferable. Values are mean ± standard deviation over the $N = 36$ Longest6 evaluation routes for a single trained model per method; the dispersion is route-to-route and does not cover variability across seeds or independent training runs (Section 5.3). The infraction columns are per-kilometer rates aggregated over all routes and carry no dispersion. The LF, GF, and TransFuser rows are reproduced from [2,6].

Because efficiency is central to LightFuser's design, each agent's inference pipeline is profiled per component—image backbone, LiDAR backbone, attention-based fusion, decoder, and output head—reporting the median latency and interquartile range per frame. The measurement covers network modules only and therefore excludes simulator stepping, sensor acquisition, preprocessing, and I/O overhead. The reported frame rate is consequently an upper bound on what a deployed pipeline could achieve, not a measurement of the deployed system itself. Peak GPU memory usage and parameter counts complete the

computational profile. Summing the per-component times yields the total network latency per frame and thus the model-only frame rate, a practical measure of compatibility with real-time control loops.

6. Results

6.1. Driving Performance

Driving performance is compared against Late Fusion (LF) and Geometric Fusion (GF), two simpler fusion strategies from the TransFuser benchmark, and against the TransFuser teacher itself [2,6]; Table 4 summarizes the results. The simple baselines reach Driving Scores only in the low-to-mid twenties, reflecting limited competence in complex scenarios. TransFuser attains a Driving Score of 47.3 with Route Completion above 93%, confirming its strength as a teacher [2,28]. LightFuser reaches a Driving Score of 42.4 at 90.9% Route Completion, clearly outperforming the simpler baselines and narrowing the gap to the teacher at a fraction of the computational cost. The infraction breakdown shows pedestrian collisions as rare as the teacher's, while vehicle collisions and red-light violations are slightly more frequent, consistent with the lower Driving Score.

6.2. Distillation Effects

When the student is trained only with supervised losses (without KD), optimization fails to produce a usable driving policy: even after 30 epochs, the agent struggles to follow routes, often spinning or getting stuck, and its auxiliary outputs (semantics, depth, BEV) remain noisy and incoherent. This indicates that the reduced-capacity architecture cannot, on its own, discover a sufficiently rich solution from the available supervision.

At the opposite extreme, enabling all KD losses from the very start of training leads to unstable behavior. Strong teacher guidance overwhelms the small student: the training loss exhibits pronounced oscillations, and the model fails to converge to a stable driving policy.

To resolve this, the task-ordered curriculum of Section 4.2 is adopted: distillation weights are increased gradually over the first third of training, beginning with the control waypoints alone, while the auxiliary targets (depth, semantics, and BEV occupancy) are introduced progressively in later epochs. The two vertical markers in Figure 5 indicate the epochs at which the depth (epoch 10) and the semantic and BEV occupancy (epoch 20) distillation terms are activated. Figure 5 illustrates the three resulting convergence regimes: without distillation, the objective remains flat; with full distillation from the first epoch, it oscillates; and under the curriculum, it decreases smoothly and monotonically within the 30-epoch budget.

Two properties of that figure should be stated explicitly. Its ordinate is the total training objective of each regime, normalized to a common relative scale: the three regimes optimize different objectives—supervised losses only, supervised losses plus all distillation terms, and supervised losses plus a time-varying subset of them—so their loss values do not share an absolute scale, and only the shape of each curve is informative. The curves were produced from the training runs reported here, but the underlying loss logs were not archived with the results and can no longer be recovered. Figure 5 is therefore presented as a qualitative record of the convergence behavior observed rather than as a quantitative loss trace; each regime is shown for a single run at a fixed seed, and no variability across seeds is depicted.

The progressive introduction of distillation losses prevents the strong teacher signals from dominating early optimization, allowing the student to establish stable task-specific representations before incorporating higher-level multimodal knowledge. Table 5 summarizes the three regimes and their outcomes.

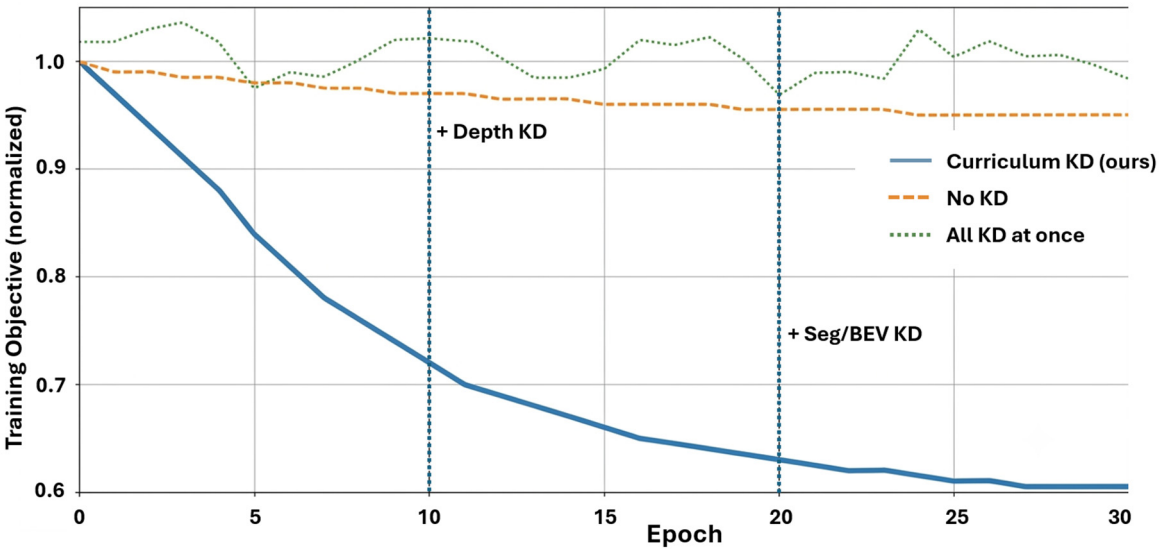

Figure 5. Indicative convergence behavior under different knowledge-distillation strategies. The ordinate is the total training objective of each regime, normalized to a common relative scale; absolute values are not comparable across the three curves, because the regimes optimize different objectives. The curves were produced from the training runs reported here, but the underlying loss logs were not archived and cannot be recovered, so the figure is a qualitative record of the three convergence behaviors rather than a quantitative loss trace, and each regime is shown for a single run at a fixed seed. The vertical markers denote the curriculum activation epochs: the depth distillation term is activated at epoch 10 and the semantic segmentation and BEV occupancy terms at epoch 20; the waypoint term is active from epoch 0.

Table 5. Convergence behavior and final driving performance under the three distillation regimes.

Distillation Regime	Convergence Behavior	DS↑	RC↑	IS↑
No KD (supervised only)	Flat, non-convergent; no usable driving policy after 30 epochs	—	—	—
Full KD from epoch 1	Oscillatory, unstable loss; fails to reach a stable policy	—	—	—
Curriculum KD (task-ordered, proposed)	Smooth, monotonic convergence within 30 epochs	42.4 ± 5.1	90.9 ± 1.3	0.45 ± 0.07

Note: DS = Driving Score; RC = Route Completion; IS = Infraction Score. Arrows indicate higher (↑) values are preferable. Dashes indicate that no meaningful score could be obtained because training did not converge to a usable policy. Convergence behavior is characterized qualitatively from the training runs; see Figure 5 and the accompanying text in Section 6.2 for what the curves do and do not represent.

Under the curriculum schedule, LightFuser converges reliably to the performance reported in Section 6.1. Distillation is therefore not merely helpful but necessary for the compact architecture, and, with respect to RQ2, the ablation identifies the task-ordered curriculum—not distillation per se—as the element that makes the lightweight multi-task student trainable at all [9,10]. The scope of this evidence should be stated precisely. The ablation contrasts the curriculum regime as a whole with its two endpoints; it does not decompose the regime into its parts. It therefore establishes that a staged, progressively weighted introduction of teacher supervision is required for this compressed student, but it does not establish that the particular task ordering, the linear shape and length of the ramp in Equation (3), or the combination of feature-level and task-level supervision is individually necessary. Each of these follows from the design rationale of Section 4.2

rather than from a controlled comparison, and separating their contributions requires the component-wise ablation set out in Section 8.

6.3. Efficiency Analysis

Table 6 presents the component-level latency comparison. Total network latency drops from about 128.4 ms per frame with TransFuser to 67.6 ms with LightFuser, an overall reduction of about 47%. This increases the estimated model-only throughput from 7.8 FPS to 14.8 FPS on the same GPU platform [2,28]—sufficient to keep pace with a 10 Hz closed-loop control cycle and thus near-real-time in the sense of RQ1.

Table 6. Component-level inference latency of TransFuser and LightFuser.

Module	TransFuser (ms) [IQR]	LightFuser (ms) [IQR]	Reduction
Image Backbone	46.04 [44.77–47.58]	23.12 [22.56–23.90]	49.8%
LiDAR Backbone	44.93 [43.48–46.15]	27.09 [26.42–27.94]	39.7%
Fusion	34.12 [33.30–35.21]	14.75 [14.36–15.24]	56.8%
Decoder	0.08 [0.07–0.08]	0.05 [0.04–0.05]	40.0%
Head	2.92 [2.71–3.00]	2.34 [2.16–2.40]	19.8%
Total Network	128.38 [124.86–132.27]	67.59 [65.96–69.67]	47.3%

As Table 6 shows, the savings are not confined to a single component: image-backbone latency halves, and LiDAR-backbone latency falls by roughly 40% owing to the lighter encoders. In comparison, the fusion stage shows the largest relative gain—about 57%—as a direct consequence of the hybrid Performer–Transformer design. Decoder and head costs were already negligible. Importantly, the improvements do not shift the bottleneck elsewhere: backbones and fusion remain the dominant costs at much lower absolute latency, and fusion's share of total network time drops from over a quarter to about one-fifth. Parameter counts decrease accordingly (Section 3.3): the encoder pair falls from 38.8 M to 22.4 M parameters. Peak GPU memory was not instrumented separately for this study. The reduction in activation memory implied by the halved channel widths and by the linear-complexity early fusion stage is therefore reported qualitatively in Table 1 rather than as a measured figure, and quantifying it is part of the embedded-platform profiling identified as future work in Section 9.

One element of the computational profile is available for the baseline but not for the student. For TransFuser the complete model comprises approximately 165.8 million parameters and about 34.9 G multiply–accumulate operations per frame at the input resolutions used here, with an inference-time GPU memory footprint of roughly 2.8 GB [4,30]. The corresponding whole-model figures for LightFuser were not recorded during the original study: the parameter and operation accounting that was archived covers the encoder pair only, 38.8 M against 22.4 M parameters and 3.2 against 1.6 GFLOPs per stream (Table 1). We report this as a gap rather than closing it with an estimate, because a whole-model total obtained by adding an unaudited fusion, decoder, and head budget to the measured encoder difference would carry more apparent precision than the underlying record supports. What is measured end-to-end for both complete models is latency: Table 6 profiles the full inference pipeline of each network on identical hardware and inputs and gives 128.38 ms per frame for TransFuser against 67.59 ms for LightFuser, a reduction of 47.3%. That measurement covers the whole model rather than the backbones alone, and it is the quantity on which the efficiency claims of this article rest. Should the whole-model parameter, multiply–accumulate, and peak-memory totals be required, the clean remedy is to recompute them under a documented and archived profiling protocol and to report them

alongside Tables 1 and 6; this is part of the embedded-platform characterization identified as future work in Section 9.

7. Discussion

The central objective of this work was to design a sensor-fusion architecture that preserves TransFuser's driving competence while substantially reducing computational cost. The empirical results indicate that LightFuser approaches this target. As shown in the benchmark comparison, its average Driving Score remains competitive and clearly above those of earlier fusion baselines. At the same time, the efficiency analysis reveals a nearly twofold reduction in network compute. Taken together, these findings suggest that the model achieves the intended balance between performance and efficiency.

RQ1—compression without catastrophic loss of driving competence. The evidence supports an affirmative answer within the tested setting. Halving network latency (128.4 ms $\rightarrow$ 67.6 ms per frame; 7.8 $\rightarrow$ 14.8 FPS) cost roughly five Driving Score points relative to the teacher (47.3 $\rightarrow$ 42.4), while Route Completion remained within three percentage points and pedestrian-collision rates stayed at the teacher's level. The degradation is graceful and concentrated in specific infraction types rather than in a global collapse of driving ability; the compressed student remains far above the simpler Late Fusion and Geometric Fusion baselines on every aggregate metric. Compression alone, however, does not yield this result—the architectural reduction becomes viable only in combination with the training regime addressed by RQ2.

RQ2—the training regime for stable convergence. The ablation in Section 6.2 delineates the answer sharply: neither pure supervised learning nor full distillation from the first epoch produces a usable policy, whereas the task-ordered curriculum of Section 4 converges smoothly within 30 epochs. The regime combines three design properties, which Section 6.2 tests jointly rather than one at a time. First, distillation signals are ramped in gradually (Equation (3)), so that the student can form stable low-level features before strong teacher gradients act on them. Second, the curriculum is task-ordered: anchoring it on the low-dimensional waypoint objective before introducing dense semantic, depth, and BEV targets prevents the auxiliary losses from dominating early optimization. Third, supervision spans both representational levels, on the reasoning that matching only final outputs would deprive the student of intermediate structure it lacks the capacity to discover on its own. What the ablation shows is that removing all three at once and imposing all three at once from the first epoch both fail, which is what identifies the regime as load-bearing. It does not attribute the effect to any single component, so the three properties above should be read as the rationale the schedule was built on rather than as separately demonstrated necessities; establishing the latter requires the component-wise ablation described in Section 8. Subject to that qualification, the regime is offered as a transferable recipe for training compact multi-task students under strong teachers.

However, the gains are not uniform across all aspects of driving. LightFuser exhibits slightly higher rates of vehicle collisions and red-light violations than its teacher, indicating that certain safety-critical cues are more sensitive to reduced model capacity. This effect is especially pronounced in challenging scenarios, such as dense traffic or adverse weather conditions, where TransFuser's richer representations retain a noticeable advantage. Under typical conditions, the performance gap is smaller, and the distilled student behaves stably and reliably.

These observations underscore the trade-off inherent in lightweight design: substantial reductions in computational load are attainable without collapsing driving ability, but specific failure modes become somewhat more frequent than in the teacher. For deployment in safety-critical settings, the efficiency gains should therefore be paired with

targeted mitigation of the affected infraction types, for example, rule-based safeguards at intersections.

The magnitude of these two effects deserves to be examined with precision. Vehicle-collision infractions rise from 2.45 to 2.58 per kilometer and red-light infractions from 0.16 to 0.17 per kilometer, increases of roughly 5% and 6% against a 47% reduction in network latency. Table 4 reports no per-route dispersion for the infraction counts, so these differences are best read as a consistent tendency rather than as separately significant effects. Their direction is nevertheless interpretable. Both infraction types depend on small, distant, or partially occluded evidence—the brake onset of a lead vehicle, a signal head at the far side of an intersection—and that evidence is carried by exactly the high-resolution image features that the FAVOR+ approximation represents only approximately and that the halved channel widths encode with less capacity. Pedestrian collisions, which turn on large, near-field, high-contrast evidence, remain at the teacher's level (0.04 versus 0.03 per kilometer), which is consistent with this reading.

8. Limitations

While the results of this study are promising, several limitations must be considered when interpreting the findings.

First, the evaluation is conducted entirely in the CARLA simulator using the official Leaderboard protocol. Although widely adopted, these benchmarks remain abstractions of reality. Aspects such as sensor noise, complex weather dynamics, and long-horizon interactions with human drivers are simplified. Consequently, real-world deployment could reveal additional weaknesses that are not captured in simulation [5,6].

Second, the Leaderboard prioritizes driving-policy robustness over computational efficiency. The efficiency metrics reported for LightFuser (latency, FLOPs, memory footprint) come from separate profiling and are not included in the official score. This creates a mismatch between how agents are ranked in the benchmark and the criteria that matter most for deployment on resource-constrained platforms [6]. The computational profile is also incomplete on the student side. Whole-model parameter and multiply–accumulate counts are reported for the TransFuser baseline but were not recorded for LightFuser, so the compression is quantified end-to-end by measured latency (Table 6) and at encoder level by parameter count (Table 1), but not by a whole-model parameter or operation total (Section 6.3).

Third, the experiments are restricted to a single GPU platform (RTX 2080). Although profiling isolates model-level costs, absolute throughput on embedded devices—such as automotive-grade SoCs or low-power GPUs—may differ significantly. Without hardware-specific optimizations, treat the reported efficiency gains as indicative rather than definitive.

Finally, LightFuser inherits structural assumptions from its teacher. Reliance on frame-level fusion without an explicit temporal component constrains long-horizon reasoning in dynamic scenes. Moreover, the distillation procedure is tailored specifically to a TransFuser teacher; transferring the same scheme to a different teacher architecture would likely require substantial retuning.

A further boundary concerns the ablation of the training regime itself. Section 6.2 compares three settings—no distillation, undifferentiated distillation from the first epoch, and the proposed task-ordered curriculum—and therefore varies the regime as a whole. It does not isolate the individual ingredients: the task ordering, the shape and length of the linear ramp in Equation (3), the choice of activation epochs, and the use of feature-level supervision alongside task-level supervision are all present together in the successful arm and all absent, or all simultaneously active, in the two failing arms. The experiments consequently support the conclusion that a staged, progressively weighted distillation

schedule is required for this compressed student, but they do not establish that any one of these components is individually necessary, and they do not rank their contributions. A component-wise ablation, varying the ordering against a fixed ramp, the ramp against a fixed ordering, and feature-level against task-level supervision in isolation, together with the sensitivity analysis over activation epochs and target weights noted in Section 4.2, would be needed to make the stronger claim; both are identified as future work in Section 9.

Beyond the simulator-to-reality gap itself, deploying on a physical platform raises three issues that this study does not address. First, the sensor interface is idealized: the model consumes perfectly time-aligned panoramas and BEV grids produced by the simulator, whereas a vehicle must absorb timing jitter, dropped frames, rolling-shutter effects, and calibration drift between camera and LiDAR—and the profiling reported in Section 6.3 explicitly excludes the acquisition and preprocessing stages in which those effects arise. Second, the latency figures are model-only and were obtained on a desktop RTX 2080; an automotive SoC has a different memory hierarchy and a different set of well-optimized operators, and the Performer kernel in particular depends on primitives that embedded inference runtimes support unevenly. Third, the residual gap to the teacher is concentrated in precisely the categories a safety case would scrutinize most closely, so a deployed variant would need the rule-based interlocks discussed in Section 7 rather than the learned policy alone.

These limitations do not invalidate the contribution of this work; rather, they define its scope and the boundaries within which the conclusions can be generalized.

One further boundary should be stated explicitly, because it delimits what the reported numbers can support. The evaluation is closed-loop by construction: teacher and student are trained on privileged-agent demonstrations recorded inside CARLA with a fixed sensor rig, and both are assessed by driving the routes rather than by scoring logged trajectories. Open-loop metrics on external driving datasets were therefore not used. Under a different sensor geometry, calibration, and label taxonomy, such an evaluation would principally measure the domain gap between CARLA and the target dataset. In contrast, the controlled quantity of interest here is the effect of compression at a fixed data distribution and a fixed sensor interface.

9. Conclusions

This work introduced LightFuser, a hybrid Performer–Transformer fusion architecture for efficient end-to-end autonomous driving. The design consolidates several key ideas into a single model: replacing heavy encoders with compact RegNetY-1.6GF backbones, employing a hybrid attention fusion scheme that combines Performer-based linear attention in high-resolution stages with full Transformer attention at reduced spatial scales, and integrating a staged knowledge distillation framework from a high-capacity TransFuser teacher. Together, these elements directly address the computational bottlenecks identified in prior analysis by reducing encoder load, limiting attention overhead, and easing sensor timing pressure—without sacrificing the benefits of multimodal reasoning.

On the CARLA Leaderboard, LightFuser maintains competitive driving performance while nearly halving network latency, and profiling confirms that the gains stem primarily from lighter backbones and a streamlined fusion stage; the remaining gap to the teacher is moderate and concentrated under adverse conditions [6,28]. More generally, the study shows that careful profiling directs optimization effort to the components with the largest impact, that curriculum-based distillation is a practical recipe for training compact yet competent models in multi-task, multimodal settings, and that the hybrid Performer–Transformer fusion pattern offers a blueprint for reconciling strong performance with strict compute and memory budgets.

Returning to the research questions, a transformer-based multimodal fusion model can indeed be compressed to near-real-time performance without catastrophic loss of driving competence (RQ1), but only when the compression is guided by component-level profiling and coupled with a knowledge-transfer mechanism; in addition, a training regime that makes such lightweight multi-task students converge stably (RQ2) is a task-ordered, progressively weighted distillation curriculum in which teacher supervision is introduced first for the control objective and only later for the dense perception tasks. That answer rests on a comparison between the regime and its two endpoints, so it identifies the staged introduction of teacher supervision as necessary without isolating the contributions of the ordering, the ramp, and the two levels of supervision within it.

Future work will proceed along three lines. First, extending LightFuser to temporal fusion with sequence-level attention would address the frame-level limitation inherited from the teacher and improve robustness to occlusions and fast-changing traffic. Second, profiling and optimizing the model on embedded automotive platforms—through quantization, structured pruning, and operator fusion, with explicit budgets in milliseconds, frames per second, memory, and power [21]—would turn the reported indicative gains into deployment-grade figures. Third, the curriculum distillation recipe itself deserves broader study: applying it to different teacher architectures and other multi-task domains would establish how far the task-ordered schedule generalizes beyond the TransFuser–LightFuser pairing examined here. A component-wise ablation of the recipe, varying the task ordering against a fixed ramp, the ramp shape and length against a fixed ordering, and feature-level against task-level supervision in isolation, together with a sensitivity analysis over the activation epochs and target weights, is a prerequisite for that broader study and is the more immediate priority.

Author Contributions: Conceptualization, S.H. and M.B.M.; methodology, S.H. and D.B.; software, D.B.; validation, S.H.; investigation, S.H., M.B.M. and D.B.; resources, S.H.; writing—original draft preparation, S.H. and D.B.; writing—review and editing, S.H. and M.B.M.; visualization, S.H. and D.B.; funding acquisition, M.B.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

Acknowledgments: This work was partly supported by the European Regional Development Fund under the "Research Innovation and Digitization for Smart Transformation" program 2021–2027 under the Project BG16RFPR002-1.014-0006, "National Centre of Excellence Mechatronics and Clean Technologies".

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

References

1. 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), Nashville, TN, USA, 20–25 June 2021; pp. 7073–7083.
2. Chitta, K.; Prakash, A.; Jaeger, B.; Yu, Z.; Renz, K.; Geiger, A. TransFuser: Imitation with Transformer-Based Sensor Fusion for Autonomous Driving. *IEEE Trans. Pattern Anal. Mach. Intell.* **2023**, *45*, 12878–12895. [CrossRef]
3. Choromanski, K.; Likhosherstov, V.; Dohan, D.; Song, X.; Gane, A.; Sarlos, T.; Hawkins, P.; Davis, J.; Mohiuddin, A.; Kaiser, L.; et al. Rethinking Attention with Performers. In Proceedings of the International Conference on Learning Representations (ICLR), Virtual Event, 3–7 May 2021.

4. Jia, X.; Wu, P.; Chen, L.; Xie, J.; He, C.; Yan, J.; Li, H. Think Twice before Driving: Towards Scalable Decoders for End-to-End Autonomous Driving. In Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 18–22 June 2023; pp. 21983–21994.
5. Dosovitskiy, A.; Ros, G.; Codevilla, F.; Lopez, A.; Koltun, V. CARLA: An Open Urban Driving Simulator. In Proceedings of the 1st Annual Conference on Robot Learning (CoRL), Mountain View, CA, USA, 13–15 November 2017; pp. 1–16.
6. CARLA Autonomous Driving Leaderboard. Available online: https://leaderboard.carla.org/ (accessed on 6 July 2026).
7. Radosavovic, I.; Kosaraju, R.P.; Girshick, R.; He, K.; Dollar, P. Designing Network Design Spaces. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 10428–10436.
8. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention Is All You Need. In *Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2017; Volume 30.
9. Hinton, G.; Vinyals, O.; Dean, J. Distilling the Knowledge in a Neural Network. *arXiv* **2015**, arXiv:1503.02531.
10. Bengio, Y.; Louradour, J.; Collobert, R.; Weston, J. Curriculum Learning. In Proceedings of the 26th Annual International Conference on Machine Learning (ICML), Montreal, QC, Canada, 14–18 June 2009; pp. 41–48. [CrossRef]
11. Fayyad, J.; Jaradat, M.A.; Gruyer, D.; Najjaran, H. Deep Learning Sensor Fusion for Autonomous Vehicle Perception and Localization: A Review. *Sensors* **2020**, *20*, 4220. [CrossRef]
12. Liang, M.; Yang, B.; Wang, S.; Urtasun, R. Deep Continuous Fusion for Multi-Sensor 3D Object Detection. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 641–656.
13. Liu, Z.; Tang, H.; Amini, A.; Yang, X.; Mao, H.; Rus, D.; Han, S. BEVFusion: Multi-Task Multi-Sensor Fusion with Unified Bird's-Eye View Representation. In Proceedings of the 2023 IEEE International Conference on Robotics and Automation (ICRA), London, UK, 29 May–2 June 2023; pp. 2774–2781.
14. Chen, D.; Zhou, B.; Koltun, V.; Krahenbuhl, P. Learning by Cheating. In Proceedings of the Conference on Robot Learning (CoRL), Osaka, Japan, 30 October–1 November 2019; pp. 66–75.
15. Shao, H.; Wang, L.; Chen, R.; Li, H.; Liu, Y. Safety-Enhanced Autonomous Driving Using Interpretable Sensor Fusion Transformer. In Proceedings of the Conference on Robot Learning (CoRL), Auckland, New Zealand, 14–18 December 2022; pp. 726–737.
16. Shao, H.; Wang, L.; Chen, R.; Waslander, S.L.; Li, H.; Liu, Y. ReasonNet: End-to-End Driving with Temporal and Global Reasoning. In Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 18–22 June 2023; pp. 13723–13733.
17. 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*; Curran Associates, Inc.: Red Hook, NY, USA, 2022; Volume 35, pp. 6119–6132.
18. Jaeger, B.; Chitta, K.; Geiger, A. Hidden Biases of End-to-End Driving Models. In Proceedings of the 2023 IEEE/CVF International Conference on Computer Vision (ICCV), Paris, France, 2–6 October 2023; pp. 8240–8249.
19. Hu, Y.; Yang, J.; Chen, L.; Li, K.; Sima, C.; Zhu, X.; Chai, S.; Du, S.; Lin, T.; Wang, W.; et al. Planning-Oriented Autonomous Driving. In Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 18–22 June 2023; pp. 17853–17862.
20. Jiang, B.; Chen, S.; Xu, Q.; Liao, B.; Chen, J.; Zhou, H.; Zhang, Q.; Liu, W.; Huang, C.; Wang, X. VAD: Vectorized Scene Representation for Efficient Autonomous Driving. In Proceedings of the 2023 IEEE/CVF International Conference on Computer Vision (ICCV), Paris, France, 2–6 October 2023; pp. 8340–8350.
21. Menghani, G. Efficient Deep Learning: A Survey on Making Deep Learning Models Smaller, Faster, and Better. *ACM Comput. Surv.* **2023**, *55*, 259. [CrossRef]
22. Tay, Y.; Dehghani, M.; Bahri, D.; Metzler, D. Efficient Transformers: A Survey. *ACM Comput. Surv.* **2022**, *55*, 109. [CrossRef]
23. Romero, A.; Ballas, N.; Kahou, S.E.; Chassang, A.; Gatta, C.; Bengio, Y. FitNets: Hints for Thin Deep Nets. In Proceedings of the International Conference on Learning Representations (ICLR), San Diego, CA, USA, 7–9 May 2015.
24. Gou, J.; Yu, B.; Maybank, S.J.; Tao, D. Knowledge Distillation: A Survey. *Int. J. Comput. Vis.* **2021**, *129*, 1789–1819. [CrossRef]
25. Kendall, A.; Gal, Y.; Cipolla, R. Multi-Task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 7482–7491.
26. Chung, J.; Gulcehre, C.; Cho, K.; Bengio, Y. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *arXiv* **2014**, arXiv:1412.3555.
27. Micikevicius, P.; Narang, S.; Alben, J.; Diamos, G.; Elsen, E.; Garcia, D.; Ginsburg, B.; Houston, M.; Kuchaiev, O.; Venkatesh, G.; et al. Mixed Precision Training. *arXiv* **2017**, arXiv:1710.03740.
28. Barhoumi, D. *AI Methods for Sensor Data Processing and Fusion*; Graduation Project Report; National Institute of Applied Sciences and Technology, University of Carthage: Carthage, Tunisia, 2025, *unpublished work*.
29. Zhou, X.; Wang, D.; Krahenbuhl, P. Objects as Points. *arXiv* **2019**, arXiv:1904.07850.

30. Lang, A.H.; Vora, S.; Caesar, H.; Zhou, L.; Yang, J.; Beijbom, O. PointPillars: Fast Encoders for Object Detection from Point Clouds. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 12697–12705. [CrossRef]
31. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. In Proceedings of the International Conference on Learning Representations (ICLR), San Diego, CA, USA, 7–9 May 2015.