FQ-PETR: Fully Quantized Position Embedding Transformation for Multi-View 3D Object Detection

Jiangyong Yu[1], Changyong Shu[1*], Sifan Zhou[2], Zichen Yu[3], Xing Hu[1], Dawei Yang[1*]
[1]Houmo AI
[2]Southeast University
[3]Dalian University of Technology
{jiangyong.yu, changyong.shu, xing.hu, dawei.yang}@houmo.ai,
sifanjay@gmail.com, yuzichen@mail.dlut.edu.cn

Abstract

Camera-based multi-view 3D detection is crucial for autonomous driving. PETR and its variants (PETRs) excel in benchmarks but face deployment challenges due to high computational cost and memory footprint. Quantization is an effective technique for compressing deep neural networks by reducing the bit width of weights and activations. However, directly applying existing quantization methods to PETRs leads to severe accuracy degradation. This issue primarily arises from two key challenges: (1) significant magnitude disparity between multi-modal features—specifically, image features and camera-ray positional embeddings (PE), and (2) the inefficiency and approximation error of quantizing non-linear operators, which commonly rely on hardware-unfriendly computations. In this paper, we propose **FQ-PETR**, a fully quantized framework for PETRs, featuring three key innovations: (1) Quantization-Friendly LiDAR-ray Position Embedding (QFPE): Replacing multi-point sampling with LiDAR-prior-guided single-point sampling and anchor-based embedding eliminates problematic non-linearities (e.g., inverse-sigmoid) and aligns PE scale with image features, preserving accuracy. (2) Dual-Lookup Table (DULUT): This algorithm approximates complex non-linear functions using two cascaded linear LUTs, achieving high fidelity with minimal entries and no specialized hardware. (3) Quantization After Numerical Stabilization (QANS): Performing quantization after softmax numerical stabilization mitigates attention distortion from large inputs. On PETRs (e.g. PETR, StreamPETR, PETRv2, MV2d), FQ-PETR under W8A8 achieves near-floating-point accuracy ($< 1\%$ degradation) while reducing latency by up to 75%, significantly outperforming existing PTQ and QAT baselines.

Code — https://github.com/JiangYongYu1/FQ-PETR

Introduction

Camera-based multi-view 3D object detection has become a practical alternative to LiDAR systems for autonomous driving, offering high-resolution perception and strong cost–efficiency. Among existing approaches, PETR and its variants (collectively PETRs) (Liu et al. 2022a,b; Wang et al. 2023a) achieve state-of-the-art performance by adapting the transformer-based DETR framework to 3D space. Their key idea is the position-aware representation: camera-ray positional embeddings map 2D image features into 3D coordinates, enabling direct interaction between queries and 3D features (Fig. 1a). This design has pushed PETRs to the top of major benchmarks such as nuScenes and Waymo. However, their substantial computational cost and memory footprint pose serious deployment challenges, making real-time inference under tight latency and power constraints difficult for automotive systems.

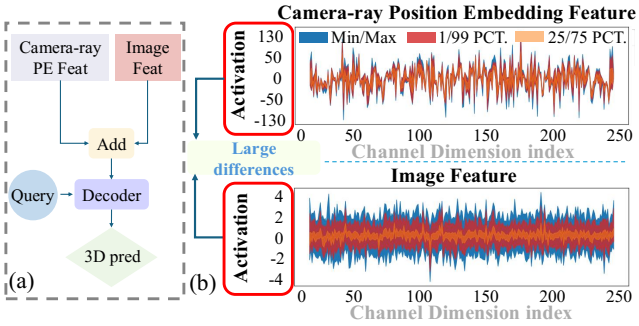

Figure 1: (a) Illustration of PETR pipeline. The camera-ray position embedding and image features are fused via element-wise addition, and then combined with object queries through a Transformer decoder to generate 3D object predictions. (b) The activation values of camera-ray position embedding feature range form -130 to 130, where image feature are primarily centered around 0.

Quantization (Yang et al. 2024) is an effective way to compress deep networks by reducing the bit width of weights and activations. However, PETRs present two quantization challenges that differ from conventional vision models. **First, modality disparity in feature fusion.** As shown in Fig. 1b, camera-ray positional embeddings span a large dynamic range (± 130), nearly two orders of magnitude wider than image features (± 4). When fused by element-wise addition, the quantization scale is dominated by positional embeddings, compressing image features into only 3–5 effective integer bins and discarding over 90% of their

*Corresponding author.

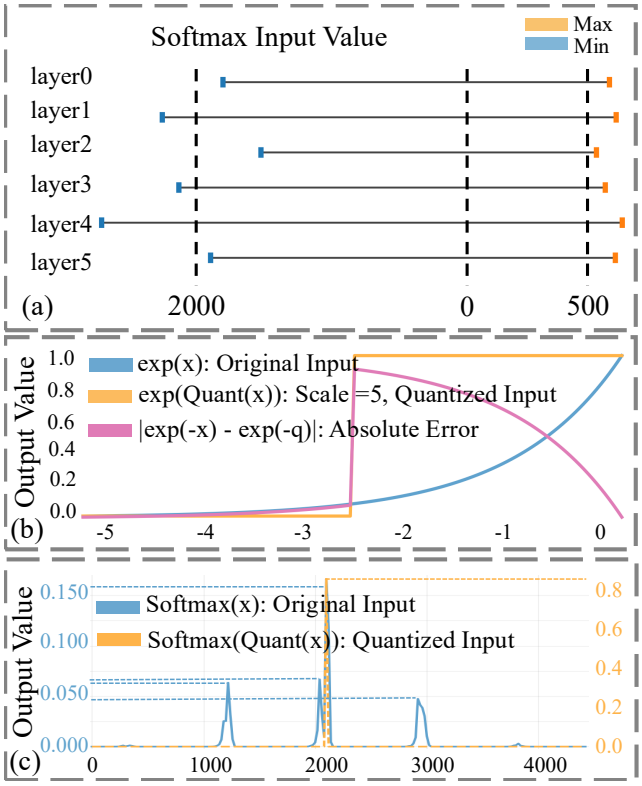

Figure 2: (a) PETR pipeline. Camera-ray position embeddings are added to image features and then fused with queries via a Transformer decoder to produce 3D object predictions. (b) Camera-ray positional embeddings exhibit a large dynamic range (-130 to 130), while image features are centered near zero.

information, resulting in up to 20.8% mAP degradation.

Second, non-linear operators introduce dual difficulties. *Challenge 1 — Attention distortion.* Inputs to operators such as Softmax exhibit extremely wide ranges (Fig. 2a); quantizing them causes large exponential errors (Fig. 2b) and severe attention shifts (Fig. 2c). *Challenge 2 — Hardware inefficiency.* Softmax and related non-linear functions are costly across platforms: (1) GPU tensor cores deliver 3–5× lower throughput on Softmax than GEMM (Dao 2023); (2) LUT-based approximation (Wang, Liu, and Foroosh 2018) suffers from exponential memory growth for linear LUTs or complex comparators for non-uniform LUTs (Yu et al. 2022); (3) software approximations (Kim et al. 2021; Li and Gu 2023) reduce precision to maintain deployability.

The co-existence of these bottlenecks—extreme modality disparity and non-linear computation overhead—creates a unique quantization challenge for PETRs that existing methods cannot address. To overcome this, we propose FQ-PETR, the first framework enabling fully integer inference for PETRs without sacrificing accuracy. Our co-design introduces three synergistic components:

Quantization-Friendly LiDAR-Ray PE (QFPE)

We redesign positional embedding from first principles. Inspired by LiDAR geometry, we sample a single 3D point per pixel along depth rays, removing multi-point interpolation and inverse-sigmoid distortion (Fig. 4b). Anchor-based embeddings with learnable bounds constrain the dynamic range and preserve spatial coherence, reducing PE magnitude by 4.4× and improving accuracy.

Dual-Lookup Table (DULUT)

We decouple non-uniform approximation from hardware constraints. By cascading two linear LUTs—the first functioning as a "nonlinear index mapper"—our method achieves exponential entry reduction (e.g., 32+32 entries for SiLU) while keeping the approximation error below 0.1%. DULUT relies only on standard LUT units, ensuring broad deployability across edge platforms.

Quantization After Numerical Stabilization (QANS)

We target Softmax quantization by applying integer conversion after logit subtraction (numerical stabilization), restricting values to $[-\beta, 0]$. Adaptive β selection reduces distribution shift, preserving attention behavior with 8-bit precision.

Comprehensive experiments across PETR, StreamPETR, PETRv2, and MV2d show that FQ-PETR achieves near-floating-point performance under W8A8 (less than 1% mAP/NDS degradation), with up to 75% latency reduction and 4× model compression—substantially outperforming existing PTQ/QAT baselines and enabling practical deployment of efficient 3D perception models.

Related Work

Multi-View 3D Object Detection. Surround-view 3D object detection is essential for autonomous driving and is typically categorized into LSS-based (Junjie et al. 2021; Yinhao et al. 2022; Junjie and Guan 2022) and transformer-based (Liu et al. 2022a; Shu et al. 2023) paradigms. LSS-based methods lift multi-camera features to dense BEV representations (Philion and Fidler 2020), but their high memory cost limits long-range perception efficiency. Transformer-based approaches exploit sparse queries for improved scalability. Among them, the PETR family has become highly influential: PETR (Liu et al. 2022a) maps 2D features to 3D space via positional embeddings; PETRv2 (Liu et al. 2022b) incorporates temporal feature indexing; and StreamPETR (Wang et al. 2023a) extends temporal query processing. Other works enhance efficiency by integrating 2D detection priors (Wang et al. 2023b; Chu et al. 2024), or by fusing vision and LiDAR signals as in CMT (Yan et al. 2023). Recent studies further refine PETR's positional embedding (Shu et al. 2023; Hou et al. 2024), and PETR has also been integrated into Omnidrive (Wang et al. 2024b) for improved large-model 3D perception.

Model Quantization. Quantization compresses models by converting weights and activations from floating-point to low-bit integer representations (Zhang et al. 2018; Banner,

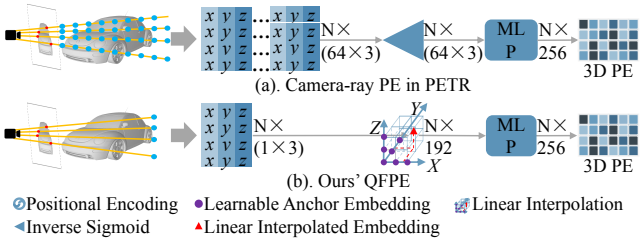

Figure 3: The overall architecture comparison of camera-ray PE, lidar-ray PE and our QFPE.

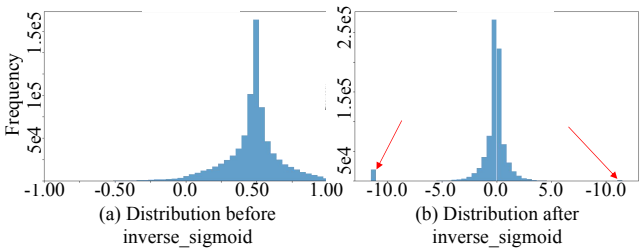

Figure 4: Distribution before and after inverse-sigmoid.

Nahshan, and Soudry 2019; Choukroun et al. 2019). Among existing methods (Wei et al. 2022; Jiang et al. 2025; Liu et al. 2024; Ashkboos et al. 2024a; Xu et al. 2025b; Chen et al. 2025; Xu et al. 2025a), we focus on uniform symmetric quantization, which maps a floating-point value x_f to a k-bit integer x_q:

$$x_q = \text{clamp}\left(\left\lfloor \frac{x_f}{s} \right\rceil, -2^{k-1}, 2^{k-1} - 1\right), \tag{1}$$

where the scale s is computed from calibration statistics as

$$s = \frac{x_f^{\max} - x_f^{\min}}{2^k}. \tag{2}$$

Quantization methods are commonly divided into Quantization-Aware Training (QAT) and Post-Training Quantization (PTQ). While QAT improves robustness at the cost of retraining (Bhalgat et al. 2020), PTQ enables efficient deployment but often degrades on transformer-based 3D detectors due to attention-heavy architectures.

For ViTs and transformer-based detectors, prior work explores PTQ, QAT, and rotation-based methods (Yuan et al. 2022; Li et al. 2023; Shi et al. 2024; Xu et al. 2023; Wang et al. 2024a; Xiao et al. 2023; Ashkboos et al. 2024b; Hu et al. 2025; Yu et al. 2025), primarily targeting GEMM-dominated computation. In contrast, nonlinear operators remain challenging; LUT-based and integer-only designs (Wang, Liu, and Foroosh 2018; Kim et al. 2021; Li and Gu 2023; Hu et al. 2024) improve hardware compatibility but face scalability or accuracy limitations.

Quantization for 3D Object Detection. Existing quantization methods for 3D detection mainly focus on CNN-based or LiDAR-centric pipelines. QD-BEV (Zhang et al. 2023) applies QAT and distillation to multi-camera 3D detection, while LIDAR-PTQ (Zhou et al. 2024) achieves near-FP32 performance on LiDAR-based *CenterPoint* (Yin, Zhou, and Krähenbühl 2020). However, PTQ solutions tailored to transformer-based 3D detection in autonomous driving remain largely unexplored.

Method

In this section, we propose a fully quantization framework called **FQ-PETR** specifically designed for PETR series models. First, we present a Quantization-Friendly LiDAR-ray position embedding (QFPE) to reduce the gap between the numerical range of 3D PE feature and image feature while maintaining floating-point accuracy. Second, we introduce a dual-lookup table (DULUT), which abstracts the index comparator required for nonlinear lookup tables into a nonlinear function, and uses a linear LUT to equal it. It maintains high approximation fidelity with fewer table entries without requiring special hardware support. Third, we propose quantization after numerical stabilization (QANS), which solves the problem of attention shift caused by quantization when the input range of Softmax is too large.

Quantization-Friendly LiDAR-ray Position Embedding (QFPE)

We first analyze the quantization failure from the perspective of magnitude imbalance, demonstrating that structural optimization of the model is essential for resolution. Based on this analysis, we present our QFPE design solution.

Magnitude Imbalance in PETR As evidenced in Fig. 1(b), a significant modality discrepancy exists: the camera-ray position embedding (Camera-ray PE) exhibits a dynamic range of approximately ±130, while multi-view image features are confined within ±4.

Crucially, when quantizing the element-wise summation of the above two features, regardless of the quantization method employed, the scale factor becomes dominated by the Camera-ray PE. This leads to severe compression of image features and consequent catastrophic accuracy degradation. Therefore, addressing this quantization challenge necessitates architectural modifications to the model.

Through systematic analysis of the Camera-Ray PE construction pipeline (shown in Fig. 3(a)) and Magnitude Propagation Analysis in Appendix A, we identify that the *inverse-sigmoid* operation amplifies the magnitude by approximately 11.5×. Furthermore, this operation distorts the originally balanced depth distribution (Fig. 4(a)), skewing it toward extreme outliers and thereby exacerbating the Camera-ray PE magnitude inflation.

QFPE Architecture. To simultaneously address both the magnitude imbalance and outliers challenges, we propose a **Q**uantization-**F**riendly LiDAR-ray **P**osition **E**mbedding (QFPE) that achieves significant magnitude reduction while maintaining computational efficiency. As illustrated in Fig. 3 (b), our design incorporates two key innovations:

1. **LiDAR-prior Guided Single-point Sampling**. Leveraging the physical characteristics of LiDAR sensors, we implement a sparse sampling strategy that selects only one

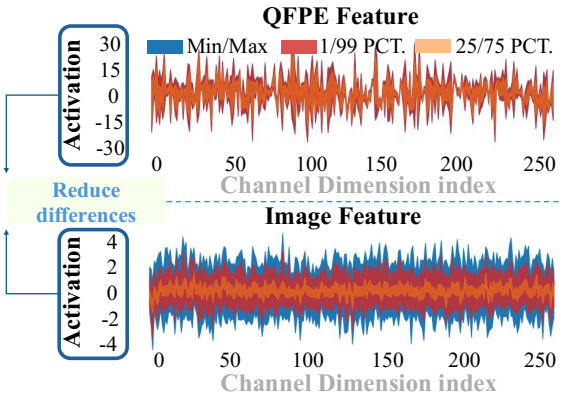

Figure 5: The activation values of QFPE feature range form -29.7 to 29.7.

3D point per pixel along each depth ray (Fig. 3(b)), contrasting with the multi-sample approach in conventional camera-ray PE. This design eliminates the need for iterative inverse-sigmoid transformations, substantially reducing embedding variance.

2. **Anchor-based Constrained Embedding with Convex Combination**. We establish three axis-aligned anchor embeddings $\{E_\alpha^i\}_{i=1}^3$ for each spatial dimension $\alpha \in \{x, y, z\}$, accompanied by corresponding anchor locations $\{L_\alpha^i\}_{i=1}^3$. For any LiDAR-sampled 3D point (x_j, y_j, z_j), we compute axis-specific embeddings through linear interpolation between adjacent anchors:

$$e_\alpha^j = \frac{p_\alpha - L_\alpha^{i_\alpha}}{L_\alpha^{i_\alpha+1} - L_\alpha^{i_\alpha}} E_\alpha^{i_\alpha+1} + \frac{L_\alpha^{i_\alpha+1} - p_\alpha}{L_\alpha^{i_\alpha+1} - L_\alpha^{i_\alpha}} E_\alpha^{i_\alpha} \quad (3)$$

where p_α denotes the coordinate along axis α. The final positional embedding vector is generated by concatenating these axis-wise embeddings and processing them through a lightweight MLP.

These two innovations ensure our QFPE remains both bounded in magnitude and free of difficult-to-quantize nonlinearities. Fig. 5 shows that the dynamic range of our QFPE (± 29.7) is only marginally wider than that of standard image features (± 4)—in stark contrast to PETR's original (± 127.3). The proposed design eliminates complex nonlinear operations (inverse-sigmoid), achieving hardware-compatible computation without compromising geometric fidelity.

DULUT for Non-linear Functions

Limitations of linear-LUT and NN-LUT. Linear-LUT requires exponentially more entries as input bit-width increases, quickly exhausting on-chip SRAM. Meanwhile, NN-LUT leverages neural networks to search non-uniform intervals, concentrating entries in high-curvature regions. However, NN-LUT heavily depends on training data distribution and random initialization, requires hardware-specific comparator trees, and only achieves real acceleration with

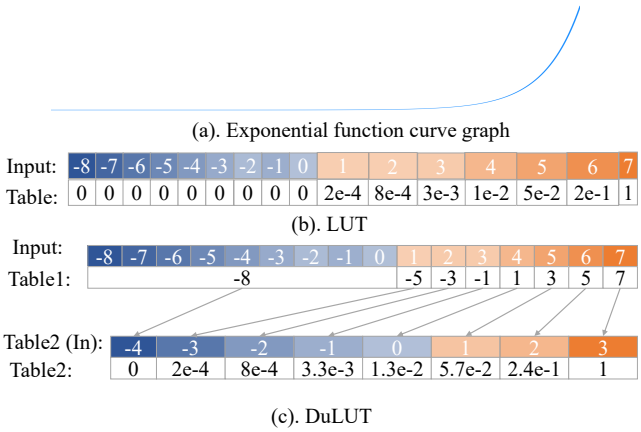

Figure 6: INT4 example for $\exp(\cdot)$ with DULUT. A 16-entry linear LUT is replaced by two 8-entry tables: (i) an index-mapping table that compresses near-flat input ranges, and (ii) a value table for interpolation. Precision matches the 16-entry LUT with fewer entries.

dedicated hardware support. Moreover, the non-uniform intervals complicate fusion and parallel optimization in existing compiler frameworks.

Proposed DULUT Algorithm. To overcome these limitations, we first establish a theoretical error bound for linear interpolation. For a twice-differentiable function $f(x)$ interpolated linearly over an interval $[x_i, x_{i+1}]$, the maximum interpolation error is bounded by:

$$\max_{x\in[x_i,x_{i+1}]} |f(x) - P(x)| \le \frac{(x_{i+1} - x_i)^2}{8} \max_{x\in[x_i,x_{i+1}]} |f''(x)| + \varepsilon_{\text{hw}}, \quad (4)$$

where $f''(x)$ denotes the second derivative (curvature), and ε_{hw} represents hardware-induced error from finite interpolation resolution and rounding. Hence, intervals must be shortened in regions with large curvature ("enlarge" regions), while flatter or near-saturated areas with lower curvature can be merged into fewer intervals ("shrink" regions).

Instead of designing hardware-based non-uniform indexing comparators, DULUT treats the nonlinear comparator itself as a nonlinear function $g(x)$ and approximates it with an additional linear LUT. Thus, the indexing operation reduces to a linear LUT lookup, and the overall problem simplifies to identifying the optimal nonlinear function $g(x)$. Ideally, without hardware error constraints, one could approximate $g(x)$ directly from curvature information. However, due to hardware limitations (e.g., fixed interpolation step size), using curvature to merge intervals can introduce significant approximation errors (more details see Appendix B).

To address this issue, we propose an iterative optimization algorithm accounting explicitly for hardware constraints. We initialize two cascaded linear LUTs equivalently to a single linear LUT. Then, we evaluate the average relative error (ARE) for each interval and iteratively merge intervals with minimal ARE while subdividing intervals with maximal ARE. LUT parameters are updated accordingly if this

Algorithm 1: Iterative Optimization Algorithm for DULUT

Require: Target nonlinear function $f(x)$ (FP32); quantization bit-width b; hardware interpolation resolution k (integer points per segment); initial LUT sizes m_1, m_2; error tolerance δ
Ensure: Optimized integer LUTs table_1, table_2, and corresponding quantization parameters (scales and zero-points)
Initialization: Uniformly partition integer domain $\mathcal{X} = [-2^{b-1}, 2^{b-1}-1]$ into m_1 intervals to construct initial table_1. Set $\text{table}_2[i] = f(i)$ and quantize according to hardware resolution k.
repeat
  Compute approximation $\hat{f}(x) = \text{table}_2[\text{table}_1[x]]$ for all $x \in \mathcal{X}$ (or dense samples).
  Calculate average relative error (ARE) within each interval.
  Identify interval $j_{\min}$ with minimal ARE and interval $j_{\max}$ with maximal ARE.
  Merge interval $j_{\min}$ with its neighbor (*shrink*), releasing one interval.
  Split interval $j_{\max}$ into two equal subintervals (*enlarge*), consuming the released interval.
  Update and requantize both LUTs; recompute global maximum ARE.
until global maximum ARE $\leq \delta$ or no further improvement achievable

Algorithm 2: Pseudo-code of DULUT with Triton

Require: Quantized input i_q (i_{bit}-bit signed); pre-computed tables $\mathbf{T}_1$ ($2^{t_{\text{bit1}}}+1$ slots) and $\mathbf{T}_2$ ($2^{t_{\text{bit2}}}+1$ slots)
Ensure: Quantized output o_q
Function LUTKERNEL(i_q, table, t_{bit}, i_{bit})
  $shift_{\text{bit}} \leftarrow i_{\text{bit}} - t_{\text{bit}}$; $shift_{\text{num}} \leftarrow 1 \ll shift_{\text{bit}}$
  $q_{\min}, q_{\max} \leftarrow -(1 \ll (i_{\text{bit}}-1)), (1 \ll (i_{\text{bit}}-1))-1$
  for each index $offset$ **in parallel do**
    $x \leftarrow i_q[offset] + (1 \ll (i_{\text{bit}}-1))$ {signed $\rightarrow$ unsigned}
    $idx \leftarrow x \gg shift_{\text{bit}}$; $p \leftarrow x \bmod shift_{\text{num}}$
    $t_l \leftarrow \text{table}[idx]$; $t_r \leftarrow \text{table}[idx+1]$
    $S \leftarrow (shift_{\text{num}} - p)t_l + pt_r$
    $S \leftarrow (S + (1 \ll (shift_{\text{bit}}-1))) \gg shift_{\text{bit}}$
    $o_q[offset] \leftarrow \text{clip}(S, q_{\min}, q_{\max})$
  end for
  return o_q
Offline table construction: Use Algorithm 1.
Online inference:
  $idx \leftarrow$ LUTKERNEL(i_q, $\mathbf{T}_1$, t_{bit1}, i_{bit})
  $o_q \leftarrow$ LUTKERNEL(idx, $\mathbf{T}_2$, t_{bit2}, i_{bit})
return o_q

operation reduces the global error. This optimization continues until no further error reduction can be achieved. The complete procedure is described in Algorithm 1.

Empirically, DULUT achieves precision comparable to much larger single-table LUT implementations while significantly reducing SRAM overhead. In Fig. 6, we illustrate an INT4 case for $\exp(\cdot)$. A linear LUT requires 16 entries to store the outputs for all 16 input codes. DULUT uses two 8-entry linear tables instead. The first table acts as an index mapper: it merges near-flat input ranges where exp varies little and allocates more indices to high-curvature ranges. The second table stores the values and performs linear interpolation. This preserves the precision of the 16-entry linear LUT while using fewer total entries.

To facilitate reproduction, we give the entire process of DULUT and Triton implementation in Algorithm 2.

Quantization After Numerical Stabilization

To solves the problem of attention shift caused by quantization when the input range of Softmax is too large (see Figure2, we propose quantization After Numerical Stabilization (QANS) (Fig. 7), and adaptively determining the optimal truncation lower bound to minimize softmax error.

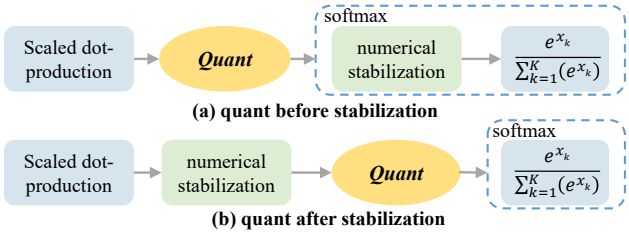

Figure 7: Illustration for quant before/after stabilization.

After NS, inputs for softmax are non-positive. Values below -20 approach zero after exponentiation, so we define a candidate set of scaling factors $S = s_1, s_2, ..., s_N$ with $s_i = \frac{i}{2^{k-1}}$ for k-bit quantization. The dequantized input is:

$$\hat{x}_s^i = s_i \cdot \text{clamp}\left(\text{round}\left(\frac{x_s}{s_i}\right), -2^{k-1}, 2^{k-1}-1\right) \quad (5)$$

ensuring $\hat{x}_s^i \in [-i, 0]$. We compute the softmax distributions $p_f = \text{softmax}(x_s)$ and $p_q^i = \text{softmax}(\hat{x}s^i)$, and select the optimal scaling factor $s\hat{i}$ that minimizes the error:

$$\hat{i} = \underset{i}{\text{argmin}} |p_f - p_q^i|, \quad i = 1, 2, ..., N. \quad (6)$$

Experiments

Validation on PETR and Its Variants

We evaluate our method comprehensively on PETR and its variants, covering both single-frame (PETR) and temporal multi-frame (PETRV2, MV2D, StreamPETR) settings, under floating-point (FP) and quantized conditions (see Tab. 1). **Firstly**, we examine improvements in floating-point performance. Our method consistently improves mAP (ranging from 0.07 to 1.08 points) and significantly improves NDS (from 0.61 to 1.09 points) for both single-frame PETR models and multi-frame PETRV2, MV2D, and StreamPETR. **Secondly**, we analyze the quantization performance. For single-frame PETR models and temporal models (StreamPETR, PETRV2, MV2d), our method effectively constrains accuracy degradation to less than 1% in both mAP and NDS metrics, thanks to the proposed QFPE, DULUT and QANS.

Ablation Study

Ablation for different quantization methods. We evaluate the Camera-ray PE module on the nuScenes dataset un-

Backbone	resolution	Feat	Model	Method	mAP↑	NDS↑	mATE↓	mASE↓	mAOE↓	mAVE↓	mAAE↓
R50	512 × 1408	c5	PETR	FP32	31.42	36.11	84.19	28.42	60.69	99.08	23.58
				SQ	20.67	29.32	107.94	31.47	75.22	121.71	25.45
				Quarot	22.81	30.00	104.87	30.99	74.16	118.54	25.07
			QF-PETR	FP32	**31.49**	**37.20**	**82.52**	**27.88**	**59.91**	**91.74**	**23.45**
				SQ*	**31.34**	**37.17**	**82.61**	**27.93**	**60.00**	**91.79**	**23.45**
				Quarot*	**31.46**	**37.19**	**82.52**	**27.89**	**59.90**	**91.77**	**23.45**
V2-99	640 × 1600	p4	StreamPETR	FP32	49.51	58.03	60.10	26.07	35.65	25.91	19.60
				SQ	18.72	35.66	74.32	30.39	41.49	30.53	20.82
				Quarot	19.97	37.49	71.28	30.11	40.16	30.23	20.76
			QF-StreamPETR	FP32	**50.48**	**58.61**	**58.78**	**26.16**	**37.05**	**25.69**	**18.59**
				SQ*	**49.44**	**57.94**	**60.30**	**26.55**	**37.07**	**25.87**	**18.59**
				Quarot*	**50.12**	**58.39**	**58.86**	**26.16**	**37.05**	**25.87**	**18.59**
R50	512 × 1408	p4	MV2d-T	FP32	45.30	54.30	61.70	26.50	38.80	38.50	17.90
				SQ	26.32	36.14	80.10	33.74	80.13	51.71	25.06
				Quarot	28.17	39.25	78.46	32.11	79.21	49.19	23.57
			QF-MV2d-T	FP32	**46.38**	**54.91**	**60.36**	**26.24**	**37.61**	**37.98**	**17.68**
				SQ*	**45.13**	**52.71**	**60.44**	**26.36**	**37.62**	**38.12**	**17.93**
				Quarot*	**46.25**	**54.16**	**60.35**	**26.24**	**37.61**	**37.98**	**17.68**
V2-99	320 × 800	p4	PETRv2	FP32	41.00	50.30	72.26	26.92	45.29	38.93	19.33
				SQ	15.12	31.04	96.14	30.11	45.71	52.14	27.11
				Quarot	19.20	34.15	95.23	29.76	45.65	50.46	26.26
			QF-PETRv2	FP32	**41.86**	**51.03**	**71.03**	**26.15**	**44.86**	**37.54**	**19.04**
				SQ*	**40.73**	**50.61**	**71.76**	**27.02**	**45.24**	**37.97**	**19.42**
				Quarot*	**41.69**	**50.94**	**71.03**	**26.15**	**44.86**	**37.54**	**19.04**

Table 1: Comparison of floating-point and quantized performance on standard PETR-series models (Wang et al. 2023a; Liu et al. 2022a,b). The default quantization setting is full-integer INT8 quantization. The DULUT configuration uses 32+32 entries. 'SQ' denotes SmoothQuant (Xiao et al. 2023), and 'Quarot' denotes rotation ptq methods (Ashkboos et al. 2024b). An asterisk (*) indicates that Quantization-Aware Nonlinear Scaling (QANS) is additionally applied.

der three configurations: FP32 Baseline (full precision as an upper bound), 8-bit PTQ Methods SmoothQuant (Xiao et al. 2023) and Quarot (Ashkboos et al. 2024b). As shown in Table 2, retaining the Softmax input in full precision ("No") yields higher mAP and NDS than when it is quantized ("Yes"), underscoring the importance of careful Softmax treatment.

Method	Quant.Softmax Input	(SQ) INT8		(Quarot) INT8	
		mAP↑	NDS↑	mAP↑	NDS↑
Camera-ray PE	FP32	31.42	36.11	31.42	36.11
	No	24.90	32.10	27.10	33.60
	Yes	20.67	29.32	22.81	30.00

Table 2: Quantization performance of Camera-ray PE on nuScenes. FP32, 8-bit SmoothQuant and Quarot methods are compared under different Softmax quantization settings.

Effect of Anchor Embedding Quantity. The QFPE uses three anchor embeddings per axis, obtained through linear interpolation. Experiments (Tab. 3) demonstrate that setting the number of anchor embeddings to 3 achieves the highest NDS and mAP scores. Adjusting this number either up or down results in lower performance, confirming that 3 is the optimal choice.

Quantity of Anchor Embedding			NDS↑	mAP↑
x-axis	y-axis	z-axis		
2	2	2	36.66	31.29
3	3	3	**37.20**	**31.49**
4	4	4	36.92	31.09
5	5	5	36.83	31.19

Table 3: Effect of Anchor Embedding Quantity.

Proof of Position embedding Equivalence. We conducted experiments to verify whether the proposed QFPE enhances floating-point performance over the original camera-ray PE. As shown in Tab. 4, QFPE provides performance improvements. On PETR, it slightly increases mAP by 0.07 but significantly boosts NDS and mATE by 1.09 and 1.67, respectively. For Stream-PETR, our method yields substantial and balanced enhancements, with increases of 0.94 in mAP, 0.46 in NDS, and 0.22 in mATE.

Local Similarity of PE Features. Figure 8 shows that QFPE significantly outperforms 3D point PE and cameraray PE in local similarity of position embedding. Its similarity distribution appears more compact and concentrated, validating the method's superiority in local spatial information

Method		mAP↑	NDS↑	mATE↓
PETR	Camera-ray PE	31.42	36.11	84.19
	QFPE	**31.49**	**37.20**	**82.52**
Stream -PETR	Camera-ray PE	49.51	58.03	60.10
	QFPE	**50.48**	**58.61**	**58.78**

Table 4: FP Performance of different 3D PE.

modeling and its capability to precisely capture neighborhood spatial relationships around target pixels.

Compare with QAT. Although QFPE requires retraining, its cost remains comparable to QAT. We implement a distillation-based QAT (inspired by QD-BEV (Zhang et al. 2023)) on the original Camera-ray PE. As shown in Table 5, QFPE consistently outperforms QAT in both mAP and NDS metrics, despite QAT's longer training epochs (24 or 36 epochs). This demonstrates the advantage of our amplitude-aware design in preserving floating-point performance and improving quantized accuracy.

Epochs	QAT (Distill)		QFPE (PTQ)	
	mAP↑	NDS↑	mAP↑	NDS↑
12	28.9	35.2	**31.46**	**37.19**
24	30.3	35.8	**31.46**	**37.19**
36	30.3	35.8	**31.46**	**37.19**

Table 5: Comparison of QAT with distillation vs. QFPE PTQ across training epochs on the nuScenes dataset.

Impact of Different Scaled Dot Product Quantization Strategy. To isolate the effect of our scaled dot-product quantization, we quantize only the softmax inputs and keep all other modules in FP. As shown in Tab. 6, the original strategy causes large drops (about 40% NDS and 50% mAP). Our "quant after stabilization" improves accuracy. Ablating the truncation range N shows that $N \geq 20$ is near-lossless, $N < 20$ harms accuracy due to over-truncation, and larger N brings no further gain. We therefore set $N = 20$.

Superior Performance of DULUT for Non-linear Functions. Using "quant after stabilization (N=20)" as the baseline (Tab. 6), we compare I-BERT, I-ViT, standard linear LUTs, and our DULUT (Tab. 7). A linear LUT requires 256 entries for lossless accuracy; with 128 entries it drops by 0.54 NDS and 0.37 mAP. DULUT achieves lossless accuracy with 128 entries and remains near-lossless with 64 entries (-0.08% NDS, -0.02% mAP), confirming its efficiency for SiLU, GELU, and Softmax.

Figure 8: Qualitative comparison of the local similarity.

Method		NDS↑	mAP↑	mATE↓
quant before ns	-	25.31	13.79	107.94
quant after ns	N = 1	3.45	1.23	150.34
	N = 5	33.86	28.77	87.12
	N = 10	34.65	29.33	85.01
	N = 20	36.10	31.42	84.19
	N = 30	36.10	31.42	84.19
	N = 40	36.10	31.42	84.19

Table 6: Performance of Different Scaled Dot Product Quantization Strategies.

Method		NDS↑	mAP↑	mATE↓
Quant after stabilization (N=20)		36.10	31.42	84.19
I-Bert		34.87	29.34	88.41
I-Vit		35.03	28.77	87.32
LUT	256 entries	36.10	31.42	84.19
	128 entries	35.56	31.05	85.61
DULUT	(16,16) entries	28.12	17.36	96.99
	(16,32) entries	34.14	27.29	90.33
	(32,32) entries	36.07	31.36	84.20
	(64,64) entries	36.10	31.42	84.19

Table 7: Performance comparison of different quantization methods for nonlinear functions.

Practical Hardware Resource Savings. Tab. 8 shows Q-PETR runs at 27.6 FPS (3.9× faster) and 1.3 GB memory (75% less) vs. PETR's 7.1 FPS/4.8 GB, demonstrating significant speedup and resource efficiency.

Method	Mode	FPS	CUDA Memory (G)
PETR	FP32	7.1	4.8
FQ-PETR	INT8	27.6	1.3

Table 8: FPS and CUDA Memory Comparison: PETR vs. Q-PETR (R50-DCN, 512×1408, RTX 4090).

Conclusion

We introduce FQ-PETR, a fully quantized framework specifically designed for the efficient deployment of PETRs on edge hardware. By addressing the magnitude imbalance between positional embeddings and image features, our proposed QFPE significantly reduces quantization difficulty without sacrificing accuracy. We further develop DULUT, a hardware-friendly algorithm for accurately approximating nonlinear functions with minimal resources. Additionally, we propose QANS, effectively mitigating the attention distortion issue from quantizing extreme softmax inputs. Extensive experiments confirm that FQ-PETR achieves near-floating-point accuracy with substantially improved inference efficiency, demonstrating its practical value for real-world autonomous driving or computer vision applications.

References

Ashkboos, S.; Croci, M. L.; Nascimento, M. G. d.; Hoefler, T.; and Hensman, J. 2024a. Slicegpt: Compress large language models by deleting rows and columns. *arXiv preprint arXiv:2401.15024*.

Ashkboos, S.; Mohtashami, A.; Croci, M. L.; Li, B.; Cameron, P.; Jaggi, M.; Alistarh, D.; Hoefler, T.; and Hensman, J. 2024b. Quarot: Outlier-free 4-bit inference in rotated llms. *arXiv preprint arXiv:2404.00456*.

Banner, R.; Nahshan, Y.; and Soudry, D. 2019. Post training 4-bit quantization of convolutional networks for rapid-deployment. In Wallach, H. M.; Larochelle, H.; Beygelzimer, A.; d'Alché-Buc, F.; Fox, E. B.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, 7948–7956.

Bhalgat, Y.; Lee, J.; Nagel, M.; Blankevoort, T.; and Kwak, N. 2020. Lsq+: Improving low-bit quantization through learnable offsets and better initialization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, 696–697.

Chen, Z.; Hu, X.; Yang, D.; Xu, Z.; Xu, C.; Yuan, Z.; Zhou, S.; and JiangyongYu. 2025. MoEQuant: Enhancing Quantization for Mixture-of-Experts Large Language Models via Expert-Balanced Sampling and Affinity Guidance. In *Forty-second International Conference on Machine Learning*.

Choukroun, Y.; Kravchik, E.; Yang, F.; and Kisilev, P. 2019. Low-bit Quantization of Neural Networks for Efficient Inference. In *2019 IEEE/CVF International Conference on Computer Vision Workshops, ICCV Workshops 2019, Seoul, Korea (South), October 27-28, 2019*, 3009–3018. IEEE.

Chu, X.; Deng, J.; You, G.; Duan, Y.; Li, Y.; and Zhang, Y. 2024. RayFormer: Improving Query-Based Multi-Camera 3D Object Detection via Ray-Centric Strategies. *arXiv preprint arXiv:2407.14923*.

Dao, T. 2023. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*.

Hou, J.; Wang, T.; Ye, X.; Liu, Z.; Gong, S.; Tan, X.; Ding, E.; Wang, J.; and Bai, X. 2024. OPEN: Object-wise Position Embedding for Multi-view 3D Object Detection. *arXiv preprint arXiv:2407.10753*.

Hu, X.; Cheng, Y.; Yang, D.; Xu, Z.; Yuan, Z.; Yu, J.; Xu, C.; Jiang, Z.; and Zhou, S. 2025. Ostquant: Refining large language model quantization with orthogonal and scaling transformations for better distribution fitting. *arXiv preprint arXiv:2501.13987*.

Hu, X.; Cheng, Y.; Yang, D.; Yuan, Z.; Yu, J.; Xu, C.; and Zhou, S. 2024. I-llm: Efficient integer-only inference for fully-quantized low-bit large language models. *arXiv preprint arXiv:2405.17849*.

Jiang, X.; Yang, H.; Zhu, K.; Qiu, X.; Zhao, S.; and Zhou, S. 2025. PTQ4RIS: Post-Training Quantization for Referring Image Segmentation. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, 12663–12670.

Junjie, H.; and Guan, H. 2022. Bevdet4d: Exploit temporal cues in multi-camera 3d object detection. *arXiv preprint arXiv:2203.17054*.

Junjie, H.; Guan, H.; Zheng, Z.; and Dalong, D. 2021. Bevdet: High-performance multi-camera 3d object detection in bird-eye-view. *arXiv preprint arXiv:2112.11790*.

Kim, S.; Gholami, A.; Yao, Z.; Mahoney, M. W.; and Keutzer, K. 2021. I-bert: Integer-only bert quantization. In *International conference on machine learning*, 5506–5518. PMLR.

Li, Z.; and Gu, Q. 2023. I-vit: Integer-only quantization for efficient vision transformer inference. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 17065–17075.

Li, Z.; Xiao, J.; Yang, L.; and Gu, Q. 2023. Repq-vit: Scale reparameterization for post-training quantization of vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 17227–17236.

Liu, Y.; Wang, T.; Zhang, X.; and Sun, J. 2022a. Petr: Position embedding transformation for multi-view 3d object detection. *arXiv preprint arXiv:2203.05625*.

Liu, Y.; Yan, J.; Jia, F.; Li, S.; Gao, Q.; Wang, T.; Zhang, X.; and Sun, J. 2022b. Petrv2: A unified framework for 3d perception from multi-camera images. *arXiv preprint arXiv:2206.01256*.

Liu, Z.; Zhao, C.; Fedorov, I.; Soran, B.; Choudhary, D.; Krishnamoorthi, R.; Chandra, V.; Tian, Y.; and Blankevoort, T. 2024. SpinQuant–LLM quantization with learned rotations. *arXiv preprint arXiv:2405.16406*.

Philion, J.; and Fidler, S. 2020. Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In *ECCV*.

Shi, H.; Cheng, X.; Mao, W.; and Wang, Z. 2024. P^2-ViT: Power-of-Two Post-Training Quantization and Acceleration for Fully Quantized Vision Transformer. *arXiv preprint arXiv:2405.19915*.

Shu, C.; Deng, J.; Yu, F.; and Liu, Y. 2023. 3DPPE: 3D Point Positional Encoding for Multi-Camera 3D Object Detection Transformers. *arXiv preprint arXiv:2211.14710*.

Wang, M.; Liu, B.; and Foroosh, H. 2018. Look-up table unit activation function for deep convolutional neural networks. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 1225–1233. IEEE.

Wang, R.; Sun, H.; Yang, L.; Lin, S.; Liu, C.; Gao, Y.; Hu, Y.; and Zhang, B. 2024a. AQ-DETR: Low-Bit Quantized Detection Transformer with Auxiliary Queries. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 15598–15606.

Wang, S.; Liu, Y.; Wang, T.; Li, Y.; and Zhang, X. 2023a. Exploring object-centric temporal modeling for efficient multi-view 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 3621–3631.

Wang, S.; Yu, Z.; Jiang, X.; Lan, S.; Shi, M.; Chang, N.; Kautz, J.; Li, Y.; and Alvarez, J. M. 2024b. OmniDrive: A Holistic LLM-Agent Framework for Autonomous Driving

with 3D Perception, Reasoning and Planning. *arXiv preprint arXiv:2405.01533*.

Wang, Z.; Huang, Z.; Fu, J.; Wang, N.; and Liu, S. 2023b. Object as query: Lifting any 2d object detector to 3d detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 3791–3800.

Wei, X.; Zhang, Y.; Zhang, X.; Gong, R.; Zhang, S.; Zhang, Q.; Yu, F.; and Liu, X. 2022. Outlier suppression: Pushing the limit of low-bit transformer language models. *Advances in Neural Information Processing Systems*, 35: 17402–17414.

Xiao, G.; Lin, J.; Seznec, M.; Wu, H.; Demouth, J.; and Han, S. 2023. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, 38087–38099. PMLR.

Xu, C.; Yue, Y.; Xu, Z.; Hu, X.; JiangyongYu; Chen, Z.; Zhou, S.; Yuan, Z.; and Yang, D. 2025a. RWKVQuant: Quantizing the RWKV Family with Proxy Guided Hybrid of Scalar and Vector Quantization. In *Forty-second International Conference on Machine Learning*.

Xu, S.; Li, Y.; Lin, M.; Gao, P.; Guo, G.; Lü, J.; and Zhang, B. 2023. Q-detr: An efficient low-bit quantized detection transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 3842–3851.

Xu, Z.; Yue, Y.; Hu, X.; Yang, D.; Yuan, Z.; Jiang, Z.; Chen, Z.; JiangyongYu; Xu, C.; and Zhou, S. 2025b. MambaQuant: Quantizing the Mamba Family with Variance Aligned Rotation Methods. In *The Thirteenth International Conference on Learning Representations*.

Yan, J.; Liu, Y.; Sun, J.; Jia, F.; Li, S.; Wang, T.; and Zhang, X. 2023. Cross Modal Transformer via Coordinates Encoding for 3D Object Dectection. *arXiv preprint arXiv:2301.01283*.

Yang, D.; He, N.; Hu, X.; Yuan, Z.; Yu, J.; Xu, C.; and Jiang, Z. 2024. Post-training quantization for re-parameterization via coarse & fine weight splitting. *Journal of Systems Architecture*, 147: 103065.

Yin, T.; Zhou, X.; and Krähenbühl, P. 2020. Center-based 3D Object Detection and Tracking. *arXiv preprint arXiv:2006.11275*.

Yinhao, L.; Zheng, G.; Guanyi, Y.; Jinrong, Y.; Zengran, W.; Yukang, S.; Jianjian, S.; and Zeming, L. 2022. BEVDepth: Acquisition of Reliable Depth for Multi-view 3D Object Detection. *arXiv preprint arXiv:2206.10092*.

Yu, J.; Park, J.; Park, S.; Kim, M.; Lee, S.; Lee, D. H.; and Choi, J. 2022. NN-LUT: Neural approximation of non-linear operations for efficient transformer inference. In *Proceedings of the 59th ACM/IEEE Design Automation Conference*, 577–582.

Yu, J.; Zhou, S.; Yang, D.; Li, S.; Wang, S.; Hu, X.; Xu, C.; Xu, Z.; Shu, C.; and Yuan, Z. 2025. Mquant: Unleashing the inference potential of multimodal large language models via static quantization. In *Proceedings of the 33rd ACM International Conference on Multimedia*, 1783–1792.

Yuan, Z.; Xue, C.; Chen, Y.; Wu, Q.; and Sun, G. 2022. Ptq4vit: Post-training quantization for vision transformers with twin uniform quantization. In *European conference on computer vision*, 191–207. Springer.

Zhang, D.; Yang, J.; Ye, D.; and Hua, G. 2018. LQ-Nets: Learned Quantization for Highly Accurate and Compact Deep Neural Networks. In Ferrari, V.; Hebert, M.; Sminchisescu, C.; and Weiss, Y., eds., *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part VIII*, volume 11212 of *Lecture Notes in Computer Science*, 373–390. Springer.

Zhang, Y.; Dong, Z.; Yang, H.; Lu, M.; Tseng, C.-C.; Du, Y.; Keutzer, K.; Du, L.; and Zhang, S. 2023. QD-BEV: quantization-aware view-guided distillation for multi-view 3D object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 3825–3835.

Zhou, S.; Li, L.; Zhang, X.; Zhang, B.; Bai, S.; Sun, M.; Zhao, Z.; Lu, X.; and Chu, X. 2024. Lidar-ptq: Post-training quantization for point cloud 3d object detection. *arXiv preprint arXiv:2401.15865*.