Quaternion Approximate Networks for Enhanced Image Classification and Oriented Object Detection

Bryce Grant[1] and Peng Wang[2]

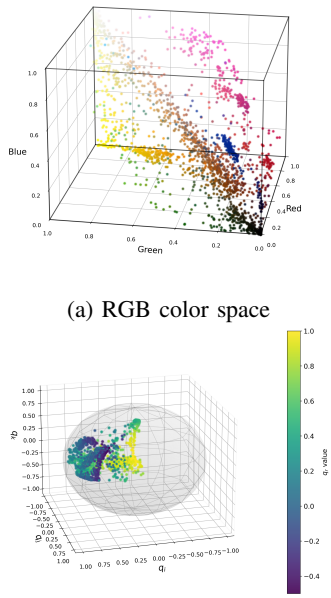
***Abstract*— This paper introduces Quaternion Approximate Networks (QUAN), a novel deep learning framework that leverages quaternion algebra for rotation equivariant image classification and object detection. Unlike conventional quaternion neural networks attempting to operate entirely in the quaternion domain, QUAN approximates quaternion convolution through Hamilton product decomposition using real-valued operations. This approach preserves geometric properties while enabling efficient implementation with custom CUDA kernels. We introduce Independent Quaternion Batch Normalization (IQBN) for training stability and extend quaternion operations to spatial attention mechanisms. QUAN is evaluated on image classification (CIFAR-10/100, ImageNet), object detection (COCO, DOTA), and robotic perception tasks. In classification tasks, QUAN achieves higher accuracy with fewer parameters and faster convergence compared to existing convolution and quaternion-based models. For objection detection, QUAN demonstrates improved parameter efficiency and rotation handling over standard Convolutional Neural Networks (CNNs) while establishing the SOTA for quaternion CNNs in this downstream task. These results highlight its potential for deployment in resource-constrained robotic systems requiring rotation-aware perception and application in other domains.**

***Keywords*—** Quaternion Networks, Oriented Object Detection, Robotic Perception, Spatial Intelligence

I. INTRODUCTION

Vision-based robotic assembly and manipulation tasks represent a critical challenge in modern robotics, which requires precise object detection, pose estimation, and spatial reasoning capabilities. Conventional computer vision approaches typically struggle with object orientation, a fundamental requirement for robotic interaction with real-world objects. Although deep learning has revolutionized object detection through frameworks such as YOLO [1], Faster R-CNN [2], and DETR [3], most approaches rely on axis-aligned bounding boxes that discard crucial orientation information. This limitation significantly affects performance in robotic applications where grasp planning and assembly operations are highly dependent on accurate object orientation [4].

Recent advances in vision-language models and self-supervised learning have transformed the computer vision landscape. DINO [5] and other contrastive learning approaches [6] have demonstrated improved performance by learning from unlabeled data, while Mixture-of-Experts (MoE) architectures [7] achieve state-of-the-art results through adaptive computation pathways. However, these sophisticated models come with substantial computational requirements, often containing hundreds of millions of parameters. Despite their impressive general-purpose capabilities, they lack the inherent mathematical structures to efficiently represent and process rotational information, a critical requirement for robotic manipulation tasks.



(a) RGB color space

(b) Quaternion Space

Fig. 1: Poincaré mapping from RGB color space to quaternion space.

Meanwhile, specialized approaches for oriented object detection [8], [9] have emerged to address rotation-aware perception needs. These methods typically use complex architectures and loss functions to estimate object orientation, but they often require substantial computational resources. This computational burden presents a significant barrier for deployment in resource-constrained robotic systems that require real-time performance.

Quaternion algebra offers a powerful mathematical framework for encoding and manipulating three-dimensional rotations [10], [11]. Originally developed by Hamilton in 1843, quaternions avoid the singularities associated with Euler angles (gimbal lock) and provide a more compact repre-

The author from [1] is with the Department of Electrical, Systems and Computer Engineering; Case Western Reserve University [2] is with Department of Mechanical and Aerospace Engineering, Case Western Reserve University, Cleveland, Ohio, USA

E-Mail: bag100@case.edu and pxw206@case.edu

Code available at: https://cwru-aism.github.io/QUANpaper/

This work is supported by the National Science Foundation under grant No. 2024614.

sentation compared to rotation matrices. This mathematical elegance has inspired researchers to explore quaternion-based deep learning approaches. Gaudet and Maida [12] established foundational concepts in deep quaternion networks, introducing quaternion backpropagation and initialization schemes for image classification, while Parcollet et al. [13] demonstrated the parameter efficiency of quaternion neural networks for audio processing tasks. Zhu et al. [14] further extended this to image denoising tasks, showing quaternion CNNs could achieve comparable performance to standard CNNs with significantly fewer parameters.

Despite these advances, existing quaternion neural networks face substantial implementation challenges. Pure quaternion networks require specialized mathematical operations that are poorly supported in mainstream deep learning frameworks, leading to significant computational overhead. Gomez and Gershenson [15] highlighted fundamental limitations such as the non-existence of bounded non-constant analytic quaternion functions, constraining the design of activation functions and complicating gradient computations. These practical limitations have prevented quaternion networks from being deployed in real-world robotic applications despite their theoretical advantages.

This paper introduces Quaternion Approximate Networks (QUAN), a novel approach that preserves the rotation equivariance benefits of quaternion representations while maintaining computational efficiency. Rather than performing operations in the quaternion domain directly, QUAN implements quaternion convolution through Hamilton product approximation using standard real-valued operations. This approach allows leveraging optimized tensor operations in existing deep learning frameworks while retaining the representational advantages of quaternion.

The key contributions are as follows:

- A quaternion approximation approach that implements Hamilton-based mixing using real-valued operations, maintaining rotation equivariance while significantly reducing computational complexity compared to previous quaternion neural networks.
- Novel quaternion-aware architectural components, including Independent Quaternion Batch Normalization (IQBN) and Quaternion Partial Spatial Attention (QC2PSA), that enhance model performance while preserving quaternion structure.
- State-of-the-art performance for quaternion networks on image classification tasks (CIFAR-10, CIFAR-100, ImageNet) with significantly reduced parameter counts compared to both standard CNNs and existing quaternion approaches.
- A systematic quaternion adaptation of YOLO architecture components (QC3k2, QSPPF, QC2PSA) for oriented bounding box detection, benchmarked on a custom robotic manipulation dataset.
- A novel orientation-aware loss function unifying quaternion angular metrics with traditional detection objectives, enabling end-to-end learning of both object location and orientation in quaternion space.

QUAN represents the first quaternion-inspired architecture that successfully bridges the gap between theoretical quaternion advantages and practical deployment needs for robotic perception systems. By combining parameter efficiency with rotation equivariance properties, this approach enables robust object detection and pose estimation for robotic manipulation tasks while maintaining suitable for real-time applications.

II. Related Works

A. General Object Detection and Pose Estimation

Object detection forms the foundation of many robotic vision systems. Landmark architectures like YOLO [1], Faster R-CNN [2], and DETR [3] have established new performance benchmarks but primarily focus on axis-aligned bounding boxes, which limit their utility for robotic manipulation tasks where orientation is crucial. For robotic applications, Wang et al. [4] developed DenseFusion, by fusing RGB and depth information, while Hodan et al. [16] introduced a benchmark for 6D object pose estimation in cluttered scenes while Kleeberger et al. [17] introduced a benchmark for industrial bin picking, highlighting the challenges of accurate orientation estimation in realistic settings. These approaches, while effective, often separate the detection and pose estimation steps rather than handling them in a unified framework.

Specialized methods for oriented object detection have emerged to address these limitations. Yang et al. [8] proposed R3Det, which refines feature maps for detecting rotated objects, while Xu et al. [9] introduced the gliding vertex mechanism to generate arbitrarily oriented bounding boxes. However, these approaches typically require complex architectural modifications and substantially increased computational resources compared to standard detectors.

B. Quaternion-Based Methods

Quaternions provide a compact and singularity-free representation for 3D rotations [10], [11]. Gaudet and Maida [12] pioneered quaternion neural networks by establishing fundamental operations, including weight initialization and backpropagation. Parcollet et al. [13] extended this work to recurrent networks and demonstrated parameter efficiency in speech processing tasks. Zhu et al. [14] applied quaternion convolutions to image processing, achieving comparable performance to standard CNNs with fewer parameters. Comminiello et al. [18] represented microphone in their spherical harmonics form, which enabled the use of quaternion networks for the detection of 3D sound events. Grassucci et al. [19] investigated quaternion generative adversarial networks, to obtain better FID scores than real-valued GANs. These works showed that quaternion CNNs could achieve comparable or even superior performance to standard CNNs with fewer parameters.

However, quaternion neural networks face significant implementation challenges. Gomez and Gershenson [15] identified limitations including higher computational complexity and the non-existence of bounded non-constant analytic quaternion functions, which constrains activation

function design. These practical challenges have prevented widespread adoption despite theoretical advantages.

C. Quaternion Pose Estimation

Rotation equivariance is particularly valuable for robotic perception. Hsu et al. [20] introduced QuatNet, using quaternions for continuous head pose estimation with a multi-regression loss function. He et al. [21] introduced PVN3D, a deep point-wise 3D keypoints voting network for 6DoF pose estimation using quaternions for rotation representation. Despite these advances, no previous work has successfully applied quaternion-based approaches to oriented object detection in practical robotic applications, which is the focus of this paper.

D. Alternative Rotation Equivariant Architectures

While our work focuses on quaternion-based approaches, several other frameworks achieve rotation equivariance through different mechanisms. Cohen and Welling [22] introduced Group Equivariant CNNs (G-CNNs) that encode symmetries using group theory, ensuring equivariance to discrete rotations. Weiler and Cesa [23] extended this with steerable CNNs using continuous groups, achieving equivariance to arbitrary rotations. Marcos et al. [24] proposed Rotation Equivariant Vector Field Networks that explicitly encode angles in the network architecture.

These approaches, require fundamental architectural changes and specialized implementations. Harmonic Networks [25] use circular harmonics to achieve rotation equivariance, but suffer from increased computational complexity. Oriented Response Networks [26] actively rotate filters during convolution, resulting in significant overhead.

In contrast, QUAN leverages quaternion algebra's inherent rotation properties within standard deep learning frameworks, achieving rotation awareness without the architectural complexity or computational burden of pure group-theory approaches.

III. Methodology

A. Quaternion Representation and Mapping

The quaternion number system extends complex numbers to four dimensions, represented as:

$$q = q_r + q_i\mathbf{i} + q_j\mathbf{j} + q_k\mathbf{k} \tag{1}$$

where $q_r, q_i, q_j, q_k \in \mathbb{R}$, and $\mathbf{i}, \mathbf{j}, \mathbf{k}$ are the fundamental quaternion units satisfying:

$$\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -1 \tag{2}$$

The Hamilton product between two quaternions $p = p_r + p_i\mathbf{i} + p_j\mathbf{j} + p_k\mathbf{k}$ and $q = q_r + q_i\mathbf{i} + q_j\mathbf{j} + q_k\mathbf{k}$ is defined as:

$$\begin{aligned} p \otimes q = &(p_r q_r - p_i q_i - p_j q_j - p_k q_k) \\ &+ (p_r q_i + p_i q_r + p_j q_k - p_k q_j)\mathbf{i} \\ &+ (p_r q_j - p_i q_k + p_j q_r + p_k q_i)\mathbf{j} \\ &+ (p_r q_k + p_i q_j - p_j q_i + p_k q_r)\mathbf{k} \end{aligned} \tag{3}$$

For image processing applications, RGB data must be mapped into quaternion space. Several mapping strategies have been explored, as explained in the Appendix subsection VI-A, with evaluation demonstrating that the Poincaré mapping [27], [28], which embeds the RGB cube within the unit quaternion sphere shown in Fig. 1, provided superior performance:

$$\begin{aligned} \mathbf{norm} &= \sqrt{r^2 + g^2 + b^2} \\ q_r &= \frac{1 - \mathbf{norm^2}}{1 + \mathbf{norm^2}} \\ q_i &= \frac{2r}{1 + \mathbf{norm^2}} \\ q_j &= \frac{2g}{1 + \mathbf{norm^2}} \\ q_k &= \frac{2b}{1 + \mathbf{norm^2}} \\ q &= [q_r, q_i, q_j, q_k] \end{aligned} \tag{4}$$

Poincaré mapping embeds RGB values into the into the Poincaré ball model of hyperbolic space, where the distance between points increases exponentially as they approach the boundary of the unit ball. This property allows the quaternion representation to better preserve hierarchical relationships in the color space compared to Euclidian embeddings.

Our approach is an approximation for two reasons: (1) we decompose the full quaternion convolution into separable operations rather than computing all 16 terms of the Hamilton product, reducing computational complexity from $\mathcal{O}(16CHW)$ to $\mathcal{O}(4CHW)$, and (2) we relax the unit quaternion constraint $\|q\| = 1$ during training, instead using Independent Quaternion Batch Normalization (IQBN) to maintain numerical stability. We found that enforcing strict constraints through Riemannian optimization [29] degraded performance, suggesting that our formulation better captures features while maintaining the quaternion structure.

B. Hamilton-Based Quaternion Approximation

Traditional quaternion convolution applies the Hamilton product between quaternion-valued kernels and features. For quaternion convolution with kernel W and input feature map X, where each quaternion has components (r, i, j, k), the full Hamilton product expansion is:

$$\begin{aligned} Y = &(W_r \otimes X_r - W_i \otimes X_i - W_j \otimes X_j - W_k \otimes X_k) \\ &+ (W_r \otimes X_i + W_i \otimes X_r + W_j \otimes X_k - W_k \otimes X_j)\mathbf{i} \\ &+ (W_r \otimes X_j - W_i \otimes X_k + W_j \otimes X_r + W_k \otimes X_i)\mathbf{j} \\ &+ (W_r \otimes X_k + W_i \otimes X_j - W_j \otimes X_i + W_k \otimes X_r)\mathbf{k} \end{aligned} \tag{5}$$

where $\otimes$ denotes standard convolution. The standard Hamilton matrix can be derived as:

$$\mathbf{W} \otimes \mathbf{X} = \begin{bmatrix} w_r & -w_i & -w_j & -w_k \\ w_i & w_r & -w_k & w_j \\ w_j & w_k & w_r & -w_i \\ w_k & -w_j & w_i & w_r \end{bmatrix} \begin{bmatrix} x_r \\ x_i \\ x_j \\ x_k \end{bmatrix} \tag{6}$$

To approximate the quaternion algebra, QUAN decomposes the left-hand operation into four separate convolutions,

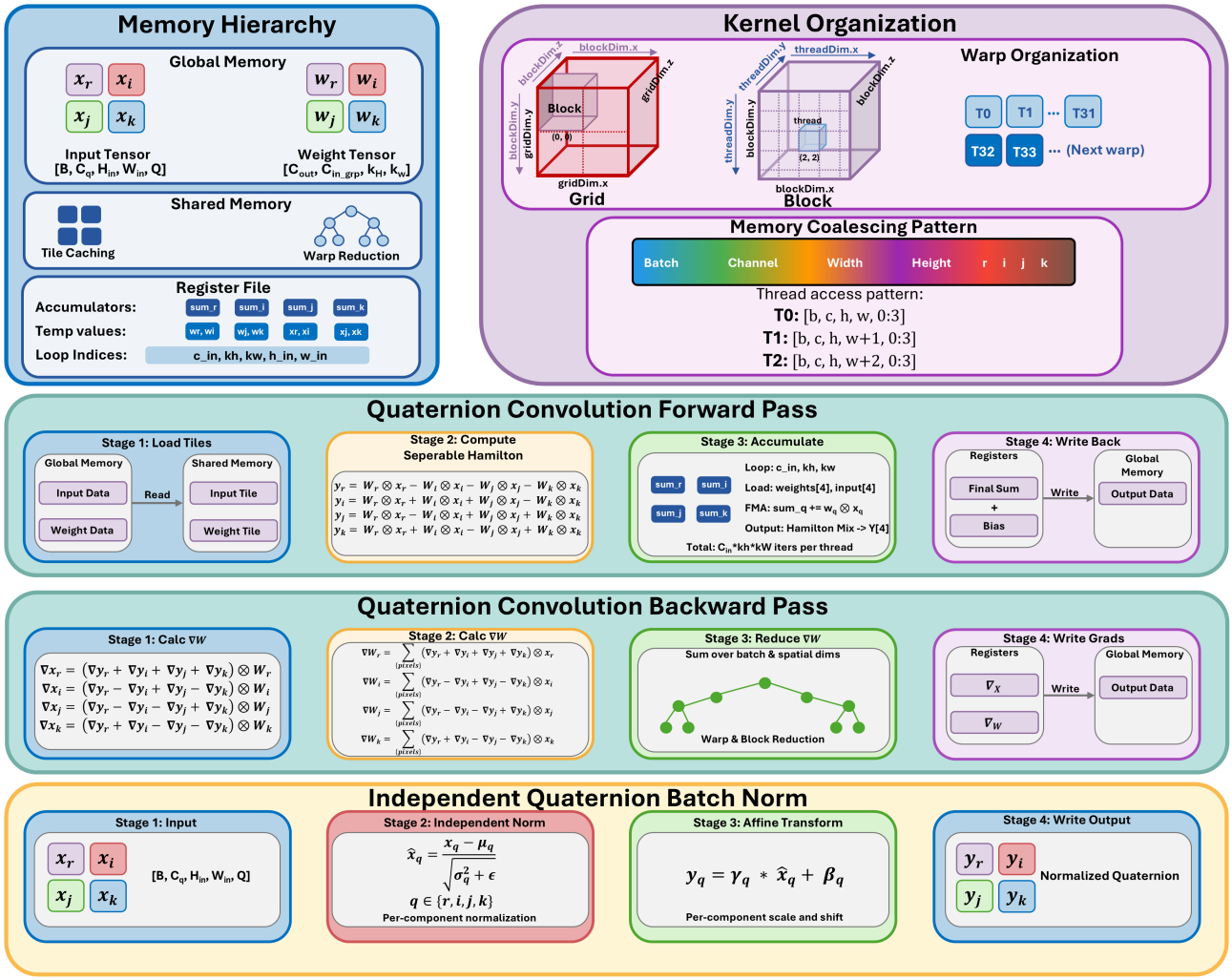


Fig. 2: CUDA kernel implementation for quaternion convolution

a method inspired by [30]. For an input tensor $X \in \mathbb{R}^{B\times C\times H\times W\times Q}$ and convolutional kernel W, the separable QUAN convolution forward pass is defined as:

$$\begin{aligned} y_r &= W_r \otimes x_r - W_i \otimes x_i - W_j \otimes x_j - W_k \otimes x_k \\ y_i &= W_r \otimes x_r + W_i \otimes x_i + W_j \otimes x_j - W_k \otimes x_k \\ y_j &= W_r \otimes x_r - W_i \otimes x_i + W_j \otimes x_j + W_k \otimes x_k \\ y_k &= W_r \otimes x_r + W_i \otimes x_i - W_j \otimes x_j + W_k \otimes x_k \end{aligned} \tag{7}$$

This separable formulation represents a valid simplification of the full quaternion convolution. While it relaxes the cross-component interactions (e.g., $W_i \otimes X_j$) present in the complete Hamilton product, it preserves the correct sign patterns essential to quaternion algebra. Our formulation maintains mathematical consistency while achieving the complexity reduction mentioned in Section subsection III-A. Despite the simplification, our experiments demonstrate that this approach effectively captures rotation-aware features while leveraging optimized tensor operations in existing deep learning frameworks. To address computational efficiency, we developed custom CUDA kernels that fuse the Hamilton product operations shown in figure 2. The backward pass gradients use the same separable structure as the forward pass:

$$\frac{\partial L}{\partial x_q} = \sum_{q\in\{r,i,j,k\}} M_q^T \cdot \left(\frac{\partial L}{\partial y_q} \star W_q\right) \tag{8}$$

where $\mathbf{M}_q^T$ is the mixing matrix derived from the forward pass coefficients in Equation 5, Equation 6:

$$M = \begin{bmatrix} 1 & 1 & 1 & 1 \\ -1 & 1 & -1 & 1 \\ -1 & 1 & 1 & -1 \\ -1 & -1 & 1 & 1 \end{bmatrix} \tag{9}$$

Similarly, for weight gradients:

$$\frac{\partial L}{\partial W_q} = \sum_{q\in\{r,i,j,k\}} M_q^T \cdot \left(\frac{\partial L}{\partial y_q} \star x_q\right) \tag{10}$$

This fusion reduces memory bandwidth requirements and achieves comparable inference time with standard convolutions.

C. Independent Quaternion Batch Normalization

Unlike previous quaternion normalization approaches that treat quaternions as unified entities, IQBN normalizes each component independently while preserving important cross-component relationships.

For quaternion feature $x = \{x_r, x_i, x_j, x_k\}$, with $B \times C \times H \times W \times Q$ shape, where $q \in \{r, i, j, k\}$ represents each quaternion component, IQBN computes:

$$\begin{aligned} \mu_q &= \mathbb{E}[X_q] \\ \sigma_q^2 &= \mathbb{E}[(X_q - \mu_q)^2] \\ \hat{x}_q &= \frac{x_q - \mu_q}{\sqrt{\sigma_q^2 + \epsilon}} \end{aligned} \tag{11}$$

During training, running statistics are computed and maintained:

$$\begin{aligned} \mu &= x.\text{mean}([0, 3]) \\ \sigma^2 &= x.\text{var}([0, 3]) \\ x_{\text{norm}} &= \frac{x - \mu.\text{view}(1, C, 1, 1, Q)}{\sqrt{\sigma^2.\text{view}(1, C, 1, 1, Q) + \epsilon}} \\ \text{Output} &= x_{\text{norm}} \cdot \gamma + \beta \end{aligned} \tag{12}$$

D. Quaternion Architectural Components

As a novel framework that integrates quaternion approximation with IQBN, QUAN can be leveraged to renovate most CNNs and quaternion CNNs, with necessary modifications to maintain quaternion operations throughout the network structure. In this section, YOLOv11 [1] is renovated with QUAN, and the following modifications, illustrated in Fig. Figure 3, were made besides replacing standard convolution with approximated quaternion convolution:

1) Quaternion Cross Stage Partial with kernel size 2 (QC3k2): The C3k2 module balances multi-scale features and gradient flow in YOLO architectures. The quaternion adaptation, QC3k2, maintains the partial cross-stage design while adapting convolutions to quaternion operations:

$$\text{QC3k2}(x) = \text{QConv}\left(\text{Concat}\left(\text{QBottleneck}(x_1), x_2\right)\right) \tag{13}$$

where x_1 and x_2 are the split components of the input feature map.

2) Quaternion Spatial Pyramid Pooling (QSPPF): SPPF efficiently handles multi-scale features through pyramid pooling. In QSPPF, the quaternion structure is preserved during pooling operations by applying pooling operations independently to each component followed by Hamilton-based mixing. This approach maintains rotation equivariance while capturing multi-scale information.

3) Quaternion Partial Spatial Attention (QC2PSA): C2PSA enhances the network's ability to focus on relevant spatial regions while maintaining quaternion structure. The module splits input features, applies quaternion attention to one branch, and recombines them. The quaternion attention mechanism computes spatial attention weights while respecting quaternion algebra, ensuring that rotational information is preserved.

E. Loss Functions for Object Detection and Orientation

For oriented object detection, QUAN employs specialized loss terms that account for both object localization, orientation, and classification:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{cls}} + \mathcal{L}_{\text{CIoU}} + \lambda_1 \mathcal{L}_{\text{angular}} + \lambda_2 \mathcal{L}_{\text{reg}} + \lambda_3 \mathcal{L}_{\text{smooth}} \tag{14}$$

where:

- $\mathcal{L}_{\text{cls}} = -\sum_{c=1}^{C} y_c \log(\hat{y}_c) + (1 - y_c) \log(1 - \hat{y}_c)$ - the classification loss
- $\mathcal{L}_{\text{CIoU}} = 1 - \text{IoU} + \frac{\rho^2(\mathbf{b}, \mathbf{b}^2 gt)}{c^2} + \alpha v$ - the complete IoU loss for bounding box regression
- $\mathcal{L}_{\text{angular}} = 2 \arccos(|\langle q_{\text{pred}}, q_{\text{target}} \rangle|)$ - the quaternion angular loss computes the geodesic angular distance between predicted and ground truth quaternion orientations, accounting for the double cover property
- $\mathcal{L}_{\text{reg}} = \|q\|_2 - 1^2$ - the quaternion regularization loss enforces unit quaternion constraints to maintain valid rotation representations
- $\mathcal{L}_{\text{smooth}} = \arccos(|\langle q_i, q_{i+1} \rangle|)$ - the orientation smoothness loss encourages consistency between sequential predictions to reduce jitter

This integrated loss formulation addresses the challenge of simultaneously optimizing detection accuracy and orientation precision, with the quaternion components designed to preserve the geometric constraints for rotation representation in 3D space.

IV. Experiments

We evaluate QUAN across two tasks: image classification and object detection. All experiments were conducted on a single NVIDIA RTX 4090 GPU. The experiments are designed to evaluate three key aspects: (1) parameter efficiency compared to standard CNNs and existing quaternion networks, (2) real-world applicability in robotic assembly tasks, and (3) rotation equivariance properties essential for robotic manipulation.

A. Datasets

1) Image Classification Datasets: We use CIFAR-10/100 [31] (60K 32×32 images), and ImageNet [32] subset (100 classes, 130K images).For **object detection**, we evaluate on COCO 2017 [33] (118K training, 5K validation), DOTA-v1.0 [34] (2,806 aerial images with 188k instances of oriented objects), and a custom robotic workspace dataset[1] (1,900 bids-eye view images with oriented and regular annotations) for pick-and place tasks.

[1] https://universe.roboflow.com/ukyaism/boxes_2-pwu7p

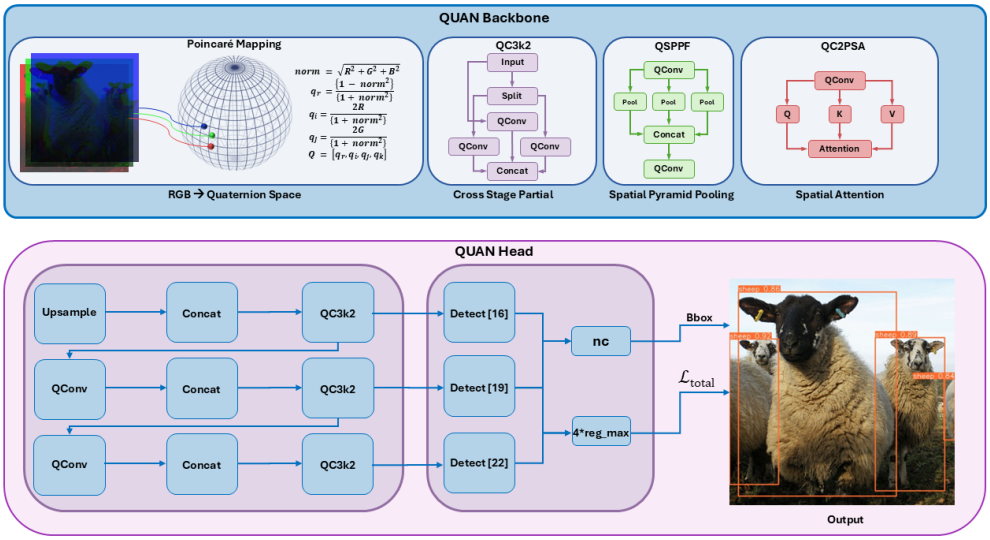


Fig. 3: QUAN architecture showing quaternion adaptations of key components: QConv, QC3k2, QSPPF, and QC2PSA

B. Network Architectures

For image classification, a Quaternion Wide ResNet (QWRN) [35] was implemented with the 16-4 configuration (16 layers, width factor 4) while YOLOv11n was implemented for detection. These architectures were selected for its efficiency and demonstrated success on state-of-the-art datasets. For classification, we use AdamW optimizer with cosine annealing, batch size 256, initial learning rate (lr) 0.1, 300 epochs. For object detection we primarily use SGD with batch size 32, automatic lr scheduling, for 100 epochs. Data augmentation includes random crops, horizontal flips, and AutoAugment for classification; mosaic augmentation, and affine transforms for detection.

C. Image Classification Performance

We compare QUAN against both standard CNNs and existing quaternion approaches. While several quaternion networks have been proposed [12] [14], most lack publicly available implementations or are limited to specific tasks. We implemented and evaluated pure quaternion operations (without our approximation), but found latency (3.8× slower) due to the lack of hardware optimization for quaternion arithmetic. For fair comparison, we focus on published results from reproducible quaternion methods and provide our implementation for community evaluation. Table I presents the classification accuracy across datasets. QUAN achieves state-of-the-art performance among quaternion networks while using significantly fewer parameters.

QUAN-WRN achieves 95.12% accuracy on CIFAR-10 with only 717K parameters—74% fewer than WRN-16-4 while outperforming all quaternion baselines. This parameter efficiency remains at larger scales, where QUAN requires only 5.39M parameters compared to 21.8M for a standard ResNet34, representing an 75% reduction without sacrificing accuracy.

TABLE I: Image classification accuracy and parameter efficiency

Model	Params	CIFAR-10	CIFAR-100	ImageNet
ResNet34	3.6M	94.98%	75.97%	73.74%*
WRN-16-4 [35]	2.75M	94.76%	76.09%	71.09%
QUAN	**717K**	**95.12%**	**76.83%**	**74.28%***
Deep QCNN [12]	932.8K	94.56%	73.99%	-
QVGG11 [36]	3.8M	85.15%	-	-
Shallow QCNN [14]	-	77.78%	-	-
QResNet34 [37]	1M	92-94%	-	-

*21.8M parameters for ResNet34, 5.39M parameters for QUAN on ImageNet

This efficiency is valuable for resource-constrained robotic systems, where model size directly impacts deployment feasibility. Despite the dramatic parameter reduction, training time only increased by 6.8% on average across our classification experiments, confirming that QUAN's approximation approach maintains computational efficiency.

D. Object Detection Results

1) Classical Object Detection: Table II shows QUAN maintains competitive performance on axis-aligned detection while dramatically reducing model size.

TABLE II: Classical Object Detection Performance

Dataset	Model	Params	mAP50	mAP50-95	ms
COCO2017	QUAN-YOLO11n	**1.187M**	0.478	0.339	1.68
	YOLO11n	2.65M	**0.547**	**0.3908**	**1.5**
Robotic Dataset	QUAN-YOLO11n	**0.67M**	0.989	**0.955**	**1.95**
	YOLO11n	2.61M	**0.992**	0.951	2.098

QUAN outperforms standard YOLO11n (0.9548 vs 0.951 mAP50-95) on the robotic dataset while using only 25.3% of the parameters and 36% more inference time. On COCO, QUAN underperforms YOLO11n (.339 vs .3908 mAP50-95) with 12% more inference time. Despite scaling up the architecture, QUAN experiences impaired generalization as the dataset scales likely due to the decreased model size.

Fig. 4: Oriented Object Detection Predictions by QUAN-YOLOv11n

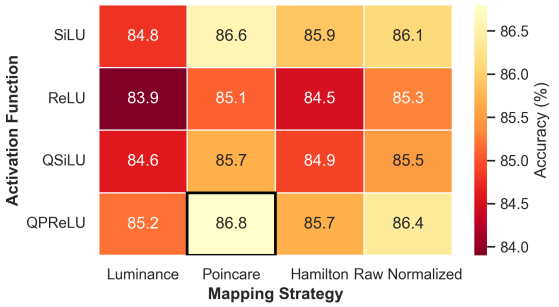


Fig. 5: Ablation studies for quaternion activation and mapping

Additionally, we did not have the computational capacity to use the same batch size and hyperparameters used on the original models when training from scratch.

2) Oriented Object Detection: For rotation-aware detection critical in robotic applications, Table III demonstrates QUAN's effectiveness across aerial and robotic datasets.

TABLE III: Oriented object detection performance

Dataset	Model	Params	mAP@50	mAP@50-95	ms
DOTA	YOLO11n-obb	2.66M	77.3	61.4	4.287
	QUAN-YOLO11n	**1.24M**	76.2	60.8	4.61
Robotic	YOLO11n-obb	2.66M	**93.3**	68.5	3.57
	QUAN-YOLO11n	**0.69M**	92.4	**76.4**	**3.44**

On the robotic dataset, QUAN outperforms YOLO11n-obb (76.4 vs. 68.5) while reducing the inference time by 3.7%. Figure 4 demonstrates QUAN's detection capabilities, showing accurate oriented bounding box annotations. On DOTAv1, we see the same performance impairment (60.8 vs 61.4 map50-95) as we did on COCO likely due to the small model size.

E. Ablation Studies

Figure 5 presents the ablation study results of quaternion activation and mapping strategies. While Poincaré mapping generally outperforms alternatives across activation functions, SiLU consistently shows superior results over ReLU and QPReLU variants. Traditional quaternion batch normalization showed on average 0.61% less accuracy than IQBN across the CIFAR-10 experiments and was not included, but may offer advantages in other contexts or with further optimization.

V. CONCLUSIONS

Quaternion Approximate Networks (QUAN) advance quaternion neural networks while maintaining practical applicability across both image classification and oriented object detection. By implementing quaternion operations through efficient real-valued approximations and independent normalization, QUAN overcomes key limitations that have hindered the widespread adoption of quaternion networks. The systematic adaptation of standard YOLO architectural blocks (C3k2, SPPF, C2PSA) to quaternion counterparts enables integration of rotation equivariance into existing object detection frameworks. However, it is not without limitations as the quaternion approximation increases arithmetic operations, resulting in slightly higher inference time that our CUDA kernels partially mitigate. The Poincare mapping, while effective, may not be optimal for all downstream tasks. Future work will explore: self-supervised learning with quaternion representations, optimization over the unique Hamilton matrices to identify optimal task-specific structures, adapting LoRA with quaternions, and live use in robotic assembly applications, where precise spatial understanding is essential for accurate manipulation.

REFERENCES

[1] R. Khanam and M. Hussain, "Yolov11: An overview of the key architectural enhancements," 2024. [Online]. Available: https://arxiv.org/abs/2410.17725

[2] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," 2016. [Online]. Available: https://arxiv.org/abs/1506.01497

[3] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," 2020. [Online]. Available: https://arxiv.org/abs/2005.12872

[4] C. Wang, D. Xu, Y. Zhu, R. Martín-Martín, C. Lu, L. Fei-Fei, and S. Savarese, "Densefusion: 6d object pose estimation by iterative dense fusion," 2019. [Online]. Available: https://arxiv.org/abs/1901.04780

[5] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, "Emerging properties in self-supervised vision transformers," 2021. [Online]. Available: https://arxiv.org/abs/2104.14294

[6] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," 2020. [Online]. Available: https://arxiv.org/abs/2002.05709

[7] W. Fedus, B. Zoph, and N. Shazeer, "Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity," 2022. [Online]. Available: https://arxiv.org/abs/2101.03961

[8] X. Yang, J. Yan, Z. Feng, and T. He, "R3det: Refined single-stage detector with feature refinement for rotating object," 2020. [Online]. Available: https://arxiv.org/abs/1908.05612

[9] Y. Xu, M. Fu, Q. Wang, Y. Wang, K. Chen, G.-S. Xia, and X. Bai, "Gliding vertex on the horizontal bounding box for multi-oriented object detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 4, p. 1452–1459, Apr. 2021. [Online]. Available: http://dx.doi.org/10.1109/TPAMI.2020.2974745

[10] W. R. Hamilton, *Elements of quaternions*. BoD–Books on Demand, 2022.

[11] J. B. Kuipers, "Quaternion algebra," in *Quaternions and Rotation Sequences: A Primer With Applications to Orbits, Aerospace and Virtual Reality*. Princeton Univ. Press, 1999, p. 125.

[12] C. Gaudet and A. Maida, "Deep quaternion networks," 2018. [Online]. Available: https://arxiv.org/abs/1712.04604

[13] T. Parcollet, M. Ravanelli, M. Morchid, G. Linarès, C. Trabelsi, R. D. Mori, and Y. Bengio, "Quaternion recurrent neural networks," 2019. [Online]. Available: https://arxiv.org/abs/1806.04418

[14] X. Zhu, Y. Xu, H. Xu, and C. Chen, "Quaternion convolutional neural networks," 2019. [Online]. Available: https://arxiv.org/abs/1903.00658

[15] G. Altamirano-Gomez and C. Gershenson, "Quaternion convolutional neural networks: Current advances and future directions," 2023. [Online]. Available: https://arxiv.org/abs/2307.08663
[16] T. Hodan, F. Michel, E. Brachmann, W. Kehl, A. G. Buch, D. Kraft, B. Drost, J. Vidal, S. Ihrke, X. Zabulis, C. Sahin, F. Manhardt, F. Tombari, T.-K. Kim, J. Matas, and C. Rother, "Bop: Benchmark for 6d object pose estimation," 2018. [Online]. Available: https://arxiv.org/abs/1808.08319
[17] K. Kleeberger, C. Landgraf, and M. F. Huber, "Large-scale 6d object pose estimation dataset for industrial bin-picking," 2019. [Online]. Available: https://arxiv.org/abs/1912.12125
[18] D. Comminiello, M. Lella, S. Scardapane, and A. Uncini, "Quaternion convolutional neural networks for detection and localization of 3d sound events," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 8533–8537.
[19] E. Grassucci, D. Comminiello, and A. Uncini, "A quaternion-valued variational autoencoder," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, June 2021. [Online]. Available: http://dx.doi.org/10.1109/ICASSP39728.2021.9413859
[20] H.-W. Hsu, T.-Y. Wu, S. Wan, W. H. Wong, and C.-Y. Lee, "Quatnet: Quaternion-based head pose estimation with multiregression loss," *IEEE Transactions on Multimedia*, vol. 21, no. 4, pp. 1035–1046, 2019.
[21] Y. He, W. Sun, H. Huang, J. Liu, H. Fan, and J. Sun, "PVN3D: A deep point-wise 3d keypoints voting network for 6dof pose estimation," *CoRR*, vol. abs/1911.04231, 2019. [Online]. Available: http://arxiv.org/abs/1911.04231
[22] T. S. Cohen and M. Welling, "Group equivariant convolutional networks," 2016. [Online]. Available: https://arxiv.org/abs/1602.07576
[23] M. Weiler and G. Cesa, "General $e(2)$-equivariant steerable cnns," 2021. [Online]. Available: https://arxiv.org/abs/1911.08251
[24] D. Marcos, M. Volpi, N. Komodakis, and D. Tuia, "Rotation equivariant vector field networks," in *2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE, Oct. 2017. [Online]. Available: http://dx.doi.org/10.1109/ICCV.2017.540
[25] D. E. Worrall, S. J. Garbin, D. Turmukhambetov, and G. J. Brostow, "Harmonic networks: Deep translation and rotation equivariance," 2017. [Online]. Available: https://arxiv.org/abs/1612.04642
[26] Y. Zhou, Q. Ye, Q. Qiu, and J. Jiao, "Oriented response networks," 2017. [Online]. Available: https://arxiv.org/abs/1701.01833
[27] W. Li, Z. Yang, W. Han, H. Man, X. Wang, and X. Fan, "Hyperbolic-constraint point cloud reconstruction from single rgb-d images," 2024. [Online]. Available: https://arxiv.org/abs/2412.09055
[28] M. Nickel and D. Kiela, "Poincare embeddings for learning hierarchical representations," 2017. [Online]. Available: https://arxiv.org/abs/1705.08039
[29] M. Kochurov, R. Karimov, and S. Kozlukov, "Geoopt: Riemannian optimization in pytorch," 2020. [Online]. Available: https://arxiv.org/abs/2005.02819
[30] H. Zhou, X. Zhang, C. Zhang, and Q. Ma, "Quaternion convolutional neural networks for hyperspectral image classification," *Engineering Applications of Artificial Intelligence*, vol. 123, p. 106234, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0952197623004189
[31] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," University of Toronto, Toronto, Ontario, Tech. Rep. 0, 2009. [Online]. Available: https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf
[32] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "Imagenet," 2015. [Online]. Available: https://arxiv.org/abs/1409.0575
[33] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft coco: Common objects in context," 2015. [Online]. Available: https://arxiv.org/abs/1405.0312
[34] G.-S. Xia, X. Bai, J. Ding, Z. Zhu, S. Belongie, J. Luo, M. Datcu, M. Pelillo, and L. Zhang, "Dota: A large-scale dataset for object detection in aerial images," 2019. [Online]. Available: https://arxiv.org/abs/1711.10398
[35] S. Zagoruyko and N. Komodakis, "Wide residual networks," 2017. [Online]. Available: https://arxiv.org/abs/1605.07146
[36] J. Pöppelbaum and A. Schwung, "Improving quaternion neural networks with quaternionic activation functions," 2024. [Online]. Available: https://arxiv.org/abs/2406.16481
[37] S. Hongo, T. Isokawa, N. Matsui, H. Nishimura, and N. Kamiura, "Constructing convolutional neural networks based on quaternion," in *2020 International Joint Conference on Neural Networks (IJCNN)*, 2020, pp. 1–6.

VI. APPENDIX

A. Mapping Strategies

Multiple RGB-to-quaternion mapping strategies have been investigated to determine the optimal approach for preserving both color relationships and geometric properties. The evaluated mappings included:

- **Luminance mapping**: Uses perceptual weights (0.299R + 0.587G + 0.114B) for the real component while preserving normalized RGB values in the imaginary components, providing human perception-aligned representations.
- **Mean brightness mapping**: Utilizes the arithmetic mean of RGB channels as the real component, offering computational simplicity.
- **Raw normalized mapping**: Directly normalizes RGB values with equal weighting across channels.
- **Hamilton mapping**: Embeds RGB values as pure imaginary components (0, R, G, B), preserving raw color information.
- **Poincaré mapping**: Projects RGB values onto the unit quaternion sphere, ensuring numerical stability during training while preserving relative color relationships.