Noto Serif

L.E.P.A.U.T.E. Framework

Achieving Precise Modeling of Geometric Transformations

Carson Wu
September 2025

Agenda

- Introduction

Agenda

- Introduction
- Lie Group Foundations

Agenda

- Introduction
- Lie Group Foundations
- Framework Overview

Agenda

- Introduction
- Lie Group Foundations
- Framework Overview
- Neural Network Architecture

Agenda

- Introduction
- Lie Group Foundations
- Framework Overview
- Neural Network Architecture
- Training and Optimization

Agenda

- Introduction
- Lie Group Foundations
- Framework Overview
- Neural Network Architecture
- Training and Optimization
- Implementation Process

Agenda

- Introduction
- Lie Group Foundations
- Framework Overview
- Neural Network Architecture
- Training and Optimization
- Implementation Process
- Practical Applications

Agenda

- Introduction
- Lie Group Foundations
- Framework Overview
- Neural Network Architecture
- Training and Optimization
- Implementation Process
- Practical Applications
- Impact Analysis

Agenda

- Introduction
- Lie Group Foundations
- Framework Overview
- Neural Network Architecture
- Training and Optimization
- Implementation Process
- Practical Applications
- Impact Analysis
- Future Directions

Agenda

- Introduction
- Lie Group Foundations
- Framework Overview
- Neural Network Architecture
- Training and Optimization
- Implementation Process
- Practical Applications
- Impact Analysis
- Future Directions
- Conclusion

Motivation

- **Challenge**: Traditional CNNs struggle with explicit geometric transformation modeling (e.g., rotation, translation).

Motivation

- **Challenge**: Traditional CNNs struggle with explicit geometric transformation modeling (e.g., rotation, translation).
- **Solution**: L.E.P.A.U.T.E. Framework uses Lie group theory to model transformations intrinsically.

¡4¿

Motivation

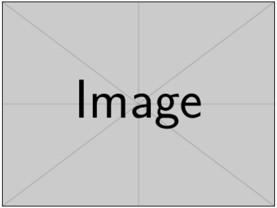
- **Challenge**: Traditional CNNs struggle with explicit geometric transformation modeling (e.g., rotation, translation).
- **Solution**: L.E.P.A.U.T.E. Framework uses Lie group theory to model transformations intrinsically.
- **Goal**: Achieve precise, robust modeling for computer vision tasks like 3D reconstruction, robotics, and medical imaging.

¡4¿

Motivation

- **Challenge**: Traditional CNNs struggle with explicit geometric transformation modeling (e.g., rotation, translation).
- **Solution**: L.E.P.A.U.T.E. Framework uses Lie group theory to model transformations intrinsically.
- **Goal**: Achieve precise, robust modeling for computer vision tasks like 3D reconstruction, robotics, and medical imaging.

¡4¿



Lie Groups and Lie Algebra

- **Lie Group ($\mathcal{G}$)**: A group with a differentiable manifold structure, e.g., $SE(3)$ for 3D transformations.

- **Lie Group ($\mathcal{G}$)**: A group with a differentiable manifold structure, e.g., $SE(3)$ for 3D transformations.
- **Examples**:

Lie Groups and Lie Algebra

- **Lie Group ($\mathcal{G}$)**: A group with a differentiable manifold structure, e.g., $SE(3)$ for 3D transformations.
- **Examples**:
  - $SE(2)$: 2D rotation and translation.

Lie Groups and Lie Algebra

- **Lie Group ($\mathcal{G}$)**: A group with a differentiable manifold structure, e.g., $SE(3)$ for 3D transformations.
- **Examples**:
  - $SE(2)$: 2D rotation and translation.
  - $SE(3)$: 3D rigid body transformations,

Lie Groups and Lie Algebra

- **Lie Group ($\mathcal{G}$)**: A group with a differentiable manifold structure, e.g., $SE(3)$ for 3D transformations.
- **Examples**:
  - $SE(2)$: 2D rotation and translation.
  - $SE(3)$: 3D rigid body transformations,

$$g = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix}, \quad R \in SO(3), \quad t \in \mathbb{R}^3$$

Lie Groups and Lie Algebra

- **Lie Group ($\mathcal{G}$)**: A group with a differentiable manifold structure, e.g., $SE(3)$ for 3D transformations.
- **Examples**:
  - $SE(2)$: 2D rotation and translation.
  - $SE(3)$: 3D rigid body transformations,

$$g = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix}, \quad R \in SO(3), \quad t \in \mathbb{R}^3$$

- **Lie Algebra ($\mathfrak{g}$)**: Tangent space at identity, e.g., $\mathfrak{se}(3)$ with generators for rotation and translation.

Lie Groups and Lie Algebra

- **Lie Group ($\mathcal{G}$)**: A group with a differentiable manifold structure, e.g., $SE(3)$ for 3D transformations.
- **Examples**:
  - $SE(2)$: 2D rotation and translation.
  - $SE(3)$: 3D rigid body transformations,

$$g = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix}, \quad R \in SO(3), \quad t \in \mathbb{R}^3$$

- **Lie Algebra ($\mathfrak{g}$)**: Tangent space at identity, e.g., $\mathfrak{se}(3)$ with generators for rotation and translation.
- **Exponential Map**: $\exp : \mathfrak{g} \to \mathcal{G}$, maps algebra to group.

L.E.P.A.U.T.E. Framework

- **Core Idea**: Embed geometric transformations using Lie groups in neural networks.

- **Core Idea**: Embed geometric transformations using Lie groups in neural networks.
- **Components**:

¡7¿

- **Core Idea**: Embed geometric transformations using Lie groups in neural networks.
- **Components**:
  - Lie group convolutional layers for equivariant feature extraction.

¡7¿

L.E.P.A.U.T.E. Framework

- **Core Idea**: Embed geometric transformations using Lie groups in neural networks.
- **Components**:
  - Lie group convolutional layers for equivariant feature extraction.
  - Lie group attention mechanisms for geometric focus.

¡7¿

L.E.P.A.U.T.E. Framework

- **Core Idea**: Embed geometric transformations using Lie groups in neural networks.
- **Components**:
  - Lie group convolutional layers for equivariant feature extraction.
  - Lie group attention mechanisms for geometric focus.
  - Geometric invariance/equivariance loss functions.

¡7¿

L.E.P.A.U.T.E. Framework

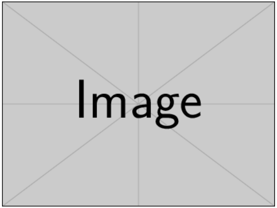
- **Core Idea**: Embed geometric transformations using Lie groups in neural networks.
- **Components**:
  - Lie group convolutional layers for equivariant feature extraction.
  - Lie group attention mechanisms for geometric focus.
  - Geometric invariance/equivariance loss functions.
- **Applications**: 3D reconstruction, robotic navigation, medical imaging, autonomous driving.

¡7¿

L.E.P.A.U.T.E. Framework

- **Core Idea**: Embed geometric transformations using Lie groups in neural networks.
- **Components**:
  - Lie group convolutional layers for equivariant feature extraction.
  - Lie group attention mechanisms for geometric focus.
  - Geometric invariance/equivariance loss functions.
- **Applications**: 3D reconstruction, robotic navigation, medical imaging, autonomous driving.

¡7¿



Lie Group Convolutional Layer

- **Definition**: Convolution on Lie group $\mathcal{G}$:

Lie Group Convolutional Layer

- **Definition**: Convolution on Lie group $\mathcal{G}$:

$$(f * k)(g) = \int_{\mathcal{G}} f(h)k(h^{-1}g)\, dh$$

Lie Group Convolutional Layer

- **Definition**: Convolution on Lie group $\mathcal{G}$:

$$(f * k)(g) = \int_{\mathcal{G}} f(h)k(h^{-1}g)\,dh$$

- **Equivariance**: Ensures $(f \circ L_g) * k = (f * k) \circ L_g$.

Lie Group Convolutional Layer

- **Definition**: Convolution on Lie group $\mathcal{G}$:

$$(f * k)(g) = \int_{\mathcal{G}} f(h)k(h^{-1}g)\, dh$$

- **Equivalence**: Ensures $(f \circ L_g) * k = (f * k) \circ L_g$.
- **Implementation**:

Lie Group Convolutional Layer

- **Definition**: Convolution on Lie group $\mathcal{G}$:

$$(f * k)(g) = \int_{\mathcal{G}} f(h)k(h^{-1}g)\, dh$$

- **Equivalence**: Ensures $(f \circ L_g) * k = (f * k) \circ L_g$.
- **Implementation**:
  - Discretize $\mathcal{G}$ (e.g., grid sampling of $SE(3)$).

Lie Group Convolutional Layer

- **Definition**: Convolution on Lie group $\mathcal{G}$:

$$(f * k)(g) = \int_{\mathcal{G}} f(h)k(h^{-1}g)\, dh$$

- **Equivariance**: Ensures $(f \circ L_g) * k = (f * k) \circ L_g$.
- **Implementation**:
  - Discretize $\mathcal{G}$ (e.g., grid sampling of $SE(3)$).
  - Use spherical harmonics for $SO(3)$ kernels.

Lie Group Convolutional Layer

- **Definition**: Convolution on Lie group $\mathcal{G}$:

$$(f * k)(g) = \int_{\mathcal{G}} f(h)k(h^{-1}g)\,dh$$

- **Equivalance**: Ensures $(f \circ L_g) * k = (f * k) \circ L_g$.
- **Implementation**:
  - Discretize $\mathcal{G}$ (e.g., grid sampling of $SE(3)$).
  - Use spherical harmonics for $SO(3)$ kernels.
  - Optimize with FFT for efficiency.

Lie Group Attention Mechanism

- **Formula**:

where $Q, K, V : \mathcal{G} \rightarrow \mathbb{R}^d$.

- **Formula**:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

where $Q, K, V : \mathcal{G} \to \mathbb{R}^d$.

¡6¿

Lie Group Attention Mechanism

- **Formula**:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) V$$

where $Q, K, V : \mathcal{G} \to \mathbb{R}^d$.

- **Geometric Compatibility**: Scores based on relative transformations,

¡6¿

Lie Group Attention Mechanism

- **Formula**:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) V$$

  where $Q, K, V : \mathcal{G} \to \mathbb{R}^d$.
- **Geometric Compatibility**: Scores based on relative transformations,

$$\text{score}(g_i, g_j) = \phi(Q(g_i), K(g_j), g_i^{-1} g_j)$$

¡6¿

Lie Group Attention Mechanism

- **Formula**:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

  where $Q, K, V : \mathcal{G} \rightarrow \mathbb{R}^d$.

- **Geometric Compatibility**: Scores based on relative transformations,

$$\text{score}(g_i, g_j) = \phi(Q(g_i), K(g_j), g_i^{-1}g_j)$$

- **Features**: Multi-head attention, geometric positional encoding via Lie algebra.

¡6¿

Lie Group Attention Mechanism

- **Formula**:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

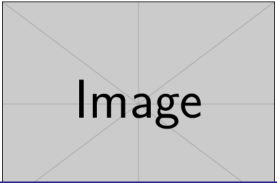
where $Q, K, V : \mathcal{G} \to \mathbb{R}^d$.

- **Geometric Compatibility**: Scores based on relative transformations,

$$\text{score}(g_i, g_j) = \phi(Q(g_i), K(g_j), g_i^{-1}g_j)$$

- **Features**: Multi-head attention, geometric positional encoding via Lie algebra.

¡6¿



Loss Functions

- **Geometric Invariance Loss**:

Loss Functions

- **Geometric Invariance Loss**:

$$L_{\text{inv}} = \sum_{i} \sum_{g \in \mathcal{G}} |f(x_i) - f(T(g)x_i)|^2$$

Loss Functions

- **Geometric Invariance Loss**:

$$L_{\text{inv}} = \sum_i \sum_{g \in \mathcal{G}} |f(x_i) - f(T(g)x_i)|^2$$

- **Equivariance Loss**:

Loss Functions

- **Geometric Invariance Loss**:

$$L_{\text{inv}} = \sum_i \sum_{g \in \mathcal{G}} |f(x_i) - f(T(g)x_i)|^2$$

- **Equivariance Loss**:

$$L_{\text{eq}} = \sum_i \sum_{g \in \mathcal{G}} |f(T(g)x_i) - T'(g)f(x_i)|^2$$

Loss Functions

- **Geometric Invariance Loss**:

$$L_{\text{inv}} = \sum_i \sum_{g \in \mathcal{G}} |f(x_i) - f(T(g)x_i)|^2$$

- **Equivariance Loss**:

$$L_{\text{eq}} = \sum_i \sum_{g \in \mathcal{G}} |f(T(g)x_i) - T'(g)f(x_i)|^2$$

- **Self-Supervised Learning**: Contrastive loss:

Loss Functions

- **Geometric Invariance Loss**:

$$L_{\text{inv}} = \sum_i \sum_{g \in \mathcal{G}} |f(x_i) - f(T(g)x_i)|^2$$

- **Equivariance Loss**:

$$L_{\text{eq}} = \sum_i \sum_{g \in \mathcal{G}} |f(T(g)x_i) - T'(g)f(x_i)|^2$$

- **Self-Supervised Learning**: Contrastive loss:

$$L_{\text{contrast}} = -\log \frac{\exp(\text{sim}(f(x_i), f(T(g)x_i))/\tau)}{\sum_j \exp(\text{sim}(f(x_i), f(x_j))/\tau)}$$

Training Process

- **Optimizer**: Adam with learning rate 10^{-4}, cosine annealing.

Training Process

- **Optimizer**: Adam with learning rate 10^{-4}, cosine annealing.
- **Regularization**: Weight decay (10^{-5}), dropout (0.1).

¡5¿

Training Process

- **Optimizer**: Adam with learning rate 10^{-4}, cosine annealing.
- **Regularization**: Weight decay (10^{-5}), dropout (0.1).
- **Data Augmentation**: Random rotations, translations in $\mathcal{G}$.

¡5¿

Training Process

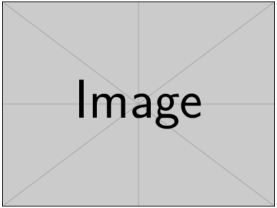
- **Optimizer**: Adam with learning rate 10^{-4}, cosine annealing.
- **Regularization**: Weight decay (10^{-5}), dropout (0.1).
- **Data Augmentation**: Random rotations, translations in $\mathcal{G}$.
- **Monitoring**: Track loss, geometric invariance metrics.

¡5¿

Training Process

- **Optimizer**: Adam with learning rate 10^{-4}, cosine annealing.
- **Regularization**: Weight decay (10^{-5}), dropout (0.1).
- **Data Augmentation**: Random rotations, translations in $\mathcal{G}$.
- **Monitoring**: Track loss, geometric invariance metrics.

¡5¿



Data Preprocessing

- **Standardization**: Normalize pixel values, adjust resolution (e.g., 256×256).

Data Preprocessing

- **Standardization**: Normalize pixel values, adjust resolution (e.g., 256×256).
- **Geometric Transformation Extraction**: Use SIFT, ORB, or RANSAC for $SE(3)$ estimation.

Data Preprocessing

- **Standardization**: Normalize pixel values, adjust resolution (e.g., 256×256).
- **Geometric Transformation Extraction**: Use SIFT, ORB, or RANSAC for $SE(3)$ estimation.
- **Lie Group Representation**: Map images to $f : \mathcal{G} \to \mathbb{R}^n$, discretize $\mathcal{G}$.

Data Preprocessing

- **Standardization**: Normalize pixel values, adjust resolution (e.g., 256×256).
- **Geometric Transformation Extraction**: Use SIFT, ORB, or RANSAC for $SE(3)$ estimation.
- **Lie Group Representation**: Map images to $f : \mathcal{G} \to \mathbb{R}^n$, discretize $\mathcal{G}$.
- **Tools**: OpenCV, PyTorch Geometric, Sophus.

Transformer Model Construction

- **Encoder**: 6-12 layers with:

Transformer Model Construction

- **Encoder**: 6-12 layers with:
  - Lie group convolution for feature extraction.

Transformer Model Construction

- **Encoder**: 6-12 layers with:
  - Lie group convolution for feature extraction.
  - Lie group attention for geometric focus.

Transformer Model Construction

- **Encoder**: 6-12 layers with:
  - Lie group convolution for feature extraction.
  - Lie group attention for geometric focus.
  - Feedforward network, LayerNorm, residual connections.

Transformer Model Construction

- **Encoder**: 6-12 layers with:
  - Lie group convolution for feature extraction.
  - Lie group attention for geometric focus.
  - Feedforward network, LayerNorm, residual connections.
- **Positional Encoding**: $\text{PE}(g) = \sin(\omega_k \cdot \xi_g)$.

Transformer Model Construction

- **Encoder**: 6-12 layers with:
  - Lie group convolution for feature extraction.
  - Lie group attention for geometric focus.
  - Feedforward network, LayerNorm, residual connections.
- **Positional Encoding**: $\mathrm{PE}(g) = \sin(\omega_k \cdot \xi_g)$.
- **Implementation**: PyTorch/JAX with Sophus.

Application Scenarios

- **3D Reconstruction**: Chamfer distance reduced by 10-15%.

Application Scenarios

- **3D Reconstruction**: Chamfer distance reduced by 10-15%.
- **Robotic Navigation/SLAM**: ATE reduced to 0.02m.

¡5¿

Application Scenarios

- **3D Reconstruction**: Chamfer distance reduced by 10-15%.
- **Robotic Navigation/SLAM**: ATE reduced to 0.02m.
- **Medical Imaging**: Dice coefficient improved to 0.90.

¡5¿

Application Scenarios

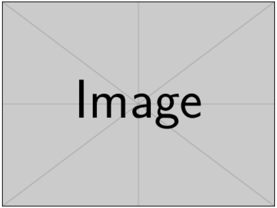
- **3D Reconstruction**: Chamfer distance reduced by 10-15%.
- **Robotic Navigation/SLAM**: ATE reduced to 0.02m.
- **Medical Imaging**: Dice coefficient improved to 0.90.
- **Autonomous Driving**: Pose errors reduced to 0.03m, mAP improved by 8%.

¡5¿

Application Scenarios

- **3D Reconstruction**: Chamfer distance reduced by 10-15%.
- **Robotic Navigation/SLAM**: ATE reduced to 0.02m.
- **Medical Imaging**: Dice coefficient improved to 0.90.
- **Autonomous Driving**: Pose errors reduced to 0.03m, mAP improved by 8%.

¡5¿



3D Reconstruction

- **Process**: Multi-view images $\longrightarrow$ Lie group features $\longrightarrow$ voxel/point cloud fusion.

3D Reconstruction

- **Process**: Multi-view images $\longrightarrow$ Lie group features $\longrightarrow$ voxel/point cloud fusion.
- **Tools**: Open3D, PyTorch3D, MeshLab.

3D Reconstruction

- **Process**: Multi-view images $\longrightarrow$ Lie group features $\longrightarrow$ voxel/point cloud fusion.
- **Tools**: Open3D, PyTorch3D, MeshLab.
- **Advantages**: Pose error $\sim$1, robust to noise.

3D Reconstruction

- **Process**: Multi-view images $\longrightarrow$ Lie group features $\longrightarrow$ voxel/point cloud fusion.
- **Tools**: Open3D, PyTorch3D, MeshLab.
- **Advantages**: Pose error $\sim$1, robust to noise.
- **Case Study**: VR gamingindoor scenes with 2 pose accuracy.

Robotic Navigation and SLAM

- **Process**: RGB-D/LiDAR $\rightarrow$ $SE(3)$ pose $\rightarrow$ map construction.

Robotic Navigation and SLAM

- **Process**: RGB-D/LiDAR $\rightarrow$ $SE(3)$ pose $\rightarrow$ map construction.
- **Tools**: ORB-SLAM3, g2o, ROS.

Robotic Navigation and SLAM

- **Process**: RGB-D/LiDAR $\rightarrow$ $SE(3)$ pose $\rightarrow$ map construction.
- **Tools**: ORB-SLAM3, g2o, ROS.
- **Advantages**: ATE $\sim$0.02m, 15% better map consistency.

Robotic Navigation and SLAM

- **Process**: RGB-D/LiDAR $\rightarrow$ $SE(3)$ pose $\rightarrow$ map construction.
- **Tools**: ORB-SLAM3, g2o, ROS.
- **Advantages**: ATE $\sim$0.02m, 15% better map consistency.
- **Case Study**: Hospital robots with 0.03m localization error.

Medical Image Processing

- **Process**: CT/MRI $\rightarrow$ $SE(3)$ registration $\rightarrow$ segmentation/classification.

Medical Image Processing

- **Process**: CT/MRI $\rightarrow$ $SE(3)$ registration $\rightarrow$ segmentation/classification.
- **Tools**: ITK, MONAI, 3D Slicer.

Medical Image Processing

- **Process**: CT/MRI $\rightarrow$ $SE(3)$ registration $\rightarrow$ segmentation/classification.
- **Tools**: ITK, MONAI, 3D Slicer.
- **Advantages**: Dice coefficient $\sim$0.90, 10% error reduction.

Medical Image Processing

- **Process**: CT/MRI $\rightarrow$ $SE(3)$ registration $\rightarrow$ segmentation/classification.
- **Tools**: ITK, MONAI, 3D Slicer.
- **Advantages**: Dice coefficient $\sim$0.90, 10% error reduction.
- **Case Study**: Brain tumor segmentation with 0.92 Dice score.

Autonomous Driving and UAVs

- **Process**: Multimodal data $\rightarrow$ $SE(3)$ pose $\rightarrow$ object detection/path planning.

Autonomous Driving and UAVs

- **Process**: Multimodal data $\rightarrow$ $SE(3)$ pose $\rightarrow$ object detection/path planning.
- **Tools**: Apollo, ROS, TensorRT.

Autonomous Driving and UAVs

- **Process**: Multimodal data $\rightarrow$ $SE(3)$ pose $\rightarrow$ object detection/path planning.
- **Tools**: Apollo, ROS, TensorRT.
- **Advantages**: Pose error $\sim$0.03m, mAP improved by 8%.

Autonomous Driving and UAVs

- **Process**: Multimodal data $\rightarrow$ $SE(3)$ pose $\rightarrow$ object detection/path planning.
- **Tools**: Apollo, ROS, TensorRT.
- **Advantages**: Pose error $\sim$0.03m, mAP improved by 8%.
- **Case Study**: Urban driving with 0.02m localization accuracy.

Advantages and Limitations

- **Advantages**:

Advantages and Limitations

- **Advantages**:
  - Robust geometric invariance/equivariance.

Advantages and Limitations

- **Advantages**:
  - Robust geometric invariance/equivariance.
  - Precise modeling for 3D tasks (pose error $\sim$1).

Advantages and Limitations

- **Advantages**:
  - Robust geometric invariance/equivariance.
  - Precise modeling for 3D tasks (pose error $\sim$1).
  - Reduced data dependency via self-supervised learning.

Advantages and Limitations

- **Advantages**:
  - Robust geometric invariance/equivariance.
  - Precise modeling for 3D tasks (pose error $\sim$1).
  - Reduced data dependency via self-supervised learning.
- **Limitations**:

Advantages and Limitations

- **Advantages**:
  - Robust geometric invariance/equivariance.
  - Precise modeling for 3D tasks (pose error $\sim$1).
  - Reduced data dependency via self-supervised learning.
- **Limitations**:
  - High computational complexity.

Advantages and Limitations

- **Advantages**:
  - Robust geometric invariance/equivariance.
  - Precise modeling for 3D tasks (pose error $\sim$1).
  - Reduced data dependency via self-supervised learning.
- **Limitations**:
  - High computational complexity.
  - Requires diverse transformation data.

Advantages and Limitations

- **Advantages**:
  - Robust geometric invariance/equivariance.
  - Precise modeling for 3D tasks (pose error $\sim$1).
  - Reduced data dependency via self-supervised learning.
- **Limitations**:
  - High computational complexity.
  - Requires diverse transformation data.
  - Steep learning curve for Lie group theory.

Comparison with Existing Methods

Method	Geometric Modeling	Invariance	Complexity
CNN (ResNet)			

Comparison with Existing Methods

Method	Geometric Modeling	Invariance	Complexity
CNN (ResNet)	Implicit		

Comparison with Existing Methods

Method	Geometric Modeling	Invariance	Complexity
CNN (ResNet)	Implicit	Limited	

Comparison with Existing Methods

Method	Geometric Modeling	Invariance	Complexity
CNN (ResNet)	Implicit	Limited	Medium

Comparison with Existing Methods

Method	Geometric Modeling	Invariance	Complexity
CNN (ResNet)	Implicit	Limited	Medium
STN			

Comparison with Existing Methods

Method	Geometric Modeling	Invariance	Complexity
CNN (ResNet)	Implicit	Limited	Medium
STN	Explicit		

Comparison with Existing Methods

Method	Geometric Modeling	Invariance	Complexity
CNN (ResNet)	Implicit	Limited	Medium
STN	Explicit	Partial	

Comparison with Existing Methods

Method	Geometric Modeling	Invariance	Complexity
CNN (ResNet)	Implicit	Limited	Medium
STN	Explicit	Partial	Medium

Comparison with Existing Methods

Method	**Geometric Modeling**	**Invariance**	**Complexity**
CNN (ResNet)	Implicit	Limited	Medium
STN	Explicit	Partial	Medium
ViT			

Comparison with Existing Methods

Method	Geometric Modeling	Invariance	Complexity
CNN (ResNet)	Implicit	Limited	Medium
STN	Explicit	Partial	Medium
ViT	Implicit		

Comparison with Existing Methods

Method	Geometric Modeling	Invariance	Complexity
CNN (ResNet)	Implicit	Limited	Medium
STN	Explicit	Partial	Medium
ViT	Implicit	Limited	

Comparison with Existing Methods

Method	Geometric Modeling	Invariance	Complexity
CNN (ResNet)	Implicit	Limited	Medium
STN	Explicit	Partial	Medium
ViT	Implicit	Limited	High

Comparison with Existing Methods

Method	Geometric Modeling	Invariance	Complexity
CNN (ResNet)	Implicit	Limited	Medium
STN	Explicit	Partial	Medium
ViT	Implicit	Limited	High
L.E.P.A.U.T.E.			

Comparison with Existing Methods

Method	Geometric Modeling	Invariance	Complexity
CNN (ResNet)	Implicit	Limited	Medium
STN	Explicit	Partial	Medium
ViT	Implicit	Limited	High
L.E.P.A.U.T.E.	Explicit		

Comparison with Existing Methods

Method	Geometric Modeling	Invariance	Complexity
CNN (ResNet)	Implicit	Limited	Medium
STN	Explicit	Partial	Medium
ViT	Implicit	Limited	High
L.E.P.A.U.T.E.	Explicit	Strong	

Comparison with Existing Methods

Method	**Geometric Modeling**	**Invariance**	**Complexity**
CNN (ResNet)	Implicit	Limited	Medium
STN	Explicit	Partial	Medium
ViT	Implicit	Limited	High
L.E.P.A.U.T.E.	Explicit	Strong	High

Future Improvements

- **Algorithm Optimization**: Sparse convolutions, steerable filters.

Future Improvements

- **Algorithm Optimization**: Sparse convolutions, steerable filters.
- **Hybrid Models**: Combine CNNs with Lie group modules.

Future Improvements

- **Algorithm Optimization**: Sparse convolutions, steerable filters.
- **Hybrid Models**: Combine CNNs with Lie group modules.
- **Data Generation**: High-fidelity synthetic datasets.

Future Improvements

- **Algorithm Optimization**: Sparse convolutions, steerable filters.
- **Hybrid Models**: Combine CNNs with Lie group modules.
- **Data Generation**: High-fidelity synthetic datasets.
- **Open-Source Tools**: Standardize Lie group vision libraries.

Conclusion

- L.E.P.A.U.T.E. Framework revolutionizes computer vision by explicitly modeling geometric transformations.

Conclusion

- L.E.P.A.U.T.E. Framework revolutionizes computer vision by explicitly modeling geometric transformations.
- **Strengths**: Precise, robust, versatile for 3D tasks.

Conclusion

- L.E.P.A.U.T.E. Framework revolutionizes computer vision by explicitly modeling geometric transformations.
- **Strengths**: Precise, robust, versatile for 3D tasks.
- **Applications**: 3D reconstruction, robotics, medical imaging, autonomous driving.

Conclusion

- L.E.P.A.U.T.E. Framework revolutionizes computer vision by explicitly modeling geometric transformations.
- **Strengths**: Precise, robust, versatile for 3D tasks.
- **Applications**: 3D reconstruction, robotics, medical imaging, autonomous driving.
- **Future**: Optimize efficiency, expand multimodal integration.

Questions?

1=101=-1 by Thank You! by= by by10

Questions?

1=101=-1 by Thank You! by= by by10

Questions?

1=101=-1 by Thank You! by= by by10

Questions?

1=101=-1 by Thank You! by= by by10

Questions?

1=101=-1 by Thank You! by= by by10

Questions?

1=101=-1 by Thank You! by= by by10

Questions?

1=101=-1 by Thank You! by= by by10

Questions?

1=101=-1 by Thank You! by= by by10

Questions?

1=101=-1 by Thank You! by= by by10

Questions?

$_{1=}$Thank You!