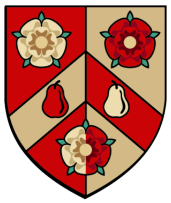
Deep Representation Learning for Dynamic Point Cloud Sequences

Jiaxing Zhong

Wolfson College, University of Oxford



Supervisors:

Prof. Andrew Markham

Prof. Niki Trigoni

March 2026

Dedication

To my beloved grandmother

(Dec. 1920–Sep. 2024)

Who witnessed a century of extraordinary change
and taught me that strength and optimism
are the greatest gifts we can carry forward.

Your wisdom, warmth, and unwavering belief in education
have guided every step of this journey.

Though you are no longer here to see this milestone,
your love and encouragement live on in every page.

Acknowledgements

First and foremost, I would like to express my deepest gratitude to my supervisors, Professor Niki Trigoni and Professor Andrew Markham, both of whom have provided exceptional guidance, unwavering support, and intellectual mentorship throughout my doctoral journey. I am equally grateful to both supervisors for their profound expertise, insightful feedback, and patient encouragement, which have shaped not only this thesis but also my development as a researcher. Their collective trust in my abilities and provision of freedom to explore ambitious research directions, while offering invaluable guidance when needed, have been fundamental to my success.

I would also like to extend my sincere gratitude to my examiners, Professor Maurice Fallon and Professor Tolga Birdal, for their time, meticulous review, and highly constructive feedback. I am particularly grateful for our inspiring discussion during the viva. The perspective they shared—that a DPhil is, above all, about maintaining a genuine passion for academia and the relentless pursuit of intellectual curiosity—resonated with me deeply and will serve as a guiding principle in my future career.

I extend my sincere appreciation to all my collaborators who have contributed significantly to this research. Special thanks go to Qian Xie, Xinyu Hou, Bo Yang, Bing Wang, Yuhang He, Kaichen Zhou, Kai Lu, Jialu Wang, Yixiong Jing, Sangyun Shin, Qianyi Deng, Ta-Ying Cheng, Yiyuan Yang, Yuzhou Zhou, Jiaojiao Ye, Chenyang Ma, Shitong Xu and He Liang, whose diverse expertise and collaborative spirit have been invaluable throughout this journey. Each collaboration has been intellectually stimulating and has enriched my research perspective through our interdisciplinary work in addressing complex challenges across mul-

tiple domains.

I would like to thank my fellow members of the Cyber-Physical Systems Group for creating a vibrant and supportive research environment. The stimulating discussions, constructive feedback during group meetings, and collaborative spirit have made this journey both productive and enjoyable. I am also grateful to my friends at the University of Oxford, whose diverse perspectives and camaraderie have made my time at Oxford truly memorable.

My heartfelt appreciation goes to the Wolfson College community for providing a welcoming and intellectually stimulating environment. The college's multicultural atmosphere and supportive community have been a source of inspiration and comfort throughout my studies.

Last but not least, I owe an immense debt of gratitude to my family. To my late grandmother, whose century of wisdom and unwavering belief in the power of education continue to inspire me every day. To my parents, your sacrifices, unconditional love, and constant support have made this journey possible. Your encouragement through every challenge, unwavering faith in my abilities even when I doubted myself, and patience with the demanding nature of doctoral studies have been my foundation and a constant source of motivation. Your dedication to my education and personal growth, often at your own expense, represents the deepest form of love and sacrifice that I will forever cherish. I also wish to acknowledge Caixuan Ji, my first love, for her understanding and invaluable emotional support during a memorable period of my doctoral studies. The shared moments of learning and discovery throughout this lengthy academic journey were deeply meaningful. Though our paths have diverged, I remain grateful for our years of companionship and its positive impacts on my personal growth.

This work would not have been possible without the collective support, encour-

agement, and inspiration from all these individuals. While this thesis bears my name, it represents the culmination of countless interactions, collaborations, and shared experiences with remarkable people who have enriched my academic and personal journey.

Abstract

Real-world environments are inherently dynamic, yet most point cloud processing methods assume static scenes. This fundamental limitation restricts the deployment of 3D perception systems in applications requiring temporal understanding, from autonomous navigation to human-robot interaction. This thesis presents three techniques for dynamic point cloud processing, each operating at a distinct level of representation—high-level semantic understanding, intermediate-level motion analysis, and low-level geometric reconstruction. While developed independently to address specific challenges at each level, these techniques collectively advance the state-of-the-art in dynamic 3D scene understanding.

At the highest level of semantic representation, where low-cost deployment is the primary challenge, we introduce Kinet—a kinematics-inspired neural network that seamlessly extends static point cloud architectures to dynamic scenes. By generalizing space-time surfaces to the feature space, Kinet achieves state-of-the-art accuracy (93.27% on MSRAction-3D) with minimal computational overhead (3.20M parameters, 10.35G FLOPs), enabling practitioners to leverage existing infrastructure with minor modifications and without expensive scene flow computation.

At the intermediate level of motion representation, where annotation-free generalization is paramount, we develop an unsupervised part-level SE(3)-equivariant approach for rigid segmentation and motion estimation. This technique exploits geometric priors and inherent structure in multi-body motion through self-supervised learning, achieving competitive performance (63.8% AP on SAPIEN) with an exceptionally lightweight architecture (0.25M parameters, 0.92G FLOPs), eliminating the need for expensive manual annotations while generalizing robustly to unseen object categories.

At the low level of geometric representation, where data efficiency is critical, we present MotionStruct4D for video-to-4D generation from single-view inputs. Through self-supervised motion structure discovery and weighted dense-to-sparse optimization, this method reconstructs complete 4D representations from minimal observations, achieving superior performance on our challenging ARTEMIS-DFA benchmark while providing interpretable motion decomposition.

Each technique addresses a distinct primary challenge at its respective level: low-cost deployment for high-level semantic tasks, annotation-free generalization for intermediate-level motion analysis, and data efficiency for low-level reconstruction. While maintaining high quality across all approaches, these targeted solutions demonstrate that effective dynamic point cloud processing requires specialized methods tailored to the unique demands of each representational level.

Extensive experiments on synthetic and real-world datasets validate the effectiveness of our techniques. By prioritizing both performance and practical deployability, our contributions provide researchers and practitioners with efficient tools for different aspects of dynamic 3D understanding. Though developed independently, these techniques are thematically connected by the shared goal of advancing dynamic point cloud representation learning.

Contents

List of Figures

List of Tables

1 | Introduction

1.1 Motivation

The cornerstone of enhanced machine autonomy lies in the capacity to comprehend and interact with the three-dimensional world. While robotics has historically relied on 2D representations from cameras due to cost, native 3D sensors such as LiDAR and radar offer richer, metric spatial data capture. Among various 3D data representations including volumetric grids and meshes, point clouds stand out for their inherent flexibility, compactness, and resilience against quantization artifacts. This advantage stems from their representation as raw coordinates, unconstrained by the fixed resolution limitations that are characteristic of grids or meshes. Consequently, point clouds have emerged as the preferred format across numerous applications in 3D sensor data processing, encompassing autonomous driving, intelligent robotics, and human-robot interaction. The evolution of deep neural networks has facilitated significant advancements in static point cloud representation learning [1–13].

Nevertheless, the dynamic nature of real-world environments, which encompasses both spatial and temporal dimensions, presents formidable challenges for static models processing time-evolving point clouds. The absence of temporal encoding in these models hampers the comprehension of inter-frame context underlying scene dynamics, necessitating specialized architectures for dynamic point cloud processing. Real-world scenarios involve continuous motion, articulated deformations, and complex interactions between multiple objects, all of which require sophisticated temporal modeling capabilities that static approaches cannot provide.

Despite substantial progress in deep static models, research on deep learning for dynamic point cloud sequences remains in its nascent stages, with limited studies focusing on specific domains such as recognition [14–18], prediction [19–23], and scene flow estimation [24–26]. This research gap motivates our endeavor to develop deep learning techniques that address distinct yet important aspects of dynamic point cloud processing, spanning semantic classification, motion analysis, and geometric reconstruction.

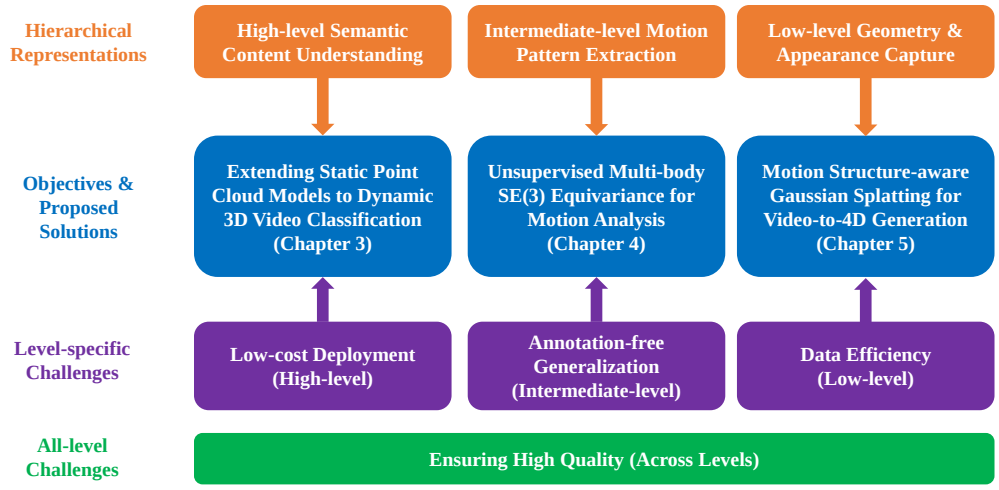


Figure 1.1: *Three proposed techniques for dynamic point cloud processing at different representation levels.* This diagram illustrates our collection of techniques (blue boxes) operating at three distinct representation levels (orange boxes), and the associated challenges addressed. All-level challenges (green box) of ensuring high quality span across all levels, while level-specific challenges (purple boxes) vary by level: low-cost deployment at the high level, annotation-free generalization at the intermediate level, and data efficiency at the low level. These techniques were developed independently to address specific challenges at each level; they are connected thematically rather than architecturally.

1.2 Problem Statement & Research Questions

The challenge of learning from dynamic point cloud sequences can be approached from multiple angles, depending on the level of abstraction at which representa-

tions are sought. In this thesis, we address three such levels through complementary but largely independent research contributions. It is important to note that Chapters 3–5 represent distinct research threads, each with its own technical approach, datasets, and evaluation criteria. They do not form a tightly coupled pipeline or a single integrated architecture; rather, they are connected by the shared theme of advancing dynamic point cloud understanding. We explore the following overarching research question:

How can we develop effective deep representations for dynamic point clouds that address fundamental challenges at different levels of abstraction?

To address this question, we investigate three sub-questions that correspond to different representational levels. Each sub-question is tackled as an independent research contribution with its own objectives and methodology:

High-level Representations: How can we understand semantic content in dynamic scenes? At the highest level of abstraction, our goal is to enable machines to understand dynamic scenes—recognizing specific behaviors, activities, or events occurring within temporal sequences. This involves developing models that can classify entire point cloud videos into meaningful semantic categories, such as distinguishing between different types of human actions, animal behaviors, object interactions, or hand gestures. The challenge lies in aggregating complex spatio-temporal information into discriminative features that capture the essence of dynamic events.

Intermediate-level Representations: How can we extract motion patterns in dynamic point clouds? To understand object dynamics, systems must be capable of analyzing and decomposing the motion patterns of individual components within dynamic scenes. This intermediate level of representation focuses on un-

derstanding how different parts of objects or scenes move over time, involving the segmentation of dynamic scenes into meaningful motion units and the estimation of their individual transformations. The key challenges include handling multi-body rigid motions, dealing with articulated deformations, and maintaining consistency across temporal sequences while preserving fine-grained motion details.

Low-level Representations: How can we capture geometry and appearance information to encode complete 3D dynamics? At the foundational level, generating complete 4D representations from limited observations requires detailed reconstruction of the entire dynamic scene, including its precise geometry, clear appearance, and temporal evolution. This is particularly challenging in scenarios involving occlusions, complex interactions, restricted viewpoints, or varying illumination conditions. The low-level representation must capture not only the static geometric properties of objects but also their dynamic appearance changes, deformations, and the underlying structure that governs their temporal behavior.

1.3 Key Challenges

As shown in Figure 1.1, each technique faces distinct challenges. While ensuring high quality is important across all levels, each level has its own specific primary challenge that drives the design of targeted solutions.

1.3.1 Cross-level Quality Requirements

Ensuring High Quality The requirement for high-quality outputs is essential for each technique, though it manifests differently at each level. At the *high level*, achieving accurate semantic classification requires models to effectively aggre-

gate spatio-temporal information while handling the geometric irregularity and temporal dynamics of point clouds. At the *intermediate level*, accurate motion segmentation and estimation must handle varying point cloud densities and articulated objects with multiple moving parts. At the *low level*, reconstructing complete 4D representations from minimal observations presents severe ill-posed problems, requiring inference of plausible geometry and motion for entirely unobserved regions while maintaining physical realism and temporal coherence.

Each technique is developed independently, and maintaining high quality standards is essential for ensuring reliable and competitive performance.

1.3.2 Level-specific Primary Challenges

Each representational level faces a distinct primary challenge that drives the design of targeted solutions.

Low-cost Deployment (High Level) At the high level, the critical challenge is enabling low-cost deployment that minimizes both computational resources and integration overhead. Dynamic point cloud processing inherently involves complex spatio-temporal computations that can quickly become computationally prohibitive, particularly when processing long sequences or high-resolution data. The computational burden grows significantly—rather than linearly—with sequence length and resolution. Beyond raw computational demands, deployment costs also encompass the practical challenges of integrating new methods into existing systems. The ideal solution must balance temporal modeling capabilities with practical deployment constraints, avoiding computationally expensive operations like dense scene flow estimation or memory-intensive temporal architectures. Solutions should integrate seamlessly with existing frameworks, leverage available pre-trained models where possible, and require minimal architectural modifica-

tions.

Annotation-free Generalization (Intermediate Level) At the intermediate level, the primary challenge is achieving annotation-free generalization that enables robust performance without expensive manual labels. Rigid-part segmentation and motion estimation traditionally require frame-by-frame annotation of individual object components and their transformations, representing a substantially greater annotation burden compared to sequence-level labels needed for high-level tasks. Moreover, the model must generalize across diverse object categories, including household items, tools, and materials unseen during training, and handle arbitrary motion patterns without category-specific supervision. This challenge necessitates developing self-supervised learning strategies that can discover motion structures from the data itself, leveraging geometric priors and physical constraints.

Data Efficiency (Low Level) At the low level, the fundamental challenge is data efficiency—generating complete 4D representations from minimal input data. While traditional 4D reconstruction assumes multi-view observations or dense temporal sampling, practical applications often provide only single-view monocular videos with limited frames. This extreme data scarcity requires methods that can efficiently exploit available information while intelligently inferring missing geometry and motion. The challenge involves not just interpolating between observed states but synthesizing plausible geometry for entirely unobserved regions, such as the back side of objects or occluded parts.

1.4 Proposed Solutions

Guided by the analysis in Figure 1.1, this thesis presents three technical contributions, each addressing a distinct challenge in dynamic point cloud processing.

These contributions are independent in their technical design and can be understood separately, though they collectively cover a broad spectrum of dynamic 3D understanding tasks. Figure 1.2 illustrates the input-output relationships of our three proposed techniques, demonstrating how each method operates on different data modalities and produces distinct representations tailored to their specific objectives.

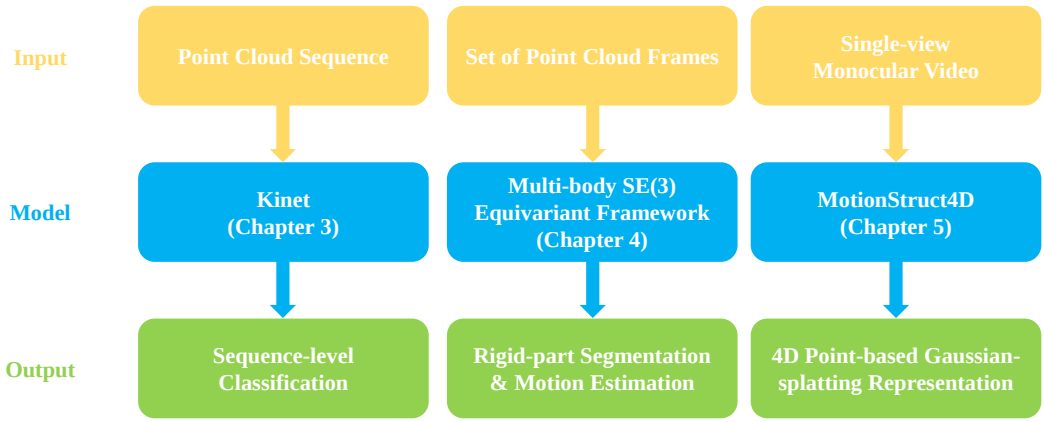


Figure 1.2: *Input-output relationships of the three proposed techniques for dynamic point cloud processing.* Each column shows the data flow from input (light yellow) through the proposed model (light blue) to output (light green). Kinet (Chapter 3) processes point cloud sequences for high-level classification, the Multi-body SE(3) Equivariant Framework (Chapter 4) analyzes sets of point cloud frames for intermediate-level motion understanding, and MotionStruct4D (Chapter 5) transforms single-view monocular videos into low-level 4D representations.

1.4.1 High-level Representations: Extending Static Point Cloud Models to Dynamic 3D Video Classification

At the high level, where low-cost deployment is the primary challenge, we develop Kinet—a kinematics-inspired neural network that seamlessly extends existing static architectures to handle temporal sequences. As shown in the first column of Figure 1.2, Kinet takes a point cloud sequence as input and outputs a sequence-

level classification label for the entire video. The primary objective is to enable dynamic point cloud video classification while minimizing both computational overhead and integration complexity.

The key innovation lies in generalizing space-time surfaces to the feature space, allowing temporal information to be incorporated through lightweight kinematic learning units that preserve the core static architecture. By unrolling the ST-normal solver differentiably in the feature space, Kinet implicitly encodes feature-level dynamics without explicitly computing point-wise correspondences. This approach achieves state-of-the-art accuracy (93.27% on MSRAction-3D) with only 3.20M parameters and 10.35G FLOPs—a fraction of the computational cost required by scene flow-based alternatives, making it suitable for real-time deployment in resource-constrained settings. More importantly, the framework enables practitioners to leverage their existing infrastructure with minimal modifications, preserving investments in pre-trained models and established codebases. The detailed methodology and experimental validation of this approach are presented in **Chapter 3**.

1.4.2 Intermediate-level Representations: Unsupervised Multi-body SE(3) Equivariance for Motion Analysis

At the intermediate level, where annotation-free generalization is paramount, we develop an unsupervised part-level SE(3)-equivariant framework for rigid segmentation and motion estimation. As depicted in the second column of Figure 1.2, our method takes a set of point cloud frames as input and outputs two complementary results: rigid-part segmentation masks for each frame and part-level rigid motion between frame pairs. The focus is on eliminating manual annotation requirements while achieving robust generalization across diverse object categories

and motion patterns.

The primary innovation lies in exploiting geometric equivariance properties to decompose multi-body motion without supervision. Our architecture features two interconnected lightweight heads: an invariant segmentation head that aggregates SE(3)-equivariant features into point-level invariant representations for mask prediction, and an equivariant motion head that computes part-level correlations for transformation estimation. By leveraging the inherent structure in rigid transformations and developing self-supervised training strategies that exploit the interrelationships among scene flow, segmentation masks, and rigid transformations, our approach transforms motion analysis from an annotation-intensive task to an unsupervised learning problem. The lightweight architecture (0.25M parameters, 0.92G FLOPs) achieves 63.8% AP on SAPIEN rigid segmentation without any manual labels, demonstrating strong generalization to unseen object categories through geometric priors rather than learned category-specific features. The comprehensive methodology and experimental evaluation are detailed in **Chapter 4**.

1.4.3 Low-level Representations: Motion Structure-aware Gaussian Splatting for Video-to-4D Generation

At the low level, where data efficiency is the fundamental challenge, we present MotionStruct4D—a novel approach for video-to-4D generation from single-view monocular inputs. As illustrated in the third column of Figure 1.2, MotionStruct4D processes a single-view monocular video captured from a certain camera viewpoint and generates a complete 4D Gaussian Splatting point-based representation that can be rendered from arbitrary viewpoints and timestamps. The objective is to reconstruct comprehensive 4D content while working with minimal observational data.

The key innovation centers on discovering and exploiting motion structures within 3D Gaussian Splatting representations. Our method establishes canonical 3D Gaussians and models their time-dependent deformations through a two-stage process: first identifying quasi-rigid parts via self-supervised spatio-temporal relationship preservation, then decomposing motion into rigid transformations and non-rigid deformations. Through weighted dense-to-sparse optimization that transitions from dense per-Gaussian parameters to sparse control points, our method efficiently balances rigid and non-rigid motion components. MotionStruct4D achieves superior performance on the ARTEMIS-DFA benchmark (PSNR: 30.68, FVD: 769.39) while providing interpretable motion decomposition. The approach transforms the severely ill-posed problem of single-view 4D reconstruction into a tractable optimization task. The detailed framework and experimental validation are presented in **Chapter 5**.

1.5 Thesis Outline

The remainder of this thesis presents our three proposed techniques and their comprehensive evaluation. Chapters 3–5 address distinct research problems and can be read independently; they are connected by the shared theme of dynamic point cloud representation learning rather than by architectural or data dependencies:

Chapter 2 — establishes a comprehensive taxonomy of tasks related to dynamic point clouds, providing a thorough review of state-of-the-art approaches. The taxonomy is structured from point-level, object-level, and video-level perspectives, offering a multifaceted analysis of the field and contextualizing the techniques developed in subsequent chapters.

Chapter 3 — addresses high-level representation through Kinet, demonstrating how static models can be efficiently extended for dynamic classifi-

cation while maintaining low-cost deployment through minimal computational overhead and seamless integration.

Chapter 4 — tackles intermediate-level representation through unsupervised multi-body SE(3) equivariant motion analysis, showing how annotation-free generalization can be achieved through self-supervised learning and geometric priors.

Chapter 5 — solves low-level representation challenges through MotionStruct4D, demonstrating how data-efficient 4D generation can be achieved from minimal single-view inputs through motion structure discovery.

Chapter 6 — concludes the contributions across all three levels, discussing how the proposed techniques address fundamental challenges in dynamic point cloud processing, their complementarity, and future research directions.

1.6 Publications

1.6.1 Publications Directly Linked to Thesis Contributions

The primary contributions of this thesis are encapsulated in the following publications, each addressing specific aspects of dynamic point cloud representation learning. Chapters 3–5 are based on these publications: while the core technical content is drawn from the respective papers, each chapter has been adapted for this thesis with expanded introductions providing additional context, transition paragraphs linking to the broader thesis theme, and supplementary discussion where appropriate.

1. **Jia-Xing Zhong**, Kaichen Zhou, Qingyong Hu, Bing Wang, Niki Trigoni,

Andrew Markham: *No Pain, Big Gain: Classify Dynamic Point Cloud Sequences with Static Models by Fitting Feature-level Space-time Surfaces*. Conference on Computer Vision and Pattern Recognition 2022 (CVPR 2022)

2. **Jia-Xing Zhong**, Ta Ying Cheng, Yuhang He, Kai Lu, Kaichen Zhou, Andrew Markham, Niki Trigoni: *Multi-body SE(3) Equivariance for Unsupervised Rigid Segmentation and Motion Estimation*. Thirty-seventh Conference on Neural Information Processing Systems (NeurIPS 2023)

3. **Jia-Xing Zhong**, Kai Lu, Jiaojiao Ye, Niki Trigoni, Andrew Markham: *MotionStruct4D: Discovering Motion Structure of Gaussian Splatting for Video-to-4D Generation*. Under review

These three core contributions address the research questions outlined in Section 1.2, providing solutions at the high-level (video classification), intermediate-level (motion segmentation and estimation), and low-level (4D geometry reconstruction) representations, respectively.

As first author on all of the three primary publications, I led the research direction, technical development, and dissemination of results. The following statements clarify my specific contributions to each work, distinguishing my role from that of co-authors who provided valuable support through discussions, feedback, and collaborative refinement of ideas.

Publication 1 - Kinet (CVPR 2022): As the first author, I was responsible for:

- Conceptualizing and designing the kinematics-inspired neural network architecture
- Developing the ST-surface generalization to feature space
- Implementing the complete framework in TensorFlow (code available at https://github.com/jx-zhong-for-academic-purpose/Kinet)

- Designing and conducting all experiments across four datasets
- Writing the manuscript and creating all visualizations

Publication 2 - Multi-body SE(3) Equivariance (NeurIPS 2023): As the first author, I contributed:

- Designing the part-level SE(3)-equivariant architecture
- Developing the unsupervised training strategy
- Implementing the complete pipeline in PyTorch (code available at https://github.com/jx-zhong-for-academic-purpose/Multibody_SE3)
- Conducting experiments and ablation studies
- Writing the paper with co-authors providing feedback on experimental design

Publication 3 - MotionStruct4D (Under Review): As the first author, I was responsible for:

- Proposing the motion structure discovery approach
- Designing the weighted dense-to-sparse optimization strategy
- Curating the benchmark dataset of 4D generation
- Implementing the complete framework
- Conducting all experiments and writing the manuscript

1.6.2 Other Papers Published during DPhil Study in Related Areas

In addition to the primary contributions that directly address the research challenges delineated in Section 1.2, substantial contributions have been made to several collaborative publications. These papers focus on relevant topics but are not directly linked to the chapters of this thesis:

4. Kaichen Zhou, **Jia-Xing Zhong**, Sangyun Shin, Kai Lu, Yiyuan Yang, Andrew Markham, Niki Trigoni: *DynPoint: Dynamic Neural Point For View Synthesis*. Thirty-seventh Conference on Neural Information Processing Systems (NeurIPS 2023)

5. Kai Lu, **Jia-Xing Zhong**, Bo Yang, Bing Wang, Andrew Markham: *Learning to Catch Reactive Objects with a Behavior Predictor*. 2024 IEEE International Conference on Robotics and Automation (ICRA 2024)

6. Qian Xie, Ta-Ying Cheng, **Jia-Xing Zhong**, Kaichen Zhou, Andrew Markham, Niki Trigoni: *Beyond Fusion: Modality Hallucination-based Multispectral Fusion for Pedestrian Detection*. IEEE/CVF Winter Conference on Applications of Computer Vision (WACV 2024)

7. Jiaojiao Ye, **Jia-Xing Zhong**, Qian Xie, Yuzhou Zhou, Andrew Markham, Niki Trigoni: *DA-VLM: Data Factory with Minimal Effort Using VLMs*. Under review at International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2026)

8. Yuhang He, Shitong Xu, **Jia-Xing Zhong**, Sangyun Shin, Niki Trigoni, Andrew Markham: *SPEAR: Receiver-to-Receiver Acoustic Neural Warping Field*. arXiv:2406.11006

1.7 Open-source Software Contributions

Beyond the theoretical and methodological contributions, this thesis includes substantial software engineering efforts that enable reproducibility and facilitate future research. All implementations prioritize computational efficiency, modularity, and ease of deployment.

1.7.1 Released Codebases

Kinet Framework (Chapter 3): A complete TensorFlow implementation of our kinematics-inspired network for dynamic point cloud classification, publicly available at https://github.com/jx-zhong-for-academic-purpose/Kinet. Key features include:

- Efficient implementation of the differentiable ST-normal solver with only 3.20M parameters and 10.35G FLOPs overhead
- Training and evaluation scripts for benchmark datasets
- Comprehensive documentation with usage examples and pre-trained model weights
- Visualization tools for Point Activation Clouds (PACs) and feature analysis

Multi-body SE(3) Framework (Chapter 4): PyTorch implementation of our unsupervised rigid segmentation and motion estimation approach, publicly available at https://github.com/jx-zhong-for-academic-purpose/Multibody_SE3. The codebase provides:

- Lightweight SE(3)-equivariant feature extractor based on EPN (0.25M parameters, 0.92G FLOPs)

- Dual-head architecture for point-level invariant segmentation and part-level equivariant motion estimation
- Self-supervised training pipeline exploiting interrelationships among scene flow, segmentation, and rigid transformations
- Evaluation tools for rigid segmentation metrics (AP, PQ, F1, mIoU, RI) and motion estimation metrics (EPE3D, AccS, AccR, Outlier)
- Pre-processing scripts and data loaders for multiple formats of datasets

1.7.2 Engineering Contributions and Design Principles

The software contributions represent significant engineering effort guided by three core principles:

Computational Efficiency: All implementations are designed for practical deployment:

- Kinet adds minimal overhead (10.35G FLOPs) compared to flow-based alternatives (>150G FLOPs)
- The SE(3) framework achieves state-of-the-art performance with only 0.25M parameters

Reproducibility: Detailed specifications ensure exact replication of results:

- Complete hyperparameter configurations documented in code and appendices
- Deterministic training procedures with fixed random seeds
- Standardized data preprocessing and augmentation pipelines
- Pre-trained model checkpoints for all major experiments

1.7.3 Impact and Adoption

These software artifacts lower the barrier to entry for dynamic point cloud research and provide practical tools for real-world deployment. The released codebases enable:

- Rapid prototyping of dynamic 3D perception systems for robotics applications
- Benchmarking of new methods against our established baselines
- Educational use in teaching advanced topics in 3D computer vision
- Industrial deployment in autonomous systems requiring real-time dynamic scene understanding

The modular nature of our implementations allows researchers to leverage individual components in their own work, amplifying the impact beyond the specific applications demonstrated in this thesis.

2 | Literature Review

2.1 Introduction

The rapid advancement of 3D sensing technologies, driven by continued miniaturization and mass production, has fundamentally transformed how machines perceive and interact with three-dimensional environments [27]. At the heart of this transformation lies the point cloud—a fundamental data structure consisting of discrete spatial coordinates that directly captures the raw geometric information from 3D sensors [1]. Unlike structured representations such as images or voxels, point clouds preserve the inherent geometric fidelity of sensed data without imposing artificial discretization or topology.

The deep learning revolution has brought remarkable progress to static point cloud processing. Pioneering architectures have achieved impressive results across diverse tasks: shape classification methods [1, 2, 28] can now recognize complex 3D objects with high accuracy; object detection frameworks [6–8] enable precise localization in cluttered scenes; and semantic segmentation approaches [12, 29, 30] can parse entire environments into meaningful components. This success has been catalyzed by the emergence of large-scale, high-quality datasets [3, 9–11] and increasingly sophisticated neural architectures designed specifically for unstructured point data [1, 2, 28, 31].

However, a fundamental limitation persists: these methods assume static, unchanging environments. Real-world scenarios are inherently dynamic—objects move, scenes evolve, and temporal relationships carry critical information that static models cannot capture [32]. This gap between static processing capabilities and dynamic reality has motivated an emerging body of research addressing

temporal point cloud understanding. Recent efforts have begun exploring various aspects of this challenge, including action recognition [14–16, 33], motion prediction [19, 20, 34], scene flow estimation [25, 35–38], and motion segmentation [39, 40], though the field remains in its early stages compared to static processing.

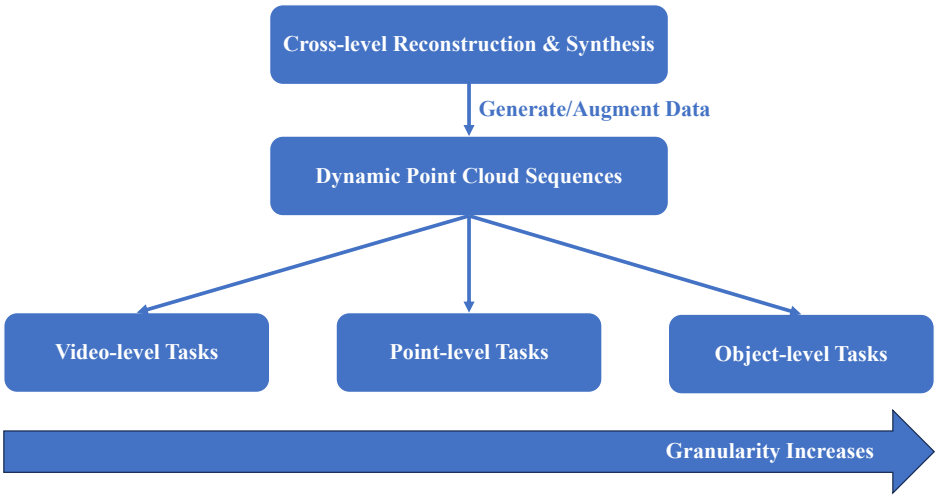


Figure 2.1: *Conceptual data flow and task hierarchy in dynamic point cloud processing.* Cross-level reconstruction and synthesis methods serve as foundational data generators. They produce the dynamic point cloud sequences that are then processed by task-specific approaches at three primary levels of increasing output granularity: coarse video-level predictions, intermediate object-level analysis, and fine-grained point-level outputs.

This chapter provides a comprehensive review of deep learning approaches for dynamic point cloud processing. We structure this review according to the conceptual relationships and data flow between different task categories, as illustrated in Figure 2.1. At the foundation are the cross-level tasks of **reconstruction and synthesis**, which serve as data generation pipelines. These methods produce or augment the dynamic point cloud sequences that form the input for a spectrum of analytical tasks. These subsequent tasks can be organized by their level of output granularity, ranging from coarse, sequence-level predictions to fine-grained, per-point analysis. This hierarchy, where granularity increases from holistic scene un-

derstanding to detailed component analysis, reflects both the computational complexity and the richness of information at each level. **To contextualize the evolution of techniques,** Table 2.1 summarizes representative methods across levels, their first-publication year, and the core modeling mechanism (3D CNN, graph models, LSTM, transformers, sparse 4D CNNs, Gaussian splatting, *etc*.).

The remainder of this chapter is structured as follows. Section 2.2 establishes foundational knowledge by examining various 3D data representations and their suitability for dynamic scenarios. Section 2.3 explores video-level classification methods that produce global predictions for entire sequences. Section 2.4 examines object-centric approaches for detection and tracking. Section 2.5 delves into fine-grained, point-level tasks including segmentation, flow estimation, and prediction. Section 2.6 covers reconstruction and synthesis methods that generate dynamic point cloud data. Section 2.7 discusses key datasets driving progress in the field.

2.2 Overview of 3D Data Representations

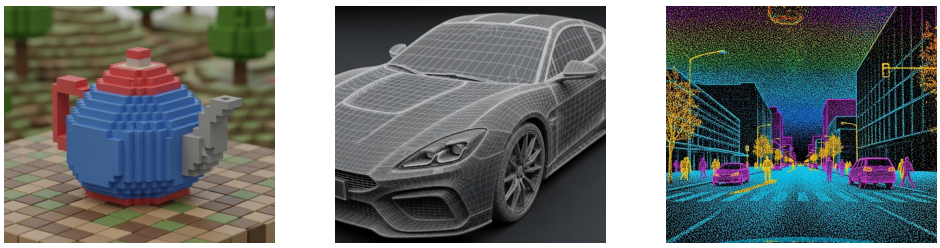
Before examining specific approaches to dynamic point cloud processing, it is essential to understand the fundamental characteristics of different 3D data representations. The choice of representation profoundly impacts algorithmic design, computational efficiency, and the types of information that can be effectively processed—considerations that become even more critical when handling temporal data.

2.2.1 Voxel Grids

Voxel grids extend the familiar concept of 2D pixels into three dimensions, discretizing 3D space into a regular grid of volumetric elements [69]. As illustrated

Table 2.1: *Chronology and mechanisms of representative methods for dynamic point cloud understanding and generation.* The table lists each method's level, first-publication year, core mechanism, and brief notes.

Level	Method	First pub.	Core mechanism	Representative task / notes
Video-level	MeteorNet [14]	2019	Spatio-temporal 3D convolutions via cell/neighbor grouping	Action recognition on point sequences
Video-level	3DV [41]	2020	Multi-stream (motion vs. appearance) with dynamic voxel extraction	Decoupled dynamics; sequence classification
Video-level	PSTNet [15]	2021	Point Spatio-Temporal convolution (PST-Conv) for irregular sets	Efficient spatio-temporal aggregation
Video-level	PointLSTM [17]	2020	Point-wise recurrent units (LSTM)	Long-range temporal modeling
Video-level	P4Transformer [16]	2021	Transformer with space-time self-attention over points	Global dependency modeling
Video-level	Spatial-Temporal Transformer [42]	2021	Efficient spatial-temporal attention blocks	Scalable long-sequence modeling
Object-level	SC3D [43]	2019	Siamese metric learning on point features	3D single-object tracking
Object-level	P2B [44]	2020	Point-to-box regression from target-consistent points	Robust LiDAR tracking
Object-level	Graph-based Online Detector [45]	2020	Spatio-temporal graphs with message passing	Online detection under occlusion
Object-level	SparseConv + LSTM [22]	2020	3D sparse U-Net with recurrent temporal state	Online detection/tracking in streams
Object-level	Center-based (anchor-free) [46]	2021	Center heatmaps + attribute regression	Efficient detection/tracking association
Object-level	Auto-labeling (offline) [47]	2021	Sequence-level optimization using future frames	High-quality sequence annotations
Point-level (Seg/Flow)	Minkowski 4D CNNs [48]	2019	Sparse 4D convolutions on occupied voxels	Efficient dense prediction
Point-level (Seg)	ASAP-Net [49]	2020	Attention-based temporal fusion + spatial correction	Temporal consistency in segmentation
Point-level (Seg)	Temporal Memory Align [50]	2020	Feature alignment with a temporal memory bank	Long sequence stability
Point-level (Motion Seg)	DeepPart [51]	2019	Mobility-aware learning; iterative corres.+part assignment	Category-agnostic motion parts
Point-level (Motion Seg)	MultiBodySync [39]	2021	Geometric synchronization across rigid bodies	Unsupervised rigid-part discovery
Point-level (Motion Seg)	OGC [40]	2022	Object geometry constraints	Label-free rigid segmentation
Point-level (Motion Seg)	SLIM [52]	2021	Self-supervised via scene-flow/rigidity links	Driving scenes; reduced labels
Point-level (Flow)	FlowNet3D [25]	2019	Flow embedding layer for correspondence	Supervised scene flow estimation
Point-level (Flow)	PointPWC-Net [35]	2020	Coarse-to-fine pyramids + cost volumes	Efficient large-motion handling
Point-level (Flow)	FLOT [36]	2020	Optimal transport-based matching	Principled correspondence
Point-level (Flow)	FH-Net [37]	2022	High-frequency preserving cues	Detail-aware flow
Point-level (Flow)	HPLFlowNet [26]	2019	Permutohedral lattice convolutions	Fast hierarchical flow
Point-level (Flow, Self-sup.)	Cycle consistency [24]	2020	Forward-backward cycle constraints	No dense labels needed
Point-level (Flow, Weak-sup.)	Weakly-supervised Flow [53]	2021	Object-level labels as weak signals	Reduced annotation cost
Prediction	PointRNN [19]	2019	Recurrent point-wise state update	Future point cloud prediction
Prediction	SPFNet [23]	2020	Scene flow + trajectory fusion	Long-horizon forecasting
Prediction	CloudLSTM [54]	2020	Density-aware LSTM for variable sampling	Robust to point sparsity
Prediction	Flow-based extrapolation [55]	2020	Additive scene flow update	Simple/efficient baseline
Reconstruction / Synthesis	OFlow [56]	2019	Time-conditioned implicit occupancy	4D reconstruction with topology changes
Reconstruction / Synthesis	CASPR [20]	2020	Canonical space + deformation fields	Factor shape/motion; completion
Reconstruction / Synthesis	D-NeRF / Nerfies [57, 58]	2021/2021	Dynamic NeRF with deformation	Dynamic novel-view synthesis
Reconstruction / Synthesis	Deformable 3D Gaussians [59]	2024	Deformable 3D Gaussian splats	Explicit, editable dynamics
Reconstruction / Synthesis	4DGS / SC-GS [60, 61]	2023/2024	4D Gaussian splatting; sparse control	Real-time rendering; longer sequences
Reconstruction / Synthesis	DreamGaussian4D / Consistent4D [62, 63]	2023/2023	SDS-guided 4D GS (text/image)	4D generation from minimal inputs
Reconstruction / Synthesis	SV4D / STAG4D [64, 65]	2024/2024	Multi-view video diffusion + 3D lifting	Single-view-to-4D synthesis
Reconstruction / Synthesis	DreamMesh4D [66]	2024	Hybrid meshes bound to GS control	Controllable motion synthesis
Reconstruction / Synthesis	L4GM / Efficient4D [67, 68]	2024/2024	Learned 4D generative models; fast optimization	Diverse scenes; efficiency-focused
Reconstruction / Synthesis	4DComposition [34]	2021	Compositional parts + motion priors	Completion; part-wise control



(a) Voxel Grid (b) Polygon Mesh (b) Point Cloud

Figure 2.2: *Visual comparison of 3D data representations.* (a) Voxel grids discretize 3D space into regular volumetric elements. (b) Polygon meshes represent surfaces through interconnected vertices and faces. (c) Point clouds maintain unordered sets of 3D coordinates directly from sensor measurements.

in Figure 2.2(a), each voxel typically stores binary occupancy information, though richer attributes like color, density, or learned features can also be encoded.

Advantages: The regular grid structure of voxels enables direct application of well-established 3D convolutional neural networks [70, 71], leveraging decades of research in convolutional architectures. This regularity also simplifies many geometric operations and provides explicit volumetric representation. Recent work has demonstrated the effectiveness of voxel-based approaches in complex tasks, from object detection [70] to dynamic scene processing with 4D Minkowski networks [72].

Limitations: The primary challenge lies in memory consumption, which scales cubically ($O(N^3)$) with resolution. This creates a fundamental trade-off: high-resolution grids necessary for capturing fine details quickly become computationally prohibitive, while coarser grids sacrifice geometric precision. Additionally, voxel representations inherently introduce quantization artifacts that can obscure subtle geometric features.

Dynamic Considerations: For temporal sequences, the storage challenge is compounded—each frame requires its own voxel grid, leading to potentially massive memory requirements. While sparse representations like octrees [71] and special-

ized libraries [48, 73] can mitigate storage for static scenes, maintaining consistency in sparse structures across time when topology changes (e.g., objects entering/leaving the scene) introduces additional complexity. Recent approaches like 4D Minkowski ConvNets [48] have made progress in efficiently processing sparse 4D data, but challenges remain.

2.2.2 Polygon Meshes

Meshes represent 3D surfaces through interconnected vertices, edges, and faces (typically triangles), providing an efficient and flexible representation for complex geometries, as shown in Figure 2.2(b).

Advantages: Meshes excel at representing surfaces with arbitrary topology while maintaining computational efficiency. They can capture intricate geometric details with relatively few primitives and provide well-defined surface properties (normals, curvature, *etc.*). The widespread adoption in computer graphics has led to mature rendering pipelines and extensive tool support. Recent advances in mesh-based generation [74, 75] have further demonstrated their effectiveness for high-quality 3D content creation.

Limitations: The irregular connectivity of meshes poses significant challenges for deep learning. Unlike regular grids, meshes require specialized architectures such as graph neural networks [31] or techniques from geometric deep learning. Furthermore, generating high-quality meshes from raw sensor data remains a complex preprocessing step, often requiring sophisticated reconstruction algorithms.

Dynamic Considerations: Dynamic mesh sequences present unique challenges. When topology remains fixed (only vertex positions change), storage can be efficient. However, real-world dynamics often involve topological changes—objects

splitting, merging, or undergoing complex deformations. Handling such changes consistently across time requires sophisticated tracking and remeshing strategies [57, 58]. Recent hybrid approaches like DreamMesh4D [66] have explored combining meshes with other representations for dynamic scene modeling.

2.2.3 Point Clouds

Point clouds represent 3D geometry directly from sensors like LiDAR, depth cameras, and photogrammetric systems [1], as unordered sets of points $\{P_i = (x_i, y_i, z_i, \text{attributes}_i)\}_{i=1}^{N}$, as depicted in Figure 2.2(c).

Advantages: Point clouds offer several compelling benefits. They directly represent sensor measurements without imposing artificial structure or topology, avoiding quantization artifacts. Their flexibility allows representation of complex, non-manifold geometries that would be challenging for meshes. Additionally, they require no complex preprocessing—raw sensor data can be processed directly.

Limitations: The unstructured nature of point clouds—lacking explicit connectivity or consistent ordering—prevents direct application of standard CNNs. This has motivated the development of specialized architectures like PointNet [1], which uses symmetric functions to achieve permutation invariance, and subsequent improvements incorporating local structure [2, 28, 31]. Additionally, varying point density and the absence of explicit surface information can complicate certain tasks.

Dynamic Considerations: Temporal point cloud sequences amplify both advantages and challenges. Storage requirements scale linearly with the number of points and frames, which can be substantial for dense, long sequences. The lack of consistent point ordering across frames makes tracking individual points challenging, necessitating sophisticated matching algorithms or learning-based

approaches [14, 15, 18]. Scene flow estimation—computing point-wise motion vectors—becomes computationally intensive due to the need for correspondence estimation [25, 36–38, 76].

2.2.4 Emerging Representations

Recent advances have introduced novel representations that address limitations of traditional approaches:

Neural Implicit Representations: Neural radiance fields (NeRFs) [77] and their extensions to dynamic scenes [57, 58, 78] represent geometry and appearance through continuous functions parameterized by neural networks. These methods excel at novel view synthesis but require extensive optimization.

3D Gaussian Splatting: Recently emerged as a powerful alternative, 3D Gaussian Splatting [79] represents scenes as collections of 3D Gaussians, enabling real-time rendering while maintaining explicit structure. Extensions to dynamic scenes [59, 60, 80] have shown promising results for 4D reconstruction and generation.

Hybrid Representations: Methods combining multiple representations leverage complementary strengths. For instance, recent 4D generation approaches [66, 81] combine Gaussian splatting with mesh-based deformations or sparse control points for efficient dynamic modeling.

2.2.5 Comparative Analysis for Dynamic Scenarios

Table 2.2 summarizes the key trade-offs among different representations. Each representation offers distinct advantages and limitations:

- **Voxel grids** provide structural regularity ideal for convolutional processing

Table 2.2: Comparison of 3D Data Representations for Dynamic Point Cloud Processing

Representation	**Memory Efficiency**	**Processing Regularity**	**Topology Flexibility**	**Sensor Fidelity**	**Surface Quality**	**Dynamic Handling**
Voxel Grids	Low	High	Low	Medium	Medium	Challenging
Polygon Meshes	High	Low	Medium	Low	High	Complex
Point Clouds	Medium	Low	High	High	Low	Moderate
Neural Implicit	Low	N/A	High	Medium	High	Good
Gaussian Splatting	Medium	Medium	High	Medium	High	Excellent

but suffer from memory scaling issues, particularly acute in 4D scenarios.

- **Meshes** offer efficient surface representation and excellent rendering properties but require complex handling of topological changes over time.
- **Point clouds** preserve raw sensor fidelity and handle arbitrary geometries but demand specialized architectures for processing unstructured data.
- **Neural implicit representations** enable high-quality reconstruction but require extensive per-scene optimization.
- **Gaussian splatting** balances explicit structure with rendering efficiency, showing promise for real-time dynamic applications.

The choice of representation fundamentally shapes the algorithmic approach. This thesis focuses primarily on point cloud representations due to their direct correspondence with sensor data, flexibility in representing diverse geometries, and avoidance of preprocessing artifacts. However, we acknowledge that hybrid approaches combining multiple representations may offer advantages for specific applications.

2.3 Video-level Classification

Video-level classification represents the most abstract level of dynamic point cloud understanding, where models must distill entire temporal sequences into semantic categories. As illustrated in Figure 2.1, this task encompasses recognizing gestures [17, 82], identifying actions [14–16, 33], and classifying dynamic events. The challenge lies in effectively aggregating spatial and temporal information while handling the irregularity of point cloud data.

2.3.1 Architectural Approaches

The evolution of video-level classification methods reflects diverse strategies for capturing spatio-temporal patterns:

3D Convolutional Approaches: Extending successful 2D video architectures [83, 84], MeteorNet [14] pioneered the application of 3D convolutions to point cloud sequences. By grouping points into spatial cells and tracking their evolution over time, MeteorNet performs spatio-temporal aggregation that captures both spatial patterns and temporal dynamics. Similarly, Wang et al. [85] introduced anchor-based spatio-temporal attention within 3D convolutions, allowing the model to focus on discriminative regions across space and time. The 3DV approach [41] further decouples motion and appearance through multi-stream architectures, though requiring offline dynamic voxel extraction.

Specialized Spatio-Temporal Convolutions: Recognizing the unique challenges of point cloud data, Fan et al. [15] developed Point Spatio-Temporal (PST) convolutions specifically designed for irregular point data. PST-Conv aggregates information from spatio-temporal neighborhoods while respecting the geometric structure of point clouds, achieving superior performance compared to naive extensions

of image-based methods. Recent extensions [18] have further refined these convolutions for improved efficiency and accuracy.

Recurrent Architectures: Drawing inspiration from sequence modeling [86], PointLSTM [17] adapts recurrent structures to point cloud sequences. By maintaining hidden states that encode temporal history and combining them with current observations, PointLSTM can capture long-range dependencies while being memory-efficient for long sequences. The recurrent formulation also allows processing of variable-length sequences, a practical advantage for real-world applications. SequentialPointNet [87] extends this concept with sequential feature learning modules tailored for point cloud data.

Transformer-based Methods: The success of transformers in various domains has motivated their adaptation to dynamic point clouds. P4Transformer [16] employs self-attention mechanisms to model relationships across both spatial points and temporal frames, while PST-Transformer [18] combines the benefits of specialized convolutions with transformer architectures. Wei et al. [42] introduced spatial-temporal transformer blocks that efficiently capture long-range dependencies. These methods excel at capturing global context but require careful design to manage computational complexity, particularly for long sequences or dense point clouds.

2.3.2 Critical Analysis

While these approaches have advanced the field significantly, several limitations persist:

- **Computational Overhead:** Most existing methods require substantial computational resources, with transformer-based approaches being particularly demanding. This limits their deployment in resource-constrained environ-

ments.

- **Generalization Challenges:** Methods trained on specific datasets often struggle to generalize across different sensor modalities or scene types, necessitating extensive retraining.
- **Temporal Resolution Trade-offs:** Balancing temporal resolution with computational efficiency remains challenging—higher temporal resolution improves motion capture but increases computational cost quadratically.
- **Limited Pre-training:** Unlike image-based video understanding that benefits from large-scale pre-trained models, point cloud video classification lacks similar foundation models, limiting transfer learning opportunities.

2.4 Object-level Detection and Tracking

Moving from global sequence understanding to object-centric analysis, this section examines methods that detect and track individual objects within dynamic point cloud sequences. These tasks are fundamental for applications requiring detailed understanding of specific entities, such as autonomous vehicles tracking surrounding traffic or robotic systems monitoring manipulated objects.

2.4.1 Detection in Temporal Context

Object detection in point cloud sequences extends beyond frame-by-frame detection by exploiting temporal continuity and motion patterns. This temporal context can significantly improve detection accuracy and robustness.

Online Detection Frameworks: Yin et al. [45] developed a graph-based framework that processes point cloud streams in real-time. By modeling objects as nodes in a spatio-temporal graph and using message passing to aggregate infor-

mation across frames, their method achieves robust detection even for partially occluded objects. The graph structure naturally handles varying numbers of objects and irregular sampling patterns.

Recurrent Detection Architectures: Huang et al. [22] integrated LSTM networks with 3D sparse convolutions for online object detection. The U-Net-style architecture extracts frame-wise features, while the LSTM module maintains temporal context in its hidden state. This design allows the model to leverage motion patterns and temporal consistency for improved localization, particularly beneficial for predicting future object positions in autonomous driving scenarios.

Offline Optimization with Future Frames: While online methods are crucial for real-time applications, offline approaches can achieve superior accuracy by considering future frames. Qi et al. [47] proposed an auto-labeling framework that processes entire sequences to generate high-quality 3D bounding boxes. By optimizing detections across multiple frames simultaneously, their method resolves ambiguities that single-frame or online methods might miss.

2.4.2 Object Tracking Paradigms

Tracking extends detection by establishing correspondences across frames, requiring models to handle appearance changes, occlusions, and complex motion patterns.

Siamese Tracking Networks: SC3D [43] pioneered 3D point cloud tracking using Siamese networks. By encoding point cloud regions into discriminative features and comparing them across frames, SC3D generates tracking proposals without relying on color or texture information. This approach proves particularly robust for LiDAR data where appearance information is limited.

Point-to-Box Regression: P2B [44] reframes tracking as a point-centric regres-

sion problem. Given a target template and search region, P2B identifies points likely belonging to the target and regresses from these points to the bounding box. This point-based approach naturally handles sparse data and provides interpretable intermediate results.

Anchor-free Tracking: Recent advances in anchor-free detection [46] have been adapted for tracking. By detecting object centers and regressing other attributes directly, these methods simplify the tracking pipeline and improve efficiency. The center-based representation also provides a natural way to associate objects across frames.

2.4.3 Comparative Assessment

The choice between online and offline approaches depends on application requirements:

- **Online methods** enable real-time decision-making but must operate with limited temporal context, achieving typical processing speeds of 10-20 Hz on modern hardware.
- **Offline methods** achieve 15-20% higher accuracy but introduce latency, making them suitable for tasks like dataset annotation or post-processing analysis.
- **Hybrid approaches** that maintain a sliding window of frames offer a middle ground, balancing accuracy and latency for many practical applications.

2.5 Point-level Segmentation, Scene Flow Analysis, and Predictions

This section explores tasks requiring fine-grained, point-wise understanding of dynamic scenes. These dense prediction tasks form the foundation for detailed scene understanding and motion analysis.

2.5.1 Semantic Segmentation in Temporal Context

Semantic segmentation assigns category labels to each point. Extending this task to temporal sequences requires maintaining consistency while adapting to scene changes.

Direct Extensions of Classification Networks: Many video-level classification architectures can be adapted for dense prediction by modifying their output layers. Networks like MeteorNet [14], PSTNet [15], and P4Transformer [16] have demonstrated effectiveness in semantic segmentation by replacing global pooling with point-wise predictions. This approach leverages proven architectures while maintaining temporal consistency.

Temporal Memory Mechanisms: Duerr et al. [50] introduced temporal memory alignment to explicitly maintain consistency across frames. By aligning features between adjacent frames and maintaining a memory bank of past observations, their method handles long sequences while preventing drift in predictions. This approach proves particularly effective for scenarios with gradual scene changes.

Efficient 4D Processing: Choy et al. [48] leveraged sparse convolutions through Minkowski Convolutional Neural Networks, enabling efficient processing of 4D point cloud data. By operating only on occupied voxels and their temporal neigh-

borhoods, MinkNet achieves significant computational savings while maintaining high accuracy. This efficiency is crucial for real-time applications.

Attention-based Temporal Fusion: ASAP-Net [49] incorporates attention mechanisms to selectively fuse features across frames. The attentive temporal embedding layer identifies relevant temporal contexts for each point, while spatial correction ensures consistency within each frame. This selective fusion prevents noise accumulation while preserving important temporal information.

2.5.2 Motion Segmentation

Motion segmentation partitions point clouds based on motion patterns rather than semantic categories, typically identifying rigidly moving parts within articulated objects or multi-body scenes. This task has seen significant advances with the introduction of geometric priors and self-supervised learning.

Learning-based Approaches: DeepPart [51] introduced a learning-based framework for mobility-based segmentation. Through iterative optimization of correspondences, flow, and part assignments, DeepPart can generalize to novel object categories by learning motion patterns rather than appearance. Recent work has extended this to self-supervised settings [88], reducing annotation requirements.

Probabilistic Formulations: Hayden et al. [89] developed a Bayesian nonparametric model that infers part structures and their motions jointly. By modeling uncertainty in both segmentation and motion, their approach handles ambiguous cases more robustly than deterministic methods. SLIM [52] exploits the relationship between scene flow and motion segmentation for self-supervised learning in autonomous driving scenarios.

Geometric Approaches: MultiBodySync [39] formulates motion segmentation as a synchronization problem, leveraging geometric constraints to identify rigid

parts. CAPTRA [90] uses canonicalization to normalize object poses before segmentation, simplifying the motion patterns to be discovered. Recent work [40] introduces object geometry constraints for unsupervised rigid segmentation, while factorization-based methods [91] provide theoretical foundations for multi-body motion analysis.

2.5.3 Scene Flow Estimation

Scene flow represents the 3D motion field of a point cloud, analogous to optical flow in 2D but with additional depth information. Recent advances have significantly improved both accuracy and efficiency of flow estimation.

Unsupervised Methods: Self-supervised approaches avoid the need for expensive flow annotations. Methods exploiting cycle consistency [24] use the principle that points should return to their origin when flow is applied forward and backward as a supervisory signal. Hierarchical estimation approaches have proven particularly effective—PointPWC-Net [35] adapts the PWC-Net architecture from 2D flow, using a coarse-to-fine hierarchy to handle large motions efficiently, while HPLFlowNet [26] further improves efficiency through hierarchical permutohedral lattice convolutions.

Supervised Methods: When ground-truth flow is available, supervised methods achieve higher accuracy. FlowNet3D [25] pioneered end-to-end learning for scene flow, introducing flow embedding layers that capture motion patterns effectively. FLOT [36] formulates flow estimation as an optimal transport problem, a principled framework for establishing correspondences. Recent advances like FH-Net [37] focus on preserving high-frequency details in flow estimation.

Weakly-Supervised Approaches: Recognizing the annotation burden of dense flow labels, researchers have explored middle ground approaches. Gojcic et al.

[53] demonstrated that using object-level annotations rather than dense flow can achieve competitive performance while significantly reducing annotation effort.

2.5.4 Future Frame Prediction

Predicting future point cloud states is crucial for anticipatory systems in robotics and autonomous driving. Recent advances have moved beyond simple flow-based extrapolation to sophisticated generative models.

Flow-based Prediction: The most straightforward approach involves adding estimated scene flow to current points [55]. While computationally efficient, this method assumes constant velocity and struggles with complex motion patterns.

Recurrent Prediction Models: PointRNN [19] pioneered recurrent networks for temporal evolution modeling. SPFNet [23] combines flow estimation with trajectory prediction for improved long-term forecasting. CloudLSTM [54] adapts LSTM architectures specifically for variable point densities across frames.

Generative Approaches: CASPR [20] learns canonical representations that factor shape from motion. Compositional approaches [34] explicitly separate objects and backgrounds for more accurate predictions. Occupancy flow methods [56] predict through implicit representations that naturally handle topology changes.

2.5.5 Performance Comparison and Trade-offs

Different approaches excel in different scenarios:

- **Accuracy vs. Efficiency:** Supervised methods achieve 20-30% higher accuracy but require extensive annotations. Unsupervised methods trade accuracy for practical deployability.
- **Temporal Consistency:** Memory-based approaches maintain better consis-

tency but struggle with sudden scene changes. Attention mechanisms offer adaptive fusion but increase computational cost.

- **Generalization:** Geometric approaches generalize better across object categories but may miss subtle non-rigid motions that learning-based methods capture.

2.6 Cross-level Reconstruction and Synthesis

Reconstruction and synthesis methods serve a unique role in the dynamic point cloud ecosystem—they generate the data that other methods process. As illustrated in Figure 2.1, these approaches create dynamic point cloud representations from various inputs, which can then be consumed by task-specific methods at different levels. This section examines both reconstruction (recovering complete representations from partial observations) and synthesis (generating novel dynamic content).

2.6.1 4D Reconstruction: From Partial to Complete

Reconstruction methods aim to recover complete 4D representations from partial observations, addressing the fundamental challenge of inferring unobserved geometry and motion.

Canonical Representations: CASPR [20] learns canonical spatio-temporal representations that factor out motion from shape. By learning a canonical space and deformation fields, CASPR can interpolate between observations and complete missing data. This factorization enables reconstruction even when only sparse temporal samples are available. RempeDynamics [21] extends this concept to human motion, incorporating physics-based priors to ensure plausible completions.

Implicit Representations: OFlow [56] extends occupancy networks to dynamic scenes by learning time-conditioned implicit functions. This approach elegantly handles topology changes and provides continuous representations of evolving geometry, enabling completion of occluded regions. Recent neural radiance field extensions [57, 58, 92–94] have achieved impressive reconstruction results, completing unseen viewpoints and temporal gaps simultaneously.

Gaussian Splatting for Dynamics: The emergence of 3D Gaussian Splatting has revolutionized dynamic reconstruction. Methods like Deformable 3D Gaussians [59], 4DGS [60], and SC-GS [61] model motion through deformation fields or sparse control points, achieving real-time rendering with high fidelity. These explicit representations offer advantages in editability and enable efficient completion of missing frames.

Compositional Approaches: 4DComposition [34] learns to decompose dynamic scenes into constituent parts and their motions. This compositional structure enables intelligent completion—missing parts can be inferred from the learned part library, while motion patterns guide temporal completion.

2.6.2 4D Synthesis: Generating Novel Dynamic Content

Synthesis methods create entirely new dynamic point cloud content, often from limited input modalities like text, images, or single-view videos.

Score Distillation Methods: Extending DreamFusion's SDS loss to 4D, methods like DreamGaussian4D [62] and Consistent4D [63] synthesize dynamic Gaussians using pre-trained diffusion models. These approaches generate plausible dynamic content from text descriptions or static images. SC4D [81] introduces sparse control for more efficient synthesis, enabling generation of longer sequences.

Multi-view Generation: Approaches leveraging multi-view diffusion models

[95–97] generate per-frame multi-view images as intermediate representations. SV4D [64] and STAG4D [65] extend this to video, synthesizing consistent 4D content from single-view inputs. This two-stage approach (2D generation followed by 3D lifting) leverages powerful 2D priors for enhanced quality.

Hybrid Representations: DreamMesh4D [66] combines sparse control Gaussians with mesh-surface bindings for physically plausible motion synthesis. L4GM [67] learns generative models for 4D content, enabling synthesis of diverse dynamic scenes. Efficient4D [68] focuses on optimization efficiency, making synthesis practical for resource-constrained applications.

Motion Structure Awareness: Recent work emphasizes understanding motion structure for robust synthesis. Methods decomposing scenes into rigid parts or learning motion priors achieve better generalization across viewpoints. This structural understanding enables more controllable generation—users can specify motion patterns while the system synthesizes appropriate geometry.

2.6.3 The Reconstruction-Synthesis Continuum

Reconstruction and synthesis exist on a continuum rather than as distinct categories:

- **Completion** tasks (partial reconstruction) fill in missing data while preserving observed information.
- **Enhancement** tasks improve quality or resolution of existing dynamic point clouds.
- **Transfer** tasks apply motion from one object to another, blending reconstruction and synthesis.
- **Generation** tasks (pure synthesis) create entirely novel content from mini-

mal constraints.

These generated dynamic point clouds then serve as inputs for the task-specific methods discussed in previous sections, creating a complete ecosystem for dynamic 3D understanding.

2.7 Datasets Driving Progress

The advancement of dynamic point cloud processing has been fundamentally enabled by the availability of diverse, high-quality datasets. These datasets not only provide training data but also establish benchmarks that drive algorithmic innovation. This section examines key datasets across different application domains.

Table 2.3: Representative Datasets for Dynamic Point Cloud Processing

Dataset	**Modality**	**Annotation**	**Primary Task**
Action/Gesture Recognition			
MSRAction-3D [98]	Depth	Action labels	Action recognition
NvGesture [99]	RGB-D	Gesture labels	Gesture recognition
SHREC'17 [100]	Depth	Hand bbox	Hand gesture
NTU RGB+D [101]	RGB-D	Action labels	Action understanding
NTU RGB+D 120 [102]	RGB-D	Action labels	Large-scale action understanding
Autonomous Driving			
KITTI [103]	LiDAR	3D boxes	Detection/tracking
nuScenes [5]	LiDAR+RGB	3D boxes	Detection/tracking
Waymo Open [46]	LiDAR	3D boxes	Detection/tracking
SemanticKITTI [10]	LiDAR	Point labels	Segmentation
KITTI-SF [104]	LiDAR	Scene flow	Flow estimation
Indoor Scenes			
ScanNet [9]	RGB-D	Instance seg.	Scene understanding
S3DIS [9]	RGB-D	Point labels	Segmentation
Synthetic/Specialized			
ShapeNet [3]	Mesh	Part labels	Shape analysis
PartNet [4]	Mesh	Hierarchical parts	Part segmentation
SAPIEN [105]	Synthetic 3D modalities	Part motion	Articulated objects tasks
FlyingThings3D [106]	Synthetic 3D modalities	Scene flow	Flow estimation

2.7.1 Dataset Characteristics and Challenges

Table 2.3 summarizes representative datasets across different domains. Several observations emerge:

Scale and Diversity: Dataset sizes vary dramatically, from hundreds of sequences (KITTI-SF) to millions of frames (ScanNet). This variation reflects different trade-offs between annotation quality and quantity. Large-scale datasets enable deep learning but often have coarser annotations, while smaller datasets provide richer labels but may limit model generalization.

Annotation Complexity: The annotation burden increases significantly from frame-level labels (action recognition) to point-level labels (segmentation) to dense correspondences (scene flow). This hierarchy influences method development—tasks with expensive annotations often motivate unsupervised or weakly-supervised approaches.

Sensor Modalities: Different sensors provide complementary information: RGB cameras offer appearance, depth sensors provide geometry, and LiDAR captures precise 3D structure. Multi-modal datasets enable fusion approaches but introduce calibration and synchronization challenges.

Real vs. Synthetic: Synthetic datasets (FlyingThings3D, SAPIEN) offer perfect ground truth and controllable conditions but may not capture real-world complexity. Real datasets provide ecological validity but suffer from annotation noise and limited diversity. The sim-to-real gap remains a significant challenge.

2.7.2 Emerging Dataset Needs

Despite the richness of existing datasets, several gaps persist:

- **Long-term Sequences:** Most datasets contain short clips (seconds to minutes). Understanding long-term dynamics requires datasets spanning hours or days.
- **Dense Temporal Sampling:** Higher frame rates would enable better mo-

tion analysis but increase storage and annotation costs.

- **Cross-domain Datasets:** Datasets spanning multiple environments (indoor/outdoor, aerial/ground) would improve generalization.
- **Dynamic Annotations:** Beyond static labels, annotations capturing motion patterns, deformations, and interactions would enable richer understanding.
- **Multi-scale Datasets:** Combining macro-scale scene dynamics with micro-scale object deformations requires multi-resolution capture.

2.8 Emerging Trends: Foundation Models, World Models, and Large-scale 3D Learning

Since the development of the methods presented in this thesis, the broader landscape of 3D understanding has undergone rapid evolution. Several emerging trends are reshaping the field and provide important context for situating our contributions.

2.8.1 3D Foundation Models and Large-scale Pre-training

The success of foundation models in natural language processing and 2D vision has inspired analogous efforts in 3D understanding. Point-BERT [107] and Point-MAE [108] introduced masked autoencoding for point cloud pre-training, demonstrating that self-supervised pre-training on large 3D datasets can yield transferable representations. More recent models such as Uni3D [109] and Point-GPT [110] have scaled this paradigm further, leveraging cross-modal alignment with language and image encoders to build unified 3D representations. Recent comprehensive surveys [111] highlight an explicit convergence of large-scale self-

supervised learning, geometric deep learning, and multimodal integration, culminating in architectures that bridge unstructured 3D data with large language models. OpenScene [112] and LERF [113] connect 3D scenes with open-vocabulary language features, enabling zero-shot 3D understanding.

Despite these rapid advancements, foundation models primarily target static 3D understanding. Extending them to dynamic point cloud sequences remains an open challenge: the temporal dimension introduces combinatorial complexity in pre-training objectives, and large-scale dynamic 3D datasets with sufficient diversity are still scarce. Our contributions are complementary to this trend: Kinet (Chapter 3) demonstrates that lightweight temporal extensions can efficiently adapt static backbones—potentially including pre-trained foundation models—to dynamic tasks without the computational overhead of training 4D models from scratch, while the SE(3)-equivariant framework (Chapter 4) shows that geometric priors can achieve strong generalisation even without large-scale pre-training.

2.8.2 World Models for 3D Perception and Generation

World models [114] that learn predictive representations of environment dynamics have gained significant traction, transitioning from simple 2D predictive models to rich representations of 3D and 4D environments [115]. In the context of autonomous driving and robotics, methods such as GAIA-1 [116], OccWorld [117], and DriveDreamer [118] learn to predict future 3D occupancy or sensor observations, effectively modelling the dynamics of the world through Video-based (VideoGen), Occupancy-based (OccGen), and LiDAR-based (LiDARGen) generation. Furthermore, breakthrough systems like Genie [119] illustrate the shift toward interactive foundation world models capable of simulating complex, playable 3D environments from minimal prompts.

These approaches share a conceptual connection with our work: both aim to capture temporal evolution in 3D environments. However, world models typically operate at a holistic, macro-level with implicit representations and require staggering computational resources. In contrast, our methods provide fine-grained, explicit, and interpretable representations at different granularities (semantic labels, rigid-part masks, or 4D Gaussian fields), offering precise geometric controls that macro-level world models often lack.

2.8.3 Data-centric Approaches and Large-scale Training

The availability of large-scale 3D datasets has been a key driver of progress, shifting the community toward data-centric 3D learning. Repositories like Objaverse [120] and its successor Objaverse-XL [121] provide millions of 3D objects, enabling data-driven approaches at unprecedented scale. More recently, the field has increasingly relied on Generative AI and synthetic data pipelines to automatically generate high-quality annotated 3D data, directly mitigating the persistent data scarcity challenge in 3D learning. End-to-end 3D learning frameworks are also emerging to process unbounded datasets for real-time downstream applications.

For dynamic temporal scenes, however, high-quality human-annotated datasets remain challenging and expensive to acquire, which strongly motivates the self-supervised and data-efficient approaches developed in this thesis. By effectively extracting motion and structure without relying on massive exhaustive annotations, our methods align with the goals of data-centric AI by maximizing the utility of available data. Furthermore, our ARTEMIS-DFA benchmark (Chapter 5) contributes to addressing the data gap for rigorous 4D generation evaluation.

2.8.4 Geometry in Generative Systems

Recent advances in 3D and 4D generative models have blurred the boundary between reconstruction and generation. Methods leveraging diffusion models for 3D generation [95, 96, 122] and their extensions to 4D [62, 64, 65] demonstrate that generative priors can compensate for limited observations. MotionStruct4D (Chapter 5) builds on this paradigm while introducing explicit motion structure awareness, which existing generative approaches lack. The integration of structured motion priors with powerful generative models represents a promising direction for future work.

2.8.5 Positioning and Justification of This Thesis

In light of these emerging trends and the rapid proliferation of large-scale vision and learning models, the contributions of this thesis are not only highly relevant but necessary to address the fundamental limitations of scaled-up "black-box" approaches. Our work is justified from four critical perspectives:

First, *computational efficiency and adaptability* continue to be paramount. Even as 3D foundation models grow exponentially larger, lightweight task-specific adaptations (as demonstrated by Kinet in Chapter 3) are essential for deploying these powerful representations to dynamic tasks in resource-constrained, real-time robotics platforms. Our approach allows existing frozen backbones to process temporal streams without incurring the prohibitive costs of full-time 4D retraining.

Second, against the backdrop of massive data scaling, *geometric and physical priors* provide indispensable inductive biases. Our SE(3)-equivariant framework (Chapter 4) achieves robust generalisation from limited data precisely because it explicitly encodes the structure of rigid motion. Unlike massive world models that must implicitly memorize physical laws from massive datasets, our methods

guarantee geometric consistencies by design.

Third, *self-supervised learning paradigms* remain critical. While synthetic 3D generation capability is expanding, acquiring varied, reliable, and perfectly aligned temporal 4D annotations natively is unscalable. The unsupervised nature of the techniques developed in Chapter 4 demonstrates how structured formulations can overcome data scarcity in dynamic 3D contexts autonomously.

Finally, *explicit interpretability and structural control* distinguish our work from implicit generative or world models. While massive world models can macroscopically simulate entire environments, they often lack fine-grained, physically-grounded control over internal components. The explicit motion structure models presented in this thesis (Chapter 5) provide the interpretability, disentanglement, and precise controllability potentially required for reliable downstream applications in autonomous driving and robotic manipulation.

2.9 Summary and Relation to Research Questions

This comprehensive review reveals both the substantial progress and significant gaps in dynamic point cloud processing. The field has evolved from simple extensions of static methods to sophisticated approaches that explicitly model temporal relationships, motion patterns, and dynamic scene structure. Recent advances in representation learning, geometric deep learning, and generative modeling have opened new possibilities for understanding and synthesizing dynamic 3D content.

Through our systematic analysis, several critical observations emerge:

Representation Trade-offs: The choice of 3D representation fundamentally constrains algorithmic possibilities. While point clouds offer direct sensor correspondence and flexibility, they require specialized architectures. The emergence of

hybrid representations suggests that future methods may adaptively combine multiple representations based on task requirements.

Task Hierarchy: The organization from video-level to point-level tasks reveals increasing computational complexity but also richer information extraction. While methods operating at different levels could in principle be combined, in practice they address distinct problems with different input requirements and evaluation criteria.

The Generation-Processing Cycle: Reconstruction and synthesis methods create dynamic point clouds that feed into task-specific processing, which in turn can guide better generation. This cyclical relationship suggests opportunities for joint optimization across generation and understanding tasks.

Data-Driven Progress: The availability of large-scale datasets has been crucial for advancing the field. However, the annotation burden for dynamic data remains a bottleneck, motivating continued research in unsupervised and self-supervised methods.

Key insights from this review directly motivate the research questions addressed in this thesis:

High-level Representation (Chapter 3): While methods like MeteorNet [14], P4Transformer [16], and recent transformer variants [42] advance dynamic classification, they often require training from scratch or significant architectural modifications. The computational overhead (often 10-100× that of static models) and lack of pre-trained foundation models limit practical deployment. The success of 2D video analysis techniques [123–126] in leveraging pre-trained models suggests similar opportunities for point clouds. This motivates our development of Kinet [33], which enables practitioners to extend existing static models for dynamic tasks with minimal modifications, addressing the practical need for efficient adap-

tation of pre-trained models.

Intermediate-level Representation (Chapter 4): Current motion segmentation approaches range from supervised methods like DeepPart [51] requiring extensive part-level annotations to complex optimization procedures in MultiBodySync [39]. Recent unsupervised methods [40] show promise but still require careful tuning and struggle with generalization. The emergence of SE(3)-equivariant networks [127, 128] and their recent application to 3D tasks [129, 130] suggests opportunities for more generalizable approaches. Our SE(3)-equivariant framework addresses this by enabling unsupervised learning of rigid segmentation and motion estimation, dramatically reducing annotation requirements while maintaining competitive performance across diverse object categories [131].

Low-level Representation (Chapter 5): The rapid progress in 4D generation, from early implicit methods [20, 56] to recent Gaussian-based approaches [62, 66, 81], demonstrates the potential for creating dynamic 3D content. However, existing methods typically assume multi-view observations or specific capture setups, limiting their practical applicability. The challenge of generating complete 4D representations from minimal inputs—particularly single monocular videos—remains largely unsolved. MotionStruct4D addresses this challenging scenario by discovering motion structures that enable robust 4D generation from minimal inputs, pushing the boundaries of what can be reconstructed from limited observations.

The progression from high-level to low-level representations reflects not only increasing granularity of analysis but also the evolution of the field itself. As fundamental capabilities mature, research attention shifts toward more nuanced understanding and generation of dynamic 3D content. This thesis contributes to three distinct aspects of this broad landscape, providing both theoretical insights and practical solutions for dynamic point cloud processing. By addressing the specific

challenges at each level—low-cost deployment at the high level, annotation-free generalization at the mid level, and data efficiency at the low level—while maintaining high quality throughout, our work advances the overarching goal of enabling machines to understand and interact with dynamic 3D environments. These three contributions are largely independent in their technical design and should be understood as addressing complementary facets of a multifaceted problem rather than as tightly coupled layers of a single system.

3 | High-level Representation Learning: Extending Static Models for Dynamic Point Cloud Classification

Having established the theoretical and methodological background in Chapter 2, the following three chapters present our contributions at progressively finer levels of abstraction. This chapter addresses the highest level—semantic classification of dynamic point cloud sequences—where the primary challenge is deploying temporal models at low computational cost. As the literature review highlighted, existing dynamic classification methods typically require training from scratch or significant architectural modifications, incurring computational overheads that are much larger than those of their static counterparts. We propose Kinet, a kinematics-inspired framework that seamlessly extends mature static backbones to dynamic recognition, with only minor structural changes and low computing overhead.

3.1 Abstract

Scene flow is a powerful tool for capturing the motion field of 3D point clouds. However, it is difficult to directly apply flow-based models to dynamic point cloud classification since the unstructured points make it hard or even impossible to efficiently and effectively trace point-wise correspondences. To capture 3D motions without explicitly tracking correspondences, we propose a **kine**matics-inspired neural **net**work (Kinet) by generalizing the kinematic concept of ST-surfaces to the feature space. By unrolling the normal solver of ST-surfaces in the feature space, Kinet implicitly encodes feature-level dynamics and benefits from the use

of mature backbones for static point cloud processing. With only minor changes in network structures and low computing overhead, it is straightforward to jointly train and deploy our framework with a given static model. Experiments on NvGesture, SHREC'17, MSRAction-3D, and NTU-RGBD demonstrate its efficiency in terms of parameters and computational complexity, as well as versatility to various static backbones. The code is available at https://github.com/jx-zhong-for-academic-purpose/Kinet.

3.2 Introduction

Due to continued miniaturization and mass production, 3D sensors are becoming less esoteric and increasingly prevalent in geometric perception tasks. These sensors typically represent scene geometry through a point cloud, which is an unordered and irregular data structure consisting of distinct spatial 3D coordinates. As a fundamental problem in point cloud understanding, classification of static scenes [9, 12, 29] or objects [1, 3, 132] has witnessed rapid advances over the past few years. Whilst impressive, these techniques do not directly account for the fact that the real 3D world is also changing, through egocentric and/or allocentric motion. To better understand our time-varying world, a handful of recent works [14–17, 19, 42, 82] have been applied to dynamic point cloud classification, a task in which the model is required to output a video-level category for a given sequence of 3D point clouds.

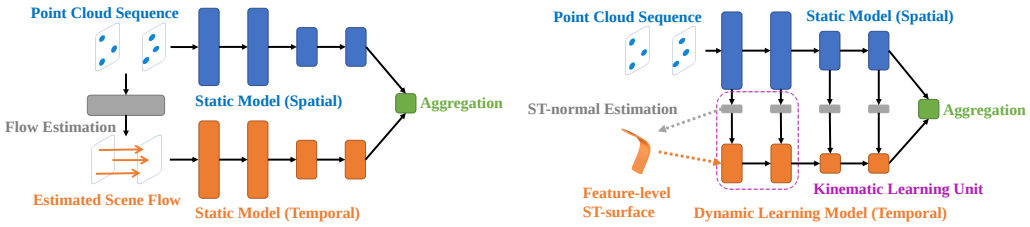
As a natural extension of 2D optical flow, 3D scene flow captures the motion field of point clouds. Based on optical flow, two-stream networks [133–137] have already proven to be successful in image-based video classification. Hence, it should be a natural choice to classify dynamic point clouds with the help of scene flow. However, to the best of our knowledge, scene flow has not been utilized in

point cloud sequences despite the prevalence of mature scene flow estimators [24–26, 35, 36, 53, 138, 139].

What then hampers us from applying scene flow to dynamic point cloud classification? Although scene flow is a powerful tool, it is difficult to estimate it efficiently and effectively from sequential point clouds - the computation of 3D scene flow has higher time expenditure, larger memory consumption, and lower accuracy than that of 2D optical flow. These challenges are mainly caused by the irregular and unordered nature of dynamic point clouds. This unstructured nature makes it difficult to track the point-wise correspondences of the moving point sets across different frames.

Why not extract dynamic information without explicitly finding the point-wise correspondences? If this were possible, it would allow us to draw benefits from inferring motion representations without incurring a high scene flow estimation cost. Similar to the advantages seen in two-stream networks in image-based models, we would be able to *preserve the benefit of mature static solutions* in inference and training, such as well-benchmarked network architectures, transferable pre-trained weights, and ready-to-use source code. At the same time, the computationally intensive scene flow estimation could be avoided, with only *minor network modification and low computational overhead*.

For this purpose, we get inspiration from kinematics and propose a neural network (**Kinet**) to bypass direct scene flow estimation by generalizing the kinematic concept of space-time surfaces [140] (ST-surfaces) from the physical domain of point clouds to the feature space. In this way, normal vectors w.r.t. these ST-surfaces (ST-normals) establish the representation field of dynamic information as shown in Figure 3.1b. Thus, *motions are implicitly represented by feature-level ST-surfaces without explicitly computing point-wise correspondences*. Inspired by



(a) Flow-based two-stream framework. (b) Kinematic-guided two-stream framework.

Figure 3.1: *Comparison between the flow-based framework and ours.* With neither explicit point-wise correspondence estimation nor the stand-alone temporal branch, our framework is lightweight and efficient.

iterative normal refinement [141], we unroll the solver for ST-normals and make it jointly trainable alongside the static model in an end-to-end manner. Inheriting intermediate features from static network layers, Kinet is lightweight in parameters and efficient in computational complexity, compared with the vanilla flow-based framework depicted in Figure 3.1a which requires extra scene flow estimation and the independent temporal branch.

How our approach differs at a glance? Figure 3.1 contrasts the conventional *flow-guided* two-stream design with our *kinematic-guided* alternative. In the flow-based pipeline (Fig. 3.1a), one must *first* estimate dense 3D scene flow between consecutive frames and *then* feed the result into a dedicated temporal branch—incurring substantial compute/memory and relying on point-wise correspondences that are hard to establish for unordered, irregular points. In stark contrast, our Kinet framework (Fig. 3.1b) removes both the explicit scene-flow estimator and the stand-alone temporal branch. Instead, it reuses intermediate features from a mature static backbone and *implicitly* encodes dynamics by fitting feature-level space–time (ST) surfaces and their normals. The temporal signal is produced by stacked kinematic learning units that unroll a differentiable ST-normal solver; the final prediction simply aggregates this temporal representation with the spatial stream. This design preserves the benefits of static models (architecture, pretraining, codebase),

avoids point correspondence tracking, and adds only minor parameter/FLOP overhead.

Experiments are conducted on four datasets (*NvGesture* [99], *SHREC'17* [100], *MSRAction-3D* [98] and *NTU-RGBD* [101]) for two tasks (gesture recognition and action classification) with three typical static backbones (MLP-based *PointNet++* [28], graph-based *DGCNN* [31] and convolution-based *SpiderCNN* [142]). Noticeably, 1) in gesture recognition, our framework outperforms humans for the first time with the accuracy of 89.1% on *NvGesture*; 2) in action classification, it achieves a new record of 93.27% on 24-frame *MSRAction-3D* with only $3.20M$ parameters and $10.35G$ FLOPs.

In summary, our main contribution is as follows:

- By introducing Kinet, we decouple temporal information from spatial features, thereby easily extending static backbones to dynamic recognition and entirely preserving the merits of these mature backbones.
- Without tracking point-wise correspondences, we encode point cloud dynamics by unrolling the ST-normal solver in the feature space. This method is jointly trainable alongside the static model, with minor structural changes and low computing overhead.
- Extensive experiments on various datasets, tasks and static backbones show its efficiency in terms of parameters and computational complexity, as well as versatility to various static backbones. The code is available at https://github.com/jx-zhong-for-academic-purpose/Kinet.

3.3 Related Work

Deep Learning on Static Point Clouds. Recently, deep learning on 3D point clouds has attracted increased attention [27], with substantial progress achieved in several fields including shape classification [1–4], object detection [5–8] and scene segmentation [9–13]. This can be mainly attributed to the availability of various high-quality datasets [9, 10, 143] and sophisticated neural architectures [1, 2, 28, 144]. From the perspective of scene representations, existing works can be roughly divided into 1) Voxel-based methods [69–73], 2) Projection-based methods [145, 146], 3) Point-based methods [1, 2, 28, 30, 31, 142], and 4) Hybrid methods [132, 147, 148]. Based on the well-developed static classification models, we attempt to apply them to dynamic point cloud recognition with minor structural surgery and low computational overhead.

Deep Learning on Dynamic Point Clouds. A handful of recent works have explored dynamic problems on point clouds, such as recognition [14–17, 42], detection [22, 45, 47], tracking [43, 44], prediction [20, 21, 23, 34, 56] and scene flow estimation [24–26, 35, 36, 53, 138, 139]. Existing works on sequence classification are based on convolutional [14, 14, 82], recurrent [17], self-attentional [16, 42], or multi-stream neural networks [41]. As a convolutional framework, MeteorNet [14] modeled point cloud dynamics via spatio-temporal neighbor aggregations [14]. Likewise, PSTNet [15] applied point spatio-temporal convolutions to capture information along the time dimension and the space domain. Derived from recurrent networks, PointLSTM [17] updated the hidden states with the combination of past and current features. Fan *et al*. [16] and Wei *et al*. [42] adopted self-attentional structures along with the popularity of video transformers [149]. By extracting the offline dynamic voxel, 3DV [41] encoded motions and appearances through multiple streams. Our Kinet shares the same idea of decoupling

spatial and temporal information as 3DV, but the presented framework requires neither the offline motion extraction nor the extra stand-alone temporal stream.

Flow-guided Classifiers for Image-based Videos. We obtain inspiration from the similar idea of encoding optical flow information into deep representations for image-based video classification [123–126, 150, 151]. By subtracting feature maps along the temporal axis, OFF [123], STM [125] and PAN [126] robustly imitated optical-flow calculations. Similarly, Piergiovanni & Ryoo [152] and Fan *et al*. [153] mimicked TV-L1 optical flow iterations [154] inside network layers. Heeseung *et al*. [151] introduced correlations to continuous feature maps. For the purpose of acceleration, temporal shift modules [124] or spatial shift filters [150] were utilized to model multi-frame interactions. The above methods rely on feature maps retaining spatial correspondences of regular pixels in images, whereas features of irregular point clouds usually cannot manifest point-wise correspondences across frames. Consequently, all of the aforementioned feature-level operations (subtraction, shift, correlation, *etc*.) are ineffective for point cloud models. To achieve the similar goal of encoding dynamics from static features, we propose a distinct approach from them.

3.4 Methodology

Denote an input point cloud sequence with T frames as $P = (P_1, P_2, ..., P_{T-1}, P_T)$. The t^{th} frame $P_t = \{p_i^{(t)} | i = 1, 2, ..., m_t - 1, m_t\}$ is a set of m_t points, in which the position of the i^{th} point $p_i^{(t)}$ is specified by its spatial coordinates $\mathbf{x}_i^{(t)} = (x_i^{(t)}, y_i^{(t)}, z_i^{(t)}) \in \mathbb{R}^3$. The goal of dynamic point cloud classification is to output the sequence-level category label y for a particular input P.

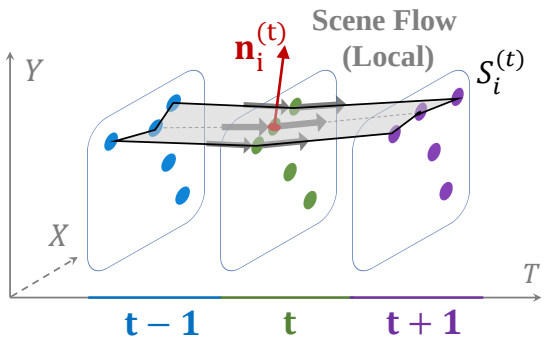


Figure 3.2: *2D local ST-surface and its normal.* Scene flow (darkgray arrows) lies on ST-surface $S_i^{(t)}$ and it is orthogonal to the normal $\mathbf{n}_i^{(t)}$. This demonstrates how motion can be implicitly encoded through surface normals.

3.4.1 Background: Kinematic ST-Surface

Our method extends the kinematic concept of ST-surfaces [140] and adopts the solver of iterative normal refinement [141] from the 3D physical world to deep representation learning.

Intuitively, the local **ST-surface** $S_i^{(t)} \subset \mathbb{R}^4$ centered at the point $p_i^{(t)}$ is *a surface that fits as many as possible space-time neighbor points* of $p_i^{(t)}$. Figure 3.2 illustrates ST-surface of 2D dynamic point clouds within 3 frames, which can be easily generalized to the 3D case and more frames. According to spatial kinematics [140], instantaneous velocity vectors always lie on the ST-surface. Equivalently, for a point cloud sequence as depicted in Figure 3.2, Mitra *et al.* [141] point out that *local scene flow lies on the same ST-surface* due to the vicinal (*i.e.*, local neighborhood) consistency of movements. As a result, the ST-normal $\mathbf{n}_i^{(t)}$ is orthogonal to local scene flow[1] and the field of those normals describes the motions of sequential point clouds.

Mathematically, the space-time neighbors of $p_i^{(t)}$ are a point set: $N_{\Delta r}^{\Delta t}(p_i^{(t)}) = \{p_j^{(\tau)} |\ |t-\tau| \leq \Delta t, ||\mathbf{x}_i^{(t)} - \mathbf{x}_j^{(\tau)}|| \leq \Delta r\}$, hereinafter referred to as $N_i^{(t)}$ for

[1] Strictly speaking, the ST-normal $\mathbf{n}_i^{(t)}$ is orthogonal to the local tangent plane of scene flow.

simplicity. $S_i^{(t)}$ is specified by its tangent plane with the surface equation $A\mathbf{x}+b = t$, of which the coefficient $A \in \mathbb{R}^{1\times 3}$ and b satisfy:

$$A\mathbf{x}_j^{(\tau)} + b = \tau, \ \ \forall p_j^{(\tau)} \in N_i^{(t)} \,. \tag{3.1}$$

In practice, Equation (3.1) may be an over-determined linear system since the local neighbor area may be too large to reflect the instantaneous velocity. In this case, space-time neighbors $N_i^{(t)}$ cannot be completely represented by coefficients of the tangent plane and there exists no exact solution. Therefore, a least-squared approximation is introduced to seek the optimal coefficients A^* and b^*:

$$A^*, b^* = \underset{A,b}{arg\,min} \sum_{p_j^{(\tau)} \in N_i^{(t)}} ||A\mathbf{x}_j^{(\tau)} + b - \tau||^2 \,. \tag{3.2}$$

To alleviate the influence of noisy point clouds, a commonly-used objective is to obtain the coefficients based on weighted neighbor points:

$$A_{i,t}^*, b_{i,t}^* = \underset{A,b}{arg\,min} \sum_{p_j^{(\tau)} \in N_i^{(t)}} w_j^{(\tau)} ||A\mathbf{x}_j^{(\tau)} + b - \tau||^2 \,, \tag{3.3}$$

where $w_j^{(\tau)}$ is the point-wise weight of these neighbors, and we use subscripts i, t to denote that these coefficients correspond to point i at time t.

As a solver of Equation (3.3), **iterative normal refinement** [141] robustly encodes dynamics via the normal field of ST-surfaces. This was earliest used in Dynamic Geometry Registration [141], a traditional method to register large-scale moving and deforming point clouds. As shown in Figure 3.3, the basic idea is to alternately re-compute the ST-surface $S_i^{(t)}$ and its normal $\mathbf{n}_i^{(t)}$ based on the neighbors' weights $w_j^{(\tau)}$ and re-weight the space-time neighbors $N_i^{(t)}$ based on

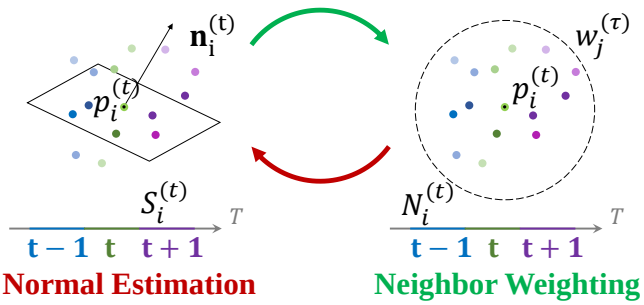


Figure 3.3: *Solver of iterative normal refinement.* The darkness of colors represents weights of neighbor points. The ST-surface is estimated upon the point-wise weights, and vice versa. This iterative process refines both surface estimation and neighbor weighting.

the estimated ST-surface $S_i^{(t)}$ until convergence.

3.4.2 Kinematic Representation Learning

3.4.2.1 Framework

The vanilla flow-based framework (Figure 3.1a) explicitly extracts scene flow (or dynamic voxels as in the case of 3DV [41]), while Kinet implicitly encodes motions with feature-level ST-surfaces. As shown in Figure 3.1b, Kinet contains three parts: 1) a spatial branch (marked in blue) identical to common static models, 2) a temporal model comprised of stacked kinematic learning units (the violet dotted frame), followed by 3) the final aggregation (marked in green) of spatial and temporal results.

3.4.2.2 Kinematic Learning Unit

Typically, a learning unit for point clouds has two crucial operations, *i.e.*, **grouping** and **abstracting**. The former selects the neighbors around centroids (*e.g.* ball query in PointNet++), while the latter encodes the local feature from these neighbors (*e.g.* PointNet layers in PointNet++). Figure 3.3 demonstrates that iterative

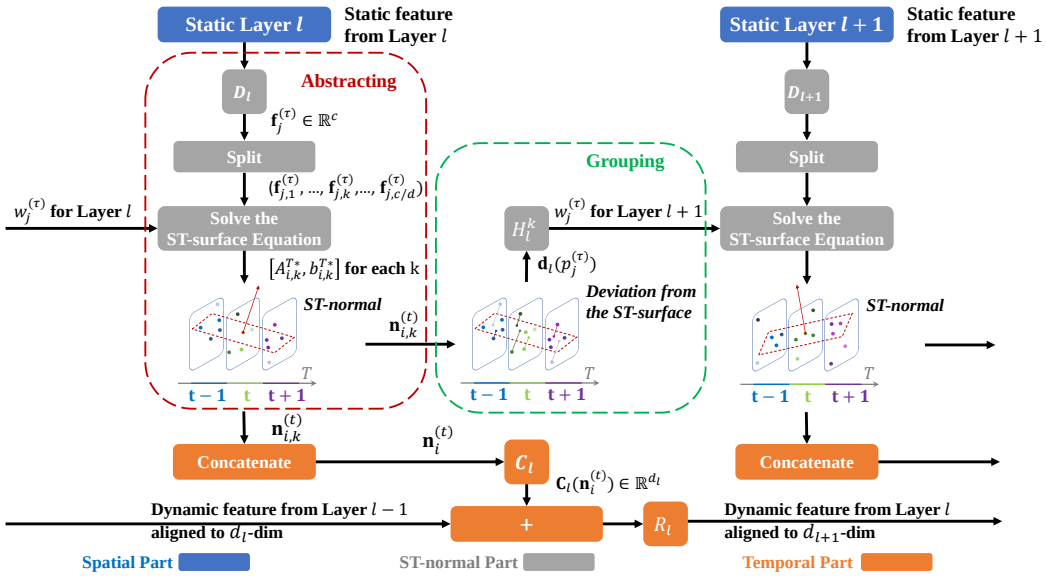


Figure 3.4: *Kinematic learning unit.* As shown in the violet dotted frame of Figure 3.1b, a stack of these units comprises the temporal branch of our framework. The alternate abstracting (red dotted frame) and grouping (green dotted frame) unroll the ST-normal solver (Figure 3.3) in the feature space. Only the **squares** are parametric operations whereas the **rectangles** introduce no parameters - kinematic learning just requires a small number of learnable parameters.

normal refinement alternates between two similar operations: **neighbor weighting** corresponding to grouping and **normal estimation** analogous to abstracting. The original integrative normal refinement works in the non-differentiable physical space of three-dimensional point sets - we unroll its solver in a fully *differentiable* fashion and generalize it to the *high-dimensional* feature space for joint optimization in neural networks as depicted in Figure 3.4.

Assume that we obtain a series of features $F_l(P) = F_l(P_1), F_l(P_2), ..., F_l(P_{T-1}), F_l(P_T)$ from the l^{th} layer (marked in blue in Figure 3.4) of a certain static model F. Based on sequential static features $F_l(P)$, our learning unit aims to obtain dynamic representations by fitting feature-level ST-surfaces.

Abstracting with Normal Estimation (Red Dotted Frame in Figure 3.4) To decrease the computational complexity, we first utilize a 1×1 convolution D_l to reduce its dimension to c, where c is proportional to the dimension of a static feature. For a given point $p_i^{(t)}$, the corresponding $c-$dimensional feature vector is denoted as $\mathbf{f}_i^{(t)} \in \mathbb{R}^c$. Similar to the physical space, the tangent hyper-plane of ST-surfaces in the feature space is specified by its surface equation $A\mathbf{f} + b = t$, where the coefficients A and b satisfy:

$$A\mathbf{f}_j^{(\tau)} + b = \tau, \ \ \forall p_j^{(\tau)} \in N_i^{(t)} . \tag{3.4}$$

Likewise, *time-varying changes of those static features lie on the corresponding ST-(hyper)surface in the representation space*. The vector field of normals w.r.t. such feature-level ST-surfaces orthogonally describes the dynamic information based on static representations.

The equation of ST-surfaces in the 3D physical space is usually over-determined ($|N_i^{(t)}| > 3$), whereas it is not the case in the $c-$dimensional feature space ($|N_i^{(t)}| <$

c). To ensure exact coefficient solutions in Equation 3.4, we split the point-wise feature $\mathbf{f}_j^{(\tau)}$ into several $d-$dimensional groups $\mathbf{f}_j^{(\tau)} = (\mathbf{f}_{j,1}^{(\tau)}, ..., \mathbf{f}_{j,k}^{(\tau)}, ..., \mathbf{f}_{j,c/d}^{(\tau)})$ where $\mathbf{f}_{j,k}^{(\tau)} \in \mathbb{R}^d$ specifies the k^{th} group. For each group $\mathbf{f}_{j,k}^{(\tau)}$, the number of neighbors $N_i^{(t)}$ is sufficiently large to solve the following weighted least-squared approximation:

$$A_{i,k,t}^*, b_{i,k,t}^* = \underset{A,b}{arg\,min} \sum_{p_j^{(\tau)} \in N_i^{(t)}} w_{j,k}^{(\tau)} ||A\mathbf{f}_{j,k}^{(\tau)} + b - \tau||^2 \,, \tag{3.5}$$

where $w_{j,k}^{(\tau)}$ is the point-wise weight of each neighbor, and we denote the coefficients with subscripts i, k, t to indicate they correspond to point i, group k, at time t. The vanilla iterative normal refinement leverages weighted-PCA [155] to solve the normals, which is unfriendly to back-propagation [156]. To this end, we attempt to directly fit this equation via its closed-form least-squared solution:

$$[A_{i,k,t}^{T*}, b_{i,k,t}^*] = (F_{i,k}^{(t)T} W_{i,k}^{(t)} F_{i,k}^{(t)})^{-1} F_{i,k}^{(t)T} W_{i,k}^{(t)} \boldsymbol{\tau}_{i,k}^{(t)} \,, \tag{3.6}$$

where the weight matrix $W_{i,k}^{(t)} = diag(w_{1,k}^{(\tau)}, ..., w_{|N_i^{(t)}|,k}^{(\tau)}) \in \mathbb{R}^{|N_i^{(t)}| \times |N_i^{(t)}|}$, the feature matrix $F_{i,k}^{(t)} = [(\mathbf{f}_{1,k}, 1), .., (\mathbf{f}_{|N_i^{(t)}|,k}, 1)] \in \mathbb{R}^{|N_i^{(t)}| \times (d+1)}$ and the time vector $\boldsymbol{\tau}_{i,k}^{(t)} \in \mathbb{R}^{|N_i^{(t)}|}$. The normal vector is as follows:

$$\mathbf{n}_{i,k}^{(t)} = \frac{(A_{i,k,t}^{T*}, -1)}{||(A_{i,k,t}^{T*}, -1)||} \,, \tag{3.7}$$

where $||\cdot||$ is the $\ell 2$-norm. The concatenated normals $\mathbf{n}_i^{(t)} = concat(\mathbf{n}_{i,1}^{(t)}, .., \mathbf{n}_{j,\frac{c}{d}}^{(t)})$ are fed into a 1×1 convolution C_l to obtain the d_l-dimensional abstracted dynamic feature $C_l(\mathbf{n}_i^{(t)})$. After the dimension alignment with another 1×1 convolution R_l, the feature is forwarded to the next layer via a residual connection.

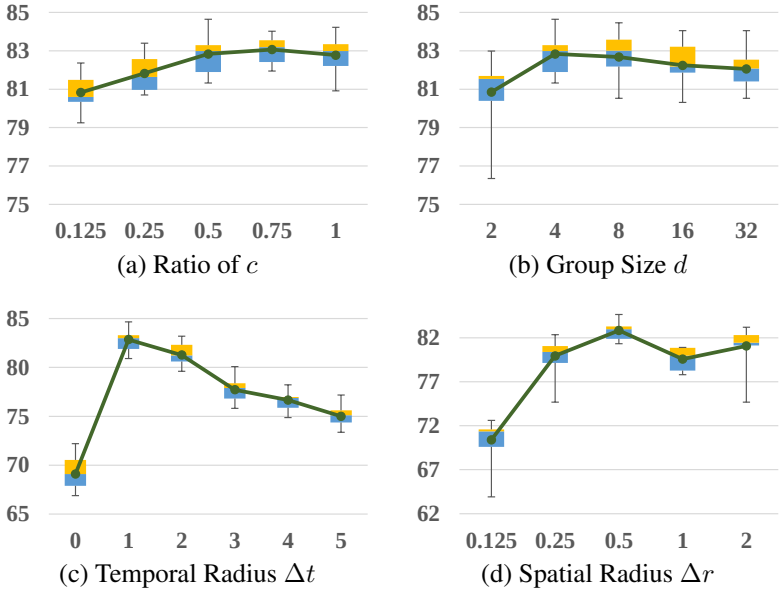


Figure 3.5: *Box-whisker plots of temporal-stream performance on validation set of* NvGesture *under different hyper-settings.* The x-axis is the value of hyper-parameters, while y-axis is the validation accuracy (%). The boxes show quartiles with median lines, and whiskers extend to extreme values.

Grouping with Weighted Neighbors (Green Dotted Frame in Figure 3.4) For each neighbor $p_j^{(\tau)}$, we compute its weight $w_{j,k}^{(\tau)}$ for better point-wise representations in the l^{th} layer. Inspired by iterative normal refinement, we feed the channel-wise fitting deviation (a.k.a. residual) from the $(l-1)^{th}$ layer into a 1×1 convolution H_{l-1}^k activated by $sigmoid$ to obtain the neighbor weights within $[0, 1]$:

$$w_{j,k}^{(t)} = H_{l-1}^k(\mathbf{d}_{l-1}(p_j^{(\tau)}))\,, \tag{3.8}$$

where $\mathbf{d}_{l-1}(p_j^{(\tau)})$ is the vector of fitting deviations in the $(l-1)^{th}$ layer. In the first layer, $w_{j,k}^{(t)}$ is set to 1 for all of the neighbors.

3.4.2.3 Aggregation

In the last layer, we leverage the dynamic features to obtain $softmax$ classification scores of a point cloud sequence. The spatial and temporal stream is respectively optimized through a cross-entropy loss in the training stage. During the testing phase, category predictions from the static model and the temporal stream are averaged as the final outputs. Since all the operations of kinematic representation learning are differentiable, it can be seamlessly plugged into a wide range of static neural architectures with minor structural surgery.

3.5 Experiments

To evaluate the performance of Kinet, we conduct experiments on three datasets (NvGesture [99], SHREC'17 [100], MSRAction-3D [98] and NTU-RGBD [101]) for gesture recognition or action classification. The proposed framework is implemented with TensorFlow [157]. All experiments are conducted on the NVIDIA DGX-1 stations with Tesla V100 GPUs. In most experiments, PointNet++ [28] is adopted as the static backbone in Kinet. For a fair comparison, we follow the identical settings of network layers to [82]. The other backbones are mainly chosen to evaluate Kinet's versatility. If not specified explicitly, we keep hyper-parameters of the original backbone (see A.1 for more details). For training stability, we first train the static backbone (spatial stream) until convergence and then freeze its weights to individually optimize the dynamic branch (temporal stream). Following [14, 15, 17], classification accuracy is the main evaluation metric and we use default data splits for fair comparisons.

3.5.1 NvGesture: Hyper-settings & Ablations

NvGesture consists of 1532 (1050 for training and 482 for testing) videos composed of 25 classes. Following [17], we uniformly sample 32 frames from a video and generate 512 points for each frame. To provide intuition behind the operation of our framework, we first investigate the impact of various network settings and components before we turn to further studies.

Influence of Hyper-settings. We explore the effectiveness of crucial hyper-parameters with the 10-fold validation protocol [82]. To exclude the interference of the static branch, we calculate the accuracy only using the classification results of the dynamic stream without ensembles. Box-whisker plots are utilized to analyze the performance and the stability of hyper-settings as shown in Figure 3.5.

Feature dimensions c controls the output size of convolution D_l to reduce computational overheads, as depicted in Figure 3.4, which is proportional to the dimension of static features. We vary the ratio from 12.5% to 100% to find a value that makes the representations as compact as possible but informative enough. As shown in Figure 3.5a, 50% is adequate to encode the dynamic information from a static layer. In comparison with the standalone flow-based branch, this saves at least half of the network parameters.

Group size d controls the dimensions in each group. Ideally, d is expected to be sufficiently large to describe motions. However, excessively large matrices cause inefficiency in the ST-normal solvers. By changing d from 2 to 32, we empirically find that $d = 4$ properly balances the accuracy and computational overheads as shown in Figure 3.5b.

Temporal radius Δt controls the receptive field of our dynamic branch along the temporal axis. It should be sufficiently small to retain details but large enough to

model long-term interactions. It is observed that multi-frame information ($\Delta t \leq 1$) indeed improves the performance over a single frame ($\Delta t = 0$). In the searching range of $[0, 5]$, the best temporal radius Δt is 1 as depicted in Figure 3.5c.

Spatial radius Δr controls the receptive field in the spatial dimension. Similar to Δt, Δr should have a moderate optimum. From the comparison of $\Delta r \in [0.125, 2.0]$ in Figure 3.5d, $\Delta r = 0.5$ consistently has decent performance.

In the remaining experiments, we set the ratio of feature reduction as 50%, group-wise dimensions $d = 4$, temporal radius $\Delta t = 1$, and spatial radius $\Delta r = 0.5$, respectively.

Settings	**Spatial (Static)**	**Temporal (Dynamic)**			**Accuracy**
No.	Pretrain	Pretrain	ST-normal	Weighted	**(%)**
i.	✗	–	–	–	82.6
ii.	✓	–	–	–	84.5
iii.	–	✗	✓	✓	80.9
iv.	–	✓	✓	✓	82.4
v.	✓	✓	✗	✗	85.3
vi.	✓	✓	✓	✗	87.9
vii.	✓	✓	✓	✓	89.1

Table 3.1: *Ablation studies on* NvGesture. ✓/✗ means that the operation is applied/not applied to the framework, while – means that the predictions of the corresponding spatial/temporal stream are excluded from the evaluation. The results of the upper/lower part is obtained from a single stream/two streams, respectively.

Ablation Studies. We conduct ablation studies on different components on the test set of *NvGesture*.

Is it beneficial to pretrain on static datasets? For two-stream models of image-based videos, it is well-known that pretraining on images significantly improves the performance on videos [133, 136]. However, this fact has not been verified for point cloud sequences. By pretraining the static PointNet++ on ModelNet40 [3], we analyze the individual performance change for each branch. Table 3.1 *i.* &

ii. demonstrate that the performance of spatial stream on multi-frame predictions increases by 1.9% (from 82.6% to 84.5%), while the temporal branch using the pretrained static features also boosts the accuracy by 1.5% as shown in Table 3.1 *iii.* & *iv.*. Obviously, both the static branch and the dynamic stream benefit from pretraining. Based on the pretrained backbone, the fusion of two-stream results achieves the accuracy of 89.1% since the two branches are complementary to each other.

Is it useful to abstract with normal estimation? To evaluate the efficacy of our abstracting operation, we remove the process of normal calculation in Figure 3.4. Instead, the reduced feature is directly fed into the convolution C_l and aggregated with max pooling. In this manner, all the trainable convolutions are unchanged but the accuracy significantly drops from 89.1% to 85.3% as shown in Table 3.1 *v.* & *vii.*. By comparing *ii.* with *v.* in Table 3.1, we find that purely introducing additional convolution parameters over the static backbone contributes only 0.8% to the performance gain. Evidently, normal estimation is a vital component in the abstracting operation.

Is it helpful to group with weighted neighbors? By replacing all the learnable weights of neighbor points with the fixed value of 1.0, we train the model to analyze the effect of weighted grouping. Table 3.1 *vi.* shows that the performance decreases to 87.9%, which means the grouping operation with weighted neighbors is capable of further improving the performance of our feature abstraction.

Comparisons. *NvGesture* is a multi-modality dataset and allows us to compare our method to state-of-the-art techniques under the standard data split [99]. As shown in Table 3.2, our approach with an accuracy of 89.1% not only outperforms all of the existing point cloud methods, but also achieves higher performance than those models using other modalities. It is worth mentioning that Kinet is even

Methods	Modalities	Accuracy (%)
R3DCNN [99]	Infrared Image	63.5
R3DCNN [99]	Optical Flow	77.8
R3DCNN [99]	Depth Map	80.3
PreRNN [158]	Depth Map	84.4
MTUT [159]	Depth Map	84.9
R3DCNN [99]	RGB Frame	74.1
PreRNN [158]	RGB Frame	76.5
MTUT [159]	RGB Frame	81.3
PointNet++ [28]	Point Cloud	63.9
FlickerNet [82]	Point Cloud	86.3
PointLSTM-base [17]	Point Cloud	85.9
PointLSTM-early [17]	Point Cloud	87.9
PointLSTM-PSS [17]	Point Cloud	87.3
PointLSTM-middle [17]	Point Cloud	86.9
PointLSTM-late [17]	Point Cloud	87.5
Human [99]	RGB Frame	88.4
Kinet	Point Cloud	**89.1**

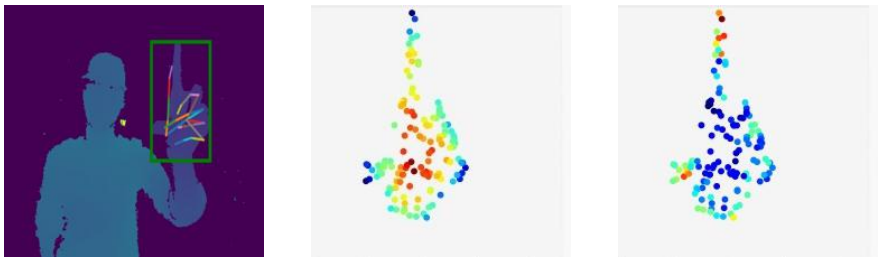
Table 3.2: *Quantatitve results achieved on* NvGesture.

superior to the human recognition on RGB videos (88.4%) *for the first time*.

3.5.2 SHREC'17: Robustness to Noisy Backgrounds

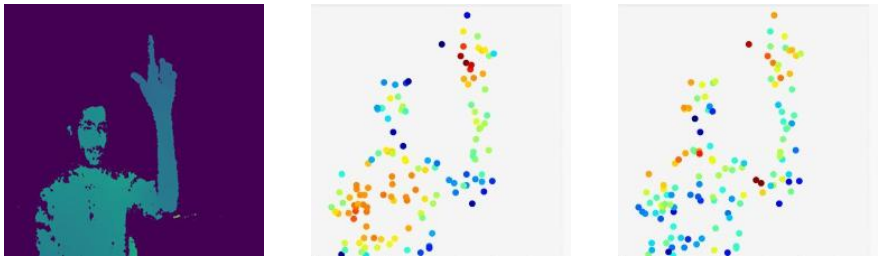
SHREC'17 is comprised of 2800 videos in 28 classes for gesture recognition, of which 70% (2960 videos) are training data and the other 30% (840 videos) are the test set. It has two types of supervisory signals, *i.e.*, video-level classification labels and bounding boxes (BBox) of hand skeletons. Unlike most of the prior works only focusing on the background-free cases, we adopt two input settings to verify whether Kinet can capture useful movements and ignore meaningless ones: 1) w/ BBox (Figure 3.6a) - used by a majority of existing methods for high accuracy, based on the area inside the bounding boxes of hand skeletons without background interference; 2) w/o BBox (Figure 3.6d) - raw videos with

With Bounding Box (w/ BBox)



(a) Depth (b) Spatial (c) Temporal

Without Bounding Box (w/o BBox)

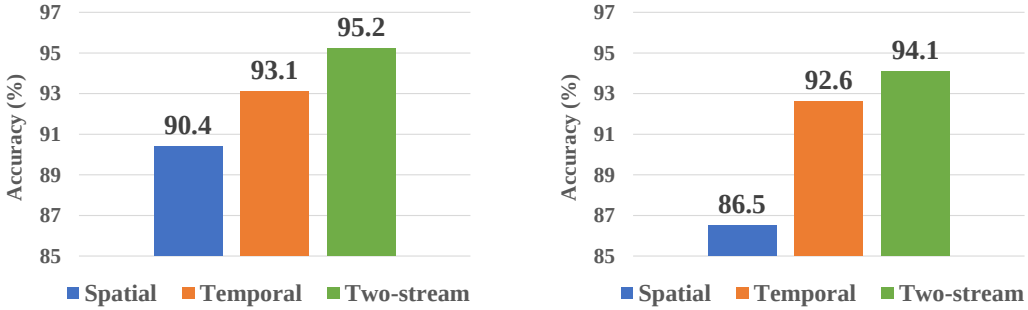


(d) Depth (e) Spatial (f) Temporal

Figure 3.6: *Raw depth inputs, and PACs on* SHREC'17. In PACs, the points in red have the highest activation values, while the blue ones are the lowest activating points. **Top section (w/ BBox):** When bounding boxes remove background noise, both spatial and temporal streams work complementarily - the spatial branch highlights palms while the temporal stream captures finger and wrist movements. **Bottom section (w/o BBox):** Without bounding boxes, the spatial stream focuses on the large background (performer's body) while the temporal stream successfully captures moving parts (arms and fingers) despite some noise.

noisy backgrounds (the performer's body).

Qualitative Analysis on Robustness. We visualize the learned point activation clouds (PACs) [15, 82] in Figure 3.6. With bounding boxes removing noisy backgrounds, the two streams work complementary - the static branch (Figure 3.6b) highlights the main parts (the palms) of spatial appearances, whereas the temporal representations (Figure 3.6c) capture key motions, such as the movement of fingers and wrists. In the case with redundant backgrounds (without bound-



(a) With Bounding Box (w/ BBox) (b) Without Bounding Box (w/o BBox)

Figure 3.7: *Stream-wise accuracy on* SHREC'17. The temporal stream shows stronger robustness to motion-irrelevant backgrounds (with bound boxes) at only a 0.5% accuracy drop compared to the spatial stream's 4% drop.

ing boxes), the static stream (Figure 3.6e) excessively focuses on the large yet useless background portions (the performer's body), while the temporal stream (Figure 3.6f) captures the moving parts (arms and fingers). Inevitably, the temporal stream also highlights several redundant points of the performer's shaking head by mistake.

Quantitative Analysis on Robustness. From Figure 3.7a & 3.7b, it is observed that the dynamic branch shows strong robustness to motion-irrelevant backgrounds, where the accuracy slightly drops from 93.1% to 92.6%, compared with that of the spatial stream which plunges by nearly 4%.

Comparison. As shown in Table 3.3, we compare the performance of Kinet to existing models. With the same bounding box supervision signal, our two-stream PointNet++ outperforms others with the accuracy of 95.2%. Noticeably, even for the highly challenging inputs without bounding boxes, our framework boosts the accuracy of static PointNet++ from only 86.5% to 94.1%, which is comparable to state-of-the-art models with bounding boxes.

Methods	Modalities	BBox	Accuracy (%)
Key frames [100]	Depth Map	✗	71.9
SoCJ+HoHD+HoWR [160]	Skeleton	✓	81.9
Res-TCN [161]	Skeleton	✓	87.3
STA-Res-TCN [161]	Skeleton	✓	90.7
ST-GCN [162]	Skeleton	✓	87.7
DG-STA [163]	Skeleton	✓	90.7
PointLSTM-base [17]	Point Cloud	✓	87.6
PointLSTM-early [17]	Point Cloud	✓	93.5
PointLSTM-PSS [17]	Point Cloud	✓	93.1
PointLSTM-middle [17]	Point Cloud	✓	94.7
PointLSTM-late [17]	Point Cloud	✓	93.5
Kinet	Point Cloud	✓	**95.2**
Kinet	Point Cloud	✗	94.1

Table 3.3: *Quantitative results achieved on* SHREC'17.

Methods	Modalities	# of Frames	Accuracy (%)
Vieira *et al.* [164]	Depth Map	20	78.20
Kläser *et al.* [165]	Depth Map	18	81.43
Actionlet [166]	GroundTruth Skeleton	Full	88.21
PointNet++ [28]	Point Cloud	1	61.61
MeteorNet [14]	Point Cloud	4 / 8 / 12 / 16 / 24	78.11 / 81.14 / 86.53 / 88.21 / 88.50
P4Transformer [16]	Point Cloud	4 / 8 / 12 / 16 / 24	80.13 / 83.17 / 87.54 / 89.56 / 90.94
PSTNet [15]	Point Cloud	4 / 8 / 12 / 16 / 24	**81.14** / 83.50 / 87.88 / 89.90 / 91.20
Kinet	Point Cloud	4 / 8 / 12 / 16 / 24	79.80 / **83.84** / **88.53** / **91.92** / **93.27**

Table 3.4: *Quantitative results achieved on* MSRAction-3D.

3.5.3 MSRAction-3D: Different Tasks & Backbones

MSRAction-3D has 567 videos of 20 action categories. Following Fan *et al.* [16], we adopt the standard data splitting protocol [14, 166] and report the average accuracy over 10 runs.

Versatility across Static Backbones. Under the taxonomy defined in [27], three types of backbones dominate the classification models of static point clouds: MLP-based, convolution-based, and graph-based methods. In this chapter, we

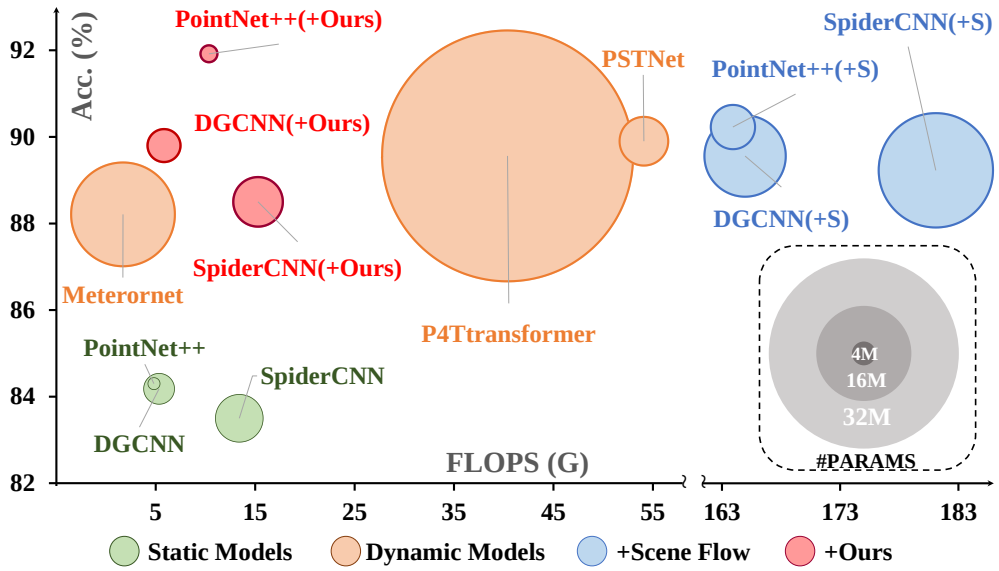


Figure 3.8: *Comparison of FLOPs, parameter number, and accuracy on 16-frame* MSRAction-3D. Quantitative details can be found in A.2.

choose one typical static architecture for each paradigm respectively, *i.e.*, MLP-based PointNet++ [28], convolution-based SpiderCNN [142] and graph-based DGCNN [31] in order to demonstrate the ease of extending these to dynamic point cloud tasks. By grouping a video into 16-frame clips as the input unit, we run 3 sets of experiments for each static model: 1) Directly feed videos into the static model; 2) Fuse the static spatial model and the temporal streams; 3) Ensemble classification scores from two static models, one is trained on the raw point clouds, while the other flow-based model is trained on scene flow estimated by Justgo [24], a self-supervised scene flow estimation tool. Apart from classification performance, we also take memory consumption and computational complexity into consideration. They are measured with the accuracy, the number of parameters, and floating-point operations per second (FLOPs), respectively.

As illustrated in Figure 3.8, the extra input modality of estimated scene flow in setting 3) (+*Scene Flow* ●) considerably improves the accuracy of the three static

models (●) to a level comparable to the state-of-the-art. However, the scene flow estimator and another flow-based classifier almost triple the number of parameters. Even worse, the estimation of scene flow introduces more than 150G FLOPs of extra calculations since point-wise dense predictions are required between every two consecutive frames. For setting 2) (+*Ours* ●), it is observed that our kinematic representations consistently increase the accuracy of the static predictions by 5.99%∼9.04% relative gains. By utilizing the kinematic representations, the FLOPs only increases to 5.83G∼15.29G, and the number of parameters increases by 0.59M∼1.08M. These computing overheads are negligible and make the fused model extremely lightweight. Compared with state-of-the-art dynamic networks (●), they achieve comparable or superior performance with fewer model parameters and lower computational complexity. Intriguingly, Kinet has higher performance gains in higher-performance static models, possibly indicating its *limitation* - the informativeness of kinematic representations is constrained by the static features. A poor static backbone would benefit from the proposed approach, but it cannot improve the underlying static representation.

Comparison. We compare the PointNet++-based Kinet with existing classification models on dynamic point clouds. Following prior research [14–16], we set the length as $\{4, 8, 12, 16, 24\}$ frames (2048 points/frame) and report the mean accuracy of 10 runs. As shown in Table 3.4, our framework shows the superiority in long videos (length≥ 8) and achieves the 24-frame accuracy of 93.27%, exceeding others by a large margin.

3.5.4 NTU-RGBD: Effectiveness on Large-scale Data

NTU-RGBD has two subsets, *i.e.*, NTU-RGBD 60 [101] and NTU-RGBD 120 [102]. We evaluate our model on the former since it is included in more comparison results of previous research. NTU-RGBD 60 is a large-scale dataset for 3D action

recognition, with 56880 RGBD videos of 60 action categories. The data is converted into point cloud sequences with the implementation of PSTNet [15].

Comparison Following the official data partition [101], cross-subject and cross-view scenarios are individually adopted as two splits. As shown in Table 3.5, Kinet is superior to the others, with an accuracy of 92.3% in the cross-subject split. In the cross-view protocol, Kinet comes a close second (with 96.4%) after 96.5% of PSTNet.

Table 3.5: *Quantitative results achieved on* NTU-RGBD 60.

Methods	**Modalities**	**Accuracy (%)**	
		Cross-subject	Cross-view
SkeleMotion [167]	Skeleton	69.6	80.1
GCA-LSTM [168]	Skeleton	74.4	82.8
Attention-LSTM [169]	Skeleton	77.1	85.1
AGC-LSTM [170]	Skeleton	89.2	95.0
AS-GCN [171]	Skeleton	86.8	94.2
VA-fusion [172]	Skeleton	89.4	95.0
2s-AGCN [173]	Skeleton	88.5	95.1
DGNN [174]	Skeleton	89.9	96.1
MS-G3D [175]	Skeleton	91.5	96.2
HON4D [176]	Depth Map	30.6	7.3
SNV [177]	Depth Map	31.8	13.6
HOG^2 [178]	Depth Map	32.2	22.3
Li *et al*. [179]	Depth Map	68.1	83.4
Wang *et al*. [180]	Depth Map	87.1	84.2
MVDI [181]	Depth Map	84.6	87.3
3DV-Appearance [41]	Point Cloud	80.1	85.1
3DV-Motion [41]	Voxel	84.5	95.4
3DV-Full [41]	Point Cloud + Voxel	88.8	96.3
P4Transfomers [16]	Point Cloud	90.2	96.4
PSTNet [15]	Point Cloud	90.5	**96.5**
Kinet	Point Cloud	**92.3**	96.4

3.6 Conclusion

In this chapter, we propose Kinet to bridge the gap between static point cloud models and dynamic sequences. By extending kinematic ST-surfaces to the high-dimensional feature space and unrolling the ST-normal solver differentiably, the presented framework gains advantages from mature static models. Without the pain of modeling point-wise correspondences, it can be seamlessly integrated into arbitrary static point cloud learning backbones, with only minor structural surgery and low computing overhead. Experiments on four datasets, two tasks, and three static networks demonstrate the efficacy of our framework in dynamic classification, the efficiency in parameters and FLOPs, and the versatility to various static backbones.

The reported efficiency metrics reflect Kinet's design as a lightweight temporal extension to existing static backbones for the specific task of sequence-level classification. In the broader multi-level framework of this thesis (Chapter 1), the efficiency advantage pertains to the high-level classification stage alone. Were the architecture generalised to incorporate additional capabilities—for instance, the intermediate-level rigid segmentation and equivariant motion estimation of Chapter 4, or dense geometric reconstruction—the computational requirements would increase substantially, as each stage introduces its own parameters and operations. A fully integrated pipeline spanning all representation levels would therefore carry a combined computational footprint far exceeding that of any individual component. The efficiency figures reported here should be understood in the context of this specific, task-focused design rather than as indicative of a general-purpose dynamic point cloud system.

Beyond architectural scope, another design dimension concerns input modality. A

deliberate choice in this work is the exclusive use of geometric (XYZ) point cloud data without incorporating RGB or other appearance information. This choice was motivated by two considerations: (1) our focus on the geometric dynamics captured by the ST-surface formulation, which operates on spatial coordinates and their temporal evolution; and (2) the desire to evaluate the contribution of pure geometric temporal modeling independently of appearance cues. We note that this practice is common in the point cloud sequence classification literature—PSTNet [15], P4Transformer [16], and PointLSTM [17] all operate on XYZ coordinates alone. Nevertheless, several datasets used in our evaluation—notably NTU-RGBD [101]—do provide colour channels alongside depth, and incorporating such multimodal inputs could improve accuracy, particularly for object-interaction actions (e.g., "reading" vs. "writing") where geometric motion alone may be ambiguous. Extending Kinet to handle multimodal point cloud attributes (*e.g.*, RGB, normals, intensity) is therefore a natural and promising direction for future work.

An obvious limitation is that Kinet is constrained by the performance of the static model—a poor static baseline will not be rectified by adding our dynamic extension. Therefore, we will explore a method to self-improve static representations in future works.

4 | Intermediate-level Motion Analysis: Unsupervised Rigid Segmentation and Motion Estimation

Whereas Chapter 3 addressed dynamic point cloud understanding at the semantic level, this chapter turns to a different and independently motivated problem: discovering rigid segments and estimating their 6-DoF motions in dynamic point clouds. The two chapters share the broad theme of dynamic point cloud representation learning but differ substantially in their technical approaches, input assumptions, and output granularity.

4.1 Abstract

A truly generalizable approach to rigid segmentation and motion estimation is fundamental to 3D understanding of articulated objects and moving scenes. In view of the closely intertwined relationship between segmentation and motion estimates, we present an SE(3) equivariant architecture and a training strategy to tackle this task in an unsupervised manner. Our architecture is composed of two interconnected, lightweight heads. These heads predict segmentation masks using point-level invariant features and estimate motion from SE(3) equivariant features, all without the need for category information. Our training strategy is unified and can be implemented online, which jointly optimizes the predicted segmentation and motion by leveraging the interrelationships among scene flow, segmentation mask, and rigid transformations. We conduct experiments on four datasets to demonstrate the superiority of our method. The results show that our method excels in both model performance and computational efficiency, with only 0.25M

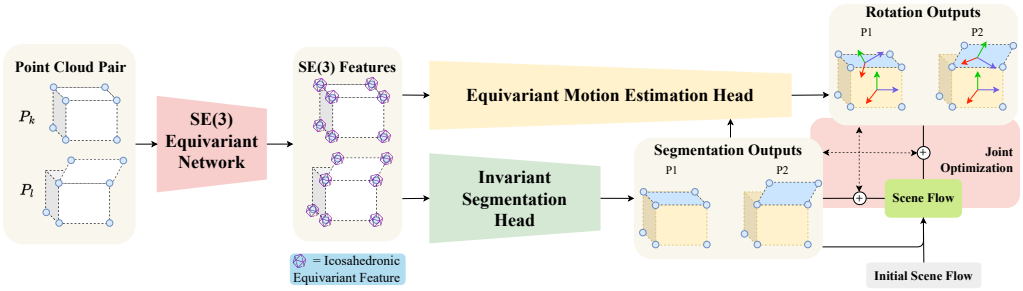


Figure 4.1: *An overview of network structure and training strategy.* We feed a pair of point clouds P_k, P_l into the SE(3) equivariant networks to obtain a pair of SE(3) icosahedronic features. These features are then fed into the two proposed heads for motion segmentation and invariant segmentation. The two outputs, combined with scene flow, are used for a joint optimization that can be done in an unsupervised manner.

parameters and 0.92G FLOPs. To the best of our knowledge, this is the first work designed for category-agnostic part-level SE(3) equivariance in dynamic point clouds.

4.2 Introduction

Comprehending point cloud motion is critical for various 3D vision tasks in dynamic scenarios, *e.g.*, tracking, animation, simulation, and manipulation. Many types of 3D motion can be described as a composition of multi-body rigid movements, such as articulated objects [105], and vehicular traffic scenes [182]. Specifically, the setting of *multi-body rigid motion* requires all moving parts (*i.e.*, bodies) to undergo only translation and rotation, without any type of deformation. The process of modeling these multi-body rigid movements typically involves two primary portions [39]: 1) the identification of distinct moving bodies (*i.e.*, *rigid segmentation*), and 2) the calculation of individual movements for each identified body (*i.e.*, *motion estimation*).

Being the first modeling portion, rigid segmentation significantly differs from tra-

ditional semantic segmentation tasks that concentrate on category information [9–13]. Based on multi-body motion instead of category-dependent semantics, rigid segmentation becomes *category-agnostic about moving parts* as rigid bodies are not limited to a specific set of shapes and naturally cannot be assigned certain category labels. As regards motion estimation, individual pose variations would occur arbitrarily, including transformations unseen in the training data. This constitutes an *open set of pose changes*.

Considering the relative inaccessibility and the constrained generalizability of manual annotations, Song and Yang [40] have proposed a seminal work for unsupervised rigid segmentation. Nevertheless, the unsupervised paradigm remains a formidable challenge because of the *generalizable requirement* in multi-body movements and the *interdependent nature* between rigid segmentation and motion estimation. Due to the aforementioned category-agnostic rigid prior and open-set pose changes in multi-body motion, a model is required to generalize well to transformations of appearance, location, and orientation. Moreover, rigid segmentation and multi-body motion estimation are highly coupled — the calculation of movements for each body is based on the output of a rigid partition, while an estimate of multi-body motion facilitates further segmentation. In absence of precise supervision, it is difficult to independently obtain effective training signals for either rigid segmentation or motion estimation.

To tackle the difficulties of generalizability and interdependence, this chapter presents a three-fold contribution in the form of architecture, training strategy, and experimental evaluation:

1. **Architecture.** *We design a part-level SE(3)-equivariant network that demonstrates strong generalization to open-set motion and category-agnostic moving bodies.* The SE(3)-equivariance allows features of a model to keep coherence with

the rigid transformations of point cloud inputs, thereby enhancing the model's robustness to such spatial transformations (*i.e.*, rigid motion) [127, 183–186] and therefore is more generalizable for open-set pose changes of global rigid targets [187–190]. Tailing the SE(3)-equivariance is two lightweight heads for segmentation and motion estimation. The former, unlike previous segmentation networks using global SE(3) equivariance, exhibits *point-level local flexibility* to SE(3)-invariance. The latter, robustly employing the probability of part consistency, utilizes *part-level "softly" matching operations* to handle SE(3)-equivariance without requiring knowledge of part categories. As a result of integrating *two lightweight heads* into a unified network, rather than utilizing large independent networks, our model is characterized by a small number of parameters (0.25 M) and low computational complexity (0.92G FLOPs). To the best of our knowledge, this is the first work designed for *category-agnostic part-level SE(3)-equivariance* in dynamic point clouds.

2. **Training Strategy.** *We leverage the interdependence between rigid segmentation and motion estimation and present a unified training strategy to jointly optimize their outputs in an online fashion.* Based on our proposed two-head network, we simultaneously filter out the noisy flow predictions and refine the estimates of rigid motion by exploiting the interrelation among scene flow, segmentation mask, and rigid transformation. Superior to previous arts, our online training process is *free from complicated components and manual intervention*, *e.g.*, the alternation of Markov Chain Monte Carlo proposals and Gibbs sampling updates [191], the offline repetition of training multiple segmentation networks from scratch [40], or the continual optimization of an accessory neural network of flow estimation [39].

3. **Experiments.** *Experiments on four datasets demonstrate the efficacy of our model in performance, as well as its efficiency in parameters and computational complexity.* We conduct comprehensive experiments on four datasets (SAPIEN [105],

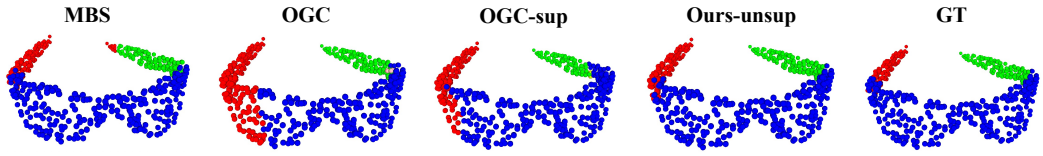


Figure 4.2: *Qualitative comparison of rigid segmentation on SAPIEN.* Selected examples comparing our **unsupervised** predictions with supervised baselines. Even without annotations, our method produces clean part boundaries that closely match the ground truth. An extended comparison covering a wider range of object categories is provided in Figure 4.6; additional visualisations appear in B.5.

OGC-DR [40], OGC-DRSV [40], and KITTI-SF [104]) across three application scenarios (articulated objects, furniture arrangements, and vehicular traffic). Noticeably, as shown in Figure 4.4, our performance on the SAPIEN dataset of articulated objects surpasses state-of-the-art results w.r.t. all evaluation metrics, achieving the relative gain of at least 14.7% AP in rigid segmentation and at least 23.3% in motion estimation, with only 0.25M parameters and 0.92G FLOPs. Figure 4.2 provides a qualitative preview: even without any annotations, our method discovers coherent rigid parts with sharp motion boundaries that closely match the ground truth, whereas existing methods—including supervised ones—often produce fragmented or noisy segmentation. The code is available at `https://github.com/jx-zhong-for-academic-purpose/Multibody_SE3`.

4.3 Related Works

Deep Learning on Dynamic Point Clouds. Deep neural networks for point cloud video modeling aim to understand our dynamic 3D surroundings. These networks enable the resolution of several downstream tasks, such as video-level classification [18, 32, 33, 87, 192, 192, 193], frame-level prediction [20, 34, 56], object-level detection [5–8] or tracking [43, 44, 46], part-level mobility pars-

ing [88, 194, 195], point-level segmentation [15, 32, 49, 192], and scene flow estimation [37, 38, 76, 196, 197]. In contrast to the above dynamic tasks that usually assume contiguous sequential input, multi-body rigid motion may be captured between discrete frames [39], presenting a unique challenge.

3D Motion Segmentation & Multi-body Rigid Motion. Although the systematic formulation of multi-body rigid motion modeling is a relatively recent development [39], a related problem, *i.e.*, motion segmentation, has been long sought after. Motion segmentation aims to group points with similar motion patterns from the input of scene flow, using such techniques as factorization [91, 198, 199], clustering[200], graph optimization [201–203], and deep learning [52, 191, 195, 204]. However, motion segmentation does not explicitly take into account multi-frame multi-body consistency. To address this issue, Huang *et al*. [39] present the seminal fully-supervised work for modeling multi-body rigid motion. Song and Yang [40] further attempt to predict the mask of rigid segmentation via three object geometry losses in an unsupervised setting. Following the same unsupervised paradigm, our approach has distinct motivations: 1) to achieve high generalizability in the presence of low-quality training signals, and 2) to simplify the training process through online optimization.

SE(3) Equivariant Networks for Point Clouds. Equivariant networks have superior discriminative and expressive ability for various data structures (*e.g.*, graphs [205, 206], images [207, 208]) under the transformation of some symmetry groups. Among them, the equivariance of a Special Euclidean group (3) (SE(3)) has drawn increasing attention to 3D point cloud processing [127, 183–190]. A vast majority of these researches focus on global, while the part-level local equivariance is under-explored. Recently, [128] has utilized part-level SE(3) equivariance for supervised bounding box detection. Concurrently with our work, Lei *et*

al. [129] and Feng *et al*. [130] apply part-level equivariance to category-specific tasks of object segmentation and human body parsing, respectively. Differently, we leverage part-level equivariance to the unsupervised category-agnostic problem.

4.4 Methodology

4.4.1 Background: SE(3)-equivariance/invariance & Discretization

Given a point cloud $X \in \mathbb{R}^{n\times 3}$ of n points and a rigid transformation $\mathbf{T} \in \mathrm{SE}(3) : \mathbb{R}^{n\times 3} \rightarrow \mathbb{R}^{n\times 3}$, a neural network mapping inputs to a feature domain $\phi : \mathbb{R}^{n\times 3} \rightarrow \mathcal{F}$ is defined as *SE(3)-equivariant* if $\phi(\mathbf{T} \circ X) = \mathbf{T} \circ \phi(X), \forall \mathbf{T} \in \mathrm{SE}(3)$, where $\circ$ means performing an SE(3) transformation. Likewise, *SE(3)-invariance* is expressed as $\phi(\mathbf{T} \circ X) = \phi(X)$. To reduce the computational cost of equivariant networks, [209] discretizes the rotation space into an icosahedral group $\mathcal{G}$ of 60 rotational angles ($|\mathcal{G}| = 60$). As a further extension, Equivariant Point Network (EPN) [127] operates the SE(3) discretization on point clouds. For a point $x \in \mathbb{R}^3$ in the input X, the feature extractor of EPN outputs the per-point representation $f(x) \in \mathbb{R}^{|\mathcal{G}|\times D}$, where D is the feature dimension. By definition, $f(x)$ is SE(3)-equivariant to its neighbors of the convolution kernel's receptive field:

1) Rotation equivariance within the icosahedral group: $f(g \circ x) = g \circ f(x), \forall g \in \mathcal{G}$.

2) Translation invariance w.r.t. arbitrary translation t: $f(t \circ x) = f(x)$.

By relaxing the strictly rotational equivariance into the equivariance w.r.t. the 60 icosahedral angles in $\mathcal{G}$, EPN is proven to be robust to many downstream tasks, such as 6D pose estimation [188], and place recognition [210]. Due to its high ro-

bustness, we choose EPN as our SE(3) equivariant backbone of feature extraction. In principle, our training strategy is versatile, and other networks with *per-point SE(3)-equivariant representations* can also serve as its feature extractor.

4.4.2 Problem Statement

We define a set of point clouds P from K frames that are not necessarily consecutive as $P = \{P_1, P_2, \cdots, P_{K-1}, P_K\}$, where each frame $P_k = \{p_k^i \in \mathbb{R}^3 | i = 1, 2, \cdots, N-1, N\}$ contains N three-dimensional points. All frames of P represent the same target,[1] which is segmented into S independently moving rigid parts. For a given frame P_k, the point-part rigid mask is defined as $\mathbf{M}_k \in \{0, 1\}^{N \times S}$. $\mathbf{M}_k^{is} = 1$ indicates that the point p_k^i belongs to the s^{th} rigid part; $\mathbf{M}_k^{is} = 0$ indicates that it does not. The rigid motion of the s^{th} part between two frames (P_k, P_l) is denoted as $\mathbf{T}_{kl}^s \in \mathrm{SE}(3)$. This rigid transformation is specified by a rotation matrix $\mathbf{R}_{kl}^s \in \mathrm{SO}(3)$ and a translation vector $\mathbf{t}_{kl}^s \in \mathbb{R}^3$. Note that the targets in the dataset belong to various categories, including classes possibly unseen in the training data.

Given the point cloud set P, we aim to segment each frame P_k with rigid-part masks $\mathbf{M}_k$, and obtain part-level rigid transformation $\mathbf{T}_{kl}^s$ between frames. The former task is called *rigid segmentation*, while the latter is named *motion estimation*. In the supervised setting, ground-truth segmentation masks and rigid motions (or scene flow) are provided in the training data. In the unsupervised formulation, neither segmentation nor motion information is available, and our training input solely consists of point cloud frames.

[1]The term "target" refers to objects, environments, or scenes associated with multi-body rigid motion, such as articulated objects, arrangements of indoor furniture, or scenes of vehicular traffic.

4.4.3 Network Architecture

As shown in Figure 4.1, our network takes a pair of point cloud frames as the input and outputs the predictions of rigid segmentation and motion estimation. There are three main components in the proposed framework, *i.e.*, a feature extractor, an invariant segmentation head, and an equivariant motion estimation head.

Per-point Feature Extractor. For an input frame P_k, the feature extractor of EPN outputs per-point SE(3)-equivariant representations $F_k \in \mathbb{R}^{N\times|\mathcal{G}|\times D}$, following the notations in Section 4.4.1 and 4.4.2. The corresponding feature $f_k^i \in \mathbb{R}^{|\mathcal{G}|\times D}$ of a point p_k^i can be viewed as a concatenation of different representations w.r.t. $g_j \in \mathcal{G}$ over the rotation group dimension:

$$f_k^i = [\theta(p_k^i, g_1), \theta(p_k^i, g_2), \cdots, \theta(p_k^i, g_{|\mathcal{G}|-1}), \theta(p_k^i, g_{|\mathcal{G}|})], \tag{4.1}$$

where θ is a D-dimensional encoder based on a stack of convolution kernels. The prediction module of vanilla EPN is designed for global SE(3)-equivariance for all input points. Differently, our unsupervised multi-body task requires the model's ability to handle *part-level local equivariance*, especially under low-quality training signals. For this purpose, we further devise two heads for rigid segmentation and motion estimation.

Point-level Invariant Segmentation Head. Rigid segmentation is an SE(3)-invariant task, as the predicted mask should remain consistent for the same points across various poses and positions. Traditional SE(3)-equivariant structures assume that all input points undergo the same rigid transformation, which is not in alignment with the multi-body setting. To *encode distinct transformations for individual input points*, the equivariant features $\theta(p_k^i, g_j)$ are aggregated into an

invariant representation u_k^i across the dimension of the rotation group, as depicted in Figure 4.3:

$$u_k^i = \sum_j^{|\mathcal{G}|} w(p_k^i, g_j)\theta(p_k^i, g_j), \tag{4.2}$$

where $w(p_k^i, g_j) \in [0, 1]$ is a selection probability of the discrete rotation g_j in $\mathcal{G}$, derived through a 1×1 convolution. The weighted sum u_k^i is invariant to rigid motion given a point with its neighbors in the convolution receptive field. As Figure 4.3 illustrates, by fusing such invariant representations $u_{k(1)}^i \in \mathbb{R}^D, \cdots, u_{k(h)}^i \in \mathbb{R}^{D'}$ from h layers in the feature extractor, the segmentation head outputs a soft prediction $\hat{M}_k \in [0, 1]^{N \times S}$ of the rigid mask.

Part-level Equivariant Motion Estimation Head. Part-level SE(3)-equivariance is desirable for motion analysis, especially rotation estimation. Based on the noisy predictions $(\hat{M}_k, \hat{M}_l)$ of the frames (k, l) from the head of rigid segmentation, the motion head is supposed to *handle these uncertain category-agnostic parts*. Figure 4.3 demonstrates the operational scheme of our motion estimation head. First of all, the part-level SE(3) feature $V_{k,j} \in \mathbb{R}^{N \times D \times S}$ w.r.t. the rotation $g_j \in \mathcal{G}$ of a single frame k is obtained from the per-point equivariant representations F_k and the predicted rigid mask $\hat{M}_k$:

$$V_{k,j} = \{\hat{m}_k^1 \cdot \theta(p_k^1, g_j), \hat{m}_k^2 \cdot \theta(p_k^2, g_j), \cdots, \hat{m}_k^{N-1} \cdot \theta(p_k^{N-1}, g_j), \hat{m}_k^N \cdot \theta(p_k^N, g_j)\}, \tag{4.3}$$

where $\hat{m}_k^i$ is the element corresponding to a point p_i^k in $\hat{M}_k$, and $\cdot$ is the broadcast operation of Hadamard product. Afterward, $V_{k,j}$ over the rotation group $\mathcal{G}$ is concatenated as $V_k \in \mathbb{R}^{N \times |\mathcal{G}| \times D \times S}$, followed by a permutation-invariant operation $\sigma : \mathbb{R}^{N \times |\mathcal{G}| \times D \times S} \to \mathbb{R}^{|\mathcal{G}| \times D \times S}$ (*e.g.*, max pooling) over all the points to produce part-level equivariant features $\tilde{V}_k = \sigma(V_k)$. Between two frames (k, l),

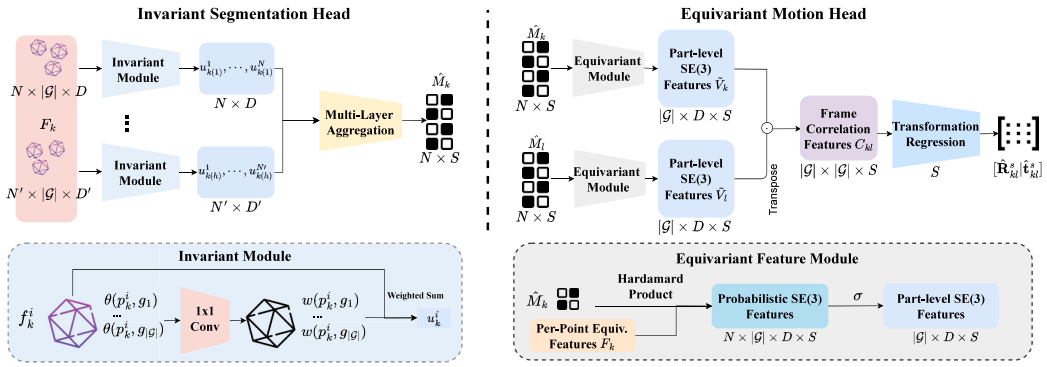


Figure 4.3: *An overview of Segmentation and Motion heads.* The invariant segmentation head comprises an invariant module that sums $\theta(\cdot)$ and $w(\cdot)$ to obtain an invariant representation which is then aggregated through multiple layers to obtain a segmentation mask. Pairs of segmentation masks $\hat{M}_k, \hat{M}_l$ are fed into the equivariant motion head along with per-point equivariant features F_k to obtain correlations features for predicting the final transformations of points.

the part-level rotation correlated feature $C_{kl} \in \mathbb{R}^{|\mathcal{G}|\times|\mathcal{G}|\times S}$ is defined as:

$$C_{kl} = \tilde{V}_k \tilde{V}_l^T, \tag{4.4}$$

where ^{T} is matrix transposition. C_{kl} is calculated upon "softly matching" within each consistent rigid part, while the specific category labels can be agnostic to the model. Based on the correlated feature C_{kl}, the motion head estimates rotation $\hat{\mathbf{R}}_{kl}^s$ and translation $\hat{\mathbf{t}}_{kl}^s$ of each rigid part s. More implementation details can be found in B.1.

4.4.4 Unsupervised Training Strategy

Ideally, the relation among scene flow $\delta_{kl} \in \mathbb{R}^{N\times 3}$, rigid segmentation M_k^s of the s^{th} part, and multi-body transformation $\mathbf{T}_{kl}^s$ is as follows:

$$P_l = P_k + \delta_{kl} = \bigcup_s^S \{\mathbf{T}_{kl}^s \circ (M_k^s \star P_k)\}, \tag{4.5}$$

wherein the operation of union set is denoted as $\bigcup$, and $\star$ signifies the removal of a point p_i^k if it does not belong to the rigid s. Nevertheless, their predictions $\hat{\mathbf{T}}_{kl}^s$, $\hat{M}_k$ and $\hat{\delta}_{kl}$ may exhibit a considerable amount of interdependent noise during the unsupervised training process. To *mitigate the noise in one prediction by leveraging the other two*, we propose an online training strategy without interrupting the end-to-end optimization of a network, as shown in Figure 4.1.

Cold Start & Scene Flow $\hat{\delta}_{kl}$ Updating. Scene flow creates a relation between motion and segmentation. As a dense vector field that maps points to points, it can directly assist point-level segmentation. At the same time, scene flow also contains movement information for motion estimation. We initialize by collecting noisy scene flow from an unsupervised flow estimator (*e.g.*, FlowStep3D [211]), and calculate rudimentary estimates of rigid masks and transformation by minimizing the discrepancy between their derived displacement and scene flow. As the accuracy of their estimates $\hat{M}_k^s$ and $\hat{\mathbf{T}}_{kl}^s$ improves, the scene flow is online corrected during a training epoch:

$$\hat{\delta}_{kl} = \alpha\hat{\delta}_{kl} + (1-\alpha)(\bigcup_{s}^{S}\{\hat{\mathbf{T}}_{kl}^s \circ (\hat{M}_k^s \star P_k)\} - P_k), \tag{4.6}$$

where $\alpha \in [0, 1]$ is a decay factor to control the updating rate. In this manner, improved scene flow is capable of providing enhanced supervision to learn segmentation masks and motion estimates.

Motion & Flow $\rightarrow$ Segmentation $\hat{M}_k^s$. Although scene flow encodes point-level supervisory signals for segmentation, accurately computing flow between non-adjacent frames is challenging, as pointed out by Song and Yang [40]. To alleviate the influence of miscalculated scene flow, we optimize our segmentation head based on the *consensus between the motion head and updated scene*

flow. Given the outputs $\hat{\mathbf{R}}^s_{kl}, \hat{\mathbf{t}}^s_{kl}$ of our motion head, and the interpolation point $p'^i_l = p^i_k + \delta^i_{kl}$ derived from scene flow, we measure how well the motion head's prediction for part s agrees with the scene flow at each point. Specifically, the consensus score $\beta^{s(i)}_{kl}$ of the point p^i_k w.r.t. the rigid part s between the frames (k, l) is defined as:

$$\beta^{s(i)}_{kl} = \exp\big(-\tau \, \|\hat{\mathbf{R}}^s_{kl} \, p^i_k + \hat{\mathbf{t}}^s_{kl} - p'^i_l\|\big), \tag{4.7}$$

where $\tau > 0$ is a temperature coefficient controlling the sharpness of $\beta^{s(i)}_{kl}$, and $\|\cdot\|$ denotes the ℓ_2-norm. A high consensus score indicates that the rigid transformation predicted by the motion head for part s closely matches the scene-flow-based displacement at point p^i_k; a low score signals disagreement. This score is then used to down-weight unreliable regions in the segmentation loss, so that the segmentation head learns primarily from points where the motion head and scene flow are consistent:

$$\mathcal{L}_{seg} = \frac{1}{NS} \sum_{i=1}^{N} \sum_{s=1}^{S} \beta^{s(i)}_{kl} \, \|\hat{M}^{s(i)}_k \, (p'^i_l - \hat{\mathbf{T}}^s_{kl} \circ p^i_k)\|, \tag{4.8}$$

where $\hat{M}^{s(i)}_k$ is the predicted soft mask value for point i belonging to part s, and $\hat{\mathbf{T}}^s_{kl}$ is the rigid transformation computed from the current estimates of scene flow and segmentation, as described in the next paragraph. In words, this loss penalises the residual displacement (the gap between the scene-flow-interpolated position and the rigid-transformation-predicted position) for each point–part pair, weighted by both the consensus score β and the predicted mask $\hat{M}$. Points with low consensus or low mask probability contribute less to the gradient, preventing noisy flow estimates from corrupting the segmentation.

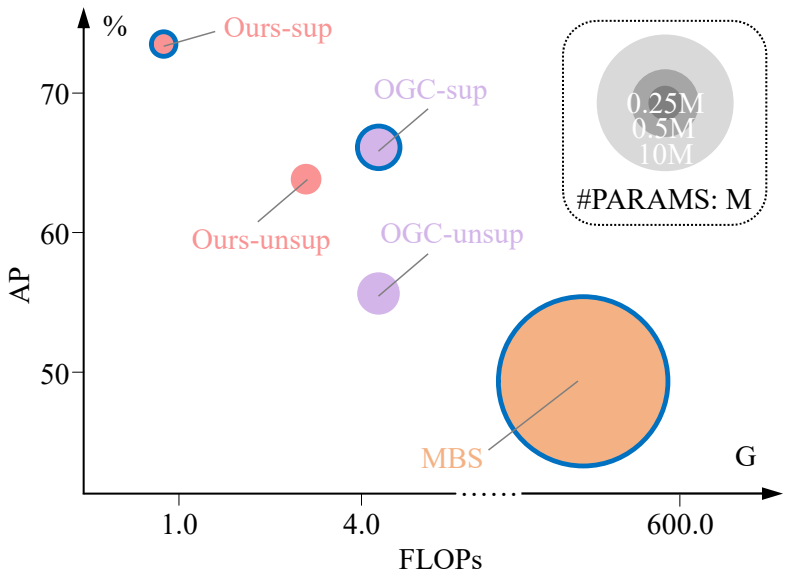


Figure 4.4: *Comparison of FLOPs, parameter number, and AP on SAPIEN.* Supervised methods are marked in ◯.

Segmentation & Flow → Motion $\hat{\mathbf{T}}^s_{kl}$. Following previous works [39, 40], we employ the weighted-Kabsch algorithm [212, 213] to determine part-level rigid transformation $\hat{\mathbf{T}}^s_{kl}$ given the estimates of scene flow and rigid masks. Further, the motion head is optimized by the rotation component of $\hat{\mathbf{T}}^s_{kl}$, and estimates corresponding translation by minimizing our probability-based part-level distance. The implementation details of this distance and our motion loss can be found in B.2.

4.5 Experiments

4.5.1 Datasets & Metrics

Our model is evaluated on three various application scenarios with four datasets: 1) SAPIEN [105] for articulated objects, 2) OGC-DR and its single-view counterpart OGC-DRSV [40] for furniture arrangements; 3) KITTI-SF [104] for vehicular traffic. Following previous works [39, 40, 214], we evaluate the rigid

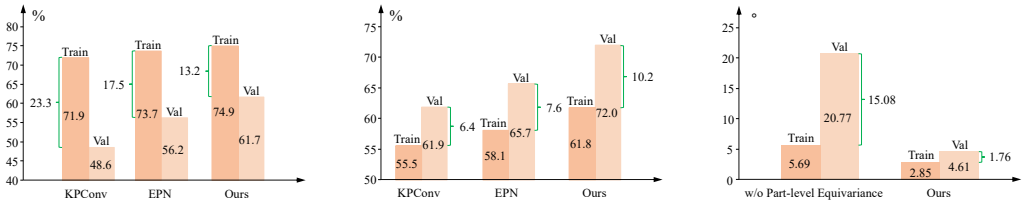


Figure 4.5: *Results of pilot studies.* (a)(b) are the experimental results of our segmentation head, while (c) is the exploration outcomes of the motion head.

segmentation performance on the following seven measures: Average Precision (**AP**), Panoptic Quality (**PQ**), F1-score (**F1**), Precision (**Pre**), and Recall (**Rec**) at an Intersection over Union threshold of 0.5, in addition to the mean Intersection over Union (**mIoU**) and Rand Index (**RI**). In terms of multi-body motion estimation, we report End-Point-Error 3D (**EPE3D**) in the main body, and other metrics (*e.g.*, Outlier) are provided in B.3.

4.5.2 Pilot Studies on The Two Heads

We first conduct experiments on SAPIEN to verify the generalizability of our two-head structure, including the point-level invariance of the segmentation head and the part-level equivariance of the motion head.

Can the point-level invariance of our segmentation head help generalize to open-set motion? On SAPIEN, each articulated target has four frames with various part-level rigid orientations and locations. By training the segmentation head on a subset of the 1^{st} frames (with the canonical pose in [39]) and testing them on all frames, we compare its performance with the segmentation results using the same parameters of the non-equivariant counterpart KPConv [29] and global equivariant EPN. As shown in Figure 4.5(a)(b), the SE(3)-equivariance can essentially boost the generalization even given the global since there are some targets

that only contain two rigid parts. By introducing point-level invariance, both precision and recall can be further improved.

Can the part-level equivariance of our motion head help generalize to category-agnostic parts? Note that the category of targets for training, validation, and testing on SAPIEN is completely disjoint. By optimizing the motion head on the training set and testing its performance on the validation data, we ensure that the targets do not overlap in categories so their parts are entirely category-agnostic. To exclude the interference of part segmentation, we directly provide the ground-truth rigid partition as the mask. By comparing the decrease in performance on the validation data when using part-level equivariance *vs*. not using it, as illustrated in Figure 4.5(c), it is observed that the mean angular error of our model only increases by less than $2°$, while that without part-level equivariance surges from $5.69°$ to $20.77°$. This demonstrates the strong generalizability of our model to category-agnostic parts.

Table 4.1: *Ablation studies on SAPIEN.*

Seg. Head		Scene Flow		Mot. Head	Metrics						
SE(3) Feat.	Point-level Flexibility	Current	Past	Consensus	AP↑	PQ↑	F1↑	Pre↑	Rec↑	mIoU↑	RI↑
					45.2	44.2	58.9	53.8	65.1	60.9	71.2
✓					51.7	50.0	65.8	64.7	67.0	61.6	72.3
✓	✓				55.3	52.8	68.3	65.9	70.0	62.3	72.7
✓	✓	✓			54.8	52.0	67.6	66.0	69.3	63.5	73.8
✓	✓	✓	✓		57.0	51.6	67.3	63.8	71.1	63.1	73.2
✓	✓	✓	✓	✓	63.8	61.3	77.3	84.2	71.3	63.7	75.4

4.5.3 Ablation Studies

We perform detailed ablation studies on SAPIEN to explore the effects of three main components: segmentation head, motion head, and scheme of scene flow

updating.

Segmentation Head. Two crucial design elements significantly enhance our segmentation head: 1) the use of SE(3)-equivariant features, and 2) point-level flexibility for invariant representations. By simply introducing the global equivariant features, AP increases from 45.2% to 51.7%. This demonstrates the importance of SE(3)-equivariance in modeling rigid transformations. The proposed point-level flexibility further boosts the performance to 55.3%, suggesting that our point-level invariance is highly advantageous to the multi-body task.

Scene Flow Updating Scheme. For the updated choice of scene flow, we empirically test three options: 1) completely relying on the past flow extracted by an unsupervised flow network, 2) ignoring the past flow and directly using the current estimation, and 3) combining them together with a decay factor. It is observed that the extreme choices of 1) or 2) result in minor performance differences (AP: 55.3% *vs*. 54.8%). The significant improvement comes from the weighted combination in option 3), achieving the AP of 57.0%.

Motion Head. The motion head contributes to the consensus score for the removal of scene flow noise. From the last row of Table 4.1, the consensus between the motion head and scene flow significantly boosts the final segmentation performance in most of the metrics. By controlling the point-level learning weight during the training process, our motion head can assist in optimizing the segmentation head.

Table 4.2: *Rigid segmentation results on SAPIEN.* * indicates that we evaluate these metrics upon the officially released model; - means that the metric is unavailable.

		AP↑	PQ↑	F1↑	Pre↑	Rec↑	mIoU↑	RI↑	EPE3D↓
Supervised Methods	PointNet++ [28]	-	-	-	-	-	51.2	65.0	-
	MeteorNet [14]	-	-	-	-	-	45.7	60.0	-
	DeepPart [195]	-	-	-	-	-	53.0	67.0	5.95
	MBS [39]	49.4*	52.6*	67.6*	61.4*	75.2*	67.3	77.0	5.03
	OGC-sup [40]	66.1	48.7	62.0	54.6	71.7	66.8	77.1	-
	Ours-sup	**73.5**	**57.8**	**71.1**	**65.6**	**77.7**	**72.6**	**81.4**	**3.86**
Unsupervised Methods	TrajAffn [215]	6.2	14.7	22.0	16.3	34.0	45.7	60.1	-
	SSC [214]	9.5	20.4	28.2	20.9	43.5	50.6	65.9	-
	WardLinkage [216]	17.4	26.8	40.1	36.9	43.9	49.4	62.2	-
	DBSCAN [217]	6.3	13.4	20.4	13.9	37.9	34.2	51.4	-
	NPP [191]	-	-	-	-	-	51.5	66.0	21.22
	OGC [40]	55.6	50.6	65.1	65.0	65.2	60.9	73.4	-
	Ours	**63.8**	**61.3**	**77.3**	**84.2**	**71.3**	**63.7**	**75.4**	**5.47**

4.5.4 Results & Comparisons

4.5.4.1 SAPIEN

SAPIEN is a challenging dataset of articulated objects since its training, validation and testing sets comprise completely disjoint categories of objects. This domain gap in the data split complicates the precise segmentation of rigid parts. Adapting our model to the supervised formulation is straightforward by optimizing the segmentation with ground-truth training labels. Therefore, we also report the supervised performance of the proposed framework. As shown in Table 4.2, our segmentation performance *sets a new benchmark across all metrics* in both supervised and unsupervised settings.

In line with the default setting of [39], we estimate motion between both consecutive frames and non-adjacent ones. Table 4.2 reports its EPE3D performance and comparison: our unsupervised performance of 5.47 EPE3D on motion estimation is on par with the best existing supervised method (MBS), while our supervised

EPE3D achieves 3.86, marking a relative error reduction of approximately 23%.

As depicted in Figure 4.4, owing to the lightweight two-head structure, our model encompasses only a small number of parameters (0.25M) and incurs a low cost in computational complexity (0.92G floating point operations, FLOPs). As previewed in Figure 4.2, our unsupervised predictions are comparable to—and often cleaner than—those of supervised baselines. To further demonstrate that this advantage is consistent across a broader range of object categories and pose variations, Figure 4.6 presents an extended side-by-side comparison with OGC [40] and MultiBodySync [39] under both supervised and unsupervised settings. Across all examples, our method produces sharper motion boundaries and fewer over-segmentation artifacts, confirming that the SE(3)-equivariant formulation generalises reliably to category-agnostic, open-set scenarios.

Table 4.3: *Rigid segmentation results on OGC-DR and OGC-DRSV.*

		AP↑	PQ↑	F1↑	Pre↑	Rec↑	mIoU↑	RI↑
Supervised Methods	OGC-sup [40]	90.7 / 86.3	82.6 / 78.8	87.6 / 85.0	83.7 / 82.2	92.0 / 88.0	89.2 / 83.9	97.7 / 97.1
	Ours-sup	**92.8 / 89.3**	**86.9 / 82.6**	**91.0 / 87.9**	**88.8 / 85.5**	**93.2 / 90.4**	**91.2 / 86.6**	**98.7 / 97.9**
Unsupervised Methods	TrajAffn [215]	42.6 / 39.3	46.7 / 43.8	57.8 / 54.8	69.6 / 63.0	49.4 / 48.4	46.8 / 45.9	80.1 / 77.7
	SSC [214]	74.5 / 70.3	79.2 / 75.4	84.2 / 81.5	92.5 / 89.6	77.3 / 74.7	74.6 / 70.8	91.5 / 91.3
	WardLinkage [216]	72.3 / 69.8	74.0 / 71.6	82.5 / 80.5	**93.9 / 91.8**	73.6 / 71.7	69.9 / 67.2	94.3 / 93.3
	DBSCAN [217]	73.9 / 71.9	76.0 / 76.3	81.6 / 81.8	85.8 / 79.1	77.8 / 84.8	74.7 / 80.1	91.5 / 93.5
	OGC [40]	92.3 / 86.8	85.1 / 77.0	89.4 / 83.9	85.6 / 77.7	93.6 / 91.2	90.8 / 84.8	97.8 / 95.4
	Ours	**93.9 / 88.1**	**87.0 / 80.0**	**91.1 / 86.1**	87.0 / 80.8	**95.6 / 92.2**	**92.4 / 86.7**	**98.1 / 96.6**

4.5.4.2 OGC-DR & Single-view Counterpart

OGC-DR is a dataset for indoor furniture arrangements, where the training, validation, and testing instances are distinct from one another. OGC-DRSV, the single-view version of OGC-DR, presents a challenge due to the incomplete nature of the furniture caused by occlusion, making it difficult to identify consistent rigidity. As demonstrated in Table 4.3, the proposed framework constantly surpasses the state-of-the-art models across all measures. Remarkably, even under the most challenging condition, *i.e.*, unsupervised single-view, our AP still

achieves the value of 88.1%. This underscores the robustness of our model in handling incomplete observations. The performance on motion estimation can be found in B.3.

Table 4.4: *Rigid segmentation results on KITTI-SF.* Our model still achieves competitive results even though the data setting is inconsistent with the model's assumption.

Method Category	Method	AP↑	PQ↑	F1↑	Pre↑	Rec↑	mIoU↑	RI↑
Supervised Methods	OGC-sup [40]	62.4	52.7	65.1	63.4	67.0	67.3	95.0
	Ours-sup	**65.1**	**56.3**	**68.6**	**69.4**	**67.8**	**69.5**	**95.7**
Unsupervised Methods	TrajAffn [215]	24.0	30.2	43.2	37.6	50.8	48.1	58.5
	SSC [214]	12.5	20.4	28.4	22.8	37.6	41.5	48.9
	WardLinkage [216]	25.0	16.3	22.9	13.7	**69.8**	60.5	44.9
	DBSCAN [217]	13.4	22.8	32.6	26.7	42.0	42.6	55.3
	OGC [40]	**54.4**	42.4	52.4	47.3	58.8	**63.7**	**93.6**
	Ours	53.6	**44.4**	**55.1**	**56.3**	54.0	61.5	93.4

4.5.4.3 KITTI-SF

Strictly speaking, KITTI-SF is not a multi-body rigid dataset, as the background in its point clouds may be deformable. This is inconsistent with the assumption of our framework, which posits that the feature is only equivariant for rigid transformations. However, as shown in Table 4.4, our framework still delivers competitive performance in rigid segmentation despite this inconsistency.

4.5.4.4 Generalization to KITTI-Det & SemanticKITTI

Following the same evaluation protocol of OGC, we further evaluate the generalizability of our model from KITTI-SF to two larger outdoor datasets, i.e., KITTI-Det [103] and SemanticKITTI [10], and compare the performance with the results reported in OGC, as shown in Table 4.5& 4.6.

It is important to clarify the scope of our comparison. The SemanticKITTI benchmark hosts numerous methods for supervised semantic segmentation (*e.g.*, Cylin-

der3D [218], SPVNAS [219], 2DPASS [220]) that achieve substantially higher performance on that benchmark's leaderboard. However, our task—rigid-body motion segmentation—is fundamentally different from semantic segmentation. The leaderboard methods classify points into predefined semantic categories (*e.g.*, car, road, building) using dense per-point category labels, whereas our methods—both supervised and unsupervised variants—segment the scene into independently moving rigid parts based on motion consistency. The supervised variants (Ours-sup and OGC-sup) use rigid-part motion annotations rather than semantic category labels; they are included following the evaluation protocol established by OGC [40] to provide an upper-bound reference for the unsupervised setting. We compare with OGC as it is the most directly comparable method addressing the same problem formulation of multi-body rigid motion segmentation.

Table 4.5: *Segmentation performance on KITTI-Det.*

Methods	**AP**↑	**PQ**↑	**F1**↑	**Pre**↑	**Rec**↑	**mIoU**↑	**RI**↑
OGC-sup [40]	51.4	41.0	49.1	43.7	56.0	66.2	91.0
Ours-sup	**52.5**	**43.3**	**51.8**	**47.5**	**57.0**	**68.0**	**92.6**
OGC-unsup [40]	40.5	30.9	37.0	30.8	**46.5**	**60.6**	86.4
Ours-unsup	**41.3**	**32.9**	**38.8**	**35.3**	43.1	60.2	**87.2**

In Table 4.6, we report results on three groupings of sequences: all sequences (00–10), sequences excluding Sequence 08 (00–07 & 09–10), and Sequence 08 alone. This separate reporting follows the standard SemanticKITTI protocol established by [10], where Sequence 08 serves as the designated validation set while the remaining sequences are used for training. We follow the same convention as OGC [40] to ensure a fair and consistent comparison. The performance difference on Sequence 08 compared to other sequences can be attributed to its distinct driving environment and traffic patterns, which differ from the training distribution.

Table 4.6: *Segmentation performance on SemanticKITTI.*

Sequences	Methods	AP↑	PQ↑	F1↑	Pre↑	Rec↑	mIoU↑	RI↑
00 - 10	OGC-sup [40]	53.8	41.3	48.1	40.1	60.0	68.3	90.0
	Ours-sup	**60.1**	**47.6**	**55.4**	**48.6**	**64.4**	**71.9**	**93.4**
	OGC-unsup [40]	42.6	30.2	35.3	28.2	47.3	60.3	86.0
	Ours-unsup	**46.9**	**31.6**	**36.9**	**29.0**	**50.6**	**63.2**	**88.7**
00 - 07 & 09 - 10	OGC-sup [40]	55.3	41.8	48.4	40.1	61.1	69.9	90.3
	Ours-sup	**60.5**	**48.1**	**55.6**	**48.8**	**64.7**	**73.2**	**93.8**
	OGC-unsup [40]	43.6	30.5	35.5	28.1	48.2	62.1	86.3
	Ours-unsup	**47.4**	**31.7**	**36.8**	**28.7**	**51.0**	**64.8**	**89.3**
08	OGC-sup [40]	49.4	39.2	46.6	40.0	55.8	60.3	88.3
	Ours-sup	**58.4**	**46.0**	**54.4**	**47.8**	**63.1**	**65.8**	**91.7**
	OGC-unsup [40]	38.6	29.1	34.7	28.6	44.0	51.8	84.3
	Ours-unsup	**44.2**	**31.0**	**37.3**	**30.0**	**49.1**	**55.8**	**86.1**

4.6 Limitations & Future Work

Our model is predicated on an assumption: the part-level motion should be a rigid transformation. Therefore, as shown in Table 4.3, if the observation of part-level movements is non-rigid (such as occluded single views), it would suffer a performance decrease. In the future, we would like to develop a model that is more robust to its observation.

4.7 Conclusion

This chapter introduces a part-level SE(3)-equivariant framework for modeling multi-body rigid motion. The two heads for rigid segmentation and motion estimation are meticulously designed based on their inherent invariant and equivariant characteristics. The relationship among scene flow, rigid segmentation, and multi-body transformation is then exploited to derive an unsupervised optimization strategy. Our approach achieves state-of-the-art results on multiple datasets

while significantly reducing the required parameters and computations.

Within the multi-level programme of this thesis, this chapter complements the high-level semantic classifier of Chapter 3 by operating at a finer granularity: rather than assigning a single label to an entire sequence, we discover part-level structure and estimate per-part rigid transformations. Both chapters share the overarching goal of learning effective dynamic point cloud representations, but they differ in abstraction level, supervision paradigm, and output format. Importantly, the unsupervised nature of the method presented here demonstrates that strong geometric priors—specifically SE(3) equivariance—can compensate for the absence of manual annotations, a theme that resonates across all three contributions.

While this chapter addresses part-level motion at an intermediate granularity, a complementary challenge lies in reconstructing complete 4D geometry from minimal observations. The next chapter tackles this low-level reconstruction problem, proposing MotionStruct4D for video-to-4D generation. Notably, MotionStruct4D also decomposes motion into rigid and non-rigid components, echoing the rigid-body assumptions explored here but extending them to a generative setting with 3D Gaussian Splatting representations.

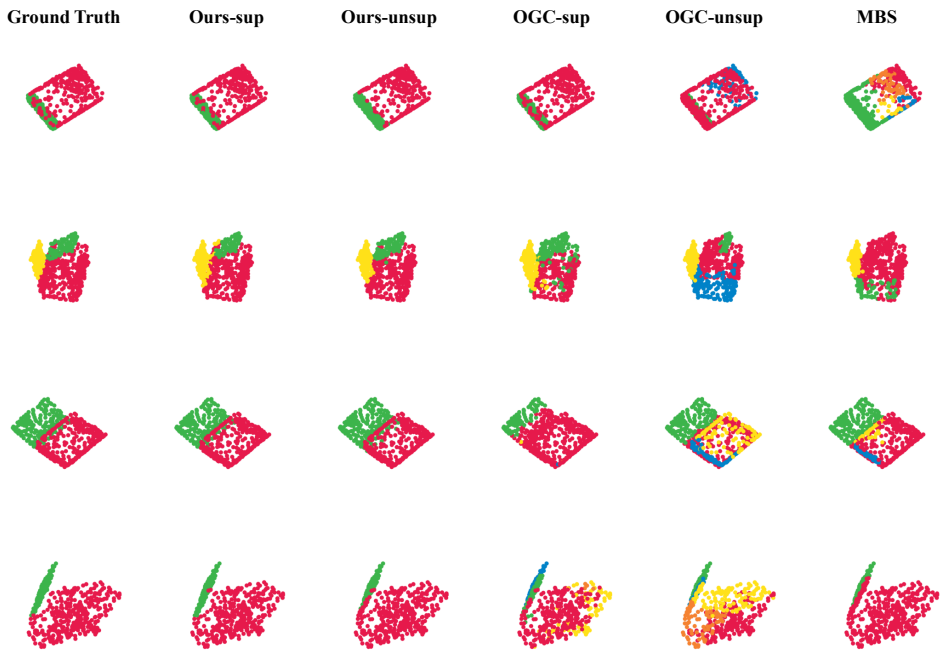


Figure 4.6: *Extended qualitative comparison of rigid segmentation on SAPIEN.* We compare our method with OGC [40] and MultiBodySync (MBS) [39] under both supervised and unsupervised settings across a diverse set of articulated objects. **Ground Truth** denotes the reference segmentation. Our unsupervised model (**Ours-unsup**) produces coherent rigid parts with sharp motion boundaries, whereas **OGC-unsup** exhibits over-segmentation and noisy label bleeding. Our supervised variant (**Ours-sup**) further captures fine-grained part details, outperforming **OGC-sup** and **MBS**. These results confirm that the SE(3)-equivariant formulation generalises reliably across open-set pose changes and diverse object categories. Additional visualisations are provided in B.5.

5 | Low-level Geometric Reconstruction: Video-to-4D Generation via Motion Structure Discovery

This chapter addresses a problem that is distinct from those in Chapters 3 and 4: reconstructing complete 4D representations from single-view monocular video. While the preceding chapters focused on classification and motion analysis of point cloud sequences, this chapter operates in the domain of 4D generation using 3D Gaussian Splatting, with its own technical formulation and evaluation criteria.

5.1 Abstract

Monocular Video-to-4D generation faces the fundamental challenge of inferring plausible 3D geometry and motion from limited single-view inputs. We present MotionStruct4D, a novel approach that discovers and exploits the underlying motion structure of 3D Gaussian Splatting for high-quality video-to-4D generation. Our key insight is that real-world motion can be effectively decomposed into coarse rigid transformations that capture principal movements, complemented by detailed non-rigid deformations that account for fine-grained details. MotionStruct4D introduces: (1) a self-supervised motion structure discovery module that identifies quasi-rigid parts by preserving spatio-temporal relationships without explicit 3D supervision, and (2) a weighted dense-to-sparse optimization architecture that transitions from dense per-Gaussian deformation to sparse control points, effectively integrating rigid and non-rigid motion components through adaptive weighted fusion. This design addresses the parametric imbalance between rigid and non-rigid motions and effectively models 3D movements across different

kinematic patterns. To evaluate our approach on challenging scenarios, we curate a comprehensive benchmark dataset featuring substantial object displacement and diverse articulated motion patterns. Experimental results demonstrate MotionStruct4D's superior performance in motion fidelity and novel viewpoint synthesis quality, while also providing interpretable motion structure decomposition that reveals meaningful quasi-rigid part segmentation. Video demonstrations of our results are available at https://jx-zhong-for-academic-purpose.github.io/MotionStruct4DPage/.

5.2 Introduction

Recent progress in 4D content generation has enabled dynamic 3D assets to be generated from a variety of inputs, including text [221], images [67, 96], and videos [63, 66, 81]. Among these modalities, video-to-4D generation presents a particularly promising yet challenging approach, as it allows users to transform ordinary monocular video captures into viewable 4D assets from various perspectives. Unlike *traditional 4D reconstruction* [57, 58, 93, 94, 222–225], video-to-4D generation faces a fundamentally more difficult ill-posed problem. 4D reconstruction typically establishes a direct correspondence between input and output views–either reconstructing forward-facing views from forward-facing inputs, or synthesizing free viewpoints from multi-directional source views. In contrast, *video-to-4D generation* is required to infer plausible 3D geometry and motion that can be *rendered from arbitrary viewpoints only given a single-view monocular video as input*. The core challenge in video-to-4D generation lies in designing effective motion representations. Despite extensive exploration of various techniques using Neural Radiance Fields (NeRFs) [64, 68], HexPlane [93], and 3D Gaussian Splatting [63, 66, 81], existing approaches frequently struggle to simultaneously

achieve high-fidelity rendering and consistent motion across novel viewpoints.

Intuitively, real-world motion usually exhibits structural patterns that can be approximated by a limited set of primary transformations. Therefore, complex movements can be represented as a combination of several rigid transformations that capture the principal global movements, complemented by local non-rigid deformations that account for fine-grained details. Based on this physical insight, we propose MotionStruct4D, a novel approach that discovers and exploits the underlying motion structure of 3D Gaussian Splatting for high-quality video-to-4D generation.

However, the implementation of this rigid-nonrigid motion approach presents two significant challenges: *supervising motion structure discovery* and *jointly modeling rigid and non-rigid transformations*. For motion structure discovery, the absence of multi-view constraints hampers the inference of underlying 3D motion structures, particularly as Gaussian point locations evolve throughout the learning process, complicating rigid-body identification. For modeling rigid-nonrigid motion simultaneously, the joint optimization of rigid and non-rigid motions is non-trivial due to their inherent parametric imbalance–rigid motions are comparatively few but structurally significant, while non-rigid motions are numerous and possess substantially greater degrees of freedom, allowing Gaussian points to traverse any directions at any distances. This asymmetry frequently causes non-rigid motion to dominate the optimization process, compromising overall motion coherence.

To address these two challenges, our approach incorporates: (1) self-supervised motion structure discovery using spatio-temporal relationship preservation, and (2) balanced rigid-nonrigid motion modeling through weighted dense-to-sparse optimization. Figure 5.1 illustrates our MotionStruct4D pipeline. For *identifying*

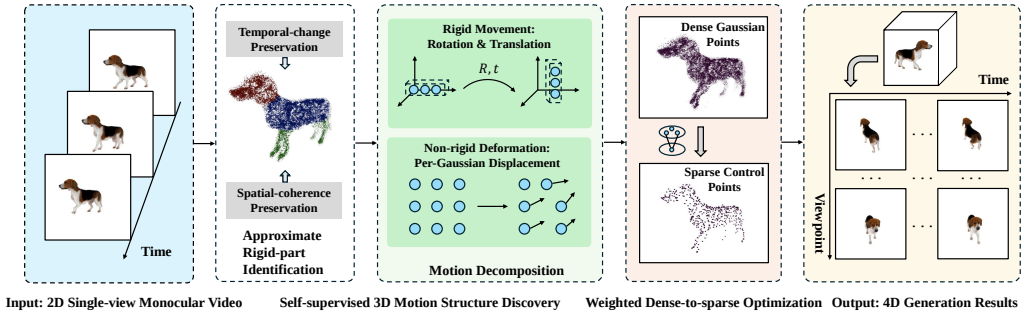


Figure 5.1: *Our pipeline for video-to-4D generation.* Given a single-view monocular video input, we first identify nearly rigid-body parts in a self-supervised way that approximates Gaussian point motion with rigid movements. We then decompose motion into rigid and non-rigid transformations. Through our weighted dense-to-sparse optimization (where darkness represents rigid-motion weights), we integrate these complementary motion components by transitioning from dense Gaussian representations to sparse control points, and produce 4D content across arbitrary viewpoints and timestamps.

quasi-rigid parts without explicit 3D supervision, we first approximate Gaussian point motion through multiple movements by preserving both temporal changes and spatial coherence. After obtaining approximate rigid parts, we capture the motion structure of 3D Gaussians via the *decomposition into rigid and non-rigid transformations*. For joint rigid-nonrigid modeling, we optimize the proposed framework in a *weighted dense-to-sparse manner*. By transitioning from dense per-Gaussian deformation to sparse control points, we significantly reduce the number of non-rigid parameters while maintaining expressiveness. Our learnable weighting mechanism further balances the influence of rigid and non-rigid components, adaptively integrating them based on their relative importance in different regions.

To rigorously evaluate our approach, particularly in scenarios with diverse motions, we curate a comprehensive dataset of moving animal videos by restructuring ARTEMIS-DFA [226]. In contrast to existing benchmarks (*e.g.* Consistent4D [63]), which typically have limited object displacement and insufficient

motion diversity for individual objects, our dataset encompasses both large global object displacement and multiple motion patterns for the same object. This constitutes a challenging and ecologically valid testbed for video-to-4D generation across various kinematic patterns. Experimental results demonstrate MotionStruct4D's superior performance in motion fidelity and novel viewpoint synthesis quality, while also providing interpretable motion structure decomposition that reveals meaningful quasi-rigid part segmentation.

In summary, our contributions are three-fold:

- A self-supervised motion structures discovery module that identifies quasi-rigid parts and decomposes Gaussian point transformations from single-view videos without explicit 3D supervision.
- A dense-to-sparse optimization architecture with adaptive weighted fusion, effectively integrating rigid and non-rigid motion components.
- A curated large-motion benchmark dataset featuring substantial object displacement and diverse articulated motion patterns of the same object, establishing a more comprehensive evaluation benchmark for video-to-4D generation.

5.3 Related Works

3D Generation. Early research in 3D generation achieved high-quality textured 3D mesh generation [74, 75, 227, 228]. More recently, diffusion model-based methods have become prominent in 3D generation algorithms [229–234]. DreamFusion [122] first introduced the score distillation sampling (SDS) loss for text-to-3D generation, optimizing 3D shapes with pre-trained 2D diffusion models. To address the multi-view Janus problem (inconsistent appearances across views), methods such as MVDream [95], SweetDreamer [235], Zero123 [96], and Sync-

Dreammer [97] fine-tune 2D diffusion models to enable explicit control over image generation from different viewpoints. With the advancement of 3D Gaussian Splatting representations, approaches like DreamGaussian [233] and GaussianDreamer [236] have pioneered its use as the underlying 3D structure, enabling efficient 3D object generation within minutes.

4D Representation. 4D representations aim to model dynamic geometry and appearance over time. Traditional approaches often extend static Neural Radiance Fields (NeRFs) [77] to dynamic scenes by incorporating deformability [57, 58, 78, 92] or time-varying components [93, 94, 237–239]. However, these methods can suffer from high computational costs and slow convergence. Recently, 3D Gaussian Splatting has emerged as a popular alternative due to its explicit structure, which facilitates real-time rendering. Methods such as Deformable 3D Gaussians and SC-GS [61] model motion via deformation fields or sparse control points (e.g., using Linear Blend Skinning (LBS) or Direct Quaternion Skinning (DQS)), enabling efficient and temporally consistent reconstruction. Compared to implicit radiance fields, Gaussian-based representations [59, 60, 66, 67, 80, 81, 221, 240, 241] generally offer superior optimization efficiency and spatio-temporal fidelity, making them well-suited for scalable 4D content generation.

Video-to-4D Generation. Building on the significant progress in 3D generation, a growing body of research now focuses on the more challenging task of video-to-4D generation [64–68, 97, 221, 242]. These methods often leverage existing multi-view diffusion models to compute an SDS loss [62, 63, 81] or to generate per-frame multi-view images [64, 65, 243] that serve as supervision signals. Among these, SC4D [81], a concurrent work to our method, optimizes a set of sparsely-controlled dynamic 3D Gaussians using a per-frame SDS loss. DreamMesh4D [66] proposes a hybrid representation, incorporating sparse con-

trol Gaussians and mesh-surface bindings to achieve physically plausible motion and efficient optimization. In contrast to these approaches, our MotionStruct4D introduces explicit motion structure awareness through self-supervised rigid-part discovery. This enables our method to maintain structural coherence across arbitrary viewpoints by understanding and decomposing the underlying motion of the scene.

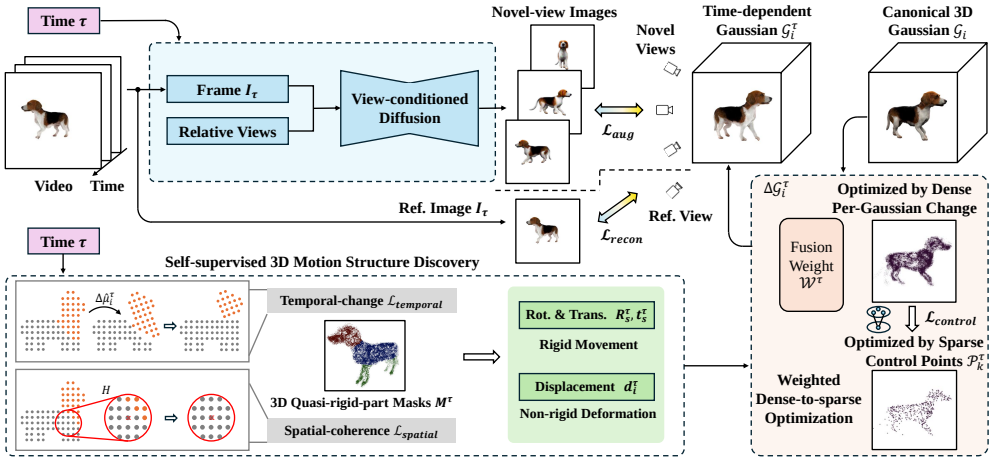


Figure 5.2: *Framework of MotionStruct4D.* There are three main components: **(1)** View-conditioned diffusion generates multi-view contents for $\mathcal{L}_{\text{aug}}$, while $\mathcal{L}_{\text{recon}}$ provides direct supervision from input frames. **(2)** 3D motion structure discovery identifies quasi-rigid moving parts through self-supervised losses ($\mathcal{L}_{\text{temporal}}$ and $\mathcal{L}_{\text{spatial}}$), enabling rigid-nonrigid motion decomposition. **(3)** Dense-to-sparse optimization from dense per-Gaussian change to sparse control points $\mathcal{P}_k^\tau$, with a learnable fusion weight $\mathcal{W}^\tau$. In this way, the canonical 3D Gaussians $\mathcal{G}_i$ are transformed to time-dependent Gaussians $\mathcal{G}_i^\tau$ via the learned variation $\Delta\mathcal{G}_i^\tau$, enabling consistent novel view synthesis across space and time.

5.4 Methodology

5.4.1 Problem Statement

Video-to-4D generation aims to create a time-varying 3D representation from a monocular video. Formally, given a T-frame single-view video $\mathcal{I} = \{I_\tau\}_{\tau=1}^{T}$ captured from a certain camera viewpoint, our goal is to infer its 4D representation $\mathcal{O}$ that can be rendered from arbitrary viewpoints v and timestamps τ. The input video $\mathcal{I}$ provides only single-view frames but a model is required to restore complete volumetric information at each timestamp. This inherently ill-posed nature distinguishes our task from conventional 4D reconstruction approaches, which typically benefit from multi-view spatial correspondences or consider forward-facing perspectives only.

5.4.2 Preliminaries: 4D Gaussian Splatting via Canonical Mapping

To extend 3D Gaussian Splatting (3D-GS) [79] to dynamic scenes, we adopt the canonical-mapping paradigm [60]. This approach establishes a canonical space of 3D Gaussians $\{\mathcal{G}_i\}_{i=1}^{N}$ and models per-Gaussian temporal variations $\Delta\mathcal{G}_i^\tau$ at any timestep τ, which effectively preserves a consistent underlying scene structure while accommodating temporal dynamics.

Canonical 3D Gaussians. By constructing a canonical space with static 3D Gaussians, we store time-invariant 3D information of the video. A static 3D Gaussian $\mathcal{G}$ is parameterized by a center position $\boldsymbol{\mu} \in \mathbb{R}^3$, a covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{3\times3}$ (typically specified by scale $\boldsymbol{s} \in \mathbb{R}^3$ and rotation quaternion $\boldsymbol{q} \in \mathbb{R}^4$),

opacity $\sigma \in \mathbb{R}$, and l-order spherical harmonic coefficients for color $\boldsymbol{h} \in \mathbb{R}^{3\times l^2}$. These Gaussians are rendered from the viewpoint $\boldsymbol{v}$ via alpha blending [244] through a differentiable splatting operation to compute the color $C(u)$ at a pixel u:

$$C(u) = \sum_i c_i \alpha_i \prod_{j=1}^{i-1} (1 - \alpha_j), \tag{5.1}$$

where $c_i = \text{SH}(\boldsymbol{h}_i, \nu)$ represents the color of the i-th Gaussian evaluated at view direction ν based on the spherical harmonic function SH, $\alpha_i = \sigma_i G_i(\boldsymbol{x})$ represents the opacity contribution of the i-th Gaussian at 3D position $\boldsymbol{x}$ corresponding to pixel u, with $G_i(\boldsymbol{x}) = \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_i)\right)$ the Gaussian function calculated at position $\boldsymbol{x}$.

Time-aware Dynamics Modeling. Based on the canonical Gaussians, we define a time-dependent deformation function $\mathcal{F}$ that maps canonical Gaussians to their corresponding states at time τ, with the change $\Delta\mathcal{G}_i^\tau$:

$$\Delta\mathcal{G}_i^\tau = \mathcal{F}(\mathcal{G}_i, \tau) = \{\Delta\boldsymbol{\mu}_i^\tau, \Delta\boldsymbol{q}_i^\tau, \Delta\boldsymbol{s}_i^\tau\}, \tag{5.2}$$

where $\Delta\boldsymbol{\mu}_i^\tau$, $\Delta\boldsymbol{q}_i^\tau$, and $\Delta\boldsymbol{s}_i^\tau$ denote the changes in position, rotation, and scaling, respectively. The final Gaussian $\mathcal{G}_i^\tau$ is obtained by applying these changes to the canonical Gaussian $\mathcal{G}_i$. Prior work using this paradigm exhibits two critical limitations: (1) insufficient motion structure awareness, as it treats all deformations as non-rigid, and (2) computational inefficiency when modeling common motion patterns such as articulated movements. Differently, our approach explicitly models motion as a composition of rigid-body transformations and non-rigid deformations, thereby more accurately representing real-world dynamic objects.

5.4.3 MotionStruct4D

As shown in Figure 5.1, the MotionStruct4D pipeline consists of two main components: (1) self-supervised 3D motion structure discovery, which comprises approximate rigid-part identification and motion decomposition of Gaussian points, and (2) a weighted dense-to-sparse optimization strategy. Figure 5.2 illustrates the detailed framework. Following common practice, we utilize view-conditioned diffusion models (*e.g.*, Zero123) to augment multi-view training data. Although these synthetic views lack temporal consistency, they provide stable 3D spatial supervision. We employ Score Distillation Sampling (SDS) [96, 242] to compute the augmentation loss $\mathcal{L}_{\text{aug}}$ (details in C.2) as a component of our objective function. Additionally, we incorporate reconstruction loss $\mathcal{L}_{\text{recon}}$ for the original frames in the input video. Both $\mathcal{L}_{\text{recon}}$ and $\mathcal{L}_{\text{aug}}$ provide content constraints for the rendered images, initially training the Gaussian Splatting representation.

Given an input video $\mathcal{I}$ with any frame I_τ, we aim to model the time-aware change $\Delta\mathcal{G}_i^\tau$ for each canonical 3D Gaussian $\mathcal{G}_i$ corresponding to this frame, obtaining $\mathcal{G}_i^\tau$ for video-to-4D generation. We view $\Delta\mathcal{G}_i^\tau$ as Gaussian "motion" that reflects the underlying 3D motion structure of the video.

5.4.3.1 Self-supervised Approximate Rigid-part Identification

To achieve motion structure discovery, our first step is identifying the main motion components of Gaussian point motion. We aim to approximate Gaussian point motion with rigid movements and identify nearly rigid-body parts in a self-supervised manner. During this stage, we focus on modeling the positional displacement $\Delta\boldsymbol{\mu}_i^\tau$ of a canonical Gaussian point $\mathcal{G}_i$ in the τ^{th} frame for simplicity. Other changes, such as each Gaussian's intrinsic rotation $\Delta\boldsymbol{q}_i^\tau$, are modeled separately in the non-rigid per-Gaussian deformation component.

For a specific canonical Gaussian point centered at $\boldsymbol{\mu}_i$, we identify potential rigid motion patterns to model its location change $\Delta\boldsymbol{\mu}_i^\tau$. We define a learnable rigid-part mask function $\mathcal{M}^\tau$ that assigns each point a soft assignment probability vector $\boldsymbol{m}_i^\tau \in \mathbb{R}^S$, where S represents the maximum number of potential rigid parts. Our objective is to learn an effective $\mathcal{M}^\tau$ through self-supervision.

Temporal-change Preservation. First, these quasi-rigid movements should preserve the temporal changes of Gaussian point locations as much as possible, with Gaussian points belonging to the same rigid part undergoing similar transformations. For the s-th rigid part, we estimate its transformation matrix $\boldsymbol{T}_s^\tau \in SE(3)$ using a differentiable weighted-Kabsch algorithm [79] (details in C.2). The rigid-part division should maximize the preservation of Gaussian point location temporal changes:

$$\mathcal{L}_{\text{temporal}} = \frac{1}{N}\sum_{i=1}^{N}\left|\left|\sum_{s=1}^{S} m_{is}^\tau(\boldsymbol{T}_s^\tau \circ \boldsymbol{\mu}_i) - (\boldsymbol{\mu}_i + \Delta\hat{\boldsymbol{\mu}}_i^\tau)\right|\right|_2^2, \tag{5.3}$$

where m_{is}^τ represents the probability of the i-th point belonging to the s-th rigid part, $\circ$ denotes the application of transformation, and $\Delta\hat{\boldsymbol{\mu}}_i^\tau$ is the location change of the Gaussian point from the previous iteration.

Spatial-coherence Preservation. Based solely on temporal-change preservation, the model tends to create more rigid parts to reduce approximation error, leading to over-segmentation. To prevent over-segmentation and enforce spatial smoothness, we preserve spatial coherence based on local neighborhood consistency. For each point $\boldsymbol{\mu}_i$, we search for H neighboring points using ball queries [28] and enforce their rigid-part assignments to be consistent:

$$\mathcal{L}_{\text{spatial}} = \frac{1}{NH}\sum_{i=1}^{N}\sum_{h=1}^{H}\delta(\boldsymbol{m}_i^\tau, \boldsymbol{m}_{i,h}^\tau), \tag{5.4}$$

where $\boldsymbol{m}_i^\tau \in (0,1)^S$ represents the soft assignment of the center point $\boldsymbol{\mu}_i$, $\boldsymbol{m}_{i,h}^\tau$ denotes the assignment of its h-th neighbor, and $\delta(\cdot)$ is the L2 distance. The weighted fusion $\mathcal{L}_{\text{identification}} = \lambda_{\text{temporal}}\mathcal{L}_{\text{temporal}} + \lambda_{\text{spatial}}\mathcal{L}_{\text{spatial}}$ serves as the self-supervised loss for learning $\mathcal{M}^\tau$, where $\mathcal{L}_{\text{temporal}}$ promotes sufficient quasi-rigid parts and $\mathcal{L}_{\text{spatial}}$ prevents excessive segmentation.

5.4.3.2 Motion Decomposition

After identifying quasi-rigid parts $\mathcal{M}^\tau$, we model the motion structure by decomposing the movements of each Gaussian from canonical space to the τ^{th} frame. The positional displacement $\Delta\boldsymbol{\mu}_i^\tau$ is decomposed into rigid motion weighted by part probabilities and additional non-rigid deformation:

$$\Delta\boldsymbol{\mu}_i^\tau = \sum_{s=1}^{S} m_{is}^\tau(\boldsymbol{R}_s^\tau\boldsymbol{\mu}_i + \boldsymbol{t}_s^\tau) + \boldsymbol{d}_i^\tau, \tag{5.5}$$

where the learnable parameters $\boldsymbol{R}_s^\tau$ and $\boldsymbol{t}_s^\tau$ represent the rotation and translation of the s-th quasi-rigid part, respectively; m_{is}^τ is the soft assignment probability, and $\boldsymbol{d}_i^\tau \in \mathbb{R}^3$ (also learnable) captures the non-rigid deformation specific to Gaussian i at time τ. This decomposition allows our model to capture both global rigid movements (*e.g.*, animals' body displacement) and fine-grained non-rigid deformations (*e.g.*, muscle bulging), providing a comprehensive representation of motion patterns.

5.4.3.3 Weighted Dense-to-Sparse Optimization

As noted above, the number of rigid motion parameters ($\boldsymbol{R}_s^\tau$ and $\boldsymbol{t}_s^\tau$ for S groups) is significantly smaller than the number of per-Gaussian non-rigid motion parameters ($\boldsymbol{d}_i^\tau$ for N groups). To balance rigid and non-rigid modeling while maintaining computational efficiency, we propose a weighted dense-to-sparse optimization

strategy that progressively refines the 4D representation. First, we introduce a learnable weight module $\mathcal{W}^\tau$ that outputs $w_i^\tau \in \mathbb{R}$ to adjust the proportion of rigid and non-rigid motion for each Gaussian point:

$$\Delta\boldsymbol{\mu}_i^\tau = w_i^\tau \sum_{k=1}^{S} m_{ik}^\tau (\boldsymbol{R}_s^\tau \boldsymbol{\mu}_i + \boldsymbol{t}_s^\tau) + (1 - w_i^\tau)\boldsymbol{d}_i^\tau, \tag{5.6}$$

Dense Per-Gaussian Optimization. In this stage, the time-aware change $\Delta\mathcal{G}_i^\tau$ of canonical Gaussians is directly established on Gaussian points. We initialize canonical Gaussians $\{\mathcal{G}_i\}_{i=1}^N$ with randomly sampled positions inside a sphere. For a specific timestep τ, we learn the canonical Gaussians' movement using our weighted rigid-nonrigid composition in Eq. 5.6, and compute their new position $\boldsymbol{\mu}_i^\tau = \boldsymbol{\mu}_i + \Delta\boldsymbol{\mu}_i^\tau$, as well as rotation quaternion $\boldsymbol{q}_i^\tau$ and scale $\boldsymbol{s}_i^\tau$. After obtaining the transformed Gaussians $\{\mathcal{G}_i^\tau\}_{i=1}^N$ at timestep τ, we project them from the reference view and augmented views to get the rendered image to compute $\mathcal{L}_{\text{recon}}$ and $\mathcal{L}_{\text{aug}}$, respectively (as defined in Sec. 5.4.3). The complete optimization objective in this stage is:

$$\mathcal{L}_{\text{dense}} = \lambda_{\text{recon}}\mathcal{L}_{\text{recon}} + \lambda_{\text{aug}}\mathcal{L}_{\text{aug}} + \lambda_{\text{identification}}\mathcal{L}_{\text{identification}}. \tag{5.7}$$

Sparse Control-point Optimization. Building upon the concept of sparse control points prevalent in Gaussian Splatting methods (*e.g.*, SC-GS [61], DreamMesh4D [66], SC-4D [81]), we extend this concept to model non-rigid changes based on rigid motion. We establish a set of K sparse control points $\{\mathcal{P}_k\}_{k=1}^K$. Each control point $\mathcal{P}_k^\tau$ at timestep τ is parameterized by its original location $\boldsymbol{p}_k$, point-wise displacement $\Delta\boldsymbol{p}_k^\tau$ and rotation $\hat{\boldsymbol{R}}_k^\tau$ (with rotation quaternion $\hat{\boldsymbol{q}}_k^\tau$). Similarly, the displacement $\Delta\boldsymbol{p}_k^\tau$ of each control point follows the same rigid-nonrigid composition principle as in Eq. 5.6. We use Linear Blend Skinning (LBS) to control the time-aware change of canonical Gaussians. For each canonical Gaussian $\mathcal{G}_i$,

the warped location $\boldsymbol{\mu}_i^\tau$ and rotation $\boldsymbol{q}_i^\tau$ can be computed as a weighted sum of its KNN control points M_i:

$$\boldsymbol{\mu}_i^\tau = \sum_{k \in M_i} \omega_{ik} (\hat{\boldsymbol{R}}_k^\tau (\boldsymbol{\mu}_i - \boldsymbol{p}_k) + \boldsymbol{p}_k + \Delta \boldsymbol{p}_k^\tau) \qquad \boldsymbol{q}_i^\tau = \left(\sum_{k \in M_i} \omega_{ik} \hat{\boldsymbol{q}}_k^\tau \right) \otimes \boldsymbol{q}_i, \quad (5.8)$$

where ω_{ik} is a weighting ratio depending on the distance d_{ik} between Gaussian $\mathcal{G}_i$ center and its neighboring control point $\boldsymbol{p}_k$, and a learned control radius γ_k (details in C.2); $\otimes$ denotes quaternion multiplication. The complete optimization objective combines all loss components:

$$\mathcal{L}_{\text{sparse}} = \lambda_{\text{recon}} \mathcal{L}_{\text{recon}} + \lambda_{\text{aug}} \mathcal{L}_{\text{aug}} + \lambda_{\text{identification}} \mathcal{L}_{\text{identification}} + \lambda_{\text{control}} \mathcal{L}_{\text{control}}, \quad (5.9)$$

where $\mathcal{L}_{\text{control}}$ denotes additional regularization terms for sparse control points that constrain their location distribution that maintains shape fidelity throughout optimization (details in C.2).

5.5 Experiments

5.5.1 Experimental Setup

Proposed Benchmark. Existing video-to-4D datasets (*e.g.*, Consistent4D [63]) suffer from (1) limited object displacement and (2) insufficient motion diversity, restricting comprehensive evaluation of motion modeling capabilities. To address these limitations, we curate a challenging benchmark with 3D CGI models sourced from ARTEMIS-DFA [226] crafted through artists' efforts. This dataset comprises 9 distinct animal species (*i.e.*, Beagle Dog, Bear, Cat, Duck, Fox, Lion, Panda, White Tiger, Wolf), each represented by 2-4 video sequences showcasing different motion patterns (*e.g.*, howling, running), resulting in a total of 25 sequences. We select one reference viewpoint for training and four for testing novel

view synthesis. For quantitative evaluation, we choose 32 frames at 512-pixel resolution with foreground-only objects to ensure compatibility with existing methods, while the subset of additional frames serves for qualitative assessment and validation. More information can be found in C.1.

Evaluation Protocol. Evaluation metrics encompass both pixel-level fidelity measures (PSNR, SSIM) computed against reference viewpoints, and perceptual quality metrics (LPIPS, FVD, FID-VID, CLIP-Score) evaluated across all viewpoints including both generated and reference views. Additionally, we conduct experiments on the Consistent4D dataset to demonstrate generalization to generic objects.

Implementation Details. Following established practices [79], we employ multilayer perceptrons (MLPs) to learn model parameters, feeding positional embeddings as inputs (6 dimensions for temporal encoding, 10 dimensions for spatial encoding). Our optimization strategy consists of 1,500 iterations: the first 1,000 iterations focus on dense per-Gaussian optimization, followed by 500 iterations of sparse control-point optimization. More implementation details are provided in C.1.

5.5.2 Hyper-setting Exploration

We conduct extensive hyperparameter analysis on our ARTEMIS-DFA validation set to investigate two critical parameters: the maximum number of sparse control points K, which governs non-rigid deformation capacity, and $\lambda_{\text{temporal}}$, which controls the weight of the temporal-change preservation loss term in our self-supervised motion structure discovery.

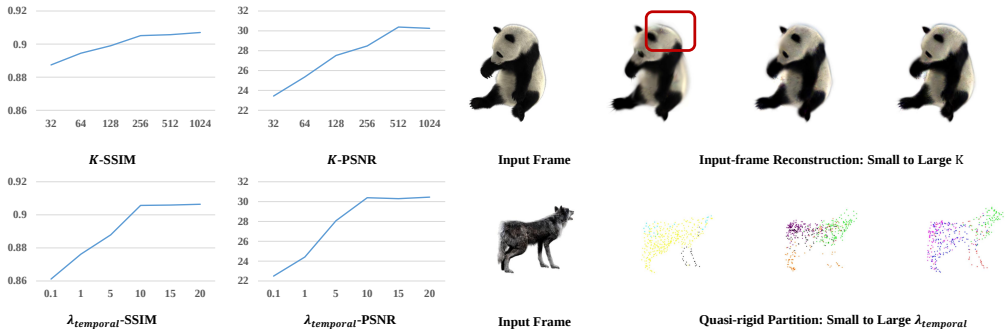


Figure 5.3: *Hyper-parameter exploration.* We evaluate key hyperparameters on the validation set using PSNR and SSIM metrics. The top row explores the maximum number of sparse control points K, where red boxes highlight distortion regions caused by insufficient control points. The bottom row demonstrates the influence of different $\lambda_{\text{temporal}}$ values, where too small values inadequately capture motion dynamics while too large values lead to over-segmentation.

Number of Sparse Control Points (K). We evaluate the effect of varying the maximum number of sparse control points while fixing the number of rigid parts to 8. As shown in the top row of Figure 5.3, insufficient control points (e.g., $K = 32$) result in noticeable distortion artifacts, particularly in regions requiring fine-grained non-rigid deformations, as highlighted by the red boxes. Performance metrics (PSNR and SSIM) demonstrate steady improvement as K increases, with diminishing returns observed beyond $K = 512$. This saturation point indicates that 512 control points provide sufficient degrees of freedom to capture the non-rigid motion details in our benchmark while maintaining computational efficiency.

Temporal Loss Weight ($\lambda_{\text{temporal}}$). With λ_{spatial} fixed at 0.1, we examine the impact of temporal preservation weight on motion structure discovery. The bottom row of Figure 5.3 illustrates the trade-off between motion completeness and segmentation quality. Small values (e.g., $\lambda_{\text{temporal}} = 1.0$) fail to adequately preserve temporal motion patterns, resulting in insufficient rigid-part identification. Conversely, excessively large values (e.g., $\lambda_{\text{temporal}} = 100.0$) lead to over-segmentation,

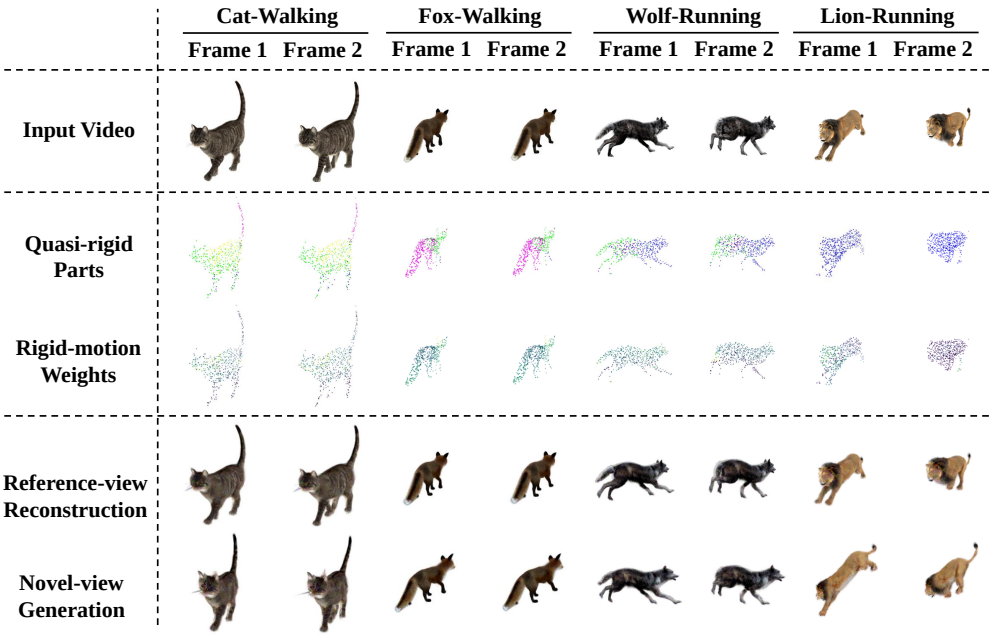


Figure 5.4: *Visualization samples of our method.*

where the model creates too many rigid parts to minimize temporal approximation error. Our analysis reveals that $\lambda_{\text{temporal}} = 10.0$ achieves optimal performance, balancing motion preservation with appropriate part segmentation.

Based on these explorations, we adopt the following hyperparameter configuration for all subsequent experiments: $\lambda_{\text{spatial}} = 0.1$, and $\lambda_{\text{temporal}} = 10.0$, utilizing up to 512 sparse control points and a maximum of 8 quasi-rigid parts. This constitutes a highly compact representation compared to per-Gaussian parameterization–canonical Gaussians typically have 50,000-80,000 points even after the pruning processes.

Table 5.1: *Quantitative comparison on ARTEMIS-DFA.*

	PSNR ↑	SSIM ↑	LPIPS ↓	FVD ↓	FID-VID ↓	CLIP ↑
Consistent4D [63]	24.73	0.918	0.157	1025.67	37.42	0.895
DreamGaussian4D [62]	28.52	0.929	0.168	1132.85	39.16	0.903
4DGen [221]	24.86	0.912	0.164	1087.33	71.58	0.881
STAG4D [65]	25.41	0.922	0.152	1189.72	44.93	0.887
SC-4D [81]	27.38	0.924	0.145	1052.19	38.84	0.898
DreamMesh4D [66]	29.12	0.922	0.138	812.53	22.18	0.916
MotionStruct4D	30.68	0.925	0.134	769.39	19.67	0.919

5.5.3 Qualitative & Quantitative Results

As shown in Tab. 5.1, experimental results on our ARTEMIS-DFA benchmark demonstrate that MotionStruct4D achieves superior performance across all evaluation metrics, with the highest PSNR (30.68) and CLIP score (0.919), and the lowest perceptual distortion metrics (LPIPS: 0.134, FVD: 769.39, FID-VID: 19.67). Our method consistently outperforms all baseline approaches, validating the effectiveness of our motion structure discovery and weighted dense-to-sparse optimization strategy. As demonstrated in Figure 5.4, the identified quasi-rigid parts provide meaningful interpretability: for instance, the cat's tail is segmented as an independent part since it exhibits distinct swaying motion while walking, contrasting with foxes and wolves where the tail moves cohesively with the posterior body and is thus grouped within the same quasi-rigid part.

Limitations. Despite the overall effectiveness of our approach, certain challenging scenarios expose limitations of our method. As illustrated in Figure 5.4 with the running lion example, when objects undergo substantial displacement combined with extensive deformation, our model encounters difficulty in identifying specific quasi-rigid parts. In such cases, the method tends to treat the entire body as a single rigid entity, with the overall body displacement captured as the primary rigid movement.

Table 5.2: *Ablation studies on validation data.*

Exp.	**Components**						**Metrics**					
Group	Temp. Cue	Spat. Cue	Rigid Mot.	Non-rigid Mot.	Dense-to-sparse Fusion	Weight Fusion	**PSNR ↑**	**SSIM ↑**	**LPIPS ↓**	**FVD ↓**	**FID-VID ↓**	**CLIP ↑**
(a)				✓			22.3	0.795	0.288	272	35.1	0.889
(b)		✓	✓	✓			25.3	0.841	0.213	195	28.2	0.918
(c)	✓	✓	✓	✓			28.5	0.875	0.171	173	24.7	0.941
(d)	✓	✓	✓	✓		✓	29.7	0.894	0.148	156	22.5	0.959
(e)	✓	✓	✓	✓	✓	✓	30.2	0.908	0.136	145	20.9	0.965

5.5.4 Ablation Studies

To validate the effectiveness of each component in our MotionStruct4D framework, we conduct comprehensive ablation studies on our curated benchmark dataset. Tab. 5.2 presents quantitative results across different component combinations.

(a) Baseline: Pure Non-rigid Motion. We begin with a baseline configuration that only employs non-rigid deformation without any motion structure discovery. This approach treats all Gaussian movements as independent per-point deformations, similar to conventional deformable 3D Gaussian methods. The results show reasonable performance (PSNR: 22.3, SSIM: 0.795) but significant limitations in handling structured motions, as evidenced by higher perceptual distortion (LPIPS: 0.288) and poor temporal consistency (FVD: 272). This baseline demonstrates the challenges of modeling complex object motion without incorporating structural priors.

(b) Adding Spatial Coherence Preservation. Incorporating spatial coherence preservation while introducing rigid motion modeling leads to substantial improvements across all metrics. The spatial cue $\mathcal{L}_{\text{spatial}}$ ensures neighboring Gaussian points maintain consistent rigid-part assignments. The reduction in FVD (272→195) and FID-VID (35.1→28.2) indicates better temporal consistency and overall motion quality.

(c) Incorporating Temporal Change Preservation. The addition of temporal cues $\mathcal{L}_{\text{temporal}}$ represents a critical enhancement in motion structure discovery. This combination enables the model to effectively identify quasi-rigid parts by preserving both temporal dynamics and spatial relationships. Performance improvements are substantial: PSNR increases to 28.5, SSIM to 0.875, while LPIPS decreases

significantly to 0.171. The further reduction in FVD (195→173) demonstrates superior temporal consistency.

(d) Weighted Fusion of Rigid-Nonrigid Motions. Introducing the learnable weight module $\mathcal{W}^{\tau}$ (Eq. 5.6) that adaptively balances rigid and non-rigid motion contributions yields consistent improvements across all metrics. The weighted fusion mechanism allows different regions to emphasize appropriate motion types—rigid parts benefit from structured transformations while detailed areas leverage non-rigid deformations. This results in PSNR reaching 29.7, SSIM improving to 0.894, and further reduction in perceptual metrics (LPIPS: 0.148, FVD: 156).

(e) Dense-to-Sparse Optimization Strategy. The complete framework incorporating dense-to-sparse optimization achieves the best performance across all metrics. The transition from dense per-Gaussian deformation to sparse control points significantly reduces computational complexity while maintaining expressiveness. Final results show PSNR of 30.2, SSIM of 0.908, and the lowest perceptual distortions (LPIPS: 0.136, FVD: 145). This configuration demonstrates that our hierarchical optimization strategy effectively integrates motion structure discovery with efficient sparse control, resulting in superior motion fidelity and novel view synthesis quality.

5.6 Conclusion

Video Demonstrations. The nature of 4D generation—rendering dynamic 3D content across arbitrary viewpoints and timestamps—demands visual evaluation beyond what static figures can convey. We therefore provide comprehensive video

demonstrations on the project webpage.[1] The webpage showcases: (1) 4D generation results rendered from multiple novel viewpoints (*e.g.*, $15°$, $-75°$, $105°$, $195°$), demonstrating the multi-view consistency of our reconstructed scenes; and (2) the discovered quasi-rigid part segmentations at each viewpoint, illustrating how MotionStruct4D disentangles global rigid movements from local non-rigid deformations. These video results complement the quantitative metrics reported in the preceding sections and provide direct visual evidence of our method's ability to generate temporally coherent, spatially consistent 4D content from single-view monocular video.

In this work, we present MotionStruct4D, a novel approach for video-to-4D generation that addresses the fundamental challenge of inferring plausible 3D geometry and motion from single-view monocular videos. Our main contributions include: (1) a self-supervised motion structure discovery module that identifies quasi-rigid parts by preserving spatio-temporal relationships without explicit 3D supervision; (2) a weighted dense-to-sparse optimization architecture that balances rigid and non-rigid motions through adaptive fusion; and (3) a comprehensive benchmark dataset featuring substantial object displacement and diverse articulated motion patterns. Extensive experimental validation demonstrates MotionStruct4D's superior performance across multiple metrics. The explicit modeling of motion structure not only enhances generation quality but also provides interpretable insights into object dynamics, making our approach particularly valuable for applications requiring motion understanding.

Together with the semantic classification framework of Chapter 3 and the motion analysis method of Chapter 4, MotionStruct4D completes the multi-level representation programme outlined in Chapter 1. Across the three contribution

[1] https://jx-zhong-for-academic-purpose.github.io/MotionStruct4DPage/

chapters, a common thread emerges: each method exploits domain-specific structural priors—kinematic surfaces, SE(3) equivariance, and rigid–non-rigid motion decomposition—to achieve strong performance under practical constraints of computational cost, annotation availability, or data scarcity. Chapter 6 synthesises these collective insights, examines the relationships and limitations of the three contributions, and charts directions for future research.

6 | Conclusion and Future Directions

This thesis began with the assertion that the cornerstone of enhanced machine autonomy lies in the capacity to comprehend and interact with the three-dimensional world. We identified a critical gap: while deep learning has achieved remarkable success in static point cloud processing, the dynamic nature of real-world environments—encompassing both spatial and temporal dimensions—presents formidable challenges that static models cannot address. This observation motivated our central research question: **How can we develop effective deep representations for dynamic point clouds that address fundamental challenges at different levels of abstraction?**

Through systematic investigation across three representational levels—high-level semantic understanding, intermediate-level motion analysis, and low-level geometric reconstruction—we have demonstrated that this question can be answered through a collection of targeted techniques, each addressing the primary challenge at its respective level while maintaining the cross-cutting requirement for high-quality outputs. This chapter reflects on our contributions, examines their collective impact, and charts directions for future research.

6.1 Summary of Contributions

Our research has resulted in three distinct yet complementary contributions that directly address the fundamental sub-questions posed in Chapter 1.2. Each technique operates at a different level of abstraction, tackling its specific primary challenge while maintaining high-quality standards.

6.1.1 Addressing High-level Semantic Understanding

At the highest level of abstraction, we confronted the challenge of **low-cost deployment** through Kinet (Chapter 3), a kinematics-inspired neural network that seamlessly extends static architectures to dynamic 3D video classification. By generalizing space-time surfaces to the feature space and unrolling the ST-normal solver differentiably, Kinet achieves state-of-the-art accuracy (93.27% on MSRAction-3D) with minimal computational overhead (3.20M parameters, 10.35G FLOPs).

This contribution directly answers our first sub-question: machines can understand semantic content in dynamic scenes by efficiently aggregating spatio-temporal information without explicit point-wise correspondence tracking. The lightweight design ensures practical deployability in real-time applications where immediate action recognition is required.

6.1.2 Achieving Intermediate-level Motion Analysis

The intermediate level presented the challenge of **annotation-free generalization**, addressed through our unsupervised part-level SE(3)-equivariant framework (Chapter 4). By exploiting geometric equivariance properties and developing self-supervised training strategies, our method achieves robust rigid segmentation (63.8% AP on SAPIEN) and motion estimation without manual labels.

This addresses our second sub-question: motion patterns can be extracted from dynamic point clouds by leveraging the inherent structure in rigid transformations and the interrelationships among scene flow, segmentation masks, and transformations. The lightweight architecture (0.25M parameters, 0.92G FLOPs) ensures efficient deployment while the geometric priors enable generalization to unseen object categories.

6.1.3 Enabling Low-level 4D Reconstruction

At the foundational level, we tackled the challenge of **data efficiency** with MotionStruct4D (Chapter 5), enabling video-to-4D generation from single-view monocular inputs. Through self-supervised motion structure discovery and weighted dense-to-sparse optimization, our method reconstructs complete 4D representations while working with minimal observational data.

This answers our third sub-question: complete 3D dynamics can be captured by discovering and exploiting motion structures within the data, decomposing movements into rigid and non-rigid components, and efficiently balancing these through adaptive optimization. The superior performance on ARTEMIS-DFA (PSNR: 30.68, FVD: 769.39) demonstrates that data-efficient 4D reconstruction is achievable even from severely limited inputs.

6.2 Synthesis and Collective Impact

6.2.1 Addressing the Overarching Research Question

Returning to our overarching research question, this thesis shows that effective deep representations for dynamic point clouds can arise from addressing distinct aspects of the problem with targeted solutions. Rather than pursuing a monolithic solution, we **address three different facets** of dynamic point cloud understanding, each with clear goals and distinct sources of difficulty. At each level, we **target the primary practical bottleneck**—low-cost deployment at the semantic level, annotation-free generalization at the motion level, and data efficiency at the reconstruction level—so that advances translate directly into deployable capability. Throughout, we **maintain output quality** as a non-negotiable requirement, ensuring that each technique can stand alone as an independent contribution. Cru-

cially, progress at all three levels is enabled by **exploiting inherent structure in the data**: kinematic relationships guide semantic aggregation, geometric equivariance constrains part-level motion, and motion decomposition organizes 4D reconstruction. Taken together, these choices demonstrate a coherent research programme for building accurate and efficient representations of dynamic 3D environments, even though the individual techniques are not architecturally coupled.

6.2.2 Relationship Among Contributions

It is important to be explicit about how the three contributions relate in practice. While they are organized along a spectrum from semantic to geometric representations, they were developed independently and do not form a tightly coupled pipeline. Each method has its own input format (point cloud sequences for Kinet, discrete frame pairs for the SE(3) framework, monocular video for MotionStruct4D), its own training paradigm (supervised classification, unsupervised self-training, optimization-based generation), and its own evaluation criteria. The methods do not share learned representations, and the output of one is not used as input to another in any experiment presented in this thesis.

Whether direct integration of these methods is feasible remains an open question. In principle, one could envision a system where Kinet provides high-level activity labels that inform the motion analysis module about expected motion patterns, or where the rigid-part segmentation from Chapter 4 provides structural priors for the 4D reconstruction in Chapter 5. However, several practical barriers prevent such unification at present: (1) the methods operate on different data modalities and at different temporal granularities; (2) the training procedures are fundamentally different (end-to-end supervised learning *vs*. self-supervised optimization); and (3) the computational requirements of a combined system would be substantially larger than any individual component, undermining the efficiency advantages that

each method offers in isolation. Among possible partial integrations, the most natural starting point would be feeding the rigid-part segmentation of Chapter 4 into MotionStruct4D's motion decomposition, since both methods share the assumption of piecewise-rigid dynamics and operate on compatible geometric representations; even this limited coupling, however, would require reconciling their different supervision signals and temporal granularities. We view the broader exploration of such integration as a promising but non-trivial direction for future research, as discussed in Section 6.3.2.

6.2.3 Theoretical and Practical Implications

6.2.3.1 Theoretical Implications

Our results suggest that complex perceptual problems benefit from **multi-level formulations** that respect natural abstraction boundaries: semantics, part-level motion, and geometry each admit different inductive biases and learning signals, and treating them separately but compatibly improves stability and sample efficiency. We further show that **domain-specific priors**—kinematics for temporal aggregation, equivariance for motion analysis, and motion structure for reconstruction—can be embedded in modern architectures without sacrificing flexibility, yielding better generalization from limited supervision. Finally, the observed performance of our intermediate module indicates that **self-supervised learning under geometric and physical constraints** is a promising route to high-quality models when annotations are scarce or unavailable.

6.2.3.2 Practical Implications

From an application standpoint, the proposed methods are designed with **computational efficiency** in mind, enabling deployment on resource-constrained plat-

forms where latency and power budgets are tight. The **annotation-free** nature of the motion analysis module lowers the cost of entering new domains, since part-level motion can be learned without curated labels. Moreover, the ability to reconstruct **complete 4D representations from single-view inputs** broadens the reach of dynamic scene understanding to settings where multi-view capture is impractical—such as mobile robotics, in-the-wild AR, and low-infrastructure industrial inspection—thereby expanding the impact of the overall framework.

6.3 Limitations and Future Research Directions

While our contributions advance the state of the art in dynamic point cloud processing, several open issues delimit the current scope and point to promising directions.

6.3.1 Current Limitations

Temporal scope. Our models are validated primarily on short to medium sequences (e.g., 16–32 frames for Kinet and 2–4 frames for the SE(3)-equivariant framework). Such windows are sufficient for local motion cues and short actions, but they under-represent long-range dependencies, delayed causal effects, and activity phases that unfold over minutes. Extending temporal coverage without incurring prohibitive memory and compute costs remains a central challenge.

Motion complexity. The intermediate-level method is tailored to piecewise-rigid dynamics and excels when object parts admit SE(3) motions. However, highly deformable phenomena—cloth, soft tissues, or fluids—violate rigid assumptions and demand representations that can express continuous non-rigid fields while retaining stability and physical plausibility. Bridging from rigid parts to smoothly varying deformation spaces is non-trivial and calls for stronger priors and regular-

ization.

Scene complexity. Most experiments target single-object or lightly interactive scenes. In densely populated environments with many agents and contact events, occlusions, multi-body constraints, and long-range interactions conspire to degrade correspondence and segmentation quality. Scaling gracefully to crowded, cluttered scenes will require stronger relational reasoning and improved handling of occlusion and re-identification.

Real-time performance. Although each component is computationally efficient in isolation, achieving end-to-end real-time performance across all three levels on embedded hardware is not yet guaranteed. Meeting tight latency and energy budgets while preserving accuracy will likely require algorithm–system co-design, model compression, and hardware-aware scheduling.

6.3.2 Future Research Directions

The landscape of dynamic 3D perception is evolving rapidly, driven by advances in foundation models, world models, and large-scale generative systems. Below we discuss how these trends, together with immediate methodological extensions, create opportunities for extending the methods presented in this thesis.

6.3.2.1 Immediate Extensions

A first avenue is a **unified multi-level architecture** that couples high-level semantics, part-level motion, and low-level reconstruction within a single optimization or training loop. Such integration would allow cross-level constraints—e.g., semantic priors guiding motion grouping, or reconstructed geometry feeding back into recognition—to resolve ambiguities that are difficult for any single level alone. As discussed in the preceding section, this remains non-trivial due to dif-

ferences in data modalities, training paradigms, and computational requirements, but represents a compelling long-term goal.

A second priority is **long-horizon temporal modeling**. Hierarchical memories, recurrent implicit states, or efficient attention schemes over sparsified keyframes could capture dependencies over thousands of frames while limiting quadratic costs.

Finally, extending the intermediate representation toward **non-rigid motion** suggests learning low-dimensional deformation manifolds or implicit deformation fields that remain compatible with equivariance where possible, thus preserving the benefits of self-supervision while broadening applicability to soft and articulated objects with complex elasticity.

6.3.2.2 Scaling to Large Datasets and Integration with Foundation Models

The rapid progress in 3D foundation models (*e.g.*, Point-BERT [107], Point-MAE [108], Uni3D [109], PointGPT [110]) and large-scale 3D datasets (*e.g.*, Objaverse [120]) opens significant opportunities for the methods presented in this thesis. Kinet's design as a lightweight temporal adapter makes it naturally suited for integration with pre-trained 3D foundation models: one could replace the static backbone with a foundation model and fine-tune only the kinematic learning units, potentially achieving stronger performance with minimal additional training. Similarly, the SE(3)-equivariant framework could benefit from foundation model features as initialization, reducing the reliance on self-supervised training signals.

For MotionStruct4D, the emergence of large-scale video generation models and multi-view diffusion models (*e.g.*, SV3D, SV4D [64]) suggests that stronger generative priors could replace or augment the SDS-based supervision currently used,

potentially improving both quality and training efficiency. The integration of explicit motion structure awareness (as in our approach) with the implicit knowledge captured by large generative models represents a particularly promising research direction.

6.3.2.3 World Models and Dynamic 3D Perception

World models that learn predictive representations of environment dynamics (*e.g.*, GAIA-1 [116], OccWorld [117]) represent an emerging paradigm for dynamic 3D understanding. These models learn to predict future states of 3D environments, effectively capturing scene dynamics at a holistic level. Our methods could contribute to this paradigm in several ways: Kinet's efficient temporal encoding could serve as a component within world model architectures; the rigid-part decomposition from Chapter 4 could provide structured state representations for world model prediction; and MotionStruct4D's motion decomposition could inform the design of structured latent spaces for world models that separate rigid and non-rigid dynamics.

Such integration is not straightforward, however. Existing world models typically operate on dense, regular representations (voxel grids, BEV maps, or pixel-aligned features) and are trained on large-scale driving or embodied-agent datasets, whereas our methods consume sparse, irregular point clouds and are developed on smaller-scale benchmarks. Bridging this representational gap—for example, through hybrid sparse-dense encoders or point-to-voxel projection layers—would be a prerequisite for practical coupling. Nevertheless, we believe the explicit structural and motion priors our methods provide could complement the learned implicit dynamics of world models, potentially improving their sample efficiency and physical plausibility.

6.3.2.4 Geometry Extraction in Generative Systems

As 3D and 4D generative systems become more capable, extracting reliable geometric information from generated content becomes increasingly important for downstream applications in robotics, simulation, and augmented reality. Current generative approaches often produce visually plausible but geometrically inconsistent outputs—thin structures collapse, genus changes appear across frames, and metric scale drifts during motion. The motion structure awareness introduced in MotionStruct4D addresses this partially by enforcing rigid-body constraints, but our experiments also reveal remaining failure modes: topology-changing deformations (e.g., a hand opening or cloth tearing) can violate the piecewise-rigid prior, and single-view inputs leave depth ambiguities that propagate into reconstructed geometry. Future work could integrate differentiable physics simulators or topology-preserving regularizers into the generation pipeline to enforce physical plausibility more broadly. The development of geometry-aware evaluation metrics—beyond visual quality scores—and physics-informed generation objectives represents an important direction for making generative 4D content practically useful.

6.3.2.5 Broader Research Agenda

Beyond short-term engineering, we envision **foundation models for dynamic point clouds** trained at scale with multi-task objectives and self-supervised signals, providing strong universal priors for downstream robotics and perception tasks. Complementarily, **active perception**—controlling viewpoints or manipulating the scene to disambiguate hypotheses—can close the loop between sensing and action, particularly in robotic settings where agency is available. Richer **cross-modal learning** that fuses point clouds with RGB, audio, language, and proprioception promises more complete scene understanding and more natural hu-

man–robot interaction. Finally, **physics-informed learning** that embeds differentiable simulators or explicit constraints can improve generalization and guarantee physically plausible predictions, especially for motion forecasting and planning.

6.3.2.6 Societal Considerations

As dynamic 3D understanding matures, its societal footprint must be considered in parallel with technical advances. **Privacy and surveillance** concerns arise when human activities are inferred at scale; privacy-preserving representations, on-device processing, and auditable data governance are essential safeguards. Equally important is **accessibility**: focusing on computational and data efficiency can lower the barrier to adoption and ensure benefits extend to cost-sensitive domains such as assistive technologies and public-interest robotics. Finally, **safety and reliability** are paramount for deployment in safety-critical applications—from autonomous driving to surgical robotics—where calibrated uncertainty, robust out-of-distribution detection, and formal verification of control loops can mitigate risk.

6.4 Concluding Remarks

This thesis embarked on a journey to bridge the gap between static point cloud processing and the dynamic reality of our three-dimensional world. Through systematic decomposition of the problem into three levels of abstraction and targeted solutions for each level's primary challenge, we have demonstrated that effective deep representations for dynamic point clouds are not only possible but can be achieved with practical efficiency.

The progression from recognizing the limitations of static models in Chapter 1 to developing three complementary techniques represents a broad exploration of

dynamic point cloud understanding across different levels of abstraction. While these contributions are thematically connected, they are technically independent and should be understood as addressing distinct facets of a multifaceted problem rather than as layers of a single integrated system.

As we stand at the intersection of continued sensor proliferation and advancing AI capabilities, the ability to understand dynamic 3D environments becomes increasingly critical. The techniques presented in this thesis—Kinet for efficient semantic understanding, SE(3)-equivariant framework for annotation-free motion analysis, and MotionStruct4D for data-efficient 4D reconstruction—provide practical tools for this understanding while respecting real-world constraints of computational resources, annotation availability, and data scarcity.

Looking ahead, the foundations laid by this work open numerous avenues for future exploration. The success of our approach—addressing distinct challenges with targeted, efficient solutions—suggests that similar problem decompositions might benefit other complex perceptual problems. The effectiveness of incorporating domain-specific priors (kinematics, equivariance, motion structure) validates the continued importance of integrating physical understanding into learning systems. Most importantly, the demonstrated ability to achieve high-quality results with practical efficiency brings us closer to the ultimate goal stated in our opening: enhancing machine autonomy through comprehensive understanding of our dynamic three-dimensional world.

The journey from static snapshots to dynamic understanding represents not merely a technical evolution but a fundamental leap toward machines that can truly perceive, comprehend, and interact with the world as it actually exists—constantly changing, richly structured, and full of motion. Through the contributions of this thesis, we have taken significant steps toward realizing this vision, providing both

theoretical insights and practical solutions that advance the frontier of dynamic point cloud processing.

Bibliography

[1] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. PointNet: Deep learning on point sets for 3D classification and segmentation. In *CVPR*, 2017.

[2] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. PointCNN: Convolution on X-transformed points. In *NeurIPS*, 2018.

[3] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3D model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[4] Kaichun Mo, Shilin Zhu, Angel X Chang, Li Yi, Subarna Tripathi, Leonidas J Guibas, and Hao Su. Partnet: A large-scale benchmark for fine-grained and hierarchical part-level 3D object understanding. In *CVPR*, 2019.

[5] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuScenes: A multimodal dataset for autonomous driving. In *CVPR*, 2020.

[6] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *CVPR*, 2019.

[7] Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. Frustum pointnets for 3D object detection from RGB-D data. In *CVPR*, 2018.

[8] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3D object proposal generation and detection from point cloud. In *CVPR*, 2019.

[9] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. ScanNet: Richly-annotated 3D reconstructions of indoor scenes. In *ICCV*, 2017.

[10] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jurgen Gall. SemanticKITTI: A dataset for semantic scene understanding of LiDAR sequences. In *ICCV*, 2019.

[11] Qingyong Hu, Bo Yang, Sheikh Khalid, Wen Xiao, Niki Trigoni, and Andrew Markham. Towards semantic segmentation of urban-scale 3D point clouds: A dataset, benchmarks and challenges. In *CVPR*, 2021.

[12] Qingyong Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, Niki Trigoni, and Andrew Markham. RandLA-Net: Efficient semantic segmentation of large-scale point clouds. In *CVPR*, 2020.

[13] Bo Yang, Jianan Wang, Ronald Clark, Qingyong Hu, Sen Wang, Andrew Markham, and Niki Trigoni. Learning object bounding boxes for 3D instance segmentation on point clouds. In *NeurIPS*, 2019.

[14] Xingyu Liu, Mengyuan Yan, and Jeannette Bohg. Meteornet: Deep learning on dynamic 3d point cloud sequences. In *ICCV*, 2019.

[15] Hehe Fan, Xin Yu, Yuhang Ding, Yi Yang, and Mohan Kankanhalli. Pstnet: point spatio-temporal convolution on point cloud sequences. In *ICLR*. ICLR, ICLR, 2021.

[16] Hehe Fan, Yi Yang, and Mohan Kankanhalli. Point 4D transformer networks for spatio-temporal modeling in point cloud videos. In *CVPR*, 2021.

[17] Yuecong Min, Yanxiao Zhang, Xiujuan Chai, and Xilin Chen. An efficient PointLSTM for point clouds based gesture recognition. In *CVPR*, 2020.

[18] Hehe Fan, Yi Yang, and Mohan Kankanhalli. Point spatio-temporal transformer networks for point cloud video modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(2):2181–2192, 2022.

[19] Hehe Fan and Yi Yang. PointRNN: Point recurrent neural network for moving point cloud processing. *arXiv preprint arXiv:1910.08287*, 2019.

[20] Davis Rempe, Tolga Birdal, Yongheng Zhao, Zan Gojcic, Srinath Sridhar, and Leonidas J. Guibas. Caspr: Learning canonical spatiotemporal point cloud representations. In *NeurIPS*, 2020.

[21] Davis Rempe, Srinath Sridhar, He Wang, and Leonidas J. Guibas. Predicting the physical dynamics of unseen 3D objects. In *WACV*, 2020.

[22] Rui Huang, Wanyue Zhang, Abhijit Kundu, Caroline Pantofaru, David A. Ross, Thomas Funkhouser, and Alireza Fathi. An lstm approach to temporal 3d object detection in lidar point clouds. In *ECCV*, 2020. ISBN 978-3-030-58523-5.

[23] Xinshuo Weng, Jianren Wang, Sergey Levine, Kris Kitani, and Nick Rhinehart. Inverting the Pose Forecasting Pipeline with SPF2: Sequential Pointcloud Forecasting for Sequential Pose Forecasting. In *CoRL*, 2020.

[24] Himangi Mittal, Brian Okorn, and David Held. Just go with the flow: Self-supervised scene flow estimation. In *CVPR*, 2020.

[25] Xingyu Liu, Charles R Qi, and Leonidas J Guibas. FlowNet3D: Learning scene flow in 3D point clouds. In *CVPR*, pages 529–537, 2019.

[26] Xiuye Gu, Yijie Wang, Chongruo Wu, Yong Jae Lee, and Panqu Wang.

HPLFlowNet: Hierarchical permutohedral lattice flownet for scene flow estimation on large-scale point clouds. In *CVPR*, 2019.

[27] Yulan Guo, Hanyun Wang, Qingyong Hu, Hao Liu, Li Liu, and Mohammed Bennamoun. Deep learning for 3D point clouds: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[28] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. PointNet++: Deep hierarchical feature learning on point sets in a metric space. In *NeurIPS*, 2017.

[29] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. KPConv: Flexible and deformable convolution for point clouds. In *ICCV*, 2019.

[30] Qingyong Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, Niki Trigoni, and Andrew Markham. Learning semantic segmentation of large-scale point clouds with random sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[31] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics*, 2019.

[32] Hehe Fan, Yi Yang, and Mohan Kankanhalli. Point 4d transformer networks for spatio-temporal modeling in point cloud videos. In *CVPR*, pages 14204–14213, 2021.

[33] Jia-Xing Zhong, Kaichen Zhou, Qingyong Hu, Bing Wang, Niki Trigoni, and Andrew Markham. No pain, big gain: classify dynamic point cloud sequences with static models by fitting feature-level space-time surfaces. In *CVPR*, pages 8510–8520, 2022.

[34] Boyan Jiang, Yinda Zhang, Xingkui Wei, Xiangyang Xue, and Yanwei Fu. Learning compositional representation for 4d captures with neural ode. In *CVPR*, 2021.

[35] Wenxuan Wu, Zhi Yuan Wang, Zhuwen Li, Wei Liu, and Li Fuxin. Pointpwc-net: Cost volume on point clouds for (self-) supervised scene flow estimation. In *ECCV*, 2020.

[36] Gilles Puy, Alexandre Boulch, and Renaud Marlet. FLOT: Scene Flow on Point Clouds Guided by Optimal Transport. In *ECCV*, pages 527–544. Springer, 2020.

[37] Lihe Ding, Shaocong Dong, Tingfa Xu, Xinli Xu, Jie Wang, and Jianan Li. Fh-net: A fast hierarchical network for scene flow estimation on real-world point clouds. In *ECCV*, pages 213–229. Springer, 2022.

[38] Ke Wang and Shaojie Shen. Estimation and propagation: Scene flow prediction on occluded point clouds. *IEEE Robotics and Automation Letters*, 7(4):12201–12208, 2022.

[39] Jiahui Huang, He Wang, Tolga Birdal, Minhyuk Sung, Federica Arrigoni, Shi-Min Hu, and Leonidas Guibas. Multibodysync: Multi-body segmentation and motion estimation via 3d scan synchronization. In *CVPR*, 2021.

[40] Ziyang Song and Bo Yang. Ogc: Unsupervised 3d object segmentation from rigid dynamics of point clouds. In *NeurIPS*, 2022.

[41] Yancheng Wang, Yang Xiao, Fu Xiong, Wenxiang Jiang, Zhiguo Cao, Joey Tianyi Zhou, and Junsong Yuan. 3dv: 3d dynamic voxel for action recognition in depth video. In *CVPR*, 2020.

[42] Yimin Wei, Hao Liu, Tingting Xie, Qiuhong Ke, and Yulan Guo. Spatial-temporal transformer for 3d point cloud sequences. In *WACV*, 2021.

[43] Silvio Giancola, Jesus Zarzar, and Bernard Ghanem. Leveraging shape completion for 3d siamese tracking. In *CVPR*, 2019.

[44] Haozhe Qi, Chen Feng, Zhiguo Cao, Feng Zhao, and Yang Xiao. P2b: Point-to-box network for 3d object tracking in point clouds. In *CVPR*, 2020.

[45] Junbo Yin, Jianbing Shen, Chenye Guan, Dingfu Zhou, and Ruigang Yang. Lidar-based online 3d video object detection with graph-based message passing and spatiotemporal transformer attention. In *CVPR*, 2020.

[46] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3d object detection and tracking. In *CVPR*, 2021.

[47] Charles R. Qi, Yin Zhou, Mahyar Najibi, Pei Sun, Khoa Vo, Boyang Deng, and Dragomir Anguelov. Offboard 3d object detection from point cloud sequences. In *CVPR*, 2021.

[48] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *CVPR*, 2019.

[49] Hanwen Cao, Yongyi Lu, Cewu Lu, Bo Pang, Gongshen Liu, and Alan Yuille. Asap-net: Attention and structure aware point cloud sequence segmentation. In *BMVC*, 2020.

[50] Fabian Duerr, Mario Pfaller, Hendrik Weigel, and Jürgen Beyerer. Lidar-based recurrent 3d semantic segmentation with temporal memory alignment. In *3DV*, 2020.

[51] Li Yi, Haibin Huang, Difan Liu, Evangelos Kalogerakis, Hao Su, and Leonidas Guibas. Deep part induction from articulated object pairs. *ACM Transactions on Graphics*, 37(6), 2019.

[52] Stefan Andreas Baur, David Josef Emmerichs, Frank Moosmann, Peter Pinggera, Björn Ommer, and Andreas Geiger. Slim: Self-supervised lidar scene flow and motion segmentation. In *ICCV*, pages 13126–13136, 2021.

[53] Zan Gojcic, Or Litany, Andreas Wieser, Leonidas J Guibas, and Tolga Birdal. Weakly supervised learning of rigid 3d scene flow. In *CVPR*, 2021.

[54] Chaoyun Zhang, Marco Fiore, Iain Murray, and Paul Patras. Cloudlstm: A recurrent neural model for spatiotemporal point-cloud stream forecasting. In *AAAI*, 2020.

[55] David Deng and Avideh Zakhor. Temporal lidar frame prediction for autonomous driving. In *3DV*, 2020.

[56] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Occupancy flow: 4d reconstruction by learning particle dynamics. In *ICCV*, 2019.

[57] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *CVPR*, pages 10318–10327, 2021.

[58] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *ICCV*, pages 5865–5874, 2021.

[59] Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. In *3DV*, 2024.

[60] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splat-

ting for real-time dynamic scene rendering. *CVPR*, pages 20310–20320, 2023. URL https://api.semanticscholar.org/CorpusID:263908793.

[61] Yi-Hua Huang, Yang-Tian Sun, Ziyi Yang, Xiaoyang Lyu, Yan-Pei Cao, and Xiaojuan Qi. Sc-gs: Sparse-controlled gaussian splatting for editable dynamic scenes. In *CVPR*, pages 4220–4230, 2024.

[62] Jiawei Ren, Liang Pan, Jiaxiang Tang, Chi Zhang, Ang Cao, Gang Zeng, and Ziwei Liu. Dreamgaussian4d: Generative 4d gaussian splatting. *arXiv preprint arXiv:2312.17142*, 2023.

[63] Yanqin Jiang, Li Zhang, Jin Gao, Weimin Hu, and Yao Yao. Consistent4d: Consistent 360 {\deg} dynamic object generation from monocular video. *arXiv preprint arXiv:2311.02848*, 2023.

[64] Zeyu Yang, Zijie Pan, Chun Gu, and Li Zhang. Diffusionˆ2: Dynamic 3d content generation via score composition of video and multi-view diffusion models. *arXiv preprint arXiv:2404.02148*, 2024.

[65] Yifei Zeng, Yanqin Jiang, Siyu Zhu, Yuanxun Lu, Youtian Lin, Hao Zhu, Weiming Hu, Xun Cao, and Yao Yao. Stag4d: Spatial-temporal anchored generative 4d gaussians. In *ECCV*, pages 163–179. Springer, 2024.

[66] Zhiqi Li, Yiming Chen, and Peidong Liu. Dreammesh4d: Video-to-4d generation with sparse-controlled gaussian-mesh hybrid representation. *NeurIPS*, 37:21377–21400, 2024.

[67] Jiawei Ren, Cheng Xie, Ashkan Mirzaei, Karsten Kreis, Ziwei Liu, Antonio Torralba, Sanja Fidler, Seung Wook Kim, Huan Ling, et al. L4gm: Large 4d gaussian reconstruction model. *NeurIPS*, 37:56828–56858, 2024.

[68] Zijie Pan, Zeyu Yang, Xiatian Zhu, and Li Zhang. Efficient4d: Fast

dynamic 3d object generation from a single-view video. *arXiv preprint arXiv:2401.08742*, 2024.

[69] Daniel Maturana and Sebastian Scherer. Voxnet: A 3D convolutional neural network for real-time object recognition. In *IROS*, 2015.

[70] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3D object detection. In *CVPR*, 2018.

[71] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. Octnet: Learning deep 3D representations at high resolutions. In *CVPR*, 2017.

[72] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4D spatio-temporal convnets: Minkowski convolutional neural networks. In *ICCV*, 2019.

[73] Benjamin Graham, Martin Engelcke, and Laurens van der Maaten. 3D semantic segmentation with submanifold sparse convolutional networks. In *ICCV*, 2018.

[74] Jun Gao, Tianchang Shen, Zian Wang, Wenzheng Chen, Kangxue Yin, Daiqing Li, Or Litany, Zan Gojcic, and Sanja Fidler. Get3d: A generative model of high quality 3d textured shapes learned from images. *NeurIPS*, 35:31841–31854, 2022.

[75] Dario Pavllo, Jonas Kohler, Thomas Hofmann, and Aurelien Lucchi. Learning generative models of textured 3d meshes from real-world images. In *ICCV*, pages 13879–13889, 2021.

[76] Shengyu Huang, Zan Gojcic, Jiahui Huang, Andreas Wieser, and Konrad Schindler. Dynamic 3d scene analysis by point cloud accumulation. In *ECCV*, pages 674–690. Springer, 2022.

[77] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

[78] Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhöfer, Christoph Lassner, and Christian Theobalt. Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a dynamic scene from monocular video. In *ICCV*, pages 12959–12970, 2021.

[79] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), July 2023. URL https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/.

[80] Zeyu Yang, Hongye Yang, Zijie Pan, and Li Zhang. Real-time photorealistic dynamic scene representation and rendering with 4d gaussian splatting. In *ICLR*, 2024.

[81] Zijie Wu, Chaohui Yu, Yanqin Jiang, Chenjie Cao, Fan Wang, and Xiang Bai. Sc4d: Sparse-controlled video-to-4d generation and motion transfer. In *ECCV*, pages 361–379. Springer, 2024.

[82] Yuecong Min, Xiujuan Chai, Lei Zhao, and Xilin Chen. Flickernet: Adaptive 3D gesture recognition from sparse point clouds. In *BMVC*, 2019.

[83] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *ICCV*, 2015.

[84] Zhaofan Qiu, Ting Yao, and Tao Mei. Learning spatio-temporal representation with pseudo-3d residual networks. In *ICCV*, 2017.

[85] Guangming Wang, Hanwen Liu, Muyao Chen, Yehui Yang, Zhe Liu, and Hesheng Wang. Anchor-based spatio-temporal attention 3-d convolutional networks for dynamic 3-d point cloud sequences. In *T-IM*, 2021.

[86] Zhenqi Xu, Jiani Hu, and Weihong Deng. Recurrent convolutional neural network for video classification. In *ICME*, pages 1–6, 2016. doi: 10.1109/ICME.2016.7552971.

[87] Xing Li, Qian Huang, Zhijian Wang, Zhenjie Hou, and Tianjin Yang. Sequentialpointnet: A strong parallelized point cloud sequence network for 3d action recognition. *arXiv preprint arXiv:2111.08492*, 2021.

[88] Yahao Shi, Xinyu Cao, and Bin Zhou. Self-supervised learning of part mobility from point cloud sequence. *Computer Graphics Forum*, 40(6): 104–116, 2021. doi: https://doi.org/10.1111/cgf.14207. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.14207.

[89] David S Hayden, Jason Pacheco, and John W Fisher. Nonparametric object and parts modeling with lie group dynamics. In *CVPR*, 2020.

[90] Yijia Weng, He Wang, Qiang Zhou, Yuzhe Qin, Yueqi Duan, Qingnan Fan, Baoquan Chen, Hao Su, and Leonidas J Guibas. Captra: Category-level pose tracking for rigid and articulated objects from point clouds. In *ICCV*, 2021.

[91] Yunuo Chen, Minchen Li, Lei Lan, Hao Su, Yin Yang, and Chenfanfu Jiang. A unified newton barrier method for multibody dynamics. *ACM Transactions on Graphics*, 41(4):1–14, 2022.

[92] Tianhao Wu, Fangcheng Zhong, Andrea Tagliasacchi, Forrester Cole, and Cengiz Oztireli. Dˆ2nerf: Self-supervised decoupling of dynamic and static objects from a monocular video. *NeurIPS*, 35:32653–32666, 2022.

[93] Ang Cao and Justin Johnson. Hexplane: A fast representation for dynamic scenes. In *CVPR*, pages 130–141, 2023.

[94] Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance. In *CVPR*, pages 12479–12488, 2023.

[95] Yichun Shi, Peng Wang, Jianglong Ye, Long Mai, Kejie Li, and Xiao Yang. Mvdream: Multi-view diffusion for 3d generation. *arXiv:2308.16512*, 2023.

[96] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3d object. In *ICCV*, pages 9298–9309, 2023.

[97] Yuan Liu, Cheng Lin, Zijiao Zeng, Xiaoxiao Long, Lingjie Liu, Taku Komura, and Wenping Wang. Syncdreamer: Generating multiview-consistent images from a single-view image. *arXiv preprint arXiv:2309.03453*, 2023.

[98] Wanqing Li, Zhengyou Zhang, and Zicheng Liu. Action recognition based on a bag of 3D points. In *CVPRW*, 2010.

[99] Pavlo Molchanov, Xiaodong Yang, Shalini Gupta, Kihwan Kim, Stephen Tyree, and Jan Kautz. Online detection and classification of dynamic hand gestures with recurrent 3D convolutional neural network. In *CVPR*, 2016.

[100] Quentin De Smedt, Hazem Wannous, Jean-Philippe Vandeborre, Joris Guerry, Bertrand Le Saux, and David Filliat. Shrec'17 track: 3D hand gesture recognition using a depth and skeletal dataset. In *3DOR*, 2017.

[101] Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang. Ntu rgb+d: A large scale dataset for 3d human activity analysis. In *CVPR*, 2016.

[102] Jun Liu, Amir Shahroudy, Mauricio Perez, Gang Wang, Ling-Yu Duan, and Alex C Kot. Ntu rgb+ d 120: A large-scale benchmark for 3d human activity understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(10):2684–2701, 2019.

[103] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, pages 3354–3361. IEEE, 2012.

[104] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *CVPR*, pages 3061–3070, 2015.

[105] Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, et al. Sapien: A simulated part-based interactive environment. In *CVPR*, pages 11097–11107, 2020.

[106] Nikolaus Mayer, Eddy Ilg, Philip Häusser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*, 2016. doi: 10.1109/CVPR.2016.438. URL https://doi.org/10.1109/CVPR.2016.438.

[107] Xumin Yu, Lulu Tang, Yongming Rao, Tiejun Huang, Jie Zhou, and Jiwen Lu. Point-bert: Pre-training 3d point cloud transformers with masked point modeling. In *CVPR*, pages 19291–19300, 2022. doi: 10.1109/CVPR52688.2022.01871.

[108] Yatian Pang, Wenxiao Wang, Francis EH Tay, Wei Liu, Yonghong Tian, and Li Yuan. Masked autoencoders for point cloud self-supervised learning. In *ECCV*, pages 604–621, 2022.

[109] Junsheng Zhou, Jinsheng Wang, Baorui Ma, Yu-Shen Liu, Tiejun Huang, and Xinlong Wang. Uni3d: Exploring unified 3d representation at scale. In *ICLR*, 2024.

[110] Guangyan Chen, Meiling Wang, Yi Yang, Kai Yu, Li Yuan, and Yufeng Yue. Pointgpt: Auto-regressively generative pre-training from point clouds. *NeurIPS*, 36:29667–29679, 2023.

[111] Vishal Thengane, Xiatian Zhu, Salim Bouzerdoum, Son Lam Phung, and Yunpeng Li. Foundational models for 3d point clouds: A survey and outlook, 2025. URL https://arxiv.org/abs/2501.18594.

[112] Songyou Peng, Kyle Genova, Chiyu Jiang, Andrea Tagliasacchi, Marc Pollefeys, Thomas Funkhouser, et al. Openscene: 3d scene understanding with open vocabularies. In *CVPR*, pages 815–824, 2023.

[113] Justin Kerr, Chung Min Kim, Ken Goldberg, Angjoo Kanazawa, and Matthew Tancik. Lerf: Language embedded radiance fields. In *ICCV*, pages 19729–19739, 2023.

[114] David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.

[115] Lingdong Kong, Wesley Yang, Jianbiao Mei, Youquan Liu, Ao Liang, Dekai Zhu, Dongyue Lu, Wei Yin, Xiaotao Hu, Mingkai Jia, Junyuan Deng, Kaiwen Zhang, Yang Wu, Tianyi Yan, Shenyuan Gao, Song Wang, Linfeng Li, Liang Pan, Yong Liu, Jianke Zhu, Wei Tsang Ooi, Steven C. H. Hoi, and Ziwei Liu. 3d and 4d world modeling: A survey, 2025. URL https://arxiv.org/abs/2509.07996.

[116] Anthony Hu, Lloyd Russell, Hudson Yeo, Zak Murez, George Fedoseev, Alex Kendall, Jamie Shotton, and Gianluca Corrado. Gaia-1: A generative

world model for autonomous driving. *arXiv preprint arXiv:2309.17080*, 2023.

[117] Wenzhao Zheng, Weiliang Chen, Yuanhui Huang, Borui Zhang, Yueqi Duan, and Jiwen Lu. Occworld: Learning a 3d occupancy world model for autonomous driving. In *ECCV*, pages 55–72. Springer, 2024.

[118] Xiaofeng Wang, Zheng Zhu, Guan Huang, Xinze Chen, Jiagang Zhu, and Jiwen Lu. Drivedreamer: Towards real-world-drive world models for autonomous driving. In *ECCV*, pages 55–72. Springer, 2024.

[119] Jake Bruce, Michael Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, Yusuf Aytar, Sarah Bechtle, Feryal Behbahani, Stephanie Chan, Nicolas Heess, Lucy Gonzalez, Simon Osindero, Sherjil Ozair, Scott Reed, Jingwei Zhang, Konrad Zolna, Jeff Clune, Nando de Freitas, Satinder Singh, and Tim Rocktäschel. Genie: Generative interactive environments, 2024. URL https://arxiv.org/abs/2402.15391.

[120] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *CVPR*, pages 13142–13153, 2023.

[121] Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, et al. Objaverse-xl: A universe of 10m+ 3d objects. *NeurIPS*, 36:35799–35813, 2023.

[122] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dream-

fusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022.

[123] Shuyang Sun, Zhanghui Kuang, Lu Sheng, Wanli Ouyang, and Wei Zhang. Optical flow guided feature: A fast and robust motion representation for video action recognition. In *CVPR*, 2018.

[124] Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *ICCV*, 2019.

[125] Boyuan Jiang, MengMeng Wang, Weihao Gan, Wei Wu, and Junjie Yan. Stm: Spatiotemporal and motion encoding for action recognition. In *ICCV*, 2019.

[126] Can Zhang, Yuexian Zou, Guang Chen, and Lei Gan. Pan: Persistent appearance network with an efficient motion cue for fast action recognition. In *ACM MM*, 2019.

[127] Haiwei Chen, Shichen Liu, Weikai Chen, Hao Li, and Randall Hill. Equivariant point network for 3d point cloud analysis. In *CVPR*, pages 14514–14523, 2021.

[128] Hong-Xing Yu, Jiajun Wu, and Li Yi. Rotationally equivariant 3d object detection. In *CVPR*, pages 1456–1464, 2022.

[129] Jiahui Lei, Congyue Deng, Karl Schmeckpeper, Leonidas Guibas, and Kostas Daniilidis. Efem: Equivariant neural field expectation maximization for 3d object segmentation without scene supervision. In *CVPR*, pages 4902–4912, 2023.

[130] Haiwen Feng, Peter Kulits, Shichen Liu, Michael J Black, and Victoria Abrevaya. Generalizing neural human fitting to unseen poses with articulated se (3) equivariance. *arXiv preprint arXiv:2304.10528*, 2023.

[131] Jia-Xing Zhong, Ta-Ying Cheng, Yuhang He, Kai Lu, Kaichen Zhou, Andrew Markham, and Niki Trigoni. Multi-body se(3) equivariance for unsupervised rigid segmentation and motion estimation. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *NeurIPS*, volume 36, pages 76085–76097. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/efca456a4e861f3b47455c44bb134424-Paper-Conference.pdf.

[132] Charles R Qi, Hao Su, Matthias Nießner, Angela Dai, Mengyuan Yan, and Leonidas J Guibas. Volumetric and multi-view cnns for object classification on 3D data. In *CVPR*, 2016.

[133] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *NeurIPS*, 2014.

[134] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Convolutional two-stream network fusion for video action recognition. In *CVPR*, 2016.

[135] Shiwen Zhang, Sheng Guo, Weilin Huang, Matthew R Scott, and Limin Wang. V4D: 4D convolutional neural networks for video-level representation learning. In *ICLR*, 2020.

[136] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, 2017.

[137] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Val Gool. Temporal segment networks: Towards good practices for deep action recognition. In *ECCV*, 2016.

[138] Aseem Behl, Despoina Paschalidou, Simon Donné, and Andreas Geiger.

Pointflownet: Learning representations for rigid motion estimation from point clouds. In *CVPR*, June 2019.

[139] Ivan Tishchenko, Sandro Lombardi, Martin R Oswald, and Marc Pollefeys. Self-supervised learning of non-rigid residual flow and ego-motion. In *3DV*, 2020.

[140] Helmut Pottmann and Johannes Wallner. *Computational Line Geometry*. Springer Science & Business Media, 2001.

[141] Niloy J Mitra, Simon Flöry, Maks Ovsjanikov, Natasha Gelfand, Leonidas J Guibas, and Helmut Pottmann. Dynamic geometry registration. In *Symposium on geometry processing*, 2007.

[142] Yifan Xu, Tianqi Fan, Mingye Xu, Long Zeng, and Yu Qiao. Spidercnn: Deep learning on point sets with parameterized convolutional filters. In *ECCV*, 2018.

[143] Qingyong Hu, Bo Yang, Sheikh Khalid, Wen Xiao, Niki Trigoni, and Andrew Markham. Sensaturban: Learning semantics from urban-scale photogrammetric point clouds. *International Journal of Computer Vision*, pages 1–28, 2022.

[144] Qingyong Hu, Bo Yang, Guangchi Fang, Yulan Guo, Ales Leonardis, Niki Trigoni, and Andrew Markham. Sqn: Weakly-supervised semantic segmentation of large-scale 3d point clouds with 1000x fewer labels. *arXiv preprint arXiv:2104.04891*, 2021.

[145] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3D object detection network for autonomous driving. In *ICCV*, 2017.

[146] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-

Miller. Multi-view convolutional neural networks for 3D shape recognition. In *ICCV*, 2015.

[147] Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. Point-voxel cnn for efficient 3D deep learning. In *NeurIPS*, 2019.

[148] Angela Dai and Matthias Nießner. 3DMV: Joint 3D-multi-view prediction for 3D semantic scene segmentation. In *ECCV*, 2018.

[149] Rohit Girdhar, João Carreira, Carl Doersch, and Andrew Zisserman. Video Action Transformer Network. In *CVPR*, 2019.

[150] Myunggi Lee, Seungeui Lee, Sungjoon Son, Gyutae Park, and Nojun Kwak. Motion feature network: Fixed motion filter for action recognition. In *ECCV*, 2018.

[151] Heeseung Kwon, Manjin Kim, Suha Kwak, and Minsu Cho. Motionsqueeze: Neural motion feature learning for video understanding. In *ECCV*, 2020.

[152] AJ Piergiovanni and Michael S Ryoo. Representation flow for action recognition. In *CVPR*, 2019.

[153] Lijie Fan, Wenbing Huang, Chuang Gan, Stefano Ermon, Boqing Gong, and Junzhou Huang. End-to-end learning of motion representation for video understanding. In *CVPR*, 2018.

[154] Javier Sánchez Pérez, Enric Meinhardt-Llopis, and Gabriele Facciolo. TV-L1 Optical Flow Estimation. *Image Processing On Line*, 3:137–150, 2013. https://doi.org/10.5201/ipol.2013.26.

[155] W.P. Krijnen and H.A.L. Kiers. An efficient algorithm for weighted pca. *Computational Statistics*, 10(3):299–306, 1995. ISSN 0943-4062.

[156] Wei Wang, Zheng Dang, Yinlin Hu, Pascal Fua, and Mathieu Salzmann. Backpropagation-friendly eigendecomposition. In *NeurIPS*, 2019.

[157] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL https://www.tensorflow.org/. Software available from tensorflow.org.

[158] Xiaodong Yang, Pavlo Molchanov, and Jan Kautz. Making convolutional networks recurrent for visual sequence learning. In *CVPR*, 2018.

[159] Mahdi Abavisani, Hamid Reza Vaezi Joze, and Vishal M Patel. Improving the performance of unimodal dynamic hand-gesture recognition with multimodal training. In *CVPR*, 2019.

[160] Quentin De Smedt, Hazem Wannous, and Jean-Philippe Vandeborre. Skeleton-based dynamic hand gesture recognition. In *CVPRW*, 2016.

[161] Jingxuan Hou, Guijin Wang, Xinghao Chen, Jing-Hao Xue, Rui Zhu, and Huazhong Yang. Spatial-temporal attention res-tcn for skeleton-based dynamic hand gesture recognition. In *ECCVW*, 2018.

[162] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convo-

lutional networks for skeleton-based action recognition. In *AAAI*, 2018.

[163] Yuxiao Chen, Long Zhao, Xi Peng, Jianbo Yuan, and Dimitris N Metaxas. Construct dynamic graphs for hand gesture recognition via spatial-temporal attention. In *BMVC*, 2019.

[164] Antonio W Vieira, Erickson R Nascimento, Gabriel L Oliveira, Zicheng Liu, and Mario FM Campos. Stop: Space-time occupancy patterns for 3D action recognition from depth map sequences. In *Iberoamerican congress on pattern recognition*, 2012.

[165] Alexander Klaser, Marcin Marszałek, and Cordelia Schmid. A spatio-temporal descriptor based on 3D-gradients. In *BMVC*, 2008.

[166] Jiang Wang, Zicheng Liu, Ying Wu, and Junsong Yuan. Mining actionlet ensemble for action recognition with depth cameras. In *CVPR*, pages 1290–1297, 2012. doi: 10.1109/CVPR.2012.6247813.

[167] Carlos Caetano, Jessica Sena, François Brémond, Jefersson A Dos Santos, and William Robson Schwartz. Skelemotion: A new representation of skeleton joint sequences based on motion information for 3d action recognition. In *AVSS*, 2019.

[168] Jun Liu, Gang Wang, Ping Hu, Ling-Yu Duan, and Alex C Kot. Global context-aware attention lstm networks for 3d action recognition. In *CVPR*, 2017.

[169] Jun Liu, Gang Wang, Ling-Yu Duan, Kamila Abdiyeva, and Alex C Kot. Skeleton-based human action recognition with global context-aware attention lstm networks. *IEEE Transactions on Image Processing*, 27(4):1586–1599, 2017.

[170] Chenyang Si, Wentao Chen, Wei Wang, Liang Wang, and Tieniu Tan. An

attention enhanced graph convolutional lstm network for skeleton-based action recognition. In *CVPR*, 2019.

[171] Maosen Li, Siheng Chen, Xu Chen, Ya Zhang, Yanfeng Wang, and Qi Tian. Actional-structural graph convolutional networks for skeleton-based action recognition. In *CVPR*, 2019.

[172] Pengfei Zhang, Cuiling Lan, Junliang Xing, Wenjun Zeng, Jianru Xue, and Nanning Zheng. View adaptive neural networks for high performance skeleton-based human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(8):1963–1978, 2019.

[173] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In *CVPR*, 2019.

[174] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Skeleton-based action recognition with directed graph neural networks. In *CVPR*, 2019.

[175] Ziyu Liu, Hongwen Zhang, Zhenghao Chen, Zhiyong Wang, and Wanli Ouyang. Disentangling and unifying graph convolutions for skeleton-based action recognition. In *CVPR*, 2020.

[176] Omar Oreifej and Zicheng Liu. Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences. In *CVPR*, 2013.

[177] Xiaodong Yang and YingLi Tian. Super normal vector for activity recognition using depth sequences. In *CVPR*, 2014.

[178] Eshed Ohn-Bar and Mohan Trivedi. Joint angles similarities and hog2 for action recognition. In *CVPRW*, 2013.

[179] Junnan Li, Yongkang Wong, Qi Zhao, and Mohan S Kankanhalli. Unsupervised learning of view-invariant action representations. In *NeurIPS*, 2018.

[180] Pichao Wang, Wanqing Li, Zhimin Gao, Chang Tang, and Philip O Ogunbona. Depth pooling based large-scale 3-d action recognition with convolutional neural networks. *IEEE Transactions on Multimedia*, 20(5):1051–1061, 2018.

[181] Yang Xiao, Jun Chen, Yancheng Wang, Zhiguo Cao, Joey Tianyi Zhou, and Xiang Bai. Action recognition for depth video using multi-view dynamic images. *Information Sciences*, 480:287–304, 2019.

[182] Andreas Geiger, Martin Lauer, Christian Wojek, Christoph Stiller, and Raquel Urtasun. 3D Traffic Scene Understanding From Movable Platforms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(5): 1012–1025, 2014. doi: 10.1109/TPAMI.2013.185.

[183] Taco Cohen and Max Welling. Group equivariant convolutional networks. In *ICML*, pages 2990–2999. PMLR, 2016.

[184] Congyue Deng, Or Litany, Yueqi Duan, Adrien Poulenard, Andrea Tagliasacchi, and Leonidas J Guibas. Vector neurons: A general framework for so (3)-equivariant networks. In *ICCV*, pages 12200–12209, 2021.

[185] Taco S Cohen, Mario Geiger, Jonas Köhler, and Max Welling. Spherical cnns. In *ICLR*, 2018.

[186] Maurice Weiler, Fred A Hamprecht, and Martin Storath. Learning steerable filters for rotation equivariant cnns. In *CVPR*, pages 849–858, 2018.

[187] Haiping Wang, Yuan Liu, Zhen Dong, and Wenping Wang. You only hypothesize once: Point cloud registration with rotation-equivariant descrip-

tors. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 1630–1641, 2022.

[188] Xiaolong Li, Yijia Weng, Li Yi, Leonidas J Guibas, A Abbott, Shuran Song, and He Wang. Leveraging se (3) equivariance for self-supervised category-level object pose estimation from point clouds. *NeurIPS*, 34:15370–15381, 2021.

[189] Evangelos Chatzipantazis, Stefanos Pertigkiozoglou, Edgar Dobriban, and Kostas Daniilidis. Se (3)-equivariant attention networks for shape reconstruction in function space. *arXiv preprint arXiv:2204.02394*, 2022.

[190] Omri Puny, Matan Atzmon, Edward J Smith, Ishan Misra, Aditya Grover, Heli Ben-Hamu, and Yaron Lipman. Frame averaging for invariant and equivariant network design. In *ICLR*, 2022.

[191] David S Hayden, Jason Pacheco, and John W Fisher. Nonparametric object and parts modeling with lie group dynamics. In *CVPR*, pages 7426–7435, 2020.

[192] Hehe Fan, Xin Yu, Yi Yang, and Mohan Kankanhalli. Deep hierarchical representation of point cloud videos via spatio-temporal decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(12): 9918–9930, 2021.

[193] Yuhang He, Lin Chen, Junkun Xie, and Long Chen. Learning 3D Semantics From Pose-Noisy 2D Images with Hierarchical Full Attention Network. In *ECCV Workshops*, pages 726–742. Springer Nature Switzerland, 2023. ISBN 978-3-031-25056-9.

[194] Yahao Shi, Xinyu Cao, Feixiang Lu, and Bin Zhou. Pˆ3-net: Part mobility parsing from point cloud sequences via learning explicit point correspon-

dence. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36 (2):2244–2252, Jun. 2022. doi: 10.1609/aaai.v36i2.20122. URL https://ojs.aaai.org/index.php/AAAI/article/view/20122.

[195] Li Yi, Haibin Huang, Difan Liu, Evangelos Kalogerakis, Hao Su, and Leonidas Guibas. Deep part induction from articulated object pairs. *ACM Transactions on Graphics*, 37(6):1–15, 2018.

[196] Wencan Cheng and Jong Hwan Ko. Bi-pointflownet: Bidirectional learning for point cloud based scene flow estimation. In *ECCV*, pages 108–124. Springer, 2022.

[197] Aseem Behl, Despoina Paschalidou, Simon Donne, and Andreas Geiger. Pointflownet: Learning representations for rigid motion estimation from point clouds. In *CVPR*, June 2019.

[198] Ting Li, Vinutha Kallem, Dheeraj Singaraju, and René Vidal. Projective factorization of multiple rigid-body motions. In *CVPR*, pages 1–6. IEEE, 2007.

[199] Xun Xu, Loong-Fah Cheong, and Zhuwen Li. 3d rigid motion segmentation with mixed and unknown number of models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(1):1–16, 2019.

[200] Jiahui Huang, Sheng Yang, Zishuo Zhao, Yu-Kun Lai, and Shi-Min Hu. Clusterslam: A slam backend for simultaneous rigid body clustering and motion estimation. In *ICCV*, pages 5875–5884, 2019.

[201] Luca Magri and Andrea Fusiello. Fitting multiple heterogeneous models by multi-class cascaded t-linkage. In *CVPR*, pages 7460–7468, 2019.

[202] Hossam Isack and Yuri Boykov. Energy-based geometric multi-model fitting. *International journal of computer vision*, 97(2):123–147, 2012.

[203] Tolga Birdal and Slobodan Ilic. Cad priors for accurate and flexible instance reconstruction. In *ICCV*, pages 133–142, 2017.

[204] Florian Kluger, Eric Brachmann, Hanno Ackermann, Carsten Rother, Michael Ying Yang, and Bodo Rosenhahn. Consac: Robust multi-model fitting by conditional sample consensus. In *CVPR*, pages 4634–4643, 2020.

[205] Johannes Brandstetter, Rob Hesselink, Elise van der Pol, Erik J Bekkers, and Max Welling. Geometric and physical quantities improve e (3) equivariant message passing. In *ICLR*, 2021.

[206] Limei Wang, Haoran Liu, Yi Liu, Jerry Kurtin, and Shuiwang Ji. Learning hierarchical protein representations via complete 3d graph networks. In *ICLR*, 2022.

[207] Taco S Cohen and Max Welling. Steerable cnns. In *ICLR*, 2016.

[208] Thomas W Mitchel, Noam Aigerman, Vladimir G Kim, and Michael Kazhdan. Möbius convolutions for spherical cnns. In *ACM SIGGRAPH 2022 Conference Proceedings*, pages 1–9, 2022.

[209] Taco Cohen, Maurice Weiler, Berkay Kicanaoglu, and Max Welling. Gauge equivariant convolutional networks and the icosahedral cnn. In *ICML*, pages 1321–1330. PMLR, 2019.

[210] Chien Erh Lin, Jingwei Song, Ray Zhang, Minghan Zhu, and Maani Ghaffari. Se(3)-equivariant point cloud-based place recognition. In Karen Liu, Dana Kulic, and Jeff Ichnowski, editors, *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pages 1520–1530. PMLR, 14–18 Dec 2023.

[211] Yair Kittenplon, Yonina C. Eldar, and Dan Raviv. FlowStep3D: Model Unrolling for Self-Supervised Scene Flow Estimation. *CVPR*, 2021.

[212] Wolfgang Kabsch. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, 32(5):922–923, 1976.

[213] Zan Gojcic, Caifa Zhou, Jan D Wegner, Leonidas J Guibas, and Tolga Birdal. Learning multiview 3d point cloud registration. In *CVPR*, pages 1759–1769, 2020.

[214] Urbano Miguel Nunes and Yiannis Demiris. 3d motion segmentation of articulated rigid bodies based on RGB-D data. In *BMVC*, page 255. BMVA Press, 2018.

[215] Peter Ochs, Jitendra Malik, and Thomas Brox. Segmentation of Moving Objects by Long Term Video Analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2014.

[216] Joe H. Ward. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58(301):236–244, 1963. doi: 10.1080/01621459.1963.10500845.

[217] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In Evangelos Simoudis, Jiawei Han, and Usama Fayyad, editors, *Proc. 2nd Int. Conf. on Knowledge Discovery and Data Mining (KDD'96)*, pages 226–231. AAAI Press, 1996.

[218] Xinge Zhu, Hui Zhou, Tai Wang, Fangzhou Hong, Yuexin Ma, Wei Li, Hongsheng Li, and Dahua Lin. Cylindrical and asymmetrical 3D convolution networks for LiDAR segmentation. In *CVPR*, 2021.

[219] Haotian* Tang, Zhijian* Liu, Shengyu Zhao, Yujun Lin, Ji Lin, Hanrui

Wang, and Song Han. Searching efficient 3d architectures with sparse point-voxel convolution. In *ECCV*, 2020.

[220] Xu Yan, Jiantao Gao, Chaoda Zheng, Chao Zheng, Ruimao Zhang, Shuguang Cui, and Zhen Li. 2dpass: 2d priors assisted semantic segmentation on lidar point clouds. In *ECCV*, pages 677–695. Springer, 2022.

[221] Yuyang Yin, Dejia Xu, Zhangyang Wang, Yao Zhao, and Yunchao Wei. 4dgen: Grounded 4d content generation with spatial-temporal consistency. *ArXiv*, abs/2312.17225, 2023. URL https://api.semanticscholar.org/CorpusID:266573572.

[222] Jiemin Fang, Taoran Yi, Xinggang Wang, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Matthias Nießner, and Qi Tian. Fast dynamic radiance fields with time-aware neural voxels. In *SIGGRAPH Asia 2022 Conference Papers*, 2022.

[223] Yifan Liu, Chenxin Li, Chen Yang, and Yixuan Yuan. Endogaussian: Gaussian splatting for deformable surgical scene reconstruction, 2024.

[224] Yifan Liu, Chenxin Li, Hengyu Liu, Chen Yang, and Yixuan Yuan. Foundation model-guided gaussian splatting for 4d reconstruction of deformable tissues. *IEEE Transactions on Medical Imaging*, 2025.

[225] Lingting Zhu, Zhao Wang, Zhenchao Jin, Guying Lin, and Lequan Yu. Deformable endoscopic tissues reconstruction with gaussian splatting. *arXiv preprint arXiv:2401.11535*, 2024.

[226] Haimin Luo, Teng Xu, Yuheng Jiang, Chenglin Zhou, Qiwei Qiu, Yingliang Zhang, Wei Yang, Lan Xu, and Jingyi Yu. Artemis: Articulated neural pets with appearance and motion synthesis. *ACM Transactions*

on Graphics, 41(4), jul 2022. ISSN 0730-0301. doi: 10.1145/3528223.3530086. URL https://doi.org/10.1145/3528223.3530086.

[227] Dario Pavllo, Graham Spinks, Thomas Hofmann, Marie-Francine Moens, and Aurelien Lucchi. Convolutional generation of textured 3d meshes. *NeurIPS*, 33:870–882, 2020.

[228] Tianchang Shen, Jun Gao, Kangxue Yin, Ming-Yu Liu, and Sanja Fidler. Deep marching tetrahedra: a hybrid representation for high-resolution 3d shape synthesis. *NeurIPS*, 34:6087–6101, 2021.

[229] Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. Magic3d: High-resolution text-to-3d content creation. In *CVPR*, pages 300–309, 2023.

[230] Yixun Liang, Xin Yang, Jiantao Lin, Haodong Li, Xiaogang Xu, and Yingcong Chen. Luciddreamer: Towards high-fidelity text-to-3d generation via interval score matching. In *CVPR*, pages 6517–6526, 2024.

[231] Guocheng Qian, Jinjie Mai, Abdullah Hamdi, Jian Ren, Aliaksandr Siarohin, Bing Li, Hsin-Ying Lee, Ivan Skorokhodov, Peter Wonka, Sergey Tulyakov, et al. Magic123: One image to high-quality 3d object generation using both 2d and 3d diffusion priors. *arXiv preprint arXiv:2306.17843*, 2023.

[232] Jingxiang Sun, Bo Zhang, Ruizhi Shao, Lizhen Wang, Wen Liu, Zhenda Xie, and Yebin Liu. Dreamcraft3d: Hierarchical 3d generation with bootstrapped diffusion prior. *arXiv preprint arXiv:2310.16818*, 2023.

[233] Jiaxiang Tang, Jiawei Ren, Hang Zhou, Ziwei Liu, and Gang Zeng. Dream-

gaussian: Generative gaussian splatting for efficient 3d content creation. *arXiv preprint arXiv:2309.16653*, 2023.

[234] Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. *NeurIPS*, 36:8406–8441, 2023.

[235] Weiyu Li, Rui Chen, Xuelin Chen, and Ping Tan. Sweetdreamer: Aligning geometric priors in 2d diffusion for consistent text-to-3d. *arxiv:2310.02596*, 2023.

[236] Taoran Yi, Jiemin Fang, Junjie Wang, Guanjun Wu, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Qi Tian, and Xinggang Wang. Gaussiandreamer: Fast generation from text to 3d gaussians by bridging 2d and 3d diffusion models. In *CVPR*, pages 6796–6807, 2024.

[237] Jiemin Fang, Taoran Yi, Xinggang Wang, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Matthias Nießner, and Qi Tian. Fast dynamic radiance fields with time-aware neural voxels. In *SIGGRAPH Asia 2022 Conference Papers*, pages 1–9, 2022.

[238] Chen Gao, Ayush Saraf, Johannes Kopf, and Jia-Bin Huang. Dynamic view synthesis from dynamic monocular video. In *ICCV*, pages 5712–5721, 2021.

[239] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *CVPR*, pages 6498–6508, 2021.

[240] Guikun Chen and Wenguan Wang. A survey on 3d gaussian splatting. *arXiv preprint arXiv:2401.03890*, 2024.

[241] Yiqing Liang, Numair Khan, Zhengqin Li, Thu Nguyen-Phuoc, Douglas

Lanman, James Tompkin, and Lei Xiao. Gaufre: Gaussian deformation fields for real-time dynamic novel view synthesis. *WACV*, pages 2642–2652, 2023. URL https://api.semanticscholar.org/CorpusID:266359262.

[242] Ruoxi Shi, Hansheng Chen, Zhuoyang Zhang, Minghua Liu, Chao Xu, Xinyue Wei, Linghao Chen, Chong Zeng, and Hao Su. Zero123++: a single image to consistent multi-view diffusion base model. *ArXiv*, abs/2310.15110, 2023. URL https://api.semanticscholar.org/CorpusID:264436559.

[243] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: learning dynamic renderable volumes from images. *ACM Transactions on Graphics*, 38 (4), July 2019. ISSN 0730-0301. doi: 10.1145/3306346.3323020. URL https://doi.org/10.1145/3306346.3323020.

[244] Jakub Bican, Daniel Janeba, Katerina Táborská, and Jaroslav Vesely. Image overlay using alpha-blending technique. *Nuclear Medicine Review*, 5(1): 53–53, 2002.

[245] Aravindh Krishnamoorthy and Deepak Menon. Matrix inversion using cholesky decomposition. In *2013 signal processing: Algorithms, architectures, arrangements, and applications (SPA)*, pages 70–72. IEEE, 2013.

[246] Diederick P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[247] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008. URL http://jmlr.org/papers/v9/vandermaaten08a.html.

[248] Li Yi, Vladimir G Kim, Duygu Ceylan, I-Chao Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, and Leonidas Guibas. A scalable active framework for region annotation in 3d shape collections. *ACM Transactions on Graphics*, 35(6):1–12, 2016.

[249] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *ECCV*, pages 523–540, 2020.

[250] Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *ECCV*, pages 628–644. Springer, 2016.

[251] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, pages 8024–8035. Curran Associates, Inc., 2019.

[252] Alain Hore and Djemel Ziou. Image quality metrics: Psnr vs. ssim. In *ICPR*, pages 2366–2369. IEEE, 2010.

[253] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[254] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

[255] Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. CLIPScore: a reference-free evaluation metric for image captioning. In *EMNLP*, 2021.

[256] Thomas Unterthiner, Sjoerd Van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski, and Sylvain Gelly. Towards accurate generative models of video: A new metric & challenges. *arXiv preprint arXiv:1812.01717*, 2018.

[257] Yogesh Balaji, Martin Renqiang Min, Bing Bai, Rama Chellappa, and Hans Peter Graf. Conditional gan with discriminative filter generation for text-to-video synthesis. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 1995–2001. International Joint Conferences on Artificial Intelligence Organization, 7 2019. doi: 10.24963/ijcai.2019/276. URL https://doi.org/10.24963/ijcai.2019/276.

A | Appendix of High-level Representation Learning (Chapter 3)

In Chapter 3, we proposed *Kinet*, a kinematics-inspired network that generalizes the concept of ST-surfaces to the feature space for dynamic point cloud classification. More specifically, we detail the model architecture and the unrolled normal solver, and report results on NvGesture, SHREC'17, MSRAction-3D, and NTU-RGBD. This appendix provides the implementation and training details necessary for reproducibility: backbone configurations, feature dimensions, temporal windowing, data preprocessing and augmentation, optimization settings, hyperparameters, and evaluation protocols. We also describe the software/hardware environment and runtime characteristics. In addition, this appendix complements the empirical study presented in Chapter 3 by providing extended ablations and diagnostics. We examine the contribution of each architectural component (*e.g.*, the ST-surface-inspired solver, temporal receptive field, and backbone choices), analyze sensitivity to hyperparameters, and report additional cross-dataset evaluations.

A.1 Implementation Details

A.1.1 Matrix Inversion

Equation (6) in the main body of this thesis is to fit the group-wise ST-surface via the closed-form least-squared solution:

$$[A_k^{T*}, b_k^*] = (F_{i,k}^{(t)T} W_{i,k}^{(t)} F_{i,k}^{(t)})^{-1} F_{i,k}^{(t)T} W_{i,k}^{(t)} \boldsymbol{\tau}_{i,k}^{(t)}, \tag{A.1}$$

where the weight matrix $W_{i,k}^{(t)} = diag(w_{1,k}^{(\tau)}, ..., w_{|N_i^{(t)}|,k}^{(\tau)}) \in \mathbb{R}^{|N_i^{(t)}| \times |N_i^{(t)}|}$, the feature matrix $F_{i,k}^{(t)} = [(\mathbf{f}_{1,k}, 1), .., (\mathbf{f}_{|N_i^{(t)}|,k}, 1)] \in \mathbb{R}^{|N_i^{(t)}| \times (d+1)}$ and the time vector $\boldsymbol{\tau}_{i,k}^{(t)} \in \mathbb{R}^{|N_i^{(t)}|}$.

Note that we do not utilize the default implementation of matrix inversion on Tensorflow [157] since the default implementation using the LU decomposition is specially designed for large matrices. In our settings, matrices are relatively small in each group and we adopt home-brew approaches to matrix inversion. Given an $n \times n$ square matrix $M \in \mathbb{R}^{n \times n}$, we inverse the matrix according to its size.

In the case of extremely small matrices ($n \leq 4$), we directly inverse the matrix with element-wise calculations. For example, let $M = \begin{pmatrix} m_1 & m_2 \\ m_3 & m_4 \end{pmatrix} \in \mathbb{R}^{2 \times 2}$. The inversion of this matrix is $M^{-1} = \frac{1}{m_1 m_4 - m_2 m_3} \begin{pmatrix} m_4 & -m_2 \\ -m_3 & m_1 \end{pmatrix}$, where $m_1 m_4 - m_2 m_3$ is clipped to the minimal value of 1×10^{-6} for the numerical stability of singular matrices. In terms of medium matrices ($4 < n \leq 16$), we recursively inverse the partitioned matrices based on the aforementioned element-wise operations for small matrices. For the case $n > 16$, we leverage Cholesky decomposition [245] to solve the matrix inversion.

A.1.2 Network Structures

A.1.2.1 Kinet: MLP-based Backbone (PointNet++)

The MLP-based methods separately model every point with shared multi-layer perceptions (MLPs), followed by a symmetric aggregation (*e.g.*, max pooling) to fuse order-invariant information. In this thesis, we choose PointNet++ [28] as our MLP-based backbone for static point clouds. For a fair comparison, we keep the number of parameters in each layer identical to FlickerNet [82], one of the state-of-the-art gesture recognition networks on sequential point clouds.

A.1.2.2 Kinet: Graph-based Backbone (DGCNN)

Unlike the MLP-based backbones independently capturing point-level representations, graph-based methods model point-wise interactions by regarding a point cloud as a graph, of which each point is the vertex and edges are established upon the neighboring distribution of these points. For the graph-based paradigm, we conduct experiments on the static backbone of DGCNN [31] and keep the default settings of layer-wise parameters including the feature aggregation of the channel-wise additions in [31].

A.1.2.3 Kinet: Conv.-based Backbone (SpiderCNN)

Different from the other two paradigms using graphs or MLPs, the conv.-based models directly devise the convolution, particularly for unstructured point clouds. We adopt SpiderCNN [142] as our convolution-based static backbone and keep the same layer-wise settings including the feature aggregation of **concatenation** and **top-k pooling**.

A.1.3 Training Configurations

The proposed framework is implemented with Tensorflow [157]. All experiments are conducted on the NVIDIA DGX-1 stations with Tesla V100 GPUs. During the training stage, the hyper-parameters are $batch\_size = 16$, $base\_learning\_rate = 0.001$, with an Adam optimizer [246]. The number of training epochs depends on various datasets: 200 epochs on *NvGesture/SHREC'17*, 150 epochs on *MSRAction-3D*, and 20 epochs on *NTU-RGBD*. For training stability, we first train the static backbone (spatial stream) until convergence and then freeze its weights to individually optimize the dynamic branch (temporal stream). To make full use of the features in various scales, multiple layers are connected to output the final results.

As for the hyper-settings of Kinet itself, we set the ratio of feature reduction as 50%, group-wise dimensions $d = 4$, temporal radius $\Delta t = 1$, and spatial radius $\Delta r = 0.5$, respectively.

A.1.4 Point Activation Clouds

This concept stems from FlickerNet [82], which indicates the highlighted points of a model. Likewise, P4Transformer [16] and PSTNet [15] have similar visualization concepts of the attentional values and the convolutional outputs, respectively. For readability, we formulate the concept of Point Activation Clouds (PACs) for the proposed two-stream framework.

Given a frame P_t of point clouds at the t^{th} time step, denote the activated feature vector of the i^{th} point $p_i^{(t)}$ in the l^{th} layer as $\boldsymbol{\phi}_l(p_i^{(t)})$. The PAC of a centroid $p_i^{(t)}$ in this layer is defined as:

$$PAC_l(p_i^{(t)}) = \max_{\phi_l \in \boldsymbol{\phi}_l} \max_{p_j^{(\tau)} \in N_i^{(t)}} \phi_l(p_j^{(\tau)}) \,, \tag{A.2}$$

where $N_i^{(t)}$ is the space-time neighbor set of $p_i^{(t)}$. Thus, the larger PACs reflect the greater activated values for these centroids, which highlights the discernible outputs in the l^{th} layer.

In the main body of this thesis, we visualize the PACs of the last layer before the first max-pooling operation in the static stream of PointNet++ [28] and the corresponding layer in our dynamic branch.

A.1.5 Point-wise Scene Flow Estimation of Flow-based Baselines

In Section 4.3 of the main body, a self-supervised framework *Justgo* [24] is trained to estimate scene flow and we adopt it as an additional input in the setting 3). Note that Kinet does not require scene flow and setting 3) serves as the flow-based baseline.

A.1.5.1 Self-supervised Training Details

Based on the model pre-trained on *Flythings3D* [106], we train the scene flow estimator with the following hyper-parameters: $batch\_size = 8$, $radius = 5$, $flip\_prob = 0.5$, $base\_learning\_rate = 0.001$, with an Adam optimizer [246]. The balancing weight of the nearest neighbor loss and the cycle loss is set as 1:1.

A.1.5.2 Inference

The scene flow is first extracted by applying the trained *Justgo* model. Then, we scale the estimated scene flow to the same range of the raw point clouds, so that a static model with default hyper-parametric settings can be directly applied to the input modality of scene flow. Note that the number of parameters in a trained *Justgo* (excluding the parameters of the Adam optimizer) achieves 3.54M. For a 16-frame snippet with 2048 points per frame, the snippet-wise computational complexity of scene flow extraction is 154.29G FLOPs. The computational overhead of scene flow estimation is considerable.

A.1.5.3 Visualization

As depicted in Figure A.1b, scene flow is decently estimated between two consecutive frames (as depicted in Figure A.1a) with the self-supervised network. It is

observed that key motions of the runner's legs are well captured but the interpolation results for the runner's head are insufficiently accurate: scene flow estimation without explicit supervision is a highly challenging task. As one of the input modalities for a vanilla two-stream model, the scaled scene flow in Figure A.1c highlights the legs' movements, still preserving crucial dynamic information.

A.2 Detailed Experimental Results

Model	FLOPs (G)	#PARAMS (M)	Accuracy (%)
Static PointNet++ [28]	4.82	2.13	84.30
Flow-based Two Streams (PointNet++)	163.93	7.79	90.23
Kinet (Pointnet++)	10.35	3.20	91.92
Static DGCNN [31]	5.33	5.25	84.18
Flow-based Two Streams (DGCNN)	164.95	14.04	89.56
Kinet (DGCNN)	5.83	5.85	89.82
Static SpiderCNN [142]	13.40	8.03	83.49
Flow-based Two Streams (SpiderCNN)	181.09	19.60	89.23
Kinet (SpiderCNN)	15.29	8.62	88.54
MeteorNet [14]	1.70	17.60	88.21
PSTNet [15]	54.09	8.44	89.90
P4Transformer [16]	40.38	42.07	89.56

Table A.1: *Quantitative results of parameter number, FLOPs and accuracy on 16-frame* MSRAction-3D.

A.2.1 Quantitative Computational Costs

In section 4.3 of the main body, we conduct 3 groups of experiments: 1) Directly feed videos into the static model; 2) Fuse the static model and the dynamic branch; 3) Ensemble classification scores from two static models, one is trained on the raw point clouds, while the other is trained on estimated scene flow. As shown in Table A.1, the extra input modality of estimated scene flow in setting 3) (+S, Flow-based Two Streams) considerably improves the accuracy of the three static models (Static PointNet++, DGCNN and SpiderCNN) to a level comparable

to the state-of-the-art. However, the scene flow estimator and another flow-based classifier almost triple the number of parameters. Even worse, the estimation of scene flow introduces more than 150G FLOPs of extra calculations since the point-wise dense predictions are required between every two consecutive frames. For setting 2) (+Ours, Kinet), it is observed that our kinematic representations consistently increase the accuracy of the static predictions by 5.99%$\sim$9.04% relative gains. By utilizing the kinematic representations, the FLOPs only increases to 5.83G$\sim$15.29G, and the number of parameters increases by 0.59M$\sim$1.08M. These computing overheads are negligible and make the fused model extremely lightweight.

A.2.2 Per-class Performance Gains

Following Liu *et al*. [14], we report the performance gains over the original static model on *MSRAction-3D*. Under the same experimental configurations as the main body, we take 16 frames as an input unit and sample 2048 points for each frame. By comparing with the baseline (*PointNet++*) accuracy of setting 1), we report the performance changes of setting 2) and 3) in per-class action categorization.

As illustrated in Figure A.4a, our dynamic branch significantly improves the classification accuracy of the static model. Noticeably, the categories of "High Throw" and "Hand Catch" show more than 30% absolute performance gains and most of the other performance changes achieve at least 20% improvements. There is only one category ("Horizontal Arm Wave") with about 6% performance decline. Figure A.4b demonstrates that the extra input modality of estimated scene flow also boosts the overall performance. It has a large performance drop of more than 10% in the category of "High Arm Wave" and limited performance gains on many classes. Compared with the raw scene flow, our kinematic approach has high ro-

bustness and considerable improvements because it does not rely on the inaccurate estimation of scene flow.

A.3 More Visualizations

A.3.1 Point-level Sequential PACs

Due to the page limitation in the main body, we visualize the depth videos and dynamic PACs in Figure 6 with the format of animations. In case the PDF reader cannot display the animations normally, we provide the sequential images as shown in Figure A.2.

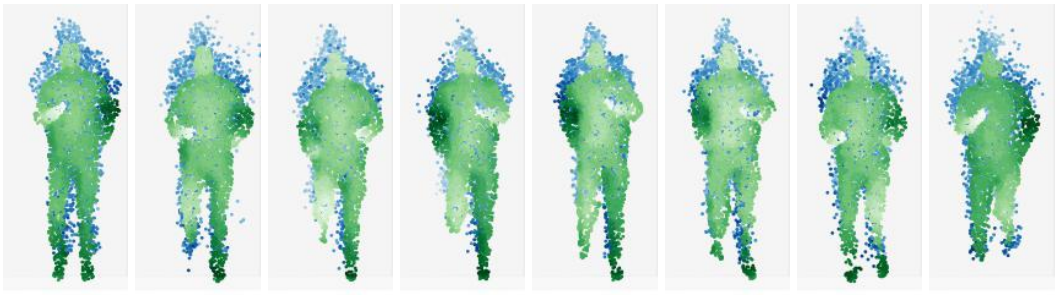
Unlike most of the prior works only focusing on the background-free cases, we adopt two input settings to verify whether Kinet can capture useful movements and ignore meaningless ones: 1) w/ BBox (Figure A.2a) - used by a majority of existing methods for high accuracy, based on the area inside the bounding boxes of hand skeletons without background interference; 2) w/o BBox (Figure A.2d) - raw videos with noisy backgrounds (the performer's body). With bounding boxes removing noisy backgrounds, the two streams work complementary - the static branch (Figure A.2b) highlights the main parts (the palms) of spatial appearances, whereas the temporal representations (Figure A.2c) capture key motions, such as the movement of fingers and wrists. In the case with redundant backgrounds (without bounding boxes), the static stream (Figure A.2e) excessively focuses on the large yet useless background portions (the performer's body), while the temporal stream (Figure A.2f) captures the moving parts (arms and fingers). Inevitably, the temporal stream also highlights several redundant points of the performer's shaking head by mistake.

A.3.2 Video-level t-SNE Features

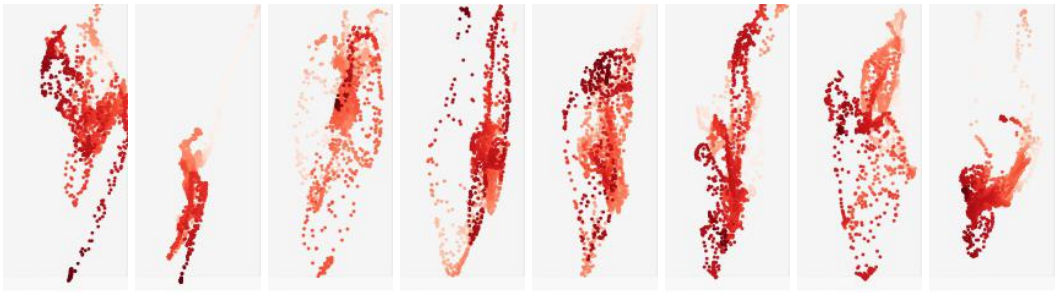
Different from PACs encoding point-level activation in intermediate layers, the last-layer features before the classifier aggregate the video-level information in the whole model. To qualitatively analyze the video-level representations, we project these last-layer features to the 2-dimensional space through t-SNE [247].

As shown in Figure A.3a & A.3b, in the cases without backgrounds (w/ BBox), our dynamic branch (the temporal stream) has better intra-class compactness and inter-class separability than the static one (the spatial stream). Though the dynamic branch shows overall superiority, the static stream is complementary to it: the dynamic branch sometimes confuses the orange data points with the yellow ones, while the static branch is capable of discriminating them correctly. In the case with backgrounds (w/o BBox), Figure A.3c & A.3d demonstrate that the background noises have negative impacts on both static and kinematic representations. In this case, our dynamic features are still highly robust to backgrounds compared with the static ones.

(a) Point Cloud

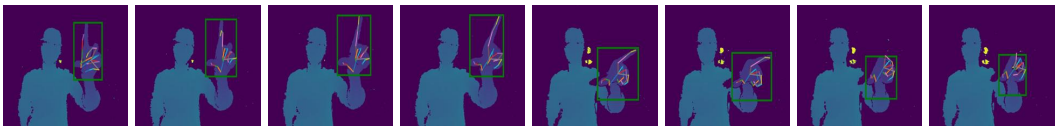


(b) Interpolation

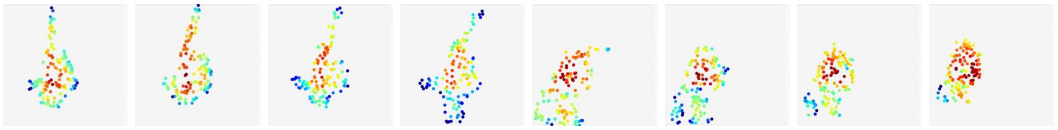


(c) Scaled Flow

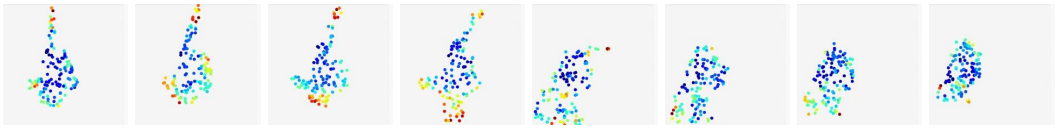
Figure A.1: *Estimated point-wise scene flow of the flow-based baselines on* MSRAction-3D. The darker color indicates the greater depths (in point clouds) or the larger motions (in scene flow). (a) is the input of raw point clouds. For the frame interpolation in (b), the green points are the ground-truth point clouds, while the blue points are the interpolation results based on the last frame and the estimated scene flow. For the estimated scene flow in (c), it is normalized to the same scale as raw point clouds.



(a) Depth Maps, w/ BBox

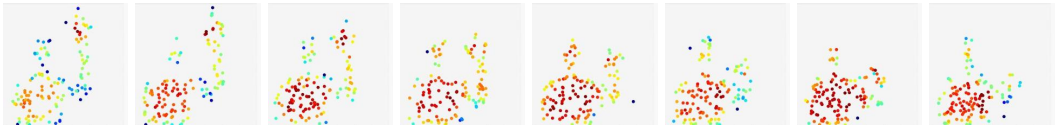


(b) Spatial (Static), w/ BBox

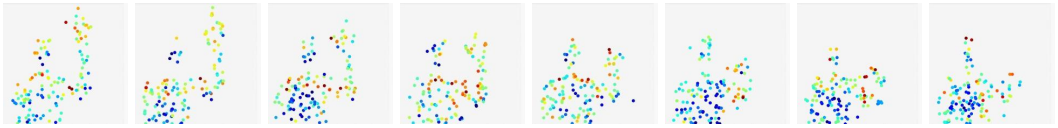


(c) Temporal (Dynamic), w/ BBox

(d) Depth Maps, w/o BBox

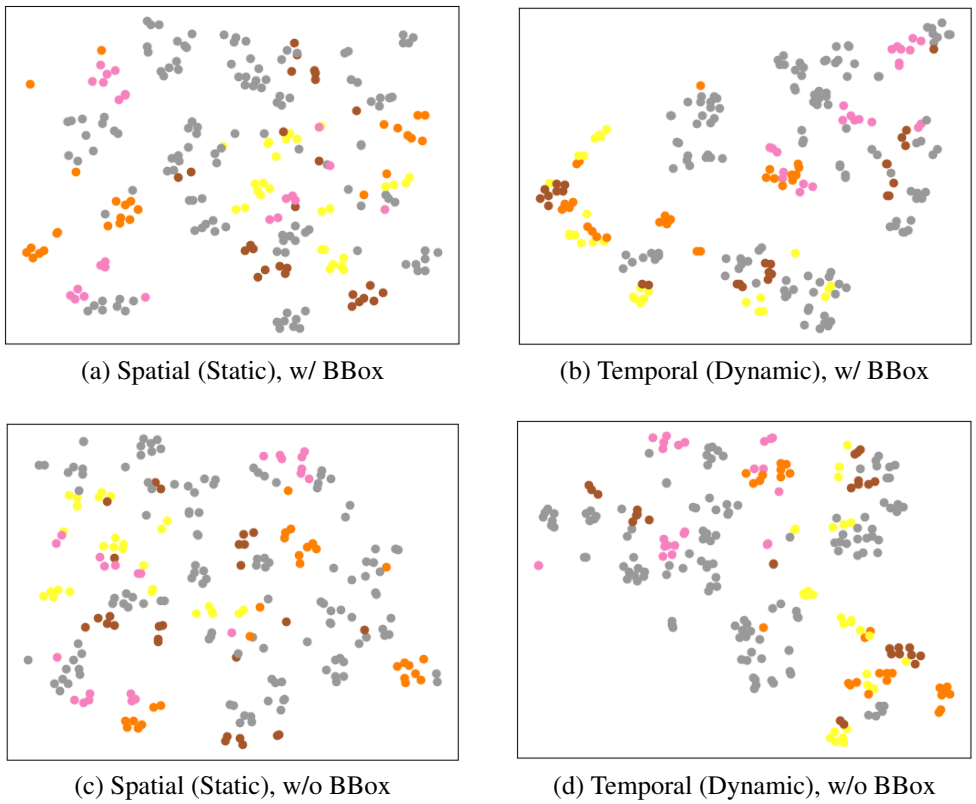


(e) Spatial (Static), w/o BBox



(f) Temporal (Dynamic), w/o BBox

Figure A.2: *Sequential raw depth inputs and point-level PACs on* SHREC'17. In PACs, the points in red have the highest activation values, while the blue ones are the lowest activating points. The animations of the above figures can be found in Figure 6 in the main body of our thesis.



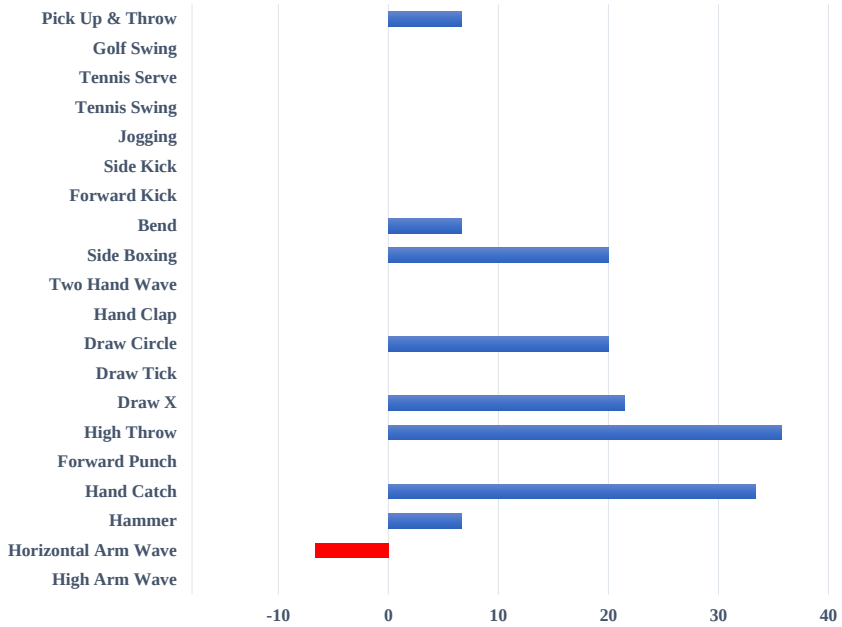
(a) Spatial (Static), w/ BBox

(b) Temporal (Dynamic), w/ BBox

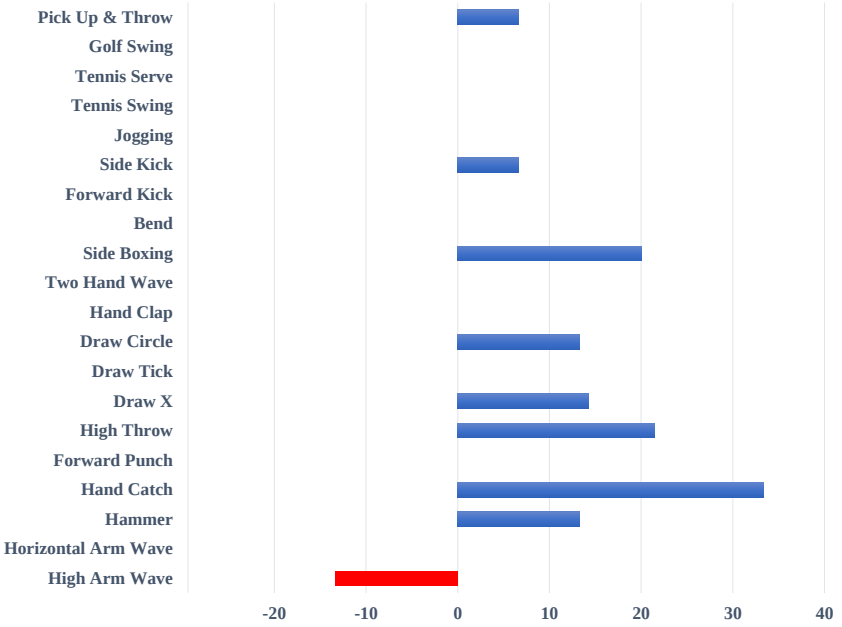
(c) Spatial (Static), w/o BBox

(d) Temporal (Dynamic), w/o BBox

Figure A.3: *Feature visualizations on* SHREC'17 *with video-level t-SNE.* For clarity, 8 out of the 28 classes are presented in the above figures. A video is visualized as a data point, of which the same color means the identical ground-truth category. The intra-class compactness and the inter-class separability reflect the representative ability of a model.



(a) +Ours (Kinet), Setting 2)



(b) +Scene Flow (Flow-based Two Streams), Setting 3)

Figure A.4: *Per-class accuracy gains (%) over the static model on 16-frame* MSRAction-3D. The blue bar indicates the postive change, whereas the red one represents the negative change.

B | Appendix of Intermediate-level Motion Analysis (Chapter 4)

In Chapter 4, we investigated the problem of discovering approximately rigid segments in dynamic point clouds. We introduce the problem formulation and objective, and present our algorithmic pipeline and key theoretical intuitions. This appendix provides (1) implementation details that were omitted for brevity, including initialization strategies and convergence criteria, and (2) proof sketches for the main propositions. We also include failure-case analyses and practical heuristics that improve robustness in challenging scenes.

B.1 Implementation Details of Network Structure

B.1.1 Per-point Feature Extractor

For an input frame P_k, the feature extractor of EPN outputs per-point SE(3)-equivariant representations $F_k \in \mathbb{R}^{N\times|\mathcal{G}|\times D}$. The corresponding feature $f_k^i \in \mathbb{R}^{|\mathcal{G}|\times D}$ of a point p_k^i can be viewed as a concatenation of different representations w.r.t. $g_j \in \mathcal{G}$ over the rotation group dimension:

$$f_k^i = [\theta(p_k^i, g_1), \theta(p_k^i, g_2), \cdots, \theta(p_k^i, g_{|\mathcal{G}|-1}), \theta(p_k^i, g_{|\mathcal{G}|})], \tag{B.1}$$

where θ is a D-dimensional encoder based on a stack of convolution kernels. The prediction module of vanilla EPN is designed for global SE(3)-equivariance for all input points. Differently, our unsupervised multi-body task requires the model's ability to handle *part-level local equivariance*, especially under low-quality training signals. For this purpose, we further devise two heads for rigid segmentation

and motion estimation. Figure B.1 demonstrates the detailed design of our EPN feature extractor on SAPIEN, OGC-DR, and OGC-DRSV, and that on KITTI-SF shares the same structure but has larger numbers of output dimensions accordingly.

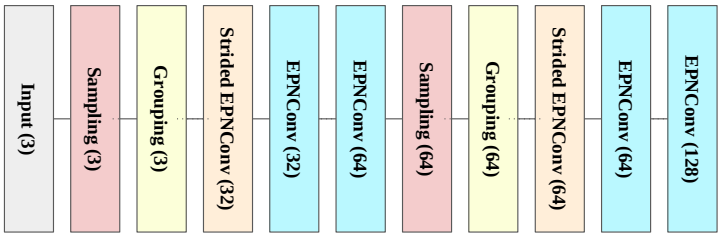


Figure B.1: *Structure of our feature extractor based on EPN.* The number in brackets denotes the output dimension of the corresponding convolution/operation. "EPNConv" is the SE(3)-equivariant convolution proposed in the vanilla EPN network.

B.1.2 Point-level Invariant Segmentation Head

Rigid segmentation is an SE(3)-invariant task, as the predicted mask should remain consistent for the same points across various poses and positions. Traditional SE(3)-equivariant structures assume that all input points undergo the same rigid transformation, which is not in alignment with the multi-body setting. To encode distinct transformations for individual input points, the equivariant features $\theta(p_k^i, g_j)$ are aggregated into an invariant representation u_k^i across the dimension of the rotation group:

$$u_k^i = \sum_j^{|\mathcal{G}|} w(p_k^i, g_j)\theta(p_k^i, g_j), \tag{B.2}$$

where $w(p_k^i, g_j) \in [0, 1]^{|\mathcal{G}|}$ is a selection probability of discrete rotations in $\mathcal{G}$, derived through a 1×1 convolution. The weighted sum u_k^i is invariant to rigid motion given a point with its neighbors in the convolution receptive field. By fusing such invariant representations $u_{k(1)}^i \in \mathbb{R}^D, \cdots, u_{k(h)}^i \in \mathbb{R}^{D'}$ from h layers in the feature

extractor, the segmentation head outputs a soft prediction $\hat{M}_k \in [0,1]^{N\times S}$ of the rigid mask. The multi-layer invariant representations are aggregated through an hourglass decoder, as shown in Figure B.2.

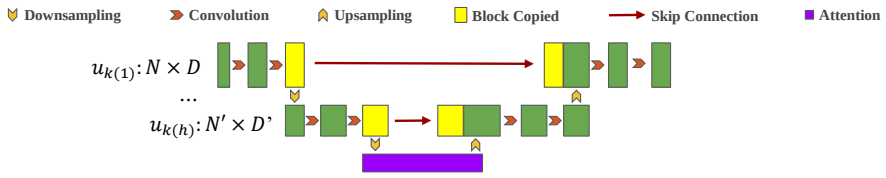


Figure B.2: *Structure of multi-layer aggregation.* N' and D' are the point number and the feature dimension of the last invariant layer, respectively.

B.1.3 Part-level Equivariant Motion Estimation Head

Part-level SE(3)-equivariance is desirable for motion analysis, especially rotation estimation. Based on the noisy predictions $(\hat{M}_k, \hat{M}_l)$ of the frames (k, l) from the head of rigid segmentation, the motion head is supposed to *handle these uncertain category-agnostic parts*. First of all, the part-level SE(3) feature $V_{k,j} \in \mathbb{R}^{N\times D\times S}$ w.r.t. the rotation $g_j \in \mathcal{G}$ of a single frame P_k is obtained from the per-point equivariant representations F_k and the predicted rigid mask $\hat{M}_k$:

$$V_{k,j} = \{\hat{m}_k^1 \cdot \theta(p_k^1, g_j), \hat{m}_k^2 \cdot \theta(p_k^2, g_j), \cdots, \hat{m}_k^{N-1} \cdot \theta(p_k^{N-1}, g_j), \hat{m}_k^N \cdot \theta(p_k^N, g_j)\}, \tag{B.3}$$

where $\hat{m}_k^i$ is the element corresponding to a point p_i^k in $\hat{M}_k$, and $\cdot$ is the broadcast operation of Hardamard product. Afterward, $V_{k,j}$ over the rotation group $\mathcal{G}$ is concatenated as $V_k \in \mathbb{R}^{N\times|\mathcal{G}|\times D\times S}$, followed by a permutation-invariant operation $\sigma : \mathbb{R}^{N\times|\mathcal{G}|\times D\times S} \to \mathbb{R}^{|\mathcal{G}|\times D\times S}$ (*e.g.*, max pooling) over all the points to produce part-level equivariant features $\tilde{V}_k = \sigma(V_k)$. Between two frames (k, l), the part-level rotation correlated feature $C_{kl} \in \mathbb{R}^{|\mathcal{G}|\times|\mathcal{G}|\times S}$ is defined as:

$$C_{kl} = \tilde{V}_k \tilde{V}_l^T, \tag{B.4}$$

where ^{T} is matrix transposition. C_{kl} is calculated upon "softly matching" within each consistent rigid part, while the specific category labels can be agnostic to the model. Based on the correlated feature C_{kl}, the motion head estimates rotation $\hat{\mathbf{R}}_{kl}^s$ and translation $\hat{\mathbf{t}}_{kl}^s$ of each rigid part s. To be specific, the rotation regression is implemented through a 1×1 convolution. Unlike the confidence-based selection in vanilla EPN for single-object pose regression, we choose the anchor g_{kl}^s from $\mathcal{G}$ by minimizing the registration error and then optimize the residual r_{kl}^s. In this case, the rotation is computed as:

$$\hat{\mathbf{R}}_{kl}^s = g_{kl}^s r_{kl}^s. \tag{B.5}$$

The translation can be derived from the minimal weighted distance between the transformed frame of P_k and the origin P_l:

$$\hat{\mathbf{t}}_{kl}^s = \underset{\mathbf{t}}{\operatorname{argmin}}\, d(\hat{\mathbf{R}}_{kl}^s P_k + \mathbf{t}, P_l), \tag{B.6}$$

where d is the chamfer loss weighted by the mask predictions.

B.2 Implementation Details of Unsupervised Training Strategy

B.2.1 Hyper-settings of Model Training

The training settings for the model include a total of 40 epochs and a batch size of 16. The learning rate is initially set to 2.0×10^{-4} and decays at a rate of 0.7 with a minimum clip value of 1.0×10^{-4}. The batch normalization momentum is set to

0.9, which controls the smoothing of the batch normalization statistics. The decay step is set to 2.0×10^5, which determines the frequency at which the learning rate decays.

B.2.2 Segmentation & Flow $\rightarrow$ Motion $\hat{\mathbf{T}}^s_{kl}$

Following previous works [39, 40], we employ the weighted-Kabsch algorithm [212, 213] to determine part-level rigid transformation $\hat{\mathbf{T}}^s_{kl}$ given the estimates of scene flow and rigid masks. Further, the motion head is optimized by the rotation component of $\hat{\mathbf{T}}^s_{kl}$, and estimates corresponding translation by minimizing our probability-based part-level distance as follows:

$$d^s_{prob} = d(\hat{\mathbf{T}}^s_{kl} \circ (\hat{M}^s_k P_k), \hat{M}^s_l P_l), \tag{B.7}$$

where d is the chamfer loss. $\hat{M}^s_k$ and $\hat{M}^s_l$ is the predicted rigid masks of the s^{th} part for the frame P_k and P_l, respectively.

B.3 Datasets & Metrics

B.3.1 Datasets

SAPIEN, as described by [105], provides a collection of 720 simulated articulated objects with annotations at the part instance level. Each object is represented by 4 sequential scans, with part instances exhibiting varying articulating states. Following the methodology of Huang *et al.* [39], we utilize the training data generated by Yi *et al.* [248]. Specifically, the dataset consists of 82092 pairs of point clouds for training and 2880 single point cloud frames for testing. Both training and testing point clouds are downsampled to 512 points.

OGC-DR, proposed by Song and Yang [40], is applicable to both scene flow estimation and object segmentation tasks. By adhering to the approach outlined by [249], they randomly positioned 4 to 8 objects from 7 classes of the ShapeNet dataset [3], including chairs, tables, lamps, sofas, cabinets, benches, and displays, within each room. A total of 3750 indoor rooms were generated for training purposes, with an additional 250 for validation and 1000 for testing. Rigid dynamics were introduced within each scene by applying continuous random transformations to each object and capturing 4 sequential frames for evaluation purposes. Each point cloud frame was subsequently downsampled to 2048 points. Song and Yang utilized the methodology proposed by Choy *et al*. [250] to partition different object instances across train/val/test sets.

OGC-DRSV, expanded upon the OGC-DR dataset by Song and Yang [40], is collected from single depth scans at each time step on the mesh models, designated as Single-View OGC-DR. Due to self- and/or mutual occlusions, all object point clouds within OGC-DRSV are severely incomplete, rendering this new dataset considerably more challenging than its predecessor, OGC-DR. Each point cloud frame within OGC-DRSV was also downsampled to 2048 points.

KITTI-SF comprises 200 pairs of point clouds from real-world traffic scenes for training purposes [104], as well as an online hidden test for scene flow estimation. Following [40], we trained our pipeline on the first 100 pairs of point clouds and subsequently tested it on the remaining 100 pairs (200 single point clouds). During the testing phase, only the human annotations of cars and trucks within each frame are retained for score computation. All other objects were considered part of the background. The entire background was not disregarded, but rather treated as a single object in our evaluation. Additionally, cars and trucks could be either static or dynamic.

B.3.2 Metrics

Rigid Segmentation. Average Precision (**AP**) is a measure that takes into account both precision and recall over all labels. It is calculated as the mean of the precision values at different recall levels. Panoptic Quality (**PQ**) is a measure proposed for evaluating panoptic segmentation, which takes into account both recognition and segmentation quality. F1-score (**F1**) is the harmonic mean of precision and recall. Precision (**Pre**) measures the proportion of true positive instances among the instances that were predicted as positive by the model, while Recall (**Rec**) measures the proportion of true positive instances that were correctly identified by the model. Mean Intersection over Union (**mIoU**) is a measure used to evaluate semantic segmentation models, which calculates the average IoU between the predicted and ground truth segmentation masks for each class. Rand Index (**RI**) is a measure of similarity between two data clusterings, which can be used to evaluate the performance of a segmentation model by comparing its predictions with the ground truth masks.

Motion Estimation. End-Point-Error 3D (**EPE3D**) measures the average error over all warped points under the estimated and the ground-truth warping functions. Accuracy Strict (**AccS**) and Accuracy Relaxed (**AccR**) refer to the ratio of points where the EPE3D or relative error is below a certain threshold. Specifically, AccS represents the ratio of points where the EPE3D is less than 0.05 or the relative error is less than 0.05, while AccR represents the ratio of points where the EPE3D is less than 0.1, or the relative error is less than 0.1. Outlier (**Outl**) refers to the ratio of points where the EPE3D is greater than 0.3 or the relative error is greater than 0.1.

B.4 Rigid Motion Estimation Performance: More Metrics & More Datasets

Figure B.3 shows the results of the motion estimation experiments on SAPIEN. The fully-supervised method outperforms the unsupervised method in terms of all of the metrics. The fully-supervised method achieved an EPE3D of 3.86, an AccS of 42.75%, an AccR of 63.58%, and an Outlier rate of 56.12% while the unsupervised method achieved an EPE3D of 5.47, an AccS of 32.76%, an AccR of 52.81% and an Outlier rate of 66.59%.

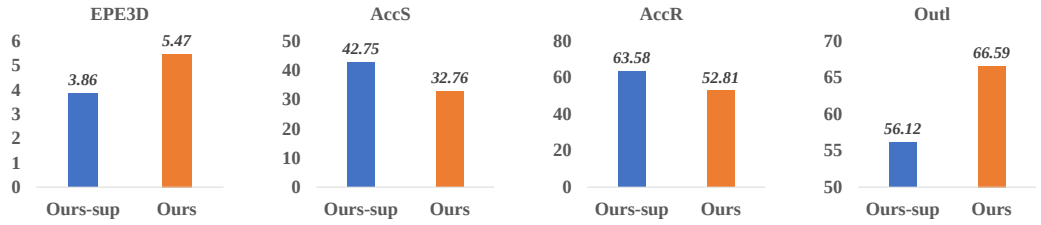


Figure B.3: *Motion estimation results on SAPIEN.*

Figure B.4 demonstrates the results of the motion estimation experiments on OGC-DR. Intriguingly, the unsupervised performance is marginally better than the fully-supervised case on this dataset, possibly because full supervision constrains the generalizability of the model of motion estimation. The fully-supervised method achieved an EPE3D of 0.91, an AccS of 67.68%, an AccR of 89.11% and an Outlier rate of 57.38, while the unsupervised method achieved an EPE3D of 0.73, an AccS of 76.38%, an AccR of 92.96% and an Outlier rate of 45.20%.

As depicted in Figure B.5, the motion estimation experiments conducted on OGC-DRSV demonstrate that the fully-supervised method surpasses the unsupervised method. The fully-supervised method attained an EPE3D of 0.87, an AccS of 75.49%, an AccR of 92.91%, and an Outlier rate of 59.43%. In contrast, the unsupervised method attained an EPE3D of 1.45, an AccS of 61.27%, an AccR of

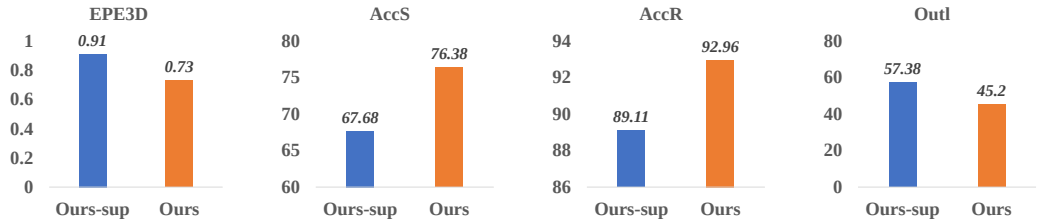


Figure B.4: *Motion estimation results on OGC-DR.*

81.82%, and an Outlier rate of 65.39%.

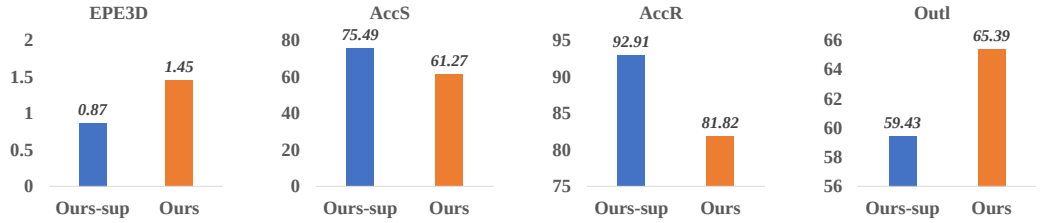


Figure B.5: *Motion estimation results on OGC-DRSV.*

B.5 Qualitative Results

Figure B.6 presents the qualitative results of our model. In general, the predictions generated by the model are satisfactory. However, errors are primarily observed in the segmentation boundaries. For instance, the border between the tap handle and the faucet spout in the 2^{nd} picture is one such area where errors occur.

We also provide additional visualizations in Figure B.7: 1) challenging scenes (multiple parts) on SAPIEN, with successful and failure cases (errors as circled in green on the final two rows); 2) visualizations on all of the other datasets (OGC-DR, OGC-DRSV, and KITTI-SF). All of the results ("Ours") are obtained from our model in the unsupervised setting, while "GT" means the ground-truth segmentation.

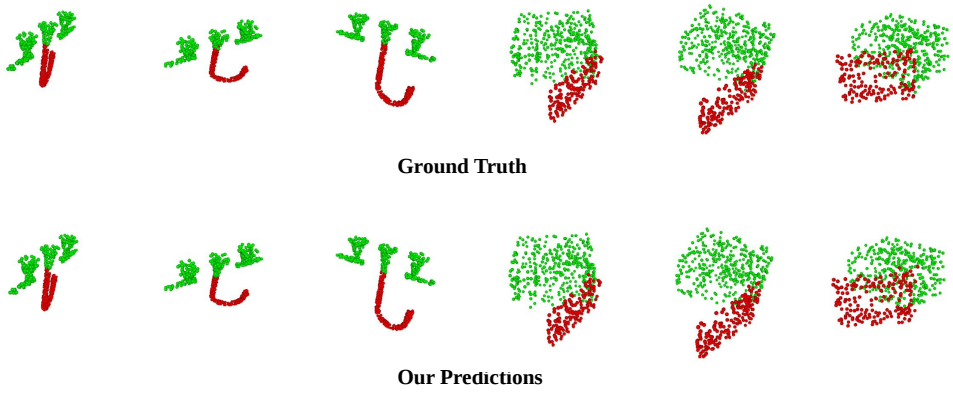


Figure B.6: *Visualizations for rigid segmentation results on SAPIEN.*

B.6 Parameter Number and Computational Complexity

Table B.1 provides a quantitative comparison of various supervised and unsupervised methods in terms of parameter number, FLOPs, and Average Precision on the SAPIEN dataset. The results demonstrate that the proposed supervised method (Ours-sup) achieves the highest Average Precision of 73.5 while utilizing the fewest parameters (0.25M) and FLOPs (0.92G). Similarly, the proposed unsupervised method (Ours) outperforms the OGC unsupervised method [7] with respect to Average Precision (63.8 *vs*. 55.6) while requiring fewer FLOPs (3.71G *vs*. 4.09G).

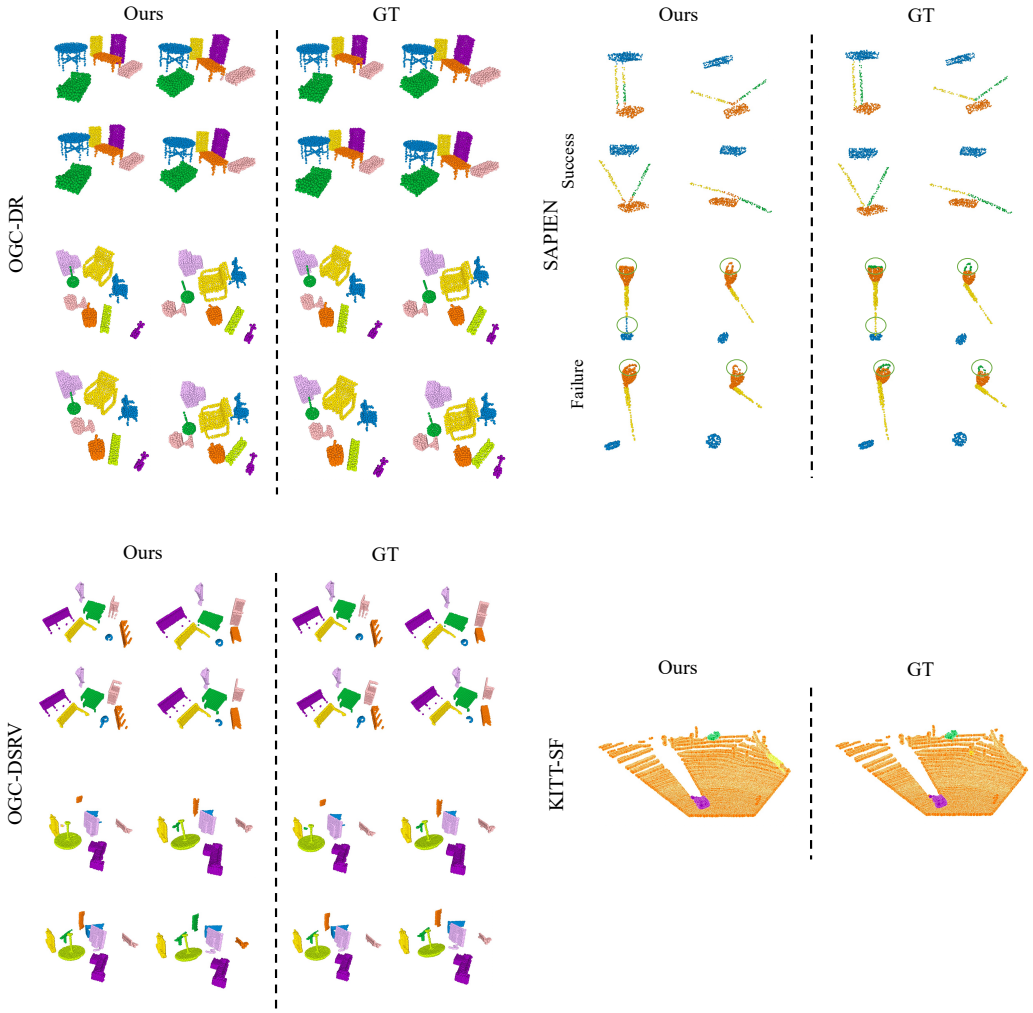


Figure B.7: *Additional visualizations for challenging scenes and more datasets.*

Table B.1: *Quantitative results of parameter number, FLOPs, and Average Precision (AP) on SAPIEN.* In this table, * indicates that we evaluate its AP upon the officially trained model.

		FLOPs (G)↓	**#PARAMS (M)↓**	**AP (%)↑**
Supervised Methods	MBS [39]	608.73	17.15	49.4*
	OGC-sup [40]	4.09	0.43	66.1
	Ours-sup	**0.92**	**0.25**	**73.5**
Unsupervised Methods	OGC [40]	4.09	0.43	55.6
	Ours	**3.71**	**0.31**	**63.8**

C | Appendix of Low-level Geometric Reconstruction (Chapter 5)

Chapter 5 addressed the estimation of segment-wise motions and their temporal consistency. We describe the motion parameterization and optimization, and report quantitative and qualitative results. This appendix also elaborates our implementation of the estimation pipeline, including parameter initialization, regularization terms, scheduling, and stopping conditions. We further specify the evaluation metrics, per-sequence diagnostics, and supplemental visualizations.

C.1 Dataset and Implementation Details

C.1.1 ARTEMIS-DFA Benchmark Dataset

The ARTEMIS-DFA benchmark is curated from the ARTEMIS Dynamic Figurative Animation dataset [226], which consists of high-quality 3D CGI models of animals created by professional artists. As illustrated in Figure C.1, our benchmark provides a diverse range of challenging scenarios for video-to-4D generation. Each animal subject performs multiple distinct motion patterns - for example, the bear exhibits both a walking sequence and a standing-up motion, the duck demonstrates different types of waddling gaits, and the tiger performs both a standard walking motion and a running/jumping sequence. This diversity of motion types for the same subject creates a more challenging evaluation scenario than datasets with only a single motion pattern per object. Our benchmark includes:

- **Number of animal species**: 9 (Beagle Dog, Bear, Cat, Duck, Fox, Lion, Panda, White Tiger, Wolf)

- **Motion sequences per species**: 2-4 distinct patterns (*e.g.*, walking, running, jumping, specific behaviors)
- **Total sequences**: 25 distinct video sequences
- **Frames per sequence**: 32-120 frames, with 32 frames used for quantitative evaluation
- **Camera viewpoints**: 5 total viewpoints (1 reference view for training, 4 views for testing)
- **Resolution**: 512×512 pixels

The dataset is specifically designed to present challenging scenarios with:

- Substantial object displacement across frames
- Complex articulated motion patterns
- Multiple distinct motion types for the same object
- Varying degrees of non-rigid deformation

Unlike existing benchmarks that often feature limited motion, our dataset provides a more comprehensive evaluation platform for video-to-4D generation methods.

C.1.2 Implementation Details

Our implementation is built upon the PyTorch [251] framework with the following specifics:

- **Network Architecture**: We use Xavier uniform initialization for most weights, with specialized initialization for output layers: zero initialization for translation outputs and identity quaternion initialization for rotation outputs. As illustrated in Figure C.2, our MotionStruct4D architecture consists of the

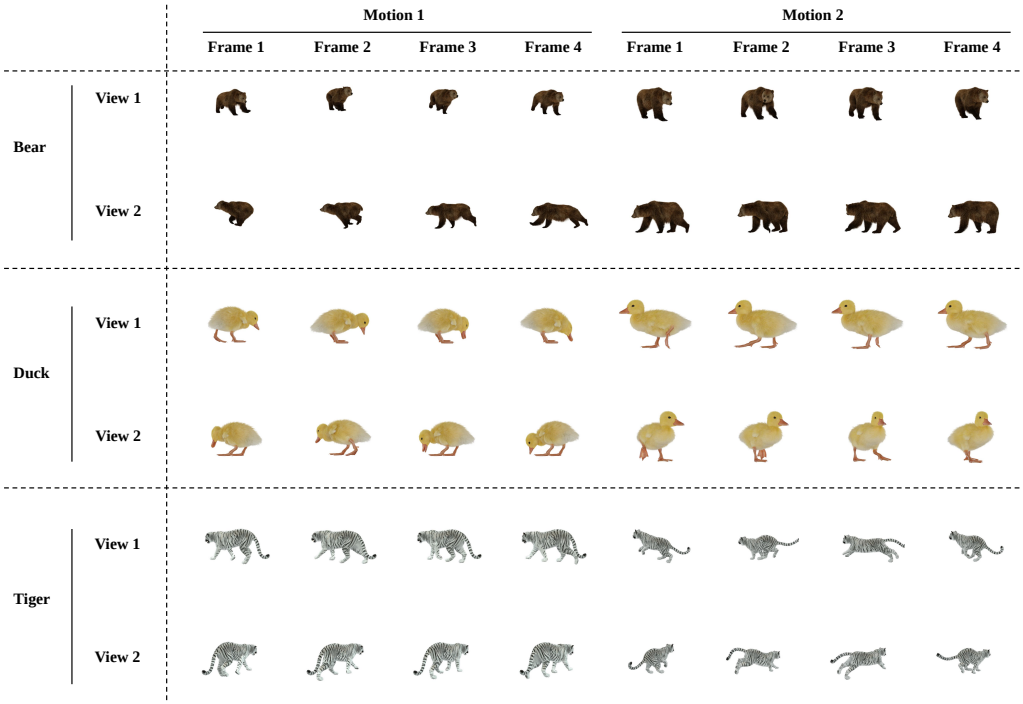


Figure C.1: *Samples from our ARTEMIS-DFA Benchmark showing different animals (Bear, Duck, Tiger) performing distinct motion patterns.* For each animal, we provide two different types of motion (walking, standing up, running, *etc*.) captured from two viewpoints across four frames, illustrating the diversity and complexity of articulated movements in our dataset.

following components:

- **Input and Embedding**: The system processes 3D Point Positions and Time Parameter t through separate embedding layers:
  - Point Positional Encoding with 10 frequencies
  - Time Positional Encoding with 6 frequencies

  These embeddings are concatenated to form a unified representation.
- **Canonical Gaussian Representation**: This branch maintains the representation of:

* Gaussian Positions

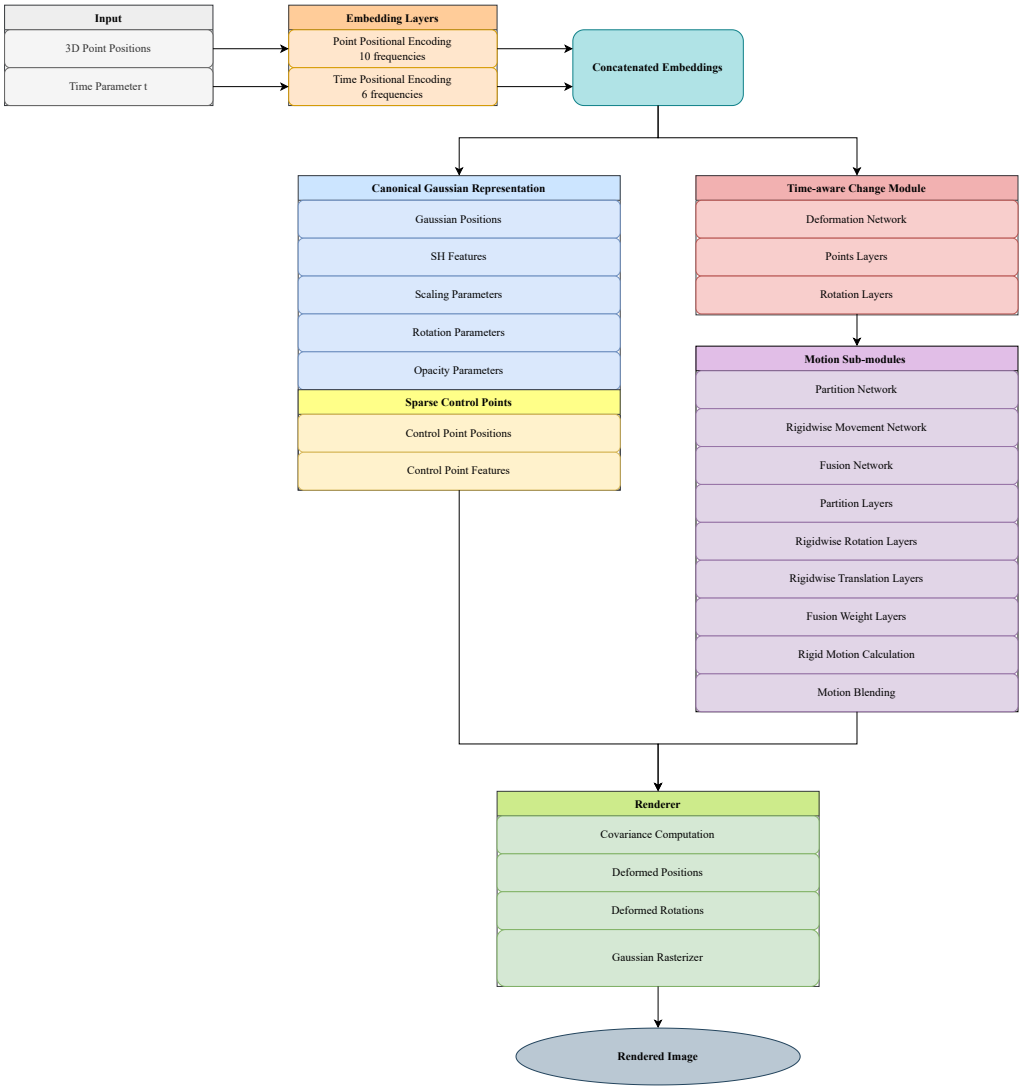


Figure C.2: *Network architecture of MotionStruct4D.* Our framework processes 3D point positions and time parameters through embedding layers, which feed into two parallel branches: (1) a canonical Gaussian representation branch that maintains the static 3D structure, and (2) a time-aware change module that computes deformations. The motion sub-modules decompose the deformation into rigid and non-rigid components through partitioning, rigid motion estimation, and motion blending. The final renderer combines all components to produce time-varying 3D content.

    - SH Features
    - Scaling Parameters
    - Rotation Parameters
    - Opacity Parameters
    - Sparse Control Points (including Control Point Positions and Features)
  - **Time-aware Change Module**: This branch processes time-dependent deformations with:
    - Deformation Network
    - Points Layers
    - Rotation Layers
  - **Motion Sub-modules**: These specialized components include:
    - Partition Network
    - Rigidwise Movement Network
    - Fusion Network
    - Partition Layers
    - Rigidwise Rotation Layers
    - Rigidwise Translation Layers
    - Fusion Weight Layers
    - Rigid Motion Calculation
    - Motion Blending

  - **Renderer**: The final stage that processes:
    * Covariance Computation
    * Deformed Positions
    * Deformed Rotations
    * Gaussian Rasterizer to produce the Rendered Image

- **Learning Rate Schedule**: We use the Adam optimizer with separate learning rates (LR) for different components:
  - Canonical Gaussian positions: Initial LR = 0.01, final LR = 0.0002
  - Gaussian opacity: LR = 0.05
  - Gaussian scaling: LR = 0.005
  - Gaussian rotation: LR = 0.005
  - Gaussian features: LR = 0.01
  - Control radius: LR = 0.005
  - Non-rigid deformation: Initial LR = 0.0002, final LR = 0.000002
  - Control points: Initial LR = 0.000002, final LR = 0.000002
  - Rigid motion parameters: LR = 0.01
- **Miscellaneous Hyper-parameters**:
  - Number of sparse control points: 512
  - Training batch size: 4
  - Background inversion probability: 0.5 (for data augmentation)

C.1.3 Computational Resources

To provide complete reproducibility and enable fair comparison with future methods, we detail the computational environment and performance characteristics of our approach. All experiments were conducted using the following hardware and software environment:

- **Hardware**:
  - GPU: 4 × NVIDIA A10 (24GB)
  - CPU: Intel(R) Xeon(R) Silver 4314 (32 cores)
  - RAM: 256GB DDR4
- **Software**:
  - PyTorch 2.0.1
  - CUDA 11.7
  - Python 3.9
- **Memory Requirements**:
  - Peak GPU memory usage during training: ~18GB

C.1.4 Evaluation Metrics

We employ a comprehensive set of metrics to evaluate the quality of video-to-4D generation across three key aspects: reference view alignment, spatio-temporal consistency, and motion fidelity.

- **Reference View Alignment Metrics**:
  - **PSNR** (Peak Signal-to-Noise Ratio) [252]: Measures the pixel-level

reconstruction quality between the rendered image and the ground truth input frame. Higher values indicate better reconstruction quality. PSNR is computed as:

$$\text{PSNR} = 10 \cdot \log_{10} \left(\frac{\text{MAX}_I^2}{\text{MSE}} \right) \tag{C.1}$$

where MAX_I is the maximum possible pixel value (255 for 8-bit images) and MSE is the mean squared error between the rendered and ground truth images.

- **SSIM** (Structural Similarity Index) [253]: Evaluates the structural similarity between rendered and ground truth images, considering luminance, contrast, and structure. SSIM ranges from 0 to 1, with higher values indicating better perceptual quality.

- **Perceptual Quality Metrics**:
  - **LPIPS** (Learned Perceptual Image Patch Similarity) [254]: A perceptual metric that uses deep features to measure similarity between images. Lower values indicate higher perceptual quality. LPIPS better aligns with human perception of image quality compared to pixel-based metrics.
  - **CLIP-Score** [255]: Measures semantic similarity between rendered images from different viewpoints using CLIP embeddings. Higher values indicate better multi-view consistency and semantic fidelity. We compute the average cosine similarity between CLIP embeddings of rendered images across different viewpoints.

- **Temporal Consistency and Motion Fidelity Metrics**:
  - **FVD** (Fréchet Video Distance) [256]: Evaluates the distribution dis-

tance between generated and reference video frames in feature space. Lower values indicate better temporal consistency and motion fidelity.

- **FID-VID** [257]: An adaptation of Fréchet Inception Distance for video data, measuring the distribution similarity between generated and reference videos. Lower values indicate better temporal quality.

Algorithm 1 Batched Weighted Kabsch Algorithm

function WEIGHTEDKABSCH($\boldsymbol{\mu}$, $\Delta\hat{\boldsymbol{\mu}}^{\tau}$, $\mathbf{m}^{\tau}$)
  Input: Canonical positions $\boldsymbol{\mu} \in \mathbb{R}^{B\times N\times 3}$, estimated flows $\Delta\hat{\boldsymbol{\mu}}^{\tau} \in \mathbb{R}^{B\times N\times 3}$, and soft assignments $\mathbf{m}^{\tau} \in \mathbb{R}^{B\times N\times S}$
  Output: Rigid-part driven flow $\Delta\boldsymbol{\mu}^{\tau} \in \mathbb{R}^{B\times N\times 3}$
  $B, N, S \leftarrow \mathbf{m}^{\tau}.size()$
  $\mathbf{m}^{\tau} \leftarrow \mathbf{m}^{\tau}.transpose(1,2).reshape(B\times S, N)$
  $\boldsymbol{\mu}_{rep} \leftarrow \boldsymbol{\mu}.unsqueeze(1).repeat(1,S,1,1).reshape(B\times S, N, 3)$
  $\Delta\hat{\boldsymbol{\mu}}^{\tau}_{rep} \leftarrow \Delta\hat{\boldsymbol{\mu}}^{\tau}.unsqueeze(1).repeat(1,S,1,1).reshape(B\times S, N, 3)$
  $\boldsymbol{R}^{\tau}_{s}, \boldsymbol{t}^{\tau}_{s} \leftarrow$ FITMOTIONSVDBATCH($\boldsymbol{\mu}_{rep}$, $\boldsymbol{\mu}_{rep} + \Delta\hat{\boldsymbol{\mu}}^{\tau}_{rep}$, $\mathbf{m}^{\tau}$)
  $\boldsymbol{\mu}^{\tau}_{s} \leftarrow \boldsymbol{R}^{\tau}_{s} \cdot \boldsymbol{\mu}_{rep} + \boldsymbol{t}^{\tau}_{s}$ ▷ Apply rigid transformation
  $\boldsymbol{\mu}^{\tau}_{s} \leftarrow \boldsymbol{\mu}^{\tau}_{s}.reshape(B, S, N, 3)$
  $\mathbf{m}^{\tau} \leftarrow \mathbf{m}^{\tau}.reshape(B, S, N)$
  $\Delta\boldsymbol{\mu}^{\tau} \leftarrow \sum_{s=1}^{S} \mathbf{m}^{\tau}_{s} \cdot \boldsymbol{\mu}^{\tau}_{s} - \boldsymbol{\mu}$ ▷ Weighted combination
  return $\Delta\boldsymbol{\mu}^{\tau}$
end function

C.2 Additional Method Details

C.2.1 Differentiable Weighted-Kabsch Algorithm

For identifying rigid motion components, we employ a differentiable weighted-Kabsch algorithm [212, 213] to estimate the transformation matrix $\boldsymbol{T}^{\tau}_{s} \in SE(3)$ for each rigid part s. The algorithm computes the optimal rigid transformation between two point sets while accounting for the soft assignment of points to rigid parts.

Given the canonical Gaussian positions $\{\boldsymbol{\mu}_i\}$ and their estimated positions at timestep τ denoted as $\{\boldsymbol{\mu}_i + \Delta\hat{\boldsymbol{\mu}}_i^\tau\}$, along with the soft assignment probabilities $\{m_{is}^\tau\}$ for rigid part s, the weighted-Kabsch algorithm proceeds as follows:

1. Compute the weighted centroids of both point sets:

$$\bar{\boldsymbol{\mu}}_s = \frac{\sum_i m_{is}^\tau \boldsymbol{\mu}_i}{\sum_i m_{is}^\tau}, \quad \bar{\boldsymbol{\mu}}_s^\tau = \frac{\sum_i m_{is}^\tau (\boldsymbol{\mu}_i + \Delta\hat{\boldsymbol{\mu}}_i^\tau)}{\sum_i m_{is}^\tau} \tag{C.2}$$

2. Compute the centered coordinates:

$$\hat{\boldsymbol{\mu}}_i = \boldsymbol{\mu}_i - \bar{\boldsymbol{\mu}}_s, \quad \hat{\boldsymbol{\mu}}_i^\tau = (\boldsymbol{\mu}_i + \Delta\hat{\boldsymbol{\mu}}_i^\tau) - \bar{\boldsymbol{\mu}}_s^\tau \tag{C.3}$$

3. Compute the weighted covariance matrix:

$$\boldsymbol{H} = \sum_i m_{is}^\tau \hat{\boldsymbol{\mu}}_i^\tau (\hat{\boldsymbol{\mu}}_i)^T \tag{C.4}$$

4. Compute the singular value decomposition (SVD):

$$\boldsymbol{H} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T \tag{C.5}$$

5. Compute the rotation matrix, ensuring a proper rotation:

$$\boldsymbol{R}_s^\tau = \boldsymbol{U} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det(\boldsymbol{U}\boldsymbol{V}^T) \end{pmatrix} \boldsymbol{V}^T \tag{C.6}$$

6. Compute the translation vector:

$$\boldsymbol{t}_s^\tau = \bar{\boldsymbol{\mu}}_s^\tau - \boldsymbol{R}_s^\tau \bar{\boldsymbol{\mu}}_s \tag{C.7}$$

The resulting transformation matrix $\boldsymbol{T}_s^\tau$ combines the rotation $\boldsymbol{R}_s^\tau$ and translation $\boldsymbol{t}_s^\tau$. This algorithm is fully differentiable, allowing gradients to flow back to update the soft assignment probabilities $\{m_{is}^\tau\}$ during training.

In our implementation, as described in Alg. 1, we use a batched version of this algorithm that handles multiple rigid parts simultaneously. The FITMOTIONSVD-BATCH function computes the optimal rotation and translation for each rigid part using SVD decomposition of the weighted covariance matrix. This implementation efficiently handles multiple rigid parts in parallel and provides a differentiable operation that can be integrated into the training pipeline.

C.2.2 Reconstruction Loss

To ensure that our generated 4D content accurately aligns with the input video frames, we employ a direct supervision through a reconstruction loss. For each frame I_τ in the input video $\mathcal{I} = \{I_\tau\}_{\tau=1}^{T}$, we render the corresponding view from our time-dependent Gaussians $\{\mathcal{G}_i^\tau\}_{i=1}^{N}$ at timestep τ.

The main term of our reconstruction loss $\mathcal{L}_{\text{recon-ref}}$ is computed as the L2 distance between the rendered image $\hat{I}_\tau$ and the ground truth input frame I_τ:

$$\mathcal{L}_{\text{recon-ref}} = \frac{1}{T}\sum_{\tau=1}^{T}\|\hat{I}_\tau - I_\tau\|_2^2 \tag{C.8}$$

To further enhance the reconstruction quality, we also incorporate a foreground loss to better preserve the foreground object silhouette:

$$\mathcal{L}_{\text{foreground}} = \frac{1}{T}\sum_{\tau=1}^{T}\|\alpha_\tau - \mathbf{F}_\tau\|_2^2 \tag{C.9}$$

where $\mathbf{F}_\tau$ is the foreground mask of the input frame at timestep τ, and α_τ is the accumulated opacity obtained during rendering. The foreground loss helps to maintain the precise shape boundary of the dynamic object across time.

The final reconstruction objective $\mathcal{L}_{\text{recon}}$ combines both the pixel-wise reconstruc-

tion and the mask loss:

$$\mathcal{L}_{\text{recon}} = \mathcal{L}_{\text{recon-ref}} + \lambda_{\text{foreground}} \mathcal{L}_{\text{foreground}} \tag{C.10}$$

where $\lambda_{\text{foreground}}$ is a weighting factor to balance the two loss terms. This combined loss ensures that both visual appearance and object silhouette are preserved in the reconstructed frames.

C.2.3 Augmentation Loss Using Score Distillation Sampling (SDS)

Following prior works [66, 81] in video-to-4D generation, we leverage view-conditioned diffusion models to generate multi-view content for improved 3D supervision. Specifically, we employ Score Distillation Sampling [122] (SDS) to compute the augmentation loss $\mathcal{L}_{\text{aug}}$ as a component of our objective function.

For each input frame I_τ at timestep τ, we maintain the 3D consistency using a pretrained diffusion model ϕ (*e.g.*, Zero123 [96]). Given the reference image I_τ, relative camera extrinsics $(\boldsymbol{R}_v, \boldsymbol{T}_v)$ between the queried novel view v and input view, and our 4D representation parameterized by θ, the augmentation loss is computed as:

$$\nabla_\theta \mathcal{L}_{\text{aug}} = \mathbb{E}_{v,\tau,\epsilon,t} \left[w(t) \left(\epsilon_\phi(z_t; I_\tau, \boldsymbol{R}_v, \boldsymbol{T}_v, t) - \epsilon \right) \frac{\partial x}{\partial \theta} \right] \tag{C.11}$$

where z_t is the noisy latent at diffusion timestep t, x is the rendered image from our 4D model at novel viewpoint v, ϵ is the sampled noise, ϵ_ϕ is the denoising network of the diffusion model, and $w(t)$ is a weighting function dependent on the timestep. This formulation guides our 4D representation to produce renderings that are consistent with the multi-view generation capabilities of the pretrained

diffusion model, effectively transferring prior knowledge of 3D geometry and appearance from the diffusion model to our representation.

C.2.4 Details of Sparse Control Points

C.2.4.1 Weighting Ratio

In the sparse control-point optimization phase, we utilize a weighting ratio ω_{ik} to determine the influence of each control point $\mathcal{P}_k$ on a Gaussian $\mathcal{G}_i$. This ratio depends on the distance d_{ik} between the Gaussian center $\boldsymbol{\mu}_i$ and the control point $\boldsymbol{p}_k$, as well as a learned control radius γ_k for each control point.

The weighting ratio is computed as:

$$\omega_{ik} = \frac{\exp(-d_{ik}^2/\gamma_k^2)}{\sum_{j \in M_i} \exp(-d_{ij}^2/\gamma_j^2)} \tag{C.12}$$

where M_i represents the set of K-nearest neighboring control points for Gaussian $\mathcal{G}_i$. We set $K = 8$ in our implementation. The learned control radius γ_k allows each control point to adaptively determine its area of influence, with some control points specializing in detailed local movements (smaller radius) and others governing broader regional transformations (larger radius).

During training, we initialize the control points using farthest point sampling (FPS) on the canonical Gaussian positions to ensure uniform coverage of the object, with initial control radii set proportionally to the average distance between neighboring points.

C.2.5 Regularization Term for Sparse Control Points

To maintain shape fidelity and prevent control point drift during optimization, we add the regularization term $\mathcal{L}_{\text{control}}$ to our loss function. At the beginning of the sparse control-point optimization stage, we preserve the positions of control points from the dense stage as a reference, following Wu *et al*. [81]. For a given timestep τ, we obtain the initial positions (denoted as $\overline{\boldsymbol{p}}^{\tau}$) of those control points. Then, we compute the regularization term for sparse control points as:

$$\mathcal{L}_{\text{control}} = \|\boldsymbol{p}^{\tau} - \overline{\boldsymbol{p}}^{\tau}\|_2^2 \tag{C.13}$$

where $\boldsymbol{p}^{\tau}$ denotes the current position of control points at timestep τ. Although this loss formulation is simple, it effectively mitigates shape degeneration issues during the optimization process.

C.3 Additional Results on Consistent4D Dataset

In addition to our ARTEMIS-DFA benchmark, we conduct experiments on the Consistent4D [63] dataset to demonstrate generalization to generic objects. The Consistent4D dataset includes 8 objects with 15 video sequences. Figure C.3 shows qualitative results on the Consistent4D dataset. Our method successfully discovers meaningful motion decomposition across diverse subjects. The quasi-rigid parts visualization (row 2) demonstrates how MotionStruct4D automatically segments the animals into intuitive rigid components - for instance, separating the crocodile's body, tail, and limbs. The rigid-motion weights (row 3) highlight regions where rigid transformation dominates the motion, versus areas requiring non-rigid deformation. Both reference-view reconstructions and novel-view generations (rows 4-5) show high fidelity to the input while maintaining temporal

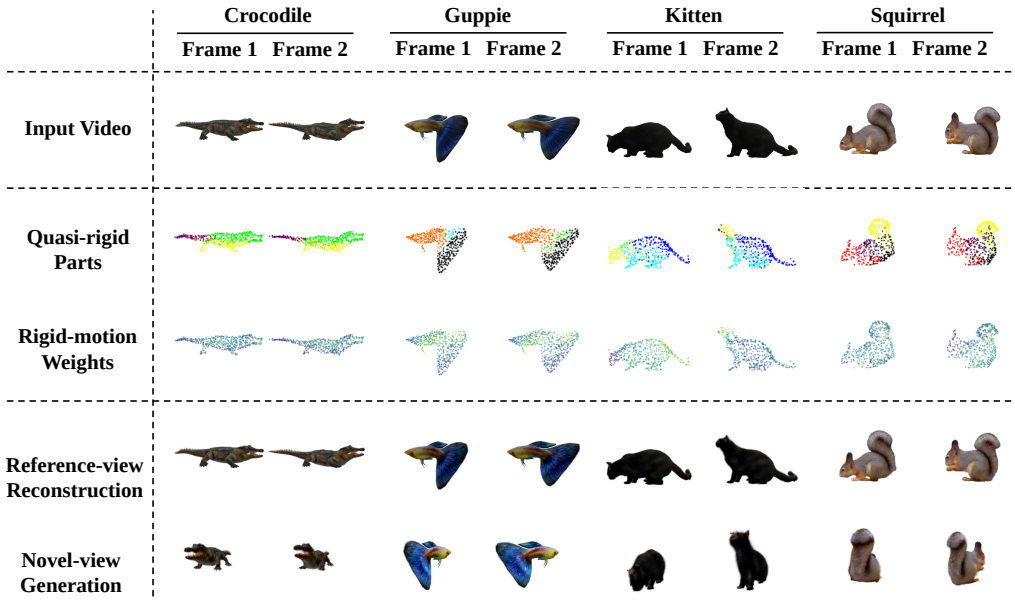


Figure C.3: *Visualization of MotionStruct4D results on the Consistent4D dataset.* For each animal (Crocodile, Guppie, Kitten, and Squirrel), we show: input video frames, discovered quasi-rigid part decomposition (colored by part assignment), rigid-motion weights (indicating areas that follow rigid motion patterns), reference-view reconstructions, and novel-view generations across two frames.

consistency. These results demonstrate that our approach generalizes well beyond the ARTEMIS-DFA benchmark to handle the natural movements of various animals in the Consistent4D dataset.

C.4 Broader Impacts

C.4.1 Potential Positive Societal Impacts

The development of MotionStruct4D and similar video-to-4D generation methods offers several potential positive impacts:

- **Content Creation Democratization**: By enabling the generation of dynamic 3D content from ordinary monocular videos, our technology democ-

ratizes content creation, allowing individuals and small studios with limited resources to produce high-quality 4D assets for applications in education, entertainment, and communication.

- **Cultural Preservation**: The ability to transform archival footage into interactive 3D assets facilitates the preservation and exploration of cultural heritage, historical events, and natural phenomena in more immersive formats.
- **Scientific Visualization**: Our approach can assist researchers in visualizing complex dynamic systems from limited observations, potentially accelerating discoveries in fields like biology, physics, and environmental science.
- **Accessibility**: 4D content generation from videos can enhance accessibility for individuals with disabilities by converting existing 2D content into more interactive, spatial formats that can be experienced through various assistive technologies.
- **Reduced Environmental Impact**: Creating 4D assets from existing videos requires significantly less computational resources compared to traditional CGI pipeline or multi-camera studio setups, potentially reducing the carbon footprint of content production.

C.4.2 Potential Negative Societal Impacts and Mitigation Strategies

While our research aims to advance the field of computer vision and graphics, we acknowledge several potential concerns:

- **Deepfake Concerns**: Enhanced capabilities for generating realistic 4D content could be misused to create convincing deepfakes. To mitigate this risk,

we focus our research on non-human subjects (animals and objects) and encourage the development of detection methods for synthesized content.

- **Copyright and Intellectual Property Issues**: The ability to convert videos into manipulable 3D assets raises questions about copyright and intellectual property rights. We emphasize that our technology should be used in compliance with existing copyright laws and with appropriate attribution to content creators.
- **Bias and Representation**: As with many AI systems, biases in training data could lead to unequal performance across different subjects or scenarios. We actively work to ensure our benchmark dataset includes diverse subjects and motion patterns to minimize such biases.
- **Resource Inequality**: Advanced computational requirements might limit access to these technologies for under-resourced communities. We are committed to optimizing our methods for computational efficiency and exploring lighter-weight variants that can run on consumer hardware.

We believe that the benefits of this research outweigh the potential risks, especially when deployed responsibly with appropriate safeguards. We encourage ongoing dialogue among researchers, policymakers, and the public regarding the ethical implications of 4D content generation technologies.