THE UNIVERSITY OF SYDNEY

DOCTORAL THESIS

Deep Learning-based Methods For Sequential Point Cloud Upsampling

Author:
Kaisiyuan WANG

Supervisor:
Prof. Luping ZHOU
Co-Supervisor:
A/Prof. Wanli OUYANG

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy

in the

School of Electrical and Information Engineering
Faculty of Engineering

November 8, 2023

Abstract of thesis entitled

Deep Learning-based Methods For Sequential Point Cloud Upsampling

Submitted by

Kaisiyuan WANG

for the degree of Doctor of Philosophy

at The University of Sydney

in November, 2023

Point cloud is a fundamental 3D representation that has received increasing attention from the computer vision community due to the recent popularity of Light Detection and Ranging (LIDAR) sensors and RGB-D cameras. However, analyzing and understanding 3D point clouds with deep networks is challenging since the raw point cloud data captured by scanning devices always suffers from sparsity and irregular data structure. To cope with this problem, several studies have been proposed to perform restoration by upsampling the input sparse point clouds via deep neural networks (DNN). Despite the successes achieved by these studies, they cannot generate satisfying results due to the insufficient geometry contexts from the input single point cloud snapshot. Considering that the 3D sensors usually capture point cloud sequences instead of each single point cloud snapshot, in this thesis, I propose three sequential point cloud sampling algorithms to significantly improve the performance of the previous single frame-based methods by exploiting temporal dependencies.

In the first method, I take inspiration from recent video super-resolution approaches to propose a recurrent framework for sequential point cloud upsampling (SPU), which leverages multi-scale long-/short-term temporal dependencies to recover the point cloud of the current

frame. Different kinds of temporal dependencies provide complementary geometric information and implicit constraints for temporal coherence. To effectively incorporate these temporal dependencies, I propose a temporal alignment module and a gating mechanism for adaptive feature enhancement.

Considering that our first method is quite limited in efficiency and generalizability, in the second method, I attempt to improve the efficiency and generalizability by extending the sequential point cloud upsampling framework to a patch-based version, namely the VPU framework. To do so, I first propose a new patch-cropping strategy compatible with the temporal demand and then carefully design a new spatio-temporal aggregation module to effectively extract, align and aggregate the rich local geometric clues from consecutive frames at the feature level. By more reliably summarizing spatio-temporally consistent and complementary knowledge from multiple frames in the resultant local structural features, our method better infers the local geometry distributions at the current frame. In addition, the new patch-based framework can be readily incorporated with various existing single frame-based point cloud upsampling methods for further extension.

Finally, I observe that the patch-based framework in our second method suffers from inevitable patch-mismatch issues when significant movements occur between adjacent frames. To address this issue, I propose a sequential point cloud upsampling framework built upon progressive multi-level refinement (PMR-Net), including three different refinement stages at the frame level, shape level, and detail level, respectively. In the first refinement stage, I perform frame-level refinement by predicting point-wise warping displacement so that the points from reference frames can be roughly aligned with the points from the target frame, which effectively alleviates the patch-mismatch issue. In order to make better use of the temporal cues provided in the point cloud sequence, I introduce a new coarse-to-fine upsampling paradigm, which consists of a shape-level refinement module and a detail-level refinement module. The completed sparse result produced by the shape-level

refinement module serves as a better prior with essential geometry components, and the subsequent detail-level refinement is able to provide much more detailed geometric decoration.

Comprehensive experiments on multiple point cloud sequence datasets demonstrate the effectiveness of our proposed frameworks SPU, VPU, and PMRNet for sequential point cloud upsampling.

Deep Learning-based Methods For Sequential Point Cloud Upsampling

by

Kaisiyuan WANG
B.E. *Harbin Institute of Technology* M.S. *Harbin Institute of Technology*

A Thesis Submitted in Partial Fulfilment
of the Requirements for the Degree of
Doctor of Philosophy

at

University of Sydney
November, 2023

Authorship Attribution Statement

Chapter 3 is published as: **Kaisiyuan Wang**, Lu Sheng, Shuhang Gu, Dong Xu, "Sequential Point Cloud Upsampling via Exploiting Multi-scale Temporal Dependency", IEEE Transactions on Circuits and Systems for Video Technology, Volume: 31, Pages: 4686 - 4696, Issue: 12 December 2021, INSPEC Accession Number: 21440881.

I designed the algorithms, performed pre-processing and experiments on the collected data, and wrote the drafts of this article.

Chapter 4 is published as: **Kaisiyuan Wang**, Lu Sheng, Shuhang Gu, Dong Xu, "VPU: a Video-based Point Cloud Upsampling Framework", IEEE Transactions on Image Processing, Volume: 31, Pages: 4062 - 4075, Issue: 18 April 2022, INSPEC Accession Number: 21780929.

I designed the algorithms, performed pre-processing and experiments on the collected data, and wrote the drafts of this article.

In addition to the statements above, for the published articles in which I am not the corresponding author, approval to include the published item has been granted by the corresponding author.

Student Name: Kaisiyuan Wang

Supervisor Name: Luping Zhou

Date: November 8, 2023

Statement of Originality

This is to claim that the content of this thesis is the product of my own work. This thesis has not been submitted for any other purpose. I certify that the intellectual content of this thesis is my own work and that all the resources and the assistance received in preparing this thesis have been well acknowledged

Student Name: Kaisiyuan Wang

Date: November 8, 2023

Declaration

I, Kaisiyuan WANG, declare that this thesis titled, "Deep Learning-based Methods For Sequential Point Cloud Upsampling", which is submitted in fulfillment of the requirements for the Degree of Doctor of Philosophy, represents my own work except where due acknowledgement have been made. I further declare that it has not been previously included in a thesis, dissertation, or report submitted to this University or to any other institution for a degree, diploma or other qualifications.

For Mama and Papa

Acknowledgements

First, I would like to thank my supervisor A/Prof. Luping Zhou and Prof. Dong Xu for all their supports provided for my research. With their inspiring guidance, I obtained sufficient research experience and many useful skills during my Ph.D. study period, which helps me to become an independent researcher.

I would also sincerely thank my co-supervisor A/Prof. Wanli Ouyang for his supportive advice and academic resources during my candidature period, and I can always extend my research interests and view through the academic seminar held by him.

Last but not least, I would like to sincerely thank my family, my girlfriend, and all my friends holding my back. Their support encouraged me to face all the difficulties encountered during my candidature period.

Kaisiyuan WANG
University of Sydney
November 8, 2023

List of Publications

JOURNALS:

[1] **Kaisiyuan Wang**, Lu Sheng, Shuhang Gu, Dong Xu, "Sequential Point Cloud Upsampling via Exploiting Multi-scale Temporal Dependency", *IEEE Transactions on Circuits and Systems for Video Technology*, 2021.

[2] **Kaisiyuan Wang**, Lu Sheng, Shuhang Gu, Dong Xu, "VPU: a Video-based Point Cloud Upsampling Framework", *IEEE Transactions on Image Processing*, 2022.

[3] **Kaisiyuan Wang**, Jinyang Guo, Luping Zhou, "Sequential Point Cloud Upsampling via Progressive Multi-level Refinement", to be submitted to *IEEE Transactions on Image Processing*, 2023.

Contents

List of Figures

List of Tables

List of Abbreviations

CD	**C**hamfer **D**istance
CFFC	**C**ross-**F**rame **F**eature **C**ombination
DNN	**D**eep **N**eural **N**etwork
EMD	**E**arth **M**over's **D**istance
FPS	**F**arthest **P**oint **S**amplng
GFFM	**G**ate **F**eature **F**usion **M**odule
HD	**H**ausdorff **D**istance
***k*NN**	*k* **N**earest **N**eighbours
LIDAR	**LI**ght **D**etection **A**nd **R**anging
MLP	**M**ulti-**L**ayer **P**erceptrons
NLA	**N**on-**L**ocal **A**lignment
PMRNet	**P**rogressive **M**ulti-level **R**efinement **Net**work
P2F	**P**oint-to-sur**F**ace distance
SFPU	**S**ingle **F**rame-based **P**oint cloud **U**psampling
SPU	**S**equential **P**oint cloud **U**psampling framework
STA	**S**patio-**T**emporal **A**ggregation
TAM	**T**emporal **A**lignment **M**odule
VPU	**V**ideo-based **P**oint cloud **U**psampling framework
WFFC	**W**ithin-**F**rame **F**eature **C**ombination

Chapter 1

Introduction

In this chapter, I first introduce the background and motivation of sequential point cloud upsampling. Then, I briefly review the main challenges in the previous approaches and the task of sequential point cloud upsampling. Finally, I introduce the outline and the contributions of this thesis.

1.1 Background and Motivation

The point cloud, as one of the most popular 3D representations, has received increasing research interest for various emerging applications such as 3D reconstruction, autonomous vehicles, and virtual/augmented reality. Although point clouds are easy to access thanks to the widely used scanning devices, the data acquired from these 3D sensors are usually noisy, sparse, and non-uniform, which will inevitably bring significant challenges when developing advanced algorithms for downstream tasks such as classification and detection. To cope with this problem, point cloud restoration techniques are proposed to enhance the distorted point clouds, among which point cloud upsampling technologies have drawn considerable attention recently due to their efficiency and generalizability in real-world scenarios. Despite the success achieved by the existing point cloud upsampling approaches [81, 78, 26, 52, 57, 53], it is still an ill-posed problem to generate satisfactory results from a sparse input within merely a single point cloud snapshot, especially when the captured point clouds suffer from severe geometry loss in some local regions due to occlusion or lighting reflection. Therefore, it

is crucial to exploit non-trivial prior knowledge related to the structural details that are often missing in the raw input data.

Considering that the scanning devices usually capture point cloud sequences instead of single point cloud snapshots, I take inspiration from the 2D video super-resolution studies to leverage temporal dependencies among multiple neighboring point cloud frames for upsampling result enhancement.

In contrast to the existing single frame-based point cloud upsampling, sequential point cloud upsampling focuses on a distinct set of points:

- Single frame-based point cloud upsampling aims to recover dense and uniform results directly from distorted point cloud data in a single frame. While sequential point cloud upsampling is committed to exploiting the geometry contexts from the neighboring frames, which can additionally provide complementary structure and auxiliary details.
- Single frame-based point cloud upsampling usually employs a patch-based strategy for efficiency and optimal generalization performance, without taking into account the full context of the 3D objects. While sequential point cloud upsampling strives for consistency in the geometry generated throughout the entire sequence.

1.2 Problem Statement and Challenges

Given a sparse point cloud sequence $\mathcal{S} = \{\mathcal{P}_t\}_{t=1}^{T}$, where the point cloud at each time step has N 3D points, sequential point cloud upsampling aims to generate a dense point cloud sequence $\mathcal{D} = \{\mathcal{Q}_t\}_{t=1}^{T}$ based on $\mathcal{S}$, where the dense point cloud at each time step has rN 3D points. Here, T denotes the length of the sequence, and r denotes the required upsampling rate.

The main challenges of sequential point cloud upsampling can be summarized into three aspects:

(1) Unlike the 2D images with grid-based regular data structure, point cloud data is unordered with irregular data structure, which leads to many difficulties in learning motion representation and aggregation. Therefore, how to represent and aggregate the motion information between the point cloud frames is one of the key challenges of sequential point cloud upsampling.

(2) The previous single frame-based methods usually adopt a patch-based strategy for efficiency and good generalization performance. However, directly using this strategy on consecutive point cloud frames results in patch mismatch issues owing to the movements between frames, in which the patches cropped from the neighboring frames cover different regions from the ones cropped from the target frame. This difference may introduce undesirable geometric errors and misguide the decision-making of the networks, which leads to an unstable learning process or incorrect upsampling results.

(3) The commonly used paradigm for single frame-based methods includes a feature extraction module and a feature expansion module, where the former aims to extract representative geometric features from the input point cloud, and the latter focuses on expanding the duplicated points into a uniform distribution. However, the paradigm normally generates average results in the over-sparse regions rather than recovering delicate details, which limits the quality of the final reconstruction.

1.3 Thesis Outline and Contributions

Based on the aforementioned challenges, I make empirical explorations on the new sequential paradigm and propose three new methods for sequential point cloud upsampling, which progressively solves all the challenging problems. I organize the main content of this thesis as the following chapters:

Chapter 2. Literature review. In this chapter, I provide a comprehensive literature review on the background of 3D point clouds and the existing

point cloud upsampling techniques. Other recent studies related to the three methods I proposed are also involved in this chapter.

Chapter 3. Sequential Point Cloud Upsampling by Exploiting Multi-scale Temporal Dependencies. In this chapter, I focus on the first challenge and propose one of the earliest sequential point cloud upsampling frameworks, SPU by making use of different kinds of temporal clues, including both short-term and long-term, as well as multi-scale information from dynamic sequences. I introduce two novel designs, including the temporal alignment module and the gating mechanism for motion representation and aggregation, respectively. Extensive experiments on three dynamic point cloud datasets validate the effectiveness of the SPU framework and the corresponding novel designs.

- The contributions in this part are included in:

> **Kaisiyuan Wang**, Lu Sheng, Shuhang Gu, and Dong Xu, "Sequential Point Cloud Upsampling by Exploiting Multi-scale Temporal Dependency," in *IEEE Transactions on Circuits and Systems for Video Technology*, 2021.

Chapter 4. Patch-based Sequential Point Cloud Upsampling framework. In this chapter, I concentrate on the second challenge and propose the first patch-based sequential point cloud upsampling framework called VPU. It proposes a temporal patch-cropping strategy and a carefully designed spatio-temporal aggregation module accordingly, which can effectively extract, align and aggregate rich local geometric clues from consecutive frames at the feature level and be readily incorporated with various existing single frame-based methods. I evaluate the VPU framework on four point cloud sequence datasets to demonstrate the effectiveness of the carefully designed modules.

- The contributions in this part are included in:

> **Kaisiyuan Wang**, Lu Sheng, Shuhang Gu, and Dong Xu, "VPU: a Video-based Point Cloud Upsampling Framework", *IEEE Transactions on Image Processing*, 2022.

Chapter 5. Sequential Point Could Upsampling via Multi-Level Refinement. In this chapter, I continue exploring better solutions for the second challenge and additionally reconsider the existing paradigm of point cloud upsampling, based on which I propose a new sequential point cloud upsampling framework via progressive multi-level refinement called PMRNet. PMRNet consists of three refinement modules performed on the frame level, shape level, and detail level. The frame-level refinement robustly aligns the reference point cloud frames with the target point cloud frame at the coordinate space by predicting point-wise displacement between the distorted source and target point clouds, which can handle more severe patch-mismatch cases than VPU. The shape-level and detail-level refinement modules construct a new coarse-to-fine upsampling paradigm instead of the simple expansion, where the former module is designed for global shape completion, and the latter module focuses on delicate geometric details generation. I conduct comprehensive experiments and ablation studies to indicate the effectiveness of the three refinement modules in the PMRNet.

- The contributions in this part are included in:

> **Kaisiyuan Wang**, Jinyang Guo, and Luping Zhou, "Sequential Point Cloud Upsampling via Progressive Multi-level Refinement". To be submitted to *IEEE Transactions on Image Processing*, 2023.

Chapter 6. Conclusion and Future Work. In this chapter, I present conclusions and summarize the contributions of this thesis. I also discuss the limitations of the three proposed methods and the potential future work accordingly.

Chapter 2

Literature Review

In this chapter, I first present the recent development of relevant 3D point cloud tasks, and then I introduce the techniques from other areas that have inspired my research.

2.1 Point Cloud Upsampling

Following the first work PointNet [50] built upon deep neural networks, the community of 3D point clouds has achieved significant success in various tasks [76, 31, 68, 72, 19, 28, 75, 74, 51, 31, 60, 36, 77, 49, 59]. Nevertheless, the point clouds captured in real-world scenarios are often sparse, incomplete, and non-uniform, which degrades the performance of downstream tasks, such as classification and detection. Therefore, rapidly increasing attention has been paid to point cloud upsampling for the 3D object or scene recovery.

Specifically, Yu *et. al.* proposed the first deep learning-based point cloud upsampling approach PU-Net [81] to upsample the input point set, whose encoder architecture is based on PointNet++ [51]. They also introduced another edge-aware point cloud consolidation network (EC-Net) [80] to enhance the restoration results based on sharp features. Wang *et. al.* [78] proposed a multi-step progressive network (MPU) to further restore the details at multiple resolutions. Li *et. al.* [26] presented a GAN-based point cloud upsampling network PU-GAN to learn the point cloud distribution through adversarial learning. To further improve the uniformity of input data, they additionally introduced a new

uniformity loss term to regularize the density of points in small surfaces. Recently, Qian *et. al.* [52] proposed a new point cloud upsampling pipeline PU-GCN, which consists of a multi-scale feature extractor based on a graph convolution network and a novel Node Shuffle operator used for feature expansion. Li *et. al.* [27] disentangled the point cloud upsampling task into two sub-goals and presented a coarse-to-fine framework with two sub-networks, including a dense generator and a spatial refiner accordingly. Akhtar *et. al.* introduced PU-Dense [1], which employs a 3D multiscale architecture using sparse convolutional networks that hierarchically reconstruct an upsampled point cloud geometry via progressive rescaling and multiscale feature extraction. Chen *et. al.* proposed a transformer-based framework PC^2-PU [37] explores patch-to-patch and point-to-point correlations for more robust point cloud upsampling.

2.2 Motion Modelling for Point Cloud Sequences

2.2.1 Dynamic Point Cloud Processing

In recent years, several works have been proposed to exploit temporal redundancy from point cloud sequences for various tasks, including 3D human action recognition [20, 13], 3D semantic segmentation [34, 35], 3D object tracking [71, 79, 25] and 3D dynamic object/scene reconstruction [48, 45, 55, 39].

A direct method to process point cloud sequences is to convert the 3D point cloud data into a regular grid structure via voxelization and then perform 4D convolution on the voxels. Therefore, a few works [7, 58, 85] were proposed to process the point cloud sequences by using 4D convolution. However, these methods suffer from low efficiency and the inevitable quantization errors caused by the voxelization operation. For example, motivated by spatio-temporal CNNs, Choy *et. al.* [7] converted the 3D point cloud data to a regular grid structure via voxelization operation and then adopted 4D CNNs to effectively capture the spatio-temporal representation.

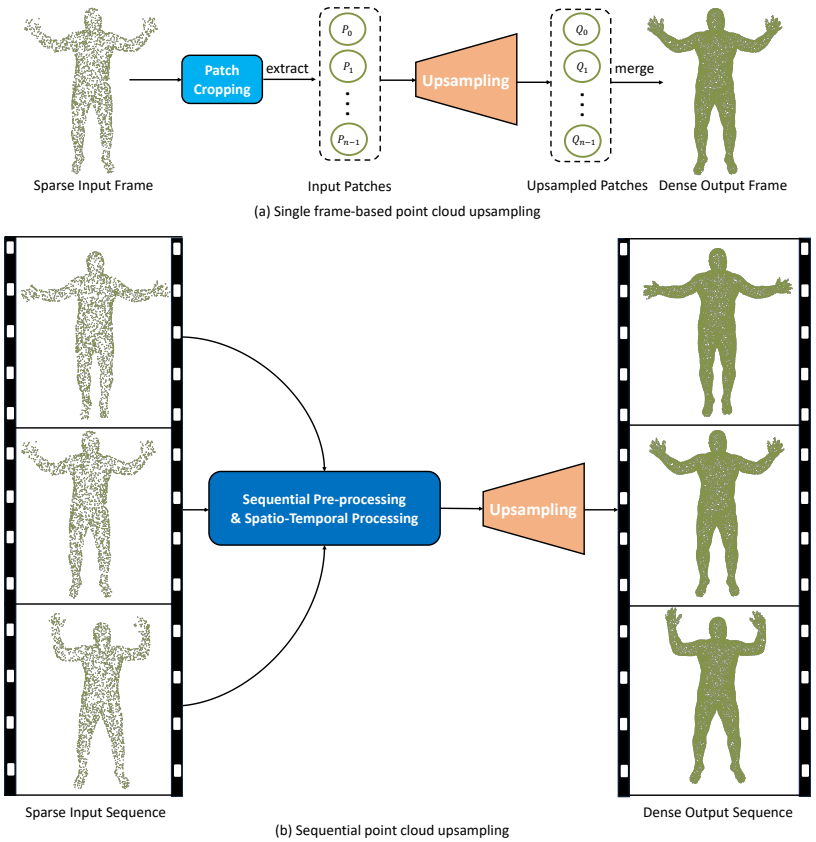


Figure 2.1: Comparison of single frame-based point cloud upsampling and the proposed sequential point cloud upsampling. (a) Single frame-based point cloud upsampling aims to recover dense and uniform results directly from an individual frame by using the patch-based strategy. (b) The proposed sequential point cloud upsampling takes advantage of the spatio-temporal hints within the point cloud sequence to achieve more consistent results with richer geometric details.

Recently, a few works [34, 16, 2, 70, 73] used 3D scene flow to represent 3D motion, which only focuses on predicting the motion between a pair of 3D points. Liu *et al.* introduced MeteorNet [35] to directly process raw point cloud sequences by constructing spatio-temporal neighborhoods. However, the explicitly collected spatio-temporal neighborhoods may contain undesirable information, which often degrades classification and segmentation performance. Fan *et al.* [38] presented MoNet to extract motion features between two consecutive point cloud frames in order to accurately predict future frames. Min *et al.* proposed the PointLSTM method [43] for human gesture recognition by propagating information from the past to the future while preserving the spatial structure. Fan *et al.* [13] proposed a point cloud 4D transformer network called P4Transformer, which depends on a transformer-based architecture to capture the appearance and motion information across the entire point cloud video by performing self-attention on the embedded local features.

In addition to these works on high-level computer vision tasks, there are also a few studies focusing on low-level vision demands for 3D data. Lukas *et al.* [48] proposed Tranquil Clouds to learn stable and temporally coherent feature spaces for points clouds changing over time, in which a temporal loss is introduced among the adjacent frames. Niemeyer *et al.* [45] extended the occupancy network [41] as a 4D version named Occupancy Flow, which uses the neural ordinary differential equation to predict the trajectory of each point explicitly. Meanwhile, Rempe *et al.* [55] adopted a latent representation to model motion information contained in the point cloud sequence.

2.2.2 Non-rigid Point Cloud Registration

Our approach is also related to point cloud registration methods. Estimating non-rigid correspondence between point clouds is a fundamental research topic in computer vision and computer graphics, which enables wide applications in many fields, such as 3D reconstruction and animation. The early studies [8, 21, 6, 44] are all non-parametric methods based on thin plate spline functions [4] or local transformation. Among these methods, the most widely used Coherent point drift (CPD) [44] is based

on the Gaussian Mixture Model (GMM), which implicitly learns the motion field between two point sets depending on the coherent penalization on the velocity field. Recently, several learning-based approaches for non-rigid point cloud registration have been proposed. Wang *et. al.* presented CPD-Net [66], which extracts shape descriptors from the source and target point clouds via the PointNet [50] backbone and predicts the point-wise displacement. After CPD-Net, they further proposed PR-Net [65], which adopts an intermediate grid-based representation to calculate the shape correlation tensor between two shape descriptors and predict the shape transformation parameters. Feng *et. al.* presented a recurrent framework RMA-Net [14], which represents the non-rigid transformation by iteratively estimating rigid transformations for several steps. Li *et. al.* introduced Lepard [30], a new architecture designed to disentangle point cloud representation into feature space and 3D position space.

2.3 Video Super-resolution

Our research is also related to the video super-resolution (VSR) methods, which exploit temporal information from video sequences for improving the spatial resolution of each video frame. Most of the existing VSR approaches rely on motion compensation to align multiple input video frames. However, it is often hard for these early VSR works to cope with the videos with large motion or significant occlusion. The recent works TDAN [61] and EDVR [67] used an implicit motion compensation strategy by performing the deformable convolution operations [10] to align the input frames in the feature space, which achieves the state-of-the-art VSR results. Unlike the VSR task where visual contents in the adjacent frames of each video sequence are often roughly aligned based on the regular coordinate system, the point cloud sequences are irregularly, sparsely and non-uniformly sampled from moving surfaces, so it is a non-trivial task to explicitly exploit temporal information and align the 3D points from adjacent frames due to the lack of one-to-one correspondence between these 3D points.

Quantization-based Image Modeling Image modeling has recently achieved significant progress after taking inspiration from Transformer networks. VQVAE [62] and VQGAN [12], as the pioneer of quantized image modeling, have received extensive attention from multiple works on high-resolution image synthesis and editing.

Quantized image modeling is performed via an autoencoder network and a learnable codebook. The key insight of these approaches is to replace the discrete representations from the encoder with the codes queried from the learned codebook. VQGAN [12] and ImageBART [11] leverage a transformer to synthesize images in an auto-regressive manner, while MaskGIT [5] proposes to model an image from multiple directions instead of the sequential prediction as in VQGAN. CodeFormer [86] achieves remarkable face restoration by learning a high-quality texture dictionary and building a reliable mapping with much less uncertainty. VQFR [17] designs a parallel decoder to replace the commonly used transformer for reconstructing high-fidelity human face details.

2.4 Summary

In this chapter, I have introduced the studies that have inspired my exploration of sequential point cloud upsampling, including the advanced point cloud processing techniques and the recent progress in 2D image/video restoration. Specifically, I first briefly review the existing single frame-based point cloud upsampling techniques. Then I continue to discuss the techniques for 3D point cloud motion modeling, which consists of dynamic point cloud processing and non-rigid point cloud registration. Finally, I introduce state-of-the-art approaches in 2D video super-resolution and image modeling.

According to the detailed analysis and discussion of these related works, I summarize the differences between these related works and the

proposed sequential point cloud upsampling task in Table 2.1. Sequential point cloud upsampling is a newly proposed research topic for low-level restoration in the 3D space rather than high-level tasks (*e.g.*, classification and detection). The key problem of this task is how to align and aggregate the spatio-temporal information from the point cloud sequence with irregular data structure instead of regular grid-based data in 2D images. However, the difficulties are not only caused by the lack of one-to-one correspondence but also derived from the challenging scenarios with distorted input point clouds.

Table 2.1: Comparison between the related studies and the proposed sequential point cloud upsampling.

Differences	Sequential point cloud upsampling	
Single frame-based point cloud upsampling	Single frame-based point cloud upsampling only focuses on the spatial context within each single point cloud frame without considering the geometry of the entire 3D object.	Sequential point cloud upsampling studies how to exploit rich temporal dependencies in the point cloud sequence for complementary recovery. The comparison between these two tasks is detailedly illustrated in Fig. 2.1 for better understanding.
Temporal modeling for dynamic point clouds	The previous works on dynamic point cloud processing either perform temporal modeling in the whole point cloud sequence or just consider the motion between adjacent frames for knowledge extraction.	Sequential point cloud upsampling makes explorations on different kinds of temporal dependencies (*e.g.*, both long-term and short-term temporal representation). More importantly, sequential point cloud upsampling focuses on the low-level task video-based point cloud upsampling, instead of the high-level vision tasks in these works.
Non-rigid point cloud registration	The studies on non-rigid point cloud registration are normally committed to estimating the correspondence between two clean point sets in different motion states caused by non-rigid deformation.	In contrast, sequential point cloud upsampling serves for more complex scenarios with greater demands for robustness, in which performing alignment between two distorted point sets (*i.e.*, sparse, non-uniform, and incomplete) becomes much more challenging.
Video super-resolution	In 2D VSR task, the 2D image pixels with the same content are roughly aligned in the regular grid space and the grid space for each frame shares the same 2D grid template.	Conversely, it is more challenging to align irregular point clouds from adjacent frames in the 3D coordinate space due to the lack of one-to-one correspondence. Furthermore, distorted point clouds from adjacent frames may suffer from geometry loss in different parts, leading to more obstacles in point-wise alignment.

Chapter 3

Sequential Point Cloud Upsampling by Exploiting Multi-scale Temporal Dependencies

In this chapter, I propose a sequential point cloud upsampling framework to generate fine-grained and temporally consistent upsampling results for dynamic point cloud sequences by leveraging different kinds of temporal clues. Comprehensive experiments performed on three point cloud sequence datasets (i.e., DYNA, MSR Action3D and COMA) demonstrate the effectiveness of the SPU framework.

3.1 Motivations and Contributions

To better understand 3D shapes, objects, and scenes captured by LIDAR sensors and RGB-D cameras, rapidly increasing attention has been paid to 3D point cloud processing algorithms in the community of computer vision, robotics, and mixed realities. Significant progress has been made in recent works, which mainly focus on high-level tasks, such as 3D object classification, segmentation, and detection [50, 51, 31, 60, 36, 87, 49, 42, 59]. However, most of these works still face considerable challenges since the point clouds captured in the real scenarios are usually noisy, sparse and non-uniform.

To cope with this issue, several works [81, 78, 26, 57] related to point cloud upsampling proposed to generate denser and more uniform 3D objects from a single point cloud frame by using deep neural networks (DNNs), which have achieved certain success for recovering spatial details. However, it is still challenging to upsample point clouds that are over-sparse and noisy. Firstly, when the geometric clues are inadequate in the raw input point cloud data, single frame-based methods may fail to recover faithful geometric details in critical areas with important semantics, e.g., hands may not be well upsampled when their input points are too few (e.g., less than 10 points in some cases). Secondly, most of the previous upsampling works adopt a patch-based strategy that performs upsampling on small patches cropped from the input 3D objects [26], and produces dense point clouds by merging the upsampling results from all patches. However, these methods [81, 78, 26] neglect necessary knowledge about the global shapes of the input point clouds, which may result in undesirable outliers around the boundaries of patches, and thereafter fail to reconstruct consistent shape details of the 3D objects.

Recently, many 2D video super-resolution works [56, 22, 32, 33, 69, 18] show that it is beneficial to use different kinds of temporal dependencies such as neighboring low-resolution frames and inferred high-resolution estimation from the preceding frames to generate high-resolution video sequences with both fine-grained details and long-term coherence. Borrowing similar ideas from these works, the issue of lack of geometric clues in a single point cloud frame could be addressed by aggregating the geometric clues from a set of temporally adjacent frames, which enables much more reliable shape representation and also leads to more faithful point cloud upsampling results. Inspired by these works, I address the aforementioned limitations from the single frame-based point cloud upsampling methods by exploring multi-scale, long-term/short-term temporal dependencies from point cloud sequences in order to achieve comprehensive and consistent shape representation of the current frame. Moreover, in contrast to the patch-based upsampling strategy used in [81, 78, 26], I instead apply a holistic upsampling strategy to generate globally consistent upsampling results that are more aware of the semantics of the input point clouds.

To this end, in this chapter, I propose a sequential point cloud upsampling framework (referred to as SPU) to generate fine-grained and temporally consistent upsampling results for dynamic point cloud sequences. Specifically, in order to gather complementary geometric information for each sparse input point cloud frame, I first extract a set of short-term features from the orderless 3D points in each low-resolution reference frame (e.g., the previous/current/subsequent frame). Meanwhile, considering that high-resolution estimation includes more concrete structural details within each local area, I also produce another short-term feature based on the upsampling result reconstructed from the previous frame. Moreover, for better preserving the temporal coherence of the generated point cloud sequence, I add a long-term latent representation accumulated throughout the sequence into the upsampling process, in order to better discover the global shape priors and also reduce the outliers when generating the underlying surfaces. This latent representation is designed to be propagated through the entire sequence and recurrently updated by each frame from the sequence, which can thus capture the common and consistent shape prior of the 3D object in the point cloud sequence.

To overcome the inherent alignment difficulty caused by the orderless characteristics of point cloud, I additionally propose the temporal alignment module (TAM) to implicitly align each temporal hint (i.e., the multi-scale short-term features or the long-term feature) with respect to the short-term feature extracted from the current frame. Motivated by the self-attention mechanism, I first calculate the cross-frame similarity matrix to measure the pair-wise similarity between every pair of points from the reference frame and the current frame at the feature space, based on which I produce the transformed feature by warping the feature of the reference frame with respect to that of the current frame. By using the TAM, soft alignment is allowed for the points from the reference frame with respect to those from the target frame at the feature space. I also propose a new gating mechanism to learn the optimal weights for the transformed features for better exploiting various temporal dependencies, which are then used to effectively aggregate these

transformed features as the final fused feature. Taking the fused feature as the input, the existing single frame-based point cloud upsampling methods (e.g., PU-Net [81], MPU [78] and PU-GAN [26]) can be readily used to generate the dense point cloud for the current frame.

Extensive experiments on three benchmark datasets DYNA, COMA, and MSR Action3D demonstrate that the proposed method SPU outperforms the existing single frame-based methods in terms of high-fidelity reconstruction.

The contributions are summarized as follows: (1) To the best of my knowledge, SPU is one of the earliest attempts to exploit rich and complementary temporal clues, including both short-term and long-term, as well as multi-scale information from dynamic sequences to enhance the point cloud upsampling performance. (2) I propose a general alignment strategy tailored for processing dynamic point cloud sequences with orderless data structure, which can be readily incorporated with the existing single frame-based point cloud upsampling methods, such as PU-Net [81], MPU [78] and PU-GAN [26]. (3) SPU achieves state-of-the-art results on three point cloud sequences datasets, with substantial performance improvement over its single frame-based counterparts.

3.1.1 Discussion

The studies most related to SPU are the proposed single frame-based point cloud upsampling approaches [81, 78, 26, 27, 52]. Unlike these point cloud upsampling methods that only consider spatial information within a single point cloud snapshot, SPU exploits different temporal dependencies within a point cloud sequence to further enhance the upsampling results.

Apart from these single frame-based point cloud upsampling methods, I also find my study related to approaches designed for processing dynamic point cloud sequences. Unlike the previous works [43, 38, 45] which pay more attention to modeling the temporal information in the whole point cloud sequence or other related works [34, 48] which only consider the motion between the adjacent frames, my method attempts

to exploit rich temporal information including both long-term latent representation from the sequence and short-term geometric details from the adjacent frames to further enhance the point cloud upsampling results.

3.2 Methodology

The proposed sequential point cloud upsampling framework (SPU) is shown in Fig. 3.1. The recent single frame-based point cloud upsampling methods like PU-GAN [26], PU-Net [81] and MPU [78] can be readily embedded as the components of the SPU framework (e.g., the orange blocks in Fig. 3.1). From Sec. 3.2.1 to Sec. 3.2.6, I will take PU-GAN as an example to introduce sequential point cloud upsampling method SPU (PU-GAN) in detail, and how to incorporate other methods like PU-Net [81] and MPU [78] into the SPU framework will be discussed in Sec. 3.2.7.

3.2.1 Overview of Sequential Point Cloud Upsampling Method

In order to reconstruct dense results with sufficient geometric details from the sparse input point clouds $\mathcal{P}_t$ at time t, the core idea of sequential point cloud upsampling focuses on exploiting rich information from the given point cloud sequence to collect spatio-temporal complementary information for $\mathcal{P}_t$. Specifically, I adopt a temporal dependency exploitation strategy which considers not only multi-scale short-term geometric details in the neighboring frames, but also long-term feature based on the point cloud sequence. For the short-term geometric features, I first respectively extract per-point features (i.e., $\mathbf{P}_{t-1}$, $\mathbf{P}_t$, and $\mathbf{P}_{t+1} \in \mathbb{R}^{N\times C}$) from 3 consecutive low-resolution point cloud frames $\mathcal{P}_{t-1}$, $\mathcal{P}_t$, and $\mathcal{P}_{t+1}$ by using a per-point feature extraction unit, where C is the number of feature channels. In addition to three low-resolution short-term features, I also produce a high-resolution short-term feature $\mathbf{Q}_{t-1} \in \mathbb{R}^{rN\times C}$ based on the reconstructed previous frame $\mathcal{Q}_{t-1}$ by using the same per-point feature extraction unit. Note that any feature extraction unit in the existing point cloud upsampling methods can be readily adopted in the SPU

framework (e.g., the DGCNN-like feature extraction method [68] used in PU-GAN [26]). For the long-term feature, I use the latent representation produced from the previous time step, namely $\mathbf{H}_{t-1} \in \mathbb{R}^{N \times C}$. Generally, this long-term feature is updated recurrently with the information of the new frame at each time step, which is able to integrate the common and consistent shape representation for the 3D object in the sequence.

Taking the short-term feature of the current frame $\mathbf{P}_t$ as the target, I perform alignment for these short-term features $\mathbf{P}_{t-1}, \mathbf{P}_t, \mathbf{P}_{t+1}, \mathbf{Q}_{t-1}$ and the long-term feature $\mathbf{H}_{t-1}$ by using the proposed temporal alignment module (TAM) to produce the transformed features $\mathbf{T}_{t-1}$, $\mathbf{T}_t$, $\mathbf{T}_{t+1}$, $\mathbf{T}_Q$, and $\mathbf{T}_H$ respectively. Moreover, I use the proposed gated feature fusion module (GFFM) to produce the optimal fused feature of these transformed features (referred to as $\mathbf{F}_{fuse}$) by calculating the gate value for each input of the GFFM. Then I feed the final fused feature $\mathbf{F}_{fuse}$, which integrates the rich temporal information, into the single frame-based point cloud upsampling module to generate the reconstructed point cloud $\mathbf{Q}_t$ for the current frame. Similar to the feature extraction unit, the decoder unit of the existing point cloud upsampling methods can also be directly adopted as the single frame-based point cloud upsampling module.

3.2.2 Temporal Alignment Module (TAM)

The inputs of the temporal alignment module consist of four short-term features including three low-resolution point cloud features $\mathbf{P}_{t-1}, \mathbf{P}_t, \mathbf{P}_{t+1}$, and a high-resolution point cloud feature $\mathbf{Q}_{t-1}$ and a long-term feature $\mathbf{H}_{t-1}$. Based on these five extracted features, I attempt to enrich the feature of the current frame $\mathbf{P}_t$ by adding rich temporal information into each corresponding point. However, due to the irregular data structure of point cloud data, these input features are not aligned in the coordinate space. Thus, feature-level fusion through per-point concatenation [16] along the feature dimensions cannot be directly performed without a prior alignment process. Inspired by the cross-shape attention strategy in [46], I perform feature alignment from each reference long-term/short-term feature (i.e., $\mathbf{P}_{t-1}, \mathbf{P}_{t+1}, \mathbf{Q}_{t-1}$ and $\mathbf{H}_{t-1}$) to the

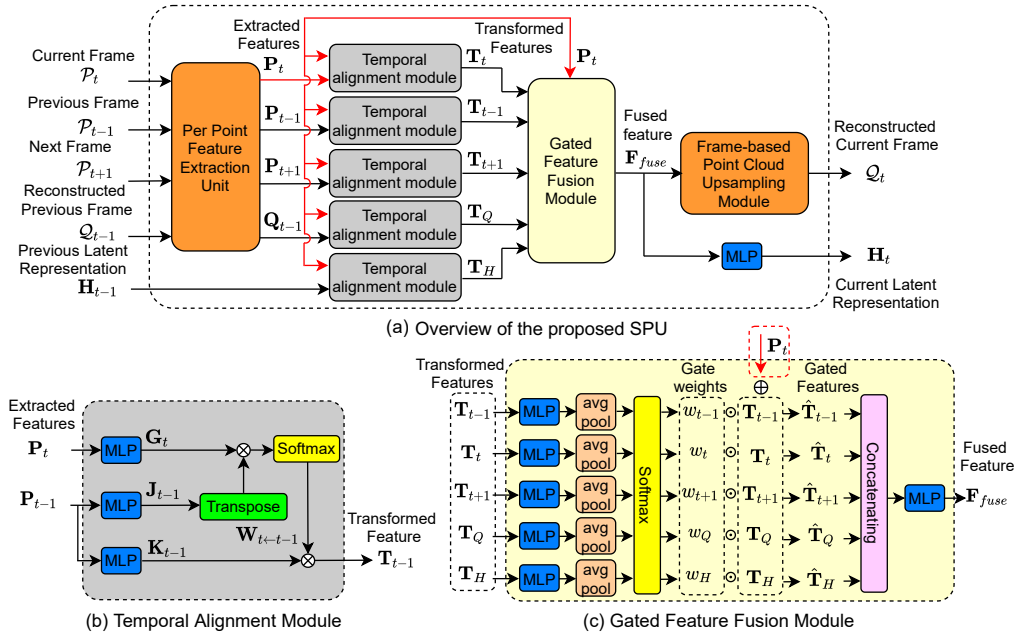


Figure 3.1: (a) Overview of the sequential point cloud upsampling (SPU) framework. Given sparse point sets from three consecutive frames $\mathcal{P}_{t-1}$, $\mathcal{P}_t$, $\mathcal{P}_{t+1}$, and the reconstructed dense output $\mathcal{Q}_{t-1}$ from frame $t-1$, the first step is to extract low-resolution short-term features $\mathbf{P}_{t-1}$, $\mathbf{P}_t$, $\mathbf{P}_{t+1}$ and high-resolution short-term feature $\mathbf{Q}_{t-1}$ by using the per-point feature extraction unit. Then I transform these extracted short-term features and the previous long-term feature $\mathbf{H}_{t-1}$ with respect to the target short-term feature $\mathbf{P}_t$ by using the Temporal Alignment Module, which outputs the transformed features $\mathbf{T}_{t-1}$, $\mathbf{T}_t$, $\mathbf{T}_{t+1}$, $\mathbf{T}_Q$, and $\mathbf{T}_H$ respectively. In the next step, these transformed features are then effectively fused by the Gated Feature Fusion Module. Lastly, I feed the fused feature $\mathbf{F}_{fuse}$ into the single frame-based point cloud upsampling module to generate the reconstructed point cloud $\mathbf{Q}_t$ for the current frame. (b) An example of the Temporal Alignment Module (TAM). $\mathbf{P}_t$, and $\mathbf{P}_{t-1}$ are the extracted low-resolution short-term features from two consecutive frames, where $\mathbf{P}_t$ is the target. $\mathbf{G}_t$, $\mathbf{J}_{t-1}$ and $\mathbf{K}_{t-1}$ indicate the projection outputs after using MLPs, and $\mathbf{W}_{t\leftarrow t-1}$ indicates the similarity matrix calculated by using $\mathbf{G}_t$ and $\mathbf{J}_{t-1}$. The `softmax` operation is performed on each row of the similarity matrix. $\otimes$ denotes matrix multiplication. (c) The Gated Feature Fusion Module. I first feed the transformed features from TAM into a set of MLPs and average pooling layers, and calculate the corresponding normalized gated values w_{t-1}, w_t, w_{t+1}, w_Q, and w_H by using the `softmax` operation. Then I concatenate the weighted transformed features in each feature dimension, and produce the final fused feature $\mathbf{F}_{fuse}$ after another MLP layer. $\oplus$ denotes element-wise sum, and $\odot$ denotes element-wise multiplication.

target feature $\mathbf{P}_t$. As shown in Fig. 3.1(b), I take the alignment from $\mathbf{P}_{t-1}$ to $\mathbf{P}_t$ as an example. The reference feature $\mathbf{P}_{t-1}$ and the target feature $\mathbf{P}_t$ are at first projected into the common feature space through two different multi-layer perceptions (MLPs), which are denoted as $\mathbf{J}_{t-1} \in \mathbb{R}^{N\times d}$ and $\mathbf{G}_t \in \mathbb{R}^{N\times d}$, respectively, where d denotes the feature dimension in the common feature space. Then I calculate a similarity matrix between $\mathbf{G}_t$ and $\mathbf{J}_{t-1}$ defined as:

$$\mathbf{W}_{t\leftarrow t-1} = \texttt{softmax}(\mathbf{G}_t\mathbf{J}_{t-1}^{\top}) \in \mathbb{R}^{N\times N}, \tag{3.1}$$

where the softmax operation is performed at each row of the similarity matrix. By further projecting $\mathbf{P}_{t-1}$ into the same feature space as $\mathbf{P}_t$ through another MLP layer (referred to as $\mathbf{K}_{t-1} \in \mathbb{R}^{N\times C}$), and the aligned reference feature (also referred to as the transformed feature) can be defined as follows:

$$\mathbf{T}_{t-1} = \mathbf{W}_{t\leftarrow t-1}\mathbf{K}_{t-1}. \tag{3.2}$$

Similarly, I can obtain the other three transformed features $\mathbf{T}_{t+1}$, $\mathbf{T}_Q$ and $\mathbf{T}_H$ accordingly. Thereafter, the geometric details from the short-term transformed features as well as the common shape prior from the long-term transformed feature can be correctly aligned with the corresponding feature in $\mathbf{P}_t$.

Moreover, I also apply the self-alignment operation towards the target feature $\mathbf{P}_t$ by using the same temporal alignment module, which outputs the transformed feature $\mathbf{T}_t$. Similarly as in self-attention [83], it enables non-local contextual feature aggregation of $\mathbf{P}_t$, which is also shown to be useful in point cloud upsampling in [26].

3.2.3 Gated Feature Fusion Module (GFFM)

Even though the transformed features have been roughly aligned by using the temporal alignment module, their temporal dependency with respect to $\mathbf{P}_t$ is different. Therefore, each transformed feature should be aggregated to the target feature $\mathbf{P}_t$ in an effective way. However, a simple fusion strategy using the uniform weights would blur the generated

point clouds due to interference from undesirable contents in the neighboring frames.

To this end, I calculate the adaptive weights by learning a gating function based on the set of transformed features. To be specific, the gating value is calculated below:

$$\begin{aligned} w_i &= \frac{\exp(\texttt{avg\_pool}(f_{\text{MLP}}(\mathbf{T}_i)))}{\sum_{j\in\Omega}\exp(\texttt{avg\_pool}(f_{\text{MLP}}(\mathbf{T}_j)))}, \\ \Omega &= \{t-1, t, t+1, Q, H\}, \end{aligned} \tag{3.3}$$

where f_{MLP} stands for a set of MLP layers. `avg_pool` means the average pooling layer, after which the averaged features are normalized by the `softmax` operation. Each transformed feature $\mathbf{T}_i$ is combined with the target feature $\mathbf{P}_t$ to further improve the robustness for extracting the fused feature. The scalar w_i measures the importance/contribution from each transformed feature. Finally, I calculate $\hat{\mathbf{T}}_i$ below:

$$\hat{\mathbf{T}}_i = w_i \odot (\mathbf{T}_i \oplus \mathbf{P}_t),\ i \in \Omega, \tag{3.4}$$

where $\oplus$ is the element-wise sum operation, and $\odot$ is the element-wise multiplication operation. $\hat{\mathbf{T}}_i$ is the gated feature, which can be readily used to augment the target feature $\mathbf{P}_t$. After concatenating four temporally aligned features $\hat{\mathbf{T}}_{t-1}$, $\hat{\mathbf{T}}_{t+1}$, $\hat{\mathbf{T}}_Q$, $\hat{\mathbf{T}}_H$ and a self-aligned feature $\hat{\mathbf{T}}_t$ along the feature channel dimension, I feed the aggregated feature to a set of MLP layers, and finally I obtain the fused feature $\mathbf{F}_{fuse}$. This feature is then utilized to generate the dense point cloud for the current frame by using the existing single frame-based point cloud upsampling methods [81, 78, 26]. This feature is also used to generate the long-term feature for the current time step (i.e., $\mathbf{H}_t$) through another set of MLP layers, which will then be employed as one of the inputs to generate the dense point cloud for the next frame $\mathbf{P}_{t+1}$. It is worth noting that by fusing the aligned long-term feature (i.e., $\hat{\mathbf{T}}_H$) with the four short-term aligned features (i.e., $\hat{\mathbf{T}}_{t-1}$, $\hat{\mathbf{T}}_t$, $\hat{\mathbf{T}}_{t+1}$ and $\hat{\mathbf{T}}_Q$) in the GFFM, I also manage to update the long-term feature recurrently based on a set of new short-term features at each time step.

3.2.4 Discussion

According to Fig. 3.1, although I conduct similar operations for the short-term features and the long-term feature at each time step, the roles they play are quite different.

The short-term features focus on the geometric information from the neighboring frames (i.e., the low resolution or high resolution frames), and I are able to enrich the short-term feature at the current time step (i.e., $\hat{\mathbf{T}}_t$) with other complementary geometric clues by effectively aggregating the temporally aligned features $\hat{\mathbf{T}}_{t-1}$, $\hat{\mathbf{T}}_{t+1}$ and $\hat{\mathbf{T}}_Q$ in GFFM 3.2.3.

Different from the short-term features that only consider the adjacent frames around the current frame, the long-term feature focuses on the 3D shape prior from the whole sequence. By recurrently accumulating optimally weighted short-term features from the GFFM 3.2.3, the long-term feature $\mathbf{H}_t$ is able to retain useful prior knowledge at each time step, which can be used as a constraint to better preserve the temporal coherence of the generated dense point cloud sequence.

3.2.5 Single Frame-based Point Cloud Upsampling Module

After obtaining the fused feature $\mathbf{F}_{fuse}$, I employ a single frame-based point cloud upsampling module to generate the output point cloud $\mathcal{Q}_t$, which will in turn be used as the input to the next round of SPU. For this module, I can readily borrow the upsampling modules used in various point cloud upsampling methods. For example, I can follow the network in PU-GAN [26], in which I feed $\mathbf{F}_{fuse}$ into the up-down-up feature expansion module, and then the 3D coordinate reconstruction module with the farthest point sampling (FPS) operation to generate the dense point cloud with an upsampling rate r. It is worth mentioning that the whole SPU framework does not require a fixed number of points in the input point clouds, thus it is capable of processing point clouds with various scales.

3.2.6 Loss Functions

For compatibility with the recent single frame-based method, such as PU-GAN [26], and demonstrating the effectiveness of the SPU framework, I follow the adversarial training strategy and completely adopt their loss functions, namely I have:

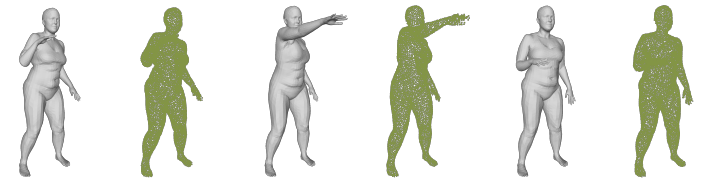
$$L_G = \lambda_{gan} L_{gan} + \lambda_{rec} L_{rec} + \lambda_{uni} L_{uni}, \text{ and } L_D = L_{dis}, \tag{3.5}$$

$\mathcal{L}_G$ is used to train SPU, and $\mathcal{L}_D$ is to train the discriminator used in PU-GAN. L_{gan} and L_{dis} are a pair of adversarial losses, which are defined based on the existing work LSGAN [40]. L_{rec} denotes the Earth Mover's Distance (EMD) between the generated dense point cloud and the ground-truth point cloud. L_{uni} is a loss term used to improve the uniformity of point clouds by enforcing additional constraints on small disks from the surface. As in [26], I empirically set $\lambda_{gan} = 0.5$, $\lambda_{rec} = 100$ and $\lambda_{uni} = 10$, please refer to [26] for more details.

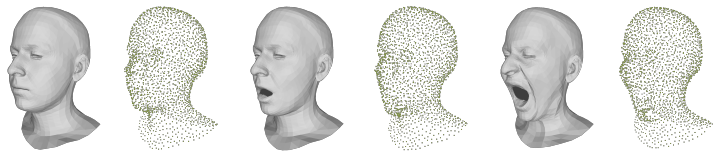
Note that in this work, I use five inputs to generate the dense point cloud for the current frame only, so I use the loss functions at the frame level in the same way as in [26]. In the experiments, I observe that the SPU framework without using any additional constraints can generate much better upsampling results than PU-GAN [26]. How to enforce more constraints to better exploit temporal coherence in the loss functions will be studied in future work.

3.2.7 SPU using PU-Net or MPU as the single frame-based point cloud upsampling method

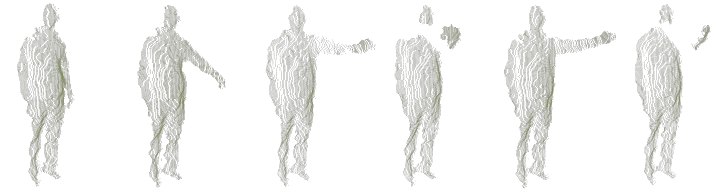
As mentioned before, other single frame-based point cloud upsampling methods can also be readily incorporated in the proposed SPU framework. Specifically, when using PU-Net [81] and MPU [78], the feature extraction unit and the single frame-based point cloud upsampling module become the corresponding modules from PU-Net [81] and MPU [78], respectively. Meanwhile, the loss function is also completely borrowed from the corresponding single frame-based method. For the loss function in PU-Net [81], the adversarial losses (i.e., L_D and L_{gan} in L_G) are



(a) Dynamic human body examples from DYNA dataset



(b) Dynamic human head examples from COMA dataset



(c) Dynamic human action examples from the MSR Action3D dataset

Figure 3.2: Visual examples from the used DYNA, COMA, and MSR Action3D datasets. DYNA and COMA are mesh-based datasets, I provide both original meshes and the sampled point clouds for better visualization. MSR Action3D is a real-scanned dataset, which only contains raw point clouds captured by the depth camera.

removed and the uniform loss is replaced by a repulsion loss as in [81]. For the loss function in MPU [78], only L_{rec} in Eq.(5) is retained and further replaced by the Chamfer Distance.

3.3 Experiments

3.3.1 Datasets

To evaluate the SPU framework, I first conduct extensive experiments on two template mesh-based datasets, DYNA [47] and COMA [54]. Then I additionally conduct another experiment on a real scanned point cloud-based dataset MSR Action3D [29] to further evaluate the generalization capability of the SPU framework under a more challenging scenario. Visual examples from these three datasets are provided in Fig. 3.2 for better understanding of the proposed sequential point cloud upsampling task.

Both DYNA and COMA datasets are captured from a number of dynamic moving 3D objects in the real world. Specifically, the DYNA dataset [47] contains 134 sequences with various dynamic deformation related to human body movement. Each sequence contains 145 to 950 frames, and the deformation is mainly due to the movement of ten persons with different human shapes. The COMA dataset [54] consists of 120 dynamic sequences collected from 10 different subjects with varying expressions, in which each sequence has 15 to 345 frames. In order to mimic the intrinsic characteristics of real point clouds and focus more on the dynamic parts of the 3D point cloud data on the COMA dataset, I only retain the frontal part of the 3D face models and remove the occluded backhead and eyeballs. To prepare for the point cloud sequences from dynamic 3D mesh-based models, I adopt the Poisson-disk sampling strategy [9] to sample the points from each mesh frame. Each input point-cloud frame and the ground-truth frame have 256 and 1024 points, respectively. Namely, I set the upsampling rate r as 4.

For both DYNA and COMA datasets, I select the sequences from 2 subjects as the testing dataset and use the sequences from the remaining subjects for training. Considering that the numbers of training and testing sequences on both datasets are limited (i.e., fewer than 110 training sequences and 30 testing sequences), I additionally prepare for a large number of short clips in order to provide sufficient training and testing data. Specifically, each short clip consists of 9 consecutive frames sampled from one sequence in the training and the testing sets. Finally, I have 6,632 (resp., 243) and 5,325 (resp., 135) training (resp., testing) point cloud clips from the DYNA and COMA datasets, respectively.

The MSR Action3D dataset [29] contains 567 point cloud sequences scanned from 10 subjects when performing 20 different human actions. Unlike the point clouds in the DYNA and COMA datasets that are sampled from the ground-truth meshes, the point clouds in MSR Action3D are reconstructed based on depth information. Therefore, I directly use the FPS method on the raw point cloud data to produce the input frames and the ground-truth frames with 256 and 1024 points, respectively. I choose 186 sequences from all the 10 subjects and all the actions as the

testing dataset to evaluate the generalization capability of the framework.

3.3.2 Implementation Details

The proposed network is implemented by using TensorFlow and the training is performed on a single NVIDIA 1080Ti GPU. When PU-GAN [26] is incorporated into the SPU framework, I set the batch size as 4 and the initial learning rates for the generator and the discriminator as 0.0005 and 0.0001, respectively, which are reduced by a decay rate of 0.8 after every 8000 iterations. When PU-Net [81] and MPU [78] are used in the framework, I follow the implementation details in [81] and [78], and use the Adam [24] optimizer with the fixed learning rates of 0.001 and 0.0005, respectively.

3.3.3 Evaluation Metrics

Following PU-GAN [26], I employ Chamfer distance (CD), Hausdorff distance (HD), point-to-surface distance (P2F), and uniformity for performance evaluation. CD is used to measure the average similarity between two point sets, while HD focuses on the outliers by measuring the maximum dissimilarity between two point sets, and P2F is adopted to measure the deviation between the output points and the ground-truth meshes. To evaluate the uniformity metric, I follow the work in [26] to crop the small disks of different sizes based on 1000 seeds randomly selected from the result of each testing frame. Specifically, I first calculate the total surface area of the ground-truth 3D mesh and calculate the radius for the small disk at each expected size (*e.g.*, 0.8% of the total surface area). Based on the ground-truth mesh, I then geodesically crop the small disks of different sizes around each seed. For each small disk, I first count the number of points in it, and then I calculate the distance of each point in the disk to its nearest neighbor, based on which I follow the work in [26] to calculate the uniformity score. Note that I provide the results at different percentages (*i.e.*, 0.8% and 1.2%) in Table 3.1, Table 3.2, Table 3.5 and Table 3.6. Following [26], I report the re-scaled evaluation

scores by multiplying a scale factor of 1000. In addition, for the signage "↓" in all the Tables below, lower metric values indicate better results.

3.3.4 Comparison with the State-of-the-art Approaches

I compare the proposed SPU with the state-of-the-art single frame-based point cloud upsampling methods including PU-Net [81], MPU [78], and PU-GAN [26]. To fairly compare different methods on the two point cloud sequence datasets, I retrain PU-Net, MPU, and PU-GAN by using the DYNA and COMA datasets, respectively.

Experimental results on the DYNA dataset

The upsampling results on the DYNA dataset are summarized in Table 3.1. The quantitative results clearly demonstrate the effectiveness of the proposed SPU method. By using PU-Net, MPU, or PU-GAN as the basic network for single frame-based point cloud upsampling, the proposed SPU methods (referred to as SPU (PU-Net), SPU (MPU), and SPU (PU-GAN)) outperform PU-Net, MPU, and PU-GAN by a large margin in terms of all the evaluation metrics. Besides the quantitative results, I also provide the visualization results in Fig. 3.3, where I present the upsampling results by the single frame-based methods PU-Net, MPU, PU-GAN, and the proposed SPU (PU-Net), SPU (MPU), SPU (PU-GAN). When compared with the single frame-based approaches, which can only exploit spatial shape priors, the proposed SPU framework can additionally take advantage of temporal information, thus it can produce high-quality upsampling results with more details (e.g., the hands in the red boxes) and fewer artifacts (e.g., the feet in the blue boxes). In addition, SPU (PU-Net), SPU (MPU) and SPU (PU-GAN) are able to generate the results with more uniformly distributed point clouds (e.g., the arms in the green boxes) when compared with PU-Net, MPU, and PU-GAN. It is worth noting that the sequences generated by SPU (PU-Net), SPU (MPU) and SPU (PU-GAN) have better visual quality (e.g., maintaining better shape boundaries) than the single frame-based counterparts thanks to the long-term latent representation.

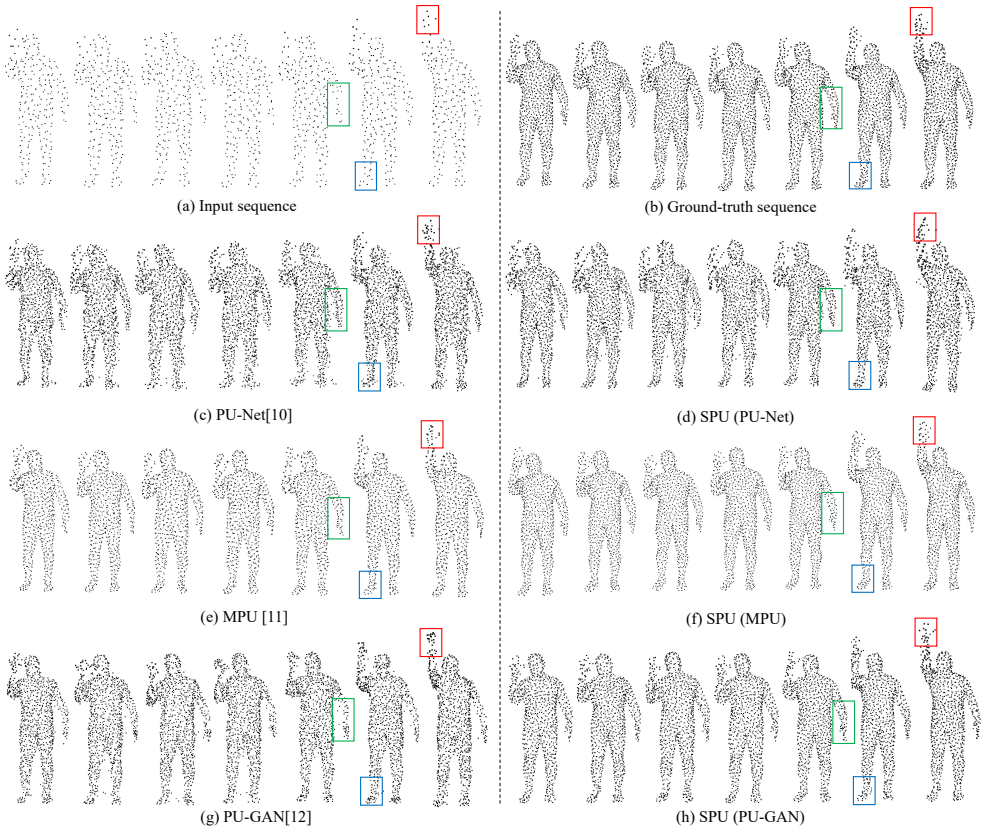


Figure 3.3: Comparison of the point cloud sequence upsampling results by using SPU (PU-Net), SPU (MPU) and SPU (PU-GAN) as well as their single frame-based counterparts PU-Net [81], MPU [78] and PU-GAN [26] on the DYNA dataset. The number of points in each input frame is 256, while the number of points in each output frame is 1024 (i.e., $r = 4$). I use a set of training sequences with each sequence consisting of 9 consecutive low-resolution frames to learn the model, which is then used to generate the high-resolution point cloud sequences with each testing sequence consisting of 7 consecutive frames.

Table 3.1: Results of different methods on the DYNA dataset.

Methods	CD ↓	HD ↓	P2F ↓	Uniformity 0.8% ↓	Uniformity 1.2% ↓
PU-Net	1.26	16.91	9.32	4.52	6.17
SPU (PU-Net)	**1.03**	**10.90**	**8.47**	**3.15**	**4.59**
MPU	0.79	5.19	4.70	1.54	2.38
SPU (MPU)	**0.75**	**4.91**	**4.42**	**1.38**	**2.16**
PU-GAN	1.26	17.24	7.60	3.76	6.11
SPU (PU-GAN)	**0.76**	**6.24**	**5.64**	**2.14**	**3.39**

Experimental results on the COMA dataset

The results on the COMA dataset are reported in Table 3.2. The proposed SPU methods, including SPU (PU-Net), SPU (MPU), and SPU (PU-GAN), still consistently outperform the single frame-based counterparts in terms of all the evaluation metrics. Since this dataset only contains the facial surfaces with different expressions, it is easier for the proposed method SPU, as well as the baseline methods, to capture the surface structures of the relatively stable faces rather than dynamically moving human bodies as on the DYNA dataset. Therefore, the P2F scores from all methods are much lower than those on the DYNA dataset, as this score indicates the distance between the generated points and the ground-truth surfaces.

In Fig. 3.4, I additionally provide some visualization results for faces with the extreme expression, from which I observe that PU-Net fails to restore the shape (e.g., the chin area) of the deformed human face with the extreme expression (see the green boxes). In contrast, SPU (PU-Net) can better recover the human shape by exploiting temporal information in adjacent frames. MPU generates a better shape for the mouth part when compared with PU-Net, however, it still produces few artifacts around the boundaries. When compared with MPU, SPU (MPU) not only recovers much clearer boundaries, but also produces more uniform upsampling results (see the orange boxes). PU-GAN generates the frames with more uniformly distributed points thanks to the uniform loss, however, it cannot reasonably recover the mouth area because of the excessive point expansion (see the blue boxes). In contrast, SPU (PU-GAN) is able to achieve better balance between the recovered details and the distribution uniformity of the recovered dense points by exploiting temporal dependency.

Testing speed comparison on the DYNA dataset

In this section, I report the testing speed of the single frame-based methods PU-Net, MPU, and PU-GAN, as well as the proposed SPU (PU-Net), SPU(MPU), and SPU (PU-GAN) under the upsampling ratio of $\times 4$ (i.e., $256 \rightarrow 1024$) on the DYNA dataset. On average, it takes 0.012, 0.008

Figure 3.4: Comparison between the single frame-based point cloud upsampling methods PU-Net [81], MPU [78], PU-GAN [26] and the proposed SPU (PU-Net), SPU (MPU), SPU (PU-GAN) for generating the faces with the extreme expression from the COMA dataset.

Table 3.2: Results of different methods on the COMA dataset.

Methods	CD ↓	HD ↓	P2F ↓	Uniformity 0.8% ↓	Uniformity 1.2% ↓
PU-Net	2.70	19.37	2.49	0.89	1.53
SPU (PU-Net)	**2.02**	**16.29**	**2.04**	**0.80**	**1.32**
MPU	0.95	5.14	0.47	0.47	0.72
SPU (MPU)	**0.87**	**4.69**	**0.42**	**0.43**	**0.69**
PU-GAN	1.37	11.00	0.93	0.55	0.83
SPU (PU-GAN)	**1.26**	**9.98**	**0.67**	**0.54**	**0.81**

Table 3.3: Results of different methods on the MSR Action3D Dataset.

Methods	CD ↓	HD ↓
PU-Net	2.601	65.414
SPU (PU-Net)	**1.648**	**27.954**
MPU	0.84	17.652
SPU (MPU)	**0.71**	**15.347**
PU-GAN	1.687	25.156
SPU (PU-GAN)	**1.074**	**17.534**

and 0.016 seconds for PU-Net, MPU, and PU-GAN to generate one single dense point cloud frame, respectively, while it takes 0.030, 0.012, and 0.020 seconds for SPU (PU-Net), SPU (MPU) and SPU (PU-GAN) to generate one dense point cloud frame, respectively. While my proposed methods are slower than the corresponding baseline methods, the testing speed is still acceptable for real-world applications.

Experiments on the MSR Action3D Dataset

To evaluate the generalization capability of SPU, I evaluate different methods on the challenging real-scanned dataset MSR Action3D [29] by directly using their models pre-trained based on the DYNA dataset, and the results are summarized in Table 3.3. Note that the ground-truth mesh data of the MSR Action3D dataset is unavailable, thus I cannot provide the P2F and uniformity scores. From Table 3.3 I observe that it is also beneficial to use the sequential point cloud upsampling method SPU to improve the upsampling results on the real-scanned dataset.

Table 3.4: Results of different methods on the DYNA datasets when using different upsampling ratios.

Methods	×2		×4		×8		×16	
	CD ↓	HD ↓	CD ↓	HD ↓	CD ↓	HD ↓	CD ↓	HD ↓
PU-Net	1.99	25.67	2.09	33.62	2.18	39.51	2.40	45.20
SPU(PU-Net)	**1.64**	**17.97**	**1.77**	**23.19**	**1.91**	**26.64**	**2.01**	**31.25**
MPU	2.01	9.39	1.95	14.70	2.06	21.72	2.20	27.96
SPU(MPU)	**1.91**	**8.92**	**1.93**	**14.12**	**1.96**	**20.42**	**2.08**	**25.73**
PU-GAN	1.61	13.13	1.58	16.92	1.64	24.89	1.76	29.12
SPU(PU-GAN)	**1.18**	**9.16**	**1.21**	**11.25**	**1.23**	**15.29**	**1.24**	**19.68**

Experiments when using different upsampling ratios

In addition to the default upsampling ratio ×4, I also take the DYNA dataset as an example to compare SPU with the single frame-based methods at four different upsampling ratios (i.e., 2, 4, 8, and 16). Specifically, I train all the methods based on the upsampling ratio ×2, and then in the testing stage, I iteratively perform the ×2 upsampling process multiple times according to the predefined upsampling ratio. The results are reported in Table 3.4. As the upsampling ratio increases, the CD score varies slightly, and the HD score becomes much larger, possibly due to the accumulated errors from the artifacts. In addition, the proposed SPU methods still outperform their single frame-based counterparts at different upsampling ratios, thanks to the long-term feature $\mathbf{H}_{t-1}$, which can well maintain the global shape of the human body.

3.3.5 Ablation Study

To verify the effectiveness of various components in SPU, I take the DYNA dataset as an example to conduct an ablation study, in which PU-GAN [26] is used as the single frame point cloud upsampling method. Specifically, I firstly show the contribution of each individual component, including the proposed TAM 3.2.2 and GFFM 3.2.3, and then I compare the results of SPU when using different temporal clues as the input (i.e., three low-resolution short-term features, one high-resolution short-term feature, and a long-term latent representation).

Table 3.5: Comparison of SPU (PU-GAN) and two simplified versions without using one or two individual modules on the DYNA dataset.

Methods	CD ↓	HD ↓	P2F ↓	Uniformity 0.8% ↓	Uniformity 1.2% ↓
w/o TAM & GFFM	0.88	6.66	7.95	2.63	3.99
w/o GFFM	0.81	11.15	6.25	2.40	3.71
SPU (PU-GAN)	**0.76**	**6.24**	**5.64**	**2.14**	**3.39**

Different modules in SPU

To analyze the contributions of the proposed TAM 3.2.2 and GFFM 3.2.3 modules, I remove either module from SPU (PU-GAN) and report the results of these two variants on the DYNA dataset. The upsampling results are reported in Table 3.5, where the first alternative method "**w/o GFFM**" denotes SPU after removing the GFFM module, and the second alternative method "**w/o TAM & GFFM**" denotes SPU after removing both TAM and GFFM modules from SPU (i.e., the inputs from multi-frames are simply aggregated to produce the fused feature $\mathbf{F}_{fuse}$ before being used as the input to PU-GAN). Without using the proposed TAM and GFFM modules, the second alternative method "**w/o TAM & GFFM**" achieves worse upsampling results in terms of all the evaluation metrics when compared with SPU. By additionally using the TAM module to better align temporal features, the first alternative method "**w/o GFFM**" can reduce the overall matching errors (e.g., CD and P2F distance) when compared with the second alternative method "**w/o TAM & GFFM**". However, some imperfect alignment introduced by the first alternative method "**w/o GFFM**" may also bring more outliers to the output sequences. As a result, the first alternative method "**w/o GFFM**" is much worse than the second alternative method "**w/o TAM & GFFM**" in terms of HD. A possible explanation is the evaluation criterion HD accounts for the largest matching errors in each frame, thus even a fraction number of outliers may lead to a large HD score. After introducing the GFFM module to effectively fuse multiple transformed features, SPU can alleviate the aforementioned outlier issue and further reduce the overall matching errors.

Table 3.6: Results of SPU (PU-GAN) when using different inputs on the DYNA dataset.

Methods	CD ↓	HD ↓	P2F ↓	Uniformity 0.8% ↓	Uniformity 1.2% ↓
w/o Q & H	0.80	9.48	6.44	2.39	3.72
w/o Q	0.81	6.37	6.06	2.27	3.55
w/o H	0.79	6.36	5.80	2.18	3.41
SPU (PU-GAN)	**0.76**	**6.24**	**5.64**	**2.14**	**3.39**

SPU using different inputs

I also analyze the contributions of different inputs in SPU (PU-GAN). To validate the contribution of each input, I either remove the short-term high quality output $\mathbf{Q}_{t-1}$ or the long-term latent representation $\mathbf{H}_{t-1}$ or both $\mathbf{Q}_{t-1}$ and $\mathbf{H}_{t-1}$ from the inputs to the SPU framework, which are referred to as "**w/o Q**", "**w/o H**" and "**w/o Q & H**", respectively. As shown in Table 3.6, when compared with the baseline method "PU-GAN" in Table 3.1, the third alternative method "**w/o Q & H**" achieves much better results in terms of all the evaluation metrics, which demonstrates it is useful to use the low-resolution short-term features from the neighboring frames. However, when compared with the proposed SPU (PU-GAN), the performance of the third alternative method "**w/o Q & H**" drops significantly in terms of the HD score, which indicates that the temporal clues merely from the low-resolution short-term features are not sufficient to handle the outliers. By separately adding $\mathbf{H}_{t-1}$ and $\mathbf{Q}_{t-1}$ into the inputs, the first alternative method "**w/o Q**" achieves significant improvement in terms of the HD score thanks to the shape prior provided by the long-term latent representation, while the second alternative method "**w/o H**" achieves more performance improvement in terms of P2F and the uniformity score due to the fine-grained local details provided by the high-resolution estimation. SPU (PU-GAN) achieves the best result, which indicates that it is beneficial to use all types of inputs in the upsampling process.

3.4 Summary

In this chapter, I have proposed a new framework SPU for the sequential point cloud upsampling task by effectively exploiting rich and mixed temporal dependency from multi-scale, long-term/short-term clues, including the neighboring low-resolution point clouds, the previous high-quality output, and the previous latent representation. In order to tackle the alignment issue between two frames caused by the intrinsic orderless characteristic of point clouds, I propose a new feature-level temporal alignment module together with a gated feature fusion module to effectively align these temporal clues and generate the fused feature for the current frame. The fused feature can be readily incorporated with various existing single frame-based point cloud upsampling methods to further enhance the point cloud upsampling result for the current frame. The proposed SPU achieves significant performance improvement when compared with the single frame-based counterparts on three benchmark datasets.

Although I have achieved significant performance gains when compared with single frame-based methods, the limitations still exist in the SPU framework. First, to produce dense point clouds with better spatial-temporal coherence, I follow the similar setting as in the video-based face hallucination work [88], which uses the whole frame as the input of the network instead of the commonly used patch-based strategy in the single frame-based methods [81, 78, 26]. This strategy may work reasonably well on the datasets with single 3D objects (*e.g.*, human body, human face, *etc.*), but it may fail for more complex scenarios with large-scale point cloud scenes, such as those in autonomous driving applications. Second, the SPU framework can also achieve satisfactory upsampling results when the number of input points increases, however, large memory is consumed under this case due to the attention mechanism used in the framework. Therefore, how to extend the SPU framework to incorporate the patch-based strategy for practical scenarios without sacrificing global shape consistency and computing efficiency still remains a challenge.

Chapter 4

Patch-based Sequential Point Cloud Upsampling Framework

According to the limitation of SPU proposed in the last chapter, I propose a simple yet effective patch-based sequential point cloud upsampling framework called VPU to extend the previous framework into a more efficient and generalized benchmark, which depends on a new temporal patch-cropping strategy and a corresponding spatio-temporal information collecting system. Extensive experiments are performed to demonstrate the effectiveness of the novel designs for this patch-based framework.

4.1 Motivations and Contributions

In Chapter 3, I have proposed the SPU framework to effectively represent and aggregate the motion between frames, which relies on a latent shape representation learned from multiple point cloud frames to generate globally consistent upsampling results. However, SPU fails to achieve satisfying performance on complex scenarios with large-scale point clouds due to its frame-based strategy and the use of the attention mechanism. Previous single frame-based point cloud upsampling methods usually adopt a patch-based strategy to improve the generalizability under more complex scenarios. However, this patch-based strategy can hardly be adopted when processing point cloud sequences [20, 71, 34] due to the patch mismatch issue between adjacent frames. Therefore, for

each local patch, how to capture reliable and complementary information from the point cloud sequence still remains an open problem.

In this chapter, I propose a new patch-based framework for the sequential point cloud upsampling task (referred to as VPU) to effectively exploit spatio-temporal information from a sequence of unordered, sparse and irregular point cloud frames. VPU consists of a spatio-temporal aggregation (STA) module and a single frame-based point cloud upsampling (SFPU) module, where the second module can be adopted from various existing single frame-based point cloud upsampling methods, such as PU-Net [81], MPU [78], PU-GAN [26] and PU-GCN [52]. Instead of explicitly using motion prediction among adjacent point cloud sequences as in the existing video super-resolution works [22, 32, 33, 69, 18], as the vital component in VPU, the newly proposed STA module can effectively *extract*, *align* and *aggregate* spatio-temporally consistent and complementary local structural clues from consecutive frames and produce rich local structural features. These aggregated features could more reliably indicate local geometry distribution around each target point at the current frame, which will thus benefit the subsequent generation process for producing more evenly distributed and geometrically consistent dense point clouds. Specifically, in the STA module, I *extract* a set of cross/within-frame features by using the cross/within-frame feature combination submodule. Concretely, for each target point in the current frame, I first use the k-nearest-neighbor (kNN) method to search for a set of neighboring points from the reference frame (*i.e.*, the previous/current/subsequent frame) at the feature space and then generate a set of 3D offset features relative to this target point, based on which I produce a set of cross/within-frame individual features encoding the local structure information of the neighborhood. For each target point, I then group these individual features of neighboring points from the reference frame to produce an intermediate feature for this target point. After that, I generate the cross/within-frame combined feature by aggregating the intermediate features from all the target points in the current frame. Thereafter, inspired by the cross-attention mechanism [46], I propose a non-local alignment submodule to softly *align* each cross-frame individual feature with respect to each

within-frame individual feature, which eventually produces the transformed feature with spatio-temporally complementary and consistent local structural information. By further adopting a set of operations including concatenation, reshaping, max-pooling and multi-layer perceptrons (MLPs) on these generated transformed features and the within-frame combined feature, I finally produce the *aggregated* feature for all the target points, which is ready to be fed into the SFPU module for generating the dense point clouds for the current frame.

Extensive experiments and ablation study on multiple point cloud sequence datasets demonstrate that the newly proposed VPU achieves remarkable performance improvement gains over the existing single frame-based counterparts, including PU-Net [81], MPU [78], PU-GAN [26] and PU-GCN [52].

The contributions are three-fold: (1) To the best of my knowledge, VPU is the first patch-based deep learning approach to explicitly exploit the temporal clues for the sequential point cloud upsampling task. (2) As a general framework, I can readily incorporate the existing single frame-based point cloud upsampling methods within the VPU framework by using them as part of the modules. (3) The VPU framework achieves the state-of-the-art results on multiple point cloud sequence datasets.

4.1.1 Discussion

The works most related to VPU are all the single frame-based point cloud upsampling methods and the sequential point cloud upsampling framework SPU proposed in the last chapter.

Unlike the existing single frame-based methods that focus on how to achieve satisfying restoration directly from the sparse and non-uniform input point clouds, VPU, similar as SPU, tends to explore how to better recover the distorted point clouds by using the spatio-temporal hints from the point cloud sequences. However, SPU aims to reconstruct fine-grained and temporally consistent point cloud sequences by exploiting different kinds of temporal dependencies in a recurrent manner, which inevitably increases the computing cost. Moreover, SPU can only be

performed on entire point cloud frames instead of local patches due to leveraging the global shape latent representation, which also causes negative effects on its efficiency and working scenarios. Different from SPU, the goal of the VPU framework is to build a more efficient and more generalized benchmark that can be applied under real-world scenarios. Therefore, the study in VPU focuses more on integrating the patch-based strategy into the sequential point cloud upsampling framework and addressing the new problems in conjunction with that.

4.2 Methodology

4.2.1 Overview of the VPU Framework

The overview of the proposed sequential point cloud upsampling framework (VPU) is shown in Fig. 4.1, where three frames are used as an example for better illustration. It consists of a spatio-temporal aggregation (STA) module and a single frame-based point cloud upsampling (SFPU) module, where the recent single frame-based point cloud upsampling methods like PU-Net [81], MPU [78], PU-GAN [26] and PU-GCN [52] can be readily incorporated in the second module SFPU. In Sec. 4.2.2, I will introduce the VPU framework in detail.

4.2.2 Network Architecture

Spatio-Temporal Aggregation (STA) Module

Step-I: Feature Extraction. As shown in Fig. 4.1, I first extract the initial features of the point cloud data from three consecutive frames $\mathcal{P}_{t-1}$, $\mathcal{P}_t$ and $\mathcal{P}_{t+1}$ by using a set of shared MLPs, which are denoted as $\mathbf{F}_{t-1}$, $\mathbf{F}_t$, and $\mathbf{F}_{t+1} \in \mathbb{R}^{N \times d}$. Then I use a cross-frame feature combination (CFFC) submodule and a within-frame feature combination (WFFC) submodule to *extract* the cross/within-frame combined feature with local geometry structure information.

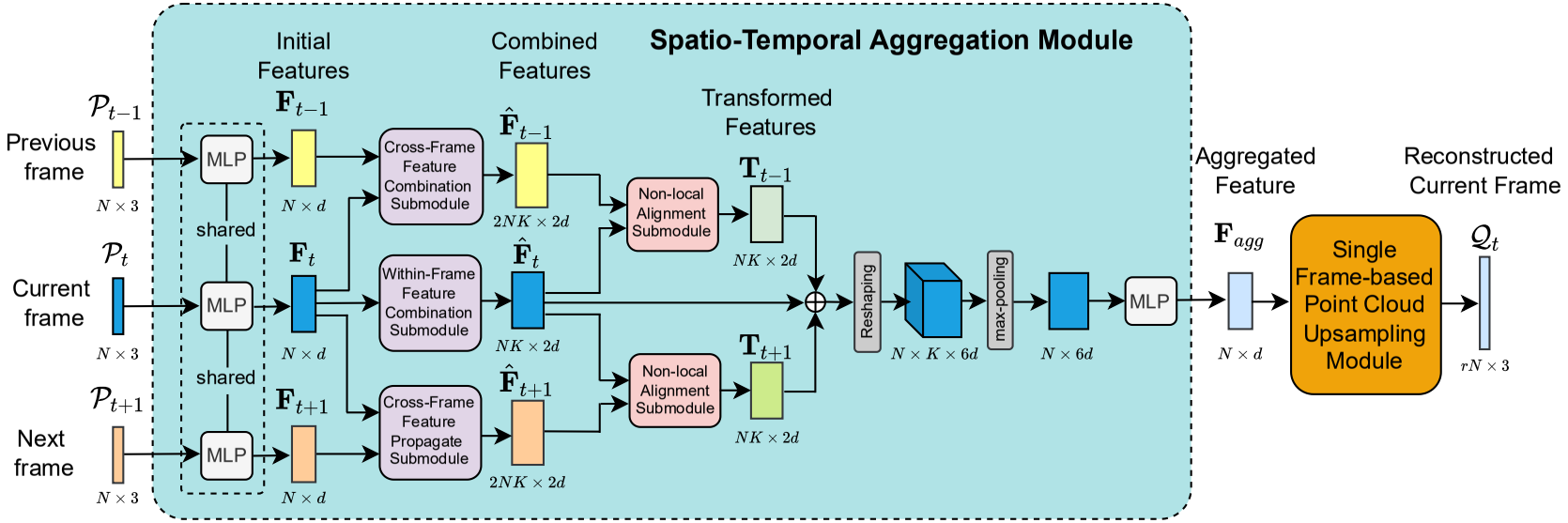


Figure 4.1: Overview of the VPU framework (three frames are used as an example for better illustration). Given the sparse and non-uniform point sets from three consecutive frames (*i.e.*, $\mathcal{P}_{t-1}$, $\mathcal{P}_t$ and $\mathcal{P}_{t+1}$) fed into the spatio-temporal aggregation (STA) module, I first project the 3D coordinates of these points into the feature space, and produce the initial features $\mathbf{F}_{t-1}$, $\mathbf{F}_t$ and $\mathbf{F}_{t+1}$ by using a set of shared MLPs. Then I extract the combined features $\hat{\mathbf{F}}_{t-1}$, $\hat{\mathbf{F}}_t$ and $\hat{\mathbf{F}}_{t+1}$ from the adjacent/current frame by using the cross/within-frame feature combination submodule. To further achieve feature-level alignment across multiple frames, I use the non-local alignment (NLA) submodule to transform the combined features $\hat{\mathbf{F}}_{t-1}$ and $\hat{\mathbf{F}}_{t+1}$ to the transformed features $\mathbf{T}_{t-1}$ and $\mathbf{T}_{t+1}$. The transformed features together with $\hat{\mathbf{F}}_t$ are used to produce the final aggregated feature $\mathbf{F}_{agg}$ after performing a set of operations including concatenation, reshaping, max-pooling and MLP. Lastly, I feed the aggregated feature $\mathbf{F}_{agg}$ into the single frame-based point cloud upsampling (SFPU) module to generate the reconstructed point cloud $\mathcal{Q}_t$ for the current frame. k denotes the number of neighbors searched from the adjacent/current frame. Note that I set $k = 2K$ and $k = K$ for the adjacent frame and the current frame, respectively. $\oplus$ denotes the concatenation operation along the feature dimension.

Specifically, when using the CFFC submodule, the reference frames can be $\mathcal{P}_{t-1}$ and $\mathcal{P}_{t+1}$, and here I take $\mathcal{P}_{t-1}$ as an example to describe the details. For the i-th 3D point (*i.e.*, the target point) $\mathbf{p}_t^i$ with its feature represented as $\mathbf{f}_t^i \in \mathbb{R}^d$ from the current frame $\mathcal{P}_t$, I search for top-k nearest-neighbors from $\mathcal{P}_{t-1}$ at the d-dimension feature space, which are denoted as $\mathbf{p}_{t-1}^j \in \mathbb{R}^3, j = 1, \ldots, k$ with their features represented as $\mathbf{f}_{t-1}^j \in \mathbb{R}^d, j = 1, \ldots, k$. Then I define $N_k(\mathbf{p}_t^i)$ as the index set of these neighboring points from the previous frame. Following the encoding strategy in DGCNN [68], the local neighborhood structure information between $\mathbf{p}_{t-1}^j$ and $\mathbf{p}_t^i$ can be expressed as the offset feature $\mathbf{f}_{t-1}^j - \mathbf{f}_t^i$ at the feature space. Then I combine the geometry information captured by the per-point feature $\mathbf{f}_t^i$ of the target point $\mathbf{p}_t^i$ from the absolute coordinate space, with the local neighborhood structure information captured by the offset feature $\mathbf{f}_{t-1}^j - \mathbf{f}_t^i$ from the relative feature space, based on which I define the cross-frame individual feature as follows:

$$\mathbf{r}_{t-1}^j = [\mathbf{f}_{t-1}^j - \mathbf{f}_t^i, \mathbf{f}_t^i] \in \mathbb{R}^{2d}, \forall j \in N_k(\mathbf{p}_t^i). \tag{4.1}$$

Note that $\mathbf{r}_{t-1}^j$ explicitly encodes cross-frame local structure information by combining both local neighborhood structure information from adjacent frames (represented by the offset feature $\mathbf{f}_{t-1}^j - \mathbf{f}_t^i$) and global geometry information (represented by $\mathbf{f}_t^i$) from the target point $\mathbf{p}_t^i$, which forms complementary local structural clues for the target point $\mathbf{p}_t^i$. By grouping all cross-frame individual features $\mathbf{r}_{t-1}^j, \forall j \in N_k(\mathbf{p}_t^i)$ in the local neighborhood from the previous frame, I can produce the cross-frame intermediate feature $\mathbf{R}_{t-1}^i \in \mathbb{R}^{k \times 2d}$ for the target point $\mathbf{p}_t^i$ in the current frame. Totally I have a set of cross-frame intermediate features $\{\mathbf{R}_{t-1}^i \in \mathbb{R}^{k \times 2d}, i = 1, \ldots, N\}$ from all points in the current frame $\mathcal{P}_t$, based on which, I define the cross-frame combined feature $\hat{\mathbf{F}}_{t-1}$ by aggregating all $\mathbf{R}_{t-1}^i$s as follows:

$$\hat{\mathbf{F}}_{t-1} = [(\mathbf{R}_{t-1}^1)^\top, (\mathbf{R}_{t-1}^2)^\top, \ldots, (\mathbf{R}_{t-1}^N)^\top]^\top \in \mathbb{R}^{Nk \times 2d}. \tag{4.2}$$

Similarly, in the WFFC submodule, I can produce the within-frame combined feature $\hat{\mathbf{F}}_t$ by using the current frame itself as the reference frame. Considering the dynamic movement of objects across the consecutive

frames, for each target point in the current frame, I use a larger searching range (*i.e.*, $k = 2K$) to assemble more related features from the adjacent frames in the CFFC submodule, and employ the normal searching range (*i.e.*, $k = K$) in the WFFC submodule. Note this strategy is similar as the direct grouping strategy used for the sequential point cloud classification/segmentation task in MeteorNet [35].

Step-II: Feature Alignment. Taking $\hat{\mathbf{F}}_t$ as the target, I feed the cross-frame combined features $\hat{\mathbf{F}}_{t-1}$ and $\hat{\mathbf{F}}_{t+1}$ into the non-local alignment (NLA) submodule to respectively produce the transformed features $\mathbf{T}_{t-1}$ and $\mathbf{T}_{t+1} \in \mathbb{R}^{Nk \times 2d}$ by using the non-local cross-attention strategy designed for estimating point correspondences [46] and feature alignment [63], which will be introduced in Sec. 4.2.2.

Step-III: Feature Aggregation. After concatenating the two transformed features $\mathbf{T}_{t-1}$, $\mathbf{T}_{t+1}$ and the within-frame combined feature $\hat{\mathbf{F}}_t$ along the feature dimension, I produce the feature $[\hat{\mathbf{F}}_t, \mathbf{T}_{t-1}, \mathbf{T}_{t+1}] \in \mathbb{R}^{Nk \times 6d}$, and then I adopt a reshaping operation to convert this feature to the three-frame feature with the dimension of $N \times k \times 6d$. Based on this three-frame feature, I finally produce the aggregated feature as $\mathbf{F}_{agg} \in \mathbb{R}^{N \times d}$ by using a max-pooling operation over each point of the current frame, and a set of MLPs.

Non-Local Alignment (NLA) Submodule

The NLA submodule is an important component in the proposed STA module. Due to the orderless characteristics of point cloud, the extracted cross-frame combined features $\hat{\mathbf{F}}_{t-1}$ and $\hat{\mathbf{F}}_{t+1}$ are not point-wisely aligned with respect to the within-frame combined feature $\hat{\mathbf{F}}_t$ in the feature space, which makes it difficult to directly perform feature-level fusion by using the per-point concatenation operation [16] along the feature dimension. To this end, similarly as the cross-shape attention mechanism in [46, 63], I perform feature alignment from either $\hat{\mathbf{F}}_{t-1}$ or $\hat{\mathbf{F}}_{t+1}$ with respect to $\hat{\mathbf{F}}_t$, which enables cross-frame non-local contextual feature aggregation in the next step.

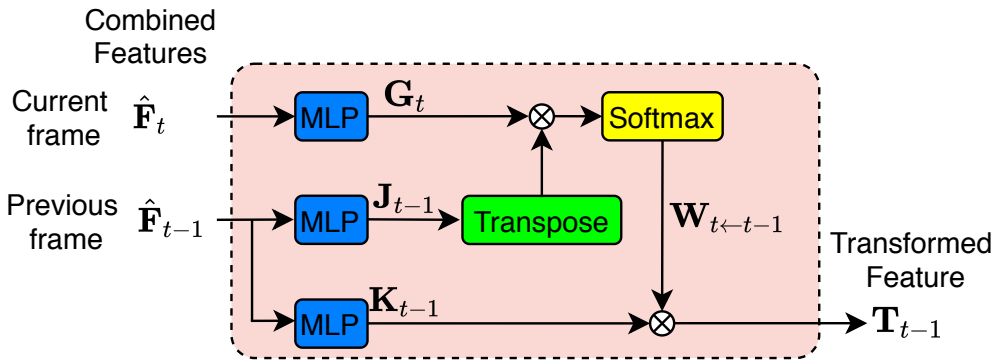


Figure 4.2: Architecture of the Non-Local Alignment Submodule.

As shown in Fig. 4.2, I take the alignment from $\mathbf{F}_{t-1}$ to $\mathbf{F}_t$ as an example. The reference feature $\hat{\mathbf{F}}_{t-1} \in \mathbb{R}^{2NK\times 2d}$ and the target feature $\hat{\mathbf{F}}_t \in \mathbb{R}^{NK\times 2d}$ (recall the searching ranges are $k = 2K$ and $k = K$ in the CFFC and WFFC submodules, respectively) are at first projected into the C-dimension common feature space through two parallel MLPs without shared weights, denoted as $\mathbf{J}_{t-1} \in \mathbb{R}^{2NK\times C}$ and $\mathbf{G}_t \in \mathbb{R}^{NK\times C}$, respectively. Similarly as in the self-attention mechanism [83], I calculate a similarity matrix

$$\mathbf{W}_{t\leftarrow t-1} = \texttt{softmax}(\mathbf{G}_t\mathbf{J}_{t-1}^{\top}), \tag{4.3}$$

where the softmax operation is conducted in each row of $\mathbf{W}_{t\leftarrow t-1} \in \mathbb{R}^{NK\times 2NK}$. I further project $\hat{\mathbf{F}}_{t-1}$ into the same $2d$-dimension feature space as $\hat{\mathbf{F}}_t$ through another separated MLP, which is referred to as $\mathbf{K}_{t-1} \in \mathbb{R}^{2NK\times 2d}$. Finally, the aligned reference feature (referred to as the transformed feature) is defined below:

$$\mathbf{T}_{t-1} = \mathbf{W}_{t\leftarrow t-1}\mathbf{K}_{t-1}. \tag{4.4}$$

Similarly, I can produce another transformed feature $\mathbf{T}_{t+1}$ based on $\hat{\mathbf{F}}_{t+1}$ and $\hat{\mathbf{F}}_t$ accordingly.

Discussion

Although the CFFC and WFFC submodules in the ***extraction*** step of the STA 4.2.2 module follow a similar local geometry encoding strategy (*a.k.a.*, encoding the offset features as local structure information) as in

DGCNN [68], their core design motivations are intrinsically different.

DGCNN focuses on *selectively* extracting necessary within-frame local structure information from a point set with complete shape, which is beneficial to the high-level classification/segmentation tasks (by using edge-conv [68]). However, for a sparse and non-uniform input, the encoded geometry information becomes less representative due to the missing points in certain areas, which will further influence the final upsampling results. To compensate the information loss at the current frame, the proposed submodules (*i.e.*, CFFC and WFFC) are designed to construct a spatio-temporal collection system by gathering *sufficient* local structural information across multiple consecutive frames. Specially, the offset feature extracted by the CFFC submodule encodes both local geometry structure and motion information. Although dynamic movements of the 3D object inevitably result in misalignment between the same part in adjacent frames, considering that the motions between adjacent frames in the point cloud sequences are usually moderate, the CFFC submodule is able to collect cross-frame local structure information that can be well ***aligned*** with respect to the within-frame local structure information by using the newly proposed NLA 4.2.2 submodule.

Also note that during the non-local alignment process in the *alignment* step of the STA module, I calculate the similarity between each pair of individual features respectively from the reference frame and the current frame to produce the corresponding weight value for soft alignment through the attention mechanism, no matter whether these individual features are gathered from the same target point, thus the STA module can produce more comprehensive local geometrical clues.

Single Frame-based Point Cloud Upsampling (SFPU) Module

As discussed in Sec.III-A, with the spatio-temporally aggregated feature $\mathbf{F}_{agg}$, the VPU directly employs the existing single frame-based point cloud upsampling methods, such as PU-Net [81], MPU [78], PU-GAN [26] and PU-GCN [52] to generate the dense point cloud for the current frame. Generally, the architecture of these single frame-based

upsampling methods can be simply regarded as an encoder-decoder network, which consists of a per-point feature extraction module and an upsampling module. The per-point feature extraction module aims to extract the features from the input point cloud $\mathcal{P}_t$, and the upsampling module is designed to upsample the extracted feature and further project the upsampled feature back to the point cloud coordinate space. Here, I briefly review the network architectures used in these single frame-based methods and for more specific details, please refer to [81], [78], [26] and [52].

PU-Net [81]: PU-Net adopts a multi-layer PointNet++ structure as the feature extraction module. Based on the extracted feature, it uses r different MLPs as the upsampling module to generate r diverse feature sets, where r denotes the upsampling ratio. These expanded feature sets are merged to generate the dense point clouds.

MPU [78]: MPU adopts a multi-layer DGCNN-like architecture [16] with dense connection as the feature extraction module. For the upsampling module, it first duplicates the extracted feature r times and assigns different codes for the duplicated features. Then it uses a set of MLPs to convert these features into the residual information, which is eventually added to the point cloud coordinates to generate the dense point clouds.

PU-GAN [26]: PU-GAN shares the same feature extraction module with MPU and additionally introduces an "up-down-up" expansion unit as the upsampling module, where the "up-feature operator" uses a self-attention layer and a set of MLPs to expand the point features, and the "down-feature operator" uses another set of MLPs to regress the original feature. In addition, PU-GAN also employs a discriminator, which consists of a two stacked PointNet layers and a self-attention layer.

PU-GCN [52]: PU-GCN employs an Inception DenseGCN block as the feature extraction module, which contains two DenseGCN blocks, a bottleneck layer and a global pooling layer. For the upsampling module, PU-GCN introduces a "NodeShuffle" layer composed of a GCN layer and a periodic shuffling operator, which first uses the GCN layer to expand the node features and then rearranges the output to the required

shape.

Instead of directly feeding the non-uniform single frame input $\mathcal{P}_t$ to the per-point feature extraction module, the VPU uses the aggregated feature $\mathbf{F}_{agg}$ as the input, which can be regarded as an improved feature for the current frame by exploiting rich spatio-temporal neighborhood information.

4.2.3 Loss Functions

For the compatibility with the existing single frame-based methods used in the SFPU module, I accordingly adopt their own loss functions to train the proposed VPU methods, which are denoted as VPU (PU-Net), VPU (MPU), VPU (PU-GAN) and VPU (PU-GCN). Following PU-Net [81], the loss function of VPU (PU-Net) consists of a reconstruction loss term and a repulsion loss term, which is defined as:

$$L_{total} = L_{rec} + \lambda_{rep} L_{rep}, \tag{4.5}$$

where L_{rec} denotes the Earth Mover's Distance (EMD) between the generated dense point cloud and the ground-truth point cloud, and L_{rep} is the repulsion loss term to prevent the points from clustering at a single position. I empirically set $\lambda_{rep} = 1$ as in [81]. For the loss function in VPU (PU-GAN), I follow the adversarial training strategy and adopt the loss functions of PU-GAN [26], which are defined below:

$$\begin{aligned} L_G &= \lambda_{gan} L_{gan} + \lambda_{rec} L_{rec} + \lambda_{uni} L_{uni}, \\ L_D &= L_{dis}. \end{aligned} \tag{4.6}$$

$\mathcal{L}_G$ is used to train the network of VPU (PU-GAN), while $\mathcal{L}_D$ is employed to train the discriminator used in PU-GAN. L_{gan} and L_{dis} are a pair of adversarial losses, which are defined based on the existing work LSGAN [40]. L_{rec} denotes the Earth Mover's Distance (EMD) between the generated dense point cloud and the ground-truth point cloud, and L_{uni} is a loss term used to improve the uniformity of the generated point clouds by enforcing additional constraints on small disks from the surface. As in [26], I empirically set $\lambda_{gan} = 0.5$, $\lambda_{rec} = 100$, and

$\lambda_{uni} = 10$. For the loss functions in VPU (MPU) and VPU (PU-GCN), I follow MPU [78] and PU-GCN [52] to use a single reconstruction loss term L_{rec} to minimize the distance between the generated point cloud and the ground truth point cloud, where L_{rec} denotes the Chamfer Distance (CD)-based loss.

4.3 Experiments

4.3.1 Experimental Setup

Dataset Preparation

To evaluate VPU, I first conduct extensive experiments on a mesh-based human action dataset, DYNA [47], and a real-scanned point cloud-based human action dataset MSR Action3D [29]. Then, to further evaluate the generalization capability of the framework, I conduct two additional experiments on the COMA [54] and KITTI [15] datasets by using the models pre-trained on the DYNA dataset.

Considering that 3D shapes are described as meshes on the DYNA and COMA datasets, I additionally apply the Poisson-disk sampling strategy [9] to sample 8,192 points from each mesh frame as the ground-truth. For the MSR Action3D dataset, I use the raw data in the point cloud sequences as the ground-truth. Note that the point clouds in MSR Action3D are converted from the depth maps, and thus they are non-uniform in nature.

The DYNA dataset [47] consists of 134 sequences from 10 subjects with significant body movement, where each sequence contains 145 to 950 frames. 107 sequences from 8 randomly selected subjects are used as the training set and the remaining 27 sequences from 2 subjects are used in the testing set. The COMA dataset consists of 120 sequences from 10 subjects with varying facial expressions, in which each sequence has 15 to 345 frames. I use the sequences from all the 10 subjects for performance evaluation. Specifically, for the sequences with more than 23 frames, I select 23 consecutive frames from each sequence, while I use all frames for the other sequences. The MSR Action3D dataset [29] contains

567 point cloud sequences captured from 10 subjects when performing 20 different human actions. I choose 100 sequences from 5 randomly selected subjects as the testing dataset and use the remaining data for training. For the KITTI [15] dataset, since the raw data is much sparser when compared with the MSR Action3D dataset, and the ground-truth dense point clouds are unavailable, I only conduct a qualitative experiment on the KITTI dataset. Following [82], I perform necessary pre-processing to align and normalize the scanned point cloud data of each object before feeding them into VPU.

Considering that the total number of sequences on DYNA, COMA, and MSR Action3D datasets is limited, I prepare a large number of 3-frame triplets in order to provide sufficient training and testing data. For example, for a sequence with 30 frames, I produce 28 overlapped triplets by using a sliding window with the length of three frames. Totally, I have 33,580 (*resp.* 1,701) triplets for training (*resp.* testing) on the DYNA dataset and 2,972 triplets for testing on the COMA dataset, respectively. Similarly, I also produce 17,854 (*resp.* 3,809) triplets for training (*resp.* testing) on the MSR Action3D dataset. It is worth noting that for each triplet, I only evaluate the upsampling result from the middle frame for fair comparison with the single frame-based approaches in the testing stage.

Implementation Details

The proposed network is implemented by using TensorFlow and the experiments are performed on the machine with one NVIDIA 1080Ti GPU. When PU-GAN [26] is incorporated into the VPU framework, I set the batch size as 4 and the initial learning rates for the generator and the discriminator as 0.0005 and 0.0001, respectively, which are reduced by a decay rate of 0.8 after every 30,000 iterations. When PU-Net [81], MPU [78] and PU-GCN [52] are used, I follow the implementation details described in their works, and use the Adam [24] optimizer with the fixed learning rate of 0.0005.

Following the patch-based training strategy in MPU [78] and PU-GAN [26], for each triplet in the training set, I first use the farthest point

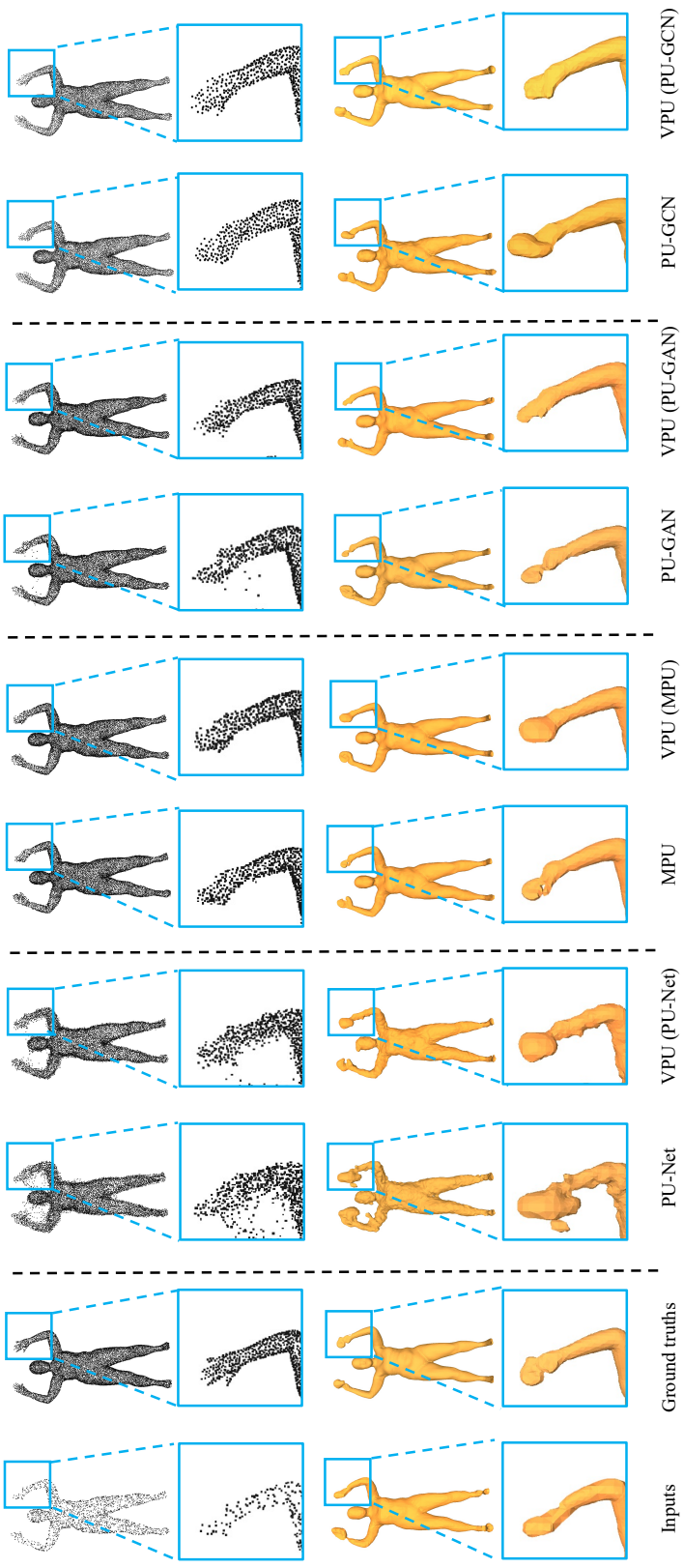


Figure 4.3: Comparison of the upsampling results (see the first row) and the surface reconstruction results (see the third row) on the DYNA dataset when using different methods under the upsampling ratio $\times 4$. I additionally provide their close-up views to compare the upsampling results (see the second row) and surface reconstruction results (see the fourth row) around the right hand (*i.e.*, the cyan boxes).

sampling (FPS) method to select a fixed number of seeds in the current ground-truth frame, and use kNN to crop the patches centered at these seeds from all three consecutive frames, where the dense patch in each frame has 512 points. Each input sparse patch has 128 points, which are randomly sampled from the ground-truth dense patches (*i.e.*, the upsampling rate $r = 4$). In the testing phase, I use the same patch-based evaluation strategy in PU-GAN [26], in which for each input frame I randomly select 2,048 points from each ground-truth frame in the testing dataset. By using the same patch extraction method as in the training phase, I produce a fixed number of sparse input patches, with each patch consisting of 128 points. Then, I feed the input patches to the network, merge the upsampling results from all patches, and use FPS to downsample the merged output to generate the final dense point cloud frame with the required number of points.

4.3.2 Experimental results

As the proposed VPU is the first sequential point cloud upsampling framework, and it is compatible with multiple single frame-based methods, I only compare VPU with four single-frame counterparts, *i.e.*, PU-Net [81], MPU [78], PU-GAN [26] and PU-GCN [52], and the corresponding VPU methods are referred to as VPU (PU-Net), VPU (MPU), VPU (PU-GAN) and VPU (PU-GCN) in the following Tables and Figures. Note that by default I set $K = 8$ in the following experiments (*i.e.*, $k = 8$ for the WFFC submodule and $k = 16$ for the CFFC submodule).

Experimental results on the DYNA Dataset

The upsampling results on the DYNA dataset are summarized in Table 4.1. The quantitative results clearly demonstrate the effectiveness of the proposed VPU framework. By using either PU-Net, MPU, PU-GAN or PU-GCN as the single-frame point cloud upsampling module in the framework, the proposed VPU methods outperform these single frame-based counterparts by a large margin in terms of all the evaluation metrics.

Table 4.1: Results of different methods on the DYNA dataset.

Methods	CD ↓	HD ↓	P2F ↓	Uniformity for different p		
				0.4% ↓	0.8% ↓	1.2% ↓
PU-Net	0.637	19.314	10.343	8.447	15.127	19.928
VPU (PU-Net)	**0.377**	**11.391**	**6.439**	**6.392**	**11.749**	**15.499**
MPU	0.181	2.928	1.545	3.151	6.277	7.924
VPU (MPU)	**0.173**	**2.620**	**1.432**	**2.931**	**5.788**	**7.308**
PU-GAN	0.215	8.912	2.741	4.763	9.329	12.193
VPU (PU-GAN)	**0.168**	**2.731**	**1.684**	**2.759**	**5.522**	**7.469**
PU-GCN	0.212	2.799	1.362	2.935	5.818	7.691
VPU (PU-GCN)	**0.158**	**2.315**	**1.114**	**2.779**	**5.644**	**7.538**

Besides the quantitative results, in Fig. 4.3 I also provide a visualization example, including point set upsampling and the surface reconstruction (by using [23]) results. From Fig. 4.3, I can observe that the baseline method PU-Net cannot preserve the human body shape due to the outliers generated above the surface, while VPU (PU-Net) is able to eliminate most of the outliers and produce relatively clear boundary for the entire human body area. MPU, PU-GAN and PU-GCN manage to generate better upsampling results with much fewer outliers, however, they still cannot correctly reconstruct the details on some local areas that are occasionally missed in the input sequences, such as hands (see the cyan boxes). After using VPU (MPU), VPU (PU-GAN) and VPU (PU-GCN), I can more reasonably generate the details on these local areas (also see the hand parts from the surface reconstruction results) and recover clearer boundaries by effectively extracting, aligning and aggregating the complementary local structural information for the current frame.

Experimental results on the MSR Action3D Dataset

In addition to the benchmark dataset DYNA, I also evaluate the proposed method on the more challenging dataset MSR Action3D [29]. The upsampling results of different methods on the MSR Action3D dataset under the ratio $\times 4$ are reported in Table 4.2. It is worth noting that the point cloud data on the MSR action3D dataset is reconstructed from the

depth maps rather than directly sampling from the ground-truth mesh, thus the evaluation results in terms of P2F and the uniformity scores are not available. From Table 4.2, I observe that the sequential point cloud upsampling method still outperforms its single frame-based counterparts on the real-scanned dataset, which indicates that it is beneficial to exploit rich temporal information from the neighboring frames to enhance the upsampling results on the real-scanned dataset.

The visualization results on the MSR Action3D dataset are provided in Fig. 4.4. The single frame-based methods like PU-Net and PU-GAN often produce large amounts of outliers above the surface. As a result, I cannot observe the clear shape boundary of the human body. VPU (PU-Net) and VPU (PU-GAN) are able to preserve the shape boundary of the original input, and even generate some reasonable details to form more smooth boundaries (see the arms in the red boxes). PU-GCN generates better results with very few outliers, however, some geometric details are still missing in the non-uniform areas. By effectively exploiting spatio-temporal information, VPU (PU-GCN) is able to generate reasonably distributed points in the non-uniform areas, and recover smooth shape boundary. In terms of uniformity, thanks to the newly proposed STA module, VPU is able to more reliably gather temporal clues from adjacent frames to provide compensation to the non-uniform areas of the current frame. When compared with MPU and PU-GAN, VPU (MPU), VPU (PU-GAN) and VPU (PU-GCN) can more effectively avoid points clustering and generate more uniform upsampling results which contain fewer vacant areas (see the close-up views).

Testing speed and model size comparison

I report the testing speed and the model size of the single frame-based methods PU-Net, MPU, PU-GAN and PU-GCN, as well as the sequential methods VPU (PU-Net), VPU (MPU), VPU (PU-GAN) and VPU (PU-GCN) in Table 4.3. As shown in Table 4.3, the VPU framework is only about 10% slower when compared with its single frame-based counterparts, and it requires only 0.1M to 0.16M additional storage, which can be neglected when compared with the origin network size.

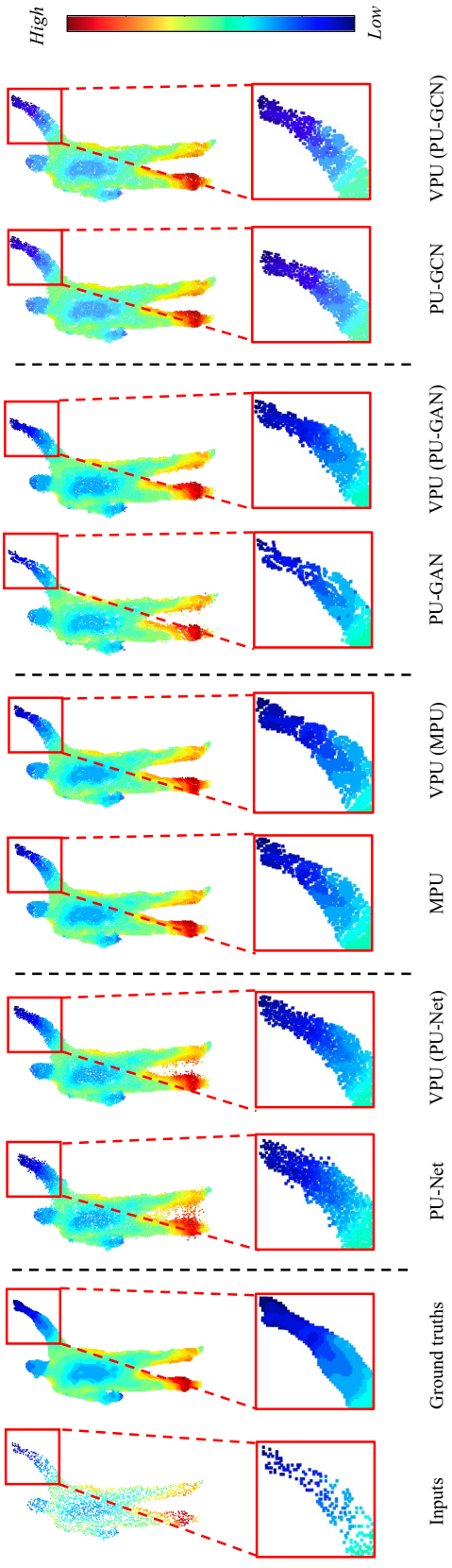


Figure 4.4: Comparison of the upsampling results on the MSR Action3D dataset when using different methods under the upsampling ratio $\times 4$. Different colors denote different depth values (*i.e.*, the red color denotes the highest depth value, while the blue color denotes the lowest depth value). Note the 3D points with the same color lie on the same surface with similar depth values. In addition, I also provide their close-up views around the right arm (*i.e.*, the red boxes) to compare the upsampling results in this local area. Best viewed on screen.

Table 4.2: Results of different methods on the MSR Action3D dataset.

Methods	CD ↓	HD ↓
PU-Net	0.826	18.624
VPU (PU-Net)	**0.787**	**16.201**
MPU	0.161	4.100
VPU (MPU)	**0.153**	**3.766**
PU-GAN	0.343	10.897
VPU (PU-GAN)	**0.317**	**6.129**
PU-GCN	0.220	5.655
VPU (PU-GCN)	**0.187**	**4.962**

Table 4.3: Comparison of different methods on the DYNA dataset in terms of testing speed and model size.

Methods	Testing speed(s)	Model size(M)
PU-Net	0.288	9.293
VPU (PU-Net)	0.297	9.431
MPU	0.289	1.658
VPU (MPU)	0.325	1.762
PU-GAN	0.391	7.830
VPU (PU-GAN)	0.434	7.934
PU-GCN	0.245	0.891
VPU (PU-GCN)	0.281	1.052

Table 4.4: Results of different methods on the DYNA dataset when using different upsampling ratios.

Methods	× 2		× 4		× 8		× 16	
	CD ↓	HD ↓	CD ↓	HD ↓	CD ↓	HD ↓	CD ↓	HD ↓
PU-Net	1.275	31.780	1.740	36.367	2.040	39.959	2.284	42.493
VPU (PU-Net)	**0.983**	**20.341**	**1.119**	**23.835**	**1.197**	**25.206**	**1.228**	**25.914**
MPU	0.634	6.809	0.511	9.341	0.467	11.022	0.475	12.495
VPU (MPU)	**0.398**	**3.861**	**0.211**	**4.245**	**0.147**	**4.702**	**0.116**	**5.007**
PU-GAN	0.854	21.199	1.145	31.670	1.503	37.267	1.786	40.447
VPU (PU-GAN)	**0.621**	**12.759**	**0.602**	**15.067**	**0.674**	**16.919**	**0.749**	**18.368**
PU-GCN	0.669	7.002	0.649	10.594	0.711	13.655	0.761	15.614
VPU (PU-GCN)	**0.485**	**4.586**	**0.322**	**5.905**	**0.267**	**7.176**	**0.236**	**8.027**

Experimental results when using different upsampling ratios

In Table 4.4, I also take the DYNA dataset as an example to provide more results of different methods when using different upsampling ratios (*i.e.*, $\times 2$, $\times 4$, $\times 8$, and $\times 16$) . Specifically, I train all the methods based on the upsampling ratio $\times 2$, and then in the testing stage, I iteratively perform the $\times 2$ upsampling process multiple times according to the predefined upsampling ratio. I observe that the proposed methods VPU (PU-Net), VPU (MPU), VPU (PU-GAN) and VPU (PU-GCN) consistently achieve better upsampling results than their single frame-based counterparts at different upsampling ratios.

4.3.3 Comparison with the potential baseline methods

Considering that a few works [35, 25, 79, 34] focus on various vision tasks (*i.e.*, classification, detection and scene-flow estimation) for dynamic point cloud sequences, several potential sequential point cloud upsampling baselines can be constructed by simply combining these dynamic point cloud processing methods with the single frame-based point cloud upsampling methods. Here I take the combination of two dynamic point cloud processing methods (*i.e.*, FlowNet3D [34] and MeteorNet [35]) and four single frame-based point cloud upsampling methods (*i.e.*, PU-Net [81], MPU [78], PU-GAN [26] and PU-GCN [52]) as eight examples to report more results, which are referred to as "**FlowNet3D + PU-Net**",

"**FlowNet3D + MPU**", "**FlowNet3D + PU-GAN**", "**FlowNet3D + PU-GCN**", "**MeteorNet + PU-Net**", "**MeteorNet + MPU**", "**MeteorNet + PU-GAN**" and "**MeteorNet + PU-GCN**". Specifically, in each baseline method, I first use the per-point feature extraction module to produce the per-point feature extracted from each input frame, and then I feed the point coordinates and the extracted per-point feature from each input frame into the Flow-embedding module [34] or the Meteor module [35] to produce the fused feature, which integrates spatio-temporal information across three consecutive frames. Based on the fused feature, the upsampling module of the single frame-based method can be readily used to generate the upsampling result of the current frame. The results of different methods are reported in Table 4.5. When compared with these baseline methods, VPU (PU-Net), VPU (MPU), VPU (PU-GAN) and VPU (PU-GCN) achieve better results than the corresponding baseline methods in terms of all the evaluation metrics. A possible explanation is that the Flow-embedding module and the Meteor module both exploit temporal clues by using the *K*NN method in the Euclidean space. However, they cannot collect sufficient and reliable geometric information when looking for a limited number of neighbors from the adjacent frames in the Euclidean space, especially when the point clouds from the input frames are very sparse and non-uniform. The results indicate that it is beneficial to exploit temporal information at the feature space for the sequential point cloud upsampling task.

4.3.4 Ablation study and analysis

To study the contributions of different components in VPU, I take PU-GAN as the single frame-based method in VPU and report the results of different variants on the DYNA dataset. Specifically, I show the contribution of each individual module in the proposed STA 4.2.2 module (*i.e.*, CFFC&WFFC and NLA 4.2.2 submodules) by removing either of them from VPU (PU-GAN). Then I also report the results when using different numbers of frames as the input in VPU (PU-GAN) and evaluate the performance of VPU (PU-GAN) when using different k values. Additionally, I provide more results by analyzing the offset mechanism and

Table 4.5: Results of the proposed methods and different potential baseline methods on the DYNA dataset.

Methods	CD ↓	HD ↓	P2F ↓	Uniformity for different p		
				0.4% ↓	0.8% ↓	1.2% ↓
FlowNet3D + PU-Net	0.591	17.420	9.968	12.308	19.127	23.928
MeteorNet + PU-Net	0.570	15.521	9.194	9.730	17.295	22.617
VPU (PU-Net)	**0.377**	**11.391**	**6.439**	**6.392**	**11.749**	**15.499**
FlowNet3D + MPU	0.178	2.695	1.491	3.011	5.940	7.963
Meteor + MPU	0.188	2.712	1.565	3.263	6.057	7.819
VPU (MPU)	**0.173**	**2.620**	**1.432**	**2.931**	**5.788**	**7.308**
FlowNet3D + PU-GAN	1.362	36.724	17.362	23.522	29.329	32.193
Meteor + PU-GAN	1.180	19.361	10.741	19.464	22.712	26.046
VPU (PU-GAN)	**0.168**	**2.731**	**1.684**	**2.759**	**5.522**	**7.469**
FlowNet3D + PU-GCN	0.197	2.580	1.302	3.364	6.260	9.257
Meteor + PU-GCN	0.198	2.584	1.319	3.492	6.394	9.366
VPU (PU-GCN)	**0.158**	**2.315**	**1.114**	**2.779**	**5.644**	**7.538**

the searching range in the WFFC/CFFC submodule.

Different modules in VPU

The results of VPU (PU-GAN) when using different modules are reported in Table 4.6, where "**w/o NLA**" denotes VPU (PU-GAN) after removing the NLA submodule from the STA module, and **w/o WFFC**", "**w/o CFFC**" and "**w/o CFFC & WFFC**" denote VPU (PU-GAN) after removing either the WFFC or CFFC submodule or both of them from the STA module, respectively. Note that in the first alternative method"**w/o NLA**", I simply set the value of k in both CFFC and WFFC submodules as $2K$ for the subsequent feature concatenation operation in order to avoid the dimension mismatch issue. In the second alternative method "**w/o CFFC & WFFC**", I use the initial features F_{t-1}, F_t and F_{t+1} to replace both the within-frame combined feature $\hat{F}_t$ and the cross-frame combined features $\hat{F}_{t-1}$ and $\hat{F}_{t+1}$. In the third alternative method "**w/o WFFC**", I replace the within-frame combined features $\hat{F}_t$ with the initial feature F_t, while I still use $\hat{F}_{t-1}$ and $\hat{F}_{t+1}$ as the cross-frame combined features. In the fourth alternative method "**w/o CFFC**", I use the initial features F_{t-1} and F_{t+1} to replace the cross-frame combined features $\hat{F}_{t-1}$ and $\hat{F}_{t+1}$, while I still use $\hat{F}_t$ as the within-frame combined feature. In addition, one may

be interested in the upsampling results of the single frame-based methods when directly using the point clouds from three consecutive frames as the input. Thus, I also conduct another experiment for comparison (denoted as **PU-GAN (3 frames)**), in which I simply merge three consecutive point cloud frames, and use the FPS method to sample 1/3 of the combined points as the input of PU-GAN. The network is then trained based on the same loss functions as PU-GAN, and the results of **PU-GAN (3 frames)** are 0.410, 8.731 and 5.880 in terms of CD, HD and P2F distance. Since it cannot effectively exploit temporal information from neighboring frames, it achieves the worst upsampling results in terms of all metrics. When compared with "**PU-GAN (3 frames)**", the first alternative method "**w/o NLA**" achieves better results in terms of the overall matching errors (*e.g.*, CD, HD and P2F distance) by simply concatenating the cross-frame combined features and the within-frame combined feature along the feature dimension without aligning these features, which indicates that it is also beneficial to adopt the simple aggregation strategy for exploiting temporal dependency across the adjacent frames. The second alternative method "**w/o CFFC & WFFC**" also achieves performance improvement in terms of all the evaluation metrics when compared with "**PU-GAN (3 frames)**". The results demonstrate that it is also effective to perform alignment at the initial feature space. When compared with VPU (PU-GAN), both variants "**w/o WFFC**" and "**w/o CFFC**" achieve worse performance due to the lack of well-extracted features from either the current frame or the neighboring frames. Based on the results of "**w/o WFFC**" or "**w/o CFFC**" versus "**w/o CFFC & WFFC**", I observe that the proposed CFFC and WFFC submodules can reliably and effectively encode desirable geometry information from both the current frame and the neighboring frames thanks to the carefully designed feature extraction strategy. After using WFFC, CFFC and NLA submodules, VPU (PU-GAN) achieves the best results in terms of all the evaluation metrics, which indicates that it is beneficial to use all the proposed submodules in the STA module to enhance the point cloud upsampling results.

Table 4.6: Results of VPU (PU-GAN) and its four variants on the DYNA dataset.

Methods	CD ↓	HD ↓	P2F ↓	Uniformity for different p		
				0.4% ↓	0.8% ↓	1.2% ↓
w/o NLA	0.189	3.095	1.870	3.066	6.122	7.981
w/o CFFC & WFFC	0.220	3.714	2.099	4.151	8.422	11.137
w/o WFFC	0.188	3.334	1.846	3.393	6.941	9.089
w/o CFFC	0.175	3.668	1.788	2.934	6.019	7.788
VPU (PU-GAN)	0.168	2.731	1.684	2.759	5.522	7.469

Table 4.7: Results of VPU (PU-GAN) on the DYNA dataset when using different numbers of input frames.

# Input frames	CD ↓	HD ↓	P2F ↓	Uniformity for different p		
				0.4% ↓	0.8% ↓	1.2% ↓
3 frames	0.168	2.731	1.684	2.759	5.522	7.469
5 frames	0.167	2.704	1.676	2.686	5.493	7.396
7 frames	0.173	2.929	1.796	2.909	5.853	7.719

Different number of input frames

I evaluate the performance of VPU (PU-GAN) when using different numbers of input frames, including 3 frames, 5 frames and 7 frames, respectively. As shown in Table 4.7, the results of VPU (PU-GAN) are slightly improved by using 5 frames, but the results become worse after using 7 frames. It is worth mentioning that the inference time also increases by 10% and 19% by using 5 frames and 7 frames as the input, respectively. Thus, in this work, I choose 3 frames as the default setting due to the trade-off between effectiveness and efficiency.

Different values of k

To test the sensitivity of the proposed method with respect to k, I also take VPU (PU-GAN) as an example to conduct the experiments based on different k values (*i.e.*, 2, 4, 8, 16 and 32). Note that when $k = 32$, the within-frame/cross-frame neighborhood is large enough (*i.e.*, 25% / 50% of the current/reference frame), since the number of the input points is 128. In addition, recalling the dimension of the similarity matrix in the

Table 4.8: Results of VPU(PU-GAN) on the DYNA dataset when using different values of k.

# The values of k	CD ↓	HD ↓	P2F ↓	Uniformity		
				0.4% ↓	0.8% ↓	1.2% ↓
$k = 2$	0.178	2.831	1.784	3.320	6.359	8.362
$k = 4$	0.172	2.779	1.713	2.965	5.838	7.869
$k = 8$	0.168	2.731	1.684	2.759	5.522	7.496
$k = 16$	0.133	2.305	1.190	2.105	5.153	6.912
$k = 32$	0.125	2.100	1.075	2.033	4.901	6.780

NLA submodule is $Nk \times 2Nk$, I require much more memory (*i.e.*, multiple 1080Ti GPUs) to conduct the experiments when k is larger than 32 . The results are reported in Table 4.8. I observe that the results become better as the value of k increases. However, when the value of k increases, the computational cost and the consumed memory also increase. Considering the trade-off between effectiveness and efficiency, I choose $k = 8$ as the default setting in this work.

More Analysis

I also design two experiments to perform more detailed analysis for the newly proposed STA module, and the results are reported in Table 4.9.

I first conduct an experiment to evaluate the contribution of the offset features $\mathbf{f}_{t-1}^{j} - \mathbf{f}_{t}^{i}$ in Eq.(1) by replacing the offset features with the initial features $\mathbf{f}_{t-1}^{j}$ of the searched neighboring points in the CFFC/WFFC submodule. The results of "**w/o offset**" are much lower when compared with those of VPU (PU-GAN). A possible explanation is that these initial features are highly related to the 3D coordinates of neighboring points, thus undesirable temporal clues will be introduced after using the initial features. The results also indicate that it is beneficial to exploit spatio-temporal local structure information at the offset feature space.

I then evaluate the performance of VPU (PU-GAN) by reducing the cross-frame searching range when using the kNN method in the CFFC submodule (*i.e.*, from $2K$ to K). When compared with the results of VPU (PU-GAN) (see Table 4.9), the results of "**w/o** $2K$" become slightly worse

Table 4.9: Results of VPU (PU-GAN) and its two variants on the DYNA dataset.

Methods	CD ↓	HD ↓	P2F ↓	Uniformity for different p		
				0.4% ↓	0.8% ↓	1.2% ↓
w/o offset	0.236	4.772	2.253	4.088	8.413	11.014
w/o $2K$	0.172	2.935	1.706	2.912	5.757	7.631
VPU (PU-GAN)	0.168	2.731	1.684	2.759	5.522	7.469

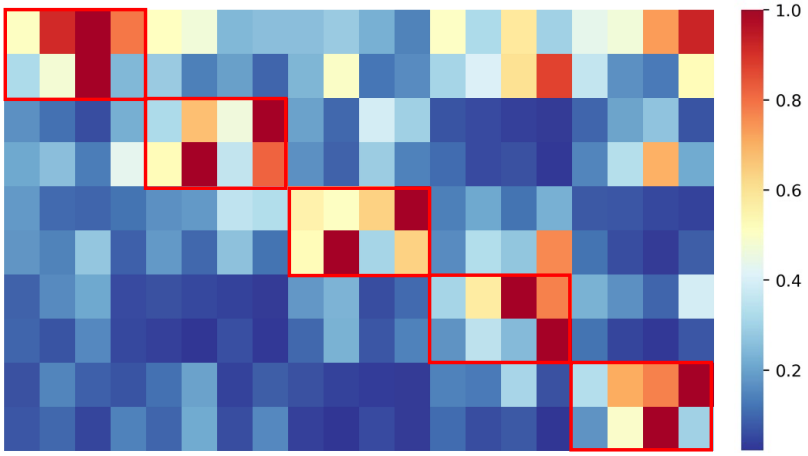


Figure 4.5: Visualization of the similarity matrix between two patches after using the NLA submodule for a toy problem.

in terms of all metrics, which demonstrates it is beneficial to use a larger searching range for searching neighboring points across frames to capture necessary shape dynamics.

4.3.5 Robustness Analysis

Following the experiment in PU-Net [81], I take PU-Net and the proposed VPU (PU-Net) as an example to perform a robustness test by upsampling the point sets with different Gaussian noise levels, and the results are reported in Table 4.10. Specifically, I first normalize the point clouds from the testing frame into a unit sphere, and then perturb their coordinates by adding Gaussian noise with different standard deviations (*i.e.*, 0.0%, 0.5% and 1.0%) based on the bounding sphere's radius. When compared with the single frame-based method PU-Net, the proposed VPU (PU-Net) is more robust at each Gaussian noise level.

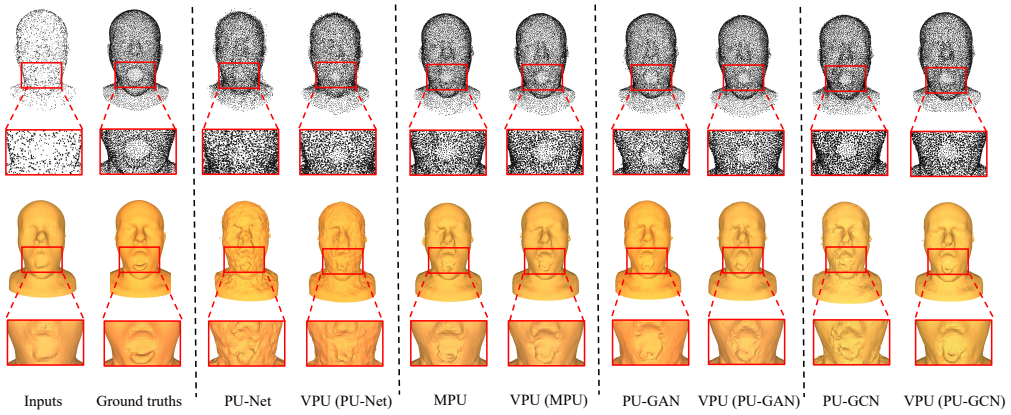


Figure 4.6: Comparison of the upsampling results (see the first row) and the surface reconstruction results (see the third row) on the COMA dataset when using different methods under the upsampling ratio $\times 4$. I additionally provide their close-up views to compare the upsampling results (see the second row) and surface reconstruction results (see the fourth row) around the mouth area (*i.e.,* the red boxes).

Table 4.10: Results of different methods on the DYNA dataset when using the input point clouds with different levels of noise.

Methods	0.0%		0.5%		1.0%	
	CD ↓	HD ↓	CD ↓	HD ↓	CD ↓	HD ↓
PU-Net	0.637	19.314	0.740	21.193	0.896	22.686
VPU (PU-Net)	**0.377**	**11.391**	**0.380**	**11.579**	**0.473**	**11.891**

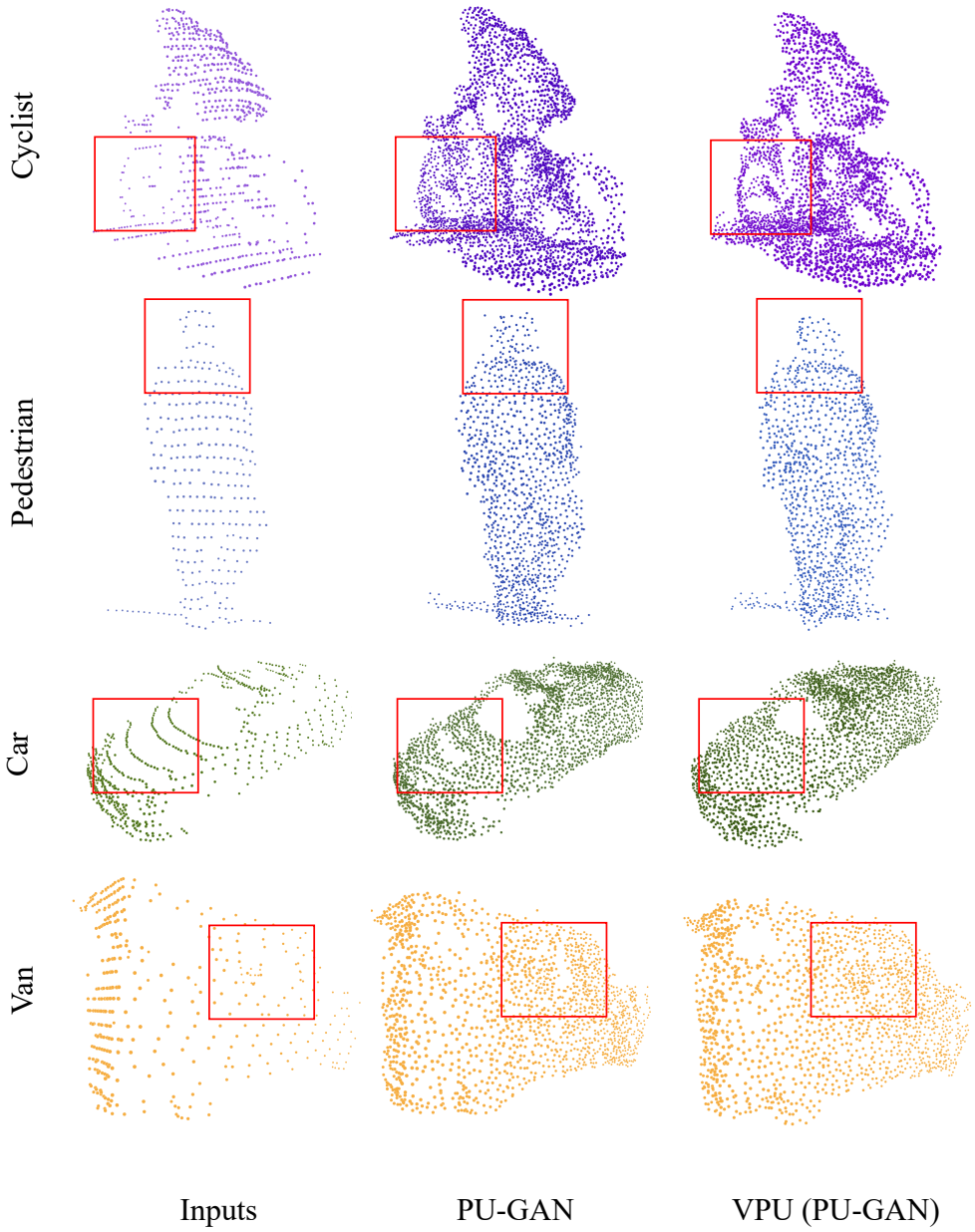


Figure 4.7: Visualization results by using PU-GAN [26] and VPU (PU-GAN) on four different types of objects from the KITTI dataset.

4.3.6 Visualization of the temporal alignment strategy in the Non-Local Alignment Submodule

For better understanding the temporal alignment results by using the newly proposed NLA submodule, I use a simple toy example to visualize the similarity matrix $\mathbf{W}_{t \leftarrow t-1}$ in Fig. 4.5. Specifically, I first randomly select two consecutive frames from the DYNA dataset as the input, then I follow the patch extraction method described in the implementation details in Sec. 4.3.1 to crop a small patch with only five 3D points from each frame to construct a toy problem, and I empirically set the neighborhood size $k = 2$ (*resp.* $k = 4$) for the current frame (*resp.* the previous frame) when calculating the similarity matrix between these two small patches at the feature space.

Each red box in Fig. 4.5 denotes the similarity values between two within-frame individual features related to one target point from the current frame (note each feature is extracted based on this target point and one of its two neighboring points in the current frame) and four cross-frame individual features related to this target point (note each feature is extracted based on this target point and one of the four neighboring points in the reference frame). As shown in Fig. 4.5, the similarity matrix generally exhibits block-diagonal characteristics, namely for most of the within-frame individual features related to each target point in the current frame, the NLA submodule can correctly align them to the corresponding cross-frame individual features related to the same target point. The result demonstrates the effectiveness of the NLA submodule.

4.3.7 Experiments results of generalization capability

As the VPU framework follows the patch-based training and testing strategy in [78, 26], I further evaluate different methods on the COMA [54] and KITTI [15] datasets by using their models pre-trained on the DYNA dataset.

Table 4.11: Results of different methods on the COMA dataset.

Methods	CD ↓	HD ↓	P2F ↓	Uniformity for different p		
				0.4% ↓	0.8% ↓	1.2% ↓
PU-Net	2.162	45.329	3.395	2.103	3.578	4.992
VPU (PU-Net)	**1.062**	**20.724**	**1.847**	**1.210**	**2.030**	**2.872**
MPU	0.948	6.956	0.480	0.532	0.902	1.363
VPU (MPU)	**0.643**	**5.339**	**0.421**	**0.489**	**0.847**	**1.294**
PU-GAN	1.358	17.153	0.661	0.790	1.305	1.836
VPU (PU-GAN)	**1.277**	**7.948**	**0.609**	**0.373**	**0.623**	**0.974**
PU-GCN	0.863	4.829	0.461	0.491	0.857	1.387
VPU (PU-GCN)	**0.839**	**3.733**	**0.455**	**0.465**	**0.820**	**1.252**

Experimental results on the COMA Dataset

The results on the COMA dataset are summarized in Table 4.11. The newly proposed VPU methods, including VPU (PU-Net), VPU (MPU), VPU (PU-GAN) and VPU (PU-GCN) still consistently outperform the single frame-based counterparts PU-Net, MPU, PU-GAN and PU-GCN in terms of all the evaluation metrics. Since this dataset only contains the human head with different facial expressions, it is easier for the proposed method VPU, as well as the corresponding baseline methods, to capture the surface structures of the relatively stable faces rather than dynamically moving human bodies as from the DYNA dataset. Therefore, the P2F and the uniformity scores from all methods are much lower than those on the DYNA dataset in Table 4.1.

I also provide a visualization example on the COMA dataset in Fig. 4.6, from which I observe that PU-Net fails to restore the human face shape due to the deformed human face with extreme expressions. In contrast, VPU (PU-Net) can better recover the human face shape by exploiting temporal information from adjacent frames, and generate smoother surface when compared with PU-Net. MPU, PU-GAN and PU-GCN are able to capture essential facial features with specific semantics (*e.g.*, eyes, nose and mouth), however, the generated boundaries of the lips are often not clear enough (see the red boxes). In contrast, VPU (MPU), VPU (PU-GAN) and VPU (PU-GCN) can better reconstruct more details for

each local area by effectively aggregating the local structural information from adjacent frames, and thus recover more clear boundaries for the lips.

Visualization results on the KITTI dataset

I take VPU (PU-GAN) as an example to evaluate the robustness, and generalizability capability of different methods on non-uniform, sparse and occluded real-scanned data. Fig. 4.7 illustrates the upsampling results by using PU-GAN and the proposed VPU (PU-GAN) for four types of representative objects, "cyclist", "pedestrian", "car", and "van" on the KITTI dataset. The proposed method VPU (PU-GAN) generates denser and more uniform surfaces with richer geometric details, which indicates the generalization capability of the VPU framework on the KITTI dataset. As suggested in the work PU-GAN [26], I cannot access the ground-truth dense point clouds on the KITTI dataset, so I cannot provide their qualitative results on this dataset.

4.4 Summary

In this chapter, I have proposed a new patch-based framework VPU for the sequential point cloud upsampling task by effectively exploiting temporal dependency among multiple clues extracted from the neighboring low-resolution point clouds. To cope with the inherent alignment difficulty between unordered point cloud frames, I propose a spatio-temporal aggregation module to effectively extract, align and aggregate the spatio-temporal hints from multiple consecutive frames at the feature level to generate the aggregated feature for the current frame. The aggregated feature can be readily combined with various existing single frame-based point cloud upsampling methods to guide the point cloud upsampling process for the current frame. The sequential framework achieves the state-of-the-art performance and is able to reliably upsample point clouds from multiple point cloud sequence datasets.

Although the VPU framework has achieved significant progress by integrating the patch-based strategy in the sequential point cloud upsampling paradigm, I still observe some unsatisfactory upsampling results under some challenging cases, especially when there are considerable movements between adjacent frames and over-sparse local regions in the target point cloud frame. I suppose these phenomena result from severe patch misalignment and the traditional upsampling paradigm via direct point expansion, respectively. Therefore, how to further alleviate the patch misalignment issue and design a more reasonable paradigm for missing geometry recovery remain open problems.

Chapter 5

Sequential Point Cloud Upsampling via Progressive Multi-Level Refinement

In the last chapter, I have presented a patch-based framework VPU for better efficiency and generalizability under real-world scenarios, which attempts to address the intrinsic patch mismatch issue by performing temporal alignment at the feature space. However, I observe that VPU tends to produce undesirable artifacts in the dynamic regions, especially when there are significant movements between the adjacent frames. Therefore, in this chapter, I propose a new paradigm for sequential point cloud upsampling via progressive multi-level refinement (PMRNet), which achieves faithful disentanglement between motion and geometry at the frame level and fine-grained upsampling results at the shape level and detail level. I have conducted comprehensive experiments to validate the effectiveness of the novel designs in PMRNet.

5.1 Motivations and Contributions

Although the proposed VPU framework has achieved substantial success by integrating the widely used patch-based strategy into the sequential point cloud framework, I still observe two challenging cases in which VPU cannot work well: (1) Its spatio-temporal aggregation (STA) module cannot work well when processing the point cloud sequences with significant movements between the neighboring frames as

shown in Fig. 5.1(a). A possible explanation is that the networks in the STA module cannot identify whether the collected spatio-temporal contexts are effective, especially when the patches from adjacent frames are severely misaligned. In this case, the STA module will inevitably introduce undesirable information due to the strong entanglement between motion and geometry, which leads to artifacts splatted in the regions with dynamic movements. (2) VPU tends to generate smooth results in the over-sparse regions instead of recovering the original geometry details as illustrated in Fig. 5.1(c). I suppose this phenomenon is derived from the current point cloud upsampling paradigm, which consists of two point-wise modules (i.e., the feature extraction module and the feature expansion module). This paradigm typically generates new points by duplicating the input points and expands them to a uniform distribution, in which the points generated in the original over-sparse regions come from simple expansion instead of direct generation with sufficient guidance.

In this chapter, I propose a new sequential point cloud upsampling framework called Progressive Multi-level Refinement Network (PMR-Net) to cope with the aforementioned problems in two steps. For the first problem, a commonly used solution is to leverage additional constraints on the global object shapes similar as in SPU. However, these constraints usually come from the whole point cloud frames and cause unaffordable memory consumption during the upsampling process, which is not an appropriate strategy for real-world applications. In contrast, I take inspiration from the recent non-rigid point cloud registration to design a robust frame-level refinement module by first warping the reference frames to the same 3D object surface of the target frame, which achieves robust disentanglement between motion and geometry. As shown in Fig. 5.1(b), the patch mismatch phenomenon can be significantly alleviated after the warping-based refinement, which enables a far more reliable spatio-temporal information collecting system than the STA module in VPU. It is worth noting that the robust frame-level refinement module is only performed on the low-resolution point clouds before the upsampling process and thus will not cause considerable memory cost.

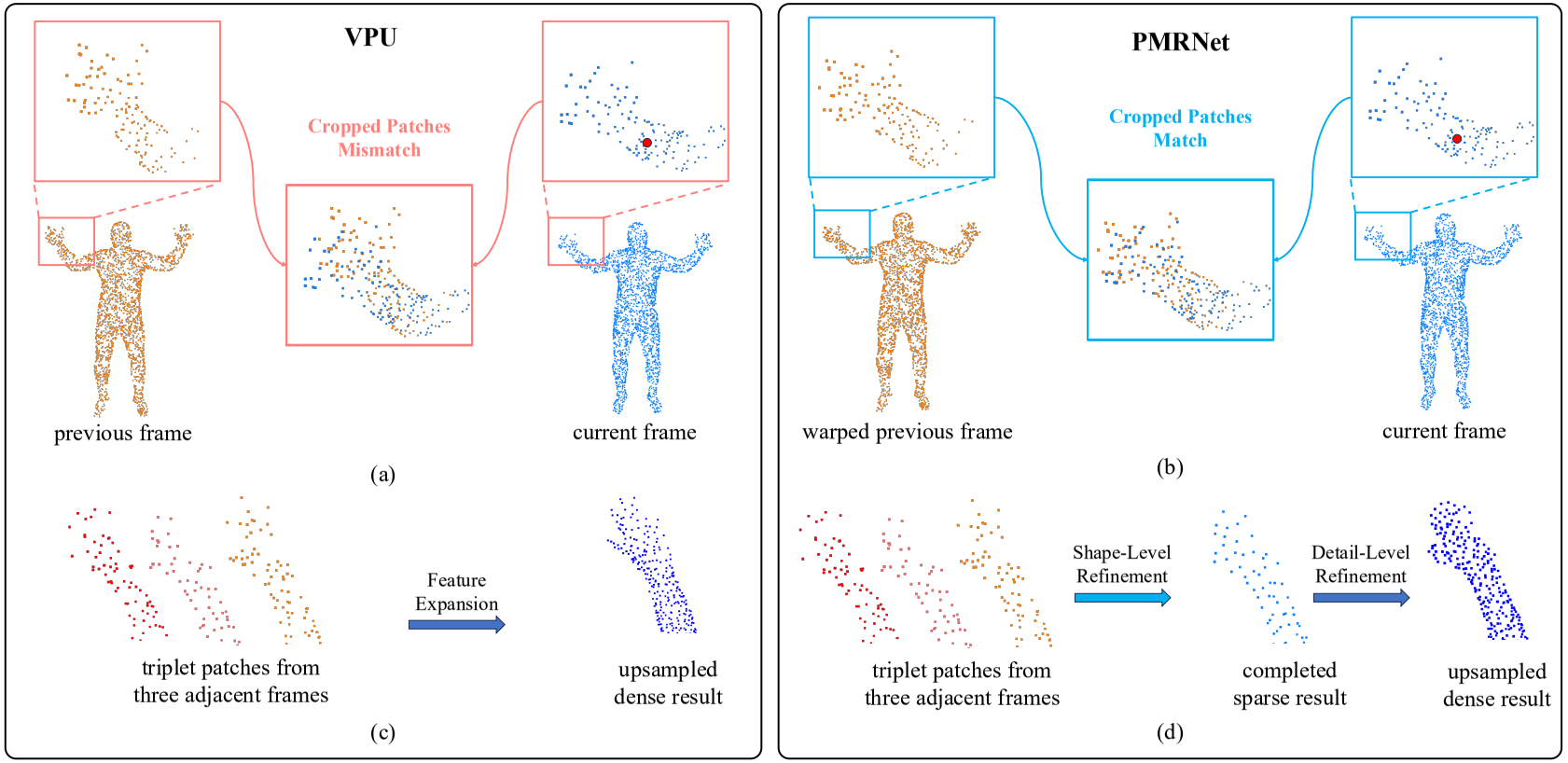


Figure 5.1: (a) The temporal patch-cropping strategy proposed by VPU suffers from severe shape mismatch issues, especially when there are substantial movements between neighboring frames. (b) The PMRnet proposes to perform frame-level refinement before the patch-cropping operation by warping the reference frame to the 3D object surface of the target frame, which significantly alleviates the patch misalignment issues. (c) VPU follows the single frame-based methods to produce upsampling results via direct feature expansion leading to unsatisfactory recovery in the over-sparse regions. (d) The PMRNet adopts a new coarse-to-fine paradigm by performing refinement on both shape level and detail level, which achieves a dense result with a complete boundary and delicate details. Close-up views of the cropped patches from consecutive frames are provided for better visualization.

To address the second problem, I propose a new coarse-to-fine paradigm, including a shape-level refinement module and a detail-level refinement module. Specifically, I propose a shape-level refinement module to first produce a coarse result, based on which I can further perform the upsampling operation. As shown in Fig. 5.1(d), although the completed coarse result is still composed of sparse point clouds, it obtains a complete geometry and uniform distribution, which thus improves the quality of the upsampling results from the feature expansion module. Moreover, to achieve the upsampled point cloud with better details, I also propose a detail-level refinement module to further decorate the generated points in local parts by referring to the geometric clues from the neighboring frames.

Equipped with the robust frame-level refinement module, shape-level refinement module, and detail-level refinement module, PMRNet can progressively refine the upsampling process in multiple levels and effectively generate upsampled point cloud sequences with more semantic meaning and details. The main contribution of this work can be summarized as follows: (1) I propose a new point cloud sequence upsampling framework called Progressive Multi-level Refinement Network (PMRNet), which refines the upsampling process at multiple levels in a progressive fashion. (2) I propose a robust frame-level refinement module to achieve the disentanglement between geometry and motion by warping different frames to one temporal space, which effectively alleviates severe patch mismatch issues among different frames. (3) I propose a shape-level refinement module to refine the incomplete and non-uniform sparse point cloud before upsampling, which can facilitate the subsequent detail generation. (4) I propose a detail-level refinement module to further refine the generated details for better point cloud sequence upsampling performance. (5) Comprehensive experiments on multiple datasets demonstrate the effectiveness of PMRNet for sequential point clou upsampling.

5.1.1 Discussion

The studies most related to PMRNet are the sequential point cloud upsampling approaches SPU and VPU proposed in the last two chapters and all the single frame-based point cloud upsampling methods. SPU and VPU employ the frame-based strategy and the patch-based strategy to achieve temporally coherent recovery and good generalizability, respectively. However, both these two frameworks cannot achieve satisfying performance on the point cloud sequences with large movements, as the geometry and motion are strongly coupled. Unlike these two approaches, PMRNet combines the advantages of the two strategies (i.e., the frame-based strategy and the patch-based strategy) to propose a two-staged sequential point cloud upsampling framework that separately deals with the motion and the geometry information. In terms of the single frame-based methods, they normally follow the same paradigm, including only feature extraction and feature expansion, which leads to unsatisfactory results in the over-sparse regions. The proposed PMRNet differently proposes a new coarse-to-fine paradigm that can better preserve the original geometry details by performing shape-level completion before upsampling.

5.2 Methodology

5.2.1 Overview

The overview of the Progressive Multi-level Refinement Network (PMRNet) for sequential point cloud upsampling is shown in Fig. 5.2(a). The proposed PMRNet consists of three modules, the robust frame-level refinement module, the shape-level refinement module, and a detail-level refinement module. Specifically, when upsampling the current frame $\mathcal{P}_t$, I first locate its adjacent frames and use the robust frame-level refinement module to eliminate the coordinate misalignment and produce refined frames. Then, I crop multiple local patches from the refined frames and use the shape-level refinement module to refine the incomplete local

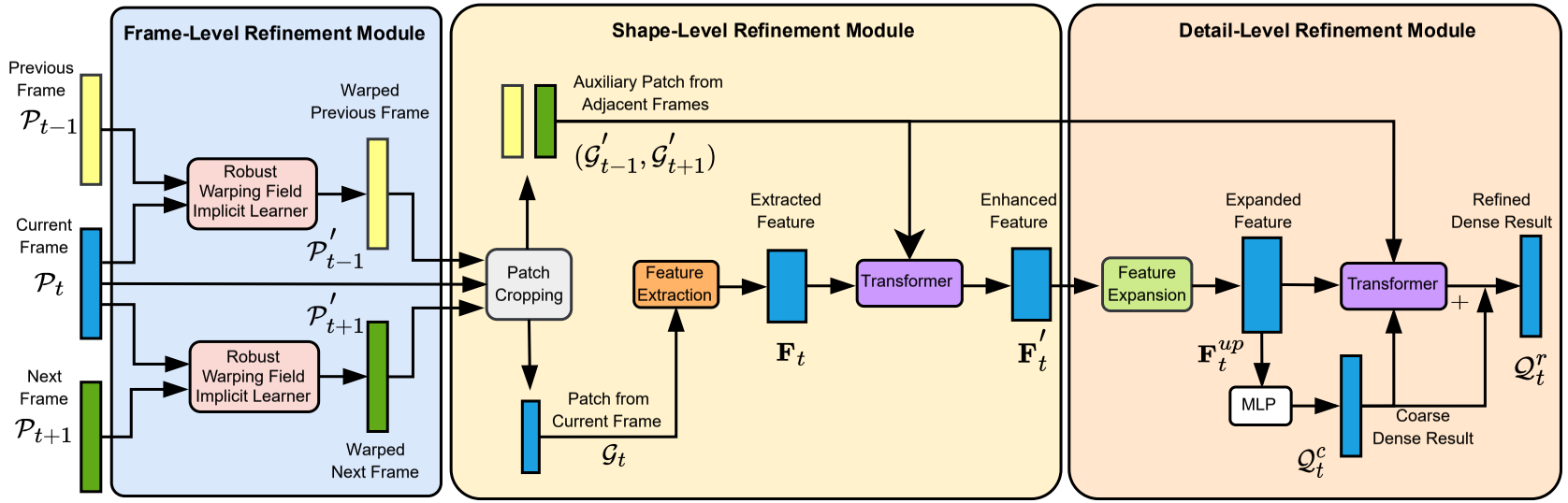


Figure 5.2: Overview of the proposed PMRNet. Given the sparse and incomplete point clouds from three consecutive frames, I first perform Robust Frame-Level Refinement to roughly align the point clouds from reference frames $\mathcal{P}_{t-1}, \mathcal{P}_{t+1}$ with the point cloud from the target frame $\mathcal{P}_t$ by using the *Robust Warping Field Implicit Learner*. Then I follow VPU to adopt its temporal patch-cropping strategy to construct triplet local patches from $\mathcal{P}_t$ and the point clouds from two warped reference frames $\mathcal{P}'_{t-1}$ and $\mathcal{P}'_{t+1}$. In the next stage, I first produce a point-wise feature $\mathbf{F}_t$ from the input patch $\mathcal{G}_t$ by using the feature extraction module. Then I take advantage of the spatio-temporal auxiliary input patches $\mathcal{G}'_{t-1}, \mathcal{G}'_{t+1}$ and $\mathbf{F}_t$ to produce an enhanced feature $\mathbf{F}'_t$ by using the proposed Shape-Level Refinement module. Lastly, I feed the expanded feature $\mathbf{F}_t^{up}$ as well as auxiliary input patches $\mathcal{G}'_{t-1}, \mathcal{G}'_{t+1}$ into the Detail-Level Refinement Module to first obtain a coarse upsampling result $\mathcal{Q}_t^c$ and produce the final refined dense point cloud $\mathcal{Q}_t^r$ via delicate refinement on geometric details.

patches into a complete one, based on which I produce a coarse upsampling result via feature expansion. Finally, I use the detail-level refinement module to decorate the coarse upsampling result with delicate details and generate the refined dense result. I will introduce them below.

5.2.2 Robust Frame-level Refinement Module

In the last chapter, I propose a patch-based sequential point cloud upsampling framework called VPU, which relies on a temporal patch-cropping strategy. To obtain temporally-paired patches, it performs the cropping operation on three consecutive frames centered on the seeds only selected from the target frame. However, this strategy obviously causes a patch-mismatch issue where patches cropped from adjacent frames cannot share the same region of the 3D object due to the dynamic movements between frames. VPU has made explorations to address this issue at the patch level by integrating commonly-used dynamic point cloud processing modules [34, 35] into its framework, while the experimental results demonstrate that the modules mainly designed for scene flow estimation (*i.e.,* rigid transformation), and cannot handle more complex scenarios with non-rigid deformation.

I rethink this issue and propose to model the motion of the 3D object at the frame level instead, which can not only achieve more accurate motion capture but also alleviate the patch-mismatch issue.

Warping Field Implicit Learner. As there are two robust warping field implicit learners in the robust frame-level refinement module, I use one of them as an example to introduce them. Given the point clouds from the previous frame $\mathcal{P}_{t-1} \in \mathbb{R}^{N\times 3}$ and the point clouds from the current frame $\mathcal{P}_t \in \mathbb{R}^{N\times 3}$, I aim to predict point-wise displacement for each point x from the previous point cloud frame $\mathcal{P}_{t-1}$ to align this frame to the current frame. To achieve so, I first produce shape descriptors $\mathbf{S}_{t-1}, \mathbf{S}_t \in \mathbb{R}^d$ to represent the crucial geometric information for both

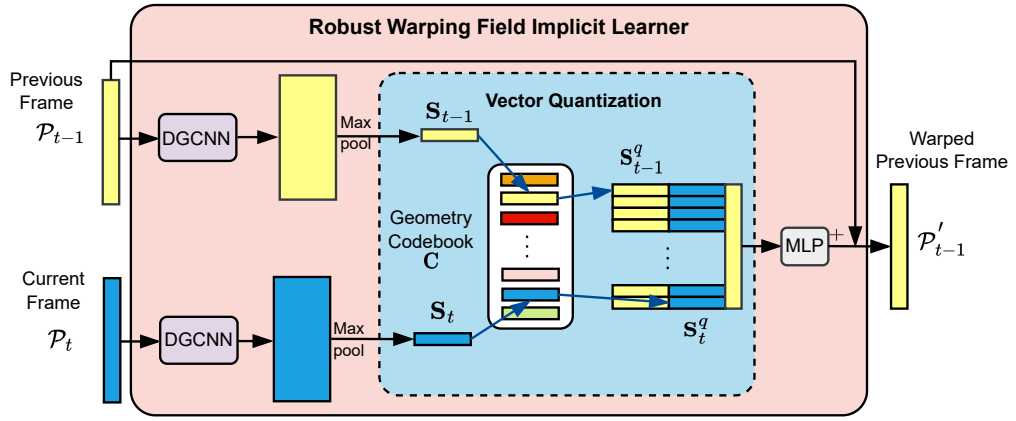


Figure 5.3: Illustration of the robust warping field implicit learner.

points sets, which are defined as:

$$\begin{aligned} \mathbf{S}_{t-1} &= \underset{i=1,\ldots,N}{Max}(E_n(p^i_{t-1})), \forall p^i_{t-1} \in \mathcal{P}_{t-1}, \\ \mathbf{S}_t &= \underset{i=1,\ldots,N}{Max}(E_n(p^i_t)), \forall p^i_t \in \mathcal{P}_t. \end{aligned} \tag{5.1}$$

Here, Max denotes the max-pooling operation, and E_n denotes the feature concentration to extract semantic information from each frame. In the practical implementation, I use the backbone network DGCNN [68] as the feature concentration. Then I concatenate these two descriptors with the coordinate of each point x from $\mathcal{P}_{t-1}$ to construct an input for an implicit function specially designed for warping field modeling, in which the displacement for each point x is defined as:

$$x_d = f_{MLP}([\mathbf{S}_t; \mathbf{S}_{t-1}; x]), \tag{5.2}$$

where f_{MLP} is a three-layer MLP network with an activation function employed for the first two layers. The entire structure of this refinement process has been proven to be continuous [66], which can guarantee the coherent geometric transformation for the whole point set. By adding the per-point displacement to its original point cloud coordinates, I obtain the warped point cloud for the previous frame $\mathcal{P}'_{t-1}$, which should share similar geometry as the current frame and the patch mismatch issues can be largely eliminated.

Vector Quantization for Robust Warping. Although the warping field implicit learner can effectively refine different frames and eliminate coordinate displacement, I observe its refinement performance is affected by the sparse and incomplete input point cloud, which leads to inaccurate displacement prediction due to their degraded shape descriptors.

Recently, several studies [86, 17] on 2D image restoration adopt the vector-quantization mechanism for robust recovery performance, which relies on an intermediate learnable codebook to store high-quality textures. Inspired by this, I propose a more robust warping field learning process by introducing the vector quantization mechanism and a geometry codebook in which the shape descriptors of point clouds with complete geometry information can be preserved. The architecture of the robust warping field implicit learner is illustrated in Fig. 5.3. Here I still take the shape descriptors $\mathbf{S}_{t-1}$ for the previous frame as an example. Instead of directly using $\mathbf{S}_{t-1}$ defined in Eq. 5.1, I replace it with the closest code in the geometry codebook $\mathbf{C} = \{c_i \in \mathbb{R}^d\}_{i=0}^{N_c-1}$ via minimizing feature distance:

$$\begin{aligned} m &= \arg\min_i \ \|\mathbf{S}_{t-1} - c_i\|_2 \text{ and } \mathbf{S}_{t-1}^q = c_m, \\ n &= \arg\min_i \ \|\mathbf{S}_t - c_i\|_2 \text{ and } \mathbf{S}_t^q = c_n. \end{aligned} \tag{5.3}$$

where N_c denotes the codebook size, $\mathbf{S}_{t-1}^q$ and $\mathbf{S}_t^q$ are the robust shape descriptors from the geometry codebook for the frame $\mathcal{P}_{t-1}$ and $\mathcal{P}_t$, respectively. By using the most similar shape descriptor from the geometry codebook instead of those from the distorted point cloud data, I are able to achieve robust warping field modeling better arbitrary adjacent frames.

5.2.3 Shape-level Refinement

Based on the pre-trained warping field implicit learner, I are able to perform frame-level refinement by transforming the point clouds from the adjacent frames $\mathcal{P}_{t-1}, \mathcal{P}_{t+1}$ into $\mathcal{P}_{t-1}^{'}, \mathcal{P}_{t+1}^{'}$, which are aligned with the

point clouds from the current frame $\mathcal{P}_t$ at the coordinate space. Based on the new triplet (*i.e.*, $\mathcal{P}^{'}_{t-1}, \mathcal{P}_t, \mathcal{P}^{'}_{t+1}$) and I follow the patch cropping strategy [64] to construct a triplet local patches $\mathcal{G}^{'}_{t-1}, \mathcal{G}_t, \mathcal{G}^{'}_{t+1} \in \mathbb{R}^{M\times 3}$ for subsequent patch-based upsampling as shown in Fig. 5.2.

I first use the feature extractor in [78, 26] to extract the feature $\mathbf{F}_t \in \mathbb{R}^{M\times C}$ from $\mathcal{P}_t$. Previous point cloud upsampling methods recover upsampled point clouds directly from $\mathbf{F}_t$ by using feature expansion [64]. However, these modules tend to generate medicore results without sufficient details in the vacant regions, where no prior geometry information is provided in the input patch.

In contrast, I propose a shape-level refinement module to complete the input geometry before feature expansion for better detail generation. Specifically, I take advantage of the spatial-temporal hints from the current frame and the adjacent frames to enhance the per-point feature $\mathbf{F}_t$, which is formulated as:

$$\mathbf{F}^{'}_t = Eh(\mathbf{F}_t, Pos(\mathcal{G}_t, \mathcal{G}^{'}_{t-1}, \mathcal{G}^{'}_{t+1}). \tag{5.4}$$

Here $Eh(\cdot)$ is a transformer-based network [84, 37] for enhancement, and $Pos(\cdot)$ is a position encoding block that integrates within-frame and cross-frame geometric information from the input consecutive frames.

Specifically, for each point p^i_t from $\mathcal{G}_t$, I first perform K Nearest Neighbour (KNN) grouping on the current frame $\mathcal{G}_t$ and the adjacent frames $(\mathcal{G}^{'}_{t-1}, \mathcal{G}^{'}_{t+1})$ to construct within-frame and cross-frame geometric representations at the coordinate space, which are denoted as $L^w_{p^i_t} = \{L^w_0, L^w_1, \ldots, L^w_{K-1}\}$ and $L^c_{p^i_t} = \{L^c_0, L^c_1, \ldots, L^c_{K-1}\}$, respectively. The $Pos(\cdot)$ function is designed to encode all these geometric representations as follows:

$$\begin{aligned} e^k_{p^i_t} = p^i_t \oplus L^w_k \oplus (p^i_t - L^w_k) \oplus \|p^i_t - L^w_k\| \\ \oplus L^c_k \oplus (p^i_t - L^c_k) \oplus \|p^i_t - L^c_k\|, \end{aligned} \tag{5.5}$$

where $e^k_{p^i_t}$ is the output of the position encoding Pos for the point p^i_t from $\mathcal{G}_t$. L^w_k, L^c_k denote the k-th neighbors of p^i_t from $L^w_{p^i_t}, L^c_{p^i_t}$, respectively and

$\oplus$ denotes the concatenation operation. $\|\|$ is the Euclidean distance calculation between points. Then I additionally construct within-frame geometric representations at the feature space $\mathbf{X}^w_{p^i_t} = \{\mathbf{X}^w_0, \mathbf{X}^w_1, \ldots, \mathbf{X}^w_{K-1}\}$ by employing KNN grouping on the per-point feature $\mathbf{F}_t$. To enhance the per-point feature $\mathbf{f}^i_t$ of each point p^i_t, $Eh(\cdot)$ is defined as:

$$\mathbf{f}^{i'}_t = \mathbf{f}^i_t + \sum_{j=0}^{K-1} \gamma(\alpha(\mathbf{f}^i_t) - \beta(\mathbf{X}^w_j) + \delta) \odot (\psi(\mathbf{X}^w_j) + \delta), \tag{5.6}$$

where $\mathbf{f}^{i'}_t$ and $\mathbf{f}^i_t$ are the ith element in $\mathbf{F}'_t$ and $\mathbf{F}_t$, respectively. $\mathbf{X}^w_j$ denotes the feature of L^w_k and $\odot$ denotes element-wise product operartion. $\gamma, \alpha, \beta, \psi$ are all single layer fully connected networks, and δ is the linear projected result of $e^k_{p^i_t}$, which is an auxiliary enhancement term derived from within-frame and cross-frame geometry. Finally, I generate a sparse but complete point cloud $\mathcal{G}'_t \in \mathbb{R}^{M\times 3}$ by projecting $\mathbf{F}'_t = \{\mathbf{f}^{i'}_t\}^{M-1}_{i=0}$ back into the coordinate space via a three-layer MLP network.

5.2.4 Detail-level Refinement

Similar to the single frame-based methods, the existing sequential point cloud upsampling frameworks [63, 64] only relies on feature expansion to expand $\mathbf{F}_t$ for upsampling, which still lead to the lack of essential details. To further recover more geometric details, I present a new detail-level refinement module that learns the correlation between the predicted high-resolution point clouds and the low-resolution geometry prior from the adjacent frames.

Given the feature $\mathbf{F}^{up}_t$ expanded by the node shuffle expansion module [52] and I first generate a coarse upsampled point cloud $\mathcal{Q}^c_t \in \mathbb{R}^{rM\times 3}$ via a three-layer MLP network. Then I predict the point-wise displacement in order to reposition each point in $\mathcal{Q}^c_t$ by leveraging a transformer-based architecture similar to the shape-level refinement module in Sec. 5.2.3. Specifically, the refined upsampled point cloud $\mathcal{Q}^r_t$ is defined as follows:

$$\mathcal{Q}^r_t = \mathcal{Q}^c_t + g_{MLP}(Eh(\mathbf{F}^{up}_t, Pos(\mathcal{Q}^c_t, \mathcal{G}'_{t-1}, \mathcal{G}'_{t+1})), \tag{5.7}$$

where g_{MLP} is a three-layer MLP network used for coordinates projection. It is worth noting that, unlike the shape-level refinement module, which focuses on geometry completion to generate a complete and sparse point cloud, the detail-level refinement module aims to delicately refine the upsampled point clouds by decorating the coarse output $\mathcal{Q}_t^c$ with more geometric details.

5.2.5 Training Objectives

To provide better supervision for PMRNet, the training objective comes from three parts: the supervision for frame-level, shape-level, and detail-level refinement.

Frame-level supervision. The goal of frame-level refinement is to accurately move the previous point cloud frame toward the current point cloud frame according to the predicted point-wise displacement, where the Chamfer Distance (CD) is adopted to supervise this refinement process:

$$L_{cd}(\mathcal{P}_{t-1}^{'}, \mathcal{P}_t) = \sum_{x \in \mathcal{P}_{t-1}^{'}} \min_{y \in \mathcal{P}_t} \|x - y\|_2^2 + \sum_{y \in \mathcal{P}_t} \min_{x \in \mathcal{P}_{t-1}^{'}} \|y - x\|_2^2, \tag{5.8}$$

where $L_{cd}(\cdot, \cdot)$ is the Chamfer Distance loss. $\mathcal{P}_{t-1}^{'}$ is the refined previous frame, and $\mathcal{P}_t$ is the current frame.

Although Eq. 5.8 can provide supervision for the frame-level refinement, it should be noted selecting robust shape descriptors from the geometry codebook is non-differential, making a stopped gradient at the vector quantization part and hindering the training of the network. To solve this issue, I follow VQVAE [62] to additionally introduce the difference between the original shape descriptor and the robust shape descriptor for gradient propagation:

$$L_{vq} = \|sg[\mathbf{S}_{t-1}] - \mathbf{S}_{t-1}^q\|_2 + \|sg[\mathbf{S}_{t-1}^q] - \mathbf{S}_{t-1}\|_2, \tag{5.9}$$

where sg denotes the "stop gradient" operation. Therefore, the total loss function for the frame-level refinement module is defined as:

$$L_f = L_{cd} + L_{vq}. \tag{5.10}$$

By using Eq. 5.10 as the objective function, I can provide effective supervision for frame-level refinement module.

Shape-level supervision. The shape-level refinement module aims to refine the incomplete point cloud to a complete and sparse one for better detail generation. Therefore, I introduce the supervision on the sparse point cloud, which can be written as:

$$L_s = L_{cd}(\mathcal{Q}_t, MLP(\mathbf{F}_t^{'})), \tag{5.11}$$

where $L_{cd}(\cdot,\cdot)$ denotes the Chamfer Distance loss. $\mathcal{Q}_t$ is the ground truth dense point cloud for the t-th frame, and $MLP(\cdot)$ is a projection implemented by a series of MLP layers to ensure the feature dimension is matched.

Detail-level supervision. Detail-level refinement module aims to refine the generated details for better upsampling result. I use the Earth Mover's Distance (EMD) between the refined dense result $\mathcal{Q}_t^r$ and the ground truth $\mathcal{Q}_t$ as the objective function for supervision, which can be written as follows:

$$L_d = \sum_{x \in \mathcal{Q}_t^r} \|x - \phi(x)\|_2, \tag{5.12}$$

where $\phi : \mathcal{Q}_t^r \rightarrow \mathcal{Q}_t$ denotes the bijection mapping.

It is worth noting that I first train the frame-level refinement module on the point cloud frame pairs, then I continue to train the other two refinement modules on local patches without updating the weights of its pre-trained model.

5.2.6 Discussion

The recent single frame-based method [37] also proposes a progressive strategy for refining upsampled point clouds, which relies on point-to-point correlation learning. However, it focuses more on the local spatial consistency of the generated points for detail refinement. For more complex cases in which essential geometry components are missing, its refinement module only performs limited point-wise displacement in very local areas, which may benefit the uniformity but barely contributes to the missing geometry.

Different from [37], I first propose a frame-level refinement module 5.2.2 to eliminate overall point cloud mismatch at the frame level, which can help the subsequent shape-level refinement 5.2.3 to effectively eliminate geometry mismatch on the low-resolution input in the single frame. Finally, I use the detail-level refinement 5.2.4 module to decorate the high-resolution coarse result with delicate geometric details. By using this progressive multi-level refinement paradigm, I can effectively generate a point cloud with more semantic meaning in a coarse-to-fine fashion for better upsampling performance.

The motivation here derives from the experimental observation that the upsampling performance of single frame-based method [26, 52, 37] is normally better on uniform and complete inputs than non-uniform and incomplete inputs. The explanation is that the feature expansion modules are well-designed for the expansion request but not good at geometry completion. My core idea is to distribute the learning tasks to appropriate modules (*i.e.*, shape-level refinement 5.2.3 module for the completion part, feature expansion module for the upsampling part, and detail-level refinement module for the final detail decoration), which incorporates multiple functions into a training-friendly framework.

5.3 Experiments

5.3.1 Experimental Setup

Dataset and Pre-processing

Following VPU, I perform training and evaluations on two human action datasets, which include a 4D mesh-based dataset DYNA [47] and a point cloud-based dataset MSR Action3D [29] captured by a depth camera. The DYNA dataset [47] consists of 134 sequences from 10 subjects with dynamic movements, and the MSR Action3D dataset [29] contains 567 point cloud sequences from 10 subjects performing 20 pre-defined human actions. I adopt the same train/test splitting strategy as in VPU for a fair comparison.

To prepare paired training data for the frame-level warping field learning, I select the source point cloud frame and the target point cloud frame from the same action sequence, and their temporal interval is randomly selected from 1 to 5 to guarantee certain movements between these two frames. In terms of the patch-based upsampling stage, I mainly follow VPU to adopt the same patch-cropping strategy to construct 3-frame triplet patches for each time step. The only difference is that the auxiliary patches $\mathcal{G}^{'}_{t-1}, \mathcal{G}^{'}_{t+1}$ come from the warped results of the first stage $\mathcal{P}^{'}_{t-1}, \mathcal{P}^{'}_{t+1}$ instead of the raw point clouds from the adjacent frames $\mathcal{P}_{t-1}, \mathcal{P}_{t+1}$.

Implementation Details

All the refinement modules in PMRNet are trained on the PyTorch platform for 40 epochs. The batch size is set as 4 and 16, respectively, when training the frame-level refinement module and the other two refinement modules. The Adam optimizer is adopted with an initial learning rate of 0.0001 and 0.001 for the two stages and a decay ratio of 0.1 after every 20 epochs.

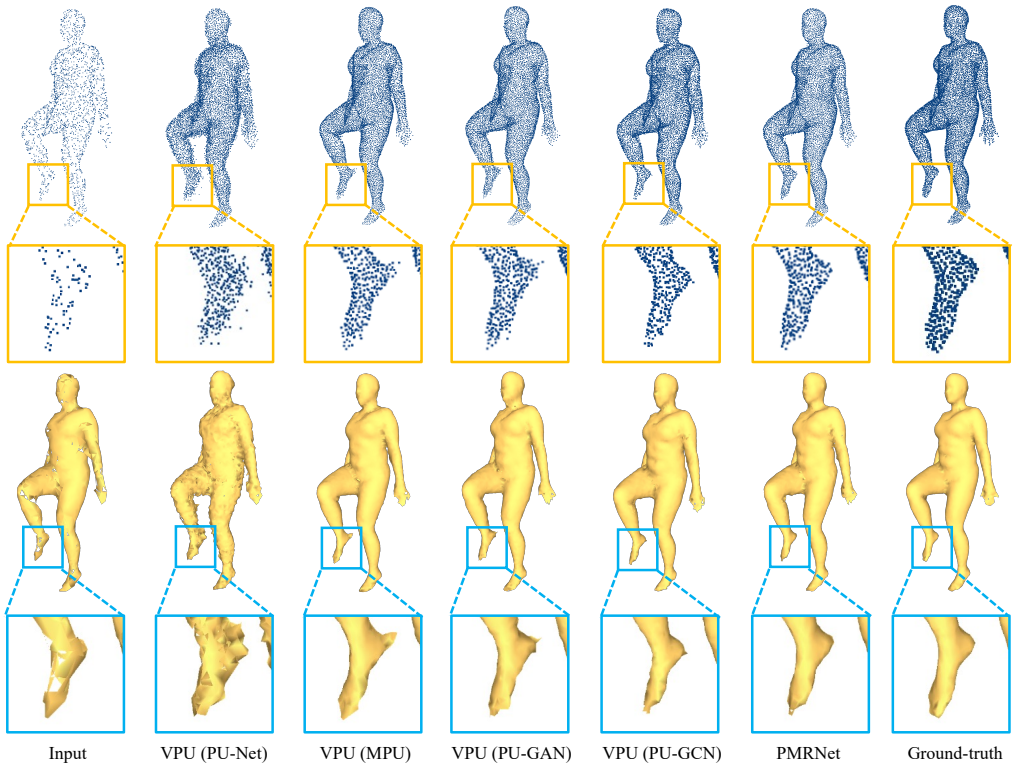


Figure 5.4: Comparison of the point cloud upsampling results (see the first row) and the surface reconstruction results (see the third row) on the DYNA dataset when using different methods under the upsampling ratio $\times 4$. I additionally provide their close-up views to compare the upsampling results (see the second row) and surface reconstruction results (see the fourth row) around the foot.

5.3.2 Comparison with the State-of-the-art Approaches

Comparison methods

Only very few attempts [63, 64] have been made for the sequential point cloud upsampling task. As SPU [63] is a frame-based approach built upon the attention mechanism and cannot be applied in scenarios with a large number of points, I only compare with the four counterparts from VPU (*i.e.*, VPU(PU-Net), VPU(MPU), VPU(PU-GAN) and VPU(PU-GCN)).

Experimental results on the DYNA Dataset

The quantitative comparison results on the DYNA dataset are reported in Table 5.1. The proposed method shows superior performance over all the counterparts in terms of all the evaluation metrics, especially the reconstruction metrics (*i.e.*, CD and HD), which demonstrates the effectiveness of the proposed two stages.

In addition to the quantitative results, I provide a visualization comparison of the human action "one leg jump", including the upsampled dense point clouds and their surface reconstruction results from [3]. From Fig. 5.4, I observe that all the counterparts built upon VPU similarly suffer from the unexpected floating points generated in the regions with dynamic movements, while the results significantly eliminate these artifacts thanks to the frame-level refinement module. This phenomenon indicates that the feature alignment module proposed in the VPU framework cannot handle the cases well when there are significant movements between the adjacent frames.

Experimental results on the MSR Action3D Dataset

Besides the DYNA dataset, I also evaluate PMRNet on a real-scanned point cloud dataset MSR Action3D [29]. The upsampling results of different methods on the MSR Action3D dataset under the ratio $\times 4$ are reported in Table 5.2. It is worth noting that the point cloud data on the MSR action3D dataset is reconstructed from the depth maps rather than

Table 5.1: Results of different methods on the DYNA dataset.

Methods	CD ↓	HD ↓	P2F ↓	Uniformity for different p		
				0.4% ↓	0.8% ↓	1.2% ↓
VPU (PU-Net)	0.377	11.391	6.439	6.392	11.749	15.499
VPU (MPU)	0.173	2.620	1.432	2.931	5.788	7.308
VPU (PU-GAN)	0.168	2.731	1.684	2.759	5.522	7.469
VPU (PU-GCN)	0.158	2.315	1.114	2.779	5.644	7.538
PMRNet	**0.142**	**1.806**	**0.825**	**2.527**	**5.365**	**7.260**

Table 5.2: Results of different methods on the MSR Action3D dataset.

Methods	CD ↓	HD ↓
VPU (PU-Net)	0.787	16.201
VPU (MPU)	0.153	3.766
VPU (PU-GAN)	0.317	6.129
VPU (PU-GCN)	0.187	4.962
PMRNet	**0.144**	**3.215**

directly sampling from the ground-truth mesh, thus the evaluation results in terms of P2F and the uniformity scores are not available. From Table 5.2, I observe that the sequentail point cloud upsampling method still outperforms its single frame-based counterparts on the real-scanned dataset, which indicates that it is beneficial to exploit rich temporal information from the neighboring frames to enhance the upsampling results on the real-scanned dataset.

The visualization results on the MSR Action3D dataset are provided in Figure 5.5. VPU(PU-Net), VPU(PU-GAN), and VPU(PU-GCN) produce large amounts of outliers above the surface. As a result, I cannot observe the clear boundary of the human arm. While VPU(MPU)is able to preserve the arm boundary, and even generate points at the original vacant regions. However, it suffers from severe points clustering issues (see the close-up views). Differently, the proposed PMRNet is able to gather reliable spatio-temporal information from neighboring frames and generate uniformly distributed point sets with complete and smooth shape boundaries as well as delicate details.

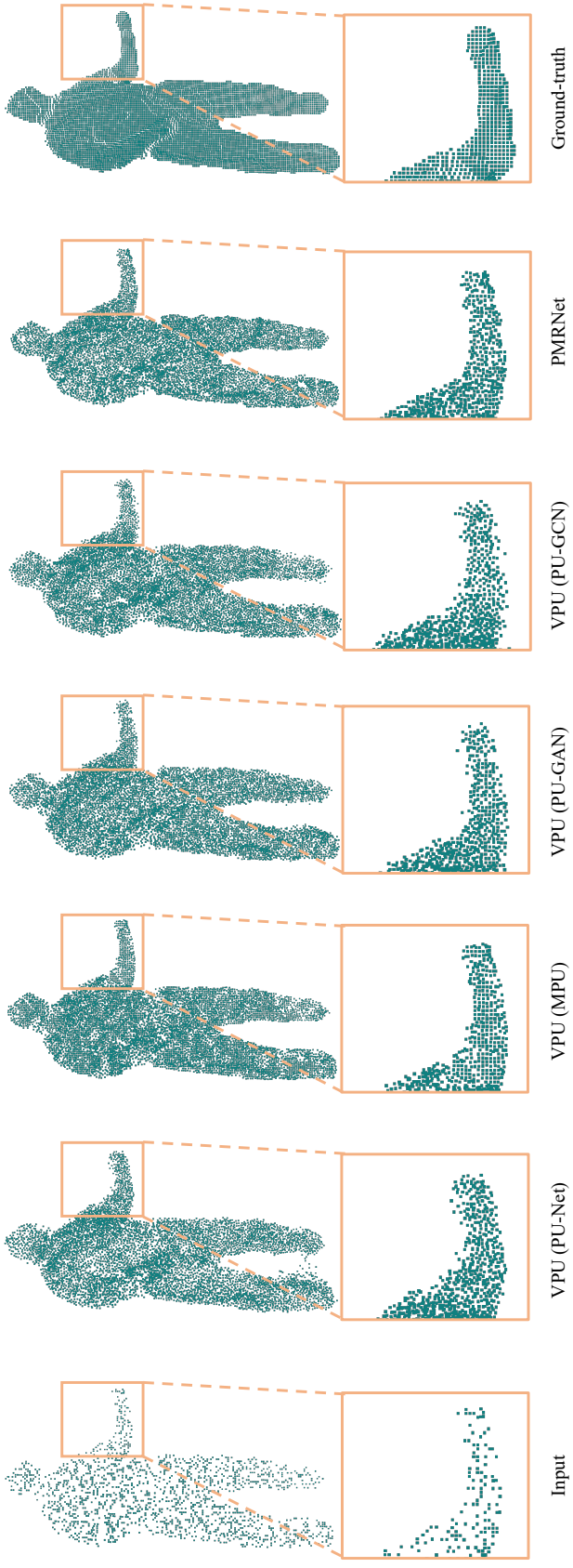


Figure 5.5: Comparison of the upsampling results on the MSR Action3D dataset when using different methods under the upsampling ratio $\times 4$. I additionally provide their close-up views around the right arm (*i.e.,* the orange boxes) to compare the upsampling results in this local area. Best viewed on screen.

Table 5.3: Results of different methods on the DYNA dataset when using different upsampling ratios.

Methods	× 2		× 4		× 8		× 16	
	CD ↓	HD ↓	CD ↓	HD ↓	CD ↓	HD ↓	CD ↓	HD ↓
VPU (PU-Net)	0.983	20.341	1.119	23.835	1.197	25.206	1.228	25.914
VPU (MPU)	0.398	3.861	0.211	4.245	0.147	4.702	0.116	5.007
VPU (PU-GAN)	0.621	12.759	0.602	15.067	0.674	16.919	0.749	18.368
VPU (PU-GCN)	0.485	4.586	0.322	5.905	0.267	7.176	0.236	8.027
PMRNet	**0.340**	**3.117**	**0.194**	**3.803**	**0.131**	**4.022**	**0.108**	**4.435**

Experimental results when using different upsampling ratios

I take the DYNA dataset as an example to additionally provide more results of different methods when using different upsampling ratios r (*i.e.*, $\times 2$, $\times 4$, $\times 8$, and $\times 16$) in Table 5.3. Specifically, I follow VPU to train all the comparison methods by using the upsampling ratio $\times 2$, and then in the testing stage, I repeat the $\times 2$ upsampling process multiple times according to the predefined upsampling ratio. I observe that PMRNet outperforms its counterparts VPU (PU-Net), VPU (MPU), VPU (PU-GAN) and VPU (PU-GCN) at different upsampling ratios. The reason behind this is that the carefully designed modules are effective in preserving local geometric details and eliminating the artifacts derived from the motion between frames.

5.3.3 Ablation Study

To further validate the effectiveness of different components in PMRNet, I take the DYNA dataset as an example to conduct a detailed ablation study. Concretely, for the frame-level refinement 5.2.2 module, I construct two variants (denoted as "w/o vq" and "w/o warping") to evaluate the performance without using the vector quantization and the entire warping process, respectively. In terms of the patch-based upsampling stage, I build another two variants "w/o shape refinement" and "w/o detail refinement" by removing either the shape-level refinement 5.2.3 module or the detail-level refinement 5.2.4 module from the full pipeline.

Table 5.4: Results of VPU (PU-GAN) and its four variants on the DYNA dataset.

Methods	CD ↓	HD ↓	P2F ↓	Uniformity for different p		
				0.4% ↓	0.8% ↓	1.2% ↓
w/o vq	0.189	3.095	1.870	3.066	6.122	7.981
w/o warping	0.220	3.714	2.099	4.151	8.422	11.137
w/o shape refinement	0.188	3.334	1.846	3.393	6.941	9.089
w/o detail refinement	0.175	3.668	1.788	2.934	6.019	7.788
Full Model	**0.142**	**1.806**	**0.825**	**2.527**	**5.365**	**7.260**

The comparison results are summarized in Table 5.4. Compared with all the variants, "w/o warping" achieves the worst performance, especially on the HD score, as directly extracting geometry information from the original adjacent frames without flow estimation may introduce unexpected artifacts. "w/o vq" achieves a significant improvement when compared with "w/o warping" but still slightly worse than the full pipeline "Full Model". On the other hand, the results of "w/o shape refinement" suffer from higher matching errors since the feature expansion cannot expand essential geometry based on incomplete input point clouds, especially in some vacant regions. By using the shape-level refinement module, "w/o detail refinement" is able to achieve comparable reconstruction scores with "Full Model"; however, it fails to preserve some local details as well as the uniformity of the predicted dense point clouds. After adding all the proposed modules, "Full Model" achieves the best performance in terms of all the evaluation metrics, which indicates it is beneficial to integrate all the refinement modules for better point cloud restoration.

5.4 Summary

In this chapter, I propose a new sequential point cloud upsampling framework called Progressive Multi-level Refinement Network (PMR-Net), which achieves the state-of-the-art sequential point cloud upsampling performance by reliably disentangling motion and geometry from the dynamic point clouds and performing refinement accordingly at

multiple levels. I first propose a robust frame-level refinement module to alleviate the severe patch mismatch issues commonly mentioned in the previous sequential point cloud upsampling approaches by warping neighboring frames to the coordinate space aligned with the target frame. I also propose a new upsampling paradigm built in a coarse-to-fine manner instead of the direct feature expansion, which relies on a shape-level refinement module to complete the over-sparse regions and a detail-level refinement module to generate delicate geometric details. The dense results recovered in the progressive and multi-level fashion are equipped with more refined details corresponding to specific semantic meanings. Extensive experiments on multiple datasets demonstrate the effectiveness of the proposed PMRNet.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this thesis, I have proposed three approaches for sequential point cloud upsampling which are committed to recovering the distorted 3D objects or scenes in real-world scenarios.

In this chapter, I first summarize the precious experience from the exploration in this research and then conclude the contributions of this thesis.

Different from single frame-based point cloud upsampling, the proposed sequential point cloud upsampling concentrates on how to effectively extract, align, and aggregate the temporal information provided by multiple point cloud frames. Normally, when the movements between frames are moderate, I simply project the geometry information of each frame into high-dimension feature space and perform attention-based alignment at the feature level. However, when the movements between frames cannot be neglected, the points with similar semantics may be too far away for capture. In this case, it's much harder for the networks to identify if the reference points are useful points or just outliers, which leads to unexpected predictions during upsampling. Thus, to further alleviate the gap between frames in these challenging cases, it's beneficial to first lift the task from the patch-level to the frame-level so that the networks are able to learn the correspondence between frames

more easily. The objective of this operation is to eliminate the spatio-temporal gap between frames so that the aligned information from reference frames shares the same distribution with the original points in the target frame.

The contributions of this thesis are summarized as follows:

- I have proposed a frame-based sequential point cloud upsampling method called SPU, which aims to upsample sparse, non-uniform, and orderless point cloud sequences by effectively exploiting rich and complementary temporal dependency from multiple inputs. By using the proposed temporal alignment module and the gating mechanism, I can perform alignment and summarization on the contributions of different kinds of temporal dependencies for producing the optimal combination for feature enhancement.
- I have proposed the first patch-based sequential point cloud upsampling framework called VPU, which achieves considerable improvement in efficiency and generalizability. To cope with the inherent alignment difficulty between local patches, I propose a spatio-temporal aggregation module compatible with the temporal patch-cropping strategy to effectively extract, align, and aggregate the spatio-temporal hints from multiple consecutive frames at the feature space. The aggregated feature can be readily incorporated with the existing single frame-based methods for regular upsampling.
- I have proposed a novel sequential point cloud upsampling paradigm via progressive multi-level refinement called PRMNet, which includes three refinement modules performed on frame-level, shape-level, and detail-level. The frame-level refinement can disentangle clean geometry contexts from the motion information via warping-based alignment, which benefits the subsequent upsampling process. While the refinement on the shape level and detail level successively extends the direct upsampling into a coarse-to-fine manner, which significantly improves the upsampling performance even in over-sparse regions.

I have conducted comprehensive experiments on multiple dynamic point cloud datasets to demonstrate the effectiveness of the SPU, VPU, and PRMNet frameworks.

6.2 Future Work

Two main research directions can be further explored in future work: (1) Sequential Point Cloud Upsampling based on Large-scale Pretrained Models (2) Sequential Point Cloud Upsampling for 3D Surface Reconstruction.

Sequential Point Cloud Upsampling based on Large-scale Pre-trained Models. Both single frame-based methods and the sequential upsampling methods focus on restoring the original geometry structure according to the distorted input point clouds. However, they fail to generate reasonable geometry when there is insufficient or no guidance. Recently, 2D generative models built upon large pre-trained models have shown remarkable generative capability in realistic image and video synthesis via learning representative priors from large-scale datasets in advance. Therefore, it is a promising and inspiring research topic to restore dynamic 3D objects or scenes under challenging cases by using large-scale pre-trained models.

Sequential Point Cloud Upsampling for 3D Surface Reconstruction. Since the point cloud is one of the explicit discrete representations of 3D surfaces, studies on surface reconstruction can provide inspiring directions for the point cloud community. Plenty of works based on neural representations (*e.g.*, neural distance field and neural radiance field) have been proposed for static/dynamic object or scene modeling by using implicit functions. These compact neural representations support infinite position querying, which is compatible with the upsampling task. Therefore, how to introduce the representing power of these neural representations into the upsampling paradigm is also a research direction full of potential.

In conclusion, sequential point cloud upsampling is currently a research field with increasing popularity and rapid development. The steadily developing deep learning techniques will motivate more real-world applications for better service.

Bibliography

[1] A. Akhtar, Z. Li, G. Van der Auwera, L. Li, and J. Chen. "Pu-dense: Sparse tensor-based point cloud geometry upsampling". In: *IEEE Transactions on Image Processing* 31 (2022), pp. 4133–4148.

[2] A. Behl, D. Paschalidou, S. Donné, and A. Geiger. "Pointflownet: Learning representations for rigid motion estimation from point clouds". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 7962–7971.

[3] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, and G. Taubin. "The ball-pivoting algorithm for surface reconstruction". In: *IEEE transactions on visualization and computer graphics* 5.4 (1999), pp. 349–359.

[4] F. L. Bookstein and W. Green. "A thin-plate spline and the decomposition of deformations". In: *Mathematical Methods in Medical Imaging* 2.14-28 (1993), p. 3.

[5] H. Chang, H. Zhang, L. Jiang, C. Liu, and W. T. Freeman. "Maskgit: Masked generative image transformer". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 11315–11325.

[6] J. Chen, J. Ma, C. Yang, L. Ma, and S. Zheng. "Non-rigid point set registration via coherent spatial mapping". In: *Signal Processing* 106 (2015), pp. 62–72.

[7] C. Choy, J. Gwak, and S. Savarese. "4d spatio-temporal convnets: Minkowski convolutional neural networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 3075–3084.

[8] H. Chui and A. Rangarajan. "A new point matching algorithm for non-rigid registration". In: *Computer Vision and Image Understanding* 89.2-3 (2003), pp. 114–141.

[9] M. Corsini, P. Cignoni, and R. Scopigno. "Efficient and flexible sampling with blue noise properties of triangular meshes". In: *IEEE transactions on visualization and computer graphics* 18.6 (2012), pp. 914–924.

[10] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei. "Deformable convolutional networks". In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 764–773.

[11] P. Esser, R. Rombach, A. Blattmann, and B. Ommer. "Imagebart: Bidirectional context with multinomial diffusion for autoregressive image synthesis". In: *Advances in Neural Information Processing Systems* 34 (2021).

[12] P. Esser, R. Rombach, and B. Ommer. "Taming transformers for high-resolution image synthesis". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 12873–12883.

[13] H. Fan, Y. Yang, and M. Kankanhalli. "Point 4D transformer networks for spatio-temporal modeling in point cloud videos". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 14204–14213.

[14] W. Feng, J. Zhang, H. Cai, H. Xu, J. Hou, and H. Bao. "Recurrent multi-view alignment network for unsupervised surface registration". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 10297–10307.

[15] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. "Vision meets robotics: The kitti dataset". In: *The International Journal of Robotics Research* 32.11 (2013), pp. 1231–1237.

[16] X. Gu, Y. Wang, C. Wu, Y. J. Lee, and P. Wang. "Hplflownet: Hierarchical permutohedral lattice flownet for scene flow estimation on large-scale point clouds". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 3254–3263.

[17] Y. Gu, X. Wang, L. Xie, C. Dong, G. Li, Y. Shan, and M.-M. Cheng. "VQFR: Blind Face Restoration with Vector-Quantized Dictionary and Parallel Decoder". In: *Proceedings of the European Conference on Computer Vision*. 2022.

[18] M. Haris, G. Shakhnarovich, and N. Ukita. "Recurrent back-projection network for video super-resolution". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 3897–3906.

[19] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, and A. Markham. "RandLA-Net: Efficient semantic segmentation of large-scale point clouds". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 11108–11117.

[20] S. Ji, W. Xu, M. Yang, and K. Yu. "3D convolutional neural networks for human action recognition". In: *IEEE transactions on pattern analysis and machine intelligence* 35.1 (2012), pp. 221–231.

[21] B. Jian and B. C. Vemuri. "Robust point set registration using gaussian mixture models". In: *IEEE transactions on pattern analysis and machine intelligence* 33.8 (2010), pp. 1633–1645.

[22] A. Kappeler, S. Yoo, Q. Dai, and A. K. Katsaggelos. "Super-resolution of compressed videos using convolutional neural networks". In: *2016 IEEE International Conference on Image Processing (ICIP)*. IEEE. 2016, pp. 1150–1154.

[23] M. Kazhdan and H. Hoppe. "Screened poisson surface reconstruction". In: *ACM Transactions on Graphics (ToG)* 32.3 (2013), pp. 1–13.

[24] D. P. Kingma and J. Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

[25] P. Li, J. Shi, and S. Shen. "Joint Spatial-Temporal Optimization for Stereo 3D Object Tracking". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 6877–6886.

[26] R. Li, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng. "Pu-gan: a point cloud upsampling adversarial network". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 7203–7212.

[27] R. Li, X. Li, P.-A. Heng, and C.-W. Fu. "Point cloud upsampling via disentangled refinement". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 344–353.

[28] R. Li, X. Li, P.-A. Heng, and C.-W. Fu. "PointAugment: an Auto-Augmentation Framework for Point Cloud Classification". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 6378–6387.

[29] W. Li, Z. Zhang, and Z. Liu. "Action recognition based on a bag of 3d points". In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition-Workshops*. IEEE. 2010, pp. 9–14.

[30] Y. Li and T. Harada. "Lepard: Learning partial point cloud matching in rigid and deformable scenes". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 5554–5564.

[31] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen. "Pointcnn: Convolution on x-transformed points". In: *Advances in neural information processing systems*. 2018, pp. 820–830.

[32] D. Liu, Z. Wang, Y. Fan, X. Liu, Z. Wang, S. Chang, and T. Huang. "Robust video super-resolution with learned temporal dynamics". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 2507–2515.

[33] D. Liu, Z. Wang, Y. Fan, X. Liu, Z. Wang, S. Chang, X. Wang, and T. S. Huang. "Learning temporal dynamics for video super-resolution: A deep learning approach". In: *IEEE Transactions on Image Processing* 27.7 (2018), pp. 3432–3445.

[34] X. Liu, C. R. Qi, and L. J. Guibas. "Flownet3d: Learning scene flow in 3d point clouds". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 529–537.

[35] X. Liu, M. Yan, and J. Bohg. "MeteorNet: Deep learning on dynamic 3D point cloud sequences". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 9246–9255.

[36] Y. Liu, B. Fan, S. Xiang, and C. Pan. "Relation-shape convolutional neural network for point cloud analysis". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 8895–8904.

[37] C. Long, W. Zhang, R. Li, H. Wang, Z. Dong, and B. Yang. "Pc2-pu: Patch correlation and point correlation for effective point cloud upsampling". In: *Proceedings of the 30th ACM International Conference on Multimedia*. 2022, pp. 2191–2201.

[38] F. Lu, G. Chen, Y. Liu, Z. Li, S. Qu, and T. Zou. "MoNet: Motion-based Point Cloud Prediction Network". In: *arXiv preprint arXiv:2011.10812* (2020).

[39] F. Lu, G. Chen, S. Qu, Z. Li, Y. Liu, and A. Knoll. "Pointinet: Point cloud frame interpolation network". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. 3. 2021, pp. 2251–2259.

[40] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley. "Least squares generative adversarial networks". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 2794–2802.

[41] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger. "Occupancy Networks: Learning 3D Reconstruction in Function Space". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 4460–4470.

[42] G. P. Meyer, A. Laddha, E. Kee, C. Vallespi-Gonzalez, and C. K. Wellington. "Lasernet: An efficient probabilistic 3d object detector for autonomous driving". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 12677–12686.

[43] Y. Min, Y. Zhang, X. Chai, and X. Chen. "An Efficient PointLSTM for Point Clouds Based Gesture Recognition". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 5761–5770.

[44] A. Myronenko and X. Song. "Point set registration: Coherent point drift". In: *IEEE transactions on pattern analysis and machine intelligence* 32.12 (2010), pp. 2262–2275.

[45] M. Niemeyer, L. Mescheder, M. Oechsle, and A. Geiger. "Occupancy Flow: 4D Reconstruction by Learning Particle Dynamics". In: *International Conference on Computer Vision*. Oct. 2019.

[46] D. Petrov and E. Kalogerakis. "Cross-Shape Graph Convolutional Networks". In: *arXiv preprint arXiv:2003.09053* (2020).

[47] G. Pons-Moll, J. Romero, N. Mahmood, and M. J. Black. "Dyna: A model of dynamic human shape in motion". In: *ACM Transactions on Graphics (TOG)* 34.4 (2015), pp. 1–14.

[48] L. Prantl, N. Chentanez, S. Jeschke, and N. Thuerey. "Tranquil Clouds: Neural Networks for Learning Temporally Coherent Features in Point Clouds". In: *arXiv preprint arXiv:1907.05279* (2019).

[49] C. R. Qi, O. Litany, K. He, and L. J. Guibas. "Deep hough voting for 3d object detection in point clouds". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 9277–9286.

[50] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. "Pointnet: Deep learning on point sets for 3d classification and segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 652–660.

[51] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. "Pointnet++: Deep hierarchical feature learning on point sets in a metric space". In: *Advances in neural information processing systems*. 2017, pp. 5099–5108.

[52] G. Qian, A. Abualshour, G. Li, A. Thabet, and B. Ghanem. "Pu-gcn: Point cloud upsampling using graph convolutional networks". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 11683–11692.

[53] Y. Qian, J. Hou, S. Kwong, and Y. He. "Pugeo-net: A geometry-centric network for 3D point cloud upsampling". In: *European Conference on Computer Vision*. Springer. 2020, pp. 752–769.

[54] A. Ranjan, T. Bolkart, S. Sanyal, and M. J. Black. "Generating 3D faces using convolutional mesh autoencoders". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 704–720.

[55] D. Rempe, T. Birdal, Y. Zhao, Z. Gojcic, S. Sridhar, and L. J. Guibas. "CaSPR: Learning Canonical Spatiotemporal Point Cloud Representations". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 13688–13701.

[56] M. S. Sajjadi, R. Vemulapalli, and M. Brown. "Frame-recurrent video super-resolution". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 6626–6634.

[57] T. Shan, J. Wang, F. Chen, P. Szenher, and B. Englot. "Simulation-based lidar super-resolution for ground vehicles". In: *Robotics and Autonomous Systems* 134 (2020), p. 103647.

[58] H. Shi, G. Lin, H. Wang, T.-Y. Hung, and Z. Wang. "SpSequenceNet: Semantic Segmentation Network on 4D Point Clouds". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 4574–4583.

[59] S. Shi, X. Wang, and H. Li. "Pointrcnn: 3d object proposal generation and detection from point cloud". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 770–779.

[60] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas. "Kpconv: Flexible and deformable convolution for point clouds". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 6411–6420.

[61] Y. Tian, Y. Zhang, Y. Fu, and C. Xu. "Tdan: Temporally-deformable alignment network for video super-resolution". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 3360–3369.

[62] A. Van Den Oord, O. Vinyals, et al. "Neural discrete representation learning". In: *Advances in Neural Information Processing Systems* 30 (2017).

[63] K. Wang, L. Sheng, S. Gu, and D. Xu. "Sequential point cloud upsampling by exploiting multi-scale temporal dependency". In: *IEEE Transactions on Circuits and Systems for Video Technology* 31.12 (2021), pp. 4686–4696.

[64] K. Wang, L. Sheng, S. Gu, and D. Xu. "VPU: A Video-Based Point Cloud Upsampling Framework". In: *IEEE Transactions on Image Processing* 31 (2022), pp. 4062–4075.

[65] L. Wang, J. Chen, X. Li, and Y. Fang. "Non-rigid point set registration networks". In: *arXiv preprint arXiv:1904.01428* (2019).

[66] L. Wang, X. Li, J. Chen, and Y. Fang. "Coherent point drift networks: Unsupervised learning of non-rigid point set registration". In: *arXiv preprint arXiv:1906.03039* (2019).

[67] X. Wang, K. C. Chan, K. Yu, C. Dong, and C. Change Loy. "Edvr: Video restoration with enhanced deformable convolutional networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2019, pp. 0–0.

[68] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon. "Dynamic graph cnn for learning on point clouds". In: *ACM Transactions on Graphics (TOG)* 38.5 (2019), pp. 1–12.

[69] Z. Wang, J. Chen, and S. C. Hoi. "Deep learning for image super-resolution: A survey". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2020).

[70] Z. Wang, S. Li, H. Howard-Jenkins, V. Prisacariu, and M. Chen. "FlowNet3D++: Geometric losses for deep scene flow estimation". In: *The IEEE Winter Conference on Applications of Computer Vision*. 2020, pp. 91–98.

[71] X. Weng, J. Wang, S. Levine, K. Kitani, and N. Rhinehart. "Sequential Forecasting of 100,000 Points". In: *arXiv preprint arXiv:2003.08376* (2020).

[72] W. Wu, Z. Qi, and L. Fuxin. "Pointconv: Deep convolutional networks on 3d point clouds". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 9621–9630.

[73] W. Wu, Z. Y. Wang, Z. Li, W. Liu, and L. Fuxin. "PointPWC-Net: A Coarse-to-Fine Network for Supervised and Self-Supervised Scene Flow Estimation on 3D Point Clouds". In: *European Conference on Computer Vision*. Springer, Cham, Switzerland. 2020, pp. 88–107.

[74] Q. Xie, Y.-K. Lai, J. Wu, Z. Wang, Y. Zhang, K. Xu, and J. Wang. "MLCVNet: Multi-Level Context VoteNet for 3D Object Detection". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 10447–10456.

[75] S. Xie, J. Gu, D. Guo, C. R. Qi, L. Guibas, and O. Litany. "Pointcontrast: Unsupervised pre-training for 3d point cloud understanding". In: *European Conference on Computer Vision*. Springer. 2020, pp. 574–591.

[76] Y. Xu, T. Fan, M. Xu, L. Zeng, and Y. Qiao. "Spidercnn: Deep learning on point sets with parameterized convolutional filters". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 87–102.

[77] X. Yan, C. Zheng, Z. Li, S. Wang, and S. Cui. "PointASNL: Robust Point Clouds Processing using Nonlocal Neural Networks with Adaptive Sampling". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 5589–5598.

[78] W. Yifan, S. Wu, H. Huang, D. Cohen-Or, and O. Sorkine-Hornung. "Patch-based progressive 3d point set upsampling". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 5958–5967.

[79] J. Yin, J. Shen, C. Guan, D. Zhou, and R. Yang. "LiDAR-based Online 3D Video Object Detection with Graph-based Message Passing and Spatiotemporal Transformer Attention". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 11495–11504.

[80] L. Yu, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng. "Ec-net: an edge-aware point set consolidation network". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 386–402.

[81] L. Yu, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng. "Pu-net: Point cloud upsampling network". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 2790–2799.

[82] W. Yuan, T. Khot, D. Held, C. Mertz, and M. Hebert. "Pcn: Point completion network". In: *2018 international conference on 3D vision (3DV)*. IEEE. 2018, pp. 728–737.

[83] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena. "Self-attention generative adversarial networks". In: *International conference on machine learning*. PMLR. 2019, pp. 7354–7363.

[84] H. Zhao, L. Jiang, J. Jia, P. H. Torr, and V. Koltun. "Point transformer". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 16259–16268.

[85] Y. Zhao, X. Li, W. Zhang, S. Zhao, M. Makkie, M. Zhang, Q. Li, and T. Liu. "Modeling 4D fMRI Data via Spatio-Temporal Convolutional Neural Networks (ST-CNN)". In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2018, pp. 181–189.

[86] S. Zhou, K. Chan, C. Li, and C. C. Loy. "Towards robust blind face restoration with codebook lookup transformer". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 30599–30611.

[87] Y. Zhou and O. Tuzel. "Voxelnet: End-to-end learning for point cloud based 3d object detection". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 4490–4499.

[88] S. Zhu, S. Liu, C. C. Loy, and X. Tang. "Deep Cascaded Bi-Network for Face Hallucination". In: *European conference on computer vision*. Springer, Cham, Switzerland. 2016, pp. 614–630.