UNIVERSITY OF CALIFORNIA,
IRVINE

Towards Efficient Deep Learning for Human-Centric Visual Understanding and Generation

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Computer Science

by

Haoyu Ma

Dissertation Committee:
Professor Xiaohui Xie, Chair
Professor Charless C. Fowlkes
Professor Erik B. Sudderth

2024

DEDICATION

To my family

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

First and foremost, I'd like to express my deep appreciation to Professor Xiaohui Xie, my dedicated advisor, for his invaluable advice, continuous support, and patience during my Ph.D. study. His vast expertise and abundant experience have been a constant source of encouragement throughout my academic research and daily life.

Besides, I am extremely grateful to my defense committee members, Professor Charless C. Fowlkes and Professor Erik B. Sudderth, for their valuable suggestions on this dissertation. Meanwhile, I would like to extend my sincere thanks to my advancement to candidacy committee member Professor Jing Zhang and Professor Jack Xin, for their constructive comments and encouragement. It is my honor to have them on my committee.

Additionally, I'd like to express my sincere gratitude to my collaborators: Professor Zhangyang Wang, Professor Tianlong Chen, Professor Yanzhi Wang, Professor Xiaolong Ma, Professor Geng Yuan, Zhe Wang, Chenyu You, Zhenglun Kong, Shiwei Liu, Yifei Chen, Ting-Kuei Hu, Mengshu Sun, Yuchao Gu, Ajay Jaiswal, Chengming Zhang, etc. Their invaluable expertise and efforts have significantly enhanced the quality and depth of my research.

Moreover, I'd like to extend my sincere thanks to my labmates, Liangjian Chen, Hao Tang, Deying Kong, Yingxin Cao, Xiangyi Yan, Xingwei Liu, Shanlin Sun, Kun Han, Thanh-Tung Le, Yifeng Xiong, Tong Zhang, Pooya Khosravi, Hasan Celik, and Junayed Naushad, for their insightful discussion and effective collaboration, and my friends, Yuxiang Lu, Wenkai Zhang, Fan Ye, Yunhan Zhao, Junze Liu, Aodong Li, Keqing Fu, Chenyang Bao, Lequan Wang, Junge Wang, etc, for enriching my life at Irvine.

Furthermore, I extend my gratitude to the mentors from my industry internships: Shahin Mahdizadehaghdam, Bichen Wu, Zhipeng Fan, Wenliang Zhao, Lior Shapira, Shu Liang, Amin Jourabloo, Tao Xu, Peihong Guo, Handong Zhao, Zhe Lin, Ajinkya Kale, Tong Yu, Jiuxiang Gu, Shi-Xiong Zhang, Meng Yu, and Dong Yu. Their guidance help me broaden my research experience and skills.

Lastly, but the most importantly, I would love to give the greatest thank to my parents and my wife, for their unconditional deep love.

VITA

Haoyu Ma

EDUCATION

Doctor of Philosophy in Computer Science **2024**
University of California, Irvine *Irvine, CA*

Bachelor of Science in Computational Sciences **2019**
Southeast University *Nanjing, Jiangsu, China*

RESEARCH EXPERIENCE

Research Scientist Intern **2023**
Meta Platforms, Inc *Menlo Park, CA*

Research Scientist Intern **2022**
Meta Platforms, Inc *Burlingame, CA*

Research Intern **2021**
Adobe Research *San Jose, CA*

Research Intern **2020**
Tencent America *Seattle, WA*

TEACHING EXPERIENCE

Teaching Assistant **2019–2024**
University of California, Irvine *Irvine, CA*

REFEREED CONFERENCE PUBLICATIONS

Nonparametric Structure Regularization Machine for 2D Hand Pose Estimation **2020**
IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)

Undistillable: Making A Nasty Teacher That CANNOT Teach Students **2021**
International Conference on Learning Representations (ICLR) (Spotlight)

PD-Net: Quantitative Motor Function Evaluation for Parkinson's Disease via Automated Hand Gesture Analysis **2021**
ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)

TransFusion: Cross-view Fusion with Transformer for 3D Human Pose Estimation **2021**
British Machine Vision Conference (BMVC)

EI-CLIP: Entity-aware Interventional Contrastive Learning for E-commerce Cross-modal Retrieval **2022**
IEEE / CVF Computer Vision and Pattern Recognition Conference (CVPR)

PPT: token-Pruned Pose Transformer for monocular and multi-view human pose estimation **2022**
European Conference on Computer Vision (ECCV)

Sparse Logits Suffice to Fail Knowledge Distillation **2022**
International Conference on Learning Representations (ICLR) Workshop

Peeling the Onion: Hierarchical Reduction of Data Redundancy for Efficient Vision Transformer Training **2023**
AAAI Conference on Artificial Intelligence(AAAI)

MaskINT: Video Editing via Interpolative Non-autoregressive Masked Transformers **2023**
Arxiv

CVTHead: One-shot Controllable Head Avatar with Vertex-feature Transformer **2024**
IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)

HRBP: Hardware-friendly Regrouping towards Block-based Pruning for Sparse CNN Training **2024**
Conference on Parsimony and Learning (CPAL)

ABSTRACT OF THE DISSERTATION

Towards Efficient Deep Learning for Human-Centric Visual Understanding and Generation

By

Haoyu Ma

Doctor of Philosophy in Computer Science

University of California, Irvine, 2024

Professor Xiaohui Xie, Chair

Human-centric visual understanding and generation are pivotal in many real scenarios, such as augmented/virtual reality, human-computer interaction, and movie industry. Over the past several years, deep learning has become the dominant approach for many human-centric visual tasks, such as pose estimation, avatar reconstruction, and character animation. Despite previous progress, these tasks remain challenging under occlusion and motion blur. Besides, most of current models are computationally extensive, which hinders real-world deployment. In this dissertation, we propose various approaches to address the aforementioned challenges in order to achieve better accuracy and higher efficiency. In the first part, the task of pose estimation (a.k.a. keypoint detection) would be widely investigated, which serves as the foundation of many human-related applications. In multi-view settings, we propose a novel transformer-based networks, named TransFusion, which effectively and efficiently fuse global and long-term visual cues from different views, and incorporate the 3D geometry constraints through a novel proposed epipolar field. Besides, we further propose the token-pruned pose transformer, named PPT, to reduce the computation in both monocular and multi-view pose estimation by pruning less important tokens with the cues from the human skeleton prior, all without the need for foreground mask annotations.

In the second part, we investigate human-centric visual editing and generation under diverse conditions. We propose a novel framework named CVThead for reconstructing head avatars from a single image, allowing for the rendering of human heads with various expressions, head shape, and camera views under the control of an explicit head mesh model. CVTHead also utilizes transformers for robust learning of appearance features and enables efficient generation through point-based rendering. Besides, we propose an advanced model that enables appearance editing in video with text instructions. This model, featuring novel non-autoregressive transformers, achieves comparable performance with previous state-of-the-art works while demonstrating a significant acceleration in running time.

Chapter 1

Introduction

Human-centric visual understanding as well as generation play an important role in many applications, such as augmented/virtual reality, human-computer interaction, and movie industry. Over the last few years, deep learning algorithms have achieved success in many human-centric visual understanding and generation tasks. However, there are still some challenging cases, such as occlusion and motion blur, have not been fully solved so far. Meanwhile, efficiency is also an important factor for deployment on real applications. In this chapter, we will first introduce the background of this problem and present the outline of this dissertation.

1.1 Background

The problem of human-centric visual understanding aims to extract biometric features from RGB images or videos. Specifically, it includes pose estimation, a.k.a. keypoint detection, which aims to locate the 2D or 3D coordinates of a set of keypoints. Besides, the understanding also includes mesh reconstruction, which aims to reconstruct the vertices on the surface.

On the other hand, human-centric visual generation aims to generate or edit RGB images or videos with human contents. Thus, it is formulated as a conditional image generation tasks. These input conditionss may include biometric features like pose and mesh data. For example, we want the person to perform a new body pose based on the skeleton. The conditions can also involve camera viewpoint, enabling the generation of images from a novel view for the depicted individuals. Additionally, text can serve as a condition, allowing for the generation of images based on textual prompts, such as altering the appearance or styles.

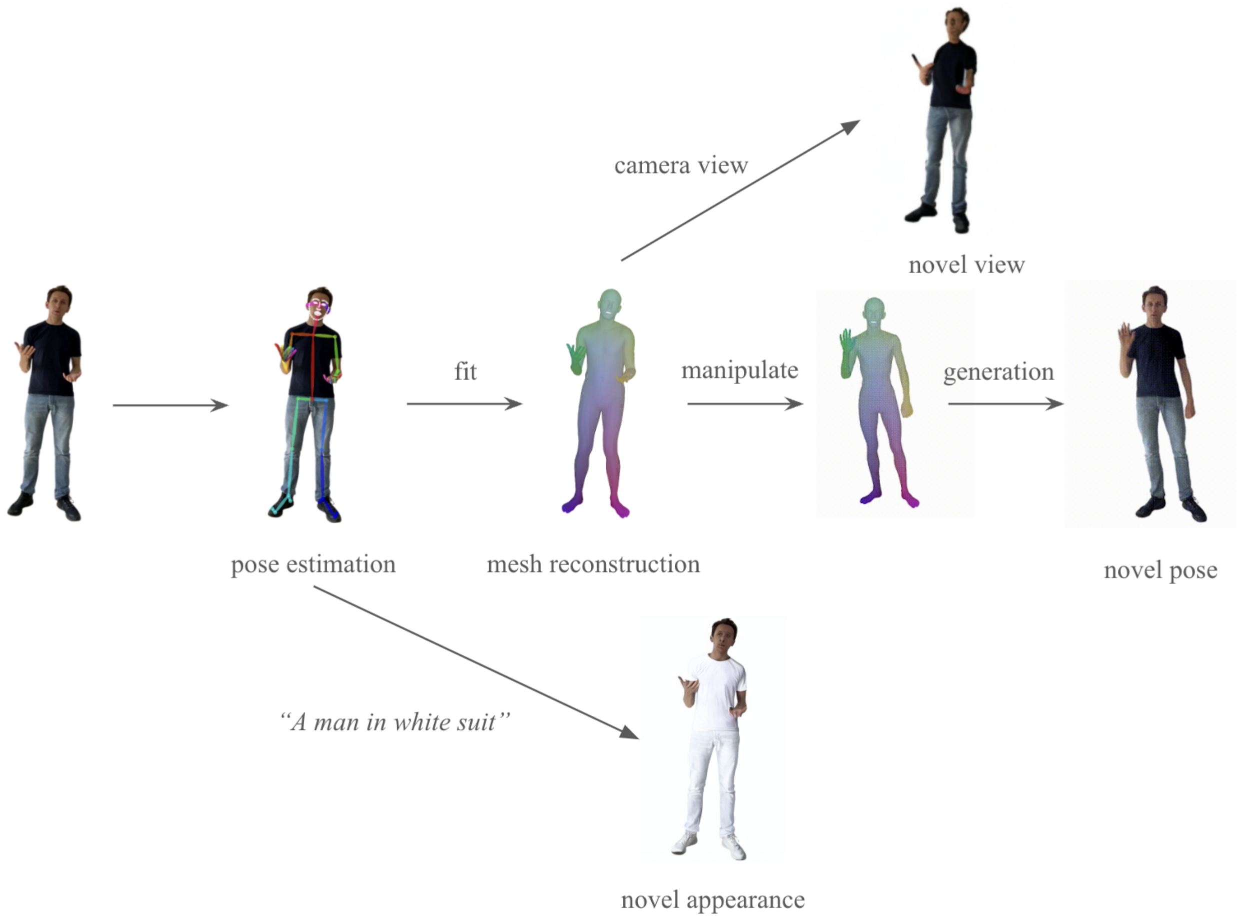


Figure 1.1: Interconnections among human-centric visual understanding and generation

Moreover, these human-centric tasks are not independent of each other, but are closely related. Pose estimation typically serves as the cornerstone for these tasks. Using estimated keypoints, we can fit parametric body mesh for these individuals. Once we've acquired the mesh, we can manipulate it by adjusting the coefficients, enabling the generation of diverse poses, shapes, and camera viewpoints. Furthermore, in an orthogonal direction, we can also modify the appearance, such as clothing or styles, using text or other reference images, all while retaining the body pose and spatial layout based on the keypoints or meshes. Fig 1.1 provides a concrete example to illustrate interconnections among human-centric visual tasks.

1.2 Outline

The subsequent structure of the dissertation unfolds as follows:

Chapter 2. In this chapter, we firstly explore human body pose estimation under the multi-view setting. Specifically, we focus on the cross-view fusion techniques, which aims to improve the 2D pose estimation by fusing corresponding information from other views. We propose a transformer-based network, named TransFusion, to globally fuse information from other views. Moreover, we propose the concept of epipolar field to encode epipolar constraints into the transformer model, which provides an efficient way of encoding correspondences between pixels of different views. Part of this chapter is published in [106].

Chapter 3. Although transformers are promising in pose estimation, efficiency remains a significant concern for real-world applications, as the computational complexity is quadratic to the number of input tokens. In this chapter, we improve the efficiency of transformer-based pose estimation networks. In detail, we propose token-pruned pose transformer, named PPT, which can automatically locate tokens of human foreground areas and prune background tokens, enabling self-attention only within selected tokens. Furthermore, we extend PPT to multi-view human pose estimation. Built upon PPT, we propose a new cross-view fusion strategy, called human area fusion, which considers all human foreground pixels as corresponding candidates. Part of this chapter is published in [108].

Chapter 4. Once we obtain the parametric mesh model of a human, we can manipulate its parameters to generate meshes with diverse pose and shape. In this chapter, we explore the generation of animatable human head image using the mesh as a conditioning factor. Existing methods for achieving explicit face control of 3D Morphable Models (3DMM) typically rely on multi-view images or videos of a single subject, making the reconstruction process complex. Additionally, the traditional rendering pipeline is time-consuming, limiting real-time animation possibilities. To this end, we introduce CVTHead, a novel approach that generates controllable head avatars from a single reference image using point-based neural rendering. CVTHead considers the sparse vertices of mesh as the point set and employs the proposed Vertex-feature Transformer to learn local feature descriptors for each vertex. This enables the modeling of long-range dependencies among all the vertices. CVTHead enables efficient rendering of novel human heads with various expressions, head poses, and camera views. These attributes can be explicitly controlled using the coefficients of 3DMMs, facilitating versatile and realistic animation in real-time scenarios. Part of this chapter is published in [109].

Chapter 5. In this chapter, we explore the task of appearance editing controlled by text prompt, which is an orthogonal direction of novel pose or view generation. More specifically, our focus lies in video-based editing, where maintaining consistency across frames is a significant challenge. Recent works suggest that Denoising Diffusion Probabilistic Models (DDPM) achieve promising results in text-based image generation and editing. However, the hundreds of denoising steps of diffusion models is extremely time-consuming, challenging the deployment in practical applications. To this end, we propose to disentangle text-based video editing into two separate stages. In the first stage, we employ existing image diffusion models to jointly edit the first and last frames of a video clip in a zero-shot way. In the second stage, we perform structure-aware frame interpolation with the proposed lightweight non-autoregressive masked generative transformers, named MaskINT. The proposed framework achieves performance comparable to other diffusion-only methods while offering a substantial acceleration in runtime during inference. Part of this chapter is published in [107].

Chapter 6. This chapter concludes the dissertation.

Chapter 2

TransFusion: Cross-view Fusion with Transformer for 3D Human Pose Estimation

Estimating the 2D human poses in each view is typically the first step in calibrated multi-view 3D pose estimation. But the performance of 2D pose detectors suffers from challenging situations such as occlusions and oblique viewing angles. To address these challenges, previous works derive point-to-point correspondences between different views from epipolar geometry and utilize the correspondences to merge prediction heatmaps or feature representations. Instead of post-prediction merge/calibration, here we introduce a transformer framework for multi-view 3D pose estimation, aiming at directly improving individual 2D predictors by integrating information from different views. Inspired by previous multi-modal transformers, we design a unified transformer architecture, named TransFusion, to fuse cues from both current views and neighboring views. Moreover, we propose the concept of *epipolar field* to encode 3D positional information into the transformer model. The 3D position encoding guided by epipolar field provides an efficient way of encoding correspondences between pixels of different views. Experiments on Human 3.6M and Ski-Pose show that our method is more efficient and has consistent improvements compared to other fusion methods. Specifically, we achieve 25.8 mm MPJPE on Human 3.6M with only 5M parameters on 256 × 256 resolution.

2.1 Problem Definition

The problem of human pose estimation aims to find a mapping from the input RGB image $\mathbf{I} \in \mathbb{R}^{h \times w \times 3}$ to the 2D or 3D positions of a set of K keypoints defined by the human body skeleton, i.e., $\mathbf{J}^{2d} \in \mathbb{R}^{K \times 2}$ or $\mathbf{J}^{3d} \in \mathbb{R}^{K \times 3}$. In multi-view human pose estimation, we have images $\{\mathbf{I^i}\}_{i=1}^{N}$ of the same subject captured from N different camera views as well as camera matrices $\{\mathbf{P^i}\}_{i=1}^{N}$ of each view. As shown in Figure 2.1, this task aims to firstly locate the 2D keypoints in each view $\{\mathbf{J}_i^{2d}\}_{i=1}^{N}$ and then perform triangulation to obtain the 3D pose in the world coordinate system $\mathbf{J}^{3d} \in \mathbb{R}^{K \times 3}$. presents the pipeline of this task.

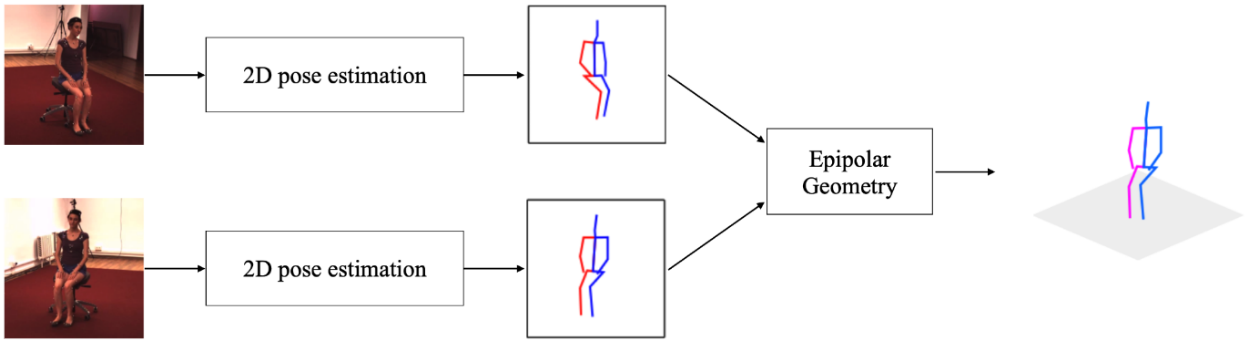


Figure 2.1: Pipeline of multi-view human pose estimation.

2.2 Introduction

Estimating the 3D locations of human joints is a critical task for many AI applications such as augmented reality, virtual reality and medical diagnosis [29]. The estimation is often carried out in two common settings: One is estimating the 3D pose from monocular images

[112, 211, 223, 48, 175, 22, 24, 160], and the other is estimating 3D poses from multiple cameras [145, 174, 23, 129, 61]. The former is challenging due to the ambiguity of depth estimation with only one view. The latter setting, the focus of this work, usually obtains better 3D pose estimation performance since the multi-view settings can help resolve depth ambiguity. Most multi-view works follow a two-step pipeline that firstly estimates 2D poses in each view and then recovers 3D pose from them. However, it is still difficult to solve challenging cases such as occlusions in the first step, and the estimated 3D poses are often inaccurate as it depends on the results from the first step.

Researchers have sought to introduce the 3D information in the first step to improve the 2D pose detector, because the challenging cases in one view are potentially easier to solve in other views. Specifically, they usually fuse the features of the neighboring view (reference view) with epipolar constraints [182, 214, 61]. Although interpretable, fusing along the epipolar line only does not fully utilize the semantic information of the reference view as the information off the epipolar line is discarded. For example, it is difficult to associate the ankle with the leg from the epipolar line in the reference view of Figure 2.2, which could be an important cue as part of the structure information for pose estimation. On the other hand, fusing all locations of other views can address this drawback. In this work, we propose the *Epipolar Field*, a more general form of the epipolar line. It assigns probabilities to all locations of the reference view and still keep the knowledge of epipolar constraints.

Recently, attention mechanisms and the transformers [165] achieve great progress in computer vision areas [170, 37, 14, 221, 99, 155]. The self-attention module [165] can capture long range dependencies and correspondences, which is difficult for the convolutional layer. Although promising, there are only a few works [99] that apply it to the 3D pose estimation tasks. To the best of our knowledge, none of the previous works have exploited the transformer architectures in the multi-view 3D pose estimation setting. Inspired by previous multi-modal transformers [151, 154, 84], we propose the *TransFusion*, a lightweight

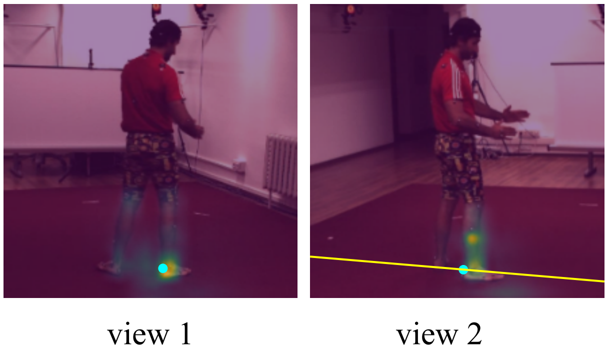


Figure 2.2: Comparison of epipolar line and attention module. Given the query pixel (cyan dot) in view 1 (current). The attention map on the view 2 (reference) indicates that the prediction relies on the image clues provided by the area of right shank, not just the corresponded right ankle. While previous methods based on epipolar line (yellow line) cannot capture this information.

framework that can utilize all pixels from both the current view itself and reference view simultaneously. As an example in Figure 2.2, the attention layer actually relies on the whole leg to infer the location of the ankle. Moreover, we add the 3D geometry positional encoding based on the epipolar field to help the transformer explicitly capture the correspondence.

Our main contributions are summarized as follows:

- We are the first to apply the transformer architecture to multi-view 3D human pose estimation. We propose the TransFusion, a unified architecture to fuse cues from multiple views.

- We propose the *epipolar field*, a novel and more general form of epipolar line. It readily integrates with the transformer through our proposed geometry positional encoding to encode the 3D relationships among different views.

- Extensive experiments are conducted to demonstrate that our TransFusion outperforms previous fusion methods on both Human 3.6M and SkiPose datasets, but requires substantially fewer parameters.

2.3 Related Work

2.3.1 Multi-view 3D Pose Estimation

Multi-view 3D pose estimation usually follows a two-step process: (1) localize 2D joints with a 2D pose estimator on each view, and (2) lift the 2D joints from multi-view images to the 3D position via triangulation. To improve the performance of 2D pose detector, researchers typically resort to sophisticated architectures to capture both low-level and high-level representations [177, 30, 119, 181, 152] or use the structural information to model the spatial constraints [161, 86, 87, 88, 28]. However, the occlusion cases are still challenging, as monocular images do not provide evidence for occlusion joints localization.

An alternative approach, more explainable, is to make the 2D pose detector 3D-aware, i.e., fusing the 2D feature heatmaps [129, 214, 182, 23, 61] from different views. Specifically, the Cross-view Fusion [129] directly learns a fixed attention weight to fuse all pairs of pixels given a pair of views. However, the learnable weight requires the multi-camera setup unchanged during the inference time, and the number of parameters is quadratic to the resolution of input images. The epipolar transformer [61] applies the non-local module [170] to obtain the weights and only fuse pixels along the epipolar line in other views. Thus it is easy to learn and flexible to use. However, sampling along the epipolar line discards off-epipolar line information and thus obtains limited information from the reference views. In the second step, researchers use graphical model with the structure of human [129] to improve the quality of triangulation or directly learn 3D pose via differentiable triangulation [74]. Our work still focuses on enhancing 2D pose by fully integrating information from different views.

2.3.2 Transformer

Vision Transformer Recently, several studies demonstrated that the transformer architectures [165] plays a significant role in a wide range of computer vision tasks, such as image classification [37, 163, 21], object detection [14, 221], and semantic segmentation [218, 173, 187]. Recently, some studies also explored applying the transformer on human pose estimation tasks [191, 94, 99, 217]. More specifically, for 2D pose estimation, TransPose [191] aims to explain the spatial dependencies of the predicted keypoints with transformers, PRTR [94] and TF-Pose [111] attempt to directly regress the joint coordinates by transformer decoders. While for the 3D pose estimation, METRO [99] firstly applies transformer to reconstruct 3D human pose and mesh from a single image, and PoseFormer [217] builds a spatial-temporal transformers with the input of 2D joint sequences for 3D pose estimation in videos. However, previous works have hardly exploited the transformer architectures on the multi-view 3D pose estimation setting, which is however an important task in the pose estimation area.

Multi-modal Transformer Transformers with multi-modality inputs such as images and texts have also been fully exploited [84, 154, 151, 93, 195, 196]. In general, these methods directly concatenate the embeddings from two sources together [84] and make the transformer itself to learn the correspondence between two modalities from millions of image-text paris [140]. Thus, these methods are quite expensive and inefficient, and difficult to apply on limited datasets. Our method, however, directly provides the correspondence between two inputs and makes the transformer explicitly learn their relationships.

2.4 Methods

2.4.1 Overview

Figure 2.3 is an overview of TransFusion. It takes two images from different views as input, and predicts the heatmaps of joints in each view. The framework consists of three modules: a CNN backbone to extract low-level features; a transformer encoder to capture both correspondence between two views and long-range spatial correlations within single view images; a head to predict the heatmaps of joints. Specifically, given images $\mathbf{I}_i \in \mathbb{R}^{3\times H_I \times W_I}$ in each view, where $i \in \{1, 2\}$ denotes view 1 and view 2, the backbone $\mathcal{F}(\cdot)$ firstly produces the low-level features $\mathbf{X}_i = \mathcal{F}(\mathbf{I}_i) \in \mathbb{R}^{d\times H\times W}$ of each image. Here d is the number of channels. H and W are the height and width of the feature map, respectively. The feature $\mathbf{X}_i$ is flattened into a sequence vector $\mathbf{X}_i^{'} \in \mathbb{R}^{L\times d}$, where $L = H \times W$. Both 2D sine positional encoding $\mathbf{E_{2D}}$ and 3D geometry positional encoding $\mathbf{E_{G}}_i$ are added onto $\mathbf{X}_i^{'}$ to make the transformer aware of position information. $\mathbf{X}_1^{'}$ and $\mathbf{X}_2^{'}$ are concatenated together to build a uniform embedding $\mathbf{X} = [\mathbf{X}_1^{'} + \mathbf{E_{2D}} + \mathbf{E_{G}}_1, \mathbf{X}_2^{'} + \mathbf{E_{2D}} + \mathbf{E_{G}}_2] \in \mathbb{R}^{2L\times d}$. The embedding $\mathbf{X}$ then enters the standard transformer encoder $\mathcal{E}(\cdot)$. Finally, the output of the transformers $\tilde{\mathbf{X}} = \mathcal{E}(\mathbf{X})$ are split into $\tilde{\mathbf{X}}_1$ and $\tilde{\mathbf{X}}_2$, which is embedding of each view, and a prediction head $\mathcal{H}(\cdot)$ takes $\tilde{\mathbf{X}}_i$ and predicts the joint heatmaps $\bar{\mathbf{H}}_i = \mathcal{H}(\tilde{\mathbf{X}}_i) \in \mathbb{R}^{k\times H_h\times W_h}$ for each view, where k is the number of joints.

2.4.2 TransFusion

Transformer Encoders The transform encoder $\mathcal{E}(\cdot)$ consists of several layers of multi-head self-attention. Let $l = 2L$ for short, given the input sequence $\mathbf{X} \in \mathbb{R}^{l\times d}$, the self-attention layer first uses linear projections to obtain a set of queries ($\mathbf{Q} \in \mathbb{R}^{l\times d}$), keys ($\mathbf{K} \in \mathbb{R}^{l\times d}$) and values ($\mathbf{V} \in \mathbb{R}^{l\times d}$) from $\mathbf{X}$. The three linear projections are parameterized

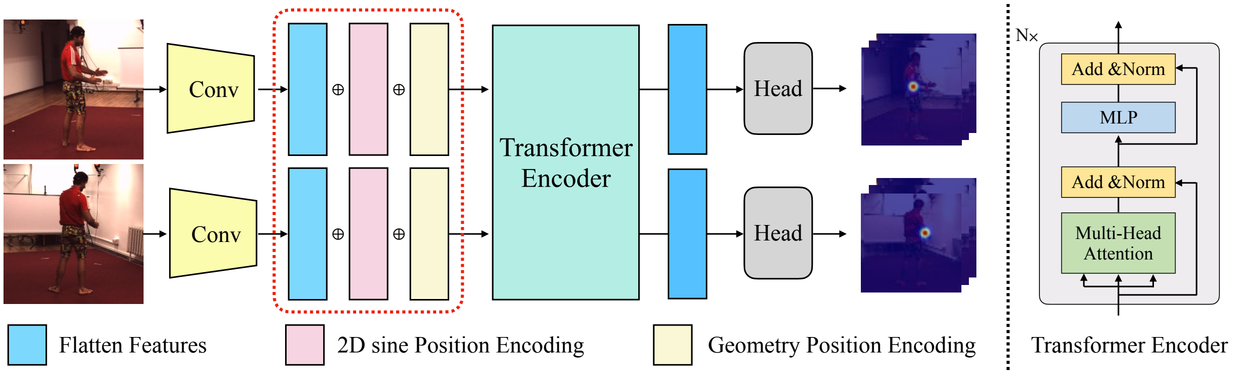


Figure 2.3: Overview of TransFusion.

by three learnable matrices $\mathbf{W}_q$, $\mathbf{W}_k$, $\mathbf{W}_v \in \mathbb{R}^{d\times d}$. Following [14], the position encoding $\mathbf{E}$ is added into the input $\mathbf{X}$ for computing the query and key. The scaled dot-product attention [165] between $\mathbf{Q}$ and $\mathbf{K}$ is adopted to compute the attention weights, and aggregate the values:

$$\mathbf{A} = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}}\right)\mathbf{V}, \tag{2.1}$$

Finally, a non-linear transformation (*i.e.*, multi layer perceptron, and the skip connection) is applied on $\mathbf{A}$ to calculate the output $\tilde{\mathbf{X}}$. As $\mathbf{X}$ is low-level features of all views, given one query pixel on the feature map, it can attend cues from the its own view and other views simultaneously through the entire network.

Positional encoding The attention layer may degenerate into a permutation-equivariant architecture without any position information. Thus, the positional encoding is necessary to make the transformer aware of position and order of input sequence. For each individual view, we follow the 2D sine positional encoding in the original transformers [37, 191], and we denote it as $\mathbf{E_{2D}}$. However, it only encodes position information from its own view, while the position information in the 3D space and that from the reference views cannot be encoded. Thus, another positional encoding $\mathbf{E}_{\mathbf{G}_i}$ (See Section 2.4.3) is required to encode the 3D location information of each view i in the 3D space.

2.4.3 Geometry Position Encoding

To make the transformers 3D-aware, we introduce 3D camera information [3, 215] into the positional encoding and propose the *Geometry Positional Encoding* (GPE). Denote the world coordinate system as $\mathbf{O}_{\text{world}}$ and the calibrated camera coordinate system as $\mathbf{O}_{\text{cam}}$. The 3D location of view i 's camera center in $\mathbf{O}_{\text{world}}$ is denoted as $\mathbf{C}_i$. Given the n-th ($n \in \{1, 2, ..., L\}$) pixel $p_i^n = (u_i^n, v_i^n)$ in the pixel array of view i, its corresponding 3D location ${\mathbf{P}_{\mathbf{c}}}_i^n = [X_i^n, Y_i^n, Z_i^n]$ in $\mathbf{O}_{\text{cam}}$ can be obtained by:

$$
{\mathbf{P}_{\mathbf{c}}}_i^n = [X_i^n, Y_i^n, Z_i^n] = [\frac{s_x^i(u_i^n - o_x^i)}{f_i} Z_i^n, \frac{s_y^i(v_i^n - o_y^i)}{f_i} Z_i^n, Z_i^n] \tag{2.2}
$$

Where f_i, (o_x^i, o_y^i), and (s_x^i, s_y^i) are the focal length, the principle point, and the scale factor from the intrinsic camera parameters, respectively [3]. Z_i is the unknown depth. Its 3D location $\mathbf{P}_i^n$ in $\mathbf{O}_{\text{world}}$ can be further achieved with rotation matrix $\mathbf{R}_i \in \mathbb{R}^{3\times3}$ and translation vector $\mathbf{T}_i \in \mathbb{R}^3$ by [3]:

$$
\mathbf{P}_i^n = \mathbf{R}_i {\mathbf{P}_{\mathbf{c}}}_i^n + \mathbf{T}_i \tag{2.3}
$$

As shown in Figure 2.4, the ray $\overrightarrow{\mathbf{C}_i\mathbf{P}_i^n}$ (gray line with arrow) indicates the direction of the pixel p_i in the world. The unit vector $\widehat{\overrightarrow{\mathbf{C}_i\mathbf{P}_i^n}}$ is its direction vector, and can encode the relative 3D location of each pixel. Thus, we design GPE based on this unit vector, and we add one linear transformation to make it fit the input dimension d. The 3D geometry positional encoding for the n-th pixel in view i is defined as:

$$
{\mathbf{E}_{\mathbf{G}}}_i^n = \mathbf{W}_e \widehat{\overrightarrow{\mathbf{C}_i\mathbf{P}_i^n}} \in \mathbb{R}^d \tag{2.4}
$$

Where $\mathbf{W}_e \in \mathbb{R}^{d\times3}$ is a learnable transformation matrix. With $\mathbf{E}_{\mathbf{G}i}$, the transformer can be aware of the 3D location of each view.

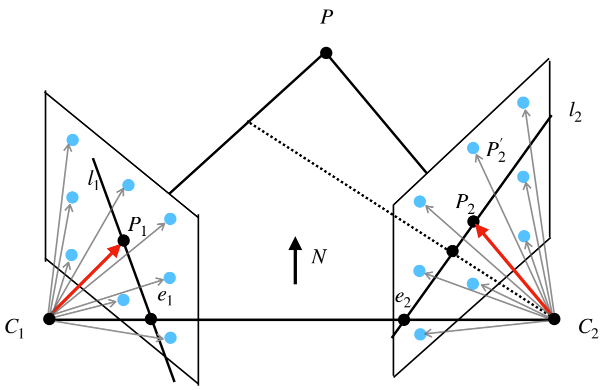


Figure 2.4: Illustration of Geometry Positional Encoding.

2.4.4 Epipolar Field

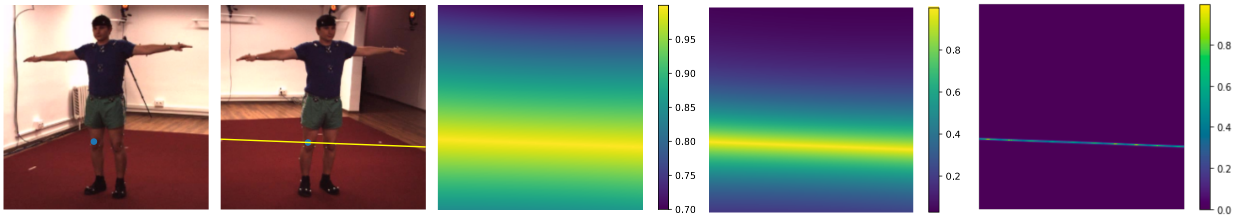
Although GPE impose the 3D space information into transformers, it does not explicitly encode the relationship between two views. As a result, given a pixel in the current view, it is still difficult to attend the corresponding regions when performing global attention between features of two views. We further impose the *Epipolar Constraints* [3] into GPE: Given one pixel p_1 in view 1, its correspondence pixel p_2 in view 2 must be on the epipolar line l_2 (Figure 2.4). However, the epipolar line does not model the relationship with pixels outside l_2. Instead, pixels close to the line and pixels away from the line should be treated differently. Thus, we propose the *Epipolar Field* to model the relationship among all pixels in the reference view. In detail, given $\mathbf{P}_1^n$, the 3D location of n-th pixel p_1^n in view 1, we calculate the normal vector $\mathbf{N}_{\mathbf{P}_1^n\mathbf{C}_1\mathbf{C}_2}$ of plane $\mathbf{P}_1^n\mathbf{C}_1\mathbf{C}_2$ by:

$$\mathbf{N}_{\mathbf{P}_1^n\mathbf{C}_1\mathbf{C}_2} = \widehat{\overrightarrow{\mathbf{C}_1\mathbf{C}_2}} \times \widehat{\overrightarrow{\mathbf{C}_1\mathbf{P}_1^n}} \tag{2.5}$$

Given $\mathbf{P}_2^m$, the 3D location of m-th pixel p_2^m in view 2, we use the angle θ between the normal vector $\mathbf{N}_{\mathbf{P}_1^n\mathbf{C}_1\mathbf{C}_2}$ and ray $\overrightarrow{\mathbf{C}_2\mathbf{P}_2^m}$ to model the relationship between p_1^n and p_2^m, and use the cosine of θ to calculate the correspondence score:

$$\mathbf{S}(p_1^n, p_2^m) = 1- \mid \cos\theta \mid = 1- \mid \mathbf{N}_{\mathbf{P}_1^n\mathbf{C}_1\mathbf{C}_2} \cdot \widehat{\overrightarrow{\mathbf{C}_2\mathbf{P}_2^m}} \mid \tag{2.6}$$

The absolute $|\cdot|$ is added to limit the score in $[0, 1]$. With Eq. 2.6, if p_2^m falls in the epipolar line l_2, the score $\mathbf{S}(p_1^n, p_2^m)$ will be 1. Otherwise, the far p_2^m is from l_2, the closer the score would be 0. We further add a soft factor γ to control the sharpness, thus the epipolar field is $\mathbf{S}'(p_1^n, p_2^m) = (\mathbf{S}(p_1^n, p_2^m))^\gamma$. Figure 2.5 gives a visualization of the epipolar field. Comparing with the epipolar line, the epipolar field model relationships with all pixels in the reference view. We can also reduce it to the epipolar line with a very large γ. Thus, epipolar field can be considered as a more general form of the epipolar line.



Query Pixel Epipolar Line Epipolar Field (γ = 1) Epipolar Field (γ = 10) Epipolar Field (γ = 1000)

Figure 2.5: Illustration of Epipolar Field. The epipolar field can reflect the distance from the epipolar line to the off-line pixels. By adjusting the soft factor γ, it can also reduce to the standard epipolar line.

We then use the epipolar field to guide the learning of $\mathbf{W}_e$ to help the $\mathbf{E}_{\mathbf{G}}{}_i^n$ encode correspondence between two views. In detail, we let the dot product of $\mathbf{E}_{\mathbf{G}}{}_1^n$ and $\mathbf{E}_{\mathbf{G}}{}_2^m$ match $\mathbf{S}'(p_1^n, p_2^m)$ with the mean square error loss during the training process:

$$L_{\text{pos}} = \frac{1}{L^2} \sum_n^L \sum_m^L (\mathbf{E}_{\mathbf{G}}{}_1^n (\mathbf{E}_{\mathbf{G}}{}_2^m)^T - \mathbf{S}'(p_1^n, p_2^m))^2 \tag{2.7}$$

Therefore, a high attention score will be achieved along the epipolar line when calculating the cross-view attention maps between $\mathbf{X}_1' + \mathbf{E}_{\mathbf{2D}} + \mathbf{E}_{\mathbf{G}1}$ and $\mathbf{X}_2' + \mathbf{E}_{\mathbf{2D}} + \mathbf{E}_{\mathbf{G}2}$, which makes the transformer easy to attend corresponding regions. Moreover, with this soft design, semantic information from offline pixels are still kept, rather than discarded like [61].

2.4.5 Implementation Details

CNN backbone We follow [191] and apply a very shallow CNN architecture as the CNN backbone $\mathcal{F}(\cdot)$, which is the initial part of the ResNet-50 [59]. Specifically, the number of parameters of the shallow CNN is 1.4 M, which is just 5.5% of the original Simple Baseline with ResNet-50 (25.6M). The output feature map has size $H = H_I/8$, $W = W_I/8$. Thus, the fine-grained local feature information can still be kept.

TransFusion Following [14, 191], we set the dimension of the feature embedding d to 256, the number of heads to 8, the number of encoder layers N to 3. Due to the limitation of resource, we only consider the fusion of 2 neighborhood views, although out framework can be easily extended to more than 2 views.

Prediction head Given $\tilde{\mathbf{X}}_i$, we reshape it back to $\tilde{\mathbf{X}}'_i \in \mathbb{R}^{d\times H\times W}$. The prediction head $\mathcal{H}(\cdot)$ applies one deconvolution layer and one 1×1 convolution layer to predict the heatmap of keypoints. By default, the height and width of heatmaps $\mathbf{H}_i$ are $H_h = H_I/4$ and $W_h = W_I/4$.

Loss function The groundtruth heatmap $\mathbf{H}_i \in \mathbb{R}^{k\times H_h\times W_h}$ of 2D keypoints is defined as a 2D a Gaussian centering around each keypoint [177]. We apply the Mean Square Error loss to calculate the difference between the output heatmaps $\bar{\mathbf{H}}_i$ and $\mathbf{H}_i$. By combining the Equation 2.7, we train the network end-to-end with loss function $L = \frac{1}{HW} \parallel \bar{\mathbf{H}}_i - \mathbf{H}_i \parallel_F^2 + L_{\text{pos}}$.

2.5 Experiments

2.5.1 Experimental Settings

Dataset We conduct extensive experiments on two public multi-view 3D human pose estimation datasets, Human 3.6M [73, 16] and Ski-Pose [135]. (1) The Human 3.6M contains joint annotations of video frames captured by four calibrated cameras in a room. We adopt the same training and test split as in [129, 74, 61], where subjects 1, 5, 6, 7, 8 are used for training, and 9, 11 are for testing. Note that 3D annotations of some scenes of the 'S9' are damaged [74], we exclude these scenes from the evaluation as in [74, 61]. (2) The Ski-Pose dataset aims to help analyze skiers's giant slalom runs with 6 calibrated cameras. It provides six camera views as well as corresponding 3D pose. In detail, 8,481 frames are used for training and $1,716$ are used for testing. We resize all images to 256×256 in all experiments.

Training As the training of transformers requires huge datasets [165, 37], while the scenes of multi-view pose datasets are quite limited, making it difficult to train the transformer from scratch. By convention [74, 61], we use the MS-COCO [101] pretrained TransPose [191] to initialize our network and fine tune it on the multi-view human pose datasets. Following the settings in [61], we apply Adam optimizer [85] and train the model for 20 epochs. The learning rate is initialized with 0.001 and decays at 10-th and 15-th epoch with ratio 0.1.

Evaluation metrics The performance of 2D pose estimation is evaluated by Joint Detection Rate (JDR), which measures the percentage of the successfully detected keypoints. A keypoints is detected if the distance between the predicted location and the ground truth is within a predefined threshold. The threshold is set to half of the head size for human pose estimation. Given the estimated 2D joints of each view, following [129, 61], direct triangulation is used for estimating the 3D poses with respect to the global coordinates. The 3D pose estimation accuracy is measured by Mean Per Joint Position Error (MPJPE) between the groundtruth 3D pose and the estimated 3D pose.

2.5.2 Results on Human 3.6M

We compare with two state-of-the-art methods, the crossview fusion [129] and the epipolar transformers [61]. For fair comparison, we use the SimpleBaseline-ResNet50 pretrained on COCO [181] as initialization and then finetuned with their official codes [129, 61].

Quantitative results The results of both 2D and 3D pose estimation are shown in Table 2.1. We also shown the number of parameters of each model, the MACs (multiply-add operations). Besides, we also report the inference time to obtain the 3D pose from 4 views on a single 2080Ti GPU of all multiview methods. For both 2D and 3D pose estimation, TransFusion consistently outperforms or achieves comparable performance with epipolar transformers [61] and cross-view fusion [129]. Note that JDR is a relative loose metric, with a wider threshold which tolerates small errors, so the improvement on 2D is not very obvious. However, on the 3D metric, which directly computes the distance, our improvement is much more significant. Moreover, as in Table 2.2, our method can achieve significant improvement on sophisticated poses sequences such as ”Phone” and ”Smoke”, which usually encounters heave occlusions for certain views. This result suggests that fusing features from the entire images of other views, instead of just features along the epipolar line [61], can bring more benefits. Besides, comparing to the single view TransPose [191], our Transfusion can achieve 4.7 mm gain on 3D. Thus, the improvement is not only from the TransPose architecture, but from the fusion with other views. Moreover, our method is lightweight and efficient. It only requires 2.1% (5M / 235M) of the parameters of cross-view fusion [129]. Benefit from the parallel computing of transformers architectures, it further reduces the inference time, while the operation of sampling along epipolar lines [61] is time-consuming.

Method	Params	MACs	Inference Time (s)	JDR (%) ↑	MPJPE (mm) ↓
Single view - Simple Baseline[181]	34M	51.7G	-	98.5	30.2
Single view - TransPose [191]	5M	43.6G	-	98.6	30.5
Crossview Fusion [129]	235M	55.1G	0.048	**99.4**	27.8
Epipolar Transformer [61]	34M	51.7G	0.086	98.6	27.1
TransFusion	**5M**	50.2G	**0.032**	**99.4**	**25.8**

Table 2.1: 2D and 3D pose estimation accuracy comparison on Human3.6M. The metric of 2D pose is JDR (%), and the metric of 3D pose is MPJPE (mm). All networks are pretrained on COCO [101] and then finetuned on Human 3.6M [73]. All images are resized to 256 × 256.

Method	Dir	Disc	Eat	Greet	Phone	Pose	Purch	Sit	SitD	Smoke	Photo	Wait	WalkD	Walk	WalkT
Crossview Fusion[129]	24.0	28.8	25.6	24.5	28.3	24.4	26.9	**30.7**	34.4	29.0	32.6	25.1	24.3	30.8	24.9
Epipolar transformers [61]	**23.2**	27.1	23.4	22.4	32.4	**21.4**	**22.6**	37.3	35.4	29.0	27.7	24.2	**21.2**	26.6	**22.3**
TransFusion	24.4	**26.4**	**23.4**	**21.1**	**25.2**	23.2	24.7	33.8	**29.8**	**26.4**	**26.8**	**24.2**	23.2	**26.1**	23.3

Table 2.2: The MPJPE of each pose sequence on Human 3.6M.

Visualization of Attention maps Given the query pixel in one view, we further visualize the attention maps on both views. We show our results in Figure 2.6. It is observed that on the view itself, typically the attention map is around the joints. If the query joint is occluded, it may resort to joints on the other side of the symmetry [191]. On the neighboring view, the network usually not just attends the corresponding keypoint, but attends the whole limbs, which cannot be located by the epipolar line. Previous methods based on epipoar line [61] actually miss this important clue.

Qualitative results We also present examples of predicted 2D keypoints on the image and 3D pose in the space, and compare our methods with baseline methods [129]. As in Figure 2.7, even if the entire arms (green line) are occluded, our method still predicts the 2D keypoints correctly by fusing information from the reference view, and further gives a better 3D pose.

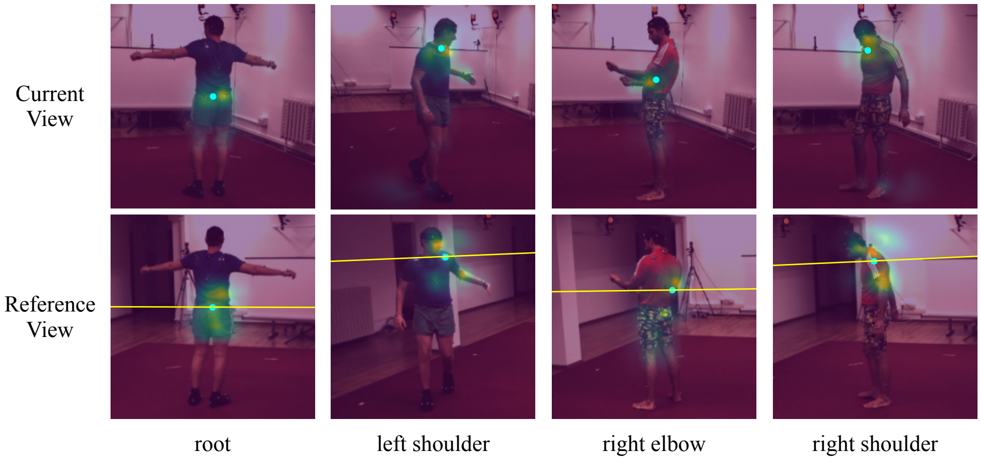


Figure 2.6: Visualization of attention maps on Human 3.6M test set. The cyan dots are groundtruth. Given the query pixel, the first rows are attention maps of current views, and the second rows are attention maps of reference views. We also visualize the epipolar line (yellow) for comparison.

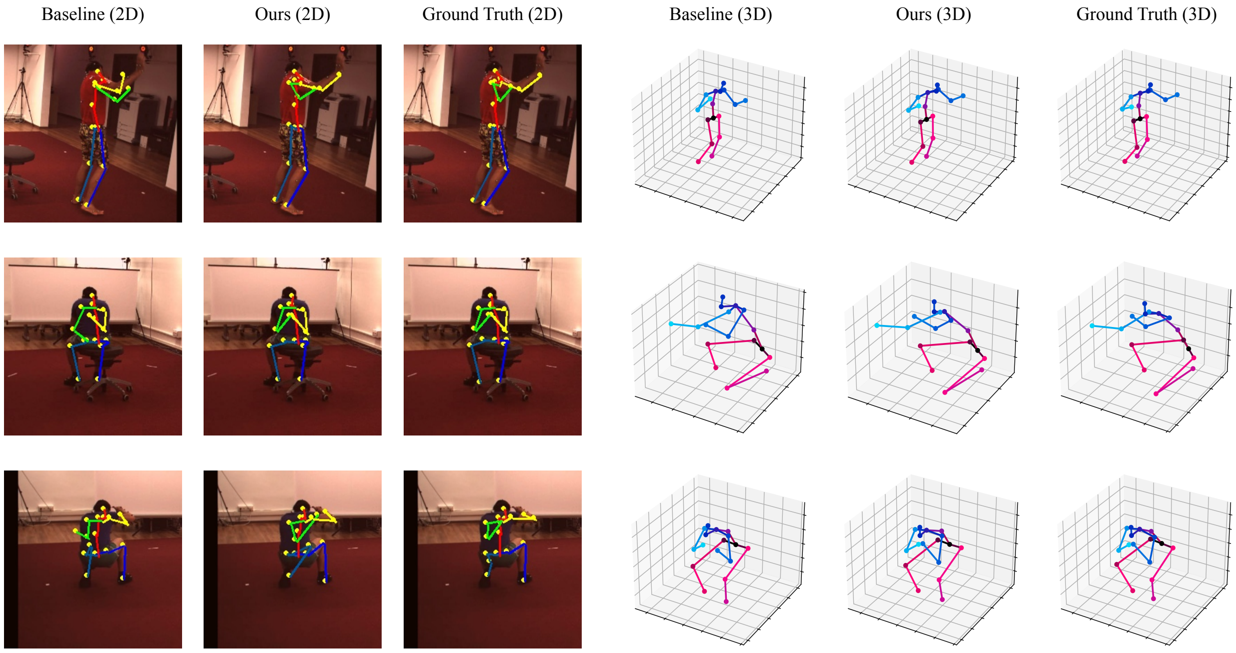


Figure 2.7: Visualization of predictions for Human 3.6M. Both skeletons of 2D keypoints on the image and 3D pose in the space are presented.

2.5.3 Ablation Studies

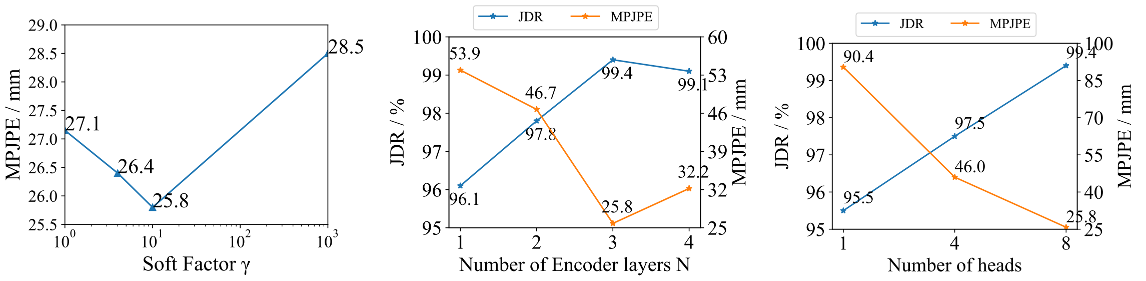
Geometry Positional Encoding We conduct ablation studies on the GPE to verify its significance. In detail, we consider 3 settings: 1) training without 3D geometry positional encoding, 2) applying a learnable 3D positional encoding, *i.e.*, directly learn $\mathbf{E}_{\mathbf{G}i}$ from scratch 3) training the 3D GPE without epipolar field constraints L_{pos}. Table 2.3 presents the results. Without the 3D location information, the performance of 1) and 2) are even worse than the single view TransPose, we hypothesize that the 2D sine PE makes the transformer easy to attend the same pixel location of all views, and the learned 3D PE is easy to overfit the training examples. Without L_{pos}, the error will also increase. Thus the guide from the epipolar field is favorable, as it imposes correspondence for cross-view attention.

Method	2D Pose / JDR (%) ↑	3D Pose / MPJPE (mm) ↓
TransFusion - without 3D positional encoding	98.5	35.9
TransFusion - learnable 3D positional encoding	96.0	57.3
TransFusion - GPE without L_{pos}	99.3	26.8
TransFusion	**99.4**	**25.8**

Table 2.3: Ablation studies on different types of 3D positional encoding

Soft Factor γ We also try different values of the soft vector γ, results are shown in Figure 2.8a. With a small γ, the epipolar field assign all locations with relative high probabilities, the performance are slightly worse (1.3 mm drop). While with a huge $\gamma = 1000$, the epipolar field reduces to the hard epipolar line, and the performance drops 2.7 mm. Thus, we verify the effectiveness of our epipolar field compared with hard-coded epipolar line.

Transformer architecture We study how performance scales with the size of the transformer. As in Figure 2.8b, with the number of layers N increasing, the performance improves significantly, as the learning ability of transformer is more powerful with more parameters. But when $N > 3$, it tends to saturate or degenerate. We hypothesize that the transformer is easy to overfit when the size is too huge. Meanwhile, as in Figure 2.8c, with the number of heads increases, the performance also improves gradually, as more heads can help attend different features [165]. In summary, our choice with $N = 3$ and 8 heas are reasonable.



(a) (b) (c)

Figure 2.8: Ablation studies on the soft factor γ of the epipolar field, the number of transformer encoder layers N, and the number of transformer heads.

Method	2D Pose / JDR (%) ↑	3D Pose / MPJPE (mm) ↓
Single view - Simple Baseline [181]	94.5	39.6
Epipolar Transformer [61]	94.9	34.2
TransFusion	**96.0**	**31.6**

Table 2.4: 2D and 3D pose estimation accuracy comparison on Ski-Pose.

2.5.4 Results on Ski-Pose Dataset

We further apply TransFusion on the Ski-Pose dataset to verify its generalization ability. Results are presented in Table 2.4. In the settings with six cameras, the Crossview Fusion is too huge (537M) to train on the 2080Ti GPU. Similar to Human 3.6M, TransFusion still outperform or achieve comparable performance with other fusion methods, while it is much lightweight. Thus, our method is also effective in outdoor multi-view settings.

2.6 Conclusion

We apply the transformer to the multi-view 3D human pose estimation for the first time. Inspired by multi-modal transformers, we propose the TransFusion network, a lightweight architecture to integrate cues from both self views and reference views. Furthermore, we propose the epipolar field, and apply it to the 3D positional encoding to encode correspondence between two views explicitly. Experimental results shows that our method outperform previous fusion methods but with a more light weighted network. In the future we plan to apply our TransFusion to regress the 3D locations with multi-view inputs in an end-to-end way to further improve 3D predictions.

Chapter 3

PPT: token-Pruned Pose Transformer for monocular and multi-view human pose estimation

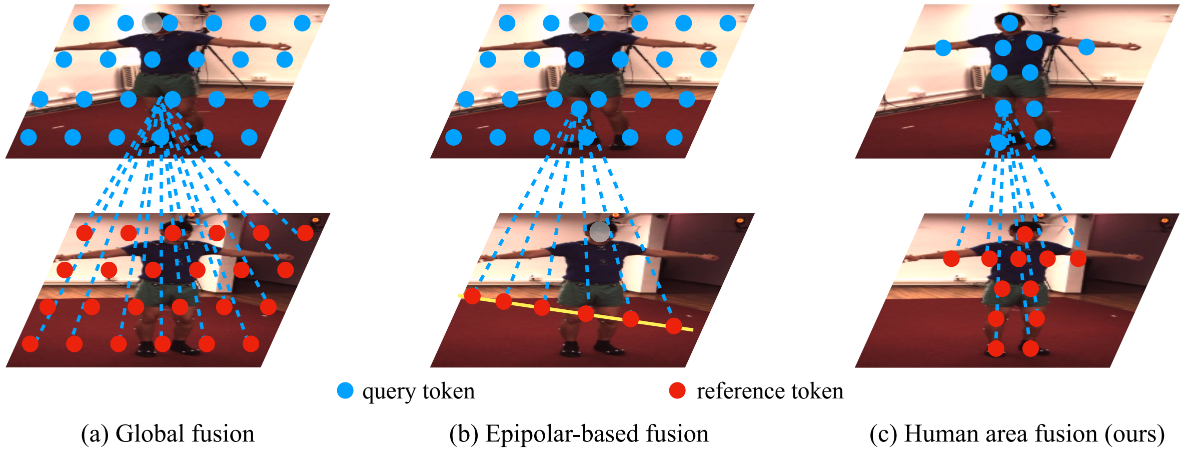
In the last chapter, we have seen that the vision transformer and its variants have played an increasingly important role in both monocular and multi-view human pose estimation. Considering image patches as tokens, transformers can model the global dependencies within the entire image or across images from other views. However, global attention is computationally expensive. As a consequence, it is difficult to scale up these transformer-based methods to high-resolution features and many views.

To this end, we propose the token-Pruned Pose Transformer (PPT) for 2D human pose estimation, which can locate a rough human mask and performs self-attention only within selected tokens. Furthermore, we extend our PPT to multi-view human pose estimation. Built upon PPT, we propose a new cross-view fusion strategy, called human area fusion, which considers all human foreground pixels as corresponding candidates. Experimental results on COCO and MPII demonstrate that our PPT can match the accuracy of previous pose transformer methods while reducing the computation. Moreover, experiments on Human 3.6M and Ski-Pose demonstrate that our Multi-view PPT can efficiently fuse cues from multiple views and achieve new state-of-the-art results.

3.1 Introduction

Human pose estimation aims to localize anatomical keypoints from images. It serves as a foundation for many down-stream tasks such as AR/VR, action recognition [70, 186], and medical diagnosis [29]. Over the past decades, deep convolutional neural networks (CNNs) play a dominant role in human pose estimation tasks [162, 177, 119, 181, 152, 176, 175]. However, cases including occlusions and oblique viewing are still too difficult to be solved from a monocular image. To this end, some works apply a multi-camera setup [145, 174, 73, 23] to boost the performance of 2D pose detection[129, 61], since difficult cases in one view are potentially easier to be resolved in other views. Meanwhile, human body joints are highly correlated, constrained by strong kinetic and physical constraints [161]. However,since the reception fields of CNNs are limited, the long-range constraints among joints are often poorly captured [97].

Recently, the ViT [37] demonstrates that the transformers [165] can achieve impressive performance on many vision tasks [163, 14]. Compared with CNN, the self-attention module of transformers can easily model the global dependencies among all visual elements. In the field of pose estimation, many tansformer-based works [97, 191, 99, 217] suggest that the global attention is necessary. In single-view 2D human pose estimation, TransPose [191] and TokenPose [97] achieve new state-of-the-art performance and learn the relationship among keypoints with transformers. In multi-view human pose estimation, the TransFusion [106] uses the transformer to fuse cues from both current and reference views. Typically, these



(a) Global fusion (b) Epipolar-based fusion (c) Human area fusion (ours)

Figure 3.1: Different types of cross-view fusion. The first row is the current view, and the second row is the reference view.

works flatten the feature maps into 1D token sequences, which are then fed into the transformer. In multi-view settings, tokens from all views are usually concatenated together to yield a long sequence. However, the dense global attention of transformers is computationally extensive. As a result, it is challenging to scale up these methods to high-resolution feature maps and many views. For example, the TransFusion [106] can only compute global attention between two views due to the large memory cost. Meanwhile, as empirically shown in Fig.3.2, the attention map of keypoints is very sparse, which only focuses on the body or the joint area. This is because the constraints among human keypoints tend to be adjacent and symmetric [97]. This observation also suggests that the dense attention among all locations in the image is relatively extravagant.

In this work, we propose a compromised and yet efficient alternative to the global attention in pose estimation, named token-Pruned Pose Transformer (PPT). We calculate attention only within the human body area, rather than over the entire input image. Specifically, we select human body tokens and prune background tokens with the help of attention maps. As the human body only takes a small area of the entire image, the majority of input tokens can be pruned. We reveal that pruning these less informative tokens does not hurt the pose estimation accuracy, but can accelerate the entire networks. Interestingly, as a by-product,

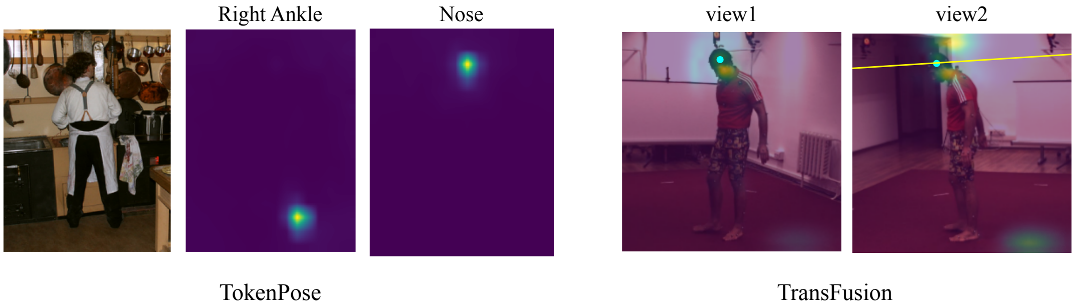


Figure 3.2: Attention map for TokenPose (monocular view) and TransFusion (multi-view). The attention maps are very sparse and only attend to a small local regions.

PPT can also predict a rough human mask without the guidance of ground truth mask annotations.

Moreover, we extend PPT to multi-view settings. As in Fig.3.1, previous cross-view fusion methods consider all pixels in the reference view (global fusion) or pixels along the epipolar line (epipolar-based fusion) as candidates. The former is computationally extensive and inevitably introduces noise from the background, and the latter requires accurate calibration and lacks semantic information. Built upon PPT, we propose a new fusion strategy, called *human area fusion*, which considers human foreground pixels as corresponding candidates. Specifically, we firstly use PPT to locate the human body tokens on each view, and then perform the multi-view fusion among these selected tokens with transformers. Thus, our method is an efficient fusion strategy and can easily be extended to many views.

Our main contributions are summarized as follows:

1. We propose the token-Pruned Pose Transformer (PPT) for efficient 2D human pose estimation, which can locate the human body area and prune background tokens with the help of a Human Token Identification module.

2. We propose the strategy of "Human area fusion" for multi-view pose estimation. Built upon PPT, the multi-view PPT can efficiently fuse cues from human areas of multiple views.

3. Experimental results on COCO and MPII demonstrate that our PPT can maintain the pose estimation accuracy while significantly reduce the computational cost. Results on Human 3.6M and Ski-Pose show that human area fusion outperforms previous fusion methods on 2D and 3D metrics.

3.2 Related Work

3.2.1 Efficient Vision Transformers

Recently, the transformer [165] achieves great progresses on many computer vision tasks, such as classification [37, 163], object detection [14, 221, 44], and semantic segmentation [218, 173, 187, 197]. While being promising in accuracy, the vanilla ViT [37] is cumbersome and computationally intensive. Therefore, many algorithms have been proposed to improve the efficiency of vision transformers. Recent works demonstrate that some popular model compression methods such as network pruning [57, 25, 26, 202], knowledge distillation [64, 163, 27], and quantization [141, 153] can be applied to ViTs. Besides, other methods introduce CNN properties such as hierarchy and locality into the transformers to alleviate the burden of computing global attention [102, 21]. On the other hand, some works accelerate the model by slimming the input tokens [203, 15, 137, 132, 89, 98, 114]. Specifically, the Token-to-tokens [203] aims to reduce the number of tokens by aggregating neighboring tokens into one token. The TokenLearner [137] mines important tokens by learnable attention weights conditioned on the input feature. The DynamicViT [132] prunes less informative tokens with an extra learned token selector. The EViT [98] reduces and reorganizes image tokens based on the classification token. However, all these models have only been designed for classification, where the final prediction only depends on the special classification token.

3.2.2 Human Pose Estimation

Monocular 2D Pose Estimation In the past few years, many successful CNNs are proposed in 2D human pose estimation. They usually capture both low-level and high-level representations [177, 30, 119, 32, 181, 152], or use the structural of skeletons to capture the spatial constraints among joints [161, 80, 122, 86, 87, 28, 88]. Recently, many works introduce transformers into pose estimation tasks [191, 97, 94, 99, 217]. Specifically, TransPose [191] utilizes transformers to explain dependencies of keypoint predictions. TokenPose [97] applies additional keypoint tokens to learn constraint relationships and appearance cues. Both works demonstrate the necessity of global attention in pose estimation.

Efficient 2D Pose Estimation Some recent works also explore efficient architecture design for real-time pose estimation [121, 118, 142, 172, 210, 199]. For example, EfficientPose [210] designs an efficient backbone with neural architecture search. Lite-HRNet [199] proposes the conditional channel weighting unit to replace the heavy shuffle blocks of HRNet. However, these works all focus on CNN-based networks, and none of them study transformer-based networks.

Multi-view Pose Estimation 3D pose estimation from multiple views usually takes two steps: predicting 2D joints on each view separately with a 2D pose detector, and lifting 2D joints to 3D space via triangulation. Recently, many methods focus on enabling the 2D pose detector to fuse information from other views [129, 214, 182, 61, 106]. They can be categorized into two groups: 1) Epipolar-based fusion. The features of one pixel in one view is augmented by fusing features along the corresponding epipolar line of other views. Specifically, the AdaFuse [214] adds the largest response on the heatmap along the epipolar line. The epipolar transformer [61] applies the non-local module [170] on intermediate features to obtain the fusion weights. However, this fusion strategy requires precise camera calibration and discard information outside the epipolar lines. 2) Global fusion. The features of one pixel in one view are augmented by fusing features of all locations in other views. In detail, the Cross-view Fusion [129] learns a fixed attention matrix to fuse heatmaps in all other views. The TransFusion [106] applies the transformers to fuse features of the reference views and demonstrates that global attention is necessary. However, the computation complexity of global fusion is quadratic to the resolution of input images and number of views. Thus, both categories have their limitations. A fusion algorithm that can overcome these drawbacks and maintains their advantages is in need.

3.3 Methodology

3.3.1 Token-Pruned Pose Transformer

Overview Fig.3.3 is an overview of our token-Pruned Pose Transformer. Following [97], the input RGB image $\mathbf{I}$ first go through a shallow CNN backbone $\mathcal{B}(\cdot)$ to obtain the feature map $\mathbf{F} \in \mathbb{R}^{H \times W \times C}$. Then $\mathbf{F}$ is decomposed into flattened image patches $\mathbf{F}_p \in \mathbb{R}^{N_v \times (C \cdot P_h \cdot P_w)}$, where (P_h, P_w) is the resolution of each image patch, and $N_v = \frac{H}{P_h} \cdot \frac{W}{P_w}$ is the total number of patches [37]. Then a linear projection is applied to project $\mathbf{F}_p$ into $\mathbf{X}_p \in \mathbb{R}^{N_v \times D}$, where D is the dimension of hidden embeddings. The 2D positional encodings $\mathbf{E} \in \mathbb{R}^{N_v \times D}$ are added to make the transformer aware of position information [165], *i.e.*, $\mathbf{X}_v = \mathbf{X}_p + \mathbf{E}$, namely the visual token. Meanwhile, following TokenPose [97], we have J additional learnable keypoint tokens $\mathbf{X}_k \in \mathbb{R}^{J \times D}$ to represent J target keypoints. The input sequence to the transformer is $\mathbf{X}^0 = [\mathbf{X}_k, \mathbf{X}_v] \in \mathbb{R}^{N \times D}$, where $N = N_v + J$ and $[\ldots]$ is the concatenation operation.

The transformer has L encoder layers in total. At the L_1^{th} layer, the Human Token Identification (HTI) module locates K most informative visual tokens where human body appears and prunes the remaining tokens. We denote $r = \frac{K}{N_v}(0 < r < 1)$ as the keep ratio. As a result,

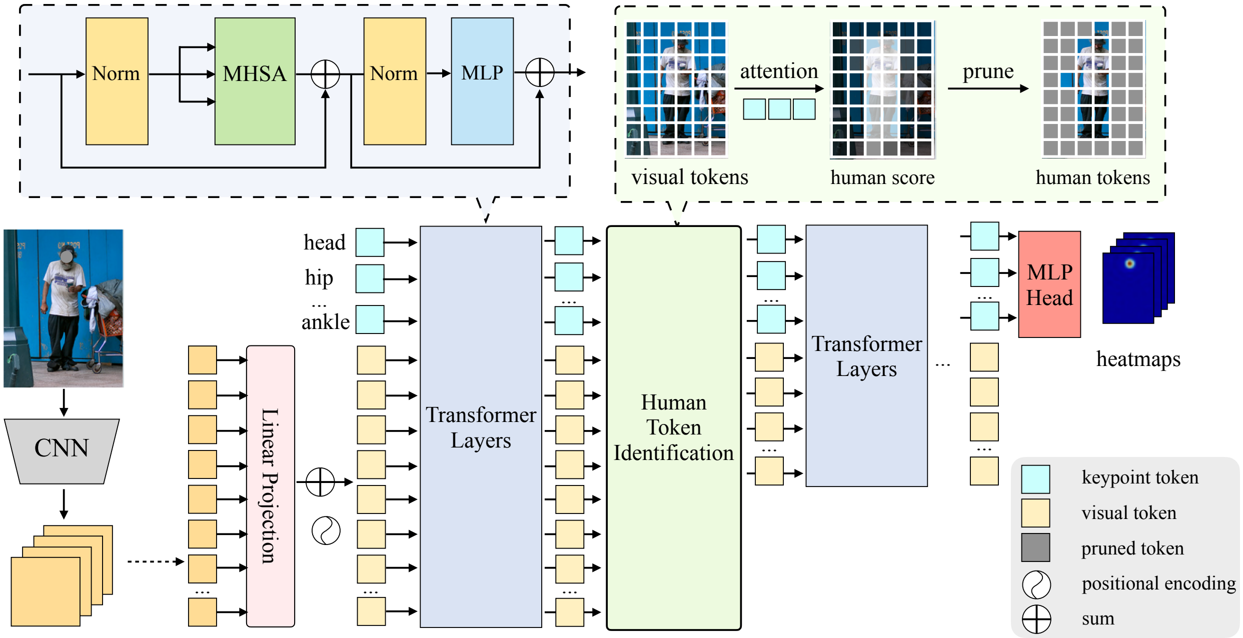


Figure 3.3: Framework of the token-Pruned Pose Transformer (PPT). The visual tokens are obtained from the flattened CNN feature maps. The keypoint tokens are added to represent each joint and predict the keypoints heatmaps. The Human Token Identification (HTI) module is inserted inside the transformer layers to locate human visual tokens and prune background tokens. Thus the followed transformer layers are only performed on these selected tokens.

the length of the sequence is reduced to $N' = rN_v + J$ for the following transformer layers. The HTI is conducted e times at the $L_1^{th}, L_2^{th}, \ldots, L_e^{th}$ layers. Thus, PPT can progressively reduce the length of visual tokens. Finally, the total number of tokens is $r^e N_v + J$. The prediction head projects the keypoint tokens in the last layer $\mathbf{X}_k^L \in \mathbb{R}^{J \times D}$ into the output heatmaps $\mathbf{H} \in \mathbb{R}^{J \times (H_h \cdot W_h)}$.

Transformer Encoder Layer. The encoder layer consists of the multi-headed self-attention (MHSA) and multi-layer perceptron (MLP). Operations in one encoder layer is shown in Fig. 3.3. The self-attention aims to match a query and a set of key-value pairs to an output [165]. Given the input $\mathbf{X}$, three linear projections are applied to transfer $\mathbf{X}$ into three matrices of equal size, namely the query $\mathbf{Q}$, the key $\mathbf{K}$, and the value $\mathbf{V}$. The self-attention (SA) operation is calculated by:

$$\mathrm{SA}(\mathbf{X}) = \mathrm{Softmax}(\frac{\mathbf{QK}^T}{\sqrt{D}})\mathbf{V}, \tag{3.1}$$

For MHSA, H self-attention modules are applied to $\mathbf{X}$ separately, and each of them produces an output sequence.

Human Token Identification (HTI). The TokenPose [97] conducts self-attention among all visual tokens, which is cumbersome and inefficient. From Equation 3.1, we know that each keypoint token $\mathbf{X}_k^j$ interacts with all visual tokens $\mathbf{X}_v$ via the attention mechanism:

$$\mathrm{Softmax}(\frac{\mathbf{q}_k^j\mathbf{K}_v^T}{\sqrt{D}})\mathbf{V}_v = \mathbf{a}^j\mathbf{V}_v, \tag{3.2}$$

where $\mathbf{q}_k^j$ denotes the query vector of $\mathbf{X}_k^j$, $\mathbf{K}_v$ and $\mathbf{V}_v$ are the keys and values of visual tokens $\mathbf{X}_v$. To this end, each keypoint token is a linear combination of all value vectors of visual tokens. The combination coefficients $\mathbf{a}^j \in \mathbb{R}^{N_v}$ are the attention values from the query vector for that keypoint token with respect to all visual tokens. To put it differently, the attention value determines how much information of each visual token is fused into the output. Thus, it is natural to assume that the attention value $\mathbf{a}^j$ indicates the importance of each visual token in the keypoint prediction [98]. Typically, a large attention value suggests that the target joint is inside or nearby the corresponded visual token.

With this assumption, we propose the Human Token Identification module to select informative visual tokens with the help of attention scores of keypoint tokens. However, each keypoint token usually only attends to a few visual tokens around the target keypoint. And some keypoint tokens (such as the eye and the nose) may attend to close-by or even the same visual tokens. Thus, it is difficult to treat the attention values of each keypoint separately. For simplicity, as all human keypoints make up a rough human body area, we use $\mathbf{a} = \sum_j \mathbf{a}^j$

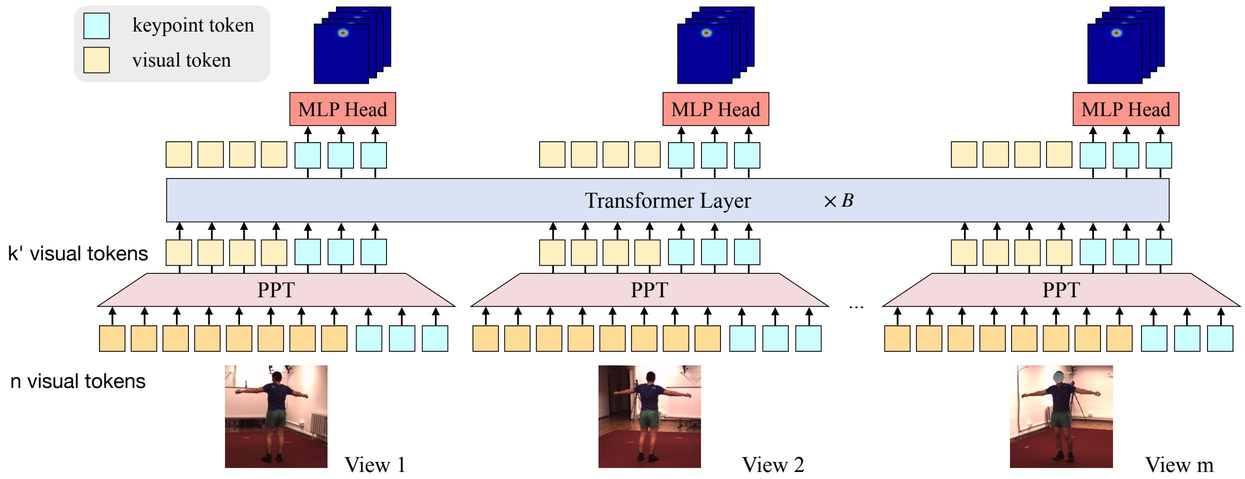


Figure 3.4: Overall framework of the Multi-view PPT. A share-weight PPT is applied to extract a subset of visual tokens for each view. Then B transformer layers are applied to the concatenated tokens from each view to perform cross-view fusion. The output head takes keypoint tokens in each view to predict heatmaps.

as the criterion to select visual tokens, which is the summation of all joints' attention maps. In detail, we keep visual tokens with the K largest corresponding values in $\mathbf{a}$ as the human tokens, and prune the remaining tokens. As a result, only K visual tokens and J keypoint tokens are sent to the following layers.

3.3.2 Multi-view Pose Estimation with PPT

Human Area Fusion. We propose the concept of *Human area fusion* for cross-view fusion in multi-view pose estimation, which considers pixels where human appears as corresponding candidates. Suppose there are m cameras, and each view maintains n pixels (tokens) in its feature map. We summarize three typical types of cross-view fusion strategies in Fig.3.1. 1) For global fusion, each pixel in each view calculates attention with respect to all n pixels in feature maps of other $m-1$ views. Thus the computational complexity is $\mathcal{O}(m^2n^2)$. 2) For epipolar-based fusion, each pixel in each view calculates attention with $k(k \ll n)$ pixels along the corresponded epipolar lines of other $m-1$ views. Thus the computational complexity is $\mathcal{O}(m^2nk)$. 3) For our human area fusion, we firstly select

k' human foreground pixels in each view. Then we perform dense attention among these foreground tokens. As we also reduce the number of query pixels, the computational complexity is $\mathcal{O}(m^2 k'^2)$. Typically, $k < k' \ll n$. Thus, our method is an efficient way to perform cross-view fusion. Moreover, it also avoids the useless or even disturbing information from the background tokens and thus makes the model focus on the constraints within the human body.

Multi-view PPT. Naturally, we can apply an off-the-shelf segmentation network [58] to obtain human foreground pixels and then perform human area fusion. However, a large amount of densely annotated images are required to train a segmentation model. To this end, we utilize PPT to efficiently locate a rough human foreground area without any mask labels, and further propose the *multi-view PPT* for multi-view pose estimation. Specifically, we design our network in a two-stage paradigm, as shown in Fig.3.4. Given the image $\mathbf{I}^m$ in each view, the share-weight PPT firstly produces selected human tokens $\tilde{\mathbf{X}}_v^m$ and keypoint tokens $\mathbf{X}_k^m$. Then we concatenate tokens from all views together and perform the dense attention among them with B transformer encoder layers. To help the network perceive the 3D space information, we also add the 3D positional encodings [106] on all selected visual tokens. Thus, each keypoint token can fuse visual information from all views. Moreover, it can learn correspondence constraints between keypoints both in the same view and among different views. Finally, a share-weight MLP head is placed on top of the keypoint token of each view to predicts keypoint heatmaps.

3.4 Experiments on monocular image

3.4.1 Settings

Datasets & Evaluation Metrics. We firstly evaluate PPT on monocular 2D human pose estimation benchmarks. COCO [101] contains $200K$ images in the wild and $250K$ human instances with 17 keypoints. Following top-down methods [181, 152, 97], we crop human instances with the ground truth bounding boxes for training and with the bounding boxes provided by SimpleBaseline [181] for inference. The evaluation is based on object keypoint similarity, which measures the distance between the detected keypoint and the corresponding ground truth. The standard average precision (AP) and recall (AR) scores are reported. MPII [4] contains about $25K$ images and $40K$ human instances with 16 keypoints. The evaluation is based on the head-normalized probability of correct keypoint (PCKh) score [4]. A keypoint is correct if it falls within a predefined threshold to the groundtruth location. We report the PCKh@0.5 score by convention.

Implementation Details. For fair comparison, we build our PPT based upon TokenPose-S, TokenPose-B, and TokenPose-L/D6 [97], namely PPT-S, PPT-B, and PPT-L/D6, respectively. For PPT-S and PPT-B, the number of encoder layers L is set to 12, the embedding size D is set to 192, the number of heads H is set to 8. They take the shallow stem-net and the HRNet-W32 as the CNN backbone, respectively. Following [132, 98], the HTI is performed $e = 3$ times and is inserted before the 4^{th}, 7^{th}, and 10^{th} encoder layers. The PPT-L/D6 has $L = 12$ encoder layers and takes HRNet-W48 as the backbone. the HTI is inserted before the 2^{th}, 4^{th}, and 5^{th} encoder layers. The number of visual tokens N_v is 256 for all networks, and the keep ratio r is set to 0.7 by default. Thus, only 88 visual tokens are left after three rounds pruning. We follow the same training recipes as [97]. In detail, all networks are optimized by Adam optimizer [85] with Mean Square Error (MSE) loss for 300

Method	#Params	GFLOPs	GFLOPs^T	AP	AP^{50}	AP^{75}	AP^M	AP^L	AR
SimpleBaseline-Res50 [181]	34M	8.9	-	70.4	88.6	78.3	67.1	77.2	76.3
SimpleBaseline-Res101 [181]	53M	12.4	-	71.4	89.3	79.3	68.1	78.1	77.1
SimpleBaseline-Res152 [181]	68.6M	15.7	-	72.0	89.3	79.8	68.7	78.9	77.8
HRNet-W32 [152]	28.5M	7.1	-	74.4	90.5	81.9	70.8	81.0	79.8
HRNet-W48 [152]	63.6M	14.6	-	75.1	90.6	82.2	71.5	81.8	80.4
Lite-HRNet-18 [199]	1.1M	0.20	-	64.8	86.7	73.0	62.1	70.5	71.2
Lite-HRNet-30 [199]	1.8M	0.31	-	67.2	88.0	75.0	64.3	73.1	73.3
EfficientPose-B [210]	3.3M	1.1	-	71.1	-	-	-	-	-
EfficientPose-C [210]	5.0M	1.6	-	71.3	-	-	-	-	-
TransPose-R-A4 [191]	6.0M	8.9	3.38	72.6	89.1	79.9	68.8	79.8	78.0
TransPose-H-S [191]	8.0M	10.2	4.88	74.2	89.6	80.8	70.6	81.0	79.5
TransPose-H-A6 [191]	17.5M	21.8	11.4	75.8	90.1	82.1	71.9	82.8	80.8
TokenPose-Small [97]	6.6M	2.2	1.44	72.5	89.3	79.7	68.8	79.6	78.0
PPT-Small (ours)	6.6M	1.6(**-27%**)	0.89(-38%)	72.2(**-0.3**)	89.0	79.7	68.6	79.3	77.8
TokenPose-Base [97]	13.5M	5.7	1.44	74.7	89.8	81.4	71.3	81.4	80.0
PPT-Base (ours)	13.5M	5.0(**-12%**)	0.89(-38%)	74.4(**-0.3**)	89.6	80.9	70.8	81.4	79.6

Table 3.1: Results on COCO validation dataset. The input size is 256 × 192. GFLOPs^T means the GFLOPs for the transformers only following equations from [89], as our method only focus on accelerating the transformers.

Method	#Params	GFLOPs	Head	Sho	Elb	Wri	Hip	Kne	Ank	Mean
SimpleBaseline-Res50 [181]	34M	12.0	96.4	95.3	89.0	83.2	88.4	84.0	79.6	88.5
SimpleBaseline-Res101 [181]	53M	16.5	96.9	95.9	89.5	84.4	88.4	84.5	80.7	89.1
SimpleBaseline-Res152 [181]	53M	21.0	97.0	95.9	90.0	85.0	89.2	85.3	81.3	89.6
HRNet-W32. [152]	28.5M	9.5	96.9	96.0	90.6	85.8	88.7	86.6	82.6	90.1
TokenPose-S [97]	7.7M	2.5	96.0	94.5	86.5	79.7	86.7	80.1	75.2	86.2
PPT-S	7.7M	1.9 (-**24%**)	96.6	94.9	87.6	81.3	87.1	82.4	76.7	87.3 (+**1.1**)
TokenPose-B [97]	14.4M	7.1	97.0	96.1	90.1	85.6	89.2	86.1	80.3	89.7
PPT-B	14.4M	6.2 (-**13%**)	97.0	95.7	90.1	85.7	89.4	85.8	81.2	89.8 (+**0.1**)

Table 3.2: Results on the MPII validation set (PCKh@0.5). The input size is 256 × 256.

epochs. The learning rate is initialized with 0.001 and decays at the 200-th and the 260-th epoch with ratio 0.1. As locating human is difficult at early training stages, the keep ratio is gradually reduced from 1 to r with a cosine schedule during the early 100 epochs.

3.4.2 Results

The results are shown in Table 3.1 and Table 3.2 for COCO and MPII, respectively. Generally, the transformer-based methods [97, 191] maintain less number of parameters. On COCO, compared with the TokenPose, PPT achieves significant acceleration while matching its accuracy. For example, PPT-S reduces 27% total inference FLOPs while only reducing

0.3 AP. Compared to SimpleBaseline-ResNet152 [181], PPT-S achieves equal performance but only requires 10% FLOPS. We can also observe consistent conclusion on PPT-B and PPT-L. Note that, for PPT-B and PPT-L, the CNN backbone takes a large portion of computation. Thus, the reduction of total FLOPs is relatively small. Meanwhile, compared with other efficient pose estimation networks [199, 210], the AP of PPT-S is 72.2, which is much better than EfficientPose-C [210] with 71.3 AP at the same FLOPs level. More over, On MPII, our PPT-S can even improve on the PCKh of TokenPose-S by 1.1%. We believe that slimming the number of tokens can also make the attention focus on key elements [221]. Thus, our PPT is efficient yet powerful, and it is applicable to any TokenPose variants. All of these results suggest that pruning background tokens does not hurt the overall accuracy and calculating attention among human foreground tokens is sufficient for 2D human pose estimation.

3.4.3 Runtime evaluation

Although the GFLOPs reflects the efficiency of networks, it is not equivalent to the real runtime on hardware due to different implementation. We further report the throughput, which measures the maximal number of input instances the network can process in time a unit. Unlike FPS (frame per second), which involves the processing of a single instance, the throughput evaluates the processing of multiple instances in parallel. During the inference time of the top-down method, given one input image, multiple human instances located by an object detector are usually cropped, resized, and combined into a minibatch to accelerate the inference. Then the minibatch of multiple human instances is fed into the pose detector. Thus, we believe throughput is a more reasonable metric to evaluate top-down 2D human pose estimation networks.

We set the batch size to 32 for all networks, and compute the throughput on a single 2080

Ti GPU. Both FPS and throughput of PPT and TokenPose [97] are shown on Table 3.3. Remarkably, pruning tokens cannot significantly improve the time of a single instance (*i.e.*, FPS). We believe the extra time introduced by the pruning operation is not negligible. Nevertheless, PPT significantly improves the throughput from TokenPose, which is consistent with the improvement of GFLOPs in Table 3.1. We further show the comparison of throughput with other methods in Figure 3.5. Our PPT consistently improves the throughput at the same AP level. Thus, pruning token does improve the runtime on hardware in practice.

Method	#Params	AP	FPS	Throughput
TokenPose-S	6.6M	72.5	120	651
PPT-S	6.6M	72.2	123	842 (+**30**%)
TokenPose-B	13.5M	74.7	50	388
PPT-B	13.5M	74.3	51	451 (+**16**%)
TokenPose-L/D6	20.8M	75.4	60	325
PPT-L/D6	20.8M	75.2	61	334 (+**3**%)

Table 3.3: FPS and Throughput on COCO validation dataset.

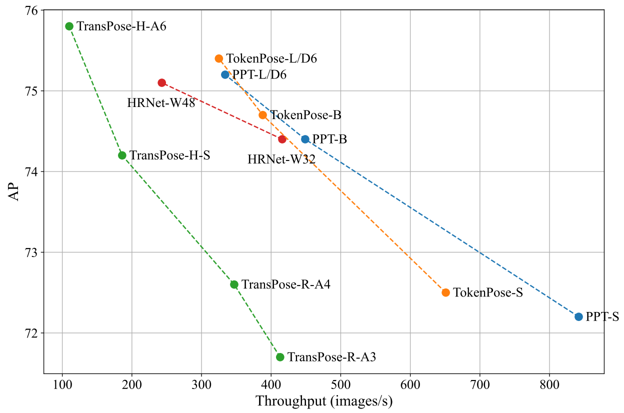


Figure 3.5: Comparison of throughput on COCO validation dataset.

3.4.4 Visualizations

We visualize the selected tokens from PPT-S in Fig. 3.6. We present the original images and the selected tokens at different layers. Remarkably, the human areas are gradually refined as the network deepens. The final selected tokens can be considered as a rough human mask. Thus, our HTI can successfully locate human tokens as expected. Moreover, the HTI can handle quite a few complicated situations such as man-object interaction (Fig.3.6b), oblique body pose (Fig. 3.6c), occlusion (Fig. 3.6d), and multiple persons (Fig.3.6e and Fig.3.6f). Nevertheless, when only part of human body appears in the image (Fig.3.6g and Fig.3.6h), the quality of the located human mask could be imperfect. In these cases, we hypothesize that some keypoint tokens such as ankle and knee cannot locate the corresponding joints as they are invisible. Thus, they may just give equal attention score, which leads to inaccurate token selection.

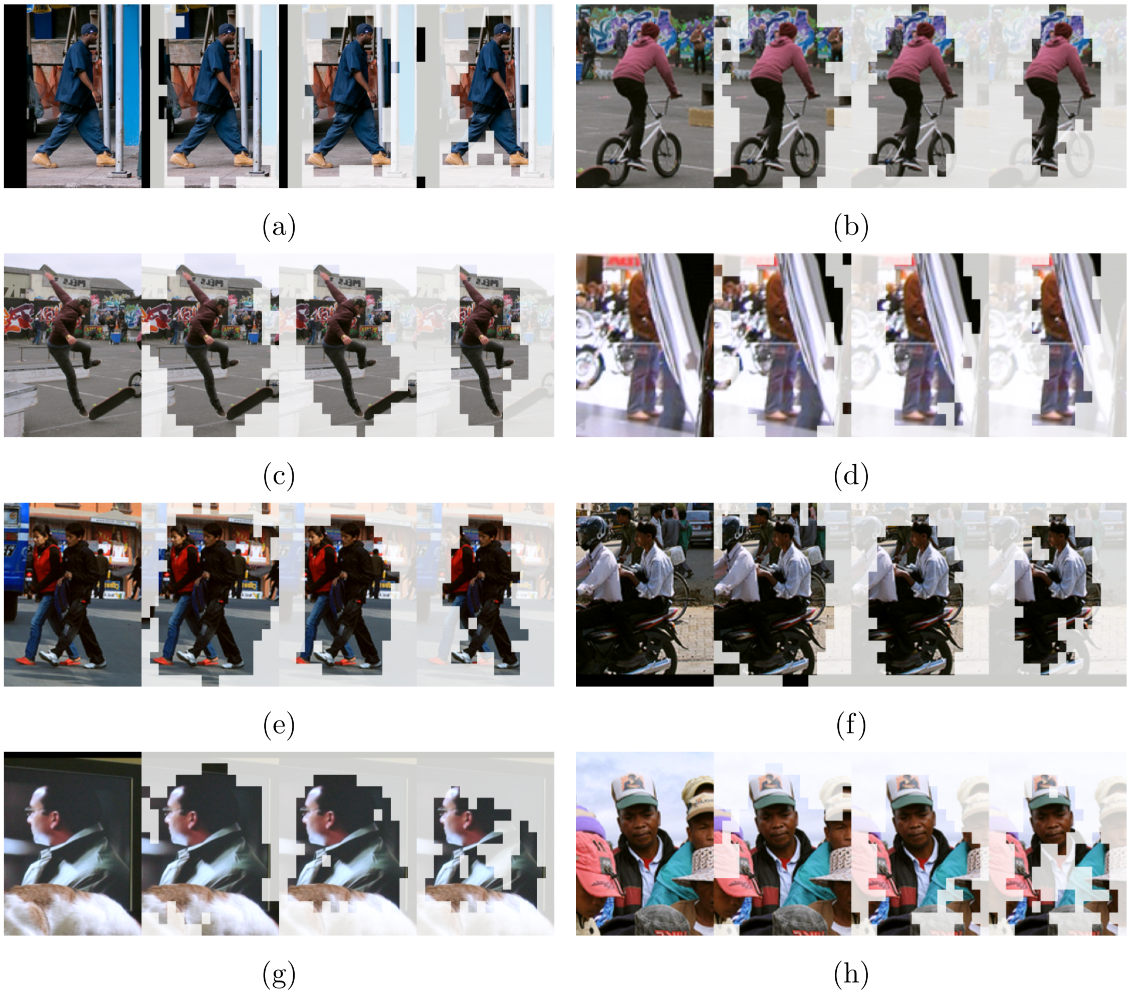


Figure 3.6: Visualizations of the selected tokens at each HTI module on COCO. The masked regions represent the pruned tokens (We use blue circles to mask out face for privacy issue). For each image group, the first column is the original image, the 2nd, 3rd, and 4th colums are the selected tokens by HTI at the 4^{th},7^{th}, and 10^{th} layers, respectively.

3.4.5 Ablation Studies

The keep ratio r controls the trade-off between the acceleration and the accuracy. Meanwhile, reducing tokens also introduces some regularization [221]. We take PPT-S and vary r from 0.6 to 0.8 on both COCO and MPII. The results are shown in Table 3.4. The reduction of AP is always less than 1%. When the r is relatively small, PPT can achieve considerable speedup but may not cover the entire human body. As a result, the accuracy of pose estimation is slightly dropped. To maintain the accuracy, we choose 0.7 as our default keep ratio.

Method	Keep Ratio	# Visual Tokens	COCO			MPII		
			AP	AR	FLOPs	PCKh@0.5	PCKh@0.1	FLOPs
TokenPose-S	1.0	256 (100%)	72.5	78.0	2.23	86.2	32.2	2.53
PPT-S	0.8	131 (51%)	72.0 (-0.5)	77.6(-0.4)	1.75 (-22%)	86.9 (+0.7)	32.9 (+0.7)	2.06 (-19%)
PPT-S	0.7	88 (34%)	72.2 (-0.3)	77.8 (-0.2)	1.61 (-27%)	87.3 (+1.1)	34.1 (+1.9)	1.92 (-24%)
PPT-S	0.6	56 (22%)	71.8 (-0.7)	77.5 (-0.5)	1.52 (-32%)	86.7 (+0.5)	32.3 (+0.1)	1.82 (-28%)

Table 3.4: Results of PPT-S on COCO and MPII with different keep ratio r.

3.5 Experiments on Multi-view Pose Estimation

3.5.1 Settings

Datasets & Evaluation Metrics. We evaluate multi-view PPT on two single-person datasets of multi-view 3D human pose estimation, *i.e.*, Human 3.6M [73, 16] and Ski-Pose [149, 45] [1]. Human 3.6M contains video frames captured by $M = 4$ indoor cameras. It includes many daily activities such as eating and discussion. We follow the same train-test split as in [129, 74, 61], where subjects $1, 5, 6, 7, 8$ are used for training, and $9, 11$ are for testing. We also exclude some scenes of $S9$ from the evaluation as their 3D annotations are damaged [74]. Ski-Pose contains video frames captured by outdoor cameras. It is created to help analyze skiers's giant slalom. There are $8,481$ and $1,716$ frames in the training

[1] Only authors from UCI downloaded and accessed these two datasets. Authors from Tencent and Meta don't have access to them.

Method	#V	MACs	shlder	elb	wri	hip	knee	ankle	root	belly	neck	nose	head	Avg
ResNet50 [181]	1	51.7G	97.0	91.9	87.3	99.4	95.0	90.8	100.0	98.3	99.4	99.3	99.5	95.2
TransPose [191]	1	43.6G	96.0	92.9	88.4	99.0	95.0	91.8	100.0	97.5	99.0	99.4	99.6	95.3
TokenPose [97]	1	11.2G	96.0	91.3	85.8	99.4	95.2	91.5	100.0	98.1	99.1	99.4	99.1	94.9
Epipolar Transformer [61]	2	51.7G	97.0	93.1	91.8	99.1	96.5	91.9	100.0	99.3	99.8	99.8	99.3	96.3
TransFusion [106]	2	50.2G	97.2	96.6	93.7	99.0	96.8	91.7	100.0	96.5	98.9	99.3	99.5	96.7
Crossview Fusion [129]	4	55.1G	97.2	94.4	92.7	**99.8**	97.0	92.3	100.0	98.5	99.1	99.1	99.1	96.6
TokenPose+Transformers	4	11.5G	97.1	97.3	95.2	99.2	98.1	93.1	100.0	98.8	99.2	99.3	99.1	97.4
PPT	1	9.6G	96.0	91.8	86.5	99.2	95.6	92.2	100.0	98.4	99.3	99.5	99.4	95.3
Multi-view PPT	2	9.7G	97.1	95.5	91.9	99.4	96.4	92.1	100.0	99.0	99.2	99.3	99.0	96.6
Multi-view PPT (w.o. 3DPE)	4	9.7G	97.6	**98.0**	96.4	99.7	98.4	93.8	100.0	99.0	**99.4**	**99.5**	**99.5**	**97.9**
Multi-view PPT	4	9.7G	**98.0**	**98.0**	**96.4**	99.7	**98.5**	**94.0**	**100.0**	**99.1**	99.2	99.4	99.3	**98.0**

Table 3.5: 2D pose estimation on Human3.6M. The metric is JDR on original image. All inputs are resized to 256 × 256. #V means the number of views used in cross-view fusion step. The FLOPs is the total computation for each view and cros-view fusion.

Method	Dir	Disc	Eat	Greet	Phone	Pose	Purch	Sit	SitD	Smoke	Photo	Wait	WalkD	Walk	WalkT	Avg
Crossview Fusion[129]	24.0	28.8	25.6	24.5	28.3	24.4	26.9	30.7	34.4	29.0	32.6	25.1	24.3	30.8	24.9	27.8
Epipolar Trans. [61]	23.2	27.1	23.4	22.4	32.4	**21.4**	**22.6**	37.3	35.4	29.0	27.7	24.2	21.2	26.6	**22.3**	27.1
TransFusion [106]	24.4	**26.4**	23.4	**21.1**	25.2	23.2	24.7	33.8	**29.8**	26.4	26.8	24.2	23.2	26.1	23.3	25.8
Multi-PPT	**21.8**	26.5	**21.0**	22.4	**23.7**	23.1	23.2	**27.9**	30.7	**24.6**	**26.7**	**23.3**	**21.2**	**25.3**	22.6	**24.4**

Table 3.6: The MPJPE of each pose sequence on Human 3.6M.

and testing sets, respectively. We use the Joint Detection Rate (JDR) on original images [129] to evaluate the 2D pose accuracy. JDR measures the percentage of successfully detected keypoints within a predefined distance of the ground truth location. The 3D pose is evaluated by Mean Per Joint Position Error (MPJPE) between the ground truth 3D pose in world coordinates and the estimated 3D pose.

Implementation Details. We build multi-view PPT upon PPT-S. The first 9 transformer layers are used to extract human tokens, and the last 3 transformer layers are used for cross-view fusion. Thus, no additional parameters are introduced. Following the settings in [61, 106], we start from a PPT-S pre-trained on COCO and finetune it on multi-view human pose datasets, as it is difficult to train the transformer from scratch with examples in limited scenes. We apply Adam optimizer and train the model for 20 epochs with MSE loss. The learning rate starts with 0.001 and later on decays at 10-th and 15-th epoch with ratio 0.1. The keep ratio r is set to 0.7 through the entire training process. We resize input images to 256 × 256 and follow the same data augmentation in [129, 106].

3.5.2 Results

The 2D results on Human 3.6m is shown in Table 3.5. The MACs (multiply-add operations) consider both single-view forward MACs of all views and cross-view fusion MACs. Noticeably, our multi-view PPT outperforms all previous cross-view fusion methods on JDR. The JDR can be further improved with the 3D positional encodings (3DPE) [106] on visual tokens. Meanwhile, it can significantly reduce the computation of all 4 view fusion, *i.e.*, the MACs is reduced from 55.1G to 9.7G. When only fusing 2 views, multi-view PPT still achieves comparable accuracy with other two-view-fusion methods [61, 106], Moreover, we add the baseline that adds transformers on top of TokenPose to perform cross-view fusion, which can be considered as multi-view PPT without token pruning. The JDR is 97.4% (-0.7% with respect to our multi-view PPT), which supports that our human area fusion is better than global attention in both accuracy and efficiency. The MPJPE of estimated 3D pose is reported in Table 3.6. We can observe that multi-view PPT also achieves the best MPJPE on 3D pose, especially on sophisticated action sequences such as “Phone” and “Smoke”, as the result of 3D pose is determined by the accuracy of 2D pose. Therefore, our “human area fusion” strategy is better than previous fusion strategies as it strikes a good balance between efficiency and accuracy. We can also observe consistent conclusion on Ski-Pose from Table 3.7. Nevertheless, it seems that the performance in this datatset tends to be saturated. The reason might be that there is limited number of training examples, thus the transformer is easy to overfit.

Method	MACs	2D Pose / JDR (%) ↑	3D Pose / MPJPE (mm) ↓
Simple Baseline-Res50 [181]	77.6G	94.5	39.6
TokenPose [97]	16.8G	95.0	35.6
Epipolar Transformer [61]	77.6G	94.9	34.2
Multi-view PPT	**14.5G**	**96.3**	**34.1**

Table 3.7: 2D and 3D pose estimation accuracy comparison on Ski-Pose.

3.5.3 Visualizations

Human Tokens. Fig.3.7 presents the selected human tokens in all views. Similar to the conclusion on COCO, our PPT accurately locates all human areas and prunes background areas in all views. Moreover, the tokens used in the cross-view fusion step can be significantly reduced.

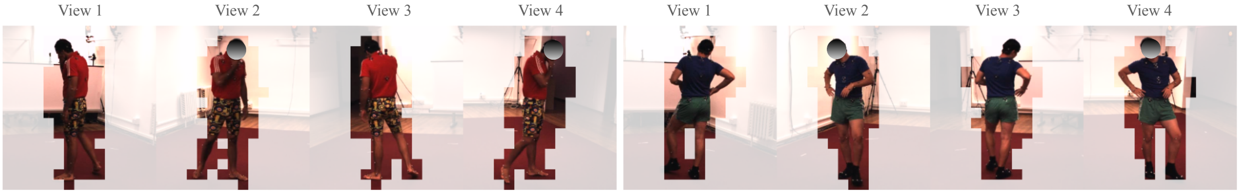


Figure 3.7: Visualizations of the final located tokens on Human 3.6M validation set. For each group, each column is an image from one view. The masked regions represent the pruned tokens. We perform cross-view fusion among these selected tokens.

Qualitative results. We present examples of predicted 2D heatmaps on the image in Fig.3.8, and compare our methods with TransFusion [106]. It is observed that our method can solve heavy occlusion cases very well, while TransFusion cannot. For two-view-fusion method, occlusion cases in current view may still be occluded in the neighbor view. For example, the heatmap marked with red box is inaccurate in both view 2 and view 4. Thus, fusing this bad quality heatmap cannot improve the final prediction. However, our method can avoid this problem by fusing clues from all views.

Attentions. We present an example of the attention map between keypoint tokens in Fig.3.9. Given keypoint tokens in one view, they pay attention to keypoints tokens in all views. For example, the left wrist in the first view (blue dot) is occluded, thus its corresponded keypoint token attends to the keypoint token in the second view, where the keypoint is visible. Therefore, the keypoint token in multi-view PPT can learn the dependencies among joints in different views.

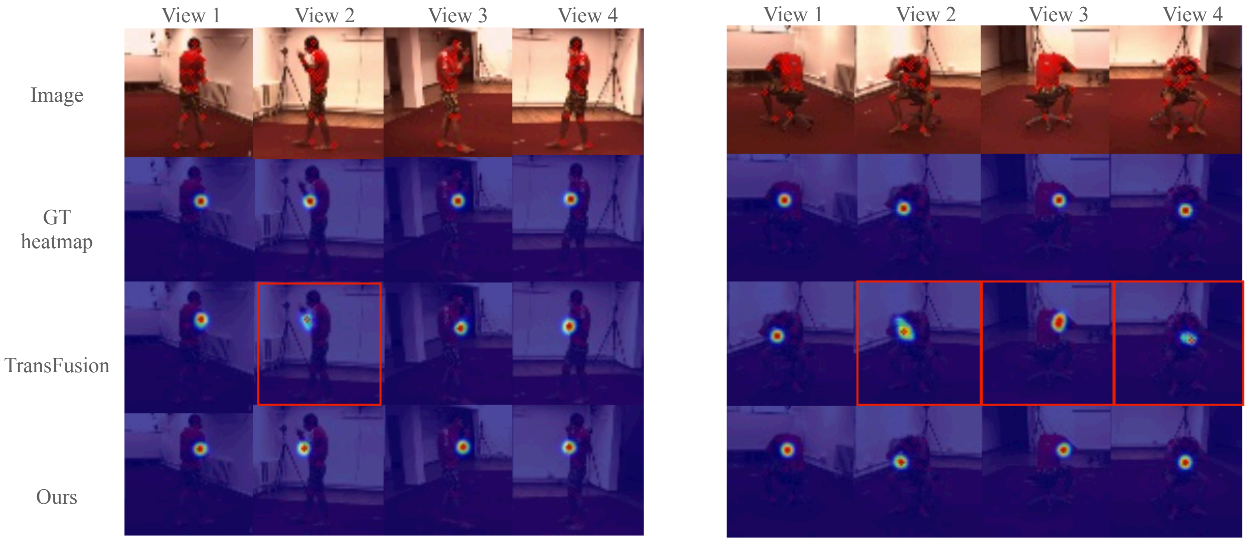


Figure 3.8: Sample heatmaps of our approach.

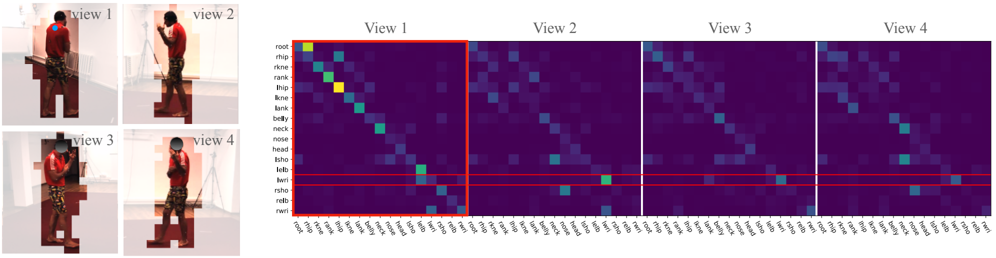


Figure 3.9: Attention maps among keypoint tokens.

3.6 Conclusion

We propose the PPT for 2D human pose estimation. Experiments on COCO and MPII show that the PPT achieves similar accuracy compared with previous transformer-based networks but reduces the computation significantly. We also empirically show that PPT can locate a rough human mask as expected. Furthermore, we propose the multi-view PPT to perform the cross-view fusion among human areas. We demonstrate that multi-view PPT efficiently fuses cues from many views and outperforms previous cross-view fusion methods on Human 3.6M and Ski-Pose.

Chapter 4

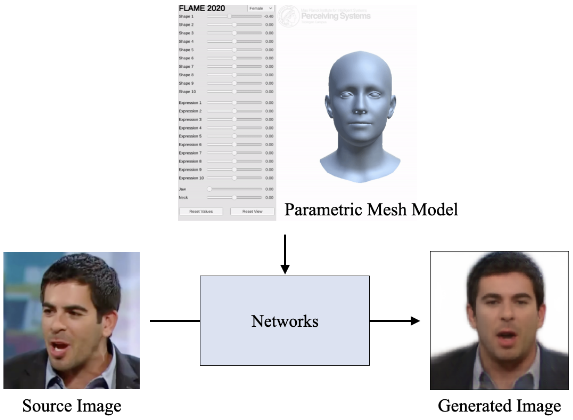
CVTHead: One-shot Controllable Head Avatar with Vertex-feature Transformer

Reconstructing personalized animatable head avatars has significant implications in the fields of AR/VR. Existing methods for achieving explicit face control of 3D Morphable Models (3DMM) typically rely on multi-view images or videos of a single subject, making the reconstruction process complex. Additionally, the traditional rendering pipeline is time-consuming, limiting real-time animation possibilities. To this end, we introduce CVTHead, a novel approach that generates controllable neural head avatars from a single reference image using point-based neural rendering. CVTHead considers the sparse vertices of mesh as the point set and employs the proposed Vertex-feature Transformer to learn local feature descriptors for each vertex. This enables the modeling of long-range dependencies among all the vertices. Experimental results on the VoxCeleb dataset demonstrate that CVTHead achieves comparable performance to state-of-the-art graphics-based methods. Moreover, it enables efficient rendering of novel human heads with various expressions, head poses, and camera views. These attributes can be explicitly controlled using the coefficients of 3DMMs, facilitating versatile and realistic animation in real-time scenarios.

4.1 Problem Definition

The problem of mesh-guided head avatar generation aims to generate realistic human head image under the control of a parametric head mesh model. As shown in Figure 4.1, the inputs of this task include an RGB image $\mathbf{I} \in \mathbb{R}^{H \times W \times 3}$ for the appearance, and also the parametric mesh $M(\beta, \phi, \theta) \in \mathbb{R}^{V \times 3}$, where V is the number of vertices for the mesh, while β, ϕ, and θ are the parameters for shape, expression, and pose, respectively. These parameters can come from other person, or manual adjustment. This task aims to learn a mapping $f(\cdot)$ to generate an RGB image $\hat{\mathbf{I}}$ that makes the person have the shape, expression, and camera view from the given mesh, i.e.,

$$\hat{\mathbf{I}} = f(\mathbf{I}, M(\beta, \phi, \theta)) \tag{4.1}$$



FLAME 2020

Female

Shape 1 -0.40
Shape 2 0.00
Shape 3 0.00
Shape 4 0.00
Shape 5 0.00
Shape 6 0.00
Shape 7 0.00
Shape 8 0.00
Shape 9 0.00
Shape 10 0.00
Expression 1 0.00
Expression 2 0.00
Expression 3 0.00
Expression 4 0.00
Expression 5 0.00
Expression 6 0.00
Expression 7 0.00
Expression 8 0.00
Expression 9 0.00
Expression 10 0.00
Jaw 0.00
Neck 0.00

Figure 4.1: The illustration of mesh-guided head avatar generation.

4.2 Introduction

Personalized head avatars play a crucial role in a wide range of applications, including AR/VR, teleconferencing, and the movie industry. Over the past few decades, there has been an extensive exploration of personalized head avatars in the fields of computer graphics and computer vision. Traditional solution [1] reconstructs a personalized mesh and texture for the source actor explicitly with 3D head scans [188, 180]. To perform full face control, 3D Morphable Models (3DMM) [8, 95] are used as a strong prior of face geometry. 3DMMs is a parametric model and uses PCA-based linear blendshapes to explicitly control face shape, expressions, texture, and head pose independently. However, 3DMM does not model the facial detail and hair region of the human face [54]. Recently, with the development of Neural Radiance Fields (NeRF) [115], reconstructing avatars with implicit models becomes popular as it can reconstruct detailed regions [47, 123, 5]. However, all these methods are subject-specific and they usually require video inputs or multi-view images of the same subject, which limits their usage in practice.

Hence, acquiring human avatars from a single image (i.e., one-shot face reenactment) becomes more and more popular [178, 157, 209, 143, 204, 169, 159, 193, 39, 83]. Given a facial image of an actor, the synthetic images can be driven by videos from other actors. A key step behind these methods is to decouple the facial appearance and motion information from the source and driven images. As a result, mesh-guided face animation has gained significant attention, primarily due to the inherent disentanglement of identity and expression offered by 3DMM. Generally, one-shot mesh-guided face animation can be roughly divided into warp-based and graphics-based. Warp-based methods [193, 39, 189, 206] employ the motion field to transfer the driving pose and expression into the source face. These methods effectively preserve fine facial details and produce high-fidelity results but only work well for a limited range of head poses. Graphic-based methods [46, 83] learn texture maps [10] from single-image and apply computer graphics pipelines to render the animated face image. Thus, it can maintain performance under large head rotations and guarantee the 3D consistency of rendered images. However, the rendering pipeline is usually computationally heavy [78], which makes efficient rendering unachievable.

Alternatively, point-based graphics [55] get rid of the surface mesh and directly use point clouds to model the 3D geometry. Later on, the point-based neural rendering techniques [2, 131, 185] augment each RGB point with a learnable neural descriptor that is interpreted by the neural renderer. The recent SMPLpix [127] further extends these techniques from static scenes to dynamic scenes and enables the efficient rendering of human body avatars under novel subject identities and human poses. Although efficient, these methods still require multi-view images with calibrated cameras to reconstruct the accurate point cloud first. Consequently, applying these methods to one-shot face reenactment is infeasible.

In this work, we utilize point-based neural rendering to achieve an efficient and realistic generation of head avatars from a single image. We direct utilize the sparse vertices from the FLAME head model [95] as our point set, instead of reconstructing a dense point cloud of the subject tediously. Specifically, given the vertices from pre-trained 3D face reconstruction networks [46], we learn a local feature descriptor aligned with each vertex. When learning the local descriptor of a 3D point from a 2D reference image, pixel-aligned features [139, 71, 60, 34] are a popular choice. However, these aligned features often become incorrect when the projected 2D location is occluded in the source image. To address this challenge, we propose the *Vertex-feature Transformer.* This approach treats each vertex as a query token [37] and utilizes transformers [165] to directly learn the canonical vertex features from the reference image. By incorporating a global attention mechanism, our model can capture long-range dependencies within the features of all vertices. Thus, the feature descriptor of invisible 3D points can still be reconstructed correctly. Next, we project the feature descriptor and depth of each vertex into image space and employ a UNet-like neural rendering to generate the RGB image. Since the feature descriptor is aligned with the vertices of the FLAME head model, the rendered face can be explicitly controlled by the shape, expression, and head pose coefficients. We name this end-to-end framework as CVTHead, in short for Controllable head avatar with Vertex-feature Transformer.

Our major contributions are summarized as follows:

- We propose CVTHead, a one-shot controllable head avatar framework using point-based neural rendering that can efficiently render novel human heads under novel expressions and camera views. To the best of our knowledge, this is the first work that performs point-based neural rendering from a monocular face image.
- We propose Vertex-feature Transformer to learn the vertex descriptor in canonical space from a single image with transformers, and demonstrate its superiority beyond projection methods.
- By conducting experiments on VoxCeleb1 and VoxCeleb2, we establish that our method achieves performance that is on par with the state-of-the-art approaches, while additionally improving efficiency.

4.3 Related Work

Mesh-guided Face Reenactment Extensive research has been conducted on employing 3DMM for the explicit animation of human face images [156, 51, 193, 39, 167, 189, 83, 40, 206]. Mesh-guided face reenactment can be divided into warp-based and graphic-based. Warping-based methods [193, 39, 167, 189] warp the source image with explicit motion fields. For example, given both source and driving meshes, Yao et al. [193] extract the motion features with Graph Convolutional Networks. HeadGAN [39] learns the dense flow field with PNCC [220] and SPADE. Face2Face$^{\rho}$ [189] calculates the motion with a set of

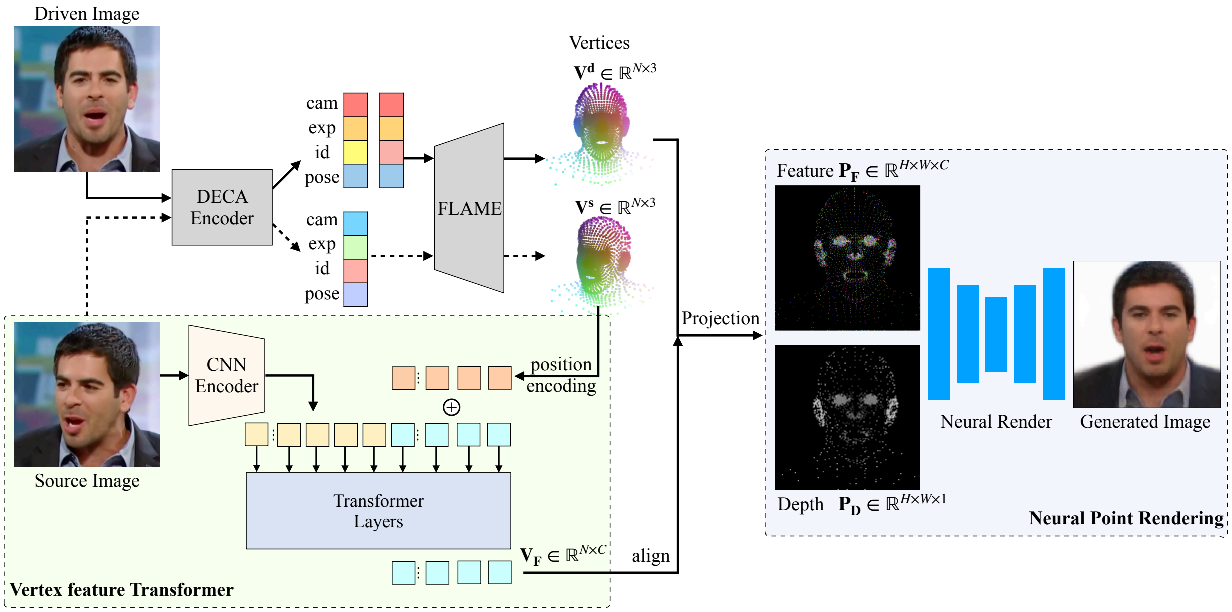

Driven Image
DECA Encoder
cam
exp
id
pose
cam
exp
id
pose
FLAME
Vertices
$\mathbf{V^d} \in \mathbb{R}^{N\times3}$
$\mathbf{V^s} \in \mathbb{R}^{N\times3}$
Projection
Feature $\mathbf{P_F} \in \mathbb{R}^{H\times W\times C}$
Depth $\mathbf{P_D} \in \mathbb{R}^{H\times W\times 1}$
Neural Render
Generated Image
Neural Point Rendering
Source Image
CNN Encoder
position encoding
Transformer Layers
$\mathbf{V_F} \in \mathbb{R}^{N\times C}$
align
Vertex feature Transformer

Figure 4.2: Overview of the CVTHead framework. We employ a pretrained face reconstruction network [46] to obtain the face mesh (Section 4.4.2) and utilize the proposed verte feature transformers to obtain the feature descriptor of each vertex from the source image (Section 4.4.3). We then consider the sparse vertex as point set and use point-based neural rendering to synthesize the image (Section 4.4.4).

pre-specified 3D keypoints. However, when faced with significant head rotations, the quality of these approaches drops significantly. Meanwhile, other methods [46, 83, 54] obtain the animated face images from the head mesh with the classic graphics rendering pipeline. In detail, DECA [46] simultaneously learns both the head mesh and the linear albedo subspace of the Basel Face Model [125]. To create realistic face photos, ROME [83] estimates a neural texture and offset for each vertex from the source image and renders the rigged mesh with deferred neural rendering technique [158]. Nevertheless, these methods still require the time-consuming classic differentiable rendering [133].

Neural Head Avatars Recently, several works extend NeRF [2] to model dynamic objects such as virtual avatars with implicit neural representations [47, 123, 77, 5, 68, 222]. For example, subject-dependent methods such as NerRACE [47] and RigNeRF [5] use 3DMM-guided deformation neural fields to enable control over head pose, facial expression, and viewpoints. Typically, these approaches employ an optimization-based head tracker [159] as a preprocessing step to extract accurate 3DMM coefficients from a monocular video of the subject. Meanwhile, subject-agnostic methods such as HeadNeRF [68] and MofaNeRF [222] learn the radiance fields from large-scale multi-view images [188]. When generating an avatar for a novel subject, these methods require time-consuming inverse rendering optimization to obtain the latent codes. Most recently, HiDe-NeRF [96] and OTAvatar [110] employ tri-plane representations [18] to efficiently extract multi-scale features for each query 3D point and also use volume rendering to reconstruct images. Although promising, a limitation of volumetric rendering is the necessity to sample hundreds of 3D points per ray and then feed them through the network to render a single pixel or feature patch. In contrast, our method utilizes a set of points to represent the head avatar, thereby necessitating only a single forward pass for rendering.

Neural Point-based Rendering Neural point-based rendering [2, 91, 131, 185, 81] has gained significant attention in recent years for its ability to generate high-quality images by directly rendering point clouds from static scenes. Aliev et al. [2] introduced Neural Point-Based Graphics (NPBG), which employs learnable neural descriptors to enhance each point for better rendering. Later on, NPBG++ [131] further predicts the descriptors with a single pass to accelerate rendering. Meanwhile, SMPLpix [127] extends point-based neural rendering to generate human avatars under the control of SMPL [103]. Although SMPLpix is a subject-agnostic method, it requires registering the SMPL model to ground truth 3D scans to obtain the RGB color of each vertex. The most recent PointAvatar [219] models the human head as an explicit canonical point cloud and continuous deformation to create realistic and relightable head avatars. However, it is still subject-dependent and requires a video caption of the subject to train the model. Our method also employs neural point-based rendering but simplifies the setting as we only require a single image of the novel subject.

Transformers in Mesh Over the past few years, transformers [165] have made significant progress in many computer vision tasks [163, 37, 14, 187, 108, 190, 90]. There are a few works that also apply transformers to mesh data [99, 100, 31, 194, 216, 38]. In detail, METRO [99] apply transformers to predict the mesh coordinates and 3D joints simultaneously of the human body [103]. Mesh Graphormer [100] further utilizes the topology of the mesh with graph convolution to improve the mesh reconstruction. Both works consider each vertex as a query token and use transformers to learn the non-local relationships among vertices. In our work, we also use transformers to learn the correspondence among vertex features.

4.4 Methodology

4.4.1 Overview

Fig 4.2 illustrates the overall framework of our CVTHead. Given both source image $\mathbf{I^s}$ and driven image $\mathbf{I^d}$, we utilize a pre-trained face reconstruction model [46] to obtain the source and driven vertex coordinates $\mathbf{V^s}$ and $\mathbf{V^d} \in \mathbb{R}^{N\times 3}$ of the FLAME model [95] (Sec. 4.4.2), where N is the number of vertices in FLAME model. Simultaneously, we employ the proposed vertex feature transformer to learn the feature descriptor for all vertices in canonical space $\mathbf{V_F} \in \mathbb{R}^{N\times C}$ from the source image (Sec. 4.4.3), where C is the number of channels of feature descriptor. Then we project driven vertices and their corresponding feature descriptors onto the vertex feature image $\mathbf{P^d_F} \in \mathbb{R}^{H\times W\times C}$ and the depth image $\mathbf{P^d_D} \in \mathbb{R}^{H\times W\times 1}$, where H and W is the height and width of the original image. Next, we conduct neural rendering with a U-Net $\mathcal{G}(\cdot)$ to generate the synthetic image $\hat{\mathbf{I}} = \mathcal{G}(\mathbf{P_F}, \mathbf{P_D}) \in \mathbb{R}^{H\times W\times 3}$ (Sec. 4.4.4). Our framework enables end-to-end training, allowing the entire process to be optimized jointly. During inference, our system enables the rendered image to be animated with novel shapes, expressions, head poses, and viewpoints by manipulating the FLAME parameters. This flexibility allows for the generation of diverse and customizable head avatars.

4.4.2 Head Mesh Reconstruction

FLAME [95] is a parametric 3D head model with $N = 5023$ vertices. It encompasses a mean template $V_b \in \mathbb{R}^{N\times 3}$, along with shape blendshapes $\mathcal{S} \in \mathbb{R}^{N\times 3\times L}$, and expression blendshapes $\mathcal{E} \in \mathbb{R}^{N\times 3\times K}$. These blendshapes are derived from a vast collection of 4D scans of human heads, allowing FLAME to capture a wide range of facial variations. Given parameters of facial identity $\beta \in \mathbb{R}^L$, expression $\phi \in \mathbb{R}^K$ and pose $\theta \in \mathbb{R}^{3k+3}$ (with $k = 4$ joints for neck, jaw, and eyeballs), FLAME first apply β and ϕ to corresponding blendshapes, resulting in modified vertex positions. Next, the linear blend skinning (LBS) technique $W(\cdot, \cdot)$ is employed to rotate the vertices based on θ. The final reconstruction of FLAME in world coordinates is calculated by:

$$M(\beta, \phi, \theta) = W(V_b + \mathcal{S}\beta + \mathcal{E}\phi, \theta) \in \mathbb{R}^{3n} \tag{4.2}$$

We employ the pre-trained DECA [46] $f_D(\cdot)$ to obtain β,ϕ,θ and camera parameters c from both source images and driven images with a single forward, i.e, $\beta^s, \phi^s, \theta^s, c^s = f_D(\mathbf{I^s})$ and $\beta^d, \phi^d, \theta^d, c^d = f_D(\mathbf{I^d})$. We also obtain the deformation of hair and shoulder regions from the source image with the pre-trained linear deformation model $f_H(\cdot)$ [83] to refine the vertices locations. Then we obtain the driven vertex coordinates by

$$\mathbf{V^d} = M(\beta^s, \phi^d, \theta^d) + f_H(\mathbf{I^s}) \in \mathbb{R}^{N\times 3} \tag{4.3}$$

4.4.3 Vertex-feature Transformer

Motivations In previous approaches that utilize pixel-aligned features [139, 34], the feature descriptor of a given 3D point is determined by the feature located at its corresponding 2D projection. In detail, given the 3D point $\mathbf{k}^s \in \mathbf{V^s}$, we project it into the 2D image space by $(u^s, v^s, d^s) = \Pi(\mathbf{k}^s, c_s)$, where $\Pi(\cdot)$ represents the orthographic projection function and c_s is the camera parameters of the reference image obtained from pre-trained DECA. The descriptor of $\mathbf{k}^s$ is defined as $I'[u^s, v^s]$, where I' is the 2D feature map of the source image. However, these methods have several limitations. First, it requires accurate mesh reconstruction to locate the correct 2D pixels. Moreover, when the point is invisible, the feature at the 2D projection cannot represent the real features of that point. For instance, if the ear is occluded by the face, the projection may result in capturing features from the eye or nose instead. As a result, relying solely on the feature at the 2D projection can lead to incomplete or misleading feature descriptors.

Vertex Feature as Tokens To tackle the aforementioned problem, we propose a solution wherein we treat each vertex as an individual query token and leverage the attention mechanism of transformers to acquire its corresponding features from the image feature tokens. This approach avoids the need for a fixed 2D projection and allows for more flexible learning. Specifically, we employ N learnable embedding vectors $\mathbf{X_v} \in \mathbb{R}^{N \times C'}$ to represent the feature descriptors associated with each vertex in canonical space and name it as *Vertex Tokens*, where C' is the number of channels. To further encode the location information of each vertex, we incorporate the sine positional encoding [165] to its corresponding image space coordinates (u^s, v^s) and depth d^s, denoting as $\mathbf{E^s_{uv}}$ and $\mathbf{E^s_{dep}}$, respectively. Finally, the vertex query token is defined as $\tilde{\mathbf{X}}_\mathbf{v} = \mathbf{X_v} + \mathbf{E^s_{uv}} + \mathbf{E^s_{dep}}$. On the other hand, we train a CNN encoder $\mathcal{E}(\cdot)$ to extract feature maps from the source image $\mathbf{I^s}$ and flatten the 2D features into a sequence of tokens $\mathbf{F^s} = \mathcal{E}(\mathbf{I^s}) \in \mathbb{R}^{hw \times C'}$. We also apply the 2D sine positional encodings [37] to encode spatial information, denoted as $\mathbf{E}$. Finally, the image token is defined as $\mathbf{X}^{\mathbf{s}}_{\mathbf{F}} = \mathbf{F}^{\mathbf{s}} + \mathbf{E}$.

Transformers The input to the transformer is the concatenation of both image tokens $\mathbf{X}^{\mathbf{s}}_{\mathbf{F}}$ and vertex tokens $\tilde{\mathbf{X}}_{\mathbf{v}}$, i.e, $\mathbf{X} = [\tilde{\mathbf{X}}_{\mathbf{v}}, \mathbf{X}^{\mathbf{s}}_{\mathbf{F}}] \in \mathbb{R}^{(N+hw)\times C'}$. The standard transformer encoder layer [165] consists of alternating layers of the multi-headed self-attention (MHSA) and multi-layer perceptron (MLP). First, three linear projections are applied to transfer $\mathbf{X}$ into three matrices of equal size, namely the query $\mathbf{Q}$, the key $\mathbf{K}$, and the value $\mathbf{V}$. The self-attention is calculated by:

$$\mathrm{SA}(\mathbf{X}) = \mathrm{Softmax}(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{D}})\mathbf{V}, \tag{4.4}$$

For MHSA, H self-attention modules are applied to $\mathbf{X}$ separately, and each of them produces an output sequence. We utilize the state of the vertex tokens at the output of the transformer encoder and employ a linear transformation to modify its dimensionality, thereby acquiring the vertex descriptor $\mathbf{V}_{\mathbf{F}} \in \mathbb{R}^{N\times C}$.

The vertex-feature transformer has several benefits. Firstly, it eliminates the need for a fixed 2D projection to determine the corresponding feature for each vertex. Instead, it leverages attention mechanisms to identify the relevant feature, introducing a higher degree of flexibility. The transformer incorporates positional encoding to encode location information, further enhancing its adaptability and versatility. Additionally, the global attention mechanism of transformers facilitates long-range correspondence among all vertex features. Even when the projection of a vertex is occluded, the vertex feature can still be obtained from neighboring regions or symmetrical vertices.

4.4.4 Neural Vertex Rendering

Given the learned vertex feature $\mathbf{V_F}$, we further use neural point-based rendering to generate synthetic images. During the training, we use the driven vertex $\mathbf{V}^d$ to reconstruct the driven image. In detail, we first project the driven vertices $\mathbf{k}^d \in \mathbf{V^d}$ into image space with the driven camera parameter c^d, i.e., $(u^d, v^d, d^d) = \Pi(\mathbf{k}^d, c^d)$. Subsequently, we create the vertex projection features $\mathbf{P^d_F} \in \mathbb{R}^{H \times W \times C}$. For each vertex $\mathbf{k}^d$, along with its corresponding descriptor $\mathbf{v_F} \in \mathbb{R}^C$, we assign the descriptor to location (u^d, v^d) in the vertex projection features [127]:

$$\mathbf{P^d_F}[\lfloor u^d \rfloor, \lfloor v^d \rfloor] = \mathbf{v_F} \tag{4.5}$$

We keep the features of the nearest vertex when two vertices are projected into the same pixel on $\mathbf{P^d_F}$. For all pixels without projection (i.e., the background pixel), we assign a constant value. Similarly, we also project the depth d^d value into a depth image P_D which satisfies $\mathbf{P^d_D}[\lfloor u^d \rfloor, \lfloor v^d \rfloor] = d^d$. Finally, we concatenate $\mathbf{P^d_F}$ and $\mathbf{P^d_D}$ and employ a U-Net $\mathcal{G}(\cdot)$ to generate the synthetic image $\hat{\mathbf{I^d}}$ as well as the binary foreground mask $\hat{\mathbf{M^d}}$, i.e.,

$$(\hat{\mathbf{I^d}}, \hat{\mathbf{M^d}}) = \mathcal{G}([\mathbf{P^d_F}, \mathbf{P^d_D}]). \tag{4.6}$$

4.4.5 Training

During the training time, we randomly sample $\mathbf{I^s}$ and $\mathbf{I^d}$ from the same video. We fixed the pre-trained DECA and only update the parameters of vertex-feature transformers and neural render. Following [83], we use the L1 loss L_{L1}, VGG perceptual loss L_{vgg} [76], face recognition loss L_{id} [13], and adversarial loss [53, 168] L_a to measure the difference between the reconstructed driven image $\hat{\mathbf{I^d}}$ and the ground truth $\mathbf{I^d}$. We use the Dice loss to match the predicted segmentation masks. The total loss is calculated by:

$$L = \lambda_{L1} L_{L1} + \lambda_{vgg} L_{vgg} + \lambda_{id} L_{id} + \lambda_{seg} L_{seg} + \lambda_a L_a \tag{4.7}$$

, where λ_{L1}, λ_{vgg}, λ_{id}, λ_{seg} and λ_a is the corresponding weights of each loss term.

4.5 Experiments

4.5.1 Experimental Set up

Dataset For a fair comparison with previous works, we conduct experiments on VoxCeleb1 [117] and VoxCeleb2 [33]. VoxCeleb1 contains around 20k video sequences of over 1000 actors and VoxCeleb2 contains around 150k videos of over 6000 actors. Note that, ROME [83] carefully selects a subset of around 15k high-quality video sequences from VoxCeleb2 for training and evaluation, which is not publicly available. We directly use all VoxCeleb2 videos instead. Following [143], each frame is cropped into 256 × 256 and normalized to $[-1, 1]$. We follow the identity-based split thus all subjects in the validation set are unseen by the model. Besides, we apply an off-the-shelf face parsing network [198] to obtain the foreground mask of each frame, which is considered as the pseudo ground truth.

Implementation Details We use the same CNN encoder $\mathcal{E}(\cdot)$ as in ROME [83], which downsamples 16× of the original image. Naturally, our vertex-feature transformer is able to process arbitrary sizes of mesh. However, due to the quadratic computation complexity w.r.t. the sequence length of the transformer, it's hard to model all $N = 5023$ tokens. Thus, we use the coarse mesh of the FLAME model with $N' = 314$ tokens in our vertex-feature transformer and use the decoder of Spiralnet++ [52] to upsample the vertex features after the transformer, which serializes the neighboring vertices based on triangular meshes. Our vertex-feature transformer has 6 transformer encoder layers and the head of MHSA is set to 4. The feature dimension is set to $C' = 128$ and $C = 32$. Our model is implemented using PyTorch and optimized with the Adam optimizer [85] for a duration of 200 epochs. The learning rate is set to $1e-4$ and the batch size is set to 16. λ_{L1}, λ_{vgg}, and λ_{seg} are set to 1.0, and λ_{id} and λ_a are set to 0.1.

Metrics Following previous works [83], we evaluate our CVTHead on both self-reenactment and cross-identity reenactment. In self-reenactment, the source and driving image come from the same video. In this scenario, the driving image can be viewed as the ground truth. We use the following metrics to measure the reconstruction quality between the driving image and the synthesized results: (1)L1 loss on the masked region; (2) peak signal-to-noise ratio (PSNR); (3) learned perceptual image patch similarity (LPIPS) with pre-trained AlexNet [208], and (4) multi-scale structured similarity (MS-SSIM). In the cross-identity reenactment, the source and driven image come from different subjects. Given the source image of one subject, We random sample a different subject in the validation set as the driving image. This evaluation requires the model to fully disentangle the identity and expression information. Since ground truth is unavailable, this task can only be evaluated by some proxy metrics. In detail, we use (1) FID [63] to evaluate the image realism; (2) CSIM [205], which measures the cosine similarity of the identity embeddings from a pre-trained model between the source image and the synthesized image; and (3) image quality assessment (IQA) [150]

4.5.2 Results of talking-face synthesis

We first evaluate the performance of our method on talking-face synthesis. To the best of our knowledge, ROME [83] is the only method that share the same setting with our method, i.e., one-shot mesh-based face reenactment based on graphics without warping field. Thus, we mainly compare our method with ROME [83]. Besides, we also compare with warping-based methods including First-Order Motion Model (FOMM) [143] and the Bi-Layer [204].

Self-reenactment The quantitative comparison results are summarized in Table 4.1. It is noteworthy that our CVTHead achieves comparable performance with previous methods over all metrics. Figure 4.3 illustrates the qualitative comparisons. We also add on the predicted soft mask as in [83] to compare its quality. The first three rows showcase scenarios with minimal head rotations and predominantly frontal source images. In such cases, both ROME and CVTHead exhibit similar performance. However, when the source images depict side views while the driving images present frontal views, ROME tends to generate images with blurry foreground masks in the occluded regions of the source image. Furthermore, ROME often renders these concealed areas in darker colors. These observations indicate that ROME struggles to effectively learn the features of occluded regions and fails to capture the correspondence between mesh vertices. Conversely, our CVTHead addresses these limitations by leveraging transformers to capture long-range dependencies among vertices.

Dataset	VoxCeleb1			
Method	L1 ↓	PSNR ↑	LPIPS ↓	MS-SSIM ↑
FOMM [143]	0.048	22.43	0.139	0.836
Bi-Layer [204]	0.050	21.48	0.108	0.839
ROME [83]	0.048	21.13	0.116	0.838
Ours	0.041	22.09	0.111	0.840

Dataset	VoxCeleb2			
Method	L1 ↓	PSNR ↑	LPIPS ↓	MS-SSIM ↑
FOMM [143]	0.059	20.93	0.165	0.793
ROME [83]	0.050	20.75	0.117	0.834
Ours	0.042	21.37	0.119	0.841

Table 4.1: Results of self-reenactment on the VoxCeleb1 and VoxCeleb2 (↑ means larger is better, ↓ means smaller is better.)

Cross-identity Reenactment We proceed to evaluate our method in comparison to other methods for cross-identity reenactment. The quantitative comparison results are presented in Table 4.2. Strikingly, we achieve similar performance on the assessed metrics as ROME,

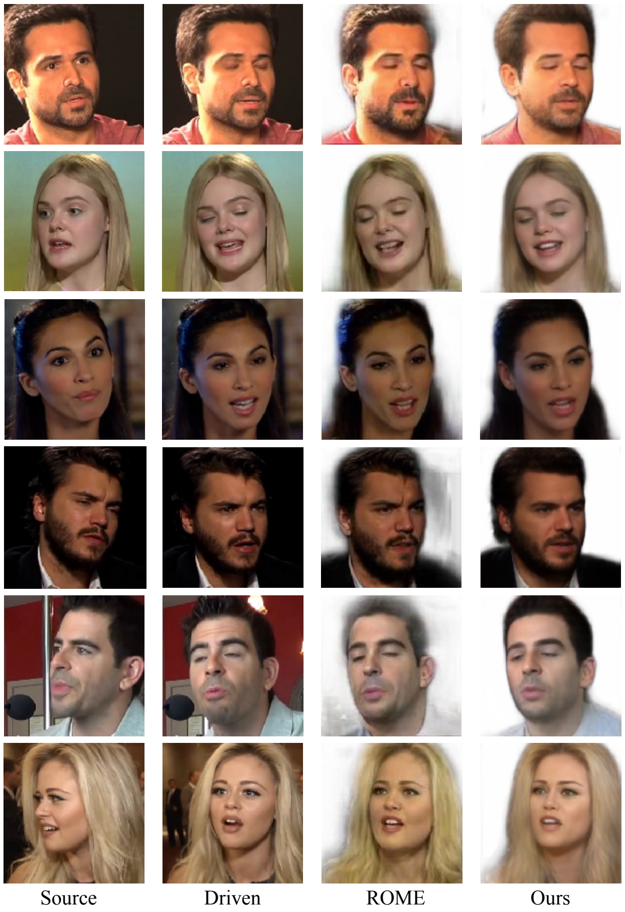


Figure 4.3: Qualitative comparisons of self-reenactment on VoxCeleb1. The 1st column is the source image. The 2nd column is the driving image, which can be considered as the ground truth. The 3rd column is the results from ROME, and the 4th column is the result from our CVTHead.

indicating the effectiveness of our method in cross-identity reenactment tasks. Furthermore, we provide qualitative results in Figure 4.4, showcasing the ability of our method to generate images with desired expressions, head poses, and other attributes. Notably, warping-based methods usually cannot maintain the identity information such as face shape from the source image. For mesh-guided methods, ROME tends to generate lower-quality images when local regions are occluded in the source image. In contrast, our method demonstrates the capability to maintain the quality of all local regions even in such challenging scenarios.

Dataset	VoxCeleb1			
Method	FID ↓	CSIM ↑	IQA ↑	FPS ↑
FOMM [143]	39.69	0.592	37.00	64.3
Bi-Layer [204]	43.8	0.697	41.4	20.1
ROME [83]	29.23	0.717	39.11	12.9
Ours	25.78	0.675	42.26	24.3

Dataset	VoxCeleb2			
Method	FID ↓	CSIM ↑	IQA ↑	FPS ↑
FOMM [143]	61.28	0.624	36.20	64.3
ROME [83]	53.52	0.729	37.34	12.9
Ours	48.48	0.712	40.27	24.3

Table 4.2: Results of cross-identity reenactment.

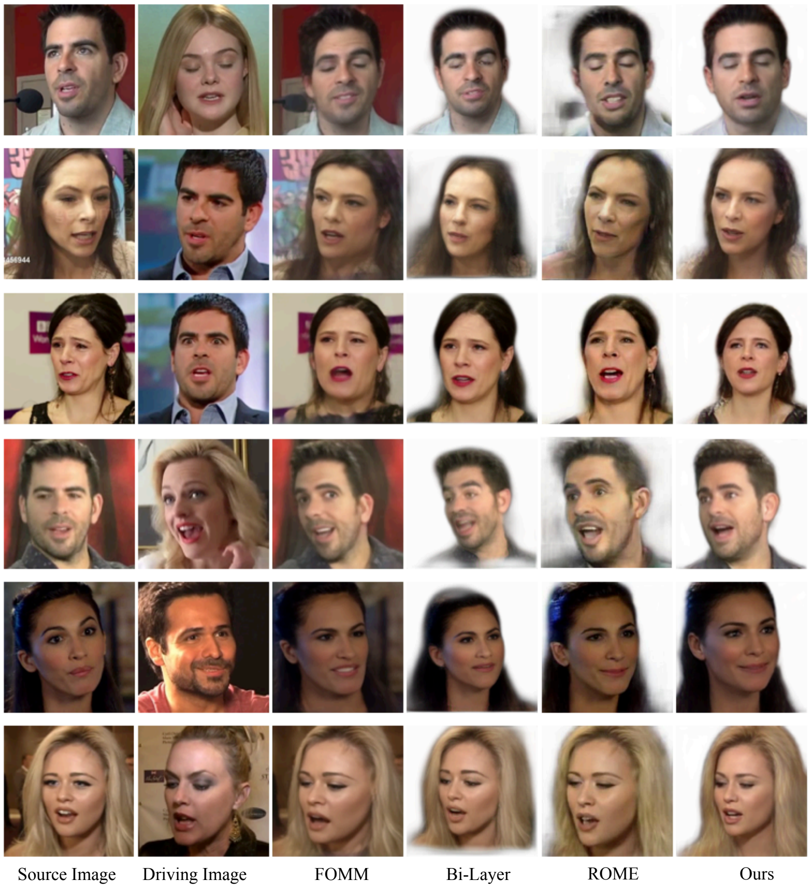


Figure 4.4: Qualitative comparisons of cross-identity reenactment on VoxCeleb1.

Inference time comparison We also evaluate the inference time of each model, considering the complete duration of 3D mesh reconstruction, the vertex deformation model, and the rendering process. To provide a comprehensive analysis, we report the average FPS

(Frames per Second) based on 1000 runs performed on a single RTX 3090Ti. The results are presented in the last column of Table 4.2. Notably, warping-based method is more efficient as they don't need the tedious rendering and mesh reconstruction. ROME achieves a modest 12.9 FPS, while our CVTHead model achieves a significantly higher rate of 24.3 FPS. This outcome highlights the superior efficiency of the point-based neural rendering approach compared to traditional graphic-based rendering methods.

4.5.3 Results of 3DMM-based Face Animation

After obtaining the vertex descriptors using the vertex feature transformer, the resulting face can be further manipulated by adjusting the coefficients of the FLAME model [95], which control expression ϕ, pose θ, face shape β, and camera views c. The ability to explicitly control these coefficients enables us to generate faces of the same subject with different expressions, face shapes, and camera views, as illustrated in Figure 4.5. This result demonstrates that the learned feature descriptors exhibit a strong alignment with the vertices in the canonical space. Consequently, neural point-based rendering can serve as a viable alternative to traditional graphic-based rendering methods. Moreover, we intentionally select two distinct source images of the same subject. Interestingly, the generated images, utilizing vertex features from these distinct sources, exhibit a striking resemblance. This intriguing observation further underscores the effectiveness and robustness of our method.

4.5.4 Ablation Studies

Vertex deformation We utilize the linear deformation model $f_H(\cdot)$ from ROME [83] to deform the vertices of the hair and shoulder region. In this study, we conduct an ablation experiment where we train CVTHead without this vertex deformation module, instead employing the default FLAME mesh with a bald head. The results presented in Table 4.3

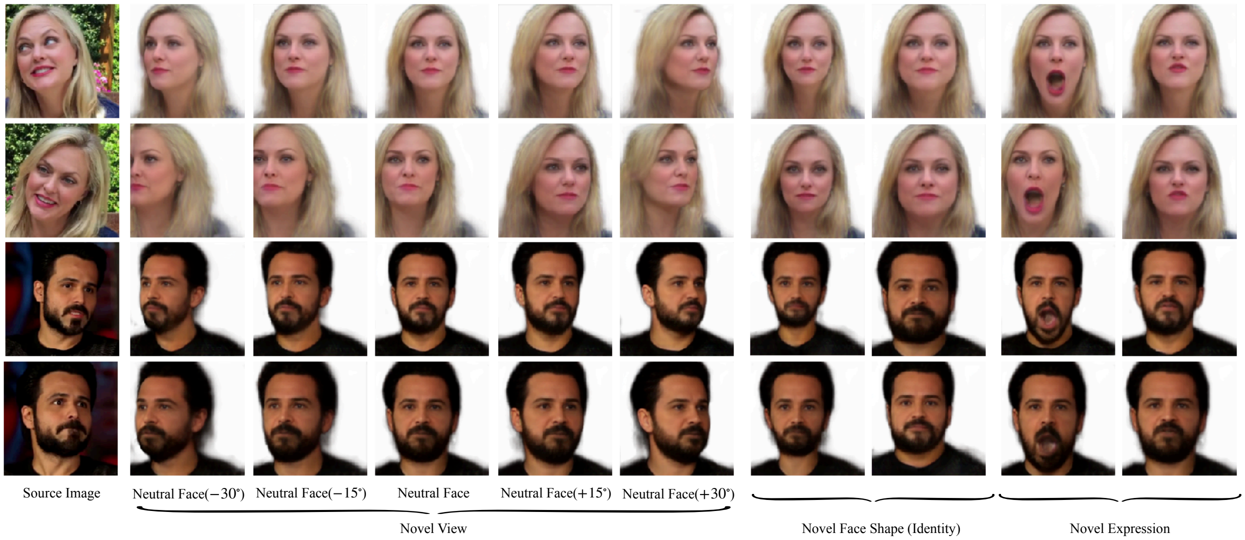


Figure 4.5: Qualitative results of face animation with novel views, novel face shapes (identity), and novel expressions.

demonstrate that the removal of the vertex deformation ("D." in short) has only a minor impact on the performance. Interestingly, Figure 4.6 reveals that the synthesized images from CVTHead, both with and without vertex deformation, appear nearly identical. Furthermore, even in cases where the subject has fluffy or long hair that extends beyond the head area, the absence of vertex deformation in CVTHead does not hinder its ability to generate the correct hairstyle. These results indicate that the local vertex descriptor can effectively capture the necessary features.

Method	L1 ↓	PSNR ↑	LPIPS ↓	MS-SSIM ↑
CVTHead (w/o D.)	0.041	22.47	0.121	0.842
CVTHead	0.041	22.09	0.111	0.840

Table 4.3: Ablation study on the vertex deformation module. We evaluate the performance of self-reenactment on the VoxCeleb1.

Method	L1 ↓	PSNR ↑	LPIPS ↓	MS-SSIM ↑
Pixel-aligned features	0.045	21.81	0.107	0.841
CVTHead	0.041	22.09	0.111	0.840

Table 4.4: Ablation study on the pixel-aligned features

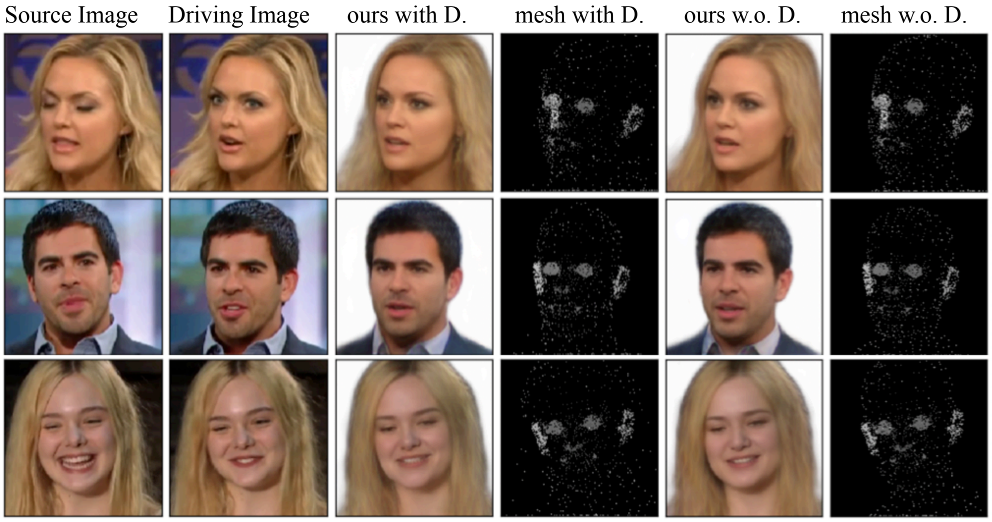


Figure 4.6: Ablation study of CVTHead with and without the vertex deformation model for hair and shoulder region (D. in short)

Pixel-aligned features In our work, we design the vertex feature transformers to learn the vertex feature. In this study, we consider the pixel-aligned features as the baseline, which project the 3D vertex into 2D and choose the corresponding pixel from the image. We follow the architecture design in S3F [34] and use a UNet-like feature extractor and sample features of each vertex with its corresponding 2D projection. Table 4.4 indicates that this approach yields a marginally lower PSNR, but a slightly improved LPIPS score. As shown in Figure 4.7, this design can maintain more detailed local features such as hair due to the high-resolution features. Thus, a slightly better LPIPS is achieved. However, when the point is occluded in the source image, the synthesized image tends to generate blur and shadow in these areas if they are visible in the driving pose, which is the reason of the worse PSNR. These results suggest that pixel-aligned methods cannot capture the correct features due to the ambiguity of depth. In this case, when a large head rotation happens, this method encounters the same issue as warp-based methods.

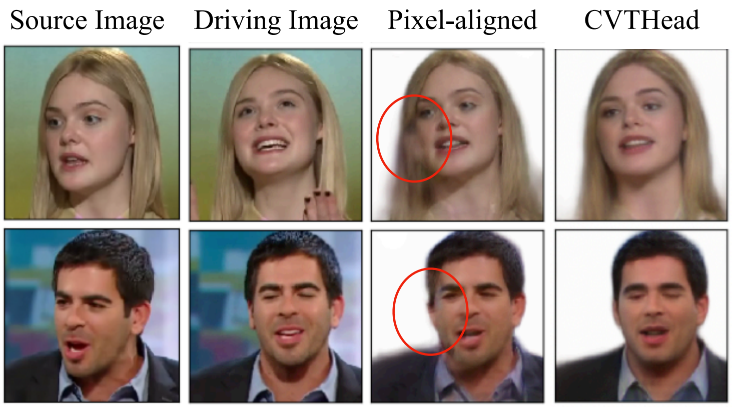


Figure 4.7: Ablation study of vertex features.

4.6 Limitations

While our method demonstrates effective face animation capabilities from a single image, one potential limitation is that the performance of our approach heavily relies on the accuracy of the 3D mesh reconstruction, specifically utilizing DECA [46] in our setup. In certain challenging scenarios, DECA may struggle to fully disentangle the shape and expression factors from the driving images. Consequently, CVTHead may generate images that differ in expressions or head poses from the intended outcome. This highlights the need for further advancements in the accuracy and robustness of 3D mesh reconstruction techniques to address such limitations.

4.7 Conclusion

We propose a novel approach for generating explicitly controllable head avatars from a single reference image, utilizing point-based neural rendering. We treat the sparse vertices of the head mesh as a point set and leverage the vertex-feature transformer to learn the local feature descriptor for each vertex. Through our research, we demonstrate that point-based rendering can effectively replace traditional graphic-based rendering methods, offering enhanced efficiency. Moreover, we envision that our method can be seamlessly integrated with various generative tools, such as diffusions, to further enhance the quality of generated images and we consider this as future work.

Chapter 5

MaskINT: Video Editing via Interpolative Non-autoregressive Masked Transformers

Recent advances in generative AI have significantly enhanced image and video editing, particularly in the context of text prompt control. State-of-the-art approaches predominantly rely on diffusion models to accomplish these tasks. However, the computational demands of diffusion-based methods are substantial, often necessitating large-scale paired datasets for training, and therefore challenging the deployment in practical applications. This study addresses this challenge by breaking down the text-based video editing process into two separate stages.

In the first stage, we leverage an existing text-to-image diffusion model to simultaneously edit a few keyframes without additional fine-tuning. In the second stage, we introduce an efficient model called MaskINT, which is built on non-autoregressive masked generative transformers and specializes in frame interpolation between the keyframes, benefiting from structural guidance provided by intermediate frames. Our comprehensive set of experiments illustrates the efficacy and efficiency of MaskINT when compared to other diffusion-based methodologies. This research offers a practical solution for text-based video editing and showcases the potential of non-autoregressive masked generative transformers in this domain.

5.1 Problem Definition

The task of text-based video editing can be formulated as a conditional image generation task. Given the input video frames $\{\mathbf{I}^i\}_{i=1}^N$ and the edit text prompt τ, this task aims to learn a mapping $f(\cdot)$ to generate video frames $\{*\mathbf{I}^i\}_{i=1}^N$ that maintain the same spatial layout as the inputs $\{\mathbf{I}^i\}_{i=1}^N$, and have the appearance based on τ, i.e.,

$$\{\mathbf{I}^i\}_{i=1}^N = f(\{\mathbf{I}^i\}_{i=1}^N, \tau) \tag{5.1}$$

Figure 5.1 show some demos for this problem. In detail, we can change the appearance of the dog in the video into a wolf, or make the video of the car in cartoon style.

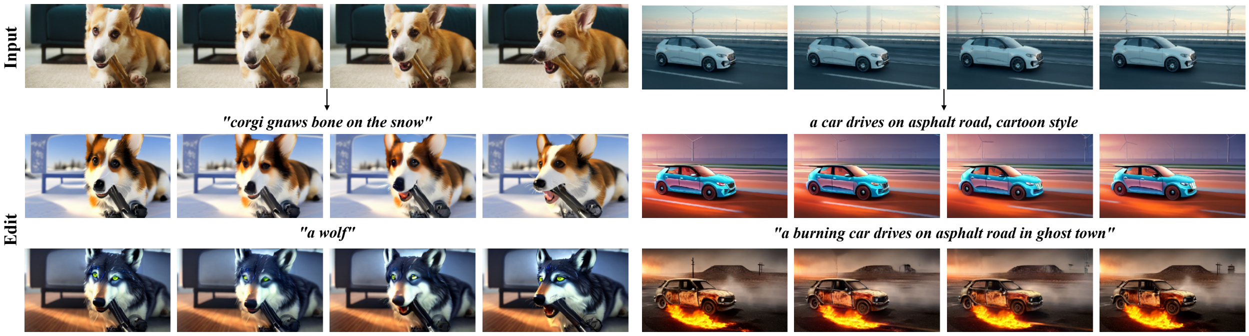

Input
"corgi gnaws bone on the snow"
a car drives on asphalt road, cartoon style
Edit
"a wolf"
"a burning car drives on asphalt road in ghost town"

Figure 5.1: Examples of video editing with MaskINT.

5.2 Introduction

Text-based video editing, which aims to modify a video's content or style in accordance with a provided text description while preserving the motion and semantic layout, plays an important role in a wide range of applications, including advertisement, live streaming, and the movie industry, etc. This challenging task requires that edited video frames not only match the given text prompt but also ensure the consistency across all video frames.

Recently, numerous studies have showcased the impressive capabilities of diffusion models [66] in the domain of text-to-image and text-to-video generation [136, 138, 9, 146]. Later on, built upon Stable Diffusion (SD) [136], several works have achieved remarkable success in the realm of text-based image editing [207, 11, 164]. When extending to text-based video editing, existing works can be mainly divided into two ways: One is to train diffusion models with temporal modules on paired text-video datasets [65, 42]. However, due to the lack of extensive text-to-video datasets, these works typically struggle to achieve the same level of editing expertise seen in the realm of image editing. The other involves leveraging a pre-trained text-to-image diffusion models in a training-free techniques [50, 128, 82, 212]. These works usually extend the self-attention across all frames to achieve an overall temporal consistency. However, this attention-based temporal constraint remains implicit and suboptimal. Moreover, while diffusion-based techniques are capable of producing high-fidelity videos, the use of diffusion models to produce all video frames proves to be a highly time-consuming process. The integration of the global attention across all frames in these video editing methods further extends the processing time, rendering them less practical for real-world applications.

Meanwhile, studies indicate that non-autoregressive masked generative transformers [20, 19, 201, 56] can attain similar levels of performance in generating images or videos compared to diffusion-based methods, while bring significant efficiency [19]. These works first tokenize image or videos into a sequence of discrete tokens [43], and then train transformers [165, 36] with masked token modeling to predict these tokens. During the inference time, they employ non-autoregressive decoding, which generates all tokens in parallel and iteratively refine predictions in a few steps. Nonetheless, extending these techniques to perform global editing tasks, such as stylization, presents a formidable challenge. This type of task requires the replacement of nearly all tokens with new ones, as opposed to merely modifying tokens within a localized region.

In this work, we disentangle text-based video editing into two separate stages. In the first stage, we utilize existing text-based image editing models to jointly edit only two keyframes (i.e., the initial and last frames) from the video, guided by the provided text prompt. In the second stage, we propose a novel Interpolative Non-autoregressive generative Transformers (MaskINT), which performs structure-aware frame interpolation by leveraging both the color information of the initial and final frames, as well as structural cues like edge or depth maps from intermediate frames. Through disentanglement of keyframe editing and frame interpolation into separate stages, our pipeline eliminates the requirement for paired video datasets during training, thereby enabling us to train the MaskINT using video-only datasets. Furthermore, thanks to its non-autoregressive decoding, MaskINT significantly accelerates the generation of intermediate frames compared to using diffusion models for this purpose. We show that our method balances the trade off between quality and efficiency, offering comparable performance with existing diffusion methods yet taking much less time in generation.

Our major contributions are summarized as follows:

- We propose to disentangle the text-based video editing into a two stage pipeline, that involves keyframes joint editing using existing image diffusion model and structure-aware frame interpolation with masked generative transformers trained on video only datasets.

- We propose MaskINT to perform structure-aware frame interpolation, which is the pioneer work that explicitly introduces structure control into non-autoregressive generative transformers.
- Experimental results demonstrate that our method achieves comparable performance with diffusion methods in terms of temporal consistency and alignment with text prompts, while providing 5-7 times faster inference times.

5.3 Related Work

Generative Transformers. Following GPT [12], many pioneer works [43, 200, 92, 49, 67] tokenize images/videos into discrete tokens, and train *Autoregressive Generative Transformers* to perform image/video generation, where tokens are generated sequentially based on previous output. However, these autoregressive methods become exceedingly time-consuming when the length of the token sequence increases. Recently, *Non-autoregressive Generative Transformers*, capable of simultaneously generating all tokens in parallel, have emerged as efficient solutions [20, 201]. Specifically, MaskGiT [20] first shows the capability and efficiency of this technique in image generation. It can be seamlessly extended to tasks like inpainting and extrapolation by applying various initial mask constraints. Muse [19] achieves state-of-the-art performance in text-to-image generation by training on large-scale text-image datasets and brings significantly efficiency improvement. StyleDrop [147] further finetunes Muse with human feedback to perform text-to-image generation guided with a reference style image. Furthermore, MaskSketch [6] introduces implicit structural guidance into MaskGiT by calculating the similarity of attention maps in the sampling step. Nevertheless, this implicit structure condition is suboptimal.

In video generation, MaskViT [56] employ 2D tokenizer and trains a bidirectional window transformer to perform frame prediction. Phenaki [166] trains a masked transformer to generate short video clips condition on text prompt and extends it to arbitrary long video with different prompts in an autoregressive way. MAGVIT [201] utilizes 3D tokenizer to quantize videos and trains a single model to perform multiple video generation tasks such as inpainting, outpainting, frame interpolation, etc. However, to the best of our knowledge, there is currently no existing literature in the field of text-based video editing utilizing masked generative transformers. Besides, there is a notable absence of research that delves into explicit structural control within this area.

Diffusion Models in Image Editing. Leveraging the advancements of SD [136], numerous studies have achieved significant success in the field of text-based image editing [207, 116, 164, 79, 62, 213, 124, 113, 35, 184]. For example, ControlNet [207], T2I-Adapter [116], and Composer [69] finetune SD with spatial condition such as depth maps and edge maps, enabling text-to-image synthesis with the same structure as the input image. The PNP [164] incorporates DDIM inversion features [148] from the input image into the text-to-image generation process alongside SD, enabling image editing without the necessity of additional training or fine-tuning. Instructpix2pix [11] trains a conditional diffusion model for text-guided image editing using synthetic paired examples, which avoid the tedious inversion. Nevertheless, employing these methods on each video frame independently often leads to inconsistencies and flickering.

Diffusion Models in Video Editing. Recently, diffusion models also dominate the field of video generation and video editing [146, 65]. For example, Gen-1 [42] trains a video diffusion models on paired text-video datasets to generate videos with both depth map and text prompts. Meanwhile, several works utilize pre-trained image diffusion models to achieve video editing in a training-free way [179, 192, 128, 82, 212, 171, 17]. To enable a cohesive global appearance among edited frames, a common approach in these studies involves extending the attention module of SD to encompass multiple frames and conducting cross-frame attention. In detail, Text2Video-Zero [82] performs cross-frame attention of each frame on the first frame to preserve appearance consistency. ControlVideo [212] extends ControlNet with fully cross-frame attention to joint edit all frames and further improves the performance with interleaved-frame smoother. TokenFlow [50] enhance PNP [164] with extended-attention to jointly edit a few keyframes at each denoising step and propagate them throughout the video based on nearest-neighbor field. On the contrary, we only utilize existing image diffusion models to edit two keyframes, rather than all video frames.

Video Frame Interpolation (VFI). VFI aims to generate intermediate images between a pair of frames, which can be applied to creating slow-motion videos and enhancing refresh rate. Advanced methods typically entail estimating dense motions between frames, like optical flow, and subsequently warping the provided frames to generate intermediate ones [120, 75, 144, 105, 134, 72]. However, these methods are most effective with simple or monotonous motion. Thus, they cannot be directly applied to the second stage. In our work, we perform frame interpolation by incorporating additional structural signals.

5.4 Preliminaries

5.4.1 Masked Generative Transformers

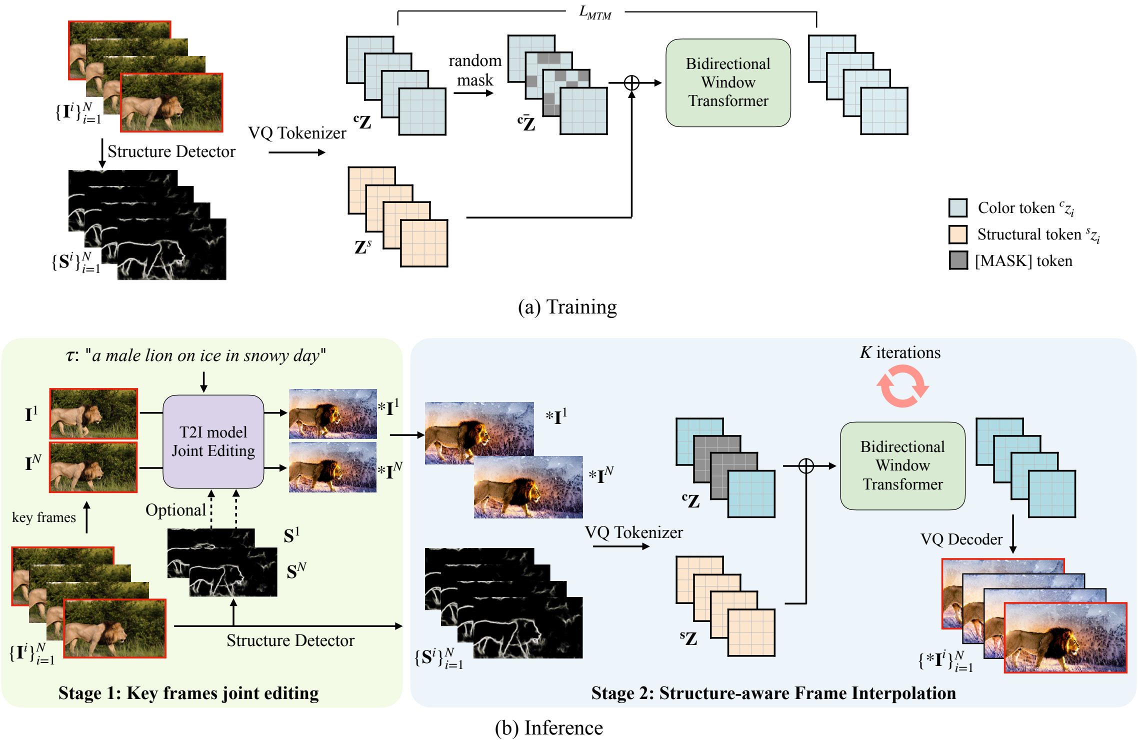
Masked generative transformers [20] follow a two-stage pipeline. In the first stage, an image is quantized into a sequence of discrete tokens via a Vector-Quantized (VQ) auto-encoder [43]. In detail, given an image $\mathbf{I} \in \mathbb{R}^{H \times W \times 3}$, an encoder $\mathcal{E}$ encodes it into a series of latent vectors and discretize them through a nearest neighbour look up in a codebook of quantized embeddings with size M. To this end, an image can be represented with a sequence of codebook's indices $\mathbf{Z} = [z_i]_{i=1}^{h \times w}, z_i \in \{1, 2, ..., M\}$, where h and w is the resolution of latent features. A decoder $\mathcal{D}$ can reconstruct the indices back to image $\mathcal{D}(\mathbf{Z}) \approx \mathbf{I}$. In the second stage, a bidirectional transformer model [165] is learned with Masked Token Modeling (MTM). Specifically, during training, a random mask ratio $r \in (0, 1)$ is selected and $\lfloor \gamma(r) \cdot h \times w \rfloor$ tokens in Z are replaced with a special `[MASK]` token, where $\gamma(r)$ is a mask scheduling function [20]. We denote the corrupted sequence with masked tokens as $\bar{\mathbf{Z}}$ and conditions such as class labels or text prompt as $\mathbf{c}$. Given the training dataset $\mathbb{D}$, a BERT [36] parameterized by Φ is learned to minimize the cross-entropy loss between the predicted and the ground truth token at each masked position:

$$\mathcal{L}_{MTM} = \mathop{\mathbb{E}}_{\mathbf{Z} \in \mathbb{D}} \left[\sum_{\bar{z}_i = \texttt{[MASK]}} - \log p_{\Phi}(z_i | \bar{\mathbf{Z}}, \mathbf{c}) \right]. \tag{5.2}$$

During inference time, non-autoregressive decoding is applied to generate images. Specifically, given the conditions $\mathbf{c}$, all tokens are initialized as `[MASK]` tokens. At step k, all tokens are predicted in parallel while only tokens with the highest prediction scores are kept. The remaining tokens with least prediction scores are masked out and regenerated in the next iteration. The mask ratio is determined by $\gamma(\frac{k}{K})$ at step k, where K is the total number of iteration steps.

5.4.2 ControlNet

Latent Diffusion Models Denoising Diffusion Probabilistic Models (DDPM) [66] generate images through a progressive noise removal process applied to an initial Gaussian noisy image, carried out over a span of $T(>> K)$ time steps. To enable efficient high-resolution image generation, Latent Diffusion models [136] operates the diffusion process in the latent space of an autoencoder. First, an encoder $\mathcal{E}'$ compresses an image $\mathbf{I}$ to a low-resolution latent code $x = \mathcal{E}'(\mathbf{I}) \in \mathbb{R}^{h \times w \times c}$. Second, a U-Net ϵ_θ with attention modules [165] is trained



(a) Training

(b) Inference

Figure 5.2: Overview of MaskINT. MaskINT disentangle the video editing task into two separate stages, i.e., keyframes joint editing and structure-aware frame interpolation.

to remove the noise with loss function:

$$\mathcal{L}_{LDM} = \mathbb{E}_{x_0,\epsilon\sim N(0,I),t\sim}\|\epsilon - \epsilon_\theta(x_t,t,\tau)\|_2^2, \tag{5.3}$$

where τ is the text prompt and x_t is the noisy latent sample at timestep t. Stable Diffusion is trained on datasets with billion scale text-image pairs, which serves as the foundation model of many generation tasks.

ControlNet In practice, it's challenging to use text prompt to describe the layout of generated image. Furthermore, ControlNet [207] is proposed to provide spatial layout conditions such as edge map, depth map, and human poses. In detail, ControlNet train the same U-Net architecture as SD and finetune it with specific conditions. We denote ControlNet as $\epsilon'_\theta(x_t, t, \tau, s)$, where s is the spatial layout condition.

5.5 Methodology

5.5.1 Overview

Fig. 5.2 shows an overview of our framework. We disentangle the video editing task into keyframe joint editing stage and structure-aware frame interpolation stage. Specifically, given a video clip with N frames $\{\mathbf{I}^i\}_{i=1}^N$, in the first stage, with the input text prompt τ, we simultaneously edit two keyframes, i.e., the initial frame $\mathbf{I}^1$ and last frame $\mathbf{I}^N$, using existing image-editing model $g(\cdot)$ that requires no additional tuning. This frame based joint editing module provides high-quality coherent edited frames ${}^*\mathbf{I}^1, {}^*\mathbf{I}^N = g(\mathbf{I}^1, \mathbf{I}^N, \tau)$ and is highly efficient on a pair of frame. In the second stage, we propose MaskINT to perform structure-aware frame interpolation via non-autoregressive transformers. The entire edited video frames are generated by $\{{}^*\mathbf{I}^i\}_{i=1}^N = f_\Phi({}^*\mathbf{I}^1, {}^*\mathbf{I}^N, \{\mathbf{S}^i\}_{i=1}^N)$, where $\mathbf{S}^i \in [0,1]^{H\times W\times 1}$ is the structural condition (i.e., the HED edge map [183]). Our MaskINT is trained with masked token modeling (MTM) on video only datasets, conditioning on the structural signal $\{\mathbf{S}^i\}_{i=1}^N$ as well as the initial frame $\mathbf{I}^1$ and last frame $\mathbf{I}^N$.

5.5.2 Keyframes Joint Editing

To maintain the structure layout of selected keyframes, we take ControlNet [207] to edit $\mathbf{I}^1$ and $\mathbf{I}^N$ based on text prompt τ as well as their edge maps $\mathbf{S}^1$ and $\mathbf{S}^N$. However, even with the identical noise, applying ControlNet to each keyframes individually, i.e., $\epsilon'_\theta(x_t^1, t, \tau, \mathbf{S}^1)$ and $\epsilon'_\theta(x_t^N, t, \tau, \mathbf{S}^N)$, cannot guarantee the appearance consistency. To address this issue, following previous work [179, 50, 212], we extend the self-attention blocks to simultaneously process two keyframes. In detail, the self-attention block projects the noisy feature map x_t^j of j^{th} frame at time step t into query $\mathbf{Q}^j$, key $\mathbf{K}^j$, value $\mathbf{V}^j$ in the original U-Net of SD. We extend the self-attention block to perform attention across all selected keyframes by concatenating their keys and values and calculate the attention by

$$\text{Softmax}(\frac{\mathbf{Q}^j[\mathbf{K}^1,\mathbf{K}^N]^{\mathbf{T}}}{\sqrt{c}})[\mathbf{V}^1,\mathbf{V}^N] \tag{5.4}$$

Note that although only two frames were used in Eq.5.4, this joint editing can seamlessly generalize to any number of frames, but at the cost of longer processing time and huge demand in resources.

5.5.3 MaskINT

Structural-aware embeddings. Previous works [56, 201] demonstrate that non-autoregressive masked generative transformers can effectively perform frame prediction and interpolation tasks. However, these architectures lack explicit control of structure, making it difficult to follow the motion of original videos. In our work, we explicitly introduce structural condition of each frame into the generation process. Specifically, we tokenize both RGB frames $\{\mathbf{I}^i\}_{i=1}^N$ and structure maps $\{\mathbf{S}^i\}_{i=1}^N$ with an off-the-shelf 2D VQ tokenizer [43]. We utilize 2D VQ rather than 3D VQ [201] to accommodate varying numbers of frames and frame rate without constraints. We denote the tokens from RGB frames as ${}^{\mathbf{c}}\mathbf{Z} = \{{}^c z_i\}_{i=1}^{N\times h\times w}$ (color token) and tokens from edge maps as ${}^{\mathbf{s}}\mathbf{Z} = \{{}^s z_i\}_{i=1}^{N\times h\times w}$ (structure token), where ${}^s z_i$, ${}^c z_i \in \{1, 2, ..., M\}$, where M is the codebook size. Subsequently, two distinct embedding layers $e^C(\cdot)$ and $e^S(\cdot)$ are learned to map token indices ${}^{\mathbf{c}}\mathbf{Z}$ and ${}^{\mathbf{s}}\mathbf{Z}$ into their respective embedding spaces. Learnable 2D spatial positional encoding $\mathbf{P}^{\mathbf{S}}$ and temporal positional encoding $\mathbf{P}^{\mathbf{T}}$ are also added [7]. Thus, the input to the following transformer layers can be formulated by $\mathbf{X} = e^c({}^{\mathbf{c}}\mathbf{Z}) + e^s({}^{\mathbf{s}}\mathbf{Z}) + \mathbf{P}^{\mathbf{S}} + \mathbf{P}^{\mathbf{T}} \in \mathbb{R}^{N\times h\times w\times c}$.

Transformer with Window-Restricted Attention. Previous masked generative transformers [20, 201] employ a pure transformer with global attention [165]. However, Given that there is no substantial motion between consecutive frames, we adopt self-attention within a restricted window to further mitigate computational overhead, following MaskViT [56]. In detail, our approach involves two distinct stages of attention. Initially, we employ spatial window attention, confining attention to tokens within a frame of dimensions $1 \times h \times w$. Subsequently, we extend this to spatial-temporal window attention, which confines attention to tokens within a tube of dimensions $N \times h_w \times w_w$, where h_w and w_w is the window size, where $h_w < h$ and $w_w < w$, which greatly reduce the complexity. Besides, to further reduce computation of transformer, we also add a shallow convolution layers to downsample $\mathbf{X}$ before the transformer encoder layers and an upsample layer at the end.

Training. By fully disentangling keyframes editing and frame interpolation into distinct stages, our model no longer necessitates paired videos for training. Consequently, we can train MaskINT using video only datasets. Denote the color token of i^{th} RGB frame as ${}^{\mathbf{c}}\mathbf{Z}^i = \{{}^{c}z^i\}_{i=1}^{h\times w}$. During the training time, we keep color tokens of the initial frame ${}^{\mathbf{c}}\mathbf{Z}^1$ and last frame ${}^{\mathbf{c}}\mathbf{Z}^N$, and randomly replace $[\gamma(r) \cdot (N-2) \cdot N]$ color tokens of intermediate frames with the `[MASK]` tokens. We denote this corrupted video color tokens as ${}^{\mathbf{c}}\bar{\mathbf{Z}} = \{{}^{\mathbf{c}}\mathbf{Z}^1, {}^{\mathbf{c}}\bar{\mathbf{Z}}^2, ..., {}^{\mathbf{c}}\bar{\mathbf{Z}}^{N-1}, {}^{\mathbf{c}}\mathbf{Z}^N\}$. The structure-aware window-restricted transformer with parameters Θ is trained by

$$\mathcal{L}_{MTM} = \mathop{\mathbb{E}}_{{}^{\mathbf{c}}\mathbf{Z},{}^{\mathbf{s}}\mathbf{Z}\in\mathbb{D}}[\sum_{{}^{c}\bar{z}_i=\texttt{[MASK]}} -\log p_{\Theta}({}^{c}z_i|{}^{\mathbf{c}}\bar{\mathbf{Z}},{}^{\mathbf{s}}\mathbf{Z})] \tag{5.5}$$

Inference. During the inference time, our MaskINT can seamlessly generalize to perform frame interpolation between the jointly edited frames, although it is only trained with regular videos. Specifically, we tokenize the the initial and last edited frames ${}^{*}\mathbf{I}^1$ and ${}^{*}\mathbf{I}^N$ from Stage 1 into color tokens ${}^{\mathbf{c}}_{*}\mathbf{Z}^1$ and ${}^{\mathbf{c}}_{*}\mathbf{Z}^N$, and initialize color tokens of all intermediate frames $\{{}^{\mathbf{c}}\bar{\mathbf{Z}}^2, ..., {}^{\mathbf{c}}\bar{\mathbf{Z}}^{N-1}\}$ with `[MASK]` tokens. We follow the iterative decoding in MaskGiT [20] with a total number of K steps. At step k, we predict all color tokens in parallel and keep tokens with the highest confidence score.

5.6 Experiments

5.6.1 Settings

Implementation Details. We train our model with $100k$ videos from ShutterStock. During training time, we random select a $T = 16$ video clip with frame interval $1, 2, 4$ from each video and resize it to 384×672. We utilize Retina-VQ [41] with 8 downsample ratio, i.e., each frame has 48×84 tokens. We employ Transformer-Base as our MaskINT and optimized it from scratch with the AdamW optimizer [104] for a duration of 100 epochs. The initial learning rate is set to $1e - 4$ and decayed with cosine schedule. During the inference time, we set the number of decoding step K to 32 and the temperature t to 4.5.

Evaluation. Following [179], we use the selected 40 object-centric videos of the DAVIS dataset [126], covering humans, animals, vehicles, etc. Besides, we also select 30 unseen videos from the ShutterStock dataset. For each video, we manually design 5 edited prompts, including object editing, background changes and style transfers. Following previous works [212, 128], we assess the quality of the generated videos using CLIP [130]. In detail, we evaluate 1) Temporal consistency, which calculates the average cosine similarity of all pairs of consecutive frames. 2) Prompt consistency, which calculates the average cosine similarity between given text prompt and all video frames. To evaluate the efficiency, we report the duration required for generating a 16-frame video clip on a single NVIDIA A6000 GPU.

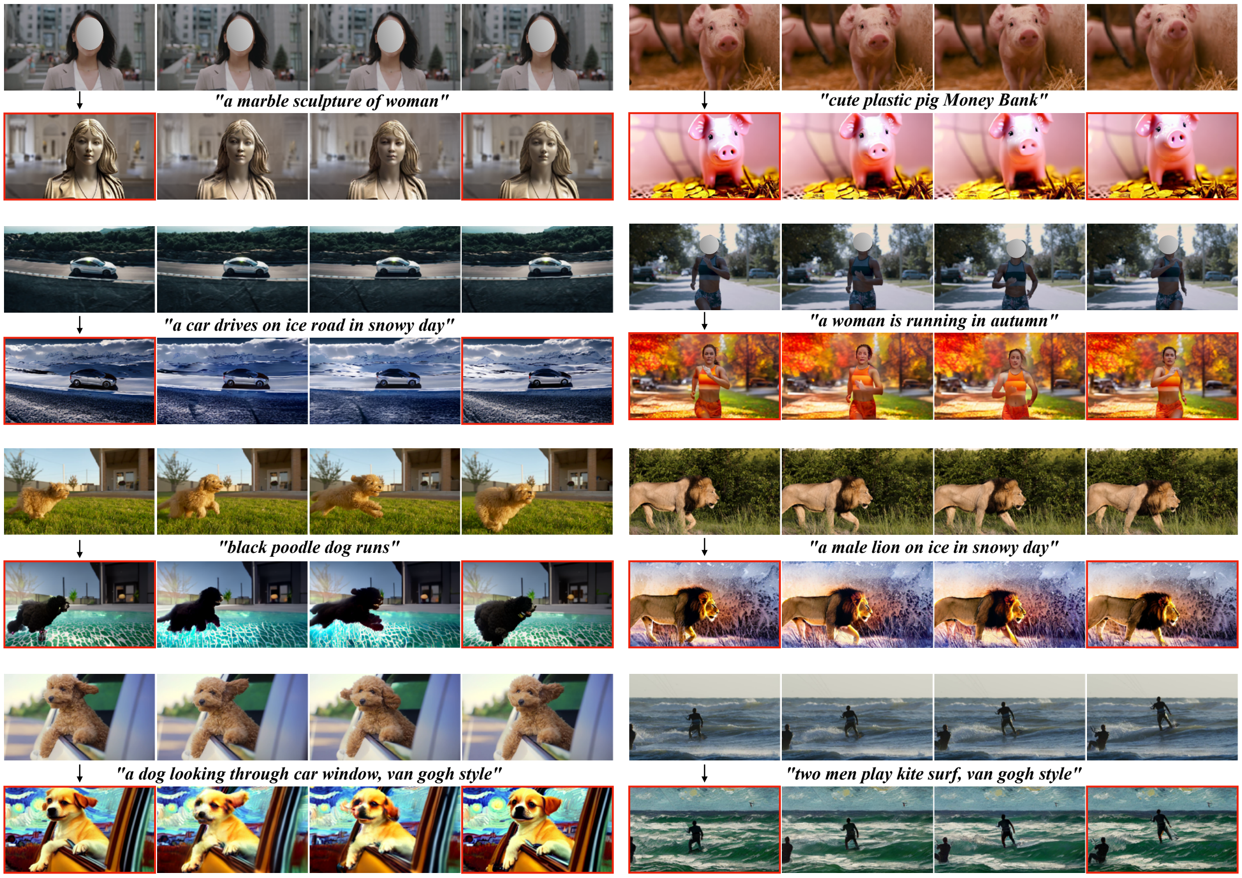

"a marble sculpture of woman"
"cute plastic pig Money Bank"
"a car drives on ice road in snowy day"
"a woman is running in autumn"
"black poodle dog runs"
"a male lion on ice in snowy day"
"a dog looking through car window, van gogh style"
"two men play kite surf, van gogh style"

Figure 5.3: Examples of video editing with MaskINT. Frames with red bounding box are jointly edited keyeframes.

5.6.2 Results

We select methods that built upon text-to-image diffusion models for comparison, including TokenFlow [50], Text-to-video zero [82], and ControlVideo [212]. We also consider apply ControlNet to each frame individually with the same initial noise as baseline. Besides, we also compare with frame interpolation with FILM [134] with the same edited keyframes.

Quantitative Comparisons. Table 5.1 summarize the performance of these methods on both DAVIS and ShutterStock datasets. Notably, our method achieves comparable performance with diffusion methods, in terms of both temporal consistency and prompt consistency, while brings a significant acceleration in processing speed. In detail, MaskINT is

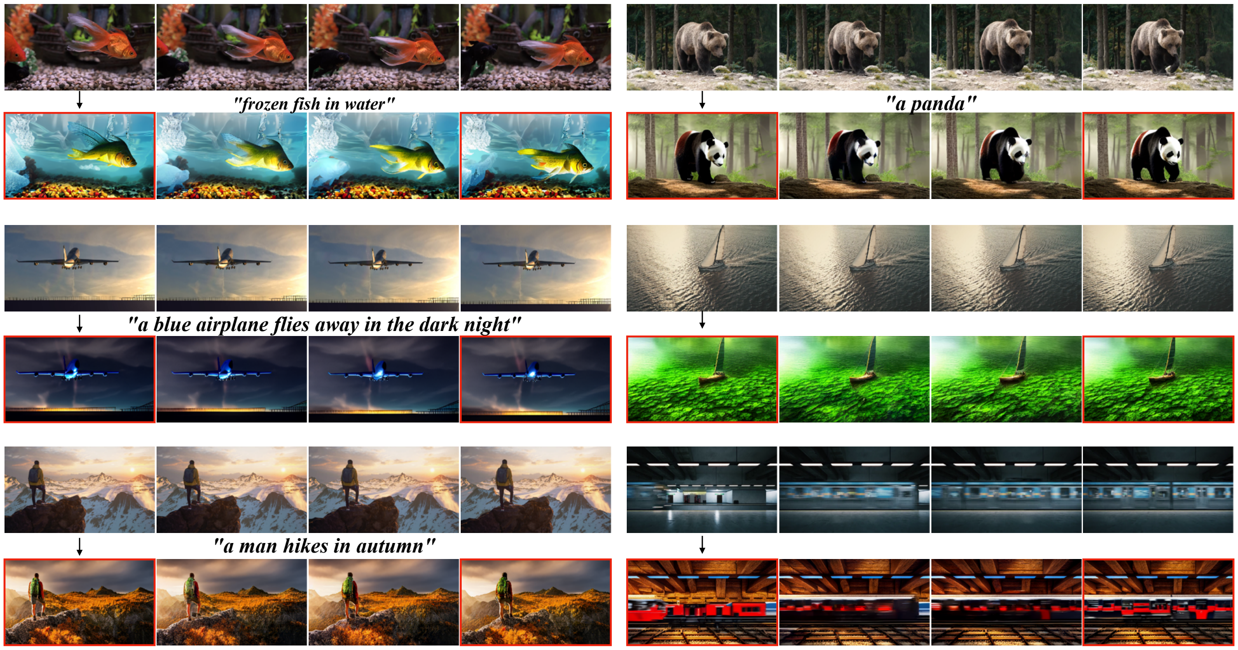


Figure 5.4: Additional editing examples with MaskINT. Frames with red bounding box are jointly edited keyeframes.

	DAVIS				ShutterStock				
Method	T.C.↑	P.C.↑	W.E.↓ $/\times 10^{-3}$	L.C.↓ $/\times 10^{-3}$	T.C.↑	P.C↑	W.E↓ $/\times 10^{-3}$	L.C.↓ $/\times 10^{-3}$	Time
ControlNet per frame [207]	0.914	0.314	36.6	58.6	0.942	0.304	25.9	45.1	50s
Tune-a-Video [179]	0.966	0.299	20.4	48.2	0.979	0.292	13.4	34.9	20min
Text2Video-zero [82]	0.964	0.312	20.7	42.5	0.981	0.304	16.0	33.0	60s
ControlVideo-edge [212]	0.975	0.314	6.9	23.9	0.986	0.303	7.4	20.5	120s
TokenFlow [50]	0.977	0.317	7.0	19.6	0.987	0.313	5.4	15.8	150s
MaskINT (ours)	0.952	0.311	9.5	27.7	0.971	0.304	8.6	22.3	22s

Table 5.1: Quantitative comparisons. "T.C." stands for "temporal consistency", "P.C." stands for "prompt consistency", "W.E" stands for "warpping-error", and "L.C." stands for "long-term temporal consistency".

almost 5.5 times faster than ControlVideo [212], whose fully cross-frame attention is computationally extensive. Moreover, MaskINT is nearly 7 times faster than TokenFlow [50], whose DDIM inversion is time-consuming. On the contrary, our acceleration is derived from a combination of a lightweight network design and a reduced number of decoding steps in masked generative transformers.

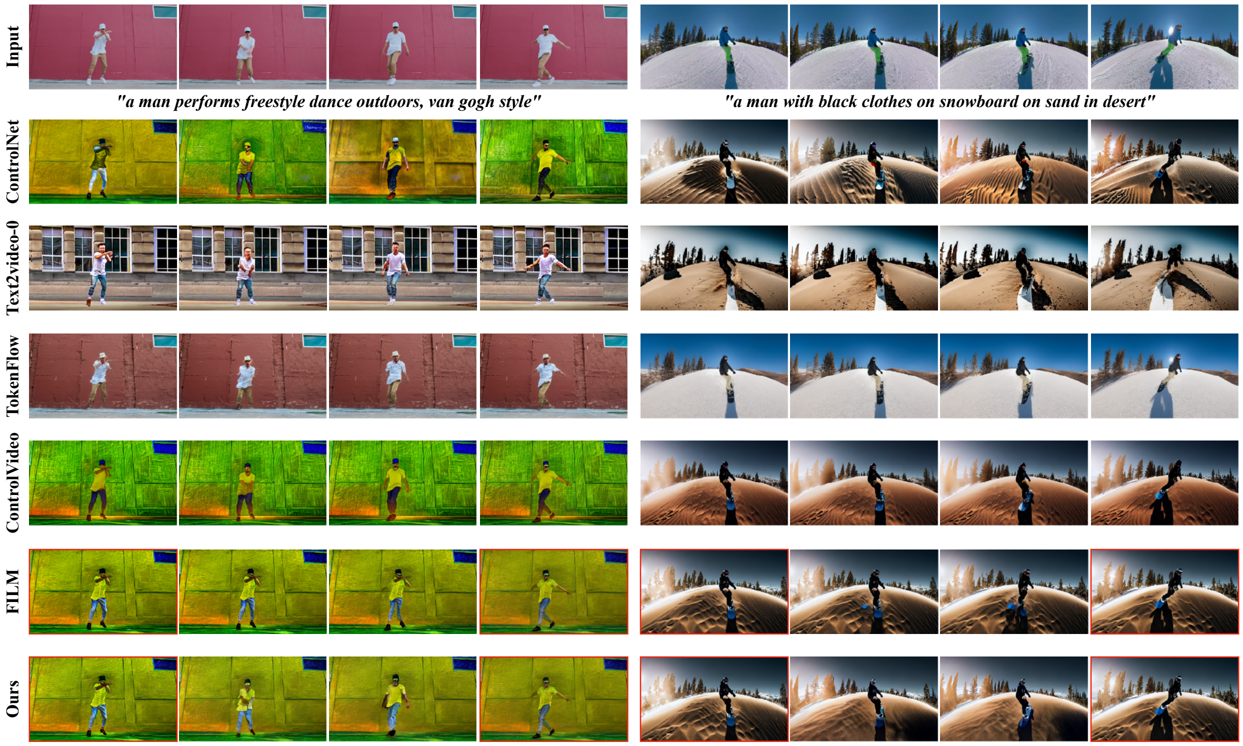

Input
"a man performs freestyle dance outdoors, van gogh style"
"a man with black clothes on snowboard on sand in desert"
ControlNet
Text2video-0
TokenFlow
ControlVideo
FILM
Ours

Figure 5.5: Qualitative comparisons with diffusion-based methods.

Qualitative Comparisons. Fig. 5.1, Fig. 5.3 and Fig. 5.4 show several samples of edited videos with MaskINT. Our method is capable of generating temporally consistent videos that adhere to the provided text prompts. This extends to a wide range of applications with text prompts, encompassing tasks such as stylization, background editing, foreground editing, and more. It also works well on challenging videos with substantial motion, such as jumping and running. Moreover, Fig. 5.5 and Fig. 5.6 provide qualitative comparisons of MaskINT to other baselines. Remarkably, diffusion methods [50, 82, 212] can ensure the consistency of overall appearance, but sometimes cannot maintain the consistency of detailed regions. For example, both TokenFlow [50] and Text2Video-Zero[82] exhibit noticeable artifacts in the leg region of the human subjects. ControlVideo [212] produces inconsistent hats. The potential explanation lies in the fact that these methods offer control over temporal consistency implicitly. Furthermore, FILM [134] produces videos that deviate from the original motions with the same edited keyframes. Our MaskINT consistently interpolates the intermediate frames based on the structure condition and even maintain better consistency in

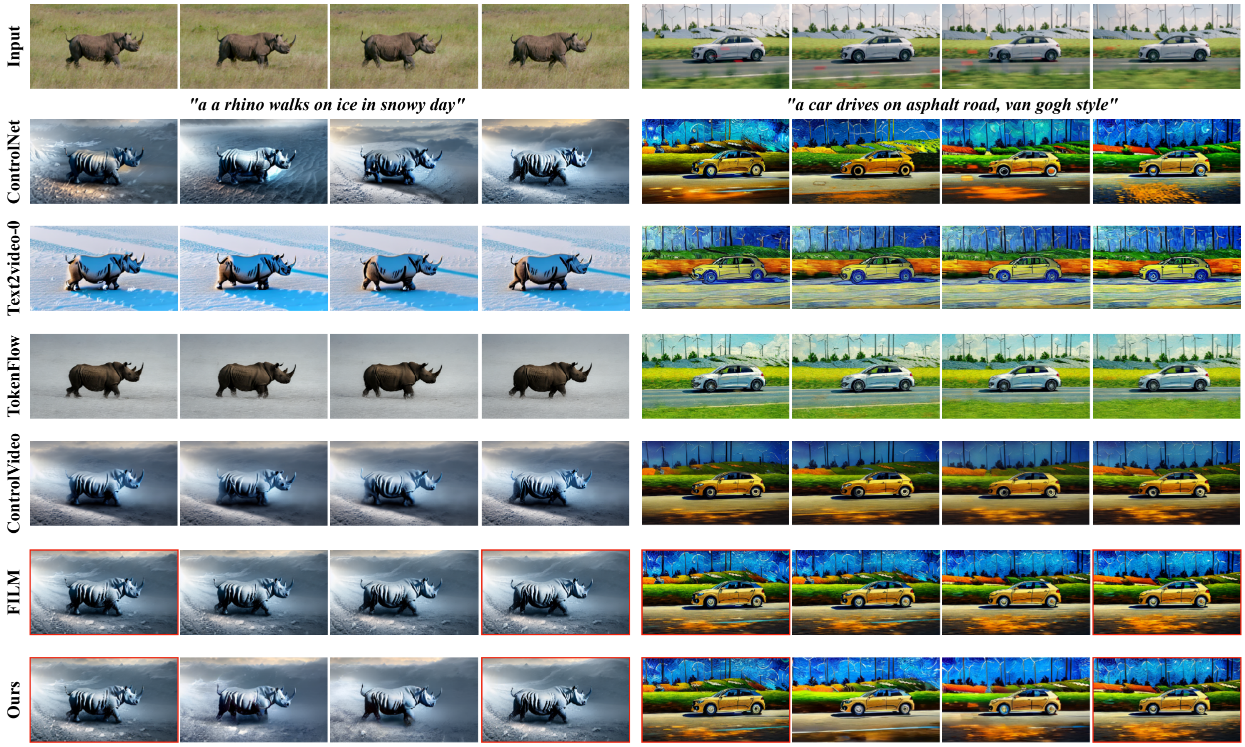

Input
"a a rhino walks on ice in snowy day"
"a car drives on asphalt road, van gogh style"
ControlNet
Text2video-0
TokenFlow
ControlVideo
FILM
Ours

Figure 5.6: Additional Qualitative comparisons with diffusion-based methods. Frames with red bounding box are jointly edited keyeframes.

local regions.

Extension on Long Video Editing. Since the non-autoregressive pipeline generates all video frames simultaneously, it's challenging for it to edit an entire long video due to GPU memory limitation. Nevertheless, our framework can still be extended to generate long videos by dividing the long video into short clips and progressively performing frame interpolation within each clip. For instance, given a video with 60 frames, we select the 1$^{\text{st}}$, 16$^{\text{th}}$, 31$^{\text{st}}$, 46$^{\text{th}}$, and 60$^{\text{th}}$ frames as keyframes. We jointly edit these selected 5 frames together and then perform structure-aware frame interpolation within each pair of consecutive keyframes. As shown in Fig. 5.7, with this design, our method can still generate consistent long videos. Besides, in this proposed extension, the generation of later frames is decoupled from the generated early frames, which differs from the autoregressive long-video generation pipeline in Phenaki [166]. Consequently, even if some early frames encounter difficulties, the generation of later frames can still proceed successfully.

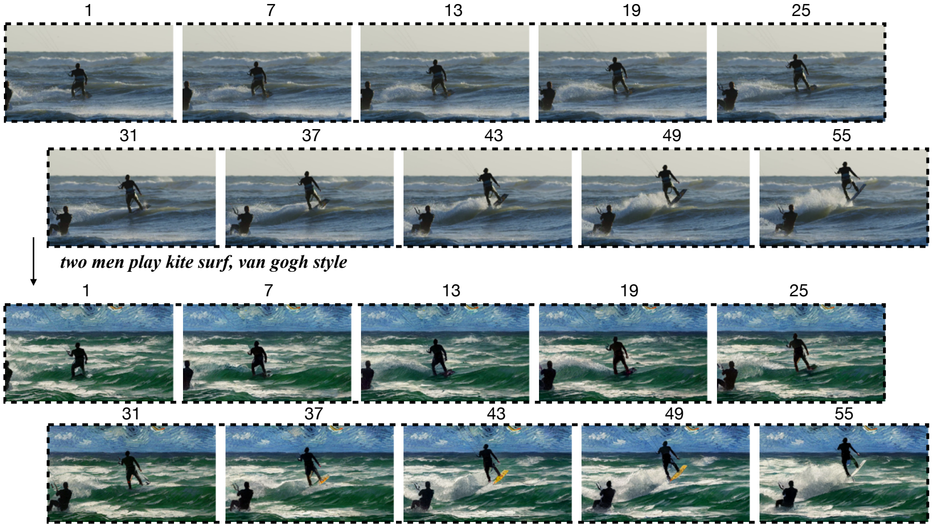


Figure 5.7: Examples of long video generation. The number indicates the index of frame.

5.6.3 Ablation Studies

Video Frame Interpolations We conduct a quantitative evaluation on the performance of our structure-aware interpolation module. In this ablation study, we perform frame interpolation using the original keyframes to reconstruct the original intermediate frames, and compare the interpolated frames with the original video frames at the pixel level. We use same testing samples from DAVIS and Shutterstock datasets, employing peak signal-to-noise ratio (PSNR), learned perceptual image patch similarity (LPIPS), and structured similarity (SSIM) as the evaluation metrics. We benchmark our method against two state-of-the-art VFI methods, namely FILM [134] and RIFE [72]. We also showcase the performance of applying VQ-GAN [43] to all video frames, serving as an upper bound for our method.

As in Table 5.2, our method significantly outperforms VFI methods on all evaluation metrics, with the benefit of the structure guidance from the intermediate frames. Furthermore, Fig. 5.8 shows qualitative comparison between video frame interpolation methods FILM [134] and MaskINT. Even when confronted with significant motion between two frames, our approach successfully reconstructs the original video, maintaining consistent motion through the aid of structural guidance. In contrast, FILM introduces undesirable artifacts, including disorted background, duplicated cat hands, and the absence of a camel's head, etc. The major reason is that current VFI models mainly focus on generating slow-motion effects and enhancing frame rate, making them less effective in handling frames with large motions, which usually requires a better semantic understanding. Additionally, the absence of structural guidance poses a challenge for these VFI methods in accurately aligning generated videos with the original motion.

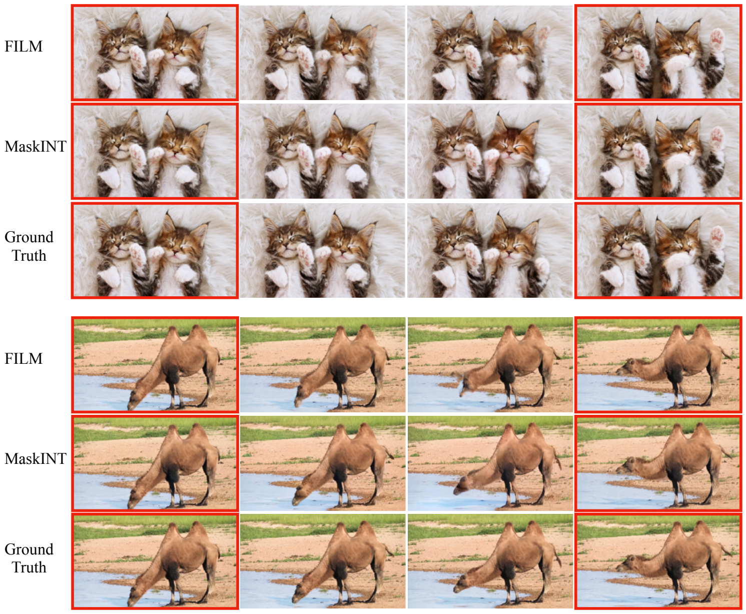


Figure 5.8: Qualitative comparisons on video reconstruction with original RGB frames. Frames with red bounding box are given.

Method	DAVIS PSNR↑	DAVIS SSIM ↑	DAVIS LPIPS↓	ShutterStock PSNR↑	ShutterStock SSIM↑	ShutterStock LPIPS↓
RIFE [72]	17.31	0.5195	0.2512	20.44	0.7210	0.1533
FILM [134]	17.00	0.5011	0.2363	20.90	0.7453	0.1246
MaskINT (ours)	22.15	0.6332	0.1483	24.19	0.7616	0.1097
VQGAN (ground truth)	25.66	0.7429	0.0784	27.81	0.8327	0.0561

Table 5.2: Quantitative comparisons on video frame interpolation with original keyframes.

Number of of keyframes Although our model is trained with frame interpolation by default, MaskINT can seamlessly generalize to an arbitrary number of keyframes without finetuning. We assess the impact of varying the quantity of keyframes on the generation performance. As shown in the left part of Table 5.3, with an increase in the number of keyframes, the model exhibits an improvement in performance. Generally, with more information, performing frame interpolation is easier. However, simultaneously editing more keyframes requires longer time due to the global attention among them.

Decoding steps We also explore the number of decoding steps of the masked generative transformers in the second stage. The right part of Table 5.3 shows that more decoding steps can bring slight improvement on the temporal consistency, but requires more time. Considering the trade-off between performance and efficiency, we chose $K = 32$ steps by default in all experiments.

# keyframes	T.C.	P.C.	Time
1	0.9690	0.2984	19s
2	0.9714	0.3038	22s
3	0.9721	0.3051	26s
4	0.9728	0.3069	29s
6	0.9737	0.3035	35s

#decoding step K	T.C.	P.C.	Time
16	0.9691	0.3038	15s
32	0.9714	0.3038	22s
64	0.9719	0.3040	33s
128	0.9720	0.3041	62s

Table 5.3: Ablation study on ShutterStock dataset of the number of keyframes and the number of decoding steps K. "T.C." stands for "temporal consistency" and "P.C." stands for "prompt consistency".

Diverse Structural Conditions Although we by default utilize HED edge map as our structure condition for both stages, our method can also employ other structural controls in both stages due to the disentanglement. In this study, we explore utilizing ControlNet with depth map to perform key frame editing, and using the depth map as the guidance to perform structure-aware frame interpolation. Additionally, we explore various combinations of these approaches. As summarized in Table 5.4, all of these combinations achieve the same level of performance. The performance of prompt consistency (P.C.) is determined by the specific key frame editing methods employed. For the second stage, depth control typically offers greater flexibility than HED edge control for frame interpolation. This could be the potential reason for the slightly worse performance in temporal consistency with edge-based key frame editing.

Stage1	Stage2	T.C.	P.C
HED edge	HED edge	0.9714	0.3038
depth map	HED edge	0.9713	0.3159
HED edge	depth map	0.9683	0.3035
depth map	depth map	0.9719	0.3171

Table 5.4: Quantitative comparisons of the combination of varied structural conditions in each stage on ShutterStock. "T.C." stands for "temporal consistency", and "P.C." stands for "prompt consistency".

5.7 Limitations

One limitation of our work is that, it can only perform structure-preserving video editing, such as altering style or appearance. Thus, it cannot handle edits that require structural changes, a limitation shared with TokenFlow [50]. We also require that no new objects should appear in the intermediate frames. Besides, performance relies on the image-editing model and structure detector. In certain challenging scenarios, the attention-based joint key frames editing stage struggles to produce consistent frames, primarily due to the complexity of the scene or the presence of exceptionally large motion. When these models fail, MaskINT enforces meaningless interpolation, resulting in artifacts. Figure 5.9 show some failure cases when the first stage fails.

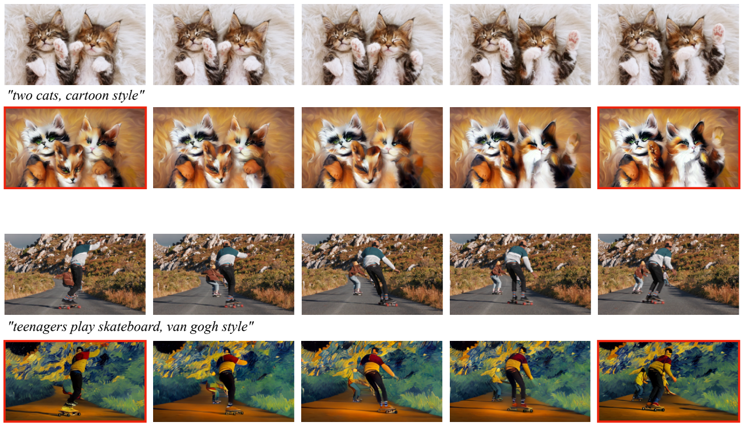

"two cats, cartoon style"
"teenagers play skateboard, van gogh style"

Figure 5.9: Examples of failure cases.

5.8 Conclusion

We propose MaskINT towards consistent and efficient video editing with text prompt. MaskINT disentangle this task into keyframes joint editing with diffusion methods and structure-aware frame interpolation with non-autoregressive masked transformers. Experimental results demonstrate that MaskINT achieves comparable performance with pure diffusion-based methods while significantly reduces the inference time. Moreover, our work demonstrates the substantial promise of non-autoregressive generative transformers within the realm of video editing.

Chapter 6

Conclusion

In this dissertation, several algorithms are proposed to tackle problems in human-centric visual understanding and generation. We first explore the understanding part and start from human body pose estimation in both monocular and multi-view setting, which usually serves as the foundation for other tasks. In detail, we focus on the cross-view fusion techniques with transformers and further improve the efficiency of pose transformers with token pruning techniques. We then explore the generation part and explore mesh-guided human head image generation. Specifically, we can efficiently generate novel face shape, expression, and pose from a single reference image under the control of parameters of mesh model. Furthermore, we explore appearance editing in videos with text instruction only, and propose a network that significantly outperform previous works in terms of running time.

Bibliography

[1] O. Alexander, M. Rogers, W. Lambeth, J.-Y. Chiang, W.-C. Ma, C.-C. Wang, and P. Debevec. The digital emily project: Achieving a photorealistic digital actor. *IEEE Computer Graphics and Applications*, pages 20–31, 2010.

[2] K.-A. Aliev, A. Sevastopolsky, M. Kolos, D. Ulyanov, and V. Lempitsky. Neural point-based graphics. In *ECCV*, pages 696–712, 2020.

[3] A. M. Andrew. Multiple view geometry in computer vision. *Kybernetes*, 2001.

[4] M. Andriluka, L. Pishchulin, P. Gehler, and B. Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *CVPR*, 2014.

[5] S. Athar, Z. Xu, K. Sunkavalli, E. Shechtman, and Z. Shu. Rignerf: Fully controllable neural 3d portraits. In *CVPR*, 2022.

[6] D. Bashkirova, J. Lezama, K. Sohn, K. Saenko, and I. Essa. Masksketch: Unpaired structure-guided masked image generation. In *CVPR*, pages 1879–1889, 2023.

[7] G. Bertasius, H. Wang, and L. Torresani. Is space-time attention all you need for video understanding? In *ICML*, 2021.

[8] V. Blanz and T. Vetter. A morphable model for the synthesis of 3d faces. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, 1999.

[9] A. Blattmann, R. Rombach, H. Ling, T. Dockhorn, S. W. Kim, S. Fidler, and K. Kreis. Align your latents: High-resolution video synthesis with latent diffusion models. In *CVPR*, pages 22563–22575, 2023.

[10] J. F. Blinn and M. E. Newell. Texture and reflection in computer generated images. *Communications of the ACM*, pages 542–547, 1976.

[11] T. Brooks, A. Holynski, and A. A. Efros. Instructpix2pix: Learning to follow image editing instructions. In *CVPR*, pages 18392–18402, 2023.

[12] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *NeurIPS*, 33:1877–1901, 2020.

[13] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman. Vggface2: A dataset for recognising faces across pose and age. In *2018 13th IEEE international conference on automatic face & gesture recognition (FG 2018)*, pages 67–74. IEEE, 2018.

[14] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020.

[15] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021.

[16] C. S. Catalin Ionescu, Fuxin Li. Latent structured models for human pose estimation. In *ICCV*, 2011.

[17] D. Ceylan, C.-H. P. Huang, and N. J. Mitra. Pix2video: Video editing using image diffusion. In *ICCV*, pages 23206–23217, 2023.

[18] E. R. Chan, C. Z. Lin, M. A. Chan, K. Nagano, B. Pan, S. De Mello, O. Gallo, L. J. Guibas, J. Tremblay, S. Khamis, et al. Efficient geometry-aware 3d generative adversarial networks. In *CVPR*, pages 16123–16133, 2022.

[19] H. Chang, H. Zhang, J. Barber, A. Maschinot, J. Lezama, L. Jiang, M.-H. Yang, K. Murphy, W. T. Freeman, M. Rubinstein, et al. Muse: Text-to-image generation via masked generative transformers. *ICML*, 2023.

[20] H. Chang, H. Zhang, L. Jiang, C. Liu, and W. T. Freeman. Maskgit: Masked generative image transformer. In *CVPR*, pages 11315–11325, 2022.

[21] C.-F. Chen, Q. Fan, and R. Panda. Crossvit: Cross-attention multi-scale vision transformer for image classification. *ICCV*, 2021.

[22] L. Chen, S.-Y. Lin, Y. Xie, Y.-Y. Lin, W. Fan, and X. Xie. Dggan: Depth-image guided generative adversarial networks for disentangling rgb and depth images in 3d hand pose estimation. In *WACV*, pages 411–419, 2020.

[23] L. Chen, S.-Y. Lin, Y. Xie, Y.-Y. Lin, and X. Xie. Mvhm: A large-scale multi-view hand mesh benchmark for accurate 3d hand pose estimation. In *WACV*, pages 836–845, 2021.

[24] L. Chen, S.-Y. Lin, Y. Xie, H. Tang, Y. Xue, X. Xie, Y.-Y. Lin, and W. Fan. Generating realistic training images based on tonality-alignment generative adversarial networks for hand pose estimation. *arXiv preprint arXiv:1811.09916*, 2018.

[25] T. Chen, Y. Cheng, Z. Gan, L. Yuan, L. Zhang, and Z. Wang. Chasing sparsity in vision transformers: An end-to-end exploration. *NeurIPS*, 2021.

[26] T. Chen, Z. Zhang, Y. Cheng, A. Awadallah, and Z. Wang. The principle of diversity: Training stronger vision transformers calls for reducing all levels of redundancy. In *CVPR*, 2022.

[27] X. Chen, Q. Cao, Y. Zhong, J. Zhang, S. Gao, and D. Tao. Dearkd: Data-efficient early knowledge distillation for vision transformers. In *CVPR*, 2022.

[28] Y. Chen, H. Ma, D. Kong, X. Yan, J. Wu, W. Fan, and X. Xie. Nonparametric structure regularization machine for 2d hand pose estimation. In *WACV*, pages 381–390, 2020.

[29] Y. Chen, H. Ma, J. Wang, J. Wu, X. Wu, and X. Xie. Pd-net: Quantitative motor function evaluation for parkinson's disease via automated hand gesture analysis. In *KDD*, pages 2683–2691, 2021.

[30] Y. Chen, Z. Wang, Y. Peng, Z. Zhang, G. Yu, and J. Sun. Cascaded pyramid network for multi-person pose estimation. In *CVPR*, 2018.

[31] J. Cho, K. Youwang, and T.-H. Oh. Cross-attention of disentangled modalities for 3d human mesh recovery with transformers. In *ECCV*, 2022.

[32] X. Chu, W. Yang, W. Ouyang, C. Ma, A. L. Yuille, and X. Wang. Multi-context attention for human pose estimation. In *CVPR*, 2017.

[33] J. S. Chung, A. Nagrani, and A. Zisserman. Voxceleb2: Deep speaker recognition. *arXiv preprint arXiv:1806.05622*, 2018.

[34] E. Corona, M. Zanfir, T. Alldieck, E. G. Bazavan, A. Zanfir, and C. Sminchisescu. Structured 3d features for reconstructing controllable avatars. In *CVPR*, pages 16954–16964, 2023.

[35] G. Couairon, J. Verbeek, H. Schwenk, and M. Cord. Diffedit: Diffusion-based semantic image editing with mask guidance. *arXiv preprint arXiv:2210.11427*, 2022.

[36] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *NAACL*, 2019.

[37] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

[38] Z. Dou, Q. Wu, C. Lin, Z. Cao, Q. Wu, W. Wan, T. Komura, and W. Wang. Tore: Token reduction for efficient human mesh recovery with transformer. In *ICCV*, pages 15143–15155, 2023.

[39] M. C. Doukas, S. Zafeiriou, and V. Sharmanska. Headgan: One-shot neural head synthesis and editing. In *ICCV*, pages 14398–14407, 2021.

[40] N. Drobyshev, J. Chelishev, T. Khakhulin, A. Ivakhnenko, V. Lempitsky, and E. Zakharov. Megaportraits: One-shot megapixel neural head avatars. In *MM*, 2022.

[41] A. Dubey, F. Radenovic, D. Mahajan, and V. Ramanathan. Retina vq. 2023.

[42] P. Esser, J. Chiu, P. Atighehchian, J. Granskog, and A. Germanidis. Structure and content-guided video synthesis with diffusion models. *ICCV*, 2023.

[43] P. Esser, R. Rombach, and B. Ommer. Taming transformers for high-resolution image synthesis. In *CVPR*, pages 12873–12883, 2021.

[44] Y. Fang, B. Liao, X. Wang, J. Fang, J. Qi, R. Wu, J. Niu, and W. Liu. You only look at one sequence: Rethinking transformer in vision through object detection. *NeurIPS*, 2021.

[45] B. Fasel, J. Spörri, J. Chardonnens, J. Kröll, E. Müller, and K. Aminian. Joint inertial sensor orientation drift reduction for highly dynamic movements. *IEEE journal of biomedical and health informatics*, 22(1):77–86, 2017.

[46] Y. Feng, H. Feng, M. J. Black, and T. Bolkart. Learning an animatable detailed 3d face model from in-the-wild images. *ACM Transactions on Graphics (ToG)*, pages 1–13, 2021.

[47] G. Gafni, J. Thies, M. Zollhofer, and M. Nießner. Dynamic neural radiance fields for monocular 4d facial avatar reconstruction. In *CVPR*, 2021.

[48] L. Ge, Z. Ren, Y. Li, Z. Xue, Y. Wang, J. Cai, and J. Yuan. 3d hand shape and pose estimation from a single rgb image. In *CVPR*, 2019.

[49] S. Ge, T. Hayes, H. Yang, X. Yin, G. Pang, D. Jacobs, J.-B. Huang, and D. Parikh. Long video generation with time-agnostic vqgan and time-sensitive transformer. In *ECCV*, pages 102–118. Springer, 2022.

[50] M. Geyer, O. Bar-Tal, S. Bagon, and T. Dekel. Tokenflow: Consistent diffusion features for consistent video editing. *arXiv preprint arXiv:2307.10373*, 2023.

[51] P. Ghosh, P. S. Gupta, R. Uziel, A. Ranjan, M. J. Black, and T. Bolkart. Gif: Generative interpretable faces. In *3DV*, pages 868–878, 2020.

[52] S. Gong, L. Chen, M. Bronstein, and S. Zafeiriou. Spiralnet++: A fast and highly efficient mesh convolution operator. In *ICCVW*, 2019.

[53] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *NIPS*, 2014.

[54] P.-W. Grassal, M. Prinzler, T. Leistner, C. Rother, M. Nießner, and J. Thies. Neural head avatars from monocular rgb videos. In *CVPR*, 2022.

[55] M. Gross and H. Pfister. *Point-based graphics*. Elsevier, 2011.

[56] A. Gupta, S. Tian, Y. Zhang, J. Wu, R. Martín-Martín, and L. Fei-Fei. Maskvit: Masked visual pre-training for video prediction. In *ICLR*, 2023.

[57] S. Han, H. Mao, and W. J. Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *ICLR*, 2016.

[58] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *ICCV*, 2017.

[59] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[60] T. He, Y. Xu, S. Saito, S. Soatto, and T. Tung. Arch++: Animation-ready clothed human reconstruction revisited. In *ICCV*, pages 11046–11056, 2021.

[61] Y. He, R. Yan, K. Fragkiadaki, and S.-I. Yu. Epipolar transformers. In *CVPR*, 2020.

[62] A. Hertz, R. Mokady, J. Tenenbaum, K. Aberman, Y. Pritch, and D. Cohen-Or. Prompt-to-prompt image editing with cross attention control. *arXiv preprint arXiv:2208.01626*, 2022.

[63] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *NeurIPS*, 30, 2017.

[64] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[65] J. Ho, W. Chan, C. Saharia, J. Whang, R. Gao, A. Gritsenko, D. P. Kingma, B. Poole, M. Norouzi, D. J. Fleet, et al. Imagen video: High definition video generation with diffusion models. *arXiv preprint arXiv:2210.02303*, 2022.

[66] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *NeurIPS*, 33:6840–6851, 2020.

[67] W. Hong, M. Ding, W. Zheng, X. Liu, and J. Tang. Cogvideo: Large-scale pretraining for text-to-video generation via transformers. *ICLR*, 2023.

[68] Y. Hong, B. Peng, H. Xiao, L. Liu, and J. Zhang. Headnerf: A real-time nerf-based parametric head model. In *CVPR*, pages 20374–20384, 2022.

[69] L. Huang, D. Chen, Y. Liu, Y. Shen, D. Zhao, and J. Zhou. Composer: Creative and controllable image synthesis with composable conditions. *arXiv preprint arXiv:2302.09778*, 2023.

[70] Z. Huang, C. Wan, T. Probst, and L. Van Gool. Deep learning on lie groups for skeleton-based action recognition. In *CVPR*, 2017.

[71] Z. Huang, Y. Xu, C. Lassner, H. Li, and T. Tung. Arch: Animatable reconstruction of clothed humans. In *CVPR*, pages 3093–3102, 2020.

[72] Z. Huang, T. Zhang, W. Heng, B. Shi, and S. Zhou. Real-time intermediate flow estimation for video frame interpolation. In *ECCV*, pages 624–642. Springer, 2022.

[73] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *TPAMI*, 2014.

[74] K. Iskakov, E. Burkov, V. Lempitsky, and Y. Malkov. Learnable triangulation of human pose. In *ICCV*, 2019.

[75] H. Jiang, D. Sun, V. Jampani, M.-H. Yang, E. Learned-Miller, and J. Kautz. Super slomo: High quality estimation of multiple intermediate frames for video interpolation. In *CVPR*, pages 9000–9008, 2018.

[76] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*, pages 694–711, 2016.

[77] K. Kania, K. M. Yi, M. Kowalski, T. Trzciński, and A. Tagliasacchi. Conerf: Controllable neural radiance fields. In *CVPR*, 2022.

[78] H. Kato, D. Beker, M. Morariu, T. Ando, T. Matsuoka, W. Kehl, and A. Gaidon. Differentiable rendering: A survey. *arXiv preprint arXiv:2006.12057*, 2020.

[79] B. Kawar, S. Zada, O. Lang, O. Tov, H. Chang, T. Dekel, I. Mosseri, and M. Irani. Imagic: Text-based real image editing with diffusion models. In *CVPR*, pages 6007–6017, 2023.

[80] L. Ke, M.-C. Chang, H. Qi, and S. Lyu. Multi-scale structure-aware network for human pose estimation. In *ECCV*, 2018.

[81] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 2023.

[82] L. Khachatryan, A. Movsisyan, V. Tadevosyan, R. Henschel, Z. Wang, S. Navasardyan, and H. Shi. Text2video-zero: Text-to-image diffusion models are zero-shot video generators. *ICCV*, 2023.

[83] T. Khakhulin, V. Sklyarova, V. Lempitsky, and E. Zakharov. Realistic one-shot mesh-based head avatars. In *ECCV*, pages 345–362, 2022.

[84] W. Kim, B. Son, and I. Kim. Vilt: Vision-and-language transformer without convolution or region supervision. *ICML*, 2021.

[85] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[86] D. Kong, Y. Chen, H. Ma, X. Yan, and X. Xie. Adaptive graphical model network for 2d handpose estimation. *BMVC*, 2019.

[87] D. Kong, H. Ma, Y. Chen, and X. Xie. Rotation-invariant mixed graphical model network for 2d hand pose estimation. In *WACV*, pages 1546–1555, 2020.

[88] D. Kong, H. Ma, and X. Xie. Sia-gcn: A spatial information aware graph neural network with 2d convolutions for hand pose estimation. *BMVC*, 2020.

[89] Z. Kong, P. Dong, X. Ma, X. Meng, W. Niu, M. Sun, B. Ren, M. Qin, H. Tang, and Y. Wang. Spvit: Enabling faster vision transformers via soft token pruning. *ECCV*, 2022.

[90] Z. Kong, H. Ma, G. Yuan, M. Sun, Y. Xie, P. Dong, X. Meng, X. Shen, H. Tang, M. Qin, et al. Peeling the onion: Hierarchical reduction of data redundancy for efficient vision transformer training. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 8360–8368, 2023.

[91] G. Kopanas, J. Philip, T. Leimkühler, and G. Drettakis. Point-based neural rendering with per-view optimization. In *Computer Graphics Forum*, pages 29–43, 2021.

[92] G. Le Moing, J. Ponce, and C. Schmid. Ccvs: context-aware controllable video synthesis. *NeurIPS*, 34:14042–14055, 2021.

[93] G. Li, N. Duan, Y. Fang, M. Gong, and D. Jiang. Unicoder-vl: A universal encoder for vision and language by cross-modal pre-training. In *AAAI*, 2020.

[94] K. Li, S. Wang, X. Zhang, Y. Xu, W. Xu, and Z. Tu. Pose recognition with cascade transformers. *CVPR*, 2021.

[95] T. Li, T. Bolkart, M. J. Black, H. Li, and J. Romero. Learning a model of facial shape and expression from 4d scans. *ACM Transactions on Graphics (ToG)*, 2017.

[96] W. Li, L. Zhang, D. Wang, B. Zhao, Z. Wang, M. Chen, B. Zhang, Z. Wang, L. Bo, and X. Li. One-shot high-fidelity talking-head synthesis with deformable neural radiance field. In *CVPR*, 2023.

[97] Y. Li, S. Zhang, Z. Wang, S. Yang, W. Yang, S.-T. Xia, and E. Zhou. Tokenpose: Learning keypoint tokens for human pose estimation. In *ICCV*, 2021.

[98] Y. Liang, C. GE, Z. Tong, Y. Song, J. Wang, and P. Xie. EVit: Expediting vision transformers via token reorganizations. In *ICLR*, 2022.

[99] K. Lin, L. Wang, and Z. Liu. End-to-end human pose and mesh reconstruction with transformers. *CVPR*, 2021.

[100] K. Lin, L. Wang, and Z. Liu. Mesh graphormer. In *ICCV*, pages 12939–12948, 2021.

[101] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

[102] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021.

[103] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black. Smpl: A skinned multi-person linear model. *ACM transactions on graphics (TOG)*, 34(6):1–16, 2015.

[104] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

[105] L. Lu, R. Wu, H. Lin, J. Lu, and J. Jia. Video frame interpolation with transformer. In *CVPR*, pages 3532–3542, 2022.

[106] H. Ma, L. Chen, D. Kong, Z. Wang, X. Liu, H. Tang, X. Yan, Y. Xie, S.-Y. Lin, and X. Xie. Transfusion: Cross-view fusion with transformer for 3d human pose estimation. In *BMVC*, 2021.

[107] H. Ma, S. Mahdizadehaghdam, B. Wu, Z. Fan, Y. Gu, W. Zhao, L. Shapira, and X. Xie. Maskint: Video editing via interpolative non-autoregressive masked transformers. *arxiv preprint*, 2023.

[108] H. Ma, Z. Wang, Y. Chen, D. Kong, L. Chen, X. Liu, X. Yan, H. Tang, and X. Xie. Ppt: token-pruned pose transformer for monocular and multi-view human pose estimation. In *ECCV*, 2022.

[109] H. Ma, T. Zhang, S. Sun, X. Yan, K. Han, and X. Xie. Cvthead: One-shot controllable head avatar with vertex-feature transformer. *WACV*, 2024.

[110] Z. Ma, X. Zhu, G.-J. Qi, Z. Lei, and L. Zhang. Otavatar: One-shot talking face avatar with controllable tri-plane rendering. In *CVPR*, pages 16901–16910, 2023.

[111] W. Mao, Y. Ge, C. Shen, Z. Tian, X. Wang, and Z. Wang. Tfpose: Direct human pose estimation with transformers. *arXiv preprint arXiv:2103.15320*, 2021.

[112] D. Mehta, S. Sridhar, O. Sotnychenko, H. Rhodin, M. Shafiei, H.-P. Seidel, W. Xu, D. Casas, and C. Theobalt. Vnect: Real-time 3d human pose estimation with a single rgb camera. *ACM Transactions on Graphics*, 2017.

[113] C. Meng, Y. He, Y. Song, J. Song, J. Wu, J.-Y. Zhu, and S. Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. *ICLR*, 2022.

[114] L. Meng, H. Li, B.-C. Chen, S. Lan, Z. Wu, Y.-G. Jiang, and S.-N. Lim. Adavit: Adaptive vision transformers for efficient image recognition. In *CVPR*, 2022.

[115] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

[116] C. Mou, X. Wang, L. Xie, J. Zhang, Z. Qi, Y. Shan, and X. Qie. T2i-adapter: Learning adapters to dig out more controllable ability for text-to-image diffusion models. *arXiv preprint arXiv:2302.08453*, 2023.

[117] A. Nagrani, J. S. Chung, and A. Zisserman. Voxceleb: a large-scale speaker identification dataset. *arXiv preprint arXiv:1706.08612*, 2017.

[118] C. Neff, A. Sheth, S. Furgurson, and H. Tabkhi. Efficienthrnet: Efficient scaling for lightweight high-resolution multi-person pose estimation. *arXiv preprint arXiv:2007.08090*, 2020.

[119] A. Newell, K. Yang, and J. Deng. Stacked hourglass networks for human pose estimation. In *ECCV*, 2016.

[120] S. Niklaus, L. Mai, and F. Liu. Video frame interpolation via adaptive separable convolution. In *ICCV*, pages 261–270, 2017.

[121] D. Osokin. Real-time 2d multi-person pose estimation on cpu: Lightweight openpose. *arXiv preprint arXiv:1811.12004*, 2018.

[122] G. Papandreou, T. Zhu, L.-C. Chen, S. Gidaris, J. Tompson, and K. Murphy. Personlab: Person pose estimation and instance segmentation with a bottom-up, part-based, geometric embedding model. In *ECCV*, 2018.

[123] K. Park, U. Sinha, J. T. Barron, S. Bouaziz, D. B. Goldman, S. M. Seitz, and R. Martin-Brualla. Nerfies: Deformable neural radiance fields. In *ICCV*, pages 5865–5874, 2021.

[124] G. Parmar, K. Kumar Singh, R. Zhang, Y. Li, J. Lu, and J.-Y. Zhu. Zero-shot image-to-image translation. In *ACM SIGGRAPH 2023 Conference Proceedings*, pages 1–11, 2023.

[125] P. Paysan, R. Knothe, B. Amberg, S. Romdhani, and T. Vetter. A 3d face model for pose and illumination invariant face recognition. In *2009 sixth IEEE international conference on advanced video and signal based surveillance*, pages 296–301, 2009.

[126] J. Pont-Tuset, F. Perazzi, S. Caelles, P. Arbeláez, A. Sorkine-Hornung, and L. Van Gool. The 2017 davis challenge on video object segmentation. *arXiv:1704.00675*, 2017.

[127] S. Prokudin, M. J. Black, and J. Romero. Smplpix: Neural avatars from 3d human models. In *WACV*, pages 1810–1819, 2021.

[128] C. Qi, X. Cun, Y. Zhang, C. Lei, X. Wang, Y. Shan, and Q. Chen. Fatezero: Fusing attentions for zero-shot text-based video editing. *ICCV*, 2023.

[129] H. Qiu, C. Wang, J. Wang, N. Wang, and W. Zeng. Cross view fusion for 3d human pose estimation. In *ICCV*, 2019.

[130] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, pages 8748–8763, 2021.

[131] R. Rakhimov, A.-T. Ardelean, V. Lempitsky, and E. Burnaev. Npbg++: Accelerating neural point-based graphics. In *CVPR*, pages 15969–15979, 2022.

[132] Y. Rao, W. Zhao, B. Liu, J. Lu, J. Zhou, and C.-J. Hsieh. Dynamicvit: Efficient vision transformers with dynamic token sparsification. *NeurIPS*, 2021.

[133] N. Ravi, J. Reizenstein, D. Novotny, T. Gordon, W.-Y. Lo, J. Johnson, and G. Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv preprint arXiv:2007.08501*, 2020.

[134] F. Reda, J. Kontkanen, E. Tabellion, D. Sun, C. Pantofaru, and B. Curless. Film: Frame interpolation for large motion. In *ECCV*, 2022.

[135] H. Rhodin, J. Spörri, I. Katircioglu, V. Constantin, F. Meyer, E. Müller, M. Salzmann, and P. Fua. Learning monocular 3d human pose estimation from multi-view images. In *CVPR*, 2018.

[136] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, pages 10684–10695, 2022.

[137] M. Ryoo, A. Piergiovanni, A. Arnab, M. Dehghani, and A. Angelova. Tokenlearner: Adaptive space-time tokenization for videos. *NeurIPS*, 2021.

[138] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. L. Denton, K. Ghasemipour, R. Gontijo Lopes, B. Karagol Ayan, T. Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *NeurIPS*, 35:36479–36494, 2022.

[139] S. Saito, Z. Huang, R. Natsume, S. Morishima, A. Kanazawa, and H. Li. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *ICCV*, pages 2304–2314, 2019.

[140] P. Sharma, N. Ding, S. Goodman, and R. Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, 2018.

[141] S. Shen, Z. Dong, J. Ye, L. Ma, Z. Yao, A. Gholami, M. W. Mahoney, and K. Keutzer. Q-bert: Hessian based ultra low precision quantization of bert. In *AAAI*, 2020.

[142] X. Shen, G. Yuan, W. Niu, X. Ma, J. Guan, Z. Li, B. Ren, and Y. Wang. Towards fast and accurate multi-person pose estimation on mobile devices. *arXiv preprint arXiv:2106.15304*, 2021.

[143] A. Siarohin, S. Lathuilière, S. Tulyakov, E. Ricci, and N. Sebe. First order motion model for image animation. *NeurIPS*, 32, 2019.

[144] H. Sim, J. Oh, and M. Kim. Xvfi: extreme video frame interpolation. In *ICCV*, pages 14489–14498, 2021.

[145] T. Simon, H. Joo, I. Matthews, and Y. Sheikh. Hand keypoint detection in single images using multiview bootstrapping. In *CVPR*, 2017.

[146] U. Singer, A. Polyak, T. Hayes, X. Yin, J. An, S. Zhang, Q. Hu, H. Yang, O. Ashual, O. Gafni, et al. Make-a-video: Text-to-video generation without text-video data. *ICLR*, 2023.

[147] K. Sohn, N. Ruiz, K. Lee, D. C. Chin, I. Blok, H. Chang, J. Barber, L. Jiang, G. Entis, Y. Li, et al. Styledrop: Text-to-image generation in any style. *arXiv preprint arXiv:2306.00983*, 2023.

[148] J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. *ICLR*, 2021.

[149] J. Spörri. Reasearch dedicated to sports injury prevention-the'sequence of prevention'on the example of alpine ski racing. *Habilitation with Venia Docendi in Biomechanics*, 2016.

[150] S. Su, Q. Yan, Y. Zhu, C. Zhang, X. Ge, J. Sun, and Y. Zhang. Blindly assess image quality in the wild guided by a self-adaptive hyper network. In *CVPR*, pages 3667–3676, 2020.

[151] W. Su, X. Zhu, Y. Cao, B. Li, L. Lu, F. Wei, and J. Dai. Vl-bert: Pre-training of generic visual-linguistic representations. *ICLR*, 2020.

[152] K. Sun, B. Xiao, D. Liu, and J. Wang. Deep high-resolution representation learning for human pose estimation. In *CVPR*, 2019.

[153] M. Sun, H. Ma, G. Kang, Y. Jiang, T. Chen, X. Ma, Z. Wang, and Y. Wang. Vaqf: Fully automatic software-hardware co-design framework for low-bit vision transformer. *arXiv preprint arXiv:2201.06618*, 2022.

[154] H. Tan and M. Bansal. Lxmert: Learning cross-modality encoder representations from transformers. *EMNLP*, 2019.

[155] H. Tang, X. Liu, K. Han, X. Xie, X. Chen, H. Qian, Y. Liu, S. Sun, and N. Bai. Spatial context-aware self-attention model for multi-organ segmentation. In *WACV*, pages 939–949, 2021.

[156] A. Tewari, M. Elgharib, G. Bharaj, F. Bernard, H.-P. Seidel, P. Pérez, M. Zollhofer, and C. Theobalt. Stylerig: Rigging stylegan for 3d control over portrait images. In *CVPR*, pages 6142–6151, 2020.

[157] J. Thies, M. Elgharib, A. Tewari, C. Theobalt, and M. Nießner. Neural voice puppetry: Audio-driven facial reenactment. In *ECCV*, pages 716–731, 2020.

[158] J. Thies, M. Zollhöfer, and M. Nießner. Deferred neural rendering: Image synthesis using neural textures. *Acm Transactions on Graphics (TOG)*, 38(4):1–12, 2019.

[159] J. Thies, M. Zollhofer, M. Stamminger, C. Theobalt, and M. Nießner. Face2face: Real-time face capture and reenactment of rgb videos. In *CVPR*, pages 2387–2395, 2016.

[160] D. Tome, C. Russell, and L. Agapito. Lifting from the deep: Convolutional 3d pose estimation from a single image. In *CVPR*, 2017.

[161] J. J. Tompson, A. Jain, Y. LeCun, and C. Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. *Advances in neural information processing systems*, 27:1799–1807, 2014.

[162] A. Toshev and C. Szegedy. Deeppose: Human pose estimation via deep neural networks. In *CVPR*, 2014.

[163] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou. Training data-efficient image transformers & distillation through attention. *ICML*, 2021.

[164] N. Tumanyan, M. Geyer, S. Bagon, and T. Dekel. Plug-and-play diffusion features for text-driven image-to-image translation. In *CVPR*, pages 1921–1930, 2023.

[165] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *NeurIPS*, 30, 2017.

[166] R. Villegas, M. Babaeizadeh, P.-J. Kindermans, H. Moraldo, H. Zhang, M. T. Saffar, S. Castro, J. Kunze, and D. Erhan. Phenaki: Variable length video generation from open domain textual descriptions. In *ICLR*, 2023.

[167] Q. Wang, L. Zhang, and B. Li. Safa: Structure aware face animation. In *3DV*, pages 679–688, 2021.

[168] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *CVPR*, pages 8798–8807, 2018.

[169] T.-C. Wang, A. Mallya, and M.-Y. Liu. One-shot free-view neural talking-head synthesis for video conferencing. In *CVPR*, pages 10039–10049, 2021.

[170] X. Wang, R. Girshick, A. Gupta, and K. He. Non-local neural networks. In *CVPR*, 2018.

[171] X. Wang, H. Yuan, S. Zhang, D. Chen, J. Wang, Y. Zhang, Y. Shen, D. Zhao, and J. Zhou. Videocomposer: Compositional video synthesis with motion controllability. *arXiv preprint arXiv:2306.02018*, 2023.

[172] Y. Wang, M. Li, H. Cai, W.-M. Chen, and S. Han. Lite pose: Efficient architecture design for 2d human pose estimation. In *CVPR*, 2022.

[173] Y. Wang, Z. Xu, X. Wang, C. Shen, B. Cheng, H. Shen, and H. Xia. End-to-end video instance segmentation with transformers. *CVPR*, 2021.

[174] Z. Wang, L. Chen, S. Rathore, D. Shin, and C. Fowlkes. Geometric pose affordance: 3d human pose with scene constraints. *arXiv preprint arXiv:1905.07718*, 2019.

[175] Z. Wang, D. Shin, and C. C. Fowlkes. Predicting camera viewpoint improves cross-dataset generalization for 3d human pose estimation. In *ECCV*, pages 523–540. Springer, 2020.

[176] Z. Wang, J. Yang, and C. Fowlkes. The best of both worlds: Combining model-based and nonparametric approaches for 3d human body estimation. In *CVPR ABAW workshop*, 2022.

[177] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh. Convolutional pose machines. In *CVPR*, 2016.

[178] O. Wiles, A. Koepke, and A. Zisserman. X2face: A network for controlling face generation using images, audio, and pose codes. In *ECCV*, pages 670–686, 2018.

[179] J. Z. Wu, Y. Ge, X. Wang, W. Lei, Y. Gu, W. Hsu, Y. Shan, X. Qie, and M. Z. Shou. Tune-a-video: One-shot tuning of image diffusion models for text-to-video generation. *ICCV*, 2023.

[180] C.-h. Wuu, N. Zheng, S. Ardisson, R. Bali, D. Belko, E. Brockmeyer, L. Evans, T. Godisart, H. Ha, A. Hypes, T. Koska, S. Krenn, S. Lombardi, X. Luo, K. McPhail, L. Millerschoen, M. Perdoch, M. Pitts, A. Richard, J. Saragih, J. Saragih, T. Shiratori, T. Simon, M. Stewart, A. Trimble, X. Weng, D. Whitewolf, C. Wu, S.-I. Yu, and Y. Sheikh. Multiface: A dataset for neural face rendering. In *arXiv*, 2022.

[181] B. Xiao, H. Wu, and Y. Wei. Simple baselines for human pose estimation and tracking. In *ECCV*, 2018.

[182] R. Xie, C. Wang, and Y. Wang. Metafuse: A pre-trained fusion model for human pose estimation. In *CVPR*, 2020.

[183] S. Xie and Z. Tu. Holistically-nested edge detection. In *ICCV*, pages 1395–1403, 2015.

[184] Y. Xiong, H. Ma, S. Sun, K. Han, and X. Xie. Light field diffusion for single-view novel view synthesis. *arXiv preprint arXiv:2309.11525*, 2023.

[185] Q. Xu, Z. Xu, J. Philip, S. Bi, Z. Shu, K. Sunkavalli, and U. Neumann. Point-nerf: Point-based neural radiance fields. In *CVPR*, pages 5438–5448, 2022.

[186] S. Yan, Y. Xiong, and D. Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *AAAI*, 2018.

[187] X. Yan, H. Tang, S. Sun, H. Ma, D. Kong, and X. Xie. After-unet: Axial fusion transformer unet for medical image segmentation. In *WACV*, 2022.

[188] H. Yang, H. Zhu, Y. Wang, M. Huang, Q. Shen, R. Yang, and X. Cao. Facescape: a large-scale high quality 3d face dataset and detailed riggable 3d face prediction. In *CVPR*, pages 601–610, 2020.

[189] K. Yang, K. Chen, D. Guo, S.-H. Zhang, Y.-C. Guo, and W. Zhang. Face2face ρ: Real-time high-resolution one-shot face reenactment. In *ECCV*, pages 55–71, 2022.

[190] S. Yang, W. Heng, G. Liu, G. LUO, W. Yang, and G. YU. Capturing the motion of every joint: 3d human pose and shape estimation with independent tokens. In *ICLR*, 2023.

[191] S. Yang, Z. Quan, M. Nie, and W. Yang. Transpose: Towards explainable human pose estimation by transformer. *ICCV*, 2021.

[192] S. Yang, Y. Zhou, Z. Liu, and C. C. Loy. Rerender a video: Zero-shot text-guided video-to-video translation. *ACM SIGGRAPH Asia Conference Proceedings*, 2023.

[193] G. Yao, Y. Yuan, T. Shao, and K. Zhou. Mesh guided one-shot face reenactment using graph convolutional networks. In *Proceedings of the 28th ACM international conference on multimedia*, pages 1773–1781, 2020.

[194] Y. Yoshiyasu. Deformable mesh transformer for 3d human mesh recovery. In *CVPR*, pages 17006–17015, 2023.

[195] C. You, N. Chen, and Y. Zou. MRD-Net: Multi-Modal Residual Knowledge Distillation for Spoken Question Answering. In *IJCAI*, 2021.

[196] C. You, N. Chen, and Y. Zou. Self-supervised contrastive cross-modality representation learning for spoken question answering. *arXiv preprint arXiv:2109.03381*, 2021.

[197] C. You, R. Zhao, F. Liu, S. Chinchali, U. Topcu, L. Staib, and J. S. Duncan. Class-aware generative adversarial transformers for medical image segmentation. *arXiv preprint arXiv:2201.10737*, 2022.

[198] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang. Bisenet: Bilateral segmentation network for real-time semantic segmentation. In *ECCV*, pages 325–341, 2018.

[199] C. Yu, B. Xiao, C. Gao, L. Yuan, L. Zhang, N. Sang, and J. Wang. Lite-hrnet: A lightweight high-resolution network. In *CVPR*, 2021.

[200] J. Yu, X. Li, J. Y. Koh, H. Zhang, R. Pang, J. Qin, A. Ku, Y. Xu, J. Baldridge, and Y. Wu. Vector-quantized image modeling with improved VQGAN. In *ICLR*, 2022.

[201] L. Yu, Y. Cheng, K. Sohn, J. Lezama, H. Zhang, H. Chang, A. G. Hauptmann, M.-H. Yang, Y. Hao, I. Essa, et al. Magvit: Masked generative video transformer. In *CVPR*, pages 10459–10469, 2023.

[202] S. Yu, T. Chen, J. Shen, H. Yuan, J. Tan, S. Yang, J. Liu, and Z. Wang. Unified visual transformer compression. In *ICLR*, 2022.

[203] L. Yuan, Y. Chen, T. Wang, W. Yu, Y. Shi, Z.-H. Jiang, F. E. Tay, J. Feng, and S. Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. In *ICCV*, 2021.

[204] E. Zakharov, A. Ivakhnenko, A. Shysheya, and V. Lempitsky. Fast bi-layer neural synthesis of one-shot realistic head avatars. In *ECCV*, pages 524–540, 2020.

[205] E. Zakharov, A. Shysheya, E. Burkov, and V. Lempitsky. Few-shot adversarial learning of realistic neural talking head models. In *ICCV*, pages 9459–9468, 2019.

[206] B. Zeng, B. Liu, H. Li, X. Liu, J. Liu, D. Chen, W. Peng, and B. Zhang. FNeVR: Neural volume rendering for face animation. In *NeurIPS*, 2022.

[207] L. Zhang and M. Agrawala. Adding conditional control to text-to-image diffusion models. *ICCV*, 2023.

[208] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, pages 586–595, 2018.

[209] W. Zhang, X. Cun, X. Wang, Y. Zhang, X. Shen, Y. Guo, Y. Shan, and F. Wang. Sadtalker: Learning realistic 3d motion coefficients for stylized audio-driven single image talking face animation. In *CVPR*, pages 8652–8661, 2023.

[210] W. Zhang, J. Fang, X. Wang, and W. Liu. Efficientpose: Efficient human pose estimation with neural architecture search. *Computational Visual Media*, 2021.

[211] X. Zhang, Q. Li, H. Mo, W. Zhang, and W. Zheng. End-to-end hand mesh recovery from a monocular rgb image. In *ICCV*, 2019.

[212] Y. Zhang, Y. Wei, D. Jiang, X. Zhang, W. Zuo, and Q. Tian. Controlvideo: Training-free controllable text-to-video generation. *arXiv preprint arXiv:2305.13077*, 2023.

[213] Z. Zhang, L. Han, A. Ghosh, D. N. Metaxas, and J. Ren. Sine: Single image editing with text-to-image diffusion models. In *CVPR*, pages 6027–6037, 2023.

[214] Z. Zhang, C. Wang, W. Qiu, W. Qin, and W. Zeng. Adafuse: Adaptive multiview fusion for accurate human pose estimation in the wild. *International Journal of Computer Vision*, 2021.

[215] Y. Zhao, S. Kong, and C. Fowlkes. Camera pose matters: Improving depth prediction by mitigating pose distribution bias. In *CVPR*, pages 15759–15768, 2021.

[216] C. Zheng, X. Liu, G.-J. Qi, and C. Chen. Potter: Pooling attention transformer for efficient human mesh recovery. In *CVPR*, pages 1611–1620, 2023.

[217] C. Zheng, S. Zhu, M. Mendieta, T. Yang, C. Chen, and Z. Ding. 3d human pose estimation with spatial and temporal transformers. *ICCV*, 2021.

[218] S. Zheng, J. Lu, H. Zhao, X. Zhu, Z. Luo, Y. Wang, Y. Fu, J. Feng, T. Xiang, P. H. Torr, et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. *CVPR*, 2021.

[219] Y. Zheng, W. Yifan, G. Wetzstein, M. J. Black, and O. Hilliges. Pointavatar: Deformable point-based head avatars from videos. In *CVPR*, 2023.

[220] X. Zhu, Z. Lei, X. Liu, H. Shi, and S. Z. Li. Face alignment across large poses: A 3d solution. In *CVPR*, pages 146–155, 2016.

[221] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai. Deformable detr: Deformable transformers for end-to-end object detection. *ICLR*, 2021.

[222] Y. Zhuang, H. Zhu, X. Sun, and X. Cao. Mofanerf: Morphable facial neural radiance field. In *ECCV*, 2022.

[223] C. Zimmermann, D. Ceylan, J. Yang, B. Russell, M. Argus, and T. Brox. Freihand: A dataset for markerless capture of hand pose and shape from single rgb images. In *ICCV*, 2019.