Generating Dance Motion using Musical Features

Wan Yi Lin

Department of Music Technology

McGill University

Montréal, Québec, Canada

August 2022

A thesis submitted to McGill University in partial fulfillment of the requirements for the degree of

Master of Arts.

Abstract

In a computer animation workflow, an animator has to plot keyframes and adjust the in-between frames to create or edit a motion, and motion is represented by keyframes and in-between frames that connect the keyframes. The sequence of significant poses, represented by keyframes, determine the general motion in animations. In-between frames are generated by interpolation strategies using keyframes, such as linear interpolation or parametric curve equation, where the starting and ending point of the parametric curve set to the two adjacent selected keyframes. In-between frames determine the trajectory and the speed an animated object moves from one keyframe to the next. Since in-between frames are reconstructed from adjacent keyframes, the selection of keyframes influences the reconstruction distance of reconstructed in-betweens, and a great amount of research has focused on strategies to select appropriate keyframes.

Dance motion and dance music are often closely related to each other. Choreography is generally designed to synchronize with the rhythm of the music, and various deep learning research topics have been proposed to choreograph dance motion from dance music. In this thesis, inspired by previous research studying the close relation between dance and music, I use musical features of dance music to select keyframes and examine the reconstruction distance of

the generated dance motion based on the AIST++ dataset. I hypothesize that the close relation between dance and music may improve reconstruction of dance motions. Three experiments are designed to evaluate the effect of musical features when reconstructing dance motions.

The first experiment is used as a baseline and does not use musical features in both keyframe selection and in-between reconstructions. Keyframes are selected evenly over the measure. The in-between dance motions are reconstructed with linear interpolation and Bézier curve fitting.

The second experiment uses musical features (onset peak positions) to select keyframes. In-between frames are reconstructed in the same way as in the first experiment, using interpolations.

In the third experiment, keyframes are selected in the same way as in the first experiment (evenly), and neural network models are used to reconstruct in-between frames using musical features, such as Mel-frequency cepstral coefficients, onset strength, and onset peak positions.

Two evaluation metrics are used for assessing the reconstructed dance motion: The Frechet Inception Distance and mean Euclidean distance between the reconstructed dance and the real dance is used to evaluate the reconstruction distance. These metrics answer the question of whether injecting musical features improves the distance of the reconstructed dance motion. Experimental results suggest that when using fewer keyframes, strategies to select keyframes and reconstruct in-between frames using dance music improve the distance of reconstructed dances.

Résumé

Dans un flux de travail d'animation par ordinateur, un animateur doit tracer des images clés et ajuster les images intermédiaires pour créer ou modifier un mouvement. Le mouvement est représenté par des images clés et des images intermédiaires qui relient les images clés. La séquence de poses significatives, représentée par des images clés, détermine le mouvement général dans les animations. Les images intermédiaires sont générées par des stratégies d'interpolation utilisant des images clés, telles que l'interpolation linéaire ou l'équation de courbe paramétrique, où le point de départ et le point d'arrivée de la courbe paramétrique sont fixés aux deux images clés adjacentes sélectionnées. Les images intermédiaires déterminent la trajectoire et la vitesse de déplacement d'un objet animé d'une image clé à l'autre. Bien qu'elles aient peu d'influence sur le contenu d'un mouvement, elles en influencent le style. Puisque les images intermédiaires sont reconstruites à partir d'images clés adjacentes, la sélection des images clés influence la précision de la reconstruction des images intermédiaires reconstruites, et de nombreuses recherches se sont concentrées sur les stratégies de sélection d'images clés appropriées.

Les mouvements de danse et la musique de danse sont souvent étroitement liés les uns aux

autres. La chorégraphie est généralement conçue pour être synchronisée avec le rythme de la musique, et divers sujets de recherche en apprentissage profond ont été proposés pour réaliser la chorégraphie de mouvements de danse à partir de la musique de danse. Dans cette thèse, inspirée par des recherches précédentes étudiant la relation étroite entre la danse et la musique, j'utilise les caractéristiques musicales de la musique de danse pour sélectionner les images clés et examiner la précision de reconstruction du mouvement de danse généré sur la base du jeu de données AIST++. J'émets l'hypothèse que la relation étroite entre la danse et la musique peut améliorer la reconstruction des mouvements de danse. Trois expériences sont conçues pour évaluer l'effet des caractéristiques musicales lors de la reconstruction des mouvements de danse.

La première expérience est utilisée comme base de référence et n'utilise pas les caractéristiques musicales dans la sélection des images clés et dans les reconstructions intermédiaires. Les images clés sont sélectionnées de manière uniforme sur la mesure. Les mouvements de danse intermédiaires sont reconstruits par interpolation linéaire et ajustement de la courbe de Bézier.

La deuxième expérience utilise des caractéristiques musicales (positions des pics de onset) pour sélectionner les images clés. Les images intermédiaires sont reconstruites de la même manière que dans la première expérience, en utilisant des interpolations.

Dans la troisième expérience, les images clés sont sélectionnées de la même manière que dans la première expérience (uniformément), et des modèles de réseaux neuronaux sont utilisés pour reconstruire les images intermédiaires à l'aide de caractéristiques musicales, telles que Mel fréquence Cepstral Coefficients, l'enveloppe d'attaque et la détection de la position d'attaque.

Deux mesures d'évaluation sont utilisées pour évaluer le mouvement de la danse reconstruite : La Fréchet Inception Distance et la distance euclidienne moyenne entre la danse reconstruite et la danse réelle sont utilisées pour évaluer la précision de la reconstruction. Ces mesures répondent à la question de savoir si l'injection de caractéristiques musicales améliore la précision du mouvement de danse reconstruit. Les résultats expérimentaux suggèrent que lorsqu'on utilise moins d'images clés, les stratégies de sélection des images clés et de reconstruction entre les images en utilisant de la musique de danse améliorent la précision des danses reconstruites.

Acknowledgements

I would like to thank my supervisor Ichiro Fujinaga for guiding me in my research and giving me feedback during the writing process of this thesis. I would like to my family for supporting me pursue my degree and research. Many thanks to my friend's weekly emotional support for helping me go through the revising process and translating and proofreading the abstract to French, and I would like to thank my colleague at the Distributed Digital Music Archives and Libraries laboratory for the inspiring idea to polish this thesis. Finally, I would like to thank Tim, Martha, and Néstor for taking the time to give feedback on my thesis proposal.

Contents

List of Figures

List of Tables

List of Acronyms

CNN	Convolution Neural Network.
FID	Frechet Inception Distance.
MFCC	Mel-Frequency Cepstral Coefficients.
MSE	Mean Squared Error.
NLP	Natural Language Processing.
NMF	Non-negative Matrix Factorization.
NMFD	Non-negative Matrix Factorization Deconvolution.
NN	Neural Network.
RNN	Recurent Neural Network.
STFT	Short-time Fourier Transform.

Chapter 1

Introduction

In a computer animation workflow, an animator has to plot keyframes and adjust the in-between frames to create an animation. An animated motion is represented by keyframes and in-between frames that connect the keyframes. *Keyframes* represent significant poses. They tell the story in each shot and provide the overall trajectory of the motion. *In-between* frames are generated by interpolation strategies using keyframes, such as linear interpolation or parametric curve equations, where the starting and ending points of the parametric curve equations are set to the two adjacent selected keyframes. In-between frames determine the detailed trajectory and the speed an animated object moves from one keyframe to the next. Though they have little influence on the overall position of a motion, they impact the style of a motion. For example, in a gait animation, keyframes decide the content of a gait animation, such as its overall trajectory, hand poses when each time the character's feet contact the floor, and the speed of the gait animation. The in-between frames do not change the content but influence

the style of a motion, such as walking calm or desperate, lazy or energetic, drunk or frightened (Williams 2012).

There are a significant amount of research topics focusing on converting animation data into keyframes and in-between frames. Given motion data, strategies are proposed to select keyframes, such as designing dance motions or selecting frames in motion data with significant features as keyframes, and using clustering algorithms to cluster frames with significant features and select the most significant frames as keyframes from each cluster. After selecting keyframes, in-between frames are derived using selected keyframes, and strategies such as linear interpolation and parametric curve equation are used to derive in-between frames. However, in these research topics, dance motion is regarded as one of the tested motion data, such as walk, run, and jump motion. While the close relation between dance motion and dance music has been investigated in previous research (Solberg and Jensenius 2019), this relation was not considered in strategies to select keyframes and in-betweens from dance motion. Consequently, in this thesis, I will use musical features to select keyframes and reconstruct in-betweens and examine the reconstruction distance of the generated dance motion. I hypothesize that the close relation between dance and music may improve reconstruction of dance motion, and I will use metrics for measuring the reconstruction distance of the generated dance motion to evaluate two main strategies to select keyframes and in-betweens: using motion only and using musical features and motion.

1.1 Thesis Organization

This thesis is organized into five chapters. Chapter 1 is the introduction, which gives this thesis's background and research objective. Chapter 2 is the Literature Overview that provides backgrounds and previous research in representing motion, neural networks, extracting keyframes and reconstructing in-betweens, analyzing dance motion and dance music, metrics evaluating dance motion, and dance datasets. Chapter 3 is the Methods that explains the experimental workflows to extract keyframes and reconstruct in-betweens frames using different features: musical features and dance motion. The reconstructed dances are evaluated with metrics to examine their reconstruction distance. Chapter 4, the Experiments section, presents and discusses the experimental results. Finally, Chapter 5 is the Conclusion that discusses and concludes the experimental results and provides an overview of possible future work.

Chapter 2

Background

This chapter presents information about analyzing dance motion and music, generating dance motion from music, neural networks, and reconstructing motion using keyframes and in-betweens. Section 2.1 provides an overview of capturing motion in photographs and motion capture systems. Section 2.2 presents general overview of representing a 3D motion in computer animation. Section 2.3 introduces neural networks and their application. Section 2.4 presents existing works on converting a motion into keyframes and in-betweens. Section 2.5 and Section 2.6 present works on analyzing dance motion and music. Section 2.7 continues its previous sections and introduces works on generating dance motion from dance music. Section 2.8 presents metrics to evaluate generated dances. Finally, Section 2.9 introduces dance datasets that are essential for machine learning models to generate dances.

2.1 Understanding Human Motion

This section discusses techniques to sample and record the motion of human, animal, or moving objects. The history of capturing motion in photographs is discussed in Section 2.1.1, and the introduction of motion capture systems is presented in Section 2.1.2

2.1.1 Motion in Photograph

Eadweard Muybridge (1830–1904) was known for capturing a galloping horse by a sequence of photographs and proving that a galloping horse had no feet touching the ground (Kitagawa and Windsor 2008). In 1879, Eadweard Muybridge invented the *zooproxiscope*, one of the earliest devices to display motion with sequential images. Etienne-Jules Marey (1830–1904) developed chronophotography, a high-speed photographic technique to capture successive frames on a single photographic plate (Blake and Shiffrar 2007). He filmed a person walking while wearing a black suit with markers attached to joints and lines connecting joints. The result was a photograph of the changing positions of the joints. In the 1970s, Gunnar Johansson devised the point-light display, where the motion was represented by positions of bright spots attached to the subject (Jensenius 2013). Taps made of reflective materials were attached to the subject's joints, and lights hitting the taps were reflected back and recorded by cameras.

2.1.2 Motion Capture System

Motion capture (mocap) systems commonly used today are able to capture the movements of a subject. They are categorized as optical mocap systems and non-optical mocap systems.

The optical mocap system describes motions as consecutive 3-d positions of the markers and requires subjects to wear either reflective or light-emitting markers. Markers are often attached to joints. A multi-camera system is in charge of capturing the lights either reflected or emitted by markers, and the 3-d position of each joint is reconstructed from frames captured by the multi-camera system (Zhu and Li 2016).

The inertial mocap system is one of the non-optical mocap systems. Subjects wear inertial measurement units (IMUs), typically including gyroscopes, accelerometers, and magnetometers. Data from IMUs are used to determine the position and movement of each IMU sensor. Cameras are not required in an inertial mocap system, and the occlusion problem in the optical mocap system is eliminated (Yahya et al. 2019).

2.2 Representing Human Motion

This section contains background knowledge for representing human motion in computer animation. Section 2.2.1 discusses 3-d coordinate systems to represent a 3-d point in computer animation. Section 2.2.2 explains the transformation matrix, which is the way to represent a point's 3-d position in a coordinate system. Section 2.2.3 explains different representations to represent a rotation in a 3-d coordinate system. Section 2.2.4 describes the tree-like data structure representing a humanoid motion. Finally, Section 2.2.5 describes methods to reconstruct 3-d positions of a humanoid motion, including multi-camera systems, camera parameters, and joint detection techniques.

2.2.1 World Coordinate and Local Coordinate

In computer graphics, when a point is represented as its 3-d coordinates, a canonical coordinate system, which is referred to as the *world coordinate system* (Shirley, Ashikhmin, and Marschner 2009) or *global coordinate system*, needs to be defined in order to represent the 3-d coordinates. Its origin can be defined at an arbitrary position, and for an object, its transformation, consisting of translation, rotation, and scaling, is defined relative to the origin. Coordinate systems can be added to the world coordinate and represented as their transformation in the world coordinate. Additional coordinate systems are referred to as *local coordinate systems*, and with a local coordinate system, instead of being represented as the transformation in the world coordinate, an object could be represented as the transformation in a local coordinate system (Salomon 2006). Multiple local coordinate systems can be chained together to form a hierarchical structure. Considering a hierarchical structure that contains multiple objects, the 3-d positions of objects are represented as transformations with respect to chained local coordinate systems of the hierarchical structure. By doing this, one can change the transformation of the top-most coordinate system in the structure to move the whole hierarchical structure while keeping the relative object positions.

2.2.2 Transformation Matrix

A 3-d transformation of an object or coordinate system represents a series of operations: moving the object to a location, scaling the object, and rotating the object (Shirley, Ashikhmin, and Marschner 2009, Chapter 6). These operations are represented by a 4×4 *transformation matrix*,

which consists of an upper-left 3×3 *rotation matrix*, an upper-right 1×3 *translation matrix*, and a lower 1×4 matrix. The transformation matrix transforms a point or vector in 3-d space, and a point or vector in 3-d space is represented as a 4-d vector. The last element of the 4-d vector is set to 1 to represent a point and 0 to represent a vector. Given a 4-d transformation matrix $M \in R^{4\times4}$ and a point or vector $x \in R^{4\times1}$, a new point or vector $y \in R^{4\times1}$ transformed from x using the transformation matrix M is computed by matrix multiplication $y = Mx$ (Bloomenthal and Rokne 1994, Bellekens et al. 2014).

2.2.3 Rotation Representation

Even though a transformation matrix uses a 3x3 rotation matrix to represent the rotation of an object, the rotation can be represented in different ways, such as Euler-angle representation, axis-angle representation, or Quaternion. These representations can be converted to 3x3 rotation matrices to form a transformation matrix. Each representation is briefly introduced in this section. For detailed mathematical proof of conversion to rotation matrix, please refer to Sarabandi and Thomas (2019), Diebel (2006), and Navaza (1990).

Euler-angle representation

Euler-angle representation defines rotation as three rotation angles regarding to x, y, and z axis. The mathematical representation of rotating θ angle regarding to x, y, and z axis is represented in Equation 2.1, 2.2, 2.3, respectively, and the rotation matrix R is represented by a series of matrix multiplication of three matrices in Equation 2.4 (Foley et al. 1996, Chapter 11).

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\theta) & -sin(\theta) \\ 0 & sin(\theta) & cos(\theta) \end{bmatrix} \tag{2.1}$$

$$R_z(\theta) = \begin{bmatrix} cos(\theta) & 0 & sin(\theta) \\ 0 & 1 & 0 \\ -sin(\theta) & 0 & cos(\theta) \end{bmatrix} \tag{2.2}$$

$$R_z(\theta) = \begin{bmatrix} cos(\theta) & -sin(\theta) & 0 \\ sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.3}$$

$$R(\theta) = R_x(\theta)R_y(\theta)R_z(\theta) \tag{2.4}$$

Axis-angle representation

A rotation in 3-d space can be represented by rotating an angle concerning a rotation axis (Foley et al. 1996, Chapter 11). This is referred to as the *axis-angle representation*. A rotation

represented in axis-angle representation is stored as a 3-d vector, where the rotation axis is the stored vector, and the norm of the vector is the rotation angle.

Quaternion

Quaternion is a four-parameter vector representing a rotation (Shepperd 1978). A quaternion q is composed of a 3-d vector v and a scalar s as $q = [s; \mathbf{v}]$. Operations for quaternions are defined, such as adding two quaternions, multiplying a vector and a quaternion, and inversing quaternion (Shirley, Ashikhmin, and Marschner 2009). A 4-d quaternion can represent the rotation of degree θ around an axis represented by a vector n, and it is formulized as $q = [cos(\frac{\theta}{2}), nsin(\frac{\theta}{2})]$. When rotating a point $\mathbf{p}$ in its quaternion representation $q_p = [0, \mathbf{p}]$, using the quaternion q defined eariler, the quaternion of the rotated point q'_p is calculated by $q'_p = qq_pq^{-1}$ (Shirley, Ashikhmin, and Marschner 2009, Chapter 16).

2.2.4 Skeleton

In computer animation, a humanoid motion is often represented by a *skeleton* (Shirley, Ashikhmin, and Marschner 2009, Chapter 16), as shown in Figure 2.1, which is a tree representing the structure of the humanoid skeleton. In the tree structure, the root represents the whole character's 3-d transformation in the world coordinate, and nodes represent joints in body parts, such as ankles, wrists, and elbows, and edges represent bones, such as upper arms and lower arms. The transformation of each node, which explains its rotation, translation, and scale, is relative to the local coordinate system centered at its parent node.

For a node containing child nodes, its transformation also defines a new coordinate system centered at the node, and for each child node, the new coordinate system serves as the local coordinate system centered at its parent node. By doing so, a skeleton is parameterized by chained transformations of each node relative to its parent node.

The transformation of each node in the world coordinate system is computed by *Forward Kinematics* (Kucuk and Bingul 2006). Forward Kinematics tracks down all the parent nodes until the target node and keeps their transformation matrices. Then it sequentially applies tracked transformation matrices to convert the target node's transformation from its local coordinate system to the world coordinate system (Shirley, Ashikhmin, and Marschner 2009, Chapter 16).

2.2.5 Detecting Joint Positions

This section introduces methods to reconstruct 3-d joint positions in a humanoid skeleton described in Section 2.2.4, which is important for representing a dance motion in terms of the 3-d positions of each joint. For an image containing people, the 2-d joint positions of the person, represented as the x and y coordinate in the image, can be detected using *2-d pose estimators* (Cao et al. 2017, Rogez, Weinzaepfel, and Schmid 2019, Newell, Yang, and Deng 2016). The 3-d joint positions of a subject can be reconstructed using 2-d pose estimators and a multi-camera system. A *multi-camera system* is an arrangement of a set of cameras filming the same location (Olagoke, Ibrahim, and Teoh 2020), and for detecting 3-d joint positions, the camera parameters of each camera need to be reconstructed. *Camera parameters* describe

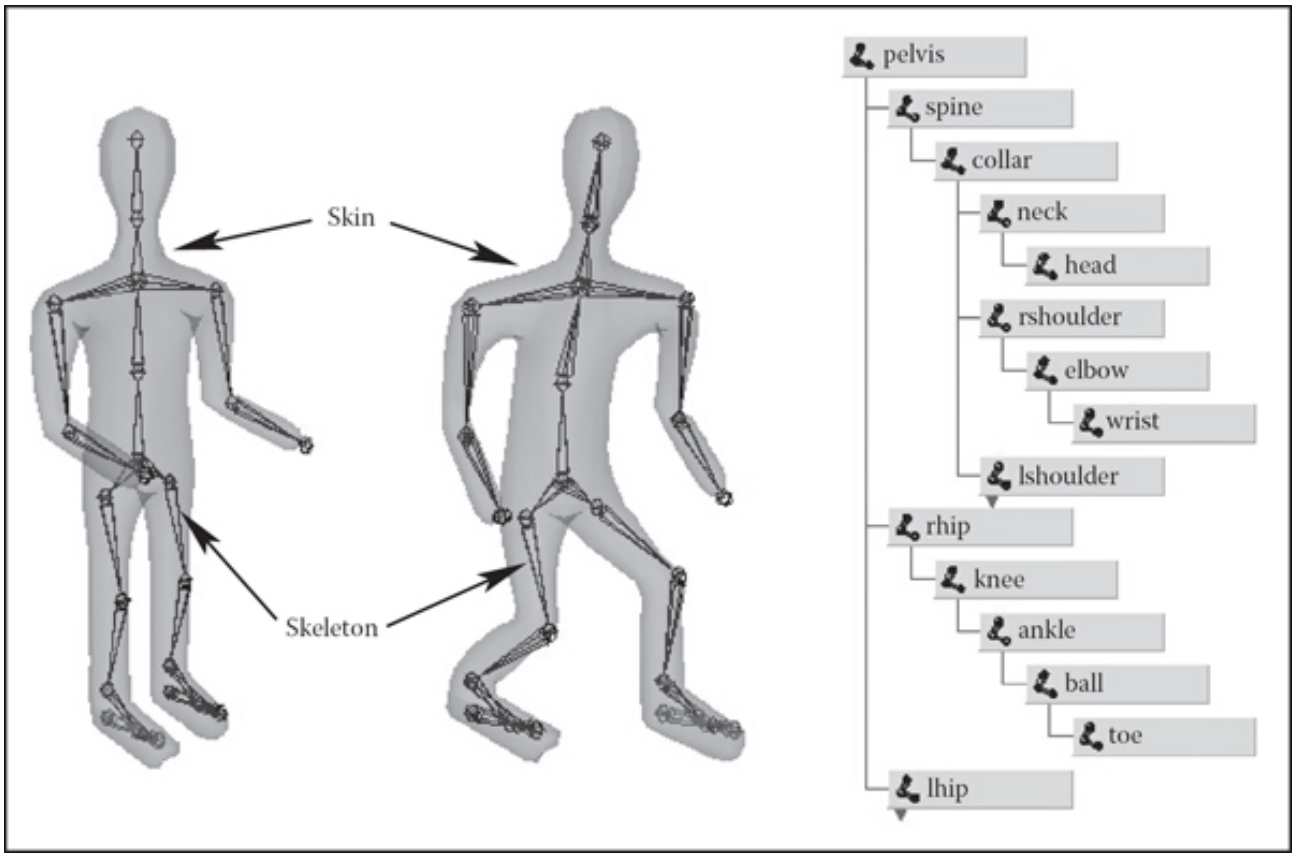

Figure 2.1: (Left, Middle) A hierarchy of humanoid skeleton (Shirley, Ashikhmin, and Marschner 2009, Chapter 16). (Right) A humanoid skeleton represented with a tree structure.

the transformation matrix of a camera in the world coordinate system and how a 3-d point is projected to its 2-d position in the image filmed by the camera. Several research topics have been proposed working on reconstructing camera parameters of a multi-camera system (Triggs et al. 1999, Zhang 2000). For detailed review on reconstructing camera parameters, please refer to Li et al. (2013) and Remondino and Fraser (2006). After reconstructing the camera parameters of each camera in a multi-camera system, a 2-d pose detector is used to detect 2-d joint positions in the image filmed by each camera. The 3-d joint positions can be reconstructed using the reconstructed camera parameters and the detected 2-d joint positions in each camera.

For details regarding the reconstruction process, please refer to Frahm, Köser, and Koch (2004).

2.3 Neural Network

This section gives an overview of neural networks and their applications. Section 2.3.1 introduces the history and basic knowledge about neural networks; Section 2.3.2 discusses the application of the convolution neural network; Section 2.3.3 introduces the recurrent neural network and its application; finally, Section 2.3.4 describes the Transformer model, its difference with the convolution neural network and recurrent neural network, and its application.

2.3.1 Introduction

With the recent development of computational resources, neural network models have become tools for academic research, and they brought benefits to research related to Computer Vision, Natural Language Processing, Audio, and Music. The most common form of deep learning is supervised learning (LeCun, Bengio, and Hinton 2015). In supervised learning, a set of data is manually annotated with ground-truth labels, and a deep neural network is trained to take data as input and map the input data to its correct label. Annotated data is split into three sets: training set, validation set, and testing set. During the training time, a deep learning model learns to map from input to its manually annotated label using the training set. The validation set is used to evaluate the performance of the model and tune parameters during the training process. The model with the best performance evaluated with the validation set is

selected as the trained neural network. Finally, after finishing training and selecting the neural network with the best performance, the testing set is used to evaluate the performance of the trained neural network (Yamashita et al. 2018).

A neural network model consists of internal trainable parameters and hyperparameters. Trainable parameters are often referred to as "weights," and they allow the neural network model to map an input to its label. Hyperparameters refer to parameters needed to be decided before the training process starts. During the training process, training data are sent to a neural network model, and their labels are predicted with weights. A loss function is used to measure the error between the predicted label and the annotated label, and optimization techniques, such as Gradient Descent, are used to update weights such that the error is minimized during the next prediction (Yamashita et al. 2018, LeCun, Bengio, and Hinton 2015).

The history of neural networks increased in the 1980s (Yoo 2015). However, it reached a significant amount of progress with improving computational resources. AlexNet, VGG, googLeNet, and Resnet are well-known neural network models in image classification tasks (Minaee et al. 2021). In these neural network models, images are sent to the neural network and processed with kernels that slide through the entire image and compute an output in each location. Kernels are the trainable weights in these neural network architectures, and a neural network may consist of multiple layers, each containing multiple kernels. Neural networks using this architecture are referred to as convolution neural networks (CNN) (Yamashita et al. 2018).

2.3.2 Convolution Neural Network

Other than image classification, convolution neural networks have multiple applications, such as object detection (Ren et al. 2015, He et al. 2017, Redmon et al. 2016, He et al. 2015), where given an input image, a convolution neural network recognizes objects' label and mark their bounding boxes (Li et al. 2021b), and image segmentation (Zhao et al. 2017, Badrinarayanan, Kendall, and Cipolla 2017, Ronneberger, Fischer, and Brox 2015), where a neural network divides an input image into different regions and predicts one label for each region. Besides images, convolution neural network can process different data, such as 1-d CNN that process audio in time-domain representation (Abdoli, Cardinal, and Koerich 2019, Li et al. 2019), 2-d CNN processing audio in time-frequency representation (Huzaifah 2017, Donahue, McAuley, and Puckette 2019, Xie et al. 2019), and 3-d CNN processing video (Huang, Guo, and Gao 2020, Qiu, Yao, and Mei 2017, Yan et al. 2019).

2.3.3 Recursive Neural Network

Different from convolution neural networks, which cannot deal with temporal states behind the input data, recursive neural network (RNN) contains cyclic connections, and the internal states, represented as trainable weights, are updated using the cyclic connection (Yu et al. 2019). Because of the cyclic connection and internal states, RNN is adequate for modeling temporal information and processing data with variable input length. With this characteristic, the application of RNN is mainly in natural language processing, including text semantic classification (Dong et al. 2014), machine translation (Cho et al. 2014), and

generating dialogue (Tran and Nguyen 2017). Other than natural language processing, RNN can be used to process video, such as action recognition (Zhao, Ali, and Van der Smagt 2017, Ullah et al. 2017) and pose detection (Luo et al. 2018). It is also used to process audio, such as speech recognition (Zhang et al. 2016, Irie et al. 2016) and music-to-score alignment (Kwon, Jeong, and Nam 2017).

2.3.4 Transformer

Besides CNN and RNN, the Transformer model has significantly impacted the deep learning research recently (Tay et al. 2022). Compared to CNN, a Transformer model can capture the long-range dependency using the self-attention mechanism (Han et al. 2022). Though RNN can keep the long-range dependencies of training data, it relies on the sequential cyclic connection, which disables parallel computing. As a result, data must be computed sequentially, which increases its training time (Vaswani et al. 2017). Transformers fix these issues by entirely relying on self-attention layers, where each sample is connected to the other sample to compute attention weights, which determine the degree of attention between each pair. By doing so, it allows the model to learn the long-range dependency. Besides, the computation of self-attention is not recursive, and it can be computed parallelly using matrix multiplication, which speeds up the training process (Han et al. 2022). Moreover, there are multiple self-attention layers in the transformer model, which allow the model to learn multiple attention weights. This implies that each sample can have different weights to describe multiple degrees of attention regarding other samples. A self-attention layer is computed from three inputs: Key, Query, and Value,

and they serve as the input to the transformer model.

Besides the self-attention layers, a Transformer model also consists of an encoder and decoder models (Tay et al. 2022). There are three variations of a Transformer model: encoder-only, decode-only, and encoder-decoder. The encoder and decoder both include multi-head self-attention layers computed with a query, key, and value as inputs. In the encoder model, training data is used as the query, key, and value to the multi-head self-attention layers. However, the output of the encoder model is sent as the key and value to the decoder model. A multi-head self-attention layer in the decoder module is computed using the output of the encoder as the key and value, and the input to the decoder as the query. These three variations can be used for different cases. For example, the encoder-only model can be used for classification tasks, the decoder-only model can be used for language modeling tasks, and the encoder-decoder model can be used for machine translation tasks (Tay et al. 2022).

The Transformer model was first proposed in Natural Language Processing (NLP) for machine translation tasks by Vaswani et al. (2017). It surpassed other models, such as recursive neural networks or convolutional neural networks. Transformer models can be pre-trained on generic tasks and fine-tuned to process downstream tasks (Wolf et al. 2020). Various research has worked on strategies to pre-train the Transformer model, including BERT (Devlin et al. 2019), where the model was pre-trained to predict masked words and fine-tuned on downstream tasks. Liu et al. (2019) conducted extensive research on hyperparameters and the procedures when training BERT, and by tuning the training procedures, it reached better results in NLP challenges such as language understanding,

Question-Answering system, and reading comprehension.

Transformer models are adapted to process data such as images, audio, and video. Dosovitskiy et al. (2021) adapted Transformer to achieve image recognition tasks, where the input image was split into patches and sent to the Transformer to recognize the content of the image. Ramesh et al. (2021) used Transformer models for text-to-image generation. A Transformer model was trained to model text and image tokens as a single stream of data, and given an input text, an image was generated using the modeled relation. Girdhar et al. (2019) proposed utilizing Transformer models to recognize and localize human actions. The action of human action depends on the spatial and temporal context of a video. For example, recognizing the action of "listening to someone" requires the prediction of another person, and recognizing the action of "watching someone" requires the understanding that the person of interest is not just staring into the distance. The model was trained to understand the spatial and temporal context of a video to recognize actions. Sun et al. (2019) proposed using contrastive learning to train the Transformer model for learning video representation, which could be fine-tuned to benefit downstream tasks such as video segmentation. In contrastive learning, the deep learning model is encouraged to minimize and maximize the distance between similar samples and different samples in a feature space. Lin, Wang, and Liu (2021) employed Transformer models to reconstruct 3D human pose and mesh from images. Dong, Xu, and Xu (2018) proposed Speech-Transformer to replace the recurrent neural network for speech recognition. However, as the Transformer model was originally proposed in NLP, the attention mechanism is designed to process 1-d temporal dependencies. To tackle

the 2-d time-frequency representation, a 2-d attention mechanism was proposed to compute attention for both temporal and spectral information. Hawthorne et al. (2021) proposed using Transformer models for piano transcription, where the input is the spectrogram of a music piece, and the model predicted events similar to MIDI-like vocabulary to describe the pitch, velocity, and time of each note.

2.4 Representing Motion in Keyframes and In-betweens

Representing a motion as keyframes and in-betweens originates from a traditional 2-d animation technique. Thomas, Johnston, and Thomas (1995) proposed twelve principles of animation. Among them, "Straight Ahead Action and Pose to Pose" is a principle regarding animation workflow. "Straight Ahead Action" means that an animator draws poses following their chronological order; "Pose to Pose" implies that an animator first draws key poses and fills up the in-between poses. The keyframe and in-between representation is applied to computer animation, where this representation allows users to edit a motion (Roberts 2018) and compress a motion, which represents a motion as keyframes and in-between frames to reduce the size of motion data and reach a more compact representation of motion data (Zhang et al. 2013). This section introduces strategies to select keyframes (Section 2.4.1) and reconstruct in-between frames (Section 2.4.2) from a motion.

2.4.1 Selecting Keyframes

Keyframes are representative poses to summarize the content of a motion (Miura et al. 2014), and they can be used to allow users efficiently browse and retrieve motion data (Lim and Thalmann 2001, Liu, Hao, and Zhao 2013). Miura et al. (2014) selected keyframes as frames whose poses are representative of describing the elementary structure of a motion, and the selected keyframes were used to summarize the content of motions. This allowed users to easily browse, edit, and reuse motions. The speed of each joint was calculated and filtered with a low-pass filter, frames whose speed was a local minimum were clustered, and frames whose speed was representative in each cluster were selected as keyframes. To efficiently store and browse motion data, Lim and Thalmann (2001) converted motion data into keyframes and in-between representation. They treated motion data as curves and recursively selected keyframes until the maximum distance between the original curve and the reconstructed curve was lower than a threshold. In each iteration, the curve was reconstructed by a straight line between selected keyframes. The frame that had a maximum distance between its value on the original curve and the reconstructed curve was selected as the new keyframe in the next iteration. Bulut and Capin (2007) converted motion data into keyframes and in-betweens to efficiently store a large amount of motion data. To select keyframes, they calculated the saliency value of each frame. For each frame on a motion curve, the saliency value was defined as the difference between two values calculated by a Gaussian weight filter centered at the frame but with a different standard deviation. Frames with higher saliency values were selected as candidate keyframes, and a clustering algorithm was applied to select the representative

keyframes from candidate keyframes to represent the motion. Halit and Capin (2011) applied Principle Component Analysis before calculating saliency values and used a clustering algorithm to select keyframes. In Togawa and Okuda (2005), the keyframe and in-between representation were used to convert motion data into a compact representation to retrieve required motion data from a large amount of motion data. All frames were selected as keyframes at first. A cost function was designed to evaluate the distortion caused by removing a keyframe. In each iteration, the keyframe with the least cost function value was removed. Keyframes were iteratively removed until the number of remaining keyframes matched the number of keyframes arbitrarily determined by users. To compress, store, browse and retrieve motion capture data, Liu, Hao, and Zhao (2013) employed a genetic algorithm to find a set of keyframes that, with the in-between motion being reconstructed from selected keyframes, reached the low reconstruction error with respect to joint positions and velocities. In order to browse and edit motion capture data, Huang et al. (2005) treated keyframe selection as a matrix factorization problem, and a sequence of motion represented as the 3-d positions of each joint is represented as a motion matrix. The motion matrix was computed by multiplying a keyframe matrix and a frame matrix, where the keyframe matrix was updated iteratively. In each round, a frame located furthest away from the reconstructed motion was selected as the new keyframe in the next round.

Roberts (2018) aimed to convert motion capture data into keyframes and in-betweens, which allows users to edit them. In the proposed method, keyframes were selected using dynamic programming, which requires iterative computation. The objective was to provide a

fast keyframe selection technique for computer animation software and, by converting a motion into keyframes and in-between frames, allow users to edit the motion. In this strategy, a sequence of dance motion with T frames in Euler-angle representation is a multidimensional curve with T points, and a *partial graph* of K points is referred to as the sub-sequence D containing the first K frames. A *keyframe matrix* records optimized selected keyframes in each partial graph, and a *cost matrix* saves the cost of selected keyframes in each partial graph. Before the iterative computation starts, costs for all combinations of selecting three keyframes from each subgraph are computed and saved to the cost matrix. The selection with the least cost in each partial graph is selected and recorded inside the keyframe matrix and cost matrix. After selecting three keyframes in each partial graph, the number of selected keyframes, starting from four, is increased by one in each iteration. In each iteration, the selected keyframes and their cost is optimized for all partial graphs, and the keyframe matrix and cost matrix are updated accordingly. The iterative process stops when the number of selected keyframes reaches the number of points inside the motion curve.

The sub-optimal structure of the subgraph reduces the computation cost to find the best keyframes, which reach the least cost in each subgraph. Reducing computation costs is important so as not to block the computer animation software. The possible costs of selecting k keyframes from a subgraph i of length t can be reduced to a collection of costs of selecting $k-1$ keyframes from subgraph j being shorter than t, plus the value of a cost function the calculated the cost when using j as the sub-optimal structure. Among all possible costs, the subgraph j with the lease cost is selected, and the selected k keyframes in subgraph i are the

selected $k-1$ keyframes inside subgraph j with the ending frame of subgraph j selected as the new keyframe. There are multiple ways to define a cost function. In Roberts (2018), the cost function takes three inputs, the ending point of subgraph i, the ending point of subgraph j, and the segment of subgraph i starting from subgraph j's ending point. The return value is calculated as the maximum distance between the value of a point on the segment and its value reconstructed by a curve linearly interpolated between two ending points.

2.4.2 Reconstructing In-betweens

The goal of reconstructing in-betweens is to use two adjacent keyframes to derive their intermediate frames. In such a case, intermediate frames can be represented by two keyframes, a reconstruction method to derive intermediate frames, and parameters for the reconstruction method to derive intermediate frames. This section introduces three methods to reconstruct in-between frames: linear interpolation, Bézier curve fitting, which are the two common methods for generating in-between frames in computer animation, and generating in-between frames with Transformer models.

For a motion represented as a series of poses, with each pose represented using a skeleton, the in-between frames can be reconstructed from performing linear interpolation using the adjacent keyframes (Lim and Thalmann 2001, Togawa and Okuda 2005, Liu, Hao, and Zhao 2013). For two keyframes kf_{t_1} and kf_{t_2} at timesteps t_1 and t_2, an in-between frames in_t at time t, where

$t_1 \leq t \leq t_2$, is reconstructed by:

$$in_t = kf_{t_1} * u(t) + kf_{t_2} * (1 - u(t)), u(t) = \frac{t_2 - t}{t_2 - t_1} \tag{2.5}$$

Poses represented in rotation angles are converted into Quaternions. For two keyframes q_{t_1} and q_{t_2} whose rotation of each joint is in quaternion representation, the quaternion of a in-between frame t can be represented by performing the Quaternion spherical linear interpolation (SLERP), which is equivalent to performing a spherical interpolation, as shown in Equation 2.6 (Shoemake 1985):

$$SLERP(q_{t1}, q_{t2}, t) = q_{t1}(q_{t1}^{-1} q_{t2})^t \tag{2.6}$$

Besides linear interpolation and quaternion SLERP, in-between frames can be represented as a Bézier curve of degree d having the form (Faraway, Reed, and Wang 2007, Shoemake 1985):

$$C(t) = \sum_{i=0}^{d} P_i B_i^d(t), t \in [0, 1] \tag{2.7}$$

where B_i^d is the Bernstein polynomials of degree represented as $B_i^d = \binom{d}{i} t^i (1-t)^{d-i}$. To represent a segment of in-between frames with a cubic bezier curve (d=3), the start and end points, P_0 and P_3, are set to two adjacent keyframes. The cubic Bézier curve will start and end at P_0 and P_3. Interior control points, P_1 and P_2, determine the shape of the cubic Bézier curve. Various optimization methods can be adapted to find the best interior control points to

reconstruct the in-between motion curve. Khan (2016) defined a cost function as the squared L2 distance between the original motion curve and a quadratic Bézier curve (d=2). The interior control point, P_1 , is updated iteratively by solving the partial differential of the cost function with respect to P_1.

Other than linear interpolation and Bézier curve fitting, in Li et al. (2022), a Transformer model was used to predict parameters to control parametric curve equations to reconstruct in-betweens. Li et al. (2022) focused on generating dance motion in keyframe and in-between representation, where in-between frames were generated by Kochanek-Bartels splines (Kochanek and Bartels 1984), and Transformer models were used to predict parameters to control Kochanek-Bartels splines.

2.5 Dance Movement Analysis

Since this thesis specifically focuses on converting dance motion into keyframes and in-betweens, which is related to analyzing dance movement, this section discusses approaches for dance movement analysis and provides insight into techniques to analyze a dance movement. In dance movement analysis, a dance movement is represented as a sequence of poses, and dance movement analysis tries to find repeating patterns in each sequence or find the relation between sequences. Dance movement analysis has wide applications, including assisting beginners in learning new dances (Yang et al. 2013, Aristidou et al. 2015) and analyzing the cross-cultural differences in dance motion (Tommi and Thompson 2011).

Tang et al. (2008) proposed using a self-similarity matrix to find repeating patterns in a

motion, and the similarity between the two poses was measured by the differences between 3-d coordinates of each joint. Poses with high similarities are shown as lines on the self-similarity. Using Dynamic Time Warping (DTW) algorithm, a segment of poses whose diagonal line ranged from top-right to bottom-left was extracted as a repeating pattern. Extracted patterns were clustered such that patterns within a cluster share a similar starting or ending time. Note that an extracted pattern could consist of multiple repeating sub-patterns. Therefore, repeating sub-patterns of each clustered diagonal pattern were further extracted by computing the correlation coefficient with respect to the extracted pattern and the lagged pattern. By changing the lagged time and calculating the correlation coefficient, a function of lagged time was computed, and a peak value at a lagged time l implied that the poses of the extracted pattern were close to the extracted pattern lagged by time l. Therefore, periodicity was detected using the computed function, and sub-patterns were extracted using the detected periodicity. Yang et al. (2013) presented a system to teach beginners learning dance. First, dance motion is analyzed and decomposed into repeating and non-repeating basic patterns using the algorithm proposed by Tang et al. (2008). Then, decomposed patterns were converted into a directed acyclic graph (DAG), which contained the pre-requisite relation between dance patterns and allowed beginners to learn a dance starting from the most basic dance pattern. Brick and Boker (2011) suggested using correlation to examine the similarity inside a dance motion. Following the equation proposed in Cohen, West, and Aiken (2014), the correlation was calculated from a sequence of dance poses and a lagged copy of itself. For each lagged time, a correlation score was calculated, and a function of lagged time was computed by changing the lagged

time. A high correlation value at lagged time l implied that the dance motion was similar to itself lagged by time l, and the periodicity in the dance motion could be found by extracting the lagged value with the highest correlation values. For a long dance sequence consisting of many dance poses, a window function was used to crop a sub-sequence of dance poses. After calculating the correlation function using the sub-sequence, the window function was moved to crop the next sub-sequence. The step was repeated until the window function reached the end of the dance sequence. By doing so, a function of lagged time and time was computed, and the changes in periodicity regarding time could be observed from the computed function. Tommi and Thompson (2011) used the similar correlation function, but instead of calculating the correlation of a dance sequence with respect to itself, it calculated the correlation between two dances performed by different dancers, and it represented to what degree two dancers dance similarly.

Instead of using the 3-d positions of each joint as features and evaluating the motion similarity using correlation functions, Aristidou et al. (2015) designed a set of features that reflects four components in Laban Movement Analysis (LMA) (Groff 1995), a method to describe and analyze a dance in four components proposed by dance theorist Rudolf von Laban. Various features were designed to reflect each component. For example, the Effort component describes the dynamic of a dance movement, and features describing it were extracted from a dance motion by calculating the velocity and acceleration of the root joint. Given a pair of dances, features to reflect four components in LMA were extracted individually from each dance. Then, for features describing each component, Pearson's linear

correlation coefficient (Benesty et al. 2009) was computed to evaluate their similarity, and the overall similarity was computed from the weighted sum of the correlation coefficient of four components.

2.6 Dance Music Analysis

Dance motion and dance music are often closely related to each other. Diverse research topics have been working on investigating the relation between dance motion and dance music. Fitch (2016) suggested that the groove of a dance music piece affects its capacity to fit in a certain type of dance, and the groove is affected by musical features such as upbeats and syncopations. Toiviainen, Luck, and Thompson (2010) studied how movements aligned with the accompanied music's metrical levels. Participants were recruited and instructed to move to the motion freely, and a motion capture system was used to capture participants' motion. Captured motion clips were analyzed and evaluated in terms of synchronization with the music. The result showed that different body parts tend to be aligned with certain metrical levels. For example, the arm movement was aligned with the tactus level, and the upper torso movement was aligned with the period of two or four beats. Krumhansl and Schenck (1997) conducted a subjective test to examine the structural and expressive mapping between dance and music. Participants were asked to watch dance videos and listen to dance music and label the level of section ends, new ideas, tension, and emotion under three conditions: music only, dance only, and both music and dance. The result suggested that music and dance shared a similar temporal organization, and the position labels for music and dance videos were aligned temporally.

Besides, the relation between human motion and the tempo and rhythm of the accompanying music are studies in Van Dyck et al. (2012), Solberg and Jensenius (2017), Leman et al. (2013), and Van Dyck et al. (2015). The results suggest that tempo and rhythm are related to motion. As a result, music source separation strategies to extract rhythmic components from dance music are discussed in Section 2.6.1. Music information retrieval techniques to extract rhythmic features and timbre from dance music are discussed in Section 2.6.2. Lastly, research topics focusing on analyzing dance music are introduced in Section 2.6.3 to provide insight into strategies to analyze dance music.

2.6.1 Music Source Separation

Diverse research topics focus on music source separation, a task to decompose a music piece into components. Each stem consists of instruments, and instruments of different stems are recorded individually and mixed together to form a song. For example, a music piece can be separated into four stems: *drums stem*, *vocals stem*, *bass stem*, and the stem for any other accompanying instruments (*others stem*). Non-negative matrix factorization (NMF) is an unsupervised blind source separation technique to decompose a music signal into components and their activation curves, where components correspond to stems, and activation curves indicate whether stems contribute to the audio sample of the music signal. In NMF, a music signal V represented in a 2-d time-frequency representation is represented by the multiplication of two non-negative matrices: the template matrix W and the activation matrix H. In other words, $V = WH$ (Müller 2015, Smaragdis et al. 2014), where separated

components are represented as column vectors of W, and a column vector contains the spectral features of a component. Their activation curves, regarding whether components contribute to the music signal in each timestep, are represented as the row vectors of H. By specifying a distance function to measure the quality of the approximated music signal V, W and H can be solved by numeric optimization techniques (Müller 2015). With the solved template matrix W and activation matrix H, a stem can be reconstructed from the column vector of W, representing the spectral features of the stem, and its corresponding activation curve in H.

In NMF, a component is represented as a column vector describing its spectral feature and does not contain temporal information. Non-negative Matrix Factor Deconvolution ($NMFD$) (Smaragdis 2004) presented an extension of NMF where components include temporal information. As a result, a component, or a stem, is not represented as a 1-d column vector but a 2-d matrix possessing its temporal and spectral features, and the activation matrix contains activation curves for each component. This fixes the shortcoming of NMF, where the temporal information of a component is discarded (Smaragdis 2004).

Recently, deep neural network models have been used to perform music source separation in a supervised manner, where the music and sound sources are provided as training data. Uhlich, Giron, and Mitsufuji (2015) used fully-connected layers to separate a magnitude spectrogram into multiple magnitude spectrograms. Each of them corresponded to a stem, and Short-time Fourier transform (STFT) was performed to recover the audio for each stem. Takahashi and Mitsufuji (2017) split the music source into multi-bands before sending it into the proposed

neural network to perform source separation. Neural network-based source separation methods typically separate a magnitude spectrogram into a set of spectrograms, each corresponding to a separated source. However, the phase of the separated spectrogram is incorrectly reconstructed, and the quality of the recovered audio is degraded (Kong et al. 2021). Kong et al. 2021 proposed an Unet structure to separate both the magnitude and phase spectrograms from the input spectrogram. *OpenUnmix* (Stöter et al. 2019) was based on the bi-directional RNN model for music source separation proposed by Uhlich et al. (2017). It provides an open-source implementation of deep-learning-based music source separation to decompose a music piece into four stems: *drums*, *vocals*, *bass*, and *others*, where the *others* stem contains any instruments excluded from the first three stems.

2.6.2 Extracting Musical Features from Dance Music

Other than music source separation to acquire stems, onset detection is used to detect the start of musical events that corresponds to a sudden change in the energy of a music signal. In onset detection algorithms, an onset novelty curve, which is a function of time representing the sudden increase of energy in a music signal, is computed. A peak detection algorithm is followed to select peak positions in the onset novelty curve as candidate onset positions. There are multiple ways to represent an onset novelty curve, and the result of the onset novelty curve affects the following peak detection algorithm. For example, an energy-based novelty curve uses a window function centered at a timestep to crop a segment of musical signal and compute the energy of the cropped musical signal. The window function is shifted until its centered

timestep reaches the end of the musical signal. Then, the deviation of the novelty function is calculated to acquire the energy difference in each timestep, and negative differences are removed. Finally, with the computed function, an arbitrary peak detection algorithm can be used to detect peak positions as onset positions (Müller 2015). Instead of computing the energy from the time domain signal, a spectral-based novelty curve first converts a music signal into a time-frequency representation. The temporal deviation of the time-frequency representation is computed. The value for a timestep of the spectral-based novelty curve is the summation of all frequency bins of the deviation at the timestep (Müller 2015). Besides musical features, Mel-Frequency Cepstral Coefficients (MFCC) are used to represent the timbre of a given music piece, and it has been used as features in music genre classification task (Baniya, Ghimire, and Lee 2015, Vishnupriya and Meenakshi 2018).

2.6.3 Dance Music Analysis

Research topics have been working on leveraging musical features to analyze dance music. Panteli, Bogaards, and Honingh (2014) analyzed the similarity of Electronic Dance Music (EDM) by extracting the rhythmic stream consisting of instruments dedicated to the rhythmic pattern of an EDM track. Downbeats were detected to split a rhythmic stream into measures. Features were extracted from the rhythmic stream to model the characteristics of the attack phase, the rhythmic pattern in each measure, and the periodicity of the rhythmic pattern. Subjective tests were conducted where subjects were asked to rate the similarity of EDM music pieces, and the results were compared with similarity evaluated with extracted

patterns. Dixon, Pampalk, and Widmer (2003) classified Latin ballroom dance music using tempo, time signature, and the distribution of periodicities. Kell and Tzanetakis (2013) investigate the transition of tracks in disk jockey (DJ) mixes in terms of musical features represented by timbre, key, loudness, and tempo. The features included Spectral Centroid, Roll-Off, Flux, and MFCC. The transition was quantized by measuring the distance between features of two ordering tracks. Besides analyzing dance music, there are research topics working on proposing dance music datasets. Knees et al. (2015) proposed a dataset consisting of EDM for key detection and tempo estimation. They provided benchmarks by using the dataset to evaluate key detection and tempo estimation accuracy using commercial products and academic research. Beauguitte, Duggan, and Kelleher (2016) proposed an annotated Irish dance music dataset, where the onset time, pitch, and duration of note events of recorded Irish dance music were labeled.

2.7 Cross-modality Analysis and Generation

Multi-modal perception is essential to capture the richness of real-world sensory data and environment. There is a long history of cross-modality learning in computer vision. Barnard et al. (2003) modeled the joint distribution between image and text, and it was able to predict labels for regions in an input image. Frome et al. (2013) proposed a model to jointly map an image and its description text into a shared embedding space, and by leveraging the shared representation, it was able to reach a performance comparable to the state-of-the-art performance in image classification tasks. Davis and Agrawala (2018) extracted musical beats

and visual beats from the music and video, where musical beats were extracted using the onset strength, and visual beats were extracted by analyzing the optical flow of the video. After acquiring musical beats and motion beats, the video was stretched to align visual beats with musical beats, and the result was a video with its visual beats synchronized with the accompanying music. Recently, deep neural networks have been used to learn cross-modality relations of the input data. Zhao et al. (2018) proposed a neural network to take videos and the spectrogram of its sound as inputs, and for each pixel, the model predicts a mask to separate the sound produced by the object located around the pixel's coordinate. Gao and Grauman (2019) designed a dataset containing aligned video and audio recorded in stereo and binaural. A neural network consisting of ResNet and Unet was used to convert stereo audio into binaural audio. The input video was sent to the ResNet to extract visual features regarding the sound source's position. The spectrogram of the stereo audio was passed through an Unet, where visual features were sent to the middle layer. The outputs were complex masks, and they were applied to the input spectrogram to get the left and right channels of the predicted binaural audio. Zhou et al. (2020) proposed using recursive neural networks to generate talking videos from a facial image and audio. The content of the audio was identified, and positions of pre-defined facial landmarks were predicted to match with the identified content. Image warping algorithms or deep-learning-based image-to-image translation were used to synthesize frames from the predicted facial landmarks and facial image.

With the possibility of learning cross-modality relation between dance and music, deep

neural networks are used to analyze and generate dance from the dance music. Lee et al. (2019) proposed decomposing a dance into basic dance units and leveraging deep learning models to generate basic dance units from music. A basic dance unit was defined as a sequence of continuous dance poses whose starting and ending poses were movements that drastically slowed down. The training process contains two stages. In the first stage, a model was trained to disentangle the dance units into its initial pose feature and movement feature, and reconstruct the dance unit back using its two features. A loss function was designed to encourage the model to discard information related to movement in the initial pose feature and the pose information in the movement feature. In the second stage, a model was trained to generate movement features from the input musical features. The musical features contained MFCC features, the first temporal derivative of MFCC features, and the log-mean energy. In the testing stage, the generated movement features and initial pose features were used as the inputs to the trained model in the first training stage to generate dance movements. Ye et al. (2020) proposed predicting dance units from musical features containing chromagram, beat, and onset. Predicted dance units were aligned with musical beats, which resulted in unnatural transitions between adjacent dance units. As a result, a model was trained to generate smooth transitions between dance units. Chen et al. (2021) employed CNN and RNN to map musical features and dance motions to the shared dance rhythm signature and their genre, which were annotated by professional dancers. In the test time, rhythm signatures and genres were predicted from the input dance music, and a graph-based motion reconstruction strategy was used to generate the accompanying dance motions by

retrieving and smooth dance motion from the annotated dance and minimizing the designed cost function. The cost function included constraints such as matching the genre of the music and dance, minimizing the distance between their rhythm signatures, and ensuring smooth transitions between motion segments. Li et al. (2021a) proposed using three Transformer models to synthesize dance motion from audio features, including MFCC, chromagram, peak positions, and onset strength. Three Transformer models consisted of an audio Transformer model, a motion Transformer model, and a cross-modal Transformer model. The audio and motion transformer models were used to extract musical and motion features. These features were concatenated and sent to the Transformer model to synthesize dance motion. Different from the related research, Li et al. (2022) reformulated the dance problem into two stages, generating keyposes and synthesizing parameters to control parametric curve equations to derive in-between poses. Keyposes were defined as poses located on the detected beat positions. A transformer model was used to synthesize key poses, and musical features located around a beat position were used as the input to the Transformer model. Musical features were defined as MFCC features and chromagram. A similar Transformer model was used for generating parameters to control a parametric curve equation, which was used to derive in-between poses between two generated keyposes.

2.8 Evaluation Metrics for Motion

When machine learning is used for generation, such as realistic images, music, or motion, it is critical to select good metrics to evaluate the results of generative tasks (Borji 2019). Therefore,

this section introduces metrics to evaluate generated dance motion. Section 2.8.1 presents the Frechet Inception Distance that assesses the distances of features of two sets of dances. Section 2.8.2 introduces metrics to evaluate the diversity of the generated dances. Section 2.8.3 presents metrics to estimate the alignment between musical beats and motion beats. Finally, Section 2.8.4 presents the mean square error to assess the reconstruction distance of a generated dance.

2.8.1 Frechet Inception Distance

Frechet Inception Distance (FID) is commonly used to evaluate generated results (Choi et al. 2020, Karras et al. 2020, Ding et al. 2021), and it was originally proposed by Heusel et al. (2017), where it was used to evaluate generated images containing faces, numbers, indoor scenes, and outdoor scenes. Considering the objective of a generative task, which is usually defined as producing generated data that match with the observed data (Heusel et al. 2017), Heusel et al. (2017) proposed using the distance between features of collections of generated data and observed data as the measurement to evaluate a generative task. The shorter the distance is, the better the generated results are. Using generated images and real images as an example, to calculate FID of collections of generated images (fake images) and real images, a feature extractor, which can be an image classification network, is used to encode all fake images and real images into their features. Two collections of features, one for fake images and the other for real images, are viewed as two continuous multivariate Gaussian distributions (Lucic et al. 2018). The distance between two multivariate Gaussian distributions is measured by Frechet Distance (Fréchet 1957). In other words: $FID(f, g) = \|\mu_f - \mu_g\|_2^2 + Tr(\sum_f + \sum_g - 2(\sum_f \sum_g)^{\frac{1}{2}})$,

where $(\mu_f, \sum_f)$ and $(\mu_g, \sum_g)$ are the mean and variance of collections of real and fake images' feature. A smaller FID result implies that the distance between two multivariate Gaussian distributions is short, and the distribution of fake images is close to real images. Other than evaluating generated images, FID has been adapted to evaluate generated videos (Unterthiner et al. 2019) and audio (Kilgour et al. 2019) using feature extractors designed for video and audio.

In the literature on dance motion generation, FID is used to measure how close, in terms of a dance feature space, the collection of generated dance is to a collection of real dance. Lee et al. (2019) used FID to measure the generated dance. As there is no standard feature extractor to extract the features of a dance sequence, they proposed a neural network trained on the dance motion as an action classifier and used it as the feature extractor. When evaluating the FID of a collection of generated dance motion, sequences of generated dance and real dance were sent to the action classifier to extract features, and for each sequence, the outputs of an intermediate layer of the action classifier were used as its feature. After calculating features, a collection of real dance features and fake dance features were acquired. Mean and variance were calculated separately, and they were used to calculate the FID to evaluate the distance between features of the generated dance and real dance.

The workflow of evaluating FID is similar in research regarding dance motion generation, yet the difference is in the way the dance feature extractor is proposed. Ye et al. (2020) and Chen et al. (2021) trained a motion auto-encoder to map dance motion into features and map features back to the dance motion, and mapped features were used as dance motion features.

Instead of training a motion classifier and using the outputs of its intermediate layer as extracted features, Li et al. (2021a) used feature extractors to extract geometric and kinematic features of poses. The geometric features were proposed by Müller, Röder, and Clausen (2005) to describe the relation of body parts, and the kinematic features were proposed by Onuma, Faloutsos, and Hodgins (2008) to describe the kinetic energy of each joint. Both features were originally used to measure the similarity of motion clips. Müller, Röder, and Clausen (2005) defined a set of features to express the geometric relations between body parts of poses represented as skeletons, and designed features were used to evaluate the similarity between motion clips to extract similar clips from a dataset. A feature was described as a Boolean function to express a geometric relation of body parts. For example, a Boolean function expressed whether the right toes lie in front of a plane spanned by the left ankle, left hip, and the root, and whether the right-hand raises above a plane whose normal vector is a vector defined by the "chest" and "neck" joint. It also expresses the distance between a joint and body part exceeding a threshold or not, such as expressing if the right hand touches the right legs, and whether the joint angle between body parts exceeds a threshold, which implies that the body part, such as the arm or thigh, is stretching or bending. The illustration of Boolean functions is shown in Figure 2.2. Müller, Röder, and Clausen (2005) designed 31-d geometric features for evaluating the similarity between video clips. Features expressed as Boolean functions were invariant to global transforms and scaling, and this made geometric features robust to variations in video clips and benefits extracting similar features. Onuma, Faloutsos, and Hodgins (2008) defined kinematic features and relied on defined features to classify motions, search for similar motions,

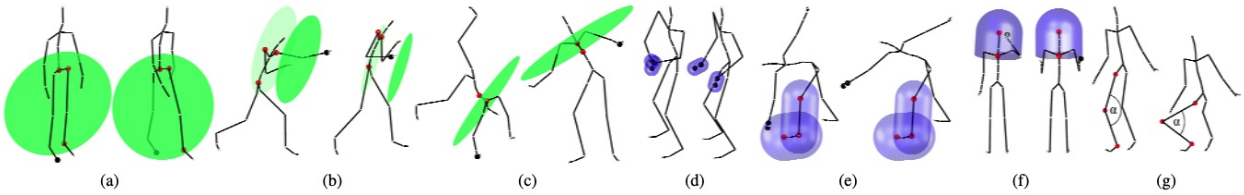

Figure 2.2: An illustration of geometric features (Müller, Röder, and Clausen 2005). (a–c) A Boolean function expressing a joint in front of or behind a plane. (d–f) A Boolean function expressing if the distance between a joint and body part exceeds a threshold. (e) A Boolean function expressing whether the distance between two body parts exceeds a threshold.

and detect outlier motions. The kinematic feature of a motion clip was calculated as follows: first, the velocity of each joint at each frame was calculated using the first derivative of the joint position. Then, the kinetic energy of each joint at each frame was calculated from the square of each velocity. Finally, for each joint, the mean kinetic energy was computed by finding the mean of the kinetic energy of all frames. Li et al. (2021a) used these features, geometric features and kinematic features, as the features used by FID to evaluate the distance between collections of generated dance and real dance. Li et al. (2022) followed Li et al. (2021a) and used the same feature extractors to evaluate FID metrics.

2.8.2 Diversity and Multimodality

Besides FID, Lee et al. (2019) evaluates the diversity and multimodality of generated dances. The diversity was evaluated among dance motions generated using different dance music pieces, and it assessed the trained model's ability to generate diverse dances from different music pieces. Diversity was calculated by randomly selecting combinations from a set of generated dance motions, computing the distance between two generated dance motions in each combination,

and finding the average distance of all combinations. Multimodality was evaluated using dance motions generated from the same music piece, and it evaluated the ability of the model to generate diverse dance conditioned on the same music piece. When evaluating multimodality, combinations of five dance motions were picked from a set of generated dance motions using the same music piece. The average distance of generated dance within each combination was calculated, and the average distance of all combinations was calculated and used to represent the multimodality of generated dance motions. Lee et al. (2019), Chen et al. (2021), Li et al. (2022), and Li et al. (2021a) used the same metrics to evaluate the diversity of generated dance motions.

2.8.3 Music-Beat Alignment

Other than only evaluating the dance motion, metrics are proposed and used to evaluate the relation between the generated dance motion and its dance music. Lee et al. (2019) proposed evaluating how well the kinematic beats match with musical beats. *Kinematic beats* B_k were defined as the number of dance poses whose velocity, calculated from the first derivative of joint positions, drastically slowed down. *Musical beats* were detected using a beat detection algorithm, and B_m was defined as the number of detected musical beats. B_a was defined as the number of aligned kinematic beats and musical beats. With B_k, B_m, and B_a, the *beat coverage* was defined as the ratio of the number of kinematic beats to musical beats. In other words, $\frac{B_k}{B_m}$. *Beat hit rate*, $\frac{B_a}{B_k}$, was defined as the ratio of the number of aligned beats to kinematic beats. Similarly, Chen et al. (2021) proposed *Beat accuracy* as the number of aligned

beats to the musical beats, where aligned beats referred to motion beats that were adjacent to musical beats. Besides, they followed the motion beats defined in Shiratori and Ikeuchi (2008), where motion beats were defined as dance motion with stopping postures. Li et al. (2021a) proposed *Beat Alignment Score* to evaluate how well generated dance correlates to the dance music. Kinematic beats referred to beats whose velocities were local minimum values of velocity curves. The beat alignment score was calculated as the average distance of kinematic beats to their nearest musical beats. Similarly, Li et al. (2022) proposed *Beat Consistency Score* calculated from the average distance of each musical beat to the nearest kinematic beat, where the definition of kinematic beats was the same as Lee et al. (2019).

2.8.4 Mean Squared Error

A great amount of research related to keyframe selection and in-between reconstruction strategies is proposed to perform motion compression, and the evaluation metric for motion compression assesses the reconstruction distance of keyframe selection and in-between reconstruction strategies. As a result, the evaluation metric for motion compression is discussed in this section, and it will be used in this thesis to evaluate the reconstruction distance of the generated dance and assess how similar the generated dance is to its original dance. Mean squared error is adapted to evaluate the difference between reconstructed motion and the original motion (Khan 2016, Zhang et al. 2013, Halit and Capin 2011, Bulut

	Music	Video	3-d joint positions	Depth	Views	Duration
MADS	×	✓	✓	✓	3 (video) 1 (depth)	58 mins
DanceDB	×	✓	✓	×	1	226 mins
ChoreoMaster	✓	×	✓	×	1	19.1 hours
AIST	✓	✓	×	×	9	118.1 hours
AIST++	✓	✓	✓	×	9	311.6 mins

Table 2.1: Dance datasets comparison.

and Capin 2007, Togawa and Okuda 2005), which is defined as:

$$MSE = \frac{1}{NJ} \sum_{n=1}^{N} \sum_{j=1}^{J} \left\| x_n^j - \hat{x}_n^j \right\|^2 \tag{2.8}$$

where N is the number of frames, J is the number of joints, x is the original motion, and $\hat{x}$ is the reconstructed motion.

2.9 Dance Datasets

Datasets that provide ground-truth data are essential for training and evaluating supervised machine learning methods, including deep neural networks. With the increasing interest in dance information retrieval, datasets corresponding to dance music and video have been proposed recently. This section describes various dancing datasets, and the comparison of dance datasets is listed in Table 2.1.

2.9.1 The MADS Dataset

Zhang et al. (2017) proposed the Martial Arts, Dancing, and Sports (MADS) dataset, which consists of martial art actions (Tai-chi and Karate), dances (Hip-hop and Jazz), and sports (basketball, volleyball, football, rugby, tennis, and badminton). It features multi-view RGB videos captured with three cameras, single-view videos filmed with a stereo camera to capture the distance of objects to the camera, and ground-truth poses obtained with a motion capture system. Motion capture data were downsampled to be in sync with videos. The multi-view videos and single-view videos were captured separately, and subjects were asked to modify their actions to face the camera as much as possible in the single-view depth videos. The total length of multi-view videos is around 53,000 frames, approximately 1 hour under 15 frames per second (FPS).

2.9.2 The Dance Motion Capture Dataset

The dance motion capture dataset (DanceDB dataset) (Aristidou, Stavrakis, and Chrysanthou 2014) is designed to preserve the Cypriot folk dance. The dataset preserves 3-d motion capture data, video data, photographs, drawings, text descriptions about dance types, and metadata about dancers, recording data, location, and recording system. While the dataset aims to preserve Greek and Cypriot dances, most of the dances in the dataset are contemporary dances. The total length is 226 minutes.

	Style 1	Duration	Style 2	Duration
Music	Chinese	20.7 hours	Mild	21.4 hours
	Japanese	40.8 hours	Exciting	32.9 hours
	English	18.7 hours	Neutral	48.2 hours
	Korean	22.3 hours	-	-
Dance	Anime	7.5 hours	Sexy	1.6 hours
	Hip-Hop	5.4 hours	Lovely	5.9 hours
	K-pop	3.9 hours	Cool	5.4 hours
	Tradition	3.1 hours	Gentle	2.7 hours
	-	-	Other	4.3 hours

Table 2.2: The style label for dance and music in the ChoreoMaster dataset.

2.9.3 The ChoreoMaster Dataset

The ChoreoMaster dataset (Chen et al. 2021) consists of dances from motion capture systems and dances animated by animation software. In total, there are 19.1 hours of dance motion; among them, 9.91 hours of dance are paired with dance music. It also collects 102.5 hours of dance music. Dance motion and music are manually labeled with two styles, as shown in Table 2.2. All dance motions are structured in four-beat meters, and professional dancers were asked to label each dance with an 8-bit rhythm signature, which is used to compare with the beat pattern of the dance music to assess the similarity between dance motion and music.

2.9.4 The AIST and AIST++ Dance Video Database

The AIST dance dataset (Tsuchida et al. 2019) consists of multi-genre dance videos filmed with a multi-camera system and dance music. In each genre, professional dancers choreographed basic and complex dance patterns and participated in the filming of dance videos. The AIST++ dance dataset (Li et al. 2021a) used the same dance videos and music in the AIST dataset, and

it reconstructed camera parameters for the multi-camera system and 2-d and 3-d dance motion from the original AIST dance videos. For details about the AIST and AIST++ datasets, please refer to Chapter 3.2.

Chapter 3

Methods

This chapter discusses the experimental setting and implementation in detail. Section 3.1 describes the experimental workflow, including preprocessing, keyframe selection, and in-between reconstruction. Section 3.2 lists the structure of the AIST and AIST++ dataset and the preprocessing steps prior to experiments. Section 3.3 and Section 3.4 describes strategies to select keyframes and reconstruct in-betweens, respectively. Finally, Section 3.5 lists the evaluation metrics to assess a reconstructed dance motion.

3.1 Overview of Workflow

The workflow for the experiment (Figure 3.1) contains four blocks: preprocessing block, keyframe block, in-between block, and evaluation block. First, the preprocessing block extracts dance motion features and musical features from the dance motion and music of the AIST++ dataset. Different strategies are used to extract features, and features are sent to

the keyframe and in-between block for keyframe selection and in-between reconstruction. Second, the keyframe block contains strategies for taking the preprocessed motion and musical features as inputs and selecting keyframes from the dance motion feature. The outputs of the keyframe block, selected keyframes, are sent to the in-between block. Third, the in-between block uses strategies that take the selected keyframes and musical features as their inputs to reconstruct the dance motion. Last, different strategies to select keyframes and reconstruct in-betweens are evaluated with metrics to assess their distance. Since keyframes and in-betweens are both required to reconstruct a dance, strategies for selecting keyframes cannot be evaluated without using a strategy for deriving in-between frames and vice versa. As a result, when assessing strategies for selecting keyframes, they are combined with two in-between reconstruction processes: linear interpolation and Bézier curve fitting, and reconstructed dances are compared among strategies using the same in-between reconstruction process. Keyframes are linearly selected when assessing strategies for deriving in-between frames.

3.2 Datasets: AIST and AIST++

The foundation of the experiment depends on the AIST++ dataset (Li et al. 2021a), which is based on the AIST dataset (Tsuchida et al. 2019). The AIST dataset includes multi-genre dance music and videos filmed with a multi-view camera system. It consists of seven categories of dance videos: basic dance, advanced dance, group dance, moving camera, showcase, cypher, and battle. Each category is explained in detail in the following section. The AIST++ dataset

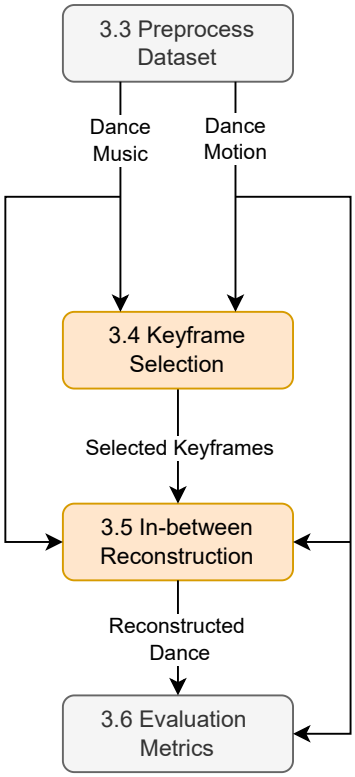

Figure 3.1: Overview of the experimental workflow.

contains the dance music and videos for the basic dance and advanced dance categories in the AIST dataset, and it further extends the dataset by recovering the camera parameters of each camera in the multi-view camera system and reconstructing dance motions‘ 2-d and 3-d positions and rotations. In this thesis, only the reconstructed 3-d positions and rotations are used for experiments. Experiments are only conducted on multi-genre dance motion in the basic and advanced dance categories.

3.2.1 Dataset Structure

Dance Genre

The AIST dataset contains ten dance genres: Break, Pop, Lock, Middle Hip-Hop, LA style Hip-Hop, House, Waack, Krump, Street Jazz, and Ballet Jazz. After consulting professional dancers, genres were selected to cover dance styles starting from the 1970s. For each genre, three professional dancers having dancing experience of more than five years were asked to select six dance music pieces and make dance choreographies (Tsuchida et al. 2019).

Dance Music

Each of the ten genres contains six dance music pieces, and, thus, there are 60 music pieces in the AIST dataset. The tempi for six different music pieces are set to 80, 90, 100, 110, 120, and 130 beats per minute (BPM), except for the six pieces for House, which are set to 110, 115, 120, 125, 130, and 135 BPM. Professional musicians were asked to create six different music pieces for each genre, and dancers were asked to choreograph with created music pieces. All dance music pieces are in four-four time signatures (Tsuchida et al. 2019).

Dance Video

Dance videos were filmed with a multi-camera system in 60 frames per second, and each dance is associated with multiple videos filmed from different angles. Videos in each dance genre contain four categories: basic dance, advanced dance, group dance, and moving camera. In addition, there are situation videos cover three scenarios: showcase, where a group of dancers performs

on stage, cypher, where a group of dancers lines up in a circle, and each dancer performs at the center of the circle, and battle, where two dancers face to each other and dance (Tsuchida et al. 2019). Details of dance videos are listed in Table 3.1. Note that each camera produces one video file, and for each dance genre, three professional dancers participated in recording dance videos.

Basic Dance The basic dance category contains ten basic dance patterns selected by participating dancers. The length of each video is four measures, and most of the dance patterns are repeated in each video. Each of the three dancers performs ten selected dance patterns with four music pieces selected from six music pieces of the dance genre, and basic dance videos were filmed with nine cameras. Dancers were not restricted to dancing to the same four music pieces. In total, there are 1,080 (three dancers $\times$ ten dance patterns $\times$ four dance music pieces $\times$ nine cameras) basic dance videos per genre. Note that the basic dance category is the only category where dancers are restricted to dancing to the same dance patterns. For the rest of the categories, dancers are allowed to choreograph their own dances, and each of the three dancers dances differently.

Advanced Dance In the advanced dance category, each dancer performs seven choreographies filmed by nine cameras. The length of each choreography is 16 measures. In total, there are 189 (three dancers $\times$ seven choreographies $\times$ nine cameras) dance videos inside advanced dance videos, and the choreography of an advanced dance video is more complicated than the basic dance video.

Group Dance The group dance category contains a group of three dancers performing ten choreographies, each of 16 measures in length. The dance group is filmed by nine cameras, and there are 90 group dance videos in total.

Moving Camera In the moving camera category, each dancer performs three or four dances, and each dance was filmed with two cameras with fixed positions and one moving camera. The length of each dance is 16 measures, and there are 30 dance videos in the moving camera category.

Showcase Showcase refers to the case where dancers perform on stage. In videos for the showcase, a group of eight dancers performs three choreographies. The duration of each choreography is 24 measures, and each choreography was filmed by eight cameras.

Cypher In cypher videos, ten dancers form a circle while one dancer improvises dance in the center. Dancers take turns improvising in the center, and the timing to change the dancer in the center is left up to the dancers themselves. The group performs two sets of dances filmed by five cameras, and the duration of a set is around ten minutes.

Battle In battle videos, two dancers take turns improvising with music. There are three groups in battle videos, and each group performs a 4-minuets dance filmed by five cameras.

	#Videos	#Cameras	Length
Basic Dance	1080	9	4 measures
Advanced Dance	189	9	16 measures
Group Video	90	9	16 measures
Moving Camera	30	3	16 measures
Showcase	24	8	24 measures
Cypher	10	5	≈10 mins
Battle	15	5	≈4 mins

Table 3.1: Statistics of dance videos for one dance genre (top) and situation videos (bottom) in the AIST dataset.

Dance Motion

The AIST dataset stores dance videos represented frame by frame, and it lacks the 2-d and 3-d information of dance motions. Li et al. (2021a) refined the dataset by reconstructing the 3-d joint coordinates of each dance motion. Compared to the AIST dataset, the AIST++ dataset contains: estimated camera parameters (see Chapter 2.2.5), 17-joints 2-d coordinates (Ruggero Ronchi and Perona 2017), 17-joints 3-d coordinates (Ruggero Ronchi and Perona 2017), and 24-joints 3-d motion in *Skinned Multi-Person Linear* (SMPL) format (Loper et al. 2015). The steps to reconstruct additional dance-related data are listed below: First, a joint detection algorithm (Papandreou et al. 2017) was employed to detect 2-d joint coordinates from multi-view videos frame by frame. Next, camera parameters were initialized, and bundle adjustment (Triggs et al. 1999) was applied to optimize camera parameters. With optimized camera parameters and detected multi-angle 2-d joint coordinates, 3-d joint coordinates were reconstructed by leveraging camera parameters to triangulate multi-angle 2-d joint coordinates to 3-d coordinates (see Chapter 2.2.5). Finally, reconstructed 3-d joint positions were fit to an SMPL model to represent 3-d joint positions as the translation of root and the rotation of each

joint in axis-angle representation (see Chapter 2.2.1, 2.2.2, and 2.2.3).

3.2.2 Dataset Format

Dance music pieces are stored in WAV files and MP3 audio files. Dance videos were filmed with 60 frames-per-second (FPS) and stored in MP4 video files. The 2-d and 3-d joint coordinates are stored in pickle files, which are serialized *Python* objects. The 3-d motion is stored in pickle file, and it is represented as the translation of the root joint, the rotation of each joint regarding its parent in the axis-angle representation, and the pose blend parameters that control the surface of the SMPL model, which is related to the body shape of a dancer.

3.3 Preprocessing

The dance motion and the music in the AIST++ dataset are preprocessed to exract features describing the dance motion and elements in music, such as timbre, rhythm, and measure, before sending them into the keyframe block and in-between block. The dance music and motions in the Ballet Jazz genre are removed first, and the keyframe selection and in-between reconstruction experiments use only the remaining nine genres. This is because the dance music pieces in Ballet Jazz do not have drums, resulting in the *drums stem* from the music source separation algorithm, one of the preprocessing steps, being silent. Since some in-between reconstruction strategies in this thesis rely on musical features of the *drums stem*, a silent *drums stem* leads to empty musical features, which I defined as invalid inputs to the strategy. As a result, I decided to remove the Ballet Jazz genre. At the beginning of preprocessing steps, input

music and dances are split into segments such that each segment contains dance motions and music within a measure. Since all dance music pieces are in four-four time signatures and at constant tempi, the number of frames per measure is calculated from the frame rate (FPS) of the dance motion, which is always 60 frames per second, and the tempo (BPM) of the dance music, by:

$$Frames\ per\ Measure = \frac{60}{BPM} * 4 * FPS \tag{3.1}$$

The Dance music is split by calculating the number of samples per measure by Equation 3.2, where SR is the sample rate of the music.

$$Samples\ per\ Measure = \frac{60}{BPM} * 4 * SR \tag{3.2}$$

A segment, which represents a measure of music, is sent to the preprocessing workflow, and the preprocessed motion and musical features are sent to the keyframe selection block and in-between-reconstruction block to reconstruct the dance motion. Procedures to preprocess and extract features are discussed in the following section, and the labels of extracted dance motion features and musical features are listed in Table 3.2.

Label	Description
D_{rotmat}	Dance motion represented in rotation matrix
D_{eulers}	Dance motion represented in Euler-angle representation
M_{full}	22-d musical features of the dance music
M_{onset}	1-d onset strength of the dance music
M_{full_drums}	22-d musical features of the *drums stem*
M_{onset_drums}	1-d onset strength of the *drums stem*
M_{onset_bass}	1-d onset strength of the *bass stem*
M_{onset_vocals}	1-d onset strength of the *vocals stem*
M_{onset_others}	1-d onset strength of the *others stem*
M_{nmfd_drums}	Activation curves of the *drums stem* processed by $NMFD$
M_{genres}	Genre vector of the dance music

Table 3.2: Labels for preprocessed motion and musical features. Four stems of a dance music are split by *OpenUnmix* (Stöter et al. 2019). Activation curves are computed by applying Non-negative Matrix Factorization Deconvolution ($NMFD$) (Smaragdis 2004) to further separate the drums track into three components. The musical features for the *bass stem*, *vocals stem*, and *others stem* (M_{full_bass}, M_{full_vocals}, M_{full_others}) are not shown as they are not used in this thesis.

3.3.1 Processing Dance Motions

In the preprocessing for motion features, axis-angle rotation representation is converted into rotation matrix and Euler-angle rotation (see Chapter 2.2.3). In the AIST++ dataset, a dance pose with 24 joints is represented with a 72-d axis-angle rotation representation and a 3-d translation of the root. In preprocessing steps, the rotation of each joint is converted to 9-d rotation matrix and 3-d Euler-angle representation, resulting in a 216-d vector (24 joints $\times$ 9-d rotation matrix) and 72-d vector (24 joints $\times$ 3-d Euler-angle representation). These vectors are further concatenated with the 3-d root translation, and following the preprocessing steps proposed in Li et al. (2021a), the rotation matrix is additionally padded with a 6-d zero vector. After preprocessing steps, a dance sequence of T frames is represented as $D_{rotmat} \in R^{T\times 225}$

using rotation matrix and $D_{eulers} \in R^{T \times 75}$ using Euler-angle representation.

3.3.2 Extracting Features from Dance Music

The musical features used here consist of features related to rhythm and timbre, and they are extracted from dance music pieces and tracks separated using a music source separation algorithm. Labels for musical features for music pieces and tracks are listed in Table 3.2. Details about features regarding each label will be discussed in this section. Dance music is re-sampled to 30,700 samples per second. 20-d Mel-frequency Cepstral Coefficients (MFCC) features relating to timbre and 1-d spectral flux onset strength envelope relating to rhythm are extracted using 512 hop size and the hann window of 2,048 fast Fourier transform window length. The hop size and window length are set to 512 and 2,048 as they are commonly used in research related to musical features (Bahuleyan 2018, Engel et al. 2020, Lu et al. 2017, Cheuk, Agres, and Herremans 2020). The sample rate is selected such that the number of frames per second in music matches with video (60 frames per second = 30,700 samples per second ÷ 512 hop size). A peak detection algorithm is used to detect the onset envelope's peak locations. The result is represented in a binary format, where 0 indicates that the value of the onset envelope at this timestep is not a peak value, and 1 indicates its counterpart. The extracted musical features in each frame is a 22-d (20-d MFCC + 1-d onset strength + 1-d onset peak positions) vector, and the dance music of T frames is represented as $M_{full} \in R^{T \times 22}$, and its onset strength is represented as $M_{onset} \in R^{T \times 1}$.

Prior to extracting 22-d musical features, a piece of dance music is processed by *OpenUnmix*

(Stöter et al. 2019) to split it into four stems: *drums*, *bass*, *vocals*, and *others* (see Chapter 2.6). This is based on the assumption that dance motion is more related to certain components of dance music. Four stems are sent to the same workflow mentioned above to acquire musical features for each stem. In addition, the drums stem is further processed by a non-negative matrix factorization deconvolution ($NMFD$, see Chapter 2.6) to split into three components, and three activations curve $M_{nmfd\_drums} \in R^{T\times 3}$ are kept as musical features. Finally, other than acquiring musical features from the audio, the genre of each dance music is converted into a one-hot dance genre vector. For a dance music in i-th genre, the genre vector is represented as $M_{genre} \in R^{9\times 1}$.

3.4 Keyframe Selection

Given processed dance motion features or musical features, six strategies select keyframes from dance motion frames while keeping the first and last frames selected as keyframes. Details about each strategy are explained in this section, and labels of each strategy are listed in Table 3.3, including selecting keyframes using the Euler-angle representation (KF_{even_rotmat}) and the rotation matrix (KF_{even_euler}) evenly, using the dynamic programming (KF_{dp}), the onset strength envelop of the dance music piece (KF_{onset}) and its four stems split by *OpenUnmix* (KF_{stems}), and using the activation curve of the *drums stem* processed with $NMFD$ (KF_{nmfd}). Among these strategies, KF_{even_rotmat}, KF_{even_euler}, and KF_{dp} only use dance motion features, and the rest depends on musical features.

Label	Description
KF_{even_rotmat}	Evenly select keyframes from dance motion represented in rotation matrix
KF_{even_euler}	Evenly select keyframes from dance motion represented in Euler-angle representation
KF_{onset}	Select keyframes from peak positions of the dance music's onset strength
KF_{stem}	Select keyframes from peak positions of the weighted onset
KF_{nmfd}	Select keyframes from peak positions of the weighted activation curve
KF_{dp}	Select keyframes with dynamic programming using the dance motion

Table 3.3: Labels for strategies to extract keyframes.

3.4.1 Evenly Selecting Keyframes

Given a sequence of dance motion features D_{eulers} in Euler-angle representation or rotation matrix, keyframes are selected to be evenly spaced. For an experiment of selecting N keyframes, the selected keyframes are defined as $KF_{even\_euler} \in R^{N\times 75}$ with Euler-angle representation and $KF_{even\_rotmat} \in R^{N\times 225}$ in rotation matrix.

3.4.2 Selecting Keyframes using Onset Curves

The following strategies select keyframes using different musical features. Given an onset curve M_{onset}, onset peak positions are detected by selecting the local maximum of the onset curve. After the detection algorithm, M peaks are found inside the onset curve, which is represented as $Peak := \{p_i : p_i \in R, i \in \{1...M\}\}$.

For selecting N keyframes, the N most significant values are selected from $Peak$, and motion frames of D_{eulers} whose timesteps are located on N peak positions are selected as keyframes, represented as $KF_{onset} \in R^{N\times 75}$. In the implementation, there exist onset curves having less than N peaks. In such a case, the motion frame located at the middle point between the

two keyframes with the highest onset value is assigned as the new keyframes. This strategy is performed iteratively until the number of selected keyframes matches the desired keyframes. Besides, peaks located within the first and the last G keyframes are removed from $Peak$. This is to avoid the misalignment leading to the peak of the first beat deviating from the first frame. Since the first frame is always a keyframe, this keeps the keyframe selection strategy away from selecting two motion frames close to each other temporally.

Onset curves of four stems, M_{onset_drums}, M_{onset_bass}, M_{onset_vocals}, and M_{onset_others}, are also used as musical features to select keyframes. Four onset curves are weighted and summed up to obtain a new onset curve (M_{onset_stem}) in Equation 3.3. The peak detection strategy is applied to M_{onset_stem} to get keyframes $KF_{stems} \in R^{N\times 75}$. Activation curves of three components, $M_{nmfd\_drums} \in R^{T\times 3}$, are also used to select keyframes. Similar to M_{onset_stems}, a new activation curve is calculated from the weighted sum of all activation curves, and the detection strategy is used to select N keyframes $KF_{nmfd} \in R^{N\times 75}$ from the new activation curve. In summary, these strategies take three types of onset curves as their inputs and select different motion frames as keyframes shown in Figure 3.2.

$$\begin{aligned} M_{onset\_stem} = {} & W_{drums} M_{onset\_drums} \\ & + W_{bass} M_{onset\_bass} \\ & + W_{vocals} M_{onset\_vocals} \\ & + W_{others} M_{onset\_others} \end{aligned} \tag{3.3}$$

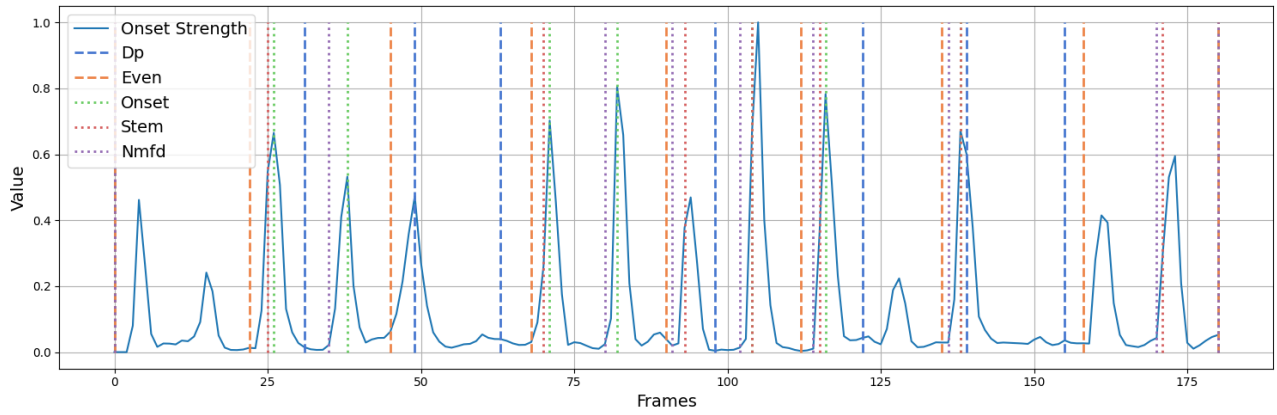

Figure 3.2: Selected keyframes using peak detection algorithm on three curves: the onset curve of a Break dance music, the weighted onset curve of four stems (*drums*, *bass*, *vocals*, *others*) from *OpenUnmix*, and the weighted activation curve of activation curves from applying $NMFD$ to the drums stem.

3.4.3 Dynamic Programming

The last strategy for the keyframe selection uses the dynamic programming method proposed in Roberts (2018) to select keyframes (see Chapter 2.4.1). Keyframes are optimized iteratively using dynamic programming, and the result of selecting N keyframes using this algorithm is defined as $KF_{dp} \in R^{N \times 75}$.

3.5 In-between Reconstruction

Dance motion features, musical features, and selected keyframes are used by different strategies to reconstruct in-between motions. The labels of each strategy are listed in Table 3.4. Strategies include linear interpolation and Quaternion SLERP ($\hat{D}_{eulers\_linear}$, $\hat{D}_{rotmat\_linear}$), Bézier curve fitting ($\hat{D}_{bezier}$), and Transformer models ($\hat{D}_{cross\_full}$, $\hat{D}_{cross\_drum}$,

Label	Description
$\hat{D}_{eulers\_linear}$	Reconstructed dance in Euler-angle representation using linear interpolation and Quaternion SLERP
$\hat{D}_{rotmat\_linear}$	Reconstructed dance in rotation matrix using linear interpolation and Quaternion SLERP
$\hat{D}_{bezier}$	Reconstructed dance using Bézier curve fitting
$\hat{D}_{cross\_full}$	Reconstructed dance using the *Full Model/Full Track*
$\hat{D}_{cross\_drum}$	Reconstructed dance using the *Full Model/Drum Track*
$\hat{D}_{merged\_full}$	Reconstructed dance using the *Full Model/Full Track/Merged*
$\hat{D}_{merged\_drum}$	Reconstructed dance using the *Full Model/Drum Track/Merged*
$\hat{D}_{genre}$	Reconstructed dance using the *Genre*
$\hat{D}_{motion}$	Reconstructed dance using the *Motion*

Table 3.4: Labels for the reconstructed dance of strategies to reconstruct in-betweens.

$\hat{D}_{merged\_full}$, $\hat{D}_{merged\_drum}$, $\hat{D}_{genre}$, $\hat{D}_{motion}$). Details are described in the following sections.

3.5.1 Linear Interpolation and Quaternion SLERP

Given N keyframes selected from an original dance motion of T frames in Euler-angle representation, the 3-d translation of the root position is reconstructed with linear interpolation, and the rotation of each joint is converted to Quaternion. Quaternion SLERP is conducted to derive in-between frames, and derived frames, represented in Quaternion, are converted back to their original representation. The result of this strategy is a reconstructed motion $\hat{D}_{euler\_linear} \in R^{T\times 75}$ in Euler-angle representation and $\hat{D}_{euler\_rotmat} \in R^{T\times 225}$ in rotaiton matrix.

3.5.2 Bézier Curve Fitting

In this strategy, in-between frames are reconstructed with cubic Bézier curves. Each cubic Bézier curve reconstructs in-between frames represented as a joint's x, y, and z rotation angle. Selected N keyframes in Euler-angle representation and the motion D_{euler} are taken as inputs. Two consecutive keyframes at time i and j are used as the first and the last control points $P0$ and $P3$ when reconstructing in-between frames located between them. *scipy*, which invoked Levenberg–Marquardt algorithm (Moré 1978, Levenberg 1944), is used to solve $P1$ and $P2$ such that the cubic Bézier curves recover the original motion D_{euler}. The reconstructed motion is denoted as $\hat{D}_{bezier} \in R^{T\times 75}$ in Euler-angle representation.

In this thesis, when Bézier Curve Fitting is in combination with evenly selecting keyframes, the position of the two intermediate control points ($P1$ and $P2$) are computed using the dance motion of the training split with Levenberg–Marquardt algorithm. The average of $P1$ and $P2$ are computed respectively from the training split's predicted $P1$ and $P2$, and they are used as the predicted $P1$ and $P2$ points of dance motion in the test split. This is to ensure that Bézier Curve Fitting is fairly compared with linear interpolation, and both have no access to the ground-truth dance data when operating on unseen dances in the dance split. However, when Bézier Curve Fitting is in combination with the dynamic programming to select keyframes, the intermediate control points are optimized using the unseen dance motion. This is to ensure that I follow the strategy Roberts (2018) proposed to reduce an animation into keyframes and in-betweens.

3.5.3 Transformer Model

The final strategy use variations of deep learning models consisting of Transformer models to reconstruct in-between frames. Musical features and selected N keyframes are used as inputs to six model variations, whose output is the reconstructed motion. Six variations are listed in the following section.

Full Model/Full Track

The *Full Model/Full Track* consists of two transformer blocks: an audio transformer and an in-between transformer. The audio transformer uses the encoder model, and the in-between transformer uses the decoder model (see Chapter 2.3). Given a sequence of evenly-spaced keyframes KF_{even_rotmat} and musical features M_{full}, musical features are sent to the audio transformer first. Evenly-spaced keyframes, KF_{even_rotmat}, are linearly interpolated such that their length is the same as the original dance. The goal of the in-between transformer is to predict the difference between the original dance and the linearly-interpolated dance. The output of the audio transformer is sent to the in-between transformer as the key and value, and linearly-interpolated keyframes are sent as the query. To get the reconstructed dance motion, the output of the in-between transformer, which is the difference between the original dance and the linearly-interpolated dance, is added with the linearly-interpolated keyframes. The reconstructed motion of Full Model is referred to as $\hat{D}_{cross\_full}$

Full Model/Drum Track

This variation, *Full Model/Drum Track*, is almost identical to *Full Model/Full Track*. The only difference is the musical features. Unlike Full Mode/Full Track, which uses the musical feature of the full track, M_{full}, *Full Model/Full Track* uses musical features of the *drums stem*, M_{full_drum}, as the input to the audio transformer. The reconstructed dance is referred to as $\hat{D}_{cross\_drum}$

Full Model/Full Track/Merged

This variation, *Full Model/Full Track/Merged* shared the same model and feature as *Full Model/Full Track*, with the only difference in the way the output of the audio transformer and linearly-interpolated keyframes are sent to the in-between transformer. In this variation, the output and the interpolated keyframes are concatenated together first, and concatenated feature is sent as the key, value, and query to the in-between transformer model. The reconstructed dance from this variation is denoted as $\hat{D}_{merged\_full}$

Full Model/Drum Track/Merged

The model (*Full Model/Drum Track/Merged*) is the same as *Full Model/Full Track/Merged*, except that the musical feature is replaced with M_{full_drum}. The reconstructed dance is referred to as $\hat{D}_{merged\_drum}$

Genre

This variation (*Genre*) only contains the in-between transformer. Musical features are replaced with the genre vector M_{genre} that indicates the style of the dance music. The genre vector is concatenated with the linearly-interpolated keyframes, and the concatenated features are sent to the in-between transformer. The reconstructed dance is referred to as $\hat{D}_{genre}$.

Motion

This variation (*Motion*) reconstructs dance motion without using musical features or genre vectors. The audio transformer is removed, and linearly-interpolated keyframes are sent to the in-between transformers. The reconstructed dance is referred to as $\hat{D}_{motion}$.

3.6 Evaluation Metrics

The reconstructed dance is evaluated with mean squared error (MSE) to evaluate the difference between its original dance and itself (see Chapter 2.8) and examine whether the reconstructed dance is close to the original dance. Instead of evaluating each pair of reconstructed dances and generated dances, FID is used to evaluate how close a set of reconstructed dances is to a set of original dances. Kinematic feature and geometric feature extractors are used to extract features from original and reconstructed dances (see Chapter 2.8), and FID using the kinematic feature extractor is referred to as FID_k, using the geometric feature extractor is referred to as FID_g.

Chapter 4

Experiments

In this chapter, the experimental workflow discussed in Chapter 3 is executed and evaluated then the result of the evaluation metrics is discussed. The overall goal of the experiments is to test whether injecting musical features improves the results of various keyframe selection and in-between-reconstruction strategies. In Section 4.1, evaluation metrics are verified by applying metrics to assess dances reconstructed from different numbers of evenly selected keyframes. In Section 4.2, keyframe selection and in-between reconstruction strategies independent of musical features are used to reconstruct dances. Reconstructed dances are evaluated with metrics verified in Section 4.1. In Section 4.3, in-between strategies are the same as Section 4.2, and keyframe selection strategies are replaced with strategies using musical features. The reconstructed dances are compared with Section 4.2 to verify whether injecting musical features in keyframe selection strategies improves the reconstructed dances. In Section 4.4, keyframe selection strategies are the same as Section 4.2, and in-between

strategies are replaced with strategies using musical features. The reconstructed dances are compared with Section 4.2 to test whether injecting musical features in in-between strategies improves the reconstructed dances or not.

4.1 Metric Verification Experiment

Before starting experimental workflows, metrics are tested to verify their usefulness. The metrics used in this thesis are expected to return a higher value when evaluating inaccurately reconstructed dance motion, and they should return a lower value when evaluating well-reconstructed dance motion. I wanted to verify empirically that this was indeed the case. This experiment uses three strategies to select keyframes in Euler-angle representation: selecting keyframes evenly (KF_{even_rotmat}), using onset curve (KF_{onset}), and dynamic programming (KF_{dp}). After acquiring keyframes, linear interpolation is used to reconstruct in-between frames. This experimental setting to verify metrics is repeated for selecting nine, five, and three keyframes, and the results are evaluated with three proposed metrics: MSE, FID_g, and FID_k. The number of selected keyframes affects the reconstruction distance. As the number of selected keyframes increases, there are fewer frames between two adjacent keyframes that the in-between reconstruction strategy needs to reconstruct, which decreases the difficulty. In other words, the reconstructed dance using more keyframes should outperform those with fewer keyframes, and their values evaluated with metrics should decrease.

	3 keyframes	5 keyframes	9 keyframes
KF_{even_rotmat}	42.73	20.89	6.75
KF_{dp}	18.3	15.04	13.82
KF_{onset}	48.12	23.11	17.46

Table 4.1: Evaluate FID_g with three, five, and nine keyframes.

	3 keyframes	5 keyframes	9 keyframes
KF_{even_rotmat}	86.70	69.25	51.48
KF_{dp}	54.40	44.35	33.16
KF_{onset}	84.26	70.26	54.62

Table 4.2: Evaluate FID_k with three, five, and nine keyframes.

4.1.1 Experimental Results

The results of evaluating FID_g, FID_k, and MSE are shown in Table 4.1, 4.2, and 4.3, respectively. As shown in each table, selecting nine keyframes reaches the best result, and the value decreases when the number of selected keyframes increases.

4.1.2 Discussions

The experimental result matches the expected behavior of metrics, where increasing the number of selected keyframes decreases the value of metrics. This confirms the hypothesis that metrics for evaluating reconstructed dance exhibit the expected behaviour of metrics.

	3 keyframes	5 keyframes	9 keyframes
KF_{even_rotmat}	0.83	0.50	0.25
KF_{dp}	0.47	0.28	0.12
KF_{onset}	0.90	0.56	0.31

Table 4.3: Evaluate MSE with three, five, and nine keyframes.

4.2 Experiment 1: Reconstructing Dance Motion without Musical Features

This section evaluates the reconstructed dances of keyframe selection strategies and in-between reconstruction strategies without using musical features. The result of this experiment will be used as the baseline for comparison with the subsequent two experiments, which use musical features. There are four combinations of keyframe selection and in-between reconstruction strategies to reconstruct dances. Strategies used in this experiment include: evenly selecting keyframes using dance motion features and reconstructing the dance motion with linear interpolation ($KF_{even\_eulers}/\hat{D}_{eulers\_linear}$), Bézier curve fitting ($KF_{even\_eulers}/\hat{D}_{bezier}$), and Transformer model ($KF_{even\_eulers}/\hat{D}_{motion}$), and selecting keyframes with dynamic programming KF_{dp} and reconstructing in-betweens with Bézier curve fitting ($KF_{dp}/\hat{D}_{bezier}$). Four strategies are tested with selecting nine or three keyframes, and the reconstructed dances are evaluated with evaluation metrics.

4.2.1 Experimental Results

The results evaluated with MSE, FID_g, and FID_k with nine and three keyframes are shown in Table 4.4 and Table 4.5, respectively. The results in Table 4.4 suggest that $KF_{dp}/\hat{D}_{bezier}$ reaches the best result in MSE and FID_k and $KF_{even\_eulers}/\hat{D}_{motion}$ has the best result in FID_g. On the contrary, $KF_{even\_eulers}/\hat{D}_{bezier}$ performs the worst in MSE, FID_g, and FID_k.

The results of four strategies are listed in Table 4.5, which show that $KF_{dp}/\hat{D}_{bezier}$

	MSE	FID_g	FID_k
$KF_{even\_eulers}/\hat{D}_{eulers\_linear}$	0.25	16.75	51.48
$KF_{even\_eulers}/\hat{D}_{bezier}$	0.91	34.37	198.08
$KF_{even\_eulers}/\hat{D}_{motion}$	0.26	**11.02**	37.90
$KF_{dp}/\hat{D}_{bezier}$	**0.12**	13.82	**33.16**

Table 4.4: Evaluated results of dance reconstructed by selecting nine keyframes and reconstruct in-between frames without musical features.

	MSE	FID_g	FID_k
$KF_{even\_eulers}/\hat{D}_{eulers\_linear}$	0.83	42.73	86.70
$KF_{even\_eulers}/\hat{D}_{bezier}$	1.07	29.74	76.01
$KF_{even\_eulers}/\hat{D}_{motion}$	0.90	**16.57**	**33.76**
$KF_{dp}/\hat{D}_{bezier}$	**0.47**	18.30	54.40

Table 4.5: Evaluated results of dance reconstructed by selecting three keyframes and reconstruct in-between frames without musical features.

achieves the best result in MSE, while $KF_{even\_eulers}/\hat{D}_{bezier}$ has the worst. For FID_g and FID_k, $KF_{even\_eulers}/\hat{D}_{motion}$ has the best result and $KF_{even\_eulers}/\hat{D}_{eulers\_linear}$ has the worst.

4.2.2 Discussions

The results of both selecting nine and three keyframes show that selecting keyframes with dynamic programming and reconstruction inbetweens with Bézier curve fitting ($KF_{dp}/\hat{D}_{bezier}$) has the best result in MSE, which implies the least reconstruction error between the real dance and reconstructed dance.

Regarding both nine keyframes and three keyframes setting, the best FID_g result in $KF_{even\_eulers}/\hat{D}_{motion}$ suggests that compared to other strategies, the geometric features of its reconstructed dance are the closest to the real dance. This means that even though

$KF_{dp}/\hat{D}_{bezier}$ has the best reconstruction distance, the relation of body parts in dance reconstructed by $KF_{even\_eulers}/\hat{D}_{motion}$, is closer to the real dance.

When the dances are reconstructed with nine keyframes and evaluated with FID_k, $KF_{dp}/\hat{D}_{bezier}$ has the best result. However, when reducing the number of keyframes to three, $KF_{even\_eulers}/\hat{D}_{motion}$ outperforms $KF_{dp}/\hat{D}_{bezier}$. This seems to imply that when reconstructing dances with fewer keyframes, using Transformer model as the in-between strategy achieves reconstructed dances whose kinematic features are the closest to the real dances. The similar kinematic features may indicate that the velocity, relating to the way a dance pose transforms to the next, is more similar to the real dance.

4.3 Experiment 2: Select Keyframes with Musical Features

This section evaluates strategies to select keyframes using musical features but reconstruct in-betweens without musical features. Three combinations of strategies are used to reconstruct dances, which include: $KF_{onset}/\hat{D}_{eulers\_linear}$ that selects keyframes using onset peak positions of the onset strength and reconstructs in-betweens with linear interpolation, $KF_{stems}/\hat{D}_{eulers\_linear}$ that uses the weighted onset strength's onset peak positions to select keyframes and reconstructs in-betweens with linear interpolation, and $KF_{nmfd}/\hat{D}_{eulers\_linear}$ that selects keyframes by peak positions of the weighted activation curve and reconstructs in-betweens with linear interpolation. Strategies are evaluated with selecting nine and three

keyframes, and reconstructed dances of each combination are evaluated with three metrics to examine the best selection of musical features. The reconstructed dances using the three strategies will be compared against each other to evaluate the selection of musical features in keyframes selection strategies, and they will be compared with reconstructed dances in Experiment 1 to test if using musical features improves the reconstructed dance.

4.3.1 Experimental Results

The results of combination of strategies tested with nine and three keyframes are shown in Table 4.6 and Table 4.7, respectively. When using nine keyframes, selecting keyframes with onset peak positions and weighted onset peak positions outperform the weighted activation curve when reconstructed dances are evaluated with MSE. When the reconstructed dances are evaluated with FID_g, the results of $KF_{stems}/\hat{D}_{eulers\_linear}$ outperforms others. As to FID_k, $KF_{onset}/\hat{D}_{eulers\_linear}$ reaches the best result.

As shown in Table 4.7, with three keyframes, the result for MSE shows that selecting keyframes with the weighted activation curve outperforms other musical features. $KF_{stems}/\hat{D}_{eulers\_linear}$ reaches the best result in FID_g. As with selecting nine keyframes, $KF_{onset}/\hat{D}_{eulers-linear}$ has the best result in FID_k.

To evaluate the impact of musical features in keyframes selections strategies, the results in Table 4.6 and Table 4.7 are compared with results in Table 4.4 and Table 4.5. As shown in the Table 4.6 and Table 4.4, when using linear interpolation to reconstruct in-betweens, evenly selecting keyframes ($KF_{even\_eulers}/\hat{D}_{eulers\_linear}$) has the best results in all metrics, and selecting

	MSE	FID_g	FID_k
$KF_{onset}/\hat{D}_{eulers\_linear}$	**0.31**	17.46	**54.62**
$KF_{stems}/\hat{D}_{eulers\_linear}$	**0.31**	**17.08**	54.63
$KF_{nmfd}/\hat{D}_{eulers\_linear}$	0.35	17.53	55.27

Table 4.6: Results of selecting nine keyframes with keyframe selection strategies using musical features, and reconstruct in-between frames using strategies independent of musical features.

	MSE	FID_g	FID_k
$KF_{onset}/\hat{D}_{eulers\_linear}$	0.90	48.12	**84.26**
$KF_{stems}/\hat{D}_{eulers\_linear}$	0.89	**48.10**	84.92
$KF_{nmfd}/\hat{D}_{eulers\_linear}$	**0.87**	48.81	86.65

Table 4.7: Results of selecting three keyframes with keyframe selection strategies using musical features, and reconstruct in-between frames using strategies independent of musical features.

keyframes with peak positions of the weighted activation curve ($KF_{nmfd}/\hat{D}_{eulers\_linear}$) has the worst results.

When selected frames are reduced from nine to three, the following can be observed by comparing Table 4.7 and Table 4.5. When deriving in-between frames with linear interpolation, $KF_{even\_eulers}/\hat{D}_{eulers\_linear}$ has the best results in MSE and FID_g, while $KF_{onset}/\hat{D}_{eulers\_linear}$ and $KF_{nmfd}/\hat{D}_{eulers\_linear}$ have the worst result separately.

4.3.2 Discussions

Comparing the Selection of Musical Features

When evaluated with MSE, the onset curve (KF_{onset}) and weighted onset curve (KF_{stems}) outperform the weighted activation under nine keyframes. However, when the number of selected keyframes is reduced to three, the weighted activation curve (KF_{nmfd}) outperforms

other musical features. As for FID_g, regardless of the number of the selected keyframes, the weighted onset strength (KF_{stems}) are better musical features that improve the geometric features of the reconstructed dance.

When evaluated with FID_k, the onset strength surpasses other musical features regardless of the number of selected keyframes. This implies that the kinematic features of its reconstructed dance are close to the real dance, and knowing the musical components of dance music failed to improves kinematic features of the reconstructed dance.

Examining the Impact of Musical Features

Regardless of the number of selected keyframes, the results show that the evenly-spaced keyframes selection strategy reaches the best MSE and FID_g. It has the best reconstruction distance, and injecting musical features does not improve the reconstruction distance (MSE) and geometric features of the reconstructed dance (FID_g). This may be because injecting musical features results in unevenly selected keyframes. Compared to unevenly selecting keyframes, evenly selecting keyframes seems to better depict the overall poses of a motion, which results in better results in MSE and FID_g. Therefore, it achieves better results in MSE, which describes the reconstruction distance, and FID_g, which evaluates the distance of reconstructed dances' geometric features to those of the real dances.

When the number of keyframes is three, $KF_{onset}/\hat{D}_{eulers\_linear}$ has better results in FID_k. This implies that when reducing the number of keyframes, even though evenly selecting keyframes reaches a better reconstruction distance, the kinematic features of dances reconstructed from the onset-curve and dynamic programming strategy are closer to real

dance motion. Kinematic features extract the velocities of dance poses' joints, which are different from the geometric features used in FID_g that focus on using Boolean functions to describe the relation of body parts in dance poses. As a result, when selecting three keyframes, the better FID_k and inferior FID_g in $KF_{onset}/\hat{D}_{eulers\_linear}$ indicates that: although the overall poses of the reconstructed dance differ from the original dance, the way the reconstructed dances move from pose to pose, which is describes by kinematic features using velocities, is close to the original dance.

4.4 Experiment 3: Reconstruct In-betweens with Musical Features

This section evaluates combinations of keyframe selection strategies without musical features and in-between strategies with musical features. Five combinations of strategies are tested to reconstruct dance motion, including evenly selecting keyframes and reconstructing in-between frames with Transformer model using musical features of the dance music ($KF_{even}/\hat{D}_{cross\_full}$), drums stem ($KF_{even}/\hat{D}_{cross\_drum}$), using genre vectors $KF_{even}/\hat{D}_{genre}$, and using the same inputs (selected keyframes and musical features) as $KF_{even}/\hat{D}_{cross\_full}$ and $KF_{even}/\hat{D}_{cross\_drum}$ but concatenating inputs along the feature dimension, resulting $KF_{even}/\hat{D}_{merged\_full}$ and $KF_{even}/\hat{D}_{merged\_drum}$. Strategies are tested with selecting nine and three keyframes, and their reconstructed dances are evaluated with three metrics. The reconstructed dances using five strategies will be evaluated against each other to determine the best selection of musical

features in in-between reconstruction strategies. They will also be compared with reconstructed dances in Experiment 1 to assess the impact of musical features in in-between reconstruction strategies.

4.4.1 Experimental Results

The results of combination of strategies tested with nine and three keyframes are shown in Table 4.8 and Table 4.9, respectively. As shown in Table 4.8, for selecting nine keyframes, using genre vector ($KF_{even}/\hat{D}_{genre}$) has the best reconstruction distance (MSE), and using musical features of the drums stem ($KF_{even}/\hat{D}_{cross\_drum}$) has the best geometric features (FID_g). For both metrics, $KF_{even}/\hat{D}_{merged\_drum}$ has the worst result. As to assessing reconstructed dances with FID_k, $KF_{even}/\hat{D}_{merged\_drum}$ has the best result, while $KF_{even}/\hat{D}_{cross\_drum}$ performs the worst. For selecting three keyframes, as show in Table 4.9, $KF_{even}/\hat{D}_{genre}$ reaches the best results in all metrics, and $KF_{even}/\hat{D}_{cross\_full}$ has the worst results in all metrics.

To evaluate the impact of musical features in in-between reconstruction strategies, the experiment results in Table 4.8 and Table 4.9 are compared with Table 4.4 and Table 4.5, respectively. Firstly, the experiment is tested by selecting nine keyframes. The results in Table 4.8 and Table 4.4 suggest that reconstructing in-between frames with Bézier curve fitting ($KF_{even\_eulers}/\hat{D}_{bezier}$) performs the best in MSE, and using musical features of the *drums stem* concatenating it with keyframes ($KF_{even}/\hat{D}_{merged\_drum}$) performs the worst. When the reconstructed dances are evaluated in terms of FID_g and FID_k, $KF_{even\_eulers}/\hat{D}_{motion}$ and $KF_{even\_eulers}/\hat{D}_{merged\_drum}$ perform the best respectively, and $KF_{even\_eulers}/\hat{D}_{eulers\_linear}$

	MSE	FID_g	FID_k
$KF_{even}/\hat{D}_{cross\_full}$	0.34	12.84	35.75
$KF_{even}/\hat{D}_{cross\_drum}$	0.30	**11.10**	38.47
$KF_{even}/\hat{D}_{merged\_full}$	0.36	15.04	29.35
$KF_{even}/\hat{D}_{merged\_drum}$	0.37	15.14	**28.53**
$KF_{even}/\hat{D}_{genre}$	**0.27**	11.38	36.88

Table 4.8: The evaluated result of dances reconstructed by evenly selecting nine keyframes and deriving in-between frames using musical features.

	MSE	FID_g	FID_k
$KF_{even}/\hat{D}_{cross\_full}$	0.96	19.09	39.10
$KF_{even}/\hat{D}_{cross\_drum}$	0.95	18.81	37.17
$KF_{even}/\hat{D}_{merged\_drum}$	0.95	17.65	36.34
$KF_{even}/\hat{D}_{genre}$	**0.88**	**15.16**	**30.24**

Table 4.9: The evaluated result of dances reconstructed by evenly selecting three keyframes and deriving in-between frames using musical features.

performs the worst for these two metrics.

When reducing the number of selected keyframes to three, the following can be observed. The results in Table 4.9 and Table 4.7 show that when evaluating the reconstructed dances in terms of MSE, the result is similar to the experiment with nine keyframes, where $KF_{even\_eulers}/\hat{D}_{bezier}$ has the best result and $KF_{even}/\hat{D}_{merged\_drum}$ has the worst. When evaluating the reconstructed dances with FID_g and FID_k, the result suggested that $KF_{even}/\hat{D}_{genre}$ performs the best, and $KF_{even}/\hat{D}_{eulers\_linear}$ performs the worst.

4.4.2 Discussions

Comparing the Selection of Musical Features

Regardless of the number of selected keyframes, using genre vectors ($\hat{D}_{genre}$) surpasses other musical features in terms of reconstruction distance (MSE). As to FID_g, when evenly selecting nine keyframes, the musical features of the *drums stem* ($\hat{D}_{cross\_drum}/\hat{D}_{merged\_drum}$) outperforms other musical features. However, when changing the number of selected keyframes to three, $\hat{D}_{genre}$ surpasses other musical features. Lastly, when reconstructing dance with nine keyframes and evaluating them with FID_k, concatenating musical features of the *drums stem* with the selected keyframes ($\hat{D}_{merged\_drum}$) outperforms other musical features. However, similar to FID_g, when the number of selected keyframes is reduced to three, $\hat{D}_{genre}$ outperforms other musical features.

In conclusion, regardless of the number of selected keyframes, using genre vectors ($\hat{D}_{genre}$) as the musical features outperforms other musical features in terms of MSE, implying that the reconstructed dance has less reconstruction error. Furthermore, when reducing the number of selected keyframes to three, it also outperforms others in terms of FID_g and FID_k, implying that the geometric and kinematic features of its reconstructed dance are the closest to the real dance. However, when the number of keyframes is increased, using musical features of the *drums stem* ($\hat{D}_{cross\_drum}/\hat{D}_{merged\_drum}$) surpasses genre vectors. This suggests that understanding the musical components helps improve the geometric and kinematic features of the reconstructed dance when using more keyframes.

Examining the Impact of Musical Features

The result suggests that regardless of the number of selected keyframes, Bézier curve fitting, the in-between strategy without musical features, has the best result in MSE. This implies that musical features may not improve the reconstructed dance's reconstruction distance. However, the results are different when FID_g and FID_k are used to evaluate the reconstructed dance. For reconstructing nine keyframes, the Transformer model ($\hat{D}_{motion}$) depending on selected keyframes has the best FID_g result. The Transformer model with musical features of the drums stem, $\hat{D}_{merged\_drum}$, has the best FID_k result. In contrast, $\hat{D}_{eulers\_linear}$, which performs the best in MSE, has the worst result. This implies that even though $\hat{D}_{eulers\_linear}$, the interpolation strategy without musical features, has the best reconstruction distance (MSE), strategies using musical features reconstruct dances whose kinematic features are closer to the real dance. The result is the same when reducing the number of keyframes to three to increase the reconstruction difficulty. When using three keyframes, even though $\hat{D}_{eulers\_linear}$ still has the best reconstruction distance, $\hat{D}_{genre}$ has the best FID_g and FID_k result. This suggests that when in-between frames are derived from fewer keyframes, knowing the dance music genre helps improve the reconstruction distance of reconstructed motion.

4.5 Summary

In the metric verification experiment (Section 4.1), three evaluation metrics: MSE, FID_g, and FID_k were verified by using them to evaluate dances reconstructed from different numbers of evenly selected keyframes. The result suggested that they were valid metrics that were able to

reflect the reconstruction distance of the reconstructed dances.

In Experiment 1 (Section 4.2), where both keyframe selection and in-between reconstruction strategies did not use musical features, the result suggested that selecting keyframes with dynamic programming and reconstructing in-between frames with Bézier curve fitting achieved the best reconstruction distance, regardless of the number of selected keyframes. However, when reducing the number of selected keyframes, deriving in-between frames with the Transformer model reconstructed dances whose geometric features and kinematic features were close to the real dance. The results of Experiment 1 were used as the baseline to compare against the results of Experiments 2 and 3.

In Experiment 2 (Section 4.3), the experimental result suggested that with nine keyframes, the onset strength and weighted onset strength outperformed the weighted activation curve in terms of the reconstruction distance; when using three keyframes, the weighted activation curve outperformed other musical features. However, regardless of the number of selected keyframes, using the weighted onset strength of the dance music improved the geometric features of the reconstructed dance. Lastly, using the onset strength enhanced the kinematic features of the reconstructed dance. The result in Experiment 2 was compared with Experiment 1 (Table 4.10 and Table 4.11), and it suggested that when using strategies to select nine keyframes, injecting musical features failed to improve the reconstruction distance (MSE) and the geometric feature (FID_g) of the reconstructed dance. However, when selecting three keyframes and using linear interpolation as the in-between strategy, the keyframe selection strategy using onset strength was able to reconstruct dances with the best kinematic features (FID_k).

	MSE	FID_g	FID_k
$KF_{even\_eulers}/\hat{D}_{eulers\_linear}$	**0.25**	**16.75**	**51.48**
$KF_{onset}/\hat{D}_{eulers\_linear}$	0.31	17.46	54.62
$KF_{stems}/\hat{D}_{eulers\_linear}$	0.31	17.08	54.63
$KF_{nmfd}/\hat{D}_{eulers\_linear}$	0.35	17.53	55.27

Table 4.10: Evaluated results of dance reconstructed by selecting nine keyframes and reconstruct in-between frames with/without musical features.

	MSE	FID_g	FID_k
$KF_{even\_eulers}/\hat{D}_{eulers\_linear}$	**0.83**	**42.73**	86.70
$KF_{onset}/\hat{D}_{eulers\_linear}$	0.90	48.12	**84.26**
$KF_{stems}/\hat{D}_{eulers\_linear}$	0.89	48.10	84.92
$KF_{nmfd}/\hat{D}_{eulers\_linear}$	0.87	48.81	86.65

Table 4.11: Evaluated results of dance reconstructed by selecting three keyframes and reconstruct in-between frames with/without musical features.

In Experiment 3, where musical features were injected into in-between selection strategies, the result indicated that using genre vectors outperformed other musical features in terms of reconstruction distance. With nine keyframes, using musical features of the *drums stem* improved the geometric and kinematic features of the reconstructed dances. With three keyframes, using genre vectors reconstructed dances whose geometric and kinematic features were close to the real dance. The result was compared with Experiment 1 (Table 4.12 and Table 4.13), and it implied that: the reconstructed dance using Bézier curve fitting had the best MSE result. However, the geometric and kinematic features of dances reconstructed from musical features were closer to the real dance.

	MSE	FID_g	FID_k
$KF_{even\_eulers}/\hat{D}_{eulers\_linear}$	**0.25**	16.75	51.48
$KF_{even\_eulers}/\hat{D}_{bezier}$	0.91	34.37	198.08
$KF_{even\_eulers}/\hat{D}_{motion}$	0.26	**11.02**	37.90
$KF_{even}/\hat{D}_{cross\_full}$	0.34	12.84	35.75
$KF_{even}/\hat{D}_{cross\_drum}$	0.30	11.10	38.47
$KF_{even}/\hat{D}_{merged\_full}$	0.36	15.04	29.35
$KF_{even}/\hat{D}_{merged\_drum}$	0.37	15.14	**28.53**
$KF_{even}/\hat{D}_{genre}$	0.27	11.38	36.88

Table 4.12: Evaluated results of dance reconstructed by selecting nine keyframes and reconstruct in-between frames with/without musical features.

	MSE	FID_g	FID_k
$KF_{even\_eulers}/\hat{D}_{eulers\_linear}$	**0.83**	42.73	86.70
$KF_{even\_eulers}/\hat{D}_{bezier}$	1.07	29.74	76.01
$KF_{even\_eulers}/\hat{D}_{motion}$	0.90	16.57	33.76
$KF_{even}/\hat{D}_{cross\_full}$	0.96	19.09	39.10
$KF_{even}/\hat{D}_{cross\_drum}$	0.95	18.81	37.17
$KF_{even}/\hat{D}_{merged\_drum}$	0.95	17.65	36.34
$KF_{even}/\hat{D}_{genre}$	0.88	**15.16**	**30.24**

Table 4.13: Evaluated results of dance reconstructed by selecting three keyframes and reconstruct in-between frames with/without musical features.

Chapter 5

Conclusion

In this thesis, I attempted to show that injecting musical information of dance music improves the reconstruction distance of the reconstructed dance. A music source separation algorithm was used to separate dance music into four stems. Musical features of each stem were extracted by computing the spectral flux onset curve, Nonnegative Matrix Factorization Deconvolution ($NMFD$), Mel frequency cepstral coefficients ($MFCC$), and peak locations of the onset curve. Dance motion was reconstructed by first selecting a subset of frames as keyframes and reconstructing in-between frames using selected keyframes. Four experiments were conducted to evaluate dances reconstructed from different keyframe selection strategies and in-between-reconstruction strategies, with and without musical features. Three evaluation metrics were used to evaluate reconstructed dances: Mean squared error (MSE) was used to evaluate the difference between the original dance motion and the reconstructed dance motion. Frechet Inception Distance with geometric features (FID_g) and kinematic features

(FID_k) was used to evaluate the distance of geometric features and kinematic features of a set of generated dances and real dances.

In metric verification experiment, evaluation metrics were verified by evaluating reconstructed dance using three, five, and nine keyframes, which were expected to have reconstruction distance from low to high. The result suggested that the value of each metric decreased as the number of evenly selected keyframes increased, which reduced the difficulty for strategies to reconstruct dances and reduced the reconstruction distance of reconstructed dances. This implied that metrics were valid and could assess the reconstruction distance of reconstructed dances.

In Experiment 1, keyframe selection and in-between reconstruction strategies without musical features were employed to reconstruct dance motion. The result suggested that disregarding the number of selected keyframes, selecting keyframes with dynamic programming and reconstructing in-betweens with Bézier curve fitting achieved the best reconstruction distance. Besides, regardless of the number of selected keyframes, evenly selecting keyframes and reconstructing in-between frames with the Transformer model was able to reconstruct dances whose geometric features were close to the real dances. When selecting nine keyframes, selecting keyframes with dynamic programming and reconstructing in-betweens with Bézier curve fitting reconstructed dances with the best kinematic features. However, when reducing the number of selected keyframes to three, replacing the in-between strategy with the Transformer model reconstructed dances with the best kinematic features.

The result suggested that when using nine keyframes, the onset strength and weighted

onset strength outperforms the weighted activation curve in term of the reconstruction distance. When using three keyframes, the weighted activation curve outperforms others. The result suggested that regardless of the number of selected keyframes, the weighted onset strength improves the geometric features, and the onset strength improves the kinematic features. Compared to Experiment 1, the result indicated that regardless of the number of selected keyframes, musical features did not improve the reconstruction distance and geometric features. When selecting three keyframes and reconstructing dances with linear interpolation, the keyframe selection strategies using onset strength as musical features was able to reconstruct dances with the best kinematic features. This suggested that it outperformed its counterparts in terms of metrics that assess the kinematic features. Compared to keyframe selection strategies without musical features, its reconstructed dances moved more like real dances.

In Experiment 3, keyframes were selected evenly, and musical features were used to reconstruct in-between frames together with the selected keyframes. The result suggested that the in-between strategy using genre vectors reconstructed dances with the best reconstruction distance. When reconstructing nine keyframes, strategies using musical features from the *drums stem* were able to reconstruct dances with the least geometric and kinematic features distances. However, as the number of keyframes was reduced to three, the in-between strategy using genre vectors outperformed others, where the reconstructed dances had the least geometric and kinematic feature distances. Compared to Experiment 1, the result showed that regardless of the number of selected keyframes, in-between strategies with musical

features did not surpass strategies without musical features in terms of reconstruction distance. However, strategies using musical features were able to reconstruct dances whose geometric and kinematic features were closer to the real dances.

In conclusion, using musical features in keyframe selection and in-between reconstruction strategies did not improve reconstruction distance. However, musical features did improves the geometric and kinematic features of its reconstructed dances. This implies that although the reconstructed dances deviate from the real dances, the overall dance poses and the way a dance pose transforms to the next pose are closer to the real dances when musical features were used.

5.1 Future Work

Two future works are discussed in this section. Different input features relating to dance and music to the neural network can be explored further to see if certain features improve the reconstructed dance. For example, the geometric and kinematic features can be used as inputs to the neural network model, and in-between frames can be reconstructed by the model that is conditioned on keyframes, musical features, and geometric and kinematic features. Since the positions of predicted keyframes influence the difficulty of the interpolation strategy, other than predicting keyframes and in-between independently, they can be predicted jointly in future work. In this way, and neural network is trained to take dance music as input and directly predict the keyframes and in-betweens.

Proposing metrics to evaluate generated dances and whether a dance fits to its music is a future research topic. Recent works normally use only three metrics to evaluate a generated

dance: FID to evaluate the realness of a dance motion, metric calculating the variance of generated dance motions to evaluate the diversity of generated dances, and metric evaluating the beat-alignment score to measure how well a dance motion fits into its dance music. Nevertheless, there are no common standards for implementing these metrics. For example, for evaluating FID metrics, Li et al. (2021a) used geometric and kinematic features defined in Müller, Röder, and Clausen (2005) and Onuma, Faloutsos, and Hodgins (2008), which were originally designed for motion recognition. However, Ren et al. (2020) used a pre-trained Graph Convolutional Network (Yan, Xiong, and Lin 2018) as the feature extractor to evaluate FID metrics. No studies focus on proposing evaluation metrics to evaluate generated dance and assess whether a dance fits its accompanying dance music. Studies on evaluation metrics are needed to establish a standard to compare and assess the generated dance motion.

Appendix A

Hyperparameters

This section includes the hyperparameters of the Transformer and the split of the AIST++ dataset. The number of multi-head self-attention layers is four, with the number of heads set to 10 in each layer. A fully-connected layer is used to transform the dimension of the input musical or motion features into 800-d vectors. Two fully-connected layers whose output dimension is 3072-d and 800-d, respectively, are attached to the end of each multi-head self-attention layer. At the end of the last multi-head attention layer, a fully-connected layer is used to transform the dimension from 800-d back to the dimension for the input data. The activation function is the softmax function, and the dropout rate is set to 0.1.

Appendix B

Dataset Split

The dataset split follows the split proposed in the AIST++ dataset, where the choreography and music in the test/validation split is not overlapped with the training split. There are 50 music pieces and eight choreographies in the training split and ten music pieces and two choreographies in the test/validation split. The numbers of dance motion in the training split and test/validation split are 980 and 40, respectively.

References

Abdoli, Sajjad, Patrick Cardinal, and Alessandro Lameiras Koerich. 2019. "End-to-End Environmental Sound Classification using a 1D Convolutional Neural Network." *Expert Systems with Applications* 136:252–263.

Aristidou, Andreas, Efstathios Stavrakis, Panayiotis Charalambous, Yiorgos Chrysanthou, and Stephania Loizidou Himona. 2015. "Folk Dance Evaluation using Laban Movement Analysis." *Journal on Computing and Cultural Heritage* 8 (4): 1–19.

Aristidou, Andreas, Efstathios Stavrakis, and Yiorgas Chrysanthou. 2014. "Cypriot Intangible Cultural Heritage: Digitizing Folk Dances." *Cyprus Computer Society Journal* 25:42–49.

Badrinarayanan, Vijay, Alex Kendall, and Roberto Cipolla. 2017. "Segnet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39 (12): 2481–2495.

Bahuleyan, Hareesh. 2018. "Music Genre Classification using Machine Learning Techniques." *arXiv preprint arXiv:1804.01149.*

Baniya, Babu Kaji, Deepak Ghimire, and Joonwhoan Lee. 2015. "Automatic Music Genre Classification using Timbral Texture and Rhythmic Content Features." In *2015 17th International Conference on Advanced Communication Technology,* 434–443. IEEE.

Barnard, Kobus, Pinar Duygulu, David Forsyth, Nando De Freitas, David M. Blei, and Michael I. Jordan. 2003. "Matching Words and Pictures." *The Journal of Machine Learning Research* 3:1107–1135.

Beauguitte, Pierre, Bryan Duggan, and John D. Kelleher. 2016. "A Corpus of Annotated Irish Traditional Dance Music Recordings: Design and Benchmark Evaluations." In *Proceedings of the International Society for Music Information Retrieval Conference,* 53–59.

Bellekens, Ben, Vincent Spruyt, Rafael Berkvens, and Maarten Weyn. 2014. "A Survey of Rigid 3d Pointcloud Registration Algorithms." In *Proceedings of the Fourth International Conference on Ambient Computing, Applications,* 8–13.

Benesty, Jacob, Jingdong Chen, Yiteng Huang, and Israel Cohen. 2009. "Pearson Correlation Coefficient." In *Noise Reduction in Speech Processing,* 1–4. Springer.

Blake, Randolph, and Maggie Shiffrar. 2007. "Perception of Human Motion." *Annual Review of Psychology* 58 (1): 47–73.

Bloomenthal, Jules, and Jon Rokne. 1994. "Homogeneous coordinates." *The Visual Computer* 11 (1): 15–26.

Borji, Ali. 2019. "Pros and Cons of GAN Evaluation Measures." *Computer Vision and Image Understanding* 179:41–65.

Brick, Timothy R., and Steven M. Boker. 2011. "Correlational Methods for Analysis of Dance Movements." *Dance Research* 29 (supplement): 283–304.

Bulut, Eyuphan, and Tolga Capin. 2007. "Key Frame Extraction from Motion Capture Data by Curve Saliency." In *Computer Animation and Social Agents,* vol. 20.

Cao, Zhe, Tomas Simon, Shih-En Wei, and Yaser Sheikh. 2017. "Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields." In *Proceedings of the IEEE / CVF Computer Vision and Pattern Recognition Conference.*

Chen, Kang, Zhipeng Tan, Jin Lei, Song-Hai Zhang, Yuan-Chen Guo, Weidong Zhang, and Shi-Min Hu. 2021. "ChoreoMaster: Choreography-Oriented Music-Driven Dance Synthesis." *ACM Transactions on Graphics* 40 (4).

Cheuk, Kin Wai, Kat Agres, and Dorien Herremans. 2020. "The Impact of Audio Input Representations on Neural Network based Music Transcription." In *Proceedings of the 2020 International Joint Conference on Neural Networks,* 1–6. IEEE.

Cho, Kyunghyun, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation." In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing,* 1724–1734.

Choi, Yunjey, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. 2020. "Stargan v2: Diverse Image Synthesis for Multiple Domains." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition,* 8188–8197.

Cohen, Patricia, Stephen G. West, and Leona S. Aiken. 2014. *Applied Multiple Regression/Correlation Analysis for the Behavioral Sciences.* Psychology Press.

Davis, Abe, and Maneesh Agrawala. 2018. "Visual Rhythm and Beat." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops,* 2532–2535.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,* 4171–4186. Association for Computational Linguistics.

Diebel, James. 2006. "Representing Attitude: Euler Angles, Unit Quaternions, and Rotation Vectors." *Matrix* 58 (15-16): 1–35.

Ding, Ming, Zhuoyi Yang, Wenyi Hong, Wendi Zheng, Chang Zhou, Da Yin, Junyang Lin, Xu Zou, Zhou Shao, and Hongxia Yang. 2021. "Cogview: Mastering text-to-image Generation via Transformers." *Advances in Neural Information Processing Systems* 34:19822–19835.

Dixon, Simon, Elias Pampalk, and Gerhard Widmer. 2003. "Classification of Dance Music by Periodicity Patterns." In *Proceedings of the International Society for Music Information Retrieval Conference,* 26–30.

Donahue, Chris, Julian McAuley, and Miller Puckette. 2019. "Adversarial Audio Synthesis." In *Proceedings of the International Conference on Learning Representations.*

Dong, Li, Furu Wei, Chuanqi Tan, Duyu Tang, Ming Zhou, and Ke Xu. 2014. "Adaptive Recursive Neural Network for Target-dependent Twitter Sentiment Classification." In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (volume 2: Short papers),* 49–54.

Dong, Linhao, Shuang Xu, and Bo Xu. 2018. "Speech-transformer: a no-recurrence Sequence-to-sequence Model for Speech Recognition." In *Proceedings of the 2018 IEEE International Conference on Acoustics, Speech and Signal Processing,* 5884–5888. IEEE.

Dosovitskiy, Alexey, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, et al. 2021. "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale." In *Proceedings of the International Conference on Learning Representations.*

Engel, Jesse, Lamtharn Hantrakul, Chenjie Gu, and Adams Roberts. 2020. "DDSP: Differentiable Digital Signal Processing." In *Proceedings of the International Conference on Learning Representations.*

Faraway, Julian J., Matthew P. Reed, and Jing Wang. 2007. "Modelling Three-dimensional Trajectories by using Bézier Curves with Application to Hand Motion." *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 56 (5): 571–585.

Fitch, W. Tecumseh. 2016. "Dance, Music, Meter and Groove: A Forgotten Partnership." *Frontiers in Human Neuroscience* 10:64.

Foley, James D., Foley Dan Van, Andries Van Dam, Steven K. Feiner, John F. Hughes, and J. Hughes. 1996. *Computer Graphics: Principles and Practice.* Vol. 12110. Addison-Wesley Professional.

Frahm, Jan-Michael, Kevin Köser, and Reinhard Koch. 2004. "Pose Estimation for Multi-camera Systems." In *Proceedings of the Joint Pattern Recognition Symposium,* 286–293. Springer.

Fréchet, Maurice. 1957. "Sur la distance de deux lois de probabilité." *Comptes rendus hebdomadaires des séances de l'académie des sciences* 244 (6): 689–692.

Frome, Andrea, Greg S. Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Marc'Aurelio Ranzato, and Tomas Mikolov. 2013. "Devise: A Deep Visual-semantic Embedding Model." *Advances in Neural Information Processing Systems* 26.

Gao, Ruohan, and Kristen Grauman. 2019. "2.5d Visual Sound." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition,* 324–333.

Girdhar, Rohit, Joao Carreira, Carl Doersch, and Andrew Zisserman. 2019. "Video Action Transformer Network." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* June.

Groff, Ed. 1995. "Laban Movement Analysis: Charting the Ineffable Domain of Human Movement." *Journal of Physical Education, Recreation & Dance* 66 (2): 27–30.

Halit, Cihan, and Tolga Capin. 2011. "Multiscale Motion Saliency for Keyframe Extraction from Motion Capture Sequences." *Computer Animation and Virtual Worlds* 22 (1): 3–14.

Han, Kai, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, et al. 2022. "A Survey on Visual Transformer." *IEEE Transactions on Pattern Analysis and Machine Intelligence.*

Hawthorne, Curtis, Ian Simon, Rigel Swavely, Ethan Manilow, and Jesse Engel. 2021. "Sequence-to-Sequence Piano Transcription with Transformers." In *Proceedings of the International Society for Music Information Retrieval Conference.*

He, Kaiming, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. 2017. "Mask R-CNN." In *Proceedings of the IEEE International Conference on Computer Vision,* 2961–2969.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37 (9): 1904–1916.

Heusel, Martin, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. 2017. "GANs Trained by a Two Time-scale Update Rule Converge to a Local Nash Equilibrium." *Advances in Neural Information Processing Systems* 30.

Huang, Ke-Sen, Chun-Fa Chang, Yu-Yao Hsu, and Shi-Nine Yang. 2005. "Key Probe: a Technique for Animation Keyframe Extraction." *The Visual Computer* 21 (8): 532–541.

Huang, Yukun, Yongcai Guo, and Chao Gao. 2020. "Efficient Parallel Inflated 3D Convolution Architecture for Action Recognition." *IEEE Access* 8:45753–45765.

Huzaifah, Muhammad. 2017. "Comparison of Time-Frequency Representations for Environmental Sound Classification using Convolutional Neural Networks." *arXiv preprint arXiv:1706.07156.*

Irie, Kazuki, Zoltán Tüske, Tamer Alkhouli, Ralf Schlüter, and Hermann Ney. 2016. "LSTM, GRU, Highway and a Bit of Attention: An Empirical Overview for Language Modeling in Speech Recognition." In *Proceedings of the Interspeech,* 3519–3523.

Jensenius, Alexander Refsum. 2013. "Some Video Abstraction Techniques for Displaying Body Movement in Analysis and Performance." *Leonardo* 46 (1): 53–60.

Karras, Tero, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. 2020. "Training Generative Adversarial Networks with Limited Data." *Advances in Neural Information Processing Systems* 33:12104–12114.

Kell, Thor, and George Tzanetakis. 2013. "Empirical Analysis of Track Selection and Ordering in Electronic Dance Music using Audio Feature Extraction." In *Proceedings of the International Society for Music Information Retrieval Conference,* 505–510.

Khan, Murtaza Ali. 2016. "An Efficient Algorithm for Compression of Motion Capture Signal using Multidimensional Quadratic Bézier curve Break-and-fit Method." *Multidimensional Systems and Signal Processing* 27 (1): 121–143.

Kilgour, Kevin, Mauricio Zuluaga, Dominik Roblek, and Matthew Sharifi. 2019. "Fréchet Audio Distance: A Reference-Free Metric for Evaluating Music Enhancement Algorithms." In *Proceedings of the International Speech Communication Association,* 2350–2354.

Kitagawa, Midori, and Brian Windsor. 2008. *MoCap for Artists: Workflow and Techniques for Motion Capture.* Focal Press.

Knees, Peter, Ángel Faraldo Pérez, Herrera Boyer, Richard Vogl, Sebastian Böck, Florian Hörschläger, and Mickael Le Goff. 2015. "Two Data Sets for Tempo Estimation and Key Detection in Electronic Dance Music Annotated from User Corrections." In *Proceedings of the International Society for Music Information Retrieval Conference,* 364–370.

Kochanek, Doris HU, and Richard H. Bartels. 1984. "Interpolating Splines with Local Tension, Continuity, and Bias Control." In *Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques,* 33–41.

Kong, Qiuqiang, Yin Cao, Haohe Liu, Keunwoo Choi, and Yuxuan Wang. 2021. "Decoupling Magnitude and Phase Estimation with Deep ResUNet for Music Source Separation." In *Proceedings of the International Society for Music Information Retrieval Conference.*

Krumhansl, Carol L., and Diana Lynn Schenck. 1997. "Can Dance Reflect the Structural and Expressive Qualities of music? A Perceptual Experiment on Balanchine's Choreography of Mozart's Divertimento No. 15." *Musicae Scientiae* 1 (1): 63–85.

Kucuk, Serdar, and Zafer Bingul. 2006. *Robot Kinematics: Forward and Inverse Kinematics.* INTECH Open Access Publisher London, UK.

Kwon, Taegyun, Dasaem Jeong, and Juhan Nam. 2017. "Audio-to-Score Alignment of Piano Music using RNN-based Automatic Music Transcription." *ArXiv* abs/1711.04480.

LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. 2015. "Deep Learning." *Nature* 521 (7553): 436–444.

Lee, Hsin-Ying, Xiaodong Yang, Ming-Yu Liu, Ting-Chun Wang, Yu-Ding Lu, Ming-Hsuan Yang, and Jan Kautz. 2019. "Dancing to Music." *Advances in Neural Information Processing Systems* 32.

Leman, Marc, Dirk Moelants, Matthias Varewyck, Frederik Styns, Leon van Noorden, and Jean-Pierre Martens. 2013. "Activating and Relaxing Music Entrains the Speed of Beat Synchronized Walking." *PloS one* 8 (7): e67932.

Levenberg, Kenneth. 1944. "A Method for the Solution of Certain Non-linear Problems in Least Squares." *Quarterly of Applied Mathematics* 2 (2): 164–168.

Li, Bo, Lionel Heng, Kevin Koser, and Marc Pollefeys. 2013. "A Multiple-camera System Calibration Toolbox using a Feature Descriptor-based Calibration Pattern." In *Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems,* 1301–1307. IEEE.

Li, Buyu, Yongchi Zhao, Shi Zhelun, and Lu Sheng. 2022. "Danceformer: Music Conditioned 3d Dance Generation with Parametric Motion Transformer." In *Proceedings of the AAAI Conference on Artificial Intelligence,* 36:1272–1279. 2.

Li, Fen, Ming Liu, Yuejin Zhao, Lingqin Kong, Liquan Dong, Xiaohua Liu, and Mei Hui. 2019. "Feature Extraction and Classification of Heart Sound using 1D Convolutional Neural Networks." *European Association For Signal Processing Journal on Advances in Signal Processing* 2019 (1): 1–11.

Li, Ruilong, Shan Yang, David A Ross, and Angjoo Kanazawa. 2021a. "AI Choreographer: Music Conditioned 3D Dance Generation with AIST++." In *Proceedings of the International Conference on Computer Vision,* 13401–13412.

Li, Zewen, Fan Liu, Wenjie Yang, Shouheng Peng, and Jun Zhou. 2021b. "A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects." *IEEE Transactions on Neural Networks and Learning Systems.*

Lim, Ik Soo, and D. Thalmann. 2001. "Key-posture Extraction out of Human Motion Data." In *2001 Conference Proceedings of the 23rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society,* 2:1167–1169.

Lin, Kevin, Lijuan Wang, and Zicheng Liu. 2021. "End-to-End Human Pose and Mesh Reconstruction with Transformers." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition,* 1954–1963.

Liu, Xian-mei, Ai-min Hao, and Dan Zhao. 2013. "Optimization-based Key Frame Extraction for Motion Capture Animation." *The Visual Computer* 29 (1): 85–95.

Liu, Yinhan, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. "Roberta: A Robustly Optimized BERT Pretraining Approach." *arXiv preprint arXiv:1907.11692.*

Loper, Matthew, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. 2015. "SMPL: A skinned multi-person linear model." *ACM Transactions on Graphics* 34 (6): 1–16.

Lu, Rui, Kailun Wu, Zhiyao Duan, and Changshui Zhang. 2017. "Deep Ranking: Triplet MatchNet for Music Metric Learning." In *Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing,* 121–125. IEEE.

Lucic, Mario, Karol Kurach, Marcin Michalski, Sylvain Gelly, and Olivier Bousquet. 2018. "Are GANs Created Equal? a large-scale Study." *Advances in Neural Information Processing Systems* 31.

Luo, Yue, Jimmy Ren, Zhouxia Wang, Wenxiu Sun, Jinshan Pan, Jianbo Liu, Jiahao Pang, and Liang Lin. 2018. "LSTM Pose Machines." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* 5207–5215.

Minaee, Shervin, Yuri Y. Boykov, Fatih Porikli, Antonio J. Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. 2021. "Image Segmentation using Deep Learning: A Survey." *IEEE Transactions on Pattern Analysis and Machine Intelligence.*

Miura, Takeshi, Takaaki Kaiga, Hiroaki Katsura, Katsubumi Tajima, Takeshi Shibata, and Hideo Tamamoto. 2014. "Adaptive Keypose Extraction from Motion Capture Data." *Journal of Information Processing* 22 (1): 67–75.

Moré, Jorge J. 1978. "The Levenberg-Marquardt Algorithm: Implementation and Theory." In *Numerical Analysis,* 105–116. Springer.

Müller, Meinard. 2015. *Fundamentals of Music Processing: Audio, Analysis, Algorithms, Applications.* Vol. 5. Springer.

Müller, Meinard, Tido Röder, and Michael Clausen. 2005. "Efficient Content-based Retrieval of Motion Capture Data." In *Association of Computing Machinery's Special Interest Group on Computer Graphics and Interactive Techniques,* 677–685.

Navaza, Jorge. 1990. "Accurate Computation of the Rotation Matrices." *Acta Crystallographica Section A: Foundations of Crystallography* 46 (7): 619–620.

Newell, Alejandro, Kaiyu Yang, and Jia Deng. 2016. "Stacked Hourglass Networks for Human Pose Estimation." In *Proceedings of the European Conference on Computer Vision,* 483–499. Springer.

Olagoke, Adeshina Sirajdin, Haidi Ibrahim, and Soo Siang Teoh. 2020. "Literature Survey on Multi-camera System and its Application." *IEEE Access* 8:172892–172922.

Onuma, Kensuke, Christos Faloutsos, and Jessica K. Hodgins. 2008. "FMDistance: A Fast and Effective Distance Function for Motion Capture Data." In *Proceedings of the Eurographics (Short Papers),* 83–86.

Panteli, Maria, Niels Bogaards, and Aline K. Honingh. 2014. "Modeling Rhythm Similarity for Electronic Dance Music." In *Proceedings of the International Society for Music Information Retrieval Conference,* 537–542.

Papandreou, George, Tyler Zhu, Nori Kanazawa, Alexander Toshev, Jonathan Tompson, Chris Bregler, and Kevin Murphy. 2017. "Towards Accurate Multi-person Pose Estimation in the Wild." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* 4903–4911.

Qiu, Zhaofan, Ting Yao, and Tao Mei. 2017. "Learning Spatio-Temporal Representation With Pseudo-3D Residual Networks." In *Proceedings of the IEEE International Conference on Computer Vision.*

Ramesh, Aditya, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. 2021. "Zero-shot Text-to-Image Generation." In *Proceedings of the International Conference on Machine Learning,* 8821–8831.

Redmon, Joseph, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. "You Only Look Once: Unified, Real-time Object Detection." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* 779–788.

Remondino, Fabio, and Clive Fraser. 2006. "Digital Camera Calibration Methods: Considerations and Comparisons." *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 36 (5): 266–272.

Ren, Shaoqing, Kaiming He, Ross Girshick, and Jian Sun. 2015. "Faster R-CNN: Towards Real-time Object Detection with Region Proposal Networks." *Advances in Neural Information Processing Systems* 28.

Ren, Xuanchi, Haoran Li, Zijian Huang, and Qifeng Chen. 2020. "Self-supervised Dance Video Synthesis Conditioned on Music." In *Proceedings of the 28th ACM International Conference on Multimedia,* 46–54.

Roberts, Richard. 2018. *Converting Motion Capture Into Editable Keyframe Animation: Fast, Optimal, and Generic Keyframe Selection.* Victoria University of Wellington.

Rogez, Gregory, Philippe Weinzaepfel, and Cordelia Schmid. 2019. "LCR-Net++: Multi-person 2d and 3d Pose Detection in Natural Images." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42 (5): 1146–1161.

Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. 2015. "U-Net: Convolutional networks for Biomedical Image Segmentation." In *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention,* 234–241. Springer.

Ruggero Ronchi, Matteo, and Pietro Perona. 2017. "Benchmarking and Error Diagnosis in Multi-Instance Pose Estimation." In *Proceedings of the IEEE International Conference on Computer Vision,* 369–378.

Salomon, David. 2006. *Transformations and Projections In Computer Graphics.* Vol. 233. Springer.

Sarabandi, Soheil, and Federico Thomas. 2019. "A Survey on the Computation of Quaternions from Rotation Matrices." *Journal of Mechanisms and Robotics* 11 (2).

Shepperd, Stanley W. 1978. "Quaternion from Rotation Matrix." *Journal of Guidance and Control* 1 (3): 223–224.

Shiratori, Takaaki, and Katsushi Ikeuchi. 2008. "Synthesis of Dance Performance based on Analyses of Human Motion and Music." *Information and Media Technologies* 3 (4): 834–847.

Shirley, Peter, Michael Ashikhmin, and Steve Marschner. 2009. *Fundamentals of Computer Graphics.* AK Peters/CRC Press.

Shoemake, Ken. 1985. "Animating Rotation with Quaternion Curves." In *Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques,* 245–254. New York, NY, USA: Association for Computing Machinery.

Smaragdis, Paris. 2004. "Non-negative Matrix Factor Deconvolution; Extraction of Multiple Sound Sources from Monophonic Inputs." In *Proceedings of the International Conference on Independent Component Analysis and Signal Separation,* 494–499. Springer.

Smaragdis, Paris, Cedric Fevotte, Gautham J. Mysore, Nasser Mohammadiha, and Matthew Hoffman. 2014. "Static and Dynamic Source Separation using Nonnegative Factorizations: A Unified View." *IEEE Signal Processing Magazine* 31 (3): 66–75.

Solberg, Ragnhild Torvanger, and Alexander Refsum Jensenius. 2017. "Pleasurable and Intersubjectively Embodied Experiences of Electronic Dance Music." *Empirical Musicology Review* 11 (3-4): 301–318.

———. 2019. "Group Behaviour and Interpersonal Synchronization to Electronic Dance Music." *Musicae Scientiae* 23 (1): 111–134.

Stöter, Fabian-Robert, Stefan Uhlich, Antoine Liutkus, and Yuki Mitsufuji. 2019. "Open-unmix-a Reference Implementation for Music Source Separation." *Journal of Open Source Software* 4 (41): 1667.

Sun, Chen, Fabien Baradel, Kevin Murphy, and Cordelia Schmid. 2019. "Learning Video Representations using Contrastive Bidirectional Transformer." *arXiv preprint arXiv:1906.05743.*

Takahashi, Naoya, and Yuki Mitsufuji. 2017. "Multi-scale Multi-band Densenets for Audio Source Separation." In *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics,* 21–25. IEEE.

Tang, Kai-Tai, Howard Leung, Taku Komura, and Hubert P. H. Shum. 2008. "Finding Repetitive Patterns in 3D Human Motion Captured Data." In *Proceedings of the 2nd International Conference on Ubiquitous Information Management and Communication,* 396–403.

Tay, Yi, Mostafa Dehghani, Dara Bahri, and Donald Metzler. 2022. "Efficient Transformers: A Survey." *ACM Computing Surveys.*

Thomas, Frank, Ollie Johnston, and Frank Thomas. 1995. *The Illusion of Life: Disney Animation.* Hyperion New York.

Togawa, Hideyuki, and Masahiro Okuda. 2005. "Position-based Keyframe Selection for Human Motion Animation." In *Proceedings of the 11th International Conference on Parallel and Distributed Systems,* 2:182–185. IEEE.

Toiviainen, Petri, Geoff Luck, and Marc R. Thompson. 2010. "Embodied Meter: Hierarchical Eigenmodes in Music-induced Movement." *Music Perception* 28 (1): 59–70.

Tommi, Himberg, and Marc R. Thompson. 2011. "Learning and Synchronising Dance Movements in South African Songs: Cross-cultural Motion-Capture Study." *Dance Research* 29 (supplement): 305–328.

Tran, Van-Khanh, and Le-Minh Nguyen. 2017. "Natural Language Generation for Spoken Dialogue System using RNN Encoder-Decoder Networks." In *Proceedings of the 21st Conference on Computational Natural Language Learning,* 442–451.

Triggs, Bill, Philip F. McLauchlan, Richard I. Hartley, and Andrew W. Fitzgibbon. 1999. "Bundle Adjustment: A Modern Synthesis." In *Proceedings of the International Workshop on Vision Algorithms,* 298–372. Springer.

Tsuchida, Shuhei, Satoru Fukayama, Masahiro Hamasaki, and Masataka Goto. 2019. "AIST Dance Video Database: Multi-Genre, Multi-Dancer, and Multi-Camera Database for Dance Information Processing." In *Proceedings of the International Society for Music Information Retrieval Conference,* 501–510.

Uhlich, Stefan, Franck Giron, and Yuki Mitsufuji. 2015. "Deep Neural Network Based Instrument Extraction from Music." In *Proceedings of the 2015 IEEE International Conference on Acoustics, Speech and Signal Processing,* 2135–2139. IEEE.

Uhlich, Stefan, Marcello Porcu, Franck Giron, Michael Enenkl, Thomas Kemp, Naoya Takahashi, and Yuki Mitsufuji. 2017. "Improving Music Source Separation based on Deep Neural Networks through Data Augmentation and Network Blending." In *Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing,* 261–265. IEEE.

Ullah, Amin, Jamil Ahmad, Khan Muhammad, Muhammad Sajjad, and Sung Wook Baik. 2017. "Action Recognition in Video Sequences using Deep Bi-directional LSTM with CNN Features." *IEEE Access* 6:1155–1166.

Unterthiner, Thomas, Sjoerd van Steenkiste, Karol Kurach, Raphaël Marinier, Marcin Michalski, and Sylvain Gelly. 2019. *FVD: A New Metric for Video Generation.* https://openreview.net/forum?id=rylgEULtdN.

Van Dyck, Edith, Dirk Moelants, Michiel Demey, Alexander Deweppe, Pieter Coussement, and Marc Leman. 2012. "The Impact of the Bass Drum on Human Dance Movement." *Music Perception: An Interdisciplinary Journal* 30 (4): 349–359.

Van Dyck, Edith, Bart Moens, Jeska Buhmann, Michiel Demey, Esther Coorevits, Simone Dalla Bella, and Marc Leman. 2015. "Spontaneous Entrainment of Running Cadence to Music Tempo." *Sports Medicine-Open* 1 (1): 1–14.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. "Attention is All You Need." *Advances in Neural Information Processing Systems* 30.

Vishnupriya, S., and K. Meenakshi. 2018. "Automatic Music Genre Classification using Convolution Neural Network." In *Proceedings of the 2018 International Conference on Computer Communication and Informatics,* 1–4. IEEE.

Williams, Richard. 2012. *The Animator's Survival Kit: A Manual of Methods, Principles and Formulas for Classical, Computer, Games, Stop Motion and Internet Animators.* Macmillan.

Wolf, Thomas, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. "Transformers: State-of-the-art Natural Language Processing." In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing,* 38–45.

Xie, Jie, Kai Hu, Mingying Zhu, Jinghu Yu, and Qibing Zhu. 2019. "Investigation of Different CNN-based Models for Improved Bird Sound Classification." *IEEE Access* 7:175353–175361.

Yahya, Muhammad, Jawad Ali Shah, Kushsairy Abdul Kadir, Zulkhairi M. Yusof, Sheroz Khan, and Arif Warsi. 2019. "Motion Capture Sensing Techniques used in Human Upper Limb motion: A review." *Sensor Review.*

Yamashita, Rikiya, Mizuho Nishio, Richard Kinh Gian Do, and Kaori Togashi. 2018. "Convolutional Neural Networks: an Overview and Spplication in Tadiology." *Insights into Imaging* 9 (4): 611–629.

Yan, An, Yali Wang, Zhifeng Li, and Yu Qiao. 2019. "PA3D: Pose-Action 3D Machine for Video Recognition." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* June.

Yan, Sijie, Yuanjun Xiong, and Dahua Lin. 2018. "Spatial Temporal Graph Convolutional Networks for Skeleton-Based Action Recognition." *Proceedings of the Association for the Advancement of Artificial Intelligence Conference on Artificial Intelligence* 32 (1).

Yang, Yang, Howard Leung, Lihua Yue, and Liqun Deng. 2013. "Generating a Two-phase Lesson for Guiding Beginners to Learn Basic Dance Movements." *Computers & Education* 61:1–20.

Ye, Zijie, Haozhe Wu, Jia Jia, Yaohua Bu, Wei Chen, Fanbo Meng, and Yanfeng Wang. 2020. "Choreonet: Towards Music to Dance Synthesis with Choreographic Action Unit." In *Proceedings of the 28th ACM International Conference on Multimedia,* 744–752.

Yoo, Hyeon-Joong. 2015. "Deep Convolution Neural Networks in Computer Vision: A Review." *IEIE Transactions on Smart Processing and Computing* 4 (1): 35–43.

Yu, Yong, Xiaosheng Si, Changhua Hu, and Jianxun Zhang. 2019. "A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures." *Neural Computation* 31 (7): 1235–1270.

Zhang, Qiang, Shao-Pei Yu, Dong-Sheng Zhou, and Xiao-Peng Wei. 2013. "An Efficient Method of Key-frame Extraction based on a Cluster Algorithm." *Journal of Human Kinetics* 39:5–13.

Zhang, Weichen, Zhiguang Liu, Liuyang Zhou, Howard Leung, and Antoni B. Chan. 2017. "Martial Arts, Dancing and Sports Dataset: A Challenging Stereo and Multi-view Dataset for 3d Human Pose Estimation." *Image and Vision Computing* 61:22–39.

Zhang, Yu, Guoguo Chen, Dong Yu, Kaisheng Yao, Sanjeev Khudanpur, and James Glass. 2016. "Highway Long Short-term Memory RNNs for Distant Speech Recognition." In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing,* 5755–5759. IEEE.

Zhang, Zhengyou. 2000. "A Flexible New Technique for Camera Calibration." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22 (11): 1330–1334.

Zhao, Hang, Chuang Gan, Andrew Rouditchenko, Carl Vondrick, Josh McDermott, and Antonio Torralba. 2018. "The Sound of Pixels." In *Proceedings of the European Conference on Computer Vision,* 570–586.

Zhao, Hengshuang, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. 2017. "Pyramid Scene Parsing Network." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* 2881–2890.

Zhao, Rui, Haider Ali, and Patrick Van der Smagt. 2017. "Two-stream RNN/CNN for Action Recognition in 3D Bideos." In *Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems,* 4260–4267. IEEE.

Zhou, Yang, Xintong Han, Eli Shechtman, Jose Echevarria, Evangelos Kalogerakis, and Dingzeyu Li. 2020. "Makelttalk: Speaker-aware Talking-head Animation." *ACM Transactions on Graphics* 39 (6): 1–15.

Zhu, Xudong, and Kin Fun Li. 2016. "Real-time Motion Capture: An Overview." In *Proceedings of the 2016 10th International Conference on Complex, Intelligent, and Software Intensive Systems,* 522–525. IEEE.