KEYFRAME EXTRACTION USING LINEAR ROTATION INVARIANT COORDINATES

ROTASYONDAN BAĞIMSIZ DOĞRUSAL KOORDİNATLAR KULLANILARAK ANAHTAR KARE ÇIKARIMI

HASAN MUTLU

ASST. PROF. DR. UFUK ÇELİKCAN

Supervisor

Submitted to

Graduate School of Science and Engineering of Hacettepe University

as a Partial Fulfillment to the Requirements

for the Award of the Degree of Master of Science

in Computer Engineering

April 2022

ABSTRACT

KEYFRAME EXTRACTION USING LINEAR ROTATION INVARIANT COORDINATES

Hasan MUTLU

Master of Science , Computer Engineering

Supervisor: Asst. Prof. Dr. Ufuk ÇELİKCAN

April 2022, 63 pages

Today, with the improvements in the processing power of video cards, SOC hardware, and smartphones, the use of 3D motion data has expanded considerably beyond video games. At the same time, through these developments, the use of computer animation also increased along with the rapid progress in areas such as augmented reality, virtual reality, and video editing software. Keyframe extraction is a widely applied remedy for issues faced with 3D motion capture based computer animation. In this work, we propose a novel keyframe extraction method. In this method, firstly the skeletal motion is represented in linear rotation invariant (LRI) coordinates. This representation creates a mesh with joint positions of the related frame in the skeletal motion and then applies the transformation of the LRI coordinates. Afterwards, by performing dimension reduction using PCA, the dimensions covering 95% of the data are automatically selected and the summary data is thus acquired. Then, by applying K-means classification, the summary data is divided into clusters and a keyframe is extracted from each cluster using the cosine similarity measure. To validate the results of our proposed method, we conducted an online user study. The results of the study show that 45% of the participants preferred the keyframes extracted using our LRI-based method, surpassing the alternative by 6%.

Keywords: Keyframe extraction, Pose extraction, Pose proposal for meshes, Pose recommendation,

ÖZET

ROTASYONDAN BAĞIMSIZ DOĞRUSAL KOORDİNATLAR KULLANILARAK ANAHTAR KARE ÇIKARIMI

Hasan MUTLU

Yüksek Lisans, Bilgisayar Mühendisliği

Danışman: Asst. Prof. Dr. Ufuk ÇELİKCAN

Nisan 2022, 63 sayfa

Günümüzde ekran kartlarının, akıllı telefonların ve gömülü entegre donanımlarının işlem güçlerinin gelişmesi ile birlikte, 3B uygulamaların kullanım alanlarıda sadece oyunlarla sınırlı kalmayarak artış göstermiştir. Bu artışın yanı sıra artırılmış gerçeklik, sanal gerçeklik ve video düzenleme yazılımları gibi alanlardaki hızı ilerlemeyle birlikte bilgisayar animasyonlarının bu alanlardaki kullanımıda artmıştır. Anahtar kare çıkarımı yöntemleri 3B hareket yakalama tabanlı bilgisayar animasyonlarında sıkça karşılaşılan sorunların çözümünde sıkça kullanılmaktadır. Bu çalışmada, yeni bir anahtar kare çıkarımı yöntemi önermekteyiz. Bu yöntem ile iskelet animasyonundaki her bir kare bir 3B şekil ile temsil edilerek, bu şekil üzerinde rotasyondan bağımsız koordinat sistemi dönüşümü uygulanmaktadır. Bu dönüşümden sonra ise temel bileşen analizi uygulanarak, animasyonun en az %95 lik kısmını temsil eden bileşenler dinamik olarak seçilip, özet kare bilgisi elde ediliyor. Daha sonrasında bu özet kare bilgilerini K-means algoritması uygulayarak kümelere ayırıp, kosinüs benzerliği metodu ile her bir kümeden bir tane anahtar kare çıkarımı gerçekleştiriyoruz. Sonuçlarımızın doğrulanması için ise hazırladığımız anket web sitesinden faydalanarak karşılaştırmamızı yapıyoruz. Çalışma sonuçlarına göre önerdiğimiz

yöntemin katılımcıların %45 i tarafından seçildiği gözlenmiş olup, alternatif yönteme göre %6 daha fazla tercih edildiği görülmüştür.

Keywords: Anahtar kare çıkarımı, Poz çıkarımı, 3D modeller için poz önermesi, Poz önermesi,

ACKNOWLEDGEMENTS

First and foremost, I would like to thank to my supervisor Asst. Prof. Dr. Ufuk Çelikcan for giving me the opportunity to work together since my undergraduate years.

Furthermore, I would like to thank my thesis committee members Prof.Dr. Haşmet Gürçay, Asst. Prof. Dr. Serdar ARITAN, Assoc. Prof. Burkay GENÇ and Assoc. Prof. Elif SÜRER for reviewing my thesis and providing their valuable comments.

I deeply thank my parents and my wife; Adnan, Hülya and Gizem for always carrying me forward with their support throughout my life.

I am sincerely grateful to my brothers; Emre, Mecit for their support in my entire life.

Finally, I would also thank to PhD. Hasan TONBUL for his constant support.

CONTENTS

Page

TABLES

FIGURES

ABBREVIATIONS

LRI	:	**L**ocal **R**otation **I**nvariant
PCA	:	**P**rincipal **C**omponent **A**nalysis
ASF	:	**A**cclaim **S**keleton **F**ile
AMC	:	**A**cclaim **M**otion **C**apture
BVH	:	**B**io **V**ision **H**ierarchy
UI	:	**U**ser **I**nterface
GPU	:	**G**raphics **P**rocessing **U**nit
API	:	**A**application **P**rogramming **I**nterface

1. INTRODUCTION

Computer animations are used in different areas, especially in movies and video games. Due to its widespread use, different computer animation techniques exist such as drawing of frames with graphic tablets by an artist or using 3D modeling and animation software. However, the most popular animation techniques are motion capture techniques. In motion capture technology, an actor wears a special suit that contains sensors communicating with a computer and then performs on the stage. Computer synchronizes incoming sensor data and matches skeletal joint with the sensor and saves the motion. This causes huge data sizes. At the same time, editing and transmission of motion capture data remain problematic due to large data sizes. Hence, representing motion capture data compactly continues to be a vital consideration of research.

Skeletal animation is the most effective and commonly used technique of exploiting motion capture data. Skeletal animation consists of two parts: a mesh and a hierarchical set of bones. The mesh part contains surface (skin) information of the character to be rendered, while the animation is realized with the spatio-temporal information by the latter. As skeletal animation is performed, the technique fills the gap between two keyframes with interpolation on the timeline. Although skeletal animation provides a solution to represent motions compactly, frame counts remain problematically large for processing, storing, and editing. Keyframe extraction has emerged as a widely applied solution for the challenges and issues faced in motion capture based skeletal animation.

The keyframe extraction method must be capable of sorting out significant keyframes from the others. Also, to improve the success rate of the solution, deriving the characteristics of vertices concerning both the vertex itself and its neighbor vertices is important. For these reasons, we argue that representing vertices of joints in alternative coordinate systems and processing the motion data accordingly can provide a better solution.

There are three commonly used approaches to solve the extracting keyframes problem. The first approach converts motion data into motion curves. The second approach applies

clustering algorithms and then selects keyframes. The last approach uses genetic algorithms to solve the problem.

In this thesis, we propose a novel keyframe extraction approach. This approach represents the joints in the frames with linear rotation invariant (LRI) coordinates [1], applying principal component analysis (PCA) [2] to reduce the data dimension and extract summary data of each keyframe. Then, it divides them into clusters with the K-means algorithm [3] and selects keyframes according to cosine similarity concerning adjacent keyframes. Also, we examine the performance effect of LRI transformation on our method against using regular Cartesian coordinates without LRI.

1.1. Scope Of The Thesis

This thesis mainly focuses on identifying the effect of using LRI coordinate systems on clustering based keyframe extraction methods.

1.2. Contributions

This research aims to cover these deficiencies by proposing a novel, simple and efficient approach. The main contributions of this work are thus threefold:

- We propose a novel keyframe extraction method.
- Unlike most of the previous works, we use LRI coordinate system in our work.
- Our user experiment results show that our LRI based solution surpasses the performance without LRI by 6%.

1.3. Organization

The organization of the thesis is as follows:

- Chapter 1 presents our motivation, contributions and the scope of the thesis.
- Chapter 2 provides background information
- Chapter 3 gives related works
- Chapter 4 introduces our method
- Chapter 5 demonstrates experiments and results
- Chapter 6 states the summary of the thesis and possible future directions.

2. BACKGROUND OVERVIEW

2.1. Motion Capture

Motion capture is the method of capturing a live motion and converting it to mathematically useable data by monitoring a series of critical points in space over time and combining them to create a single three-dimensional representation of the performance. In a nutshell, it is the technology that facilitates the conversion of a live performance to a digital one. The captured subject might be anything that exists and moves in the actual world; the essential points are the locations that best depict many moving components of the subject. These points should serve as pivots or link rigid sections of the subject. For a human being, for example, some of the critical locations are the joints, which serve as pivot points and link the bones. Each of these locations is designated by one or more sensors or markers that are put on the subject and act as conduits for information to the primary collecting device.

There are several methods for recording motion. Certain systems include cameras to capture several views of the performance, which are then combined to determine the location of critical spots, each of which is represented by one or more reflecting markers. Others monitor a set of sensors using electromagnetic waves or ultrasound. Additionally, mechanical systems based on connected constructions or armatures are available that use potentiometers to calculate the rotation of each link.

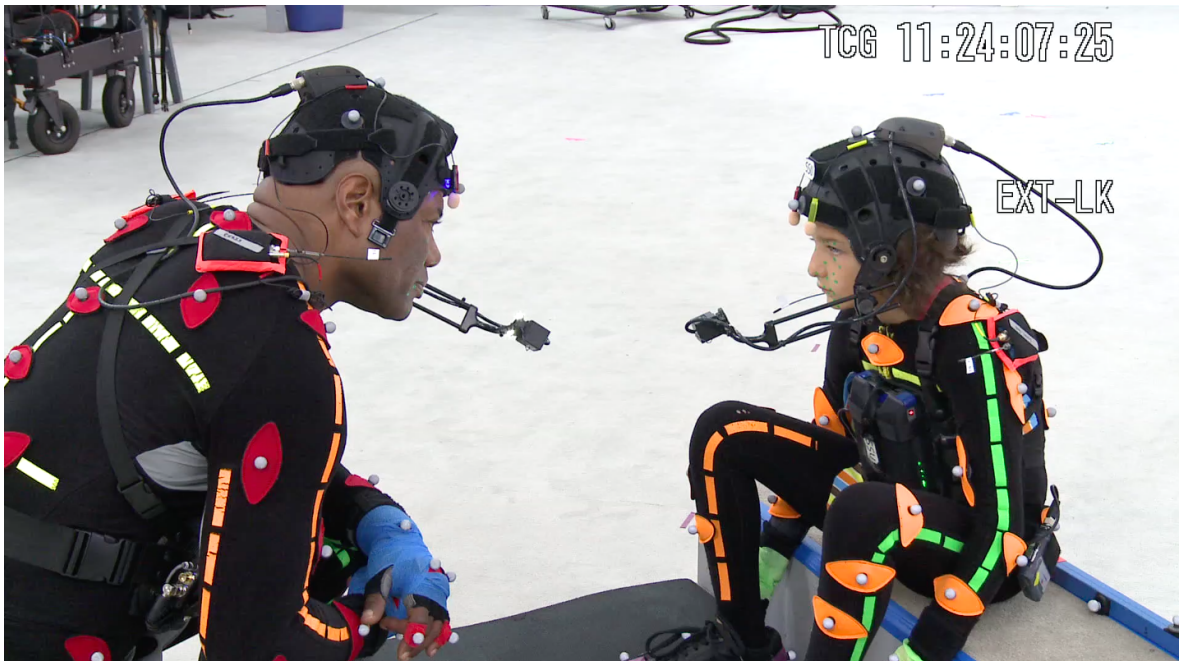


Figure 2.1 Example of a scene from the God of War game that uses the motion capture technique.

There are many animation file formats available for storing motion capture data. In this work, we used AMC and BVH file formats.

2.1.0.1. ASF / AMC Files

The ASF file is much more than a simple skeleton hierarchy definition. It provides all the data necessary to understand how the skeleton works mechanically, including units, multipliers, degrees of freedom, limitations, and documentation. The only thing missing from this file is the data itself, albeit it includes the starting posture or base position needed for character setup. AMC files are all relative to the given base location. The AMC file includes the motion data stream in its entirety. All data in this file is relative to the definitions in the ASF file, and the fields are ordered identically to the dof field of the ASF file.

2.1.0.2. BVH Files

The BVH file is another animation format developed by Biovision, a collection of optical motion capture companies specialized in sports analysis and animation. The file is organized into two distinct sections: hierarchy and motion. The hierarchy section contains all of the definitions required to construct a skeleton in an animation software application. The data stream is stored in the motion section.

2.2. OpenGL and WebGL

OpenGL (Open Graphics Library) is a graphics hardware-independent program interface. The interface consists of a variety of function calls that can be used to create sophisticated two- and three-dimensional scenarios using basic geometric primitives such as points, lines, and polygons. Additionally, there are methods for rendering the scenarios that allow for precise control of lighting, object surface attributes, transparency, anti-aliasing, and texture mapping. OpenGL is a lightweight, hardware-independent interface that can be implemented on a wide variety of graphics hardware platforms.

Modern browsers have gained a slew of sophisticated capabilities in recent years that can be accessed simply from JavaScript. Using HTML5 technologies, one can quickly build interactive components. In addition to HTML5, current browsers support WebGL. With WebGL, one can make use of GPU's processing power directly and construct high-performance 2D and 3D online applications. However, WebGL programming directly from JavaScript to construct and animate 3D scenes is an extremely sophisticated, verbose, and error-prone technique that requires understanding of OpenGL.

Figure 2.2 An example scene by rendered with Threejs

Threejs[4] is a framework that significantly simplifies this process. The framework has a lot of functions and APIs such that includes animation API, skeletal animation control functions, etc. We can utilize to create fantastic 3D scenes directly in our browser. Figure 2.2 shows an example page rendered with Threejs framework.

3. RELATED WORK

There have been a number of different approaches for keyframe extraction. These previous methods either convert the motion data into various spaces, use motion/frame data as trajectory/motion curves, apply clustering algorithms, handle a matrix factorization problem, or solve a kind of machine learning problem with a genetic algorithm.

3.1. Motion Curve Based Methods

Solutions based on trajectory or motion curves convert skeletal animation to motion curves and then apply their algorithms to these curves. After the curves are extracted from the motion data, algorithms usually apply methods such as curve simplification, saliency detection, curve fitting, PCA, or segmentation.

Miura et al. [5] combines curve-simplification and Bayesian information criterion to extract keyframes from given motion capture data. After the algorithm generates the motion curve, the method divides the curve into two segments at the point most distant from the straight line connecting the endpoints. For the calculated error between the curve and simplified line, the method uses the Bayesian information criterion to select keyframes. Bulut and Capin [6] defined a metric named curve saliency. The solution detects salient parts of the curve and uses Gaussian weighted average value distribution to select keyframes. In the method by Togawa and Okuda[7] after the joints in the animation are converted into curves, the algorithm calculates the cost value for all frames and eliminates frames accordingly. These steps repeat until the most important keyframes remain. The algorithm by Yang et al. [8] applies Butterworth filtering and PCA to the input data and then selects keyframes with zero-crossing points of velocity. The method asserted by Zhang et al. [9] creates motion curves from the amplitude of motion of joints, applies PCA and defines a distance characteristic curve, and eventually uses this curve to extract keyframes. The method proposed by Halit and Capin [10] defined a metric named 'motion saliency'. With this metric, their method analyzes the motion curve of the animation and extracts keyframes. Ik Soo Lim and Thalmann [11]

approximates the polygonal chain using a portion of the original chain's vertices through a curve simplification method. The technique begins by approximating the polygonal chain with a single straight line segment connecting its two ends. A distance criteria is used to validate this estimate, and keyframes are extracted. Matsuda et al.'s method [12] indicates a time series of the rotation angles of each axis for each joint and creates a technique dubbed interactive sequential sketching. The algorithm utilizes this stated approach to retrieve keyframes. The method proposed by Chenxu Xu et al. [13] presents a solution for extracting keyframes from motion capture data using curve fitting. Motion is represented by a series of rotation information curves in this manner. Then it identifies locations where the slope varies considerably and fits the curves using binomial to obtain segment points. The algorithm then clusters these segment points in order to retrieve keyframes. Miura et al. [14] provides a hybrid strategy that combines a curve-simplification algorithm with a principal component analysis-based initialization process.

3.2. Clustering Based Methods

The main idea of the clustering-based approaches is converting motion data into different systems, clustering these obtained data, and then implementing their proposed algorithms. While some of the solutions handle the problem as a shortest-path problem, others try to solve the problem with their similarity detection-based algorithms.

The method by Roberts et al. [15] simplifies the motion frames by around 10% while retaining most of its detail. The method considers each frame as a node in a weighted graph and calculates the weights of the graph with the perpendicular distance between joint positions in each node(frame). After these calculations, the algorithm selects the nearest N keyframes according to weights. The method suggested by Sun et al. [16] defines the inter-frame similarity metric based on a group of motion joints and uses affine propagation clustering to extract keyframes. The method by Qiang Zhang et al. [17] uses an unsupervised clustering algorithm to divide frames into two classes by similarity distances and, in the last step, uses dynamic clustering ISODATA to centralize similar frames and eliminate them. The

method developed by Xin Wang et al. [18] introduces a new extended K-means algorithm denoted by the acronym SK–means. SK-means enhances the classical K-means approach by calculating the logic similarity of each node's warping-direction energy to that of its parent node. The approach utilizes this described SK-means algorithm to extract keyframes.

3.3. Methods Based on Machine Learning

Machine learning solutions mostly use genetic algorithms to determine keyframes from the animation data. The method by Zhang et al. [19] uses a multiple-population-based genetic algorithm and defines a fitness method to meet minimizing the reconstruction error to select keyframes. The method raised by Liu et al. [20] uses genetic optimization algorithms and calculates the sparseness of the frames for determining keyframes. The method proposed by Sapinski et al. [21] presents a representation of emotional motions that are based on joint sequences. The proposed technique constructs a sequential model of emotional movement using low-level information derived from the spatial position and orientation of joints within the skeleton. This low-level derived information is used with several neural networks for selection and recognition.

3.4. Other Methods

The method by Kapadia et al. [22] encodes movements with the use of keys that reflect a variety of structural, geometric, and dynamic characteristics of human motion. Users may define sequences of key values as well as numerous combinations of key sequences to search for complicated movements. It makes optimal use of a trie-based data structure to map key sequences to movements. This established trie-based structure comprises the majority of the animation's salient keyframes and may be utilized for retrieval and extraction.

The method introduced by Jin et al. [23] focuses on determining the saliency of the frames. The method computes the saliency of each frame and selects groups from these frames. After this step, the solution uses a non-linear dimension reduction algorithm and extracts keyframes. The method proposed by Voulodimos et al. [24] creates physic-based temporal

summaries to calculate salient primitives of the motion. It determines different keyframes with these calculated salient primitives. Xia et al. [25] introduces a novel representation paradigm named joint kernel sparse representation (SR). The suggested model completes the SR using a geodesic exponential kernel in a kernel-induced space. Additionally, the solution can make use of the relationships between joints and resolves the difficulty of extracting keyframes. Choensawat et al. [26] defines an algorithm named GENLABAN, the algorithm calculates a score for each frame by analyzing body motion, body postures, and weight of the body parts. With these scores, the algorithm extract keyframes.

Matrix factorization solutions represent given skeletal animation data as matrices and solve the problem as a matrix problem. The algorithm by Huang et al. [27] provides a solution handled as a constrained matrix factorization problem with a least-squares optimization technique. This method represents the animation as matrices that contain key weights and non-keyframe weights. The algorithm uses these two matrices to extract keyframes according to user-specified error tolerance iteratively.

The method by Yang Li and Dongsheng Zhou et al. [28] eliminates joints that have a negligible effect on human posture. Then, each frame of motion data is represented as a vector in Euclidean space. Then, by computing the cosine similarity between vectors, the keyframes are extracted. The method put forward by Ming-Hwa Kim et al. [29] uses a motion analysis approach in sampling windows. The approach computes the difference between motion changes in sampling windows with and without frame skipping. According to these difference calculations, the algorithm determines the keyframes. The method by Guiyu Xia et al. [30] presents a nonconvex low-rank learning framework for learning a kernel to replace the specified kernel in the sparse subspace model in an unsupervised manner. The program divides the motion capture data into distinct subspaces and then extracts keyframes using this definition. The method proposed by Shaofan Wang et al. [31] presents an unsupervised model for learning human pose distance metrics termed sparsity locality preserving projection with adaptive neighbors (SLPPAN). The program estimates similarity values and extracts keyframes using this measure.

4. PROPOSED METHOD

Although our method uses a clustering approach, unlike other solutions, it applies LRI and PCA methods before the clustering process. Applying LRI and PCA algorithms summarizes the characteristic information of each keyframe. Also, our solution makes use of the cosine similarity measure to estimate the similarity between summarized keyframes.

Our proposed method consists of two main steps. The first step comprises the representation of skeletal motion frames in LRI local frames and dimension reduction by applying PCA. At the end of the first step, we obtain summarized data for each frame in the animation. For the second step, we divide obtained data from the first step into clusters with the K-means algorithm. Then we use cosine similarity to determine the selected keyframe for each cluster.

In the following, A = (F_1,F_2,F_3,...,F_k) defines skeletal motion where k is the keyframe count of the animation. F defines a keyframe of a skeletal motion such that $F_i = (j_1, j_2, j_3, ..., j_n), j_n \in \mathbb{R}^3$ where n is the number of joints j in the skeleton model so that F_i defines the set of joint positions for the i^{th} keyframe of the given animation. As mentioned above, our solution $O(A)$ outputs; $O(A)$ = (C_1^1,C_2^1,C_3^1,...,$C_{n_1}^2$,C_1^2,C_2^2,C_3^2,...,$C_{n_2}^2$, ... ,$C_{n_m}^m$)

where O applies LRI, PCA, and K-means algorithms, respectively over A. C defines a cluster in the result that contains summarized data for each keyframe in the same order after applied LRI conversion and PCA algorithm. m is the total number of clusters. n_i is the element count of the i^{th} cluster. Accordingly, n_m is the element count of the related extracted cluster.

After obtaining clusters from the first step, we use cosine similarity S as a measure of detecting similarity between two summarized keyframes for the clusters of summarized keyframes as follows.

$$S(X,Y) = \frac{\sum_{n=1}^{3} X_n \times Y_n}{\sqrt{\sum_{n=1}^{3} X_n^2} \times \sqrt{\sum_{n=1}^{3} Y_n^2}} \tag{1}$$

Our algorithm selects a keyframe from the obtained cluster iteratively. To accomplish that, we define two vectors for each iteration. The first vector is the difference between the candidate summarized frame data and the previous one. The second vector is the difference between the next one and the candidate summarized frame data. With these two vectors, our algorithm gathers information on the motion changes. If these vectors are similar, that means these frames are also similar. For this reason, initially, the algorithm determines the second summarized keyframe C_m^2 as the first candidate keyframe where m is the iterating cluster in the algorithm and calculates two vectors using that. The first one of these is v_1 = C_m^i - C_m^{i-1} where i is the iterating (candidate) summarized keyframe. The equation gives the difference between the candidate summarized keyframe and the previous one. The second one v_2 = C_m^{i+1} - C_m^i is the difference between next one and candidate summarized keyframe. With these two vectors, the first similarity value σ initialized by using the equation 1 above as

$$\sigma = S(v_1, v_2) \tag{2}$$

and the selected pose sp is initialized as 2.

After this initialization, σ will be updated when the new similarity in the processed iteration is less than the current value. The algorithm tries to find the keyframe that has the least similarity with the rest iteratively, as follows.

$$(sp, \sigma) = \begin{cases} sp = i, \sigma = S(V_{i-1}^c, V_{i+1}^c) & \text{if } S(V_{i-1}^c, V_{i+1}^c) \leq \sigma \\ resume & otherwise \end{cases} \tag{3}$$

In this equation, V_{i-1}^c defines the vector difference between the current summarized keyframe in the iteration and the previous one in the cluster c. Similarly, V_{i+1}^c defines the vector difference between the current summarized keyframe in the iteration and the next one in the related cluster c.

4.1. Representing Animation as LRI Local Frames

As a representation, LRI defines a separate local frame for each vertex, where the discrete forms encode the relationship and change between adjacent local frames. A local frame contains all characteristic properties of the vertex it belongs to and encodes properties relative to the neighboring vertices.

LRI defines two discrete forms. The first discrete form is for the projections of the neighboring vertices into the tangent plane of the vertex. It also denotes lengths of the projected edges on the tangent plane and signed angles between every two adjacent projected edges. The first discrete form provides invariability for positions of vertices, but it lacks information in the normal direction of neighboring vertices. For this reason, LRI also provides a second discrete form. The second discrete form can be considered as a function that defines height distances from vertex to tangent plane. LRI calculates unit vectors as the differences of these discrete forms of neighbor vertices. In the last step, LRI uses only the coefficients of this calculation to represent meshes.

The critical feature of the LRI representation is that the vertices of a given mesh are represented in relative coordinates using these specified local frames. Since this relative definition contains no global information on the mesh's location or orientation, it also ensures invariance under rigid transformations.

For the discrete equations, vertices are denoted by x^i, their corresponding positions in $\mathbb{R}^3$ are denoted by $\hat{x}^i$. The edge towards the k^{th} neighbor of i is x^i_k. Mesh edges in $\mathbb{R}^3$ are denoted by $\hat{x}^i_k$, and their projection onto the tangent plane by $\tilde{x}^i_k$. Each vertex and their neighboring vertices are parameterized as U_i and triangles are denoted by $\triangle^i_k$ for defined set U_i consisting of vertices x^i, x^i_k and x^i_{k+1}. The first discrete form uses the standard inner product of triangles corresponding in the tangent plane T_iM. Let $\mu = \mu_1 x^i_k + \mu_2 x^i_{k+1}$ be a vector in $\triangle^i_k$. Here, μ_1 and μ_2 are the vector components of the defined triangle. According to this equation, μ becomes the diagonal vector of the triangle.

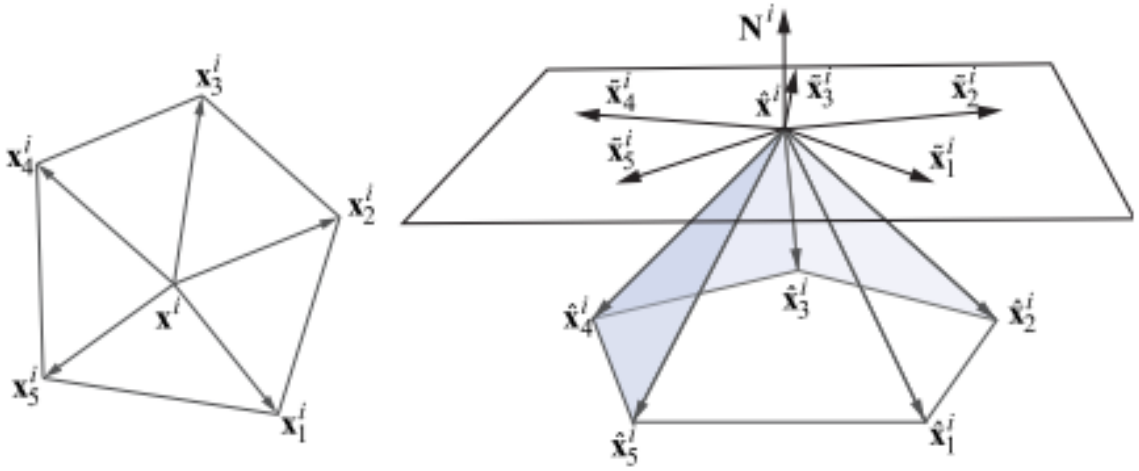


Figure 4.1 Representing 1-ring neighborhood mesh and tangent plane. This figure shown is taken from Lipman et al.'s work [1].

The first discrete form equation is given as

$$\tilde{I}(.) : \bigcup_{k1}^{d_i-1} \triangle_k^i \to \mathbb{R}. \tag{4}$$

where

$$\tilde{I}(\mu) = \langle \mu, \mu \rangle_{\mathbb{R}^3} = \langle \mu_1 \tilde{x}_k^i + \mu_2 \tilde{x}_{k+1}^i, \mu_1 \tilde{x}_k^i + \mu_2 \tilde{x}_{k+1}^i \rangle_{\mathbb{R}^3} = \\ \mu_1^2 \tilde{g}_{k,k}^i + 2\mu_1\mu_2 \tilde{g}_{k,k+1}^i + \mu_2^2 \tilde{g}_{k+1,k+1}^i \tag{5}$$

and the second discrete form equation is given as

$$\tilde{I}I(.) : \bigcup_{k1}^{d_i-1} \triangle_k^i \to \mathbb{R}. \tag{6}$$

where

$$\tilde{II}^i(\mu) := \mu_1 \langle \hat{x}_k^i, N^i \rangle_{\mathbb{R}^3} + \mu_2 \langle \hat{x}_{k+1}^i, N^i \rangle_{\mathbb{R}^3} = \mu_1 \tilde{L}_k^i + \mu_2 \tilde{L}_{k+1}^i \quad (7)$$

in which the coefficients $\tilde{L} = \langle \tilde{x}_k^i, N^i \rangle_{\mathbb{R}^3}$ and N^i is the normal of the vertex i in the tangent plane. LRI defines the local frame with a triplet b_1^i, b_2^i, N^i using these two discrete forms, where $b_1^i \in T_i M$ is a unit vector parallel to $\tilde{x}_1^i$, b_2^i is a unit vector orthogonal to $\tilde{x}_1^i$ and δ is the difference operator on the discrete frame vectors:

$$\delta_j(b_1^i) = b_1^j - b_1^i$$
$$\delta_j(b_2^i) = b_2^j - b_2^i$$
$$\delta_j(N^i) = N^j - N^i$$

Finally, the discrete local frame equations

$$\delta_j(b_1^i) = \Gamma_{j,1}^{i,1} b_1^i + \Gamma_{j,1}^{i,2} b_2^i + A_{j,1}^1 N^i$$
$$\delta_j(b_2^i) = \Gamma_{j,2}^{i,1} b_1^i + \Gamma_{j,2}^{i,2} b_2^i + A_{j,2}^1 N^i$$
$$\delta_j(N^i) = \Gamma_{j,3}^{i,1} b_1^i + \Gamma_{j,3}^{i,2} b_2^i + A_{j,3}^1 N^i$$

As previously stated, LRI representation defines local frames that filter out global positions and rotations from the mesh. We use these local frames in our solution. Since the LRI method is defined for meshes, in the first step of our solution, we transform skeleton data for each frame into a 1-ring neighborhood mesh. We assume that each joint position of the skeleton data is a vertex of a 1-ring neighborhood mesh (see Figure 4.2). Then, we apply LRI to this assumed mesh to extract LRI local frames for each vertex. LRI local frames are

a matrix that consists of 9 values and encode characteristic properties of the related vertex and the relation between neighborhood vertices. We construct a matrix whose dimension is 9 times the total number of employed joints for each keyframe. Although the extracted local frames are enough to detect similarity between adjacent vertices, we apply dimension reduction by PCA to all keyframe matrices. This way, PCA provides to eliminates the sparse density of the matrices, improving the performance of selecting keyframes computing and obtaining more meaningful data (Figure 4.3)

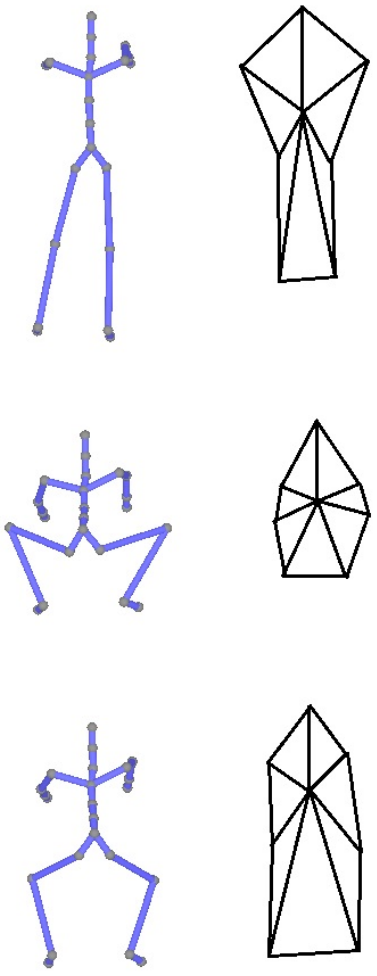


Figure 4.2 An example of representing skeletal data as a 1-ring neighborhood mesh for 3 frame.

In the dimension reduction step, instead of representing LRI data in a fixed number of dimensions, our algorithm use representations of dynamically changing dimensions. This

implies that the dimensionality adapts to the given animation. This is carried out according to principles of the PCA method on the condition that the sum of them explains the given LRI data with at least 95% accuracy. Our tests show that between 4 and 12 dimensions are sufficient to explain LRI data with at least 95% accuracy, in general. After these steps, the processed data can be used for extraction.

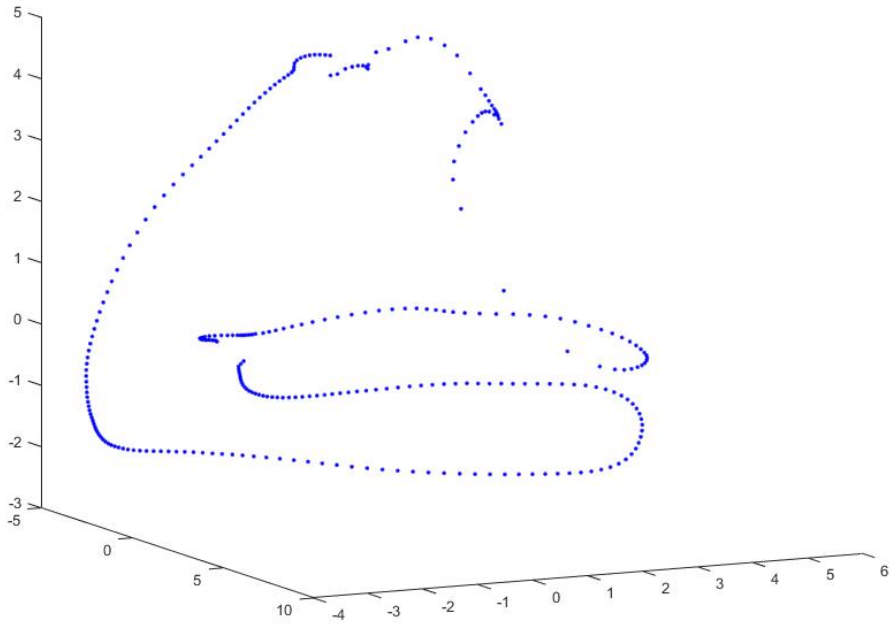


Figure 4.3 Motion data representation after LRI and PCA steps are applied. The graph shows the distribution of the frames in the motion in 3D space.

4.2. Clustering and Extracting Keyframes

Our approach uses the K-means classification algorithm for clustering and cosine similarity to measure similarity between keyframes. Firstly, we apply the K-means clustering algorithm to the dimensionally reduced data (Figure 4.3 and Figure 4.4).

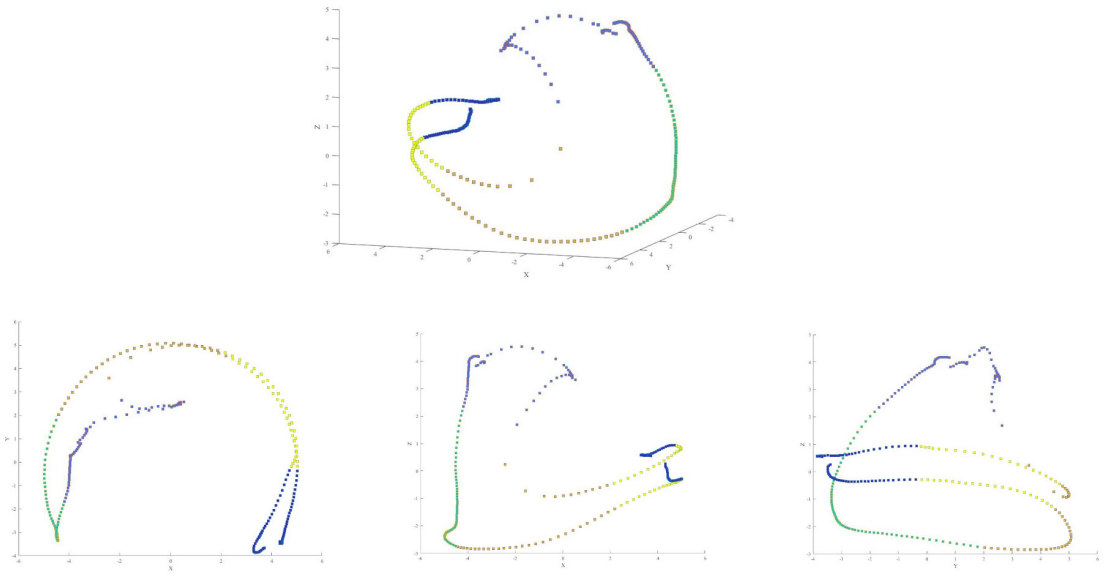


Figure 4.4 Clustering result of the motion after K-Means is applied. This graph contains the distribution of the motion as divided into 5 clusters. Each color (yellow, blue, gold, green, and purple) in the graph represents a distinct cluster. Accordingly, one keyframe for each cluster will be extracted.

Our method dynamically clusters up to the desired number of keyframes and calculates the cosine similarity between sequential candidate keyframe changes. In the initial state of the algorithm, we assume the second frame is the selected keyframe in the related cluster, and calculate the initial similarity value according to difference vectors between the second frame and neighboring frames in that first and third frame. Thus, our method selects a keyframe that has the minimum similarity value relative to the rest of the cluster values for each cluster. As a result, the most different keyframes are selected. Figure 4.5 shows examples of the result. Each row illustrates a set of 5 keyframes extracted from kick, punch, and knee to elbows motions, respectively.

Figure 4.5 Sample sets of keyframes extracted using our method.

4.3. Development Environment

In this study, We prefer to use the C++ programming language and MATLAB for implementing our solution. In our solution, we generate a MEX file that provides us to use MATLAB functions from the C++ environment. This MEX file contains a method that takes LRI data as input and applies PCA and K-Means implementation. After converting animation data to LRI coordinate system and getting the result from the MEX function, we

calculate similarities for each cluster. After these steps, our solution selects keyframes that have the least similarity value.

We also use the QT framework for desktop application. QT is a framework written in C++ and provides users to develop cross-platform applications easily. Also, QT supports OpenGL as a built-in component. Figure 4.6 denotes desktop application of our solution.

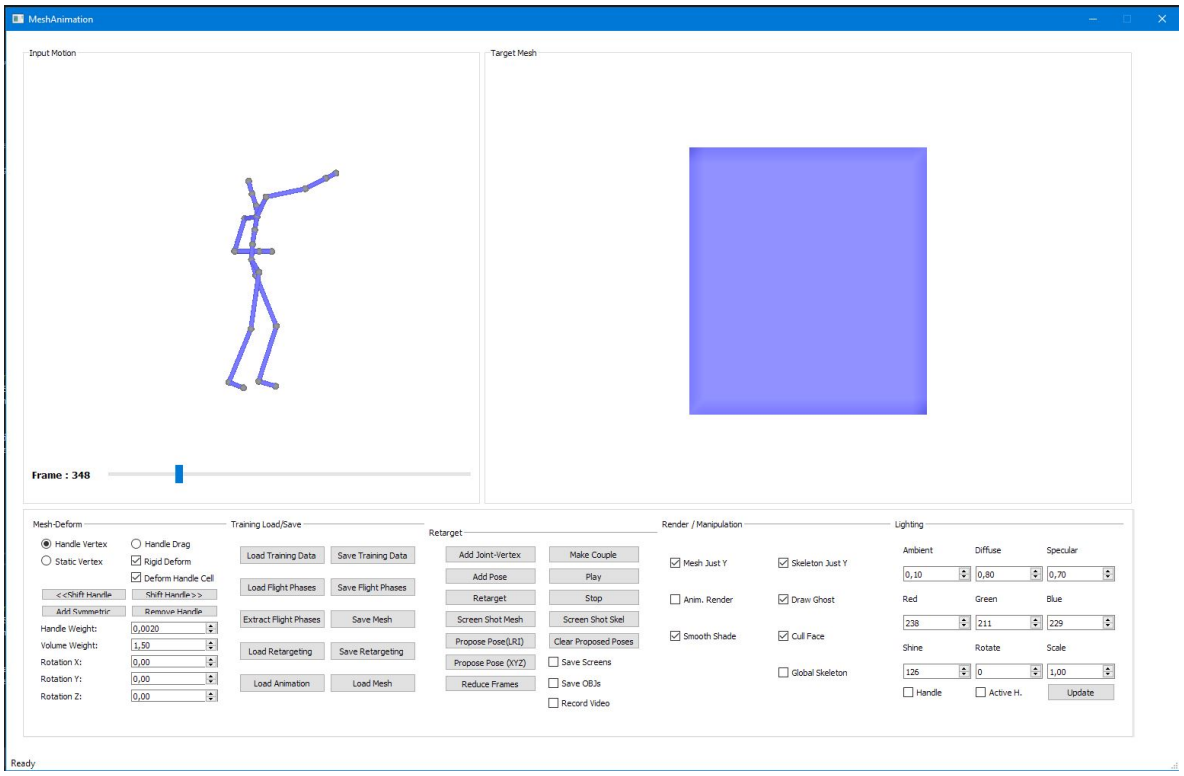


Figure 4.6 Screenshot of the developed desktop application

As shown in Figure 4.6, the user can extract keyframes after making the desired adjustments on this application.

4.4. Test Environment

It is generally agreed today that people find it easier to visit websites rather than using an external application and this helps website owners reach out to lots of people. For these reasons, we decided to create a website that can be visited by the participants and they can

contribute to our research easily. To accomplish that, we used the advantages of the full-stack javascript technologies for both server side and UI side.

4.4.1. Development Stage

Javascript is one of the most popular script languages nowadays. With NodeJS environment, It can be used for both the server-side and UI side and also enables us to develop scalable applications with cloud computing. In this work, we decided to use javascript technologies on both the server-side and UI sides.

For the server-side backend, we use the ExpressJS framework in the NodeJS environment. ExpressJS is a lightweight and versatile Node.js web application framework that includes a wide range of functionality for developing online apps. On the UI side of our application, the Jquery library is used to handle interactions between participants and our application. For 3D processes, the most popular WebGL framework named ThreeJS is used. ThreeJS contains lots of predefined WebGL methods so it gives us the opportunity to develop 3D applications that run on GPU based on WebGL and without performance issues easily. Figure 4.7 shows the application structure.

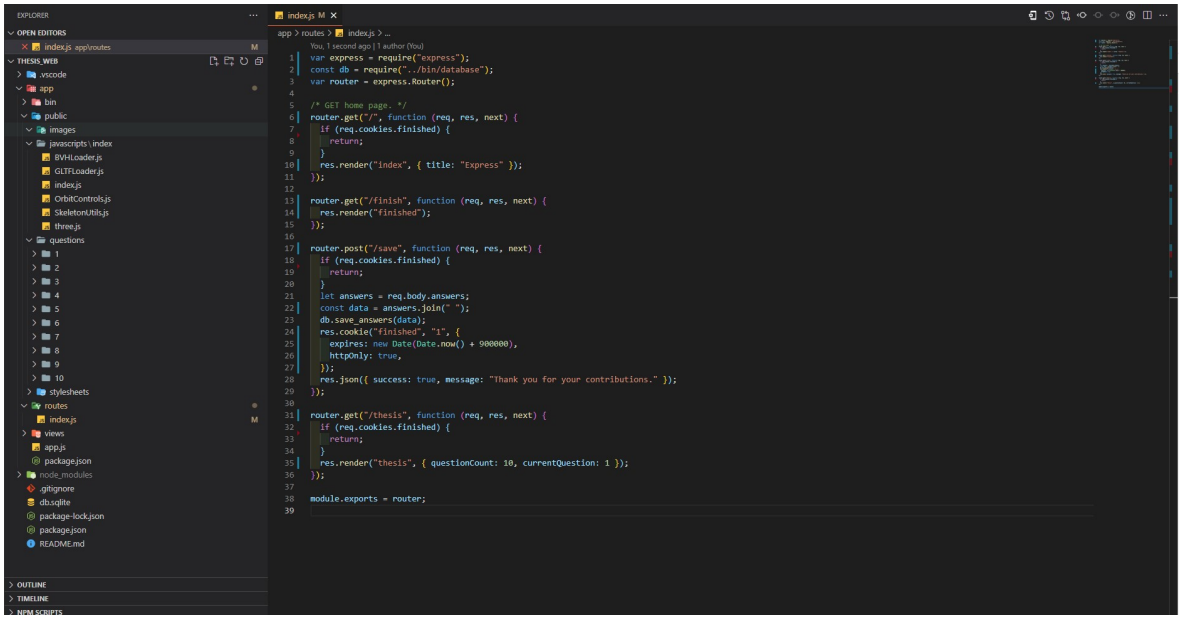


Figure 4.7 Application structure.

On the other hand, for the storage solution, we use the SQLite database. SQLite is a lightweight, standalone database. It can be used easily without any library requirement. When the participant completed the survey, the participant's answers were saved as an array in the database.

4.4.2. Publishing Stage

For publishing our application, we used a cloud provider named Heroku, which supports numerous development architectures. It has been possible to publish our application easily on Heroku with little configuration changes on our configuration file. All one needs to do is to create a project on Heroku and use the git repository provided by it. With this configuration and provided git repository, when application changes are pushed to the repository, Heroku starts the build process and then publishes our application. Besides these advantages, Heroku allows us to bind the custom domain to the application.

4.4.3. Application

The survey procedure took place as follows. When a participant visited our online survey website, they were first briefly informed of the study, and their gender and age information was collected at this step (Figure 4.8). Then, the participant started the evaluation of the results. For each motion queried, the participant initially watched the motion in a skeletal animation twice as given in Figure 4.9. Next, each set of extracted keyframes from the original motion was shown to the participant where the order of the two sets was randomized (Figure 4.10). Afterwards, the participant watched the original motion with the sets of extracted keyframes shown flanking the original motion on each side as in Figure 4.11 so that the participant could further assess the differences between the two sets of results. On this page, the participant responded by choosing either of the sets as the best representation of the original motion or 'no noticeable difference' if no significant difference was observed. This is repeated until the participant registered their responses for all 10 motions.

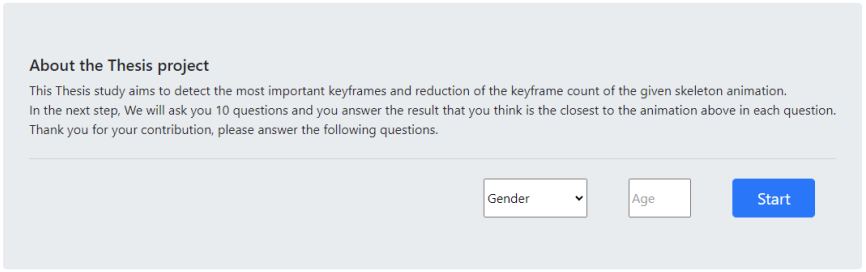


Figure 4.8 Test step where a participant is informed of the study and reports their gender and age.

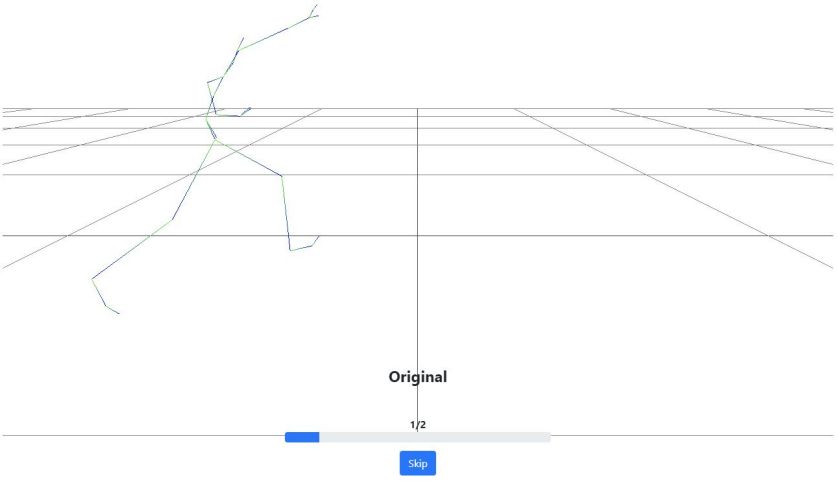


Figure 4.9 Sample preview of an original motion as shown to the participants with the online survey interface.

As seen in the Figure 4.9, the participant watches the original motion at a speed that is 20% slower than the normal speed. Also, the participant can change camera properties such as zoom in/out, rotation, and translation via mouse or touches.

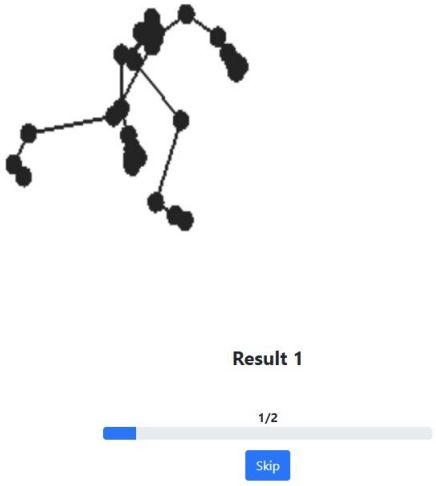


Figure 4.10 Sample instance where the participant is shown one of the extracted keyframes with the online survey interface.

The participant displays each selected keyframe for 1 second, and in total the participant watches each result for 10 seconds and 2 times.

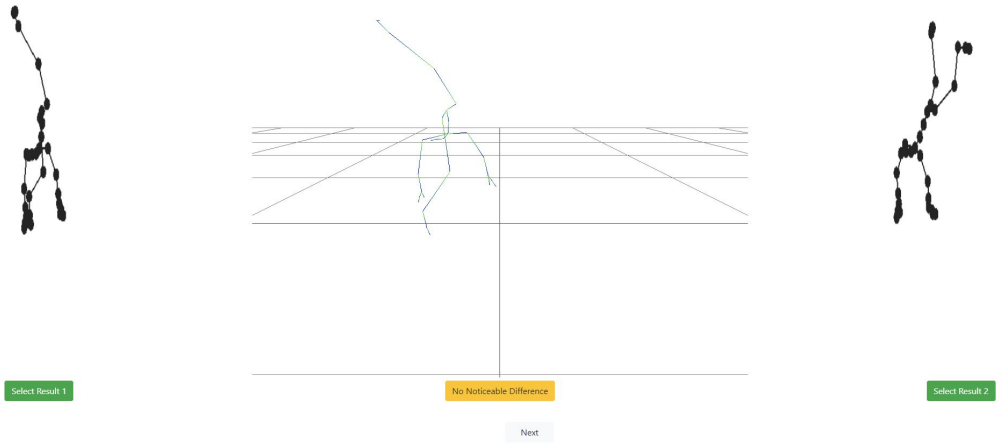


Figure 4.11 Sample preview of the online survey instance where an original motion is shown to the participant with extracted keyframes of the two alternatives shown consecutively on each side of it.

Figure 4.11 presents the screenshot of the application that shows the user the original motion and the answers simultaneously in an endless loop. In addition, the user can change the camera settings via mouse or touch and then choose the answer that seems most descriptive. After the participant decided to make a choice, click the button under the related answer indicated in green and yellow and then click the next button to answer the next question.

5. RESULTS

In this study, we used the HDM05 [32] dataset created by M. Müller et al. for testing and evaluation of our proposed method. HDM05 is a royalty-free motion capture dataset that may be used for research purposes. It features about 70 motion classes in ten to fifty realizations performed by a variety of performers. The sampling rate of all performances in the dataset is 120 Hz. For evaluation, we selected 10 relatively short motions 5.1 from the dataset and our algorithm is used to extract five keyframes from each of these selected motions. Also, table 5.1 shows the count of selected dimensions dynamically on the LRI step for each animation.

Question	Motion	Frame Count	Dimension(After PCA)
1	Cartwheel	401	9
2	Elbow to Knee	319	8
3	Jump Down	272	10
4	Jumping Jack	142	6
5	Kick	295	8
6	Lie Down	621	7
7	Punch	115	6
8	Squat	191	4
9	Throw Basketball	452	8
10	Throw ball	427	12

Table 5.1 The motions used in the experiment and their corresponding frame counts.

Furthermore, as our study results can be subjective, we prepared a website to survey subjective performance evaluations of the participants comparing the results obtained by our method using LRI representations to the ones without. The survey included the ten pre-selected motions (Table 5.1) under consideration with two sets of 5 extracted keyframes for each motion, one set including the results of our method and the other including the

results obtained using the standard Cartesian coordinate representation. Figures 5.1 and 5.2 demonstrate sample results for cartwheel and punch motions in both representations.

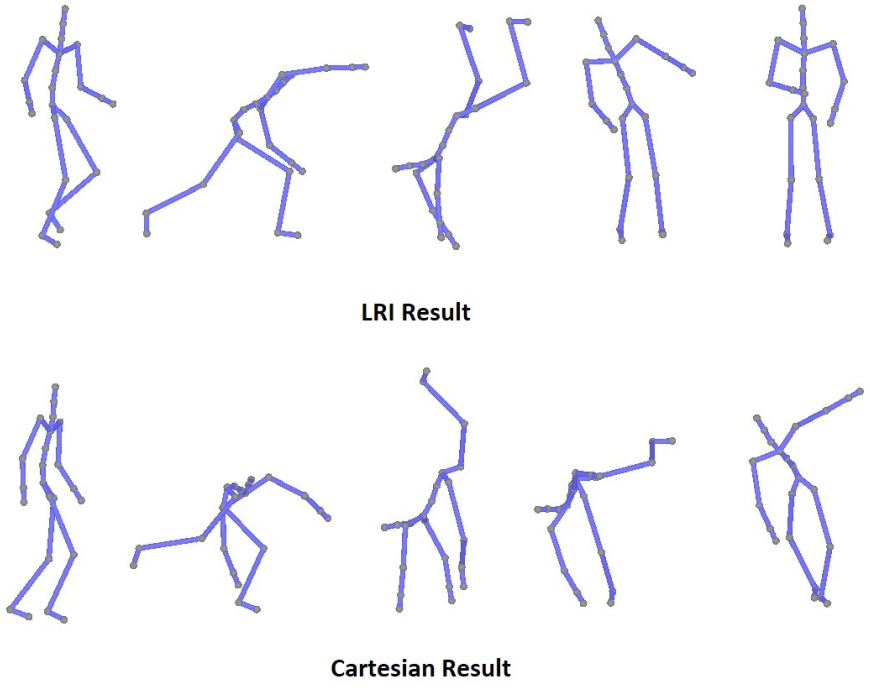


Figure 5.1 Cartwheel motion results for LRI and Cartesian.

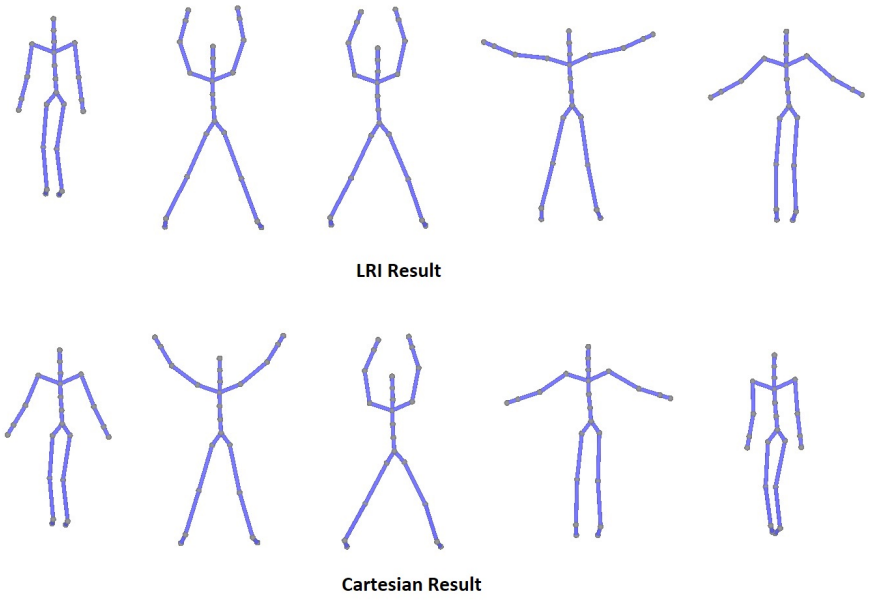


Figure 5.2 Jumping jack motion results for LRI and Cartesian.

With the survey, only the gender and age information was collected from the participants, remaining otherwise anonymous. All participants volunteered to take the online survey and none of them have been compensated in any way.

A total of 30 people, 12 female (40%) and 18 male (60%), participated in this study. The average age of the participants was 28 ± 3.14. Evaluation results are provided in Figures 5.3 and 5.4. Figure 5.3 gives participants' preferences for each motion, while Figure 5.4 gives the overall distribution of all participants' preferences.

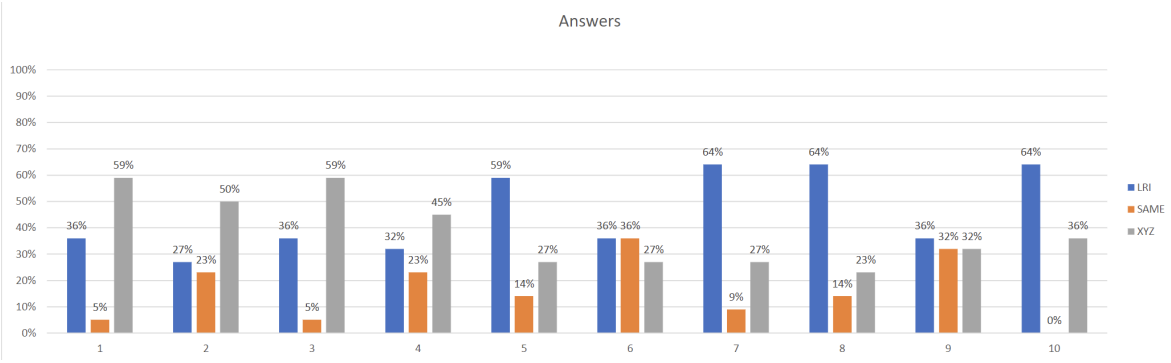


Figure 5.3 Participants' answers by each question.

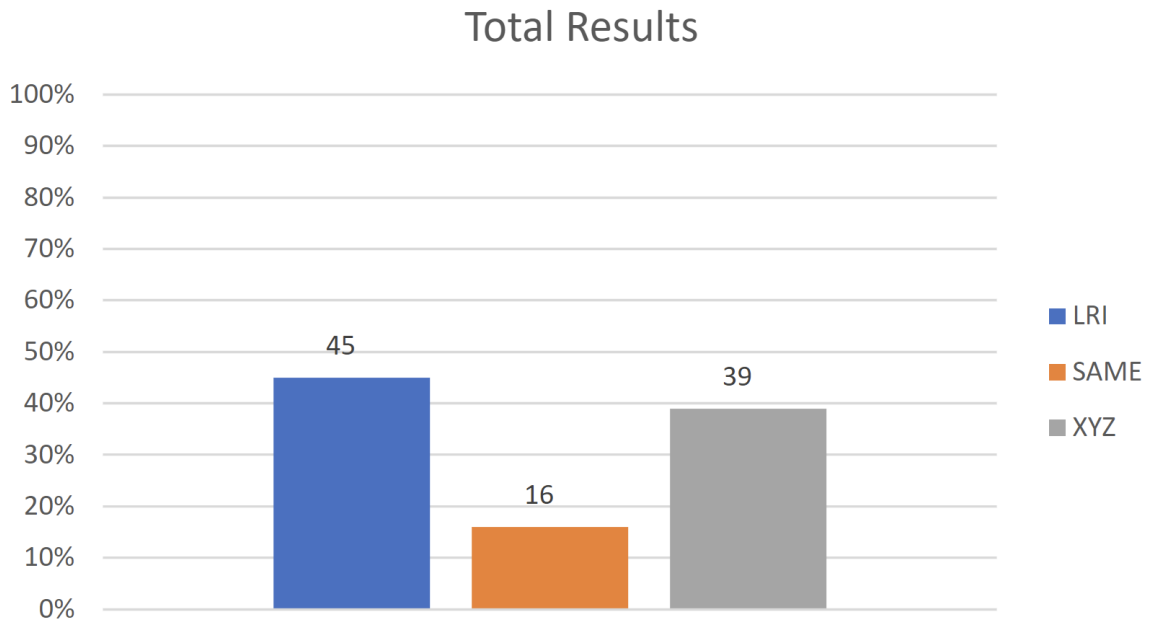


Figure 5.4 Total distribution of the answers.

The survey outcomes per motion as shown in Figure 5.3 demonstrate that the participants preferred mostly the sets of keyframes extracted using the Cartesian coordinate system representation for the first four motions under consideration. However, the keyframes obtained using the LRI representation were preferred by more participants for the remaining six motions. Figure 5.5 shows the first 4 results that contain LRI-based solution and Cartesian solution side by side, Figure 5.6 denotes the last 6 results except the 6th and 9th results. When the Figure 5.3 was examined, in the 6th and 9th answers, it was observed that the LRI method was preferred with a slight difference. Figure 5.7 indicates these motion results.

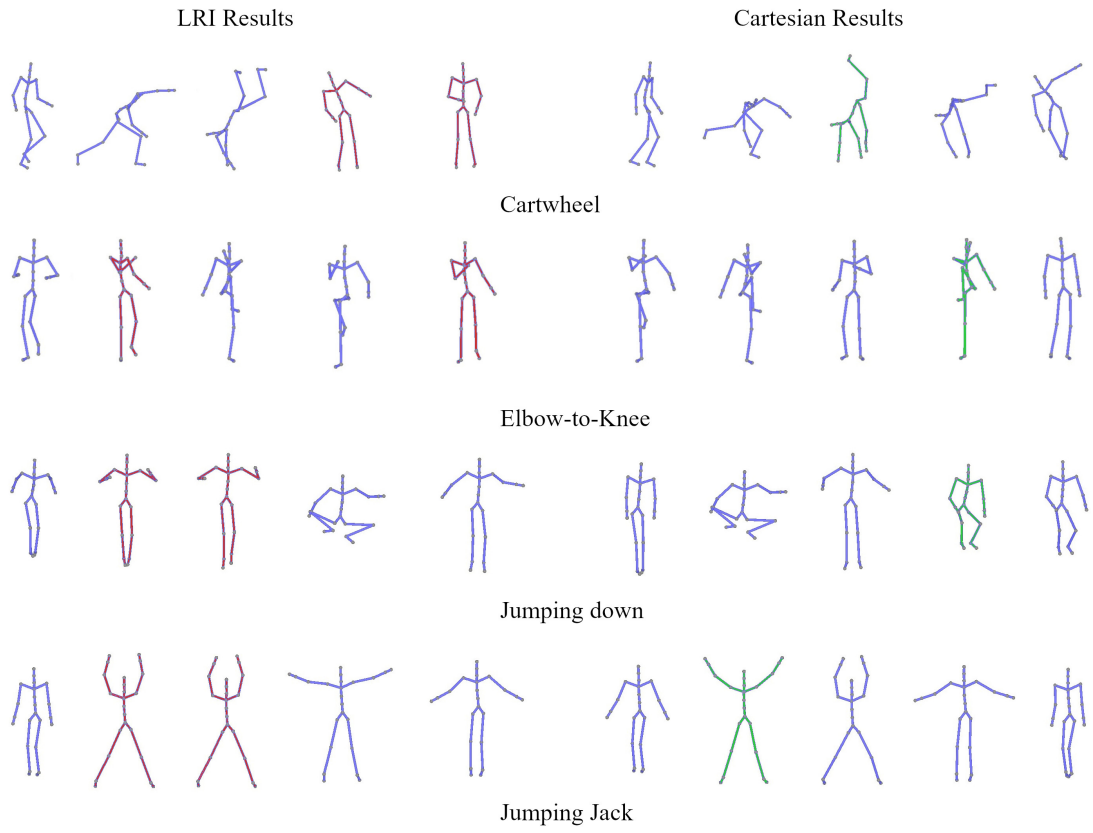


Figure 5.5 Participants' answers for the cartwheel, elbow-to-knee, jump down and jumping jack

When the first 4 results are investigated, the figure 5.5 shows that while the selected frames by Cartesian results don't break the integrity of the motion, LRI results fail to select keyframes that provide integrity of the motion. Also, we can observe that selected keyframes by the LRI method contain very similar keyframes that indicated the red color in the figure. On

the other hand, Cartesian results contain the most notable keyframes that indicated the green color in the figure.

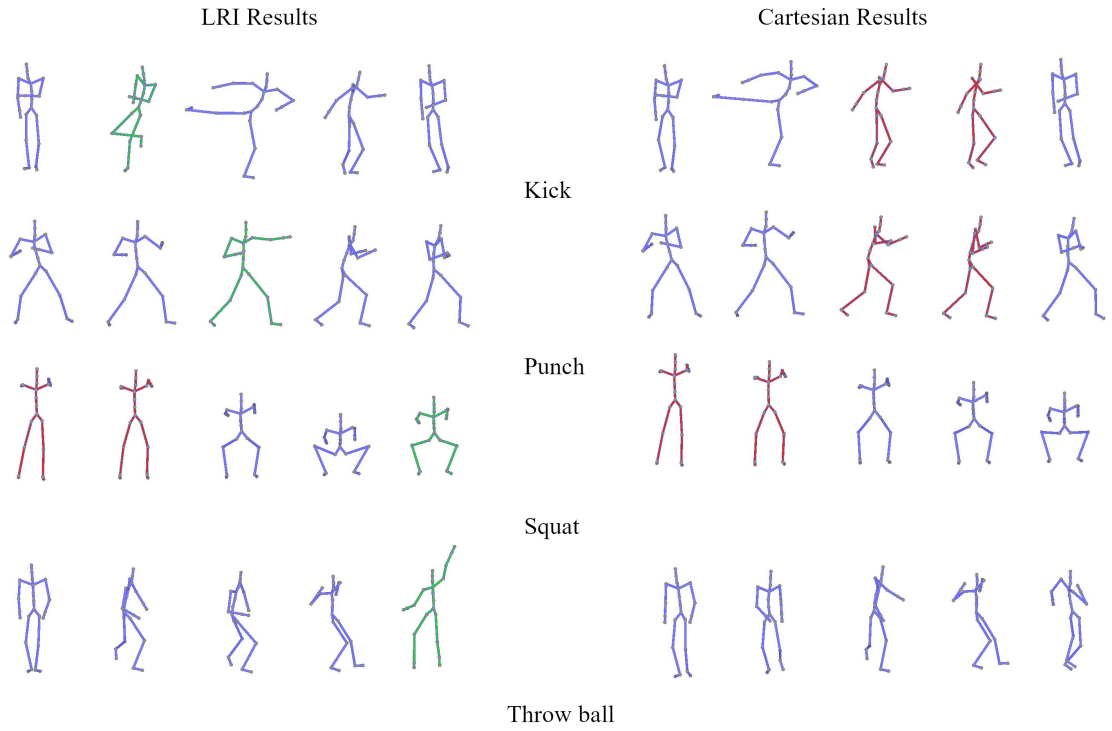


Figure 5.6 Participants' answers for the kick, punch, squat and throw ball

When the results where the LRI method was chosen the most out of the last 6 motions that kick, punch, squat and throw ball motions are examined, the figure 5.6 indicates that even similar keyframes exist for the squad motion, selected keyframes by LRI contains most different keyframes in the related motion. For example, bending the knee on the kick motion, the punch is straight, getting up from crouching and the moment that ball is thrown.

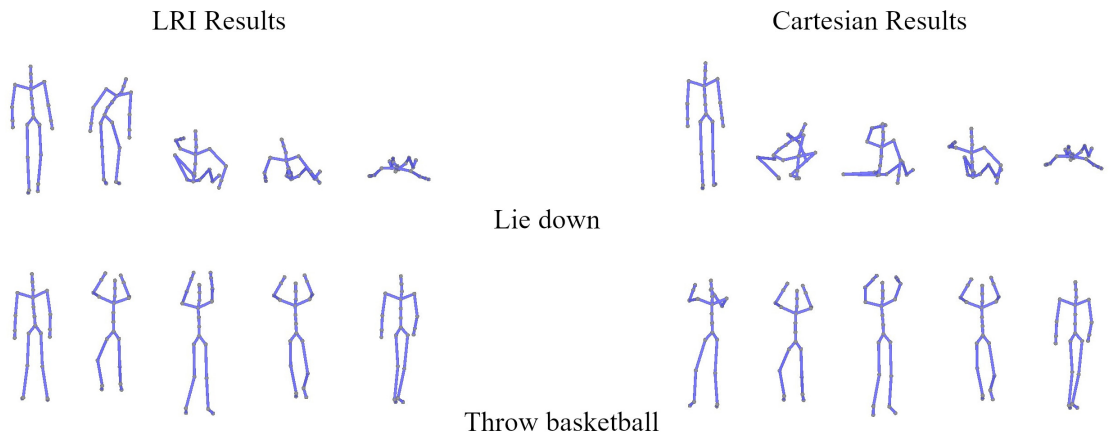


Figure 5.7 Participants' answers for the lie down and throw basketball

According to the user experiment, we can see that the LRI method is very slightly preferred for the lie-down and throw basketball motions. When we examined the figure 5.7, it shows that on the LRI keyframes the skeleton is laying on the ground step by step, in the alternative solution, this step was passed quickly. This similar situation is also valid for the last throw basketball motion.

Over the whole set of 10 motion queries, the average preference count of the LRI results by the participants was 13.6 ± 4 (45%) while the average for the alternative was 11.6 ± 3.64 (39%). Moreover, when the answers per user were examined, it was seen that each user chose an LRI for 4.53 ± 1.63 out of 10 questions on average when the average for the Cartesian was 3.87 ± 2.08. At the same time, the average number of the no difference was 1.72 ± 1.6. The aggregated preference results given in Figure 5.4 show that the participants preferred the keyframes extracted by our proposed approach more than the standard alternative by 6% in general while 16% of the votes indicated no preference.

6. DISCUSSION AND CONCLUSION

6.1. Discussion

We have investigated various types of keyframe extraction solutions in the literature. Most of the solutions use the Cartesian coordinate system without applying any transformation process. As mentioned in the literature review, While Roberts et al.'s method [15] converts given motion frames into a weighted graph and extracts keyframes according to distance between these nodes in the graph, Sun et al.'s method [16] uses affine clustering approach. In the another study proposed by Qiang Zhang et al [17], the method proposes a dynamic clustering solution. In the Xin Wang et al.'s method [18], which is last study examined in the literature, the solution extends to classify the K-Means algorithm according to similarity and extracts center frames of the clusters as keyframes.

This thesis work presented a different clustering approach rather than the previously reviewed studies. In this proposed method, unlike other proposed solutions, the method converts each frame into a mesh, applies LRI transformation (this transformation extracts characteristic information of each vertex in the mesh), uses PCA and K-Means algorithms, and then decides keyframes according to similarity values of each cluster with cosine similarity function. These steps make our work different from others and allow us to handle the keyframe extraction problem from a different perspective. Besides proposing a novel keyframe extraction method, the purpose of this thesis work was to determine the effect on performance using LRI coordinates according to the user experiment.

As mentioned before, the use of LRI brought with it several changes such as representing every frame as a mesh and using PCA to reduce high dimensions coming from LRI transformation. In the first stages of our work, we tried to use fixed-length dimensions after applying the PCA step. But, the fixed-length dimension did not outperform enough to select distinct keyframes. So, we used adaptive, dynamic dimension selection in the PCA step of the method. Our algorithm selects dimensions that the PCA result covers 95% of the motion. Our test results show that selecting dimensions between 4 and 12 is enough for motions used

in our user experiment. Selecting dimensions less or more than this range causes extracting similar keyframes in the results and does not affect the extracting keyframe performance.

Considering the data obtained as a result of this work, the results of this study indicate that our method was preferred by the participants by slightly more than 6% than the alternative method. A possible explanation for this low-performance increase might be by our mesh representation of each frame or the used similarity metrics. The 1-ring neighbor network representation used in this thesis may be insufficient to describe each frame in motion and with this mesh representation, the LRI transform may not be able to extract enough characteristic features of the vertices. For these reasons, a complete similarity-based separation could not be able to complete in the cosine similarity method. To improve or increase the effect on performance, mesh representation or similarity calculation steps can be changed or replaced with a new representation or a new similarity metric. Also, using LRI with different solutions such as motion curves, matrix factorization, or machine learning methods can help to improve extracting performance of the related solutions.

6.2. Conclusion

In this thesis, we have presented a keyframe extraction method based on LRI coordinates representation and evaluated the performance of the method against the ones based on Cartesian coordinates. Unlike other solutions, Our method represents skeletal animation data as a mesh for each frame, applies LRI transformation and PCA, uses the K-Means clustering algorithm, and then extracts keyframes with cosine similarity metric. Besides, unlike other solutions that use PCA, our method decides the dimension of the represented motion data dynamically. These differences and the user experiment show that our study results underline the potential of LRI for keyframe extraction since using our solution based on it outperforms the standard alternative with slightly better performance (6%).

This thesis was undertaken to propose a novel keyframe extraction method and show that representing animation frames using different representations, such as LRI, can provide better extraction performance for skeletal animations. For future research, it is possible to combine our LRI-based approach with deep learning methods such as using transformer networks or one-shot learning. These deep learning methods can provide better performance for the similarity detection step of our solution. Moreover, our method can use different similarity measures rather than the cosine similarity, or it can be combined with other keyframe extraction methods such as curve simplification or matrix factorization approaches towards achieving better performance.

Finally, most of the keyframe extraction methods in the literature are also used for keyframe reduction. To provide this, the methods eliminate frames according to similarity metrics without applying any clustering or another algorithm. In this context, our method can be used as a keyframe reduction solution instead of extracting the desired number of keyframes with a few changes and additions.

REFERENCES

[1] Yaron Lipman, Olga Sorkine, David Levin, and Daniel Cohen-Or. Linear rotation-invariant coordinates for meshes. *ACM Trans. Graph.*, 24(3):479–487, **2005**. ISSN 0730-0301. doi:10.1145/1073204.1073217.

[2] Andrzej Maćkiewicz and Waldemar Ratajczak. Principal components analysis (pca). *Computers & Geosciences*, 19(3):303–342, **1993**. ISSN 0098-3004. doi:https://doi.org/10.1016/0098-3004(93)90090-R.

[3] S. Lloyd. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137, **1982**. doi:10.1109/TIT.1982.1056489.

[4] Jos Dirksen. *Learn Three.js*. Packt Publishing, Livery Place, 35 Livery Street, Birmingham, B3 2PB, UK., **2018**. ISBN 978-1-78883-328-8.

[5] Takeshi Miura, Takaaki Kaiga, Naho Matsumoto, Hiroaki Katsura, Katsubumi Tajima, and Hideo Tamamoto. Application of the bayesian information criterion to keyframe extraction from motion capture data. In *SIGGRAPH Asia 2011 Posters*, SA '11. Association for Computing Machinery, New York, NY, USA, **2011**. ISBN 9781450311373. doi:10.1145/2073304.2073345.

[6] Eyuphan Bulut and Tolga Capin. Key frame extraction from motion capture data by curve saliency. *CASA*, **2007**.

[7] H. Togawa and M. Okuda. Position-based keyframe selection for human motion animation. In *11th International Conference on Parallel and Distributed Systems (ICPADS'05)*, volume 2, pages 182–185. **2005**. doi:10.1109/ICPADS.2005.239.

[8] Yang Yang, Lanling Zeng, and Howard Leung. Keyframe extraction from motion capture data for visualization. In *2016 International Conference on Virtual Reality and Visualization (ICVRV)*, pages 154–157. **2016**. doi:10.1109/ICVRV.2016.33.

[9] Qiang Zhang, Xiang Xue, Dongsheng Zhou, and Xiaopeng Wei. Motion key-frames extraction based on amplitude of distance characteristic curve. *International Journal of Computational Intelligence Systems*, 7(3):506–514, **2014**. doi:10.1080/18756891.2013.859873.

[10] Cihan Halit and Tolga Capin. Multiscale motion saliency for keyframe extraction from motion capture sequences. *Computer Animation and Virtual Worlds*, 22(1):3–14, **2011**. doi:https://doi.org/10.1002/cav.380.

[11] Ik Soo Lim and D. Thalmann. 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*, volume 2, pages 1167–1169 vol.2. **2001**. doi:10.1109/IEMBS.2001.1020399.

[12] Koichi Matsuda and Kunio Kondo. Keyframes extraction method for motion capture data. *Journal for Geometry and Graphics Volume*, 8:81–90, **2004**.

[13] Chenxu Xu, Wenjie Yu, Yanran Li, Xuequan Lu, Meili Wang, and Xiaosong Yang. Keyframe extraction for human motion capture data via multiple binomial fitting. *Computer Animation and Virtual Worlds*, 32(1):e1976, **2021**. doi:https://doi.org/10.1002/cav.1976.

[14] Takeshi Miura, Takaaki Kaiga, Takeshi Shibata, Hiroaki Katsura, Katsubumi Tajima, and Hideo Tamamoto. A hybrid approach to keyframe extraction from motion capture data using curve simplification and principal component analysis. *IEEJ Transactions on Electrical and Electronic Engineering*, 9(6):697–699, **2014**. doi:https://doi.org/10.1002/tee.22029.

[15] Richard Roberts, J. P. Lewis, Ken Anjyo, Jaewoo Seo, and Yeongho Seol. Optimal and interactive keyframe selection for motion capture. In *SIGGRAPH Asia 2018 Technical Briefs*, SA '18. Association for Computing Machinery, New York, NY, USA, **2018**. ISBN 9781450360623. doi:10.1145/3283254.3283256.

[16] Bin Sun, Dehui Kong, Shaofan Wang, and Jinghua Li. Keyframe extraction for human motion capture data based on affinity propagation. In *2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, pages 107–112. **2018**. doi:10.1109/IEMCON.2018.8614862.

[17] Qiang Zhang, Shao-Pei Yu, Dong-Sheng Zhou, and Xiao-Peng Wei. An efficient method of key-frame extraction based on a cluster algorithm. *Journal of Human Kinetics*, 39(1):5–14, **2013**. doi:doi:10.2478/hukin-2013-0063.

[18] Xin Wang, Liangxiu Chen, Jiali Jing, and Herong Zheng. Human motion capture data retrieval based on semantic thumbnail. *Multimedia Tools and Applications*, 75(19):11723–11740, **2016**. ISSN 1573-7721. doi:10.1007/s11042-015-2705-3.

[19] Qiang Zhang, Shulu Zhang, and Dongsheng Zhou. Keyframe extraction from human motion capture data based on a multiple population genetic algorithm. *Symmetry*, 6:926–937, **2014**. doi:10.3390/sym6040926.

[20] Xian-mei Liu, Ai-min Hao, and Dan Zhao. Optimization-based key frame extraction for motion capture animation. *The Visual Computer*, 29, **2012**. doi:10.1007/s00371-012-0676-1.

[21] Tomasz Sapiński, Dorota Kamińska, Adam Pelikant, and Gholamreza Anbarjafari. Emotion recognition from skeletal movements. *Entropy*, 21(7), **2019**. ISSN 1099-4300. doi:10.3390/e21070646.

[22] Mubbasir Kapadia, I-kao Chiang, Tiju Thomas, Norman I. Badler, and Joseph T. Kider. Efficient motion retrieval in large motion databases. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, I3D '13, page 19–28. Association for Computing Machinery, New York, NY, USA, **2013**. ISBN 9781450319560. doi:10.1145/2448196.2448199.

[23] Chao Jin, Thomas Fevens, and Sudhir Mudur. Optimized keyframe extraction for 3d character animations. *Computer Animation and Virtual Worlds*, 23(6):559–568, **2012**. doi:https://doi.org/10.1002/cav.1471.

[24] Athanasios Voulodimos, Ioannis Rallis, and Nikolaos Doulamis. Physics-based keyframe selection for human motion summarization. *Multimedia Tools and Applications*, 79(5):3243–3259, **2020**. ISSN 1573-7721. doi:10.1007/s11042-018-6935-z.

[25] Guiyu Xia, Huaijiang Sun, Xiaoqing Niu, Guoqing Zhang, and Lei Feng. Keyframe extraction for human motion capture data based on joint kernel sparse representation. *IEEE Transactions on Industrial Electronics*, 64(2):1589–1599, **2017**. doi:10.1109/TIE.2016.2610946.

[26] Worawat Choensawat, Minako Nakamura, and Kozaburo Hachimura. Genlaban: A tool for generating labanotation from motion capture data. *Multimedia Tools and Applications*, 74(23):10823–10846, **2015**. ISSN 1573-7721. doi:10.1007/s11042-014-2209-6.

[27] Ke-Sen Huang, Chun-Fa Chang, Yu-Yao Hsu, and Shi-Nine Yang. Key probe: A technique for animation keyframe extraction. *The Visual Computer*, 21:532–541, **2005**. doi:10.1007/s00371-005-0316-0.

[28] Yang Li, Dongsheng Zhou, and Qiang Zhang. Key frames extraction of human motion capture data based on cosine similarity. **2017**.

[29] Ming-Hwa Kim, Lap-Pui Chau, and Wan-Chi Siu. Keyframe selection for motion capture using motion activity analysis. In *2012 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 612–615. **2012**. doi:10.1109/ISCAS.2012.6272106.

[30] Guiyu Xia, Beijia Chen, Huaijiang Sun, and Qingshan Liu. Nonconvex low-rank kernel sparse subspace learning for keyframe extraction and motion

segmentation. *IEEE Transactions on Neural Networks and Learning Systems*, 32(4):1612–1626, **2021**. doi:10.1109/TNNLS.2020.2985817.

[31] Shaofan Wang, Yongjia Xin, Dehui Kong, and Baocai Yin. Unsupervised learning of human pose distance metric via sparsity locality preserving projections. *IEEE Transactions on Multimedia*, 21(2):314–327, **2019**. doi:10.1109/TMM.2018.2859029.

[32] M. Müller, T. Röder, M. Clausen, B. Eberhardt, B. Krüger, and A. Weber. Documentation mocap database hdm05. Technical Report CG-2007-2, Universität Bonn, **2007**.

Appendix A

Result Images

This section contains screenshots of results used in user experiment. Each image contains either LRI based and Cartesian based results.

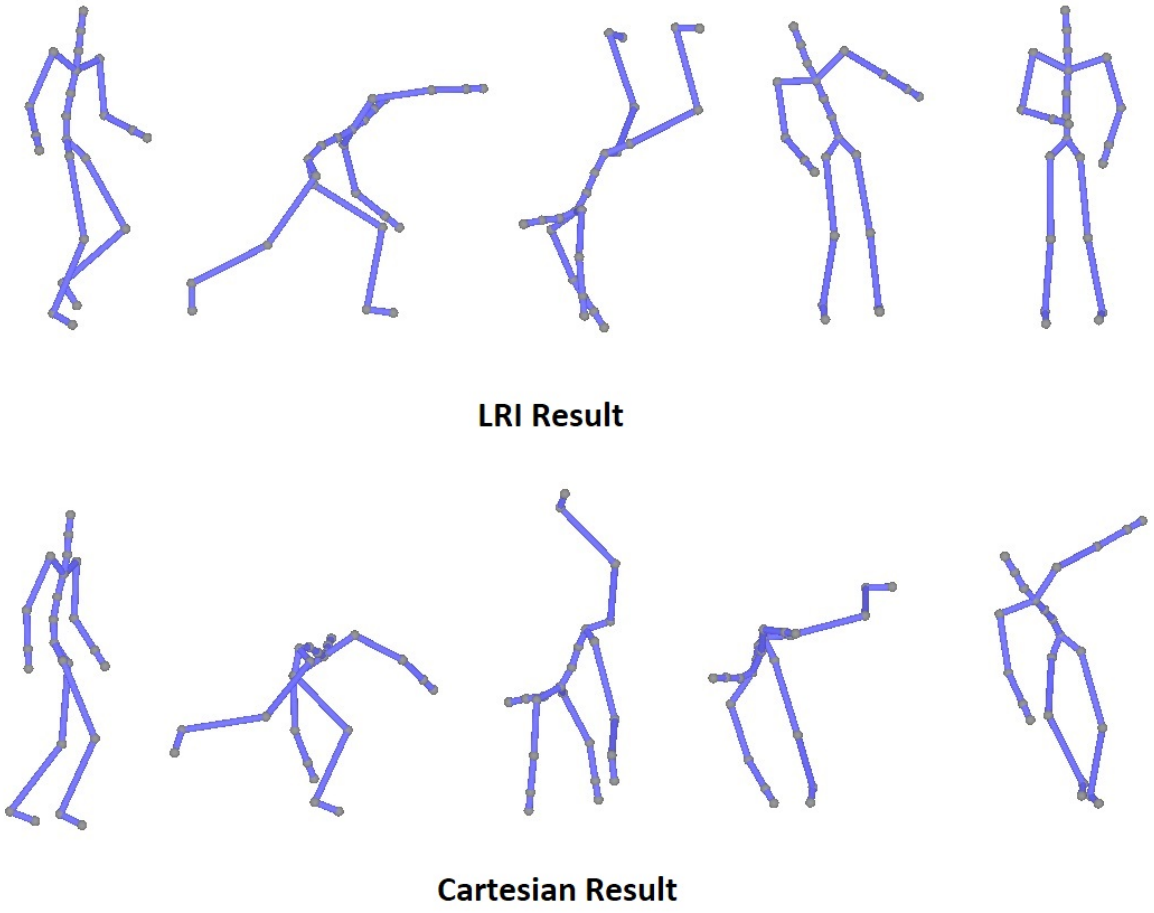
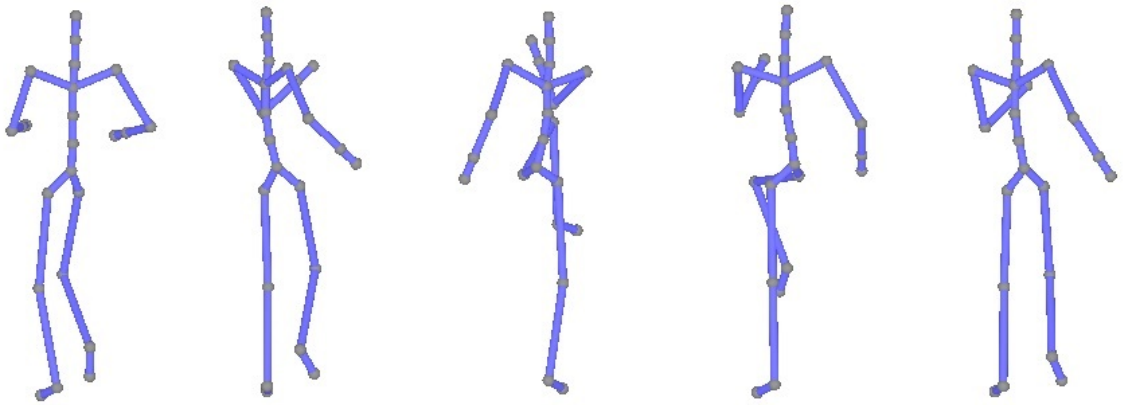
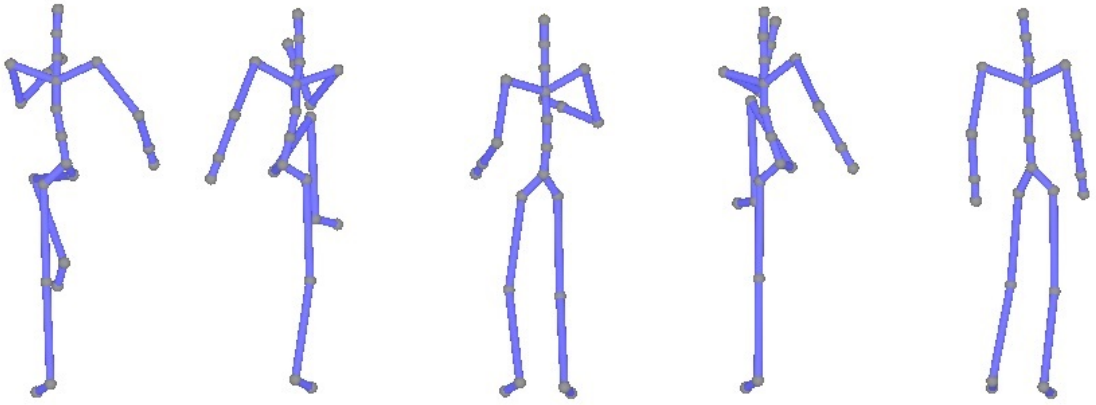


Figure A.1 Cartwheel motion result in used user experiment.

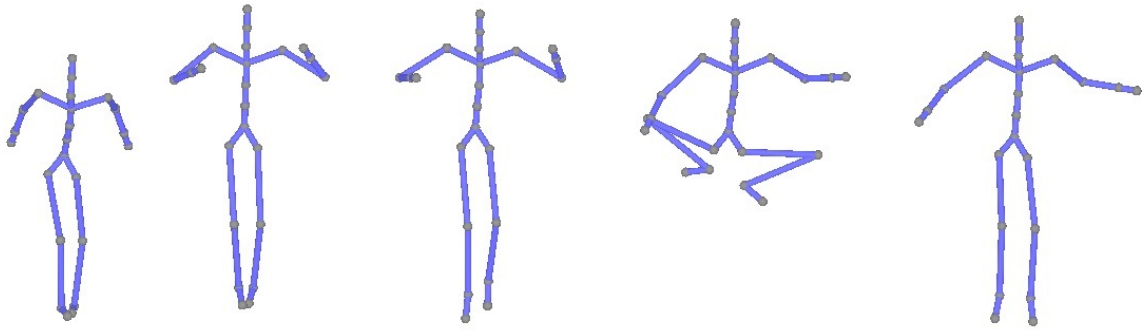


LRI Result

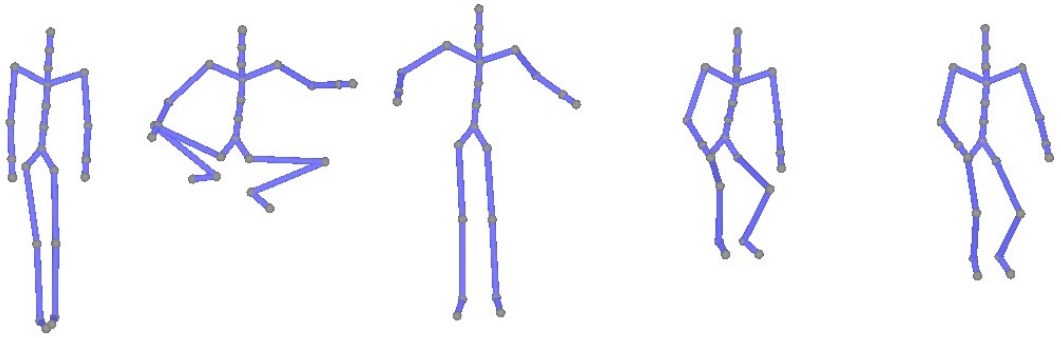


Cartesian Result

Figure A.2 Elbow-to-knee motion result in used user experiment.



LRI Result



Cartesian Result

Figure A.3 Jump down motion result in used user experiment.

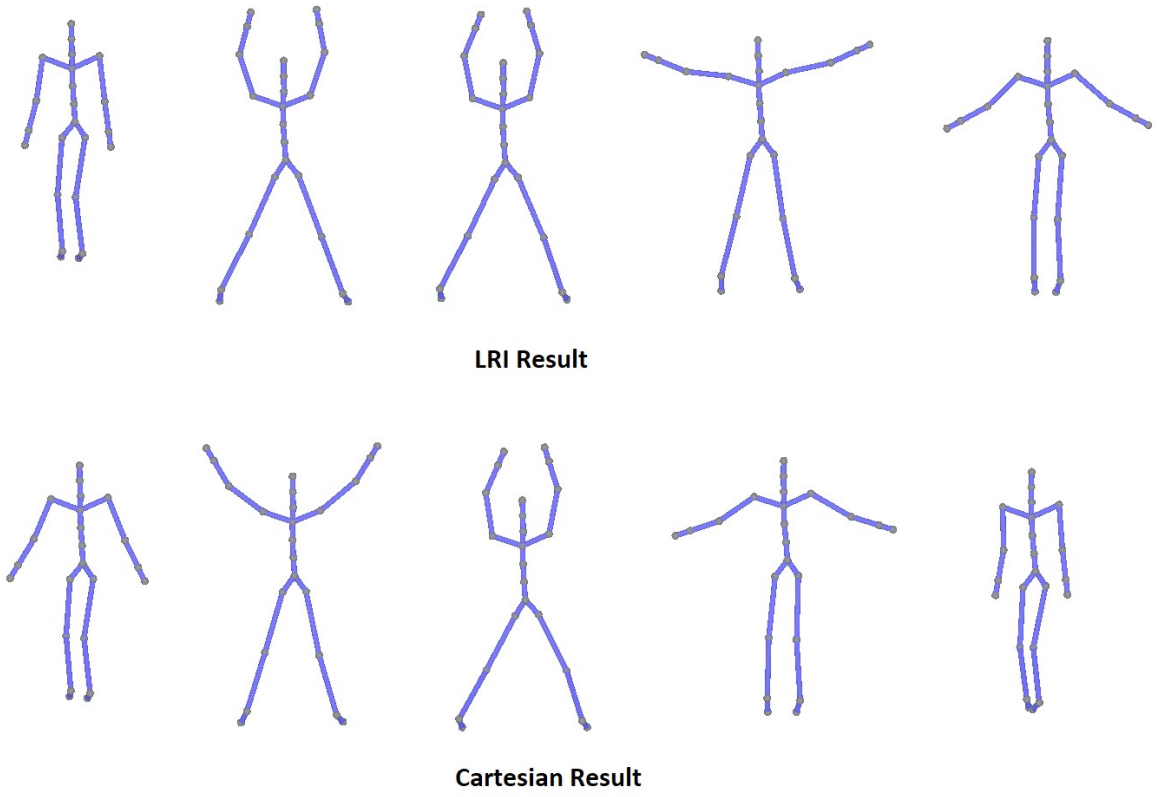


Figure A.4 Jumping jack motion result in used user experiment.

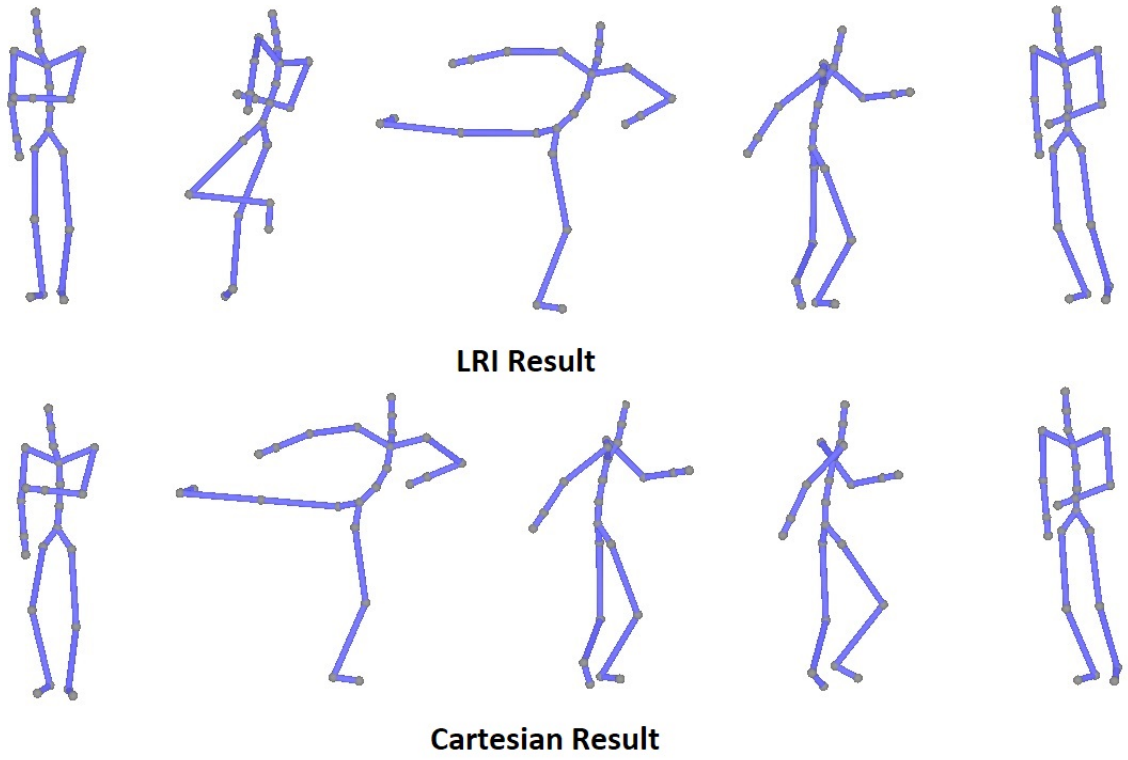


Figure A.5 Kick motion result in used user experiment.

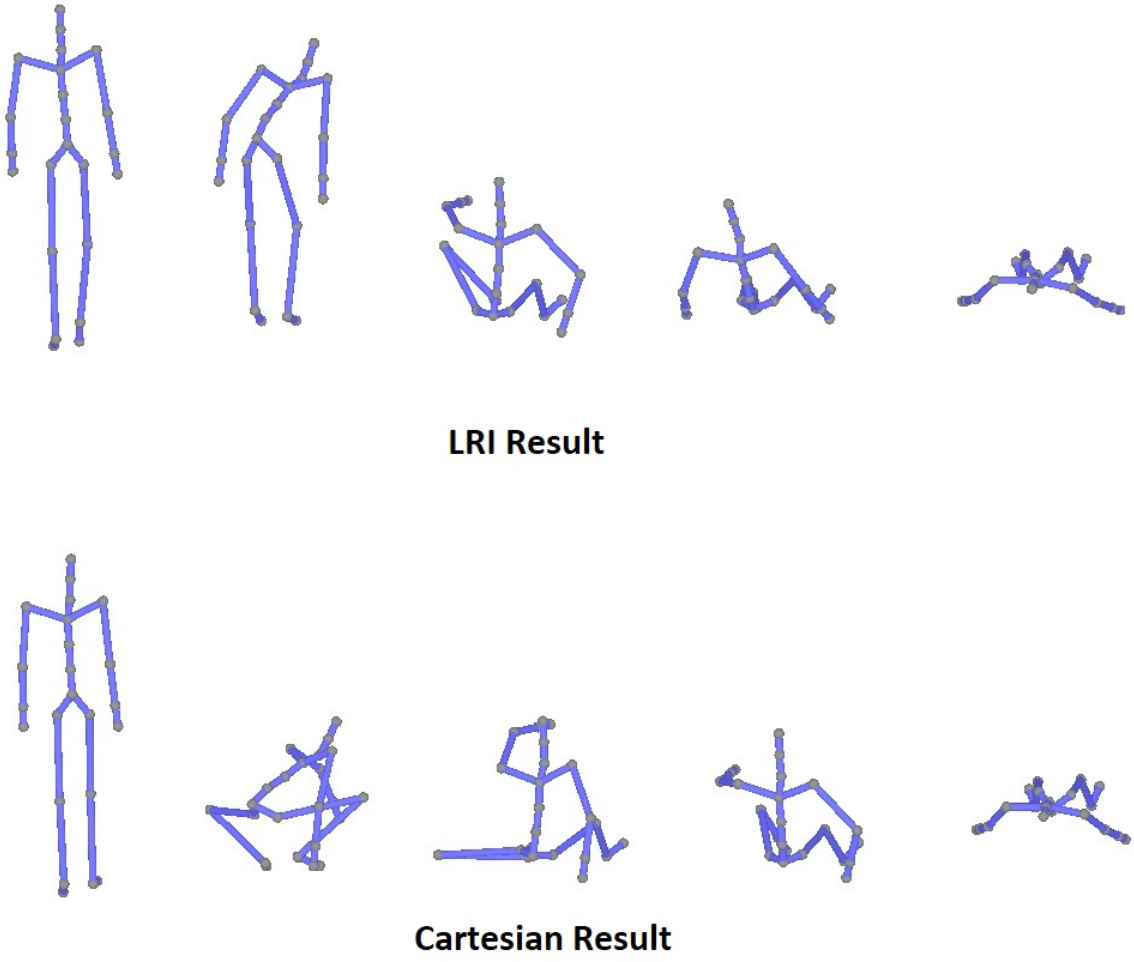


Figure A.6 Lie down motion result in used user experiment.

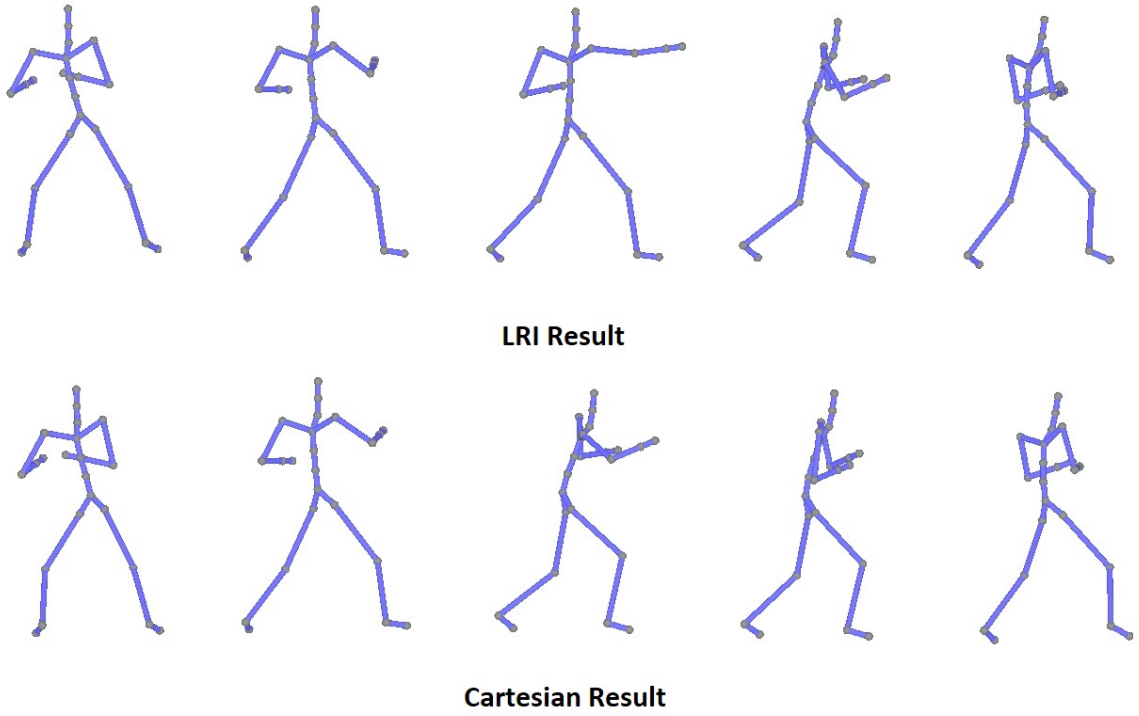


Figure A.7 Punch motion result in used user experiment.

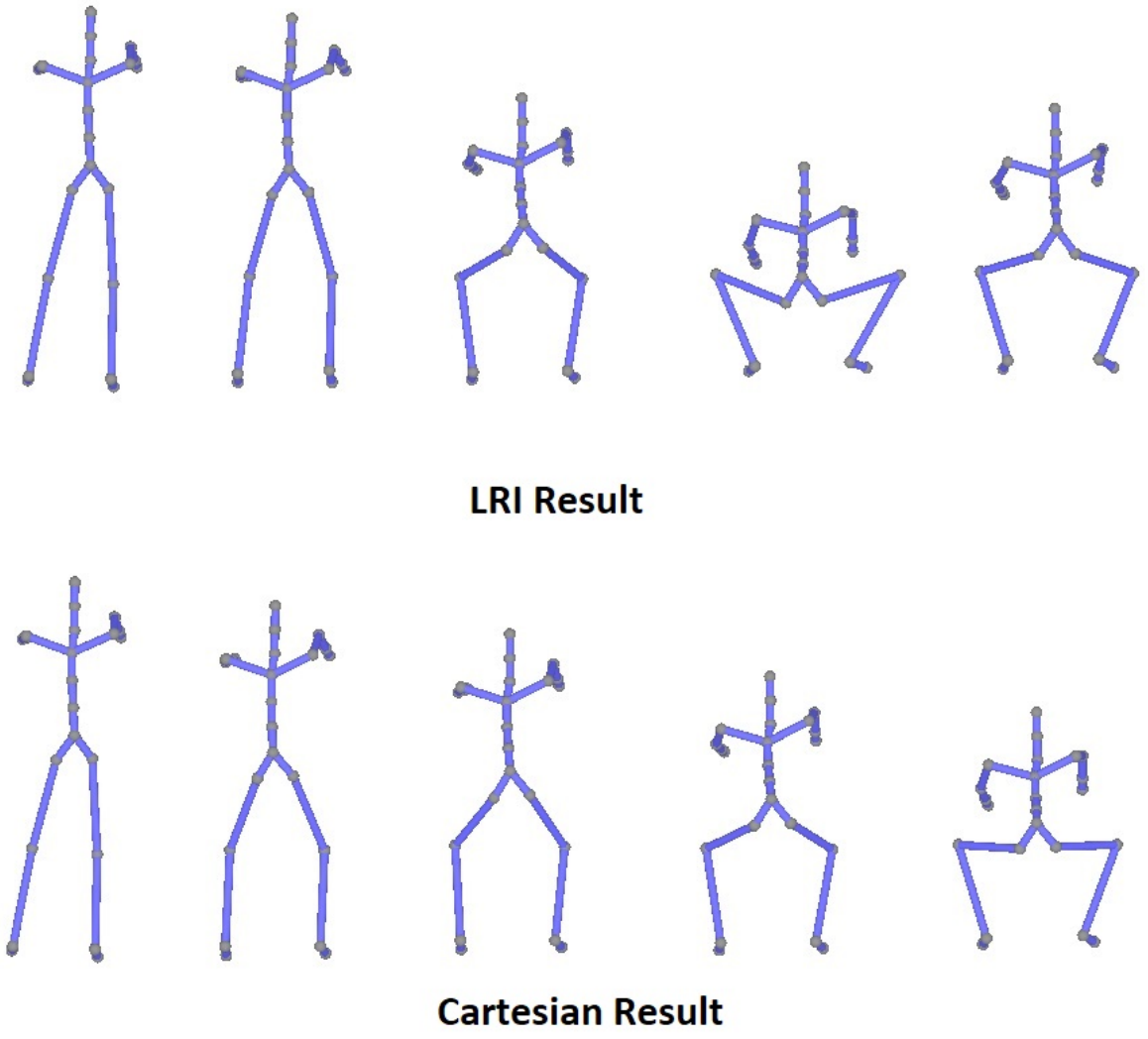


Figure A.8 Squat motion result in used user experiment.

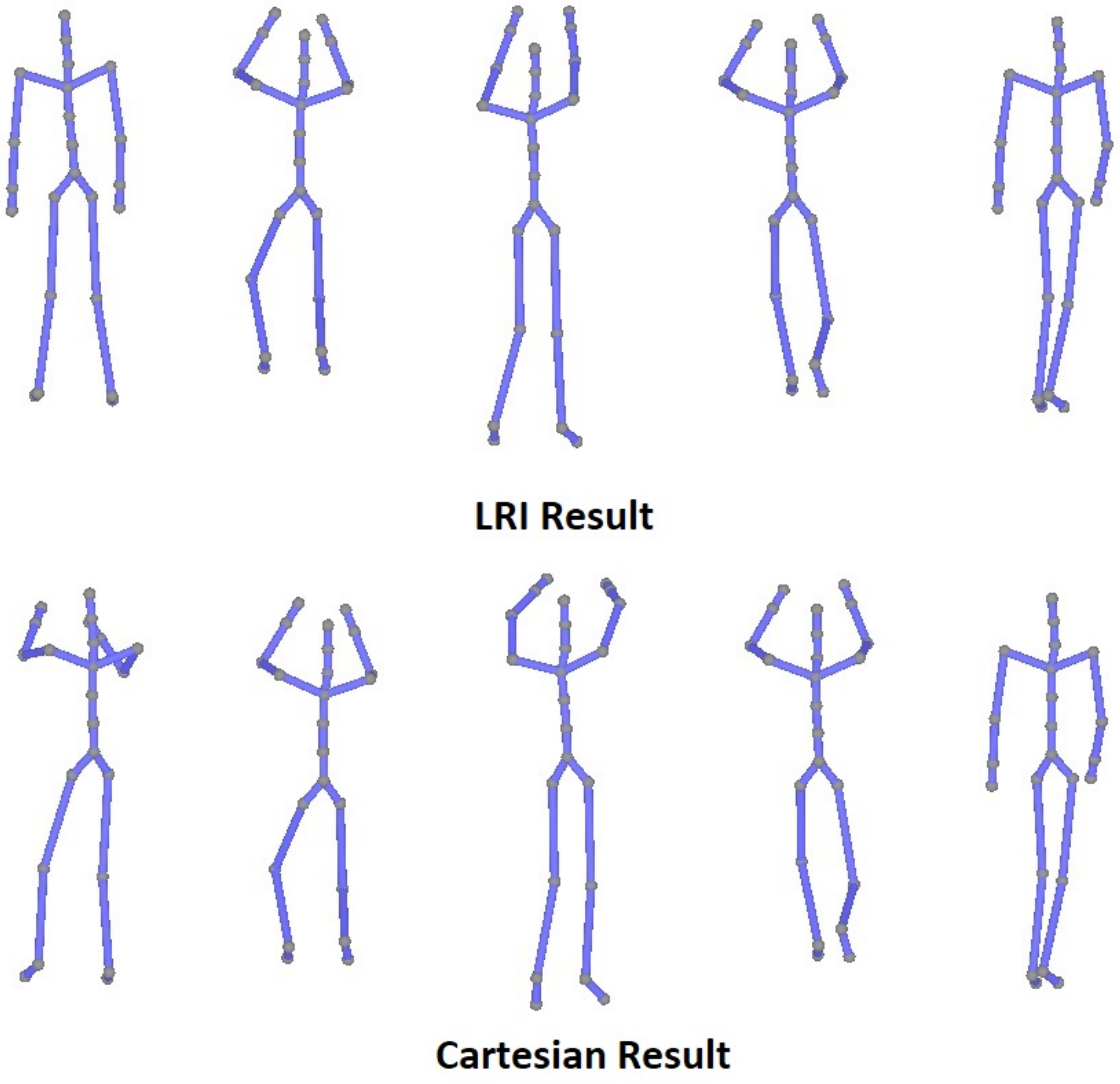
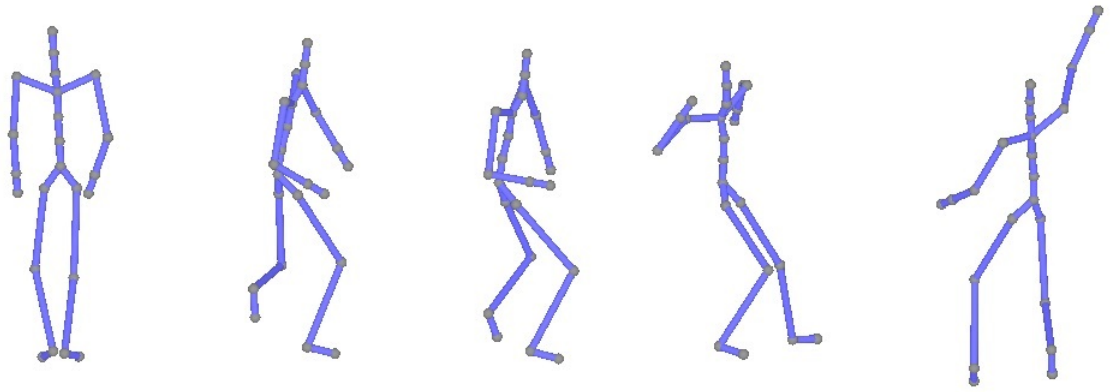
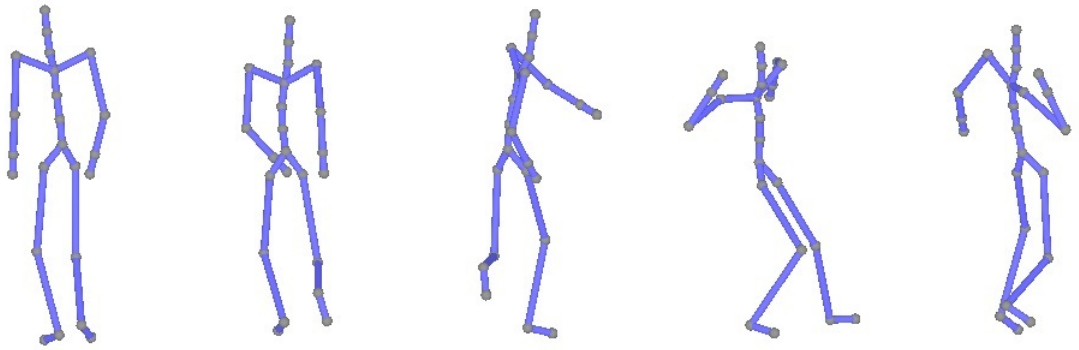


Figure A.9 Throw basketball motion result in used user experiment.



LRI Result



Cartesian Result

Figure A.10 Throw ball motion result in used user experiment.