Quantifying the Accuracy of Deep Learning Algorithms for 3D Human Skeleton Prediction

By

Razieh Shahsavar

A thesis submitted to the
Department of Computer Science
in conformity with the requirements for
the degree of Master of Science

Bishop's University
Canada

Abstract

Recent years have seen an increase in the number of Machine learning methods for extracting 3D animation information from 2D video input. An emerging area within this field is the use of animation skeleton prediction to enhance the accuracy. However, these methods usually rely on datasets where the ground truth is not precisely known. Instead, it is estimated by motion capture suits or manual labeling, both of which are prone to artifacts and human error. In this study, this thesis generates a synthetic dataset with 100% accurate ground truth using the Unity Game Engine's animation system, and tests some widely used 3D skeleton extraction methods on this dataset to determine their accuracy. This thesis defines the optimal angle, distance, and pose for positioning a camera in real-world applications. Results provide insight into the significance of angle, distance, and pose for better predictions, as well as the overall accuracy of 3D animation recognition methods.

Keywords: deep learning, 3D skeleton extraction, Unity Game Engine, Detectron 2, BlazePose, camera positioning.

Acknowledgments

I would like to express my deepest gratitude to my supervisor, Dr. Russell Butler, for his guidance, support, and invaluable expertise throughout the course of MSc project. His unwavering encouragement and insightful feedback have been instrumental in shaping the direction and quality of this thesis.

I would like to acknowledge the support and encouragement received from my husband, daughter and my parents who have provided me with the necessary motivation and understanding throughout this challenging journey.

I am grateful to the faculty and staff at Bishop's University for providing a conducive academic environment and access to necessary resources.

Finally, I would like to thank all the research participants and contributors who made this study possible.

Table of Contents

List of Figures

List of Equations

Chapter 1 Introduction

1.1 Background

3D animation plays a crucial role in diverse domains including entertainment, gaming, virtual reality, and augmented reality. The animation skeleton, comprised of bones and joints representing the human figure, is fundamental for animating humanoid models. Typically, to animate a 3D character requires some advanced motion capture suits which are expensive and require significant setup time. To address these limitations, recent research has focused on using Deep Learning to extract the 3D pose information directly from 2D video input [1], enabling 3D motion capture with a simple webcam [2]. While much progress has been made recently in predicting 3D animation from 2D video input, less effort has been spent on evaluating these algorithms, in part because it is difficult to obtain an accurate ground truth.

1.2 Problem Statement

Existing methods for predicting 3D animation from 2D video input often face challenges in achieving accurate predictions, especially when variables such as camera angles and distances come into play. Furthermore, there is a lack of comprehensive understanding regarding how these variables impact the prediction quality of the animation skeleton, making it difficult to optimize machine learning-based animation capture methods. The absence of a standardized ground truth, like one that could be created using the Unity Game Engine, further complicates the evaluation of these methods' performance, underscoring the need for a reliable benchmark against which different techniques can be tested and compared.

1.3 Thesis Objectives

The thesis aims to achieve the following objectives:

1. Develop a synthetic dataset of human-being skeleton animation.
2. Evaluate different ML techniques of 3D animation prediction based on the developed dataset.
3. Determined optimal camera and distances for capturing 3D animation from 2D Image.

Chapter 2 Literature Review

2.1 Animation Skeleton Prediction

This section provides an in-depth exploration of the basic models used for 3D pose detection and the current state of the art in this field.

2.1.1 Basic Models for 3D Pose Detection

Before the era of deep learning, early methods for 3D human pose estimation were based on handcrafted features [4]–[6]refers to methods where features used for modeling and prediction are manually designed and selected by domain experts rather than being learned from the data by a machine learning model. This process often involves applying domain-specific knowledge to derive features from raw data that the model might not be able to learn itself. Some commonly used handcrafted features in pose estimation include Scale-Invariant Feature Transform (SIFT), Sped Up Robust Features (SURF), Histogram of Oriented Gradients (HOG), and edge features.

For instance, the HOG feature descriptor, used in many pose estimation applications, captures the distribution of local intensity gradients or edge directions, which can provide useful information about the pose of a person in an image.

However, there are several disadvantages to using handcrafted features in pose estimation:

1. Limited Generalization: Handcrafted features may not generalize well to new data or different tasks. The features that work well on one dataset or problem might not be effective on another.
2. High Dimensionality: The dimensionality of handcrafted features can become quite high, especially for complex problems like pose

estimation, which can increase computational cost and make the model prone to overfitting.

3. Time and Effort: Designing and selecting effective handcrafted features can require a significant amount of time, effort, and domain expertise.
4. Lack of Flexibility: Handcrafted features are static and do not adapt based on the data or task. This contrasts with learned features, which can automatically adjust themselves based on the data and task.

In the field of 3D pose detection, various handcrafted feature-based methods have been explored to estimate the spatial configuration of human bodies. One notable approach is presented in the paper [4]. This method focuses on 3D pose estimation from RGB images and takes inspiration from the RGB-D imagery capabilities observed in Kinect systems (Figure 1).

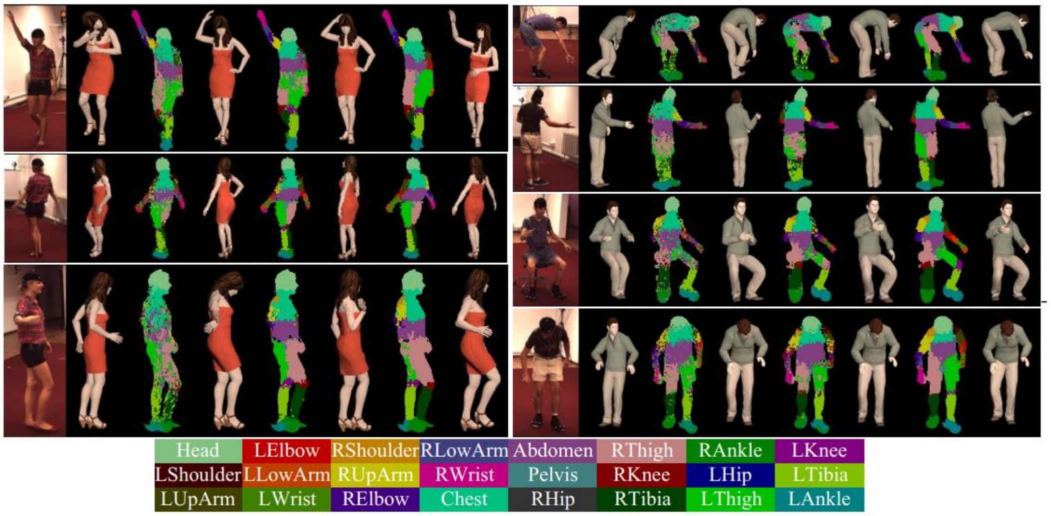


Figure 1: Examples of labeling inference and 3D human pose estimation for the different models [4].

The approach operates on three layers to accurately estimate human poses. Firstly, the method performs 2D human body part labeling by identifying and labeling the various parts of the human body in each 2D image. Next, it employs label-sensitive pooling over a hierarchical region decomposition of the body. This step dynamically computes regions in a hierarchical manner and performs a second-order label-sensitive pooling over these regions. Lastly, the approach utilizes a continuous-valued pose regression technique that employs iterative structured-output modeling to provide contextualization based on 3D pose estimates.

To evaluate the performance and robustness of the method, the Human3.6M dataset is utilized. This dataset offers a large-scale collection of 3D human motion capture data with body part labeling annotations. The dataset was created by recording the activities of multiple subjects from different viewpoints, encompassing a wide range of typical human activities. This dataset includes synchronized 2D and 3D data, consisting of time-of-flight data, high-quality images, and motion capture data. Additionally, accurate 3D body scans of the subjects are provided, along with controlled mixed reality evaluation scenarios. By leveraging this dataset, the model generates robust and stable pose descriptors that adapt to various human pose configurations (Figure 2).

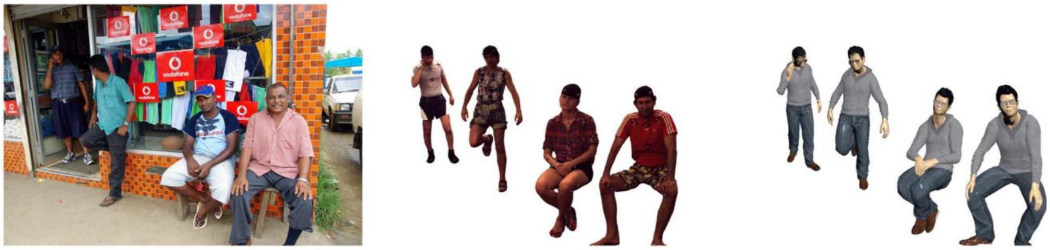


Figure 3: Real image showing multiple people in different poses (left), matching sample of the actors in similar poses (middle) and reconstructed 3D poses from the dataset (right) [5].

Another noteworthy study [5], also uses the Human3.6M dataset. The researchers behind this study present a set of large-scale statistical models and evaluation baselines for the dataset. The experiments demonstrate the improved performance achieved by training models on the Human3.6M dataset compared to existing public datasets. The authors believe that the availability of this dataset and the developed tools will facilitate advancements in computer vision and machine learning, leading to the development of more robust 3D human sensing systems for real-world settings.

Furthermore, the paper [6] introduces an activity-independent method for recovering the 3D configuration of a human figure using 2D locations of anatomical landmarks in a single image. The method leverages a large motion capture corpus as a surrogate for visual memory and solves for an anthropometrically regular body pose, considering camera perspective. By representing human pose as a sparse linear embedding, the proposed method achieves efficient convergence through closed-form computations (Figure 3).

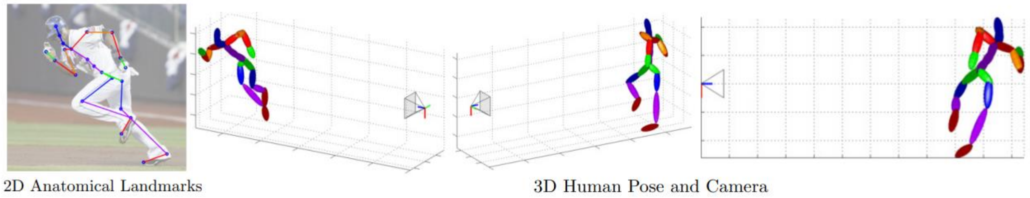


Figure 4: Estimating 3D joint configuration and relative camera pose based on 2D anatomical landmarks [6].

It is important to note that while these handcrafted feature-based methods have shown promising results, they also have certain limitations. For instance, the third method mentioned may encounter challenges in images with strong perspective effects or poses that deviate significantly from the mean pose. Despite these limitations, these basic models provide valuable insights and form the foundation for the development of more advanced and sophisticated methods in 3D pose detection.

2.1.2 State of the Art in Animation Skeleton Prediction

In recent years, significant advancements in 3D animation prediction have been achieved using deep neural networks. These approaches can be categorized into two types, each leveraging unique methods to improve accuracy and performance.

The first type focuses on predicting 3D animation directly from 2D images. For instance, the paper [7] introduces a deep convolutional neural network (CNN) for 3D human pose estimation from monocular images. This method employs a multi-task framework that simultaneously trains pose regression and body part detectors. Additionally, a pre-training strategy is utilized, where the pose regressor is initialized using a network trained for body part detection. The authors demonstrate that the deep CNN has effectively learned the correlations and dependencies between different body parts, leading to improved performance on the Human3.6M dataset.

Another approach, presented in [8], directly regresses the 3D pose from an aligned spatial-temporal feature map. This method utilizes motion information from consecutive video frames to determine the 3D pose of individuals. Unlike traditional methods that compute candidate poses in individual frames and link them in post-processing, this approach directly regresses from a spatio-temporal volume of bounding boxes to a 3D pose. By compensating for motion in consecutive frames, the method achieves better pose estimation results and outperforms existing methods on benchmarks such as Human3.6M, HumanEva, and KTH Multiview Football (Figure 4).

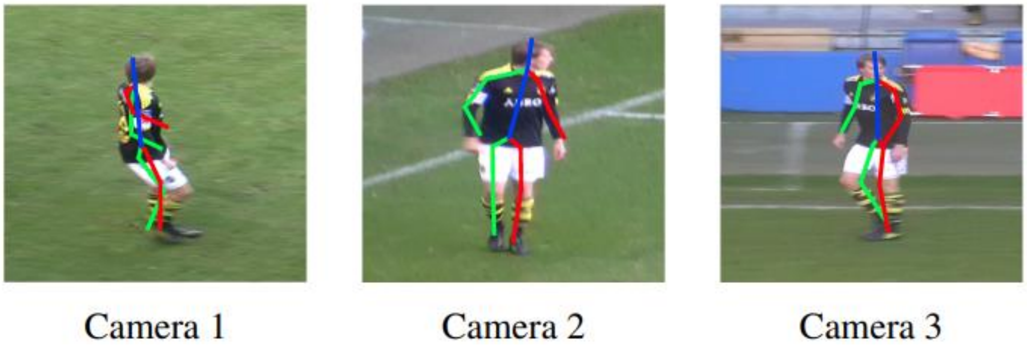


Figure 5: 3D skeleton results on KTH Multiview Football [8] .

In the second type, approaches adopt a two-stage pipeline. Initially, a 2D pose sequence is predicted by a 2D pose estimator from a video frame-by-frame. Subsequently, another estimator lifts the 2D poses to the 3D space. For example, [9] proposes a simple baseline composed of fully-connected layers for 3D human pose estimation (Figure 5). This work explores the limitations of existing methods and focuses on understanding the sources of error in 3D pose estimation. The authors demonstrate that by training their system on the output of a conventional 2D detector, it achieves state-of-the-art results on the Human3.6M benchmark.

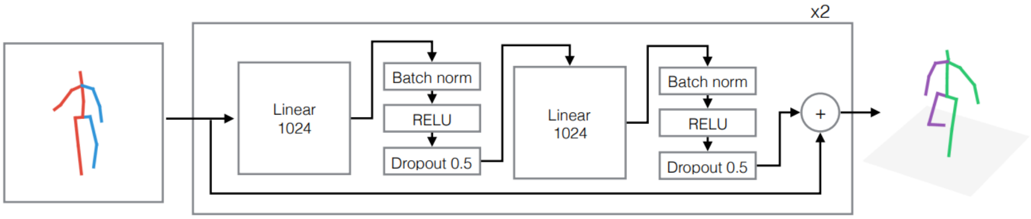


Figure 6: Two-stage pipeline for 3D pose estimation [9].

Other researchers have made significant contributions to improving the representation and modeling of 3D human poses. The paper [10] introduces a fine discretization of the 3D space and trains a ConvNet to predict per-voxel likelihoods for each joint. This approach provides a more natural representation for 3D human pose estimation and surpasses direct regression methods (Figure 6).

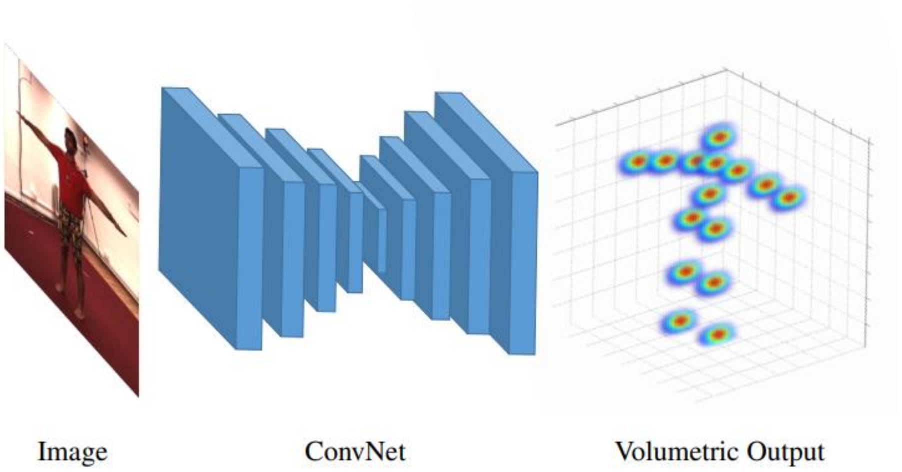


Figure 7: Training convnet to predict per-voxel likelihoods for each joint [10].

Moreover, [11] utilizes an auto-encoder to learn a latent pose representation that captures the dependencies between joints. The combination of Convolutional Neural Networks (CNNs) with auto-encoders leads to enhanced prediction accuracy compared to state-of-the-art methods (Figure 7).

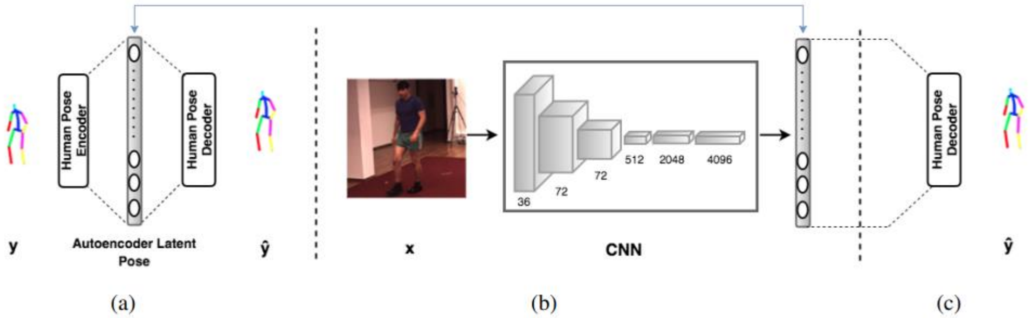


Figure 8: Autoencoder for latent pose representation [11].

Further advances have been made in incorporating temporal information for video-based 3D human pose estimation. The paper [12] proposes a fully convolutional model that utilizes dilated temporal convolutions over 2D keypoint trajectories. By exploiting temporal information, the model achieves superior results on benchmark datasets such as Human3.6M and HumanEva-I. Similarly [13] introduces a framework that leverages matrix factorization for sequential 3D pose estimation. This approach addresses the limitations of existing frameworks and offers an efficient solution to estimate 3D human pose from sequential inputs (Figure 8).

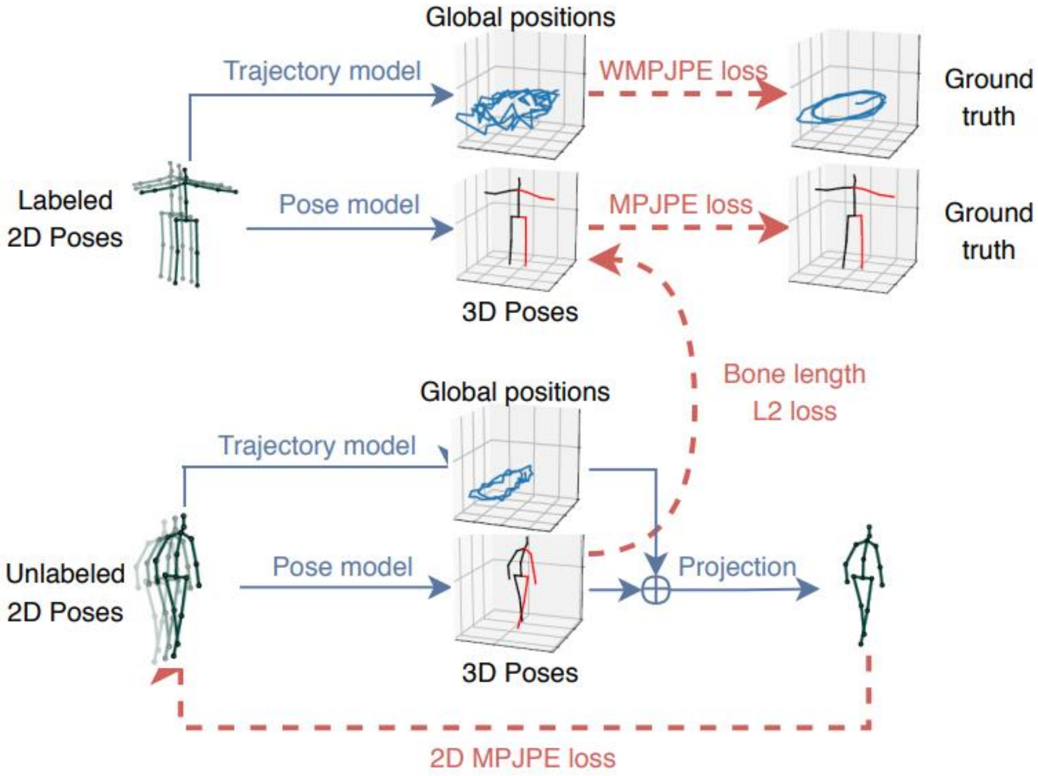


Figure 9: Semi-supervised training with a 3D pose model [12].

The effectiveness of graph-based models has been demonstrated in 3D pose estimation. The Spatial-Temporal Graph Convolutional Networks (ST-GCN) introduced in [14] leverage spatial-temporal graphs to capture both spatial and temporal patterns in skeleton sequences. The ST-GCN model outperforms previous state-of-the-art approaches in action recognition tasks, such as those evaluated on the Kinetics and NTU-RGBD datasets (Figure 9).

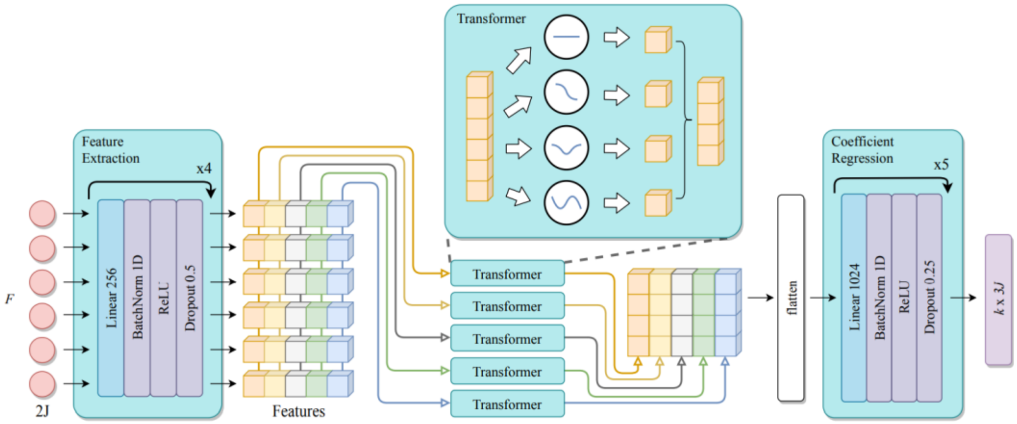


Figure 10: Spatio-temporal graph convolutional network captures both spatial and temporal patterns in skeleton sequence [13].

A recent publication [15] presents a new loss function called motion loss for the training of models designed for monocular 3D human pose estimation from videos. The unique feature of this loss function is its operation principle: it calculates the loss by comparing the motion pattern of the model's prediction against the ground truth key point trajectories. To facilitate the calculation of motion loss, the authors have introduced pairwise motion encoding, a straightforward yet potent representation for keypoint motion (Figure 10).

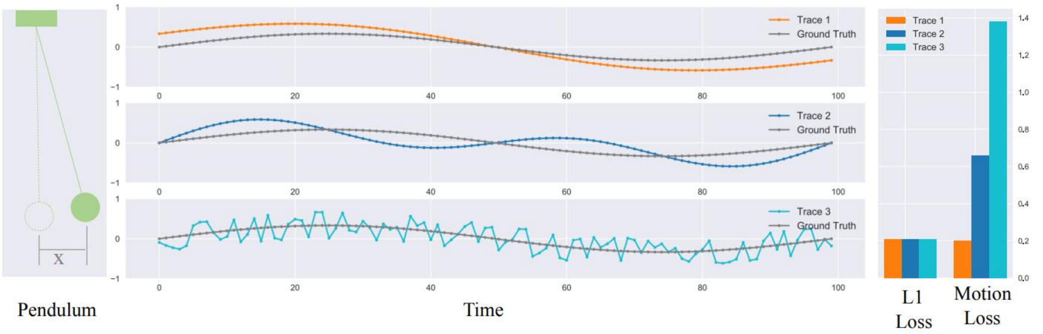


Figure 11: Location estimation of pendulum motion shows the horizontal location as time varies, a sine curve, denoted in gray, and three estimated traces, denoted in blue, orange and cyan [15].

Finally, new graph convolutional network architecture, named U-shaped GCN (UGCN) was designed to better optimize the model using motion loss. This architecture is unique because it is capable of capturing both short-term and long-term motion information, thereby effectively leveraging the supervision from the motion loss.

The authors tested UGCN with the motion loss on two large scale benchmarks: Human3.6M and MPI-INF-3DHP. The results surpassed other state-of-the-art models by a significant margin. Additionally, the model demonstrated a strong ability to produce smooth 3D sequences and recover keypoint motion (Figure 11).

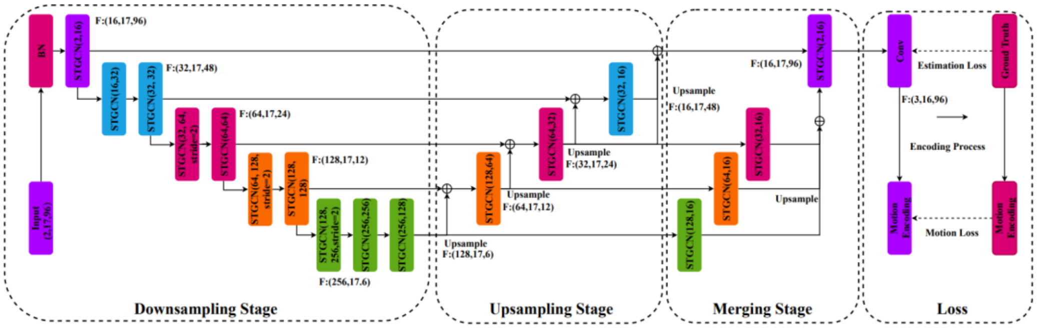


Figure 12: UGCN network structure. Consists of three stages: downsampling, upsampling and merging [15].

Ultimately, the proposed motion loss could inspire other skeleton-based tasks such as action forecasting, action generation, and pose tracking, showcasing its potential for broader applications.

These recent advances in deep learning-based models for 3D human pose estimation showcase the progress made in improving accuracy, handling occlusions, and leveraging temporal information. These methods have surpassed previous benchmarks and offer valuable insights for further research in the field of computer vision and human pose analysis.

2.3 Background on Unity and Synthetic Dataset Creation

In the era of deep learning, where models demand voluminous amounts of data, synthetic data generation has become increasingly important. This is especially pertinent given the GDPR [16], [17] restrictions on acquiring large-scale personal visual data. Synthetic datasets have proven crucial in the thermal spectrum domain, particularly for sensitive data. Two main methods exist for the generation of synthetic data: (1) Direct mapping from the RGB domain and (2) Using virtual environment engines [18].

The former approach often employs Generative Adversarial Networks (GANs) in supervised (paired data) and unsupervised (unpaired data) contexts [18]–[23]. Research shows that supervised GANs, due to the presence of RGB-thermal image pairs, provide superior results [19], [21]. However, for projects where only thermal video footage is available, this method is unfeasible.

Virtual environment engines comprise the latter approach. Despite numerous such engines for visual spectrum synthetic data generation, including CARLA for Advanced Driver Assistance System (ADAS) [24], VIVID for indoor navigation [25], Gazebo for multi-robot simulations [26], and Habitat 2.0 for home assistants [26], there have been comparatively fewer implementations for thermal datasets. Nevertheless, some researchers have successfully used game engines like Unity [27] and Unreal

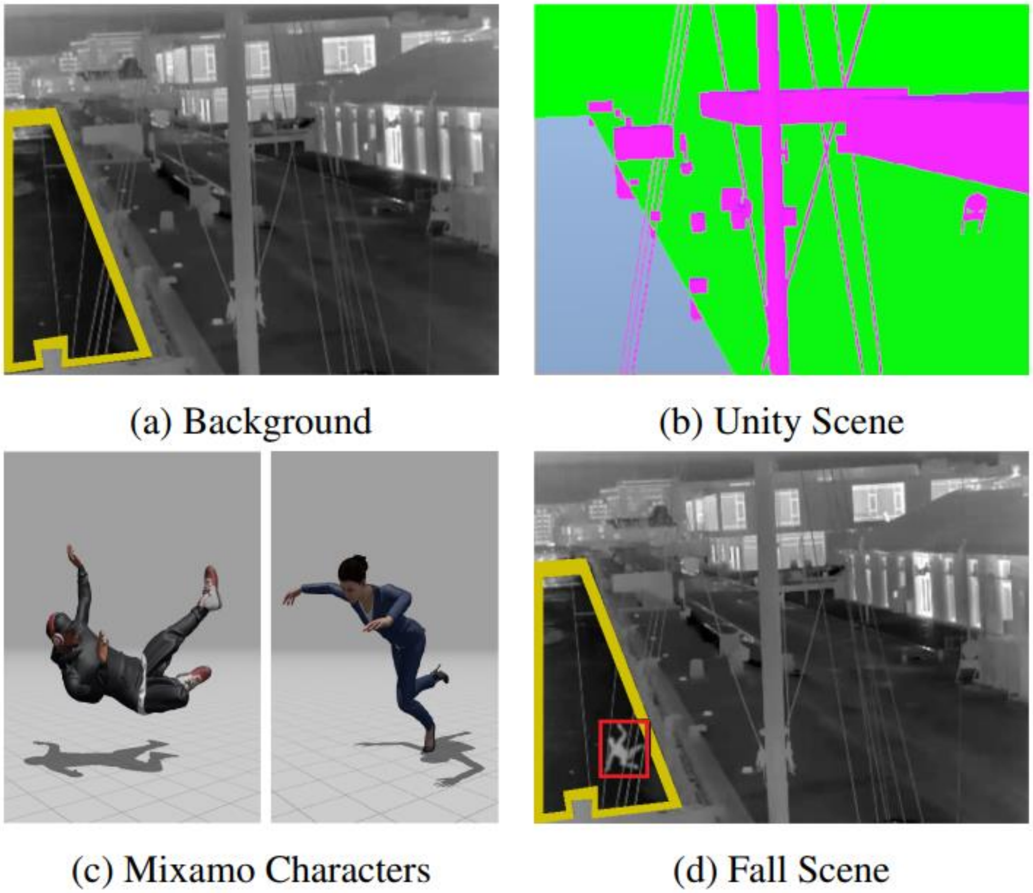
[28] to generate photorealistic synthetic data. Pramerdorfer et al. [29] and Bongini et al. [30], for example, have used Blender [31] and Unity [27] respectively, combining 3D foreground objects with real background images to create synthetic thermal datasets.

The process of generating synthetic thermal foreground videos in the paper employs the game engine Unity [27]. Scenarios of individuals performing actions like walking, running, jumping, and falling into water at a harbor are digitally depicted. These synthetic videos are later combined with generated background instances.

This generation process has two main stages. Initially, 3D models of human figures, along with a variety of animations, are selected. Mixamo [32] is utilized for this task, which is a free library offering a variety of human-like characters and motion-captured animations. For this project, a collection of 79,998 unique foreground video sequences was chosen, each sequence comprising three consecutive frames and illustrating actions such as jumps and falls.

In the subsequent stage, the parameters of the thermal camera used for recording the Long-term Thermal Drift (LTD) dataset [33] are replicated in the Unity camera. This is accomplished using the Universal Render Pipeline in Unity in conjunction with physical camera settings. A synthetic scene is then modeled within Unity, featuring primitive objects placed in locations where real-world objects might block the camera's view of individuals walking on the street. These real-world objects are chosen heuristically after analyzing videos from the LTD dataset. The position and orientation of the Unity camera are adjusted to closely resemble the real-world camera's perspective.

The resulting synthetic scene in Unity, as shown in (Figure 12), includes objects that may obscure the camera's view (colored pink), the background from real images (green), and a waterfront area (gray). A synthetic individual shown in the process of falling is also included in the scene. The real-world background is presented on a rendered texture behind the synthetic scene.



(a) Background

(b) Unity Scene

(c) Mixamo Characters

(d) Fall Scene

Figure 13: Synthetic data generation. a) Example background image from LTD dataset [34] b) same scene synthesized in Unity, c) example fall animations d) synthetically generated falling person merged with (a), yellow enclosure highlighting ROI.

The Unity camera provides versatile settings that allow for capturing animations from various viewpoints, facilitating diverse perspectives and comprehensive visualization of animated sequences.

Utilizing the Perception package [35] provided by Unity [27], a multitude of combinations of 3D meshes, animations, and backgrounds are generated. Rendered masks of people are subsequently employed in post-processing to seamlessly merge the synthetic foreground with the background.

Chapter 3 Proposed method

3.1 Synthetic Dataset Creation

The pivotal role of the synthetic dataset in this research aims to address the challenges posed by the lack of precise ground truth data. The chosen tool for this ambitious task is the Unity Game Engine, recognized for its high fidelity and flexible rendering capabilities.

3.1.1 Virtual Environment and Animation

A robust virtual environment was designed to emulate real-world scenarios. Within this space, the placing a humanoid model and subsequently animated it. These animations weren't random; they were precisely choreographed using Mixamo [32], a platform known for its vast array of human movement simulations. These movements spanned a range of activities, from mundane daily tasks to complex athletic maneuvers, ensuring a diverse data set.

3.1.2 Camera Configurations and Physical Properties

The versatility of the dataset demanded a vast array of camera angles and distances. Programmatically, using C# scripts in Unity, it dynamically adjusted camera attributes. These scripts governed aspects such as distance, angle, speed, focal length, aperture, and shutter speed, ensuring the perfect capture setting for every frame. This programmatic approach not only achieved precision but also allowed for replicability and consistency across multiple captures, as visualized in (Figure 13).

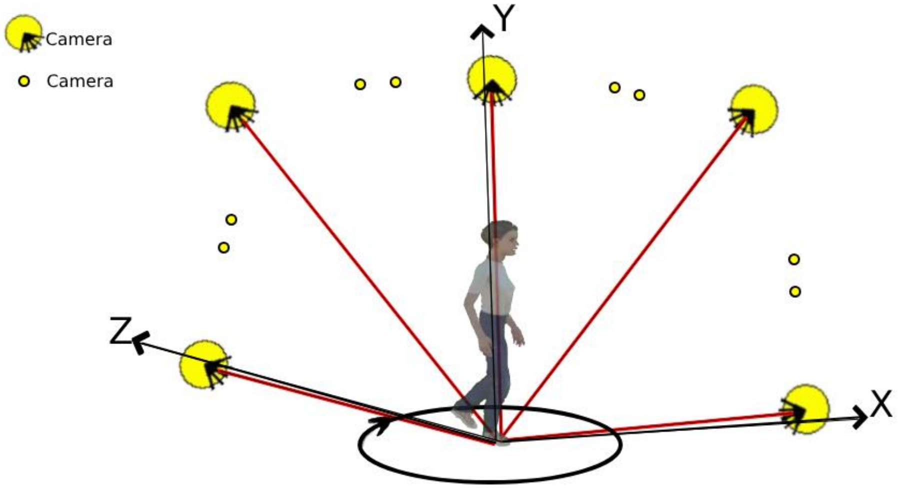


Figure 14: Cameras positioned around the character to create a synthetic dataset from many different angles and distances.

3.1.3 Data Acquisition and Frame Extraction

With the humanoid model enacting the pre-selected animations, the cameras, configured through the C# scripts, recorded the sequences. Key frames were then algorithmically extracted from this extensive footage, ensuring it encapsulated the nuances of the model's movements from an array of perspectives (Figure 14).

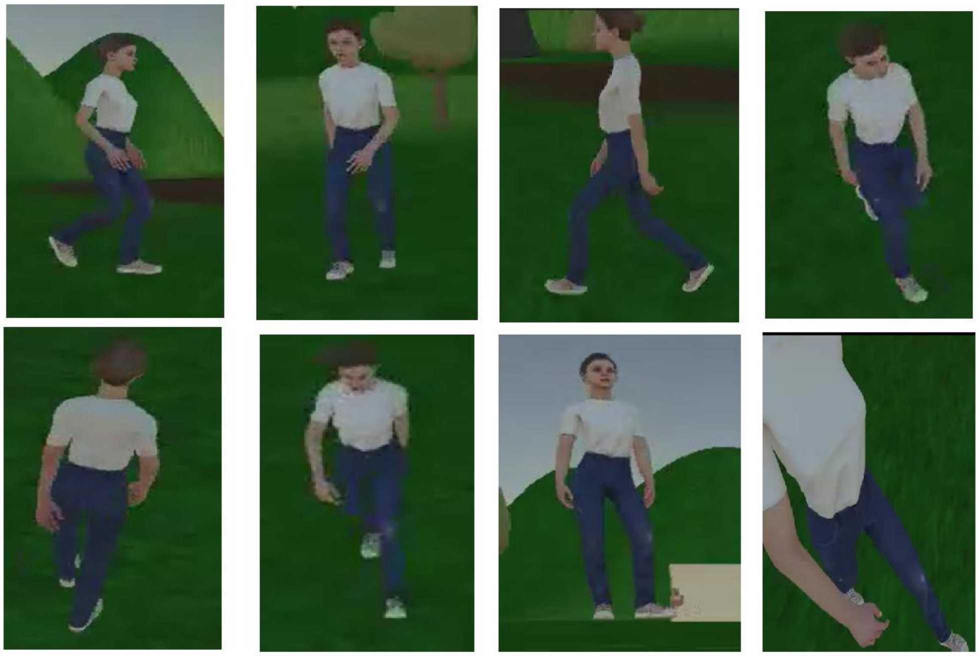


Figure 15: Creating synthetic dataset using Unity Game Engine and Mixamo.

3.1.4 Ground Truth Extraction

A cornerstone of the synthetic dataset's reliability is its meticulous representation of the ground truth. For every extracted frame, it accurately identified both the position and rotation of each joint in the animation skeleton. This detailed information was obtained through an automated extraction process designed within Unity using C#. To facilitate easy access and further evaluations, all the data points – positions and rotations – for each joint across all frames were systematically saved in a CSV file. This comprehensive approach ensures that each frame in the dataset is a precise representation of the model's pose, capturing both its structure and orientation.

Harnessing the flexibility of Unity, combined with the precise programmatic strategy in C#, the synthetic dataset emerges as a paradigm of accuracy and detailed granularity in 3D animation prediction.

3.2 Deep Learning Models

In this thesis, two state-of-the-art predicted skeleton deep learning models were tested on the synthetic data: 1) Detectron 2 [5] and 2) BlazePose [36].

Detectron 2 is a computer vision framework developed by Facebook AI Research [5]. Built on the PyTorch library[1], Detectron 2 offers a flexible and efficient platform for a wide range of computer vision tasks, including pose estimation. Detectron 2 leverages advanced object detection algorithms and convolutional neural network (CNN) architectures to precisely estimate the 2D pose landmarks for each frame of the video (Figure 15). The model initially locates various body parts in every frame, extracting valuable information about the body posture and the locations of 17 key points on the person's body (Figure 16) [9]. This accurate and fine-grained pose estimation forms the foundation for subsequent analysis.

[1] https://github.com/facebookresearch/detectron2

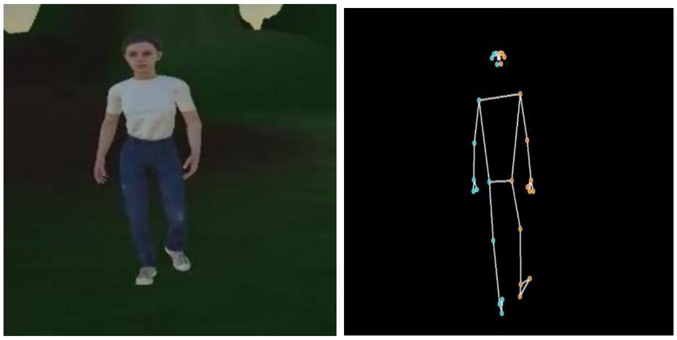
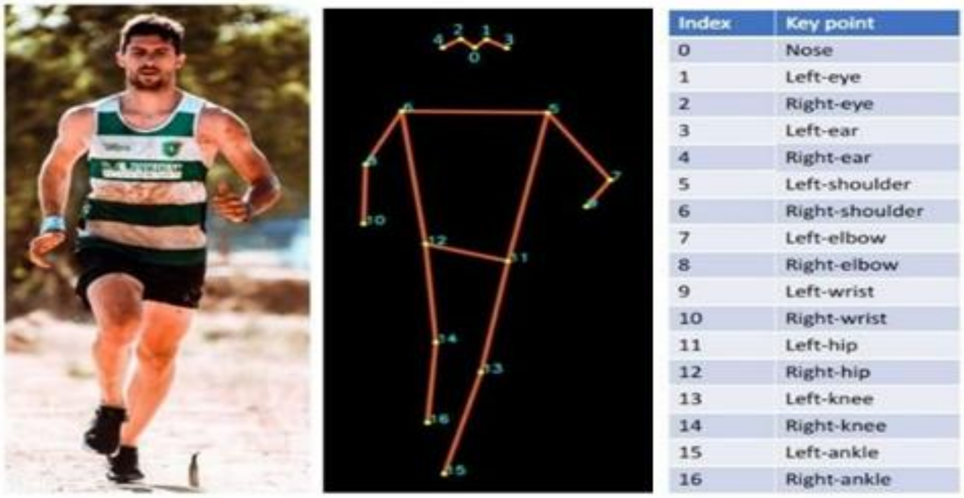


Figure 16: example of 2D pose landmarks for one skeleton of one frame of video.



Index	Key point
0	Nose
1	Left-eye
2	Right-eye
3	Left-ear
4	Right-ear
5	Left-shoulder
6	Right-shoulder
7	Left-elbow
8	Right-elbow
9	Left-wrist
10	Right-wrist
11	Left-hip
12	Right-hip
13	Left-knee
14	Right-knee
15	Left-ankle
16	Right-ankle

Figure 17: Seventeen(17) keypoints on a human body. Left: original image, middle: list of keypoints, right: location of keypoints on the person's body [5].

To further classify and analyze the action being performed, the motion of the body parts over time is carefully examined. This task is accomplished using the Long Short-Term Memory (LSTM) network, a recurrent neural network (RNN). The LSTM network takes the sequence of keypoints from multiple frames and learns to capture the temporal dynamics of the body movement. By leveraging the capabilities of the

LSTM network, the model can accurately classify actions based on the observed motion patterns (Figure 17).

The combined workflow of Detectron 2 and the LSTM network facilitates end-to-end action recognition. Detectron 2 provides the initial pose estimation, capturing the detailed posture information in each frame, while the LSTM network analyzes the temporal evolution of the body's motion. By integrating these two models, the research project achieves a comprehensive understanding of the animation skeleton's behavior and enables accurate classification of actions.

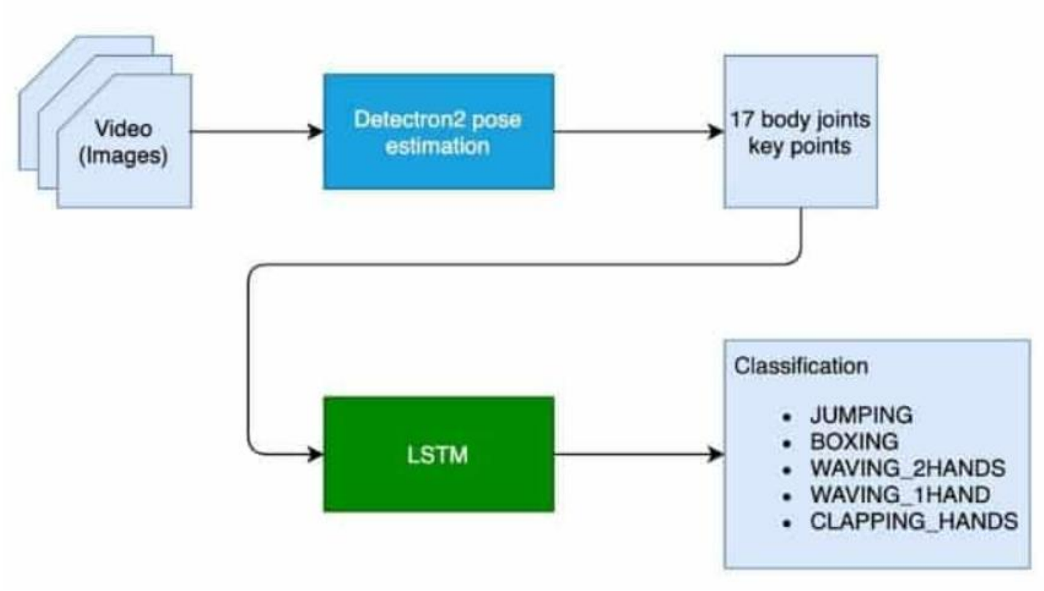


Figure 18: End to end action recognition workflow using Detectron2 and LSTM [37].

In the conducted experiment, when utilizing Detectron 2 to predict 2D poses for approximately 2,000 frames, the prediction time was found to be around 15 minutes. This was achieved with GPU acceleration enabled on Google Colab.

Following the prediction of 2D coordinates by Detectron 2, BlazePose (Full Body) [38] model [39] was employed to predict the corresponding 3D coordinates for each skeleton in every frame. BlazePose, developed by Mediapipe, utilizes a convolutional neural network (CNN) architecture to estimate the 3D coordinates of human pose landmarks.

The current standard for human body pose is the COCO topology, which comprises 17 landmarks distributed across the torso, arms, legs, and face. However, the COCO keypoints exclude ankle and wrist points, thereby lacking essential information about the scale and orientation of hands and feet. This limitation significantly affects practical applications such as fitness and dance. To address this issue, a more comprehensive set of keypoints is required to facilitate the subsequent use of domain-specific pose estimation models, including those for hands, face, or feet.

BlazePose introduces a novel topology consisting of 33 keypoints, encompassing the COCO, BlazeFace, and BlazePalm topologies. This expanded set of keypoints enables the determination of body semantics solely through pose prediction, maintaining consistency with the face and hand models (Figure 18) [36].

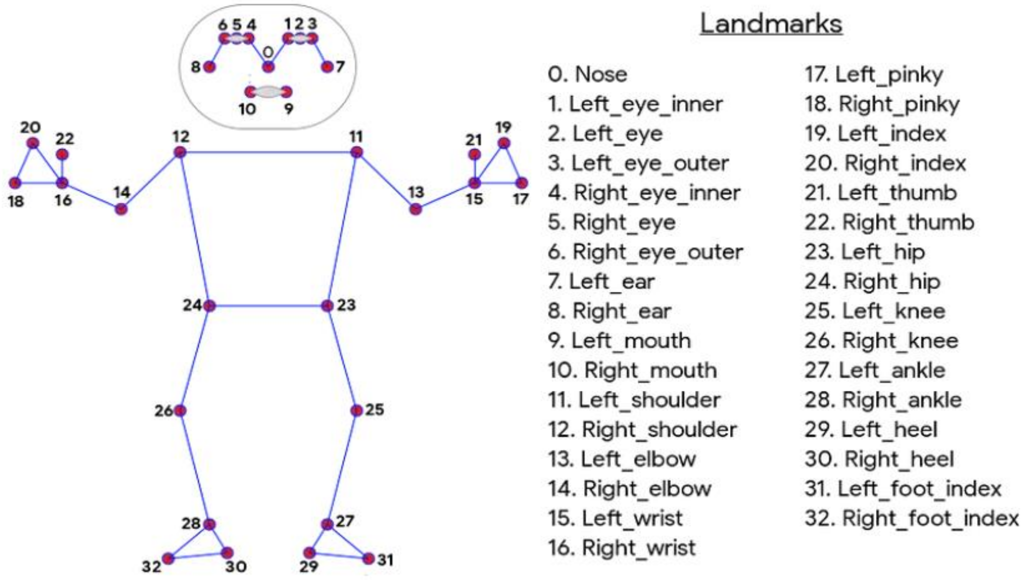


Figure 19: BlazePose keypoints [36]

For pose estimation, BlazePose consists of two machine learning models: a Detector and an Estimator (Figure 19). Using a detector, this pipeline first locates the pose region-of-interest (ROI) within the frame. The tracker subsequently predicts all 33 pose keypoints from this ROI. Note that for video use cases, the detector is run only on the first frame. For subsequent frames it derives the ROI from the previous frame's pose keypoints as discussed below.

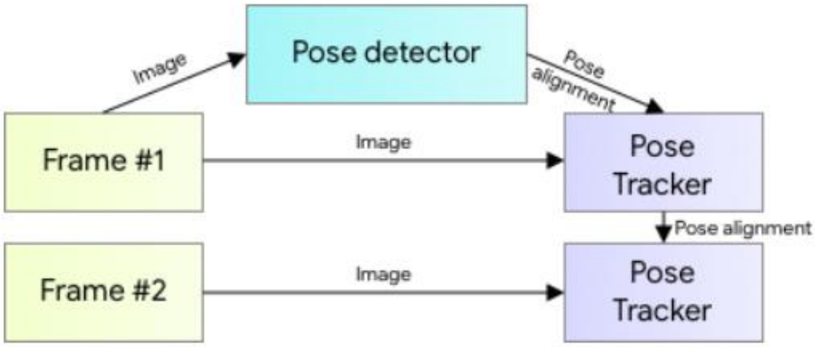


Figure 20: Human pose estimation pipeline overview [36].

To achieve real-time performance of the complete machine learning pipeline, which includes pose detection and tracking models, each component must exhibit high efficiency, processing frames within a few milliseconds. To address this requirement, the strongest signal regarding the position of the torso is obtained from the person's face. This is due to its high-contrast features and relatively consistent appearance. Therefore, a fast and lightweight pose detector is developed by leveraging the assumption that the head should be visible in the context of a single-person use case.

Consequently, a face detector is trained based on the sub-millisecond BlazeFace model, serving as a proxy for the pose detector. It should be noted that this model solely detects the person's location within the frame and cannot be utilized for individual identification. In contrast to the Face Mesh and MediaPipe hand tracking pipelines [36], where the region of interest is derived from predicted keypoints, the human pose tracking approach incorporates the explicit prediction of two additional virtual keypoints. These keypoints effectively describe the human body's center, rotation, and scale in the form of a circle.

Inspired by Leonardo's Vitruvian man, the midpoint of a person's hips, the radius of the circle encompassing the entire person, and the incline angle of the line connecting the midpoints of the shoulders and hips are predicted. This methodology ensures consistent tracking, even in complex scenarios like specific yoga asanas. The figure below illustrates this approach, highlighting how the inclusion of these additional keypoints and circle-based representations contributes to robust pose tracking (Figure 20).

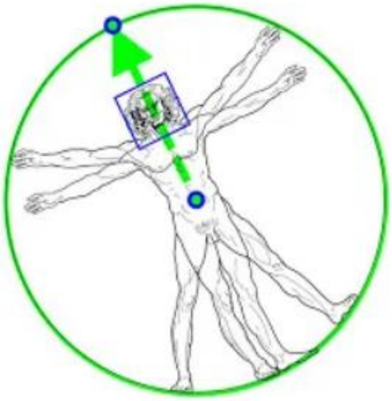


Figure 21: Vitruvian man aligned via two virtual keypoints predicted by the BlazePose detector in addition to the face bounding box [36].

The pose estimation component of the pipeline is responsible for predicting the precise locations of all 33 keypoints associated with the human body. Each keypoint is characterized by three degrees of freedom, namely the x and y coordinates and visibility. Furthermore, the two virtual alignment keypoints mentioned earlier are also taken into account in the prediction. In contrast to existing approaches that rely on computationally intensive heatmap prediction, a regression-based approach is employed in the model. This approach is supervised by a combined heatmap and offset prediction for all keypoints.

Regarding the tracking network architecture [36], the training process consists of two stages. Initially, a loss function incorporating both heatmap and offset information is used to train the center and left tower of the network. Subsequently, the heatmap output is removed, and the focus shifts to training the regression encoder, which corresponds to the right tower of the network. This approach effectively utilizes the heatmap as a supervisory signal to guide the training of a lightweight embedding. The diagram below illustrates this architecture, demonstrating the integration of regression-based prediction and heatmap supervision (Figure 21) [36].

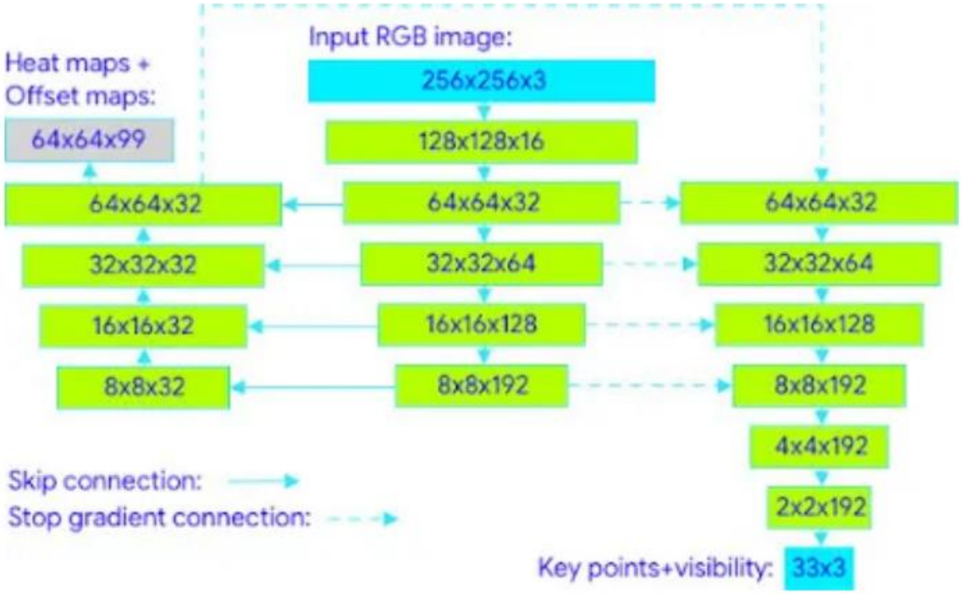


Figure 22: Tracking network architecture: regression with heatmap supervision [36].

Finally the model was trained on a synthetic dataset [36] to learn the mapping between the 2D coordinates of frames and the corresponding 3D coordinates of the animation skeleton.

During the study, when employing BlazePose for 3D animation predictions on roughly 2,000 frames, the processing duration was observed to be close to 20 minutes. This computation was conducted with GPU acceleration activated on Google Colab.

By leveraging the power of both Detectron 2 and BlazePose, this research project successfully predicted the 3D coordinates (Figure 22) of the animation skeleton, enabling accurate and comprehensive analysis of the skeletal motion.

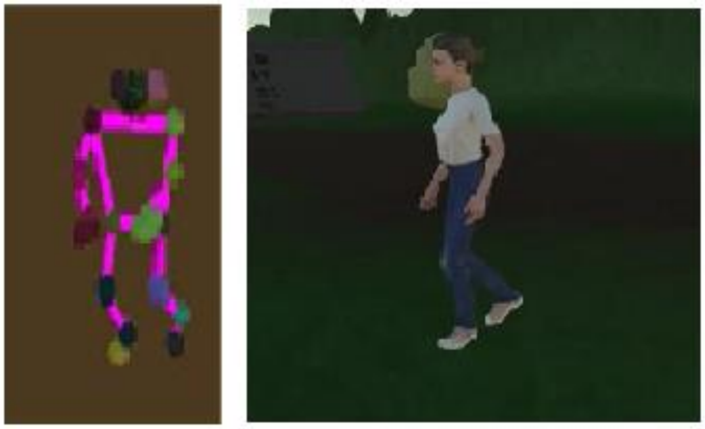


Figure 23: example of 3D animation landmarks for one skeleton of one frame of video.

Chapter 4 Experimental Results

4.1 Training and Evaluation

The synthetic dataset was partitioned into training, validation, and testing sets. The pre-trained deep learning model was employed on the training set using an appropriate loss function and optimization algorithm. The model's performance was evaluated on the validation set to monitor training progress and select the best-performing model based on predefined metrics.

As mentioned above, the experiments it conducted involved applying deep learning models, Detectron and BlezePose, to the synthetic datasets it generated. The primary objective was to measure their performance and the accuracy of the landmark predictions it made on the animation skeletons. Due to differences in how Unity’s mecanim animation system and Blazepose keypoint represent the human skeleton, the performance of the model was evaluated using a customized loss function, which incorporates the center of mass and normalized coordinates for each joint in the animation skeleton. This custom loss function allows for the comparison of the predicted joint positions with the ground truth data, taking into account variations in scale and position.

4.2 Proposed Loss Function

In this project we deigned a customized Loss function to quantify the dissimilarity between the predicted 3D coordinates of the animation skeleton (blazepose keypoints) and the ground truth coordinates (Unity mecanim). The loss function consideres the Center of Mass (COM) of the skeleton and accountes for the differences in positions and orientations across the video frames. This comprehensive loss function served as a metric to evaluate the accuracy of the deep learning model's predictions. The loss function is defined as follows:

a. Calculate the Center of Mass (COM)[40] for both the ground truth and predicted joint positions.

   To calculate the COM for each dimension (x,y,z) separately :
   Let [x1,y1,z1],[x2,y2,z2],…,[xn,yn,zn] represent the joint positions.
   Calculate the mean position along each dimension :

$$x\_mean = ((x1\ +\ x2\ +\ \dots\ +\ xn))/\,n$$
$$y\_mean = ((y1\ +\ y2\ +\ \dots\ +\ yn))/\,n$$
$$z\_mean = ((z1\ +\ z2\ +\ \dots\ +\ zn))/\,n$$

   For each landmark of all frames,the new predicted or ground truth landmarks can be obtained by subtracting the COM:
   Let landmarks(x,y,z)represent the original landmarks.
   Subtract the mean of all joint positions for each joint by Calculate the new_(pred_landmarks or) new_(gt_landmarks as):

$$new\_(predicted\_landmark\ ) = \ predicted\_landmarks(x,y,z) - (x\_mean, y\_mean, z\_mean\ )$$
$$new\_(groundTruth\_landmark\ ) = groundTruth\_landmarks(x,y,z)\ -\ (x\_mean, y\_mean, z\_mean\ ).$$

(Equation 1)

b. Normalize all joint coordinates to ensure consistent scaling[41].
   To normalize the coordinates between 0 and 1:
   Find the maximum value for each dimension (x,y,z)of each landmark.

$$gt\_(coord\_\mathrm{max}\_x\ \ ) = \ \mathrm{max}\,(x\_coord\ )$$
$$gt\_(coord\_\mathrm{max}\_y\ \ ) = \ \mathrm{max}\,(y\_coord\ )$$
$$gt\_(coord\_\mathrm{max}\_z\ \ ) = \ \mathrm{max}\,(z\_coord\ )$$

   Divide each coordinate (x,y,z)separately by the corresponding maximum value.
   Normalize the coordinates and store them in "gt_normalized" and "pred_normalized".

Now, the loss function can be calculated between the ground truth frame and predicted frame. (Equation 2)

c. Evaluate the performance of the model by comparing the normalized and mean-subtracted ground truth and predicted joint positions (As shown in Equation 1).

In the analysis, the computation of the customized loss function for approximately 2,000 frames took about 5 hours. This time frame was achieved with GPU acceleration enabled on Google Colab .

4.3 Performance Analysis

The performance of the deep learning models was assessed using the testing set, comprising video frames and their corresponding animation skeleton coordinates. The evaluation results revealed promising performance in predicting the animation skeleton solely from 2D video input. Instead of using traditional evaluation metrics such as MSE, RMSE, and MAE, the focus was placed on analyzing the predicted skeleton in conjunction with the associated loss function.

The loss function provided insights into the accuracy of the model's predictions. Landmarks associated with a loss value greater than 4 were visualized in red, indicating areas where the model struggled to accurately predict the skeleton and the predicted value is far from the ground truth value. Conversely, landmarks with a loss value less than 4 were displayed in green, indicating satisfactory predictions (Figure 23).

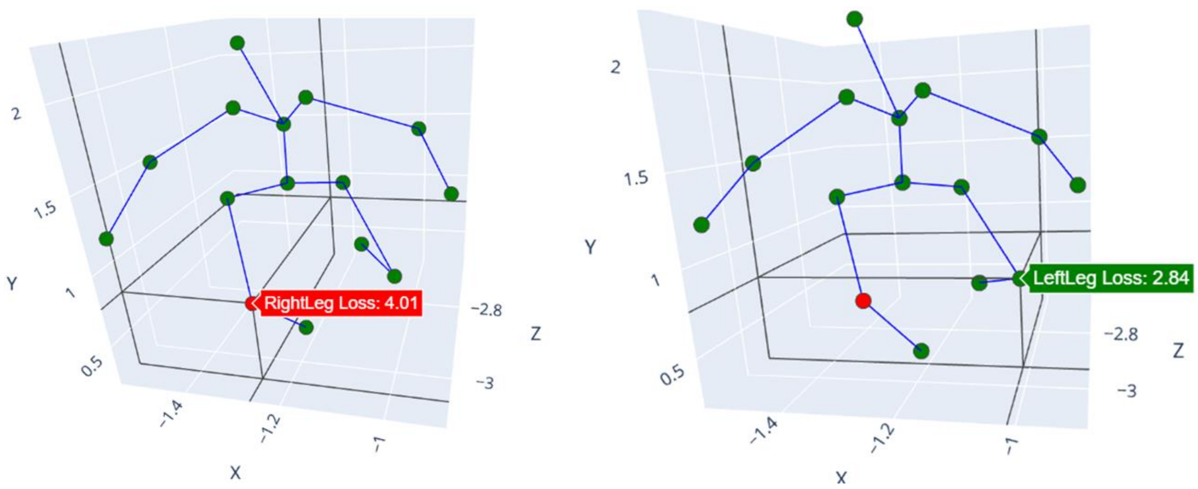


Figure 24: Performance analysis of the Detectron 2 and BlazePose deep learning model.The loss value greater than 4 visualized in red, indicating areas where the predicted value is far from the ground truth value, while green landmarks represent satisfactory prediction.

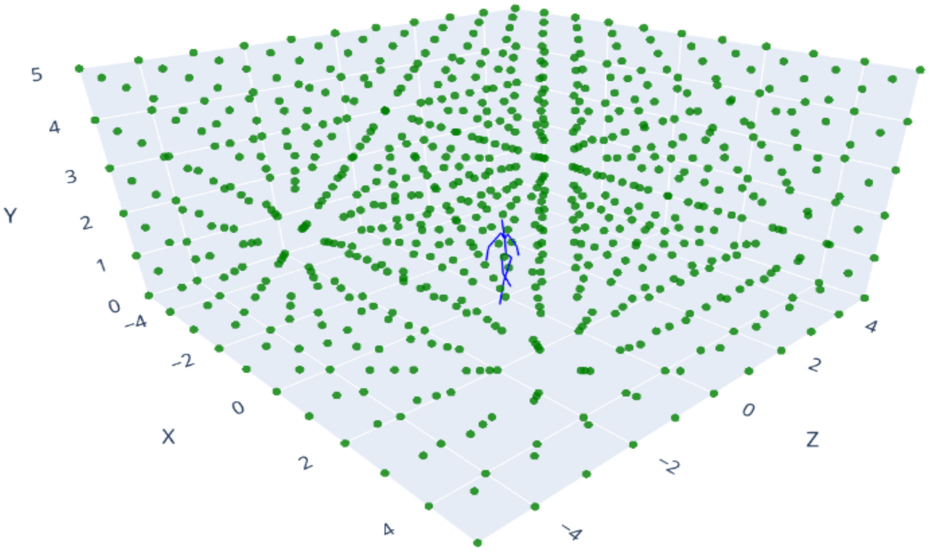


Figure 25: Animation skeleton surrounded by a 3D grid of green points where Each point is a separate camera position.

To quantify model performance, this thesis calculated loss function for each point in the grid, and then converted the grid points to spherical coordinates (Radius, Azimuth, and Elevation). Radius is distance from camera to skeleton. Azimuth and Elevation, traditionally used in applications such as the tracking of celestial bodies, are measures that define coordinates in a three-dimensional space. For instance, they can identify a point's position in the sky. Azimuth indicates the direction to face, while Elevation specifies the vertical angle to look up. Both are measured in degrees or radians.

Azimuth ranges from 0° to 360°, beginning with North at 0°. As one turns to the right (in a clockwise direction), East corresponds to 90°, South to 180°, and West to 270°, eventually returning to North, which is 360° and also 0°. Thus, an Azimuth of, say, 45° implies that the point of interest is located towards the northeast.

Elevation, also measured in degrees, can range from 0°, representing a point just at the observer's horizon, to 90°, indicating a point directly overhead, often referred to as "the zenith".

In the ensuing diagram (Figure 25), the yellow circles (cameras) serve as an exemplification of such points. The green circle bears an Azimuth of approximately 200°, signifying a location southwest of the observer, and an Elevation of around 60°, indicating a position about 2/3 of the way up in the sky from the observer's perspective.

Hence, in a given context, "START AZIMUTH" indicates the direction where the point(camera) of interest appears at one horizon, "MAX ELEVATION" specifies the maximum height the point(camera) reaches in the sky, and "END AZIMUTH" indicates the direction where the point (camera) of interest disappears at the other horizon.

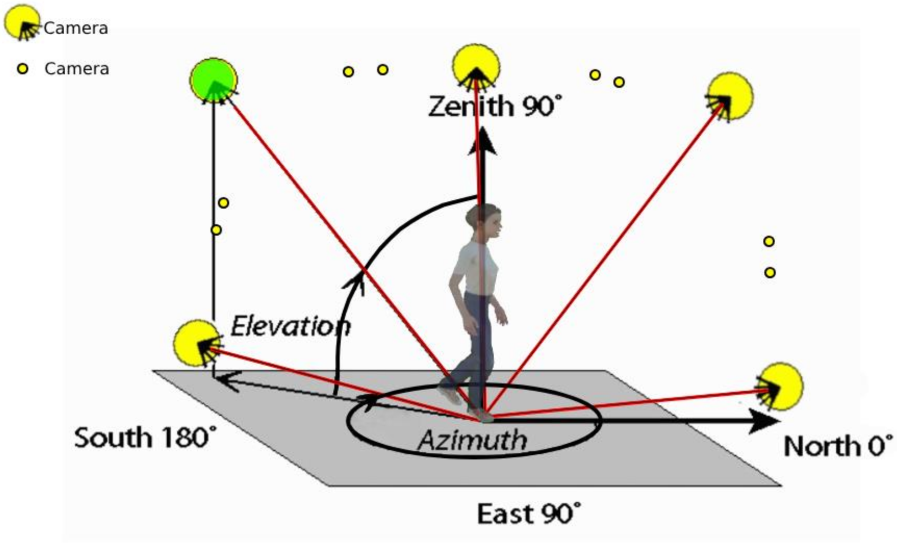


Figure 26: Azimuth and Elevation values in Cartesian coordinates.

The three plots (Figure 26) represent the analysis of camera configurations and their impact on the prediction accuracy of character joints.

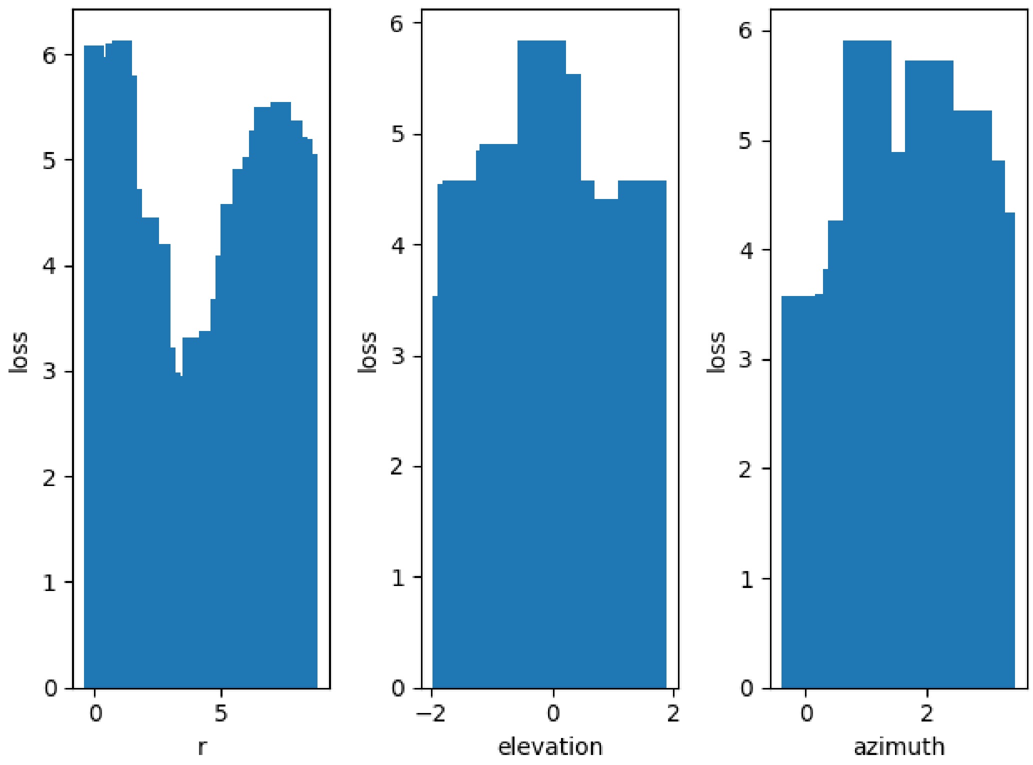


Figure 27: Loss_value for each r, elevation and azimuth value (radians).

The first plot, the "r plot," displays the relationship between the distance of the camera from the character (r-axis) and the corresponding loss value (y-axis). The x-axis of the r plot ranges from 0 to approximately 8, indicating the variation in camera distances in meters. A local-minima is observed at a distance of ~4 meters, indicating that the camera should be placed 4 meters from the subject for optimal results.

The second plot “elevation” illustrates the relationship between the elevation angle of the camera (x-axis) and the associated loss value (y-axis). The x-axis of the elevation plot ranges from -2 to 2, representing the range of elevation angles in radians. This plot allows us to analyze how changes in camera elevation impact the accuracy of predicted character joints.

The third plot, the "azimuth plot," presents the relationship between the azimuth angle of the camera (x-axis) and the corresponding loss value (y-axis). The x-axis of the azimuth plot ranges from 0 to approximately 5, indicating the azimuth angle variations. The face of the character is positioned in front of the azimuth angle 0, indicating that the camera should be placed head-on to the subject.

These three plots collectively provide a comprehensive understanding of the camera configurations used in the project and their effects on the accuracy of character joint predictions. In practical applications, when using BlazePose and MediaPipe, the camera should be placed facing the subject at a distance of 5 meters.

Chapter 5 Discussion

5.1 Implications of Findings

Using a synthetic dataset created in Unity, it was able to constructs a 100% accurate ground truth to investigate the performance of MediaPipe and Blazepose approach to animation prediction based on raw video input. Based on the outcome of our experiment, the camera should be placed facing the subject at a distance of approximately 4 meters. Other angles and distances will yield accurate animation predictions.

5.2 Limitations and Challenges

While the proposed deep learning-based methods shows promising results, several limitations and challenges should be acknowledged. The reliance on synthetic datasets introduces a gap between the synthetic environment and real-world scenarios. The model's performance may be impacted by variations in lighting conditions, occlusions, and other factors encountered in real-world video inputs. Additionally, the complexity of the animation and the inherent limitations of deep learning algorithms can pose challenges in accurately predicting intricate movements and poses.

5.3 Future Research Directions

Future research should focus on three main priorities:

1. **Improving the synthetic dataset.** The virtual environment had basic lighting, simple character models and background. A more realistic virtual environment would give results more relevant to a real-world application.

2. **Testing more models**. Here, this thesis tested only BlazePose and MediaPipe, but as shown in the literature review there are many animation skeleton prediction models to choose from. It would be interesting to see which performs best using this thesis's automated approach with a known ground truth.

3. **Examine if averaging the results across multiple cameras simultaneously could improve performance.** The thesis's loss function was based on a single camera, which inevitably suffers from occlusion artifacts and scaling problems. If multiple cameras were positioned around the model, recording video simultaneously, this thesis could combine their predictions and possibly achieve a more accurate result

Chapter 6 Conclusion

In this research, the primary focus was the evaluation of current methods for predicting animation skeletons from monocular video inputs. The study illuminated significant insights, pinpointing the optimal combinations of angles and distances that enhance prediction accuracy and filling a notable gap in the existing literature. The methodology employed encompasses data preprocessing, the use of pre-trained models, the customization of a loss function, and an in-depth performance analysis.

Beyond presenting results, this research offers a deeper understanding of the myriad factors influencing animation skeleton prediction. The broader ramifications of these findings are manifold:

6.1 Implications and Benefits for the Readership

6.1.1 Practical Utility

The findings serve as a roadmap for those venturing into the realm of animation skeleton prediction using monocular video inputs, guiding them to the configurations that yield the best results.

6.1.2 Academic Contributions

We are the first to systematically evaluate the performance of state-of-the-art deep learning algorithm for animation skeleton prediction. This study will lay the groundwork for future investigations into the accuracy of 3D animation prediction.

6.1.3 Industry Impact

With the animation and gaming industry in a state of rapid evolution, this research offers insights that can help studios and developers refine their animation capture techniques.

In conclusion, this study stands as a comprehensive guide for those delving into the complexities of animation prediction. By emphasizing the nuances of angles, distances, and poses, it paves the way for future innovations in this field.

Bibliography

[1] L. Mourot, L. Hoyet, F. Le Clerc, F. Schnitzler, and P. Hellier, 'A Survey on Deep Learning for Skeleton-Based Human Animation', Computer Graphics Forum, vol. 41, no. 1, pp. 122–157, Feb. 2022, doi: 10.1111/cgf.14426.

[2] C.-Y. Yang et al., 'CameraPose: Weakly-Supervised Monocular 3D Human Pose Estimation by Leveraging In-the-wild 2D Annotations'.

[3] N. Sarafianos, B. Boteanu, B. Ionescu, and I. A. Kakadiaris, '3D Human pose estimation: A review of the literature and analysis of covariates', Computer Vision and Image Understanding, vol. 152, pp. 1–20, Nov. 2016, doi: 10.1016/J.CVIU.2016.09.002.

[4] C. Ionescu, J. Carreira, and C. Sminchisescu, 'Iterated Second-Order Label Sensitive Pooling for 3D Human Pose Estimation'.

[5] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu, 'Human3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments', IEEE Trans Pattern Anal Mach Intell, vol. 36, no. 7, pp. 1325–1339, Jul. 2014, doi: 10.1109/TPAMI.2013.248.

[6] V. Ramakrishna, T. Kanade, and Y. Sheikh, 'Reconstructing 3D Human Pose from 2D Image Landmarks', 2012, pp. 573–586. doi: 10.1007/978-3-642-33765-9_41.

[7] S. Li and A. B. Chan, '3D Human Pose Estimation from Monocular Images with Deep Convolutional Neural Network'.

[8] B. Tekin, A. Rozantsev, V. Lepetit, and P. Fua, 'Direct Prediction of 3D Body Poses from Motion Compensated Sequences', Nov. 2015, [Online]. Available: http://arxiv.org/abs/1511.06692

[9] J. Martinez, R. Hossain, J. Romero, and J. J. Little, 'A simple yet effective baseline for 3d human pose estimation', May 2017, [Online]. Available: http://arxiv.org/abs/1705.03098

[10] G. Pavlakos, X. Zhou, K. G. Derpanis, and K. Daniilidis, ‘Coarse-to-Fine Volumetric Prediction for Single-Image 3D Human Pose’, Nov. 2016, [Online]. Available: http://arxiv.org/abs/1611.07828

[11] B. Tekin, I. Katircioglu, M. Salzmann, V. Lepetit, and P. Fua, ‘Structured Prediction of 3D Human Pose with Deep Neural Networks’, May 2016, [Online]. Available: http://arxiv.org/abs/1605.05180

[12] D. Pavllo, C. Feichtenhofer, D. Grangier, and M. Auli, ‘3D human pose estimation in video with temporal convolutions and semi-supervised training’, Nov. 2018, [Online]. Available: http://arxiv.org/abs/1811.11742

[13] J. Lin and G. H. Lee, ‘Trajectory Space Factorization for Deep Video-Based 3D Human Pose Estimation’, Aug. 2019, [Online]. Available: http://arxiv.org/abs/1908.08289

[14] Y. Cai et al., ‘Exploiting Spatial-temporal Relationships for 3D Pose Estimation via Graph Convolutional Networks *’.

[15] J. Wang, S. Yan, Y. Xiong, and D. Lin, ‘Motion Guided 3D Pose Estimation from Videos’, Apr. 2020, [Online]. Available: http://arxiv.org/abs/2004.13985

[16] The EU General Data Protection Regulation (GDPR).

[17] ‘General Data Protection RegulationGDPR’. https://gdpr-info.eu/ (accessed Jul. 16, 2023).

[18] N. Madan et al., ‘ThermalSynth: A Novel Approach for Generating Synthetic Thermal Human Scenarios’.

[19] M. Kieu, L. Berlincioni, L. Galteri, M. Bertini, A. D. Bagdanov, and A. Del Bimbo, ‘Robust pedestrian detection in thermal imagery using synthesized images’, in Proceedings - International Conference on Pattern Recognition, Institute of Electrical and Electronics Engineers Inc., 2020, pp. 8804–8811. doi: 10.1109/ICPR48806.2021.9412764.

[20] J. Y. Zhu, T. Park, P. Isola, and A. A. Efros, ‘Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks’, in

Proceedings of the IEEE International Conference on Computer Vision, Institute of Electrical and Electronics Engineers Inc., Dec. 2017, pp. 2242–2251. doi: 10.1109/ICCV.2017.244.

[21] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, 'Image-to-Image Translation with Conditional Adversarial Networks', Nov. 2016, [Online]. Available: http://arxiv.org/abs/1611.07004

[22] L. Zhang, A. Gonzalez-Garcia, J. Van De Weijer, M. Danelljan, and F. S. Khan, 'Synthetic Data Generation for End-To-End Thermal Infrared Tracking', IEEE Transactions on Image Processing, vol. 28, no. 4, pp. 1837–1850, Apr. 2019, doi: 10.1109/TIP.2018.2879249.

[23] L. Leal-Taixé and S. Roth, Eds., Computer Vision – ECCV 2018 Workshops, vol. 11134. in Lecture Notes in Computer Science, vol. 11134. Cham: Springer International Publishing, 2019. doi: 10.1007/978-3-030-11024-6.

[24] A. Dosovitskiy, G. Ros, F. Codevilla, A. López, and V. Koltun, 'CARLA: An Open Urban Driving Simulator'.

[25] K. T. Lai, C. C. Lin, C. Y. Kang, M. E. Liao, and M. S. Chen, 'ViviD: Virtual environment for visual deep learning', in MM 2018 - Proceedings of the 2018 ACM Multimedia Conference, Association for Computing Machinery, Inc, Oct. 2018, pp. 1356–1359. doi: 10.1145/3240508.3243653.

[26] N. Koenig and A. Howard, 'Design and use paradigms for gazebo, an open-source multi-robot simulator', in 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566), IEEE, pp. 2149–2154. doi: 10.1109/IROS.2004.1389727.

[27] J. Haas, 'A History of the Unity Game Engine An Interactive Qualifying Project Submitted to the Faculty of WORCESTER POLYTECHNIC INSTITUTE in partial fulfillment of the requirements for graduation'.

[28] ‘UnrealROX+: An Improved Tool for Acquiring Synthetic Data from Virtual 3D Environments’.

[29] C. Pramerdorfer, J. Strohmayer, and M. Kampel, ‘SDT: A SYNTHETIC MULTI-MODAL DATASET FOR PERSON DETECTION AND POSE CLASSIFICATION’, 2020.

[30] F. Bongini, L. Berlincioni, M. Bertini, and A. Del Bimbo, ‘Partially Fake it Till you Make It’, in Proceedings of the 29th ACM International Conference on Multimedia, New York, NY, USA: ACM, Oct. 2021, pp. 5482–5490. doi: 10.1145/3474085.3475679.

[31] ‘Community, B. O. (2018). Blender - a 3D modelling and rendering package. Stichting Blender Foundation, Amsterdam. Retrieved from http://www.blender.org’.

[32] ‘Mixamo-Create Characters and Animations’. https://www.mixamo.com/#/?page=1&type=Character (accessed Jul. 15, 2023).

[33] ‘Hikvision’. Hikvision: Ds-2td2235d-25/50. https://us.hikvision.com/ en/products/more-products/ discontinued-products/ thermal-camera/ thermal-network-bullet-camera-ds (2015), accessed: 2021-09-27 (accessed Jul. 16, 2023).

[34] I. Nikolov et al., ‘Seasons in Drift: A Long-Term Thermal Imaging Dataset for Studying Concept Drift’.

[35] ‘Unity Technologies: Unity Perception package’. https://github.com/ Unity-Technologies/com.unity (accessed Jul. 16, 2023).

[36] ‘BlazePose : A 3D Pose Estimation Model’, Accessed: Jul. 14, 2023. [Online]. Available: https://ai.googleblog.com/2020/08/on-device-real-time-body-pose-tracking.html

[37] ‘Detectron 2 model outputs’.

[38] ‘Blazepose’, Accessed: May 09, 2022. [Online]. Available: https://github.com/vietanhdev/tf-blazepose

[39] ‘Model card MediaPipe Blazepose GHUM 3D’. Accessed: Jun. 21, 2023. [Online]. Available: https://developers.google.com/static/ml-kit/images/vision/pose-detection/pose_model_card.pdf

[40] G. G. Muscolo, C. T. Recchiuto, C. Laschi, P. Dario, K. Hashimoto, and A. Takanishi, ‘A method for the calculation of the effective Center of Mass of humanoid robots’, in 2011 11th IEEE-RAS International Conference on Humanoid Robots, IEEE, Oct. 2011, pp. 371–376. doi: 10.1109/Humanoids.2011.6100864.

[41] T. Wehrbein, M. Rudolph, B. Rosenhahn, and B. Wandt, ‘Probabilistic Monocular 3D Human Pose Estimation with Normalizing Flows’. [Online]. Available: https://github.com/twehrbein/Probabilistic-Monocul