REP3D: 3D Human Motion Capture Dataset for Athletic Movement

by

Jonathan Smith

B.Sc., Acadia University, 2016

Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in the School of Computing Science Faculty of Applied Sciences

© Jonathan Smith 2017
SIMON FRASER UNIVERSITY Fall 2017

Copyright in this work rests with the author. Please ensure that any reproduction or re-use is done in accordance with the relevant national copyright legislation.
Approval

Name: Jonathan Smith

Degree: Master of Science (School of Computing Science)

Title: REP3D: 3D Human Motion Capture Dataset for Athletic Movement

Examining Committee: Chair: Dr. Hao Zhang
Professor

Dr. Greg Mori
Senior Supervisor
Professor

Dr. Ze-Nian Li
Supervisor
Professor

Dr. Angelica Lim
Internal Examiner
Assistant Professor

Date Defended: December 12, 2017
Abstract

The field of human 3D pose estimation suffers from a small population of diverse public motion capture datasets, each with a low number of environments and subjects. We propose a new dataset including 45 participants and 22 environments, using motion capture technology that allows data collection in arbitrary locations. The dataset is composed of video and motion capture data for athletic actions selected from golf and baseball, recorded from a plurality of angles and distances. The annotation process for semi-automatically aligning video data with ground truth 3D joint locations is fully outlined. The performance of a modern human 3D pose estimation model on a subset of the dataset is reported.

Keywords: computer vision; human pose estimation; motion capture
Acknowledgements

First and foremost, I’d like to thank the team at Curv Labs, where I completed this work. Thanks to Shea, Nic, Jason, Ben, Pat, Jake, Tom, Kabir, and Nate. In particular, thanks to Nico and Leah, who coordinated all the data collection sessions and recruited the subjects. The dataset would not exist without them.

For guidance, I’d like to thank my supervisor, Dr. Greg Mori, for providing key insights that were essential to the progress of this research. Furthermore, thanks to Dr. Graham Taylor, Dr. Xiaowei Zhou, and Devinder Kumar for their feedback on the original research plan. Gratitude also goes out to NEXT Canada and NextAI, for providing company funding and a workspace for the summer. This research was also made possible through funding from the MITACS Accelerate program. Solid thanks to my examining committee, as well, for reading the thesis and coming out to a 9:30AM defence.

Final thanks go out to my friends: Tevon, Kirsty, Aiden, Ranee, Daniel Caesar’s music, and coffee, for keeping me sane and grounded throughout this whole process.
# Table of Contents

Approval ii

Abstract iii

Acknowledgements iv

Table of Contents v

List of Tables vii

List of Figures viii

1 Introduction 1

2 Related Works 3
   2.1 2D Human Pose Estimation ........................................ 3
   2.2 3D Human Pose Estimation ........................................ 4
   2.3 3D Pose Estimation Datasets .................................... 6
       2.3.1 HumanEva .................................................... 6
       2.3.2 Human3.6m .................................................. 7
       2.3.3 MPI-INF-3DHP .............................................. 7

3 Dataset 9
   3.1 Structure ....................................................... 9
   3.2 Hardware ........................................................ 10
       3.2.1 Motion Capture ............................................. 10
       3.2.2 Cameras .................................................... 10
   3.3 Collection Process .............................................. 12
   3.4 Camera Calibration .............................................. 13
   3.5 Annotation ..................................................... 16
       3.5.1 Time ......................................................... 16
       3.5.2 Space ...................................................... 16
       3.5.3 Refinement and Alignment ................................ 16
List of Tables

Table 3.1  Table outlining the actions included in the dataset. Each action was performed up to 8 times at different angles. . . . . . . . . . . . . . . . 10
Table 3.2  Brief descriptions of all the locations included in the dataset. . . . . 11
Table 4.1  Validation PCK values for a set of CTF models and thresholds (in cm). 29
Table 4.2  Validation and test reconstruction error for each action and actor in millimetres. Annotations are incomplete for subject 5. . . . . . . . . . 29
Table 4.3  Results on a fully held out test for an individual subject. . . . . . . 30
### List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>A result from our pose estimation dataset. The blue skeleton shows the ground truth joint locations, while the red skeleton has been estimated given only the image on the left.</td>
<td>2</td>
</tr>
<tr>
<td>2.1</td>
<td>An example of a result from a 2D pose estimation algorithm. (Copyright ©2016 IEEE)</td>
<td>4</td>
</tr>
<tr>
<td>2.2</td>
<td>A visualization of the model structure from [32], along with the visual modes used for estimating the pose. (Copyright ©2013 IEEE)</td>
<td>4</td>
</tr>
<tr>
<td>2.3</td>
<td>A set of examples of 3D human pose data, paired with the associated monocular RGB images, from [36]. (Copyright ©2017 IEEE)</td>
<td>5</td>
</tr>
<tr>
<td>2.4</td>
<td>An example of a frame from the HumanEva motion capture dataset, from [5].</td>
<td>7</td>
</tr>
<tr>
<td>2.5</td>
<td>Examples of frames and poses from the Human3.6m dataset [4].</td>
<td>8</td>
</tr>
<tr>
<td>3.1</td>
<td>The XSens MVN Awinda, shown here on a mannequin.</td>
<td>11</td>
</tr>
<tr>
<td>3.2</td>
<td>3D skeleton created using joints from the MVN Awinda.</td>
<td>12</td>
</tr>
<tr>
<td>3.3</td>
<td>Frame taken from a video of a golf swing, recorded by one of the three GoPro Hero Session cameras.</td>
<td>13</td>
</tr>
<tr>
<td>3.4</td>
<td>Diagram of the camera setup. All three cameras are GoPro Hero Session cameras mounted on tripods, and synced to a GoPro Wifi Remote.</td>
<td>14</td>
</tr>
<tr>
<td>3.5</td>
<td>The N-pose used for calibrating the MVN Awinda. The subject must hold this pose for approximately 5 seconds for each calibration.</td>
<td>14</td>
</tr>
<tr>
<td>3.6</td>
<td>A frame from a video containing a planar calibration pattern. The points in this pattern are used to optimally estimate the intrinsic camera parameters.</td>
<td>15</td>
</tr>
<tr>
<td>3.7</td>
<td>A snapshot of the graphical user interface used for aligning videos and motion capture data in time. The user can move forward and backward through the video and MVN frame-by-frame to find a single frame that matches in both.</td>
<td>17</td>
</tr>
</tbody>
</table>
Figure 3.8 Visualization of the alignment between MVN data and video data. A key frame is selected that matches between the two, to align the entire video to MVN frames.

Figure 3.9 GUI used for annotating joints. The user is directed to click on the listed joint on the given image. If the joint cannot be located due to occlusion, the annotator can skip to another joint/image.

Figure 3.10 3D joint locations reprojected onto the 2D camera plane, as a result of a successful camera calibration process.

Figure 3.11 Projected 3D joint locations in a case where the calibration routine has not converged. The right hand is mistakenly projected far away from the subject.

Figure 3.12 Number of annotated calibration points $N$ vs. the resulting optimal projection error after time shifting, $E^*$. No clear correlation appears to exist between these two variables.

Figure 3.13 Histogram of optimal time adjustments $i^*$, with a maximum shift value $K$ of 40 frames. The level of frequencies away from 0 imply that the human time annotation is often far from optimal.

Figure 3.14 An example of a bounding box selection, given 2D joint locations on an image. The width of the padding is determined by the padding factor $\delta$.

Figure 3.15 Visualization of the joint depth calculation. The angle $\theta_i$ is calculated using the joint’s projection onto the camera sensor.

Figure 4.1 A high-level visualization of the architecture of the stacked hourglass network. [24] (Copyright ©2016 IEEE)

Figure 4.2 A more fine-grained visualization of the stacked hourglass architecture. Note the recombination of upsampling features with prior convolutional features, as well as the re-routing of the heatmap output to a new hourglass. [24] (Copyright ©2016 IEEE)

Figure 4.3 A frame from the Human3.6m dataset, with corresponding predicted joint heatmaps and predicted skeletons. [17] (Copyright ©2017 IEEE)

Figure 4.4 A high-level visualization of the iteratively increased z-resolution between hourglasses in the Coarse-to-Fine model. [26] (Copyright ©2017 IEEE)

Figure 4.5 Example of limbs being accurately located despite heavy occlusion. Ground joint locations are on the left, and predictions are on the right.

Figure 4.6 Successful case of pose estimation mid-swing. Ground joint locations are on the left, and predictions are on the right.
Figure 4.7 Failure case, due to the dataset containing very few poses with raised arms. Ground joint locations are on the left, and predictions are on the right.

Figure 4.8 Mild failure case. The structure of the pose is correct, but joint depths are not entirely accurate. Ground joint locations are on the left, and predictions are on the right.

Figure 4.9 Mild failure case. All joints are accurately located, but the spine joint is completely missed. Ground joint locations are on the left, and predictions are on the right.

Figure 4.10 Success case of estimation mid-swing, with some small discrepancies in arm depth. Ground joint locations are on the left, and predictions are on the right.
Chapter 1

Introduction

Athletic movement is a key aspect of life for humans. Amateur sport is a massive industry, with teams composed of people across the globe, both young and old. Movement in athletics can be analyzed through the use of large expensive motion capture studios, but they are expensive and not accessible to everybody. Thus, the ability to provide movement analytics through widely available technology is motivated. A wide array of athletic activities can benefit from automatic analysis of movement from video, such as golf, baseball, yoga, weightlifting, and basketball, just to name a few.

To approach this problem, 3D human pose estimation from monocular RGB images will be employed. In other words, the movement of a human is analyzed by estimating the location of their joints in 3D space, given only an image. An example of what this process looks like can be seen in figure 1.1. The key to this step is the collection of a new dataset for training models to perform 3D human pose estimation, the development of which is the primary focus of this document. For the deployment of a pose estimation system based on deep learning, a large-scale dataset is crucial. This dataset will be named REP3D to denote the repetition of movements recorded with 3D motion capture.

The applications of the estimation of human pose extend beyond sport – surveillance and analysis of pedestrian movement can benefit from the information of the person’s pose. The motivation for restricting ourselves to monocular RGB images is that most people carry monocular RGB cameras with them at all times on their mobile phones. Thus, this approach is the most sensible for large-scale product deployment.

In the past, the development and proliferation of large-scale datasets has been a strong driver for progress in deep learning applications. Beyond the considerable advancements in image understanding that came from the ImageNet competition [31], deep learning developments have come in response to other large-scale datasets such as MS-COCO [20], Sports-1M [19], Human3.6M [17], and KITTI [15], to name a few. Thus, there always exists a motivation to expand the field of current datasets, and to thoroughly investigate the shortcomings of existing datasets.
Figure 1.1: A result from our pose estimation dataset. The blue skeleton shows the ground truth joint locations, while the red skeleton has been estimated given only the image on the left.

Contributions

In this thesis, we propose a new dataset for 3D human pose estimation from RGB image/video. The dataset contains videos of 45 people performing 5 athletic actions in 22 locations. Each video was taken from 3 angles and has corresponding 3D joint locations for every frame. Annotations currently exist for over 50,000 frames, and the dataset contains 1,213,452 frames in total. Benchmark performance values are reported on the annotated data.
Chapter 2

Related Works

2.1 2D Human Pose Estimation

Before taking a look at the problem of 3D human pose estimation, the history of the approach to a similar problem, 2D human pose estimation, will be investigated. For the purpose of this document, we will be restricted to analysis of pose estimation from monocular RGB data i.e. from a single view of the pose, using a standard RGB camera with no depth channel. An example of what the result of this estimation might look like can be seen in figure 2.1.

The problem is posed as the estimation of keypoint (i.e. joint) locations in pixel coordinates on an image. Before the advent of deep neural networks, a myriad of more traditional methods were employed to solve this problem. For example, in [32], the possible human poses are clustered into a set of fundamental modes, and a Support Vector Machine (SVM) is trained on top of Histogram of Oriented Gradients (HOG) [16] features for classifying the pose. A visualization of this model, along with the discovered modes, can be seen in figure 2.2. Another approach in [28] uses an extension of the pictorial structure model in [14] to model relationships between body parts to estimate part and joint locations, also using HOG features. The model in [30] employs an iterative approach by predicting joint locations based on increasingly large image patches and previous predictions of all joints. It uses a set of classifiers for identifying the likelihood of a patch containing a joint, followed by a combination of the predictions of all classifiers across the entire image. Following multiple iterations of this process, significant improvements in performance were seen.

Moving forward to the use of convolutional neural networks (CNNs), some modern approaches use a much deeper architecture than older methods. For example, the approach in [39] extends [30] using a deep CNN to jointly predict all keypoint locations. With intermediate supervision, the model iteratively predicts the joint locations at a number of repeated stages, effectively using previous predictions to refine its estimates. The idea of iterative improvement continues through another state-of-the-art approach in [24], which
uses multiple iterations of downsampling and upsampling to predict heatmaps instead of coordinates.

### 2.2 3D Human Pose Estimation

The problem of 3D human pose estimation from monocular RGB images is posed as follows: given a single camera image containing a human, identify the location of the joints of the person in a 3D coordinate system. A full recent review of the literature on this topic can be found in [33]. Extensions of this problem include adding a depth channel to the image [34], or combining multiple camera views [27]. In our case, the data is restricted only to that which is obtainable from a standard camera. An example of a standard case of input and output can be seen in figure 2.3.

One traditional approach to the problem involved using hand-crafted image features [16], in which a random forest classifier is trained from histograms of oriented gradients. This approach poses 3D pose estimation as a pure classification problem, which is insufficient for the purpose of generalization. Some other classical approaches avoid direct matching by
directly estimating pose from the image. For example, the approach in [9] uses relevance vector machine (RVM) regressors to directly predict 3D pose from human silhouettes. Another example is the approach in [18], which uses both labelled and unlabelled data to learn the manifold of 3D poses from a large set of hierarchical image features.

Another approach employs a graphical model, attempting to directly model relationships between body parts [43], which is more of an approach for jointly predicting body shape as well as pose. A natural progression is to impose geometric constraints on body part orientation i.e. encouraging a model to predict joint positions and body part length that seem physically feasible [29]. The approach in [29] uses this geometry constraint as a refinement on pose estimation from structure-from-motion. More recently, ignoring geometric constraints and instead allowing for the model to learn geometric pose structure in an unconstrained fashion has achieved better results.

With the recent advent of deep learning approaches for solving vision-related problems, the state-of-the-art performance has been achieved with CNNs. Some modern approaches perform a direct matching process between the 2D video and a large stockpile of 3D poses [11] [42], which is reliant on having an extremely diverse and comprehensive dataset. This approach lifts an estimation of the 2D pose to 3D either by direct matching in [11] or by expectation-maximization across the entire movement video [42]. Another approach that considers both 2D and 3D cues is proposed in [36], which directly fuses 2D joint predictions with image data, using a CNN for the 2D predictions and fusion.
A considerably simpler CNN approach yields impressive results – a simple deep feedforward neural network acting directly on the output of the 2D pose estimation model in [24] shows the ability to regress 3D joint locations directly from estimated 2D joint locations. Considering the entire video instead of single frames has also had its advantages – temporal consistency over 3D pose in videos is achieved with CNNs in [23].

One particular recent state-of-the-art method [26] uses a combination of convolutions and upsampling originally proposed in [24] to generate confidence heatmaps, directly identifying the approximate locations of joints in a 3D space. The shift from coordinate regression to heatmap prediction has significantly improved pose estimation results from CNNs.

Within the realm of athletically-focused pose estimation, the work in [38] applies a 2D golf club tracker to infer the 3D pose of a golfer during their swing from a monocular video. Golf swing pose estimation is further explored in [25] with the collection of a dataset of golf swings with single depth cameras and using it to develop an efficient model architecture for estimating the pose of the golfer throughout the swing. Besides golf, the work in [13] includes a method for recovering 3D skeletons from low-resolution soccer videos through a matching process between different sources of motion video.

2.3 3D Pose Estimation Datasets

The landscape of existing 3D pose estimation datasets (i.e. sets of images with corresponding 3D ground truth joint locations) is sparse, likely due to the inherent difficulty of the collection process. The two primary datasets that are used for reporting performance baselines are Human3.6m [17] and HumanEva [35]. Both were collected using camera based motion capture systems in studio environments.

2.3.1 HumanEva

The HumanEva dataset [35] consists of approximately 50,000 frames of 3D motion capture data paired with video from 7 camera viewpoints. The commercial ViconPeak motion capture system [7] is used, which employs on-body reflective markers and multiple calibrated cameras for recovering the 3D positions of the markers. For camera calibration, the extrinsic parameters of the cameras are calculated semi-automatically by waving a single reflective marker through the air in view of the video and ViconPeak cameras. The tracking of the reflective marker in both 3D space by ViconPeak and 2D by the cameras allows for efficient calculation of the extrinsic camera parameters for each camera.

For the creation of HumanEva, 4 actors performed 6 different actions in a motion capture studio, with 3 trials for each action. In the interest of capturing natural-looking motion video, the reflective markers were placed on the actor’s clothing as opposed to a motion capture bodysuit. This results in a slight decrease in location accuracy, as a trade-off for including the natural visual quality of common clothing.
2.3.2 Human3.6m

Human3.6m [17] consists of approximately 3,600,000 frames of 3D motion capture data, paired with video from 4 different viewpoints. 11 professional actors (chosen to maximize variation of body mass index) were hired to perform long-form actions from a set of 15 different scenarios, such as eating or waiting. The collection process for Human3.6m is very similar to that of HumanEva, with some key additional details – 3D body scans and depth cameras were included for adding an extra modality to the data. Besides the 4 video cameras and 10 VICON motion capture cameras, one time-of-flight depth sensor was added on top of a video camera.

Also included in the Human3.6m data are human bounding box annotations, and background segmentation. A small subset of the diverse poses, actors, and outfits included in the Human3.6m dataset can be seen in figure 2.5.

2.3.3 MPI-INF-3DHP

MPI-INF-3DHP [22] is a newer dataset that consists of over a million frames of 3D motion capture data with video. The videos are recorded in three separate environments: a standard studio, a green screen studio, and one outdoor environment. The dataset includes actions from 8 subjects, taking actions from a set of 8 activity sets each lasting about 1 minute.

The dataset is further augmented by including viewpoints from 14 cameras at different angles and heights, and by providing body segmentation masks for automatically altering clothing in post-processing.
Figure 2.5: Examples of frames and poses from the Human3.6m dataset [4].
Chapter 3

Dataset

In this chapter, the proposed dataset will be fully outlined, in terms of its structure, collection, and manual annotation.

To summarize, the dataset is composed of a large number of videos of people performing athletic actions, in which the position of their joints in 3D space is known. In order to train a model for accurately estimating the joint locations of a person in 3D, a large amount of this data is required. While the current datasets are useful for the purpose of benchmarking new algorithms, their restriction to studio locations or only a single outdoor environment limits their utility for general purpose applications. Furthermore, if the domain of actions for an application is known a priori, e.g. for sports, then it is sensible to design a dataset with a focus on those actions. Thus, the collection of a motion capture dataset for athletic motions in a variety of locations is motivated.

3.1 Structure

With the planning of a dataset for pose estimation, there are four variables to consider for each data instance: actor, location, action, and number of takes. In order to construct a robust dataset, we will consistently vary actors, move between different locations, and include multiple different actions. Actors were all volunteers, all of whom signed consent forms confirming their willingness to appear in the dataset.

Considering the intended application of the resulting pose estimation system, the actions will be geared towards golf and baseball. Thus, the actions consist of swinging golf clubs and baseball bats, putting, and pitching. The intuition behind approaching a small number of golf+baseball movements is that athletes involved in these sports are likely concerned with fine-grained details of their movement, which may be extracted with a 3D pose estimation algorithm. Further details on the movements can be seen in table 3.1.

The dataset consists of actions from 45 people, in 22 locations. The locations are each briefly described in table 3.2. Not every location selected is sport inclined, since it is more important that we have location variety than including golfing greens or baseball fields.
<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Golf Swing 1</td>
<td>Subjects performed a swing with a standard golf club.</td>
</tr>
<tr>
<td>Golf Swing 2</td>
<td>Subjects performed a swing with a heavier golf club (driver).</td>
</tr>
<tr>
<td>Putting</td>
<td>Subjects performed a golf putt with a putter.</td>
</tr>
<tr>
<td>Baseball Swing</td>
<td>Subjects performed a swing with a wooden bat.</td>
</tr>
<tr>
<td>Baseball Pitch</td>
<td>Subjects performed a baseball pitch.</td>
</tr>
</tbody>
</table>

Table 3.1: Table outlining the actions included in the dataset. Each action was performed up to 8 times at different angles.

specifically. For each take of each action, the subjects were asked to add variation to their movements, in the interest of providing the dataset with as much diversity as possible. Effort was made to include variation in lighting, clothing, and distance from camera.

3.2 Hardware

3.2.1 Motion Capture

For obtaining 3D ground truth joint locations, we use the XSens MVN BIOMECH Awinda system [8]. The Awinda uses a set of inertial measurement units (IMUs) to calculate accurate joint locations. 17 IMUs are strapped to various locations on the actor’s body as in figure 3.1, and after a calibration process, the locations of 23 joints are wirelessly recorded live to a computer. The accuracy of the XSens system has been validated against the performance of camera-based motion capture [40]. The setup process takes a few minutes per person, and the calibration process takes a few seconds. The MVN records motion capture data at a rate of 60hz. An example of a frame of data output from the Awinda can be seen in figure 3.2.

There are a variety of other existing motion capture systems available, including camera-based [7] and sensor-based approaches. Since environment variety is one of the most fundamental qualities of the dataset, using a camera-based motion capture studio, such as VICON, is not an option. The Awinda is chosen for its high degree of portability.

A key drawback of the use of the MVN Awinda is the lack of an anchor between the camera coordinate system and the MVN’s internal coordinate system, which is generally present in camera-based motion capture systems. Without this anchor, the MVN Awinda’s estimated joint positions can drift from their true positions. Thus, in order to maintain high accuracy throughout all trials, the MVN Awinda is re-calibrated after every motion.

3.2.2 Cameras

To accompany the 3D ground truth joint locations, the video of each motion is captured using 3 GoPro Hero Session cameras [2] mounted on tripods. The cameras each capture a separate viewpoint of the action. The cameras are controlled remotely and simultaneously...
<table>
<thead>
<tr>
<th>Location</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Outdoor park 1</td>
</tr>
<tr>
<td>2</td>
<td>Outdoor park 2</td>
</tr>
<tr>
<td>3</td>
<td>Tennis court 1</td>
</tr>
<tr>
<td>4</td>
<td>Basketball court</td>
</tr>
<tr>
<td>5</td>
<td>Study room</td>
</tr>
<tr>
<td>6</td>
<td>Outdoor park 3</td>
</tr>
<tr>
<td>7</td>
<td>Balcony</td>
</tr>
<tr>
<td>8</td>
<td>Backyard 1</td>
</tr>
<tr>
<td>9</td>
<td>Lecture Hall</td>
</tr>
<tr>
<td>10</td>
<td>Outdoor park 4</td>
</tr>
<tr>
<td>11</td>
<td>Indoor lounge</td>
</tr>
<tr>
<td>12</td>
<td>Tennis court 2</td>
</tr>
<tr>
<td>13</td>
<td>Fitness studio</td>
</tr>
<tr>
<td>14</td>
<td>Field 1</td>
</tr>
<tr>
<td>15</td>
<td>Living room</td>
</tr>
<tr>
<td>16</td>
<td>Indoor recreation hall</td>
</tr>
<tr>
<td>17</td>
<td>Backyard 2</td>
</tr>
<tr>
<td>18</td>
<td>Front yard</td>
</tr>
<tr>
<td>19</td>
<td>Gymnasium</td>
</tr>
<tr>
<td>20</td>
<td>Field 2</td>
</tr>
<tr>
<td>21</td>
<td>Backyard 3</td>
</tr>
<tr>
<td>22</td>
<td>Field 3</td>
</tr>
</tbody>
</table>

Table 3.2: Brief descriptions of all the locations included in the dataset.

Figure 3.1: The XSens MVN Awinda, shown here on a mannequin. [8]
Figure 3.2: 3D skeleton created using joints from the MVN Awinda.

[3], and capture 1080 × 1920 resolution video at 30fps. An example of a frame from a camera in a recording session can be seen in figure 3.3.

The GoPro was chosen for its low price and versatility. Additionally, since the GoPro does not need to be re-focused for new environments, the internal camera parameters remain the same across all trials. The portability of the GoPro camera is achieved with a wide-angle lens, which unfortunately creates a significant fisheye distortion effect. This effect can be removed by using camera calibration techniques for internal camera parameter estimation [1].

3.3 Collection Process

The collection process is organized into a number of recording sessions. A session proceeds as follows:

1. The subject and coordinators arrive at the collection location
2. The subject’s body dimensions are recorded and entered into the XSens BIOMECH software
3. The GoPro cameras are set up on tripods in a semicircle, as in figure 3.4, and synced to a GoPro wi-fi remote
4. The 17 MVN Awinda IMUs are strapped to the subject’s body by the coordinators
5. The system is calibrated, with the subject standing still in an N-pose, as seen in figure 3.5.
6. The actor performs an action up to 8 times, facing in a different direction each time, and their movement is captured by the XSens software.

7. Steps 5 and 6 are repeated for each action.

8. The IMUs are removed from the subject and the camera setup is fully dismantled.

3.4 Camera Calibration

In order to fully prepare the data for processing by a pose estimation model, an explicit correspondence must exist between the 2D RGB video data and the 3D motion capture data. This correspondence can be calculated using camera calibration techniques defined in [41] and [37]. More specifically, given a point \((x_r, y_r, z_r)\) in real-world 3D coordinates, the corresponding point \((x_p, y_p)\) in pixels on the camera plane must be recoverable. For a pinhole camera without distortion, this correspondence is defined as follows,

\[
z_c \begin{pmatrix} x_p \\ y_p \\ 1 \end{pmatrix} = \begin{pmatrix} \alpha_x & \gamma & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{pmatrix} (R \; T) \begin{pmatrix} x_r \\ y_r \\ z_r \end{pmatrix}
\]

where \(\alpha_x, \alpha_y, \gamma, x_0, y_0\) represent the internal camera parameters, \((R \; T)\) is a 3x4 matrix defining the external camera parameters (rotation and translation, respectively), and \(z_c\) is an arbitrary scaling factor which will be assumed to be equal to 1. \(\alpha_x\) and \(\alpha_y\) represent the focal length i.e. the distance between the camera focal point and the sensor, in pixels.
Figure 3.4: Diagram of the camera setup. All three cameras are GoPro Hero Session cameras mounted on tripods, and synced to a GoPro Wifi Remote.

Figure 3.5: The N-pose used for calibrating the MVN Awinda. The subject must hold this pose for approximately 5 seconds for each calibration.
Figure 3.6: A frame from a video containing a planar calibration pattern. The points in this pattern are used to optimally estimate the intrinsic camera parameters.

\( \gamma \) represents the skew of the camera sensor, which is 0, in this case. \((x_0, y_0)\) is the center pixel of the camera sensor, which is (960, 540) for a 1080 \( \times \) 1920 resolution. Even so, this model does not fully describe the camera model for the GoPro cameras, since they suffer from heavy fisheye distortion.

In our case, the fisheye distortion parameters and internal parameters can be calculated in advance once per camera, thanks to the `calibrateCamera()` function available via the OpenCV API [10]. We estimate intrinsic and distortion parameters by displaying a planar checkerboard calibration pattern to each camera, as in figure 3.6, which allows us to remove the distortion in each camera. After intrinsic calibration, we are left with only \( R \) and \( T \) to be directly estimated via optimization techniques defined in [41] and implemented in [10].

Algorithms for estimating the external parameters require a set of points with known correspondence between 2D and 3D. In [41], the technique generally assumes that these points are in a planar calibration pattern i.e. lying along the same plane in 3D space. In our case, the correspondence is along the subject’s body, giving a non-planar calibration pattern.

From the MVN Awinda data, we have access to the 3D ground truth locations for the joints, but the joint locations in the 2D camera plane are not known a priori. Thus, annotating the 2D locations of the joints is necessary. However, not every joint needs to be annotated – once \( R \) and \( T \) have been estimated for some frames of the video, their values remain the same for the entire video. This motivates an approach that involves selecting only a sufficient number of 2D points across the frames of the video such that the calibration quality across the entire video is adequate.
3.5 Annotation

The annotation process is split into two parts, as follows:

3.5.1 Time

Existing datasets have the privilege of being able to operate the video cameras and motion capture cameras on the same clock, resulting in no need to temporally align the videos after collection. The MVN Awinda + GoPro setup does not have such a luxury, so the time alignment must be selected by hand. It is important to note that the capture rates of the MVN Awinda and GoPro differ, at 60hz and 30hz respectively. Since the MVN Awinda capture rate is exactly twice that of the GoPro, the assumption can be made that every 2 MVN frames correspond to 1 video frame.

A graphical user interface (GUI) is built for the purpose of making this annotation process user-friendly and efficient. It allows a user to manually inspect the video frames and MVN skeletons individually, in order to locate a frame where they match. Further automatic refinement of time annotations is described below. A snapshot of the GUI can be seen in figure 3.7.

3.5.2 Space

As defined mathematically in the calibration section, each video requires a set of \( \{(x_p, y_p), (x_r, y_r, z_r)\} \) pairs in order to estimate the external camera parameters \( R \) and \( T \). From the raw output of the MVN Awinda, we have access to 23 \((x_r, y_r, z_r)\) points per frame, and hand annotation is required to accurately pinpoint the corresponding \((x_p, y_p)\) points in the image.

A GUI is built, similar to the one for time annotation, that allows a user to efficiently select locations of joints with corresponding ground truth in MVN Awinda data. The annotator is shown a random frame and asked to annotate a random joint, and the joint can be skipped if not visible from occlusion. Randomness is chosen for two reasons. First, it removes the necessity of the annotator to decide which joints to annotate, allowing for quicker annotations. Secondly, it ensures a relatively uniform distribution of calibration points across the entire video. An image of this GUI can be seen in figure 3.9.

3.5.3 Refinement and Alignment

Every individual video needs to be accurately annotated in time and space in order to be properly calibrated. Since both calibration processes rely on human judgement, it can be prone to small errors. Thus, a method to refine and evaluate the quality of the annotations is presented.

For a given video aligned with MVN data, consider a set of \( N \) 2D points \( \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\} \), given by hand annotation. For each point \((x_i, y_i)\), there is a corresponding point in 3D space...
Figure 3.7: A snapshot of the graphical user interface used for aligning videos and motion capture data in time. The user can move forward and backward through the video and MVN frame-by-frame to find a single frame that matches in both.

Figure 3.8: Visualization of the alignment between MVN data and video data. A key frame is selected that matches between the two, to align the entire video to MVN frames.
Figure 3.9: GUI used for annotating joints. The user is directed to click on the listed joint on the given image. If the joint cannot be located due to occlusion, the annotator can skip to another joint/image.
Consider re-projection of a 3D point onto the camera plane as a parameterized function $P((x, y, z); R, T)$, defined as follows:

$$P((x, y, z); R, T) = \begin{pmatrix} \alpha_x & \gamma & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} R \\ T \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$ \hspace{1cm} (3.1)$$

where intrinsic parameters $\alpha_x, \alpha_y, \gamma, x_0,$ and $y_0$ remain constant for all cameras, and $R, T$ are extrinsic camera parameters. When $R$ and $T$ are the correct extrinsic camera parameters, each point $(x_i, y_i, z_i)$ is perfectly re-projected to its location in the camera plane. However, in general, these parameters are not known, and the estimates $\tilde{R}, \tilde{T}$ are used.

The quality of $\tilde{R}, \tilde{T}$ can be estimated using the following error $E$:

$$E = \frac{1}{N} \sum_{i=1}^{N} \left\| P((x'_i, y'_i, z'_i); \tilde{R}, \tilde{T}) - (x_i, y_i) \right\|_2$$ \hspace{1cm} (3.2)$$

Equation 3.2 effectively calculates mean pixel projection error between estimated projections and annotated points.

The method for checking the time annotations is comparatively simpler. Imagine that each video frame $F_i$ is associated with an MVN frame $M_j$. If the human annotation was mistakenly placed one frame too early, then the optimal pairing would be $F_{i+1}$ with $M_j$. Consider a function $shift(i)$ which shifts the time alignment ahead by $i$ MVN frames and returns the re-projection error $E_i$ that results from this shift. Thus, the optimal amount of frames to shift, with associated optimal error, is:

$$i^* = \arg \min_{i \in \{-K,K\}} shift(i) \hspace{1cm} E^* = \min_{i \in \{-K,K\}} shift(i)$$ \hspace{1cm} (3.3)$$

where $K$ is the maximum amount of time shift in frames. A visualization of the level of shift required to minimize error can be seen in figure 3.13. It can be seen that a large amount of annotations are not optimal, and the alignment is often behind by a number of frames. The effect of the value of $N$ (i.e. the number of annotated calibration points) against optimal error can be seen in figure 3.12. Figure 3.12 makes it clear that once enough points have been annotated (approximately 40), there is no correlation with the level of error.

If the optimal error $E^*$ for a video is above a predefined threshold value, the video is re-annotated. However, it was often found through trial and error that a failure in calibration would still result in a permissible value of $E^*$. An example of such a failure case can be seen in figure 3.11, where the calibration has clearly failed, yet the error $E^*$ fell below the threshold value. Thus, the thresholding of $E^*$ serves as an initial step for identifying re-annotation, followed by manual inspection.
Figure 3.10: 3D joint locations reprojected onto the 2D camera plane, as a result of a successful camera calibration process.

Figure 3.11: Projected 3D joint locations in a case where the calibration routine has not converged. The right hand is mistakenly projected far away from the subject.
Manual inspection involves re-projecting all 3D points onto the video using the current candidates for $\tilde{R}, \tilde{T}$. In other words, the subject’s skeleton position is overlayed on the video frame, as seen in figure 3.10. It is up to the annotator to decide if the video is indicative of a failed calibration, and if so, the video must be re-annotated.

3.6 Reparameterization

For use with the model defined in [26], the pose data must undergo a reparameterization into a common discretized 3D space. The MVN Awinda represents joint locations in a continuous 3D coordinate space, placing the coordinate $(0,0,0)$ at the subject’s feet at the time of MVN calibration. However, in [26], a 3D skeleton is represented in a discretized $64 \times 64 \times 64$ space, with the first two dimensions corresponding to 2D pixel locations within a bounding box, and the third dimension corresponding to joint depth.

To perform this re-parameterization, bounding boxes and joint depths must be calculated. Given camera parameters $\tilde{R}, \tilde{T}$, and all 3D joint locations $\{(x_1, y_1, z_1), \ldots, (x_J, y_J, z_J)\}$ the bounding box center $c = (c_x, c_y)$ is defined as:

$$
c = \frac{\sum_{i=1}^{J} P((x_i, y_i, z_i); \tilde{R}, \tilde{T})}{J}
$$

where $J$ is the number of joints in the skeleton. Note that equation 3.4 calculates the mean of all projected joint locations. The size of the bounding box is given by the maximum
vertical or horizontal pixel distance between projected points, plus a padding factor $\delta$, which was selected here to be 60 pixels. An example of such a bounding box, with padding, can be seen in figure 3.14. If this bounding box in practice is not known, one could use an object detection algorithm to estimate it from the image.

We calculate joint depth for a set of joints $\{(x_1, y_1, z_1), \ldots, (x_J, y_J, z_J)\}$ as follows. Let $\tilde{R}, \tilde{T}$ be the estimated extrinsic camera parameters for the associated video. We need to begin by calculating the location of the camera in 3D coordinate space. Let the relationship between a point $v_c$ in camera space (i.e. with the camera at $(0, 0, 0)$ facing in the direction of the Z-axis) and $v_w$ in real space be given by the following equation:

$$v_c = \tilde{R}v_w + \tilde{T}$$  \hspace{1cm} (3.5)

where $v_w$ is the point in world space, and $v_c$ is the point in camera space. Since the camera is at point $(0, 0, 0)^T$ in camera space, we set $v_c$ to $(0, 0, 0)^T$ and solve for $v_w$, giving:

$$v_w = -\tilde{R}^{-1}\tilde{T}$$  \hspace{1cm} (3.6)

$$= -\tilde{R}^T\tilde{T} \quad \text{(since $R$ is a rotation matrix)}$$  \hspace{1cm} (3.7)

Now, we define the joint depth of a joint as its distance from the camera’s focal plane. The focal plane is defined as the plane orthogonal to the camera’s optical axis extending
directly ahead of the camera’s sensor. Let the camera’s focal length be $f$, measured in pixels, and let $P_x(j ; R, T)$ be the $x$-position of a projected 3D point $j$. Given the center pixel of the camera sensor, $c = (x_0, y_0)$, and a joint position $j_i = (x_i, y_i, z_i)$, the joint depth $d_i$ is found by the following equation:

$$\theta_i = \arctan\left(\frac{x_0 - P_x(j_i ; \tilde{R}, \tilde{T})}{f}\right)$$

$$d_i = \|\tilde{j}_i - v_w\|_2 \cdot \cos \theta_i$$

The above equation uses the joint’s projection onto the camera sensor, so an accurate camera calibration is crucial to obtain accurate results. A visual representation of this calculation can be seen in figure 3.15.

Following the prior calculations, the 2D joint locations within the bounding box are bucketized to $64 \times 64$ bins to discretize the data. For the joint depths, the pelvis joint is selected as the root joint for the center bucket i.e. 33, and the other joints are discretized around it.
Figure 3.15: Visualization of the joint depth calculation. The angle $\theta_i$ is calculated using the joint’s projection onto the camera sensor.
Chapter 4

Experiments

In the following section, the model used for performing 3D human pose estimation will be outlined, as well as its predecessor. The results from re-training the model on our dataset will be conveyed with a set of descriptive performance metrics and qualitative examples.

4.1 Metrics

Two key metrics are used for evaluation of a pose estimation algorithm: mean per joint position error (MPJPE) and percentage of correct keypoints (PCK). Given the joints in the human 3D representation, MPJPE, often measured in millimetres, is the mean euclidean 3D error of the estimates for the location of all joints across the entire set. When this metric is performed on a 3D skeleton that has undergone an affine similarity transformation (i.e. shifting, rotation, and scaling) to approximately match the ground truth, it is referred to as *reconstruction error*. The PCK metric conveys the percentage of keypoints (i.e. joints) which are within a specified distance threshold of their true locations. As a normalization technique, especially in 2D pose estimation, the PCKh is sometimes used. PCKh is equivalent to PCK with the threshold value set to half of the length of the predicted head segment length.

4.2 Model

The model used here for 3D human pose estimation is defined in [26] and is an extension of the stacked hourglass network defined in [24].

4.2.1 Stacked Hourglass

The stacked hourglass network (SHN) is a convolutional neural network (CNN) that predicts 2D human joint locations given an input image. The key advancement of SHN is that it predicts *heatmaps* instead of coordinates. The intuition behind this development is that convolutional neural networks learn spatial representations of its input – not coordinate-
based. Thus, predicting joint location heatmaps requires less work from the network, since it no longer needs to learn a mapping to a new domain.

An SHN uses a set of downsampling and upsampling operations to generate a heatmap. It uses a set of convolutional and pooling operations to reduce the input image down to a small representative latent space, followed by nearest-neighbour upsampling to increase the resolution to the heatmap size of $64 \times 64$. Convolutional features from downsampling layers are concatenated with the upsampling features to provide intermittent supervision to the heatmap generation. Since the network predicts a heatmap for each joint location, the effective output dimensions are $J \times 64 \times 64$, where $J$ is the number of joints. One instance of this image-to-heatmap mapping is referred to as an hourglass.

Following the success of repeated stacking in other approaches [39] [30], the results of the SHN can be improved by stacking multiple instances of an hourglass, routing the heatmap output of one hourglass as the image input of another. This architecture can be seen in figure 4.1 and 4.2.

The SHN is trained using a pixel-wise mean squared error between the predicted heatmap and the ground truth heatmap. Since the ground truth is originally in pixel coordinates, its heatmap is generated as a 2D gaussian with a standard deviation of 1 pixel, centered at the joint location.

### 4.2.2 Coarse To Fine

The coarse-to-fine (CTF) model is an extension of the SHN for performing 3D pose estimation. One key difference is the form of output – while the SHN outputs a $J \times 64 \times 64$ heatmap, CTF must output a $J \times 64 \times 64 \times 64$ heatmap to predict the joint depth as well as the 2D joint location. An example of a result of the CTF on Human3.6m can be seen in figure 4.3. The extremely high dimensionality of this output representation requires a more intuitive approach to stacking the hourglasses.

The SHN approach to stacking involves having the same dimensionality of output at each hourglass. The success of CTF relies on iteratively increasing the depth of the heatmaps with each stacked hourglass, while keeping the X and Y resolutions constant at $64 \times 64$. 

---

Figure 4.1: A high-level visualization of the architecture of the stacked hourglass network. [24] (Copyright ©2016 IEEE)
Figure 4.2: A more fine-grained visualization of the stacked hourglass architecture. Note the recombination of upsampling features with prior convolutional features, as well as the re-routing of the heatmap output to a new hourglass. [24] (Copyright ©2016 IEEE)

Figure 4.3: A frame from the Human3.6m dataset, with corresponding predicted joint heatmaps and predicted skeletons. [17] (Copyright ©2017 IEEE)
More specifically, the best CTF model stacked 4 hourglasses, using respective z-resolutions of 1, 2, 4, and finally, 64. A visualization of this iterative depth increase can be seen in figure 4.4.

The best CTF model achieves a mean reconstruction error of 51.9mm on Human3.6m, and 24.3mm on HumanEva [26].

4.3 Results

4.3.1 Training Details

For the purpose of training, 20% of the frames from annotated and calibrated data will be used as a validation set, with the remaining 80% being used for training. The validation data is sampled uniformly randomly from the data for 5 subjects. Since the annotation process is ongoing, results will only be reported on the portion of the data that has been annotated. Results will also be shown for a naive baseline, which compares every ground truth to one normative static pose from the validation set. All of the data from a 6th individual subject will be held out for testing.

The code for the CTF model was written using Torch [12] by the original developers of the model, and was adapted for the new dataset. Results are reported for training the model from scratch i.e. with random initialization of weights, and for re-training a model trained on Human3.6m. The model was trained on an NVIDIA GRID K520 GPU [6] with a batch size of 1, for 50,000 iterations, and optimized using RMSProp with a learning rate of 0.0005.

4.3.2 Discussion

The results for the PCK accuracy metric can be seen in table 4.1. On the retrained model with four stacked hourglasses, the reconstruction error for each action is reported in table 4.2. The PCK results for varying threshold values appear sensible – as the threshold is decreased, the accuracy decreases in turn. It is important to note that the PCK threshold should not be close to zero – since the MVN Awinda data is not 100% accurate, we cannot
Table 4.1: Validation PCK values for a set of CTF models and thresholds (in cm).

<table>
<thead>
<tr>
<th>Model</th>
<th>Z Resolution</th>
<th>Pretrained</th>
<th>PCK@10</th>
<th>PCK@5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naive</td>
<td>–</td>
<td>–</td>
<td>0.41</td>
<td>0.16</td>
</tr>
<tr>
<td>1 Stack</td>
<td>64</td>
<td>No</td>
<td>0.72</td>
<td>0.31</td>
</tr>
<tr>
<td>2 Stack</td>
<td>1, 64</td>
<td>No</td>
<td>0.71</td>
<td>0.32</td>
</tr>
<tr>
<td>3 Stack</td>
<td>1, 4, 64</td>
<td>No</td>
<td>0.73</td>
<td>0.32</td>
</tr>
<tr>
<td>4 Stack</td>
<td>1, 2, 4, 64</td>
<td>No</td>
<td>0.73</td>
<td>0.32</td>
</tr>
<tr>
<td>4 Stack (not retrained)</td>
<td>1, 2, 4, 64</td>
<td>Yes</td>
<td>0.08</td>
<td>0.02</td>
</tr>
<tr>
<td>4 Stack (retrained)</td>
<td>1, 2, 4, 64</td>
<td>Yes</td>
<td>0.78</td>
<td>0.40</td>
</tr>
</tbody>
</table>

Table 4.2: Validation and test reconstruction error for each action and actor in millimetres. Annotations are incomplete for subject 5.

<table>
<thead>
<tr>
<th>Action Name</th>
<th>P1</th>
<th>P2</th>
<th>P3</th>
<th>P4</th>
<th>P5</th>
<th>Average</th>
<th>P6 (Test)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Golf Swing 1</td>
<td>85</td>
<td>83</td>
<td>116</td>
<td>75</td>
<td>70</td>
<td>81</td>
<td>94</td>
</tr>
<tr>
<td>Golf Swing 2</td>
<td>75</td>
<td>72</td>
<td>73</td>
<td>84</td>
<td>74</td>
<td>78</td>
<td>98</td>
</tr>
<tr>
<td>Putting</td>
<td>74</td>
<td>71</td>
<td>61</td>
<td>71</td>
<td>NA</td>
<td>69</td>
<td>65</td>
</tr>
<tr>
<td>Baseball Swing</td>
<td>92</td>
<td>71</td>
<td>97</td>
<td>84</td>
<td>NA</td>
<td>88</td>
<td>120</td>
</tr>
<tr>
<td>Baseball Pitch</td>
<td>104</td>
<td>87</td>
<td>77</td>
<td>104</td>
<td>NA</td>
<td>94</td>
<td>119</td>
</tr>
<tr>
<td>Average</td>
<td>89</td>
<td>79</td>
<td>85</td>
<td>84</td>
<td>71</td>
<td>83</td>
<td>103</td>
</tr>
</tbody>
</table>

possibly expect the model predictions to be more accurate than the data itself. Error thresholds for the MVN Awinda data are not available, but for example, if we can only expect the data to be within 5cm of the true joint locations, then the lowest the PCK threshold can be is 10cm.

The extremely low accuracy from the pretrained out-of-the-box model trained on Human3.6m implies that the original model may have overfit to the Human3.6m data, likely as a result of the non-varying environment. The retrained model achieved reconstruction errors on the order of 6cm to 11cm. The retrained model was also seen to achieve high accuracy in a much shorter time than the models trained from scratch, implying that the model significantly benefited from the weights of the pretrained model and took little time to adjust to the new domain.

The test results in table 4.3 show little improvement over the naive baseline with a small threshold of 5cm, but more significant with a larger threshold. The implication here is that the model has some difficulty generalizing to the new subject, and may require a wider variety of annotated data to improve its testing accuracy.

A small set of qualitative results can be seen in figures 4.5 through 4.10. For each one, the ground truth is on the left, and predictions are on the left.

Since results are only reported on the small annotated subset of the data, better generalization would be expected after annotating more data to include in the training set.
Table 4.3: Results on a fully held out test for an individual subject.

<table>
<thead>
<tr>
<th>Model</th>
<th>Test Acc @ 10cm</th>
<th>Test Acc @ 5cm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naive Pose</td>
<td>0.43</td>
<td>0.25</td>
</tr>
<tr>
<td>4 Stack (retrained)</td>
<td>0.65</td>
<td>0.25</td>
</tr>
</tbody>
</table>

Figure 4.5: Example of limbs being accurately located despite heavy occlusion. Ground joint locations are on the left, and predictions are on the right.

Figure 4.6: Successful case of pose estimation mid-swing. Ground joint locations are on the left, and predictions are on the right.
Figure 4.7: Failure case, due to the dataset containing very few poses with raised arms. Ground joint locations are on the left, and predictions are on the right.

Figure 4.8: Mild failure case. The structure of the pose is correct, but joint depths are not entirely accurate. Ground joint locations are on the left, and predictions are on the right.

Figure 4.9: Mild failure case. All joints are accurately located, but the spine joint is completely missed. Ground joint locations are on the left, and predictions are on the right.
Figure 4.10: Success case of estimation mid-swing, with some small discrepancies in arm depth. Ground joint locations are on the left, and predictions are on the right.
Chapter 5

Conclusion

The planning, collection, and post-processing pipeline for an in-the-wild sport-oriented motion capture dataset has been fully relayed and evaluated. A 3D pose-estimation model has been trained on the currently annotated subset of the dataset, and the evaluation results have been shown. At the moment, only a subset of the data has been annotated, so the data annotation process is likely to continue. Considering the scale of the dataset, crowd-sourcing the data annotation process will be a time-efficient option. Once a larger quantity of data is annotated, testing on a held-out set of unseen subjects will effectively convey the model’s ability to generalize.

Future direction will likely include a shift to video-oriented 3D pose estimation techniques [23], with attempts made to impose temporal consistency in predictions. Although lower error is generally attained with single-image methods [26] [21], temporal smoothness over a video can be an attractive feature in commercial applications. Beyond the realm of 3D pose estimation, the movement of a subject throughout the athletic motion can be further analyzed through the lens of biomechanics. Assuming prior knowledge of ideal characteristics of a golf swing, for example, feedback can be given based on the estimated pose.

Regarding the future state of data collection, the objective should be to focus on recruiting actors with experience in golf and baseball. The importance of including data that matches a typical use case of a resulting model cannot be understated – one expects that experienced golfers and baseball players would be inclined to apply the pose analysis to their movements. The inclusion of more actions is also a potential avenue to explore, to expand the model’s capability to more sports.
Bibliography


