A Deep Learning Based Paradigm in 3D Human Pose Detection and Estimation in Multi-View Videos

By

Feiyu Zhang

Fengkai Cheng

Han Zheng

Zhuoting Han

Final Report for ECE 445, Senior Design, Spring 2021

TA: Sophie Liu, Xinyi Xu

21 May 2021

Project No. 14

Abstract

Recent studies have witnessed the successes in applying deep learning approach 3D pose estimation. However, most of the 3D pose estimation models are either monocular or operating on single frames, which may not fully exploit the multi-view videos. To fill this gap, we propose a Bi-LSTM based 3D pose estimation model in multi-view videos, which can refine a 3D pose over an entire sequence from multi-view cameras. Our method outperforms the single frames-based model on dataset like Campus – where we show quantitative improvements – and successfully reconstruct 3D pose videos. By temporal refinement on single frames 3D poses, we effectively encode multi-view pose within a unified 3D framework. Furthermore, we propose a method to use temporal refinement module to improve the existing single frames-based 3D pose estimator. We show that our model can be extensive, which allows other deep learning-based models to retrain on previous estimated 3D pose following our paradigm.

Contents

i

1. Introduction 1

1.1 Problem Statement and Overview 1

1.2 High-level Requirements 2

2. Design 3

2.1 2D Pose Detection Module 3

2.2 Multi-view Matching Module 5

2.2.1 Appearance Matching Sub-module 6

2.2.2 Geometric Matching Sub-module 6

2.3 3D Pose Reconstruction Module 7

2.4 Temporal Refinement Module 8

3. Design Verification 9

3.1 Result and Verification 9

3.1.1 2D Pose Detection Module & Multi-view Matching Module 9

3.1.2 3D Pose Reconstruction Module 9

3.1.3 Temporal Refinement Module 10

3.2 R&V Table 11

4. Costs 12

4.1 Parts 12

4.2 Labor 12

4.3 Grand Total 12

5. Conclusion 13

5.1 Accomplishments 13

5.2 Ethical considerations 13

5.3 Future work 14

References 15

Appendix A Requirement and Verification Table 16

1. Introduction

Human Pose estimation and reconstruction is a widely researched topic in the recent decades. Its main idea is detecting location of people's joints which form a skeleton, and to estimate the posture and movement of human body. Estimating the pose of a human in 3D given an image or a video has recently received significant attention from the scientific community. The main reasons for this trend are the ever-increasing new range of applications (e.g., human-robot interaction, gaming, sports performance analysis) which are driven by current technological advances [1].

1.1 Problem Statement and Overview

Although recent approaches have reported remarkable results in 3D pose estimation from static images, it remains an unsolved problem in continues-time videos. This is because the time-varying overlaps of human bodies in consecutive video frames impose several challenges in detecting the joints from human bodies, which are not fully addressed by existing methods.

The objective of this project is to propose a 3D Pose Estimation paradigm for video setting via leveraging machine learning and optimization technique. In particular, we will first use Neural Networks (NN) to detect 2D human bodies (poses) from video clips captured by our multi-view camera system. The detected 2D poses are indicated by a set of bounding boxes. Then we apply multi-view matching algorithm to cluster the detected 2D poses in the resulting bounding boxes. Next, 2D to 3D attitude reconstruction is accomplished by some mature methods, such as 3D pictorial structure (3DPS) based model [2]. In the end, the temporal refinement module will utilize the information from time-series and optimize our results. If four main modules mentioned above function properly, all high-level requirements will be achieved.

The multi-way matching algorithm aims at finding 2D poses of the same person in a group of videos clips captured by several cameras (our camera system). For example, there are five people (labeled in 1 to 5) in the room, and we have three cameras shooting from different directions. Then the multi-way matching algorithm matches the 2D pose of person 1 in video from camera 1 to his 2D poses in videos from camera 2 and camera 3. The matched 2D poses of person 1 are categorized by bounding boxes of a specific color (a color corresponds to a labeled person), and 2D poses inside the bounding boxes with same color will be cut-out from the video for 3D pose reconstruction.

A schematic of our design is shown in figure 1.1.

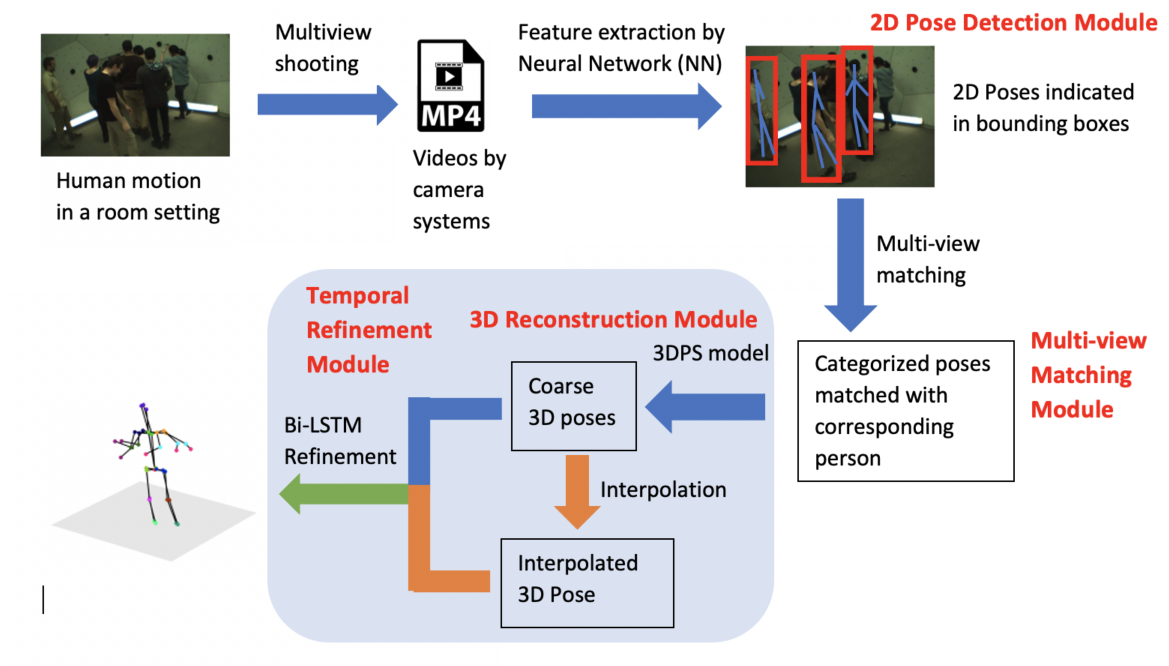
Human motion in a room setting
Multiview shooting
MP4
Videos by camera systems
Feature extraction by Neural Network (NN)
2D Pose Detection Module
2D Poses indicated in bounding boxes
Multi-view matching
Multi-view Matching Module
Categorized poses matched with corresponding person
3DPS model
3D Reconstruction Module
Coarse 3D poses
Interpolation
Interpolated 3D Pose
Temporal Refinement Module
Bi-LSTM Refinement

Figure 1.1 Visual Aid: Design schematic

The designing process of this project consists of programming environment configuration (setting up parallel-computing-based programming platform CUDA (Compute Unified Device Architecture) and machine learning packages such as TensorFlow in Linux system), neural network design (adopting convolutional layer and recurrent layer etc.) and architecture validation (finding the optimal size of network layers, optimal layer concatenations etc.), multi-view matching algorithm implementation, and experiment and demonstration.

As for the metric for evaluating our design, we introduce the Mean Per-Joint Position Error (MPJPE) proposed in [12]. MPJPE is the average Euclidean distance between the location of real-life joints on human bodies and the location of predicted joints on 3D pose model. As mentioned in [7], the MPJPE is as low as 150 (the lower the better) on the dataset Human3.6M (an open-source video data set). Considering we will implement a model that can detect videos instead of photos, the error will be higher. We plan to have similar reconstruction error on the promising datasets like CMU Panoptic [12], Human3.6M and TotalCapture.

1.2 High-level Requirements

- Number of joints in 3D pose of every person should be correct (at least 13, the least number to represent a human 3D pose).
- We expect a MPJPE to be 200(±15) mm.
- Total number of human bodies predicted in 3D pose model should equal to the true number of human bodies in video clips.

2. Design

2.1 2D Pose Detection Module

This 2D pose detection module serves to produce 2D locations of people in each view, and every detected human pose will be labeled with box, which is called “bounding box”. For example, in the figure below, a walking man is detected and marked with some joints to represent his pose. Our result will be a bit different from figure 2.3, because we use bounding box to label each human pose.

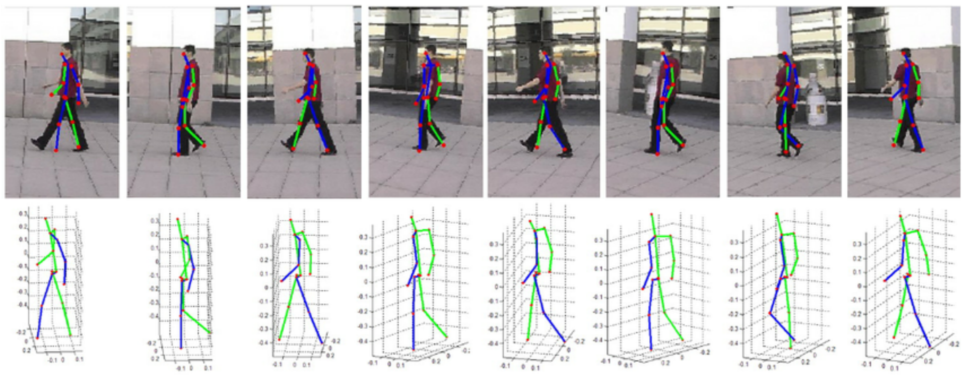

Figure 2.1 Example of 2D Pose Detection Module [9]

Our approach will be using a Cascaded Pyramid Network (CPN) [14] pretrained on MSCOCO [10] dataset for 2D pose detection in images to finish this task. CPN achieved the best 2D pose estimation accuracy in Face++ Keypoint Detection on COCO competition in 2018. The overall framework of CPN adopts a top-down detection strategy. First, a convolution neural network is used as a human detector to generate bounding box. Then, the CPN regresses the key joints of the human body for each detected bounding box, and then outputs the detected 2D poses. The paradigm of 2D pose detection using CPN is shown in figure 2.4.

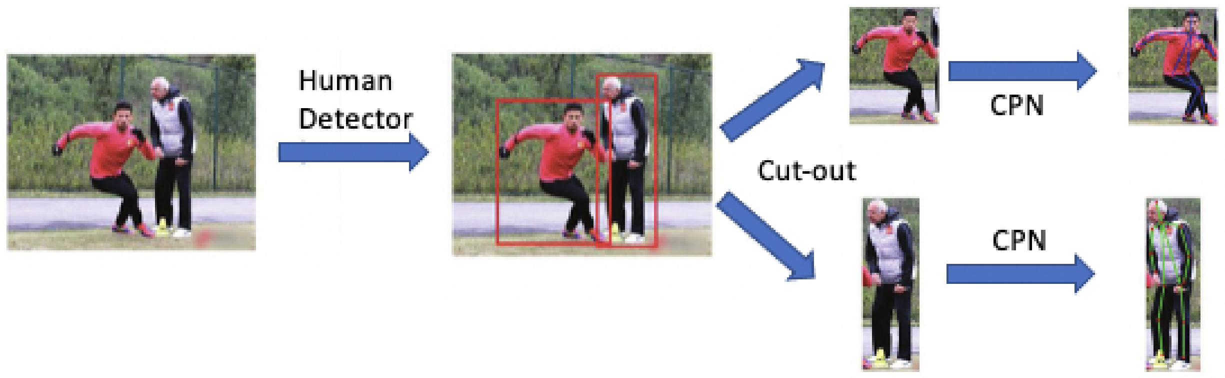

Figure 2.2 2D Pose detection via CPN

As for the human detector which generates the bounding box before CPN, I choose the Mask-RCNN model proposed by Facebook AI Research Team in [4], which is the state-of-art and the most cited object detector. The detailed description of Mask-RCNN model is omitted in this report and one should refer to [4] for further reading.

The CPN consists of two parts: GlobalNet and RefineNet are shown in the block diagram in Figure 1.2. GlobalNet carries out crude extraction of easily detected (visible) key joints such as head and arms (in fact, it is to use the network architecture of a usual convolutional neural network to return a heatmap of visible key joints), while RefineNet explicitly addresses the 2D joints that are hard to detected, such as hips. For those "hard" joints, the context can be used to predict (i.e., the information of multiple receptive fields can be integrated). In order to improve the efficiency and keep integrity of information transmission, our RefineNet transmits the information across different levels and finally integrates the information of different levels via up-sampling and concatenating as HyperNet [3]. An illustration of CPN from [3] is given in figure 2.5.

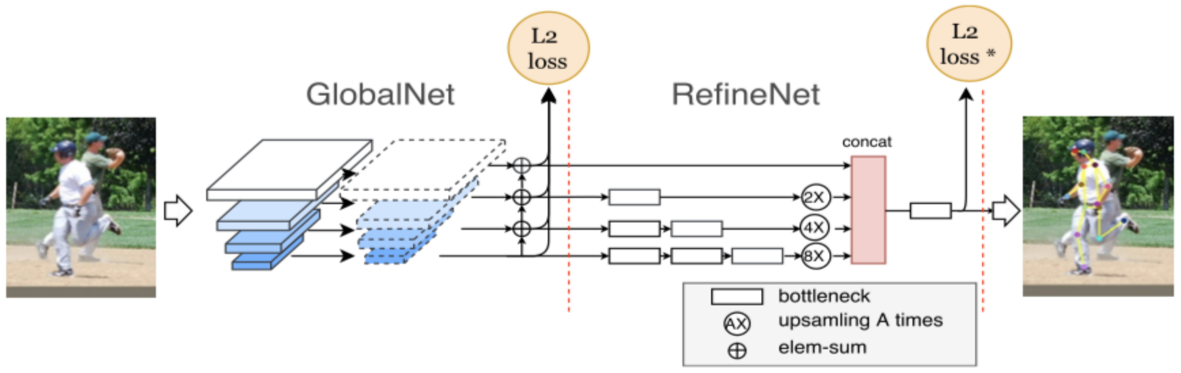

Figure 2.3 Cascaded Pyramid Network, adopted from [3]

For the network model for GlobalNet, we choose the wildly adopted convolutional neural network for visual recognition called Residual Network (ResNet) [5]. In the process of information transmission, traditional convolutional network or fully connected network will more or less have problems such as information loss. At the same time, training a deep fully-connected network also leads to gradient vanishing or gradient explosion, resulting instability in training process. ResNet solves this problem to some extent by directly routing the input information to the output to protect the integrity of the information. The whole network only needs to learn the difference between the input and output, simplifying the learning objective and difficulty. The comparison between a deep fully-connected (plain) network and ResNet is shown in Figure 2.6. The big highlight with ResNet is that there are many bypasses that connect the input directly to the next layer. Such structure is also called Shortcut or Skip Connections.

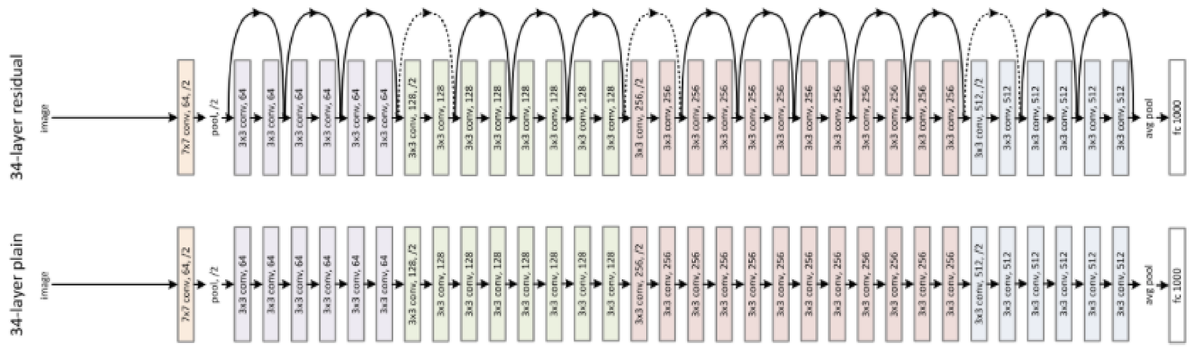

Figure 2.4 Comparison between ResNet and plain network

For specific model design of ResNet, we first implement a 34-layer ResNet backbone in personal computer, however, the memory overflowed due to large amount of network parameters. Then with the assistance of other groupmates, we transfer the model to the campus High Performance Computer (HPC), and we successfully pushed the size of the ResNet model to 50 layers to achieve better 2D joints detection. To have higher training efficiency, we first download a pretrained ResNet model and then fine-tuned the model on our dataset on HPC. We also used multi-core GUP on HPC to accelerate training. One should note the performance of ResNet on 2D joint detection of has already been investigated in [5].

2.2 Multi-view Matching Module

Match the detected 2D poses across views, i.e., find in all views the 2D poses belongs to the same person. We will use a discriminative metric to measure the likelihood that two 2D poses belong to the same person and a matching algorithm to establish the correspondences of across multiple views. The initial solution to this problem is based on: (1) find the discriminative visual difference between combination of 2 poses from all views, and to determine if each pair of view belongs to the same person; (2) match the poses to each person based on the result in the previous step.

The solution can be described as: Suppose there are V cameras in the scene and p_i detected bounding boxes in view i. For a pair of views (i, j), the discriminative scores can be calculated between the two sets of bounding boxes in view i and view j. Same as [6], we use $D_{ij} \in R^{p_i \times p_j}$ to measure discriminative visual difference, whose elements represent the discriminative scores. The correspondence between the two poses being estimated is expressed by a partial permutation matrix $P_{ij} \in \{0, 1\}^{p_i \times p_j}$. Then the solution should take D_{ij} as input and output the optimal P_{ij} that corresponds to the minimum discriminative scores. It is also noticeable that due the fact that a pose in one view should be only matched to the unique one pose in any another view, thus P_{ij} must satisfy:

$$0 \le P_{ij} 1 \le 1,\ 0 \le P_{ij}^{T} 1 \le 1$$

We follow the formular below to compute each element (discriminative scores) in D_{ij},

$$D_{ij} = \sqrt{D_{ij}^{a} \circ D_{ij}^{g}}$$

where D_{ij}^{a} denotes the appearance discriminative score given by the Appearance Matching Sub-module in Figure 2.1, D_{ij}^{p} denotes the geometric discriminative score given by the Geometric Matching Sub-module in Figure 2.1, and $\circ$ denotes the elementwise multiplication. The matching algorithm takes the smallest discriminative score of each pose and identify the another pose that this score corresponds to. Then these two poses should belong to the same person. Then the optimal P_{ij} is given by:

$$P_{ij}(\cdot) = \{1,\ if\ i, j = argmax_{i,j}(D_{i})\ 0, \qquad\qquad else$$

And P_{ij} is refined using the algorithms proposed in [6].

2.2.1 Appearance Matching Sub-module

2D pose within bounding boxes is feed to another CNN and the out-put vector of the last layer is taken as the feature of the input pose. We compute the Euclidean distance between two feature vectors and normalize this distance using sigmoid function to range (0,1) as the appearance discriminative score of these two poses. Two poses from the same person should have near-zero appearance discriminative score.

In particular, D_{ij}^{a} is determined by: (1) Feed 2D poses to another 34-layer ResNet and take the out-put vector of the last layer as the feature of the input pose. (2) Compute the Euclidean distance between two feature vectors and normalize this distance using sigmoid function to range $(0, 1)$ as the appearance discriminative score of these two poses.

2.2.2 Geometric Matching Sub-module

We measured the average point-to-line distance between joints in one pose to the epipolar line associated with these joints in another pose. Then normalize this distance using sigmoid function to range $(0, 1)$ as the appearance geometric discriminative score.

$$L(x_i, x_j) = \frac{1}{2N}\sum_{n=1}^{N} d(x_i^n, l_{ep}(x_j^n)) + d(x_j^n, l_{ep}(x_i^n))$$

$$D_{ij}^p = sigmoid(L(x_i, x_j))$$

Where N denotes the total number 3D joints on an actor's body, $d(\cdot,\cdot)$ denotes the point to line distance, x_i^n denotes the 2D location of the n-th joint of pose i, and $l_{ep}(\cdot)$ the epipolar line corresponding with the joint in another view.

The intuition of defining D_{ij}^p is based on a joint on a body in the first camera view should lie on the epipolar line (the straight line of intersection of the epipolar plane) with the image plane. as sociated with its correspondence in the second camera view, which can be explained in the figure 2.7:

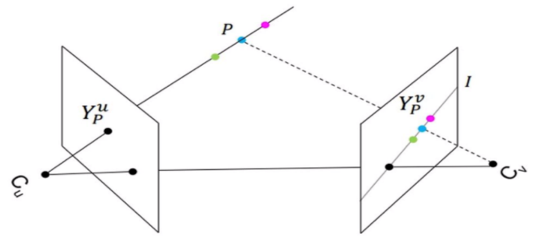

Figure 2.5 Explanation of epipolar line and projection

An image point Y_P^u back-projects to a ray in 3D defined by the camera C_u and Y_P^u. This line is imaged as I in the camera C_v. The 3D point P which projects to Y_P^u must lie on this ray, so the image of P in camera C_v must lie on I.

2.3 3D Pose Reconstruction Module

Given the estimated 2D poses of the same person from different views, we can directly reconstruct 3D pose by simple triangulation. However, the error in 2D pose estimation may significantly degrade the reconstruction process. So we are going to use 3D pictorial structure(3DPS) [13] model to reconstruct the 3D pose. 3DPS is kind of similar to search algorithm, which search in the 3D space for the points with highest posterior probability that may occur. The theorem is like following:

Let $S = \{s_i | i = 1, 2,.. N\}$, where $s_i \in R^3$ denotes the predicted 3D position of joint i on the reconstructed 3D pose. If we have 2D poses from total of M camera views, ie. $R = \{r_j | j = 1, 2,.. M\}$. Then we have the posterior distribution of 3D poses can be written as:

$$P(R) \propto \prod_{j}^{V} \prod_{i=1}^{N} P\left(\pi_j(s_i)\right)$$

Where $\pi_j(s_i)$ denotes the 2D projection of s_i the view of camera j. We get $P\left(\pi_j(s_i)\right)$ by feeding the grouped 2D poses from the Multi-view Matching Module to the Neural Network Model proposed in [14], which determines the 2D spatial distribution of each joint. Then the optimal 3D pose reconstruction S^* is achieved by maximizing $P(R)$ via searching among all possible predicted joints placement in 3D space.

2.4 Temporal Refinement Module

The estimated 3D poses can be generated from 3D pose reconstruction (3DPS) module. However, the results might be incorrect under some certain circumstances, for example, overlapping. In this temporal refinement module, we will apply Bidirectional Long Short-Term Memory (Bi-LSTM) to utilize the time-series information. The Bi-LSTM is a recurrent neural network which stops gradients from vanishing during training process, and we feed the pose sequence to Bi-directional-LSTM [5] and take the hidden state at the last time step as the refined pose. As shown in figure 2.8, to optimize one 3D pose, we will have the poses from its adjacent frames as the input.

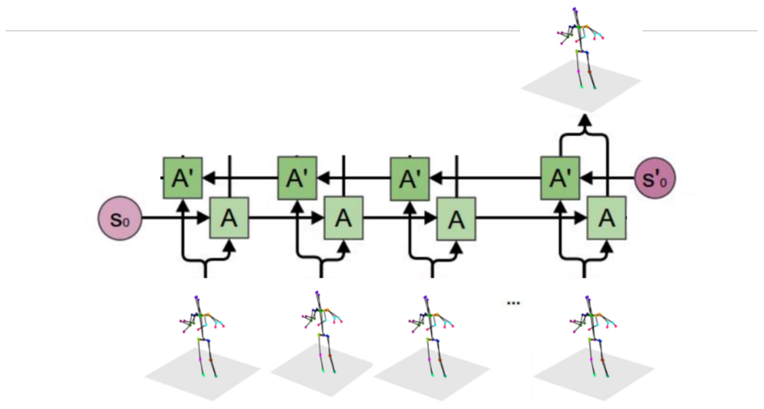

Figure 2.6 Bidirectional Long Short-Term Memory

For the training loss function, we use mean per joint precision error (MPJPE), and the formula can be written as:

$$L = \frac{1}{k}\frac{1}{n}\sum_{i=1}^{K}\sum_{i=1}^{N}\|\hat{y}_i - y_i\|$$

Here, k represents the number of samples in one training batch, n is the number of joints. $\hat{y}_i$ is the ground truth 3D poses, while y_i is the predicted poses after the refinement of Bi-LSTM.

3. Design Verification

3.1 Result and Verification

3.1.1 2D Pose Detection Module & Multi-view Matching Module

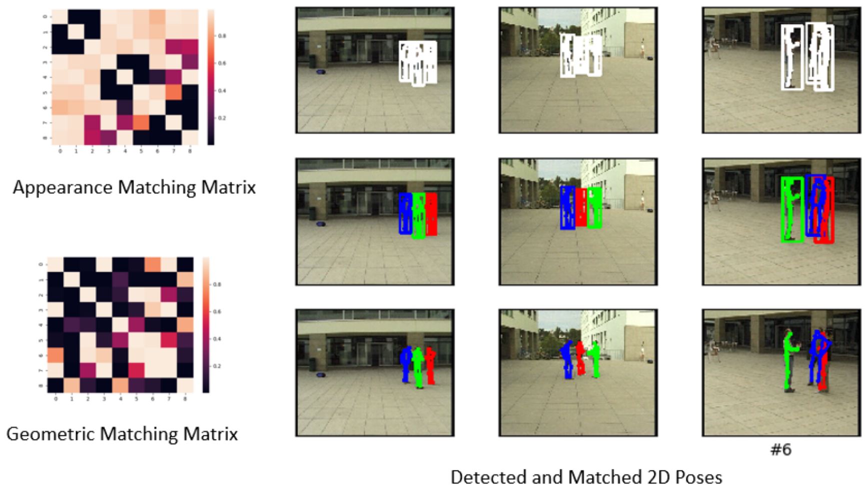
Appearance Matching Matrix
Geometric Matching Matrix
#6
Detected and Matched 2D Poses

Figure 3.1 The result of 2D pose detection module (first row on the right), multi-view matching module (second row on the right), result without bounding box (third row on the right) and matching matrices (on the left).

3.1.2 3D Pose Reconstruction Module

The 3D pose reconstruction module generates 3D poses successfully. However, the results might be incorrect due to overlap. As you can see in Figure3.2, the green man's arm is missing.

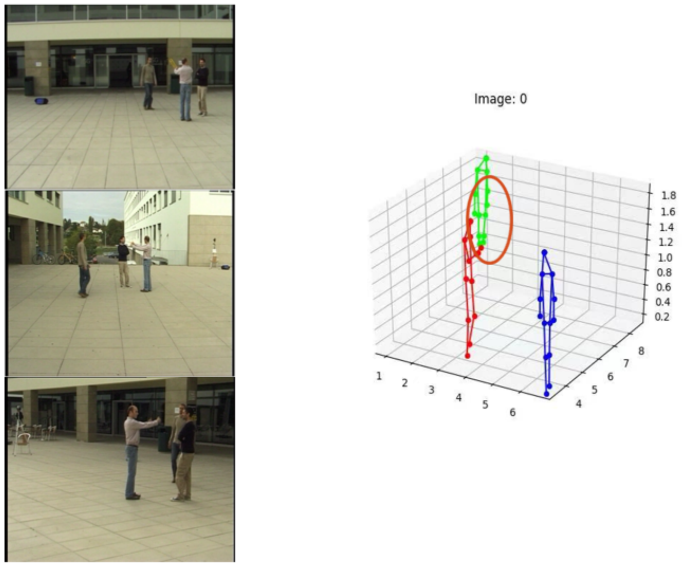

Figure 3.2 The original images and the result of 3D Pose Reconstruction Module

3.1.3 Temporal Refinement Module

After the temporary refinement module, the 3D poses are refined even there is overlap, just like in Figure 3.3.

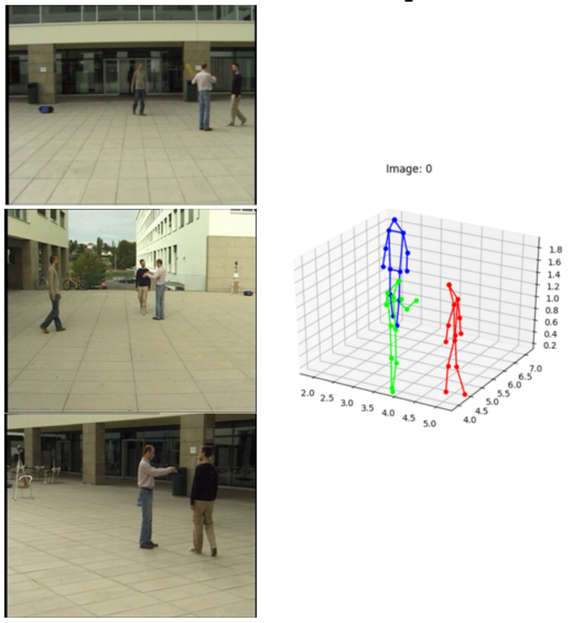

Figure 3.3 The original images and the result after temporal refinement.

3.2 R&V Table

We have verified every sub-module to make sure they meet our requirements. See Appendix A for details.

4. Costs

4.1 Parts

Description	Quantity	Manufacturer	Vendor	Cost/Unit	Total Cost
Webcam	5	Logitech	Amazon	$27.47	$137.35
1TB SSD	1	Western Digital	Amazon	$89.99	$89.99
Colab Pro	3	/	Google	$9.99 /Month	$29.97
Total					$257.31

Table 1 Parts Cost (Our Project mainly based our camera, PC and Colab)

4.2 Labor

Name	Hourly Rate	Hours	Total	Total*2.5
Fengkai Chen	$30	240	$7200	$18000
Han Zheng	$30	240	$7200	$18000
Zhuoting Han	$30	240	$7200	$18000
Feiyu Zhang	$30	240	$7200	$18000
Total				$72000

Table 2 Labor Cost

4.3 Grand Total

Section	Total
Labor	$72000
Parts	$145.43
Grand Total	$72257.31

Table 3 Grand Total Cost

5. Conclusion

5.1 Accomplishments

Our accomplishments can be concluded into three points. First, we proposed a temporal model to refine the estimated static multi-view 3D poses, which can be used with 3D poses generated by any existed model with high adaptability and change into video poses. Secondly, while existing approaches are mostly either single-view or single frame, our work explores the possibility of multi-view and temporal 3D pose estimation, with Bi-LSTM integrated into our model. Thirdly, we achieved the reconstruction accuracy such that the Mean Per Joint Position Error (MPJPE) is lower than 50mm with high reconstruction efficiency.

5.2 Ethical considerations

Our project has several potential safety and ethics concerns. The first concern is network intrusion. Currently we are using campus network to transmit our information and signals. However, every network has a possibility to be attacked, and this rule also applies to our campus network. This is against #7 and #9 of the IEEE Code of Ethics – "the people committing piracy are not properly crediting the work of others, and they could be injuring the copyright holders by sharing content without paying for it." [4] Once the network is controlled, we may lose our control over the whole system, such that our core codes and algorithms may leak. Actually, we do not have a perfect plan for this. Our current solution is that use version control tools, like SVN and git, to store our codes and do not publish it before some sense of agreement is made.

The second concern is the private pictures/video disclosure. The disclosure violates the ACM code of Ethics, #1.6, "Therefore, a computing professional should become conversant in the various definitions and forms of privacy and should understand the rights and responsibilities associated with the collection and use of personal information." [5] Due to the high volume of picture/videos used for network training, saving all data in our personal laptop is not recommended. For convenience in calling data, we plan to store our data on an online server, which may be cyber-attacked and cause data disclosure. To minimize such risk, we suggest shutting down network acceleration software such as Cisco AnyConnect Mobility Client and Express VPN when testing online algorithms.

Besides all the concerns above, we still want to make sure that the model will treat everyone equally. If we use a biased training dataset, like some dataset mostly containing videos/pictures of white people, the model may have worse effects on black, Asian and Hispanic people. If we use a training dataset that mostly involves men moving and acting, this model may have worse effects on women. All these violate the #8 of the IEEE Code of Ethics, “to treat fairly all persons and to not engage in acts of discrimination based on race, religion, gender, disability, age, national origin, sexual orientation, gender identity, or gender expression” [4]. To avoid such things, we will carefully choose our dataset, including the percentage of different races, genders, ages and other tags that may divide people into different groups, to ensure an unbiased development process.

5.3 Future work

Our method is based on stored videos and takes some time to process. The processing time can be cut down so that it can process in real-time. We also want to shoot videos with calibrated cameras, so that we can use our own videos for demonstration. In addition, the dataset we use for training is quite small. We want to apply for the Human3.6M dataset in order to train our model on more diverse data.

References

[1] Sarafianos, Nikolaos & Boteanu, Bogdan & Ionescu, Bogdan & Kakadiaris, Ioannis. 3D Human Pose Estimation: A Review of the Literature and Analysis of Covariates. In Computer Vision and Image Understanding, 2016.

[2] V. Belagiannis, S. Amin, M. Andriluka, B. Schiele, N. Navab, and S. Ilic. 3d pictorial structures for multiple human pose estimation. In CVPR, 2014.

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

[4] Kaiming He, Georgia Gkioxari, Piotr Dollár and Ross Girshick. Mask R-CNN. In ICCV, 2017

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

[6] J. Dong, W. Jiang, Q. Huang, H. Bao, and X. Zhou. Fast and robust multi-person 3d pose estimation from multiple views. In CVPR, 2019.

[7] Belagiannis, Vasileios Amin, Sikandar Andriluka, Mykhaylo Schiele, Bernt Navab, and Nassir Ilic, 3D Pictorial Structures Revisited: Multiple Human Pose Estimation. In IEEE TPAMI, 2015

[8] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-kin Wong, Wang-chun Woo, Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting. In NeurIPS, 2015.

[9] Fast Human Pose Detection Using Randomized Hierarchical Cascades of Rejectors - Scientific Figure on ResearchGate.

[10] T. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Doll´ar, and C. Zitnick. Microsoft coco: Common objects in context. In ECCV, 2014.

[11] Kocabas M, Karagoz S, Akbas E. Self-Supervised Learning of 3D Human Pose using Multi-view Geometry[J]. In CVPR, 2019.

[12] Umar.Iqbal, Pavlo.Molchanov, Jan.Kautz, NVIDIA Research. Weakly-Supervised 3D Human Pose Learning via Multi-view Images in the Wild. In CVPR, 2020

[13] Joo, Hanbyul and Simon, et al. Panoptic Studio: A Massively Multiview System for Social Interaction Capture, IEEE Transactions on Pattern Analysis and Machine Intelligence,2017

[14] ieee.org, IEEE Code of Ethics, 2020. [Online]. Available: http://www.ieee.org/about/corporate/governance/p7-8.html[Accessed: 28- Feb- 2021].7

[15] ethics.acm.org, ACM Code of Ethics, 2020. [Online]. Available: https://ethics.acm.org. [Accessed: 28- Feb- 2021].7

Appendix A Requirement and Verification Table

Table 4 System Requirements and Verifications

Requirement	Verification	Verification status (Y or N)
1. 2D Pose Detection Module a. Generate joints and bounding boxes with precision higher than 90% on CMU Panoptic [11] dataset. b. Every human pose must contain at least 13 joints (the least number to represent a human pose using joints)	1a. We use this module for several different test datasets which are selected from CMU Panoptic dataset, and we check every time if the precision is higher than 95%. 1b. After every test of 1A, we will observe each detected human pose to check if the number of joints is higher than 13.	Y
2. Multi-view Matching Module a. For images from the same person in multi-view, this sub-module should give a diversity score lower than threshold ε (An empirical value which will be set by some pre-testing). b. For images from different persons, this sub-module should give a higher diversity score than ε.	2a. To begin with, we will do some test on small test datasets to determine threshold ε. After determining threshold ε, we will input several images from the same person from our team members to check if the diversity score is lower than threshold ε. 2b. After determining threshold ε, we will input several images from different persons from our team members to check if the diversity score is higher than ε.	Y
3. 3D Pose Reconstruction Module c. Given a set of 2D poses from the same person in different views, 3D Pose Reconstruction Module should be able to generate a single 3D pose in 3D space that has a correct joints number (at least 13).	3a. We will use several test datasets that includes pre-matched 2D poses of the same person, then we use this module to generate corresponding 3D pose. After processing, we will manually check if the result is a	Y

d. Given 2D poses that are matched into different 3D persons, this module should not reconstruct them into a single 3D pose.	single 3D pose with correct joints number. 3b. We use pre-matched 2D poses from different 3D persons as input, then we check if the output will generate an error output or wrongly reconstructed single 3D pose.	
4. Temporal Refinement Module a. The training should converge. b. The model refines the result. c. The model predicts the 3D poses accurately.	4a. The training loss is asymptotically converging (decreasing). 4b. The evaluating error during training is decreasing. 4c. The final mean per joint position error (MPJPE) should be lower than 75 mm.	Y