CO-PLANAR MULTI-PERSON 3D VIDEO POSE ESTIMATION AND LOCALIZATION WITH MOVING-CAMERA BASED SYNTHETIC-REAL DATASET

Author(s) Name(s)

Zhejiang University - University of Illinois at Urbana-Champaign Institute, ZJUI

ABSTRACT

In this work, we demonstrate a algorithm that aims to achieve multi-person 3D pose estimation and localization with moving-camera in video. Meanwhile, because of the scarcity of multi-person datasets, we synthesize a multi-person dataset based on Human3.6m and validated the algorithm on it. The algorithm is two-stage, which is combined with a temporal convolutional network and a iterative regression model. Code are available at https://github.com/ReselfC/Dataexpand.

***Index Terms*—** 3D Pose, multi-person, video, synthetic-real dataset, localization

1. INTRODUCTION

At present, 3D pose estimation methods for single person have been relatively mature, and there are also large and reliable datasets like Human3.6M [1] and HumanEva [2]. But because of the higher production costs, 3D multi-person pose datasets are scarce, so there are not many algorithms that aim to solve pose estimation and localization based on. Meanwhile, considering the accuracy, universality and usage scenarios of the algorithm, we believe that it is more beneficial to implement the algorithm in the form of video rather than simple pictures. We believe that our algorithm can be combined with other 2D pose detection algorithms, and it can be widely used to assist motion capture, posture positioning of characters, and even in some network broadcast.

In view of this topic, there are also some previous partly related work. Significant works like *DeeperCut* to ameliorate the efficacy of multi-person pose estimation [3], *RMPE* to ameliorate pose estimation in the presence of inaccurate human bounding boxes [4], and also a large amount of researches to facilitate multi-person pose estimation in a variety of contexts [5, 6, 7, 8]. There are also a considerable number of researches which can be applied on multi-object datasets for training and estimating but did not [9, 10, 11, 12], because the datasets they use are single-object, like Human3.6M [1], HumanEva[2], a large part of Coco[13] and VGG [14].

In this paper, we first suggest a algorithm that synthesizing the multi-person dataset based on the large single person 3D Pose dataset Human3.6m. And in view of some research gaps and practical needs described above, we propose our algorithm in this paper, which is aims to achieve co-planar multi-person 3D video pose estimation and localization with moving-camera based on our synthetic-real dataset. For this algorithm, the only need to input 2D pose coordinates detected from the video on every frame, while the output is going to be all the joint positions of all the people in the video in camera coordinates on every frame.

Our pose estimation and localization algorithm is a two-stage algorithm. In the first stage, we normalize the 2D pose and then fed it into a temporal convolutional network person by person to get the predicted 3D pose with each person's root as origin. Then in the second stage, we design a iterative regression model. It first regress the root position of each person from the predicted 3D pose information in the last stage, so we get all the joints position in camera coordinates, than regress the ground equation from that and add ground equation constraint back to the first regression to build a iterative loop.

It has been experimentally proved that in comparison to the baseline VideoPose3D, our two-stage model performs better and the size of the model is significantly reduced.

2. DATASET

In this part we take multiple subjects with different actions in the same dataset (e.g. *Human3.6M*) as input, after which we merge the sequences of images into one sequence of images with multiple subjects using the least number of frames in the original sequences for the new dataset. This is followed (in a chronological order) by translational and rotational movements of each subject to ameliorate variety of the new dataset, a special algorithm for eliminating collisions of the shifted subjects (†)[1], and an integrated camera generator for a multi-view solution (†).

2.1. Collision Elimination

In this part we developed an algorithm called SAEIC (Sequential Approach for Eliminating Individual Collisions) and it's shown logically in **Algorithm 1**[2], where n is the num-

[1] Important parts are notated by (†) and will be discussed in details.

[2] We notate the complex part as (∗), which will be illustrated further.

ber of subjects (often less than 10), x represents the number of frames in the new dataset, m stands for the number of vertices utilized (in our case 17), and each vertex has coordinates for 3 dimensions. Evidence has shown that SAEIC can eliminate **strictly 100%** collisions on *Human3.6M*, and it's theoretically proved to be within a computational complexity of $O(n \cdot x)$.

Algorithm 1: SAEIC Algorithm

Data: The original **tensor** of shape $(n, x, 17, 3)$.
Result: The shifted **tensor** of shape $(n, x, 17, 3)$.
for *i* ***from*** *the second* ***to*** *the last subject* **do**
  for *frame* ***in*** *x* ***step*** *3* **do**
    if *this frame has collision* **then**
      find the shift vector for this frame; $(*)$
      append it to the shift vector list;
    end
  end
  find the max shift vector for this subject;
  add it to all frames of this subject;
end
return *the new* ***tensor***;

Like its name, SAEIC utilizes a sequential approach, which means the order for each input subject is substantial. All subjects must be processed one after one, because the shift vector of the latter one must guarantee **all** subjects that are chronologically earlier than it. The algorithm for finding the shift vector is shown in **Algorithm 2**.

Algorithm 2: Shift Vector Calculation Algorithm

Data: i, $frame$, 17 vertices with (x, y) coordinates each.
Result: The shift vector.
while *True* **do**
  for *j in all previous subjects* **do**
    if *i and j have collision* **then**
      calculate the shift vector for *i*; $(*)$
      move subject *i* using the shift vector;
      set a flag;
    end
    if *already the last epoch and* ***no*** *flag* **then**
      break while;
    end
  end
end
return *the shift vector*;

The calculation process of the shift vector is marked with $(*)$. In this process, we only consider vertical shift vectors and ignore all horizontal displacements. The detailed approach is, firstly determine the relative vertical positions of the two bounding boxes of the subjects, then we calculate the value of shift vector in this frame for subject i using a relationship between the highest edge of one subject and lowest edge of another.

2.2. Camera View Generation

In this part we transform the initial *Human3.6M* dataset from the world coordinate into camera coordinate. The dataset was first created with three components: the world coordinate, the 3D camera coordinate, and also the 2D camera coordinate [1]. Our research aims to generate exactly the same varieties to fulfill the tasks of *Human3.6M*.

For generation, our dataset is firstly transformed into a moving 3D camera coordinate by an extrinsic parameter matrix, and is furthermore transformed to a 2D image using an intrinsic parameter matrix with both of the generated datasets (3D and 2D, camera coordinate) being used for the same tasks in *VideoPose3D* [9] in next part of this paper.

The extrinsic parameter matrix introduces the camera movement with a cosine-like trajectory along z-axis and circle-like trajectory on the x-y plane, (see Figire 1) and the camera rotation to focus the center of the camera screen at the center of mass of the subjects in the world coordinate to avoid the subjects leaving the screen (see Figure 2). The intrinsic parameter matrix includes different types of camera (e.g. phone camera, surveillance camera and SLR camera) and their parameters (e.g. imitated tiny shaking and lens distortion). We decide the parameters according to the book *Multiple View Geometry in Computer Vision* [15] and it turns out that the right choice of the parameters magnificently improved the quality of the 2D images.

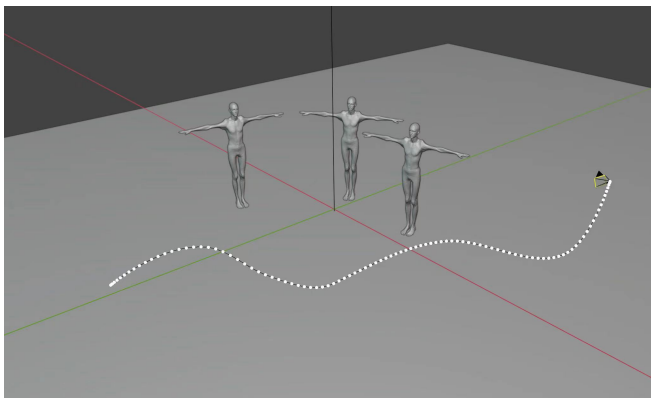

Fig. 1. The camera movement trajectory pattern. There are 3 objects moving in this scenario and the camera follows the predefined path to generate 2D images.

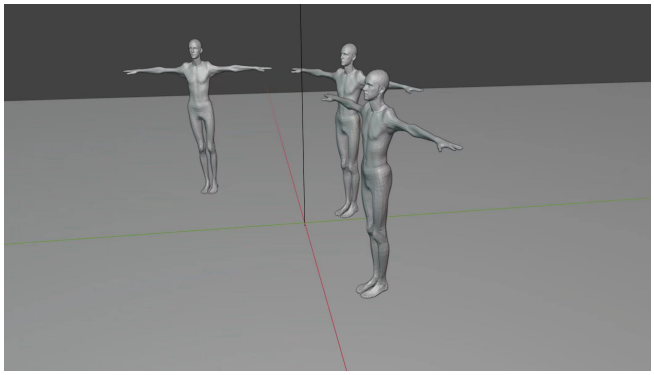

Fig. 2. The camera view of Figure 1 at the right-most watchpoint. The camera always try to include all subjects in its screen to generate a series of images with all subjects in the world coordinate.

3. METHOD

3.1. Temporal Convolutional Network

We use the same temporal convolutional network to get 3D pose estimation from 2D pose input. However, the obtained 3D pose of each person is all about their own root to construct the coordinate system for the origin, and we need to unify them in the camera coordinate system.

3.2. Iterative Regression Model

3.2.1. Initial the Iterative Loop

After temporal convolutional network, we get the 3D pose for each person in different coordinates. To initial the iterative loop, for multi-person scenarios, we can estimate all the rough individual trajectories separately.

Considering the case of single person, denote its root position in camera coordinate at frame t as C_t, number of joints as N_j, 2D pose of joint i as p_i which represented $[x_i, y_i]^T$ in pixels, 3D pose of joint i as P_i which represented $[X_i, Y_i, Z_i]^T$ in real scale. Then we can define the projection error from 2D to 3D as follows,

$$\epsilon_t = \frac{1}{N_j}\sum_{i=1}^{N_j}||\rho(P_{i,t} + C_t) - p_{i,t}||^2, \tag{1}$$

where function ρ is the projection of the pinhole camera, which maps the 3D coordinate into image coordinate for x,y separately,

$$\rho_{x,y}(P) = \frac{f_{x,y}(X,Y)}{Z} + c_{x,y} \tag{2}$$

where f_x, f_y is the focal length of camera in x, y direction. c_x, c_y is the center of the image coordinate in pixels.

Meanwhile, considering that we want the trajectory to be smooth, we also add the temporal constraint when solving C, denote the number of total frame is F, we can define the temporal error containing all frames as follows,

$$\sigma = \sum_{t=2}^{F}||C_t - C_{t-1}||^2 \tag{3}$$

Obviously all the joints have the same trajectory at each frame, then the root trajectory C_t can be estimated by minimizing the projection error,

$$\hat{C} = \arg\min_{C}\sum_{t=1}^{F}\epsilon_t + \lambda_1\sigma, \tag{4}$$

where λ_1 is the parameter to control the strength of temporal constraint. And we built a big matrix operation to solve C.

Because the we synthetic the dataset co-planar, we can use this rough root trajectory C to initial parameters in the ground equation. We select the point identified as the foot and denote as subscript f. We can define the error with the mean distance between each foot and regressed ground as follows,

$$\gamma_t = \frac{1}{N_f}\sum_{f=1}^{N_f}||G(P_{f,t} + C_{f,t}) - 1||^2, \tag{5}$$

where $G_t = [a_t, b_t, c_t]^T$ represents the parameters in ground equation $a_tX_t + b_tY_t + c_tZ_t = 1$.

And so G_t can be solved as follows,

$$\hat{G}_t = \arg\min_{G_t}\gamma_t, \tag{6}$$

So that we get the rough trajectory C and ground parameters G on every frames.

3.2.2. Iterative Loop Regression

Aiming to make trajectory more accurate, we demonstrate a iterative loop regression algorithm.

According to the equations(1)(3)(5)(6), a complete iterative loop i can be calculated as follows in order,

$$\begin{aligned}\hat{C}^{(i)} &= \arg\min_{C}\textstyle\sum_{t=1}^{F}\epsilon_t^{(i)} + \lambda_1\sigma^{(i)} + \lambda_2\sum_{t=1}^{F}\gamma_t^{(i)},\\ \hat{G_t}^{(i)} &= \arg\min_{G_t}\gamma_t^{(i)},\end{aligned} \tag{7}$$

where λ_1, λ_2 is the parameter to control the strength of temporal and ground constraint, and $\gamma_t^{(i)}$ should be calculated as follows,

$$\gamma_t^{(i)} = \frac{1}{N_f}\sum_{f=1}^{N_f}||G^{(i-1)}(P_{f,t} + C_{f,t}^{(i)}) - 1||^2, \tag{8}$$

4. EXPERIMENTS

In this part we completed the estimation of the capabilities of a dataset, and we visualized the result with 6 subfigures (3 for 3D view and 3 for 2D view) in Figure 3.

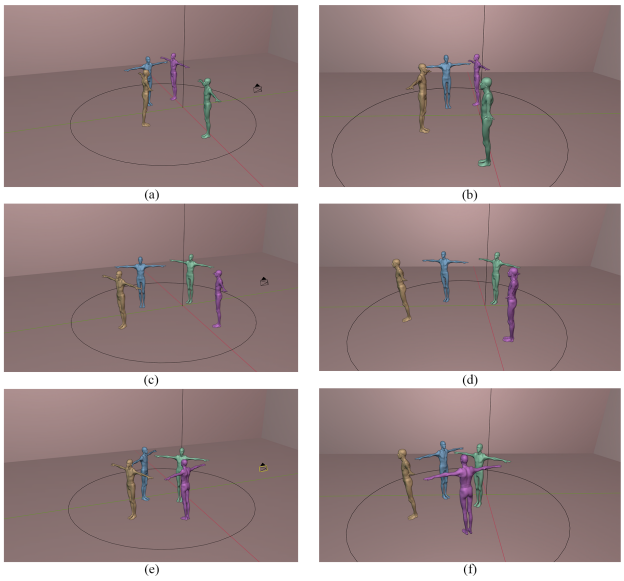

Fig. 3. The schematic diagram of our experimental results. **Top**: the results on the ground truth dataset. **Middle**: the results on the baseline dataset. **Bottom**: the results on our dataset. **Left**: the 3D results in the world coordinate. **Right**: the 2D results in the camera coordinate.

5. REFERENCES

[1] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu, "Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments," *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, no. 7, pp. 1325–1339, 2013.

[2] Leonid Sigal, Alexandru O Balan, and Michael J Black, "Humaneva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion," *International journal of computer vision*, vol. 87, no. 1-2, pp. 4, 2010.

[3] Eldar Insafutdinov, Leonid Pishchulin, Bjoern Andres, Mykhaylo Andriluka, and Bernt Schiele, "Deepercut: A deeper, stronger, and faster multi-person pose estimation model," in *European Conference on Computer Vision*. Springer, 2016, pp. 34–50.

[4] Hao-Shu Fang, Shuqin Xie, Yu-Wing Tai, and Cewu Lu, "Rmpe: Regional multi-person pose estimation," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2334–2343.

[5] George Papandreou, Tyler Zhu, Nori Kanazawa, Alexander Toshev, Jonathan Tompson, Chris Bregler, and Kevin Murphy, "Towards accurate multi-person pose estimation in the wild," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4903–4911.

[6] Umar Iqbal and Juergen Gall, "Multi-person pose estimation with local joint-to-person associations," in *European conference on computer vision*. Springer, 2016, pp. 627–642.

[7] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh, "Realtime multi-person 2d pose estimation using part affinity fields," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7291–7299.

[8] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh, "Openpose: realtime multi-person 2d pose estimation using part affinity fields," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 1, pp. 172–186, 2019.

[9] Dario Pavllo, Christoph Feichtenhofer, David Grangier, and Michael Auli, "3d human pose estimation in video with temporal convolutions and semi-supervised training," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7753–7762.

[10] Xiaowei Zhou, Menglong Zhu, Spyridon Leonardos, Konstantinos G Derpanis, and Kostas Daniilidis, "Sparseness meets deepness: 3d human pose estimation from monocular video," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4966–4975.

[11] Julieta Martinez, Rayat Hossain, Javier Romero, and James J Little, "A simple yet effective baseline for 3d human pose estimation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2640–2649.

[12] Francesc Moreno-Noguer, "3d human pose estimation from a single image via distance matrix regression," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2823–2832.

[13] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.

[14] J. Charles, T. Pfister, D. Magee, D. Hogg, and A. Zisserman, "Personalizing human video pose estimation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[15] Alex M Andrew, "Multiple view geometry in computer vision," *Kybernetes*, 2001.