3D Human Pose Estimation based on Convolutional Neural Network and Event-based cameras: a proof of concept on a DHP19 dataset

Leonardo Lucarelli,[1] *Alessandrino* Manilii[1]

[1] *Department of Information Engineering, Università Politecnica delle Marche*

Abstract. Nowadays Human Pose Estimation (HPE) represents one of the main research theme in the field of computer vision. Despite innovative methods and solutions introduced for frame processing algorithms, the use of standard frame-based cameras still has several drawbacks such as data redundancy and fixed frame-rate. The use of event-based cameras guarantees higher temporal resolution with lower memory and computational cost, while preserving the significant information to be processed and so representing a new solution for real time applications. In this paper we make use of DHP19 dataset [1], the first and, to date, the only one with HPE data recorded from Dynamic Vision Sensor (DVS) event-based cameras. Starting from their baseline CNN model, we use and test the dataset to implement a new multi-input CNN architecture in order to achieve better performance. Results show that the proposed approach outperforms single-input model in terms of accuracy and execution time.

1 Introduction

Human pose estimation (HPE) is a classic computer vision challenge aiming at generating 2D or 3D human skeleton from single or multiple view of one or more subjects. This tasks need great computational capacity to achieve good performance since big amount of data need to be processed and event cameras represent a great solution, guaranteeing high temporal resolution, high dynamic range and fewer storage requirements. These cameras, which record only changes in pixel intensity values, produce asynchronous streams of events, reducing the amount of data and so redundancy caused by fixed and insignificant background information. Moreover, it also prevents problems such as fixed frame rate that does not change if the registered object is static or in movement, producing higher quality output. In this scenario, the Dynamic Vision Sensor (DVS) Human Pose dataset (DHP19), the first DVS dataset for 3D human pose estimation, introduced in [1], could represent the standard for future works in real-time context in which lightness and speed are crucial features. The aim of this work is to explore the dataset and provide some test results for future works, integrating those described in [1] with a new multi-input architecture able to improve inference performance. We introduce a detailed description of values used for all settable parameters and proposed solutions to problems encountered during the different steps: from big size dataset management through the use of a data generator to the choice of training hyperparameters and the optimization of testing results with the use of a confidence threshold mechanism. The proposed multi-input architecture represents the real novelty compared to [1] approach and results show higher accuracy with lower processing time.

2 Related Work

2.1 HPE Datasets

Until today, there are multiple existing datasets for 3D HPE recorded using frame-based cameras. Between these, the most used are HumanEva [2], Human3.6M [3] and MPI-INF-3DHP [4]. All of them include a whole body recording obtained with multiple cameras on different subjects, performing different movements. Moreover, they include ground truth 3D pose recording from a motion capture system. To date, however, state of art contributions on HPE through innovative datasets, as those built using event-based cameras, are not many. The only existing dataset build for HPE purpose with whole body joints position and recorded through event-based cameras is the one introduced in [1]. Among other event-based datasets, only two relate to human gestures or body movements [5, 6]. Main features are described in the section 3.1.

2.2 CNN architectures

To the best of our knowledge, Convolutional Neural Networks (CNN) have been applied to the output of event-based cameras only to solve classification problems as in [6–8] or regression problem with single output in [9] but never for multiple output regression problems. For 3D HPE, there are different approaches for reconstructing 3D pose from multi-camera view [10, 11]. The method we

have chosen for 3D triangulation is the same described in [1]: firstly the model predicts 2D joints position, then we use triangulation to project the 2D prediction on the 3D space knowing the camera position. Another approach consists on using a single camera view, avoiding predicting 2D pose and directly predict the pose in 3D space [12, 13].

3 Materials

DHP19 is the first human pose dataset with data collected from DVS event-based cameras. This specific feature implies a more complex preprocessing step in order to generate standard frame from sequence of events. It also introduces great enhancements, i.e. avoiding redundancy and so saving space for more information, making dataset more valuable. A description of the instruments used, the dataset contents and the preprocessing procedures applied is provided.

3.1 Dataset

DHP19 contains the records of 17 subjects, 12 female and 5 male who perform specific movements in a recording volume of $2x2x2m^3$ in a therapy environment. All 33 movements, grouped in 5 session, can be categorized in upper-limb movements (1, 2, 5, 6, 15-20, 27-33), lower-limb movements (3, 4, 7, 8, 23-26), and whole-body movements, and they are performed by each subject 10 times consecutively. Subject position is represented through 13 labeled joints positions corresponding to head, both shoulder, both elbow, hands, left/right hip, both knees and feet as can be seen in Fig.1. In our experiments, subjects 1-9 (52% of dataset) were used as training set, subjects 10-12 (18% of dataset) as validation set and remaining subjects 13-17 (30%) as testing set.

3.2 Preprocessing and frame generation

Data have been acquired with the simultaneous use of 4 Davis cameras, a complex version of standard DVS cameras able to record both sequence of events and standard static APS (Active Pixel Sensor) frames, and Vicon motion capture system, made of 10 Bonita Motion Capture (BMC) infrared (IR) cameras. An event e = (x, y, t, p, c) is made of: position (x,y) in pixel array, time t in microsecond, polarity of the brightness change p and camera ID (0-3) c. During preprocessing several filters are applied to remove noise due to background activity and hot pixels, i.e. pixels with abnormally low event thresholds, and to mask out spots where events are generated due to the infrared light emitted from the BMC cameras in case of IR filters not working properly. In order to make use of pre-existent frame-based deep learning algorithms for event cameras [1], following the strategy from [14], we transform event stream into frames. To do so a fixed amount of events, about 7.5k per camera, are grouped in each frame which is finally normalized in the range [0,255]. Labels, instead, are constructed knowing initial and final event timestamps for each generated frame calculating the average position in that time window. Last step consist of mapping obtained 3D label position into frame space, rounding to the nearest pixel, making use of projection matrices for each camera view. The projected 2D labels represent the absolute position in pixel space. Finally a smoothing filter is applied on each heatmap through Gaussian blurring with a sigma of 2 pixels.

4 Human Pose Estimation task

The baseline approach followed for 3D HPE is the same described in [1], in which a single CNN is used to predict joints' positions in two dimensions. The model takes as input a single frame at a time and so, for each camera, a different output of the same moment is obtained. From these predictions, 3D pose estimation is inferred using triangulation and knowing cameras position. Instead in multi-view approach two frames from different cameras are given as input and multiple output is given as a couple of heatmap representing estimated joints positions for the two frames.

4.1 Baseline: Single-Input Multiple-Output (SIMO) approach

In this approach, the CNN takes a single frame as a Numpy array of shape (260, 344) and processes it; output is a Numpy array of shape (260, 344, 13), where the 13 heatmaps represent the probability for each joint to be in a certain position among all 260x344 possibilities. To get the final prediction, the position corresponding to the max value is taken, and then compared with the label through mean squared error. A detailed description of model architecture and methods used for training and testing is provided in the following paragraphs.

4.1.1 Model architecture

CNN is organized in 17 convolutional layers as shown in Table 1. Each layer is followed by Rectified Linear Unit (RELU) activation and it has a variable number of 3x3 filters from 16 to 64 contributing to a total amount of about 220 thousands trainable parameters. A bidimensional max pooling is also called 2 times decreasing computational cost and affecting on possible overfitting. In order to reproduce the baseline results in [1], we implemented their architecture considering the same setting for each layer parameter (i.e. number of filters, kernel size, strides, dilation rate).

4.1.2 Training setting

Due to the large size of the dataset (160 GB), we had to face several problems related to memory management solving by using a data generator. The basic idea is to split dataset into constant-frames files and loading into memory only those used to fill the batch. We first generate single-frame files and related labels as 13 concatenated 260X344 matrices, obtained projecting Vicon data

Table 1: SIMO model architecture

Layer	Out dimension	Stride	Dilatation
(1) Conv2D	(260, 344, 16)	1	1
MaxPooling2D	(130, 172, 16)	1	1
(2) Conv2D	(130, 172, 16)	1	1
(3) Conv2D	(130, 172, 32)	1	1
(3) Conv2D	(130, 172, 32)	1	1
MaxPooling2D	(65, 86, 32)	1	1
(4) Conv2D	(65, 86, 64)	1	2
(5) Conv2D	(65, 86, 64)	1	2
(6) Conv2D	(65, 86, 64)	1	2
(7) Conv2D	(65, 86, 64)	1	2
(8) Conv2DTc	(130, 172, 32))	2	1
(9) Conv2D	(130, 172, 32)	1	2
(10) Conv2D	(130, 172, 32)	1	2
(11) Conv2D	(130, 172, 32)	1	2
(12) Conv2D	(130, 172, 32)	1	2
(13) Conv2DTc	(260, 344, 16)	2	1
(14) Conv2D	(260, 344, 16)	1	1
(15) Conv2D	(260, 344, 16)	1	1
(16) Conv2D	(260, 344, 16)	1	1
(17) Conv2D	(260, 344, 13)	1	1

into frame's dimensions space through event-cameras projection matrices; then we define a data generator to constantly fill batches at every epochs. Mean Squared Error (MSE) and RMSProp are used respectively as cost function and optimizer. The model was trained for 20 epochs using a batch size of 64 and a variable learning rate of 10e-3 for the first 10 epochs, of 10e-4 from epoch 11 to 15 and of 10e-5 from 16 to 20. The training took about 80 hours on an NVIDIA RTX-2080 Ti.

4.1.3 Testing

We first tested the model trained by [1] as described in the paper in order to reproduce same results. After the preprocessing step, we tested the model on the same test subjects (13-17) with and without confidence threshold in range [0.1 - 0.5]. The confidence threshold mechanism works comparing the max value of the heatmap with the set threshold for each joint and, in case condition isn't satisfied, last valid joint position is taken. This can lead to take better or worst prediction depending on which is the average max value in the output heatmaps: this is the reason why using an higher confidence threshold does not improve performance, as shown by results in following paragraphs.

For evaluation purposes the mean per joint position error (MPJPE) metric is used, expressed with the following formula:

$$MPJPE = \frac{1}{J}\sum_{i}^{J}\left\|x_i - \hat{x}_i\right\| \quad (1)$$

where i represent a different joint at every iteration, ranging in the interval [1,J], and J is the number of joints.

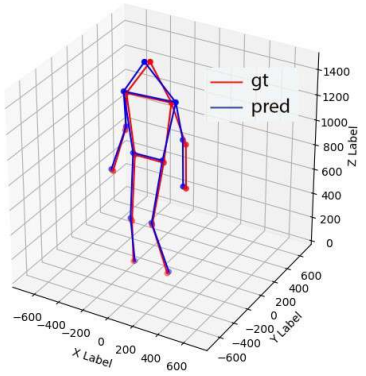

Figure 1: Ground truth and prediction overlapping in 3D space

4.2 Multiple-Input Multiple-Output (MIMO) approach

In order to deeply adapt model architecture to the multi-view case-study, we implemented a multi-input and multi-output (MIMO) CNN, which employs shared layers. In particular, as shown in Fig.2, the network takes as input two frames of the same instant time respectively from camera 2 and 3 and it outputs 13 heatmaps, one per joint, for each one as in single-input CNN. So differently from the single-input approach, in which same layers were trained on both cameras and batch are made up of a casual sequence of frames related to different subject, session, movement and alternating cams, in this case the not-shared layers are only trained on a specific cam view, generating batch as sequence of couple of frames always related to two different cameras and randomly put in sequence.

4.2.1 Model architecture

As shown in Fig.2, the CNN is constituted by two separated single input heads which share a group of central layers. Shared layers are updated on frames of both cam while the others are only related to inputs from a specific cam. The increased number of layers due to this new configuration rises the number of trainable parameters from 220k to 308k.

4.2.2 Training setting

The training procedure and network parameters remain the same described in the single-input approach, except for the batch size set to 32 in order to generate batch with the same number of frame used for single-input case. It requires lower execution time than the single-input case thanks to the use of shared layers which, for each couple of frames given in input, decrease the amount of trainable parameters

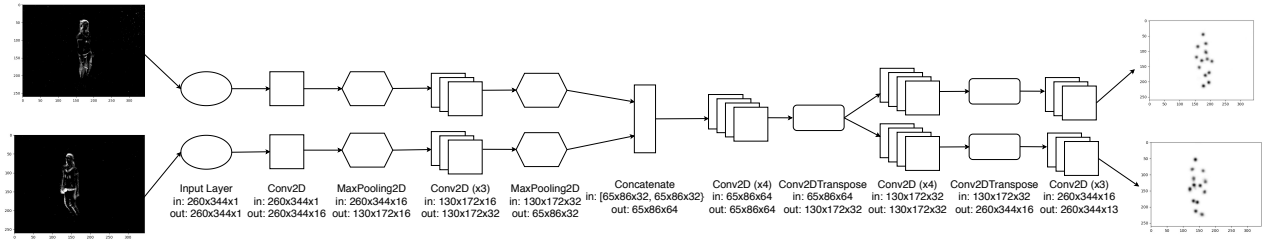

Figure 2: MIMO model architecture

updates. The training phase took about 70 hours on an NVIDIA RTX-2080 Ti.

4.2.3 Testing

To test the model we used the same procedure and metric described for the single-input approach.

4.3 3D prediction

In order to get 3D joint position from 2D prediction obtained after heatmap processing, as described in 4.1.3, we followed the same method used in [1] in which 3D projection is obtained through triangulation.

5 Results

Table 2 and 3 show the MPJPE trend for different confidence thresholds. The optimal threshold for 2D pose estimation is calculated as a function of the average max of the predicted value for each pixel. In fact, a too high threshold, means that a large number of prediction are discarded and for each one of them this mechanism takes as actual prediction the last one (in terms of time) that has satisfied the threshold's condition. So higher confidence threshold means higher probability that the temporal gap between the current prediction and the last good one is elevated. On the other hand, a too low confidence threshold can leads to problems when the heatmap values are very sparse since the maximum value of the prediction can be very lower than the average of the max value. This means that the prediction is uncertain, so it should be discarded.

5.1 Validation results

Despite we use MSE as validation loss for both models, it's more explanatory to calculate MPJPE on the validation subjects in order to analyse how training performance evolves during epochs. Figures 3 and 4 represent the trend of the MPJPE through the 20 epochs respectively for the single-input model and the multi-input model, calculated with the corresponding optimal confidence threshold identified in the next sections. We can notice how both models converge and the trend follow the learning rate scheduler.

5.2 2D pose estimation

In table 2 are shown the testing results for the pre-trained model introduced in [1] and the models described in sections 4.1.1 and 4.2.1.

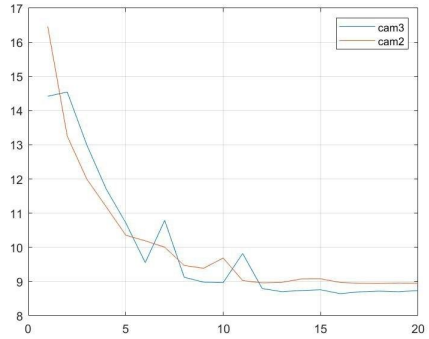

Figure 3: MPJPE trend for SIMO model through the 20 epochs

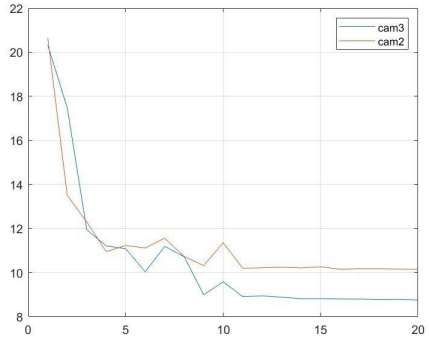

Figure 4: MPJPE trend of the MIMO model through the 20 epochs

5.3 3D pose estimation

Table 3 summarises the triangulation results described in section 4.3. The best results are obtained for the second section (precisely movements 9-14, 21, 22), which correspond to movements where all the human shape is visible in the DVS frames.

5.4 Baseline model

In this section we discuss the results obtained from the model trained by [1].

Table 2: Summary table comparing MPJPE score on test subjects for the baseline pre-trained model, the SIMO model and the MIMO model, all trained on the two frontal camera (camera 2 and 3) and tested with various confidence thresholds. In bold the selected confidence threshold used for 3D projection.

Conf. Thr.	None	0.01	0.1	0.3	0.4
Baseline					
Cam 2	7.70	7,55	7.42	**7.22**	7.26
Cam 3	7.92	7.69	7.25	**6.91**	6.98
SIMO					
Cam 2	11.47	11.12	**10.56**	11.09	11.65
Cam 3	10.86	10.72	**10.16**	10.36	10,77
MIMO					
Cam 2	10.49	**10.40**	11.82	14.85	16.54
Cam 3	9.64	**9.58**	10.51	13.75	15.93

Table 3: Mean of the 3D MPJPE (in mm) of the 5 testing subject through the 5 session for different CNNs. In bold the overall mean of the 3D MPJPE for each model.

\	Subj. 13	Subj. 14	Subj. 15	Subj. 16	Subj. 17	Subj. Mean
Baseline						
Session 1	84,67	95,87	86,74	125,37	107,41	90,95
Session 2	40,69	53,29	48,65	72,88	79,43	66,43
Session 3	91,80	134,50	104,70	125,16	134,74	124,79
Session 4	77,57	101,05	105,97	92,53	99,63	80,53
Session 5	75,70	107,24	106,48	147,81	111,46	113,84
Session Mean	60,44	80,91	75,85	92,12	86,22	**80,31**
SIMO						
Session 1	175,96	190,79	166,43	202,26	175,70	150,09
Session 2	64,21	73,77	68,12	101,60	102,92	87,17
Session 3	162,35	186,77	171,70	214,22	182.45	209,74
Session 4	129,42	147,47	149,90	135.82	123.36	116,29
Session 5	147,47	198,07	177.16	240,46	180,28	198,40
Session Mean	95,27	121,50	116,74	136,84	116,08	**118,23**
MIMO						
Session 1	147,02	168,42	179,63	173,78	263,26	165,38
Session 2	55,70	61,70	60,24	93,12	104,13	85,06
Session 3	149,17	166,00	150,37	216,58	191,38	183,94
Session 4	131,73	142,51	154,79	126,71	129,93	115,48
Session 5	144,58	161,27	124,75	224,25	177,62	174,22
Session Mean	96,90	101,78	107,61	127,60	126,65	**115,48**

5.4.1 2D pose estimation

In the first section of the table 2 the best results correspond to the confidence threshold equal to 0.3, with a relative improvement (compared to no-confidence results) of 6% and 12% for camera 2 and 3 respectively.

5.4.2 3D pose estimation

Starting from the estimation obtained from the baseline model, we projected the predicted 2D joint position to 3D space with triangulation. With a confidence threshold of 0.3, the average 3D MPJPE over all trained movements and test subject is 80,31mm, as shown in table 3.

5.5 SIMO model

In this section we discuss the results obtained from the model described in 4.1 trained by us.

5.5.1 2D pose estimation

Results showed in the second section of table 2 are worse compared to the baseline results, with an average MPJPE increment of %30. This may be due to a different setting of training parameters or a different type of data normalization.

5.5.2 3D pose estimation

In table 3 is reported the testing results of the 3D triangulation with an average MPJPE of 118,23mm and an increment of 32% compared to the one obtained from the pre-trained model. Using a confidence threshold leads to an average improvement of 66%.

5.6 MIMO model

In this section the results of 2D testing and 3D triangulation obtained with the model described in the paragraph 4.2.1 are discussed. As already mentioned, we have used the same training hyperparameters of the single-input model in order to perform a fair comparison between the two approaches.

5.6.1 2D pose estimation

The third section of table 2 shows the 2D testing results of the multi-input model. Modifying architecture leads to an average improvement of the results of 3.5% compared to SIMO model (referring to the value corresponding to the best confidence threshold of each model).

5.6.2 3D pose estimation

Table 3 summarize 3D triangulation results. Compared to SIMO model, the MPJPE overall mean has decreased of %2.5.

6 Discussions

Starting from testing results presented in [1] we firstly reproduced those results and used them as a baseline. Then we retrained the same CNN with our hyperparameters and finally we proposed our multi-input architecture using same hyperparameters configuration for training. From the results it is possible to notice that SIMO and MIMO models don't show enhancements in terms of MPJPE scores compared to the baseline results. This can be due to several reasons. The more reasonable one regards the training hyperparameters, especially for SIMO model in which CNN architecture has not be modified. For example, the setting of a different batch size or learning rate scheduling, according to the evaluation metrics trend among epochs, could lead to performance improvements. Other reasons could be linked to the data generator mechanism, which may led to a different training procedure, or to another kind of output normalization. However, the new MIMO model architecture allowed to get improvements

compared to SIMO model. This can be explained by considering that the inputs given to the model have very similar features since they correspond to 2 different points of view of the same instant of time, and so using shared layers could increase accuracy, as well as guaranteeing lower inference time. Future works could be related to explore different training hyperparameters, as those above mentioned, in order to find additional improvements. About MIMO model architecture, it could be interesting to find the optimal configuration considering to share a different number of layers or a new method of selection. As described in 5.2, also the use of a final softmax layer could introduce new test and observations. Finally, more ambitious projects could aim at implementing 3D CNN for obtaining a direct 3D HPE, in order to overcome geometric methods as triangulation or imposing kinematic constraining in state estimation, avoiding impossible joint position's prediction.

References

[1] E. Calabrese, G. Taverni, C. Awai Easthope, S. Skriabine, F. Corradi, L. Longinotti, K. Eng, T. Delbruck, *DHP19: Dynamic Vision Sensor 3D Human Pose Dataset*, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops* (2019)

[2] L. Sigal, A. Balan, M.J. Black, International Journal of Computer Vision **87**, 4 (2010)

[3] C. Ionescu, D. Papava, V. Olaru, C. Sminchisescu, IEEE Transactions on Pattern Analysis and Machine Intelligence **36**, 1325 (2014)

[4] D. Mehta, H. Rhodin, D. Casas, O. Sotnychenko, W. Xu, C. Theobalt, CoRR **abs/1611.09813** (2016), 1611.09813

[5] Y. Hu, H. Liu, M. Pfeiffer, T. Delbruck, Frontiers in Neuroscience **10**, 405 (2016)

[6] A. Amir, B. Taba, D. Berg, T. Melano, J. McKinstry, C. Di Nolfo, T. Nayak, A. Andreopoulos, G. Garreau, M. Mendoza et al., *A Low Power, Fully Event-Based Gesture Recognition System*, in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017), pp. 7388–7397

[7] H. Liu, D.P. Moeys, G. Das, D. Neil, S. Liu, T. Delbrück, *Combined frame- and event-based detection and tracking*, in *2016 IEEE International Symposium on Circuits and Systems (ISCAS)* (2016), pp. 2511–2514

[8] I. Lungu, F. Corradi, T. Delbrück, *Live demonstration: Convolutional neural network driven by dynamic vision sensor playing RoShamBo*, in *2017 IEEE International Symposium on Circuits and Systems (ISCAS)* (2017), pp. 1–1

[9] A.I. Maqueda, A. Loquercio, G. Gallego, N. García, D. Scaramuzza, CoRR **abs/1804.01310** (2018), 1804.01310

[10] S. Amin, M. Andriluka, M. Rohrbach, B. Schiele, *Multi-view Pictorial Structures for 3D Human Pose Estimation* (2013), pp. 45.1–45.11, ISBN 1-901725-49-9

[11] H. Rhodin, N. Robertini, D. Casas, C. Richardt, H. Seidel, C. Theobalt, CoRR **abs/1607.08659** (2016), 1607.08659

[12] D. Mehta, S. Sridhar, O. Sotnychenko, H. Rhodin, M. Shafiei, H.P. Seidel, W. Xu, D. Casas, C. Theobalt, *VNect: Real-time 3D Human Pose Estimation with a Single RGB Camera* (2017), Vol. 36, http://gvv.mpi-inf.mpg.de/projects/VNect/

[13] G. Pavlakos, X. Zhou, K.G. Derpanis, K. Daniilidis, CoRR **abs/1611.07828** (2016), 1611.07828

[14] D.P. Moeys, F. Corradi, E. Kerr, P.J. Vance, G.P. Das, D. Neil, D. Kerr, T. Delbrück, CoRR **abs/1606.09433** (2016), 1606.09433