JAISCR, 2022, Vol. 12, No. 4, pp. 281 – 298
10.2478/jaiscr-2022-0019

COMBINED YOLOV5 AND HRNET FOR HIGH ACCURACY 2D KEYPOINT AND HUMAN POSE ESTIMATION

Hung-Cuong Nguyen[1], Thi-Hao Nguyen[1], Jakub Nowak[2], Aleksander Byrski[3], Agnieszka Siwocha[4], Van-Hung Le[5,*]

[1]*Faculty of Engineering Technology, Hung Vuong University, Vietnam*

[2]*Czestochowa University of Technology, Czestochowa, Poland*

[3]*AGH University of Science and Technology, Institute of Computer Science, 30-059 Kraków, Poland*

[4]*University of Social Sciences, Institute of Information Technologies, 9 Sienkiewicza Street, 90-113 Łódź, Poland*

[5]*Tan Trao University, Vietnam*

[*]*E-mail: van-hung.le@mica.edu.vn*

Submitted: 15th June 2022; Accepted: 18th October 2022

Abstract

Two-dimensional human pose estimation has been widely applied in real-world applications such as sports analysis, medical fall detection, human-robot interaction, with many positive results obtained utilizing Convolutional Neural Networks (CNNs). Li et al. at CVPR 2020 proposed a study in which they achieved high accuracy in estimating 2D keypoints estimation/2D human pose estimation. However, the study performed estimation only on the cropped human image data. In this research, we propose a method for automatically detecting and estimating human poses in photos using a combination of YOLOv5 + CC (Contextual Constraints) and HRNet. Our approach inherits the speed of the YOLOv5 for detecting humans and the efficiency of the HRNet for estimating 2D keypoints/2D human pose on the images. We also performed human marking on the images by bounding boxes of the Human 3.6M dataset (Protocol #1) for human detection evaluation. Our approach obtained high detection results in the image and the processing time is 55 FPS on the Human 3.6M dataset (Protocol #1). The mean error distance is 5.14 pixels on the full size of the image (1000×1002). In particular, the average results of 2D human pose estimation/2D keypoints estimation are 94.8% of PCK and 99.2% of PDJ@0.4 (head joint). The results are available.

Keywords: YOLOv5, HRNet, 2D key points estimation, 2D human pose estimation.

1 Introduction

Human pose estimation is defined as the process of localizing human joints (also known as keypoints – elbows, wrists, etc) in images or videos. In the last five years, this task has been gaining a lot of attention. Human pose estimation is applied in many fields such as sports analysis [56, 61]; medical fall event detection [55]; identification and analysis in traditional martial arts [49]; robot interaction, construction of actions and movements of people in the game [56]. There are two study directions for estimating human pose from image/video: 2D human pose estimation and 3D human posture estimation. The 2D human pose estimation is an intermediate result for the 3D human pose estimation. Based on the approach of Zhou et al. [63], 3D human pose estimation results are highly dependent on 2D human pose estimation.

Two deep learning-based approaches can be used to estimate 2D human poses. The first is the regression methods, which apply a deep neural network to learn a mapping from the input image to body joints or parameters of human body models to predict the key points on the human (*keypoints-based*). The second is the body part detection methods to predict the approximate locations of body parts and joints (*bodyparts-based*). Deep learning networks have achieved remarkable results for the estimation task. However, they still face many challenges such as heavy occlusion, partially visible human body, image resolution.

Sudharshan [3] presented some typical studies [51, 50, 54, 8, 35, 47, 57, 53] on estimating 2D human posture in images or videos. In Table 2 of [47], the authors have shown the results of the High-Resolution Network (HRNet) comparing the above methods for 2D human pose estimation on the COCO [31] dataset. HRNet is the most accurate across different configurations. Li et al. [29] used HRNet as a backbone for 2D human pose estimation on cropped human images of the Human 3.6M [23] dataset. As the Human 3.6M dataset contains 548,819 images of Protocol #1 for testing, manually marking the data area of the person in the image will take a long time. This difficulty is very dependent on the person conducting the crop and the HRNet's estimated data area in the human data region, without regard for other regions in the image. This problem of detecting people in the image is considered to be 100% accurate. However, when applied to real problems, no approach is appropriate.

In this paper, we combine the advantages of processing speed and a **C**ontextual **C**onstraint (**CC**) into a pre-trained YOLO v5 network [25, 24] for detecting people in a crowd and a pre-trained model of HRNet for estimating 2D keypoints/ 2D human pose in the image of Human 3.6M dataset. This study is a step in the process of estimating the 3D human pose in the scene/3D point cloud that we will perform in the next studies. Our proposed method is called (**YOLOv5 + HRNet combined**), as shown in Figure 1.

The main contribution of the paper is as follows:

– We have manually marked human regions by bounding boxes on the image with our tool developed in MATLAB to evaluate human detection.

– We have proposed some efficient Contextual Constraints (**CC**) for human detection in images or video.

– Evaluating the human detection in the images of the Human 3.6M dataset with pre-trained models which trained on widely used CNNs architectures (e.g. YOLOv5, Mask R-CNN, VGG, SSD, Mobilenet).

– We proposed an efficient matching strategy, called **YOLOv5 + HRNet combined** for estimating 2D human pose/2D keypoints from the Human 3.6M dataset.

– We performed step-by-step detailed evaluations of human detection results, 2D human pose estimation/2D keypoints in images or videos.

The paper is organized as follows. Section 1 introduces human detection, 2D keypoints estimation and 2D human pose estimation on the image. Section 2 discusses related works by the methods, results of 2D keypoints estimation and 2D human pose estimation. Section 3 presents a combination of YOLOv5, context constraints, and HRNet for 2D keypoints estimation/2D human pose estimation. Section 4 introduces and presents the Human 3.6M dataset, evaluation metrics, implementation,

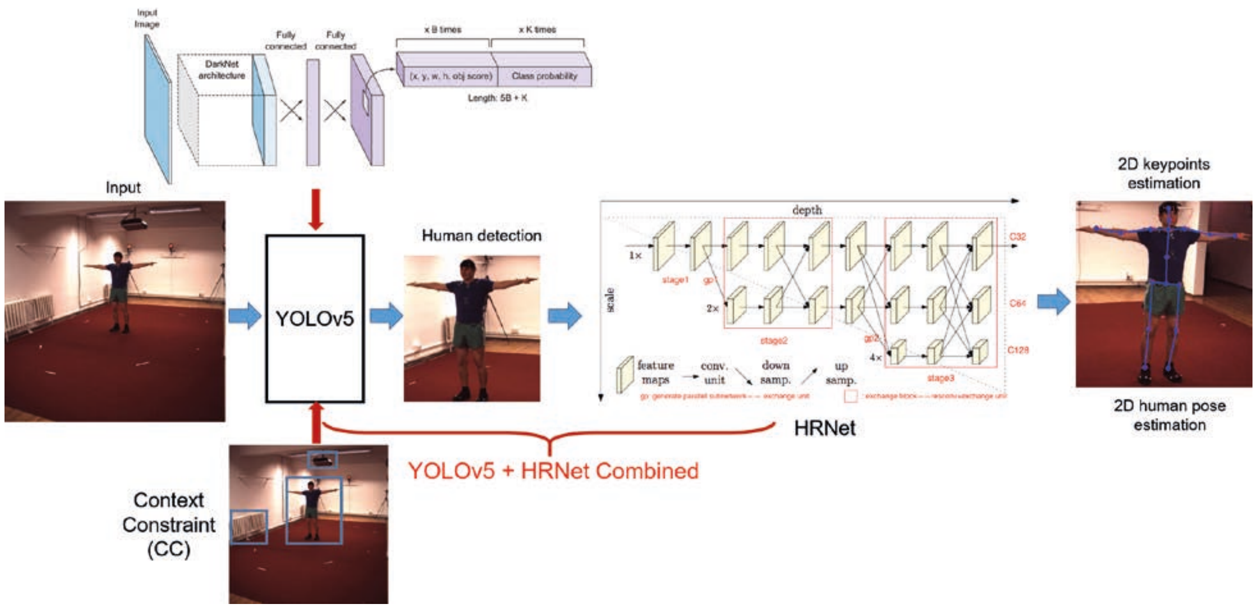

Figure 1. Proposed combined model of YOLOv5 and HRNet for highly accurate 2D human pose estimation.

some results, discussions on human detection as 2D keypoints estimation, and 2D human pose estimation. Section 5 concludes the paper and proposes some future work.

2 Related Works

Estimating human pose is an important issue that has been studied by the computer vision community for the past decade, especially when the advent of convolutional neural networks has yielded impressive results when solving this problem. The process of estimating human posture can be done based on the results of human detection or performed directly on the image. Here we present some studies on these two approaches.

Human detection in images or videos is one of the most important problems in computer vision. In recent years, most studies and applications have employed CNNs to detect persons and objects in general and demonstrated many impressive results. Girshick et al. [15] proposed a Region-based Convolutional Neural Network (R-CNN) for object detection. This network can be applied as a bottom-up method for localizing and segmenting objects of region proposals, and it improved classification efficiency by using supervised pre-training for labelled training data.

He et al. [19] proposed SPPnet (Spatial Pyramid Pooling network) to train the object detection model. Traditional CNNs include two main components: convolutional layers and fully connected layers. To overcome the fixed-size constraint of the network, SPPnet adds an SPP layer to the last convolutional layer. The fixed-length output features are generated from the SPP layer pools. SPP is relatively robust to object deformations. The extracted features of variable scales are pooled by SPP. Karen et al. [46] based their assumptions on the characteristics of CNNs that the depth of the CNNs affects the accuracy. The greater the depth, the greater the identification detection accuracy. Therefore, the authors have proposed the VGG16 network with the input 224×224 RGB image to the convolutional layer. After that, the input image passed a stack of convolutional layers. The final output size of the convolutional layer is 3×3. Recently, Xiangyu et al. [62] improved the VGG model in Fast R-CNN for object classification and detection; Haque et al. [17] also applied the VGG model to ResNet to detect objects.

To improve the results of R-CNN and SPPnet, Girshick et al. [14] proposed Fast R-CNN, which takes as input the entire image and a set of region proposals. Fast R-CNN performs two main types of computational steps: processes several convolutional and max-pooling layers on the whole image to generate a feature map. Each proposal interest region of the pooling layer then extracts a fixed-length feature vector from the generated feature map, and the input of a sequence of fully connected layers is the extracted feature vector. The SPPnet [19] and Fast R-CNN [14] models work on region proposals that could be the object, which reduces the computational burden of these CNNs. However, the accuracy of these networks has not been greatly improved. Ren et al. [43] proposed an RPN (Region Proposal Network) that shares the full-image convolutional features with the detection network, which makes nearly cost-free region proposals. The architecture of the Faster R-CNN consists of two parts: a deep, fully convolutional network (RPN) and a Fast R-CNN detector that uses the proposed regions. Recently, Goon et al. [22] used the Faster R-CNN for detecting pedestrians from drone images. The CNNs presented so far (R-CNN, SPPnet, VGG, Fast R-CNN, Faster R-CNN) are mainly concerned with high accuracy, but the computational burden for object detection is high. Therefore, Redmon et al. [40] proposed a YOLO-based network with a computation speed of about 67 fps of YOLO version 2 on the VOC 2007 dataset. The bounding boxes are predicted directly using the fully connected layers on top of the convolutional feature extractor. Currently, the YOLO network has four versions (YOLO version 1 to 4). Lui et al. [32] proposed the Single Shot Detector (SSD) network for object detection. The following mechanism is employed: the base network is used for high-quality image classification; fixed-size bounding boxes and scores are generated from a feed-forward convolutional network, and the final detections are generated by a non-maximum suppression step. Jonathan et al. [21] have performed a comparative study for object detection, which focuses on comparing object detection results based on typical CNNs: Faster R-CNN [43], R-FCN [10], and SSD [32]. The CNNs used the feature extractors as VGG or ResNet, calling them "meta-architectures". The authors evaluated many configurations of each CNN and analyzed the effect of configurations and the image size on the detection results.

Generally, immediate estimation of a human pose in images or videos using CNN can be divided into two families [12]: regression methods and body part detection methods.

Regression methods use CNNs to learn a mapping from the input image to predict the key points. Toshev et al. [52] proposed a Deep Neural Network based on a cascade technique for regressing the location of body joints. The proposed CNN includes seven layers, the input image size of CNN is resized to 220×220 pixels. The cascade of pose regressor technique is applied to train the multi-layer prediction model. In the first stage, the cascade starts with the initial position predicted over the entire input image. In the next stage, Deep Neural Network regressors are trained to predict a displacement of the joint locations with the correct locations in the previous stage. Thus, each subsequent stage can be thought of as a refinement of the currently predicted pose. Liang et al. [30] proposed a strategy of a compositional pose regression based on the ResNet50 [20]. The authors used a re-parameterized and bone-based representation that contains human body information and pose structure but did not use joint-based representation. The loss function is calculated based on each part of the human body, the joints are defined with respect to a constant origin point in the image coordinate system J_0. Each bone has a directed vector pointing from it to its parent. Luvizon et al. [34] proposed an end-to-end regression based on two Soft-argmax functions (Block-A and Block-B) for 2D human pose estimation from images, Block-A provides refined features and Block-B provides body-part and contextual activation maps. Two blocks are used to build one prediction block. Block-A used a residual separable convolution, the Block-B is used to transform input feature maps into part-based detection maps and context maps. As for the body part detection, body part detection methods train a body part detector to predict the positions of body joints. Newell et al. [36] proposed the stacked hourglass architecture for training model to predict the positions of body joints on the heatmap in which the heatmap is generated by a 2D Gaussian centered at the 2D ground truth annotation. The stacked hourglass repeats the bottom-up and top-down processing with intermediate supervision with the eight hourglasses. This CNN used the convolutional and max-pooling layers at a very low resolution and used then the top-down sequence of upsampling (the nearest neighbor upsampling of the lower resolution) and a combination of features across scales.

With the body part detection methods, Cao et al. [6] proposed a two-branch CNN to jointly predict heatmaps for body part detection that generate the ground truth confidence maps from the annotated 2D keypoints. The first branch predicts confidence maps and the second branch predicts the part affinity fields based on a novel feature representation called part affinity fields that preserves both location and orientation information across the region of support of the limb.

3 YOLOv5 + HRNet models combined for 2D Human pose estimation and 2D keypoint estimation

3.1 Human detection

Detecting humans in images using CNNs has been studied extensively and has achieved impressive results. Especially, CNNs such as R-FCN [11] Faster R-CNN [44], SSD [33], YOLO [39, 41, 42, 4] obtain good results as compared and presented by Jonathan [26]. The Faster R-CNN [44] is an improvement to the Fast R-CNN [14], it also integrates the region recommendation algorithm into the CNN model. The Faster R-CNN is based on two main ideas: building a single model consisting of a region proposal network (RPN) and the Fast R-CNN with a shared CNN. The Faster R-CNN works with several steps:

1. Using a pre-trained CNN such as VGG or ResNet (backbone) to classify images.
2. Fine-tune RPN (region proposal network) for the region proposal task, initialized by the pre-trained image classifier model (backbone). Positive examples are recommended regions when $IoU > 0.7$ by sliding a small window of size $n \times n$ over the entire CNN feature of the image. At the center of each window, multiple regions with different scales and ratios at the same time are predicted. Anchor is a combination of sliding window center, scale, and ratio.
3. Training the Fast R-CNN model using region proposals generated from the current RPN.
4. Using the Fast R-CNN network to initialize RPN training. While keeping the convolution layers shared, just fine-tune the specific RPN layers. At this stage, RPN and detection networks have shared convolution layers.
5. Finally, fine-tuning separate layers of the Fast R-CNN.

Then came the introduction of the Mask R-CNN based on the Faster R-CNN as the backbone for detecting and segmenting people in images.

Especially, YOLO is a CNN network with average accuracy, very fast processing speed, up to 91 fps. Since the input is the input image, YOLO uses some simple steps of a network of convolution, pooling, and fully connected layers to get the output. This architecture can be optimized to run on the GPU with a single forward pass, and thus achieve very high speeds. The main idea of YOLOv1 [39] is to divide the image into a grid cell with size (7×7). For each grid cell, the model will make predictions for a bounding box (B) of humans. Each box B includes 5 parameters (the coordinates of the center of human (x, y), width (w) of human, the height of human (h), and confidence (cof_h) of the human prediction. Given the grid cells in the other (7×7) grid, the model also predicts the probability into each class human, the YOLOv1 architecture is shown in Figure 2. (cof_h) is defined as Equation 1.

$$cof_h = P(h) * IOU^{prediction}_{groud-truth} \tag{1}$$

where $P(h)$ is the probability that there is a human in the cell, $IOU^{prediction}_{groud-truth}$ is the intersection over a union of the prediction region and the ground truth.

YOLOv1 [39] imposes spatial constraints on bounding boxes, each grid cell can predict only very few bounding boxes and only one class. During training, the loss function does not have a separate evaluation between the error of the small bounding box versus the error of the large bounding box. To improve the disadvantages of YOLOv2, YOLOv2, and YOLO 9000 have come up with some strategies.

– Batch Normalization: Batch Normalization is added after all convolution layers of YOLOv2 [41].
– High-resolution classifier: YOLO is trained with 2 phases. The first phase will train a classifier network with a small input image size $(224 \times$

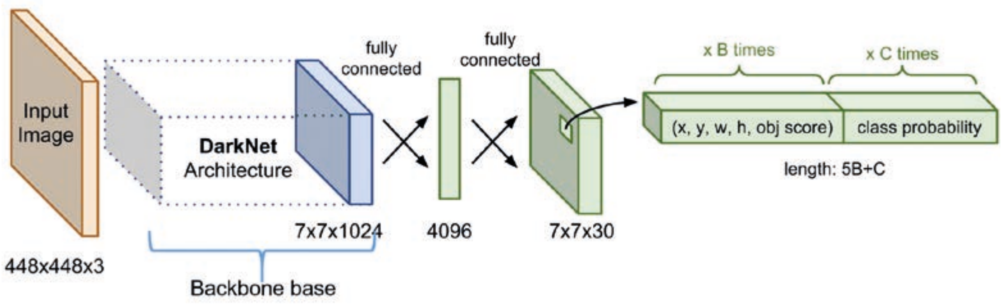

Figure 2. YOLO architecture [39] for object detection in the image.

224) and the second phase will remove the fully connected layer and use this classifier network as the backbone to train the detection network. In the YOLOv2, fine-tune the backbone network under the larger input image is (448×448).

- Using anchor box architecture to make predictions: In YOLOv2 remove the fully connected layer in the middle of the network and use anchor box architecture to predict bounding boxes. It is much easier to predict offsets relative to the anchor box than to predict the bounding box coordinates.

- K-mean clustering for anchor selection: Instead of having to manually select anchor boxes, YOLOv2 uses a k-means algorithm to make the best anchor box choices for the network.

- Direct location prediction: YOLOv1 has no limitations in predicting the position of the bounding box. When the weights are initialized randomly, the bounding box can be predicted anywhere in the image. This makes the model unstable in the early stages of training. The position of the bounding box can be very far from the position of the grid cell. YOLOv2 uses the sigmoid (σ) function to restrict the value between 0 and 1, thereby limiting the bounding box predictions around the grid cell, thereby making the model more stable during training.

- Add fine-grained features: Faster R-CNN and SSD make predictions at different layers of the network to take advantage of feature maps of different sizes. YOLOv2 also combines features at different levels to make predictions, specifically YOLOv2's original architecture combines a 26x26 feature map taken from the near end with a 13×13 feature map at the end to make predictions. Specifically, these feature maps will be merged together to form a block used for prediction.

- Multi-Scale Training: After adding the anchor box technique to YOLOv2, the input image size changed to 416×416 instead of 448×448. However, YOLOv2 is designed with only convolution and pooling layers, so it can adapt to much different input image sizes.

- Light-weight backbone: In YOLOv2 use Darknet-19. This network includes 19 layers of convolution and 5 layers of max-pooling which makes it faster than previous YOLO version.

YOLOv3 [42] has a similar architecture to YOLOv2, but it also brings some improvements.

- Using logistic regression to predict the confidence of the bounding box.

- YOLOv3 uses logistic classifiers instead of softmax for object classification. This works better if the labels are not "mutually exclusive", i.e. there can be objects belonging to two or more different classes.

- YOLOv3 uses Darknet-53 as backbone.

- YOLOv3 uses the Feature Pyramid Networks (FPN) architecture to make predictions from various scales of feature maps. This helps YOLOv3 take advantage of feature maps with different coarseness — fineness for prediction.

- YOLOv3 also adds associations between prediction classes. Model upsample the prediction classes at the later layers and then concatenate with the prediction classes in the earlier layers. This method helps to increase accuracy when predicting small objects.

The object detection challenge is now more accessible to those who do not have powerful computer resources thanks to the architecture of YOLOv4 [4]. Using YOLOv4, we can train an object detection network with extremely high accuracy using only a 1080ti or 2080ti GPU. To bring computer vision applications into practice in the future, current networks will need to be re-optimized to tolerate weak computing resources or develop high parallelism in servers.

In this paper, we use a pre-trained model trained on the COCO dataset of YOLOv5 [24] for head and human detection in a crowd and a context constraint to get the bounding box of the detected human in the image. The result of human and head detection when not using the constraint is shown in Figure 3. In Figure 3, the wall camera was mistakenly detected as a person. And in the image, the person has the largest bounding box in the image. Therefore, we propose that the bounding box of the person is the bounding box with the highest height among the bounding boxes detected and marked as the person.

3.2 HRNet-based for for 2D Human pose estimation/2D keypoints estimation

For human pose estimation/2D keypoints estimation of human, can use backbones like ResNet [20], Stacked Hourglass Networks [36], or some studies like Openpose [37], 2D Pose Estimation using Part Affinity Fields [7], CPM (Convolutional pose machines) [54]. In this paper, we used the study's Ke et al. [47] that called HRNet for 2D human pose estimation on the images.

HRNet has the first stage of using a high-resolution subnetwork, after that repeatedly fusing the representations produced by the high-to-low subnetworks. The feature maps are created by simply generating high-resolution representations by 2D Gaussian on the ground truth position of each keypoint from the heatmaps. On the COCO and MPII datasets, HRNet uses the pre-trained ResNet as the backbone.

In this paper, we use the pre-trained model of HRNet on the Human 3.6M dataset to evaluate the 2D human pose estimation results on the detected human, as presented in Section 3.1.

4 Experimental Results

4.1 Data collection

To train and evaluate the model and the estimated model, we use the benchmark Human 3.6M dataset [23]. Human3.6M is the most widely used indoor dataset for 3D human pose estimation from single-view or multi-view (4 different views in an indoor Lab scene) of the cameras. This dataset is captured from 11 subjects/people (6 males and 5 females). The people perform six types of action (upper body directions movement, full body upright variations, walking variations, variations while seated on a chair, sitting on the floor, various movements), which includes 16 daily activities (directions, discussion, greeting, posing, purchases, taking photos, waiting, walking, walking dog, walking pair, eating, phone talk, sitting, smoking, sitting down, miscellaneous). The frames are captured from TOF (Time-of-Flight) cameras, and the data frame rate of the cameras is from 25 to 50 Hz. This dataset contains about 3.6 million images (1,464,216 frames for training from 5 people (2 female and 3 male), 646,180 frames for validation from 2 people (1 female and 1 male), 1,467,684 frames for testing from 4 people (2 female and 2 male)), 3.6 million 3D human pose annotations captured by the marker-based MoCap system. 3D human pose annotation of Human 3.6M dataset consists of 17 key points arranged in order as shown in Figure 4.

3D human pose annotations of Human 3.6M are annotated based on the Mocap system. The coordinate system of this data is the real-world coordinate system. To evaluate the estimation results, we convert this data to the camera coordinate system. We based on the parameter set of the cameras and the formula for converting data from 2D to 3D of Nicolas [5] by Equation 2.

$$\begin{aligned} P3D_c.x &= \frac{(x_d - cx) * depth(x_d, y_d)}{fx} \\ P3D_c.y &= \frac{(y_d - cy) * depth(x_d, y_d)}{fy} \\ P3D_c.z &= depth(x_d, y_d) \end{aligned} \tag{2}$$

where fx, fy, cx, and cy are the intrinsics of the depth camera. $P3D_c$ is the coordinate of the key point in the camera coordinate system.

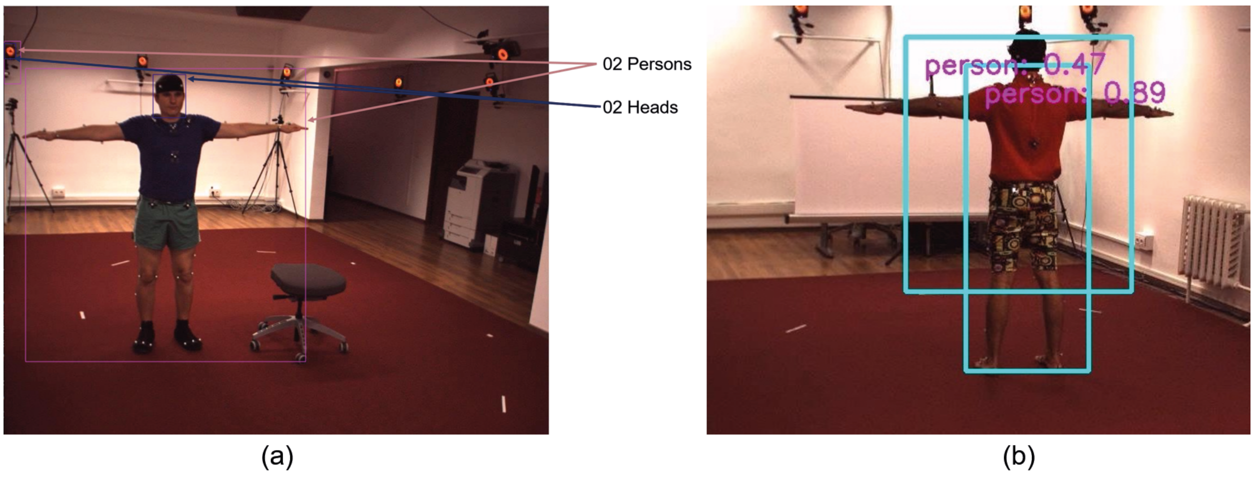

Figure 3. Human and head detection results using YOLOv5 [24] on Subject 11 of the Human 3.6M dataset.

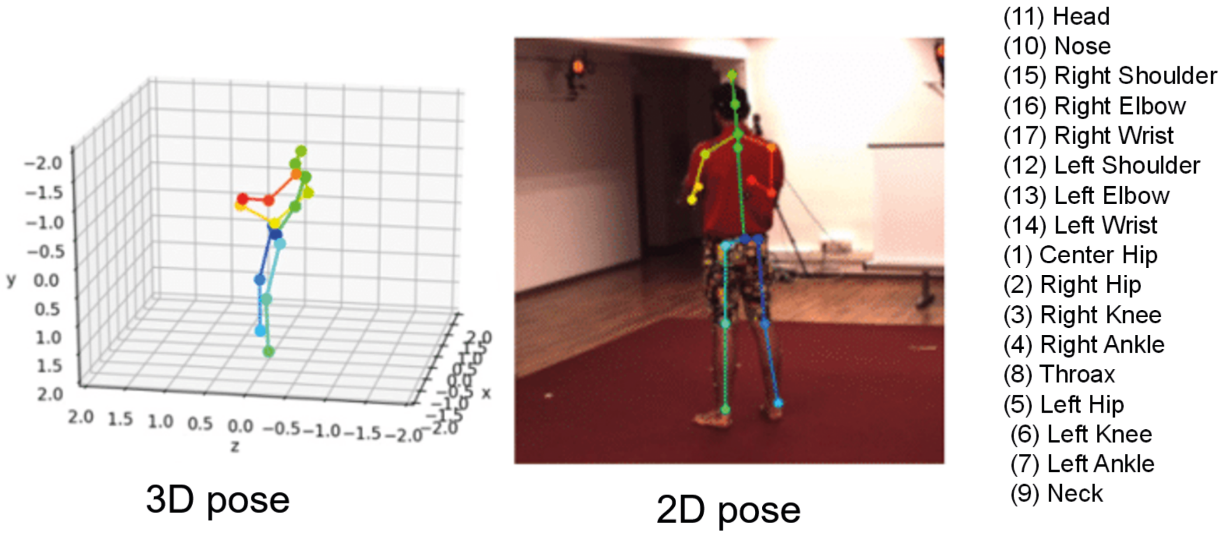

Figure 4. An illustration of human pose in Human 3.6M dataset. The left is a human skeleton in the real-world coordinate system. The middle is the human skeleton on the 2D image. The right is the order and names of the joints of the human skeleton.

We reprojected the 3D human pose annotation from the real-world coordinate system to the camera coordinate system using Equation 3.

$$P3D_c = (P3D_w - T) * R^{-1} \tag{3}$$

where R and T are the rotation and translation parameters to transform from the real-world coordinate system to the camera coordinate system. $P3D_w$ is the coordinate of the keypoint in the world coordinate system. We projected 3D human pose annotation to 2D human pose annotation according to Equation 4.

$$\begin{aligned} P2D.x &= \frac{P3D_c.x * fx}{P3D_c.z} + cx \\ P2D.y &= \frac{P3D_c.y * fy}{P3D_c.z} + cy \end{aligned} \tag{4}$$

where $P2D$ is the coordinate of the key point in the image.

The source code mark and 2D annotation of Human 3.6M database is shown in the link [1].

The authors have divided the Human 3.6M dataset into three protocols to train and test the estimation models. Protocol #1 uses the subjects S1, S5, S6, and S7 for training, and the subjects S9 and S11 for testing. Protocol #2 is divided into training-testing similar to Protocol #1, but the predictions are further post-processed by a rigid transformation before comparing to the ground-truth. Protocol #3 uses the subjects S1, S5, S6, S7, and S9 for training, and the subjects S11 for testing. This dataset is saved in path [2].

To evaluate the results of human detection on the image of Human 3.6M (testing set includes Protocol #1 (Subject 9, Subject 11), Protocol #3 (Subject 11)), we marked the person in the image with a bounding box ($bbox(x,y,w,h)$), as shown in Figure 5. The number of frames manually marked bounding box is 548,819 images.

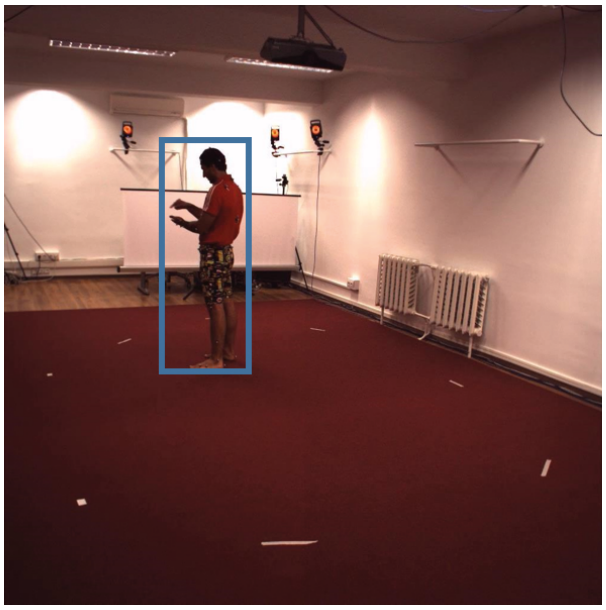

Figure 5. Illustration of a person's bounding box on the image.

4.2 Implementation and Evaluation measurement

In this paper, we used PC with GPU GTX 970, 4GB for fine-tuning, training, testing human detection, human pose estimation. The processing steps, code fine-tuning, training, testing, and development process were performed in Python language ($\geq$3.6 version) with the support of the OpenCV, Pytorch ($\geq$3.6 version), CUDA/cuDNN libraries, gcc/& g++ ($\geq$5.4 version), In addition, there are a number of other libraries such as Numpy, scipy, Pillow, cython, matplotlib, scikit-image, tensorflow $\geq$ 1.3.0, keras $\geq$ 2.0.8, opencv-python, h5py, imgaug, and IPython. The source code for training and testing is shown in link [3].

In particular, the 2D human pose estimation process using HRNet is performed on the image with resolution (228×384), while the image size of human detection by **YOLOv5 + CC** is $(w \times h)$, this image is resized to the size of (228×384), and the 2D human pose image of the Human 3.6M database has the resolution of (1000×1002). In this paper, we use Affine transformation (A_m) [48] to transform the estimated 2D human pose onto a resolution image of $(1000x1002)$, as illustrated in Figure 6.

[1]https://drive.google.com/drive/folders/1xtiI0VqSXXytAG5NxmlK3ci-ObR1MgT-?usp=sharing

[2]http://vision.imar.ro/human3.6m/

[3]https://drive.google.com/drive/folders/1-Hu2842xWDtZWBo762iT$_v iBYcuaAR7V?usp = sharing$

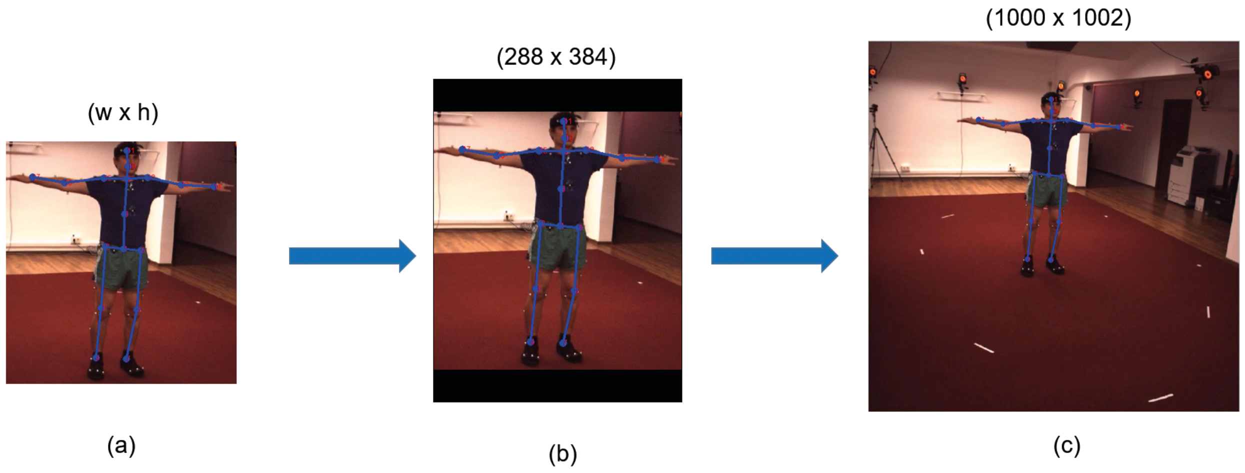

Figure 6. An illustration of human pose in Human 3.6M dataset. (a) is 2D human pose estimation results on the image of size ($w \times h$) from human detection results by **YOLOv5 +CC**. (b) is 2D human pose estimation results on the image of size (288×384). (c) is 2D human pose estimation results on the image of size (1000×1002).

The Affine transformation (A_m) includes the Rotation (Equation 5), Translation (Equation 6) and Scaling (Equation 7) matrices as shown.

$$\begin{bmatrix} X_{ro} \\ Y_{ro} \\ 1 \end{bmatrix} = \begin{bmatrix} cos(\alpha) & -sin(\alpha) & 0 \\ sin(\alpha) & cos(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \tag{5}$$

where (X_{ro}, Y_{ro}), (X, Y) are the resulting rotated coordinates, the coordinates of the pixel to be rotated, respectively. α is the rotation angle.

$$\begin{bmatrix} X_{tr} \\ Y_{tr} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & T_x \\ 0 & 1 & T_y \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \tag{6}$$

where (X_{tr}, Y_{tr}), (X, Y) are the resulting translated coordinates, the coordinates of the pixel to be translated, respectively. (T_x, T_y) is the translation vector.

$$\begin{bmatrix} X_{sd} \\ Y_{sd} \\ 1 \end{bmatrix} = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \tag{7}$$

where (X_{sd}, Y_{sd}), (X, Y) are the resulting scaled coordinates, the coordinates of the pixel to be scaled, respectively. (S_x, S_y) is the scales of the image.

In this paper, the coordinate rotation has ($\alpha = 0$). The process of converting the pose estimated (P_E) on the resolution image (288×384) to the resolution image (1000×1002) (P_{EN}) is performed based on the Equation 8.

$$P_{EN} = A_m^{-1} * P_E \tag{8}$$

In this paper, we perform the assessment in two stages: The results of detecting people in the image are first evaluated; The second is to evaluate the 2D keypoints estimation results of humans on the image.

At the first stage, we use the Jaccard index (P_{ji}) [16] for the evaluation, it is illustrated in Equation 9.

$$P_{ji} = \frac{R_g \cap R_e}{R_g \cup R_e} \tag{9}$$

where R_g is the person's ground truth bounding box, R_e is the estimated bounding box of human. We use the thresholds ($t = \{50\%, 55\%, 60\%, 65\%, 70\%\}$) to evaluate the accuracy of the results of human detection in the image with ($\{AP_{50}, AP_{55}, AP_{60}, AP_{65}, AP_{70}\}$), if $P_{ji} \geq t$ then it is a true detection.

To evaluate the results of 2D human pose estimation/2D keypoints estimation, we evaluate on three measures: the average distance between the 2D key point of the 2D ground truth and the estimated 2D key point (Err_{avg}), as shown in Equation 10.

$$Err_{avg} = \frac{1}{N}\Sigma_1^N \frac{1}{J}\ \Sigma_1^J Dis(p_i, \widetilde{p_i})(10)$$

where N and J are the numbers of frames and number of joints ($J = 17$) respectively, $\widetilde{p_i}$ and p_i are predicted and ground-truth coordinates of i^{th} joint of the hand, Err_{avg} is the Euclidean distance between two points.

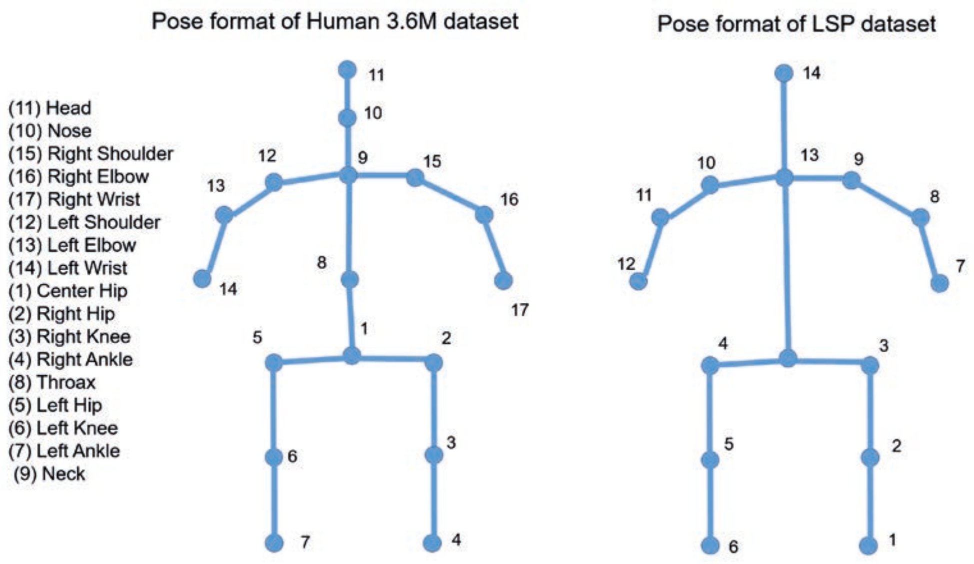

Figure 7. An illustration of human pose format in Human 3.6M dataset, LSP dataset.

The second and third are the PCk (Percentage of Correct Key-points) and the PDJ (Percentage of Detected Joints) measurements, respectively [59, 38, 45]. In this paper, we use the format of the LSP dataset implemented in Yang et al.'s study [59], the skeleton data includes 14 keypoints representing the evaluation of PCK, PDJ [58] of the Human 3.6M dataset. The data normalization process is illustrated in Figure 7. In [59], PCK and PDJ are computed on seven representative keypoints in the human skeleton, called: "Ankle"– Ankle, "Knee" – Knee, "Hip" – Hip, "Wris" – Wrist, "Elbo" – Elbow, "Shou" – Shoulder, "Head" – Head. The evaluation code is shown in [60].

4.3 Results and Discussions

We first evaluate the results of human detection in images when using the proposed method (**YOLOv5 +CC**). At the same time, we compare this method with some studies on human detection (e.g. Mask R-CNN, VGG, SSD, Mobilenet) in images combined with constraints (**CC**). The results are shown in Table 1.

In Table 1, we evaluate the number of the detected human (Number detected), each frame with only one detected human marked with a bounding box (rectangle) on the Human 3.6M dataset (Protocol #1 - 548,819 images). In which, Mask R-CNN has the highest detection rate (100%). Next, we evaluate the accuracy of human detection (AP) on the images that have been detected (Number detected). Our proposed method has the highest accuracy (AP) at all thresholds ($AP_{50}, AP_{55}, AP_{60}, AP_{65}, AP_{70}$). In particular, we also evaluate the processing speed of human detection on images, which we recommend as having the fastest processing speed ($55fps$). This is a very fast speed on an average configuration computer. The results of human detection in the image are shown in link [4].

Based on the human detection results of our proposed method (Table 1), we evaluate of 2D keypoints estimation/2D human pose estimation. The results of human pose estimation based on Err_{avg} measurement on the Human 3.6M dataset (Protocol #1) are shown in Table 2.

In Table 2, the average error value of our proposed model is 5.14 pixels, this error value is very low, it is only 2 pixels larger than (HRN + U + S) [28], [29] method. While we evaluate the whole image. In particular, the PCKh@50 measure has a very high accuracy, 94.8% on average.

The results of the 2D keypoints estimation on the image are illustrated in Figure 8.

We next evaluate the 2D keypoints estimation/ 2D human pose estimation results by PCK, PDJ measurements, they are shown in Table 3.

[4]https://drive.google.com/drive/folders/1F0GKwKVx6Wp4_YzjmJEwAJKXd5fIUP80?usp=sharing

Table 1. The results of human detection on the Human 3.6M dataset (Protocol #1) evaluated on the CNNs.

Measurement/ Methods	**Number testing samples**	**Number detected**	AP_{50} (%)	AP_{55} (%)	AP_{60} (%)	AP_{65} (%)	AP_{70} (%)	**Processing time (fps)**
YOLOv5 [24] + CC	548,819	548,346 (99.91%)	**99.78**	**99.38**	**98.42**	**97.07**	**94.16**	**55**
Mask R-CNN [2, 18] + **CC**	548,819	548,819 (**100%**)	97.17	96.93	96.61	96.12	95.51	2
MobilenetV1 SSD [1, 13] + **CC**	548,819	507,991 (92.56%)	96.87	95.66	93.59	89.52	81.38	10
VGG SSD [13] + CC	548,819	536,496 (97.75%)	99.14	98.60	97.66	95.99	92.81	12
Mobilenet SSD [27] + **CC**	548,819	548,801 (99.99%)	77.04	75.93	73.43	68.34	59.99	4.34

Table 2. The results of 2D keypoints estimtion/2D human pose estimation ((Err_{avg} - pixels)) on the Human 3.6M dataset (Protocol #1 - 548,346 images) that based on the human detection results of **YOLOv5+CC** method.

Methods	**Average Joint Localization Error** (Err_{avg}) **(pixels)**
CPN (CVPR' 18) [9]	5.4
HRN + U + S [28], [29]	**4.4**
Our (YOLOv5 +CC+ HRNet)(Full size of image)	5.14

Table 3. The results of 2D keypoints estimtion/2D human pose estimation ((PCK, PDJ - %)) on the Human 3.6M dataset (Protocol #1 - 548,346 images) that based on the human detection results of **YOLOv5+CC** method.

Part/ Joints/ Measurement	**Ankle** (%)	**Knee** (%)	**Hip** (%)	**Wris** (%)	**Elbo** (%)	**Shou** (%)	**Head** (%)	**Mean** (%)
				Part				
PCKh@0.5	94.4	95.0	96.0	92.2	94.0	96.1	96.1	**94.8**
				Joints				
PDJ@0.10	88.2	92.1	93.1	86.1	88.9	92.6	93.9	-
PDJ@0.20	94.4	95.0	96.0	92.2	94.0	96.1	96.1	-
PDJ@0.30	95.9	96.1	97.1	94.7	96.2	97.2	97.3	-
PDJ@0.40	98.2	98.3	99.1	97.4	98.4	99.1	99.2	-

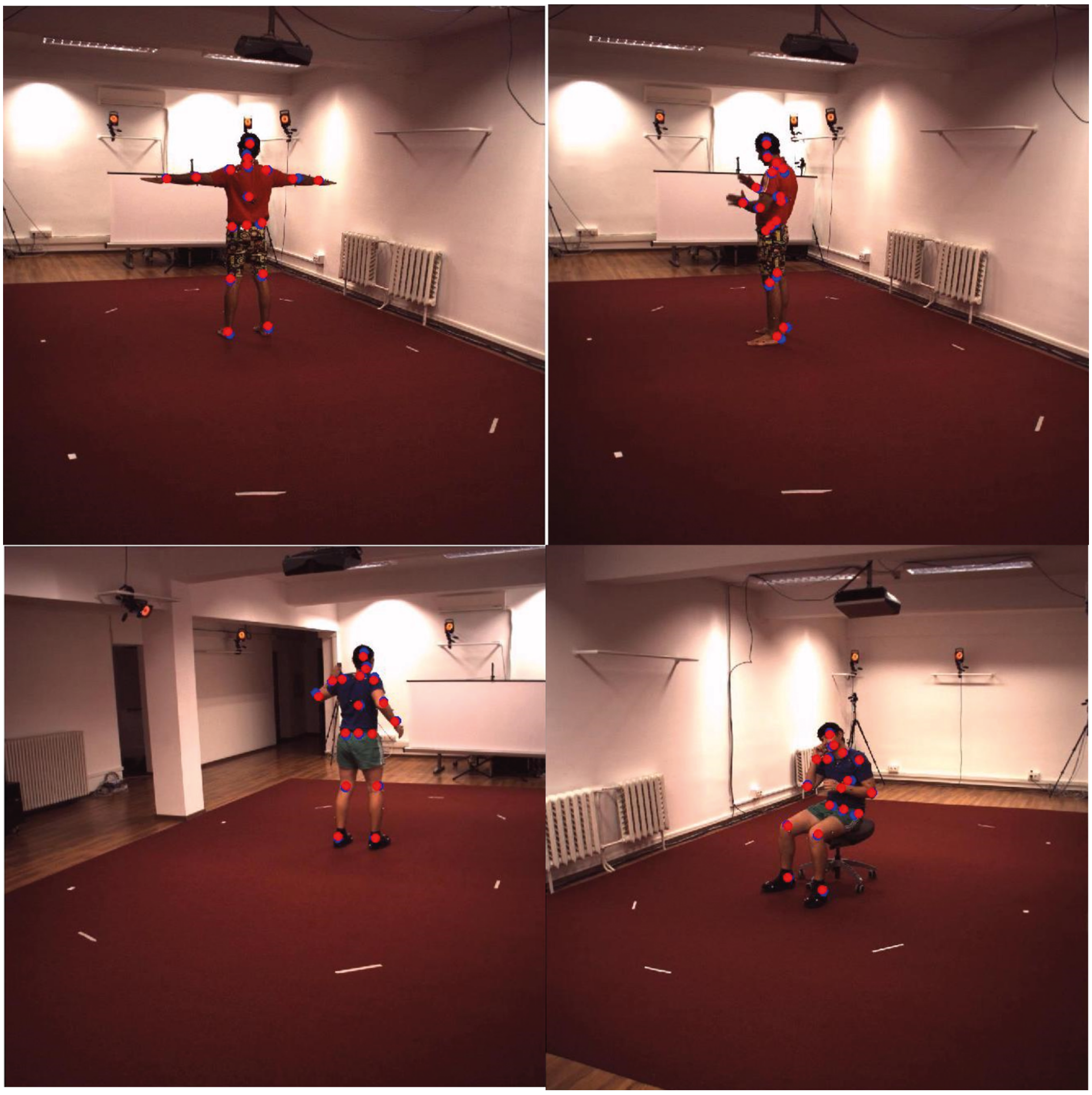

Figure 8. An illustration of 2D keypoints estimation on the Human 3.6M dataset (Protocol #1 - Subject 9, Subject 11). The blue keypoints are the ground truth keypoints, the red keypoints are the estimated keypoints.

The estimated results of 2D keypoints/2D human pose is shown in link [5].

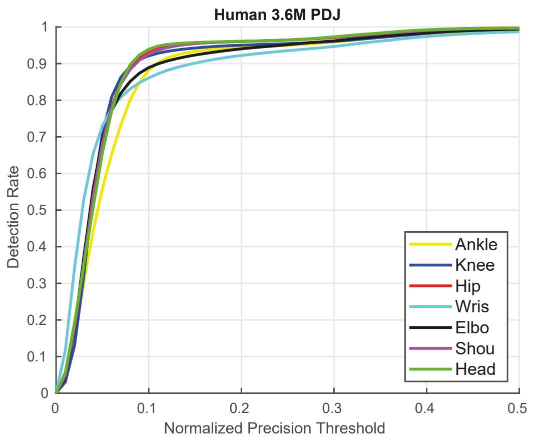

Figure 9. Human 3.6M - Area Under the Curve (AUC) is computed for a range of PDJ thresholds.

The results of the 2D human pose estimation on the image are illustrated in Figure 10.

Full source code for human pose estimation, evaluation is shared in link [6].

5 Conclusion

In this paper, we implement a combination of **YOLOv5**, contextual constraints (**CC**), and **HRNet** to build an automated application for detecting and estimating human posture with high accuracy. The person detection result is ($AP_{50} = 99.78\%, AP_{55} = 99.38\%, AP_{60} = 98.42\%, AP_{65} = 97.07\%, AP_{70} = 94.16\%$) and the processing time is 55 fps. The distance error is **5.14** pixels, the average accuracy according to PCKh@0.5 is **94.8** % and PDJ@0.4 is **99.2**% of the head joint. The results are shown in Table 1, 2, 3 and illustrated in Figure 8, 10. The model that we proposed is fully automated and has fast computation times, which can be run on computers with typical configurations. It is suitable for developing practical applications of human detection and human pose estimation for sports analysis, sign language development. Especially the intermediate step for automatic estimation of 3D human pose. In the near future, we plan to develop a high-precision automatic model of 3D human pose estimation that can run on a regular desktop computer.

Acknowledgements

This research is supported by Hung Vuong University under grant number 03/2022/HDHV03.2022.
This research is supported by Tan Trao University.

References

[1] Ssd mobilenet v1 architecture (2018). [Accessed 22 Dec 2021]

[2] Abdulla, W.: Mask r-cnn for object detection and instance segmentation on keras and tensorflow. https://github.com/matterport/Mask_RCNN (2017). [Accessed 20 Dec 2021]

[3] Babu, S.C.: A 2019 guide to human pose estimation with deep learning. https://nanonets.com/blog/human-pose-estimation-2d-guide/. [Online: Accessed 5 December 2021]

[4] Bochkovskiy, A., Wang, C.Y., Liao, H.Y.M.: YOLOv4: Optimal Speed and Accuracy of Object Detection. arXiv (2020)

[5] Burrus, N.: Kinect calibration. http://nicolas.burrus.name/index.php/Research/KinectCalibration

[6] Cao, Z., Simon, T., Wei, S.E., Sheikh, Y.: Real-time multi-person 2D pose estimation using part affinity fields. In: IEEE Conference on CVPR, vol. 2017-Janua, pp. 1302–1310 (2017). DOI 10.1109/CVPR.2017.143

[7] Cao, Z., Simon, T., Wei, S.E., Sheikh, Y.: Realtime multi-person 2d pose estimation using part affinity fields. In: CVPR (2017)

[8] Carreira, J., Agrawal, P., Fragkiadaki, K., Malik, J.: Human pose estimation with iterative error feedback. CoRR abs/1507.06550 (2015)

[9] Chen, Y., Wang, Z., Peng, Y., Zhang, Z., Yu, G., Sun, J.: Cascaded Pyramid Network for Multi-person Pose Estimation. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 7103–7112 (2018). DOI 10.1109/CVPR.2018.00742

[10] Dai, J., Li, Y., He, K., Sun, J.: R-FCN: Object detection via region-based fully convolutional networks. Advances in Neural Information Processing Systems pp. 379–387 (2016)

[11] Dai, J., Li, Y., He, K., Sun, J.: R-fcn: Object detection via region-based fully convolutional networks. In: Advances in Neural Information Processing

[5]https://drive.google.com/drive/folders/11-lxGGI3SJOyiB5YI1Qo2b0NztXmLYZC?usp=sharing
[6]https://drive.google.com/drive/folders/1xtiI0VqSXXytAG5NxmlK3ci-ObR1MgT-?usp=sharing

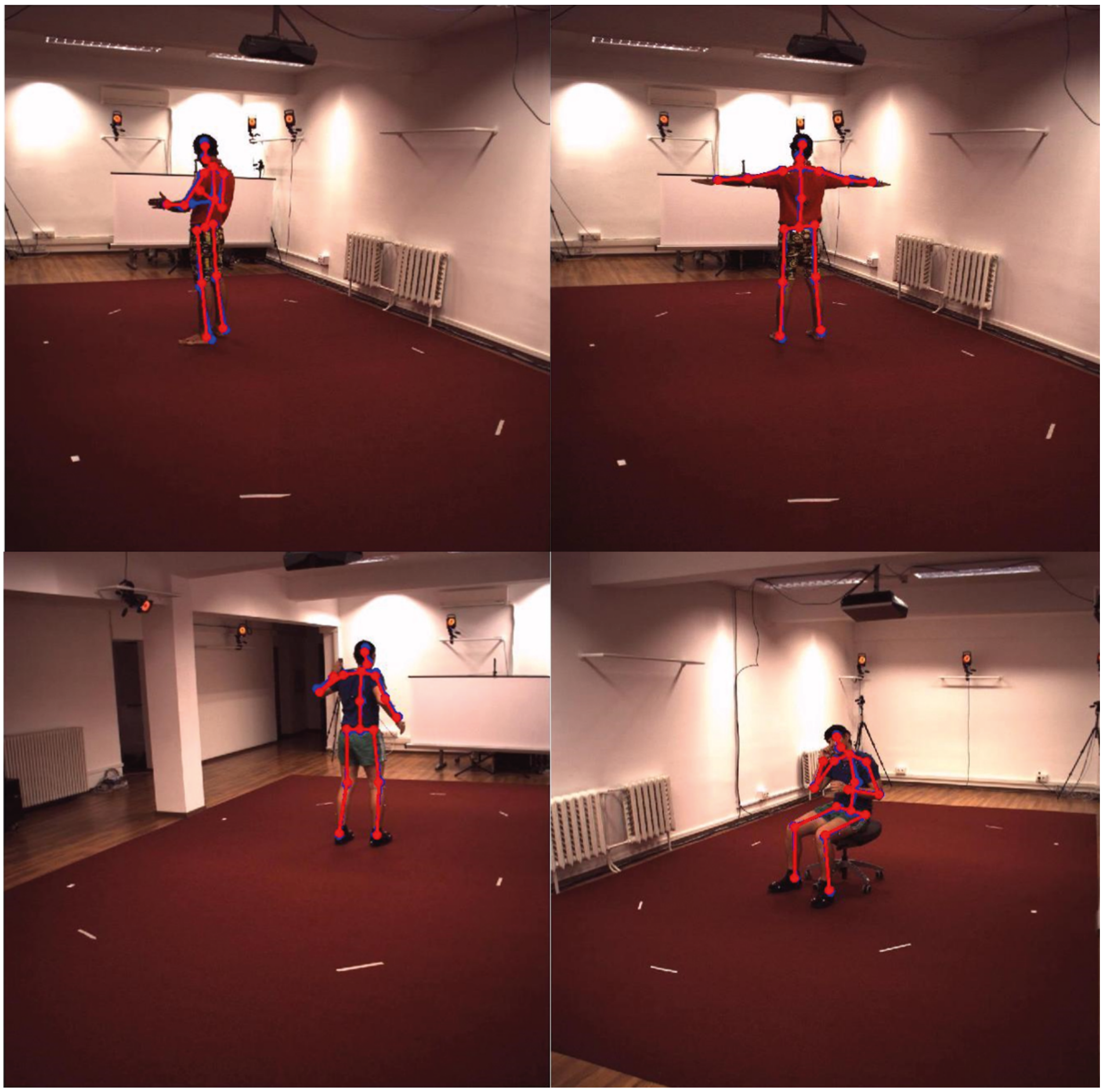

Figure 10. An illustration of 2D human pose estimation on the Human 3.6M dataset (Protocol #1 - Subject 9, Subject 11). The blue skeleton are the ground truth skeleton, the red skeleton are the estimated skeleton.

Systems, vol. 29. Curran Associates, Inc. (2016). https://proceedings.neurips.cc/paper/2016/file/577ef1154f3240ad5b9b413aa7346a1e-Paper.pdf

[12] Dang, Q., Yin, J., Wang, B., Zheng, W.: Deep learning based 2D human pose estimation: A survey. TPAMI 24(6), 663–676 (2021). DOI 10.26599/TST.2018.9010100

[13] Gao, H.: Single shot multibox detector implementation in pytorch. https://github.com/qfgaohao/pytorch-ssd (2020). [Accessed 20 Dec 2021]

[14] Girshick, R.: Fast R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision, vol. 2015 Inter, pp. 1440–1448 (2015). DOI 10.1109/ICCV.2015.169

[15] Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 580–587 (2014). DOI 10.1109/CVPR.2014.81

[16] Glen., S.: "jaccard index/similarity coefficient" from statisticshowto.com: Elementary statistics for the rest of us! https://www.statisticshowto.com/jaccard-index/. Online; accessed 6 December 2021

[17] Haque, M.F., Lim, H.y., Kang, D.s.: Object Detection Based on VGG with ResNet Network. In: 2019 International Conference on Electronics, Information, and Communication (ICEIC), pp. 1–3. Institute of electronics and information engineers (IEIE)

[18] He, K., Gkioxari, G., Dollar, P., Girshick, R.: Mask R-CNN. In: ICCV (2017)

[19] He, K., Zhang, X., Ren, S., Sun, J.: Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 37(9), 1904–1916 (2015). DOI 10.1109/TPAMI.2015.2389824

[20] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: IEEE Conference on CVPR, vol. 2016-Decem, pp. 770–778 (2016). DOI 10.1109/CVPR.2016.90

[21] Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A., Fischer, I., Wojna, Z., Song, Y., Guadarrama, S., Murphy, K.: Speed/accuracy trade-offs for modern convolutional object detectors. In: Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, vol. 2017-January, pp. 3296–3305 (2017). DOI 10.1109/CVPR.2017.351

[22] Hung, G.L., Sahimi, M.S.B., Samma, H., Almohamad, T.A., Lahasan, B.: Faster R-CNN Deep Learning Model for Pedestrian Detection from Drone Images. In: SN Computer Science, vol. 1, pp. 1–9. Springer Singapore (2020). DOI 10.1007/s42979-020-00125-y. https://doi.org/10.1007/s42979-020-00125-y

[23] Ionescu, C., Papava, D., Olaru, V., Sminchisescu, C.: Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. TPAMI 36(7), 1325–1339 (2014)

[24] Jocher, G.R.: Head and person detection model. https://github.com/deepakcrk/yolov5-crowdhuman. Online; accessed 6 December 2021

[25] Jocher, G.R.: Yolov5 tutorials. https://github.com/ultralytics/yolov5. Online; accessed 6 December 2021

[26] Jonathan, H.: Object detection: speed and accuracy comparison (faster r-cnn, r-fcn, ssd, fpn, retinanet and yolov3) (2018). [Accessed 18 Dec 2021]

[27] Krishnan, S.: Person-detection. https://github.com/SusmithKrishnan/person-detection (2021). [Accessed 20 Dec 2021]

[28] Li, N.: Evoskeleton, cascaded 2d-to-3d lifting. https://github.com/Nicholasli1995/EvoSkeleton. Online; accessed 25 December 2021

[29] Li, S., Ke, L., Pratama, K., Tai, Y.W., Tang, C.K., Cheng, K.T.: Cascaded deep monocular 3d human pose estimation with evolutionary training data. In: The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2020)

[30] Liang, S., Sun, X., Wei, Y.: Compositional Human Pose Regression. In: ICCV, vol. 176-177, pp. 1–8 (2017). DOI 10.1016/j.cviu.2018.10.006

[31] Lin, T.Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C.L., Dollár, P.: Microsoft coco: Common objects in context (2014). http://arxiv.org/abs/1405.0312

[32] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: SSD: Single shot multibox detector. In: European Conference on Computer Vision, vol. 9905 LNCS, pp. 21–37 (2016). DOI 10.1007/978-3-319-46448-0_2

[33] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S.E., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: B. Leibe, J. Matas, N. Sebe, M. Welling (eds.) ECCV (1), Lecture Notes in Computer Science, vol. 9905, pp. 21–37. Springer (2016). http://dblp.uni-trier.de/db/conf/eccv/eccv2016-1.htmlLiuAESRFB16

[34] Luvizon, D.C., Tabia, H., Picard, D.: Human pose regression by combining indirect part detection and contextual information. Computers and Graphics (Pergamon) 85, 15–22 (2019). DOI 10.1016/j.cag.2019.09.002

[35] Newell, A., Yang, K., Deng, J.: Stacked hourglass networks for human pose estimation. In: Computer Vision – ECCV 2016, pp. 483–499. Springer International Publishing (2016)

[36] Newell, A., Yang, K., Deng, J.: Stacked Hourglass Networks for Human Pose Estimation. In: ECCV (2016)

[37] openpose: openpose. https://github.com/CMU-Perceptual-Computing-Lab/openpose (2019). [Accessed 23 April 2019]

[38] Ramanan, D.: Learning to parse images of articulated bodies. In: In NIPS (2006)

[39] Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2016-Decem, pp. 779–788 (2016). DOI 10.1109/CVPR.2016.91

[40] Redmon, J., Farhadi, A.: Yolo9000: Better, faster, stronger. arXiv preprint arXiv:1612.08242 (2016)

[41] Redmon, J., Farhadi, A.: YOLO9000: Better, faster, stronger. In: Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, vol. 2017-Janua, pp. 6517–6525 (2017). DOI 10.1109/CVPR.2017.690

[42] Redmon, J., Farhadi, A.: Yolov3 an incremental improvement (2018). http://arxiv.org/abs/1804.02767. [Accessed 18 April 2021]

[43] Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems 28, pp. 91–99 (2015)

[44] Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence 39(6), 1137–1149 (2017). DOI 10.1109/TPAMI.2016.2577031

[45] Sapp, B., Taskar, B.: In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. DOI 10.1109/CVPR.2013.471

[46] Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings, pp. 1–14 (2015)

[47] Sun, K., Xiao, B., Liu, D., Wang, J.: Deep high-resolution representation learning for human pose estimation. In: CVPR (2019)

[48] Tan, D.: Image geometric transformation in numpy and opencv. https://towardsdatascience.com/image-geometric-transformation-in-numpy-and-opencv-936f5cd1d315 (2019). Online; accessed 6 December 2021

[49] Thanh, N.T., Hùng, L.V., Công, P.T.: An Evaluation of Pose Estimation in Video of Traditional Martial Arts Presentation. Journal of Research and Development on Information and Communication Technology 2019(2), 114–126 (2019). DOI 10.32913/mic-ict-research.v2019.n2.864

[50] Tompson, J., Goroshin, R., Jain, A., LeCun, Y., Bregler, C.: Efficient object localization using convolutional networks. In: CVPR, pp. 648–656. IEEE Computer Society (2015)

[51] Toshev, A., Szegedy, C.: Deeppose: Human pose estimation via deep neural networks. CoRR abs/1312.4659 (2013). http://dblp.uni-trier.de/db/journals/corr/corr1312.htmlToshevS13

[52] Toshev, A., Szegedy, C.: DeepPose: Human Pose Estimation via Deep Neural Networks. In: IEEE Conference on CVPR (2014)

[53] Wang, J., Sun, K., Cheng, T., Jiang, B., Deng, C., Zhao, Y., Liu, D., Mu, Y., Tan, M., Wang, X., Liu, W., Xiao, B.: Deep high-resolution representation learning for visual recognition. TPAMI

[54] Wei, S.E., Ramakrishna, V., Kanade, T., Sheikh, Y.: Convolutional pose machines. In: CVPR (2016)

[55] Weiming Chen , Zijie Jiang, H.G., Ni, X.: Fall Detection Based on Key Points of of human-skeleton using openpose. Symmetry (2020)

[56] Willett, N.S., Shin, H.V., Jin, Z., Li, W., Finkelstein, A.: Pose2Pose: Pose Selection and Transfer for 2D Character Animation. In: International Conference on Intelligent User Interfaces, Proceedings IUI, pp. 88–99 (2020). DOI 10.1145/3377325.3377505

[57] Xiao, B., Wu, H., Wei, Y.: Simple baselines for human pose estimation and tracking. In: European Conference on Computer Vision (ECCV) (2018)

[58] Yang, W.: Human Pose Estimation 101. https://github.com/cbsudux/Human-Pose-Estimation-101percentage-of-correct-key-points—pck (2019). [Accessed 18 April 2021]

[59] Yang, W., Ouyang, W., Li, H., Wang, X.: End-to-end learning of deformable mixture of parts and deep convolutional neural networks for human pose estimation. In: CVPR (2016)

[60] Yang, W., Ouyang, W., Li, H., Wang, X.: End-to-end learning of deformable mixture of parts and deep convolutional neural networks for human pose estimation. https://github.com/bearpaw/eval_pose (2016). Online; accessed 20 December 2021

[61] Zhang, H., Sciutto, C., Agrawala, M., Fatahalian, K.: Vid2Player: Controllable Video Sprites That Behave and Appear Like Professional Tennis Players. ACM Transactions on Graphics 40(3), 1–16 (2021). DOI 10.1145/3448978

[62] Zhang, X., Zou, J., He, K., Sun, J.: Accelerating Very Deep Convolutional Networks for Classification and Detection. IEEE Transactions on Pattern Analysis and Machine Intelligence 38(10), 1943–1955 (2016). DOI 10.1109/TPAMI.2015.2502579

[63] Zhou, X., Huang, Q., Sun, X., Xue, X., Wei, Y.: Towards 3d human pose estimation in the wild: A weakly-supervised approach. In: The IEEE International Conference on Computer Vision (ICCV) (2017)

Hung-Cuong Nguyen received Engineer in Computer Science (2009), Master in Software Engineering (2010), and Ph.D. in Software Engineering (2015) in Hanoi University of Science and Technology, respectively. He is currently a lecturer at Hung Vuong University, Phu Tho, Vietnam. His research interests include software quality standards, software reliability modeling, computer science, and deep learning.
https://orcid.org/0000-0001-7292-3611

Thi-Hao Nguyen received Bachelor degree at Hung Vuong university (2009). She received M.Sc. degree at Hanoi University of Science and Technology (2013). Currently, she is a lecture of Hung Vuong university. Her research interests include computer science, deep learning.
https://orcid.org/0000-0002-8809-0201

Jakub Nowak received his M.Sc. degree in computer science from the Czestochowa University of Technology, Poland, in 2016, and his Ph.D. degree in computer science at Czestochowa University of Technology in 2021. His present research interests include machine learning, neural networks, text and image analysis for computer system security.
https://orcid.org/0000-0002-1572-3426

Aleksander Byrski obtained Ph.D. in 2007, D.Sc in 2013 and full professorship in 2019 in computer science. He works at AGH University of Science and Technology, Institute of Computer Science. His research focuses on computational intelligence, mostly on metaheuristics, agent-based systems and simulations.
https://orcid.org/0000-0001-6317-7012

Agnieszka Siwocha received M.Sc. the degree from Lodz University of Technology, Faculty of Technical Physics, Computer Science and Applied Mathematics, and her a Ph.D. in 2015 in the field of computer science, computer graphics at the University of Social Sciences, Łódź, Poland. She is currently an assistant professor at the Social Academy of Sciences in Łódź. She has a title of Adobe Certified Expert, and provides training on Adobe software and computer graphics. Author of over 20 publications related to various problems of computer science, computer graphics and IT applications. Her present research interests include fractal coding, compression, and the quality of digital images, computer graphics, machine learning, and multifractal analysis.
https://orcid.org/0000-0003-2562-236X

Van-Hung Le received M.Sc. degree at Faculty Information Technology Hanoi National University of Education (2013). He received Ph.D. degree at International Research Institute MICA HUSTCNRS/UMI - 2954 - INP Grenoble (2018). Currently, he is a lecture of Tan Trao University. His research interests include Computer vision, RANSAC and RANSAC variation and 3-D object detection, recognition; machine leaning, deep learning.
https://orcid.org/0000-0003-4302-0581