Manual Annotations on Depth Maps for Human Pose Estimation

Andrea D'Eusanio, Stefano Pini, Guido Borghi, Roberto Vezzani, and Rita Cucchiara

University of Modena and Reggio Emilia, Italy
Department of Engineering "Enzo Ferrari"
{s.pini, name.surname}@unimore.it

Abstract. Few works tackle the *Human Pose Estimation* on depth maps. Moreover, these methods usually rely on automatically annotated datasets, and these annotations are often imprecise and unreliable, limiting the achievable accuracy using this data as ground truth. For this reason, in this paper we propose an annotation refinement tool of human poses, by means of body joints, and a novel set of fine joint annotations for the *Watch-n-Patch* dataset, which has been collected with the proposed tool. Furthermore, we present a fully-convolutional architecture that performs the body pose estimation directly on depth maps. The extensive evaluation shows that the proposed architecture outperforms the competitors in different training scenarios and is able to run in real-time.

Keywords: Human Pose Estimation · Depth Maps · Body Joints

1 Introduction

In recent years, the task of estimating the human pose has been widely explored in the computer vision community. Many deep learning-based algorithms that tackles the 2D human pose estimation have been proposed [5,19,22] along with a comprehensive set of annotated datasets, collected both in real world [1,8,11] or in simulations [7,17]. However, the majority of these works and data collections are based on standard intensity images (*i.e.* RGB and gray-level data) while datasets and algorithms based only on depth maps, *i.e.* images in which the value of each pixel represents the distance between the acquisition device and that point in the scene, have been less explored, even though this kind of data contains fine 3D information and it can be used in particular settings, like the automotive one [4,18], since depth maps are usually acquired through IR.
A milestone in the human pose estimation on depth maps is the work of Shotton *et al.* [15], based on the *Random Forest* algorithm, that has been implemented in both commercial versions of the *Microsoft Kinect SDK*. This real time algorithm has been widely used to automatically produce body joints annotations in depth-based public datasets. However, these annotations have limited accuracy: in [15], the authors report a mean average precision of 0.655 on synthetic data with full rotations.

For these reasons, in this paper we present *Watch-R-Patch*, a novel refined set of annotations of the well-known *Watch-n-Patch* dataset [20], which contains annotations provided by Shotton *et al.* 's method [15].
Original wrong, imprecise, or missing body joints have been manually corrected for 20 training sequences and 20 testing sequences, equally split between the different scenarios of the dataset, *i.e. office* and *kitchen*.
Furthermore, we present a deep learning-based architecture, inspired by [5], that performs the human pose estimation on depth images only. The model is trained combining the original *Watch-n-Patch* dataset with the manually-refined annotations, obtaining remarkable results. Similar to [15], the proposed system achieves real time performance and can run at more than 180 fps.

2 Related Work

The majority of the literature regarding the Human Pose Estimation task is focused on intensity images [6,13,22]. In [19] a sequential architecture is proposed in order to learn implicit spatial models. Dense predictions, that corresponds to the final human body joints, are increasingly refined through different stage into the network model. The evolution of this method [5] introduces the concept of *Part Affinity Fields* that allows learning the links between the body parts of each subject present in the image.

Only a limited part of works is based on depth maps, *i.e.* images that provide information regarding the distance of the objects in the scene from the camera, One plausible limitation of depth-based methods is the lack of rich depth-based datasets which have been specifically collected for the human pose estimation task and contains manual body joint annotations. Indeed, available datasets are often small, both in terms of number of annotated frames and in terms of subjects. limiting their usability for the training of deep neural networks. In 2011, a method to quickly predict the positions of body joints from a single depth image was proposed in [15]. An object recognition approach is adopted, in order to shift the human pose estimation task in a per-pixel classification problem. The method is based on the random forest algorithm and on a wide annotated dataset, which has not been publicly released. A viewpoint invariant model for the human pose estimation was recently proposed in [9], in which a discriminant model embeds local regions into a particular feature space. This work is based on the *Invariant-Top View Dataset*, a dataset with frontal and top-view recordings of the subjects.

Recently, approaches performing the head detection directly on depth maps were proposed in [2, 3]. In [3], a shallow deep neural network is exploited to classify depth patches as head or non-head in order to obtain an estimation of the head centre joint. The *Watch-n-Patch* dataset [20,21] has been collected for the unsupervised learning of relations and actions task. Its body joints annotation are obtained applying an off-the-shelf method [15], therefore they are not particularly accurate, in particular when subjects stand in a non-frontal position.

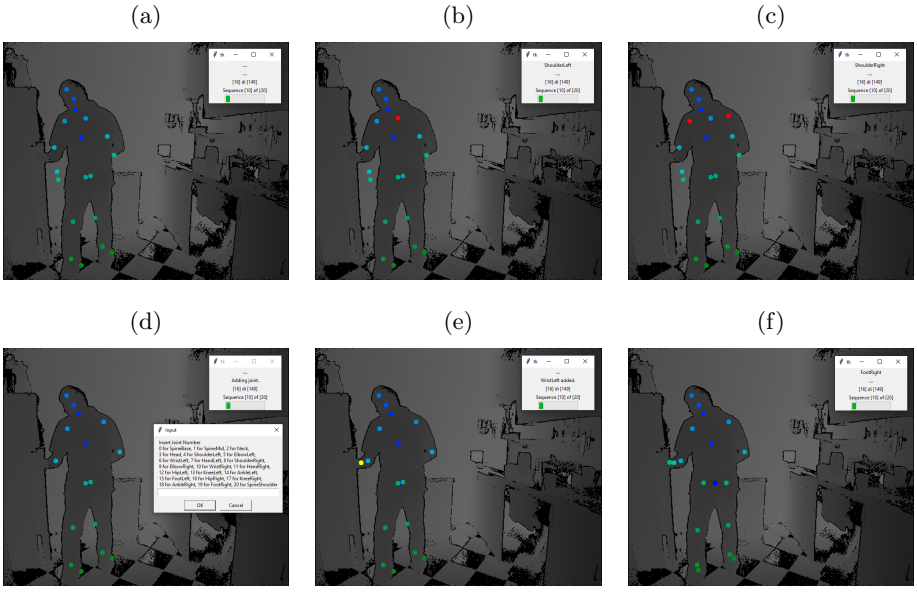

Fig. 1. Annotation tool overview. The tool initially shows the original joint locations (a). Then, each joint can be selected to view its name (b) or to move it in the correct location (c). Missing joints can be added in the right position (d) (e). Finally, the annotations (f) can be saved and the next sequence frame is shown.

3 Dataset

In this section, we firstly report an overview of the *Watch-n-Patch* dataset [20]. Then, we present the procedure we used to improve the original joint annotations and the statistics of the manually refined annotations which are referred as *Watch-R(efined)-Patch*. The dataset will be publicly available[1].

3.1 *Watch-n-Patch* dataset

Watch-n-Patch [20] is a challenging RGB-D dataset acquired with the second version of the *Microsoft Kinect* sensor: differently from the first one, it is a *Time-of-Flight* depth device. The dataset contains recordings of 7 people performing 21 different kinds of actions. Each recording contains a single subject performing multiple actions in one room chosen between 8 offices and 5 kitchens.
The dataset contains 458 videos, corresponding to about 230 minutes and 78k frames. The authors provide both RGB and depth frames (with a spatial resolution of 1920×1080 and 512×424, respectively) and human body skeletons (composed of 25 body joints) estimated and tracked with the method proposed in [15].

[1] *Watch-R-Patch*: http://imagelab.ing.unimore.it/depthbodypose

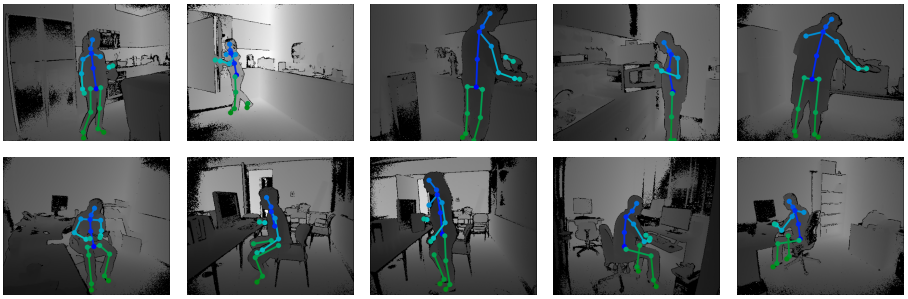

Fig. 2. *Watch-R-Patch* dataset overview. Kitchen and office sequences are shown in the first and second row, respectively.

3.2 Annotation Procedure

We collect refined annotations for the *Watch-n-Patch* dataset using a quick and easy-to-use annotation tool. In particular, we develop a system that shows the original body joints (*i.e.* the *Watch-n-Patch* joints) on top of the acquired depth image. The user is then able to move the incorrect joints in the proper positions using the mouse in a drag-and-drop fashion. Once every incorrect joint has been placed in the correct location, the user can save the new annotation and move to the next frame. It is worth noting that, in this way, the user has only to move the joints in the wrong position while already-correct joints do not have to be moved or inserted. Therefore, original correct joints are preserved, while improving wrongly-predicted joints. We have ignored finger joints (tip and thumb) since original annotations are not reliable and these joints are often occluded. An overview of the developed annotation tool is shown in Figure 1. The annotation tool is publicly released[2].

3.3 Statistics

We manually annotate body joints in 20 sequences from the original training set and 20 sequences from the original testing set. Sequences are equally split between office and kitchen sequences. To speed up the annotation procedure and increase the scene variability, we decided to fine-annotate a frame every 3 frames in the original sequences. In some test sequences, every frame has been fine-annotated. The overall number of annotated frames is 3329, 1135 in the training set, 766 in the validation one, and 1428 in the testing one. We also propose an official validation set for the refined annotations, composed of a subset of the testing set, in order to standardize the validation and testing procedures.
For additional statistics regarding the annotated sequences and the proposed train, validation, and test splits, please refer to Table 1. A qualitative overview of the dataset is reported in Figure 2.

[2] Annotation tool: https://github.com/aimagelab/human-pose-annotation-tool

Table 1. Statistics of the *Watch-R-Patch* dataset.

Split	Sequences		Frames	Annotated frames	Modified joints (%)	mAP
	Kitchen	Office				
Train	data_02-28-33 data_03-22-44 data_03-38-20 data_03-42-37 data_03-46-49 data_03-50-38 data_04-07-17 data_04-17-37 data_04-31-11 data_04-34-13	data_01-50-09 data_03-28-59 data_04-02-43 data_04-31-13 data_04-41-55 data_04-47-41 data_04-56-00 data_05-31-10 data_05-34-47 data_12-03-57	3385	1135	0.757	0.574
Val	data_01-52-55 data_03-53-06 data_04-52-02	data_02-32-08 data_02-50-20 data_03-25-32	995	766	0.643	0.600
Test	data_02-10-35 data_03-45-21 data_04-13-06 data_04-27-09 data_04-51-42 data_05-04-12 data_12-07-43	data_03-04-16 data_03-05-15 data_03-21-23 data_03-35-07 data_03-58-25 data_04-30-36 data_11-11-59	2213	1428	0.555	0.610
Overall	-	-	6593	3329	0.644	0.595

4 Proposed Method

In the development of the human pose estimation architecture, we focus on both the performance (in terms of *mean Average Precision* (mAP)) and the speed (in terms of *frames per second* (fps)).
To guarantee high performance, we decided to develop a deep neural network derived from [5] while, to guarantee high fps, even on cheap hardware, we do not include the *Part Affinity Fields* section (for details about PAF, see [5]).

4.1 Network Architecture

An overview of the proposed architecture is shown in Figure 3.

The first part of the architecture is composed of a VGG-like feature extraction block which comprises the first 10 layers of VGG-19 [16] and two layers that gradually reduce the number of feature maps to the desired value. In contrast to [5], we do not use ImageNet pre-trained weights and we train these layers from scratch in conjunction with the rest of the architecture since the input is represented by depth maps in place of RGB images.

The feature extraction module is followed by a convolutional block that produces an initial coarse prediction of human body joints analyzing the image features extracted by the previous block only. The output of this part can be expressed as:

$$\mathbf{P}^1 = \phi(\mathbf{F}, \theta^1) \tag{1}$$

where $\mathbf{F}$ are the feature maps computed by the feature extraction module and ϕ is a parametric function that represents the first convolutional block of the architecture with parameters θ^1. Here, $\mathbf{P}^1 \in \mathbb{R}^{k \times w \times h}$.

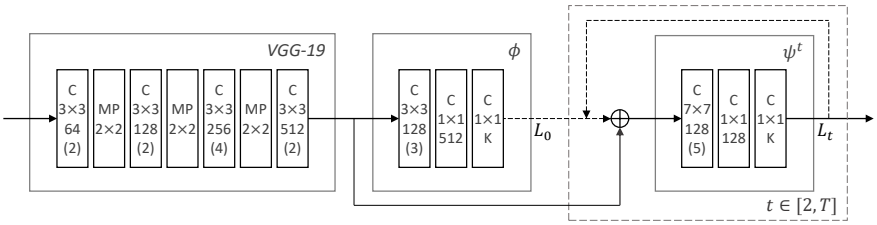

Fig. 3. Proposed architecture overview. Each block contains its type (C: convolution, MP: max pool), the kernel size, the number of feature maps, and the number of repetitions (if higher than 1). In our experiments, $K = 21$ and $T = 6$.

Then, a multi-stage architecture is employed. A common convolutional block is sequentially repeated $T - 1$ times in order to gradually refine the body joint prediction. At each stage, this block analyzes the concatenation of the features extracted by the feature extraction module and the output of the previous stage, refining the earlier prediction. The output at each step can be represented with

$$\mathbf{P}^t = \psi^t(\mathbf{F} \oplus \mathbf{P}^{t-1}, \theta^t) \quad \forall t \in [2, T] \tag{2}$$

where $\mathbf{F}$ are the feature maps computed by the feature extraction module, $\mathbf{P}^{t-1}$ is the prediction of the previous block, $\oplus$ is the concatenation operation, and ψ^t is a parametric function that represents the repeated convolutional block of the architecture with parameters θ^t. As in the previous case, $\mathbf{P}^t \in \mathbb{R}^{k \times w \times h}$.

The model is implemented in the popular framework *Pytorch* [14]. Further details regarding the network architecture are reported in Figure 3.

4.2 Training Procedure

The architecture is trained in an end-to-end manner applying the following objective function

$$L^t = \sum_{k=1}^{K} \alpha_k \cdot \sum_{\mathbf{p}} \|\mathbf{P}_k^t(\mathbf{p}) - \mathbf{H}_k(\mathbf{p})\|_2^2, \tag{3}$$

where K is the number of considered body joints, α_k is a binary mask with $\alpha_k = 0$ if the annotation of joint k is missing, t is the current stage, and $\mathbf{p} \in \mathbb{R}^2$ is the spatial location.
Here, $\mathbf{P}_k^t(\mathbf{p})$ represents the prediction at location $\mathbf{p}$ for joint k while $\mathbf{H}_k \in \mathbb{R}^{w \times h}$ is the ground-truth heatmap for joint k, defined as

$$\mathbf{H}_k(\mathbf{p}) = e^{-||\mathbf{p} - \mathbf{x}_k||_2^2 \cdot \sigma^{-2}} \tag{4}$$

where $\mathbf{p} \in \mathbb{R}^2$ is the location in the heatmap, $\mathbf{x}_k \in \mathbb{R}^2$ is the location of joint k, and σ is a parameter to control the Gaussian spread. We set $\sigma = 7$.
Therefore, the overall objective function can be expressed as $L = \sum_{t=1}^{T} L^t$ where T is the number of stages. In our experiments, $T = 6$.

As outlined in [5], applying the supervision at every stage of the network mitigates the *vanishing gradient* problem and, in conjunction with the sequential refining of the body joint prediction, leads to a faster and more effective training of the whole architecture.

The network is trained in two steps. In the first stage, the original body joint annotations of *Watch-n-Patch* are employed to train the whole architecture from scratch. It is worth noting that the *Watch-n-Patch* body joints are inferred by the *Kinect SDK* which makes use of a random forest-based algorithm [15].
In the second stage, the network is finetuned using the training set of the presented dataset. During this phase, we test different procedures. In the first tested procedure, the whole architecture is fine-tuned, in the second one the feature extraction block is frozen and not updated, while in the last procedure all the blocks but the last one are frozen and not updated.

During both training and finetuning, we apply data augmentation techniques and dropout regularization to improve the generalization capabilities of the model. In particular, we apply random horizontal flip, crop (extracting a portion of 488×400 from the original image with size 512×424), resize (to the crop dimension), and rotation (degrees in $[-4°, +4°]$). Dropout is applied between the first convolutional block and each repeated block.

In our experiments, we employ the *Adam* optimizer [10] with $\alpha = 0.9$, $\beta = 0.999$, and weight decay set to $1 \cdot 10^{-4}$. During the training phase, we use a learning rate of $1 \cdot 10^{-4}$ while, during the finetuning step, we use a learning rate of $1 \cdot 10^{-4}$ and apply the dropout regularization with dropout probability of 0.5.

5 Experimental Results

5.1 Evaluation Procedure

We adopt an evaluation procedure following what proposed for the COCO Keypoints Challenge on the COCO website [12].

In details, we employ the *mean Average Precision* (mAP) to assess the quality of the human pose estimations compared to the ground-truth positions. The mAP is defined as the mean of 10 Average Precision calculated with different *Object Keypoint Similarity* (OKS) thresholds:

$$\text{mAP} = \frac{1}{10} \sum_{i=1}^{10} \text{AP}^{\text{OKS}=0.45+0.05i} \tag{5}$$

The OKS is defined as

$$\text{OKS} = \frac{\sum_i^K [\delta(v_i > 0) \cdot \exp \frac{-d_i^2}{2s^2k_i^2}]}{\sum_i^K [\delta(v_i > 0)]} \tag{6}$$

where d_i is the Euclidean distance between the ground-truth and the predicted location of the keypoint i, s is the area containing all the keypoints, and k_i is

Table 2. Comparison of the mAP reached by different methods computed on the *Watch-R-Patch* dataset. See Section 4 for further details.

	Shotton *et al.* [15]	Ours$_{orig}$	Ours$_{last}$	Ours$_{blk}$	**Ours**
AP$^{\text{OKS=0.50}}$	0.669	0.845	0.834	0.894	**0.901**
AP$^{\text{OKS=0.75}}$	0.618	0.763	0.758	0.837	**0.839**
mAP	0.610	0.729	0.726	0.792	**0.797**

defined as $k_i = 2\sigma_i$. Finally, v_i is a visibility flag: $v_i = 0$ means that keypoint i is not labeled while $v_i = 1$ means that keypoint i is labeled.
The values of $\boldsymbol{\sigma}$ depend on the dimension of each joint of the human body. In particular, we use the following values: $\sigma_i = 0.107$ for the spine, the neck, the head, and the hip joints; $\sigma_i = 0.089$ for the ankle and the foot joints; $\sigma_i = 0.087$ for the knee joints; $\sigma_i = 0.079$ for the shoulder joints; $\sigma_i = 0.072$ for the elbow joints; $\sigma_i = 0.062$ for the wrist and the hand joints.

5.2 Results

Following the evaluation procedure described in Section 5.1, we perform extensive experimental evaluations in order to assess the quality of the proposed dataset and method. Results are reported in Table 2.

Firstly, we have assessed the accuracy obtained by our architecture after a training step employing the original *Watch-n-Patch* dataset. This experiment corresponds to Ours$_{orig}$ in Table 2. As expected, when trained on the Kinect annotations, our model is capable of learning to predict human body joints accordingly to the Shotton *et al.* 's method [15], reaching a remarkable mAP of 0.777 on the *Watch-n-Patch* testing set.
We also test the performance of the network employing our annotations as the ground-truth. In this case, our method reach a mAP of 0.729, outperforming the Shotton *et al.* 's method with an absolute margin of 0.119. It is worth noting that our method has been trained on the Kinect annotations only, but the overall performance on the manually-annotated sequences is considerably higher than the one of [15]. We argue that the proposed architecture has better generalization capabilities than the method proposed in [15], even if it has been trained on the predictions of [15], therefore it obtains a higher mAP when tested on scenes with actual body joint annotations.

Then, we report the results obtained applying different finetuning procedures. In particular, we firstly train the proposed network on the original *Watch-n-Patch* annotations then we finetune the model with the proposed annotations updating different parts of the architecture. In the experiment Ours$_{last}$, we freeze the parameters of all but the last repeated block, which means updating only the parameters θ^6 of the last convolutional block ψ^6. In Ours$_{blk}$, we freeze the parameters of the feature extraction block, *i.e.* only the parameters θ^t of ϕ and ψ^t are updated. Finally, we finetune updating the whole network in the experiment Ours. As shown in Table 2, finetuning the whole architecture leads

Table 3. mAP of each body joint present in the *Watch-R-Patch* dataset.

Joint	Shotton *et al.* [15]	Ours$_{orig}$	**Ours**
SpineBase	0.832	0.841	**0.905**
SpineMid	0.931	0.911	**0.935**
Neck	**0.981**	0.975	0.978
Head	**0.971**	0.961	0.962
ShoulderLeft	0.663	0.673	**0.819**
ElbowLeft	0.490	0.635	**0.772**
WristLeft	0.456	0.625	**0.677**
HandLeft	0.406	0.599	**0.680**
ShoulderRight	0.538	0.547	**0.782**
ElbowRight	0.454	0.618	**0.748**
WristRight	0.435	0.642	**0.727**
HandRight	0.412	0.641	**0.712**
HipLeft	0.646	0.766	**0.824**
KneeLeft	0.494	0.743	**0.788**
AnkleLeft	0.543	0.771	**0.800**
FootLeft	0.497	0.743	**0.801**
HipRight	0.696	0.778	**0.860**
KneeRight	0.493	0.670	**0.763**
AnkleRight	0.508	0.630	**0.648**
FootRight	0.388	**0.605**	0.605
SpineShoulder	**0.969**	0.942	0.955

to the highest $AP^{OKS=0.50}$, $AP^{OKS=0.75}$, and mAP scores. The proposed model, trained on the original *Watch-n-Patch* dataset and finetuned on the presented annotations, reaches a remarkable mAP of 0.797, outperforming the Shotton *et al.* 's method with an absolute gain of 0.187.

Finally, we report per-joint mAP scores in Table 3. As it can be observed, the proposed method outperforms the competitor and the baseline in nearly every joint prediction, confirming the capabilities and the quality of the model and the employed training procedure. Qualitative results are reported in Figure 1.
The model is able to run in real-time (5.37ms, 186fps) on a workstation equipped with an *Intel Core* i7-6850K and a GPU *Nvidia 1080 Ti*.

6 Conclusions

In this paper we have investigated the human pose estimation on depth maps. We have proposed a simple annotation refinement tool and a novel set of fine joint annotations for a representative subset of the *Watch-n-Patch* dataset, which we have published free-of-charge. We have presented a deep learning-based architecture that performs the human pose estimation by means of body joints, reaching state-of-the-art results on the challenging fine annotations of the *Watch-R-Patch* dataset. As future work, we plan to publicly release the annotation tool and to complete the annotation of the *Watch-n-Patch* dataset.

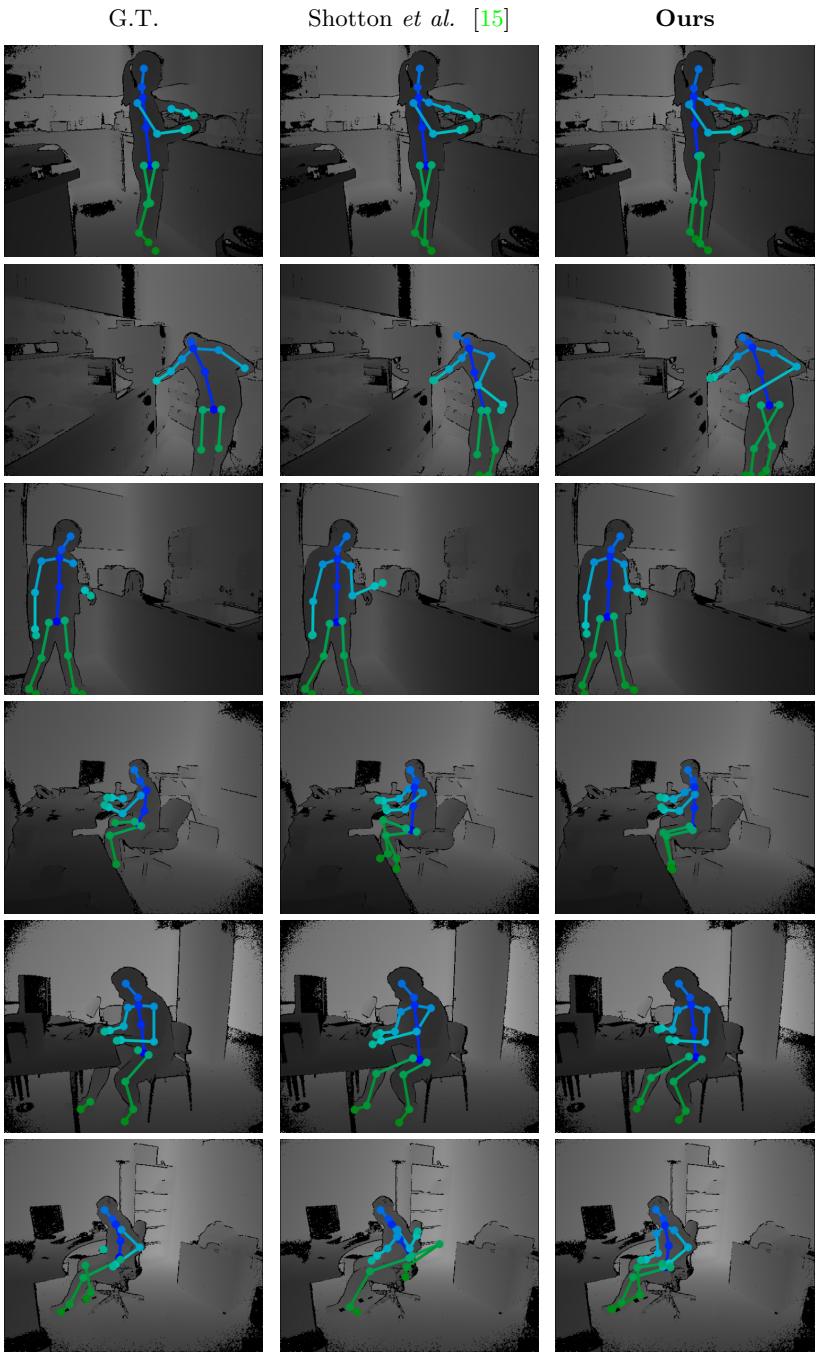

Fig. 4. Qualitative results obtained on the *Watch-R-Patch* dataset.

References

1. Andriluka, M., Pishchulin, L., Gehler, P., Schiele, B.: 2d human pose estimation: New benchmark and state of the art analysis. In: CVPR (2014) 1
2. Ballotta, D., Borghi, G., Vezzani, R., Cucchiara, R.: Fully convolutional network for head detection with depth images. In: 2018 24th International Conference on Pattern Recognition (ICPR). pp. 752–757. IEEE (2018) 2
3. Ballotta, D., Borghi, G., Vezzani, R., Cucchiara, R.: Head detection with depth images in the wild. In: VISAPP (2018) 2
4. Borghi, G., Fabbri, M., Vezzani, R., Cucchiara, R., et al.: Face-from-depth for head pose estimation on depth images. IEEE TPAMI (2018) 1
5. Cao, Z., Simon, T., Wei, S.E., Sheikh, Y.: Realtime multi-person 2d pose estimation using part affinity fields. In: CVPR (2017) 1, 2, 5, 7
6. Carreira, J., Agrawal, P., Fragkiadaki, K., Malik, J.: Human pose estimation with iterative error feedback. In: CVPR (2016) 2
7. Chen, W., Wang, H., Li, Y., Su, H., Wang, Z., Tu, C., Lischinski, D., Cohen-Or, D., Chen, B.: Synthesizing training images for boosting human 3d pose estimation. In: International Conference on 3D Vision (3DV) (2016) 1
8. Güler, R.A., Neverova, N., Kokkinos, I.: Densepose: Dense human pose estimation in the wild. In: CVPR (2018) 1
9. Haque, A., Peng, B., Luo, Z., Alahi, A., Yeung, S., Fei-Fei, L.: Towards viewpoint invariant 3d human pose estimation. In: ECCV. Springer (2016) 2
10. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014) 7
11. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: ECCV. Springer 1
12. COCO - Keypoint Evaluation. http://cocodataset.org/#keypoints-eval 7
13. Newell, A., Yang, K., Deng, J.: Stacked hourglass networks for human pose estimation. In: ECCV. Springer (2016) 2
14. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in PyTorch. In: NIPS Autodiff Workshop (2017) 6
15. Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., Blake, A.: Real-time human pose recognition in parts from single depth images. In: CVPR (2011) 1, 2, 3, 7, 8, 9, 10
16. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014) 5
17. Varol, G., Romero, J., Martin, X., Mahmood, N., Black, M.J., Laptev, I., Schmid, C.: Learning from synthetic humans. In: CVPR (2017) 1
18. Venturelli, M., Borghi, G., Vezzani, R., Cucchiara, R.: Deep head pose estimation from depth data for in-car automotive applications. In: International Workshop on Understanding Human Activities through 3D Sensors. pp. 74–85 (2016) 1
19. Wei, S.E., Ramakrishna, V., Kanade, T., Sheikh, Y.: Convolutional pose machines. In: CVPR (2016) 1, 2
20. Wu, C., Zhang, J., Savarese, S., Saxena, A.: Watch-n-Patch: Unsupervised understanding of actions and relations. In: CVPR (2015) 2, 3
21. Wu, C., Zhang, J., Sener, O., Selman, B., Savarese, S., Saxena, A.: Watch-n-patch: unsupervised learning of actions and relations. IEEE Transactions on Pattern Analysis and Machine Intelligence **40**(2), 467–481 (2018) 2
22. Xiao, B., Wu, H., Wei, Y.: Simple baselines for human pose estimation and tracking. In: ECCV (2018) 1, 2