Top-Down Human Pose Estimation with Depth Images and Domain Adaptation

Nelson Rodrigues[1,*], Helena Torres[1], Bruno Oliveira[1], João Borges[1], Sandro Queirós[1,2], José Mendes[1], Jaime Fonseca[1], Victor Coelho[3] and José Henrique Brito[2],

[1]*Algoritmi Center, University of Minho, Guimarães, Portugal*

[2]*2Ai - Polytechnic Institute of Cávado and Ave, Barcelos, Portugal*

[3]*Bosch, Braga, Portugal*

Keywords: Human Pose, Depth Images.

Abstract: In this paper, a method for estimation of human pose is proposed, making use of ToF (Time of Flight) cameras. For this, a YOLO based object detection method was used, to develop a top-down method. In the first stage, a network was developed to detect people in the image. In the second stage, a network was developed to estimate the joints of each person, using the image result from the first stage. We show that a deep learning network trained from scratch with ToF images yields better results than taking a deep neural network pretrained on RGB data and retraining it with ToF data. We also show that a top-down detector, with a person detector and a joint detector works better than detecting the body joints over the entire image.

1 INTRODUCTION

The main motivation for this project was to develop a system capable of monitoring passengers inside a vehicle. With the evolution of autonomous vehicles, the interaction that the humans will have in the car will have a paradigm completely different from the current one. With autonomous vehicles, the time that was previously spent driving will be used for other activities. Consequently, there is a need to monitor and predict the actions of all passengers inside the vehicle. For this purpose, it is necessary to detect humans and their respective body posture, namely the spatial location of the skeletal joints. To capture quality images of the interior of the vehicle, ToF cameras can be used, as these have a great advantage over RGB cameras, namely their immunity to light conditions. With this type of images, it is possible, through algorithms based on Deep Learning (DL), to estimate the body posture of individuals. There are already methods able to determine the human pose in both RGB and depth images. In this article, a YOLO object detection method was used, to develop a top-down method.

In the first stage, a DL network was developed to detect people in the image. In the second stage, a network was developed to estimate the joints of each person, using the image region of interest (RoI) detected in the first stage. The rest of the paper is organized as follows. Section 2 gives an introduction to the different existing human pose detection methods, as well as some methods used in this article. The modifications made to the object detection method are described in section 3. The results of these same detections are shown in section 4. And finally, section 5 provides the overall conclusions.

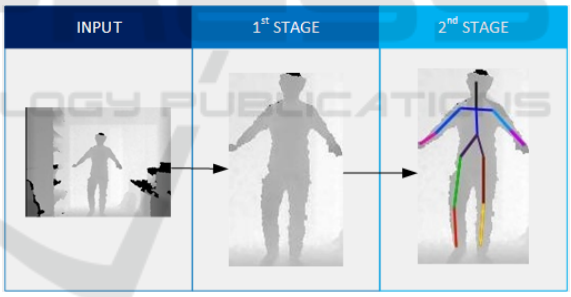

Figure 1: Overview of the proposed top-down YOLO method. The first stage uses a person detector to produce a bounding box around each candidate person. In the second stage, a pose estimator is applied to the image cropped arround each candidate person in order to localize their skeleton's joints.

Rodrigues, N., Torres, H., Oliveira, B., Borges, J., Queirós, S., Mendes, J., Fonseca, J., Coelho, V. and Brito, J.
Top-Down Human Pose Estimation with Depth Images and Domain Adaptation.
DOI: 10.5220/0007344602810288
In *Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2019)*, pages 281-288
ISBN: 978-989-758-354-4

2 PREVIOUS WORK

Human pose estimation in 2D images is usually treated as an object detection task, where the objects to be detected are the skeleton joints of the people appearing in the images.

Felzenszwalb et al. (2010) proposed an object detection system that uses local appearances and spatial relations to recognize generic objects of an image. Generally, this method consists of defining a model that represents the object. The model is constructed by defining a root filter (for the object) and a set of filters (for the parts of the object). These filters are used to study the features of the image. More specifically, the characteristics of the oriented gradient histogram (HoG) are analyzed within each filter to represent an object category. The descriptor calculates the gradients of a region of the image HoG, assuming the object within the image can be described by its intensity transitions. This method uses a sliding window approach, where the filters are applied to all image positions. For the creation of the final model, a discriminative approach is used, where the model learns from annotated data, using bounding boxes around the object. This part is usually performed by an support vector machine (SVM). After the training phase, the model is used to detect the objects in test images. Detection is performed by computing the convolution of the trained part models with the feature map of the test image and selecting the regions of the image with the highest convolution score. One can notice that this method, despite having a discriminative basis, can be interpreted as an adjustment of the image to a model, which involves generative concepts. For this reason, it can be considered a hybrid methodology, and may thus not be trivial to adapt this method to depth images.

The random tree walk (RTW) method presented by Jung et al. (2015) estimates 3D joints from depth images. This work is an evolution of an earlier method proposed by Shotton et al. (2013). The main difference is in the fact that it does not apply a pixel regression for all the pixels in the image and trains a tree to estimate the direction to a specific joint from a random point instead of the distance. RTW only evaluates one pixel at each iteration. When it reaches a leaf in the tree, it will choose a direction. The RTW method will then iteratively converge to the desired joint. This method is executed hierarchically, which means the position resulting from a joint search will be used as the starting point for the next joint to be calculated.

Regarding DL approaches, Cao et al. (2017) proposed a method that uses a VGG (Simonyan and Zisserman, 2015) network to extract features from the image and these features are used as inputs for a CNN with two branches. The first branch is trained and used for joint detection and the second branch is trained with the segments between them, so it is able to detect limbs connecting joints. In the first branch, a feed-forward network is used to provides the confidence maps of the different parts of the body corresponding to their probability maps. These probability maps are a representation of the confidence in each position of the joint that occurs in each pixel and is expressed in a Gaussian function. In the second branch, the part affinity vector fields are constructed, encoding the association between the parts. The part affinity fields allow joint's positions to be assembled into a body posture. A part affinity field is constructed for each member of the body and encodes location and orientation information. The predictions for joint and limb detections produced by the two branches of the network are refined over several stages through an iterative process. The predictions of each branch are used as the input of the next stage. This method is designed to better handle images with more than one person. For this reason, it is unnecessary to implement a method for detecting people, to later detect the joints of each person, which allows to avoid bad detections on the people detector and increases the computation time. As major disadvantages, it requires significant training data and requires the analysis of the entire image.

The method presented by Papandreou et al. (2017) consists of a two-stage approach. The first stage predicts the location and scale of bounding boxes containing people using a Faster R-CNN (Ren et al., 2017) detector. Both the region proposal components and the bounding box classification used in the Faster R-CNN detector were trained using only the person category of the MS COCO (Lin et al., 2014) dataset, with all other categories being ignored. In the second step, for each bounding box proposed in the first step, the 17 key points of the person potentially contained in the box are estimated. For better computational efficiency, the bounding box proposals of people are only sent to the second stage if their score is higher than a threshold (0.3). Using a fully convolutional ResNet, the system predicts two targets, (1) disk-shaped heatmaps around the key points and (2) magnitude of the offset fields toward the precise position within the disk. The system then aggregates these results, producing the activation maps aggregating the results in a weighted voting process, on highly localized activation maps.

The method presented by He et al. (2017), named Mask R-CNN, is an extension of Faster R-CNN

that also outputs a segmentation mask, in addition to the bounding box and class probabilities. Faster R-CNN adopts a two-stage methodology: first, a region proposal network (RPN) produces candidates regions, and then a second stage extracts features for each candidate and outputs its class and bounding box offsets. Mask R-CNN adds an additional branch to estimate segmentation masks for each candidate region, in parallel to the branch that predicts the class and bounding box offsets.

Like Faster-RCNN, the You Only Look Once (YOLO) method presented by Redmon et al. (2016) is an object detector. YOLO is faster because it reformulated the detection of objects as a single regression problem, directly from the pixels of the image to the bounding box coordinates and the class probabilities. A single convolution network simultaneously provides several bounding boxes and their respective classification probabilities. YOLO trains directly with full images. This method can be adapted to follow the same procedure as the Papandreou et al. (2017) method, but just using the object detections in both stages.

In general, DL-based systems for human pose detection take in RBG images and are structured around a DNN trained on a large RGB dataset with joint annotations, such as PASCAL VOC (Everingham et al., 2005) or MS COCO. The most popular human pose detection systems that use ToF images are usually adaptations of the method from Shotton et al. (2013) trained on ToF images also with joint annotations. In our work, the challenge was to leverage DL-based systems, usually designed for RGB images, and to find the best way to adapt them to the new domain provided by ToF cameras. We therefore used a deep learning based detector as a starting point and repurposed it for human joint detection with ToF images, using different strategies, to see what strategies produced the best results. We evaluate our experiments on the iTop dataset Haque et al. (2016), which includes ToF frames and ground truth 2D locations of a 15 joint skeleton.

3 IMPLEMENTATION

This work aims to estimate human pose (joint positions) from depth images. All implementations of CNN-based detectors used the YOLOv3 (Redmon and Farhadi, 2018) as a starting point. YOLO is a DL object detector that outputs bounding box coordinates. YOLOv3 has a hybrid architecture between the YOLOv2 (Redmon and Farhadi, 2017) version (darknet-19) and residual networks with several small improvements. YOLOv3 is available with pre-trained networks for different datasets, like PASCAL VOC and MS COCO, which allows detecting people easily.

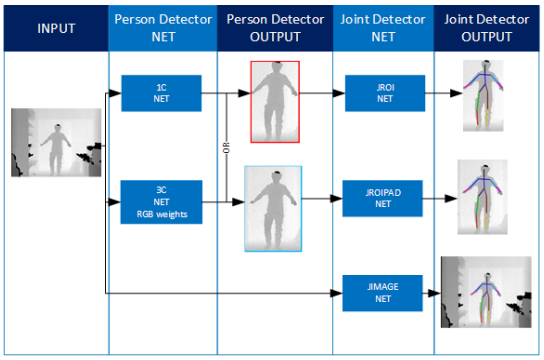

Figure 2: Implementations of all different networks for person detector and joints detector.

All implemented detectors take 416x416 pixel inputs, as in the original implementation. Some of the implemented networks were trained from pre-trained weights for the convolutional layers (using the darknet53.conv.74 weights as a starting point), while some were trained from scratch: for detectors that use 3-channel inputs, training was initialized with the pre-trained weights, while for detectors that use 1-channel inputs, training was initialized with random weights. We tried both approaches to verify if it would make sense to leverage knowledge transfer across domains (RGB to ToF) or if training from scratch in the new domain worked better.

The person detector was implemented using the original YOLOv3 implementation, simply by changing the number of classes to 1, the person class. For the detection of joints, the network was adjusted for the number of classes, so that it would detect the 15 classes corresponding to the 15-joint skeleton provided for each frame in the dataset. For the development of the top-down human pose estimation detectors, as shown in Figure 1, we simply concatenated a person detector with a joint detector, to compose a two-stage system. The first network was trained only to detect people in the image, and the second network was trained with RoIs for joint detection. For the joint detectors, we tried 3 different versions: joint detectors trained on person bounding boxes, trained on padded person bounding boxes, and trained on the whole image (without person detection).

3.1 Person Detector

These networks were trained only to detect persons in the images, so there is only one class, person. We tried using pre-trained weights (darknet53.conv.74) with 3 channel images (simply feeding 3 channels with the same depth information into the network)

Table 1: Parameters values for the person detector network.

Parameter	Value
Classes	1
Coords	4
Number of Masks	3
Filters	18

and training the network from scratch for 1 channel (depth). The RoIs defined by the bounding boxes produced by the person detector are then fed as input to the second stage of the hierarchical pose detector. The RoI can be used as is (red bounding box in Figure 2) or with a 20-pixel padding (blue bounding box in Figure 2).

3.2 Person Pose Estimation

Table 2: Parameters values for the pose estimation network.

Parameter	Value
Classes	15
Coords	4
Number of Masks	3
Filters	60

A separate network detects the position of the joints inside a region. The input region may be the whole image or a RoI provided by the person detector. We use the joint structure provided by the iTOP dataset, a skeleton with 15 joints. The 15 joints are the object classes detected by the joint detectors. Joint detection is formulated as an object detection problem by defining bounding boxes around the ground truth coordinates of each joint provided in the dataset. The bounding boxes are all square in shape, but their size depends on the type of joint. The bounding box sizes for each joint class are presented in Table 3.

Table 3: Bounding box sizes for each joint.

Joint	Size
head	35
neck	35
rShoulder	25
lShoulder	25
rElbow	25
lElbow	25
rHand	30
lHand	30
torso	15
rHip	30
lHip	30
rKnee	25
lKnee	25
rFoot	25
lFoot	25

Figure 3: A) Frame with zero padding arround the pose detection, B) Frame with twenty pixels of padding arround the pose detection.

At inference time, the detector outputs the center of the bounding box as the estimated coordinates of the joint.

Since the bounding box boundaries provided by the person detector may be very close to the boundaries of the silhouette, namely close to the hands, it is difficult to train a method with good results for the bounding boxes of the joints, since important context information might be missed (Figure 3 left). If a 20-pixel padding is added to the person bounding box, the joint bounding boxes will contain more context information, which will be extremely useful for training the joint detector. On the other hand, training the joint detector in the whole image will use all available context information, but might be an unnecessary waste of computational resources. For this reason, as mentioned before, we trained 3 variants for the joint detector: detecting the joints inside the person bounding box, inside a padded person bounding box, and in the whole image. As for the person detectors, the 3-channel version were trained from pretrained weights and 1-channel version were trained from random initializations.

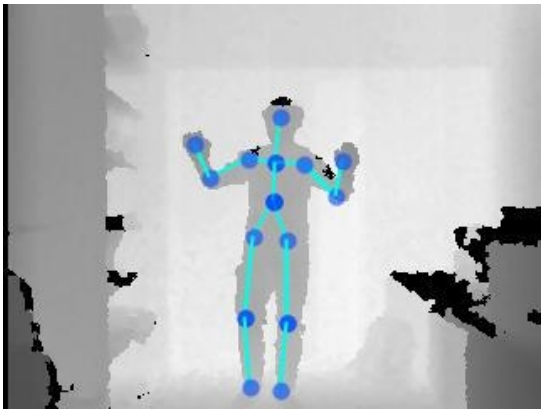

Figure 4: Position of each joint in the silhouette.

4 EXPERIMENTAL EVALUATION

4.1 Experimental Setup

For the training of the method, a server with a NVIDIA Tesla V100 GPU with 16GB was used. All experiments used a momentum of 0.9, a learning rate of 0.001, a batch size of 64 with 4 subdivisions which means that in every step the network reads 16 images. The methods for person detectors were trained for 5 000 iterations and the joints detection were trained for 10 000 iterations.

The iTOP dataset was used both for training and testing. The dataset includes 22660 front view depth images with joint annotations, which was split into a training set with 17991 images and a test set with 4669 images. The original ground truth annotations had to be corrected, as some joints were placed outside the human silhouette. In those situations, the joint coordinates were moved so that they would be placed on the edge of the silhouette. For this procedure, a region growing method was used in order to obtain the RoIs. The human silhouette was then isolated by selecting the object that included the torso joint. After having a segmented human silhouette, a k nearest neighbours (KNN) algorithm was applied in order to move the joints outside the silhouette onto the edges of the silhouette (Figure 5).

4.2 Person Detections

To evaluate person detection, the classic metrics were used, namely Intersection over Union (IoU) above some thresholds, Average Precision and Average Recall, Precision and Recall at 0.5, Precision and Recall at 0.75, following standard practice in object detection challenges such as COCO. As mentioned above, different person detectors were trained for 3-channel images and 1-channel images, using pretrained weights and from random initializations respectively.

Results are shown in Figure 6 for AP, Figure 7 for AR and in supplemental material for P0.5, R0.5, P0.75 and R0.75.

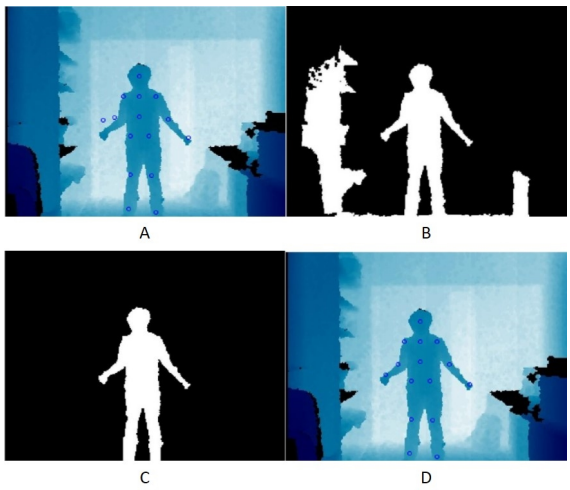

Figure 5: A) Annotations outsite the human silhouette, B) Result of the region growing algorithm, C) Human silhouette object selected, D) Applied KNN algorithm in order to move the joints to the edge of the human silhouette.

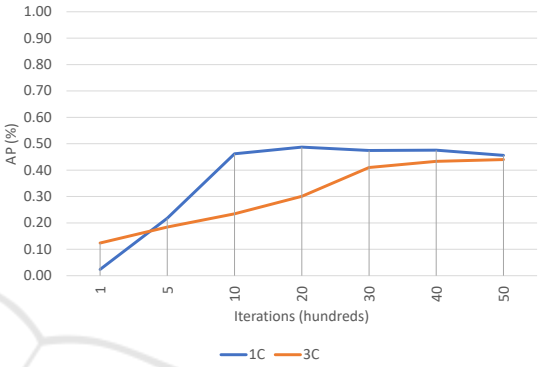

Figure 6: Average precision results over training iterations.

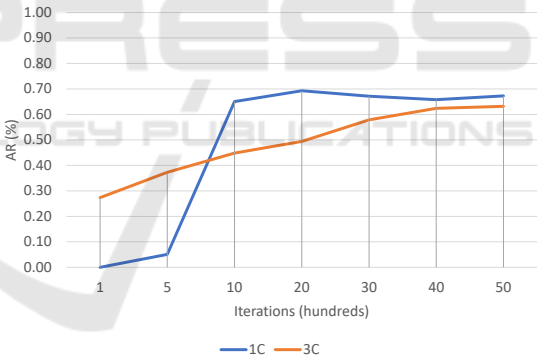

Figure 7: Average recall results over training iterations.

In the first few iterations, the pretrained RGB weights seem to be able to encapsulate some information about the behavior of depth images, and retraining them on 3-channel ToF images yields better results than random weights with trained on very little 1-channel ToF data. However, after a few hundred training iterations, the more compact 1-channel representation allows the 1-channel network to learn better and faster from ToF images, although the difference in performance is not very large, if enough training iterations are allowed to be executed. We therefore conclude that training a 1-channel detector from scratch is better than retraining RGB weights, as the 1-channel representation is a more compact represen-

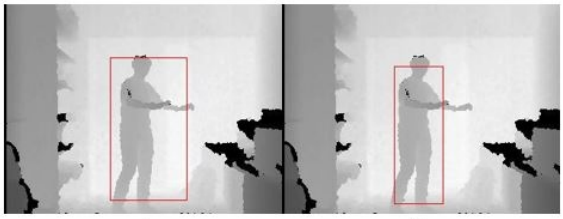

Figure 8: A) Person detection for one channel without pre-trained weights, B) Person detection for three channels network with pre-trained weights.

tation, from which a detector learns more efficiently than from a 3-channel representation, where the same information is repeated in the 3 channels.

4.3 Person Pose Estimations

Having determined that training with 1-channel ToF images is more efficient for person detection, all pose estimation detectors took 1-channel ToF images with random initial weights for joint detection.

To evaluate pose estimation, the considered metrics were not based on IoU. Instead, we compute Average (Euclidian) Distance (in cm) between the detected joint coordinates and ground truth coordinates (AvD), mean Precision and mean Recall of joint detections considering detections within some threshold distance (5 cm and 10 cm) as true positives (mP5cm, mP10cm, mR5cm, mR10cm), and again Average Distance but considering only joints that were detected within some threshold distance (5 cm and 10 cm) (AvDT5cm, AvDT10cm). These metrics make more sense than classic region-based object detection metrics, as the system is truly estimating point positions, rather than object positions.

Figure 9 and Figure 10 show the results for AP5cm and AR5cm respectively, and AP10cm, AR10cm, AD, AD5cm, AD10cm are included in supplemental materials. The results are shown for different topologies, considering joint detection on the whole image, joint detection in padded RoIs and joint detection for standard RoI, for different numbers of training iterations, and using 3-channel data.

Overall, the topology that yielded the best results was the one where the joint detector uses standard padded ROI as input. The results using the whole image are also similar but not quite as good as using just the padded ROI. Results for joint detection using the standard ROI are significantly worse. When the whole image is used for joint detection, the network is more prone to make mistakes in joint detection. If the network is progressively fed with inputs that are more constrained to the true position of the joints, the performance also progressively increases, so feeding the

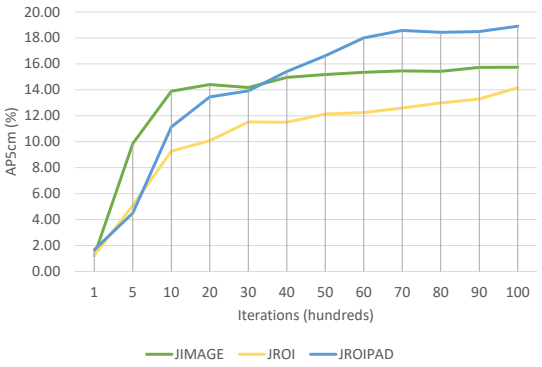

Figure 9: Average precision for a threshold of 5 cm over training iterations.

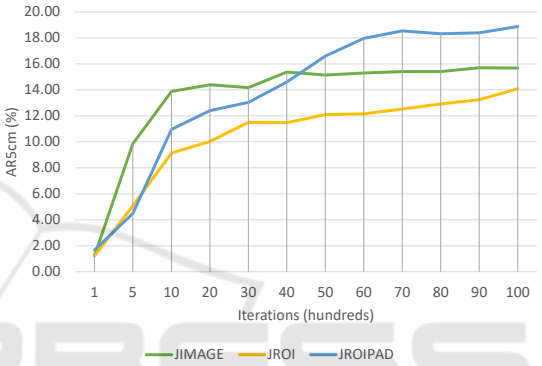

Figure 10: Average recall for a threshold of 5 cm over training iterations.

standard padded ROI gives the best result. We originally anticipated that using padded ROIs would yield better results than using standard ROIs, as the detection of joints that are closer to the edge of the ROI would benefit from having more visual context available for those detections. Indeed, the results achieved with padded ROI were much higher than the standard.

5 CONCLUSION

In this work, we have shown how to repurpose a deep learning object detector, originally trained with RGB images, for a different task using ToF images. We have shown that it is preferable to train the whole network from scratch with ToF images, rather than take trained RGB weights and retrain them with ToF images. We have also shown that a top-down hierarchical detector works better than just using the joint detector on the entire image, as the person detector constrains the search for the joint detector, enabling it to make less mistakes during joint detection. However, constraining the search to ROIs hampers the

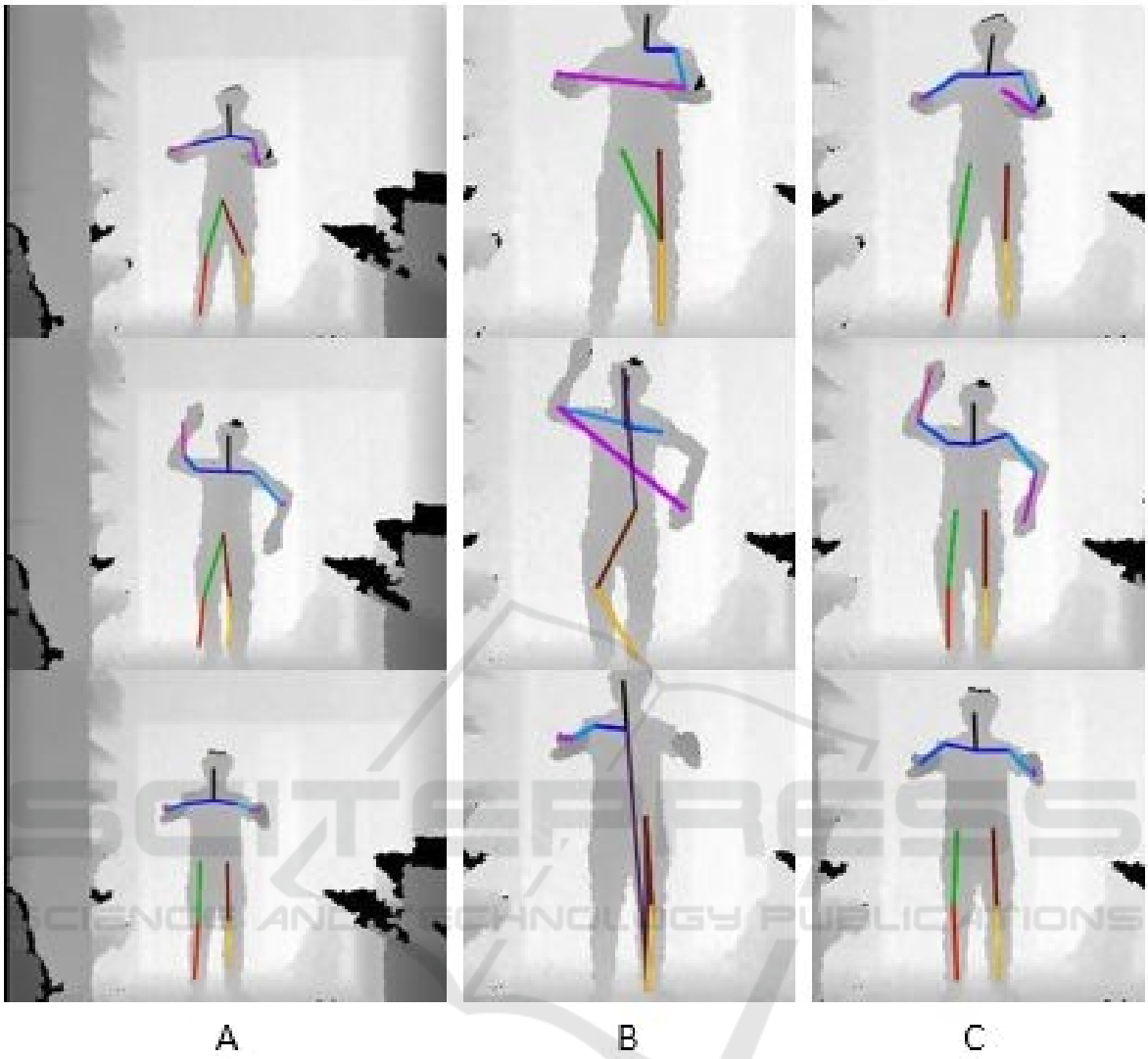

Figure 11: Pose estimations results for the differents networks: A) JImage, B) JRoI, C) JRoIPad.

body joint detector for joints that are close to the ROI boundary, as less visual context information is available for those joints. Detecting joints on padded ROIs did in fact significantly change the results, and enabled the system to be more effective for joints near the ROI boundary.

For future work, we plan to try the same approach using other deep learning based detectors, possibly combining the YOLO based ToF person detector with a different joint detector, such as Cao et al. (2017) also trained from scratch with random weights with ToF images. To be able to address the in-car scenario, which is our ultimate goal, we are currently developing a dataset with in-car images in order to apply this solution in this type of images.

ACKNOWLEDGEMENTS

This work is supported by: European Structural and Investment Funds in the FEDER component, through the Operational Competitiveness and Internationalization Programme (COMPETE 2020) [Project no 002797; Funding Reference: POCI-01-0247-FEDER-002797].

REFERENCES

Cao, Z., Simon, T., Wei, S. E., and Sheikh, Y. (2017). Realtime multi-person 2D pose estimation using part affinity fields. *Proceedings - 30th IEEE Conference*

on Computer Vision and Pattern Recognition, CVPR 2017, 2017-Janua:1302–1310.

Everingham, M., Zisserman, A., Williams, C. K. I., Van Gool, L., and Al., A. (2005). The 2005 PASCAL Visual Object Classes Challenge. *First PASCAL Machine Learning Challenges Workshop, MLCW 2005*, 3944:117–176.

Felzenszwalb, P. F., Girshick, R. B., McAllester, D., and Ramanan, D. (2010). Object Detection with Discriminative Trained Part Based Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645.

Haque, A., Peng, B., Luo, Z., Alahi, A., Yeung, S., and Fei-Fei, L. (2016). Towards Viewpoint Invariant 3D Human Pose Estimation. In *ECCV*, pages 160–177.

He, K., Gkioxari, G., Dollar, P., and Girshick, R. (2017). Mask R-CNN. *Proceedings of the IEEE International Conference on Computer Vision*, 2017-Octob:2980–2988.

Jung, H. Y., Lee, S., Heo, Y. S., and Yun, I. D. (2015). Random tree walk toward instantaneous 3D human pose estimation. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 07-12-June, pages 2467–2474.

Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft COCO: Common objects in context. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8693 LNCS(PART 5):740–755.

Papandreou, G., Zhu, T., Kanazawa, N., Toshev, A., Tompson, J., Bregler, C., and Murphy, K. (2017). Towards accurate multi-person pose estimation in the wild. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017-Janua:3711–3719.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). YOLO You Only Look Once: Unified, Real-Time Object Detection. *Cvpr 2016*, pages 779–788.

Redmon, J. and Farhadi, A. (2017). YOLO9000: Better, faster, stronger. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017-Janua:6517–6525.

Redmon, J. and Farhadi, A. (2018). YOLOv3: An Incremental Improvement.

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

Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., and Blake, A. (2013). Real-time human pose recognition in parts from single depth images. *Studies in Computational Intelligence*, 411:119–135.

Simonyan, K. and Zisserman, A. (2015). VGG : Very Deep Convolutional Networks for Large-Scale Image Recognition. *International Conference on Learning Representations (ICRL)*, pages 1–14.

APPENDIX

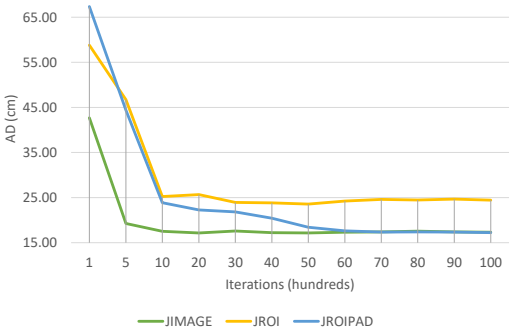

Figure 12: Average distance over the training iterations.

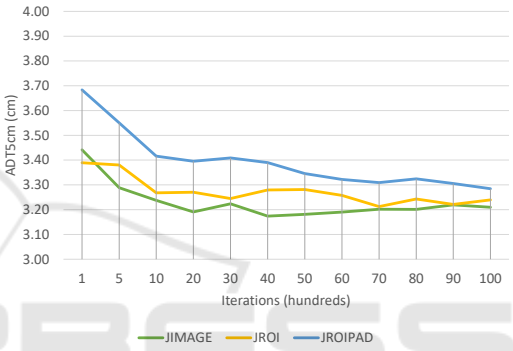

Figure 13: Average distance for a threshold of 5 cm for the joints correctly detected over the training iterations.

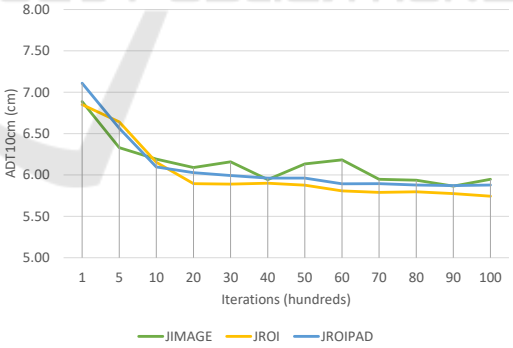

Figure 14: Average distance for a threshold of 10 cm for the joints correctly detected over the training iterations.