Revolutionizing Media and Gaming with AI: Advancements in Body Measurement Calculation, Motion Tracking, Gesture Recognition, and Upper Limb Segmentation

Gilda Manfredi[1,*], Nicola Capece[1], Ugo Erra[1] and Monica Gruosso[1]

[1]University of Basilicata, via dell'Ateneo Lucano 10, Potenza, 85100, Italy

Abstract

This contribution discusses the potential of Artificial Intelligence (AI) to enhance Human-Computer Interaction (HCI) methods. Researchers at the Laboratory of Computer Graphics and Parallel Computing at the University of Basilicata have developed several AI-based systems for HCI applications. These systems include an upper limb segmentation system, an XR gesture recognition system, and a virtual dressing room that utilizes Body Tracking and Anthropometric Measurement Systems. The systems use deep learning algorithms to accurately track body movements and interpret hand gestures in real-time, creating a more natural and intuitive interaction with XR environments. The virtual dressing room enables users to create a 3D model of themselves and try on virtual clothing and accessories, ensuring a perfect fit through Anthropometric Measurement System calculations. These AI-based systems have significant potential to enhance user experience and interaction in the HCI field.

Keywords

human-computer interaction, motion tracking, gesture recognition, deep learning, extended reality

1. Introduction

In recent years, researchers have considered AI a transformative technology [1, 2] as it affects nearly every aspect of human life. One of the areas where AI is showing tremendous potential is in the realm of media and gaming. In particular, the development of eXtended Reality (XR) applications, which encompass virtual, augmented, and mixed-reality experiences, has led to a search for new and innovative ways to engage and immerse users. Starting from the development of XR applications, researchers are exploring new ways of interaction with these platforms that go beyond traditional HCI methods, such as mouse and keyboard input. Such researches are due because traditional HCI methods limit the ability of users to immerse themselves in XR environments [3]. Therefore, to enhance the user experience, researchers are turning to AI to develop more natural and intuitive forms of interaction.

This contribution presents an overview of some of the main research activities focused on AI for the HCI field, conducted by the Laboratory of Computer Graphics and Parallel Computing of the University of Basilicata. This includes several AI-based systems for HCI applications. One of these systems is an upper limb segmentation system, which utilizes deep learning techniques for upper limb segmentation in egocentric vision and unconstrained real-world scenarios. The laboratory has also developed an XR gesture recognition system, which utilizes machine learning algorithms to recognize and interpret hand gestures in real-time within virtual and augmented reality environments. Furthermore, the laboratory has created a virtual dressing room that utilizes Body Tracking and Anthropometric Measurement Systems. This system enables users to create a realistic 3D model of themselves, which can be used to try on virtual clothing and accessories. The Body Tracking system uses deep learning algorithms to accurately track the user's body movements and apply them to the virtual model. At the same time, the Anthropometric Measurement System calculates the user's body measurements to ensure a perfect fit for virtual clothing. Overall, these AI-based systems have great potential to enhance user experience and interaction in the field of HCI.

2. Egocentric Upper Limb Segmentation

One promising area of research in HCI is the development of egocentric vision-based approaches that enable users to control their virtual avatars using their body movements. Many applications involving the use of hands are based on hand segmentation, which is usually used as a pre-processing step in various contexts such as HCI, human-robot interaction, hand gesture recognition, and

Ital-IA 2023: 3rd National Conference on Artificial Intelligence, organized by CINI, May 29–31, 2023, Pisa, Italy

*Corresponding author.

gilda.manfredi@unibas.it (G. Manfredi); nicola.capece@unibas.it (N. Capece); ugo.erra@unibas.it (U. Erra); gruosso.monica@gmail.com (M. Gruosso)

0000-0003-0633-862X (G. Manfredi); 0000-0002-1544-3977 (N. Capece); 0000-0003-2942-7131 (U. Erra); 0000-0001-9609-8919 (M. Gruosso)

mixed reality. With the increasing popularity of wearable devices, there has been a growing interest in egocentric or first-person vision (FPV) systems for hand segmentation. However, most existing approaches focused only on the hand up to the wrist or bare arm. They were limited in their ability to deal with occlusions, varied lighting conditions, and dynamic camera and wearer movement. This study extended the hand segmentation task to focus on upper limb segmentation in egocentric vision and unconstrained real-world scenarios. We trained an encoder-decoder deep convolutional neural network using the DeepLabv3+ [4] architecture to overcome the limitations of the existing methods. The DeepLabv3+ encoder consists of a backbone network, followed by an atrous spatial pyramid pooling module (ASPP) [5] and a 1×1 convolutional layer. We selected atrous rates of 6, 12, and 18 for the ASPP atrous convolutions. We utilized convolutional and bilinear upsampling operations for the decoder to obtain spatial information from the encoder features and refine the segmentation result, resulting in detailed object boundaries.

Although various neural networks are available as the network backbone, we opted for the Xception model [6] due to its favourable qualitative and quantitative results for image classification tasks, surpassing prior networks like VGG-16, ResNet-152, and Inception V3 while still achieving a fast computation time. Our experimental testing revealed that it was the most effective model for our case study [7]. Specifically, we utilized the Xception-65 model adapted by Chen et al. [4] for semantic segmentation tasks. It comprises 65 layers and replaces the original max-pooling layers with atrous depthwise separable convolutions. These convolutions factorize a standard convolution into a depthwise convolution (spatial convolution carried out independently for each channel) with atrous convolution, followed by a pointwise (1×1) convolution. Additionally, batch normalization and ReLU were included after each 3×3 depthwise convolution.

Our dataset includes about $46,000$ varied RGB images with accurate labels, which enables our model to learn a wide range of realistic activities without any fine-tuning or domain adaptation. The RGB images used in this study were obtained from various sources and captured in unconstrained real-world scenarios, showing various situations such as different indoor and outdoor environments, lighting conditions, skin tones, hand-to-hand and hand-to-object occlusions, and a variable amount of motion blur. The images were well-annotated and collected from an egocentric perspective. The upper limb segmentation task dataset was compiled from three different sources: EDSH [8], TEgO [9], and EgoCam. EDSH dataset includes indoor and outdoor video frames showing different lighting conditions and a user's bare limb during real-life actions. TEgO is a large dataset of high-resolution indoor images showing two subjects' hands and forearms with different skin tones, lighting, and object occlusions. The EgoCam dataset, which we manually labelled, shows four male and female people in simple and cluttered environments, indoor and outdoor real-life scenes, and inter-hand occlusions. A subset of the first two datasets was used, and data with labelling errors were discarded. The images were cropped and resized to 360×360 to accelerate the training. The upper limb segmentation dataset was divided into training and test subsets. Additionally, we evaluated the network's generalization level on challenging cases using another test set named EgoGestureSeg. This subset was taken from a benchmark dataset for egocentric hand gesture recognition called EgoGesture and consisted of 235 images manually labelled by Gonzalez-Sosa et al. [10]. The images were captured in challenging indoor/outdoor scenarios and showed clothed and bare limbs, natural or artificial light, and various occlusions and motion blur.

We trained the network using our upper limb segmentation train set, and a training method similar to Chen et al. [11]. We utilized the stochastic gradient descent optimization algorithm with momentum set at 0.9, the base learning rate α_0 set to 0.0001, and a batch size of 8. We also used a cross-entropy loss function and a polynomial learning rate policy, which was more effective than other policies and resulted in faster convergence [12, 5]. This policy adjusts the learning rate α_t during training according to the following equation:

$$\alpha_t = \alpha_0 \times \left(1 - \frac{t}{T}\right)^p \tag{1}$$

Here, α_t represents the learning rate at the current iteration step t, T represents the total number of iterations (set at $90K$ for our training phase), and p represents the power value, set to 0.9.

During network training, we utilized pre-trained weights from the ImageNet [13] and MS-COCO [14] datasets and accelerated the process with one Nvidia Titan Xp GPU with 12GB memory. We employed Python 3.6 and the tensorflow [15] machine learning library, which was tested on Microsoft Windows 10 Pro. Finally, we applied data augmentation by randomly flipping images and labels left/right during training to prevent model overfitting. Our trained network achieved impressive results for both whole upper limb and hand-only segmentation tasks in egocentric view and unconstrained real-life scenarios, significantly outperforming the state-of-the-art (SOTA). We assessed our outcomes against the SOTA arm and hand segmentation techniques in egocentric vision, namely Ego2Hands [16] and EgoArm [10]. Moreover, we also evaluated HGR-Net [17], which presented promising results in demanding situations, although it was not particularly developed for egocentric vision segmentation. The results from HGR-Net were the worst (as shown in the fourth row of Figure 1), with a poor classifi-

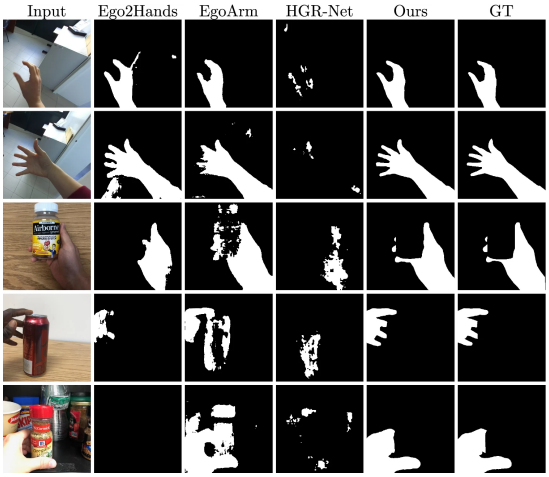

Figure 1: Visual examples of the Upper Limb Segmentation test set, including input images and corresponding ground-truth (GT) segmentation masks. The first two images are from EgoCam, and the last three are from TEgO.

cation of the limbs evident from the per-class metrics in Table 1. It is possible that HGR-Net was not specifically designed to segment limbs captured from an egocentric perspective, and a general-purpose approach may not be enough to produce optimal results. EgoArm performed well in identifying a large part of the limb but had difficulty with background pixels and tended to classify objects incorrectly as limbs. Ego2Hands also struggled with accurate segmentation (as seen in the last image of Figure 1). In contrast, our network performed exceptionally well in various scenarios, including different lighting conditions, skin tones, and occlusions caused by objects, as illustrated in the fifth row of Figure 1.

Our work is the first to evaluate and prove the effectiveness of a deep learning model for upper limb segmentation in such cases. It provides a promising direction for future research in egocentric vision-based HCI.

The study case was published in the Virtual Reality journal with the title *"Egocentric upper limb segmentation in unconstrained real-life scenarios"* [18].

3. XR Hand Gesture Recognition System

In recent years, there has been a growing interest in developing HCI systems that provide users with more intuitive and natural ways to interact with technology. Hand gesture recognition (HGR) has emerged as one of the most promising techniques for achieving this goal. With the latest HMDs, such as Oculus Quest 2 and Vive Focus 3, incorporating onboard hand-tracking sensors, HGR has become an increasingly popular way to enhance the user experience. However, these devices are not always accessible due to their high cost and usability issues, but new technologies have emerged that allow HGR through general-purpose low-cost devices. In this context, we propose a deep learning approach that enables HGR to use a simple, low-cost RGB camera. This approach promises to democratize access to HGR and enhance the user experience of a wide range of HCI systems. To create a system for HGR independent of camera features, we developed a pipeline that uses landmarks predicted by the MediaPipe Hands solution as input to a feed-forward neural network (FFNN). Our FFNN is trained on a dataset of 15 static and dynamic gestures and can predict corresponding hand gestures based on the input landmarks. These gestures involve a range of combinations of open and closed fingers and hands. The input for the model is represented by 21 hand landmarks obtained from MediaPipe, and each landmark is made up of x, y, and z coordinates. However, only the x and y coordinates were used for this model, and the z coordinate was ignored. The FFNN is characterized by a single hidden layer of 32 units that uses the pixel space of the hand landmark model as input. A rectified linear unit (ReLU) activation function was used on the hidden layer, while a Softmax [19] activation function was used on the last FFNN layer. Dynamic gestures were handled differently in this model; instead of using recurrent neural networks (RNNs) [20] and other data sequences-based approaches, a simple FFNN was used. This decision was made due to the real-time performance requirement of the dynamic gestures, as XR collaborative applications are well-suited for synchronous dynamic gestures. The proposed gestures were divided into two templates; static gestures and dynamic gestures. Static gestures were represented by a fixed hand pose that the FFNN directly predicts. Dynamic gestures were characterized by moving hands and can be further divided into single and combo gestures based on the number of tracked hands. A dataset of 130,000 hand pose samples was manually labelled and used to train the FFNN model. The FFNN was trained for 2000 epochs using the Adam optimizer [21] algorithm with a learning rate of 0.0001, resulting in a prediction accuracy of 98%. Testing was conducted in real-time using RGB sensors from the Intel RealSense D455 camera, the 40MP Huawei P30 back camera, and the 1080p MacBook Pro (M1 Pro) camera. The HGR system proposed in this study was designed for collaborative use in a multi-user XR 3D environment. Each gesture was mapped to a specific action in the XR 3D environment, and the approach was validated in a Unity 3D game engine environment. The Netcode mid-level networking library was utilized to enable multi-user scene authoring. The application was tested with two clients equipped with a simple RGB camera for hand landmark tracking and gesture prediction. Users could interact with

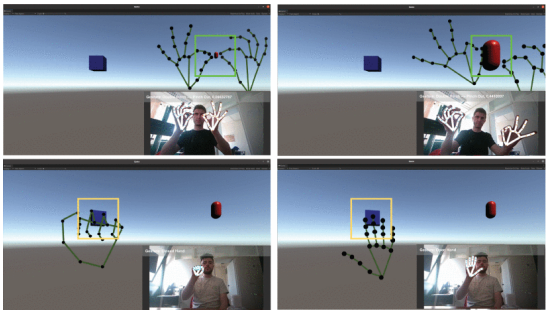

Figure 2: An example of our HGR system in use. This Figure shows how to move objects in a virtual scene using hand gestures and showing their tracking in AR. If necessary, it is possible to view and interact with the elements of the scene in AR.

visible gameObjects using the designed hand gestures, and a set of user interaction actions were implemented and associated with the hand gestures. The scene could be viewed using various devices and modes, including desktop mode, virtual, AR, and MR, using a smartphone, HMD, or Google Cardboard. The HGR system was also trained to recognize egocentric hand tracking. Figure 2 shows an example of the scene in desktop mode, with hand movement visualization for debugging purposes.

The proposed work was presented at the 2022 IEEE International Conference on Metrology for Extended Reality, Artificial Intelligence and Neural Engineering (MetroXRAINE) with the title *"An easy Hand Gesture Recognition System for XR-based collaborative purposes"* [22].

4. Virtual Dressing Room with Body Tracking

Virtual Dressing Rooms (VDRs) are an emerging technology that allows users to try on clothing virtually without physically wearing the clothes. The system uses computer vision and deep learning to track the user's body and simulate the clothing in real-time. This technology has the potential to revolutionize the retail industry, providing customers with a more personalized shopping experience and reducing the need for physical inventory. To improve the usability of VDRs, there have been efforts to incorporate HCI principles and anthropometric measurement systems. HCI aims to improve the interaction between humans and computers, making the virtual dressing room more intuitive and user-friendly. On the other hand, anthropometric measurement systems use body measurements better to simulate the fit of clothing on a specific individual. In this context, deep learning has been shown to be a promising technology for virtual dressing rooms, as it can learn from vast amounts of data and adapt to different body shapes and clothing styles. This approach can lead to more accurate simulations and a more realistic virtual shopping experience.

Our research aims to address the on-the-market solutions' limitations, such as the absence of dress animations, the presence of artefacts due to the incorrect tracking of body measurements, and clothes that do not adapt to the user's body. We developed a 3D virtual dressing room application called TryItOn using Unreal Engine 4.27 (UE4), known for its ability to create hyper-realistic environments that provide an immersive virtual reality experience. With TryItOn, users can try on digital garments and choose from various sizes. A single RGB-D camera system accurately captures the user's body measurements and tracks their movements in real-time. This information is utilized to create a 3D model of the user that is as realistic as possible. Additionally, a third-party plugin called uDraper[1] is used for modelling and simulating garment movement based on the specific physical characteristics of the fabric. Using deep learning in our system allows for accurate anthropometric measurements and tracking of the user's body movements. The pipeline for TryItOn consists of two types of operations: one-time and real-time. The former type of operation is executed solely during the modelling phase. This includes the creation of the base character before the release of the application, as well as the modelling of new garments during the creation and updating of the clothes catalogue. Meanwhile, real-time operations are executed during the application runtime. As previously stated, TryItOn is built on the UE4 platform. Hence, the models that were created during non-runtime operations are imported into the UE4 project, and real-time operations are executed within the game engine. In the group of one-time operations, a base 3D model character is created with realistic body measurements. This was created using the MB-Lab[2] add-on for Blender, which offers a character editor that can generate a realistic 3D rigged character model. This can be morphed through the Blender shape keys, with each corresponding to a specific anthropometric measurement parameter. Our methodology involves exporting the shape keys, also known as "morph targets", in an FBX file, alongside the mesh and skeleton, which can be imported into a UE4 project. To ensure proper sizing and alignment, we created a Python script that converts the Blender and UE4 coordinate systems and includes a collision mesh with lower LOD to optimize performance. The collision mesh includes relative morph targets, allowing the garment physics engine to interact with complex meshes while remaining hidden from the user's view. To ensure easy

[1] https://udraper.com/
[2] https://mb-lab-community.github.io/MB-Lab.github.io/

integration of the base avatar with existing plugins in UE4, we replaced the skeleton generated by MB-Lab with the standard mannequin skeleton included in UE4, which involved renaming and adjusting bone orientations and eliminating unnecessary bones. Finally, the modified skeleton was set to the same starting pose as the UE4 mannequin to maintain consistency within the virtual environment. The one-time operations group also includes garment modelling, which can be accomplished using the uDraper modelling software. This software enables the creation of a 3D garment by starting from a 2D pattern, which can also be designed using other software. After designing the 2D pattern, the different sections must be stitched together and wrapped around the 3D character to create the 3D garment.

The real-time operations group includes the anthropometric measurement calculation, the body tracking, and the physically-based garment simulation. To begin the calculation of a customer's anthropometric measurements, advanced deep learning techniques for computer vision tasks are employed using the FrankMocap framework [23, 24]. This involves analyzing 2D images of the customer captured through an RGB camera. FrankMocap employs deep learning models trained to reconstruct a human 3D mesh and a corresponding 3D skeleton, complete with body joints, in real-time. Anthropometric Measurement Calculation (AMC), a Python algorithm, has been developed to compute the anthropometric measurements, which are classified as linear or circular. AMC calculates linear measurements by directly measuring the distance between relevant joints on the skeleton. For circular measurements, AMC uses FrankMocap's body joints as landmarks to detect the points on the 3D mesh for accurate measurements. These measurements are used to morph the customer's 3D avatar within the VICO-DR application. Certain guidelines must be followed to ensure accurate measurements, such as being at the correct distance from the camera, wearing form-fitting clothing, and having proper lighting conditions. The system includes instructions, prompts, and real-time feedback to ensure precise anthropometric measurement calculation, delivering personalized and engaging virtual avatars for customers. As we needed the 3D character model to follow the user's movements, we developed a Body Tracking System Plugin (BTSP) for UE4. This plugin supports three types of cameras: the Azure Kinect camera, the ZED camera, and a simple RGB camera. We utilized their proprietary APKs for body tracking for the Azure Kinect and ZED cameras. However, for the simple RGB camera, we use the MediaPipe Pose estimation system. Both the camera APKs and MediaPipe utilize deep learning to analyze the images captured by the cameras and estimate the 3D position of the joints in real-time. The ZED module for body tracking focuses on detecting and tracking a person's bones, represented by two endpoints, also known as keypoints. The ZED camera can provide 2D and 3D information on each detected keypoint and local rotation between neighbouring bones. This information, including each person's 3D position and velocity, is shared in the outputs. To detect keypoints, the body tracking module also employs a neural network. It utilizes the depth and positional tracking of the ZED SDK module to obtain the final 3D position of each keypoint. The Azure Kinect body tracking system uses deep learning for detecting and tracking human bodies. The system begins by acquiring depth and infrared images through the Azure Kinect SDK. The infrared image is then passed through a convolutional neural network, which extracts the users' 2D joint coordinates and silhouette. Each pixel in the 2D image is assigned the corresponding depth value from the depth frame, which provides its position in 3D space. The results are then post-processed to produce accurate human body skeletons. The MediaPipe Pose estimation system uses a convolutional neural network to detect the joints in the image and estimate their 3D positions. The system is trained on large labelled image datasets to ensure accurate joint detection and tracking. Once the joints are detected, and their 3D positions are estimated, BTSP maps them to the UE4 mannequin skeleton to animate the avatar.

Regarding garment simulation, we used the physics simulation method of uDraper plugin to create realistic movement and deformation of virtual garments, allowing them to interact with the avatar's body and other objects in the scene. This approach is different from traditional keyframe animation techniques. The plugin calculates deformation based on internal and external forces and requires pre-modeling and pre-simulation of the virtual garment on the avatar's collision mesh. To address the limitation of pre-modeling, we developed a solution that involves implementing a base avatar that can be deformed at runtime, allowing garments to adapt to the avatar's shape during simulation.

Figure 3 shows the interface of the TryItOn application.

The proposed system is a work-in-progress project that was presented at the Extended Reality: First International Conference, XR Salento 2022, with the title *"TryItOn: A Virtual Dressing Room with Motion Tracking and Physically Based Garment Simulation"* [25].

References

[1] R. Gruetzemacher, J. Whittlestone, The transformative potential of artificial intelligence, Futures 135 (2022) 102884.

[2] N. Capece, F. Banterle, P. Cignoni, F. Ganovelli, R. Scopigno, U. Erra, Deepflash: Turning a flash selfie into a studio portrait, Signal Processing: Image Communication 77 (2019) 28–39.

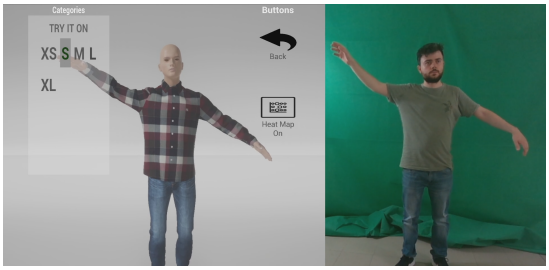

Figure 3: Interface of the TryItOn application.

[3] G. Caggianese, N. Capece, U. Erra, L. Gallo, M. Rinaldi, Freehand-steering locomotion techniques for immersive virtual environments: A comparative evaluation, International Journal of Human–Computer Interaction 36 (2020) 1734–1755.

[4] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, H. Adam, Encoder-decoder with atrous separable convolution for semantic image segmentation, in: Proceedings of the European conference on computer vision (ECCV), 2018, pp. 801–818.

[5] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, A. L. Yuille, Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs, IEEE transactions on pattern analysis and machine intelligence 40 (2017) 834–848.

[6] F. Chollet, Xception: Deep learning with depthwise separable convolutions, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 1251–1258.

[7] M. Gruosso, N. Capece, U. Erra, Exploring upper limb segmentation with deep learning for augmented virtuality (2021).

[8] C. Li, K. M. Kitani, Pixel-level hand detection in ego-centric videos, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2013, pp. 3570–3577.

[9] K. Lee, H. Kacorri, Hands holding clues for object recognition in teachable machines, in: Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, 2019, pp. 1–12.

[10] E. Gonzalez-Sosa, P. Perez, R. Tolosana, R. Kachach, A. Villegas, Enhanced self-perception in mixed reality: Egocentric arm segmentation and database with automatic labeling, IEEE Access 8 (2020) 146887–146900.

[11] L.-C. Chen, G. Papandreou, F. Schroff, H. Adam, Rethinking atrous convolution for semantic image segmentation, arXiv preprint arXiv:1706.05587 (2017).

[12] W. Liu, A. Rabinovich, A. C. Berg, Parsenet: Looking wider to see better, arXiv preprint arXiv:1506.04579 (2015).

[13] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al., Imagenet large scale visual recognition challenge, International journal of computer vision 115 (2015) 211–252.

[14] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C. L. Zitnick, Microsoft coco: Common objects in context, in: Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13, Springer, 2014, pp. 740–755.

[15] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, et al., Tensorflow: Large-scale machine learning on heterogeneous distributed systems, arXiv preprint arXiv:1603.04467 (2016).

[16] F. Lin, T. Martinez, Ego2hands: A dataset for egocentric two-hand segmentation and detection, arXiv preprint arXiv:2011.07252 (2020).

[17] A. Dadashzadeh, A. T. Targhi, M. Tahmasbi, M. Mirmehdi, Hgr-net: a fusion network for hand gesture segmentation and recognition, IET Computer Vision 13 (2019) 700–707.

[18] M. Gruosso, N. Capece, U. Erra, Egocentric upper limb segmentation in unconstrained real-life scenarios, Virtual Reality (2022) 1–13.

[19] C. M. Bishop, N. M. Nasrabadi, Pattern recognition and machine learning, volume 4, Springer, 2006.

[20] F. Yang, Y. Wu, S. Sakti, S. Nakamura, Make skeleton-based action recognition model smaller, faster and better, in: Proceedings of the ACM multimedia asia, 2019, pp. 1–6.

[21] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980 (2014).

[22] N. Capece, G. Manfredi, V. Macellaro, P. Carratù, An easy hand gesture recognition system for xr-based collaborative purposes, in: 2022 IEEE International Conference on Metrology for Extended Reality, Artificial Intelligence and Neural Engineering (MetroXRAINE), IEEE, 2022, pp. 121–126.

[23] Y. Rong, T. Shiratori, H. Joo, Frankmocap: A monocular 3d whole-body pose estimation system via regression and integration, in: IEEE International Conference on Computer Vision Workshops, 2021.

[24] H. Joo, N. Neverova, A. Vedaldi, Exemplar finetuning for 3d human pose fitting towards in-the-wild 3d human pose estimation, 3DV (2021).

[25] G. Manfredi, N. Capece, U. Erra, G. Gilio, V. Baldi, S. G. Di Domenico, Tryiton: A virtual dressing room with motion tracking and physically based garment simulation, in: Extended Reality: First International Conference, XR Salento 2022, Lecce, Italy, July 6–8, 2022, Proceedings, Part I, Springer, 2022, pp. 63–76.