Mobile Human Shape Superimposition: an initial approach using OpenPose

Roman Bajireanu, João A.R. Pereira, Ricardo J.M. Veiga, João D.P. Sardo,
Pedro J.S. Cardoso, Roberto Lam, João M.F. Rodrigues

LARSyS & ISE, University of the Algarve
Campus da Penha, Faro
PORTUGAL
{rbajireanu, japereira, rjveiga, jdsardo, pcardoso, rlam, jrodrig}@ualg.pt

Abstract: When it comes to visitors at museums and heritage attractions objects speak for themselves. Nevertheless, it is important to give those visitors the best experience possible as, this will lead to an increase in the visits number, enhance the perception and value of an organisation, and boost the sales. With the aim of enhancing a traditional museum visit, a mobile Augmented Reality (AR) framework is being developed as part of the Mobile Five Senses Augmented Reality (M5SAR) system for museums project. This paper presents an initial approach to human shape detection and AR content superimposition, achieved by combining information of human body joints with shape segmentation or texture overlapping. The OpenPose model was used to compute the body joints and the GrabCut algorithm for person segmentation, allowing to fit clothes segments to persons moving in real environments. The initial results and proof-of-concept are shown.

Keywords: Pose Estimation, Segmentation, Human Shape Superimposition, Convolutional Neural Networks

1 Introduction

This work is part of the Mobile Image Recognition based Augmented Reality (MIRAR) framework [18], one of the Mobile Five Senses Augmented Reality (M5SAR) project's modules [22]. The M5SAR project aims at developing an Augmented Reality (AR) system for museums that acts as guide for cultural, historical and museum's events, see e.g. [18, 22]. The use of mobile applications in museums, with or without AR systems, is not a novelty (see e.g., [11, 20, 26, 30]). The novelty proposed by the project is to extend the AR to the five human senses.

MIRAR module focuses on the development of a mobile multi-platform AR framework, with the following main goals: a) to detect museum's pieces (e.g., paintings and statues) [18], b) detect museum's walls/environments [31], and c) detect human shapes in order to overlay AR contents – Human Shape Superimposition submodule. This paper focuses on the last component, the detection of human shapes and the overlay of different clothes over those shapes. For background information see Sec. 2.

To achieve the objective of this paper the Convolution Neural Network (CNN) implemented in TensorFlow [8] and an image processing algorithm for foreground (person) segmentation are used. The main contribution of the paper is the overlapping of different types of clothes on persons that are in real environments using a mobile device.

Section 2 presents an overview on related works. Section 3 details the development of the human shape detection and superimposition model. Followed by Sec. 4, where conclusions and future work are drawn.

2 Background

The detection of human shapes in computer vision is a challenging problem due to several factors such as body parts occlusions, cluttered backgrounds, and/or, different viewpoints. Nevertheless, there are many developments and studies about human shapes detection, see e.g. [6, 15, 24, 28, 33].

Nowadays, human shape detection investigations have two types of approaches: top-down or bottom-up. While in the top-down method an estimation of the pose is made by first computing the body shape [9, 10, 16], in the bottom-up, the humans' parts are individually detected, generating groups of body parts in order to form a person instance [4, 6]. Some bottom-up approaches outperformed the top-down methods to which they were compared with [13, 19]. In this context, it is important to stress that, the runtime of top-down approaches is affected by the number of people in the image, i.e., more people means greater computational cost. In contrast, bottom-up approaches main-

tain efficiency even as the number of people in the image increases.

In recent years, the capacity of CNNs has been demonstrated in a large variety of computer vision tasks, such as object classification and detection, face and text recognition, etc. [1, 2, 21]. One of the areas at usage is human-pose estimation (in-the-wild), where the current state-of-the-art is achieved using CNNs [13, 19]. Recently, two popular CNN frameworks for human shape detection and segmentation have standout, the OpenPose [4, 25, 32] and the Mask R-CNN [9]. However, experimental tests showed that the original implementations are not suited for mobile devices. However, those CNN are the basis for the most recent implementations, that make possible the use on mobile devices. One approach that achieves full body pose estimation and segmentation is Mask R-CNN2Go [5], based on the original Mask R-CNN framework. The main reason for the processing time being reduced was the optimization of the number of convolution layers and the width of each layer.

Another approach to computing human pose estimation on mobile devices was the modification of the original architecture of OpenPose for mobiles [14, 27] and its combination with MobileNets [12]. MobileNets, as the name suggests, is designed for mobile applications, making use of depthwise separable convolutions layers, when compared to traditional layers. This modification reduces the processing time, but also reduces the accuracy in pose estimation, when compared to the original architecture.

Other methods have been applied with success to the human body segmentation subject. One of those methods is GrabCut [23], which defines the pixels of the image as connected regions of a graph and runs a graph cut based optimization to extract foreground pixels. For example, this methodology was applied in [10, 17].

It is also important to stress that, despite this subject being a hot topic of research nowadays, there are already applications available in the market, such as Pozus [7]. Pozus runs on the iOS mobile operating system and uses the smartphone camera as input for the human pose estimation applying textures over the human body. Nevertheless, Pozus does not do fitting of textures (clothes) to the person's shape.

3 Human Shape Superimposition

As already mentioned, the objective of the M5SAR's human superimposition submodule is to project AR content (clothes) over persons that are in a museum, using a mobile device. On other words, the goal is "to dress" museums' users with clothes from the epoch of the museums' objects. The human shape superimposition submodule as two main steps: (i) the human shape detection, and the (ii) clothes overlapping. Those steps will be explained in detail in the following sections.

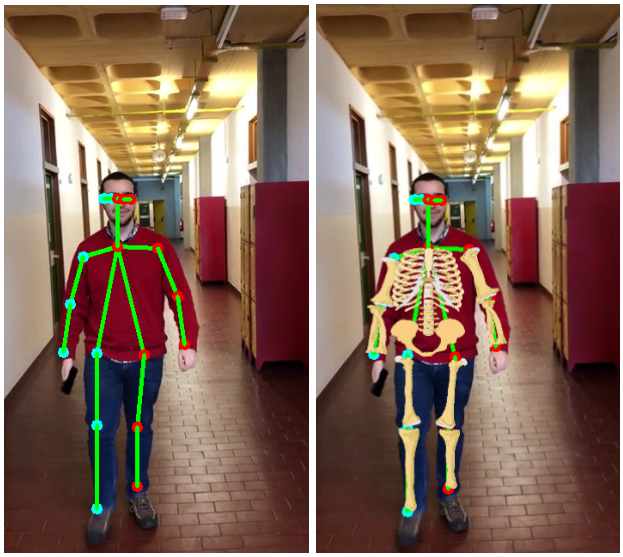

Figure 1: Example of pose estimation (left) and a skeleton superimposed over a person (right).

3.1 Human Shape Detection

In the M5SAR's context, the human shape detection, step (i), has to be accomplished in real-time on a mobile device, while the user is moving through six degrees of freedom (6DOF), which increases the computational complexity level [3, 17].

To achieve the detection, a variation of the OpenPose method (implemented with TensorFlow [8]) that works on the mobile devices [14, 27] was used. In our case, the features extracted with MobileNets [12] serve as input for the OpenPose algorithm. The OpenPose's output has 2D locations of keypoints with 57 layers of depth: 18 layers for body parts locations, 1 layer for the background, 19 layers for limbs information in the x direction and 19 layers for limbs information in the y direction (2D vector). Here, a body part is a component (joint) of the body, like a right knee, the right hip, or the left shoulder (see Fig. 1 left, the red and blue circles, where blue indicates the person's right body parts), and a limb is a pair of connected parts, like the right shoulder connection with the neck (see Fig. 1 left, the green line segments).

In more detail, the process can be summarized as follows. (a) Before entering the model, for each frame, the input color pixels are normalized to the interval [-1,1]. Then, the (b) MobileNets model is applied, returning the extracted features, which are used as input for (c) OpenPose. (d) OpenPose outputs 2 tensors: heatmap and part affinity fields (PAFs). (d.1) The heatmap gives the locations where particu-

lar joints are, which in reality correspond to 18 + 1 heatmaps (as already mentioned), i.e., one for each body part plus one for the background. (d.2) The PAF maps tell how to associate two joints. There are 38 PAF maps that represent the connections between two body joints, in a total of 17 connections, plus two connections between the ears and respective shoulders (left ear paired with left shoulder and right ear with right shoulder).

With the above information computed, it is now possible to overlap images on the human body. One example of this can be seen in Fig. 1 right, where a skeleton is overlapped over a person (see more details how the overlapping is done in Sec. 3.2).

The mobile application was implemented in Unity [29] using the OpenCV library (Asset for Unity). Furthermore, the frozen pre-trained weights for the OpenPose given in [14] were used as input parameters.

In order to verify the implementation's reliability and to compare running times of "mobile" OpenPose, tests were done in desktop and mobile platforms; nevertheless, the video used for the tests was acquired by the mobile device.

Tests were done using an ASUS Zenpad 3S 10" tablet and a Windows 10 desktop with an Intel i7-6700 running at 3.40 GHz. The tested video consisted of a total amount of 573 frames of expected user navigation (vertical orientation used).

Two frame sizes for the CNN were also used in the tests (in MobileNets), namely, 368×368 and 184×184 pixels (px). Depending on the input's size, each frame takes a mean value of 236ms (milliseconds) and 70ms to be processed on the desktop, respectively, while in the mobile device it takes a mean time of 2031ms and 599ms, respectively.

For each heatmap, a confidence threshold of 30% was used as minimal good accuracy for pose estimation. Further tests showed that, as expected, increasing the threshold decreases the number of frames with body parts matched, while the processing times continues very similar .

Reducing the size of the CNN from 368×368px to 184×184px also, as expected, allowed to attain improvements on the time performance, but the accuracy of the results dropped. One example of missing accuracy is the confusion between right and left hands/legs, as seen in Fig. 2 top row left. The same figure, bottom row, shows one example of pose estimation returning different body parts and limbs when applying the two spatial sizes, 368×368px (left) and 184×184px (right). In this instance, it is possible to observe that body joints and limbs were lost in the 184×184px case.

Both error types (confusion between right and left hands/legs and missing body parts) are expected to be solved using historical data, from the body parts. In other words, it is intended to project one human shape overlapping per second, so for each n consecutive frames (the value of n changes from device to device; for the example presented with 184×184px, it was used $n = 14$ for the desktop, and $n = 2$ for the mobile) the center of mass (centroid) of each body part is located, and the limbs are projected from there. Nevertheless, at the moment of writing, this is ongoing work.

Figure 2: Top row, example of confusion between left and right ankle (left) and a well detected pose (right). Bottom row, example of pose estimation with spatial size of the CNN equal to 368×368px (left) and 184×184px (right).

3.2 Clothes Overlapping with Segments

This sub-section explains one of the method used to fit the clothes into the persons. The algorithm was di-

Figure 3: examples of the clothes segments.

vided into 4 main components: (a) split clothes into segments, (b) for each limb (or group of limbs) place the clothes segment, (c) segment the person, and (d) (re)fit clothes segment to the correspondent person segment.

In Step (a), the clothes are divided into several segments depending on their type and shape. For example, a suit is divided into 9 segments, while a dress is divided into only 2 segments, as shown in Fig. 3. Currently, these segments are manually computed. Each clothes' segment is associated with two or more OpenPose's body parts. For instance, in the suitcase (Fig. 3 left), segment number 9 uses both shoulders and hips, while segment 5 uses the right shoulder and right elbow. In the case of the sleeveless dress (Fig. 3 right), the projection of segment 1 also uses the shoulders and the hips, while segment 2 uses the hips combined with the ankles.

Regarding Step (b), in order to properly project contents over the person's body, it is necessary to calculate the angle (α) of each limb relative to a vertical alignment (see Fig. 4 top left), and rotate the respective clothes segment (see the resulting segment in Fig. 5 top row).

To fit the clothes to each body part, it is necessary to segment the person, Step (c). For this purpose the GrabCut segmentation algorithm [10] was used. The grabcut algorithm is a semi-automatic procedure because it receives the foreground and background areas as input. To create a fully automatic algorithm, the body parts output by OpenPose was used. By using the bounding coordinates from the body parts, it is possible to create a bounding box area around the person, see Fig. 5 middle row left. In more detail: (c.1) use the coordinates from neck, hands and ankles to create a bounding box area around the person. (c.2) Increase the bounding box area by 10%; this will put a bounding box around the human body and is used as the foreground in the GrabCut algorithm. (c.3) Cut the input image, with double the size of the initial bounding box (up to the image limits), with the same center, and use the cropped area as the background; this to optimize the GrabCut processing time. Finally, (c.4) use the GrabCut algorithm to do the segmentation. The resulting segmentation can be seen in Fig. 5 middle row right image.

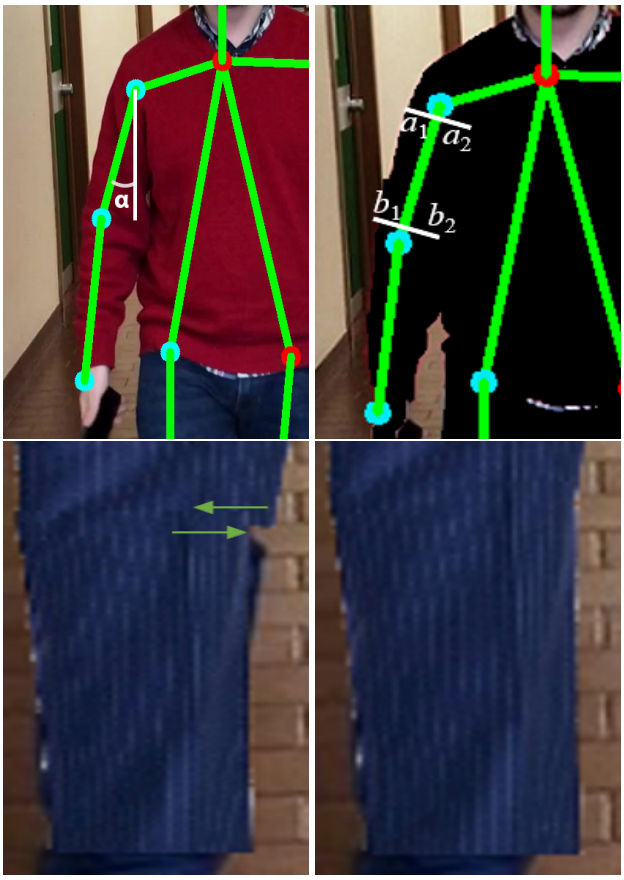

Figure 4: Top row, limb's angle (left), and the distances calculated to fit the clothes (right). Bottom row, segment stretch directions (left), and final result (right) (see Sec. 3.2).

In the last Step (d), each segment is computed and fit to the person's body as follows. (d.1) For each limb (the ones that match with the clothes segment), a line is calculated between the parts that create the limb, then the upper and lower distances, in a perpendicular, are computed between this line and the segmented area previously determined in (c), see Fig. 4 top row right, $a1$ and $b1$. (d.2) This distance is increased by 10% (this value was empirically chosen). (d.3) The same values are projected into the opposite perpendicular direction of the limb, $a2$ and $b2$. Now, in possession of the coordinates where the segments should be placed, (d.4) a warp perspective of the segment is done.

Finally, (d.5) segments that are nearby, in a way they have continuity in their contours, are adjusted. For instance, if after the above process (d.1-4) seg-

ments numbered 3 and 4 (Fig. 3 left) still do not have contour continuity, one is stretched out and the other is stretched in until the coordinates $a1$ and $b1$ of one segment exactly match the coordinates $a2$ and $b2$ of the other segment. Figure 4 bottom row left shows the directions of the stretch for this specific case and on the right the final result.

Figure 5 bottom row shows the final results obtained for the suit and for the dress when overlapped in the same person. Figure 6 shows a sequence of 3 frames of another person moving in a different environment using the same suit and dress.

The clothes overlapping process takes an average processing time of 127ms (per frame) using the desktop and 1086ms on the mobile device. The complete process takes 70ms + 127ms equals 197ms in the desktop and 559ms + 1086ms equals 1645ms in the mobile for each frame. This means that, we are still far from the intended results, i.e., to do at least one clothes overlap per second. Nevertheless, this is the initial proof-of-concept and optimizations are required.

3.3 Clothes Overlapping with Textures

A different approach was also tested to overlap clothes with textures. The algorithm was divided into 3 main components: (a) add an skeleton to the textures, (b) resize textures, (c) project textures over the person.

In step (a), a modelling tool to add a skeleton (Fig. 8) to the textures that allows deformations is used. In the example it was done a set-up of *bones*, parameters related to geometry and weights of texture. The suit has 14 *bones* and the dress has 10 (the position of *bones* are made based on the results of OpenPose). For the geometric transformations, vertices and edges were created automatically at the borders of the textures. The geometric skeleton that will be generated was refined adjusting the following parameters: a density and accuracy of the skeleton to the texture outline, and a threshold for transparency considered when generating the outline, and for last subdivide the skeleton to increase tessellation. The weights of textures are automatically generated based on the texture skeleton.

Regarding step (b), the textures are resized taking into consideration the distance between ankles and nose (to obtain an approximation to the person's height).

For step (c), the textures are placed over the detected poses and it is necessary to calculate the angle (α) of each limp relative to a vertical alignment and rotate the respective textures *bones*. For example, each *bone* from texture skeleton is placed over the respective body parts (Fig. 8) that will deform the textures over the persons body. The dress has less *bones* (elbows and wrists are excluded) than the suit because it does not have arms.

Figure 5: Top row, example clothes overlapping. Middle row, example of a bounding box (left) and segmentation (right). Bottom row, projection of contents on person body after clothes fitting.

Figure 6: Sequence of frames of a human shape superimposition using "segmentation".

Figure 7: Sequence of frames of a human shape superimposition using "textures".

Figure 9 shows the final results obtained for the suit and for the dress when overlapped in the same person. Figure 7 shows a sequence of 3 frames of another person moving in a different environment using the same suit and dress.

The textures overlapping process takes an average processing time of 6ms using the desktop and 29.31ms on the mobile device. The complete process takes 70ms + 6ms equals 76ms in the desktop and 559ms + 29.31ms equals 588.31ms in the mobile.

Figure 8: example of created *bones*.

4 Conclusion

This paper presented an initial proof-of-concept for human shape superimposition. The first proposed method combines OpenPose and the segmentation done with the GrabCut algorithm to fit textures to the human body in mobile devices. Despite already working in the mobile device, optimizations to achieve performances of at least one shape superimposition per second are still needed. The second method presents the superimposition using textures. As the latter approach presents better results, this is the one that we are expecting to use.

For future work, beyond the mentioned optimizations and the use of "historical" data, the GrabCut segmentation process, needs to be complemented in order to achieve a better human segmentation, since this will allow the projection of contents onto those shapes/persons with better quality. Finally, 3D clothes models will be used, and fit those to the segmentation results from GrabCut, instead of the clothes segments.

Acknowledgements: This work was supported by the Portuguese Foundation for Science and Technology (FCT), project LARSyS (UID/EEA/50009/2013),

Figure 9: example of human shape superimpostion using “textures”.

CIAC, and project M5SAR I&DT nr. 3322 financed by CRESC ALGARVE2020, PORTUGAL2020 and FEDER. We also thank Faro Municipal Museum and the M5SAR project leader, SPIC - Creative Solutions [www.spic.pt].

References:

[1] Brandon Amos, Bartosz Ludwiczuk, and Mahadev Satyanarayanan. Openface: A general-purpose face recognition library with mobile applications. *CMU School of Computer Science*, 2016.

[2] Batuhan Balci, Dan Saadati, and Dan Shiferaw. Handwritten text recognition using deep learning. *CS231n: Convolutional Neural Networks for Visual Recognition, Stanford University, Course Project Report, Spring*, 2017.

[3] Chetan Bhole and Christopher Pal. Automated person segmentation in videos. In *21st International Conference on Pattern Recognition (ICPR)*, pages 3672–3675. IEEE, 2012.

[4] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7291–7299, 2017.

[5] Amit Jindal et. all. Enabling full body AR with Mask R-CNN2Go. `https://bit.ly/2jfnn8S`, 2018. Retrieved: Apr. 10, 2018.

[6] Haoshu Fang, Shuqin Xie, Yu-Wing Tai, and Cewu Lu. Rmpe: Regional multi-person pose estimation. In *The IEEE International Conference on Computer Vision (ICCV)*, volume 2, 2017.

[7] GENTLEMINDS. Pozus. `http://pozus.io`, 2018. Retrieved: Apr. 10, 2018.

[8] Google. TensorFlow - an open-source machine learning framework for everyone. `https://www.tensorflow.org/`, 2018. Retrived: January 14, 2018.

[9] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 2980–2988. IEEE, 2017.

[10] Antonio Hernández-Vela, Miguel Reyes, Víctor Ponce, and Sergio Escalera. Grabcut-based human segmentation in video sequences. *Sensors*, 12(11):15376–15393, 2012.

[11] HMS. Srbija 1914 / augmented reality exhibition at historical museum of Serbia, Belgrade. `https://vimeo.com/126699550`, 2017. Retrieved: April 04, 2018.

[12] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[13] Eldar Insafutdinov, Mykhaylo Andriluka, Leonid Pishchulin, Siyu Tang, Evgeny Levinkov, Bjoern Andres, and Bernt Schiele. Arttrack: Articulated multi-person tracking in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 4327, 2017.

[14] Ildoo Kim. tf-pose-estimation. `https://bit.ly/2HJxxcq`, 2018. Retrieved: Apr. 10, 2018.

[15] Wanli Ouyang and Xiaogang Wang. Joint deep learning for pedestrian detection. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 2056–2063. IEEE, 2013.

[16] George Papandreou, Tyler Zhu, Nori Kanazawa, Alexander Toshev, Jonathan Tompson, Chris Bregler, and Kevin Murphy. Towards accurate multiperson pose estimation in the wild. *arXiv preprint arXiv:1701.01779*, 8, 2017.

[17] Sohee Park and Jang-Hee Yoo. Human segmentation based on grabcut in real-time video sequences. In *IEEE International Conference on*

Consumer Electronics (ICCE), pages 111–112. IEEE, 2014.

[18] João A.R. Pereira, Ricardo J.M. Veiga, Marco A.G. Freitas, J.D.P. Sardo, Pedro J.S. Cardoso, and João M.F. Rodrigues. MIRAR: Mobile image recognition based augmented reality framework. In *International Congress on Engineering and Sustainability in the XXI Century*, pages 321–337. Springer, 2017.

[19] Leonid Pishchulin, Eldar Insafutdinov, Siyu Tang, Bjoern Andres, Mykhaylo Andriluka, Peter V Gehler, and Bernt Schiele. Deepcut: Joint subset partition and labeling for multi person pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4929–4937, 2016.

[20] Qualcomm. Invisible museum. `https://goo.gl/aS0NKh`, 2017. Retrieved: April 04, 2018.

[21] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 91–99. Curran Associates, Inc., 2015.

[22] João MF Rodrigues, João AR Pereira, João DP Sardo, Marco AG de Freitas, Pedro JS Cardoso, Miguel Gomes, and Paulo Bica. Adaptive card design UI implementation for an augmented reality museum application. In *International Conference on Universal Access in Human-Computer Interaction*, pages 433–443. Springer, 2017.

[23] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. In *ACM transactions on graphics (TOG)*, volume 23(3), pages 309–314. ACM, 2004.

[24] Pierre Sermanet, Koray Kavukcuoglu, Soumith Chintala, and Yann LeCun. Pedestrian detection with unsupervised multi-stage feature learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3626–3633. IEEE, 2013.

[25] Tomas Simon, Hanbyul Joo, Iain Matthews, and Yaser Sheikh. Hand keypoint detection in single images using multiview bootstrapping. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, 2017.

[26] SM. Science museum - atmosphere gallery. `https://vimeo.com/20789653`, 2017. Retrieved: April 04, 2018.

[27] Ale Solano. Human pose estimation using openpose with tensorflow. `https://goo.gl/7t7SGS`, 2018. Retrieved: Apr. 10, 2018.

[28] Yonglong Tian, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning strong parts for pedestrian detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1904–1912, 2015.

[29] Unity. Unity3D. `https://unity3d.com`, 2018. Retrieved: Jan. 10, 2018.

[30] Natalia Vainstein, Tsvi Kuflik, and Joel Lanir. Towards using mobile, head-worn displays in cultural heritage: User requirements and a research agenda. In *Proc. 21st International Conference on Intelligent User Interfaces*, pages 327–331. ACM, 2016.

[31] R.J.M. Veiga, R. Bajireanu, J.A.R. Pereira, J.D.P. Sardo, P. J.S. Cardoso, and João M.F. Rodrigues. Indoor environment and human shape detection for augmented reality: an initial study. *In Procs23rd edition of the Portuguese Conference on Pattern Recognition, Aveiro, Portugal, 28 Oct., pp. 21.*, 2017.

[32] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. In *CVPR*, 2016.

[33] Liliang Zhang, Liang Lin, Xiaodan Liang, and Kaiming He. Is faster R-CNN doing well for pedestrian detection? In *European Conference on Computer Vision*, pages 443–457. Springer, 2016.