Human Body Pose Estimation in Multi-View Environments

Jorge L. Charco, Angel D. Sappa, Boris X. Vintimilla and Henry O. Velesaca

Abstract This chapter tackles the challenging problem of human pose estimation in multi-view environments to handle scenes with self-occlusions. The proposed approach starts by first estimating the camera pose—extrinsic parameters—in multi-view scenarios; due to few real image datasets, different virtual scenes are generated by using a special simulator, for training and testing the proposed convolutional neural network based approaches. Then, these extrinsic parameters are used to establish the relation between different cameras into the multi-view scheme, which captures the pose of the person from different points of view at the same time. The proposed multi-view scheme allows to robustly estimate human body joints' position even in situations where joints are occluded. The chapter concludes by presenting experimental results in real scenes by using state-of-the-art and the proposed multi-view approaches.

Jorge L.Charco
Escuela Superior Politécnica del Litoral, ESPOL, Campus Gustavo Galindo Km. 30.5 Vía Perimetral, P.O. Box 09-01-5863, e-mail: jlcharco@espol.edu.ec
Universidad de Guayaquil, Delta and Kennedy Av., Guayaquil, Ecuador, e-mail: jorge.charcoa@ug.edu.ec

Angel D. Sappa
Escuela Superior Politécnica del Litoral, ESPOL, Campus Gustavo Galindo Km. 30.5 Vía Perimetral, P.O. Box 09-01-5863, e-mail: asappa@espol.edu.ec
Computer Vision Center, Campus UAB, 08193 Bellaterra, Barcelona, Spain, e-mail: asappa@cvc.uab.es

Boris X. Vintimilla
Escuela Superior Politécnica del Litoral, ESPOL, Campus Gustavo Galindo Km. 30.5 Vía Perimetral, P.O. Box 09-01-5863, e-mail: boris.vintimilla@espol.edu.ec

Henry O. Velesaca
Escuela Superior Politécnica del Litoral, ESPOL, Campus Gustavo Galindo Km. 30.5 Vía Perimetral, P.O. Box 09-01-5863, e-mail: hvelesac@espol.edu.ec

1 Introduction

During the last decade, several works have focused on the Human Pose Estimation (HPE) problem, which is usually tackled by detecting human body joints such as wrist, head, elbow, etc. to build the human body skeleton by the connection of all detected joints. The solutions proposed in the state-of-art are robust when all body joints are visible by the detector. However, when joints are occluded, due to moving objects in the scene (i.e., bicycles, cars) or natural human body pose self-occlusions, it becomes a challenging problem, particularly, in monocular vision system scenarios. An important aspect to take into account is that other areas of research take advantage of the accuracy of human pose estimation to develop on totp of it different solutions, for instance: human action recognition, gaming, surveillance, just to mention a few. Nowadays, most of computer vision tasks (e.g., segmentation, camera pose, object detection), including the human pose estimation problem, are tackled by convolutional neural networks (CNN), reaching a better performance with respect to classical approaches (e.g., [30, 6, 29, 22, 10, 2, 26]).

CNN architectures have been used in monocular vision system scenarios to solve the human pose estimation problem, using as an input a set of images with single or multiple-persons from one camera to feed the architecture. Regarding this latter point, multiple-persons, the computational cost could be increased due to the number of body joints of each subject that the architecture has to detect in the image. Although the obtained results are appealing, the complex poses are a challenging problem since certain parts of the body joints could be occluded, including when other moving objects are part of the scene. This problem could be overcome by the multi-view approaches since the human body can be captured from different points of view at the same time. This could allow to recover body joints occluded in one view by using information from other cameras, with other point of view, where these body joints are not occluded.

Some tasks such as camera pose, 3D-reconstruction, object detection (e.g., [31, 25, 5, 27], just to mention a few), where the principal problem is the occluded regions, have been tackled by multi-view approaches. However, few works have been proposed to tackle the human pose estimation problem by using the approach mentioned above. Some works propose to fuse features from both views (i.e., two cameras located in different points of view) to predict the human pose (e.g., [23, 13]), through either the epipolar line of the images across all different views or using the intermediate layers in early stages of the architecture to find corresponding points in other views.

On the contrary to previous works, a compact architecture, which has been proposed for monocular scenarios [16], is adapted in the current chapter to leverage the relative camera pose in multi-view scenarios, and thus to find the relationship between the different features of the images, which are acquired at the same time from different views, to tackle the self-occlusion problems of the body joints. This chapter is organized as follows. Section 2 summarizes deep learning based camera pose estimation used to obtain the relative rotation and translation between two cameras, which includes experimental results on real-world datasets and transfer learning from virtual environments to real-world. Then, Section 3 describes the architecture used to estimate the human body pose in multi-view scenarios, including experimental results and comparisons with other approaches of the state-of-art. Finally, conclusions and future works are provided in Section 4.

2 Camera Pose Estimation

Most computer vision tasks require a calibration process to get camera intrinsic and extrinsic parameters, which are used to have correspondences between the camera and the world reference system. In general, these calibration processes are performed by using special calibration patterns. However, during the last decade, different proposals have been developed to estimate these parameters in an automatic way, more specifically the extrinsic camera parameters (relative translation and rotation), by using just the context of the images without considering special calibration patterns.

Some difficulties are presented during the process of calibration such as illumination, low resolution, few features images, becoming a challenging problem to solve. For doing it, features detection is a key point to consider since they are used in any camera calibration process. Different proposals have been used for feature point detection, for instance SURF [1], ORB [24], SIFT [20] just to mention a few. However, the accuracy of these algorithms decrements when there are not enough features points to be matched. In order to overcome this problem, different CNN architectures have been used in this process due to the power to extract features on images, showing better results than classical approaches. Nowadays, these methods are used for camera calibration, which could be divided into two categories: single or multi-view environments.

In the single view approaches (e.g. [17, 18]), a sequence of images are used, which are captured from the same angle and point of view. The main limitation of these approaches is that some important features could be occluded since depend on the angle and position of the camera in the world coordinate system, becoming a challenging problem to solve. In the second category, i.e., multi-view approaches, the occluded feature problem could be solved since the scene is captured from different points of view at the same time. However, it is necessary to ensure the overlap between the acquired images. A few works have been proposed considering multi-view scenarios; it could be mentioned the approaches proposed by [19, 6, 9], where a set of pairs of images is used to feed a Siamese CNN architecture for relative camera pose estimation. In this chapter, the multi-view environment problem is considered, where a Siamese CNN architecture is used to estimate the relative camera pose. This architecture and its main features are presented in Section 2.1. Then, experimental results from this Siamese architecture, on real world scenarios, are presented in Section 2.2. Finally, a transfer learning strategy, which takes advantage of virtual scenarios to initialize the network weights, is introduced in Section 2.3.

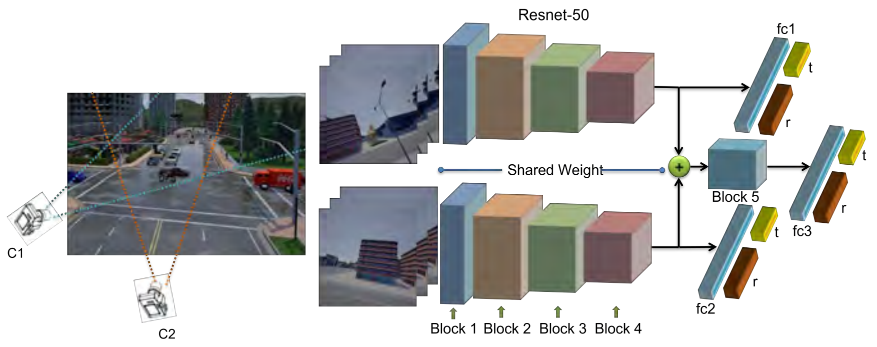

Fig. 1 Siamese CNN architecture fed with pairs of images of the same scene captured from different points of views at the same time. The extrinsic camera parameter estimation is obtained by using the regression part, which contains three fully-connected layers.

2.1 Siamese Network Architecture

This section summarizes the Siamese CNN architecture, referred to as RelPoseTL, that has already been presented in our previous work [4]. It is used to estimate the relative pose between two synchronized cameras (Fig. 1 shows an illustration of this architecture). The RelPoseTL is based on a modified Resnet-50, proposed in [12], which has two identical branches with shared weights up to the fourth residual block. The output of each branch is concatenated to feed the fifth residual block. ELUs activation functions are used instead of RELUs in the residual block structures, since it helps to speed up convergence and to avoid the vanishing gradient, as it was mentioned in [7]. A pair of images are used to feed the architecture. These images are acquired at the same time, but from different point of views, i.e., different positions and orientations with respect to the real-world system. Three fully connected (*fc*) layers are added after to the fourth and fifth residual block. The first two fully connected layers *fc1* and *fc2* have a dimension of 1024 each one and are added after the fourth residual block to predict the **global camera pose** followed by two regressors of (3×1) and (4×1), which correspond to the global translation and rotation concerning the real-world system respectively. The last fully-connected layer *fc3* is added to the fifth residual block. Similarly to mentioned above, this fully connected layer has a dimension of 1024, but it is used to predict the **relative camera pose**, followed by two regressors of (3×1) and (4×1) to predict the relative translation and rotation between the given pair of images.

Global camera pose is represented by: $\Delta p = [\hat{t}, \hat{r}]$, where $\hat{t}$ represents the translation as a 3-dimension vector and $\hat{r}$ represents the quaternion values of the rotation as a 4-dimension vector. The Euclidean distance is used to estimate them:

$$T_{Global}(I) = \left\| t - \widehat{t} \right\|_{\gamma}, \tag{1}$$

$$R_{Global}(I) = \left\| r - \frac{\widehat{r}}{\|\widehat{r}\|} \right\|_{\gamma}, \tag{2}$$

where t and $\widehat{t}$ represent the ground truth and prediction of the translation respectively, and r is the ground truth rotation represented as quaternion values, and $\widehat{r}$ denotes its prediction. The predicted rotation is normalized to a unit length as $\frac{\widehat{r}}{\|\widehat{r}\|}$. $L2$ Euclidean norm is defined as γ. Due to the difference in scale between both terms (translation and rotation), the authors in [17] propose to use two learnable parameters called $\widehat{s}_x$ and $\widehat{s}_y$ to balance translation and rotation terms. The effect of these parameters are similar to the one proposed in [18], where a β parameter is used to balance both terms. In RelPoseTL, a modified loss function that uses $\widehat{s}_y$ as learnable parameter is used:

$$Loss_{Global}(I) = T_{Global} + (exp(\widehat{s}_y) * R_{Global} + \widehat{s}_y). \tag{3}$$

On the other hand, the relative pose is estimated from the output of the fifth residual block. The relative translation and rotation are obtained similarly to the global pose estimation, which is defined as:

$$T_{Relative}(I) = \left\| t_{rel} - \widehat{t}_{rel} \right\|_{\gamma}, \tag{4}$$

$$R_{Relative}(I) = \left\| r_{rel} - \frac{\widehat{r}_{rel}}{\|\widehat{r}_{rel}\|} \right\|_{\gamma}, \tag{5}$$

where $T_{Relative}$ and $R_{Relative}$ are the differences between the ground truth (t_{rel} and r_{rel}) and the prediction from the trained model ($\widehat{t}_{rel}$ and $\widehat{r}_{rel}$). As the rotation ($\widehat{r}_{rel}$) is predicted directly from the trained model, then it has to be normalized before. In order to get t_{rel} and r_{rel}, the following equations are used:

$$t_{rel} = t_{C1} - t_{C2}, \tag{6}$$

$$r_{rel} = r^{*}_{C2} * r_{C1}, \tag{7}$$

where Ci corresponds to the pose parameters of the (i) camera (i.e., rotation and translation); these parameters are referred to the real world system; r^{*}_{C2} is the conjugate quaternion of r_{C2}. Similarly to the Eq. (3), the loss function used to obtain the relative pose is defined as:

$$Loss_{Relative}(I) = T_{Rel} + (exp(\widehat{s}_y) * R_{Rel} + \widehat{s}_y). \tag{8}$$

Note that $Loss_{Global}$ in Eq. (3) and $Loss_{Relative}$ in Eq. (8) are applied for different purposes. The first one is used to predict global pose through each branch of the trained model, where each branch is fed by images captured at the same scenario from different points of view. While the second one predicts the relative pose between pairs of images by concatenating the Siamese Network (see Fig. 1).

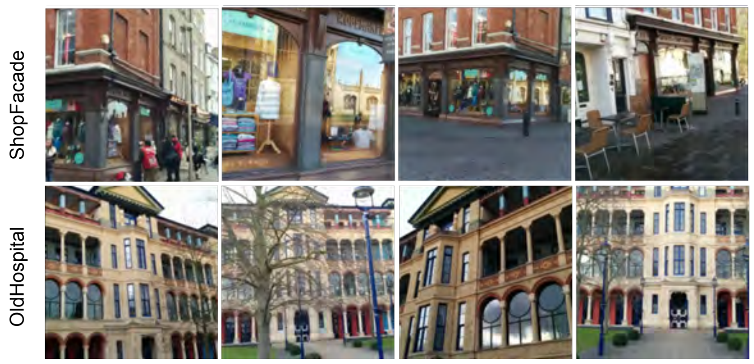

Fig. 2 Real world images of Cambridge Landmarks dataset [18] used for training and evaluating the RelPoseTL architecture.

Table 1 Comparison of average errors (extrinsic parameters) between RelPoseTL and Pose-MV on ShopFacade and OldHospital of Cambridge Landmarks dataset.

Scene / Models	Pose-MV [6]	RelPoseTL[4]
ShopFacade	1.126m, 6.021º	**1,002m, 3.655º**
OldHospital	5.849m, 7.546º	**3.792m, 2.721º**
Average	3.487m, 6.783º	**2.397m, 3.188º**

The RelPoseTL was jointly trained with Global and Relative Loss, as shown in Eq. (9):

$$L = Loss_{Global} + Loss_{Relative}. \tag{9}$$

2.2 Results from Real World Datasets

The RelPoseTL architecture presented in the previous section has been trained and evaluated using the Cambridge Landmarks dataset [18]; Figure 2 shows some images of this dataset, which were captured in outdoor environments. All images are resized to 224 pixels along the shorter side; then, the mean intensity value is computed and subtracted from the images. For the training process, the images are randomly cropped at 224×224 pixels. On the contrary to the training stage, during the evaluation process, a central crop is used instead of a random crop.

For the training process, the weights of Resnet-50 pretrained on ImageNet were used to initialize layers of RelPoseTL up to the fourth residual block, for the remaining layers the normal distribution was used. The RelPoseTL was trained on ShopFacade and OldHospital of Cambridge Landmarks dataset. A set of 5900 pairs of images of OldHospital dataset was used for the training process. A similar process was performed but with 1300 pairs of images of ShopFacade dataset. The architecture was trained during 500 epoch for both datasets, which approximately took 7 hours and 3 hours respectively. For the evaluation process, a set of 2100 pairs of images from the OldHospital dataset and a set of 250 pairs of images from ShopFacade dataset have been considered.

In order to evaluate the performance of RelPoseTL, angular error and Euclidean distance error are considered. The first is used to compute the rotation error; it is computed with a 4-dimensional vector. While, the second one is used to computed the distance error between ground truth and the estimated value using a 3-dimensional vector. Table 1 depicts average errors on rotation and translation for both datasets, obtained with the RelPoseTL network and with the Pose-MV network [6]. It can be appreciated that the translation error obtained by the RelPoseTL improves the results of Pose-MV in about 32%, and about 53% when the rotation error is compared between the RelPoseTL and Pose-MV.

2.3 From Virtual Environments to Real World

As an extension to the results obtained in the previous section, where the RelPoseTL architecture has been trained with real data, in this section a novel training strategy is proposed. It consists on first training the network with synthetic images obtained from a virtual environment and then, use this weights as an initialization when training again the network but with images from real scenarios. This strategy is intended to improve results from real data since the training of the network does not start from scratch but from a pre-trained set. An advantage of using virtual environments is the possibility of generating an almost unlimited set of synthetic images considering different conditions, actors and scenarios, i.e., weather, illumination, pedestrian, road, building, vehicles. Furthermore, an additional advantage lies on the fact that the ground truth is automatically obtained, reducing human error when the datasets are manually annotated. Different engines could be used to design virtual environments and aquire such a kind of synthetic pairs of images (e.g., CARLA Simulator [8], Virtual KITTI [11], Video Game engines, just to mention a few). In the current work different synthetic datasets have been generated using the CARLA simulator, an open-source software tool [8]. Among the tools offered by CARLA simulator, there is an editor that allows you to modify existing virtual worlds, as well as to create new scenarios from scratch; this editor integrates both CARLA simulator and Unreal Engine, which is a video game engine that CARLA is based on.

It should be mentioned that synthetic images generated from virtual environments and the real images used for final training could have different features spaces and distribution, hence a Domain Adaptation (DA) strategy should be used to transfer the knowledge from one domain to other. Depending on the relationship between both domains, transferring the knowledge learned from virtual to real environments can be performed in one-step or multi-step DA. For the first case, both domains are directly related since the features spaces are similar. In the second case, a intermediate domain, which should be highly related with both domains, is necessary. In this section, the strategy proposed in [4] will be followed. It consist on creating 3D virtual scenarios with a similar structure to the datatsets of real images, in our case OldHospital and KingsCollege datasets. This similarity should be not only on the shape and distribution of objects in the scene, but also on the way images are acquired (i.e., similar relative distance and orientation between the cameras and objects in the scene for the generation of synthetic images).

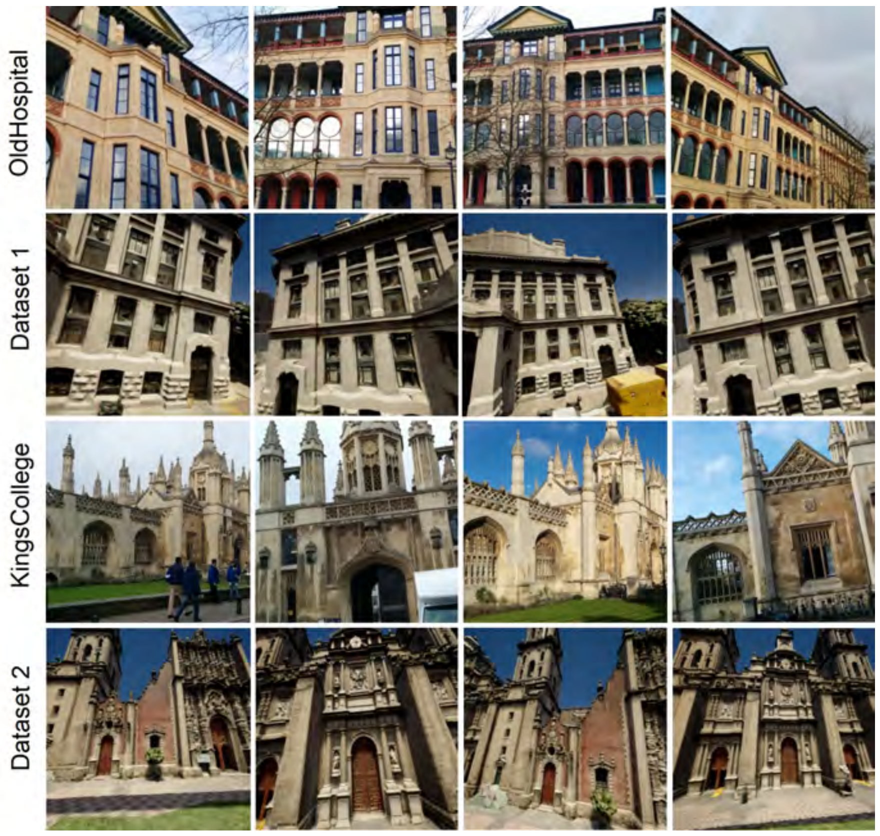

Fig. 3 Synthetic images generated using CARLA Simulator tool. (*2nd row*) Virtual scenario similar to OldHospital dataset. (*4th row*) Virtual scenario similar to KingsCollege dataset.

In order to generate the two datasets with synthetic images similar to OldHospital and KingsCollege datasets, two 3D virtual scenarios are considered. These 3D virtual scenarios have similar structure to the real scenarios, i.e., they should have the same feature spaces (objects' geometry and camera point of view). Figure 3

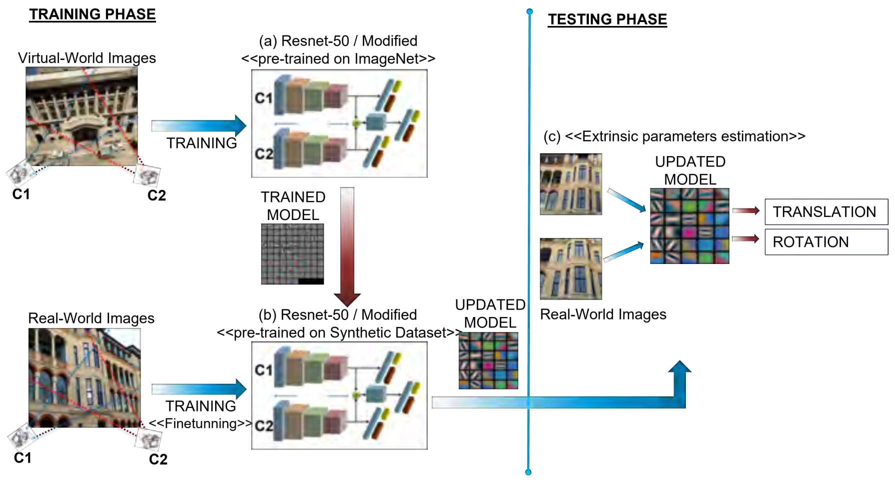

Fig. 4 (a) RelPoseTL is trained using synthetic images generated from CARLA Simulator. (b) The learned knowledge is used to apply DA strategy using real images. (c) Updated weights after DA strategy are used to estimate relative camera pose (i.e., relative translation and rotation).

shows illustrations of the virtual scenarios generated from the CARLA Simulator tool. The Dataset 1 and Dataset 2 are similar to the OldHospital and KingsColleges datasets respectively. Additionally, two virtual cameras were also considered. The cameras move in these virtual scenarios and acquire pair of images. The cameras start the trajectories with a given initial position and orientation with respect to the world reference system. Then, the pair of cameras moves together randomly and change their relative orientation also randomly. The images are simultaneously acquired for each camera when there is enough overlap between their field of view. This process generates about 3000 synthetic images from both synchronized cameras, for each scenario; it takes about three hours for each scenario. The overlap between pairs of synthetic images is computed with OpenMVG [21], where a minimum of 60% overlap between acquired synthetic images is imposed.

Once datasets with synthetic images have been generated as mentioned above, the RelPoseTL model is trained by initializing all layers of the architecture as presented in Section 2.2. The Adam optimizer is used for the training process with batch size of 32 and learning rate of 10^{-4}. The synthetic images were resized to 224×224 pixels, including data normalization for training process. A set of 8124 pairs of images was considered to train the architecture, which took about 30 hour per dataset till convergence was reached. During the evaluation phase, images were pre-processed as mentioned above. A set of 2048 pairs of images from each of the two datasets was used.

Finally, in order to transfer the knowledge learned in the synthetic image domain to the new domain (i.e., real-world), the Domain Adaptation (DA) strategy is applied. It consists on training again the RelPoseTL architecture but now with just a few pairs of real-world images; the weights from the synthetic image domain are used as initialization. It helps to speed up the training process as well as to avoid the time consuming ground truth generation on real-world images. This strategy is also referred in the literature to as transfer learning. Figure 4 shows an illustration of this DA strategy. In details, each layer of RelPoseTL is initialized with the learned weights from the training process using synthetic images, which should be similar to the real-world scenarios. In order to show the advantages of this strategy, in this section, sets of (256, 512 and 1024) pairs of images from the real-world scenario are considered. These images are from the OldHospital and KingsCollege of Cambridge Landmark dataset [18]. They are used to train the model and results are compared with those obtained by initializing the network with the weights obtained from the synthetitc image domain. In order to compare the results obtained when the network is initialized with ImageNet weights or with those from the synthetic scenario—the DA strategy—three sets of 64, 128 and 256 pairs of real-world images are considered. Quantitative results are presented in Table 2 and Table 3. Angular error is used to evaluate the rotation error from the estimated quaternion. On the other hand Euclidean error is used to measure the error between the estimated translation and ground truth value. In the case of OldHospital dataset, the DA strategy, which consists on using initially the Dataset 1 for training the model from scratch and then transferring the learned knowledge for retraining the model with just few real images, reaches the best result. This best result has been obtained in all cases, even if they are compared with the model trained using just the real images dataset. A similar process was performed with KingsCollege dataset using Dataset 2 where the results are the best even if the model is just trained using real images dataset, showing that the similarity of features spaces (i.e., 3D scenario used to represent the real environments) as well as in the camera pose (i.e., relative distance between the cameras and the objects in the scene as well as their relative orientation) helps to improve the camera pose estimation when DA is applied.

Table 2 Angular and Euclidian distance errors of RelPoseTL [4] trained with real datat (RD) and with the domain adaptation (DA) strategy on pairs of images (PoI) from OldHospital dataset; on the first row RelPoseTL is initialized with ImageNet weights, while in the second row the weights are obtained by pre-training RelPoseTL with synthetic datatset (SD1).

	DA strategy on OldHospital dataset		
Trained with	Train: 256 PoI Test: 64 PoI	Train: 512 PoI Test: 128 PoI	Train: 1024 PoI Test: 256 PoI
RD (Init. ImageNet)	4.29m, 5.72º	3.93m, 4.04º	3.48m, 3.95º
RD (Init. SD1)	3.55m, 5.59º	3.40m, 3.70º	3.20m, 3.54º

Table 3 Angular and Euclidian distance errors of RelPoseTL [4] trained with real datat (RD) and with the domain adaptation (DA) strategy on pairs of images (PoI) from KingsCollege dataset; on the first row RelPoseTL is initialized with ImageNet weights, while in the second row the weights are obtained by pre-training RelPoseTL with synthetic datatset (SD2).

	DA strategy on KingsCollege dataset		
Trained with	Train: 256 PoI Test: 64 PoI	Train: 512 PoI Test: 128 PoI	Train: 1024 PoI Test: 256 PoI
RD (Init. ImageNet)	5.28m, 5.29º	3.86m, 5.08º	2.95m, 4.06º
RD (Init. SD2)	4.89m, 4.96º	3.13m, 4.18º	2.35m, 3.32º

3 Human Pose Estimation

The 2D human pose is estimated by detecting human body joints (e.g., elbow, wrist, head, etc.) from images and then connecting them to build the human body figure. During last years, different approaches have been proposed for the human pose estimation (HPE), for instance OpenPose [2], DeepPose [28], Convolutional pose machine [29], just to mentioned a few; appealing results have been shown from these approaches, mainly when all body joints are detected. However, the natural pose of human body generally generally involves self-occlusions that make the HPE a challenging problem in monocular vision system scenarios.

An alternative to overcome the problems of monocular vision systems could be by considering multi-view approaches. In these approaches, since the human body is captured from different points of view at the same time, occluded joints from one view can be observed from another view. The multi-view approaches have been already used to tackle the region occlusion problem in certain tasks such as 3D-reconstruction, camera pose, autonomous driving, object detection (e.g., [31, 25, 5, 14, 27]). However, few works have leveraged the advantages of multi-view approaches to overcome the occlusion problem in the human pose estimation. Some works exploit the epipolar geometry of multi-view approaches to solve the region occlusion problem as the authors in [23]. In the current work, a CNN architecture is used to fuse all features on epipolar line of the images of all points of view as previous step to get the predicted joints. Another approach has been proposed in [13], where the author leveraged the extracted feature of intermediate layers to find its corresponding points in a neighboring view to combine and robustly extract features of each view.

The performance of these approaches allow that certain applications such as action recognition, healthcare, or augmented reality, take advantage of the obtained accuracy to develop different solutions. In this section a novel multi-view scheme is presented to robustly estimate the human body pose. The architecture to tackle the human pose estimation problem from a multi-view scheme is presented in Section 3.1. Then, experimental results are presented in Section 3.2.

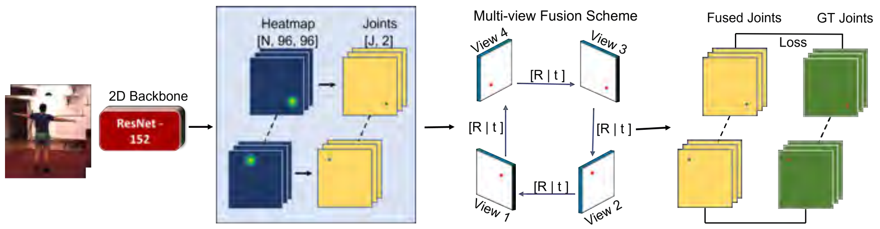

Fig. 5 CNN backbone feeds with a set of pairs of images of the same scene simultaneously acquired from different points of view. The multi-view fusion scheme allows to estimate occluded joints with information from other views across of the relative camera pose.

3.1 Multi-View Scheme

The multi-view scheme presented in [3], which is referred to as Mview-Joints, tackles the self-occlusion problem in the 2D human pose estimation by considering at least two views. It uses the CNN backbone proposed by [16], which is a variant of Resnet-152 with learnable weights, and as output, the number of body joints is considered.

The model is fed by a set of images containing just a single-person, which has been captured from a multi-view system of C calibrated and synchronized cameras with known parameters. The images captured by the multi-view system are organized in pairs of images using two different close views, namely, reference view Im^{ref} and source view Im^{src}. Heatmaps obtained from each image, like those resulting from the usage of the backbone [16], are fused across source view considering the confidence of each joint and the relative camera pose, improving the accuracy of joints from each image (see Fig. 5).

In order to estimate the 2D position of the joints ($p_{(x,y)}$) in the image plane, the center of mass of each heatmap is computed as follow:

$$p_{(x,y)} = \sum_{u=1}^{W} \sum_{v=1}^{H} h_{i_{(u,v)}}.(\zeta_{\Theta}(h_{i_{(u,v)}})), \tag{10}$$

where ζ_{Θ} represents the function softmax; h represents the ROI of the heatmaps of i-th joint and W and H correspond to the size of the heatmap ROI. For each 2D position of each joint obtained using Eq. (10), its position in the world coordinate system $P = (X, Y, Z)$ is obtained, as shown below:

$$x_i = f\frac{X}{Z} \qquad y_i = f\frac{Y}{Z}, \tag{11}$$

where (x, y) is the 2D position of the i-th joint obtained in Eq. (10). The focal length of the camera is defined as f. Since the depth (Z) of the joint is unknown, two values are empirically defined to solve the Eq.(11). The first one corresponds to a depth value close to zero while the second value corresponds to a depth near to the size of the scene, in our case it has been set to $10m$.

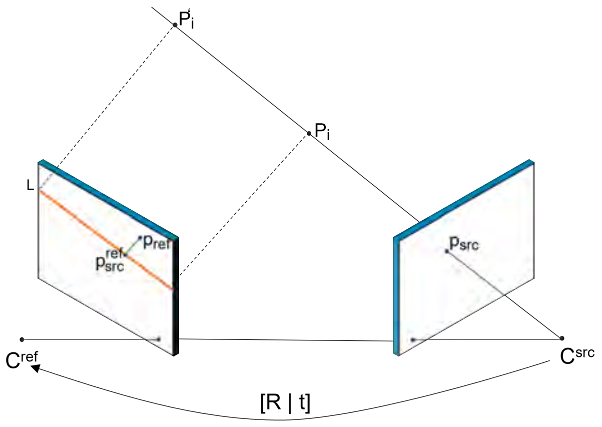

Fig. 6 An image point p_{src} back-projects to a ray in 3D defined by the point p_{src} and two depth values p_i and p'_i; it is then projected to the image plane of reference view to generate the epipolar line (L).

The position in the world coordinate system of each joint is transformed using the relative camera pose between both points of view (see Fig. 6), i.e, source and reference view, and then, projected to the image plane, as shown below:

$$T_{rel} = Rot_{src} \cdot (T_{ref} - T_{src}), \tag{12}$$

$$Rot_{rel} = Q(Rot_{ref}.T)^{-1} * Q(Rot_{src}.T), \tag{13}$$

$$p^{ref}_{src_{(x,y)}} = \Delta 2D_{ref}(Rot_{rel} \cdot (P_i - T_{rel})), \tag{14}$$

where $Q(.)$ represents the quaternion. The rotation matrix and the translation vector are defined as $Rot \in \mathbb{R}^{3x3}$ and $T \in \mathbb{R}^{3x1}$ respectively. P_i corresponds at i-th joint in the world coordinate system obtained in Eq. (11). The projected line on image plane of reference view L is obtained using the linear equation and the point computed in the Eq. (14). In order to obtain the depth of i-th joint of the source view, which should be on the projected line on image plane of reference view, the intersection between the projected line and the 2D-point of the joint computed in the reference view $p_{ref_{(x,y)}}$ is performed.

The confidence of the two different 2D positions in the plane of the reference image of the i-th joint, where the first corresponds to the reference view $p_{ref_{(x,y)}}$ and the second, a projected joint from source to reference view $p^{ref}_{src_{(x,y)}}$, is computed as the distance between the ground truth of 2D position of i-th joint and the estimated 2D positions of i-th joints. These confidence values are used as shown in Eq. (16).

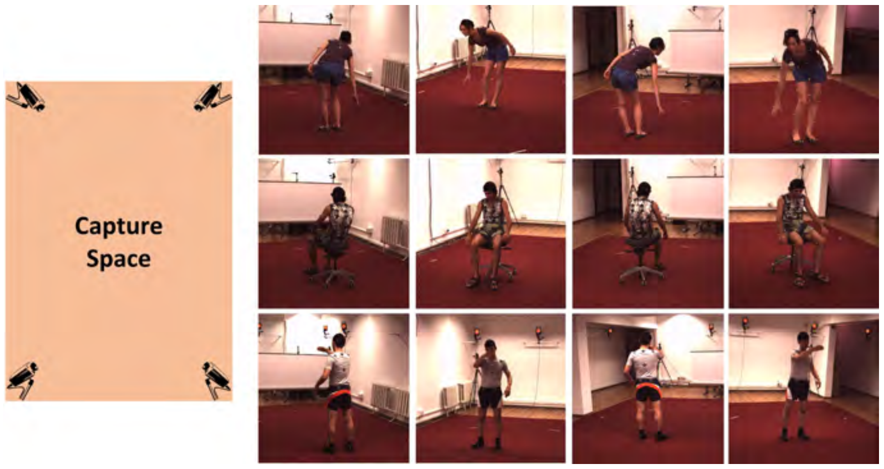

Fig. 7 Human3.6m dataset used for the training and evaluation processes. The subject is captured from different point of views considering the four calibrated and synchronized cameras located in each corner of the room.

$$\omega = 1 - \left| \frac{D_\Delta(\hat{\gamma}_i, \gamma_i)}{\sum D_\Delta(\hat{\gamma}_i, \gamma_i)} \right|, \tag{15}$$

$$\delta_{upd_{i(x,y)}} = \omega * p_{i(x,y)}, \tag{16}$$

where $(\hat{\gamma}$, $\gamma)$ represent the ground truth and prediction of 2D position of i-th joint respectively, and ω corresponds to the confidence of the points of i-th joints in the reference view.

The enhanced 2D position of i-th joint is denoted as $\delta_{upd_{i(x,y)}}$, which considers the information and confidence of i-th joint projected from the source to reference view. In order to minimize the error between the enhanced 2D position of i-th joint and its ground truth in the learning process of the proposed multi-view scheme, a loss function is defined as:

$$Loss = \sum_{i=1}^{N} \left\| \delta_{upd_{i(x,y)}} - \hat{p}_{i(x,y)} \right\|_2, \tag{17}$$

where N corresponds to the number of joints, and $\hat{p}_{i(x,y)}$ is the ground-truth of i-th joint in image plane.

3.2 Results from Multi-View Approach

The Mview-Joints architecture presented in the previous section has been trained and evaluated using the Human3.6m dataset [15]. Human3.6m is one of the largest publicly available benchmark, with a multi-view setup, for human pose estimation. In details, four synchronized and calibrated cameras are considered to generate a set of images with a single-person doing different activities (Fig. 7 shows three different captures from the four cameras).

Sets of images of Human3.6m dataset were cropped according to the bounding box of the person and resized to 384x384 pixels as a previous step for training the model. Mview-Joints was firstly initialized with the weight pretrained by [16] and trained on a set of 60k pre-processed images of Human3.6m dataset, in an end-to-end way, until 20 epochs. This training process takes about 120 hours. The same pre-processing is used during the evaluation phase with a set of 8k images. In order to evaluate the performance of Mview-Joints architecture for 2D human pose estimation, the Joint Detection Rate (JDR) is used. This metric measures the percentage of $successfully$ detected joints, which uses a threshold to validate if a joint has been $successfully$ detected. The threshold is defined as half of the head size. In addition, the Euclidean distance error is used to compute the accuracy of each human body joint estimated by the network.

Results and comparisons with state-of-the-art CNN-based approaches are presented in Table 4 and Table 5 using JDR metric. As it can be appreciated, the improvement is most significant for shoulder, elbow and ankle joints, which increment from 96.44% to 99.65%, from 95.00% to 97.31% and from 96.62% to 97.45%, respectively. In term of average JDR, the results obtained by Mview-Joints improves the Epipolar transformer model [13] about 1%, and with respect to Cross-View fusion [23] about 3%.

Table 4 Comparison of 2D pose estimation accuracy on Human3.6m dataset using JDR(%) as metric. " – ": these entries were absent. $*$: approach presented in [23]. Υ trained again by [13]. ψ approach presented in [13]. R50 and R152 are ResNet-50 and ResNet-152 respectively. Scale is the input resolution of the network.

	Net	scale	shlder	elb	wri	hip	knee	ankle	root	neck	head	Avg
Sum epipolar line *	R152	320	91.36	91.23	89.63	96.19	94.14	90.38	-	-	-	-
Max epipolar line *	R152	320	92.67	92.45	91.57	97.69	95.01	91.88	-	-	-	-
Cross-View fusion $^{*\Upsilon}$	R50	320	95.6	95.0	**93.7**	96.6	95.5	92.8	96.7	96.5	96.2	95.9
Cross-View fusion $^{*\Upsilon}$	R50	256	86.1	86.5	82.4	96.7	91.5	79.0	**100**	93.7	95.5	95.1
Epipolar transformer $^{\psi}$	R50	256	96.44	94.16	92.16	98.95	**97.26**	96.62	99.89	99.68	99.63	97.01
Mview-Joints	R152	384	**99.65**	**97.31**	**93.70**	**99.22**	97.24	**97.45**	99.83	**99.82**	**99.75**	**98.22**

Table 5 shows the Euclidean distance errors. In this case the accuracy of estimated body joints using the Mview-Joints architecture is compared with respect to the CNN backbone proposed by [16]. As it can be appreciated, body joints such as elbow, wrist, knee, nose, head improve by 15.88%, 4.32%, 8.46%, 18.11% and 50.53% respectively, when compared to the results obtained with CNN backbone proposed by [16]. Since the multi-view scheme leverages the different views of the scenario, the challenging human body pose are better estimated. Figure 8 shows some scenes

Table 5 Comparison of average Euclidean distance error between Mview-Joints and Learning triangulation backbone proposed by [16] on Human3.6m dataset ($Backbone$: Resnet 152 with pretrained weight [16]).

	Net	shlder	elb	wri	hip	knee	ankle	root	neck	nose	belly	head	Avg
Learning triangulation	Backbone	**7.84**	8.00	7.40	**7.55**	7.45	9.70	5.75	**5.86**	6.46	6.47	6.57	7.18
Mview-Joints	Backbone + Multi-view	7.88	**6.73**	**7.08**	7.62	**6.82**	**9.19**	**5.24**	6.05	**5.29**	**6.15**	**3.25**	**6.48**

with self-occlusions where it can be appreciated that Mview-Joints architecture is able to estimate the human body pose better than single view approach [16].

4 Conclusions

This chapter focuses on the challenging problem of human body pose estimation in multi-view scenarios. It is intended to tackle self-occlusion problems by accurately estimating the human body pose. Firstly, in order to put the different views in a common framework, the relative position and orientation—extrinsic camera parameters—between the different cameras is estimated by using a deep learning based strategy, instead of classical calibration-pattern based approaches. In order to train this extrinsic camera calibration network, synthetic datasets of outdoor scenarios are generated overcoming the limitation of lack of annotated real-world data. Then, once relative pose between cameras is estimated, human body pose in the multi-view scenario is obtained by using an adaptation of an architecture initially intended for single view scenarios. Experimental results of estimated human pose and comparisons with state-of-the-art approaches are provided showing improvements on challenging scenarios. This chapter shows how information from different views can be fused in order to reach a more accurate representation than single view approaches, in particular when self-occlusions are considered. An important aspect to consider is that the precision of body joint estimations is the base to solve other related problems such as action recognition, surveillance, 3D human pose estimation among other. Future work will be focused on extending the usage of multi-view environments to leverage the geometry of the scene, and thus, improve the 3D human pose. Additionally, the usage of attention modules will also be considered to tackle the occluded regions into the multi-view scheme.

Acknowledgements This work has been partially supported by the ESPOL projects EPASI (CIDIS-01-2018), TICs4CI (FIEC-16-2018) and PhysicalDistancing (CIDIS-56-2020); and the "CERCA Programme/Generalitat de Catalunya". The authors acknowledge the support of CYTED Network: "Ibero-American Thematic Network on ICT Applications for Smart Cities" (REF-518RT0559) and the NVIDIA Corporation for the donation of the Titan Xp GPU. The first author has been supported by Ecuador government under a SENESCYT scholarship contract CZ05-000040-2018.

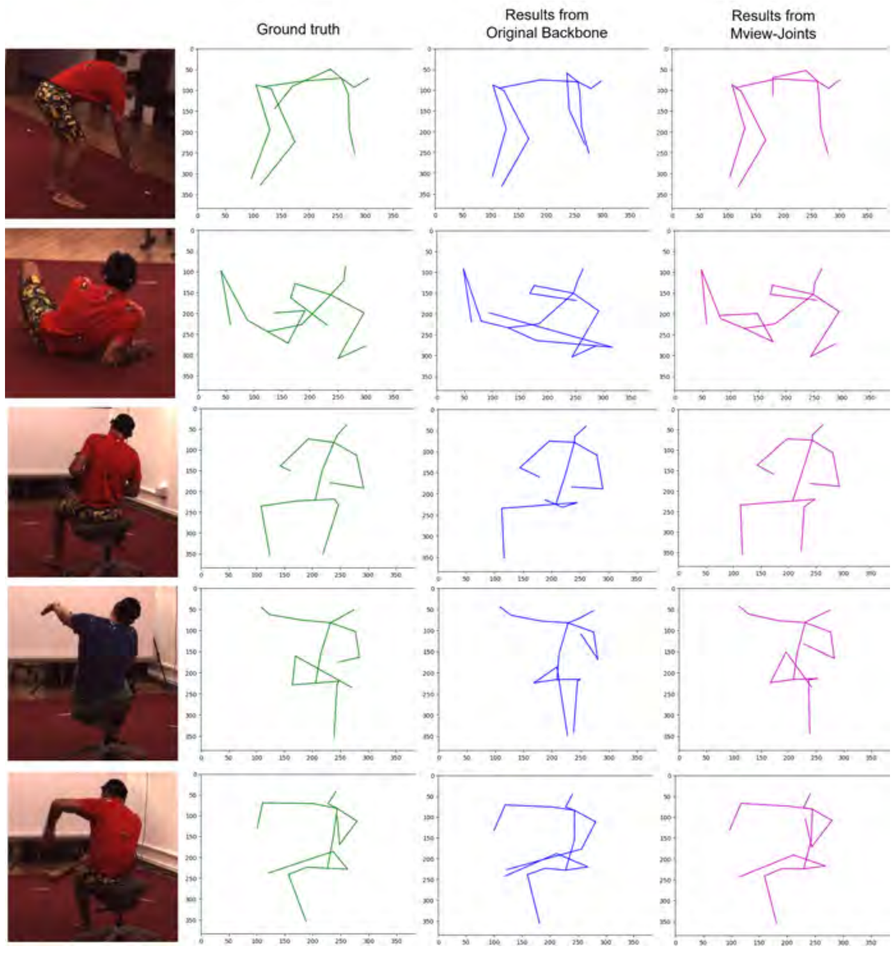

Fig. 8 Qualitative results on challenging scenarios— Mview-Joints architecture obtains better estimations than the backbone proposed by [16].

References

1. Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008.
2. Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Openpose: realtime multi-person 2d pose estimation using part affinity fields. *IEEE transactions on pattern analysis and machine intelligence*, 43(1):172–186, 2019.
3. Jorge L Charco, Angel D Sappa, and Boris X Vintimilla. Human pose estimation through a novel multi-view scheme. In *Proceedings of the 17th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 5: VISAPP,*, pages 855–862. INSTICC, SciTePress, 2022.

4. Jorge L. Charco., Angel D. Sappa., Boris X. Vintimilla., and Henry O. Velesaca. Transfer learning from synthetic data in the camera pose estimation problem. In *Proceedings of the 15th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 4: VISAPP,*, pages 498–505. INSTICC, SciTePress, 2020.
5. Jorge L Charco, Angel D Sappa, Boris X Vintimilla, and Henry O Velesaca. Camera pose estimation in multi-view environments: From virtual scenarios to the real world. *Image and Vision Computing*, 110:104182, 2021.
6. Jorge L Charco, Boris X Vintimilla, and Angel D Sappa. Deep learning based camera pose estimation in multi-view environment. In *2018 14th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS)*, pages 224–228. IEEE, 2018.
7. Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015.
8. Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017.
9. Sovann En, Alexis Lechervy, and Frédéric Jurie. Rpnet: an end-to-end network for relative camera pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.
10. Hao-Shu Fang, Shuqin Xie, Yu-Wing Tai, and Cewu Lu. Rmpe: Regional multi-person pose estimation. In *Proceedings of the IEEE international conference on computer vision*, pages 2334–2343, 2017.
11. Adrien Gaidon, Qiao Wang, Yohann Cabon, and Eleonora Vig. Virtual worlds as proxy for multi-object tracking analysis. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 4340–4349, 2016.
12. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
13. Yihui He, Rui Yan, Katerina Fragkiadaki, and Shoou-I Yu. Epipolar transformers. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, pages 7779–7788, 2020.
14. Markus Hofbauer, Christopher B Kuhn, Jiaming Meng, Goran Petrovic, and Eckehard Steinbach. Multi-view region of interest prediction for autonomous driving using semi-supervised labeling. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–6. IEEE, 2020.
15. Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1325–1339, jul 2014.
16. Karim Iskakov, Egor Burkov, Victor Lempitsky, and Yury Malkov. Learnable triangulation of human pose. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7718–7727, 2019.
17. Alex Kendall, Roberto Cipolla, et al. Geometric loss functions for camera pose regression with deep learning. In *Proc. CVPR*, volume 3, page 8, 2017.
18. Alex Kendall, Matthew Grimes, and Roberto Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *Proceedings of the IEEE international conference on computer vision*, pages 2938–2946, 2015.
19. Iaroslav Melekhov, Juha Ylioinas, Juho Kannala, and Esa Rahtu. Relative camera pose estimation using convolutional neural networks. In *International Conference on Advanced Concepts for Intelligent Vision Systems*, pages 675–687. Springer, 2017.
20. Eric N Mortensen, Hongli Deng, and Linda Shapiro. A sift descriptor with global context. In *Computer vision and pattern recognition, 2005. CVPR 2005. IEEE computer society conference on*, volume 1, pages 184–190. IEEE, 2005.
21. Pierre Moulon, Pascal Monasse, Renaud Marlet, and Others. Openmvg. an open multiple view geometry library. https://github.com/openMVG/openMVG, 2016.
22. Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European conference on computer vision*, pages 483–499. Springer, 2016.

23. Haibo Qiu, Chunyu Wang, Jingdong Wang, Naiyan Wang, and Wenjun Zeng. Cross view fusion for 3d human pose estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4342–4351, 2019.
24. Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *Computer Vision (ICCV), 2011 IEEE international conference on*, pages 2564–2571. IEEE, 2011.
25. Hamid Sarmadi, Rafael Muñoz-Salinas, MA Berbís, and RJIA Medina-Carnicer. Simultaneous multi-view camera pose estimation and object tracking with squared planar markers. *IEEE Access*, 7:22927–22940, 2019.
26. Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5693–5703, 2019.
27. Cong Tang, Yongshun Ling, Xing Yang, Wei Jin, and Chao Zheng. Multi-view object detection based on deep learning. *Applied Sciences*, 8(9):1423, 2018.
28. Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1653–1660, 2014.
29. Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 4724–4732, 2016.
30. Minghu Wu, Hanhui Yue, Juan Wang, Yongxi Huang, Min Liu, Yuhan Jiang, Cong Ke, and Cheng Zeng. Object detection based on rgc mask r-cnn. *IET Image Processing*, 14(8):1502–1508, 2020.
31. Haozhe Xie, Hongxun Yao, Xiaoshuai Sun, Shangchen Zhou, and Shengping Zhang. Pix2vox: Context-aware 3d reconstruction from single and multi-view images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2690–2698, 2019.