applied sciences

MDPI

Article

CFAM: Estimating 3D Hand Poses from a Single RGB Image with Attention

Xianghan Wang [1], Jie Jiang [1,*], Yanming Guo [1], Lai Kang [1,*], Yingmei Wei [1] and Dan Li [2]

[1] College of Systems Engineering, National University of Defense Technology, Changsha 410073, China; 18373140341@163.com (X.W.); guoyanming@nudt.edu.cn (Y.G.); weiyingmei@hotmail.com (Y.W.)
[2] Graduate College, National University of Defense Technology, Changsha 410073, China; 13317310232@163.com
* Correspondence: jiejiang@nudt.edu.cn (J.J.); kanglai123@yeah.net (L.K.)

Received: 28 November 2019; Accepted: 10 January 2020; Published: 15 January 2020

Abstract: Precise 3D hand pose estimation can be used to improve the performance of human–computer interaction (HCI). Specifically, computer-vision-based hand pose estimation can make this process more natural. Most traditional computer-vision-based hand pose estimation methods use depth images as the input, which requires complicated and expensive acquisition equipment. Estimation through a single RGB image is more convenient and less expensive. Previous methods based on RGB images utilize only 2D keypoint score maps to recover 3D hand poses but ignore the hand texture features and the underlying spatial information in the RGB image, which leads to a relatively low accuracy. To address this issue, we propose a channel fusion attention mechanism that combines 2D keypoint features and RGB image features at the channel level. In particular, the proposed method replans weights by using cascading RGB images and 2D keypoint features, which enables rational planning and the utilization of various features. Moreover, our method improves the fusion performance of different types of feature maps. Multiple contrast experiments on public datasets demonstrate that the accuracy of our proposed method is comparable to the state-of-the-art accuracy.

Keywords: hand pose estimation; CFAM; 3D keypoint; RGB image; attention

1. Introduction

Gesture estimation plays a significant role in computer science, and related tasks aim toward understanding human gestures through algorithms. It is robust to environment changes such as mutative shooting distance and glare light. Human–computer interaction (HCI) can be implemented wherever and whenever, has fewer constraints, and enables computers to efficiently and precisely understand user commands without any mechanical assistance. Gestures for HCI are quick, vivid, intuitive, flexible, and visual; they can enable soundless interactions and bridge the gap between the real world and virtual worlds.

To recognize gestures, 3D hand poses are required. Computer-vision-based hand pose estimation enables people to communicate with machines more naturally. With the development of computer vision, pose estimation no longer relies on traditional wearable devices in specific scenes but can be directly implemented based on image recognition. The research on pose estimation in computer vision includes three main categories: depth images, multivision RGB images, and single RGB images. Many studies have estimated hand poses through depth images [1–9] and achieved good results. However, depth images must be obtained using indoor depth cameras and are thus not as convenient as RGB images. Multivision has successfully achieved hand tracking and hand pose estimation through RGB images [10] but there are still some constraints on users due to the requirements of multivision.

The goal of hand pose estimation based on computer vision is to free users from the constraints of depth equipment and multivision images and to facilitate HCI through mobile phones and other devices. Therefore, it is of great significance to be able to estimate hand poses based on a single RGB image, which allows full 3D hand poses to be learned from a single RGB image and does not rely on any special equipment or environment.

At present, estimating 3D hand poses from 2D score maps is the most commonly used pose estimation method based on single RGB images. Although some methods for human body posture estimation can turn RGB images into 3D postures, they cannot be directly applied to hand pose estimation. Moreover, hands have a more serious self-shielding problem than other parts of the human body as the inside of each hand is asymmetrical while the human body is symmetrical. A recent hand pose estimation method based on a single RGB image first estimates the hand state and the rotation angle relative to the camera, and then calculates the 3D coordinates. Unfortunately, for this method, the estimation is based on only 2D keypoints, and the texture features of the RGB image are ignored. In this paper, a fusing channel attention method is introduced to combine a 2D pose and an RGB image to estimate a 3D pose, which effectively solves the problem of different types of input data. The input is an RGB image of a human hand. After applying the end-to-end neural network, we obtain a 3D array. The 3D array is the spatial location of the 21 keypoints of the hand in the input image, as shown in Figure 1.

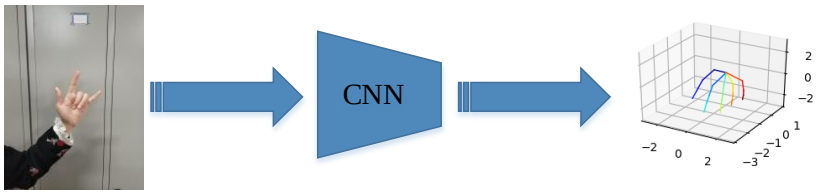


Figure 1. Our task. The 3D hand pose is estimated by an end-to-end convolutional neural network (CNN). The input is an RGB image, and the output is the 3D coordinate of each keypoint on the hand.

2. Related Work

A computer-vision-based 3D hand pose can be estimated from an RGB image or a depth image by using computer vision. We introduce estimation methods based on depth images and RGB images in this chapter.

2.1. 3D Hand Pose Estimation Based on Depth Images

Traditional computer-based 3D hand pose estimation uses depth images. Depth images include depth information, which is helpful for obtaining the distances between keypoints.

Markus et al. [1] proposed a preliminary positioning and optimization method, HandDeep, based on a convolutional neural network. It can accurately locate a hand in a single depth image after training with multiple labeled depth images. In 2016, Ayan et al. [2] proposed an acceleration method using matrix completion, and it can be applied to large-scale, real-time hand pose estimation without relying on a GPU. In 2017, Overweger et al. [3] optimized several aspects of the network and the training process and improved the accuracy. The method included data expansion, dropout, the addition of residual modules, and the optimization of hand segmentation. In 2014, Tomsons et al. [4] proposed a hand pose recognition method that combines the generation method and the data-driven method. In 2016, Sun et al. [5] proposed an algorithm for matching depth images with models. By constructing a hand model, this method matched the keypoints from the palm to fingertip. Wan et al. [11] proposed a method for dense pixels that aggregated local estimates using nonparametric mean shift variables, explicitly forcing consistency between the estimated 3D joint coordinates and the 2D and 3D local estimations. This approach provided a better fusion between 2D detection

and 3D regression than prior mechanisms and various baselines. In 2018, Aisha et al. [12] set the gesture segmentation under the first-person perspective and the presence of occlusion with the aid of a conditional random field (CRF). For the first time, a method performed hand segmentation and detection from a self-centered perspective and under occlusion, and the hand pose estimation accuracy was improved by improving the segmentation accuracy. However, this method still did not solve the problem of occluded objects or the similarity between background objects and hands in RGB images. Motivated by CycleGAN [13], Baek et al. [14] proposed a method for expanding datasets. This method can actively generate keypoint data by training datasets and restore depth images through a GAN (Generative Adversarial Networks) after CycleGAN training. To some extent, the lack of training data for partial perspectives was solved. The proposed solution was useful to some extent. However, a complicated cyclical relationship was used, which made the training process cumbersome and the network complicated. Wan et al. [15] matched depth images to bone images based on a hidden space transformation. Although the accuracy was mediocre, the method could achieve a speed of 90 frames per second (FPS) on a CPU, improving the efficiency of realizing image-based hand pose estimation. The method mapped paired depth and bone images to the same position in the hidden space and restored the original image from the hidden space via deconvolution. The depth-image-based pose estimation method has gradually matured, but the depth acquisition device, which is sensitive to illumination, jitter, and distance, imposes constraints on the user. In addition, it is expensive.

2.2. 3D Hand Pose Estimation Based on RGB Images

Due to the lack of depth information, hand pose estimation based on RGB images, especially single RGB images, developed relatively late. The accuracy of the RGB-based method is not as good as that of the depth-based method. However, an RGB image is easier to obtain, and the equipment is cheaper. Thus, an increasing amount of research focusses on the RGB-based method.

Zhang [10] proposed estimating poses based on multivision and using binocular vision to restore the exact distance information and realized RGB-based hand pose estimation. However, this method still places a large number of constraints on users. In 2017, Zimmermann [16] realized 3D hand pose estimation through a single RGB image based on deep learning; the method used deep networks to learn reasonable prior information from the data to solve the fuzzy problem without relying on any special equipment. A feasible network framework for deriving 3D keypoints from 2D keypoints was generated. The method consisted of three deep networks: the first network performed hand segmentation to locate the hand in the image, the second network estimated the 2D keypoint score map from the output of the first network using convolutional pose machines (CPMs) [17], and the third network derived a 3D keypoint from the 2D keypoints. Furthermore, the method proposed a normalized coordinate system that regarded the hand position in the normalized coordinate system as a rotation in the camera coordinate system; the hand position was calculated in the normalized coordinate system, and the rotation angle was calculated by using the neural network to restore the position of the 3D keypoint. This method was the first to achieve 3D hand pose estimation with a single RGB image. Spur [18] used a variational encoder hand pose estimation method to project the image and keypoint information onto the hidden space and optimized the accuracy by minimizing the distance between the image and the information in the hidden space. Dibra [19] used weakly supervised learning to estimate hand poses. This method does not directly perform supervision training through the three-dimensional hand keypoints, but rather generates depth images of the estimated 3D hand pose through a GAN. Muller [20] restored occluded hand areas through a GAN, which solved the problem of hand area occlusion to a certain extent.

Of the computer-vision-based hand pose estimation methods, the depth-based method requires more expensive equipment, but the multivision-based method still places certain constraints on the user. Most methods based on a single RGB image use a 2D score map but ignore the information contained in the RGB image. Based on 3D hand estimation from a single RGB image, we propose a method that uses the attention mechanism to fuse the 2D score map and the RGB image channel

features. In Section 3, we introduce the prior methods and our methods. In Section 4, we introduce our experimental dataset and compare it with the baseline in the dataset and with the state-of-the-art methods. In Section 5, we summarize our paper.

3. Method

As shown in Figure 2, the task is divided into three steps: first, the hand bounding box is cropped from the input image; second, 2D keypoints are calculated from the hand bounding box; third, the 3D hand pose is estimated from 2D keypoints and the hand bounding box. The main step is the third one. We introduce the first and second steps in Section 3.1, and we introduce 3D hand pose estimation from 2D keypoints in Section 3.2. For fusion, we used the RGB image and 2D keypoint method, as described in Section 3.3.

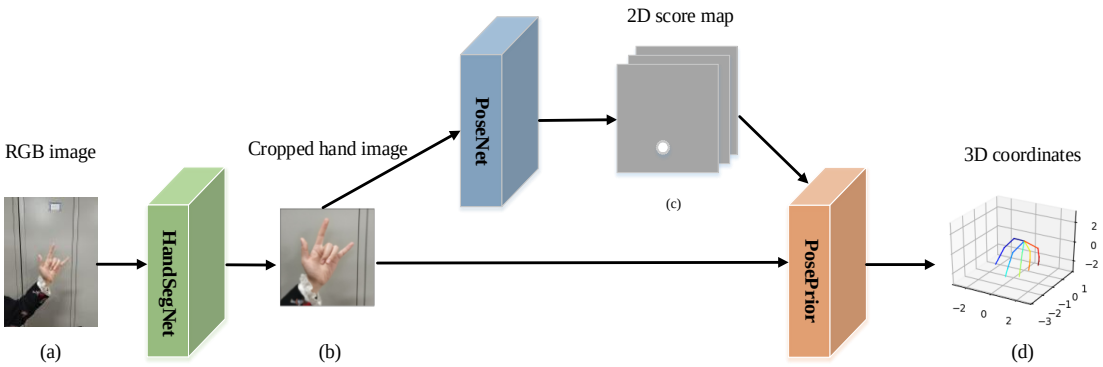


Figure 2. The framework of our method. The process is divided into three steps. First, the RGB image (**a**) is brought into the the network as the input. The hand area (**b**) is cropped from the whole RGB image using *HandSegNet* [16]. Then, the 2D score map (**c**) is estimated according to the hand area. Finally, 3D coordinates (**d**) are estimated from the 2D score map and cropped hand area by *PosePrior*.

3.1. 2D Keypoint Calculation

In this section, we introduce *HandSegNet* to obtain the bounding box of the hand from the input image and *PoseNet* [16] to get the 2D keypoints.

We use J to represent the different hand keypoints; $J = \{1, 21\}$ since we found 21 useful keypoints on one hand. $W = \left\{w_J = (x, y, z), J \in [1, 21]\right\}$ represents the 3D position of each hand keypoint. The input $I \in R^{w \times h \times 3}$ is the RGB image. The picture of the cropped hand mask is $I_{mask} \in \mathbb{R}^{w_m \times h_m \times 3}$, which is smaller than the whole input I and includes only the hand mask. $R = \left(R_x, R_y, R_z\right)$ represents the rotation angle of the camera coordinate system relative to the world coordinate system. We use 2D Gaussian keypoint score maps $P = p_J(u, v), J \in [1, 21]$ to present the 2D keypoints, where each score map corresponds to one keypoint. (u, v) is the position of the keypoint where the Gaussian score map $p_J(u, v)$ is centered. It is beneficial for the network to learn possible positions of the keypoints during the training process.

2D keypoint calculation is key to 3D keypoint estimation. To calculate the 2D keypoints, *HandSegNet* is first used to estimate the region of the hands, namely, *handmask* I_{mask}, from the original image $I \in \mathbb{R}^{w \times h \times 3}$.

The first neural network used in this method is *HandSegNet*, whose task is to crop the hand region from the image. Directly cropping the horizontal and vertical coordinates of the hand region to obtain a rectangular block is a regression problem. Neural networks are not as good at regression tasks as they are at classification tasks [2]. Thus, we first get the mask of the hand and regard our goal as a task for computing bool images. Next, we determine whether each pixel in the image belongs to the area of the hand. For each pixel, we calculate the probability (P_i) that it belongs the hand mask. When the probability is greater than a threshold, the point is considered to belong to the hand mask.

Then, the center of mass of the hand mask is calculated. The hand area is cropped around the center of mass.

Then, I_{mask} is fed into *PoseNet* to estimate score maps of different 2D keypoints. The 2D keypoints are estimated from the cropped hand images. Traditional methods directly predict the x and y values of each keypoint, which ignores the connection between the fingers and is a task at which the neural network is not good. We do not just estimate the x and y values of the keypoints like traditional methods; instead, we obtain a score map of each keypoint, such as that shown in Figure 2c. The score map also represents the location of the keypoint, and it can be understood better by neural network.

3.2. 3D Hand Pose Estimation

The 3D hand pose can be estimated by *PosePrior* by using 2D keypoints. After *PoseNet*, the score map is sent to the *PosePrior* network to estimate the 3D hand pose.

The *PosePrior* network proposes to train the network to estimate coordinates within a canonical frame rather than to directly estimate absolute 3D coordinates. Additionally, it estimates the transformation from the relative 3D coordinates to the canonical frame during parallel processing, which is a 3D rotation matrix called the viewpoint. Two similar streams are used to estimate viewpoint and canonical coordinates. In the end, two estimates are combined to estimate the 3D coordinates.

The 3D coordinates W are divided into a world coordinate system W^{world} and a camera coordinate system W^{camera}. The camera rotation angle is introduced as $R = \left(R_x, R_y, R_z\right)$ to convert the two coordinate systems:

$$W^{world} = W^{camera} R. \tag{1}$$

3D hand pose estimation can be divided into two tasks. One is to estimate the hand pose in the camera coordinate W^{camera}, and the other is to estimate the angle of view, i.e., the camera rotation angle R. Finally, the two results are fused to get the final coordinates W^{world}. Thus, the 3D coordinate transformation network is divided into two subnetworks with the same architecture, as shown in Figure 3, and the task is divided into an above network and a below network. The above network estimates the coordinates of the 3D hand keypoint W^{camera} in the standard coordinate system. Meanwhile, the rotation angle R of the standard coordinate system relative to the camera coordinate system is estimated in the below network.

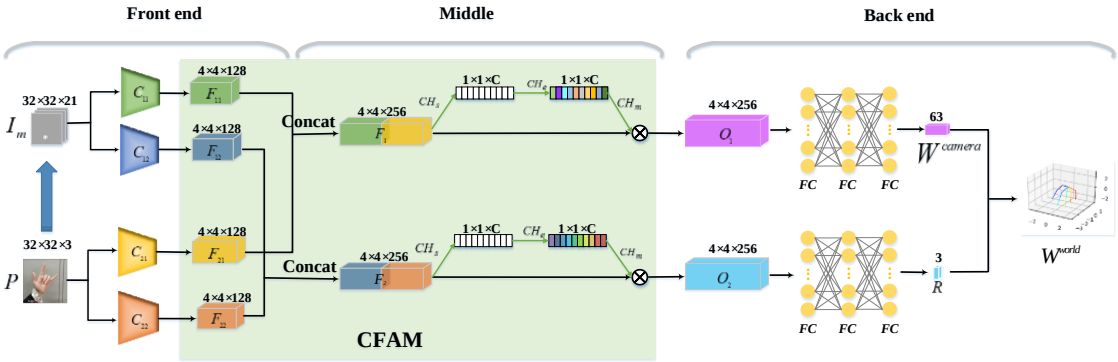
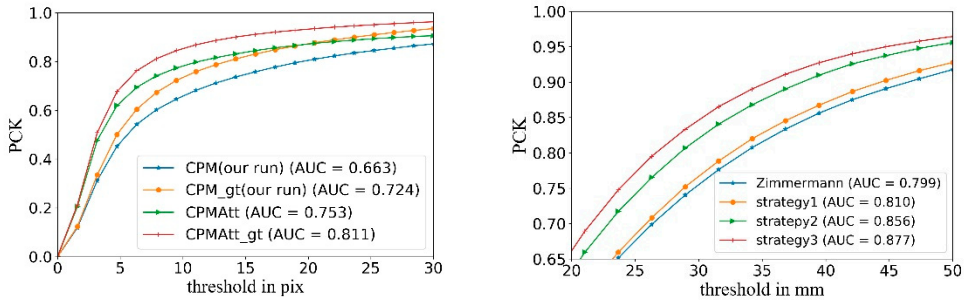


Figure 3. The framework of our channel fusion attention mechanism (CFAM), where C represents six convolution operations, F and O represent data, and FC represents fully connected operations. The framework is divided into three parts: frontend, middle, and backend. Our proposed CFAM is highlighted in the figure. In the frontend, we use a cropped hand region to estimate the 2D score map. The features of the 2D score map and the cropped hand are extracted using a CNN. Then, in the middle, we concatenate the feature maps to obtain F_i and process F_i using a channel attention mechanism. Finally, in the backend, the fully connected layer is used to estimate the camera rotation angle and the 3D hand pose in the camera coordinate system, and the 3D hand pose in the world coordinate system is calculated.

Although the above method performs well, it uses only 2D keypoints, and the RGB information is lost. The RGB image contains texture information and implicit spatial information, which are not included in 2D keypoints. The information in the RGB image is essential for ensuring the accuracy of 3D hand pose estimation. To overcome this drawback, we propose a CFAM (channel fusion attention mechanism; see Section 3.3 for details) which makes full use of the RGB image and takes the spatial information into consideration.

3.3. Channel Fusion Attention Mechanism (CFAM)

In this section, we introduce CFAM, which fuses the information contained in the RGB image with the score map. If we directly merge them, the RGB image influencing factor with less information will overstep the amount of information it contains. As shown in Figure 4b, while adding the RGB image can improve the result to a certain degree, CFAM obtains a better result.



(**a**) (**b**)

Figure 4. (**a**) Area under the curve (AUC) of 2D score map estimation on the rendered hand pose (RHD) dataset. "_gt" in the legend means the method is from the ground truth cropped hand image. Methods without "_gt" are from the crop hand image. (**b**) 3D hand pose estimation on the RHD dataset from the ground truth (GT) data. Zimmermann and strategy 1 utilize GT score map. Strategy 2 and strategy 3 fuse the score map and GT-cropped image. The GT score map is the Gaussian score map from the GT 2D keypoint locations. The GT-cropped image is the cropped hand area. PCK: percentage of correct keypoints.

Although the score map already contains the vital keypoint position information, the RGB image has information that is not included in the score map, such as the implicit spatial information and the local texture information. The texture features are represented by the gray distribution of the surrounding space and the pixel, and they have a rotation invariance and strong resistance to noise. Statistical calculations are required in regions that contain multiple pixels but are not pixel-based features. In pattern matching, this regional feature has greater advantages and can be matched due to local deviations. In addition, the local texture information is repeated to varying degrees, forming the global texture information. The texture feature in *handmask* reflects the nature of the global feature and describes the surface properties of the hand corresponding to the image.

We show the architecture of Figure 3 in Tables 1 and 2. Table 1 shows the structure of the frontend, while Table 2 shows the structures of the middle and backend architectures.

The supplementary information from the RGB image can provide strong guidance for restoring the 3D coordinates for use with the score map. However, if the importance of the RGB image is considered to be the same as that of a score map, then the guiding effect of the RGB image could become too powerful, ultimately affecting the accuracy of the model. To make full use of the RGB image, we introduce a channel attention mechanism to constrain the influence of each input on the final result.

Table 1. The frontend architecture, i.e., networks C_{11}, C_{12}, C_{21}, and C_{22}, is shown below. The networks have the same structure but differ in terms of the number of input channels. The number of channels is 3 when the input is an RGB image, while it is 21 when input is a 2D score map. In addition, the networks have different weights. Conv: convolution, ReLU: rectified linear units.

ID	Name	Kernel	Dimensionality
	Input		$32 \times 32 \times$ Channel
1	Conv. + ReLU	3×3	$32 \times 32 \times 32$
2	Conv. + ReLU	3×3	$16 \times 16 \times 32$
3	Conv. + ReLU	3×3	$16 \times 16 \times 64$
4	Conv. + ReLU	3×3	$8 \times 8 \times 64$
5	Conv. + ReLU	3×3	$8 \times 8 \times 128$
6	Conv. + ReLU	3×3	$4 \times 4 \times 128$

Table 2. The middle and backend network architectures in Figure 3 consist of two independent streams: the upper stream and the lower stream. The two network architectures are similar. One stream calculates the rotation angle of the camera, while the other calculates the coordinate. Therefore, the dimensions of the network output are different. When calculating the rotation angle, the Output in the 'Dimensionality' is 3; when calculating the coordinate, the Output in the 'Dimensionality' is 64. The following table is a stream network structure. Concat: concatenation.

ID	Name	Kernel	Dimensionality
1	F_{1i}		$4 \times 4 \times 128$
2	F_{2i}		$4 \times 4 \times 128$
3	Concat(1,2)		$4 \times 4 \times 256$
4	Average Pooling	4×4	$1 \times 1 \times 256$
5	FC		$1 \times 1 \times 32$
6	FC		$1 \times 1 \times 256$
7	Sigmoid		$1 \times 1 \times 256$
8	Multiply(3,7)		$4 \times 4 \times 256$
9	FC + ReLU		512
10	FC + ReLU		512
11	FC		Output

The framework of our proposed method is provided in Figure 3, where the CFAM consists of two components: the frontend and the middle.

3.3.1. The Frontend: A Fusion Model of the *Handmask* and the Score Maps

We first propose a fusion model of the *handmask* and the score maps to consider the implicit spatial information in the RGB image in our CFAM.

There are four parallel processing streams (called $C_{ij}, i, j = 1, 2$ in Figure 3) in the frontend of the network with almost identical architectures, including six convolutions with rectified linear unit (ReLU) nonlinearities. However, their parameters are not shared. We set the *handmask* I_{mask} as the input of the first two streams C_{1j} and the score maps p_J as inputs of the latter two streams C_{2j}. After being fed into C, the output of the first stream C_{11} and the third stream C_{21} is concatenated to estimate the camera coordinate, while the second stream C_{12} and the fourth stream C_{22} are concatenated to estimate the camera rotation angle. The procedure is illustrated by the following formulas:

$$\mathrm{d}F_{1j} = I_{mask} * C_{1j}, \tag{2}$$

$$F_{2j} = p_J * C_{2j}, \tag{3}$$

$$F_k = F_{1j} \oplus F_{2j}, k = 1, 2, \tag{4}$$

where F_{ij} is the output of the convolution process, $*$ represents the operation on the feature maps that C_{ij} acts on, and $\oplus$ is the concatenation of F_{1j} and F_{2j}. To make full use of I_{mask}, the implicit spatial information and the texture information are utilized in 3D hand pose estimation, which remedies the problem of insufficient context. More spatial and context information is obtained by the network.

3.3.2. The Middle: A Channel Attention Block on the Fused Mode

Before the two feature maps are further processed by fully connected layers, the attention mechanism is added. Attention mechanisms are widely used in various computer vision tasks, such as image classification, segmentation, and object detection. The benefits of such a mechanism have been shown for those tasks. Generally, an attention mechanism biases the allocation of available processing resources toward the most informative components of the input. Hu [21] proposed a squeeze-and-excitation (SE) block to enhance the representational power of basic modules throughout the network. Inspired by the attention model, we use the channel attention block in the latter part of the convolutional layers. The dimensions of features from C are $4 \times 4 \times 256$. The feature maps F_k are first passed through a squeeze operation. Global average pooling is used to aggregate the feature maps across a 4×4 spatial dimension to produce a channel descriptor. A statistic L_k is generated by shrinking F_k through a 4×4 spatial dimension, where the ith ($i \in [1, 256]$) element of L_k is calculated using:

$$l_{ki} = CH_s(f_{ki}) = \frac{1}{4 \times 4} \sum_{m=1}^{4} \sum_{n=1}^{4} f_{ki}(m, n). \tag{5}$$

This descriptor embeds the global distribution of channelwise feature responses, enabling information from the global receptive field of the network to be leveraged by its lower layers. Then, an excitation operation acts on the descriptor. The operation is given by:

$$\mathrm{R}_k = CH_e(L_k, U) = \sigma(g(L_k, U)) = \sigma(U_2 \delta(U_1 L_k)), \tag{6}$$

where δ refers to the ReLU function, $U_1 \in \mathbb{R}^{\frac{256}{h} \times 256}$ and $U_2 \in \mathbb{R}^{256 \times \frac{256}{h}}$. To limit model complexity and aid generalization, we first feed the descriptor L_k to a fully connected layer U_1 around the dimensionality reduction layer with a reduction ratio h, followed by a ReLU. Then, a fully connected layer U_2 is used to increase the dimensionality, followed by sigmoid activation. After the excitation operation, R_k is obtained to describe the weight of each feature map from F_k. Finally, the feature maps F_k from C are reweighted using channelwise multiplication (represented by) between F_k and R_k to generate the output of the channel attention block O_k. The activation is given by:

$$O_k = F_k \cdot R_k. \tag{7}$$

The frontend and the middle constitute our CFAM.

3.3.3. The Backend: Calculate the World Coordinates of the Keypoints

Through the activation above, the network can recalibrate features and learn to use global information to selectively emphasize informative features and suppress less useful ones. The output feature maps from channel attention block O_k are concatenated with the information to determine whether the hand is a left or right hand and is processed further using two fully connected layers. Then, the two parallel streams are fed directly into fully connected layers to estimate the camera coordinate W^{camera} and the camera rotation angle R. Both estimations are combined with an estimation of the world coordinate W^{world}. The final process is as follows, where FC_k is the operation of the full connection:

$$W^{camera} = O_1 * FC_1, \tag{8}$$

$$R = O_2 * FC_2, \tag{9}$$

$$W^{world} = W^{camera}R. \tag{10}$$

4. Experiments

We conducted experiments to verify the proposed model. Our method was implemented in TensorFlow [22]. All experiments were conducted on a Linux computer with one NVIDIA 1080Ti GPU (with 11 GB memory). The batch size is the number of samples selected for one training epoch, and we set our batch size to 8. We trained the model with the Adam optimizer until the loss did not decrease. The learning rates used were 1×10^{-5}, 1×10^{-6}, and 1×10^{-7}. The learning rates changed after 30,000 and 60,000 steps. The improved score map detection model and the 3D hand pose estimation model were tested. We highlight the results of the best method in each experiment in bold. In the table, some wrist prediction errors were 0 because we kept only two decimal places and thus errors that were less than 0.01 are rounded to zero; such errors indicate wrist predictions that were accurate.

4.1. Dataset

Our proposed method is based on a single RGB image whose labels are required for supervision. Traditional depth-based hand pose estimation datasets, such as MSRA [5] and the NYU Hand Pose Dataset by Tompson et al. [4], are not suitable for our method. Thus, we chose two open datasets: a real-world dataset from the Stereo Hand Pose Tracking Benchmark (STB) [10] and a generated dataset called the Rendered Hand Pose (RHD) [16] dataset, both of which contain RGB images of human hands and the position coordinates of 3D keypoints. Twenty-one hand keypoints are included in both datasets, including the palm center (not the wrist or hand center) and four keypoints per finger. Each sample in both datasets includes an RGB image, a handmask image, the rotation of the camera, and the 2D and 3D coordinates of each keypoint.

The RHD dataset is a generated dataset that consists of 39 different actions performed by 20 different persons. The dataset contains 41,258 training samples and 2728 testing samples. An image pixel in the dataset is 320×320. The STB dataset is a real image dataset collected from different cameras. It contains 36,000 images and can be divided into six scenarios. Each scene contains two RGB images and a depth image of the same action in different positions. There are 30,000 training images and 6000 testing samples in the 640×480 dataset. The two datasets include images of the hands of different people.

4.2. Assessment Criteria

The error and the area under the curve (AUC) were used to evaluate the experimental results. The error of each keypoint was calculated as follows:

$$E_J = \left| gt_J - pre_J \right|, \tag{11}$$

where gt_J is the coordinate of the ground truth for keypoint J and pre_J is the estimated coordinate of keypoint J. The AUC curve was based on the percentage of correct keypoints (PCK):

$$AUC_J = \int PCK_J. \tag{12}$$

In addition to evaluating the average error and the average AUC, we used 21 keypoints to calculate the error and accuracy for each finger. For convenience, in the following subsections in this section, we use "wrist" to represent the palm and "thumb," "index," "middle," "ring," and "little" to represent each finger. We use "GT" to represent the ground truth in the following experiments.

4.3. 2D Score Map Detection

3D hand pose estimation largely depends on 2D pose estimation, and it can be effectively improved by enhancing 2D score map estimation. Motived by Zimmermann et al. [16], who used a CPM to

locate 2D keypoints, as Figure 4a and Table 3 show, we improved the CPM and enhanced the location accuracy. Table 4 shows the result for each finger. The original RGB images and the RGB images of the cropped hand region are provided in the dataset. We resized the original RGB images to 240 × 320 and the images of the segmented hand region to 256 × 256 during training. The channel attention mechanism was added to improve the score map estimation accuracy. For a channel attention block to obtain a better feature map, we added this block to the CPM. CPMAtt represents the method after adding a channel attention mechanism to the CPM. CPMAtt_gt and CPM_gt are the CPMs that were used on the ground truth hand cropped images. CPMAtt and CPM are the CPMs that were used on the original image, which needed to be cropped by *HandSegNet*.

Table 3. The mean error and AUC of the 2D keypoints estimation results on the RHD dataset. By adding the channel attention mechanism, CPMAtt was superior to a convolutional pose machine (CPM) [17] in terms of the AUC and error. Even in the *HandSegNet* cropped picture, our experimental AUC was better than that of the CPM method on the GT-cropped picture. Regardless of whether the image being tested was segmented by the GT or *HandSegNet*, our model was an improvement. The AUC was increased by nearly 9 percentage points, and the mean error was reduced by nearly 3 pixels. The best results are highlighted in bold.

Evaluation Index	CPMAtt	CPMAtt_gt	CPM (Our Run)	CPM_gt (Our Run)
AUC (0–30 pix)	0.753	**0.811**	0.663	0.724
Median Error (pix)	3.81	**3.25**	5.83	5
Mean Error (pix)	14.49	**6.26**	17.04	9.14

Table 4. 2D keypoint estimation results on the RHD dataset. Our method achieved the best results on different fingers. The best results are highlighted in bold.

Evaluation Index	Method	Wrist	Thumb	Index	Middle	Ring	Little
Mean Error (pix)	CPMAtt	24.8	13.47	13.66	13.23	14.12	15.39
	CPMAtt_gt	**4.34**	**8.53**	**7.4**	**4.5**	**5.04**	**6.32**
	CPM (our run)	24.62	17.23	16.29	14.3	16.04	19.45
	CPM_gt (our run)	6.11	12.2	9.59	6.1	7.65	10.89
Media Error (pix)	CPMAtt	16.12	**3.4**	**2.87**	3.13	3.25	3.31
	CPMAtt_gt	**3.36**	3.66	3.31	**3.06**	**3.14**	**3.05**
	CPM (our run)	15.97	5.84	4.67	4.59	5.59	5.93
	CPM_gt (our run)	4.47	6.15	4.6	4.36	4.99	5.04
AUC (0–50 pix)	CPMAtt	0.44	0.75	0.77	0.8	0.78	0.75
	CPMAtt_gt	**0.86**	**0.76**	**0.79**	**0.85**	**0.83**	**0.8**
	CPM (our run)	0.44	0.64	0.67	0.74	0.7	0.62
	CPM_gt (our run)	0.8	0.65	0.71	0.8	0.75	0.68

4.4. 3D Hand Pose Estimation with CFAM

To better estimate the 3D hand poses based on the previous pose, we propose the CFAM module, which includes the attention mechanism and the fusion of RGB images and 2D heat map information. To show that every step of our network design is effective, we used three strategies for comparison:

Strategy 1: adding the channel attention mechanism.

Strategy 2: adding the fusion of RGB images and 2D score maps without adding the channel attention mechanism.

Strategy 3: adding the full CFAM.

Table 5 and Figure 4b show the effect of our approach on the RHD dataset, while Table 6 shows the effect on each finger. The channel attention mechanism can have certain auxiliary effects on feature acquisition. Therefore, the result of strategy 1 was slightly better than that of Zimmermann's method, but the effect was not significantly improved, and the AUC was increased by approximately

one percent. Strategy 2 added RGB image-assisted training, and the improvement was significant. Our CFAM (strategy 3) combined the features of RGB images and the 2D score map, reducing the error from strategy 2 by more than 1 mm and reducing the error from Zimmermann's framework by more than 4 mm.

Table 5. 3D hand pose estimation on the RHD dataset from a GT 2D score map and a GT-cropped RGB image. The best results are highlighted in bold.

Evaluation Index	Zimmermann	Strategy 1	Strategy 2	Strategy 3
AUC (0–50 mm)	0.585	0.591	0.629	**0.648**
Median Error (mm)	18.84	18.6	16.65	**15.85**
Mean Error (mm)	22.43	21.88	19.4	**18.2**

Table 6. 3D hand pose estimation on the RHD dataset from the GT 2D score map and the GT-cropped RGB image for each finger. The best results are highlighted in bold.

Evaluation Index	Methods	Wrist	Thumb	Index	Middle	Ring	Little
Mean Error (mm)	Zimmermann [16]	0.28	24.98	27.11	22.27	20.98	22.36
	Strategy 1	0.8	24.55	25.49	21.44	20.64	22.54
	Strategy 2	1.39	20.09	22.46	20.63	18.56	19.77
	Strategy 3	**0.21**	**18.88**	**20.95**	**18.83**	**17.35**	**19.47**
Media Error (mm)	Zimmermann [16]	0.27	20.67	22.98	18.76	17.68	18.76
	Strategy 1	0.77	20.79	21.51	18.26	17.53	19.37
	Strategy 2	1.32	17.3	18.91	17.8	15.87	17.21
	Strategy 3	**0.2**	**16.56**	**17.81**	**16.32**	**15**	**17.47**
AUC (0–50 mm)	Zimmermann [16]	0.97	0.55	0.52	0.58	0.6	0.58
	Strategy 1	0.97	0.55	0.53	0.6	0.61	0.57
	Strategy 2	0.97	0.62	0.58	0.61	0.64	0.62
	Strategy 3	0.97	**0.64**	**0.6**	**0.64**	**0.66**	**0.62**

Strategy 3 was better than strategy 2, and strategy 1 was better than Zimmermann's framework. The main reason for the improvement was the addition of channel attention, but the improvement of strategy 3 was greater than that of strategy 1, and the accuracy was also improved. It was more difficult to improve the accuracy when the accuracy was already high, indicating that the attention mechanism in our CFAM was effective, and it not only played a role in channel attention, but also blended the characteristics of the RGB images and 2D score maps; only then could the results be greatly improved. Among the different methods, strategy 3 (CFAM) maximized the AUC. Since the CFAM method had the highest accuracy, we tested it on the STB dataset.

Table 7 and the left graph of Figure 5 show the results of our CFAM on the STB dataset from the GT 2D score maps, and our CFAM outperformed Zimmermann's in terms of both the error and AUC. Table 8 shows the result for each finger; for most hand keypoints, our CFAM outperformed Zimmermann's framework.

Table 7. 3D hand pose estimation on the STB dataset from the GT 2D score map and GT-cropped RGB image. The best results are highlighted in bold.

STB	Zimmermann [16]	CFAM
AUC (0–50 mm)	0.83	**0.837**
Median Error (mm)	7.3	**7.06**
Mean Error (mm)	8.47	**8.07**

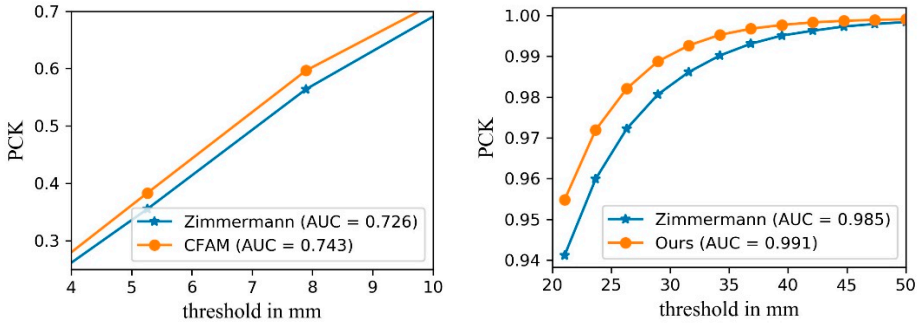


Figure 5. AUC of 3D hand pose estimation on STB dataset from the GT 2D score maps (**left**) and the GT-cropped RGB image (**right**).

Table 8. 3D hand pose estimation on the STB dataset from the GT 2D score map and the GT-cropped RGB image for each finger. The best results are highlighted in bold.

Evaluation Index	Methods	Wrist	Thumb	Index	Middle	Ring	Little
Mean Error (mm)	Zimmermann [16]	0	**8.74**	10.04	8.93	8.35	8.42
	CFAM	0	9.02	**9.1**	**8.68**	**7.76**	**7.79**
Media Error (mm)	Zimmermann [16]	0	**7.72**	8.79	7.57	6.99	7.24
	CFAM	0	7.85	**8.45**	**7.76**	**6.71**	**6.31**
AUC (0–50 mm)	Zimmermann [16]	0.97	**0.83**	0.8	0.82	0.83	0.83
	CFAM	0.97	0.82	**0.82**	**0.83**	**0.84**	**0.84**

4.5. Estimating 3D Hand Poses from a Single RGB Image

When estimating the 3D hand pose based on the GT 2D score map, it was found that our method was superior to Zimmermann's framework. To prove that our method was feasible throughout the process, we estimated 3D keypoints from a single RGB image, and we verified the results via the original RGB image that needed to be cropped by *HandSegNet*. The GT-cropped RGB image is the ground truth cropped RGB image, and the RGB image is the image without cropping, which needs *HandSegNet* to crop the hand image. The method called "Ours" is the method that used CPMAtt to estimate the 2D keypoints and CFAM to estimate the 3D keypoints. Due to the lack of depth information, estimating a 3D hand pose from a single RGB image is challenging. The hand side information was used for the processing step. The picture will slip if the hand side changes.

The right graph of Figure 5 shows the result using the STB dataset from GT-cropped images, while the left graph of Figure 6 shows the result using the RHD dataset. Tables 9–12 show the results using both datasets and the details for each finger. Our method obtained a better result on the GT-cropped RGB image. We also tested it on RGB images without GT cropping. This kind of image was cropped by *HandSegNet* first, which may have caused a bigger error because of the error in the segmentation strategy. Under all conditions, our method attained better results; therefore, the method was robust and useful for solving this kind of task.

Table 9. Estimating the 3D hand pose from the GT-cropped RGB image. The best results are highlighted in bold.

Datasets	RHD		STB	
Method	**Zimmermann [16]**	**Ours**	**Zimmermann [16]**	**Ours**
AUC (0–50 mm)	0.48	**0.561**	0.823	**0.824**
Median Error (mm)	24.47	**19.61**	**7.58**	7.78
Mean Error (mm)	30.36	**24.6**	8.8	**8.75**

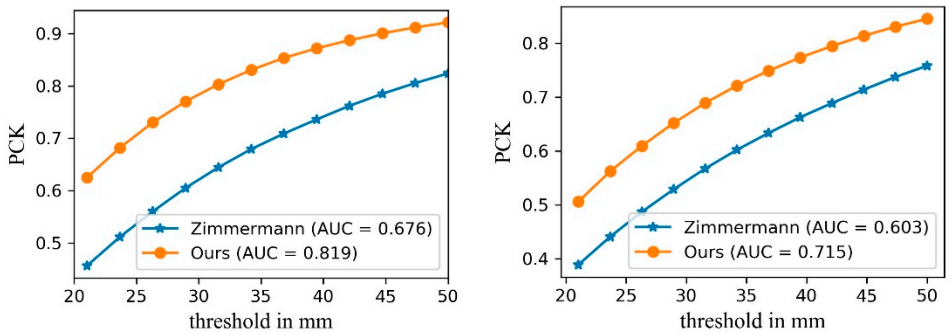


Figure 6. AUC of the 3D hand pose estimation on the RHD dataset from the GT-cropped RGB images (**left**) and the RGB images (**right**).

Table 10. Mean error of the 3D hand pose estimation from the GT-cropped RGB image (mm). The best results are highlighted in bold.

Datasets	Method	Wrist	Thumb	Index	Middle	Ring	Little
RHD	Zimmermann [16]	**0**	24.57	28.58	25.5	24.46	25.33
	Ours	0.13	**18.46**	**20.01**	**18.1**	**16.83**	**18.11**
STB	Zimmermann [16]	0	9.41	10.06	9.62	8.61	8.51
	Ours	0	**9.11**	**9.04**	**10.04**	**8.97**	**8.75**

Table 11. Median error of the 3D hand pose estimation from the GT-cropped RGB image (mm). The best results are highlighted in bold.

Datasets	Methods	Wrist	Thumb	Index	Middle	Ring	Little
RHD	Zimmermann [16]	**0**	33.81	35.55	30.33	28.5	31.2
	Ours	0.14	**24.2**	**25.7**	**22.35**	**20.65**	**22.48**
STB	Zimmermann [16]	0	8.45	8.83	**8.16**	**7.19**	**7.17**
	Ours	0	**8.24**	**8.29**	8.79	7.89	7.62

Table 12. AUC of the 3D hand pose estimation from the GT-cropped RGB image (0–50 mm). The best results are highlighted in bold.

Datasets	Methods	Wrist	Thumb	Index	Middle	Ring	Little
RHD	Zimmermann [16]	0.97	0.47	0.41	0.46	0.48	0.45
	Ours	0.97	**0.58**	**0.54**	**0.59**	**0.61**	**0.58**
STB	Zimmermann [16]	0.97	0.81	0.8	**0.81**	**0.83**	**0.83**
	Ours	0.97	**0.82**	**0.82**	0.8	0.82	0.82

Images cropped using *HandSegNet* had a certain degree of error; we experimented with the images cropped by *HandSegNet* to prove that our CFAM was not sensitive to these errors. As the right graph of Figure 6 and Tables 13 and 14 show, our method was still better than Zimmerman's framework, which proved that our CFAM can be used in end-to-end 3D hand pose estimation, and our method was superior to the original method in all modules. Many hand pose estimation methods are based on segmented hand images, indicating that they are sensitive to errors in the segmentation process. Our method can more accurately estimate hand poses in the presence of segmentation errors than other methods, and it can be used in tracking and undivided hand images.

Table 13. 3D hand pose estimation on the RHD dataset from the RGB image for each finger. The best results are highlighted in bold.

	Methods	Wrist	Thumb	Index	Middle	Ring	Little
Mean Error (mm)	Zimmermann [16]	**0**	39.8	42.51	35.98	33.04	35.59
	Ours	0.21	**32.83**	**34.43**	**29.81**	**26.85**	**29.02**
Media Error (mm)	Zimmermann [16]	**0**	29.46	34.66	29.8	27.98	28.7
	Ours	0.2	**23.51**	**25.35**	**23.08**	**21.05**	**22.68**
AUC (0–50 mm)	Zimmermann [16]	0.97	0.4	0.35	0.41	0.43	0.4
	Ours	0.97	**0.48**	**0.45**	**0.49**	**0.53**	**0.49**

Table 14. 3D hand pose estimation on the RHD dataset from the RGB image. The best results are highlighted in bold.

Methods	Zimmermann [16]	Ours
AUC (0–50 mm)	0.424	**0.512**
Median Error (mm)	28.69	**22.04**
Mean Error (mm)	35.61	**29.14**

Although our method performed better on most fingers, in some experiments, Zimmermann's method obtained a better result for the thumb. Our method focuses more on the global optimum, while Zimmermann's method pays more attention to the accuracy of single fingers. In general, our method worked best, but Zimmermann's method performed better on the thumb.

4.6. Comparison with the State-of-the-Art Methods

To prove the superiority of our approach, we compared it with the state-of-the-art methods. Since many methods are performed on segmented hand images and most methods are based on the STB dataset, we also compared them on the segmented hand images of the STB dataset. Table 15 shows that, among all methods, our method achieved the best AUC value. Dibra's method performs weak supervised learning by reducing keypoints into a depth image and can learn some implicit depth information through weak supervised learning, but the learned depth information is still less useful than that implied in the original RGB image. Zimmermann, Panteleris, and Supr use only 2D information to restore the 3D positions and lose some information from the RGB image. Muller's method restores the occluded hand areas through a GAN, but due to the error of the picture restored by the GAN, the error is magnified during intermediate error transmission. Since we have used the proposed CFAM module to take the information in the 2D score map and RGB image of the hand into account, our method achieved the best result.

Table 15. Comparison of state-of-the-art methods in terms of AUC for the STB dataset. The best result is highlighted in bold.

Methods	AUC (20–50 mm)
Dibra [19] (CVPR2018 workshop)	0.923
Panteleris [23] (2018 WACV)	0.941
Mueller [20] (CVPR2018)	0.965
Spur [18] (2018 CVPR)	0.983
Zimmermann [16] (2017 ICCV)	0.985
Ours	**0.991**

5. Conclusions

We proposed CFAM for estimating the 3D hand pose from a single RGB image. As far as we know, we are the first to use this attention mechanism as a block in the application of 3D hand pose estimation, and the accuracy was clearly an improvement over other commonly used methods. We reasonably used the missing information in the color image by combining a 2D score map and an RGB image. In addition, we used an attention mechanism as a weighting scheme to clarify the guiding effect of the two features of color images and the 2D joint points on the 3D joint point estimation. We validated our method on the RHD and STB datasets. Multiple contrast experiments on public datasets demonstrated that our proposed method could achieve state-of-the-art accuracy, and an ablation experiment showed that an RGB image and a 2D score map could be combined to improve the result of the 3D hand estimation, which means that the information in the RGB image was also very important. In future research, we will improve the efficiency of the program and simplify the model. Our method can be used in virtual reality equipment to accurately locate joint points.

Author Contributions: Conceptualization, X.W.; data curation, X.W. and L.K.; formal analysis, L.K. and D.L.; funding acquisition, Y.G.; methodology, X.W., Y.G., and Y.W.; resources, Y.W.; software, X.W.; validation, Y.W.; writing—original draft, X.W.; writing—review and editing, J.J. All authors have read and agreed to the published version of the manuscript.

Funding: This work is partially supported by the National Natural Science Foundation of China (NSFC) under grant nos. 61806218, 61873274, and 61806215.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Markus, O.; Paul, W.; Vincent, L. Hands Deep in Deep Learning for Hand Pose Estimation. *Comput. Vis. Winter Workshop* **2015**, *2015*, 21–30.
2. Ayan, S.; Chiho, C.; Karthik, R. Deep Hand Robust Hand Pose Estimation by Computer a Matrix Imputed with Deep Features. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016; pp. 4150–4158.
3. Markus, O.; Vincent, L. DeepPrior++: Improving Fast and Accurate 3D Hand Pose Estimation. In Proceedings of the IEEE Internatio-nal Conference on Computer Vision Workshop (ICCVW), Venice, Italy, 22–29 October 2017; pp. 585–594.
4. Tompson, J.; Stein, M.; Lecun, Y.; Perlin, K. Real-time continuous pose recovery of human hands using convolutional networks. TOG. *ACM Trans. Graph.* **2014**, *33*, 169. [CrossRef]
5. Sun, X.; Wei, Y.; Liang, S.; Tang, X.; Sun, J. Cascaded hand pose regression. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015.
6. Edgar, S.-S.; Arnau, R.; Guillem, A.; Carme, T.; Francesc, M.N. Single image 3D human pose estimation from noisy observations. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Rhode, Island, 18–20 July 2012; pp. 2673–2680.
7. Athitsos, V.; Sclaroff, S. Estimating 3d hand pose from a cluttered image. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Madison, WI, USA, 18–20 June 2003; Volume 2, p. II-432.
8. Zhou, X.; Wan, Q.; Zhang, W.; Xue, X.; Wei, Y. Model-based deep hand pose estimation. In Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence IJCAI, New York, NK, USA, 9–15 July 2016.
9. Garcia-Hernando, G.; Yuan, S.; Baek, S.; Kim, T.-K. First-person hand action benchmark with RGB-D videos and 3D hand pose annotations. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 409–419.
10. Zhang, J.; Jiao, J.; Chen, M.; Qu, L.; Xu, X.; Yang, Q. 3d hand pose tracking and estimation using stereo matching. *arXiv* **2016**, arXiv:1610.07214.
11. Chengde, W.; Thomas, P.; Luc, V.G.; Angela, Y. ETH Zurich Dense 3D Regression for Hand Pose Estimation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016.

12. Aisha, U.K.; Ali, B. Analysis of Hand Segmentation in the Wild. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018.
13. Zhu, J.-Y.; Park, T.; Isola, P.; Efros, A.A. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Venice, Italy, 22–29 October 2017.
14. Seungryul, B.; Kwang, I.K.; Kim, T.-K. Augmented Skeleton Space Transfer for Depth-based Hand Pose Estimation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018.
15. Wan, C.; Probst, T.; Gool, L.V.; Yao, A. Crossing Nets: Combining GANs and VAEs with a Shared Latent Space for Hand Pose Estimation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017.
16. Zimmermann, C.; Brox, T. Learning to estimate 3d hand pose from single rgb images. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 4903–4911.
17. Wei, S.-E.; Ramakrishna, V.; Kanade, T.; Sheikh, Y. Convolutional pose machines. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 4724–4732.
18. Spur, A.; Song, J.; Park, S.; Hilliges, O. Cross-modal Deep Variational Hand Pose Estimation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018.
19. Dibra, E.; Melchior, S.; Balkis, A.; Wolf, T.; Öztireli, C.; Gross, M. Monocular rgb hand pose inference from unsupervised refinable nets. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, Salt Lake City, UT, USA, 18–22 June 2018; pp. 1075–1085.
20. Mueller, F.; Bernard, F.; Sotnychenko, O.; Mehta, D.; Sridhar, S.; Casas, D.; Theobalt, C. Ganerated hands for real-time 3d hand tracking from monocular rgb. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 22 May 2018; pp. 49–59.
21. Hu, J.; Shen, L.; Sun, G. Squeeze-and-excitation networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7132–7141.
22. Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G.S.; Davis, A.; Dean, J.; Devin, M.; et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv* **2016**, arXiv:1603.04467.
23. Panteleris, P.; Oikonomidis, I.; Argyros, A. Using a single rgb frame for real time 3d hand pose estimation in the wild. In Proceedings of the 2018 IEEE Winter Conference on Applications of Computer Vision (WACV), Lake Tahoe, NV, USA, 12–15 March 2018; pp. 436–445.