Journal of Computers Vol. 29 No. 2, 2018, pp. 55-64
doi:10.3966/199115992018042902006

Image Face Calibration Positioning Based on 3000FPS Algorithm

Xi-bao Wu[1*], Chen Jin[1], Xiang-feng Chen[1], Rentian He[1], Ke Lv[2]

[1] Beijing Information Science & Technology University, Beijing, 100192, China
wuxibao@bistu.edu

[2] Chinese Academy of Sciences University, Beijing, 100192, China

Received 24 July 2017; Revised 19 September 2017; Accepted 19 October 2017

Abstract. In the face recognition tasks, face pose can have influence on the results of recognition. To improve the accuracy of recognition, we introduced 3000FPS algorithm for face image calibration positioning. Random forests is used to establish each key point, and the output of every tree in random forests, binary feature, is exploited to compose local binary feature, which is used to train a softmax classifier. The experiment show that the recognition accuracy of the algorithm can reach 93.33% on the Helen face dataset, and 94.14% on the LFPW. Meanwhile, multi-face can be aligned and located even though the input image is partially obscured.

Keywords: 3000FPS, face calibration positioning, global linear regression, random forest trees

1 Introduction

In the face recognition process [1], the most important key is calibration positioning of the face. Facial calibration positioning usually aim to locate facial features, such as eyes, nose, mouth and chin, which is essential steps and can contribute to the result of face recognition. With the rapid development of the network, the number of photos produced by people grew increasingly. Therefore, face image alignment is essential for the need to improve efficient and accurate results.

Currently, face alignment consist of three method including local-basedmethods, holistic-basedmethods and mixture ways. Early, ASM [2] is proposed to detect face feature point by Cootes, who made ASM further improvement, introducing the AAM [3] algorithm. A joint shape and texture appearance model is used in [4], presenting the more robust model than AAM, which depends on the image reconstruction error to update the model parameters. To address the limitation of the connection to optimization theory in learning a mapping from image features to problem parameters, cascaded regression is proposed in [5] to solve facial deformable model fitting model. Due to the advantage of deep learning, convolutional Neural Network (CNN) is firstly exploited in the face feature point location by Sun [6]. The prior works concentrated on sparse alignment due to the limited number of facial landmark points, [7] provided a dense 3-D alignment for multi-pose face images to address the drawback of training CNN with multiple datasets. The paper [8] presented Deep Alignment Network (DAN) including multiple stages, which improved the locations of the facial landmarks estimated by the previous stage.

In some extend, the previous work has enhanced the effectiveness and robustness of facial feature points alignment and location, however, it ignored the multi-face alignment and the facial key points detection under the condition of partial shelter. Motivated by the 3000FPS algorithm proposed in [9], we introduced a calibration and positioning of human face based on 3000FPS algorithm, which can increase detection accuracy. The rest of paper is organized as follows: the proposed approach is described in details in Section 2. Experimental settings and results are demonstration and discussed respectively in Section 3. Section 4 draws the conclusion of the work.

* Corresponding Author

2 Face Calibration Based on 3000FPS Algorithm

Image face calibration positioning based on a 3000FPS algorithm was introduced in this paper. First of all, the face image preprocessing, the key points in the picture make the difference. Then, use each key point establish random forests, binary feature from each tree output in the random forest trees, to form a local binary feature by the cascade, and use this feature through the global linear regression training a regression device. Finally realize facial face calibration and positioning.

2.1 3000FPS Principle

The combination of random forest and global linear regression is the core method of 3000FPS. Compared with the convolution neural network used in depth learning, the method adopted by 3000FPS is the traditional method of machine learning. As shown in 1.

$$S^t = S^{t-1} + R^t(I, S^{t-1}) \tag{1}$$

Among them, the formula 1 is the absolute shape of the image, generating regression device, said the image, said the number of layers cascaded, we predict the shape of the use of multi-level cascade is used to predict. The regression device predicts a deformation based on the position information of the image and shape, and then superimposes the previously predicted deformation with the previous shape to obtain a new shape.

3000FPS predicted by random forest, whose output combination becomes a feature, and the use of combined features to do the forecast. In addition to using the structure of random forest to predict, the 3000FPS also for each key point to give a random forest to predict, and all the key points corresponding to the local characteristics of the random forest output connected to each other, known as the local binary features LBF, and then use this local binary feature to perform global regression, used to predict the deformation.

2.2 Regression Device

Regression device training and the prediction process is shown in Fig. 1.

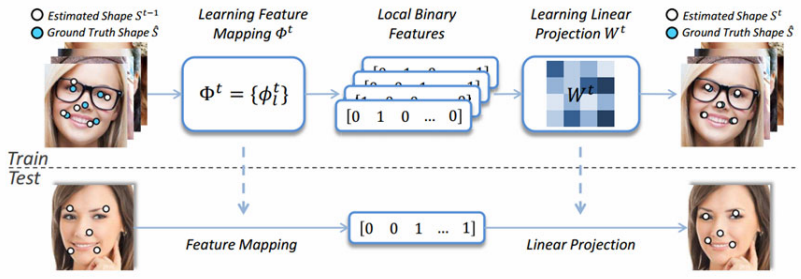

Fig. 1. Regression device training and prediction process

Where Φ_l^t represents that the random forest of the l key points in the t level. The output of Φ_l^t and Φ_l^t is the LBF characteristic. And then use the LBF feature to perform global linear regression or to predict deformation.

Fig. 2 describes that regression device R^t generated the LBF feature, for each map of each landmark of each tree will eventually output a value, as shown in Fig. 2(b), the first tree traverses the leftmost child node, so it is recorded as [1, 0, 0, 0]. For one of the trees, if the leaf node is accessed, the node is denoted as 1, or it is recorded as 0, and then a landmark has a forest that is a tree, then all the results together is $\phi_l^t = [1,0,0,0,0,1,0,0,0,0,1,0,...]$, the real Local Binary Features is all the landmark of these features are linked together, as shown in 2.

$$\phi^t = [\phi_1^t, \phi_2^t, ..., \phi_L^t] \tag{2}$$

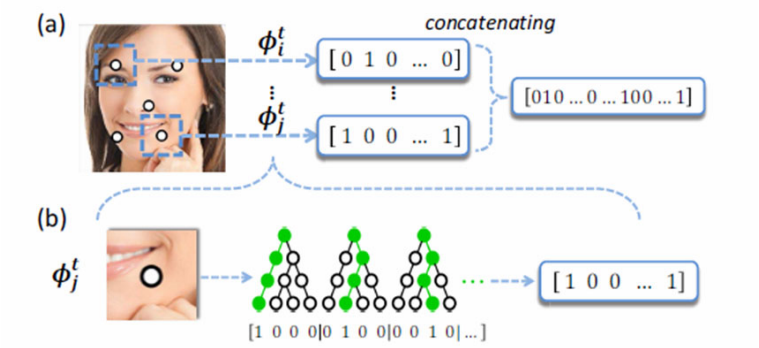

Fig. 2. Regression device training and prediction process

So we can see that this is a very sparse vector, the middle of the number of 1 is the total number of trees in the landmark, the rest of the 0. And then is to train a global linear regression.

2.3 Pixel Difference Features

Each of the key points in the image corresponds to a random forest, but each random forest is made up of several independent trees. The feature used by the tree is called the Pixel Difference Features, which is the pixel difference between two points in the face region. ESR, RCPR, 3000FPS. These three algorithms use different strategies.

ESR algorithm uses the strategy is in the face of two key points near the random out of two new points, do the difference between the two new points as Pixel Difference Features characteristics.

The strategy used by the RCPR algorithm is to select the midpoint of the two key points and a random offset in the face key to generate the feature point, using the difference between the two such feature points as the Pixel Difference Features.

In the 3000FPS, because the random forest is aimed at a single key point, all the feature points used by the tree are not associated with other key points, and only two feature points are randomly generated in the vicinity of the current key point. The pixel difference between the two feature points is the Pixel Difference Features.

With the increasing depth of the cascade, the range of random points will gradually become smaller, in order to obtain more accurate local features, as shown in Fig. 3.

Fig. 3. Different cascades produce different results

For a particular landmark, some of the points around it are geometrically invariant, for example we have a landmark that is the left eye of the left eye, with a certain position above it, such as the eyebrow on the nearby Black color, which for everyone is the same, of course, in the light or other factors under the influence of the color here is still very different, so it is necessary Pixel Difference Features characteristics.

Pixel Difference Features In fact, it is very simple, that is, the value of the two points minus a difference, and this difference in a threshold range will have a large degree of floating, but also can remove the impact of light or other factors, which is shown in 3.

$$feature = I(x + \Delta x_1, y + \Delta y_1) - I(x + \Delta x_2, y + \Delta y_2) \quad \textbf{(3)}$$

2.4 The Establishment of Random Forest

Random Forest is composed of many trees, compared to a single tree can prevent the model over fitting. Random Forest can be used for regression and classification.

When training the tree, the input is $X = \{I, S\}$, the goal of the forecast is $Y = \Delta S$. In the actual training tree, the tree in all the nodes of the training process is the same. When we train on a node, we first select a Pixel Difference Features that has been generated in advance. The selected feature maps all sample points into a set of real numbers, and then randomly assigns a sample point to the Left and right sub-tree, and the purpose is to want left-child and right-child in the sample point of y has the same pattern. Feature selection can be described by the following formulations 4.

$$\begin{gathered} f = \arg\max_{f \in F} \Delta \\ Delta = S(y \mid y \in Root) - [S(y \mid y \in Left) + S(y \mid y \in Right)] \\ y \in \begin{cases} Left, f(x) < \delta \\ Right, f(x) < \delta \end{cases} \end{gathered} \tag{4}$$

The above F formulation shows the Pixel Difference Features feature, f represents the selected feature function, δ represents the randomly generated threshold, S is used to describe the similarity between the sample points or the entropy of the sample set.

For each node, the training data (X, Y) will be divided into two parts (X_1, Y_1) and (X_2, Y_2). We expect the sample data in the left and right sub-trees of each tree to have the same pattern, so we expect the variance to be the smallest when we select the feature function f.

Each node on the tree is trained in this way, and the training of each tree is independent of each other, all the training process is the same, so that a single key point of the random forest training is completed.

2.5 The Establishment of Random Forest

In the usual way, we can store the predicted deformation on the leaf nodes of the tree, and the average or weighted average of the predicted output of each tree in the random forest is tested. However, 3000FPS does not take this method, it will be the output of the tree expressed as a binary feature, the binary characteristics of all the tree before and after the connection to form a local binary features, that is, LBF features. Use this feature to do a linear regression, the deformation as a prediction target, training a linear regression to predict. Linear regression can be expressed by the following 5.

$$W_t = \arg\min_{W_t} \| \Delta S - W_t \cdot lbf \|_2 + \lambda \| W_t \|_2 \tag{5}$$

Where ΔS indicates the deformation target, lbf is the feature, W_t represents linear regression parameters, λ is used to suppress the parameters of the model to prevent the phenomenon of over-fitting. Use the equation 6 to forecast.

$$\Delta S = W_t \cdot lbf \tag{6}$$

Each stage of the multi-level cascade linear regression method can be split into two parts according to the above formula, using the random forest to extract the LBF, and then use the LBF to do the global linear regression prediction shape increment ΔS.

3 Experimental Analysis of Image Face Calibration

3.1 Image Preprocessing

To improve the accuracy and accuracy of calibration, you need to tailor the image and scale the cropped image. Image trimming mainly obtains the area of the face part we are interested in in the whole image, cuts the image, reduces the size of the image, and greatly reduces the use of memory, greatly increasing the speed of image recognition. The crop method of picture shown in Fig. 4.

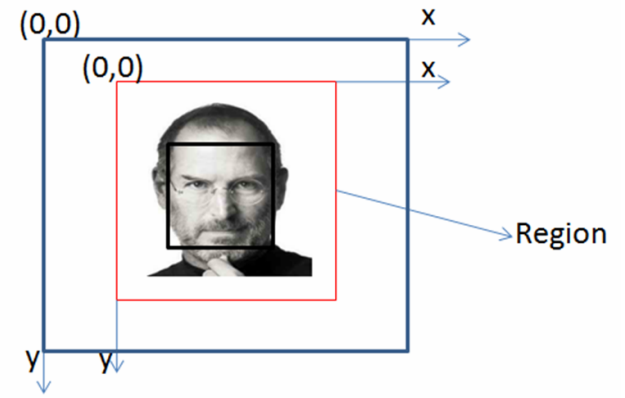

Fig. 4. Image cropping diagram

The method of image cropping is from the coordinate system point of view, in fact, from the original coordinate system to the expanded bounding box based on the coordinate system. Changes in the coordinate system can be expressed by the equation 7.

$$\widetilde{shapes\_gt} = shapes\_gt - [region(1), region(2)] \tag{7}$$

The formula is just the original shape after the cut of the coordinates, in order to further save memory, we also need to add the zoom step, as shown in the formula 8.

$$\widetilde{shapes\_gt} = \widetilde{shapes\_gt} \cdot Ratio \tag{8}$$

3.2 Preparation of the Initial Data

Once the image is preprocessed, we need to prepare the initial data to facilitate the training and testing of the model.

First, we need to get mean_shape, mean_shape is the average of the ground_truth points of all the images. The method used to get mean_shape is to add all the feature points directly to the average, but the mean_shape obtained is rush and inaccurate, Because each picture is a variety of faces between the various, by the light, posture and other aspects of the impact is different, so if we directly get the average will certainly have a lot of error, so we should A relatively uniform frame below is the mean_shape.

The use of a unified framework can be represented by the following formulation 9.

$$\frac{\{shape\_gt - [Region(1), Region(2)]\}}{[Region(3), Region(4)]} \Rightarrow [0,1] \cdot [0,1] \tag{9}$$

3000FPS core idea is as shown in formulation 10.

$$\Delta S^t = W^t \Phi^t (I, S^{t-1}) \tag{10}$$

Where $\Delta S = S^{gt} - S^t$ is the residual difference between the images after t stage, and W^t is linear regression matrix, $\Phi^t(I, S^{t-1})$ is an operator that extracts an image feature. In the process of model training, we need to give the characteristics and residuals, using the least squares method or other optimization methods to get the regression matrix W^t.

Next, we analyze the influence of the in-plane rotation of the face on the formulation $\Delta S = W^t \Phi^t (I, S^{t-1})$. Suppose that when we use the training set, there are two images in Fig. 5, and the left image is rotated to get the image on the right. The green feature points in the figure are the shapes of the current stage, S_1^t and S_2^t, and the red feature points are S_1^{gt} and S_1^{gt} , both of which are based on the plane coordinate system of the image.

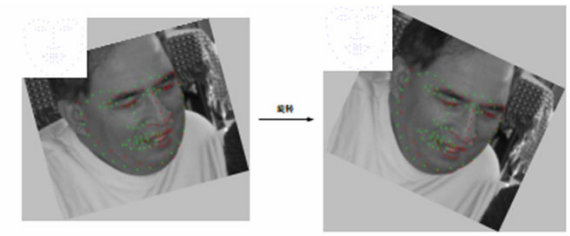

Fig. 5. Face rotation inside

After giving the shapes S_1^t and S_2^t of the current stage, use the corresponding feature, such as SIFT, HOG, or LBP, and so on. Since the left and right images are the same in position relative to the face, the extracted features are similar in some way. So,However, there is a clear difference between them. Due to the rotation of the image, we can get the following formula. Extract the corresponding characteristics by $\Phi^t(I, S^{t-1})$.

Since the positions of S_1^t and S_2^t in the left and right images are the same relative to the position of the face, the extracted features are similar in some way. So $\Phi(S_1^t) \approx \Phi(S_2^t)$. However, there is a significant difference between ΔS_1^t and ΔS_2^t. Due to the rotation of the image, we can get the following formula 11.

$$\begin{cases} S_2^t = cR(S_1^t) + \hat{t} \\ S_2^{gt} = cR(S_1^{gt}) + \hat{t} \end{cases} \tag{11}$$

Thus, we can deduce the transformation between ΔS_1^t and ΔS_2^t. ΔS_2^t is obtained by rotating and scaling by ΔS_1^t.

$$\Delta S_2^t = cR(\Delta S_1^t) \tag{12}$$

At this point, we define an average shape S^0 in the image coordinates. After the centering of S_1^t, S_2^t, and S^0 is expressed as $\overline{S}_1^t$, $\overline{S}_2^t$, and $\overline{S}^0$, the following transformations can be made.

$$\begin{cases} \overline{S}^0 = c_1 R_1(\overline{S}_1^t) \\ \overline{S}^0 = c_2 R_2(\overline{S}_2^t) \end{cases} \tag{13}$$

In summary, we get the results shown in Equation 14.

$$\Delta S_2^t = \frac{c_1}{c_2} R_2^{-1} R_1(\Delta S_1^t) \Rightarrow c_2 R_2 \Delta S_2^t = c_1 R_1 \Delta S_1^t \tag{14}$$

The transformation of the residual coordinate system is described above, which can eliminate the negative influence of the intra-plane rotation on the cascade regression training process. As far as possible, it is guaranteed that the target value to be returned, if the input feature is consistent or similar. The residuals are also consistent. Which helps to reduce the training error of the model, ensure the consistency of the input and output, and thus reduce the test error of the model. In addition to ensuring that in-plane rotation does not affect the residual value, we also need to ensure that in-plane rotation has no effect on feature extraction.

In summary, the main process of ΔS^t similarity transformation is shown following.

S^t and S^0 for central transformation, when the central transformation are completed after the solution, to solve the following transformation matrix according to equation 15 for cR, after that, we need change ΔS^t by 16.

$$S^0 = cRS^t \tag{15}$$

$$\tilde{S}^t = cR\Delta S^t \tag{16}$$

Use the change to solve the linear regression matrix. Mean_shape is mapped to intermediate_bboxes, so that S_0 and S_1 are at the same scale and roughly the same position, using the formula can be expressed as 17.

$$S_0\_resize = S_0 * Ratio + [Region(1), Region(2)] \tag{17}$$

Where the Ratio is actually the size of the intermediate-bboxes. So the above method can be the same calculation by 18, 19.

$$\begin{aligned} \widetilde{S_0} &= S_0\_Resize - mean(S_0\_Resize) \\ &= S_0 * Ratio - mean(S_0\_Resize) * Ratio \\ &= (S_0 - mean(S_0)) * Ratio = \overline{S_0} * Ratio \end{aligned} \tag{18}$$

$$\widetilde{S_0} = Ratio * \overline{S_0} = \widetilde{c_1}\widetilde{R_1}\overline{S_1} \tag{19}$$

Suppose $\overline{S_0} = c_1 R_1 \overline{S_1}$, equation 20 can be obtained according to the above formula.

$$c_1 R_1 = \widetilde{c_1}\widetilde{R_1} / Ratio = \widetilde{c_1}\widetilde{R_1} / intermediate\_bboxes \tag{20}$$

Therefore

$$\widetilde{\Delta S} = c_1 R_1 / \text{int}ermediate\_bboxes * \Delta S \tag{21}$$

So far, preparation of the initial data has been completed.

3.3 Training and Testing Phase

The training is divided into T stages, the number of regression times is T, and the training samples are first loaded. The training samples are Labeled Face Parts in the Wild (LFPW) dataset. Preprocess the image in the data set, and prepare the initial data. Prepare the initial data and prepare the training model. First extract the LBF feature in each stage, and then according to the real $\Delta\hat{S}_i$. A global regression can be used to train a regression device that can be used. Then use the trained regression to get the approximate value of $\Delta\hat{S}_i$, and use the resulting $\Delta\hat{S}_i$ to update the previous stage to get the shape, in order to get more accurate shape. This process is repeated until the number of i is greater than the number of regression times. Finally, the training of the model output, save to the appropriate path, to complete this training.

The overall goal of the training is to train five random trees for each feature point of the 68 feature points of each image of the LFPW dataset. The depth of each random tree is 4 layers, and all random tree merges are called random forests.

Now we have N samples, because we use the LFPW data set, LFPW data set with images and shapes a total of 1622 samples, that is, N = 1622. In addition, we have countless Pixel Difference Features. For each random tree, if we use the general method, we should randomly select several samples from the N samples and then select the M feature points. And then use these materials to be trained. However, in order to simplify the training process, we divide the N samples into five equal parts equally, and there are some overlapping parts between the five equal parts, that is, the training number of the first tree is 1-541, the training of the second tree No. 325-865, the third tree training number 649-1189, the fourth tree training number 973-1573, the first tree training number 1297-1622. And then distribute the assigned training samples as a common material for training five random trees of random forests.

In the training process, we need to give each training sample an initial shape, the initial shape can be randomly selected from the sample, by selecting multiple initial shapes, we can also achieve the effect of increasing the data set.

The following describes the whole process of node split, First prepare the data, the data contains training samples N, pixel difference features M, make $\Delta S = (\Delta X, \Delta Y)$. And let $var\_overall = N * (\text{cov}\,\Delta X + \text{cov}\,\Delta Y)$, Then select the first feature, for the first feature, randomly select a sample of the pixel difference features as the threshold Z, when the eigenvalue is less than the threshold Z, the sample on the left, the new variance left_var, and vice versa On the right, solve the new variance right_var, as

shown in 22.

$$\Delta \text{var} = \text{var}\_overall - (left\_\text{var} + right\_\text{var}) \quad \textbf{(22)}$$

The above steps are repeated until the M-th feature is trained. And then select the maximum number of T, the output threshold Z and feat location, the sample according to the characteristics of the left and right is divided into two parts. The process of splitting the nodes of the random tree is all done.

After training the random forest, we can get the LBF feature of the training data. According to this LBF feature, we can train the global linear regression model with the relative projection of relative shape. So far, all the training process is completed.

The process of the test phase is similar to the training phase. The test phase is loaded at the beginning and the model is trained before the image is predicted.

3.4 Experimental Results

First, prepare the face database, and then need to correspond to the Path_Images.txt file. Setting the global parameters in the LBF.cpp file You can find the three parameters of “modelPath”, “dataPath”, “cascadeName” in the file, where the “modelPath” parameter is the save path of the training phase model and the model in the test phase The loading path. The “dataPath” parameter is the path to load the dataset image, which should be the same as the path in Path_Images.txt that was previously generated. “CascadeName” parameters do not need to modify.

When testing the Helen dataset, the result is shown in the Fig. 6, the test data is loaded slowly, so a small sample of the Helen dataset is used for testing. The average predicted time is about 77.1906 ms and the variance is 17.3457%. When testing the LFPW data set, the test data is 215, the average test time is 97.128ms, and the variance is 17.6097%, the result of which is shown in the Fig. 7.

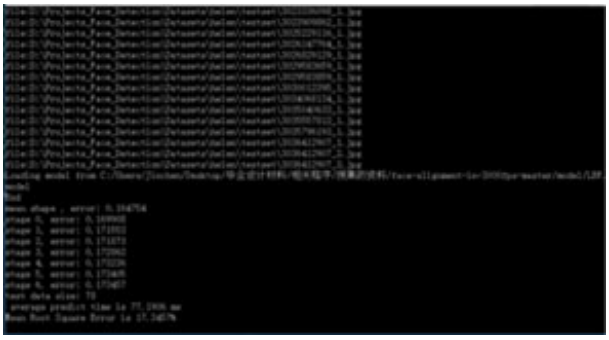

Fig. 6. Helen dataset test results

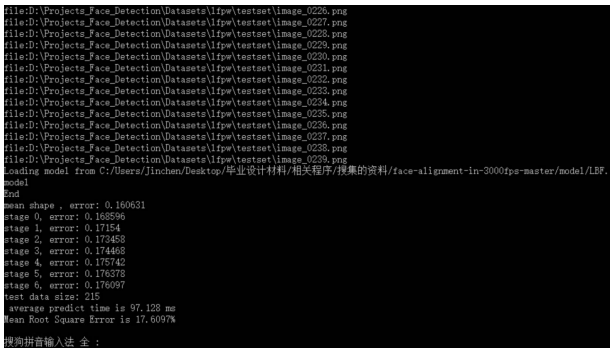

Fig. 7. LFPW dataset test results

In the process of identification, we can identify the feature points of the face at a fairly rapid rate, so as to realize the calibration of the face, as shown in Fig. 9 When we have multiple faces to identify, we can also identify the feature points of all faces, as shown in Fig. 9. When the camera has a foreign body or wear a mask and other reasons caused by the face is blocked, the program can also fill the face, but the results of the completion of a slight deviation from the real situation, as shown in Fig. 10 and Fig. 11.

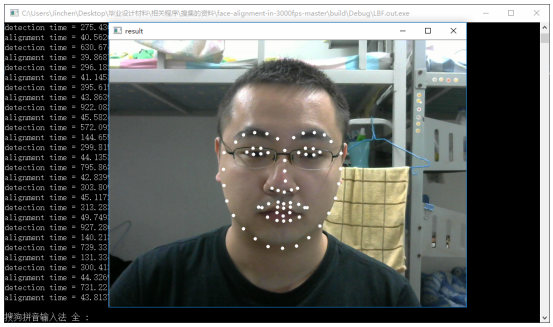

Fig. 8. Face calibration positioning

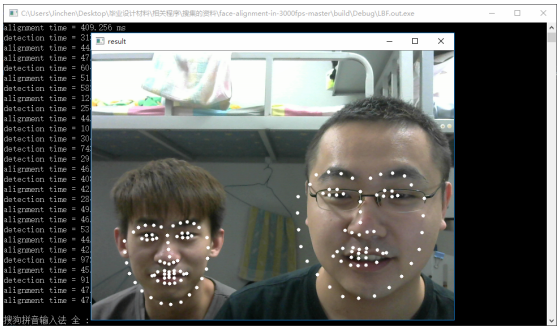

Fig. 9. Multi-face calibration positioning

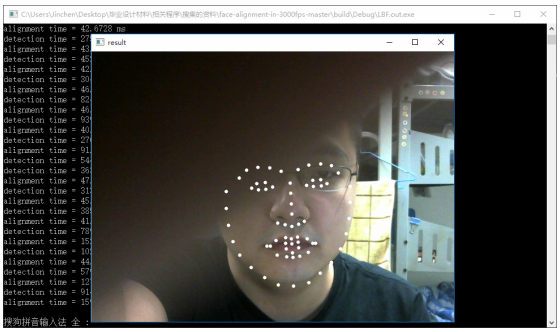

Fig. 10. Camera block recognition

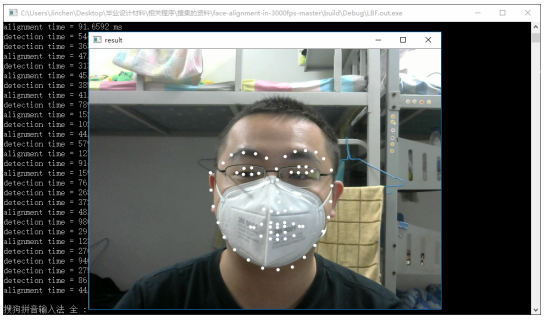

Fig. 11. Camera block recognition

4 Conclusion

In this paper, the method based on 3000FPS is used to complete the calibration of human face. The principle of 3000FPS algorithm is described in detail. At the same time according to the relevant papers to complete the implementation of 3000FPS algorithm, you can initially complete the face calibration

positioning function. Program for real-time use, the image calibration positioning speed of about 44ms or so. It can meet the real-time requirements.

Acknowledgements

This work is supported by the Training Plan of High Level Talents in Beijing Higher Education, the Virtual Teaching Team Plan of Intelligent Science and Technology in Beijing Higher Education.

References

[1] Cao X, Wei Y, Wen F, J. Sun, Face alignment by explicit shape regression, International Journal of Computer Vision 107(2)(2014) 177-190.

[2] T.F. Cootes, C.J. Taylor, D.H. Cooper, J. Graham, Active shape models- their training and application [J]. Computer Vision & Image Understanding 61(1)(1995) 38-59.

[3] K.Y.E. Leung, M.V. Stralen, M.M. Voormolen, N. de Jong, A.F.W. van der Steen, J.H.C. Reiber, J.G. Bosch, Improving 3D active appearance model segmentation of the left ventricle with Jacobian tuning, in: Proc. Medical Imaging 2008: International Society for Optics and Photonics, 2008.

[4] D. Cristinacce, T.F. Cootes, Feature detection and tracking with constrained local models, in: Proc. British Machine Vision Conference 2006, 2006.

[5] G. Tzimiropoulos, Project-out cascaded regression with an application to face alignment, in: Proc. Computer Vision and Pattern Recognition, 2015.

[6] X. Cao, Y. Wei, F. Wen, J. Sun, Face alignment by explicit shape regression, International Journal of Computer Vision 107(2)(2014) 177-190.

[7] Y. Liu, A. Jourabloo, W. Ren, X. Liu, Dense face alignment, in: Proc. 2017 IEEE International Conference on Computer Vision Workshop (ICCVW), 2017.

[8] M. Kowalski, J. Naruniec, T. Trzcinski, Deep alignment network: a convolutional neural network for robust face alignment, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition Workshop (CVPRW), 2017.

[9] S. Ren, X. Cao, Y. Wei, J. Sun, Face alignment at 3000 FPS via regressing local binary features, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2014.