An Iterative Regression Approach for Face Pose Estimation from RGB Images

Wenye He

This paper presents an iterative optimization method, explicit shape regression, for face pose detection and localization. The regression function is learnt to find out the entire facial shape and minimize the alignment errors. A cascaded learning framework is employed to enhance shape constraint during detection. A combination of a two-level boosted regression, shape indexed features and a correlation-based feature selection method is used to improve the performance. In this paper, we have explain the advantage of ESR for deformable object like face pose estimation and reveal its generic applications of the method. In the experiment, we compare the results with different work and demonstrate the accuracy and robustness in different scenarios.

Introduction

Pose estimation is an important problem in computer vision, and has enabled many practical application from face expression \(^1\) to activity tracking \(^2\). Researchers design a new algorithm called explicit shape regression (ESR) to find out face alignment from a picture \(^3\). Figure 1 shows how the system uses ESR to learn a shape of a human face image. A simple way to identify a face is to find out facial landmarks like eyes, nose, mouth and chin. The researchers define a face shape \(S\) and \(S\) is composed of \(N_{fp}\) facial landmarks. Therefore, they get \(S = [x_1, y_1, ..., x_{N_{fp}}, y_{N_{fp}}]^T\). The objective of the researchers is to estimate a shape \(S\) of a face image. The way to know the accuracy
of the estimation is to minimize the alignment error. Equation 1 show how to find alignment error and used for the cascaded learning framework.

$$||S - \hat{S}||_2,$$

(1)

where $\hat{S}$ is the ground true shape of the image. Nevertheless, the true shape is unknown in testing procedure, so optimization-based method and regression-based method are the most popular approaches for this problem. The researchers use the regression-based method, and, then, they create the algorithm, ESR. Compared to most previous works, ESR does not use any parametric shape models. Considering all facial landmarks are regressed jointly, the researchers train the regressor by reducing the alignment error over training data. This regressor solve large shape variations and guarantee robustness. Moreover, the researchers design the second regressor to solve small variation and ensure accuracy. Besides, ESR used the improved version of the cascaded pose regression framework. The layer based regressor has also be adopted for many other applications, such as depth completion. The next section goes over some related works.
Related Works

In previous works, active appearance models (AAM) is one of influential models in alignment approaches. In 1998, Cootes et al. introduce AAM to interpret an image. In the same year, they publish how this model interpret face image and recognize face. In 2007, Saragih and Goeckke propose a nonlinear discriminative approach to AAM. In 2010, Laurens and Emile make an extension of AAM to solve the large variation in face appearance. In 2011, Sauer and Cootes use random forest and boosting regression with AAM. In face alignment, regression-based method is a well-known one. In 2007, Cristinancce and Cootes construct constrained local models for face interpretation. In 2010, Valstar et al. apply boosted regression find facial features. Cascaded pose regressor (CPR) was originally derived from the boosted regression category. CPR has also been successfully used for face detection in videos.

Method

This section illustrates the ESR. A basic and essential terms are required to declared here, the normalized shape $M_S \circ S$. $M_S$ is produced from Equation 2 that looks for a $M$ that can make $S$ as close to as the mean shape $\bar{S}$ as possible.

$$M_S = \arg \min_M ||\bar{S} - M \circ S||_2,$$  \hspace{1cm} (2)

where $\bar{S}$ is the mean shape and $S$ is an input shape.

There are $N$ training samples $\{I_i, \hat{S}_i, S^{(0)}_i\}_{i=1}^N$, and the stage regressors $(R^1, ..., R^T)$. In every Stage
In each Stage $t$, the stage regressor $R^t$ is learnt like Equation 3.

$$R^t = \arg \min_R \sum_{i=1}^{N} ||y_i - R(I_i, S^t_{i-1})||^2$$

(3)

$$y_i = M_{S^t_{i-1}} \circ (\hat{S}_i - S^t_{i-1}),$$

where $S^t_{i-1}$ is the estimated shape in previous Stage $t-1$, and $M_{S^t_{i-1}} \circ (\hat{S}_i - S^t_{i-1})$ is the normalized regression target.

In testing, in each Stage $t$, the normalized shape $S^t_i$ is computed as follow,

$$S^t_i = S^t_{i-1} + M_{S^t_{i-1}}^{-1} \circ R^t(I_i, S^t_{i-1}),$$

(4)

where the normalized shape is updated by the regressor $R^t$ from $S^t_{i-1}$. The normalization reduces the complication of the regression. Suppose there are two facial images, $I_1$ and $I_2$ with the estimated shape and $I_2$ is transformed from $I_1$. The results of both regressions are different due to transformation. However, normalization, simplifying the problem, would give the same result in both regressions. Algorithm 1 below shows how ESR processing in both training and testing. In ESR training, the researchers initialize the training data, compute normalized targets and get the stage regressors. In the ESR testing, the researchers initialize the testing data, normalize targets and output the shape using the regressor.

In the initialization, the researchers has an $InitSet$ composed of exemplar shapes which are viewed as representative shapes or groundtruth shapes from the training data. Each exemplar shape generates a number of initial shapes. The researchers’ implementation here is 20. The initialization returns a triple $\{I^0_i, \hat{S}^0_i, S^0_i\}_{i=1}^{CD}$, the facial image, the groundtruth shape and the initial shape. Therefore, the number of a triple that has identical facial image and identical groundtruth shape is 20.
Algorithm 1 Explicit Shape Regression (ESR)

Variables: Training images and labeled shapes $\{I_l, \hat{S}_l\}_{l=1}^L$; ESR model $\{R^t\}_{t=1}^T$; Testing image $I$; predicted shape $S$; \textit{TrainParams}\{}times of data augment $N_{aug}$, number of stages $T$\}; \textit{TestParams}\{}number of multiple initializations $N_{int}$\}; \textit{InitSet} which contains exemplar shapes for initialization

ESRTraining($\{I_l, \hat{S}_l\}_{l=1}^L$, \textit{TrainParams}, \textit{InitSet})

// augment training data
$\{I_i, S_i, S_0^i\}_{i=1}^{N_{int}} \leftarrow$ Initialization($\{I_l, \hat{S}_l\}_{l=1}^L$, $N_{aug}$, \textit{InitSet})

for $t$ from 1 to $T$ do

$Y \leftarrow \{M_{S_{t-1}} \circ (\hat{S}_i - S_{t-1}^i)\}_{i=1}^{N_{int}}$ // compute normalized targets

$R^t \leftarrow$ LearnStageRegressor($Y$, $\{I_l, S_{t-1}^i\}_{i=1}^{N_{int}}$) using Eq. (3)

for $i$ from 1 to $N_{int}$ do

$S_i^t \leftarrow S_{t-1}^i + M_{S_{t-1}}^{-1} \circ R^t(I_l, S_{t-1}^i)$

end for

end for

return $\{R^t\}_{t=1}^T$

ESRTesting($I$, $\{R^t\}_{t=1}^T$, \textit{TestParams}, \textit{InitSet})

// multiple initializations
$\{I_i, *, S_0^i\}_{i=1}^{N_{int}} \leftarrow$ Initialization($\{I, *\}$, $N_{int}$, \textit{InitSet})

for $t$ from 1 to $T$ do

for $i$ from 1 to $N_{int}$ do

$S_i^t \leftarrow S_{t-1}^i + M_{S_{t-1}}^{-1} \circ R^t(I_l, S_{t-1}^i)$

end for

end for

$S \leftarrow$ CombineMultipleResults($\{S_T^i\}_{i=1}^{N_{int}}$)

return $S$

Initialization($\{I_c, \hat{S}_c\}_{c=1}^C$, $D$, \textit{InitSet})

$i \leftarrow 1$

for $c$ from 1 to $C$ do

for $d$ from 1 to $D$ do

$S_0^i \leftarrow$ sampling an exemplar shape from \textit{InitSet}

$\{I_0^i, S_0^i\} \leftarrow \{I_c, \hat{S}_c\}$

$i \leftarrow i + 1$

end for

end for

return $\{I_0^i, \hat{S}_0^i, S_0^i\}_{i=1}^{CD}$
To improve the performance of the experiments, the researchers introduce two-level boosted regression, external-level and internal-level. The stage regressor \( R^{t} \) is internal-level and is called primitive regressor. The number of iterations is \( TK \), where \( T \) is the number of iterations in the external-level and \( K \) is that in the others. Figure 2 shows the tradeoffs between two level boosted regression. When the total iterations is 5000, the researchers find the lowest mean error as \( T = 10 \) and \( K = 500 \).

| Stage regressors (\( T \) ) | 1  | 5  | 10 | 100 | 5000 |
|-----------------------------|----|----|----|-----|------|
| Primitive regressors (\( K \) ) | 5000 | 1000 | 500 | 50  | 1    |
| Mean error (\( \times 10^{-2} \)) | 15 | 6.2 | 3.3 | 4.5 | 5.2  |

![Figure 2: Tradeoffs between two level boosted regression](image)

A fern\(^{[16]}\) is used as the internal-level regressor. Fern can be considered as pixel domain comparison in contrast to features, and can be used for image matching\(^{[6]}\). In the researchers’ implementation, a fern contains \( F = 5 \) features. Thus, the feature space and all training samples \( \{ \hat{y}_{i} \}_{i=1}^{N} \) are divided into \( 2^F \) bins and every bin \( b \) means a regression output \( y_{b} \). The prediction of a bin is calculated like Equation 5,

\[
y_{b} = \arg\min_{y} \sum_{i \in \Omega_{b}} ||\hat{y}_{i} - y||_{2},
\]

where the set \( \Omega_{b} \) implies the samples in the \( b \)th bin. The best solution is the average,

\[
y_{b} = \frac{\sum_{i \in \Omega_{b}} \hat{y}_{i}}{|\Omega_{b}|}.
\]

To solve the over-fitting problem, Equation 6 is revised into Equation 7.

\[
y_{b} = \frac{1}{1 + \beta/|\Omega_{b}|} \frac{\sum_{i \in \Omega_{b}} \hat{y}_{i}}{|\Omega_{b}|},
\]

where \( \beta \) is a parameter to control the smoothness of the regression. The value of \( \beta \) varies from 0 to \( N \), and it is used as a penalty term to balance the tradeoff between training error and smoothness.
where $\beta$ is a free shrinkage parameter. Given that the final regressed shape $S$ starts from the initial shape $S^0$ and updates by catching the information from the groundtruth shapes, $S$ is computed like Equation 8,

$$S = S^0 + \sum_{i=1}^{N} w_i \hat{S}_i. \quad (8)$$

Before demonstrating the procedure of the internal-boosted regression, shape indexed features is introduced as an important ingredient in the regression. The researchers use pixel-difference features, which means the intensity difference of two pixels in the image. The extracted pixels are unchangeable to similarity transform and normalization. Each pixel is indexed by the local coordinate $\delta^l = (\delta x^l, \delta y^l)$, where $l$ is a landmark associated with the pixel. Considering that pixels indexed by the same global coordinates may be variant because of different face shape while those in the same local coordinates are invariant though in different face shape (detail in Figure 3), the researchers find the local coordinates first and, then, turn them back to the global coordinates for the intensity difference of two pixels. The transform is showed in Equation 9.

$$\pi_l \circ S + M_{S^{-1}} \circ \Delta^l, \quad (9)$$

Figure 3: Left pair of images show the local coordinates and right pair of images show the global coordinates. The global coordinates have different meaning due to different images.

7
where \( \pi_i \) is the operator to get the \( x \) and \( y \) coordinates of a landmark from the shape. In Figure 3 and the explanation above, different samples has the identical \( \delta l \). Algorithm 2 shows how the researchers get shape indexed features. In Algorithm 2, \( P \) numbers of pixels are generated, so

**Algorithm 2** Shape indexed features

**Variables:** images and corresponding estimated shapes \( \{I_i, S_i\}_{i=1}^N \); number of shape indexed pixel features \( P \); number of facial points \( N_{fp} \); the range of local coordinate \( \kappa \); local coordinates \( \{\Delta l_\alpha\}_{\alpha=1}^P \); shape indexed pixel features \( \rho \in \mathbb{R}^{N \times P} \); shape indexed pixel-difference features \( X \in \mathbb{R}^{N \times P^2} \);

\[
\text{GenerateShapeIndexedFeatures} \left( \{I_i, S_i\}_{i=1}^N, N_{fp}, P, \kappa \right)
\]

\[
\{\Delta l_\alpha\}_{\alpha=1}^P \leftarrow \text{GenerateLocalCoordinates} \left( \text{FeatureParams} \right)
\]

\[
\rho \leftarrow \text{ExtractShapeIndexedPixels} \left( \{I_i, S_i\}_{i=1}^N, \{\Delta l_\alpha\}_{\alpha=1}^P \right)
\]

\[
X \leftarrow \text{pairwise difference of all columns of } \rho
\]

**GenerateLocalCoordinates** \((N_{fp}, P, \kappa)\)

for \( \alpha \) from 1 to \( P \) do

\[
l_\alpha \leftarrow \text{randomly drawn a integer in } [1, N_{fp}]
\]

\[
\Delta l_\alpha \leftarrow \text{randomly drawn two floats in } [-\kappa, \kappa]
\]

end for

**ExtractShapeIndexedPixels** \((\{I_i, S_i\}_{i=1}^N, \{\Delta l_\alpha\}_{\alpha=1}^P)\)

for \( i \) from 1 to \( N \) do

for \( \alpha \) from 1 to \( P \) do

\[
\mu_\alpha \leftarrow \pi_{l_\alpha} \circ S_i + M_{S_i}^{-1} \circ \Delta l_\alpha
\]

\[
\rho_{i\alpha} \leftarrow I_i(\mu_\alpha)
\]

end for

end for

**Variables:**

\[
\{\Delta l_\alpha\}_{\alpha=1}^P, \rho, X
\]

there are \( P^2 \) number of pixel-difference features. This makes a huge number of calculations. To optimize the performance and improve the efficiency, the researchers create a correlation-based feature selection to reduce unnecessary computations.

The researchers selects \( F \) out of \( P^2 \) features to form a fern regressor. Two properties are required
for a good fern: high correlation between each feature in the fern and the regression target and low correlation between features enough to composed complementarily. They propose Equation 10 to maximizing feature’s correlation:

$$ j_{opt} = \arg\min_j \text{corr}(Y_{prob}, X_j), $$

where $Y$ is the regression target with $N$ rows and $2N_{fp}$ columns, and $X$ is pixel-difference features matrix with $N$ rows and $P^2$ columns. $N$ is the number of samples. Each column $X_j$ of feature matrix represents a pixel-difference feature. $y_{prob}$ means a projection $Y$ into a column vector from unit Gaussian. The researchers write a pixel-difference feature as $\rho_m - \rho_n$, and they get Equation 10 about the correlation below:

$$ \text{corr}(Y_{prob}, \rho_m - \rho_n) = \frac{\text{corr}(Y_{prob}, \rho_m) - \text{corr}(Y_{prob}, \rho_n)}{\sqrt{\sigma(Y_{proj})\sigma(\rho_m - \rho_n)}} $$

$$ \sigma(\rho_m - \rho_n) = \text{cov}(\rho_m, \rho_m) + \text{cov}(\rho_n, \rho_n) - 2\text{cov}(\rho_m, \rho_n). $$

The pixel-pixel covariances $\sigma(\rho_m - \rho_n)$ can be pre-computed and reused with in each internal-level boosted regression due to fixity of the shape indexed pixels, which reduces the complexity from $O(NP^2)$ to $O(NP)$. Algorithm 3 is the method to selecting correlation-based feature.

After introduction of shape indexed features and correlation-based feature selection, the internal-boosted regression is explained in Algorithm 4. The regression consists of $K$ primitive regressors $\{r_1, ..., r_K\}$, which are ferns. $F$ thresholds are sampled randomly from an uniform distribution provided the range of pixel difference feature is $[-c, c]$, the range of the uniform distribution is $[-0.2c, 0.2c]$. In each iteration a new primitive regressor is learn from the residues left by previous regressors.
Algorithm 3 Shape indexed features

**Input:** regression targets $Y \in \mathbb{R}^{N \times 2N_{fp}}$; shape indexed pixel features $\rho \in \mathbb{R}^{N \times N}$; pixel-pixel covariance $\text{cov}(\rho) \in \mathbb{R}^{P \times P}$; number of features of a fern $F$;

**Output:** The selected pixel-difference features $\{\rho_{m_f} - \rho_{n_f}\}_{f=1}^F$ and the corresponding indices $\{m_f, n_f\}_{f=1}^F$;

**CorrelationBasedFeatureSelection**($Y, \text{cov}(\rho), F$)

for $f$ from 1 to $F$ do
  $v \leftarrow \text{randn}(2N_{fp}, 1)$ // draw a random projection from unit Gaussian
  $Y_{\text{prob}} \leftarrow Yv$ // random projection
  $\text{cov}(Y_{\text{prob}}, \rho) \in \mathbb{R}^{1 \times P}$ // compute target-pixel covariance
  $\sigma(Y_{\text{prob}}) \leftarrow$ compute sample variance of $Y_{\text{prob}}$
  $m_f = 1; n_f = 1$;
  for $m$ from 1 to $P$ do
    for $n$ from 1 to $P$ do
      $\text{corr}(Y_{\text{prob}}, \rho_m - \rho_n) \leftarrow$ compute correlation using Eq.(11)
      if $\text{corr}(Y_{\text{prob}}, \rho_m - \rho_n) > \text{corr}(Y_{\text{prob}}, \rho_{m_f} - \rho_{n_f})$ then
        $m_f = m; n_f = n$;
      end if
    end for
  end for
return $\{\rho_{m_f} - \rho_{n_f}\}_{f=1}^F, \{m_f, n_f\}_{f=1}^F$
Algorithm 4 Internal-level boosted regression

Variables: regression targets $Y \in \mathbb{R}^{N \times 2N_{fp}}$; training images and corresponding estimated shapes $\{I_i, S_i\}_{i=1}^N$; training parameters $TrainParams\{N_{fp}, P, \kappa, F, K\}$; the stage regressor $R$; testing image and corresponding estimated shape $\{I, S\}$;

$LearnStageRegressor(Y, \{I_i, S_i\}_{i=1}^N, TrainParams)\$

$\{\Delta^l_{\alpha}\}_{\alpha=1}^{P} \leftarrow GenerateLocalCoordinates(N_{fp}, P, \kappa)\$

$\rho \leftarrow ExtractShapeIndexedPixels(\{I_i, S_i\}_{i=1}^N, \{\Delta^l_{\alpha}\}_{\alpha=1}^{P})\$

$\text{cov}(\rho) \leftarrow \text{pre-compute pixel-pixel covariance}\$

$Y^0 \leftarrow Y \parallel \text{initialization}\$

for $k$ from 1 to $K$
do

$\{\rho_{mf} - \rho_{nf}\}_{f=1}^F, \{m_f, n_f\}_{f=1}^F \leftarrow CorrelationBasedFeatureSelection(Y^{k-1}, \text{cov}(\rho), F)\$

$\{\theta_f\}_{f=1}^F \leftarrow \text{thresholds from an uniform distribution}\$

$\{\Omega_b\}_{b=1}^{2^F} \leftarrow \text{partition training samples into } 2^F \text{ bins}\$

$\{y_b\}_{b=1}^{2^F} \leftarrow \text{randomly drawn two floats in } [-\kappa, \kappa]\$

$\text{compute the outputs of all bins using Eq.(7)}\$

$r_k \leftarrow \{(m_f, n_f)_{f=1}^F, \{\theta_f\}_{f=1}^F, \{y_b\}_{b=1}^{2^F}\} \parallel \text{construct a fern}\$

$Y_k \leftarrow Y^{k-1} - r^k(\rho_{mf} - \rho_{nf})_{f=1}^F \parallel \text{update the targets}\$

end for

$R \leftarrow \{\{r^k\}_{k=1}^K, \Delta_{\alpha}^l\}_{\alpha=1}^P \parallel \text{construct stage regressor}\$

return $R$

$ApplyStageRegressor(I, S, R) \parallel \text{i.e. } R(I, S)$

$\rho \leftarrow ExtractShapeIndexedPixels(\{I, S\}, \{\Delta_{\alpha}^l\}_{\alpha=1}^P)\$

$\delta S \leftarrow 0$

for $k$ from 1 to $K$
do

$\delta S \leftarrow \delta S + r^k(\rho_{mf} - \rho_{nf})_{f=1}^F$

end for

return $\delta S$
Experiments and results

The researchers compare their approach to previous approaches. Figure 4-6 show samples of images from different datasets.

Figure 4: Selected results from LFPW

Figure 5: Selected results from LFPW87

Comparison with CE on LFPW  Compared to the consensus exemplar approach\cite{ref}, ESR has more than 10% accurate on most landmarks estimation and smaller overall error.
Figure 6: Selected results from Helen dataset

Figure 7: 29 facial landmarks on the left image. The radius of circle implies the average error of ESR. The color of the circles suggests accuracy improvement over the CE method. Green has more than 10% accuracy, while cyan has less than 10% accuracy. The bar graph shows the different accuracy between two methods among all landmarks. The small table shows average error of all landmarks in both methods.

Comparison with CDS on LFPW87  A component-based discriminative search (CDS) method is proposed by Liang et al. ESR beats CDS due to lower error rate.

| RMSE       | <5 Pixels | <7.5 Pixels | <10 Pixels |
|------------|-----------|-------------|------------|
| CDS (%)    | 74.7      | 93.5        | 97.8       |
| Our method (%) | 86.1      | 95.2        | 98.2       |

Figure 8: Percentages of test images with root mean square error less than given thresholds on the LFW87 dataset

Comparison with STASM and CompASM on Helen ESR has around 40% – 50% lower mean error on Helen dataset than STASM and CompASM have.
| Method     | Mean | Median | Min  | Max  |
|------------|------|--------|------|------|
| STASM      | 0.111| 0.094  | 0.037| 0.411|
| CompASM    | 0.091| 0.073  | 0.035| 0.402|
| Our method | **0.057** | **0.048** | **0.024** | **0.16** |

Figure 9: Different errors in ESR, STASM and CompASM

Reference

1. S.B. Gokturk, J.-Y. Bouguet, “Model-based face tracking for view-independent facial expression recognition,” Automatic Face and Gesture Recognition, pp. 55-37, 2012.

2. J. Shen and J. Yang, “Automatic human animation for non-humanoid 3d characters,” International Conference on Computer-Aided Design and Computer Graphics (CAD/Graphics), pp. 220-221, 2015.

3. Cao, Xudong, et al. ”Face alignment by explicit shape regression.” International Journal of Computer Vision 107.2 (2014): 177-190.

4. Dollr, Piotr, Peter Welinder, and Pietro Perona. ”Cascaded pose regression.” Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on. IEEE, 2010.

5. Cootes, Timothy F., Gareth J. Edwards, and Christopher J. Taylor. ”Active appearance models.” European conference on computer vision. Springer Berlin Heidelberg, 1998.

6. J. Shen and S. C. S. Cheung, “Layer Depth Denoising and Completion for Structured-Light RGB-D Cameras,” IEEE Conference on Computer Vision and Pattern Recognition, pp. 1187-1194, 2013.
7. Edwards, Gareth J., Christopher J. Taylor, and Timothy F. Cootes. "Interpreting face images using active appearance models.” Automatic Face and Gesture Recognition, 1998. Proceedings. Third IEEE International Conference on. IEEE, 1998.

8. Edwards, Gareth, Timothy Cootes, and Christopher Taylor. "Face recognition using active appearance models.” Computer VisionECCV98 (1998): 581-595.

9. J. Shen and W. Tan, “Image-based indoor place-finder using image to plane matching,” IEEE International Conference on Multimedia and Expo, pp. 1-6, 2013.

10. Saragih, Jason, and Roland Goecke. ”A nonlinear discriminative approach to AAM fitting.” Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on. IEEE, 2007.

11. Van Der Maaten, Laurens, and Emile Hendriks. “Capturing appearance variation in active appearance models.” Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on. IEEE, 2010.

12. R. Liu, J. Shen, Q. Sun, J. Yang, S. C. S. Cheung, “Cascaded Pose Regression Revisited: Face Alignment in Videos,” IEEE Third International Conference on Multimedia Big Data (BigMM), pp. 291-298, 2017.

13. Patrick Sauer, Tim Cootes and Chris Taylor. Accurate Regression Procedures for Active Appearance Models. In Jesse Hoey, Stephen McKenna and Emanuele Trucco, Proceedings of the British Machine Vision Conference, pages 30.1-30.11. BMVA Press, September 2011. http://dx.doi.org/10.5244/C.25.30
14. Cristinacce, David, and Timothy F. Cootes. "Boosted regression active shape models.” BMVC. Vol. 2. 2007.

15. Valstar, Michel, et al. "Facial point detection using boosted regression and graph models.” Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on. IEEE, 2010.

16. Ozuysal, Mustafa, et al. "Fast keypoint recognition using random ferns.” IEEE transactions on pattern analysis and machine intelligence 32.3 (2010): 448-461.

17. Belhumeur, Peter N., et al. "Localizing parts of faces using a consensus of exemplars.” IEEE transactions on pattern analysis and machine intelligence 35.12 (2013): 2930-2940.

18. Liang, Lin, et al. "Face alignment via component-based discriminative search.” Computer Vision ECCV 2008 (2008): 72-85.

19. Le, Vuong, et al. "Interactive facial feature localization.” Computer Vision ECCV 2012 (2012): 679-692.

20. Milborrow, Stephen, and Fred Nicolls. "Locating facial features with an extended active shape model.” European conference on computer vision. Springer Berlin Heidelberg, 2008.