Online Smart Face Morphing Engine with Prior Constraints and Local Geometry Preservation

Quan Wang(✉), Yu Wang, and Zuoguan Wang

Department of Electrical, Computer, and Systems Engineering,
Rensselaer Polytechnic Institute, Troy, NY 12180, USA
wangq10@rpi.edu

Abstract. We present an online system for automatic smart face morphing, which can be accessed at http://facewarping.com. This system morphs the user-uploaded image to "beautify" it by embedding features from a user-selected celebrity. This system consists of three major modules: facial feature point detection, geometry embedding, and image warping. To embed the features of a celebrity face and at the same time preserve the features of the original face, we formulate an optimization problem which we call prior multidimensional scaling (prior MDS). We propose an iterated Levenberg-Marquardt algorithm (ILMA) to efficiently solve prior MDS in the general case. This online system allows the user to configure the morphing parameters, and has been tested under different conditions.

Keywords: Feature point detection · Geometry embedding · Image warping · Multidimensional scaling

1 Introduction

What features make a face look aesthetically pleasing? Though the answers may differ from person to person, depending on his cultural background, gender, age, experiences, and personal taste, we know that many celebrities are widely recognized as aesthetically appealing in their appearance. The underlying mechanism is that there are some widely existing standards people subconsciously use to judge the aesthetics of a face, such as eye distance and size of mouth. There has been work to study these underlying standards, or to learn them from data, such as [1–4]. In all these papers, facial geometry is used as an important feature to represent the attractiveness. In [1], the authors also use color and texture features. In [2], the authors consider hair color, facial symmetry, and skin smoothness. In [3], more higher-level semantics-sensitive features are used, including HOG, GIST, L*a*b* color histograms, and SIFT features.

In our morphing system, we only change the facial geometry of the user-uploaded image, while preserving other factors of the original image such as color and texture. This ensures that the resulting image still looks natural without conspicuous artifacts. Like many previous efforts, we detect salient facial feature

F. Schwenker et al. (Eds.): MPRSS 2014, LNAI 8869, pp. 130–140, 2015.
DOI: 10.1007/978-3-319-14899-1_13

points (also referred to as *landmarks*) from the face image, and use the distances between each pair of these points as our geometry features. Though similar to Leyvand *et al.*'s engine [4], our system differs in numerous aspects: (1) The facial points detection algorithm we use is the extended active shape model (ASM) with 2D profiles and trimmed covariance matrix [5], while in [4] the authors use the Bayesian Tangent Shape Model (BTSM) [6]. (2) We embed the prior constraint of the original face directly into the target function of the multidimensional scaling (MDS) problem, and solve it with iterated Levenberg-Marquardt algorithm (ILMA), while in [4] the authors simply assume that the solution to the standard MDS should be close to original feature points if the algorithm is well initialized. (3) For image warping, we use the thin-plate splines method [7], while in [4] the authors use multilevel free-form deformation (MFFD) [8]. The choice of these primitive algorithms for each module of our system is a comprehensive consideration of not only efficiency and performance, but also configurability, difficulty of implementation, and code maintainability.

2 System Overview

A flowchart of our system pipeline is provided in Fig. 1. First, the user can either upload an image, provide the URL to an image on the Internet, or click a test image from the AR face dataset [9]. This user-specified image will be referred to as the *original image* in the context. Next, the user selects a *reference image* from the MSRA-CFW dataset of celebrity faces [10]. Then the user specifies several parameters of the morphing process. Current parameters include whether to warp the eye, whether to warp the mouth, and the morphing degree $1-\mu$. With these inputs, our system first detects the facial landmarks of the original image, and looks up the pre-computed facial landmarks of the reference image. It then calls the iterated Levenberg-Marquardt algorithm to compute the positions of

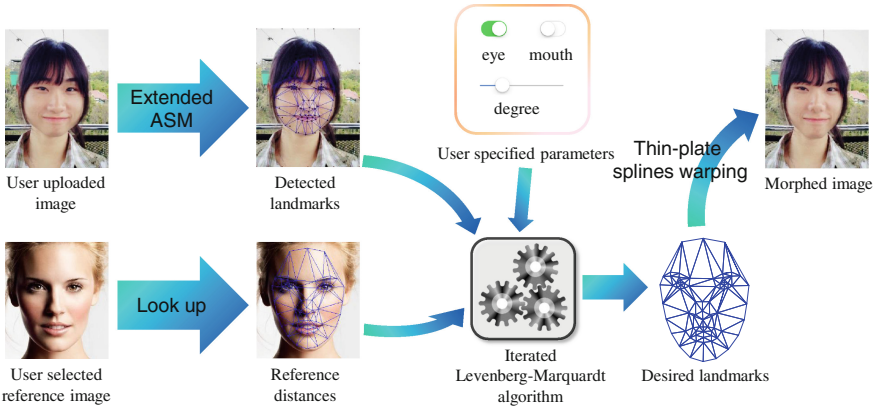

Fig. 1. A flowchart of our online morphing system.

desired landmarks. Finally, the system uses thin-plate splines warping to generate the warped image and displays it on the web page.

3 Feature Point Detection

Facial feature points, such as eye centers, nose tips and mouth corners, are semantically well defined to represent the shape of a face. Existing algorithms for facial feature points detection include random ferns regression [11], convolutional neural network (CNN) regression [12], and the Active Shape Model (ASM) [13] method, which has been extensively used in many other applications. The basic idea of ASM is to iteratively fit feature points in a face image by alternating between the steps of adjusting feature points with template matching and conforming the face shape to a global shape model. A number of extensions of ASM have been proposed. Bayesian Tangent Shape Model (BTSM) [6] is one of them, which designs a Bayesian approach for shape registration, and has been used in [4] for face beautification. Our system adopts the latest extension of ASM [5], which integrates a bunch of techniques to improve ASM in a sound manner. It achieves state-of-the-art accuracy with an improved efficiency, and thus is more suitable for practical (especially web-based) applications. Our model uses $N = 77$ landmarks (Fig. 2), and is trained on the MUCT data [14].

Before applying ASM for face detection, the bounding box of the face is detected using Haar feature-based cascade classifiers [15].

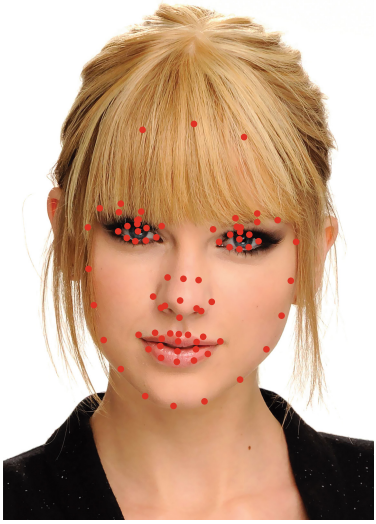

Fig. 2. An example of the 77 facial feature points.

4 Distance-Based Landmark Determination

Let the N landmark points of the reference face be $Q = \{q_1, q_2, \ldots, q_N\}$, and the distance between two landmarks q_i and q_j be $d_{i,j}$. For the original face, the landmark points are $P = \{p_1, p_2, \ldots, p_N\}$, and we morph the face such that each original landmark point p_i moves to a new position x_i. To perform a distance-preserving morphing, we hope the distances between the new positions are close to those of the reference face. This is a standard multidimensional scaling (MDS) problem [16], where we minimize a stress function $S(\cdot)$:

$$S(x_1, \ldots, x_N) = \sum_{1 \le i < j \le N} (||x_i - x_j|| - d_{i,j})^2. \tag{1}$$

There are lots of methods to solve the standard MDS problem:

$$\min_{x_1, \ldots, x_N} S(x_1, \ldots, x_N), \tag{2}$$

such as the iterative steepest descent approach by Kruskal [17] and the iterative majorization algorithm (SMACOF) by de Leeuw [18]. According to Williams, metric MDS problems can also be solved by recasting it to kernel PCA [19].

4.1 Prior Constraints by Original Face

The minimization problem in Eq. (2) has infinite many solutions since by representing a face with distances, we lose the location information of the face. Which solution we get by solving the minimization problem depends on the algorithm we use and the initialization of the algorithm. In [4], the authors did not add any extra constraints to make the solution unique. Instead, they assumed that using the original facial landmarks for initialization, the solution to the MDS problem in Eq. (2) should guarantee a minute modification from the original face. Although this assumption is empirically acceptable in many cases, if we generalize to higher dimensions and more complicated problems, the quality of different solutions of Eq. (2) may be very different. In our system, we address this concern and ensure the uniqueness of the solution by elegantly adding the prior knowledge of the original face directly into the stress function that we are minimizing:

$$\widetilde{S}(x_1, \ldots, x_N) = \frac{\mu}{N} \sum_{1 \le i \le N} ||x_i - p_i||^2 + \frac{2(1-\mu)}{N(N-1)} \sum_{1 \le i < j \le N} (||x_i - x_j|| - d_{i,j})^2. \tag{3}$$

Here $\mu \in [0, 1]$ is a weight parameter balancing between the constraints of the original face and the reference face. The $\frac{1}{N}$ and $\frac{2}{N(N-1)}$ are normalization factors of the two summations. The first summation in Eq. (3) is the prior constraint of the original face, and the second summation is the constraint of the reference face, which is just the raw stress in Eq. (1) multiplied by a constant. If μ is small, then the reference face has a larger weight. For this reason, we refer to the value of $1 - \mu$ as the *morphing degree* in the context (Fig. 5).

To minimize the new stress function:

$$\min_{x_1,\ldots,x_N} \widetilde{S}(x_1,\ldots,x_N), \tag{4}$$

we cannot directly use the standard SMACOF algorithm or kernel PCA method. Here we propose an iterative least squares solution to the *prior MDS* optimization problem Eq. (4). We note that in Eq. (4), the new stress function $\widetilde{S}(x_1,\ldots,x_N)$ is minimized with respect to $X = \{x_1,\ldots,x_N\}$, which has $N \times m$ dimensions, where m is the dimension of each x_i. In our face morphing problem, $N = 77$ and $m = 2$. In a more general case of the prior MDS problem, when both N and m are large, this nonlinear optimization problem becomes computationally intractable if we attempt to solve for all dimensions in one step. Instead, we iteratively optimize each x_i while fixing all other points. In each step, we minimize the stress function $\widetilde{S}(x_1,\ldots,x_N)$ with respect to only an m-dimensional variable, instead of $N \times m$, which greatly reduces the complexity of the optimization problem. The subproblem can be viewed as a least squares problem, and can be solved by the standard Levenberg-Marquardt algorithm [20,21]. Since the total stress Eq. (3) is monotonically non-increasing through time, the convergence of the optimization is guaranteed. We call our method the iterated Levenberg-Marquardt algorithm (ILMA), which is detailed in Algorithm 1.

Algorithm 1. The iterated Levenberg-Marquardt algorithm (ILMA).

input :Target distances $\{d_{i,j}\}$, where $1 \le i < j \le N$;
Prior points $P = \{p_1,\ldots,p_N\}$;
Balancing parameter $\mu \in [0,1]$;
Max number of iterations T;
Exit threshold ϵ;
output: New points $X = \{x_1,\ldots,x_N\}$;

begin
  for $i \leftarrow 1$ **to** N **do**
    Initialize $x_i = p_i$;
  end
  for $t \leftarrow 1$ **to** T **do**
    Generate a random permutation $(r_1, r_2, \ldots, r_N)$ of integers 1 to N;
    for $i \leftarrow 1$ **to** N **do**
      Use the standard Levenberg-Marquardt algorithm to find the x_{r_i} that minimizes:

$$\frac{\mu(N-1)}{2(1-\mu)}||x_{r_i} - p_{r_i}||^2 + \sum_{j \neq i}(||x_{r_i} - x_j|| - d_{r_i,j})^2;$$

    end
    if $\Delta\widetilde{S}/\widetilde{S} < \epsilon$ **then** break
  end
done

4.2 Partial Distances

For all the distances $\{d_{i,j}\}_{1\le i<j\le N}$, we may not want to give them all equal weights. Some of the distances should be considered more important than others. Thus for the second term of stress $\widetilde{S}(x_1,\ldots,x_N)$ in Eq. (3), we may rewrite it as:

$$\frac{2(1-\mu)}{N(N-1)}\sum_{1\le i<j\le N}\beta_{i,j}(||x_i-x_j||-d_{i,j})^2\,, \tag{5}$$

where $\beta_{i,j}$ is the weight for distance $d_{i,j}$. With this change in the target stress function, we can simply modify Algorithm 1 by changing the target function in step 8 to:

$$\frac{\mu(N-1)}{2(1-\mu)}||x_{r_i}-p_{r_i}||^2+\sum_{j\neq i}\beta_{i,j}(||x_{r_i}-x_j||-d_{r_i,j})^2\,. \tag{6}$$

4.3 Scale Invariance

To make sure our morphing algorithm is scale invariant, distance normalization is necessary. We simply normalize the target distances of the celebrity such that the mean distance of all landmark pairs of the celebrity equals that of the original image. This operation is performed before the ILMA.

4.4 Local Geometry Preservation

People are sensitive to even very small shape changes of facial organs such as eyes, the nose and the mouth. The optimization of landmarks in Sect. 4.1 gives no consideration to this issue and in some cases the resulting facial organs look unnatural. The work in [2] noticed this problem to the eye shapes, and applied a linear transform on original facial points that relocates eyes by minimizing the mean square distance between transformed position and the position resulted from their beauty engine. While linear transform mitigates this problem, it still cannot get rid of distortions. Our system addresses this issue by applying a rigid transform to the points of each facial organ. It preserves the original organ shape while adjusting its position close to the one optimized by our morphing engine. Assume the two point sets $P=\{p_i\}_{1\le i\le N}$ and $X=\{x_i\}_{1\le i\le N}$ are the original points of a facial organ and the ones from morphing engine, respectively. The rigid transform takes the form:

$$\widetilde{p}_i = R\cdot p_i + T, \tag{7}$$

where R is a rotation matrix and T is a translation vector:

$$R=\begin{bmatrix}\cos\theta & -\sin\theta\\ \sin\theta & \cos\theta\end{bmatrix}, \tag{8}$$

$$T=\begin{bmatrix}t_x\\ t_y\end{bmatrix}. \tag{9}$$

We find R and T by minimizing $\sum_{i=1}^{N} ||\widetilde{p}_i - x_i||^2$, which can be efficiently solved by using singular value decomposition (SVD) [22]. Then we replace x_i with $\widetilde{p}_i$ for this facial organ in the system.

5 Image Warping

Given the original landmarks $P = \{p_1, p_2, \ldots, p_N\}$ and the desired landmarks $X = \{x_1, x_2, \ldots, x_N\}$, we want to warp the entire original image I_o to a new target image I_t, such that each p_i is warped to the corresponding x_i. We simply use the thin-plate splines (TPS) warping method [7], where we find a *backward* thin-plate splines function $g(\cdot)$, such that $g(x_i) = p_i$ for $i = 1, \ldots, N$. Then for any pixel x in the target image I_t, we project it to $p = g(x)$, interpolate the pixel value of p in the original image I_o, and assign the value to x in I_t.

6 Implementation Details

6.1 Hosting

We implemented our smart face morphing system as a web-based application that can be accessed from any device equipped with a web browser. The front-end of the system is implemented using JavaScript and CherryPy — a light-weight Python web framework. The back-end of the system, which is the face morphing engine, is implemented in C++, and we use OpenCV2 for basic image handling. The website is currently hosted on a free-tier instance (known as t1.micro) of Amazon Elastic Compute Cloud (EC2).

6.2 User Interface

The system allows users to upload an image, use a web image, or simply select from sample testing images (AR face database [9]) as the input. The reference dataset is a mixture of the MSRA-CFW dataset of celebrity faces [10] and images that we scraped from Google Images using a Python script. For aesthetic reasons, we only displayed a few reference images on the web page (Fig. 3). The system also gives users options to enable/disable warping of the eyes or mouth, and adjust the morphing degree $1 - \mu$ of the prior MDS.

6.3 Border Preservation

When we warp the original image using the new desired facial feature points $X = \{x_i\}_{1 \le i \le N}$ with thin-plate splines method, not only the facial region in the image is warped, but also the background region is warped and distorted, which is undesired. To preserve the background region, especially the image borders which may cause severe artifacts, we add auxiliary points on the image borders in the warping process. The four image corners plus three points on each border are added to both the original and the reference landmark sets during the warping.

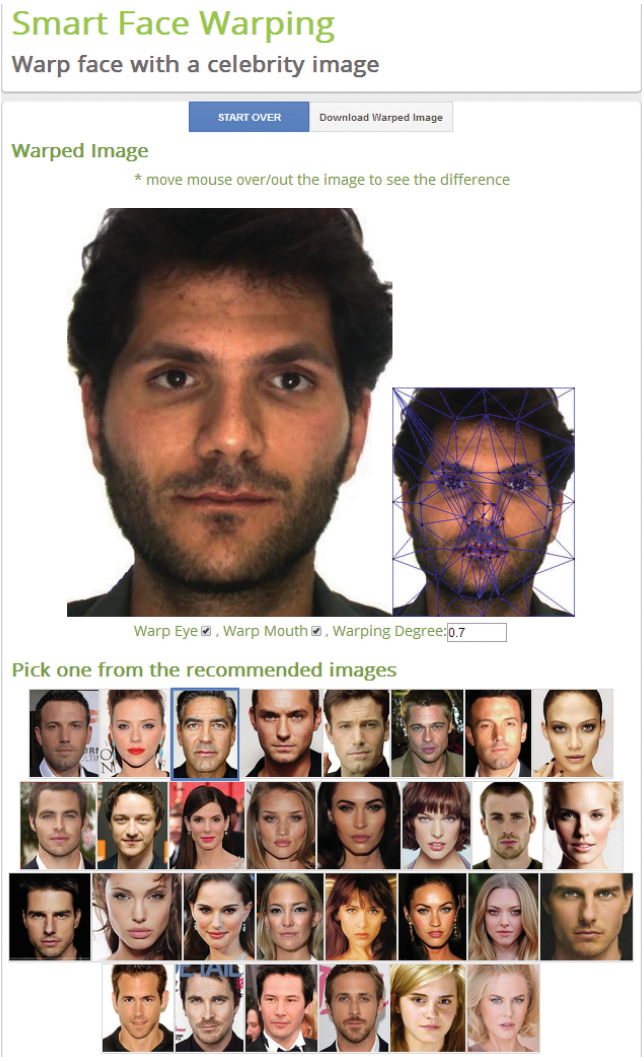

Fig. 3. A screenshot of our web-based smart face morphing system. The image on the left is the morphing result of an AR face image using a MSRA-CFW celebrity image as the reference. The smaller image by the morphed image shows detected facial feature points (blue) and desired facial feature points (red) superimposed on the original image. Note that we added sixteen points on the image borders to both sets of facial feature points to avoid deformation in the boundary region. In this example, both eyes warping and mouth warping are enabled, and the morphing degree is 0.7 (Color figure online).

An example case of using and not using auxiliary points for border preservation is shown in Fig. 4. We can observe that, without border preservation, the border regions are distorted. If there are straight lines in the background region, they

(a) (b) (c)

Fig. 4. Border preservation avoids distortion in the background. (a) Before morphing. (b) With border preservation. (c) Without border preservation.

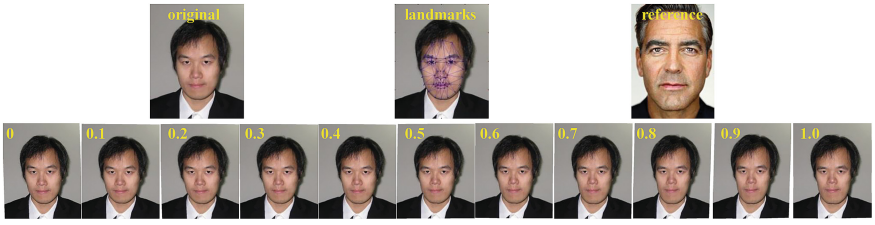

Fig. 5. The morphing degree $1 - \mu$ determines the level of deformation.

will likely be curved and look very unnatural. Adding the auxiliary points does not affect the warping effect in the facial region.

6.4 Running Time

On a 64-bit Windows 8 machine with 2.0 GHz Intel Core i7 CPU and 8 GB memory, provided a 576×768 color image as input, the feature point detection takes about 0.3 s, solving the prior MDS problem with ILMA takes about 0.005 s, and the thin-plate splines warping takes about 0.4 s. On the Amazon web service EC2, the entire morphing process also takes less than 1 s.

7 Conclusion

In this paper, we have presented an online smart face morphing engine. We formulated a prior MDS problem which directly integrates the prior constraints of the original face into the geometry embedding problem, and provided an iterated Levenberg-Marquardt algorithm to find its unique solution. Although the other algorithms in our system are pretty standard, one of our contribution

is to assemble them into a fully-featured publicly accessible real-time web-based system. Currently one limitation of our system is that when the subject in the user-uploaded image wears glasses or other decorations, these objects will very likely be distorted in the final image as the result of warping. In the future, one improvement direction of our system is to re-train the active shape model using a more comprehensive dataset which includes subjects from different ethnic groups and different age groups, and also with different expressions, to ensure more robust facial point detection. Another improvement direction is that we can use GPU to further speed up our algorithms, especially the ASM and the thin-plate splines warping.

References

1. Kagian, A., Dror, G., Leyvand, T., Cohen-Or, D., Ruppin, E.: A humanlike predictor of facial attractiveness. In: Advances in Neural Information Processing Systems (NIPS), Vancouver, B.C., Canada, pp. 649–656 (2006)
2. Eisenthal, Y., Dror, G., Ruppin, E.: Facial attractiveness: beauty and the machine. Neural Comput. **18**(1), 119–142 (2006)
3. Altwaijry, H., Belongie, S.: Relative ranking of facial attractiveness. In: Workshop on the Applications of Computer Vision (WACV), Clearwater Beach, FL, USA, pp. 117–124. IEEE (2013)
4. Leyvand, T., Cohen-Or, D., Dror, G., Lischinski, D.: Data-driven enhancement of facial attractiveness. ACM Trans. Graph. (TOG) **27**(3), 38 (2008)
5. Milborrow, S., Nicolls, F.: Locating facial features with an extended active shape model. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part IV. LNCS, vol. 5305, pp. 504–513. Springer, Heidelberg (2008)
6. Zhou, Y., Gu, L., Zhang, H.J.: Bayesian tangent shape model: Estimating shape and pose parameters via Bayesian inference. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Madison, WI, USA, vol. 1, pp. I–109. IEEE (2003)
7. Bookstein, F.L.: Principal warps: thin-plate splines and the decomposition of deformations. IEEE Trans. PAMI **11**(6), 567–585 (1989)
8. Lee, S., Wolberg, G., Chwa, K.Y., Shin, S.Y.: Image metamorphosis with scattered feature constraints. IEEE Trans. Vis. Comput. Graph. **2**(4), 337–354 (1996)
9. Martinez, A.M.: The AR face database. CVC Technical Report 24 (1998)
10. Zhang, X., Zhang, L., Wang, X.J., Shum, H.Y.: Finding celebrities in billions of web images. IEEE Trans. Multimedia **14**(4), 995–1007 (2012)
11. Cao, X., Wei, Y., Wen, F., Sun, J.: Face alignment by explicit shape regression. Int. J. Comput. Vis. **107**(2), 177–190 (2014)
12. Zhou, E., Fan, H., Cao, Z., Jiang, Y., Yin, Q.: Extensive facial landmark localization with coarse-to-fine convolutional network cascade. In: IEEE International Conference on Computer Vision Workshops (ICCVW), Sydney, NSW, Australia, pp. 386–391. IEEE (2013)
13. Cootes, T.F., Taylor, C.J., Cooper, D.H., Graham, J.: Active shape models-their training and application. Comput. Vis. Image Underst. **61**(1), 38–59 (1995)
14. Milborrow, S., Morkel, J., Nicolls, F.: The MUCT landmarked face database. Pattern Recogn. Assoc. S. Afr. 201 (2010)

15. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Kauai, HI, USA, vol. 1, p. I-511. IEEE (2001)
16. Borg, I., Groenen, P.J.F.: Modern Multidimensional Scaling: Theory and Applications. Springer Series in Statistics, 2nd edn. Springer, New York (2005)
17. Kruskal, J.B.: Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. Psychometrika **29**, 1–27 (1964)
18. de Leeuw, J.: Applications of convex analysis to multidimensional scaling. In: Barra, J.R., Brodeau, F., Romier, G., Cutsem, B.V. (eds.) Recent Developments in Statistics, pp. 133–146. North Holland Publishing Company, Amsterdam (1977)
19. Williams, C.K.: On a connection between kernel PCA and metric multidimensional scaling. Mach. Learn. **46**(1), 11–19 (2002)
20. Levenberg, K.: A method for the solution of certain non-linear problems in least squares. Q. Appl. Math. **2**, 164–168 (1944)
21. Marquardt, D.: An algorithm for least-squares estimation of nonlinear parameters. J. Soc. Ind. Appl. Math. **11**(2), 431–441 (1963)
22. Arun, K.S., Huang, T.S., Blostein, S.D.: Least-squares fitting of two 3-D point sets. IEEE Trans. PAMI **9**(5), 698–700 (1987)