This paper presents a novel system framework of face beautification. Unlike prior works that deal with single images, the proposed beautification framework is designed for an input video and it is able to improve both the appearance and the shape of a face. Our system adopts a state-of-the-art algorithm to synthesize and track 3D face models using blendshapes. The personalized 3D model can be edited to satisfy personal preference. This interactive process is needed only once per subject. Based on the tracking result and the modified face model, we present an algorithm to beautify the face video efficiently and consistently. Furthermore, we develop a variant of content preserving warping to reduce warping distortions along the face boundary. Finally, we adopt real time bilateral filtering to remove wrinkles, freckles, and unwanted blemishes. This framework is evaluated on a set of videos. The experiments demonstrate that our framework can generate consistent and pleasant results over video frames while the original expressions and features are preserved naturally.

**Index Terms**— face beautification, blendshape, image warping

1. **INTRODUCTION**

People use different ways, such as makeup and cosmetic surgery, to make their faces look more attractive. For example, we use foundation to change the skin tone, use concealer to make blemish, wrinkle and freckle invisible. We could also make our cheek thinner, eyes larger and nose more pointed via cosmetic surgery. Nowadays face beautification through picture editing becomes popular as it could be extremely useful to preview and find pleasant face shapes and appearance before applying actual makeup and cosmetic surgery. One kind of beautification algorithms mainly concentrates on automatic makeup transformation without changing face shapes (e.g., [1,2,3]), or modifying the face models (e.g., [4,5]). The makeup transformation could also be accomplished by commercial software such as Photoshop™ or even dedicated hardware such as Casio Exilim EX-TR15 [6], which is limited to appearance changes (such as brightening and smoothing the face skins).

Current algorithms are mainly applying on a single input image. Usually they require user-touch up on a per-frame basis, frontal views, and/or natural expressions. Therefore, they could not be easily extended to work on a video input. In this paper, we propose a novel face beautification system framework that is designed to change both face shape and appearances on a video input.

In this framework, we first choose state-of-the-art algorithm [7] to synthesize and track 3D face models. Comparing with 2D models used in prior works, 3D models could generate more realistic beautification results and are robust to facial expression, pose, and illumination changes in the input video. In order to beautify a face, we choose to allow the user to directly modify the face shapes instead of applying changes based on statistics (e.g., [4]), since we have noticed that the concept of beauty does vary from person to person. Given a beautified 3D model, we develop an algorithm to beautify all the video frames efficiently and consistently. Furthermore, we develop a variant of content...
Face beautification algorithms could be divided in to two categories. The first category concentrates on appearance/makeup changes [1, 2, 3]. In [1], Guo and Sim first decompose face images into face structure layer, skin detail layer, and color layer. The skin detail layer contains skin flaws such as wrinkles. The face structure layer contains facial components such as eyes, nose and mouth. The color layer represents color tone. Makeup information is transferred between corresponding layers. Although the 3D face morphable model is used during synthesis in [2], their goal is to generate a textured 3D face model from single images captured under a controlled-light setup. Yang et al. proposed a real time bilateral filtering which time complexity is invariant to filter kernel size [3]. This technique is applied on single face images to remove skin flaws. In our paper, we adopt this implementation in the last step of the framework.

Another category of the face beautification concentrates on shape changes [4, 5]. In [4], Leyvand et al. compute a set of distances between 2D facial feature points. These distances could be considered as a high dimensional point in a face space. The face space also contains training points that are computed from a set of face images that have been rated offline. Given a new point, they search for a nearby point in the face space that has a higher attractiveness rating. Chou et al. apply the ASM model to align 2D face image and use the Poisson image editing technique to insert a facial component to the target image [5]. This approach is only suitable for one single image.

There are also other similar techniques in which the purposes are not face beautification. For example, Dale et al. propose an algorithm to replace faces in video [8]. This algorithm uses a 3D multi-linear model to track the facial performance. The source video is warped to the target video after retiming and blending. As two videos contain two users with totally different face shapes and appearances, beautification is not the goal of this algorithm and users are not allowed to beautify their faces according to their own preferences. In [9], the goal of the proposed algorithm is to manipulate (e.g., magnify and suppress) facial expressions by adjusting expression coefficients. This adjustment is somehow related to our first step of the framework. However, our algorithm is different from the algorithm in [9] and is more suitable for face beautification.

The algorithms for face synthesis and tracking step in our framework are the state-of-the-art algorithms described in [7, 10]. The core part is the rigid and non-rigid tracking of the blendshape weights and position of head and its orientation. A statistical model is also proposed in [7] to prevent unrealistic poses by regularizing the blendshape weights. These algorithms have been integrated in the commercial software [11].

The content preserving warping algorithm proposed in [12] is modified and improved for the purpose of the face image warping. The original algorithm is designed to capture arbitrary scenes from a hand-held camera.

3. OUR VIDEO-BASED SYSTEM

In order to beautify faces in a video input, we propose a system framework as illustrated in Figure 2. First, we use Kinect sensor to collect data. The Kinect sensor supports simultaneous capture of a 2D image and a depth map at 30 frames per second. The user is instructed to perform a set of different expressions in front of the Kinect sensor. Second, as the depth maps often exhibit high noise levels and missing data, we processed the depth maps offline with algorithms proposed in [7] to generate a set of personalized blendshapes. There are 49 blendshapes, with one neutral expression and 48 other expressions such as eye blink, smile, etc. The user’s expression is reconstructed by a linear PCA model based on the blendshapes.

\[
S = \tilde{S} + \sum_{t=1}^{M} \alpha_t \tilde{S}_t
\]

where \(S\) is the target face expression, \(\tilde{S}\) is the mesh of neutral expression, \(\tilde{S}_t\) is the additive displacements between \(i_{th}\) blendshape and neutral expression, and \(\alpha_t\) is the blending weight corresponding to \(i_{th}\) blendshape.

In the third step, we beautify the personalized blendshapes. One possible way to beautify these blendshapes is to manually edit all of them. However, this process could be very time consuming and the edited blendshapes may not be consistent to each other. In Section 3.1, we describe an algorithm to beautify the blendshapes that only requires manual editing of a few blendshapes.
By applying the linear PCA model on the beautified blendshapes, we are able to reconstruct a beautified 3D face mesh for every image frame in an input video. The weights \( \alpha_i \) in the linear PCA model are estimated by the rigid and non-rigid tracking proposed in [10]. The rigid tracking is mainly done by ICP with point-plane constraints and computes position of the head and its orientation. The non-rigid tracking estimates the blendshape weights \( \alpha_i \).

In many existing algorithms, accumulated facial texture is often mapped to the 3D mesh to reconstruct a complete 3D face model. This is a necessary step for 3D face animation. However, our goal is to generate a new video with the beautified face. Therefore, in the fifth step, we project the 3D face meshes back to the image frame using the position of the head and its orientation estimated from the rigid tracking. Then we apply an image warping between the original and beautified 2D face meshes. We notice that a direct warping would often cause some unwanted distortions around the face boundary. Hence, we design an image warping algorithm to minimize the distortions around the face boundary (Section 3.2). As a result, the image background and the beautified face can be composited together seamlessly.

In the last step, we apply filtering on the image sequences to remove wrinkles, freckles, and unwanted blemishes (Section 3.3).

### 3.1. Beautification of 3D Face Mesh

Our goal of the algorithm is to estimate \( M = 49 \) beautified blendshapes efficiently. A beautified face mesh \( S' \) is also modeled by the linear PCA,

\[
S' = \mathcal{S} + \sum_{i=1}^{M} \alpha_i S_i
\]  

(2)

As the difference between \( S' \) and \( S \) could not be very large, we approximate \( \alpha_i' \) by the original weight \( \alpha_i \) in equation (1). Our experiments show that the approximation is reasonable and can generate stable results. After manually editing the neutral expression \( \mathcal{S} \) and target expression \( S' \), the 48 additive displacements are unknown and need to be estimated. Here we first divide the face mesh into five different regions (i.e., left eye, right eye, nose, mouth, and chin regions). A linear relation is used to model the transformation between \( \mathcal{S} \) and \( S_i \) for each local region.

Thus, equation (2) is converted to,

\[
S'_j = \mathcal{S}'_j + \sum_{i=1}^{M} k_{ij} \alpha_i S_i
\]  

(3)

where \( \alpha_i' = \alpha_i \), \( S'_j = k_{ij} S_i \), and \( j \) is the index of local region. We set \( \beta_{ij} = k_{ij} \alpha_i \), and \( \beta_{ij} \) is solved by minimizing the energy function for each local region,

\[
E_j = \left\| S'_j - \mathcal{S}'_j - \sum_{i=1}^{M} \beta_{ij} S_i \right\|^2_2 + \lambda \sum_{i=1}^{M} (\beta_{ij} - \alpha_i)^2
\]  

(4)
where the second term is the regularization term that makes \( k_{ij} \) close to \( \alpha_i \) to prevent unrealistic transformation, and \( \lambda \) is the parameter to balance regularization and data fitting.

However, when \( \alpha_i \) is zero or very small, \( k_{ij} = \beta_{ij}/\alpha_i \) is infinity or highly unstable. To deal with this problem, we first increase the number of target expressions \( S' \). Two or three different target expressions \( S' \) could greatly reduce the number of \( \alpha_i \) that is zero or close to zero. For the remaining cases, we simply set \( k_{ij} \) to 1 to prevent unrealistic changes. As two or more local regions have a large overlapping area, we use the mean value of multiple \( k_{ij} \) when updating \( S'_{ij} \). Figure 3 shows some beautified 3D face meshes.

We manually edit the neutral expression \( \tilde{S} \) and few target expressions \( S' \). Mesh editing is a well-studied topic in computer graphics and has been integrated into many modeling software. We choose a simple and free 3D modeling editor kHED [13]. Other commercial software also could be used here. Since the interactive Laplacian mesh editing (e.g., [14]) is often used in many of them, the edited results could be smoother.

### 3.2. Face Image Warping

After projecting the original and beautified 3D face meshes back to the image frame, we obtain two corresponding 2D face meshes \( \tilde{P} \) and \( P \). In order to warp from \( \tilde{P} \) to \( P \) naturally without causing obvious distortions along the face boundary, we first define an image warping patch that is larger than the face region, which could reduce the warping artifacts along patch boundaries. We then design a variant of the content preserving warping algorithm proposed in [12].

The original image patch is divided into a \( n \times m \) uniform grid mesh \( \tilde{V} \). The warping problem is then converted to finding warping version \( V \) of this grid mesh. The warping is formulated as a linear least squares problem in [12]. The energy function is defined as

\[
E = E_d + \alpha E_s
\]  
(5)

\( E_d \) is the data term that is defined as

\[
E_d = \sum_i \| w_i V_i - P_i \|^2
\]  
(6)

This data term assumes bilinear interpolation coefficients \( w_i \) for each face point \( P_i \) remain unchanged after warping. \( V_i \) represents the four grid vertices that enclose \( P_i \).

\( E_s \) is the similarity transformation term defined as

\[
E_s(V_1) = \| V_1 - V'_1 \|^2
\]  
(7)

\( V_1 \) is a vertex of one grid cell and \( V'_1 \) is another version of \( V_1 \) that is represented by a linear combination of vectors between the other two neighboring vertices \( V_2 \) and \( V_3 \),

\[
V'_1 = V_2 + u(V_3 - V_2) + v \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} (V_3 - V_2)
\]  
(8)

where \( u \) and \( v \) are the coordinates in the local coordinate system. The salience term defined in [12] is omitted in our implementation.

This term assumes that the transformation between each pair of local grid cells is close to a similarity transformation. This could be a reasonable assumption when the grid cell is relatively small. Details of these two terms could be found in [12].

In practice, however, we find that these two terms are not enough to reduce the distortions around the face boundary. Hence, we add another term \( E_b \) to fix the vertices outside the face region.

\[
E_b = \sum_{i \in S} \| V_i - \tilde{V}_i \|^2
\]  
(9)

where \( S \) is the non-face region in the image patch. The term is used to reduce the transformation outside the face region. Along with the previous two terms, our energy function is defined as,

\[
E = E_d + \alpha E_s + \beta E_b
\]  
(10)
where $\alpha$ and $\beta$ are the parameters to control weights of the second and the third terms. Figure 4 shows the comparison of warping results with and without adding the third term.

### 3.3. Filtering

We apply real time $O(1)$ bilateral filtering proposed in [3] to remove wrinkles, freckles, and unwanted blemishes. Unlike other bilateral filtering algorithms, this algorithm could have arbitrary spatial and range kernels and run in constant time. The basic idea is to decompose the bilateral filter into two sets of spatial filters as pixel intensity is discrete. Spatial filters (e.g., box filter and Gaussian filter) can also be computed or approximated in $O(1)$ time. Once the 2D face mesh is generated we only apply the bilateral filtering inside the face region to avoid undesirable smoothing on other regions.

### 4. EXPERIMENTAL RESULTS

We evaluate our proposed system on six different users including different genders and races. Each user is instructed to perform a set of expressions at the beginning, which are used to generate personalized blendshapes. Beautified blendshapes are generated based on the algorithm described in Section 3.1 and the user’s preferences. Then we collect a short video for each user and convert it to a new video that contains beautified faces.

Figure 5 shows the image frames selected from videos for six different users before and after beautification. We can easily find that both face shape and skin region are more

---

**Figure 5.** Results of our face beautification system. (Top) six images are original frame from videos. (Middle) images are the results after shape modification. (Bottom) images are the outputs with shape changed and skin smoothed.

**Figure 6.** The different results on cheek size and skin smooth level. (a) and (d) are original image frames from videos. (b) and (c) show the different cheek sizes for the same person. (e) and (f) show the difference skin smooth levels.
attractive after beautification. The attraction is defined according to the user’s preferences. On average, users prefer to make cheek thinner, eyes larger, and skin smoother. However, different users still could have different definitions of beauty. Figure 6 shows different results (i.e., different cheek sizes and smoothing levels) of the same user.

Figure 7 shows a set of image frames selected from a user’s video. We can see that the shapes and color appearances are changed consistently over the video. **Video demo is presented in the supplemental material.**

In terms of processing time, the face tracking part [11] is still offline. The most time-consuming part is the interactive editing of the face shape. It is usually an iterative process that takes 20-30 minutes. The warping and blending process can run at interactive rate.

5. CONCLUSION

The major contribution of this paper is the novel system framework to beautify face videos. In this framework, we adopt the state-of-the-art algorithms and further propose a beautification algorithm to change a large set of personalized blendshapes and a variant of content preserving warping algorithm. Our experimental results demonstrate the effectiveness of the framework. Looking into the future, we plan to develop a better user interface to facilitate quick and easy 3D face changes using semantics, and explore the option for user to choose more attractive faces under different expressions and apply them seamlessly to videos.

**ACKNOWLEDGEMENTS**

This work is supported in part by US National Science Foundation award HRD 1345219, US National Science Foundation grants IIS-1208420, and Natural Science Foundation of China (No.61173105, 61373085, 61332017).

6. REFERENCES