A Virtual Makeup Augmented Reality System

Aline de Fátima Soares Borges
Institute of Mathematics and Statistics (IME)
University of São Paulo
São Paulo, Brazil
alinefsb@ime.usp.br

Carlos H. Morimoto
Institute of Mathematics and Statistics (IME)
University of São Paulo
São Paulo, Brazil
hitoshi@ime.usp.br

***Abstract*—Virtual makeup systems allow users to try makeup on remotely, without wasting products or spending time cleaning up later. Some virtual makeup systems just show the final makeup result on the user's face, while other systems allow some interaction when applying makeup on a photo. In this paper we introduce an augmented reality system that allows users to apply virtual makeup directly on their face using a physical applicator, simulating a virtual mirror experience. Facial features are detected and tracked using an RGBD camera and mapped to a normalized 2D facial mesh composed of 124 triangles. Finger touches on the face are also detected on the RGBD video stream, and used to store the applied makeup texture representation on the corresponding 2D facial triangle. Rendering the face with the virtual makeup is performed by back-projecting the makeup stored on the facial mesh to the image captured by the camera. Our initial prototype demonstrates the feasibility of our technique, that detects touches very accurately (about 2.2 mm), and that achieves real-time interactive performance (about 15 fps) when tracking and rendering makeup using a regular PC and an Intel RealSense RGBD camera.**

***Index Terms*—Virtual Makeup, Augmented Reality, Interactive Video Processing**

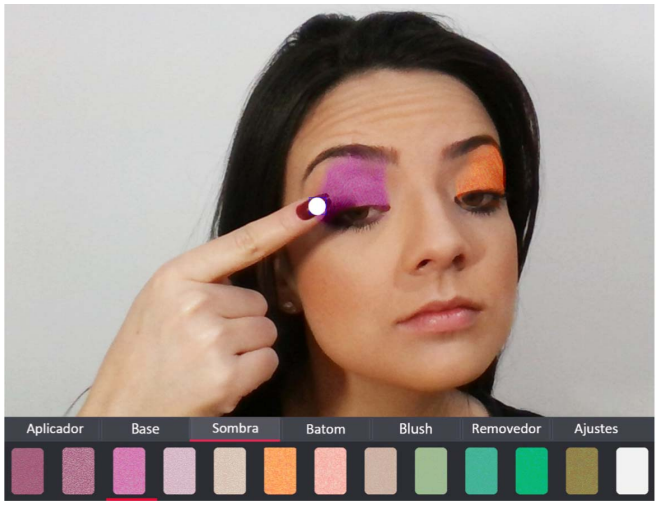

Fig. 1. User experimenting the virtual makeup augmented reality system.

I. Introduction

For centuries, makeup products have been used by the most diverse societies and are today part of many people's daily lives. Facial makeup is a way to change the facial appearance by modifying optical properties and texture of the skin and can be used in different contexts, such as to make people more attractive, to create a personality of an artist or a model, for medical purposes, or just for entertainment. However, the experimentation of real facial makeup requires the use and waste of many products, such as the makeup itself, cleansing products, applicators, etc. Thus, a virtual makeup application system, as seen in Figure 1, using augmented reality (AR) offers several advantages to the experimentation process, allowing the user to visualize the results quickly and cleanly, even remotely, without any waste.

Commercial examples of virtual makeup application systems are ModiFace and YouCam Makeup. Similar systems have been the target of investments from many makeup manufacturers such as Natura, Avon, Clinique, Shiseido, and others. However, despite interesting results, there are still several challenges and limitations, particularly with regards to the lack of rendering realism and poor user interaction experience.

One important limitation is the use of the mouse or finger touching on a touch sensitive screen to apply the virtual makeup. Though the face displayed on the screen is the user's own face, the user must act as putting makeup on someone else's face. In this paper we apply a digital mirror metaphor to explore the AR potential of augmenting the user's face with makeup, to offer a more natural and enjoyable user experience.

The remainder of this paper is organized as follows. In section II we present a brief review of the literature about virtual makeup systems. Section III shows the proposed system design. Section IV presents the results of evaluation experiments and Section V concludes the paper.

II. Related Work

In this section, we present some relevant methods proposed in the literature about the use virtual makeup and how they can be applied on a user's face. In addition, considering facial touch detection, we present papers that deal with hand position estimation and interactive surfaces.

Makeup transfer methods focus on the makeup realism and are non interactive. Given an image A containing a face F without makeup; an image $A*$ containing a face F with makeup M; an image B containing a face G without makeup, the process of makeup transfer is defined as a composition of an image $B*$ containing G with the makeup M, the same way that F is makeup at $A*$ [1].

978-1-7281-5434-3/19/$31.00 ©2019 IEEE
DOI 10.1109/SVR.2019.00022

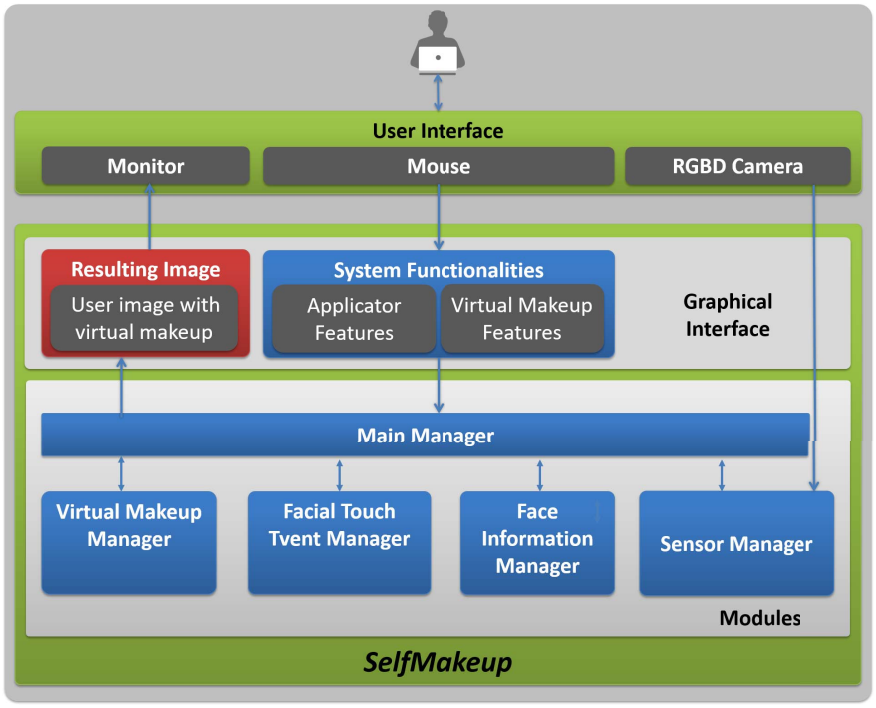

Fig. 2. Block diagram of a Self Makeup System architecture composed by UI components, graphical interface and processing modules.

Tong et al. [2] presented a method to makeup transfer using two images of the same person as a model to learn the makeup effects; one without makeup (A) and the other with makeup A^*. Both images are assume to have the same light conditions and facial pose. To perform makeup transfer to an target image B with realism, [2] proposed that features such as freckles on the skin should not be transferred from the model to the target and that such imperfections if present on the skin of the person in target image, should remain. Then, before transferring the makeup, the eyebrows, eyelashes, blemishes and imperfections of the skin are extracted of all three images (A, A^* and B). The faces of the three images are deformed to accommodate in a canonical face model according to the *Thin Plate Splines* method. Assuming the skin of the face as a Lambertian surface, the makeup mapping is performed pixel by pixel, using the same 2D facial geometry, resulting in the image B^*. In the last step, the facial characteristics extracted in the pre-processing are transfer back to B^*.

The method proposed by Guo and Sim [3] differs from the method of [2] by using just one image A^* as model and by decomposing the images A^* and B into layers. First, 83 facial control points are detected using an *Active Shape Model (ASM)* and are aligned using the Thin Plate Splines method, deforming A^* and B. Then, A^* and B are split into two layers of color and of luminosity. Therefore, initially the images are converted to La*b* space color, where L is used to luminosity and a* and b* to color. In the following, the luminosity layer is split into one layer of the face structure and another layer of skin details. When compared to the method of [2], the method of Guo and Sim [3] shows the following advantages: they are more faithful to the style of makeup of the model; present more vivid colors; improve the preservation of the facial structure; and improve the appearance of the lips.

The methods of [2] and [3] are not sufficient for the implementation of the virtual makeup application method of this study. Such approaches depend on model images with makeup applied to the face and require that the face pose of the target image be as close as possible to the face pose of the model image. In addition to these undesirable requirements, they do not present a generalizable way to simulate the effect of makeup with physical products on the image of a person's face.

A. *Virtual makeup application*

Virtual Makeup application methods eliminate the use of model images. Dhall et al. [4] present a procedure for application of automatic makeup based on a person's gender and skin color. This procedure starts with the segmentation of the face in the image by skin color. Viola-Jones [5] and *Active Shape Models* [6] are used to to find the eye and mouth regions. Then, to remove marks on the skin, a Gaussian softening is applied followed by a dilation. Then the image receives the makeup, which is applied by modifying the hue and saturation values of the HSV color space.

The method proposed by Kim and Choi [7] focuses on applying makeup by combining the makeup color desired with the colors of the skin texture, using the weighted average of the facial skin color with the makeup color, using a constant

for the makeup intensity. However, the methods of [4] and [7] are limited in calculating a weighted average to simulate the effects of makeup on the skin and do not take into account facial features, implying an unrealistic result.

A method to simulate the makeup application by rendering a 3D facial model was proposed by Huang et al. [8]. It considers the light interaction with skin and cosmetic product features. A special equipment called Light Stage X was used to capture the geometry and appearance of the face and makeup product models. After building the 3D facial model, this system allows the application of foundation, blush and shadow, in addition to modifying applied makeup in real-time. To do so, they used the Kubelka-Munk model [9] to calculate the total transmittance when light passes through the cosmetic layers and the *screen-space* method [10] to simulate scattering of rays of light on human skin.

A system for simulating makeup procedures on rendered avatars was presented by Jang et al. [11]. The goal was to make the virtual makeup color more realistic given a skin tone. Capturing images by the system was done with a 3D scanner. Initially, color calibration of the system occurs with the use of LEDs from the three cameras of the scanner. Then the avatar's skin colors based on skin and cosmetic spectrum data is estimated, after applying specific cosmetics.

The methods of Huang et al. [8] and Jang et al. [11] present interesting results for rendering virtual makeup. Nonetheless, because they employ special equipment to scan the skin and makeup products to obtain the optical properties, it is not possible to use them in real-time interactive system for applying virtual makeup.

B. Interactive virtual makeup systems

While some authors focused on the realism of rendering the virtual makeup, a few systems were also proposed with particular attention to user interaction. Kim and Choi [12] presented an interactive system to cosmetic makeup applied to a 3D face model by using a haptic device, but limit the user experience by not considering that in the real world self-makeup scenarios the user touches his own face to apply makeup.

Some authors have considered real-world makeup self-application scenarios to provide the user interface. Iwabuchi [13] presented the system called "Smart Makeup Mirror" (SMM), composed of a high resolution camera and infrared (IR) sensors, with functionalities to automatically zoom the facial image, simulate illumination conditions, and publish the result on the Internet. The automatic zoom is performed by detecting a marker fixed at a real makeup product and processing the information about the distance of the user in relation to the IR sensors. However, SMM does not render digital makeup. Nakagawa et al. [14] proposed the "Smart Makeup System" to help users find new makeup methods using their own makeup products.

A "Sensory Augmented Smart Interaction Mirror" (SIM) to display an augmented 3D makeup face was proposed by Rahman et al. [15]. To allow users to choose makeup products, SIM contains an RGB camera to capture the facial image and an IR camera to track the hands and facial movements through the IR emitters. Experiment results showed that some users were unhappy about the visual quality of the system. In addition, the devices used in the interface and the system calibration process limit the application scenarios.

Campos and Morimoto [1] presented a smart mirror that allows users to apply virtual makeup by touching the screen showing the user's face. Their method decomposes the facial image in layers, as an improvement of [3] through the elimination of a facial model image containing the virtual makeup. However, [1] does not consider the shading and highlighting effects caused by makeup, which reduces the rendering realism.

C. Hand pose estimation and interactive surfaces

A number of studies have been proposed to detect and track known markers in makeup applicators such as [13]–[17]. The use of color markers or special gloves to track the hand has been addressed by the investigations of [18]–[20]. The use of known objects or bare hand in contrast with background scenario was present by [21]–[27]. However all of these approaches present limitations regarding the context of this study since the use of extra tools or markers might lower the user experience, and object segmentation using background extraction is not useful because the face is also moving.

With more consumer depth sensors available, a number of distinct methods have been developed to hand tracking and hand pose estimation by processing the depth and color image streams such as [28]–[31]. However these methods not work when the hand is too close to the face.

III. System Design

Based on an abstraction of real-world self makeup scenarios, the following requirements were considered during our design process:

- Display the virtual makeup products and allow the user to select one of them;
- Detect the facial position touched by the user;
- Capture the image of user face in real-time and render the virtual makeup application results.

We established a conceptual model based on the mirror metaphor as a central component of the *SelfMakeup* design. Using this metaphor, the user applies a makeup product by touching his own face using a physical applicator. This conceptual model allowed the definition of the *SelfMakeup* architecture as shown in Figure 2, which is composed of the user interface (UI) and three key processing modules: **Face Information Manager**, **Facial Touch Event Manager**, and **Virtual Makeup Manager**.

Besides the main modules, two auxiliary modules have been implemented. The **Sensor Manager** module is responsible for buffering the captured RGBD data and provide it to other processing modules. Also, the **Main Manager** was idealized to manage and synchronize the information and communication with the other modules.

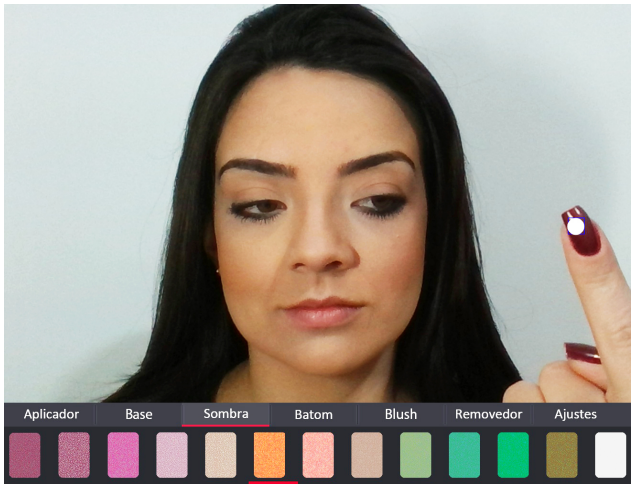

Fig. 3. Applicator tip detected and the user starting the virtual makeup application.

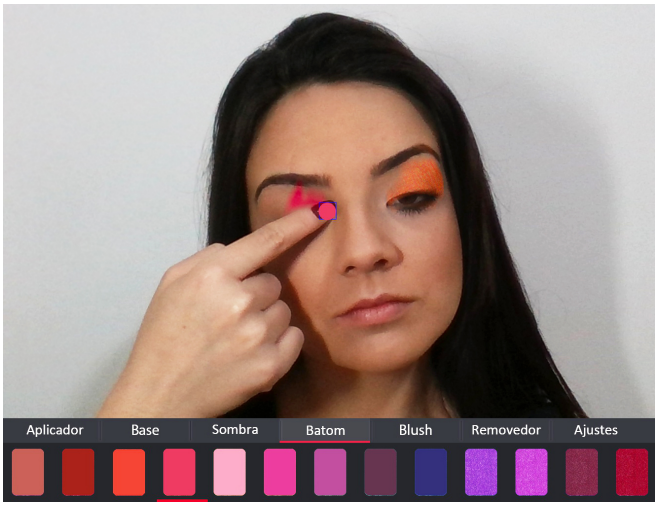

Fig. 4. Applicator tip touching the face to apply a virtual makeup product.

A. System overview

The main idea to construct the 2D facial map is to divide the face into regions, and each region is decomposed into triangles. During user interaction with the system, the information of virtual makeup stored in each triangle can be deformed and aligned according to the user's face position and facial expression. The 2D normalized facial map eliminates the need for system calibration to construct a 3D facial model of the user.

The **Facial Touch Event Manager** estimates the position of a finger touch on the face. Initially, the applicator (finger) is assumed to be the closest object to the camera and it is not in contact with the face. The color and depth images are used to estimate the touch position of the applicator tip over the face by detecting and tracking the face and the applicator in multiscale color and depth images. Once the facial touch event is identified, the information is stored in coordinates (u, v) of the normalized 2D facial map.

The **Virtual Makeup Manager** module was designed after investigating the methods of transfer and application of virtual makeup. We verified that the method proposed by [1] presents some realism but lacks the effects of shading and prominence of the virtual make-up that could be remedied with some features of the method of [3] for treating the facial image's luminosity layer.

Our method for rendering the virtual makeup extends the method described in [1] with shading and prominent features similar to the method described in [3] for makeup transfer.

B. Graphical User Interface

The current implementation of the UI allows the user to try on blush, eyeshadow, and foundation. The UI components assume a typical desktop computer composed of a monitor with a keyboard and mouse, with the extension of an RGBD camera. The **monitor** allows the user to visualize the result of virtual makeup application and the virtual makeup products available. The **mouse** is used to select a virtual product and could be substituted by touch on a touch sensitive monitor. The **RGBD camera** is used to capture the color and depth images.

To begin interacting with this system, users must position their face in the range of 30 cm to 80 cm in relation to the RGBD camera, which is placed on top or bottom of the computer screen used as "mirror".

The interface shows a product menu that allows the user to select virtual makeup products such as foundation, blush, and shadow. For each of these products there are several colors that can be chosen. In addition, the user can select the applicator features such as the size, the intensity and the edge type. After setting up the applicator and choosing a product color, the user can start the application of the product by simply touching his/her face with her/his finger, or any other stick that resembles makeup brushes. When the system detects the tip of the applicator, a white circle is displayed at the position of the corresponding image, as shown in the figure 3.

The user can move the applicator and touch the face to apply the chosen virtual makeup. When a facial touch event is detected, the applicator circle turns the same color as the selected makeup, indicating that the product is being applied. Figure 4 illustrates this scenario in which the tip of the applicator is touching the face. Makeup is applied only when the system detects that there is contact of the finger with the face, as described in Section III-D.

C. Face Information Manager

The **Face Information Manager** is responsible for the real-time detection and tracking of the face and facial features. In addition to the head motion, the face can assume a diversity of expressions. For the virtual makeup to become head pose and facial expression invariant, our system relies on a 2D normalized facial map that stores the applied virtual makeup.

Our current implementation uses an Intel RealSense RGBD camera with full HD RGB resolution. The RealSense SDK contains routines to detect and track faces and 78 facial landmarks that include the location of the mouth, eyebrows, eyes, nose, face contour, and eye pupils. The two landmarks corresponding to the pupils are not used and our method. We have extended the set of landmarks with 15 additional points based on human face anthropometric data around the forehead and nose contour.

We represent coordinates in a normalized 2D map by (u, v). The facial map M_{map} is subdivided into 8 regions (R_{face}) which correspond to the right cheek, left cheek, forehead, nose, lower lips of mouth and chin, upper lips and region between mouth and nose, right eyelid, and left eyelid. According to the topology of the face, each $r_i \in R_{face}$ is subdivided into a number of triangles until the desired resolution is reached. Our current implementation of M_{map} contains 124 triangles, as shown in Figure 5.

For each input frame, a 2D facial map F_{map} is generated containing the same structure of points, triangles, and facial regions as M_{map}. That is, for each point $p_i(u, v) \in M_{map}$ there is a corresponding point $p'_i(x, y) \in F_{map}$, for each triangle $t'_i \in M_{map}$ there is a corresponding triangle $t_i \in F_{map}$ and for each facial region $r_i \in M_{map}$ there is a corresponding region $r'_i \in F_{map}$. Coordinate (x, y) represent pixels in the RGB image.

When a touch is detected on a triangle in the F_{map}, the corresponding triangle in the M_{map} updates its stored makeup, as shown in Figure 6. Similarly, for rendering the virtual makeup on the users face image, affine transformations are used to map the triangles in M_{map} back to F_{map}.

D. Facial Touch Event Manager

The iterative process of identifying the facial touch event initiates by capturing color images I_c and depth images I_d from the RGDB camera. These images are downsampled by constructing an image pyramid with two levels. The lowest resolution images I'_c and I'_d are used for further processing in real-time.

The depth image I'_d is mapped to I'_c using the sensors intrinsic parameters such that the depth of a pixel $p(x, y) \in I'_c$ can be accessed using $I'_d(x, y)$. For each frame, the face position is identified in I'_d and then, the average distance θ_f between the face and the RGBD sensor is calculated using (1), where $d_i \in I'_d$ represent the 91 facial landmarks provided by the **Face Information Manager** module.

$$\theta_f = \frac{1}{N}\sum_{i=1}^{91} d_i \tag{1}$$

The processing of I'_d is performed to verify when the applicator is over the face and if it is touching the face. Using I'_d and the information inherent to the face position, the applicator is assumed to be the closest object to the camera and its depth θ_a is defined as $\theta_a < (\theta_f$-50). The value 50 has been empirically established to define a region-of-interest (ROI) and is used to eliminate a large number of pixels belonging to the user's head and torso and to the background scene. Then, the pixels corresponding to the applicator form an image of depth I'_{da}, which is estimated using (2).

$$I'_{da} = \{p(x, y) \in I'_d | p(x, y) < (\theta_f - 50)\} \tag{2}$$

The remaining pixels are then refined using an erosion morphological operation followed by a dilation to eliminate outliers due to camera noise around the contours. We have used an elliptical structuring element of size 3×3.

After the dilatation operation, the remaining pixels of I'_{da} constitute the *blob* corresponding to the applicator. We consider that the applicator tip is cylindrical with approximately 15 mm of diameter and is about 300 mm in front of the sensor. Top most point of I'_{da} is assumed to be the top point p_{at} of the applicator tip. This procedure allows us to initiate the tracker of the applicator tip.

We have used the *Kernelized Correlation Filters (KCF)* algorithm for tracking the applicator tip. *KCF* is an approach that uses the frequency domain of the image to create correlation filters, which filter frequencies of consistent parts of the training images and provide invariance to the displacement. These features enable real-time performance, making tracking faster than using *Tracking,Learning and Detection (TLD)* or *Multiple Instance Learning (MIL)* [32].

KCF requires a kernel constructed from a target image. The size $s_k(a_d, a_d)$ of the *KCF* kernel is established according to the applicator diameter a_d selected by the user. The *KCF* kernel K_{ap} is created using (3).

$$K_{ap} = \{p(x, y) \in I'_c | (x_{p_{at}} - l/2) < x < (x_{p_{at}} + l/2) \cap (y_{p_{at}}) < y < (y_{p_{at}} + l)\} \tag{3}$$

The *KCF* search window is defined using I'_c with size $s_w(2 * a_d, 2 * a_d)$, i.e., using twice the size of $s_k(a_d, a_d)$. The search window contains the region in I'_c that corresponds to the applicator tip. This search window W_{ap} is established according to (4).

$$W_{ap} = \{p(x, y) \in I'_c | (x_{p_{at}} - l) < x < (x_{p_{at}} + l) \cap (y_{p_{at}}) < y < (y_{p_{at}} + l * 2)\} \tag{4}$$

After building K_{ap} and W_{ap}, the applicator tip tracking is initiated. For each frame, the tracking updates the applicator tip position. Then to each frame is verified if the K_{ap} is over the face. If the the applicator tip is not over the face, every ten frames I'_{da} is processed to check if it contains the *blob* of the applicator. This step is a way to recover from possible faults or in scenarios where the user is moving the applicator but does not want to use it.

When the applicator tip is over the face, the maximum depth $d_f = max(K_{ap})$ of the region corresponding to the applicator tip is considered to belong to the face. The depth of the applicator d_{ap} is considered to be the distance of the center point of K_{ap}. Then, the facial touch is detected if

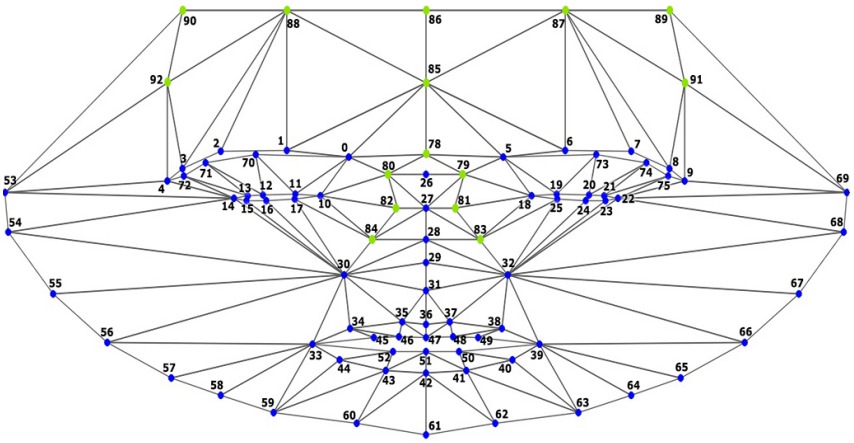

Fig. 5. Facial model map M_{map} containing 124 triangles in the normalized space. The blue points are the points provided by the RealSense SDK. The green dots are extended points from anthropometric data.

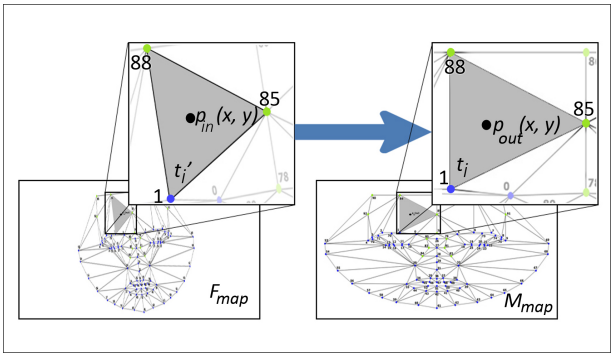

Fig. 6. Process for transforming the coordinates of a point $p_{in}(x, y)$ of F_map to a point $p_{out}(u, v)$ of M_{map}. In this image, p_{in} is situated in t_i', which is formed by vertices representing the fiducial points 1, 88 and 85 and has corresponding t_i. Thus, the coordinates of vertices 1, 88 and 85 of F_{map} and M_{map} are used to obtain the affine transformation that will be applied to $p_{in}(x, y)$ to find the coordinates of $p_{out}(u, v)$.

$(d_f - d_{ap}) < 20mm$. Once the applicator is touching the face, the next step is to estimate the corresponding position in the normalized space. This verification is performed using the transformation function detailed in Section III-C, passing as parameter the center point of K_{ap} in coordinates (x, y). This function returns the point $p_{touch}(u, v)$, in normalized coordinates (u, v).

E. Virtual Makeup Manager

In the context of this study, applying a virtual makeup product means altering the optical and texture properties of the image of the face, modifying the way that the facial image is perceived. For each virtual makeup product, we consider a small texture element we call *maquilet*, which consists of an RGBA image containing information about the hue variations of that product.

Once a product is first selected, a mask m_{ask} is created, initially filled with a value of 0. A mask for each product is used to store the information of the facial regions where that product has been applied. Each m_{ask} is in normalized facial space coordinates.

The filling of a m_{asc} depends on the type of event triggered by the user. The event can be:

- the tip of the physical applicator has just touched the face;
- the tip of the physical applicator was already touching the face but it changed position;
- the tip of the physical applicator is no longer in contact with the face;

When one of the first two event types are detected, the mask is updated with the product information within a region of radius $r_{applicator}$ centered at the point $p_{touch}(u_c, v_c)$ that corresponds to the center of the applicator tip. Observe that different applicators might have different radius. We also model an attenuation factor $\alpha_{applicator}$ per applicator, to simulate different application intensities.

When the user removes the tip of the physical applicator from the facial surface, it is detected that the applicator is no longer touching the face and the system stops updating the mask.

Once the mask has been loaded with the selected product information on the regions that had been touched, a Gaussian filter is applied on m_{asc} to smooth edges. The smoothed m_{asc} is used to add the *maquilet* to the overall texture $E_{maquilets}$, which contains all the *maquilets* already applied.

To render the combined virtual makeup $E_{maquilets}$ registered to the user's face in the input video stream, we first perform the mapping of the virtual makeup to image coordinates respecting the pose and facial expression. This mapping is accomplished using the function defined in the **Face Information Manager** module passing as parameter $E_{maquilets}$ and getting $T_{maquilets}$ containing the positions of the virtual makeup aligned with the pose and expression of the user's face in the current video frame.

To render the face with makeup using $T_{maquilets}$, we use the following 5 layer model similar to [1].

- C_{fb}= low frequency layer, which contains the overall characteristics of the user's face, such as basic strokes;
- C_{f3}= intermediate frequency layer 3, which contemplates the less subtle details of the user's face, such as varying skin color tone;
- C_{f2}= intermediate frequency layer 2, which is composed of facial details of intermediate intensity and presents some characteristics of skin texture;
- C_{f1}= intermediate frequency layer 1, which contains the finer facial details;
- C_{fa}= high frequency layer, which present facial details such as hairs, hair strands and some contours of facial regions;

This 5 layer model allows the contours of the face and its characteristics to be preserved even if the texture of the skin is altered, in order not to deform the image of the face of the user. In addition, it enables the treatment of individual components of the user's image according to facial characteristics such as color and texture of the skin, lighting, specularity, and facial hair [1]. Thus, by acquiring the original RGB image I_c captured by the camera and the face position data, the processing module crops I_c according to the face position, obtaining $I_{faceRGB}$. Then $I_{faceRGB}$ is split into 5 layers by applying multi-resolution Gaussian filters.

After obtaining these five layers, if the user has applied virtual base makeup, a smoothing is performed in the intermediate layers C_{f1}, C_{f2} e C_{f3}, which are layers that contain skin details. As a result, the skin looks more homogeneous, producing an effect similar to the physical foundation makeup.

The luminosity layer contains important details of the makeup and also includes contour details that are essential for the face characterization. To add to the facial image the highlight and shade details of the makeup without compromising the important details of the face contour, we transform C_{f3}, which is the layer containing the least subtle facial features, from the RGB color space to CIELAB. This enables a better separation of the brightness and color than RGB [3]. Thus, the image $C_{f3La*b*}$ is obtained, which is layered according to the channels La*b*. The image C_{f3L} is acquired from the luminosity channel L.

After computing C_{f3L}, we also transform $T_{maquilets}$ from RGBA to the CIELAB color space. Similarly, the image $T_{maquiletsL}$, from the L channel, contains the highlight and shading effects of the virtual makeup. The $T_{maquilets}$ alpha channel controls the opacity of the virtual makeup for each pixel, defined as $\alpha(x, y)$. The image and makeup is composed in layer C_{f3} using (5).

$$C_{f3L}(x,y) = C_{f3L}(x,y) * (1 - \alpha(x,y)) + T_{maquiletsL}(x,y) * \alpha(x,y) \quad (5)$$

Once it has been processed, the layer C_{f3L} is used to convert $C_{f3La*b*}$ back to RGB, resulting in C_{f3} with the contribution of the luminosity of the virtual makeup. This assumption holds because the illumination of the *maquilets* is assumed uniform.

Next we merge the layers C_{fa}, C_{f1}, C_{f2}, C_{f3}, and C_{fb} resulting in $I_{faceRGB}$, which now contains the contribution of the shading and highlighting effects of the virtual makeup and the smoothing produced by the virtual foundation.

After modifying the user's image by adding the effects related to skin texture, the makeup color information is added to $I_{faceRGB}$. We first transform $I_{faceRGB}$ to the CIELAB color space, and add the corresponding components from the $a*$ and $b*$ channels of $T_{maquilets}$.

Lastly, $I_{faceRGB}$ is placed in I_{RGB}, closing the processing of the **virtual Makeup Manager** module for that frame of the video. The image I_{RGB} is then sent to the graphical interface for rendering the augmented face with makeup to the user.

IV. Experimental Results

The accuracy of real-time face touch detection is an important aspect to offer a more natural and enjoyable user experience. Therefore, experiments were performed in order to evaluate the accuracy of the proposed method. The experiment was conduced with 5 participants (3 female) with ages between 27 and 34 years old.

The color and depth streams were captured using an Intel RealSense RS300 camera connected to a notebook computer (2.4GHz Intel I5 CPU, 12GB RAM). Instead of the notebook screen, a 23" LED monitor with a resolution of 1920×1080 pixels was used. The monitor was placed about 45 cm from the participants face. The experiment was conducted in a room with artificial illumination and no direct sun light.

The system was developed in C++ with OpenCV. Although there are alternatives to Intel's implementation to retrieve information about face position and its landmarks points, we decided to use the one provided by the RGBD camera manufacturer to detect and track the face due to its improved performance when the applicator occludes facial landmarks. To track the applicator tip we used the KCF implemented provided by OpenCV. To avoid the influence of face pose change relative to the camera, the information about facial markers and touched regions was transformed to normalized facial space. The experiments were performed by capturing the Intel RealSense RGB stream with resolution of 1280×720 pixels and depth images with resolution 640×480 pixels, both at 30 fps.

Before the beginning of a session, participants were introduced to the experiment and answered a brief questionnaire. Following the introduction, 9 yellow circles of 3 mm radius were fixed as target points on the participants face: 2 on the right cheek, 2 on the left cheek, 1 on the chin, 1 on the nose and 3 on the forehead. Participants were asked to use their index finger as the applicator in the experiment. Participants performed a short training session interacting with the system to get familiar with the system initialization. Data from the training session was not considered for data analysis.

Participants were asked to move their head naturally to touch each of the target points. Each session began with the system initialization in order to detect the position of the face, the finger tip and the 9 targets on the face. Participant were instructed to touch each target 10 times. The duration of a session was approximately 10 minutes.

For every session the software recorded all 9 target positions and all facial touch event positions in coordinates (u, v) of the F_{map}.

Although participants were asked to touch each target ten times, each participant had their own speed to touch a target, which entailed in a different number of samples collected for each touch event. Thus, for each session, the average position p_{at} of each touch event was first calculated using (6).

$$p_{at}(u, v) = \frac{1}{N} \sum_{i=1}^{N} s(u, v)_i \tag{6}$$

where $s(u, v)$ is the position of each sample i and N is the number of samples. The values of $p_{at}(u, v)$ for each facial touch event are shown in Figure 7.

To interpret the results of our method in millimeters we use anthropometric facial data [33], where the face of an adult female individual is considered to have the height of 137.1 ± 4.3 mm and width of $129,9 \pm 5.3$ mm, while the face of an adult male individual has height of 187.0 ± 8.1 mm and width of $172,5 \pm 7.5$ mm. Then the accuracy of the our approach in estimating the facial positions touched by an applicator tip is computed as the error for each facial touch event, defined as the Euclidean distance between each facial touch position $p_{at}(u, v)$ and the target position $p_t(u, v)$, in millimeters. The grand mean error computed from all touches was 2.14 ± 1.41 mm.

V. Conclusion

This paper has presented an augmented reality system that allows the user to apply virtual makeup directly on his/her face using one finger or any similar shape object. Its interface allows the user to choose the product and other features during the interaction. Our current real-time prototype can simulate the application of foundation, blush, and eyeshadow.

We have implemented a real-time prototype using a regular PC and an Intel RealSense RGBD camera to capture the color and depth images. The user's face and 91 facial landmarks are tracked and mapped to a normalized facial space that is invariant to facial motion and expressions. This normalized space is used to store the virtual makeup applied on the face. The normalized face is represented by a mesh with 124 triangles.

Another contribution of the paper is an accurate real-time face touch detection algorithm, that tracks the face and the tip of the finger in the depth images, and detects touch events when the distance between the finger and face is below a predefined threshold. The touch is mapped to one or more triangles that update their state with the current active makeup texture. Rendering the face with the virtual makeup is performed by back-projecting the facial mesh and combining the makeup texture with the facial texture in the current video frame using the tracked facial landmarks.

Our experimental results show that our touch detection algorithm is very efficient (can run at about 15 fps when tracking and rendering) and accurate (about 2.2 mm). Although rendering realism was not the focus of this work, our system extends the real-time rendering method described in [1] with aspects of the [3] for makeup transfer, to include shading and some prominent features that preserve facial details. When the applicator is not visible (not being tracked), our virtual makeup rendering pipeline runs at 30 fps. Future work include experiments to evaluate the user experience when using our system based on a virtual mirror scenario instead of applying makeup on another person and the inclusion of other makeup products such as lipstick.

Acknowledgments

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001, and the São Paulo Research Foundation (FAPESP), grant 2016/10148-3.

References

[1] F. M. S. de Campos and C. H. Morimoto, "Espelho virtual interativo para simulacao de maquiagem," in *Proceedings of the 13th Brazilian Symposium on Human Factors in Computing Systems*, ser. IHC '14. Porto Alegre, Brazil, Brazil: Sociedade Brasileira de Computacao, 2014, pp. 345–348. [Online]. Available: http://dl.acm.org/citation.cfm?id=2738055.2738113

[2] W.-S. Tong, C.-K. Tang, M. S. Brown, and Y.-Q. Xu, "Example-based cosmetic transfer," in *Proceedings of the 15th Pacific Conference on Computer Graphics and Applications*, ser. PG '07. Washington, DC, USA: IEEE Computer Society, 2007, pp. 211–218. [Online]. Available: http://dx.doi.org/10.1109/PG.2007.21

[3] D. Guo and T. Sim, "Digital face makeup by example," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 73–79.

[4] A. Dhall, G. Sharma, R. Bhatt, and G. M. Khan, "Adaptive digital makeup," in *Advances in Visual Computing*. Springer, 2009, pp. 728–736.

[5] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 1. IEEE, 2001, pp. I–511.

[6] S. Milborrow and F. Nicolls, "Locating facial features with an extended active shape model," in *Computer Vision–ECCV 2008*. Springer, 2008, pp. 504–513.

[7] J.-S. Kim and S.-M. Choi, "Interactive cosmetic makeup of a 3d point-based face model," *IEICE transactions on information and systems*, vol. 91, no. 6, pp. 1673–1680, 2008.

[8] C.-G. Huang, T.-S. Huang, W.-C. Lin, and J.-H. Chuang, "Physically based cosmetic rendering," *Computer Animation and Virtual Worlds*, vol. 24, no. 3-4, pp. 275–283, 2013.

[9] G. Kortüm, *Reflectance spectroscopy: principles, methods, applications*. Springer Science & Business Media, 2012.

[10] J. Jimenez, V. Sundstedt, and D. Gutierrez, "Screen-space perceptual rendering of human skin," *ACM Transactions on Applied Perception (TAP)*, vol. 6, no. 4, p. 23, 2009.

[11] I.-S. Jang, J. W. Kim, J.-Y. You, and J. S. Kim, "Spectrum-based color reproduction algorithm for makeup simulation of 3d facial avatar," *ETRI Journal*, vol. 35, no. 6, pp. 969–979, 2013.

[12] J.-S. Kim and S.-M. Choi, "Interactive cosmetic makeup of a 3d point-based face model," *IEICE transactions on information and systems*, vol. 91, no. 6, pp. 1673–1680, 2008.

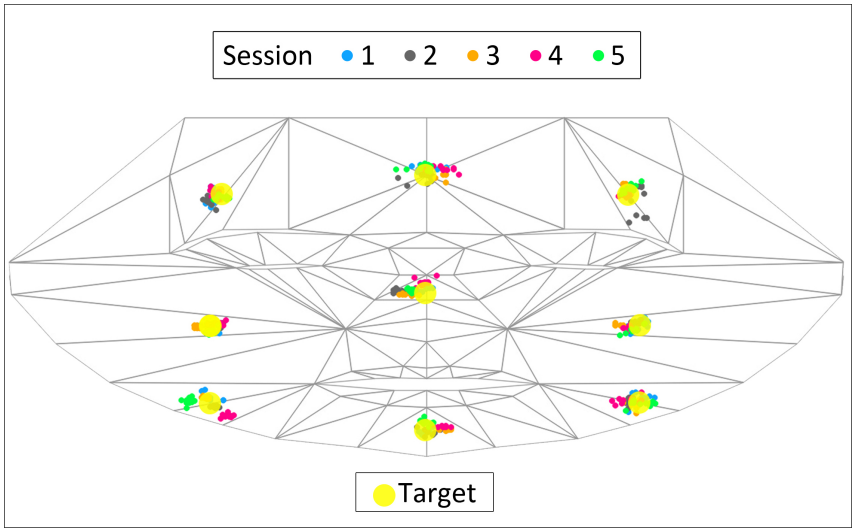

Fig. 7. Distribution of the estimated position for each facial touch in each of the five sessions. The yellow circles represent the position of the targets fixed on the face.

[13] E. Iwabuchi, M. Nakagawa, and I. Siio, "Smart makeup mirror: Computer-augmented mirror to aid makeup application," in *Human-Computer Interaction. Interacting in Various Application Domains*. Springer, 2009, pp. 495–503.

[14] M. Nakagawa, K. Tsukada, and I. Siio, "Smart makeup system: Supporting makeup using lifelog sharing," in *Proceedings of the 13th International Conference on Ubiquitous Computing*, ser. UbiComp '11. New York, NY, USA: ACM, 2011, pp. 483–484. [Online]. Available: http://doi.acm.org/10.1145/2030112.2030182

[15] A. Rahman, T. T. Tran, S. Hossain, and A. El Saddik, "Augmented rendering of makeup features in a smart interactive mirror system for decision support in cosmetic products selection," in *Distributed Simulation and Real Time Applications (DS-RT), 2010 IEEE/ACM 14th International Symposium on*. IEEE, 2010, pp. 203–206.

[16] A. Hanafusa, S. Terada, Y. Miki, C. Sasagawa, T. Ikeda, and T. Fuwa, "Makeup support system for visually impaired persons: Overview of system functions," in *Computers Helping People with Special Needs*. Springer, 2010, pp. 338–345.

[17] M. Cabral, G. Roque, D. dos Santos, L. Paulucci, and M. Zuffo, "Point and go: Exploring 3d virtual environments," in *2012 IEEE Symposium on 3D User Interfaces (3DUI)*, March 2012, pp. 183–184.

[18] L. Dipietro, A. M. Sabatini, and P. Dario, "A survey of glove-based systems and their applications," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 38, no. 4, pp. 461–482, July 2008.

[19] R. Y. Wang and J. Popović, "Real-time hand-tracking with a color glove," in *ACM SIGGRAPH 2009 Papers*, ser. SIGGRAPH '09. New York, NY, USA: ACM, 2009, pp. 63:1–63:8. [Online]. Available: http://doi.acm.org/10.1145/1576246.1531369

[20] A. Aristidou and J. Lasenby, "Motion capture with constrained inverse kinematics for real-time hand tracking," in *2010 4th International Symposium on Communications, Control and Signal Processing (ISCCSP)*, March 2010, pp. 1–5.

[21] J. M. Rehg and T. Kanade, *Visual tracking of high DOF articulated structures: An application to human hand tracking*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1994, pp. 35–46. [Online]. Available: http://dx.doi.org/10.1007/BFb0028333

[22] B. Stenger, P. R. S. Mendonca, and R. Cipolla, "Model-based 3d tracking of an articulated hand," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 2, 2001, pp. II–310–II–315 vol.2.

[23] A. Erol, G. Bebis, M. Nicolescu, R. D. Boyle, and X. Twombly, "A review on vision-based full dof hand motion estimation," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Workshops*, June 2005, pp. 75–75.

[24] B. Stenger, A. Thayananthan, P. H. S. Torr, and R. Cipolla, "Model-based hand tracking using a hierarchical bayesian filter," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 9, pp. 1372–1384, Sept 2006.

[25] M. de La Gorce, D. J. Fleet, and N. Paragios, "Model-based 3d hand pose estimation from monocular video," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 9, pp. 1793–1805, Sept 2011.

[26] F. Guo, D. Kimber, and E. Rieffel, "Featured wand for 3d interaction," in *2007 IEEE International Conference on Multimedia and Expo*, July 2007, pp. 2230–2233.

[27] A. M. F. de Sousa and C. H. Morimoto, "5* magic wand: An rgbd camera-based 5 dof user interface for 3d interaction," in *XVII Symposium on Virtual and Augmented Reality (SVR)*, May 2015, pp. 15–22.

[28] H. Hamer, K. Schindler, E. Koller-Meier, and L. V. Gool, "Tracking a hand manipulating an object," in *2009 IEEE 12th International Conference on Computer Vision*, Sept 2009, pp. 1475–1482.

[29] I. Oikonomidis, N. Kyriazis, and A. A. Argyros, "Full dof tracking of a hand interacting with an object by modeling occlusions and physical constraints," in *2011 International Conference on Computer Vision*, Nov 2011, pp. 2088–2095.

[30] I. Oikonomidis, M. I. A. Lourakis, and A. A. Argyros, "Evolutionary quasi-random search for hand articulations tracking," in *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, ser. CVPR '14. Washington, DC, USA: IEEE Computer Society, 2014, pp. 3422–3429. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2014.437

[31] S. Sridhar, A. Oulasvirta, and C. Theobalt, "Interactive markerless articulated hand motion tracking using rgb and depth data," in *2013 IEEE International Conference on Computer Vision*, Dec 2013, pp. 2456–2463.

[32] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, 04 2014.

[33] C. J. Burstone, "Lip posture and its significance in treatment planning," *American journal of orthodontics*, vol. 53, no. 4, pp. 262–284, 1967.