Rapid Photorealistic Blendshape Modeling from RGB-D Sensors

Dan Casas[1], Andrew Feng[1], Oleg Alexander[1], Graham Fyffe[1], Paul Debevec[1], Ryosuke Ichikari[1], Hao Li[2], Kyle Olszewski[2], Evan Suma[1], and Ari Shapiro[1]

[1]Institute for Creative Technologies, University of Southern California

[2]University of Southern California

ABSTRACT

Creating and animating realistic 3D human faces is an important element of virtual reality, video games, and other areas that involve interactive 3D graphics. In this paper, we propose a system to generate photorealistic 3D blendshape-based face models automatically using only a single consumer RGB-D sensor. The capture and processing requires no artistic expertise to operate, takes 15 seconds to capture and generate a single facial expression, and approximately 1 minute of processing time per expression to transform it into a blendshape model. Our main contributions include a complete end-to-end pipeline for capturing and generating photorealistic blendshape models automatically and a registration method that solves dense correspondences between two face scans by utilizing facial landmarks detection and optical flows. We demonstrate the effectiveness of the proposed method by capturing different human subjects with a variety of sensors and puppeteering their 3D faces with real-time facial performance retargeting. The rapid nature of our method allows for just-in-time construction of a digital face. To that end, we also integrated our pipeline with a virtual reality facial performance capture system that allows dynamic embodiment of the generated faces despite partial occlusion of the user's real face by the head-mounted display.

Keywords

face modeling, blendshapes, RGB-D, animation

3D characters are an important element of many 3D games, simulations, feature films, and other media that use 3D content. Of particular interest is the ability to represent the human face for purposes of expression, speech and recognition. The generation of a highly realistic, emotive digitally-based human face has been shown in feature films and high end video games which utililize a combination of traditional 3D art techniques and high-quality scanning.

Scan-based facial modeling techniques are capable of capturing the appearance and expression of a human subject through a combination of still images, video, or 3D images from a depth sensor or laser scanner. In contrast to traditional 3D art pipelines, scan-based techniques allow for subtle and realistic variations in shape and color between subjects. However, the challenge when using scan-based data is in manipulating the data into a form that be controlled in a simulation. Scanning techniques are susceptible to problems such as noisy data, inconsistent topologies, texture discolorations and so forth. Thus scan-based techniques are challenged to transform complex, unstructured data into well-formed, structured data, which can be complicated and time-consuming. And thus, the widespread use of high quality, scanned 3D faces in simulations is limited due to the complexity and time needed to transform such scanned data into a well-formed set of controllable data.

In this work, we demonstrate a method of capturing a set of 3D facial data from a commodity depth sensor, then transform these facial poses into a set of blendshapes and textures that can be used in a standard 3D animation pipeline. Our capture system takes only 15 seconds to capture and construct a single facial pose, and approximately 1 minute to process that pose and generate a topologically consistent blendshape. The processing is done near-automatically, and requires no artistic expertise. Also, the expression capture does not require a separate operator, and can be performed by the capture subject themself. Thus, our method allows a single subject to capture their own expressions and subsequently generate a data set that can be used on standard 3D facial animation systems that require blendshapes and textures. By allowing the rapid capture and transformation of human subject's facial data, numerous 3D facial images suitable for animation can be generated efficiently. Since the capture subject's own faces are used to generate the data, the facial models retain a photorealistic appearance and by definition reside within the limits of human facial appearances. Thus our method is capable of quickly generating variations in a human face, while simultaneously retaining the efficiency of traditional 3D facial pipelines.

Our method is data-agnostic; any set of unstructured, individual poses can be used with our method and similarly transformed into blendshapes and textures. Thus data captured from higher-quality sources, such as depth sensors with higher scan and RGB resolution or data obtained from photogrammetry systems can also benefit from this method.

Our method has two main contributions: 1) a registration method capable of operating on non-coherent 3D scan data, and 2) an end-to-end system for generating a photorealistic set of blendshapes and textures from any set of 3D input scans.

CASA 2016, May 23-25, 2016, ,

DOI: http://dx.doi.org/10.1145/2915926.2915936

1. RELATED WORK

1.1 3D Reconstruction

Capturing realistic 3D face models from a human subject has been an important goal in computer graphics. Highly detailed facial geometry can be obtained with photogrammetry in a controlled studio setup. Work from [1] utilizes multiple DSLR cameras to perform multiview face scans resulting in high quality facial geometry with detailed normal and reflectance maps. Other methods initially reconstruct each frame independently [2], resulting in a per-frame geometry, using a passive stereo system that enables submillimeter accuracy. The unstructured mesh sequence is then temporally aligned by identifying *anchor frames* to propagate a consistent topology across the sequence using multi-view optical flow [3]. Motion capture techniques are used in combination with 3D scans to produce high fidelity images, but requires motion capture markers. Similarly, the work in [4] captures the full facial performance from multi-view video sequences. Unaligned meshes are first reconstructed at each video frame and are used as constraints to produce the aligned mesh sequences. Additional skin details are captured using photometric stereo with color lights. The above methods produce very high quality faces but usually require a studio setup and controlled lighting to extract facial details.

The recent advancements in RBG-D sensor technologies provide low cost solutions to obtain high quality 3D models.

KinectFusion [5] reconstructs a 3D environment in real-time using inputs from a moving Kinect [6] sensor. The method works well in reconstructing 3D static scenes but is not suitable for capturing non-rigid scene such as human bodies. Methods for full-body scan from low-cost commodity depth sensors have also been proposed. For the sake of simplicity, common approach is to turn around a single depth sensor [7, 8] to capture multiple depth scans of the actor from different views. Alignment techniques based on Iterative Closest Point, ICP, [9] are generally used due to their efficiency to align the depth scans and create a single point cloud. Non-rigid registration techniques have been also proposed [7, 10, 11] but have not been considered in this work due to the rigid nature of a single expression scan.

Work in [12] uses a low-cost RGB-D sensor to infer an accurate face model. It works by setting a cylinder around the reference frame and continuously transforming the 3D point clouds to 2D cylindrical map to update the face model. Since low-cost depth sensors tend to produce noisy point clouds, they also apply bilateral filtering on the cylinder map to produce smooth models. More recently, the method proposed in [13] utilize a template-based technique for real-time non-rigid reconstruction. It first captures a base template geometry from the subject that is moving rigidly. The template mesh is then deformed to align with the input scans from RGB-D sensor in real-time to obtain a non-rigid reconstruction. This allows real-time 3D geometry acquisition from a deforming subjects using only low-cost depth sensors.

1.2 Face Rigging and Blendshapes

The movement of a human face is complex and it is a challenging task to animate a face realistically. The most widely used technique for modeling facial animation is blendshapes, which define a linear space for facial expressions [14]. Building such a blendshape rig typically requires efforts from 3d artists to craft each shape manually. The work in [15] can produce blendshapes by fitting a template model onto multiple images. The method is mostly automatic and only requires the user to mark up some feature points in the photos. The work in [16] captures high quality 3D face scans using multiple synchronized cameras and structured light projectors. A template mesh is then used to register with each scan to obtain a consistent face mesh sequences that can also be used as blendshapes. Reconstruction of detailed face blendshapes from monocular video is performed by [17], but also requires an initial manual creation of a blendshape model, as well as utilizing a video stream to tune the facial model. In addition, since the method copies textures from the video, there is no explicit step to enforce texture coherency, thus extraction of blendshapes from such a method could result in texture drift that our method explicitly handles. Another work [18] proposed an automatic method to learn the segmentations of local shape deformations from a set of blendshapes and build a more expressive control rig. The work in [19] produces a full set of blendshapes models from the 3D model of a specific character. It uses a pre-defined blendshape of a generic face model and transfers the expressions to the user-provided 3D face model. This method is useful as it only requires a reduced set of example poses for a specific model to produce fully expressive face blendshapes. One advantage of blendshapes is that the control is simple and robust, yet it produces expressive results. Recently there have been a number of advancements in the area of facial tracking, where the goal is to efficiently and accurately track the movements of a face over time. Such tracking can be done from 2D video frames [20, 21] or from RGB-D sensors [22, 23]. Our work differs in that we are presenting a method for the generation of blendshapes that can subsequently be controlled with such tracking methods. Our examples show the use of a real time tracking system as a control signal to our generated blendshapes, but we do not address the facial tracking problem at all. While those methods do produce 3D geometric models during the course of their tracking results, the models lack the fidelity of the original scan data and likewise do not account for texture correspondences. By contrast, our method attempts to preserves the original scan data, resulting in imagery that mirrors the original scans.

FaceShift [24] is a facial performance capture system using an RGB-D sensor. Although at a glance it seem to share the same goal as our work and use the same accumulated face models from RGB-D sensor, our work differs in how we make use of the input raw scans and the resulting textured face blendshape models. FaceShift utilizes template fitting methodology to deform a generic template model toward raw face scans. Therefore the resulting face model represents only an *approximation* of the original geometry instead of an exact reconstruction. Moreover, since the template model is deformed under geometric constraints, there is no guarantees on exact texture alignments between different facial expressions. On the other hand, our method starts from raw face scans and directly extract *consistent mesh* from the scans. Therefore the geometric representations are exact to the original shapes. Our method unwrap both geometric and texture information into 2D images and solve for alignments in the UV space using facial feature detection and optical flow. Therefore the geometric and texture alignments are more accurate in the reconstructed facial expressions. For better visualization of the surface alignment results, Figure 1 illustrates a comparison of the generated texture maps for expressions 'neutral' and 'mouthLeft'. Our approach improves upon FaceShift results, which suffer from texture drifting artifacts, because it aligns both texture and geometry using a novel optical-flow based surface alignment approach in a 2D domain.

Closely related to our work is [25], which has shown how multiple images from a smartphone can be used as a data source to generate a template rig with localized texture details. Our work differs in that we are using scans from RGB-D sensors instead of photogrammetric processes, and we are using per-shape textures, rather than extracting high resolution details. In addition, since our

(a) Ours 'neutral' (b) Ours 'mouthLeft' (c) FaceShift 'neutral' (d) FaceShift 'mouthLeft'

Figure 1: Comparison of the texture maps generated with our approach, (a) and (b), and FaceShift, (c) and (d), for 'neutral' and 'mouthLeft' scans, respectively. Notice that the surface drifting artifacts in the eyebrow area when using FaceShift is not present using the proposed approach. Please check the supplementary video for better visualization.

method does not use templates, it also generates photorealistic results, in contrast to a stylized appearance that the templates generate. In addition, our method has much faster processing, and it does not on its own produce an entire head, which a template method is capable of doing.

[26] demonstrates a method to use data from an RGB-D sensor and populate a template model while adding textures from specific regions of the face, such as the wrinkles above the brow. By contrast, our method utilizes the entire texture from each facial expression, rather than enhancing specific expressions with limited details. [27] demonstrates facial reenactment using a video-based capture of source and target while finding matching areas between the two. [28] uses a template model-based face replacement to perform a realtime puppeteering of some facial expressions by overlaying onto video. High frequency detail is transferred to the target actor. By contrast, our method is not based on video processing, and directly captures texture details, such as blood flow that could not be captured by high frequency filtering.

Our goal of photorealistic blendshapes requires the manipulation not only of the underlying face geometry but also the appearance. Texture maps need to be aligned in the UV space to avoid ghosting artifacts in the final texture. Texture alignment to synthesize parametric textures for a 3D model has been recently tackled by using online optical flow computed from the virtual view-point [29]. In addition, a combination of image, shape and directable forces has been used to create scan correspondences in [30]. Similarly, multi-camera setups also use texture alignment techniques to synthesize blended view-dependent appearances [31, 32]. The work in [33] reconstructs mesh sequences using 14 cameras and optical flow to recover motion sequences at 30 frames per second. Our method likewise uses optical flow, but attempts to recover a set of controllable, user-specified blendshapes, rather than a sequence from a performance, and uses only a single RGB-D sensor.

The typical artist-driven transformation of a 3D scan into a blendshape involves manually manipulating the geometry of a neutral face scan until it matches the appearance of a different expressive scan. Commercial tools help to automate the geometry processing, but do not account for the simultaneous texture and geometry registration as our method does.

2. SYSTEM OVERVIEW

The goal of our work is to build an end-to-end system that can quickly capture a user's face geometry using a low-cost commodity sensor and convert the raw scans into a blendshape model automatically. Figure 2 summarizes the workflow of our pipeline. First, the face geometry will be captured and reconstructed using an RGB-D sensor (Section 3). Since the raw face scans have different positions and orientations, we run rigid alignment between expressions using iterative closest point (ICP) to obtain a set of aligned scans. These scans are then unwrapped into a 2D representation of point clouds and texture UV maps and stored in EXR float images to be used for surface tracking (Section 4.1). The surface tracking then utilizes the 2D representation of the face scans and finds correspondences from a source face pose to the target neutral face pose. To guide the surface tracking, we first apply face feature detection to find a set of facial landmark points on each scan (Section 4.2). These feature points are used to build a Delaunay triangulation on the UV map as the intial constraints. This triangulation is used to pre-warp the 2D map of each face scan to the target neutral face pose. Then a dense image warping is done using optical flow to transform the source image to the target image (Section 4.3). Once the dense correspondences are established, the blendshape models can be produced by extracting a consistent mesh from each face point cloud image using an artist mesh (Section 5.1).

3. FACE ACQUISITION

We obtain individual face scans from an RGB-D sensor and reconstruct each face pose using the method in [12]. We choose this method for face acquisition since it is fast and requires only a single depth sensor. However, the result of the pipeline works regardless of the capturing methods. Thus higher quality face models obtained from photogrammetry or laser scans will also work. To capture a face, the user would sit in front of the RGB-D sensor to capture the depth image frame of a near frontal face as the reference frame. This reference frame is used to generate 3D point clouds. The point clouds are then unwrapped onto a reference 2D cylindrical map. Then the user can freely move his head to obtain depth scans from different views. Each of the subsequent depth scans are then converted to 3D points and then registered with the reference point clouds. The registered points from the new scans are then unwrapped and aggregated onto the reference map to obtain a new map. In order to avoid failed registration results from corrupting the final model, some depth scans are discarded if they can not be aligned properly with the reference frame. Since the resulting model may contain noise, the method applies a bilateral filter on the cylindrical 2D map to remove noises while keeping the sharp features. Finally, a smooth face geometry is produced by triangulating the neighboring pixels in the final cylindrical map. The texture is then obtained by projecting the final face model onto the reference RGB image. The aforementioned face capture process is very fast and requires about only 20 seconds to capture and process a face pose using an Intel RealSense sensor. Since the raw face poses are captured separately, they may have different positions and orientations. Thus they need to be first aligned into the same reference frame before the surface tracking stage. To achieve this, we first extract the points cloud from face geometry and use iterative closest point (ICP) [34] to rigidly align points cloud of each face pose with the neutral face. Since there are non-rigid deformations at the lower parts of faces, a naive rigid registration of the whole face scan tends to be less accurate. Thus we perform the rigid alignment only at the forehead regions of each face between different poses, similar to [35]. This adjustment improves the accuracy of face pose registration since the deformations at the lower parts of faces do not affect the alignment results. The aligned points cloud are then mapped back to the 2D images to be used in the next stage. Note that in our work, we already have the cylindrical mapping from the capture session to map the points cloud back to the 2D image. If the input geometries are captured with a different method such as laser scans, the cylindrical mapping can be applied to convert the

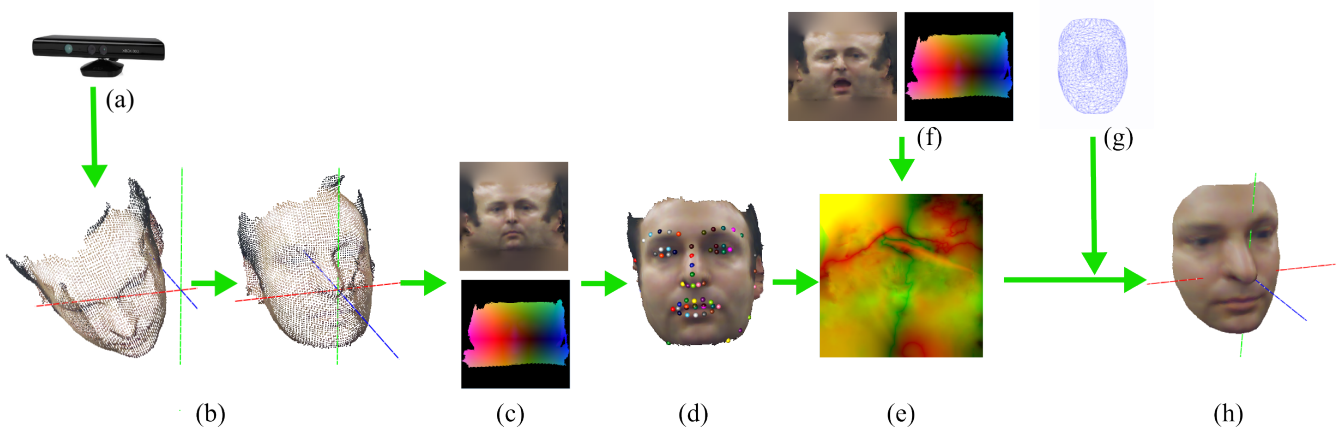


Figure 2: Diagram depicting the proposed pipeline for photorealistic blendshapes from RGB-D: (a) A set of facial expressions are scanned; (b) Rigid alignment between expressions is obtained by automatic ICP registration; (c) and (f) 3D textured meshes are converted into a 2D representation and stored in EXR floating image format, in this particular case (c) is the source scan and (f) the target scan; (d) Automatic facial landmark detection is used to detect common features in scans; (e) A combination of Delaunay triangulation over the detected landmarks and 2D optical flow is used for dense warping between source scan (c) and target scan (f); (g) Reference mesh sharing the same UV space as the target scan is used to extract the final blendshape (h).

geometries back to 2D images.

4. SURFACE TRACKING

Triangulated pointclouds present a per-frame independent topology, which hinders the reutilization of the reconstructed models in the traditional animation pipelines. When capturing an actor's face in multiple static facial expressions, it is often desirable for the resulting mesh to all have the same topology and for the textures to all be in the same UV space. Such "corresponded", or topologically-consistent, meshes would enable the straightforward creation of blendshape-based facial rigs, extensively used in face animation pipelines.

Marker-based motion capture systems that track a set of markers across video has been used to generate temporally-consistent 3D models from multicamera capture [36]. However, marker-based approaches have not been considered in this work due to the implicit problem of visible markers in the final texture maps, which would decrease the realism of the final models. In the context of multi-video capture, markerless approaches have also been recently introduced [37, 2, 4]. These video-based automatic methods rely on the fact that there is a small difference in either pixel color values or surface deformation between consecutive frames, and therefore optical flow based tracking performs well. Similarly, non-sequential approaches [38] aim to find similar frames across video sequences.

This section targets the problem of surface tracking across a sparse set of scans from a commodity RGB-D camera, with large deformations between each sample. Current video-based surface tracking approaches cannot handle such sparse sets due to the limitations of the optical flow in large displacements. Our goal also requires *fast* processing times, in the order of seconds, to enable the automatic construction of a complete set of photorealistic blendshapes in minutes.

4.1 Data Format

Rather than storing our scans as geometry and textures, we choose instead to store our scans as images. Each one of our scans is stored as a 32 bit float EXR [39] texture map image, and a high resolution point cloud. As it is discussed later on in this paper, the main motivation for this format conversion is the fact that this work tackles the 3D mesh surface tracking problem in the 2D UV domain.

The maps are in a cylindrically unwrapped UV space, representing our ear to ear data. The UV space of each expression falls roughly in the same area because each expression has been previously aligned with respect to the *neutral*, which is used as a reference, removing the rigid transformations. However, due to inherent changes in surface details across scans, the UV space differs slightly for each expression.

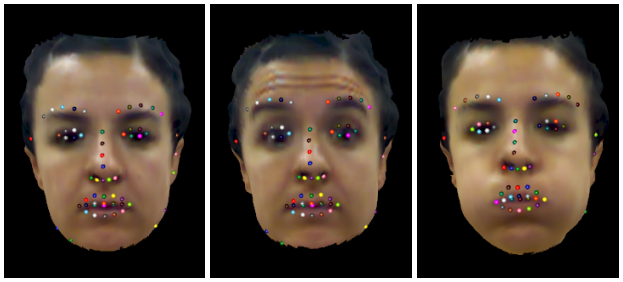
4.2 Face Landmarks Detection

Our approach for 3D surface tracking exploits the image-based scan representation described in Section 4.1 by doing the scan correspondence in 2D rather than 3D. For each shape to process, two scans are taken as input data: one of the actual expression as the source and the *neutral* expression as the target.

For each of the two inputs, we need to find a set of facial landmarks that will be used as a starting set of features to build a new triangulated mesh on the UV domain. This marking process, which could also be done manually by means of a GUI interface that helps the user to select corresponding points in a 3D rendered model [40], is automatically done by an state-of-the-art facial landmark detection framework [41].

A frontal view of each of the scans is rendered and processed by the landmark detector, which returns a set of 2D features that are then projected into the 3D model. These features are not required to be specific face features such as corners of the eyes and lips, any pair of features from corresponding positions between faces are valid. In particular, the landmark detector used in this work [41] is trained to find 38 face features, depicted in Figure 3, but any other existing method for landmark detection could be used. Notice that our approach only takes as valid features those that are successfully detected in both input scans, therefore marking a corresponding point. It is also important to notice that these correspondence points do not have to be exact –the points are used only as an initialization for the deforming algorithm.

4.3 Geometry and Texture Warping



(a) Neutral (b) Eyebrows up (c) Puff

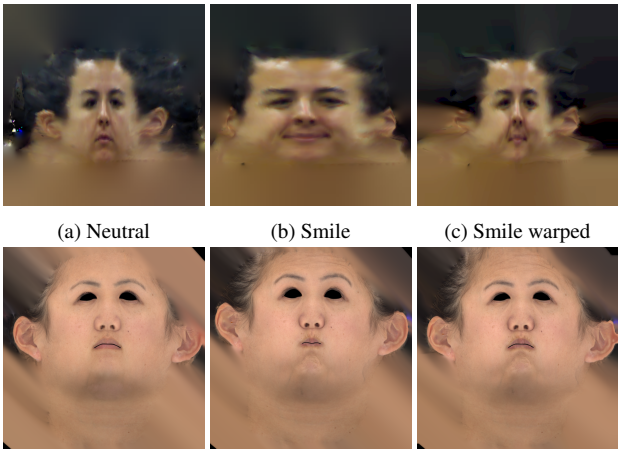
Figure 3: Automatic landmark detection results obtained using [41].

A geometry and texture 2D warping algorithm is used to align the *source* scan into the *target* scan [42], following three steps. First, a Delaunay triangulation is built between the set of landmarks of both the *source* and *target* scans. The constructed mesh is used to roughly pre-warp the source texture map to the target using affine triangle transformations. Second, a GPU-accelerated optical flow is used to compute a dense warp field from the pre-warped source texture map to to the target. Finally, dense warp is used to deform both the texture map and the point cloud from the *source* to the *target* scan. This results in the *source* scan warped to the *target* UV space.

Some expressions are more challenging to correspond than others. Especially expressions with lots of occlusions, like mouth open to mouth closed. In such cases, optical flow may fail to get a good result, but our pipeline provides also a semi-automatic tool that allows the user to interactively manipulate the set of correspondences. Also, we can assist the optical flow in two ways. First, by painting black masks around occlusion regions in both source and target diffuse textures. Second, by marking some points as "pinned" and those points are rasterized into small black dots at runtime. Using both of these techniques in combination usually produces good results even in the most challenging cases.

Figure 4 presents a visualization of texture warping results achieved with the proposed approach. Notice how the UV space of the warped texture aligns with the *neutral*.

Analogously, Figures 4d, 4e and 4f present texture warping results for a character captured in a multi-camera setup with controlled lighting [1].



(a) Neutral (b) Smile (c) Smile warped

(d) Neutral (e) Puff (f) Puff warped

Figure 4: Example texture alignment results. Left and center columns are the target and source texture maps respectively, generated from RGB-D capture (top) and multi-camera capture (bottom). Right column contains the results after applying the proposed approach to warp the source UV space to the target.

5. BLENDSHAPES

In this section, we describe the process of automatically generating blendshapes from the aligned *source* and the *target* scans. Additionally, we also present a masking approach for combining blendshapes in a real-time system.

5.1 Blendshape Generation

First, the *neutral* scan is remeshed creating what we refer to as *artist mesh*. Such mesh, whose topology will be propagated to all the scans, can be generated using a standard automatic decimation process or manually produced by an artist. The latter option allows the artist to manually arrange the triangle topology, which ideally must have a smaller triangles in the areas with large non-rigid surface dynamics, such as the mouth, to ease the mesh deformation [40]. However, the results presented in this paper use an automatically decimated mesh and demonstrate that successful results can be also generated with out artist help. The UV coordinates of the *artist mesh* is generated by transferring the UV space previously automatically generated for the *neutral* expression. Notice that this step is just done one single time per dataset of shapes.

Second, the *artist* mesh is propagated into the *source* mesh by looking up the vertex positions in the warped point clouds. The texture map is also warped into the artist UV space, which is simply an additional affine triangles 2D warp. This results in a set of blendshapes and textures ready to plug into the standard facial animation pipeline.

At run time, any requested blended shape is computed following the *delta blendshape formulation* [14].

$$\mathbf{f} = \mathbf{b_0} + \sum_{k=1}^{n} w_k(\mathbf{b_k} - \mathbf{b_0}) \tag{1}$$

where

$$\mathbf{f} = (x_0, y_0, z_0, \dot{sc}, x_n, y_n, z_n)^T \tag{2}$$

is the resulting shape, n the number of vertices, k the number of shapes, $w_k \in [0,1]$ the weight for the k^{th} shape, and $\mathbf{b_k}$ the k^{th} shape and $\mathbf{b_0}$ the *neutral* shape.

Equation 1 treats each shape as a whole, considering that each blending weight w_k associated to the mesh $\mathbf{b_k}$ affects equally to all areas of the shape (i.e: all vertices are associated with the same w_k). However, in many cases we may be interested in combining two shapes that affect two different local areas of the face, for example, *eyes_close*, which modifies upper area of the face, and *kiss* expressions, which modifies the lower area. In order to satisfy this need for local blending weight control, matrix $\mathbf{m_k}$ is incorporated into Equation 1 as follows

$$\mathbf{f} = \mathbf{b_0} + \sum_{k=1}^{n} w_k \mathbf{m_k}(\mathbf{b_k} - \mathbf{b_0}) \tag{3}$$

where $\mathbf{m_k}$ is a diagonal matrix of size $3n \times 3n$ containing the local weights for each vertex.

5.2 Combining Blendshapes

Blendshapes can be utilized most effectively in a real-time system by creating masks to localize blendshapes to particular regions

of the face. As an input to a real-time system, masking effects enable the combination of multiple blendshapes to produce a number of versatile behaviors, such as blinking and eyebrow movement during arbitrary facial expressions, see Figure 5. Additionally, this approach can also be used for generating the mouth poses necessary for visual speech or lip syncing.

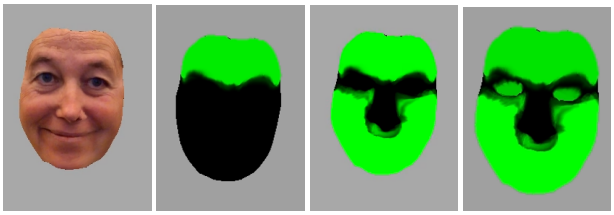


Figure 5: A set of blendshapes with localized effects from masking textures during rendering. The masks separate the upper face, eyes and lower face into separate regions, allowing for blinking, lip movement, and eyebrow expressions by combining several blendshapes simultaneously, shown in green.

6. RESULTS AND APPLICATIONS

The blendshapes generated by the proposed method can be used in many animation and simulation environments that utilize blendshapes. Our approach is data agnostic and can utilize scan input from depth-sensors. We demonstrate results using the Kinect v1, Intel RealSense F200, and Occipital Structure Sensor. We expect our method to produce results according to the quality of the sensors, including depth and color specifications. Thus our method should produce higher levels of details with better sensors as they have already shown from low (Kinect v1), medium (RealSense) and high (Structure Sensor). Additionally, the method introduced in this paper is also compatible with 3D data generated from photogrammetry techniques. A single user can quickly capture scans, process the data, and puppeteer the generated face without artist intervention in a matter of minutes.

Our accompanying videos demonstrate the acquisition, processing, and use of the blendshape data with a real-time animation system and real-time facial tracking software [24, 22]. Noting the recent proliferation of consumer virtual reality technology, we have integrated our face scanning and processing pipeline with a recently developed head-mounted display facial performance capture system [43, 44], see Figure 10. This system uses a head-mounted RGB-D camera to capture lower facial expressions combined with strain sensors embedded in the foam lining of the display to sense expressions in the occluded upper region. This makes it possible for dynamic real-time embodiment of one's own (or someone else's) face within an immersive virtual reality environment.

7. DISCUSSION

We do not include any explicit handling or separation of lighting. The RGB-D scan examples shown in this paper have been captured under typical indoor lighting conditions.

By using the scans from a depth sensor and RGB camera, our method is able to generate face data that has a photorealistic appearance. This is in contrast to a stylized or cartoonish appearance that are often generated through traditional artist-driven 3D facial construction and through many template-based methods that use priors [25, 19]. The quality of scan data is dependent on the sensor quality. Thus, sensors that can obtain higher resolutions or greater detail will produce higher quality results through our method.

(a)

(b)

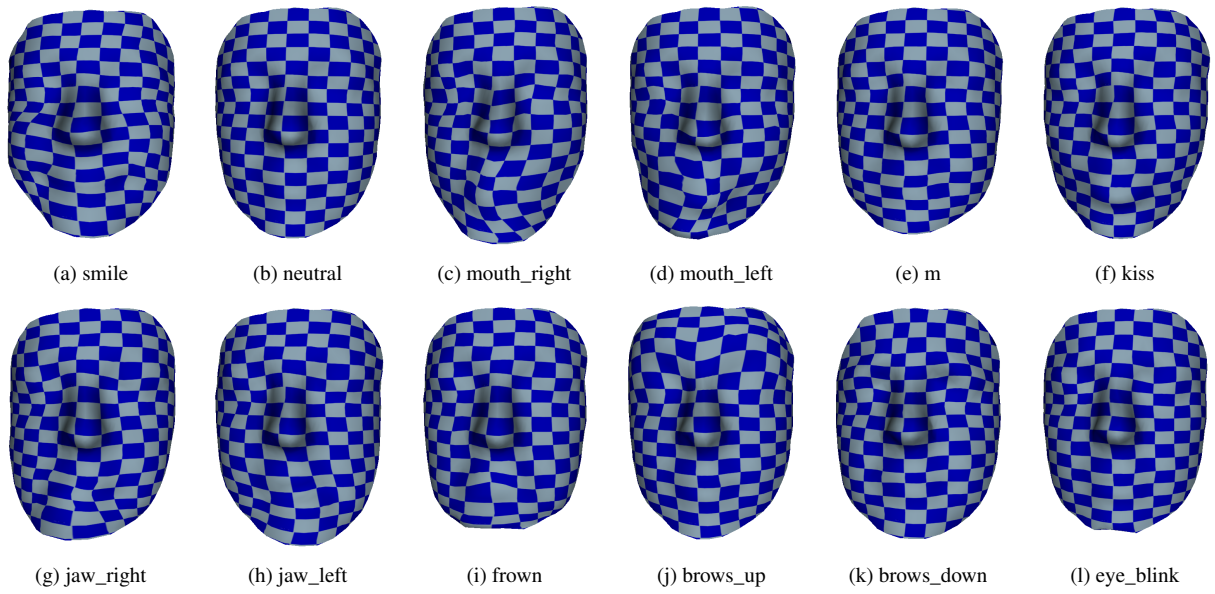
Figure 8: Real-time puppeteering a photorealistic character built with the Intel RealSense F200 sensor, first by the original capture subject, then by a different person.

8. CONCLUSION

Our method can rapidly generate a set of photorealistic, expressive facial poses as blendshapes from a single commodity depth sensor within a relatively short amount of time, while requiring no artistic or technical expertise on the part of the capture subject. We demonstrate our approach as part of a complete end-to-end system for scanning, processing, and real-time control. The rapid nature of model acquisition and automatic processing enables the ability to generate a controllable 3D face model for environments where the fast construction of an new face model is desirable. For example, in a virtual environment while wearing a head-mounted display. Thus, this work advances the state-of-the-art for the rapid creation of photorealistic digital representations of real people that can enable multi-user communication and collaboration in virtual reality.

9. REFERENCES

[1] Abhijeet Ghosh, Graham Fyffe, Borom Tunwattanapong, Jay Busch, Xueming Yu, and Paul Debevec. Multiview face capture using polarized spherical gradient illumination. *ACM Transactions on Graphics (TOG)*, 30(6):129, 2011.



(a) smile (b) neutral (c) mouth_right (d) mouth_left (e) m (f) kiss

(g) jaw_right (h) jaw_left (i) frown (j) brows_up (k) brows_down (l) eye_blink

Figure 6: Surface tracking results using the approach described in Section 4. Shapes from dataset used in Figure 7.

(a)
jaw_right = 0.05
mouth_right = 0.04

(b)
open = 0.84
mouth_left = 0.05

(c)
brows_up = 0.71
open = 0.15
jaw_right = 0.10

(d)
kiss = 0.24
puff = 0.94

(e)
kiss = 0.85
puff = 0.21

(f)
mouth_right = 0.15
smile = 0.75

Figure 7: Real-time puppeteering a photorealistic character built within minutes using Kinect v1 sensor and the framework proposed in this paper.

Figure 9: Blendshape results using our method with data from an RGB-D sensor (Occipital Structure Sensor) and iPad. Our method is capable of preserving facial details; for example, notice the pronounced vertical ridge for the eyebrows down shape (lower left image).

Figure 10: Facial performance capture head-mounted display (left and center) that enables live puppeteering of the generated 3D faces in virtual reality (right), despite the fact that the user's real face is partially occluded by the display hardware.

MultiMedia, IEEE, 19(2):4–10, 2012.

[2] Thabo Beeler, Fabian Hahn, Derek Bradley, Bernd Bickel, Paul Beardsley, Craig Gotsman, Robert W. Sumner, and Markus Gross. High-quality passive facial performance capture using anchor frames. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2011)*, 30:75:1–75:10, 2011.

[3] Thabo Beeler, Bernd Bickel, Paul Beardsley, Bob Sumner, and Markus Gross. High-quality single-shot capture of facial geometry. *ACM Transactions on Graphics (TOG)*, 29(4):40, 2010.

[4] Martin Klaudiny and Adrian Hilton. High-Detail 3D Capture and Non-sequential Alignment of Facial Performance. In *International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission, 2012 (3DIMPVT)*, pages 17–24, 2012.

[5] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, et al. KinectFusion: real-time 3D reconstruction and interaction using a moving depth camera. In *Symposium on User Interface Software and Technology*, pages 559–568. ACM, 2011.

[6] Zhengyou Zhang. Microsoft kinect sensor and its effect.

[7] Hao Li, Etienne Vouga, Anton Gudym, Linjie Luo, Jonathan T. Barron, and Gleb Gusev. 3D Self-Portraits. *ACM Transactions on Graphics (Proceedings SIGGRAPH Asia 2013)*, 32(6), November 2013.

[8] Ari Shapiro, Andrew W. Feng, Ruizhe Wang, Hao Li, Mark T. Bolas, Gérard G. Medioni, and Evan A. Suma. Rapid avatar capture and simulation using commodity depth sensors. *Journal of Visualization and Computer Animation*, 25(3-4):201–211, 2014.

[9] Szymon Rusinkiewicz and Marc Levoy. Efficient variants of the ICP algorithm. In *3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on*, pages 145–152. IEEE, 2001.

[10] Jing Tong, Jin Zhou, Ligang Liu, Zhigeng Pan, and Hao Yan. Scanning 3d full human bodies using kinects. *Visualization and Computer Graphics, IEEE Transactions on*, 18(4):643–650, 2012.

[11] Will Chang and Matthias Zwicker. Range scan registration using reduced deformable models. In *Computer Graphics Forum*, volume 28, pages 447–456. Wiley Online Library, 2009.

[12] Matthias Hernandez, Jongmoo Choi, and Gérard Medioni. Laser Scan Quality 3-D Face Modeling Using a Low-cost Depth Camera. In *European Signal Processing Conference (EUSIPCO), 2012*, pages 1995–1999. IEEE, 2012.

[13] Michael Zollhöfer, Matthias Nießner, Shahram Izadi, Christoph Rehmann, Christopher Zach, Matthew Fisher, Chenglei Wu, Andrew Fitzgibbon, Charles Loop, Christian Theobalt, and Marc Stamminger. Real-time Non-rigid Reconstruction using an RGB-D Camera. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2014)*, 33(4), 2014.

[14] JP Lewis, Ken Anjyo, Taehyun Rhee, Mengjie Zhang, Fred Pighin, and Zhigang Deng. Practice and Theory of Blendshape Facial Models. In *EUROGRAPHICS - State of the Art Reports*, pages 199–218, 2014.

[15] Frédéric Pighin, Jamie Hecker, Dani Lischinski, Richard Szeliski, and David H. Salesin. Synthesizing Realistic Facial Expressions from Photographs. In *Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '98, pages 75–84, 1998.

[16] Li Zhang, Noah Snavely, Brian Curless, and Steven M. Seitz. Spacetime faces: High resolution capture for modeling and animation. In *ACM SIGGRAPH 2004 Papers*, SIGGRAPH

'04, pages 548–558, New York, NY, USA, 2004. ACM.

[17] Pablo Garrido, Levi Valgaerts, Chenglei Wu, and Christian Theobalt. Reconstructing detailed dynamic face geometry from monocular video. *ACM Trans. Graph.*, 32(6):158, 2013.

[18] Pushkar Joshi, Wen C Tien, Mathieu Desbrun, and Frédéric Pighin. Learning controls for blend shape based realistic facial animation. In *ACM SIGGRAPH 2005 Courses*, page 8. ACM, 2005.

[19] Hao Li, Thibaut Weise, and Mark Pauly. Example-based facial rigging. In *ACM SIGGRAPH 2010 Papers*, SIGGRAPH '10, pages 32:1–32:6, New York, NY, USA, 2010. ACM.

[20] Chen Cao, Qiming Hou, and Kun Zhou. Displaced dynamic expression regression for real-time facial tracking and animation. *ACM Transactions on Graphics (TOG)*, 33(4):43, 2014.

[21] Chen Cao, Yanlin Weng, Stephen Lin, and Kun Zhou. 3d shape regression for real-time facial animation. *ACM Trans. Graph.*, 32(4):41, 2013.

[22] Sofien Bouaziz, Yangang Wang, and Mark Pauly. Online modeling for realtime facial animation. *ACM Transactions on Graphics (TOG)*, 32(4):40, 2013.

[23] Hao Li, Jihun Yu, Yuting Ye, and Chris Bregler. Realtime facial animation with on-the-fly correctives. *ACM Trans. Graph.*, 32(4):42, 2013.

[24] Thibaut Weise, Sofien Bouaziz, Hao Li, and Mark Pauly. Realtime performance-based facial animation. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2011)*, 30(4):77, 2011.

[25] Alexandru Eugen Ichim, Sofien Bouaziz, and Mark Pauly. Dynamic 3d avatar creation from hand-held video input. *ACM Trans. Graph.*, 2015.

[26] Jun Li, Weiwei Xu, Zhiquan Cheng, Kai Xu, and Reinhard Klein. Lightweight wrinkle synthesis for 3d facial modeling and animation. *Computer Aided Design*, 58:117–122, 2015.

[27] Pablo Garrido, Levi Valgaerts, Ole Rehmsen, Thorsten Thormaehlen, Patrick Perez, and Christian Theobalt. Automatic face reenactment. In *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*, pages 4217–4224, 2014.

[28] J. Thies, M. Zollhöfer, M. Nießner, L. Valgaerts, M. Stamminger, and C. Theobalt. Real-time expression transfer for facial reenactment. *ACM Transactions on Graphics (TOG)*, 34(6), 2015.

[29] Dan Casas, Marco Volino, John Collomosse, and Adrian Hilton. 4D Video Textures for Interactive Character Appearance. *Computer Graphics Forum (Proceedings of EUROGRAPHICS 2014)*, 33(2):371–380, 2014.

[30] Cyrus A Wilson, Oleg Alexander, Borom Tunwattanapong, Pieter Peers, Abhijeet Ghosh, Jay Busch, Arno Hartholt, and Paul Debevec. Facial cartography: interactive high-resolution scan correspondence. In *ACM SIGGRAPH 2011 Talks*, page 8. ACM, 2011.

[31] Martin Eisemann, Bert De Decker, Marcus Magnor, Philippe Bekaert, Edilson De Aguiar, Naveed Ahmed, Christian Theobalt, and Anita Sellent. Floating textures. *Computer Graphics Forum*, 27(2):409–418, 2008.

[32] Marco Volino, Dan Casas, John Collomosse, and Adrian Hilton. Optimal Representation of Multiple View Video. In *British Machine Vision Conference*. BMVA Press, 2014.

[33] Derek Bradley, Wolfgang Heidrich, Tiberiu Popa, and Alla Sheffer. High resolution passive facial performance capture. *ACM Transactions on Graphics (TOG)*, 29(4):41, 2010.

[34] Radu Bogdan Rusu and Steve Cousins. 3D is Here: Point Cloud Library (PCL). In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–4. IEEE, 2011.

[35] Thabo Beeler and Derek Bradley. Rigid stabilization of facial expressions. *ACM Transactions on Graphics (TOG)*, 33(4):44, 2014.

[36] Bernd Bickel, Mario Botsch, Roland Angst, Wojciech Matusik, Miguel Otaduy, Hanspeter Pfister, and Markus Gross. Multi-scale capture of facial geometry and motion. In *ACM Transactions on Graphics*, volume 26, page 33. ACM, 2007.

[37] Cedric Cagniart, Edmond Boyer, and Slobodan Ilic. Probabilistic deformable surface tracking from multiple videos. In *European Conference on Computer Vision (ECCV)*, pages 326–339. Springer, 2010.

[38] Chris Budd, Peng Huang, Martin Klaudiny, and Adrian Hilton. Global non-rigid alignment of surface sequences. *International Journal of Computer Vision*, 102(1-3):256–270, 2013.

[39] Florian Kainz, Rod Bogart, and Drew Hess. The openexr image file format. *GPU Gems: Programming Techniques, Tips and Tricks for Real-Time Graphics, R. Fernando, Ed. Pearson Higher Education*, 2004.

[40] Ryosuke Ichikari, Oleg Alexander, and Paul Debevec. Vuvuzela: A facial scan correspondence tool. In *ACM SIGGRAPH 2013 Posters*, SIGGRAPH '13, pages 89:1–89:1, New York, NY, USA, 2013. ACM.

[41] Tadas Baltrušaitis, Louis-Philippe Morency, and Peter Robinson. Continuous Conditional Neural Fields for Structured Regression. In *European Conference on Computer Vision (ECCV)*. a, 2014.

[42] Manuel Werlberger, Werner Trobin, Thomas Pock, Andreas Wedel, Daniel Cremers, and Horst Bischof. Anisotropic Huber-L1 optical flow. In *Proceedings of the British Machine Vision Conference (BMVC)*, London, UK, September 2009.

[43] Pei-Lun Hsieh, Chongyang Ma, Jihun Yu, and Hao Li. Unconstrained realtime facial performance capture. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1675–1683, 2015.

[44] Hao Li, Laura Trutoiu, Kyle Olszewski, Lingyu Wei, Tristan Trutna, Pei-Lun Hsieh, Aaron Nicholls, and Chongyang Ma. Facial performance sensing head-mounted display. *ACM Transactions on Graphics (TOG)*, 34(4):47, 2015.