Local Geometric Indexing of High Resolution Data for Facial Reconstruction from Sparse Markers

Matthew Cong, Lana Lan, and Ronald Fedkiw

Abstract—When considering sparse motion capture marker data, one typically struggles to balance its overfitting via a high dimensional blendshape system versus underfitting caused by smoothness constraints. With the current trend towards using more and more data, our aim is not to fit the motion capture markers with a parameterized (blendshape) model or to smoothly interpolate a surface through the marker positions, but rather to find an instance in the high resolution dataset that contains local geometry to fit each marker. Just as is true for typical machine learning applications, this approach benefits from a plethora of data, and thus we also consider augmenting the dataset via specially designed physical simulations that target the high resolution dataset such that the simulation output lies on the same so-called manifold as the data targeted.

Index Terms—Computer Graphics, Image Processing and Computer Vision, Interpolation.

1 Introduction

Realistic facial animation has a wide variety of applications in both computer vision and the entertainment industry [1]. It is typically achieved through a combination of keyframe animation, where an animator hand-adjusts controls corresponding to the motion of different parts of the face, and facial performance capture, which uses computer vision to track the motion of an actor's face recorded from one or more cameras. Despite the many techniques developed over the years, facial performance capture remains a difficult task, and the high degree of accuracy required to generate realistic facial animation severely suppresses its widespread impact.

One class of techniques which has a proven track record uses markers painted on an actor's face in conjunction with a stereo pair of head mounted cameras [1], [2]. These markers are tracked in each camera in 2D and triangulated to obtain a sparse set of animated 3D bundle positions representing the motion of the actor's face. In order to reconstruct a full 3D facial pose for each frame of 3D bundles, one often uses a parameterized (blendshape) model [3], [4]. However, these parameterized models often have large infeasible spaces. While a skilled animator can aim to avoid these infeasible spaces, an optimization algorithm would need them explicitly specified which is typically not practical. Another commonly used approach interpolates bundle displacements across the face [2]. However, this results in reconstructed geometry that is overly smooth since the sparse bundle positions cannot represent high-resolution details between the bundles, especially details that appear during expressions, e.g. folds, furrows, and wrinkles. To address these shortcomings, we follow the current trend in the deep learning community of adding more and more data by using a large dataset of facial shapes to inform the reconstruction of the face surface geometry from the tracked bundle positions.

Our approach to this problem can be thought of as local geometric indexing wherein each bundle needs to identify relevant associated geometry from the dataset. To accomplish this, we envision the dataset as a separate point cloud for each bundle; this point cloud is obtained by evaluating the 3D position of the relevant bundle on each face shape in the dataset. These point clouds are then used to index the dataset in order to figure out the most relevant shapes given a bundle position. A bundle position that lies outside of its associated point cloud indicates a lack of data and can be projected back towards the point cloud. On the other hand, it is also possible for many candidate points to exist in the point cloud in which case neighboring bundles and their associated point clouds can be used to disambiguate. Finally, the shapes chosen for each bundle are combined to obtain a high-resolution dense reconstruction of the facial geometry.

We begin the exposition by describing the creation of our facial shape dataset, which is initially bootstrapped via a combination of dense performance capture and hand sculpting for a small set of expressions and is further augmented using physical simulation. Then, we detail our local geometric indexing scheme and show how it can be used to find the shapes that are most relevant to a bundle given its position. This is followed by a discussion of the various smoothness considerations that are used to inform our approach for spatially blending the relevant shapes across the face to recover a high-resolution dense reconstruction of the full face. Finally, we apply our algorithm to a series of feature film production examples and compare the results to other popular approaches.

- *The authors are with Industrial Light & Magic, San Francisco, CA, 94129. R. Fedkiw is also with the Department of Computer Science, Stanford University, Stanford, CA, 94305. E-mail: mdcong@cs.stanford.edu, lanalan@gmail.com, fedkiw@cs.stanford.edu.*

Manuscript received October 18, 2022; revised June 6, 2023; accepted June 19th, 2023

2 Prior Work

2.1 Capture

High-resolution facial geometry can be captured using dense performance capture techniques such as [5], [6], [7], [8]. However, these methods typically require environments with controlled lighting and dedicated camera hardware. These restrictions, along with the limitations on the actor's motion, often make these techniques unsuitable for on-set capture where an actor often needs to interact with the set and/or other actors. On-set capture typically involves painting a marker pattern on an actor's face and recording the actor's performance with a set of helmet mounted cameras. The markers can be tracked in the resulting footage and triangulated to recover a sparse set of bundle positions that follow the actor's facial performance.

2.2 Reconstruction

In order to animate the neutral mesh of an actor, one could compute a smooth deformation of the face mesh by interpolating the motion of the bundles (see e.g. the corrective shape computed in [2] and the non-rigid ICP approach of [9]). However, this usually results in a deformed mesh that contains too much of the high-frequency detail of the neutral mesh and too little of the high-frequency detail associated with a particular expression. In order to add high frequency details to the reuslting deformed mesh, [9] projects the smoothly deformed mesh towards per-frame dense scans. Several approaches that do not require additional per-frame scans have also been proposed including [10], [11], [12], [13], and [14] which use deformation gradients, a nonlinear shell energy, feature graph edge strains, polynomial displacement maps, and neural networks respectively. Masquerade [15] combines some of these approaches for facial performances solved from helmet mounted cameras. However, such approaches may not remove high-frequency details in the neutral mesh that are not present in the expression. Furthermore, if the smooth deformation interpolates the bundles, the addition of fine scale details in this manner can potentially move the surface farther away from the bundles.

2.3 Blendshapes

Instead of interpolating the motion of the bundles directly, one could use the markers and/or bundles to drive a blendshape facial rig [3] which specifies the deformation of the face as a linear combination of facial shapes. These facial shapes are acquired using dense performance capture (see e.g. [5], [6], [7], [8]) and/or sculpted by an experienced modeler [16], [17]. Then, one can optimize for the shape weights that minimize the differences between the marker and bundle positions and their associated projected surface positions and surface positions respectively on the resulting mesh [2], [18], [19], [20]. Alternatively, one could minimize the difference between synthetic renderings of the face and the corresponding input images (see e.g. [21], [22]). However, such approaches often result in unnatural combinations of shapes with weights that are difficult to interpret [23], [24], [25], [26]. These infeasible combinations can be avoided by experienced animators but are extremely problematic for optimization algorithms. In order for an optimization algorithm to avoid these combinations, one would need to specify all such invalid combinations in the high-dimensional Cartesian space of facial shapes, which is intractable.

2.4 Patch-Based Approaches

The patch-based model of [27] is particularly notable because it uses a smaller number of facial shapes compared to a traditional blendshape rig. Despite the small number of facial shapes, the resulting per-patch shape in this model still lies in the Cartesian product of the input shapes. Thus, as the size of the dataset increases, one would still expect the model to overfit on a per-patch basis. The FaceIK editing technique of [28] also uses a localized blendshape deformation model by adaptively segmenting the face mesh based on user specified control points, solving for blendshape weights for each control point based on its position, and spatially blending the resulting weights across the mesh using a radial basis function. In order to improve sparsity of the blendshape weights and reduce overfitting, blendshapes that are farther away from the control points are penalized. Unlike [28], which uses an interpolatory approach, our approach uses a non-manifold mesh and other considerations to boost the domain from $\mathbb{R}^3$ into higher dimensions. Other localized models have also been proposed such as [29] which uses PCA-based patches.

3 Dataset

Given the high-resolution mesh of an actor in the neutral or rest pose, we construct a dataset of high-quality facial shapes that sufficiently samples the actor's range of motion and expression. We bootstrap this process by acquiring high-resolution facial geometry for a selection of the actor's (extreme) facial poses taken from a range of motion exercise using the Medusa performance capture system [5], [6], [30]. For each facial pose, Medusa both deforms the neutral mesh to the pose based on images from multiple cameras and estimates the cranium rigid frame associated with the deformed mesh. The cranium rigid frame is manually refined (if necessary), validated against the images from each of the cameras, and then used to stabilize the associated deformed mesh. Each stabilized deformed mesh is then stored as a per-vertex displacement from the neutral mesh.

These stabilized facial shapes are further improved using physical simulation. Starting from the high-resolution neutral mesh, we build a simulatable anatomical face model by morphing an anatomically and biomechanically accurate template model following the approach of [31]. Then, we use the art-directed muscle simulation framework of [32] to target each captured facial shape to obtain a corresponding simulated facial shape with improved volume conservation, more realistic stretching, and a more plausible response to contact and collision. The captured and simulated facial shape are then selectively blended together by a modeler to obtain a combined facial shape that incorporates both the high degree of detail obtained from capture as well as the physical accuracy obtained from simulation. Finally, this combined facial shape is further refined by a modeler based on the images in order to resolve any remaining artifacts before being added to the dataset. See [16], [17].

At this point, the dataset consists of facial shapes corresponding to various extreme poses. We augment the dataset with in-betweens to better represent subtle motions and combinations of expressions. To do this, one could construct a blendshape system using the facial shapes already in the dataset and evaluate this blendshape system at fixed intervals in the high-dimensional Cartesian space; however, the resulting in-betweens would suffer from well-known linear blendshape artifacts such as volume loss. Instead, one could use the aforementioned process targeting

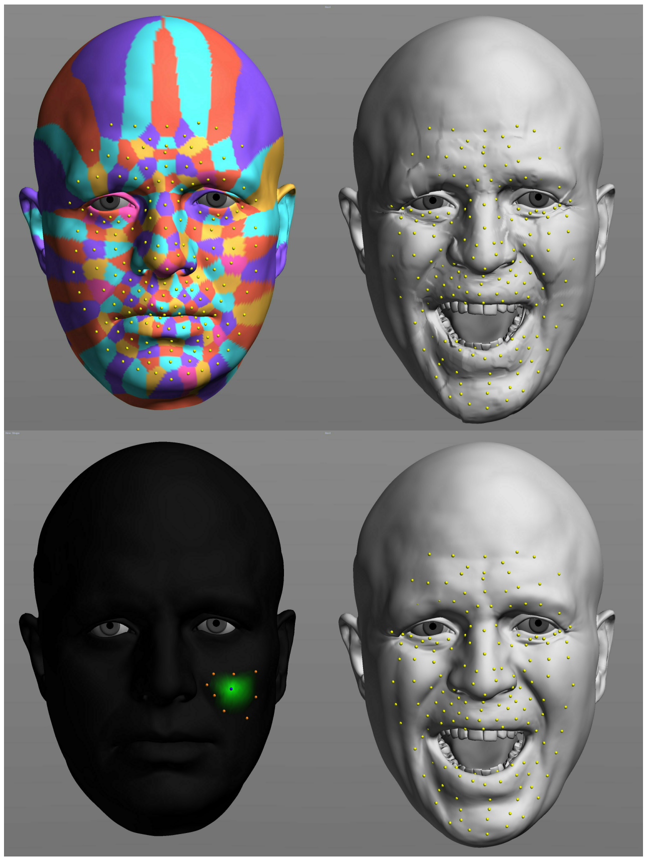

Fig. 1. Top Left: Voronoi diagram on the neutral mesh. Top Right: Applying the locally indexed surface geometry to each Voronoi cell results in discontinuities across cell boundaries. Bottom Left: Natural neighbor weights for a single bundle. The weight is 1 at the bundle surface position and 0 at the surface positions corresponding to neighboring bundles. Bottom Right: Using natural neighbor weights, we obtain a smoother reconstruction that interpolates the bundle positions.

the high-dimensional Cartesian space blendshapes with the art-directed muscle simulation framework of [32], or alternatively one could use the approach of [32] alone to move between various extreme facial poses creating in-betweens. We utilize a combination of these options to add anatomically motivated nonlinear in-betweens to the dataset.

While our algorithm provides the best results with a set of individualized high-quality facial shapes for the actor, it is also possible to use our method with a set of generic facial shapes and just the actor's neutral mesh. However, this can yield lower-quality results due to differences in range of motion and expression. One could mitigate this loss of quality to some degree by transferring an existing set of facial shapes to another actor using approaches such as [33].

4 Local Geometric Indexing

Our local geometric indexing scheme begins by creating a bundle $(\vec{p}_i, \mathcal{C}_i)$ for each painted dot on the actor's face where $\vec{p}_i$ denotes the 3D position of the painted dot in 3D space. Since the dots are painted on and follow the motion of the skin, we also compute correspondences $\mathcal{C}_i$ between the bundles and the actor's face mesh. For each bundle, we find the triangle on the neutral mesh where it is located and calculate the barycentric coordinates of the bundle position with respect to the triangle's vertices. This enables us to

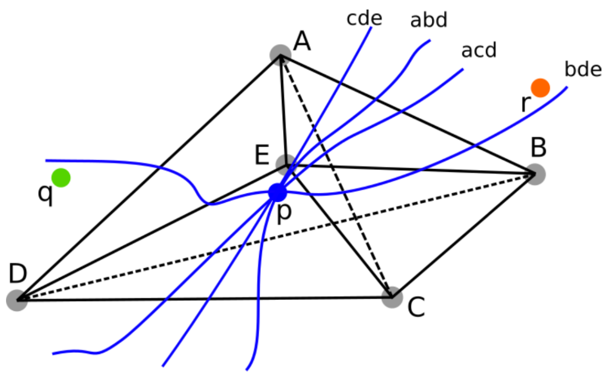

Fig. 2. In the nonoverlapping manifold triangulation of vertices A, B, C, D, and E formed by the solid black lines, interior bundle $\vec{p} \in \mathbb{R}^2$ is located in exactly one triangle CDE and therefore has a unique candidate surface geometry shown by blue line cde. In the overlapping non-manifold triangulation obtained by adding the dashed black lines between AC and DB, $\vec{p}$ is now located in triangles CDE, BDE, ABD, and ACD and has multiple candidate surface geometries shown by the blue lines cde, bde, abd, and acd respectively. Thus, we have removed the uniqueness of the candidate surface geometry with respect to $\mathbb{R}^2$. In order to disambiguate among the candidate surface geometries and minimize kinks in the reconstruction, we choose triangle BDE which yields local surface geometry bde minimizing the distance to neighboring bundles $\vec{q}, \vec{r} \in \mathbb{R}^2$. Consequently, the triangle associated with $\vec{p}$ depends on $\vec{p}$, $\vec{q}$, and $\vec{r}$ which boosts the domain from $\mathbb{R}^2$ to $\mathbb{R}^6$. Note that the nonoverlapping triangulation resulting in cde would have yielded the most discontinuous reconstruction. The generalization to tetrahedra in $\mathbb{R}^3$ is straightforward.

evaluate the surface position of a bundle for a given deformed mesh via barycentric interpolation.

For each bundle, we construct a point cloud $\mathcal{P}_i$ by evaluating the surface position of the bundle on each facial shape in the dataset. The brute force version of our algorithm would tetrahedralize each point cloud $\mathcal{P}_i$ with all possible combinations of four points resulting in a per-bundle non-manifold tetrahedralized volume $\mathcal{V}_i$ (See Sec. 4.1). Then, given an input bundle position $\vec{p}_i$, we find $\mathbb{Q}_i$, the set of all the tetrahedra in the associated tetrahedralized volume $\mathcal{V}_i$ that contain $\vec{p}_i$. Since the tetrahedralized volumes are only dependent on the dataset, this process can be accelerated by precomputing a uniform grid spatial acceleration structure [34], [35].

For each of tetrahedron $(a, b, c, d) \in \mathbb{Q}_i$, we compute the convex barycentric weights $(\lambda_a, \lambda_b, \lambda_c, \lambda_d)$ of the bundle position and use these to blend together the four facial shapes $\vec{s}_a$, $\vec{s}_b$, $\vec{s}_c$, and $\vec{s}_d$ corresponding to the vertices of the tetrahedron. Then, we use these barycentric weights to evaluate candidate surface positions $\vec{x}$ via

$$\vec{x} = \vec{x}_0 + \sum_{k \in \{a,b,c,d\}} \lambda_k \vec{s}_k \tag{1}$$

where $\vec{x}_0$ represents the neutral mesh positions. By construction, the candidate surface geometry is guaranteed to intersect the bundle position and lie within the convex hull of the facial shapes. Repeating this process for each tetrahedron which contains $\vec{p}_i$ yields a set of per-bundle candidate surface geometries from which we choose the local surface geometry following the criteria outlined in Section 5.

If there are no tetrahedra that contain the bundle position, we project the bundle position to the convex hull of the associated

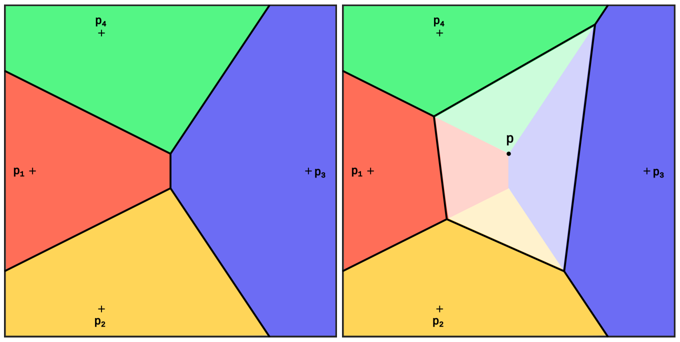

Fig. 3. Left: The Voronoi diagram for bundles $\vec{p}_1$, $\vec{p}_2$, $\vec{p}_3$, and $\vec{p}_4$ consists of the distinctly colored Voronoi cells C_1, C_2, C_3, and C_4 respectively. Right: In order to compute natural neighbor interpolation weights for a given mesh vertex $\vec{p}$, we insert $\vec{p}$ into the Voronoi diagram and recompute the Voronoi cells to obtain $C'_1 \dots C'_4$ and C'. The latter is the Voronoi cell associated with $\vec{p}$ and is shown as the highlighted region that overlaps prior Voronoi cells $C_1 \dots C_4$. The natural neighbor interpolation weight for the inserted vertex $\vec{p}$ with respect to $\vec{p}_j$ is then given by $w_j = \parallel C' \cap C_j \parallel / \parallel C' \parallel, j = 1 \dots 4$ i.e. the area of the intersection of the Voronoi cell C' and the prior Voronoi cell C_j divided by the total area of Voronoi cell C'. It follows that $\sum w_j = 1$ since the sum of the areas of the intersections is equal to the total area of Voronoi cell C'.

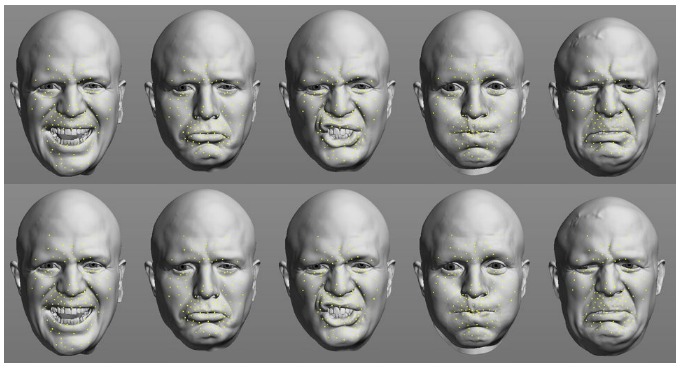

Fig. 4. In order to verify our approach, we input 3D bundle positions from each shape in our dataset into our local geometric indexing algorithm; the results obtained are nearly identical to the original shapes. Top: Shape. Bottom: Local geometric indexing. A video showing the results on the entire dataset is available in the supplementary material.

point cloud by using the barycentric coordinates for the closest point on the associated tetrahedralized volume. The lack of tetrahedra containing a bundle position indicates a need for additional facial shapes in the dataset; however, this projection approach gives reasonable results in such scenarios.

Local geometric indexing can be viewed as a piecewise linear blendshape system, although the pieces are difficult to describe due to overlapping non-manifold tetrahedra and various nonlinear, nonlocal, and higher dimensional strategies for choosing between multiple overlapping tetrahedra. Still, by augmenting the dataset with more in-betweens, we can insert so-called Steiner points [36] allowing for increased efficacy – stressing the importance of collecting more and more data.

4.1 Tetrahedralization

As the size of the point cloud increases, the construction of all possible tetrahedra quickly becomes unwieldy. Thus, we aggressively prune redundancies from the point cloud, e.g. removing points corresponding to expressions that do not involve them. For example, we do not add bundle evaluations to a forehead bundle's point cloud from expressions that only involve the lower half of the face. Besides reducing the number of points, we may also eliminate tetrahedra especially those that are poorly shaped: too thin, too much spatial extent, etc. Moreover, tetrahedra which are known to be problematic admitting shapes that are locally off-model can also be deleted. Similar statements hold for unused or rarely used tetrahedra, etc. Importantly, through continued use and statistical analysis, our tetrahedral database can evolve for increased efficiency and quality. Alternatively, one could construct ad-hoc tetrahedra on-the-fly following the approach of [37] which demonstrates that even a sparse subset of tetrahedra is sufficient to interpolate high-dimensional unstructured data.

Instead of considering all possible combinations of four points, one could tetrahedralize each point cloud using a space-filling tetrahedralization algorithm such as constrained Delaunay tetrahedralization [38]. However, this would restrict a bundle position to lie uniquely within a single tetrahedron and create a bijection between a bundle position and local surface geometry. This is problematic because different expressions can map to the same bundle position with different local curvature. For example, a bundle along the midline of the face on the red lip margin can have the same position during both a smile and a frown. Thus, it is better to construct an overlapping non-manifold tetrahedralization in order to allow for multiple candidate local surface geometries for a bundle position, later disambiguating using additional criteria as discussed in Section 5. Moreover, as discussed later, one may create more than one point cloud for an associated bundle with each point cloud corresponding to different criteria. For example, the shapes one uses for an open jaw could differ significantly when comparing a yawn and an angry yell; different point clouds for sleepy, angry, happy, etc. would help to differentiate in such scenarios.

Again, we stress that a space-filling manifold tetrahedralized volume allows a bundle only three degrees of freedom as it moves through the manifold tetrahedralized volume in $\mathbb{R}^3$, whereas overlapping non-manifold tetrahedra remove uniqueness in $\mathbb{R}^3$ boosting the domain to a higher dimensional space; then, other considerations may be used to ascertain information about other dimensions and select the appropriate tetrahedron.

5 Smoothness Considerations

Our local geometric indexing scheme generates local surface geometry for (the neighborhood of) each bundle independently, and we subsequently sew the local surface geometry together to create a unified reconstruction of the full face. Because only local geometry is required, we only need to store small surface patches (and not the full face geometry) for each point in the point cloud making the method more scalable. To sew the local patches together, we first construct a Voronoi diagram on the neutral mesh using the geodesic distances to the surface position of each bundle in the rest pose. See Figure 1 (Top Left). These geodesic distances are computed using the fast marching method [39]. The local surface geometry for each bundle could then be applied to its associated Voronoi cell on the mesh, although the resulting face shape would typically have discontinuities across Voronoi cell boundaries as shown in Figure 1 (Top Right).

We have experimented with a number of scattered interpolation methods aimed at smoothing the local patches across Voronoi cell faces including, for example, radial basis functions as in

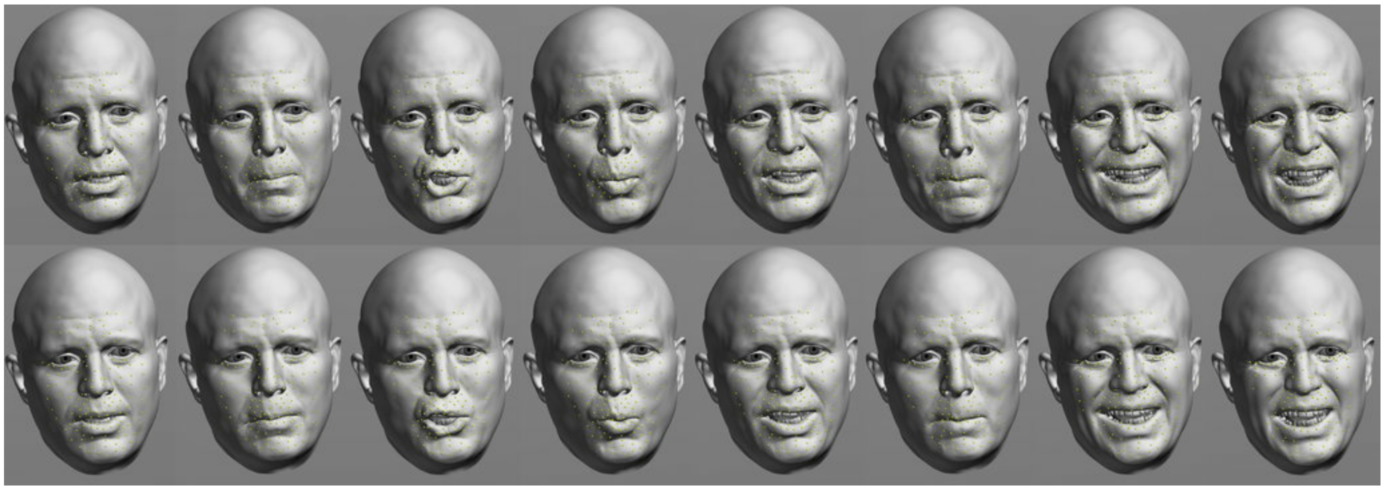

Fig. 5. Top: A high-resolution facial performance processed using the Medusa performance capture system [5], [6], [30]. The shapes from this facial performance are *not* included in our dataset. Bottom: Reconstruction obtained using local geometric indexing driven by the bundle positions on the captured geometry. None of the high-resolution facial shapes in the performance were included in the dataset used by our algorithm. A number of the differences, such as those in the mouth corners and eyebrows, are actually due to artifacts in the Medusa performance capture geometry that are cleaned up by our reconstruction indicating that our approach provides some degree of regularization. The remaining differences are outside of the region spanned by the bundles where we would expect less accuracy due to limited data. A video showing the results on the entire performance is available in the supplementary material.

[27]. We experimentally achieved the best results leveraging our Voronoi diagram using natural neighbor interpolation [40], [41]. For a given vertex on the neutral mesh, natural neighbor weights are computed by inserting the vertex into the precomputed Voronoi diagram, computing the areas stolen by the new vertex's Voronoi cell from each of the pre-existing neighboring Voronoi cells, and normalizing by the total stolen area. See Figure 3 for details. For each vertex, the natural neighbor weights are used to linearly blend the shapes used for each surrounding bundle. Note that a vertex placed at a bundle position would not change the Voronoi cells of surrounding bundles and would merely adopt the Voronoi cell from the bundle it is coincident with; this guarantees that the resulting blended surface still exactly interpolates the bundle positions. In this way, we obtain a C^0 continuous reconstructed surface [42] that passes through all of the bundle positions. See Figure 1 (Bottom Right). We found that constructing the Voronoi diagram and calculating the natural neighbor weights in UV/texture space and subsequently mapping them back onto the 3D mesh yielded smoother natural neighbor weights than performing the equivalent operations on the 3D mesh directly.

5.1 Choosing Tetrahedra

In order to minimize kinks in the C^0 continuous reconstructed surface, we use an additional smoothness criterion when choosing between overlapping tetrahedra and their resulting candidate surface geometries. If there are multiple tetrahedra which contain the bundle position, we choose the tetrahedron that results in local surface geometry that minimizes the distances from neighboring bundle positions to their respective surface positions as shown in Figure 2. This indicates that the local surface geometry is representative of the bundle as well as the neighborhood between the bundle and its neighboring bundles.

In the case where no tetrahedra contain the bundle position, one can apply a similar criterion to project the bundle back to the dataset in a smooth manner. When deciding which tetrahedron to project to, one could consider not only the distance from the bundle under consideration to the resulting surface, but also the distances that neighboring bundles would be from the resulting surface.

In the case of an animated bundle with time-varying position, we apply additional criteria to prevent disjoint sets of shapes from being chosen in neighboring frames, ameliorating undesirable oscillations in the animated reconstructed surface. To do this, we assign higher priority to tetrahedra which share more points and therefore facial shapes with the tetrahedron used on the previous frame, biasing towards a continuous so-called winding number on the non-manifold representation.

6 Jaw Articulation

So far, we have considered facial shapes and bundle positions relative to the neutral mesh. However, these shapes and bundle positions may include displacements due to rotational and prismatic jaw motion [43], [44]. This can result in significant linearized rotation artifacts in the reconstruction which reduces the generalizability of our approach. In order to address this, we hybridize our approach by using linear blend skinning to account for the jaw pose.

To do this, we modify Eq. 1 with a block diagonal matrix of spatially varying invertible transformations $T(\theta)$ calculated using linear blend skinning from the jaw parameters θ and a set of unskinned facial shapes $\vec{s}_k^*$ to obtain

$$\vec{x} = T(\theta)\left(\vec{x}_0 + \sum_{k\in\{a,b,c,d\}} \lambda_k \vec{s}_k^*\right). \tag{2}$$

For a shape with known jaw parameters θ_k, setting Eq. 1 equal to Eq. 2 and rearranging terms gives an expression for the unskinned facial shape

$$\vec{s}_k^* = T(\theta_k)^{-1}\left(\vec{x}_0 + \vec{s}_k\right) - \vec{x}_0$$

as a function of the facial shape $\vec{s}_k$. See [3], [45]. In order to utilize this approach, every shape in the database needs the jaw

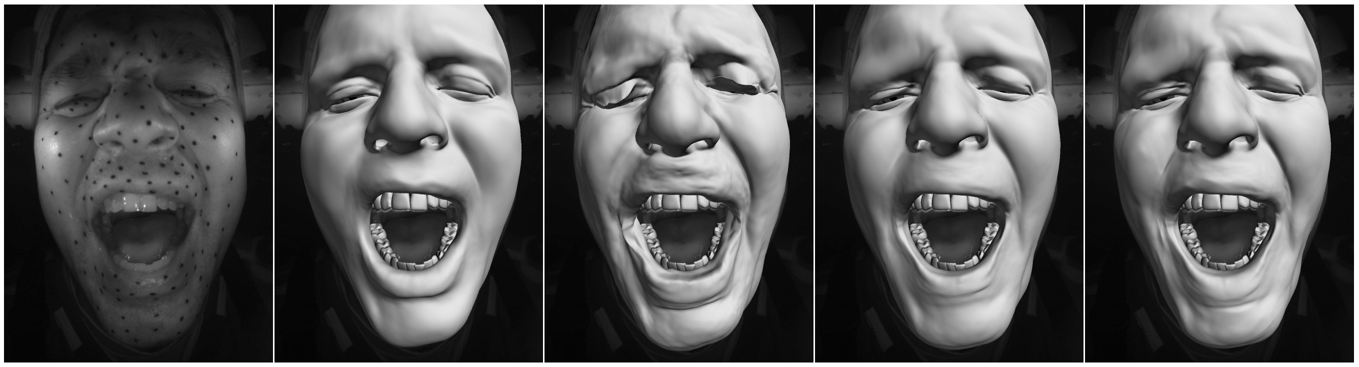

Fig. 6. Far Left: Helmet mounted camera footage. Middle Left: The reconstruction obtained by interpolating the bundle displacements across the mesh using [2] conveys a yawn as opposed to the anger/tension because it does not utilize any additional high-resolution detail beyond that of the neutral mesh. Middle: The typical overfitting symptomatic of blendshape rigs; with enough regularization, one would expect the detail to fade similar to the result using [2]. Middle Right: Using Gaussian RBF interpolation instead of natural neighbor interpolation in our approach results in additional high-resolution detail but does not interpolate the bundle positions. Far Right: Our approach passes through the bundles, conveys the expression, and captures high-resolution details that are not present in the neutral mesh.

parameters θ_k estimated so that we may store $\vec{s}_k^*$ instead of $\vec{s}_k$. Similarly for each frame, θ must be estimated using one of the usual methods for head and jaw tracking so that the bundle positions can be unskinned before indexing into the point cloud.

As mentioned in Sec. 4.1, having a large number of points can result in an unwieldy number of tetrahedra. Thus, one could bin points into different point clouds based on a partition computed using the jaw parameters θ; each point cloud would only contain a range of jaw parameters and would therefore be smaller. Moreover, it makes more sense to interpolate between shapes with similar jaw parameters as opposed to significantly different jaw parameters. One should likely still unskin all of the shapes in the point cloud to have the same jaw parameter value for better efficacy; however, choosing a non-neutral reference shape for the unskinning (e.g. in the middle of the relevant jaw parameter range) could be wise.

7 Experiments

7.1 Verification

In order to verify our algorithm, we calculated a set of 3D bundle positions for each facial shape in our dataset by evaluating the surface position of each bundle on the facial shape. Then, we inputted each set of bundle positions into our local geometric indexing algorithm, and verified that the resulting reconstruction is nearly identical to the original facial shape. See Figure 4.

7.2 High-Resolution Capture Comparison

Next, we evaluate our algorithm on a high-resolution performance outputted from the Medusa performance capture system [5], [6], [30]. The jaw is tracked using the lower teeth during the portions of the performance where they are visible and interpolated to the rest of the performance using the chin bundles as a guide. Like the previous experiment, we calculate a set of 3D bundle positions for each frame of the performance and use this animated set of 3D bundle positions as input into our local geometric indexing algorithm. The resulting high-resolution reconstruction of the performance using our dataset is very similar to the original performance. See Figure 5. The differences in the mouth corners and lips are due to artifacts in the Medusa performance capture. By indexing the most relevant cleaned up shapes in our dataset, we obtain a cleaner reconstruction while also adding detail sculpted

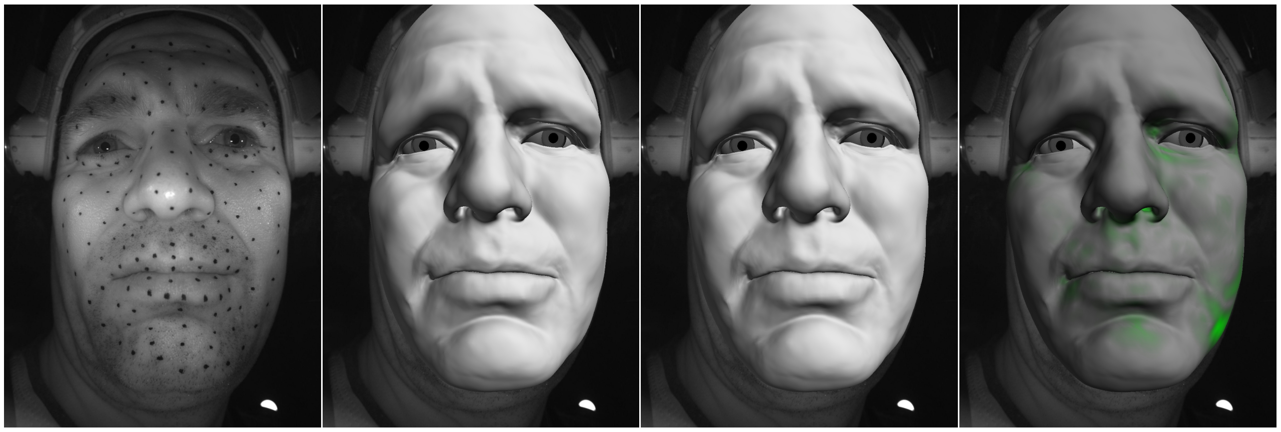

Fig. 7. Far Left: Helmet mounted camera footage. Middle Left: The reconstruction obtained using our approach captures a subtle expression in the helmet mounted camera footage. This performance also shows the effectiveness of our temporal smoothness constraints. See video in supplementary material. Middle Right: Adding simulated in-betweens allows us to improve the smoothness of the reconstruction in the philtrum and the right jowl while also improving the lift in the upper right cheek. Far Right: Heatmap highlighting the differences between (Middle Left) and (Middle Right).

by a modeler such as lip wrinkles. Other differences, such as those on the forehead and side of the face, occur because there are no bundles in those locations and thus our algorithm extrapolates from the nearest bundle.

7.3 Comparison to Other Approaches

In Figure 6, we compare our approach to other popular approaches on a performance captured using two vertically stacked helmet mounted fisheye cameras. Footage from the top camera placed at nose level is shown in Figure 6 (Far Left). The images from both cameras are undistorted and the cameras are calibrated using the markers on the helmet. Given 2D marker positions $\vec{m}_i^{top}$ and $\vec{m}_i^{bottom}$ in each camera obtained from tracking the dot pattern painted on the actor's face, we can estimate the 3D positions of the corresponding bundles $\vec{p}_i$ visible in both cameras using triangulation such that $\vec{p}_i$ minimizes the reprojection error with respect to both $\vec{m}_i^{top}$ and $\vec{m}_i^{bottom}$. The triangulated bundle positions rigidly aligned to the neutral mesh using a combination of the bundles on the nose bridge, forehead, and the cheeks with varying weights based on the amount of non-rigid motion in those regions. The jaw is tracked in the same manner as the previous experiment. In Figure 6 (Middle Left), we calculate bundle displacements by subtracting the neutral pose bundle positions from the triangulated bundle positions followed by interpolating these bundle displacements across the mesh using [2] in order to obtain per-vertex displacements on the mesh. Adding these displacements to the neutral mesh reconstructs a yawn instead of the angry face in the corresponding helmet mounted camera footage because it does not contain any additional high-resolution detail beyond that of the neutral mesh. Since the neutral mesh represents one's face while expressionless, similar to that when asleep, using the displacements relative to the neutral mesh and its features often leads to expressions that appear tired. In order to obtain Figure 6 (Middle), we first constructed a blendshape rig using the facial shapes in our dataset. Then, we solved for the blendshape weights that minimize the Euclidean distances from the bundles to their relevant surface points subject to a soft constraint that penalizes the weights to lie between 0 and 1. The result incorporates more high-resolution details than Figure 6 (Middle) but suffers from overfitting resulting in severe artifacts around the mouth and eyes. Even though the resulting weights lie between 0 and 1, they are neither convex nor sparse which leads to unnatural combinations. Of course, increased regularization would smooth the artifacts shown in the figure creating a result that looks more like Figure 6 (Middle Left). In comparison, the reconstruction obtained using our local geometric indexing algorithm shown in Figure 6 (Far Right) captures many of the high-resolution details that are not present in the neutral mesh including the deepened nasolabial folds, jowl wrinkles, and lip stretching without the overfitting artifacts of Figure 6 (Middle).

7.4 RBF Interpolation

Alternatively, instead of using natural neighbor interpolation, one could use radial basis functions to smooth with our local geometric indexing algorithm. As long as the radial basis function is applied on the facial shape weights as opposed to the vertex positions themselves, this still yields high-resolution features from the dataset in the reconstruction; however, the reconstructed surface will typically not pass through the bundles. This can be corrected by smoothly interpolating the remaining displacements needed to properly interpolate the bundles across the mesh (with e.g. [2]). As shown in Figure 6 (Middle Right), the reconstruction obtained using a combination of radial basis function interpolation and a smoothly interpolated deformation has a higher degree of detail than smoothly interpolating the deformation from neutral mesh. In a similar manner, additional rotoscoped constraints such as lip occlusion contours [2], [46], markers visible in only a single camera, etc. can be incorporated as a postprocessing step on top of our approach; in fact, we utilized [2] to incorporate lip occlusion contours in Figure 6.

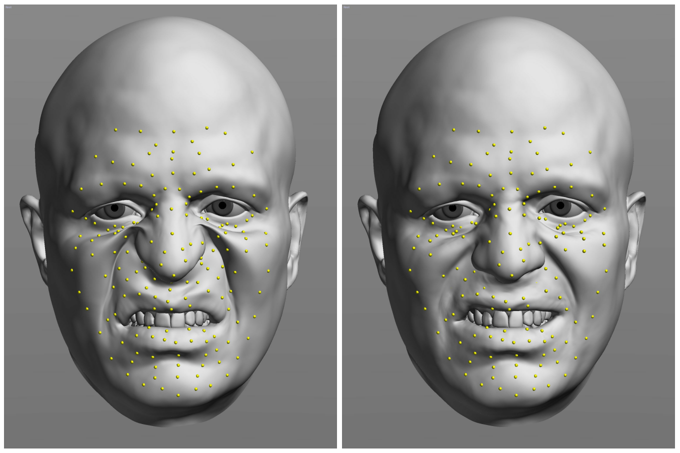

Fig. 8. Left: The combination of sneer, snarl, and upper lip raiser blendshapes leads to severe pinching artifacts in the cheeks and excessive deformation in the nose. These blendshapes are often used in conjunction to animate an angry face. Right: Reconstruction obtained using our local geometric indexing algorithm with the bundles calculated from (Left) as input. The reconstruction fixes the aforementioned artifacts in the cheek and nose, improves the shape of the upper lip, and preserves the emotional intent associated with each of the individual blendshapes.

7.5 Temporal Smoothing

Figure 7 demonstrates the ability of our method to capture subtle expressions while also maintaining temporal coherency in the presence of bundle positions with random and systematic errors (e.g. errors in depth due to the limited parallax between the two cameras). If necessary, one can obtain a smoother performance by either temporally smoothing the input bundle positions or smoothing the barycentric weights on each bundle. In this performance, we apply temporal smoothing by taking a central moving average of the barycentric weights associated with each bundle relative to the jaw skinned neutral mesh in order to avoid smoothing the jaw animation. Because transitions between different sets of shapes typically occur when the same bundle position is achievable using multiple tetrahedra, we found this straightforward temporal smoothing scheme to have negligible impact on the ability for the reconstruction to interpolate the bundles.

7.6 Data Augmentation via Simulation

Figure 7 also illustrates the efficacy of augmenting the dataset using the art-directed muscle simulation framework of [32]. Figure 7 (Middle Left) was the result obtained without augmenting and Figure 7 (Middle Right) was the improved result obtained by adding a number of new facial shapes via [32] as outlined in Section 3.

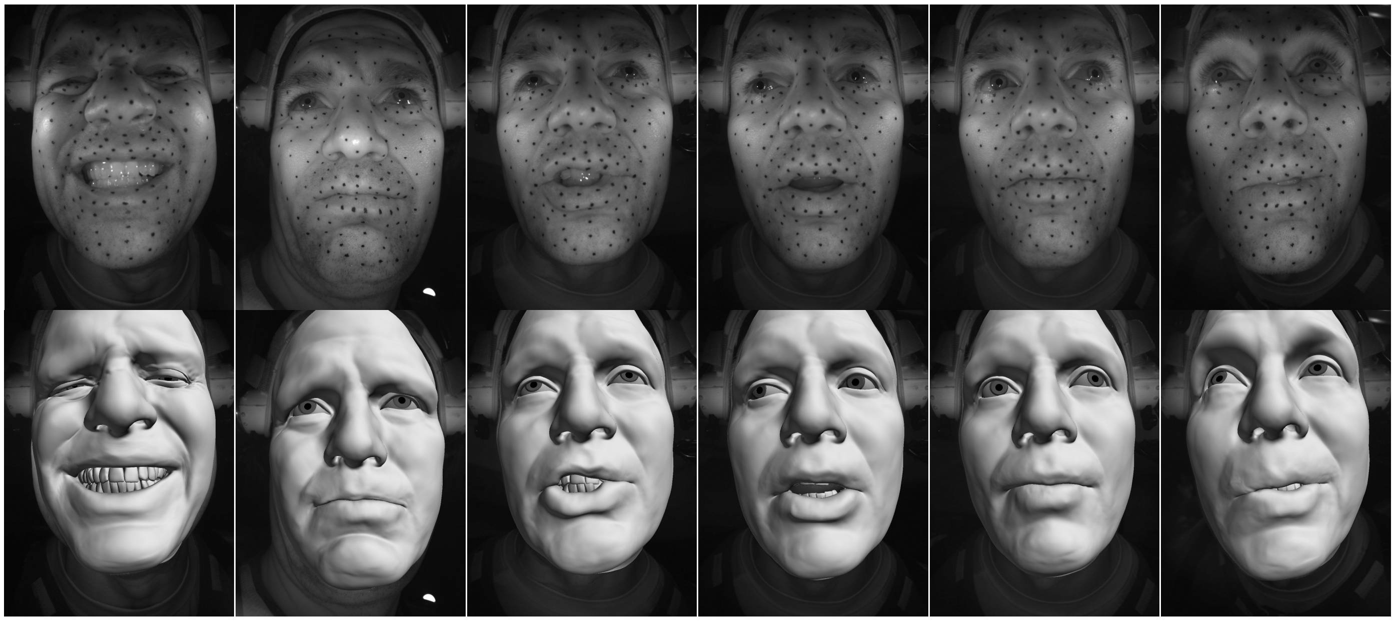

Fig. 9. Top: Helmet mounted camera footage. Bottom: Reconstructions obtained using our method.

7.7 Generating/Correcting Rigs

Facial rigs often use a standardized set of controls for keyframe animation. For example, facial expressions in the mouth region may be split into many different subregions in order to provide fine-tuned localized control to the animator. These synthetically decomposed facial shapes rarely appear on their own during a real-world performance capture session and are usually sculpted by a skilled modeler based on a set of facial scans. Our local geometric indexing algorithm can automate this time intensive process and generate actor-specific facial rigs from a generic template blendshape rig with the standardized set of controls. Given a generic template blendshape rig applied to the actor neutral mesh, we evaluate bundle positions for individual blendshapes and use these bundle positions as input into our local geometric indexing algorithm to reconstruct corresponding actor-specific blendshapes. We apply the same approach to combinations of blendshapes in order to obtain corresponding actor-specific corrective shapes [2] that do not exhibit the artifacts commonly found in combinations of blendshapes. See Figure 8. These actor-specific blendshapes and corrective shapes can be incorporated into an actor-specific nonlinear blendshape facial rig for use in keyframe animation and other facial capture applications.

Unlike many of the prior use cases where the bundle layout is dictated by the on-set facial performance capture, this particular use case allows the user to choose their bundle layout. Thus, it may be possible to further improve the fidelity of the output shapes and correctives by tuning the bundle layout and consequently the associated Voronoi diagram. This process could be accomplished either by hand and/or via an optimization procedure such as [47] and has the potential to improve the semantics of the patches and/or increase the orthogonality of deformations across patches. We defer this avenue of investigation to future work.

7.8 Usability

The natural neighbor weights are a function of the neutral mesh and the bundle positions in the neutral pose and can therefore be precomputed once and reused for each frame of the bundle animation. This precomputation can be futher accelerated using the GPU implementation of natural neighbor interpolation demonstrated in [40]. Furthermore, our local geometric indexing calculations can be performed independently for each bundle and as expected, our parallel CPU implementation using Intel Threading Building Blocks scales linearly yielding runtimes of approximately 1-2 seconds per frame with eight CPU threads. Given this degree of parallelism in the local geometric indexing scheme [40] and the significant speedup of traditional blendshape models on the GPU (see e.g. [48]), our algorithm has the potential to run at interactive rates on the GPU. Already, our approach is general and efficient enough to have been incorporated for use in the production of a major feature film. It has been tested on a wide range of production examples by a number of different users with significant creative and technical evaluation on the results. We show a small selection of our test results in Figure 9.

8 Conclusion

We have presented a data-driven approach for high-resolution facial reconstruction from sparse marker data. Instead of fitting a parameterized (blendshape) model to the input data or smoothly interpolating a surface displacement to the marker positions, we use a local geometric indexing scheme to identify the most relevant shapes from our dataset for each bundle using a variety of different criteria. This yields local surface geometry for each bundle that is then combined to obtain a high-resolution facial reconstruction.

We have applied our method to real-world production helmet mounted camera footage to obtain high-quality reconstructions. Rotoscoped features, including lip occlusion contours, can be readily incorporated as a postprocess. Finally, our approach has already been deployed for use in a film production pipeline for a major feature film where it has been leveraged by many users to obtain production quality results.

Acknowledgements

We would like to thank Cary Phillips, Brian Cantwell, Kevin Sprout, and Industrial Light & Magic for supporting our research into facial performance capture. This research was supported in part by ONR N00014-19-1-2285 and ONR N00014-21-1-2771.

References

[1] L. Williams, “Performance-driven facial animation,” in *Proceedings of the 17th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH ’90. New York, NY, USA: Association for Computing Machinery, 1990, p. 235–242. [Online]. Available: https://doi.org/10.1145/97879.97906

[2] K. S. Bhat, R. Goldenthal, Y. Ye, R. Mallet, and M. Koperwas, “High fidelity facial animation capture and retargeting with contours,” in *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ser. SCA ’13. New York, NY, USA: Association for Computing Machinery, 2013, p. 7–14. [Online]. Available: https://doi.org/10.1145/2485895.2485915

[3] J. P. Lewis, K. Anjyo, T. Rhee, M. Zhang, F. Pighin, and Z. Deng, “Practice and Theory of Blendshape Facial Models,” in *Eurographics 2014 - State of the Art Reports*, S. Lefebvre and M. Spagnuolo, Eds. The Eurographics Association, 2014.

[4] F. I. Parke, “A parametric model for human faces.” Ph.D. dissertation, The University of Utah, 1974, aAI7508697.

[5] T. Beeler, B. Bickel, P. Beardsley, B. Sumner, and M. Gross, “High-quality single-shot capture of facial geometry,” *ACM Trans. Graph.*, vol. 29, no. 4, Jul. 2010. [Online]. Available: https://doi.org/10.1145/1778765.1778777

[6] T. Beeler, F. Hahn, D. Bradley, B. Bickel, P. Beardsley, C. Gotsman, R. W. Sumner, and M. Gross, “High-quality passive facial performance capture using anchor frames,” *ACM Trans. Graph.*, vol. 30, no. 4, Jul. 2011. [Online]. Available: https://doi.org/10.1145/2010324.1964970

[7] G. Fyffe, A. Jones, O. Alexander, R. Ichikari, and P. Debevec, “Driving high-resolution facial scans with video performance capture,” *ACM Trans. Graph.*, vol. 34, no. 1, Dec. 2015. [Online]. Available: https://doi.org/10.1145/2638549

[8] A. Ghosh, G. Fyffe, B. Tunwattanapong, J. Busch, X. Yu, and P. Debevec, “Multiview face capture using polarized spherical gradient illumination,” *ACM Trans. Graph.*, vol. 30, no. 6, p. 1–10, Dec. 2011. [Online]. Available: https://doi.org/10.1145/2070781.2024163

[9] H. Li, B. Adams, L. J. Guibas, and M. Pauly, “Robust single-view geometry and motion reconstruction,” in *ACM SIGGRAPH Asia 2009 Papers*, ser. SIGGRAPH Asia ’09. New York, NY, USA: Association for Computing Machinery, 2009. [Online]. Available: https://doi.org/10.1145/1661412.1618521

[10] A. H. Bermano, D. Bradley, T. Beeler, F. Zund, D. Nowrouzezahrai, I. Baran, O. Sorkine-Hornung, H. Pfister, R. W. Sumner, B. Bickel, and M. Gross, “Facial performance enhancement using dynamic shape space analysis,” *ACM Trans. Graph.*, vol. 33, no. 2, Apr. 2014. [Online]. Available: https://doi.org/10.1145/2546276

[11] B. Bickel, M. Botsch, R. Angst, W. Matusik, M. Otaduy, H. Pfister, and M. Gross, “Multi-scale capture of facial geometry and motion,” in *ACM SIGGRAPH 2007 Papers*, ser. SIGGRAPH ’07. New York, NY, USA: Association for Computing Machinery, 2007, p. 33–es. [Online]. Available: https://doi.org/10.1145/1275808.1276419

[12] B. Bickel, M. Lang, M. Botsch, M. A. Otaduy, and M. Gross, “Pose-Space Animation and Transfer of Facial Details,” in *Eurographics/SIGGRAPH Symposium on Computer Animation*, M. Gross and D. James, Eds. The Eurographics Association, 2008.

[13] W.-C. Ma, A. Jones, J.-Y. Chiang, T. Hawkins, S. Frederiksen, P. Peers, M. Vukovic, M. Ouhyoung, and P. Debevec, “Facial performance synthesis using deformation-driven polynomial displacement maps,” *ACM Trans. Graph.*, vol. 27, no. 5, Dec. 2008. [Online]. Available: https://doi.org/10.1145/1409060.1409074

[14] L. Huynh, W. Chen, S. Saito, J. Xing, K. Nagano, A. Jones, P. Debevec, and H. Li, “Mesoscopic facial geometry inference using deep neural networks,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8407–8416.

[15] L. Moser, D. Hendler, and D. Roble, “Masquerade: Fine-scale details for head-mounted camera motion capture data,” in *ACM SIGGRAPH 2017 Talks*, ser. SIGGRAPH ’17. New York, NY, USA: Association for Computing Machinery, 2017. [Online]. Available: https://doi.org/10.1145/3084363.3085086

[16] M. Cong, L. Lan, and R. Fedkiw, “Muscle simulation for facial animation in kong: Skull island,” in *ACM SIGGRAPH 2017 Talks*, ser. SIGGRAPH ’17. New York, NY, USA: Association for Computing Machinery, 2017. [Online]. Available: https://doi.org/10.1145/3084363.3085040

[17] L. Lan, M. Cong, and R. Fedkiw, “Lessons from the evolution of an anatomical facial muscle model,” in *Proceedings of the ACM SIGGRAPH Digital Production Symposium*, ser. DigiPro ’17. New York, NY, USA: Association for Computing Machinery, 2017. [Online]. Available: https://doi.org/10.1145/3105692.3105693

[18] C. Cao, Y. Weng, S. Lin, and K. Zhou, “3d shape regression for real-time facial animation,” *ACM Trans. Graph.*, vol. 32, no. 4, Jul. 2013. [Online]. Available: https://doi.org/10.1145/2461912.2462012

[19] C. Cao, Q. Hou, and K. Zhou, “Displaced dynamic expression regression for real-time facial tracking and animation,” *ACM Trans. Graph.*, vol. 33, no. 4, Jul. 2014. [Online]. Available: https://doi.org/10.1145/2601097.2601204

[20] H. Li, T. Weise, and M. Pauly, “Example-based facial rigging,” *ACM Trans. Graph.*, vol. 29, no. 4, Jul. 2010. [Online]. Available: https://doi.org/10.1145/1778765.1778769

[21] J. Thies, M. Zollhöfer, M. Stamminger, C. Theobalt, and M. Nießner, “Facevr: Real-time gaze-aware facial reenactment in virtual reality,” *ACM Trans. Graph.*, vol. 37, no. 2, Jun. 2018. [Online]. Available: https://doi.org/10.1145/3182644

[22] H. Kim, P. Garrido, A. Tewari, W. Xu, J. Thies, M. Niessner, P. Pérez, C. Richardt, M. Zollhöfer, and C. Theobalt, “Deep video portraits,” *ACM Trans. Graph.*, vol. 37, no. 4, Jul. 2018. [Online]. Available: https://doi.org/10.1145/3197517.3201283

[23] S. Bouaziz, Y. Wang, and M. Pauly, “Online modeling for realtime facial animation,” *ACM Trans. Graph.*, vol. 32, no. 4, jul 2013. [Online]. Available: https://doi.org/10.1145/2461912.2461976

[24] E. Chuang and C. Bregler, “Performance driven facial animation using blendshape interpolation,” *Computer Science Technical Report, Stanford University*, vol. 2, 01 2002.

[25] R. B. i. Ribera, E. Zell, J. P. Lewis, J. Noh, and M. Botsch, “Facial retargeting with automatic range of motion alignment,” *ACM Trans. Graph.*, vol. 36, no. 4, Jul. 2017. [Online]. Available: https://doi.org/10.1145/3072959.3073674

[26] Y. Seol, J. Lewis, J. Seo, B. Choi, K. Anjyo, and J. Noh, “Spacetime expression cloning for blendshapes,” *ACM Trans. Graph.*, vol. 31, no. 2, Apr. 2012. [Online]. Available: https://doi.org/10.1145/2159516.2159519

[27] C. Wu, D. Bradley, M. Gross, and T. Beeler, “An anatomically-constrained local deformation model for monocular face capture,” *ACM Trans. Graph.*, vol. 35, no. 4, Jul. 2016. [Online]. Available: https://doi.org/10.1145/2897824.2925882

[28] L. Zhang, N. Snavely, B. Curless, and S. M. Seitz, “Spacetime faces: High resolution capture for modeling and animation,” *ACM Trans. Graph.*, vol. 23, no. 3, p. 548–558, Aug. 2004. [Online]. Available: https://doi.org/10.1145/1015706.1015759

[29] J. R. Tena, F. De la Torre, and I. Matthews, “Interactive region-based linear 3d face models,” in *ACM SIGGRAPH 2011 Papers*, ser. SIGGRAPH ’11. New York, NY, USA: Association for Computing Machinery, 2011. [Online]. Available: https://doi.org/10.1145/1964921.1964971

[30] T. Beeler and D. Bradley, “Rigid stabilization of facial expressions,” *ACM Trans. Graph.*, vol. 33, no. 4, Jul. 2014. [Online]. Available: https://doi.org/10.1145/2601097.2601182

[31] M. Cong, M. Bao, J. L. E, K. S. Bhat, and R. Fedkiw, “Fully automatic generation of anatomical face simulation models,” in *Proceedings of the 14th ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, ser. SCA ’15. New York, NY, USA: Association for Computing Machinery, 2015, p. 175–183. [Online]. Available: https://doi.org/10.1145/2786784.2786786

[32] M. Cong, K. S. Bhat, and R. Fedkiw, “Art-directed muscle simulation for high-end facial animation,” in *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ser. SCA ’16. Goslar, DEU: Eurographics Association, 2016, p. 119–127.

[33] R. W. Sumner and J. Popović, “Deformation transfer for triangle meshes,” *ACM Trans. Graph.*, vol. 23, no. 3, p. 399–405, aug 2004. [Online]. Available: https://doi.org/10.1145/1015706.1015736

[34] A. Fujimoto, T. Tanaka, and K. Iwata, “Arts: Accelerated ray-tracing system,” *IEEE Computer Graphics and Applications*, vol. 6, no. 4, pp. 16–26, 1986.

[35] M. Pharr and G. Humphreys, *Physically Based Rendering: From Theory to Implementation*, ser. Morgan Kaufmann series in interactive 3D technology. Elsevier Science, 2010. [Online]. Available: https://books.google.com/books?id=9nJBAJhTxt8C

[36] M. De Berg, M. Van Kreveld, M. Overmars, and O. Schwarzkopf, “Computational geometry: Algorithms and applications,” in *Computational Geometry: Algorithms and Applications*. Springer-Verlag TELOS, 2008.

[37] S. Lüders and K. Dolag, “Psi: Constructing ad-hoc simplices to interpolate high-dimensional unstructured data,” *J. Comput. Phys.*, vol. 467, no. C, oct 2022. [Online]. Available: https://doi.org/10.1016/j.jcp.2022.111476

[38] J. R. Shewchuk, “Constrained delaunay tetrahedralizations and provably good boundary recovery.” in *IMR*. Citeseer, 2002, pp. 193–204.

[39] R. Kimmel and J. A. Sethian, “Computing geodesic paths on manifolds,” *Proceedings of the National Academy of Sciences*, vol. 95, no. 15, pp. 8431–8435, 1998. [Online]. Available: https://www.pnas.org/content/95/15/8431

[40] S. W. Park, L. Linsen, O. Kreylos, J. D. Owens, and B. Hamann, “Discrete sibson interpolation,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 2, p. 243–253, Mar. 2006. [Online]. Available: https://doi.org/10.1109/TVCG.2006.27

[41] R. Sibson, “A vector identity for the dirichlet tessellation,” in *Mathematical Proceedings of the Cambridge Philosophical Society*, vol. 87, no. 1. Cambridge University Press, 1980, pp. 151–155.

[42] B. Piper, *Properties of Local Coordinates Based on Dirichlet Tessellations*. Berlin, Heidelberg: Springer-Verlag, 1993, p. 227–239.

[43] E. Sifakis, I. Neverov, and R. Fedkiw, “Automatic determination of facial muscle activations from sparse motion capture marker data,” *ACM Trans. Graph.*, vol. 24, no. 3, p. 417–425, Jul. 2005. [Online]. Available: https://doi.org/10.1145/1073204.1073208

[44] G. Zoss, D. Bradley, P. Bérard, and T. Beeler, “An empirical rig for jaw animation,” *ACM Trans. Graph.*, vol. 37, no. 4, Jul. 2018. [Online]. Available: https://doi.org/10.1145/3197517.3201382

[45] V. Orvalho, P. Bastos, F. I. Parke, B. Oliveira, and X. Alvarez, “A facial rigging survey.” in *Eurographics (STARs)*, 2012, pp. 183–204.

[46] D. Dinev, T. Beeler, D. Bradley, M. Bächer, H. Xu, and L. Kavan, “User-Guided Lip Correction for Facial Performance Capture,” *Computer Graphics Forum*, 2018.

[47] E. Zell and R. McDonnell, “Compact facial landmark layouts for performance capture,” *Computer Graphics Forum*, vol. 41, no. 2, pp. 121–133, 2022. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.14463

[48] J. Seo, G. Irving, J. P. Lewis, and J. Noh, “Compression and direct manipulation of complex blendshape models,” *ACM Trans. Graph.*, vol. 30, no. 6, p. 1–10, dec 2011. [Online]. Available: https://doi.org/10.1145/2070781.2024198

Matthew Cong received his Ph.D. in Department of Computer Science at Stanford University in 2016 where he was advised by Professor Ron Fedkiw and supported by a NDSEG Fellowship. Currently, Matthew works as a Senior Research Scientist at NVIDIA on the Omniverse team. From 2012 to 2020, he worked as a R&D Engineer at Industrial Light & Magic focusing on facial animation and simulation. He has received screen credits on movies such as ”Kong: Skull Island” and ”Avengers: Endgame”.

Lana Lan worked on creatures and characters at Industrial Light & Magic as a modeler and model supervisor for 17 years before moving to real time character work at Epic Games.

Ronald Fedkiw received his Ph.D. in Mathematics from UCLA and spent part of his postdoctoral studies at Caltech in Aeronautics before joining the Stanford Computer Science Department. He was awarded an Academy Award from the Academy of Motion Picture Arts and Sciences (twice: 2008 and 2015), the National Academy of Science Award for Initiatives in Research, a Packard Foundation Fellowship, a Presidential Early Career Award for Scientists and Engineers (PECASE), a Sloan Research Fellowship, the ACM Siggraph Significant New Researcher Award, an Office of Naval Research Young Investigator Program Award (ONR YIP), the Okawa Foundation Research Grant, the Robert Bosch Faculty Scholarship, the Robert N. Noyce Family Faculty Scholarship, three distinguished teaching awards including the Tau Beta Pi Teaching Excellence Award (for 2020-21), etc. He has published over 135 research papers in computational physics, graphics, learning, and vision, a book on level set methods, and is currently working at the interface between physical simulation and machine learning - having joined the Stanford Artificial Intelligence Laboratory (SAIL) in 2017. Currently, he serves on the editorial board of the Journal of Computational Physics. He was a consultant with Industrial Light + Magic for over 19 years, receiving screen credits on movies such as ”Terminator 3: Rise of the Machines”, ”Star Wars: Episode III - Revenge of the Sith”, ”Poseidon”, ”Evan Almighty”, ”Kong: Skull Island”, etc. Currently, he is a consultant at Epic Games (for 3+ years). He has graduated 38 Ph.D. students so far, and is very proud of their various amazing accomplishments!