Representing 3D Faces with Learnable B-Spline Volumes

Prashanth Chandran Daoye Wang Timo Bolkart
Google
{prchandran, daoye, tbolkart}@google.com

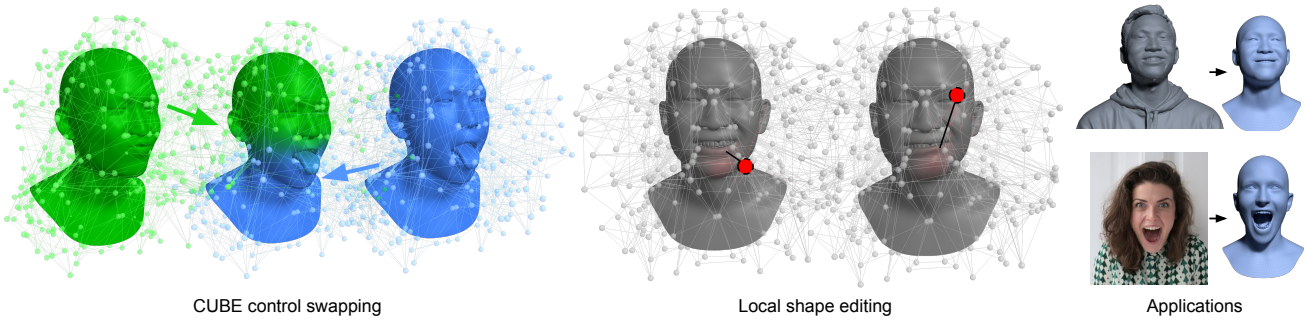

Figure 1. We present CUBE (Control-based Unified B-spline Encoding), a new geometric representation for faces that extends traditional B-Spline volumes with learned features. CUBE's control features locally influence a face shape and therefore allow for precise shape editing by (left) control swapping or (middle) interactive editing. We demonstrate the usefulness of CUBE with two (right) applications; feed-forward facial scan registration where CUBE achieves state-of-the-art results and image-based regression.

Abstract

We present CUBE (Control-based Unified B-spline Encoding), a new geometric representation for human faces that combines B-spline volumes with learned features, and demonstrate its use as a decoder for 3D scan registration and monocular 3D face reconstruction. Unlike existing B-spline representations with 3D control points, CUBE is parametrized by a lattice (e.g., $8 \times 8 \times 8$) of high-dimensional control features, increasing the model's expressivity. These features define a continuous, two-stage mapping from a 3D parametric domain to 3D Euclidean space via an intermediate feature space. First, high-dimensional control features are locally blended using the B-spline bases, yielding a high-dimensional feature vector whose first three values define a 3D base mesh. A small MLP then processes this feature vector to predict a residual displacement from the base shape, yielding the final refined 3D coordinates. To reconstruct 3D surfaces in dense semantic correspondence, CUBE is queried at 3D coordinates sampled from a fixed template mesh. Crucially, CUBE retains the local support property of traditional B-spline representations, enabling local surface editing by updating individual control features. We demonstrate the strengths of this representation by training transformer-based encoders to predict CUBE's control features from unstructured point clouds and monocular images, achieving state-of-the-art scan registration results compared to recent baselines.

1. Introduction

High-fidelity 3D digital humans are central to applications in virtual reality, telepresence, and immersive entertainment. A core challenge in computer vision and graphics is developing a representation that balances computational efficiency with the expressivity required to model diverse identities, poses, and facial expressions.

Current approaches for 3D faces typically rely on explicit surfaces, most notably 3D Morphable Models (3DMMs) [4, 16, 29]. These are the standard for reconstruction [14, 18], tracking [49, 50], speech animation [13, 56], or acting as geometry proxies for neural head avatars [43, 58]. However, 3DMMs are often limited by a fixed mesh topology and low-dimensional parameter spaces that fail to capture person-specific high-frequency details. While learning-based non-linear models [9, 45] offer increased flexibility, they generally lack interpretability and localized control. Conversely, implicit representations [20] provide high surface detail but lack inherent semantic correspondence and require computationally expensive isosurface extraction (e.g., Marching Cubes) for rendering and compatibility with standard graphics pipelines.

To address these limitations, we propose CUBE, a hybrid representation that combines the localized control of B-splines with the expressivity of learning-based models. Inspired by B-spline surfaces [38, Chapter 3] with their localized control through small grids of control points, and the spline-based transformer [10], we leverage B-spline volumes [54]. However, B-spline volumes commonly reconstruct surfaces via isosurfacing. Instead, we utilize the volume to define a continuous mapping from a fixed template mesh to the target geometry, ensuring dense semantic correspondence in the output.

Unlike traditional B-splines that rely on 3D control points, CUBE is parameterized by a lattice of learnable, high-dimensional control features. To evaluate the model, we query the B-spline volume using vertex coordinates from a fixed template mesh. The interpolated high-dimensional features are processed in two stages: the first three dimensions define a coarse base mesh, while the full feature vector is passed through a lightweight Multilayer Perceptron (MLP) to predict fine-scale geometric residuals. This architecture significantly enhances expressivity compared to standard B-splines. Crucially, it retains desirable local support properties, enabling local surface editing by updating individual control features.

We validate this representation by training transformer-based encoders to predict CUBE parameters from unstructured point clouds and monocular images. Our experiments demonstrate state-of-the-art performance in scan registration and reconstruction compared to recent geometric and multi-view baselines.

In summary, our main contributions are:

- CUBE, a novel geometric representation that parameterizes a B-spline volume with high-dimensional learnable control features.
- A hybrid decoding strategy that combines B-spline interpolation with a lightweight MLP to recover fine-scale geometric residuals while maintaining the interpretability and local support of traditional B-splines.
- A framework to predict CUBE control features from unstructured 3D head scans, demonstrating superior registration accuracy over recent learning-based methods.

2. Related work

Common face representations. 3D Morphable Models (3DMMs) [4, 16, 29] based on triangle meshes have remained the go-to method for representing human faces in both academia and industry. 3DMMs for faces represent geometry in a compressed, disentangled latent space and provide a simple linear decoder for shape reconstruction. 3DMMs continue to be used as one of the main driving representations for full head avatars [43, 55, 58]. On the other hand, nonlinear face models [9, 45] rely on the expressivity of neural networks to represent highly detailed faces with a low dimensional latent space. The latent spaces of these neural shape models have also evolved to provide semantic controls [8, 19, 28] similar to traditional 3DMMs. Another emerging paradigm for modeling facial geometries are implicit neural representations [36]. Implicit face models [20, 39, 57] use MLPs to model signed distance fields (SDFs) from which a face mesh can be extracted using marching cubes. The ability to continuously evaluate these models in space makes them an attractive representation for driving full head avatars [21]. In comparison to existing face representations, CUBE is a hybrid representation based on B-Spline volumes. CUBE allows for localized control over an underlying geometry through high-dimensional B-Spline control features, and also uses a lightweight residual MLP to boost the expressivity of the linear B-Spline bases. CUBE can be queried at 3D coordinates sampled from a fixed template mesh, but can also be evaluated continuously like an implicit model.

CAD representations. Non-uniform rational B-Splines (NURBS), on which CUBE is built, are a standard representation for designing and manipulating 3D geometries with high precision in Computer-Aided Design (CAD) [53]. Unlike polygon-based modeling (ex. meshes), NURBS are continuous mathematical objects representing smooth surfaces that can be manipulated interactively through a sparse set of control points. NURBS volumes [24, 53] are an extension of B-Spline surfaces for modeling continuous volumetric fields. For example, Li [31] proposed the use of a NURBS volume as a compact representation of a volumetric scalar field that is optimized to match multiview projection constraints. A shortcoming of NURBS surfaces and volumes is their inability to represent complex geometries such as face shapes with a limited number of control points. They also have limited support in standard rendering software and often have to be discretized into a polygon mesh for downstream use. CUBE replaces the 3D control points used in standard B-Spline volumes with high-dimensional control features, and a neural residual MLP. CUBE can therefore represent complex geometries with only a few control points, while leveraging its residual MLP to model high-frequency details. CUBE is also parameterized with an underlying template mesh, removing the need for an additional surface extraction step to produce polygon meshes and thus can be easily plugged into existing graphics pipelines.

Cage-based deformation. Free-form geometric deformation using volumetric control lattices is a well studied technique in graphics, wherein the vertices of a mesh are obtained through an affine sum of control points in $\mathcal{R}^3$. The weights for a given vertex are determined by mapping its location to the parametric space of the control lattice. Trivariate B-Splines [22] are one of several geometric representations that have been used for this purpose. We refer to this

detailed survey [47] for a comprehensive discussion. Jung *et al*. [25] proposed the use of such a free-form deformation representation as the output representation to regress 3D face shapes from input images. CUBE extends standard free-form deformation for geometry with two main differences. First, we propose the use of high dimensional control lattice along with a refinement MLP to increase the level of detail in the reconstructed geometry. Second, we parameterize the control lattice through a normalized template mesh that allows us to establish correspondences between the control features and known semantic locations on the template mesh.

Applications-specific face representations. In this paper, we demonstrate how CUBE can be useful in practice through two applications: i) facial scan registration and ii) image-based regression. We now provide a short review of the different representations used in these applications.

Facial scan registration: Scan registration deforms a standard template mesh to match a 3D face scan [3]. Unprocessed scans vary drastically in the number and ordering of vertices and are not amenable to statistical analysis, making registration a crucial step for practical utility. Methods are generally classified into optimization-based and learning-based approaches. Salazar *et al*. [46] proposed a method for mesh-based scan registration that automatically annotates sparse landmarks on a scan using a markov network. The annotated landmarks are then used to rigidly align a template mesh with the scan, followed by an iterative optimization with closest point constraints to guide the registration. Topo4D [32] optimizes the properties of 3D Gaussians [26] attached to a template mesh to achieve topologically consistent face meshes and 8K textures from multi-view videos of a single subject.

Learning-based scan registration methods deal with the challenging task of mapping an unstructured representation (i.e. scan) to a structured shape (i.e. template mesh) through a neural network. Prokudin *et al*. [40] introduced Basis Point Sets (BPS), encoding scans with arbitrary point counts into fixed-size embeddings using randomly distributed basis points $\in \mathbb{R}^3$. The embedding consists of the computed distances from each basis point to its closest scan neighbor. This vector is then passed through an MLP to decode the registered mesh. Liu *et al*. [33] employ a PointNet [41] encoder to reduce input point clouds into disentangled identity and expression embeddings. These are transformed into their respective offsets and summed to produce the final shape. Similarly, Shape-my-face [1] also uses a PointNet encoder to produce identity and expression embeddings from a scan, but uses a graph-convolution-based decoder to predict the final shape. ImFace [57] is a morphable model based on implicit neural representations. Trained as an auto-decoder, it predicts the signed distance field value at a 3D query point conditioned on identity and expression vectors.

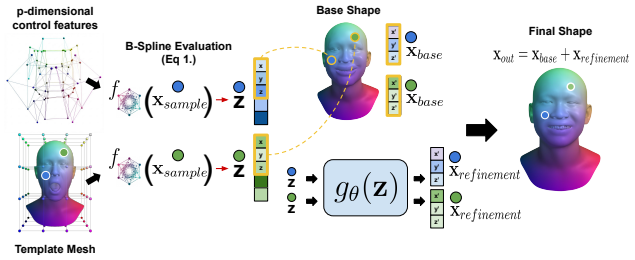

Figure 2. Defined by a lattice of high-dimensional control features, CUBE reconstructs a 3D face in a two-stage process. First, 3D coordinates are sampled from a fixed template mesh. These coordinates are then mapped to high-dimensional features via B-spline interpolation using the lattice of control features. The first three values of the resulting high-dimensional feature vector form a coarse base shape. Finally, the full feature vector is input to a small MLP, which predicts coordinate offsets (residuals) from the base shape, resulting in the refined 3D point coordinates.

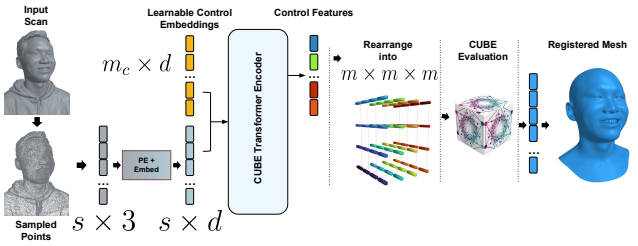

Figure 3. We register scans to a common mesh topology by directly predicting the control features of CUBE. For this, input scan vertices are tokenized, concatenated with trainable control tokens, and then passed through multiple transformer layers. The resulting control token embeddings are extracted and reshaped to form the feature lattice of CUBE. Querying CUBE with a fixed template mesh points then reconstructs the registered 3D mesh.

Image reconstruction. Most methods for monocular [13, 18] and multi-view face reconstruction [6, 27, 30, 34] use triangular meshes as their underlying representation. VHAP [42] is one such approach that fits FLAME [29] and an appearance prior to monocular videos or multi-view images using a differentiable renderer. Bai *et al*. [2] proposed the adaptive face model: a person specific linear shape basis that is applied on top of an underlying 3DMM and is learned from an input video through an iterative non-rigid MVS. Wang *et al*. [51] use a tri-plane representation [7] along with an MLP decoder to fit an implicit neural representation to multi-view images.

3. Method

3.1. B-spline feature volumes

Representation. A NURBS (Non-Uniform Rational B-Spline) feature volume is a generalization of a NURBS vol-

ume [31, 37, 54], where the d-dimensional NURBS output $f(u, v, w)$ represents not 3D spatial coordinates, but an abstract, high-dimensional feature vector. Given a $(m \times m \times m)$ lattice of d-dimensional control features $\mathbf{c}_{ijk} \in \mathbb{R}^d$, each with a control weight $h_{ijk} \in \mathbb{R}$, this NURBS feature volume is defined as a function $f : \mathbb{R}^3 \to \mathbb{R}^d$ that maps a parameter vector (u, v, w) to a d-dimensional feature vector. Formally, f is defined as:

$$f(u, v, w) = \frac{\sum_{i,j,k=1}^{m} N_i^r(u) N_j^r(v) N_k^r(w) h_{ijk} \mathbf{c}_{ijk}}{\sum_{i,j,k=1}^{m} N_i^r(u) N_j^r(v) N_k^r(w) h_{ijk}}. \quad (1)$$

Here, $N_i^r : \mathbb{R} \to \mathbb{R}$ are B-spline basis functions [38, Section 2.2] of degree r. The B-spline basis is defined over a sequence of non-decreasing real numbers $T = \{t_1, ..., t_{m+r+1}\}$, called *knot vector*, that define where and how control features influence the volume. With this knot vector, the B-spline basis function of degree r is defined recursively as $(r \geq 1)$:

$$N_i^r(u) = \frac{u - t_i}{t_{i+r} - t_i} N_i^{r-1}(u) + \frac{t_{i+r+1} - u}{t_{i+r+1} - t_{i+1}} N_{i+1}^{r-1}(u),$$

with the base case $(r = 0)$:

$$N_i^0(u) := \begin{cases} 1 & \text{if } t_i \leq u < t_{i+1}, \\ 0 & \text{otherwise.} \end{cases}$$

One interesting property of the B-spline basis functions is their locality, as the $N_i^r(u) = 0$, for parameters $u \notin [t_i, t_{i+r+1})$. This local support property propagates to the B-spline volume, where each output feature vector is influenced by at most $(r + 1)^3$ control features. For simplicity, the parameters (u, v, w) share the same knot vector $T = \{0, ..., 0, t_{r+2}, ..., t_m, 1, ..., 1\}$, with the first and last $r+1$ values fixed to zero and one, respectively, defining the parameter domain to the interval $[0, 1]$. We choose $r = 2$ for all experiments.

Background. The NURBS volume of Li [31] is a special case of Equation (1) with $d = 1$, resulting in a scalar field output, from which a surface is reconstructed with iso-surface extraction. The NURBS feature volume generalizes the standard NURBS surface [38, Chapter 4] by extending its parameter domain from (u, v) to (u, v, w), and by representing the control features $\mathbf{c}_{ijk}$ in an arbitrary d-dimensional space, rather than constraining it to $d = 3$.

3.2. CUBE

Parametrized by the $(m \times m \times m)$ lattice of d-dimensional control features $\mathbf{c}_{ijk}$, CUBE reconstructs a 3D point $\mathbf{x}_{\text{out}} \in \mathbb{R}^3$ corresponding to a parameter point $\mathbf{x}_{\text{sample}} \in \mathbb{R}^3$. As shown in Fig. 2, this is done in two stages: first, a base shape is reconstructed from the B-spline volume, before in a second stage, this is refined with an MLP.

Coarse base shape. The 3D parametric point $\mathbf{x}_{\text{sample}}$ is first mapped to a d-dimensional feature vector $\mathbf{z}$ through the B-spline volume $f(\mathbf{x}_{\text{sample}}) \to \mathbf{z}$ (Equation (1)). The resulting feature vector is designed to encode both coarse shape and refinement details. The first three dimensions of $\mathbf{z}$ are the 3D coordinates of base point $\mathbf{x}_{\text{base}} := \mathbf{z}_{1:3}$.

Residual displacement. The second stage refines the base point by incorporating high-frequency details encoded in the feature vector $\mathbf{z}$. For this, $\mathbf{z}$ is passed through a lightweight 4-layer MLP $g : \mathbb{R}^d \to \mathbb{R}^3$. This MLP predicts the residual displacement $g(\mathbf{z}) \to \mathbf{x}_{\text{refinement}}$. The final 3D point coordinates $\mathbf{x}_{\text{out}}$ is then computed as the sum of the base point and the refinement, $\mathbf{x}_{\text{out}} = \mathbf{x}_{\text{base}} + \mathbf{x}_{\text{refinement}}$.

Point sampling. To guarantee dense semantic correspondence across different reconstructed 3D face surfaces, each defined by different control features, we query CUBE at fixed parametric coordinates $\mathbf{x}_{\text{sample}}$. To reconstruct surface points, we consider the parametric coordinates $\mathbf{x}_{\text{sample}} \in [0, 1]^3$ derived from the surface of a fixed template mesh (normalized to the unit cube $[0, 1]^3$). By querying CUBE at the vertex locations of the template mesh (i.e., the parameter points $\mathbf{x}_{\text{sample}}$ corresponding to those vertices), we reconstruct 3D meshes with the same mesh topology as the template mesh. Further, by sampling CUBE at non-vertex locations (e.g., intermediate points on the template surface), we can reconstruct 3D points for arbitrary mesh topologies, all while maintaining dense semantic correspondence.

3.3. Mapping scans to CUBE control features

Scans acquired from multi-view capture setups consist of an unordered, variable number of 3D points with diverse positions and orientations in space. Our objective is the registration of such a scan to a common mesh topology using a feed-forward prediction model. This model leverages CUBE as the decoder by directly predicting CUBE's control features, specifically the tensor $\mathbf{c}$, as illustrated in Fig. 3. First, we center the scan by subtracting the mean of its vertices. Unlike prior feed-forward scan registration work [40], the scans are not pre-processed into a canonical orientation. This simplifies the model's application at test time by eliminating the need for additional pre-processing. We optionally apply a 10-band Fourier position embedding [48] on the scan vertices. While this does not impact the model's final accuracy, it accelerates training convergence. The scan vertices are then tokenized into s tokens of size d, where $d \gg 3$. Additionally, we define $m_c = m \times m \times m$ trainable control tokens of size d. These m_c control embeddings are concatenated with the tokenized scan vertices, yielding a final sequence of input tokens of length $m_c + s$. These tokens are subsequently input into a sequence of feed-forward Transformer blocks [9, 11] utilizing XCiT attention [17]. Finally, at the output of the Transformer encoder, we extract the embeddings corresponding to the first m_c tokens.

These extracted embeddings are then reshaped to form the $m \times m \times m$ lattice of CUBE control features. The remaining s scan tokens from the encoder are discarded. The mesh in correspondence is then reconstructed by querying CUBE at the vertex locations of the normalized template mesh.

3.4. Loss functions

We train our model for scan registration by jointly training the encoder and the decoder in a supervised manner. Given a dataset (Sec. 4) of scans and corresponding registrations, we compute an L1 loss on both the predicted base mesh B and the final mesh M with respect to the ground truth registration mesh. Both losses are equally weighted.

4. Implementation details

4.1. Training data

We train CUBE for scan registration using pairs of meshes and scans obtained from a synthetic dataset [12]. Crucially, to avoid the need for manual registration, we produce the scans starting from a randomly sampled mesh using an approach shown in Fig. 4. Our approach is identical to that of Chen *et al*. [12] where a procedural human approach [52] was taken to first create a complete 3D human head with skin textures, clothing, hair, accessories, etc. The procedurally generated head is then rendered using Blender [5] from multiple camera views. These synthetic images are then fed into a multi-view reconstruction pipeline [44] to produce their corresponding scans. Samples from our synthetic dataset are shown in Fig. 4. We generate 300,000 examples and follow a 90:10 train:test split.

4.2. Model and Training Details

The CUBE transformer encoder in Fig. 3 is based on standard Vision Transformer (ViT) [15] sizes. We experiment with three models CUBE-S(mall), CUBE-M(edium), and CUBE-L(arge) which match the specifications of ViT-S, ViT-M, and ViT-L respectively (#layers, #embedding size, #self-attention heads). As training augmentations, we randomly orient the input scan and sample 50,000 points from it at random to form a point cloud. We center the sampled point cloud at the origin by subtracting their mean to result in the final set of input points to our model. As the B-Spline bases only depend on the query point $((u, v, w)$ and the index (i, j, k) of a control point, we can evaluate their value for every sample location on the normalized template shape during model initialization. This makes the B-Spline volume evaluation extremely fast during training. We train our model on 16 TPUs with a batch size of 64 for 250,000 iterations. Training our biggest model CUBE-L with 16^3 controls takes roughly 1 day.

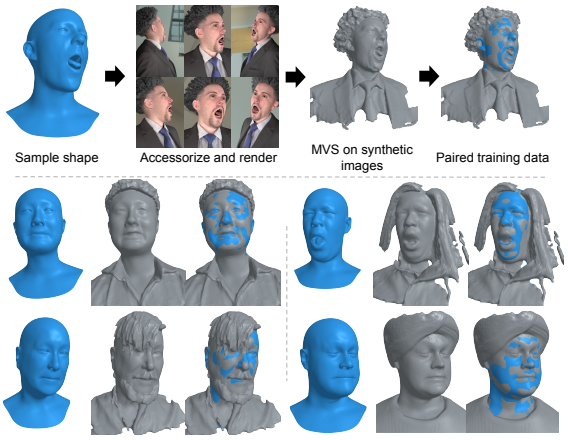

Figure 4. Starting from a randomly sampled mesh, we accessorize it with skin appearance, clothing, and hair and render it from 16 cameras using Blender. The synthetic images along with the cameras are then used to reconstruct a scan [44] to provide paired training data. We show some examples of mesh-scan pairs generated using this approach in the last two rows.

5. Evaluation

Ablation. We perform an ablation study to understand the CUBE representation and analyze the impact of design choices on its representation power. As we know from Sec. 3.2, CUBE is parameterized by the number of control points (m_c), the dimensionality of each control point (d), and the MLP (g). We use scan registration as our application of choice to analyze CUBE. We naturally also take the size of the CUBE transformer encoder into account.

We train multiple scan registration models of different sizes on our synthetic scans dataset (see Sec. 4.1) and evaluate its performance on a test set of 30,000 synthetic scans. In Table. 1 we present the results of this study. We report two different metrics which are the i) point to scan distance (PTS) and the ii) vertex to vertex (V2V) distance. The point to scan distance measures the closest distance between the model's prediction and the input scan, while the vertex to vertex distance measures the difference between the predicted mesh and the ground truth registration. The performance of CUBE scales with the size of the transformer encoder and also the number of control points m_c. The shape offsets predicted by the residual MLP g improve the base shape across all model configurations. In Fig. 5, we can also visually observe that models with a larger encoder and a larger number of control points produce reconstructions that closely match the input scan. From the last row of Fig. 5, we can also see that the magnitude of the predicted residuals increases for models with a lower number of control points (such as 4^3), highlighting the effectiveness of the CUBE residual MLP g in compensating for the limited expressivity of standard B-Spline volumes. From Table. 1,

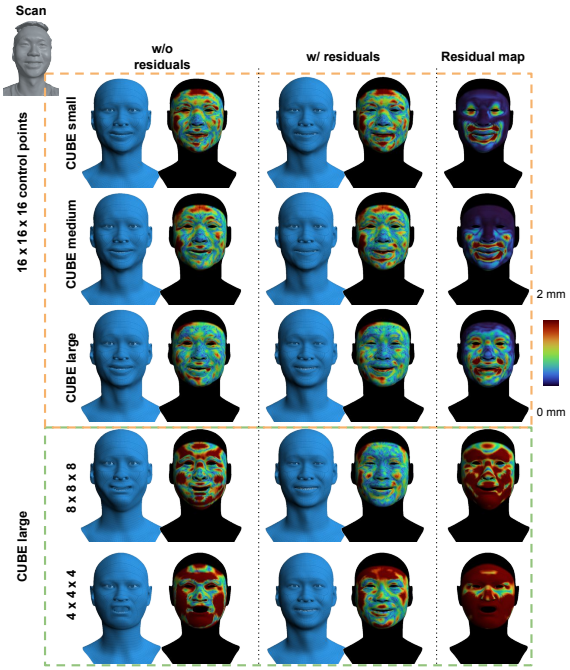

Figure 5. For a test scan shown in the top left, we evaluate CUBE models with encoders of different sizes and a varying number of control points. For each model, we show the predicted mesh and a corresponding error map visualizing the point-to-scan distance. The first columns contain shapes without applying the residuals predicted by the CUBE residual MLP, while the third and fourth columns are of shapes with the residuals applied. The last column visualizes the magnitude of the residuals across the face. The first three rows show models with 16x16x16 control points with encoders of different sizes (small, medium and large), while the last two rows are from a CUBE-Large model with $8 \times 8 \times 8$ and $4 \times 4 \times 4$ control points respectively.

we identify that a CUBE-Large encoder with (16^3) control points gives us the best results. Unless mentioned otherwise, all results in this paper are generated with this model.

Quantitative evaluation. Next, we quantitatively compare CUBE against two feed-forward scan registration baselines. Our first baseline is the Basis Point Set (BPS) method [40], which requires only the raw scan. BPS encodes scans with a variable number of points into fixed-size embeddings using distances to a set of random basis points; these are then processed by an MLP to predict the mesh. For our evaluation, we vary the model capacity by training three BPS models with 512, 1024, and 2048 basis points respectively. Following [40], we sample these points from a uniform sphere tightly enclosing the data. We also compare against TEMPEH [6], a state-of-the-art registration method utilizing multiview images. In contrast to BPS and CUBE, TEMPEH does not take the 3D scan as input. Instead, it extracts per-pixel image features and fuses them across views into a 3D feature volume. The feature volume is processed by a 3D convolutional network that predicts a 'Global' estimate of the face shape. This is followed by a refinement step, where a local volume of features in the neighborhood of every vertex is processed independently by a refinement network to further localize its position in space.

Table 1. **Impact of model sizes, number of control points m_c, and residual offsets.** Ablation study showing the performance of our different model configurations on our test set of synthetic scans. We report point-to-scan (PTS) and vertex-to-vertex (V2V) distances.

Encoder Size	m_c	MLP	PTS (mm)			V2V (mm)		
			Mean	Median	Std	Mean	Median	Std
CUBE-S ($d = 384$)	4^3	w/o	2.59	1.70	2.77	3.66	3.04	2.34
		w/	1.77	0.99	2.32	2.19	1.92	1.24
	8^3	w/o	2.16	1.35	2.49	2.70	2.30	1.66
		w/	1.73	0.94	2.33	2.04	1.78	1.18
	16^3	w/o	1.74	0.95	2.31	2.09	1.81	1.24
		w/	**1.64**	**0.87**	**2.27**	**2.01**	**1.76**	**1.17**
CUBE-M ($d = 512$)	4^3	w/o	2.62	1.72	2.81	3.59	2.94	2.36
		w/	1.68	0.89	2.31	1.95	1.71	1.12
	8^3	w/o	2.16	1.35	2.49	2.61	2.22	1.61
		w/	1.68	0.88	2.31	**1.83**	**1.59**	**1.06**
	16^3	w/o	1.70	0.89	2.32	1.93	1.66	1.18
		w/	**1.63**	**0.84**	**2.29**	1.87	1.62	1.12
CUBE-L ($d = 1024$)	4^3	w/o	2.63	1.73	2.80	3.62	3.01	2.30
		w/	1.62	0.81	2.32	1.81	1.58	1.06
	8^3	w/o	2.04	1.21	2.47	2.44	2.06	1.55
		w/	1.47	0.67	2.27	1.50	1.27	0.94
	16^3	w/o	1.65	0.81	2.33	1.78	1.52	1.11
		w/	1.40	0.67	1.63	1.54	1.31	0.96

For a fair comparison, we train all three methods BPS, TEMPEH and CUBE, on the same training dataset (Sec. 4.1). We test these models on 18,000 scans derived from captures of real humans in a multi-view setup [44]. Our real test set contains a wide range of identities performing a variety of facial expressions. We also run a standard offline registration pipeline to obtain ground truth registrations for these captures.

As we see in Table. 2, our CUBE model for scan registration outperforms previous baselines for feed-forward facial scan registration. We observe that BPS is sensitive to the orientation of the scan and can collapse to average predictions for some scan orientations. TEMPEH-Refined, while achieving metrics comparable to CUBE-S, produces noisy reconstruction. CUBE-M and CUBE-L significantly outperform BPS and TEMPEH in our evaluations.

6. Applications

Localized editing. A key advantage of CUBE is that its control features are localized within the B-Spline vol-

Table 2. **Comparison with State-of-the-Art Methods.** Quantitative results for BPS [40], TEMPEH [6], and CUBE across different model sizes and residual configurations. We report the point-to-scan (PTS) and vertex-to-vertex (V2V) distances.

Method	PTS (mm)			V2V (mm)		
	Mean	Median	Std	Mean	Median	Std
Ground Truth	0.60	0.40	0.72	n/a	n/a	n/a
BPS [40]						
512 points	4.90	4.33	3.35	14.84	14.72	3.92
1024 points	4.66	4.01	3.30	15.16	15.03	3.81
2048 points	5.03	4.21	3.69	17.78	17.79	3.87
TEMPEH [6]						
w/o Refinement	1.45	1.15	1.10	2.46	2.24	1.26
w/ Refinement	1.13	0.80	0.98	2.27	2.01	1.27
(CUBE-S)						
w/o residuals	1.24	0.96	0.99	2.20	1.91	1.30
w/ residuals	1.17	0.90	0.93	2.13	1.86	1.24
(CUBE-M)						
w/o residuals	1.18	0.89	0.97	2.05	1.76	1.24
w/ residuals	1.11	0.85	0.91	1.99	1.72	1.19
(CUBE-L)						
w/o residuals	1.12	0.82	0.96	1.90	1.63	1.17
w/ residuals	0.95	0.71	0.82	1.69	1.45	1.02

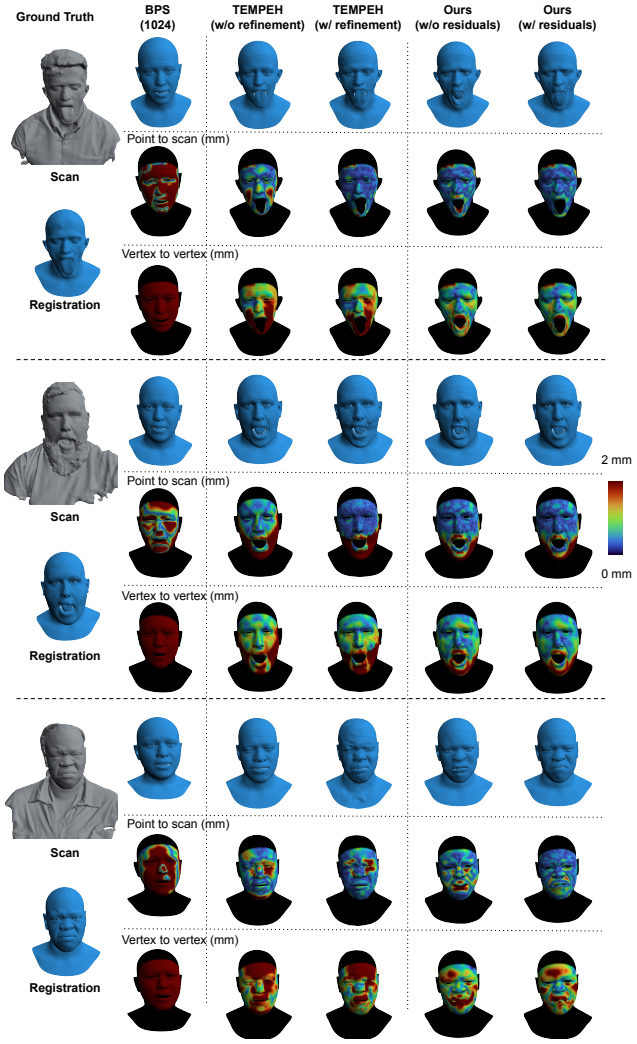

Figure 6. We show qualitative comparisons of BPS [40], TEMPEH [6] and CUBE for predicting registered meshes from three different real scans in various expressions. BPS struggles with scans of varying orientations and collapses to predicting an average face shape that only rigidly matches the location of scan vertices. While TEMPEH-Refined improves on the predictions of its Global stage, it produces noisy reconstructions with geometric artifacts (last example). In contrast, CUBE produces reconstructions with the lowest errors, while matching the identity and expression of the scan closely even for challenging expressions (ex: a person having their tongue out in the first row).

ume. CUBE readily allows a user to perform local edits on a shape by directly modifying its control features. In Fig. 1 (left), we show how CUBE's control features can be swapped across two different expressions represented with CUBE. We replace the control features in the lower half of the volume with those derived from a different expression. CUBE also allows for the editing of individual control points to make precise edits to shapes. In Fig. 1 (middle), we show how displacing a single control feature shown in red, influences the predicted shape locally. To make this local edit, we modify only the first three dimensions of a control feature that corresponds to its location in Euclidean space Fig. 2. The rest of the control feature is left unchanged. As we see in Fig. 1 (middle), modifying one such control point in this manner only influences a local region (in this case, the lower lip). We highlight that CUBE's control features are local by design and do not require any locality-inducing losses or particular types of training data to enable localized control.

Control interpolation. As CUBE's control features have geometric meaning, they interpolate smoothly. In Fig. 7, we encode a source and target scan independently through the CUBE transformer encoder and obtain two sets of control features. We linearly interpolate the control features and evaluate CUBE following Fig. 2 at each interpolated time step to produce the interpolated shapes. In Fig. 7, we see that as the interpolation progresses, the interpolated shape gradually changes in identity and expression from the source to the target, even for extreme facial expressions.

Expression transfer. We can also perform arithmetic operations on our control features to enable applications such as expression transfer from raw scans. Given two scans of the same person (source) in two different expressions, we obtain control features corresponding to both scans from our encoder. We then compute the difference between the control features, apply it as an offset to control features ob-

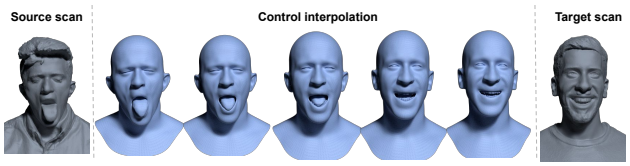

Figure 7. Control features derived from a source and a target scan are linearly interpolated and then evaluated through our CUBE decoder. The shapes resulting from the interpolated control features transition smoothly from the source to the target.

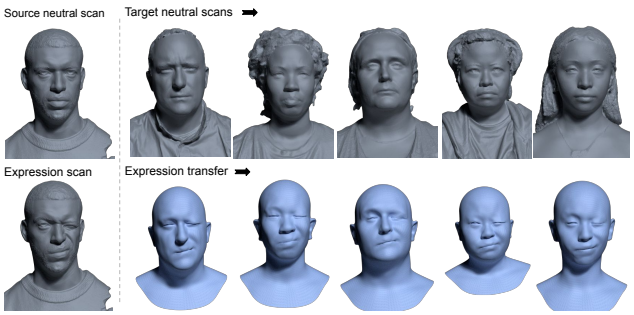

Figure 8. We compute the difference between the control features of a neutral and expression scan of a source subject (first column). This feature difference is then added to control features derived from neutral scans obtained of different target subjects (columns 2-6). In this case, CUBE control arithmetic transfers facial expressions from the source to the target subjects.

tained from a neutral scan of a target subject. In this case, control feature arithmetic results in transferring expressions from the source to target subjects as we can see in Fig. 8.

Generalization to in-the-wild scans. Our method generalizes well to in-the-wild scans produced by other capture setups and MVS algorithms without finetuning. In Fig. 9, we run our CUBE-L model on scans produced by two public datasets; CoMA [45] and FaMoS [6]. Pre-processing is limited to centering each scan at the origin.

Image-based regression. By introducing an additional patchify layer before our CUBE transformer encoder in Fig. 3, we can process tokenized image patches instead of a point cloud, resulting in a ViT-like encoder with a CUBE latent space. With this simple adaptation, we can leverage the flexibility of CUBE representations for image-based regression. In Fig. 10, we show results of training an image-based CUBE transformer encoder for a face reconstruction from in-the-wild images. More details and results can be found in our supplemental material.

7. Conclusion

We presented CUBE, a novel geometric representation for 3D faces, which successfully hybridizes traditional B-spline volumes with high-dimensional learned control features. CUBE is parameterized by a lattice of high-dimensional

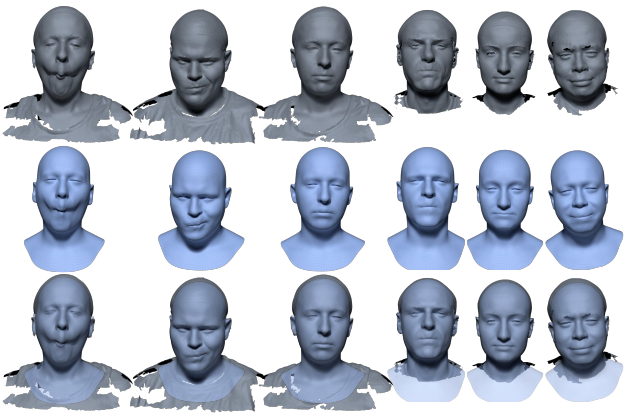

Figure 9. The CUBE scan encoder generalizes to scans captured by different setups. Here we show scans from two different datasets CoMA [45] (columns 1-3) and FaMoS [6] (columns 4-6). The input scans are shown in the first row, the predictions from our CUBE-L model are shown in the second row, and an overlay of the predictions with the input scan is shown in the third row.

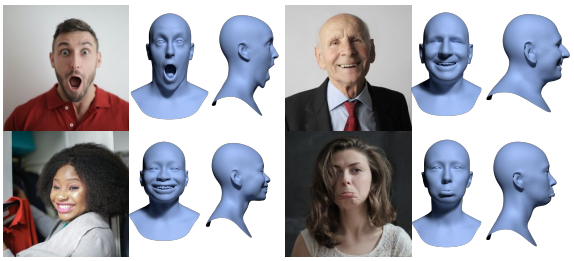

Figure 10. CUBE can also be leveraged for image understanding tasks like in-the-wild face reconstruction. CUBE is versatile and expressive enough to capture a wide range of identities and expressions without relying on an underlying 3DMM.

control features, and to reconstruct a 3D surface, it is continuously evaluated on the surface of a fixed template mesh. This continuous evaluation ensures dense semantic correspondence in the resulting 3D face meshes and enables the reconstruction of different mesh topologies simply by adjusting the point sampling. We present a transformer-based model to regress the CUBE representation from unstructured point clouds and portrait images. When predicted by our transformer-based encoder, the CUBE representation achieves state-of-the-art accuracy in feed-forward facial scan registration. By leveraging B-spline basis functions, CUBE's control features have localized control over the surface, allowing for the local editing of the reconstructed meshes by manipulating individual control features. Overall, we demonstrated that CUBE is a versatile representation, with promising properties that can benefit future applications in the modeling and animation of digital humans.

References

[1] Mehdi Bahri, Eimear O' Sullivan, Shunwang Gong, Feng Liu, Xiaoming Liu, Michael M. Bronstein, and Stefanos Zafeiriou. Shape my face: Registering 3D face scans by surface-to-surface translation. *International Journal of Computer Vision (IJCV)*, 129(9):2680–2713, 2021. 3

[2] Ziqian Bai, Zhaopeng Cui, Jamal Ahmed Rahim, Xiaoming Liu, and Ping Tan. Deep facial non-rigid multi-view stereo. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5849–5859, 2020. 3

[3] Thabo Beeler, Bernd Bickel, Paul Beardsley, Bob Sumner, and Markus Gross. High-quality single-shot capture of facial geometry. *ACM Trans. on Graphics (Proc. SIGGRAPH)*, 29 (3):40:1–40:9, 2010. 3

[4] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3D faces. In *SIGGRAPH*, pages 187–194, 1999. 1, 2

[5] Blender Foundation. Cycles renderer. 5

[6] Timo Bolkart, Tianye Li, and Michael J. Black. Instant multi-view head capture through learnable registration. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 768–779, 2023. 3, 6, 7, 8

[7] Eric R. Chan, Connor Z. Lin, Matthew A. Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas Guibas, Jonathan Tremblay, Sameh Khamis, Tero Karras, and Gordon Wetzstein. Efficient geometry-aware 3D generative adversarial networks. In *CVPR*, 2022. 3

[8] P. Chandran, D. Bradley, M. Gross, and T. Beeler. Semantic deep face models. In *International Conference on 3D Vision (3DV)*, pages 345–354, Los Alamitos, CA, USA, 2020. IEEE Computer Society. 2

[9] Prashanth Chandran, Gaspard Zoss, Markus Gross, Paulo F. U. Gotardo, and Derek Bradley. Shape transformers: Topology-independent 3D shape models using transformers. *Computer Graphics Forum (CGF)*, 41(2):195–207, 2022. 1, 2, 4

[10] Prashanth Chandran, Agon Serifi, Markus Gross, and Moritz Bächer. Spline-based transformers. In *European Conference on Computer Vision (ECCV)*, 2024. 2

[11] Prashanth Chandran, Loïc Ciccone, Gaspard Zoss, and Derek Bradley. Neural Facial Deformation Transfer. In *Eurographics 2025 - Short Papers*. The Eurographics Association, 2025. 4

[12] Victoria Yue Chen, Daoye Wang, Stephan Garbin, Jan Bednarik, Sebastian Winberg, Timo Bolkart, and Thabo Beeler. Pixels2Points: Fusing 2D and 3D Features for Facial Skin Segmentation. In *Eurographics 2025 - Short Papers*. The Eurographics Association, 2025. 5

[13] Radek Daněček, Kiran Chhatre, Shashank Tripathi, Yandong Wen, Michael Black, and Timo Bolkart. Emotional speech-driven animation with content-emotion disentanglement. In *SIGGRAPH Asia Conference Papers*. ACM, 2023. 1, 3

[14] Yu Deng, Jiaolong Yang, Sicheng Xu, Dong Chen, Yunde Jia, and Xin Tong. Accurate 3D face reconstruction with weakly-supervised learning: From single image to image set. In *Computer Vision and Pattern Recognition Workshops*, pages 285–295, 2019. 1

[15] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2021. 5, 12

[16] Bernhard Egger, William A. P. Smith, Ayush Tewari, Stefanie Wuhrer, Michael Zollhoefer, Thabo Beeler, Florian Bernard, Timo Bolkart, Adam Kortylewski, Sami Romdhani, Christian Theobalt, Volker Blanz, and Thomas Vetter. 3D morphable face models - past, present and future. *Transactions on Graphics (TOG)*, 39(5), 2020. 1, 2

[17] Alaaeldin El-Nouby, Hugo Touvron, Mathilde Caron, Piotr Bojanowski, Matthijs Douze, Armand Joulin, Ivan Laptev, Natalia Neverova, Gabriel Synnaeve, Jakob Verbeek, et al. Xcit: Cross-covariance image transformers. *arXiv preprint arXiv:2106.09681*, 2021. 4

[18] Yao Feng, Haiwen Feng, Michael J. Black, and Timo Bolkart. Learning an animatable detailed 3D face model from in-the-wild images. *Transactions on Graphics, (Proc. SIGGRAPH)*, 40(8), 2021. 1, 3

[19] Simone Foti, Bongjin Koo, Danail Stoyanov, and Matthew J Clarkson. 3d shape variational autoencoder latent disentanglement via mini-batch feature swapping for bodies and faces. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18730–18739, 2022. 2

[20] Simon Giebenhain, Tobias Kirschstein, Markos Georgopoulos, Martin Rünz, Lourdes Agapito, and Matthias Nießner. Learning neural parametric head models. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21003–21012. IEEE, 2023. 1, 2

[21] Simon Giebenhain, Tobias Kirschstein, Martin Rünz, Lourdes Agapito, and Matthias Nießner. Npga: Neural parametric gaussian avatars. In *SIGGRAPH Asia 2024 Conference Papers (SA Conference Papers '24), December 3-6, Tokyo, Japan*, 2024. 2

[22] Josef Griessmair and Werner Purgathofer. Deformation of Solids with Trivariate B-Splines. In *EG 1989-Technical Papers*, 1989. 2

[23] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. *arXiv:2111.06377*, 2021. 12

[24] Zhou Jianwen, Lin Feng, and Seah Hock Soon. A volume modeling component of cad. In *Proceedings of the 2001 Eurographics Conference on Volume Graphics*, page 103–117, Goslar, DEU, 2001. Eurographics Association. 2

[25] Harim Jung, Myeong-Seok Oh, and Seong-Whan Lee. Learning free-form deformation for 3d face reconstruction from in-the-wild images. In *2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 2737–2742, 2021. 3

[26] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42 (4), 2023. 3

[27] Jing Li, Di Kang, and Zhenyu He. GRAPE: generalizable and robust multi-view facial capture. In *European Confer-*

ence on Computer Vision (ECCV), pages 403–418. Springer, 2024. 3

[28] Ruilong Li, Karl Bladin, Yajie Zhao, Chinmay Chinara, Owen Ingraham, Pengda Xiang, Xinglei Ren, Pratusha Prasad, Bipin Kishore, Jun Xing, and Hao Li. Learning Formation of Physically-Based Face Attributes . In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3407–3416, Los Alamitos, CA, USA, 2020. IEEE Computer Society. 2

[29] Tianye Li, Timo Bolkart, Michael. J. Black, Hao Li, and Javier Romero. Learning a model of facial shape and expression from 4D scans. *Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 36(6):194:1–194:17, 2017. 1, 2, 3

[30] Tianye Li, Shichen Liu, Timo Bolkart, Jiayi Liu, Hao Li, and Yajie Zhao. Topologically consistent multi-view face inference using volumetric sampling. In *International Conference on Computer Vision (ICCV)*, pages 3824–3834, 2021. 3

[31] Wanwan Li. Multi-view NURBS volume. In *International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP)*, pages 228–235. SCITEPRESS, 2022. 2, 4

[32] Xuanchen Li, Yuhao Cheng, Xingyu Ren, Haozhe Jia, Di Xu, Wenhan Zhu, and Yichao Yan. Topo4D: Topology-preserving gaussian splatting for high-fidelity 4D head capture. In *European Conference on Computer Vision (ECCV)*, pages 128–145. Springer, 2024. 3

[33] Feng Liu, Luan Tran, and Xiaoming Liu. 3D face modeling from diverse raw scan data. In *International Conference on Computer Vision (ICCV)*, pages 9407–9417, 2019. 3

[34] Shichen Liu, Yunxuan Cai, Haiwei Chen, Yichao Zhou, and Yajie Zhao. Rapid face asset acquisition with recurrent feature alignment. *Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 41(6):214:1–214:17, 2022. 3

[35] Ilya Loshchilov and Frank Hutter. Fixing weight decay regularization in adam. *CoRR*, abs/1711.05101, 2017. 12, 14

[36] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, Los Alamitos, CA, USA, 2019. IEEE Computer Society. 2

[37] SK Park. Volumetric NURBS representation of multidimensional and heterogeneous objects: Concepts and formation. *Korean Journal of Computational Design and Engineering*, 10(5):303–313, 2005. 4

[38] Les Piegl and Wayne Tiller. *The NURBS book*. Springer Science & Business Media, 1997. 2, 4

[39] Rolandos Alexandros Potamias, Stathis Galanakis, Jiankang Deng, Athanasios Papaioannou, and Stefanos Zafeiriou. Imhead: A large-scale implicit morphable model for localized head modeling. In *ICCV*, 2025. 2

[40] Sergey Prokudin, Christoph Lassner, and Javier Romero. Efficient learning on point clouds with basis point sets. In *International Conference on Computer Vision (ICCV)*, pages 4332–4341, 2019. 3, 4, 6, 7

[41] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 3

[42] Shenhan Qian. VHAP: Versatile head alignment with adaptive appearance priors, 2024. 3

[43] Shenhan Qian, Tobias Kirschstein, Liam Schoneveld, Davide Davoli, Simon Giebenhain, and Matthias Nießner. GaussianAvatars: Photorealistic head avatars with rigged 3D gaussians. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20299–20309, 2024. 1, 2

[44] Di Qiu, Yinda Zhang, Thabo Beeler, Vladimir Tankovich, Christian Häne, Sean Fanello, Christoph Rhemann, and Sergio Orts-Escolano. CHOSEN: Contrastive hypothesis selection for multi-view depth refinement. In *ECCV-W*, 2024. 5, 6

[45] Anurag Ranjan, Timo Bolkart, Soubhik Sanyal, and Michael J. Black. Generating 3D faces using convolutional mesh autoencoders. In *European Conference on Computer Vision (ECCV)*, pages 725–741, 2018. 1, 2, 8

[46] Augusto Salazar, Stefanie Wuhrer, Chang Shu, and Flavio Prieto. Fully automatic expression-invariant face correspondence. *Machine Vision and Applications*, 25(4):859–879, 2014. 3

[47] D. Ströter, J. M. Thiery, K. Hormann, J. Chen, Q. Chang, S. Besler, J. S. Mueller-Roemer, T. Boubekeur, A. Stork, and D. W. Fellner. A survey on cage-based deformation of 3d models. *Computer Graphics Forum*, 43(2):e15060, 2024. 3

[48] Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. In *Advances in Neural Information Processing Systems (NeurIPS)*, Red Hook, NY, USA, 2020. Curran Associates Inc. 4, 12

[49] Felix Taubner, Prashant Raina, Mathieu Tuli, Eu Wern Teh, Chul Lee, and Jinmiao Huang. 3D face tracking from 2D video through iterative dense UV to image flow. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1227–1237, 2024. 1

[50] Justus Thies, Michael Zollhöfer, Marc Stamminger, Christian Theobalt, and Matthias Nießner. Face2Face: Real-time face capture and reenactment of RGB videos. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2387–2395, 2016. 1

[51] Yating Wang, Ran Yi, Xiaoning Lei, Ke Fan, Jinkun Hao, and Lizhuang Ma. Reconstructing topology-consistent face mesh by volume rendering from multi-view images, 2025. 3

[52] Erroll Wood, Tadas Baltrušaitis, Charlie Hewitt, Sebastian Dziadzio, Thomas J Cashman, and Jamie Shotton. Fake it till you make it: face analysis in the wild using synthetic data alone. In *ICCV*, pages 3681–3691, 2021. 5

[53] Zhongke Wu, Hock Soon Seah, and Feng Lin. *NURBS Volume for Modelling Complex Objects*, pages 159–167. Springer London, London, 2000. 2

[54] Zhongke Wu, Hock Soon Seah, and Feng Lin. *NURBS Volume for Modelling Complex Objects*, pages 159–167. Springer London, 2000. 2, 4

[55] Jun Xiang, Xuan Gao, Yudong Guo, and Juyong Zhang. Flashavatar: High-fidelity head avatar with efficient gaussian

embedding. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 2

[56] Qingcheng Zhao, Pengyu Long, Qixuan Zhang, Dafei Qin, Han Liang, Longwen Zhang, Yingliang Zhang, Jingyi Yu, and Lan Xu. Media2Face: Co-speech facial animation generation with multi-modality guidance. In *SIGGRAPH Conference Papers*, page 18. ACM, 2024. 1

[57] Mingwu Zheng, Hongyu Yang, Di Huang, and Liming Chen. ImFace: A nonlinear 3D morphable face model with implicit neural representations. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20311–20320, 2022. 2, 3

[58] Wojciech Zielonka, Timo Bolkart, and Justus Thies. Instant volumetric head avatars. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4574–4584. IEEE, 2023. 1, 2

Supplemental: Representing 3D Faces with Learnable B-Spline Volumes

Prashanth Chandran Daoye Wang Timo Bolkart
Google
{prchandran, daoye, tbolkart}@google.com

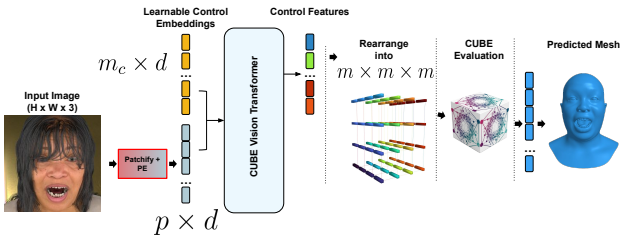

Figure S1. CUBE control features can be regressed from images by concatenating m_c learnable control embeddings to image patch tokens, and processing them with a standard vision transformer. With the exception of the *patchify* layer before the transformer encoder, our architecture for image-based regression is identical to the model we introduced for scan registration in the main paper. For image-based regression, we used a ViT-Large backbone with $8 \times 8 \times 8$ control tokens.

S1. Additional Implementation Details

S1.1. Image-based Regression

Architecture. As we briefly mentioned in the main paper, CUBE can also be used as the output representation when regressing 3D faces from an RGB image. To achieve this, we pass an input image $(H \times W \times 3)$ through a standard *patchify* layer and obtain patch tokens of shape $(p \times d)$. A learnable position encoding is applied to these patch tokens [15]. Similar to the CUBE scan encoder, we append m_c learnable control token embeddings; each of size d, along the sequence length of the p position encoded patch tokens. We then use a standard vision transformer [15] to regress CUBE control features from the input collection of $(p+m_c)$ tokens as shown in Fig. S1. The vision transformer outputs the control features that are evaluated using CUBE as shown in figure 2 of the main paper.

Dataset. We use a synthetic dataset of 500K portraits rendered at a resolution of 224×224 to train our image-to-CUBE model. This synthetic dataset was generated using the same process described in section 4.1 of the main paper. As we are regressing the shape directly from a single input image, we only render an accessorized 3D head from a single randomly placed camera and skip the scan reconstruction step.

Training. We use a ViT-Large model as our encoder and initialize its weights from the official MAE's encoder checkpoint [23] to speed up convergence. We use $8 \times 8 \times 8$ control features for CUBE. We apply standard photometric and geometric augmentations on the input images during training. We train our model with an L1 loss on the predicted shape. We use a batch size of 128 and train the model for 100,000 steps on 8 TPUs using the AdamW optimizer [35] and a learning rate of 1e-4.

S2. Additional Results

S2.1. Performance-cost trade off

Table 1 of the main paper confirms a tradeoff between control point density and the level of detail added by the residual mlp g. With fewer control points (4^3), the difference in the point-to-scan distance between predictions w/ and w/o the refinement MLP is 3-7x larger, depending on the encoder size. The inference times (measured on a V100 GPU) of our models with a varying number of control points ranges from 0.20s (CUBE-S, 4^3) to 2.92s (CUBE-L, 16^3).

S2.2. Sampling robustness for scans

Our model for scan registration is robust to varying sample counts and sampling strategies. In Fig. S2, we sample the same input scan three times to obtain point clouds of different sizes consisting of 25,000, 50,000 and 75,000 points respectively. The sampled point clouds are processed with CUBE-L model with $16 \times 16 \times 16$ control points to obtain the corresponding registered meshes. As we see in Fig. S2, the CUBE scan registration model produces perceptually similar outputs with only minor differences in face shape.

S2.3. Effect of position encoding

In Fig. S3, we compare the training curves of two identical scan-to-mesh models w/ and w/o position encoding (PE) [48] to show its effect on the convergence. We used a CUBE-Large model with $16 \times 16 \times 16$ controls for this experiment. While we did not observe a significant difference in the final training error, we noticed that PE speeds up the convergence of our model in the earlier stages of training.

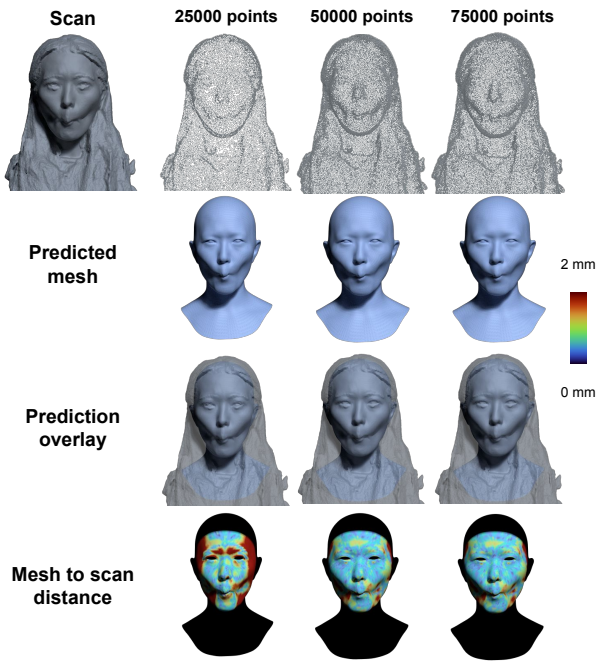

Figure S2. Given a high resolution scan, we randomly sample 25K, 50K and 75K points from the scan to result in point clouds of different sizes as seen in the first row. These sampled point clouds are independently processed by our CUBE-L model to result in the predictions seen in the second row. The third row overlays the predicted meshes on the original scan. The last row shows the mesh to scan distance for the predicted meshes. The predicted shapes remain plausible and capture the geometry of the scan in all cases demonstrating that our model is able to gracefully handle point clouds of different sizes.

Figure S3. Position encoding (PE) the input scan points has a positive effect on the convergence of our CUBE encoder. While the difference in final training error is not significant, PE did seem to accelerate convergence in the earlier iterations of training.

S2.4. Processing scans with colors

Scans captured from multiview setups often contain additional information such as the color for each vertex in the mesh. These per-vertex colors may provide semantic textural cues for scan registration, allowing a model to disambiguate regions in the scan devoid of geometric features. To validate this hypothesis, we train a CUBE scan encoder to take color as an additional input for each sampled scan point. The sampled point positions and their corresponding colors are positionally encoded separately and concatenated before being fed as input to the CUBE transformer encoder. We find that providing the scan colors as an additional input lowers the reconstruction error by roughly 12% on our test set of 18,000 real scans.

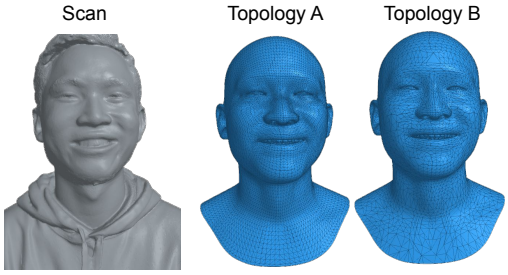

Figure S4. We show an example where by sampling CUBE at different locations on the template mesh surface, we can dictate the topology of the output shape. When combined with our transformer based encoder for scan registration, this allows CUBE to be used not only with input scans with a varying number of points, but also to represent output meshes in different topologies.

S2.5. Topology changes

CUBE is a hybrid representation of geometry that can be evaluated continuously like an implicit representation, while still maintaining correspondence with a template mesh. Therefore CUBE can produce meshes in arbitrary output topologies without any retraining. As we saw in figure 2 of the main paper, CUBE is evaluated in two stages. First, we evaluate a B-Spline volume using the predicted high dimensional control features at locations sampled from a normalized template mesh. This is followed by evaluating a point-wise residual MLP to obtain residual geometric details. By changing the topology of the template shape at inference, we can control the topology in which the output mesh is generated without having to re-train CUBE. In Fig. S4, we show an example of a predicting a registered mesh in two different output topologies from an input scan using the same model. This flexibility makes CUBE a very useful geometric representation in practice.

S2.6. Optimizing CUBE representations

CUBE as a representation can also be optimized or fit to target constraints without an encoder. We present two examples for optimization-based fitting with CUBE.

Fitting to a mesh. Given a target mesh with proxy surfaces for the scalp and the beard (see Fig. S5 left), our objective is to optimize for CUBE's latent control features $\mathbf{c}_{ijk}$ along

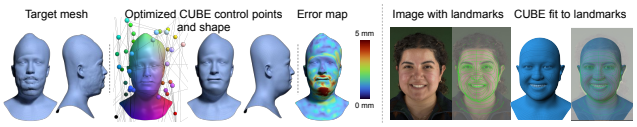

Figure S5. Left: We show how CUBE can be optimized to match the shape of a target mesh with proxy surfaces for the scalp and facial hair. The optimized control points along with the final mesh and the approximation error are also shown. Right: We show the result of fitting CUBE to match detected dense landmarks on an input image. CUBE can reasonably approximate the facial geometry with a small number of $4 \times 4 \times 4$ control features in both cases.

with the residual MLP g to approximate this target mesh as closely as possible. We use $4 \times 4 \times 4$ control features for this experiment where the dimension of each control feature is 16. We randomly initialize the CUBE control features and the MLP g and optimize them to minimize the vertex to vertex distance to the target scan using gradient descent. We use the adamw [35] optimizer with a learning rate of 1e-3 and optimize for 2000 steps. As seen in Fig. S5 (left), CUBE can approximate the facial geometry with proxy surfaces with only a small number of control points.

Fitting to 2D landmarks. We can optimize CUBE to fit detected 2D landmarks on images using a standard landmark re-projection energy (see Fig. S5 right). Given a portrait image of a subject, we run a dense landmark detector to detect 600 landmarks in the image. We optimize for the CUBE control features in camera space so that the recovered shape when projected on the image, matches the detected 2D landmarks. We assume that the intrinsics are fixed for this experiment. The optimization parameters are the same as for the mesh fitting experiment, and we solve for $4 \times 4 \times 4$ control features of dimension 16 using gradient descent. We optimize with a learning rate of 1e-2 for 500 steps.

S2.7. Temporal predictions

Our model produces temporally smooth predictions when applied to scan and image sequences. In Fig. S6 and Fig. S7 we show predictions from our model for sequential scan and image data respectively. Kindly have a look at our supplemental video for more results.

S2.8. Qualitative results

Finally we provide additional qualitative results for our scan registration and image-based face reconstruction models in Fig. S8 and Fig. S9 respectively. These results highlight CUBE's ability to represent diverse face shapes and expressions while also being a continuous and localized shape representation.

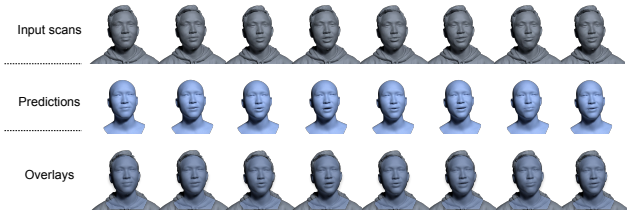

Figure S6. The CUBE scan encoder produces temporally smooth results for input scan sequences and can be used for performance registration. In this figure, we show the input scan, the prediction of our model and an overlay of the prediction on the input scan for a sequence of scans reconstructed from a facial performance of a subject. Note that the scans were processed independently for each frame.

Figure S7. The CUBE image encoder can process frames from a video sequence to reconstruct facial performances in the wild. In this example, we show the captured images in the first row and the corresponding predictions from our model for each image in the second row. Our model produces a faithful reconstruction of the input facial performance.

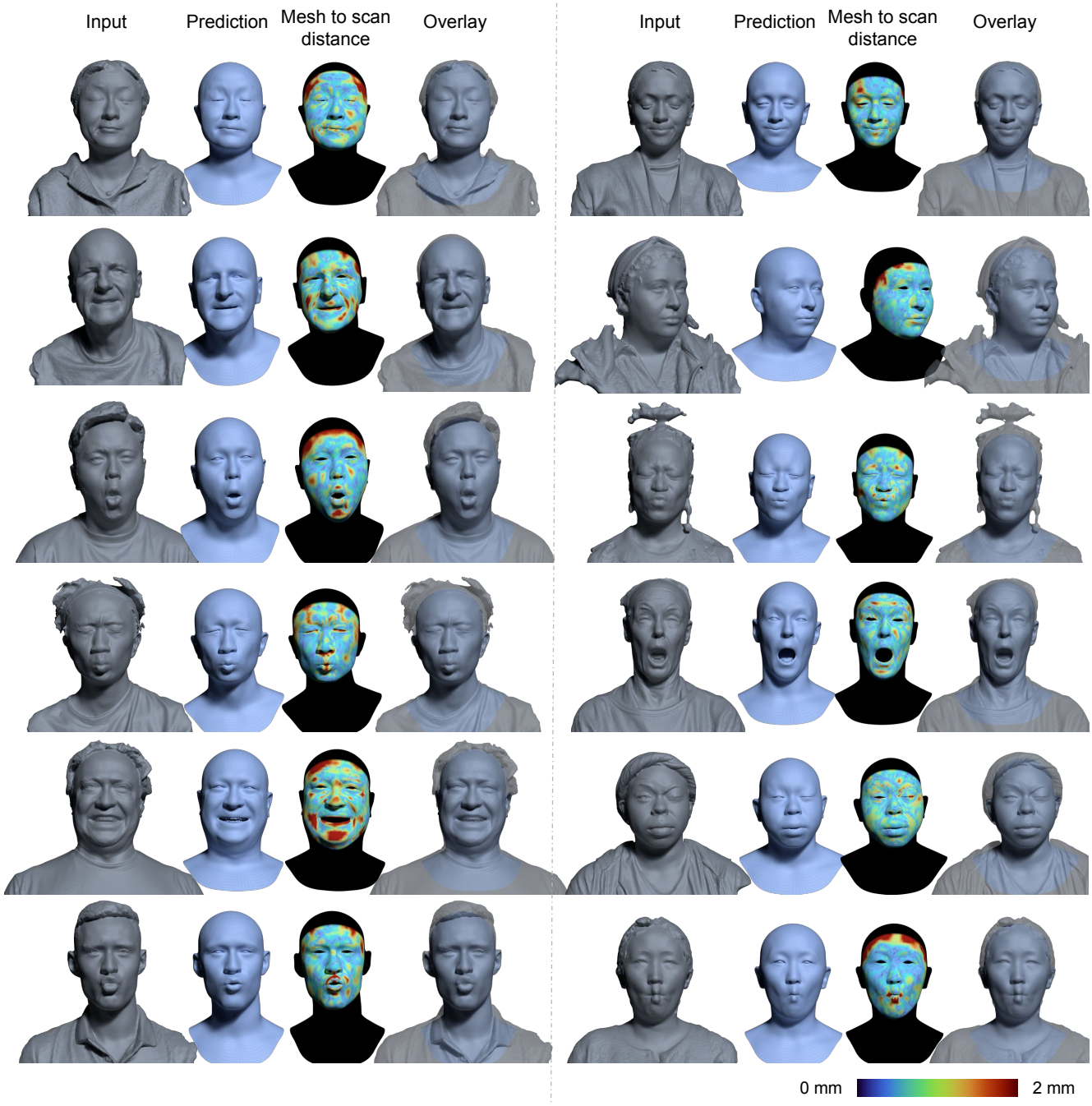

Figure S8. Additional qualitative results from our CUBE-Large $16 \times 16 \times 16$ model for scan registration. For each input scan, we visualize the prediction from the model and the point to scan distance (mm). Our method can handle input scans in varying topologies acquired from a diverse collection of subjects in various expressions and accessories.

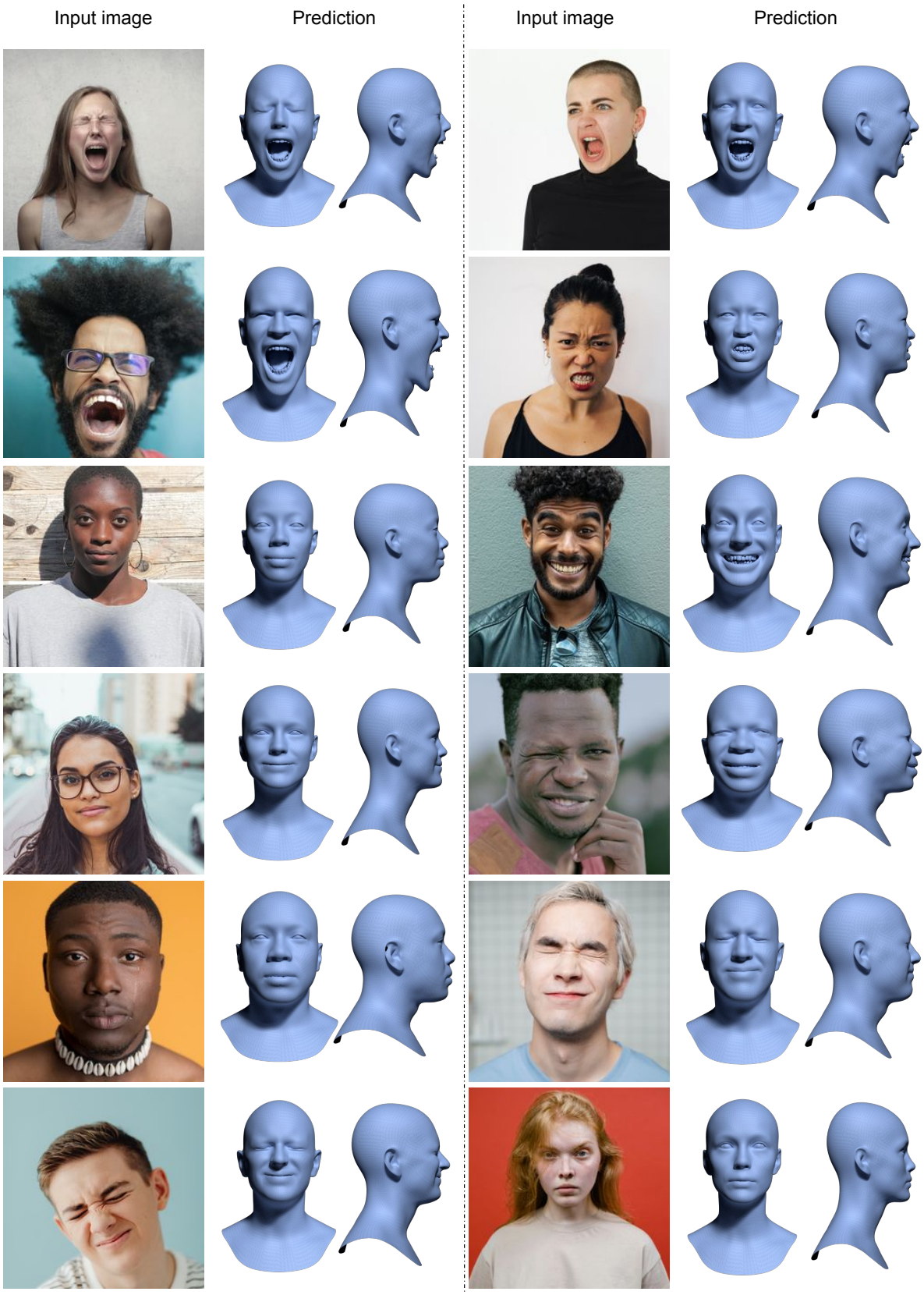

Figure S9. Additional qualitative results from our CUBE-Large $8 \times 8 \times 8$ model for face reconstruction from monocular images. For each input image, we visualize the predicted from the front and the side.