EUROGRAPHICS 2024 / A. Bermano and E. Kalogerakis
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 43 (2024), Number 2

ShellNeRF: Learning a Controllable High-resolution Model of the Eye and Periocular Region

G. Li[1,2], K. Sarkar[1], A. Meka[1], M. Buehler[1,2], F. Mueller[1], P. Gotardo[1], O. Hilliges[2], T. Beeler[1],

[1]Google [2]ETH Zürich

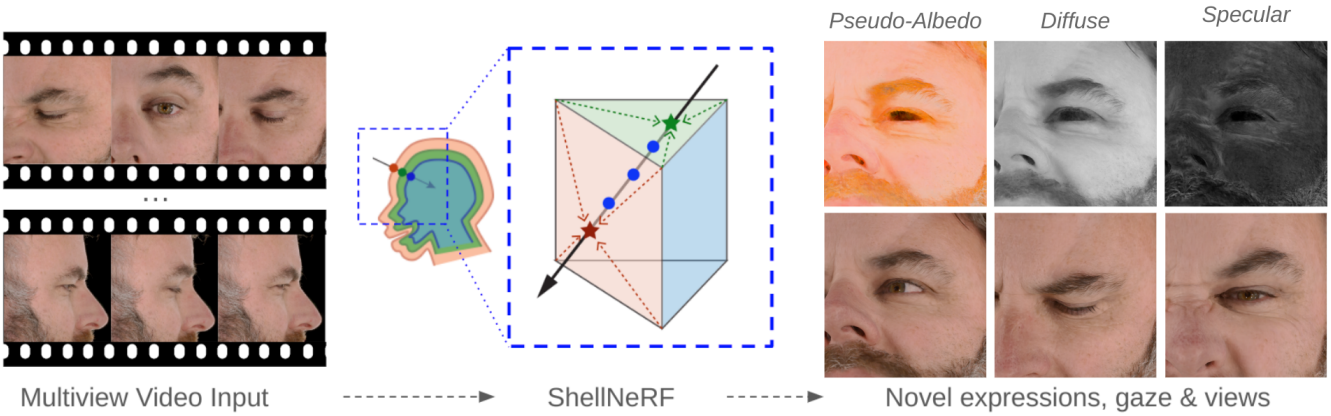


Figure 1: *We present ShellNeRF - a novel method for high-resolution novel view synthesis and animation of the periocular face region. Our method allows for controlling expressions and eye gaze and renders novel views at an unprecedented level of detail.*

Abstract

Eye gaze and expressions are crucial non-verbal signals in face-to-face communication. Visual effects and telepresence demand significant improvements in personalized tracking, animation, and synthesis of the eye region to achieve true immersion. Morphable face models, in combination with coordinate-based neural volumetric representations, show promise in solving the difficult problem of reconstructing intricate geometry (eyelashes) and synthesizing photorealistic appearance variations (wrinkles and specularities) of eye performances. We propose a novel hybrid representation - ShellNeRF - that builds a discretized volume around a 3DMM face mesh using concentric surfaces to model the deformable 'periocular' region. We define a canonical space using the UV layout of the shells that constrains the space of dense correspondence search. Combined with an explicit eyeball mesh for modeling corneal light-transport, our model allows for animatable photorealistic 3D synthesis of the whole eye region. Using multi-view video input, we demonstrate significant improvements over state-of-the-art in expression re-enactment and transfer for high-resolution close-up views of the eye region.

CCS Concepts

• ***Computing methodologies*** → ***Motion capture; Physical simulation;*** *Image-based rendering; Mixed / augmented reality;* ***Volumetric models;*** *Parametric curve and surface models; Appearance and texture representations; Shape representations; 3D imaging; Reconstruction;*

1. Introduction

"...the impression of life was gleaned from the eyes more than from other facial features" [LW10] – eyes play a crucial role in interpersonal communication, more so than any other form of body-language [CH19]. Such non-verbal signals extend beyond just the eye gaze, involuntary eye expressions ranging from blinking to gaping also play distinct roles [Kre15, LA17, DJJ22]. But the interpretation of such facial expressions varies significantly across individuals and cultures [EP13, JGY*12]. Facial performance capture systems for media or telepresence can achieve authentic immersive

communication only if the underlying face models are *personalized* and faithfully reconstruct and render *gaze* and *eye expressions* of the subject.

The challenge of building adequate models stems from the complex physics of the eye region – thin geometry of eyelashes, intricate deformation of eyelid and surrounding wrinkles, complex light transport of eyeball surface and subsurface scattering of skin. This is further exacerbated by the wide range of motions – eyeballs exhibit rigid movements like fixation, saccades and smooth-pursuit, while the 'periocular' region (face region around the eyeball) show several voluntary and autonomic expressions like winking, blinking, squinting, drooping and widening. This requires solving 3 crucial sub-problems for the eye region – i) 3D reconstruction & tracking ii) driveable animation iii) photorealistic 3D view-synthesis.

Traditional graphics-based techniques focus on modeling subregions such as eyeballs [BBGB19, BBGB16, BBN*14], eyelids [BBK*15] or the surrounding skin [KADM22]. Parametric face and eye models have been particularly successful at geometric tracking and animation of the eyeball pose and eyelid contours [WBM*16,WXLY17]. These methods excel in tracking and animation, but suffer from the limitation of the underlying mesh model – they cannot reconstruct the thin geometry of eyelashes, often cannot drive eye expression meaningfully and achieve only a synthetic rendering quality.

Recent developments in neural volume rendering for dynamic faces have enabled reconstruction of volumetric effects (including hair and eyelashes) and achieve perceptual photorealism [PSB*21, PSH*21]. They jointly optimize a deformation network that performs 3D correspondence search and an implicit canonical volume with localized surface boundaries using a coarse-to-fine importance point sampling strategy, which is inherently ambiguous. They rely on dense data from thousands of frames from slow-moving videos with slowly-changing expressions to successfully constrain this optimization. [LMM*22] combine such a dynamic implicit volume with an explicit eyeball mesh that models ray-surface interactions such as corneal reflection and refraction to achieve photorealistic view-synthesis and relighting of the eye region. While such methods work well for performance capture and may interpolate within the training frames, they do not provide any means to control or drive more general expression animations.

State-of-the-art techniques have achieved drivable animations for faces by anchoring such canonical neural volumes to a parametric face model [GTZN21,AXS*22,BTH*23,LSS*21,CSK*22, GKE*22, MESK22]. The underlying 3DMM face models, while good at pose tracking and broad expression changes, are low-dimensional and have limited capacity to model fine-scale wrinkles, hence rely on learning an additional 3D deformation field. Our experiments show that they still struggle with finer deformations required to faithfully generate eye expressions, possibly due to the inherent ambiguity of surface localization and dense non-rigid registration in 3D, particularly with sparse input data.

We propose a novel technique for tracking, animation and photoreal synthesis of gaze and eye expressions of a subject captured in a sparse multi-view camera setup. Taking inspiration from Shell Maps [PBFJ05], we develop a novel volumetric representation – *ShellNeRF* – defined by equidistantly placed concentric surfaces around a parametric face mesh. It defines a fully differentiable, one-to-many mapping from world space to a canonical UVD space – defined by the UV layout of the 3DMM and indexed 'depth' of the shell. The constrained scene volume between the first and last shell reduces the space of the correspondence search, whereas the intermediate shells allow for efficient parameterization of sample locations through barycentric interpolation. We add a residual deformation field conditioned on the local geometry gradient to solve for fine-scale errors. We combine this with an explicit eyeball mesh resulting in a holistic pipeline for animation and synthesis. To summarize:

1. We propose a novel volumetric "shell" representation anchored on a parametric face 3DMM that models personalized appearance and dynamics of the eye region.
2. We show state-of-the-art quality in animating and re-enacting gaze and eye expressions using only 3DMM expression parameters, particularly for close-up high-resolution views.
3. We experimentally evaluate our proposed model and show significant improvements over baseline SOTA methods.

2. Related Work

2.1. 3D eye modeling and synthesis

Personalized graphics model of an eyeball have been physically modeled from multi-view images [BBN*14] or even a single in-the-wild image using data-driven priors from shape and texture databases [BBGB16]. Artist designed physical parametric eyeball models have been fit to multi-view images using annotated keypoints and contours [BBGB19]. The folding motion of eyelid and resulting self-occlusion have also been temporally reconstructed [BBK*15]. 3D morphable models have been crucial to automated tracking and animation of eyeballs and periocular region [WBM*16], with special emphasis on tracking eyelid contours [WXLY17]. The periocular skin region exhibits complex deformations; data-driven techniques from multi-view captures have successfully achieved accurate reconstruction using either mesh [KADM22] or volumetric representations [LMM*22], with limited capacity to animate by interpolating between captured training gazes/expressions. [SWW*20, CSK*22] generate plausible gaze and periocular animations by conditioning their underlying face mesh and implicit volume respectively on the gaze direction for egocentric cameras in VR applications.

2.2. Volumetric face tracking & reconstruction

Many techniques have been proposed for face tracking and synthesis in the literature [ZTB*18, KRP*15], we limit the discussion to neural volumetric given their unparalleled photorealism. [PSB*21] modeled face expressions using a learnt warp field that mapped image-space observations to a canonical volume, with incredible success. This was extended to [PSH*21] expressions that involve topological changes such as mouth opening. But these methods demand dense views from a slowly moving camera to solve the dense tracking, and are not animatable.

SOTA methods use 3DMMs that provide approximate rigid and non-rigid tracking of the face geometry, and then learn a

residual 3D correspondence similar to previous methods to model higher-order effects like wrinkles and challenging regions like hair and eyelashes [HPX*22, AXS*22, GTZN21, CSK*22, BTH*23, LHR*21, XFM22].

2.3. Discrete neural volume representations

Concurrent work [GKE*22] naturally extends the 3DMM surface mesh to a tetrahedral volumetric mesh to model deformable faces. Recent techniques have also used features learnt on the nodes of a regular grid and interpolating them to arbitrary points in 3D space to define a volumetric field [CFHT23]. The complexity of training a dense neural network to model a volume has been eased by sacrificing network parameters and instead storing learnt feature vectors in a multi-resolution hash grid that is quicker to access [MESK22]. This scheme has also been utilized by concurrent work [ZBT22] to efficiently learn a personalized dynamic face model by creating a multi-resolution hash encoding around a canonical 3DMM mesh, and mapping corresponding points from the tracked mesh in observation space. One representation somewhat similar to ours is [DYXT22]. They learn a series of 2D manifolds or isosurfaces for an object category, such as faces, and sample points on the ray-surface intersections rather than in the 3D space. This representation is extended to animate faces by learning a warp field conditioned on 3DMM expression parameters [WDY*22], but they achieve only generic expression animation without personalization.

Our method is broadly derived from developments in each of these fields, but presents a novel way to constrain and control the periocular volume using concentric shells around the 3DMM mesh. We take inspiration from shell maps [PBFJ05], which defines a mapping between a world space shell, constructed using underlying surface to a 3D texture space. A volume defined in this space implicitly allows for control over the entire volume by deforming the mesh. However, this method was proposed only in the context of traditional computer graphics and is being used in the context neural deformable tracking and reconstruction for the first time.

3. Preliminaries

A Neural Radiance Field (NeRF) [MST*20] represents a scene as a radiance volume $f : (\mathbf{x}, \mathbf{d}) \rightarrow (\mathbf{c}, \sigma)$ which maps 3D locations $\mathbf{x}$ and view direction $\mathbf{d}$ to color $\mathbf{c}$ and density σ. In the original setting, this function was represented and optimized using a multi-layer perceptron (MLP), with a view-independent density. Pixel colors are rendered by integrating these values along each corresponding camera ray. Given some predetermined near and far camera plane t_n and t_f, the color of a pixel $\mathbf{C}(\mathbf{r})$ on a ray $\mathbf{r}$ can be computed using the following equation:

$$\mathbf{C}(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\sigma(\mathbf{r}(t))\mathbf{c}(\mathbf{r}(t), \mathbf{d})dt,$$

$$\text{where } T(t) = \exp\left(-\int_{t_n}^{t} \sigma(\mathbf{r}(s))ds.\right)$$

In practice, this is estimated using ray-marching with a discrete number of sample points. In the original NeRF formulation, a two pass method is utilized, by first uniformly sampling between the near and far planes, and then importance sampling based on the sample weights defined by $T(t)\sigma(\mathbf{r})$.

Nerfies [PSB*21] extends NeRFs to dynamic temporal scenes. They map all observed world space points $\mathbf{x}$ to a canonical space using a warp function $w : (\mathbf{x}, \omega) \rightarrow (\mathbf{x'})$. This warp function is conditioned on a learned per-temporal-frame latent code ω. Evaluating $f(w(\mathbf{x}, \omega), \mathbf{d})$ instead of $f(\mathbf{x}, \mathbf{d})$ yields the NeRF at the temporal frame associated with code ω.

4. Method

Our focused goal is to model the dynamic periocular face region as a deformable canonical volume with explicit control over the expression. To this end, we present a novel hybrid surface-volume representation based on a 3D mesh that is encoded and controlled by a morphable face model (3DMM). The morphable face model can be tracked across temporal frames with dynamic pose and expression changes through a traditional optimization process [BV99, GMFB*18], but it can only accumulate radiance on the mesh surface and hence cannot model volumetric effects such as hair and sub-surface scattering in the skin, limiting the photorealism. We use a *shelling* process to lift the 2D surface of the morphable model into a 3D volume with embedded appearance (radiance) field. The resulting shell volume is conveniently parameterized by extending the 2D UV parameterization of the mesh to 3D UVD by including a shell index or depth.

As in traditional facial capture pipelines, at modeling (training) time, we align and track our face model over subsequent video frames and use the registered model to photorealistically encode the appearance of each expression in the UVD space. At inference time, by adjusting gaze and expression coefficients of the model, we can explicitly control and render the periocular region for novel expressions and viewpoints.

We complete the full scene model by combining our novel model for the periocular region with an eyeball model from EyeNeRF [LMM*22]. This parametric eyeball model consists of two spheres smoothly blended into each other as defined by the cornea radius, eyeball radius, and the blending angle. More details of the eyeball model can be found in [LMM*22]. Note that their periocular model consists of a Nerfies-style deformable volume and hence cannot explicitly control expressions. For the parameteric face model, we employ a linear 3DMM similar to [BV99], using 157 expression blendshapes [Pol23]. Next, we detail the key components of our novel hybrid representation shown in the schematic overview in Fig. 2.

4.1. Shelling

We introduce a *shelling* procedure, which constructs a volumetric cage around the original surface of the morphable face model mesh by extruding the vertices by a preset amount along the vertex normal, both inwards and outwards. This step is repeated multiple times to create a number of shell layers, leading to a concentric onion-like structure. This construction allows all the shell layers to have a common surface topology. By connecting corresponding

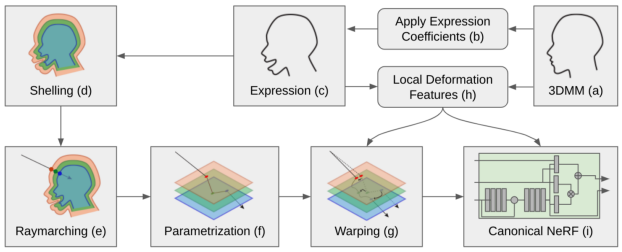


Figure 2: *Method overview: our hybrid model (a) is first posed by applying the 3DMM expression coefficients (b), Sec. 4. The resulting face shape (c) is extruded into a layered volume (d) via a "shelling" procedure, Sec. 4.1. The shells map sampled ray points from the XYZ world space (e) into a new UVD shell space (f), Sec. 4.2. The UVD points are then displaced by a warp field (g), which is conditioned on the local geometry deformation (h), Sec. 4.4. Lastly, the warped UVD points are used to query the canonical NeRF (i) for rendering, Sec. 4.6.*

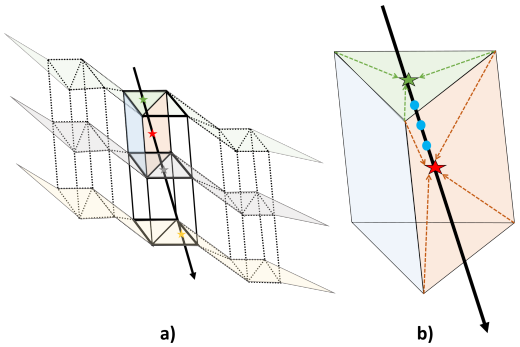


Figure 3: *(a) For each ray, we compute all intersections with all wedges, and find a successor within the same wedge. (b) The UVD coordinates of intersections on the wedge surfaces (red and green stars) are computed using barycentric interpolation. The UVD locations of samples within the wedge (blue dots) can be approximated by linearly interpolating the surface samples.*

vertices across subsequent layers, each triangle in the original mesh is lifted into a stack of *volumetric wedges*, Fig. 3. Each wedge delimits a volumetric cell within the UVD shell space. We show that using a stack of such shell layers allows for better modeling of our deformable scene by compensating for inaccuracies and limitations of morphable model tracking by accumulating volumetric features that encode i) a 3D radiance field and ii) a small additional deformation field. These encoded fields help model the complex motion, intricate geometry and appearance of hair and skin.

We observe in our experiments that more outwards shells are needed than inwards shells to model volumetric effects like hair and eyelashes. The outermost shells is extruded 12mm from the 3DMM mesh and the innermost is intruded by 1mm. The rest of the layers are equidistantly placed along this range. We show ablations with different number of shell layers (2, 6, 12 and 20), we notice no improvements for more than 20 shells and hence use it as the default setting.

4.2. Shell mapping and sampling

We now describe the mapping from the world space (XYZ) to our shell space (UVD) defined in the previous section. As in the original NeRF, each pixel is rendered by sampling 3D points along the camera ray in XYZ world space. But instead of sampling between a near and far plane, our sample points are defined by the intersections of the camera ray with the multiple wedges in our shell model. For each ray, we first compute the intersection with all triangles within the scene, including shell triangles and wedge walls in Fig. 3(a). Given the known UV parameterization and depth index D of each wedge vertex, the UVD location of each intersection point is computed through linear (barycentric) interpolation, Fig. 3(b). As barycentric interpolation is continuous across neighboring triangle boundaries, we obtain a continuous UVD mapping across wedges. For proper positional encoding, this mapping is composed with a global scaling and translation of the UV subspace from $[0,1]$ onto $[-\pi,\pi]$; the same scaling is applied on the depth axis.

4.2.1. Subsampling intervals and rendering

The ray-wedge intersections provide a sparse number of sample points. They are also an unequal count over different rays, which prohibits fixed-size batch processing, leading to wasted compute performance. We solve this problem by further densifiying the samples by adding "spare" sample points along the rays to make up a fixed count across all rays. We empirically find that a total of 380 sample points (including the wedge intersections) are sufficient for sampling across 20 shells. To place these additional sample points, we first define the *sampling intervals* as the ray segments between the intersections with each wedge. The "spare" points are then allocated across these intervals proportional to their length. The allocated points are then placed equidistantly within each interval. In some corner cases (discussed ahead) the sample points may not be sorted contiguously along the ray, hence a final sorting operation is performed to ensure the right ordering.

Once all the sample points are defined, the color of a ray $\mathbf{r}$ is rendered as:

$$\hat{C}(\mathbf{r}) = \sum_{i=0}^{N} T_i(1-\exp(-\sigma_i\delta_i))\mathbf{c}_i, \quad T_i = \exp\left(-\sum_{j=1}^{i-1}\sigma_j\delta_j\right), \quad (1)$$

where σ_i and $\mathbf{c}_i$ refer to density and radiance samples taken within UVD space and δ_i denotes the distance from a sample point to the next one.

Concave geometry of the face model mesh causes interesting corner cases during the shelling process, such as overlapping wedges and inverted faces. Overlapping wedges are caused by the "collision" of either intruded shells (in regions like the eyelids) or extruded shells (such as on the nose bridge). A naïve approach would require the definition of a one-to-many mapping, where a single XYZ world point is included into multiple overlapping wedges. By relying on ray-wedge intersections to define our sampling intervals directly in UVD space, our formulation allows for a simple and elegant way to handle the otherwise complex mapping due to overlapping wedges. We can thus sample from multiple over-

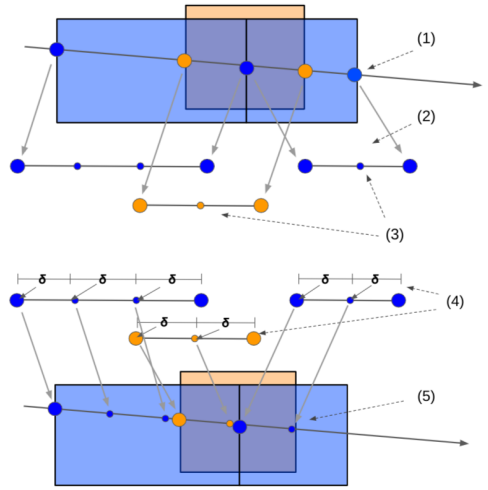


Figure 4: *There are 5 steps in the sampling of ray points within overlapping wedges (illustrated by their 2D rectangular cross-sections in blue and orange): (1) compute ray-wedge intersections; (2) pair-up intersection points into sampling intervals for each wedge; (3) allocate samples evenly across all intervals, based on their size; (4) compute the XYZ-distance δ between subsequent samples within the same interval; (5) gather the samples sorted based on their distance along the ray. The last sample in each interval is ignored.*

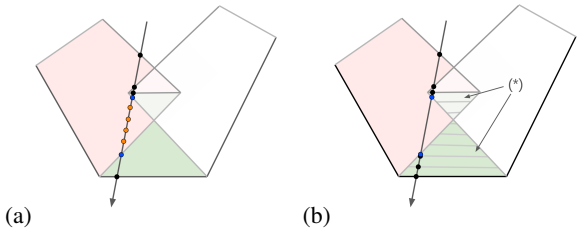


Figure 5: *Resolving an inverted wedge: the 2d cross-sections of 3 different wedges are shown in red, green and white. In (a), the two blue intersections are erroneously paired into an interval and generate orange samples outside of the wedge, leading to artifacts near the inverted wedge. In (b), new layers are added (*) and the blue samples are now in different wedge layers; they are no longer paired into an interval.*

lapping wedges without special treatment, as illustrated in Fig. 4. During rendering, as the number of sample points increases, δ_i decreases and naïvely reusing Eq. (1) leads to decreased opacity in each of the overlapping wedges, as density is split up between the volumes (which "share" the integration distance). To avoid this issue, we instead re-define δ_i as the distance to the next point *within the same wedge*, meaning that δ_i would then remain constant regardless of how many intersection points from other wedges are in the same space. The ordering of each sample point however remains the same, leading to correct integration of radiance along the ray. Inverted mesh faces, on the other hand, require special handling. We discuss this in more detail in the next section.

4.3. Wedge Inversion

As we construct the shells by extruding vertices along their normals, at certain convex locations such as near the eye edges, the in/extruded shell surfaces may contain inverted faces, resulting in self overlapping wedges.

For the human face, the actual wedge inversions occurs almost exclusively in areas where the volume should be transparent (and therefore would not intrinsically cause issues), as the only areas where the volume needs to extend above the surface are hairs which exclusively are seated above convex sections of the face where mesh inversion cannot occur. Therefore, the primary concern lies within the fact that our sampling strategy through the use of interpolation could produce invalid samples throughout and even outside the wedge (see Fig. 5 (a)), and not just the region where the wedge self-intersects.

This is mitigated by the fact that we employ multiple layers within our shells. Although wedge inversion still occurs, each wedge would only be inverted in one layer, and incorrect inbetween samples would not be generated outside that layer, effectively isolating the problem to a much smaller region (see Fig. 5 (b)).

4.4. Warp Field

3DMMs can model the coarse motion involved in various facial expressions but do not have the resolution to model more subtle details, especially finer structures like the eyelids. While slight misalignments might not be visible to human observers, shifts of even a few pixels could cause significant degradation of quality as our model integrates data over multiple frames/expressions. In order to address these residual errors, we learn a small additional 3D warp field.

While Nerfies [PSB*21] employs such as warp field to directy learn the mapping from the observation space to an implicit canonical space, we instead first map world space 3D points to the shell space (in UVD coordinates) for each frames using the mapping previously described in Sec. 4.2. The additonal 3D warp field is used only to compute a delta from the shell UVD space of each frame to the final canonical UVD space.

While the warp field in Nerfies is conditioned on a per-frame learnt latent code, this cannot be generalized to novel unseen expressions with explicit control. A simple solution to this would be to instead condition it on the 3DMM expression paramaters for each frame. However, the expression basis of our 3DMM model contains 157 dimensions, which may result in overfitting. Furthermore, this strategy would entangle the warp field and the full face expression whereas the warp should only be a spatially local phenomenon. Hence, we choose to condition the warp field only on local geometry features (defined ahead in the next section).

$s : \mathbf{x} \rightarrow \mathbf{x'}$ is our mapping from world space to the shell UVD space, as defined in Sec. 4.2. $w : (\mathbf{x'}, \omega) \rightarrow (\mathbf{x''})$ is the residual warp field, conditioned on local geometry features $\omega(\mathbf{x})$. Each sample

point is mapped from the given world space location **x** to the canonical location **x''** as $\mathbf{x''} = w(s(\mathbf{x}), \omega(\mathbf{x}))$. Note that as mentioned in Sec. 4.2, although s is not a function as it is a one to many mapping, each sample point along a camera ray is still mapped to exactly one resulting canonical location. We use the same parameterization for our warp field w as used in Nerfies [PSB*21] – an MLP with positionally encoded input **x'**, using annealing and arap regularization, only with the difference of conditionining on local geometry features instead of a learnt latent code. We refer you to their paper for more schematic details.

4.4.1. Local Geometry Features

We condition the warp field on the local geometry, which changes with mesh deformation across different expressions. We formulate these geometry features to be both well defined and continuous across the mesh. These properties are defined on a per vertex basis and can then be computed at any point in any wedge using the same interpolation as used to compute the UVD coordinates.

Let v_c be the vertex for which we compute the geometry features, and $v_{0..N}$, be the list of neighboring vertices, and let n_c be the corresponding normals. Let $w_{0..N}$ be a corresponding list of weights, computed based on the mean of the angles of the two triangles adjacent to that vertex: $w_i = \frac{\alpha_i + \alpha_{i+1}}{2\sum \alpha_k}$ with $k \in 0..N$. This corresponds to weights commonly used for computing shading vertex normals, where neighboring triangles are weighted based on their angles.

Specifically, we compute the following values:

- The area of the faces around the vertex: $a_c = \sum ||(v_i - v_c) \times (v_{i+1} - v_c)||$
- The scaled cotangent laplacian of the vertex: $\frac{1}{\sqrt{a_c}} \sum w_i (v_c - v_i) \cdot n_c$
- The rotational stretch, estimated using the shifting weights: $\sum w_i (\mathrm{UV}_i - \mathrm{UV}_c)$
- The normal of the vertex, in the canonical head pose: n_c
- The location of the vertex, in the canonical head pose: v_c

The first value roughly corresponds to the local horizontal stretch, defining how much the triangles have increased or decreased in size. The second value estimates the local curvature, using the cotangent laplacian as a heuristic. The third value measures the relative change of the weights w_i. As the UV location of each vertex is constant across all frames, the value changes only if the corresponding angle weights change. Specifically, if the nearby vertices are all rotated towards a certain UV direction, then the weights increase for vertices closer to the opposite direction, shifting the resulting weighted UV difference in that direction. We normalize the values by subtracting the values computed from the neutral expression, and scale the result to have a standard deviation of 1, and assemble them into a 10-dimensional local geometry feature descriptor to condition our warp field. To avoid outliers, we clip the values to be strictly with [-2, 2].

We store the range of values of the features for each vertex and each feature dimension observed during training. During rendering of novel expressions, we constrain the evaluated features to lie within those ranges. This prevents our warp field and the NeRF features from extrapolating to unstable values, instead relying purely on the 3DMM motion for poses which are outside of the convex hull of the training set.

4.5. Explicit Eye Model Integration

Although our shell formulation allows for control over most of the face, the 3DMM face model by itself does not contain the eyeball and hence is unsuitable for modeling eye motion. We therefore add an explicit parametric eyeball model similar to EyeNeRF [LMM*22]. The volume inside the eyeball is modeled separately than the canonical UVD volume. During the rendering process, we compute ray intersections with the eyeball surface and split it into a reflected and refracted ray. The refracted ray that enters into the eyeball through the cornea is marched through an "inner-eye" volume defined in the world XYZ space. The reflected ray is marched through the UVD shell space. Instead of using a separate MLP to model the "inner-eye" volume, it is parameterized with the same MLP as the shell UVD volume. The range of XYZ values for the "inner-eye" volume and the UVD values for the shell space are scaled and an offset is added to ensure that they do not overlap. This avoids the computational cost of an additional network without any noticeable performance drop.

Splitting the ray at the exact location of the eyeball surface can lead to artifacts since the eyeball model parameters might not be accurate and the eyeball surface might intersect with the eyelid, especially for unseen poses. As a solution, we extend the ray by 2 mm into the eyeball after the intersection. This ensures that the entire eyelid is always integrated.

We also model and estimate the scene illumination using an environment map model parameterized by a lat-long image. This image is queried at the end of the reflected ray and is optimized to generate the specularities/glints on the eyeball surface. Unlike EyeNeRF which required the ground-truth environment map, we estimate this scene lighting, which is later used for generating accurate specularities on the eyeball while synthesizing novel views and gaze.

4.6. Shading

The combination of shell map and warp field is very flexible and handles most facial deformations. However, it is incapable of correctly modeling shading caused by wrinkles that appear and shift as the face deforms. Similarly, as the face moves, the normals and specular highlights also shift. We address this issue by decomposing the output radiance of our network into view-independent albedo and diffuse shading and view-dependent specular shading.

Features from the MLP trunk (Fig. 6, green) are branched off to predict three outputs: a 3 dimensional albedo color, a single dimensional diffuse shading factor and a single dimensional specular shading. The diffuse shading factor is first multiplied with the albedo, and then the specular shading is added to obtain the final radiance. Note that we do not claim a physically accurate intrinsic decomposition of the radiance field, our primary objective of performing such a decomposition is to condition the diffuse and specular shading components with the local geometry features, so that they can model the high-frequency shading effects caused by wrinkles and folds. Our ablation studies clearly show the effectiveness of this disentangled representation in achieving more accurate and photorealistic synthesis.

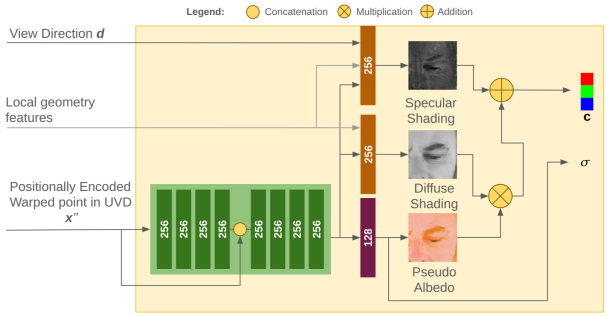


Figure 6: *Architecture of the proposed canonical NeRF network, which is an MLP that takes the positionally encoded canonical UVD-space point* $\boldsymbol{x''}$*, local geometry features, and the view direction as inputs. The model outputs a density* σ*, a diffuse shading factor that is multiplied with the pseudo albedo and finally added to a specular shading to yield the final color* $\mathbf{c}$*.*

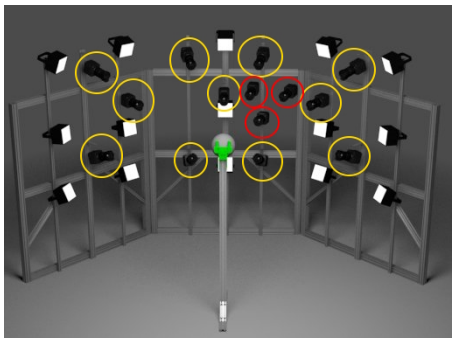


Figure 7: *The capture setup includes 11 training cameras (yellow circles) and 3 test cameras (red circles). The subject's head is positioned at the central gray sphere.*

5. Model Training

In this section we describe how we train our model, the losses and regularization terms, and the training data.

5.1. Data Capture

To demonstrate our method, we capture 5 subjects in a calibrated multi-view camera rig consisting of 14 cameras distributed quasi-uniformly in the frontal hemisphere as shown in Figure 7. The capture sequence consists of a series of momentary videos of the subjects making 30 different expressions. Of the 14 camera views, 11 are used for training, while 3 of the frontal cameras are used for evaluation. We use only 1-3 frames of each captured expression for training our model.

We also capture each subject looking at the 10 frontal cameras while making 4 different expressions (neutral, brows up, smiling, squinting) while keeping their head pose still. Since the camera positions are known through the camera calibration process, capturing this "gaze-target" sequence provides frames with ground-truth 3D gaze direction, which helps to achieve explicit control of the synthesized gaze direction during rendering. The frames where the subject is looking at the evaluation cameras are also held out in order to evaluate regazing quality. Note that the subject can only look at 10 cameras and not all 14 cameras because 4 cameras are located beyond the peripheral vision of a frontal gaze, so the subject cannot look at them without turning their head.

Finally, we eliminate any frames with noticeable motion blur through a quick manual inspection. This results in 120 - 150 total training frames per subject. Each image initially has a resolution of 4096×6144, containing the entire face. We focus on the eye and periocular region and crop the image to a 1500×1500 pixel region around the left eye. As the shell-volume is constrained to the close proximity of the face, our method is incapable of reconstructing the background. We segment out the background in the training frame using an off-the-shelf face segmentation method [PEL*21].

5.2. Network Training

We use mean-squared error in sRGB space as the primary training loss. For a given ray $\mathbf{r}$, we compute the mean-squared error between the color of the target pixel $\mathbf{C}_{\text{srgb}}$ and the output of our network $\hat{\mathbf{C}}_{\text{srgb}}(\mathbf{r})$.

$$l_{\text{im}} = ||\mathbf{C}_{\text{srgb}} - \hat{\mathbf{C}}_{\text{srgb}}(\mathbf{r})||_2^2 . \tag{2}$$

5.3. Geometry Training

Both the 3DMM shell and warp field serve the same purpose and map correspondences across frames. They are inherently interchangeable, and therefore ambiguous. We resolve this ambiguity by staggering our training into two phases. i) In the first phase, we train our model without the warp field. Our 3DMM parameters are initialized using a traditional model fitting process consisting of minimization of sparse facial landmark reprojection error across multi-view images. These 3DMM parameters are further jointly optimized while training the canonical NeRF network with our primary photometric loss, Eq. 2. This achieves 3DMM parameters that best represent the coarse to mid-level deformation. We also reset the NeRF volume 4 times (every 50k iterations) to improve the final geometry accuracy, in particular that of the eyeball model. Inspired by [LMM*22], we also learn a per-vertex offset for the eyeball model δ_i, which we regularize using a small l_2 loss $l_{\text{off}} = \sum_{i=0}^{N_v} \delta_i^2$. To prevent the drift of the 3DMM geometry during the joint optimization, we compute the average drift as the difference of the initial and optimized 3DMM parameters and apply them as bias for re-animation. ii) In the second phase, we freeze the 3DMM parameters and continue training the canonical NeRF network, this time also learning the warp field. This addresses the residual deformations that require the more flexible local warp field. During this phase we train for 300k iterations.

5.4. Environment Map

We learn an approximate environment map to construct high quality corneal reflections. We define the environment map as a 256×128 latlong image I_{env}, which is sampled bilinearly for each reflection query. The environment map is trained together with the other

networks using l_{im} (Eq. 2). We further impose a small l_1 regularization on the environment map in order to encourage sparsity and mitigate noise: $l_{\text{env}} = ||I_{\text{env}}||_1$

5.5. Eyelid Opacity Regularization

Eyelid and eyeball motion are strongly correlated to gaze, i.e. the upper eyelid is automatically being lowered when looking down. This introduces an ambiguity. The appearance of the eyelid can be on the eyeball or the eyelid. This is not an issue for reconstructing training frames but artifacts may occur when rendering novel expressions, especially for eyeball motion or blinking. In order to resolve this ambiguity, we leverage the fact that certain training frames include entirely closed eyelids. For these frames, the appearance should remain the same for every eyeball pose or color. Instead of querying the volume inside the eyeball for the refracted ray, we replace the eyeball volume with randomly chosen colors during training. This encourages the rest of the volume outside the eyeball to become more opaque and hide the incorrect eyeball color.

5.6. Additional Regularizers and Total Loss

In order to improve overall quality and reduce floaters, we use the distortion loss from MipNeRF360 [BMV*22], which penalizes non sparse volumes. In order to achieve spatial consistency, the warp field is regularized using the ARAP loss used in Nerfies [PSB*21]. We refer the reader to the respective papers for more details.

This results in the total loss:

$$l_{\text{tot}} = \lambda_{\text{im}} l_{\text{im}} + \lambda_{\text{off}} l_{\text{off}} + \lambda_{\text{env}} l_{\text{env}} + \lambda_{\text{distort}} l_{\text{distort}} + \lambda_{\text{arap}} l_{\text{arap}},$$

with the empirically chosen weights $\lambda_{\text{im}} = 1$, $\lambda_{\text{off}} = $ 1e-6, $\lambda_{\text{env}} = $ 1e-7, $\lambda_{\text{distort}} = 1$, $\lambda_{\text{arap}} = 0.1$. We train our models using the Adam [KB14] optimizer with an initial learning rate of 5-e4, exponentially decayed by a factor of 0.1 every 250k iterations.

5.7. Split training

Training our model with multi-frame data with several expressions inherently leads to blurred results, when compared with training a NeRF on a single frame. This is consistent with most other prior works such as Nerfies [PSB*21], MonoAvatar [BTH*23] or NerFACE [GTZN21]. This is however not a binary effect; as can be seen in Fig. 8, the more frames that are trained on, the blurrier the end result becomes.

We speculate that this is due to the warp field not being able to accurately model complex sub-pixel deformations experience by the skin, and that this mismatch results in blur as the network attempts to aggregate the data across frames. Furthermore, when training on fewer frames, finding correspondences also becomes easier and likely improves the warp field quality. But when training on only a few frames, we lose generalizability leading to very noticeable artifacts when rendering novel expressions.

In order to retain details without losing generalizability, we distribute training samples such that ⅔ of them are distributed across 6 key expressions (tightly closed eyes, closed eyes, neutral, looking left and right, snarl), while the rest are distributed evenly across all expressions. The latter ensures that the warp field does not deviate too much from any given expression, while the former training samples allow the unwarped canonical volume to be trained primarily on only a few frames, resulting in a less blurry volume overall.

Figure 8: *Results of the same frame rendered after training the network on an increasing number of frames. The first 2 rows show the ground truth training expression followed by synthesized result with different number of frames used in training. The last row show the same for another training expression. The more frames used for training, the blurrier the result. However, training with fewer than 144 frames results in artifacts when rendering novel expressions.*

6. Results and Evaluation

In this section, we demonstrate the ability to independently control the gaze (Fig. 9), viewpoint (Fig. 10), and eye expressions of a captured subject (Fig. 11, 12, and 13). We train our model on 5 different subjects with diverse appearances, and evaluate it on multiple different settings. As described in Sec. 5.1, we capture 4 sequences of the subject looking at each camera with various expressions. To enable evaluations, we hold out parts of the data during training as detailed below. In addition, we capture two test sequences, each consisting of the subject making a variety of expressions for 10 seconds. The second sequence also contains head motion, with the subject rotating their head around.

6.1. State-of-the-Art Comparisons

We compare our method with several state-of-the-art methods – NerFACE [GTZN21], Mixture of Volumetric Primitives (MVP)

Table 1: *Results from our quantitative baseline comparison. The colors indicate best, second best, and third best.*

	Regazing		Test View		Test Expression		Gaze+Expression	
	SSIM ↑	LPIPS ↓	SSIM ↑	LPIPS ↓	SSIM ↑	LPIPS ↓	SSIM ↑	LPIPS ↓
MonoAvatar	0.7633	0.4673	0.6940	0.5110	0.7511	0.4665	0.7558	0.4691
Nerface+Eye	0.70854	0.4989	0.6778	0.4963	0.7006	0.5093	0.7012	0.5018
EyeNeRF+	0.6911	0.4871	0.6930	0.4831	0.6851	0.4969	0.6912	0.4865
Nerface+EyeNeRF	0.6932	0.4868	0.6953	0.4811	0.6840	0.4896	0.6933	0.4855
MVP	0.7816	0.3705	0.7179	0.4324	0.7381	0.4090	0.7509	0.3865
ShellNeRF	0.7516	0.3924	0.7124	0.4358	0.7206	0.4198	0.7390	0.3941

Table 2: *Results from our user study. We show the preference of users for each method and task.*

	Regazing Realism	Regazing Accuracy	Reenactment Accuracy	Expression Realism	Novel View Realism
MonoAvatar	4.9 %	9.3 %	2.2 %	4.9 %	0.0 %
MVP	1.3 %	8.0 %	15.6 %	12.0 %	11.1 %
ShellNeRF	93.8 %	82.7 %	82.2 %	83.1 %	88.9 %

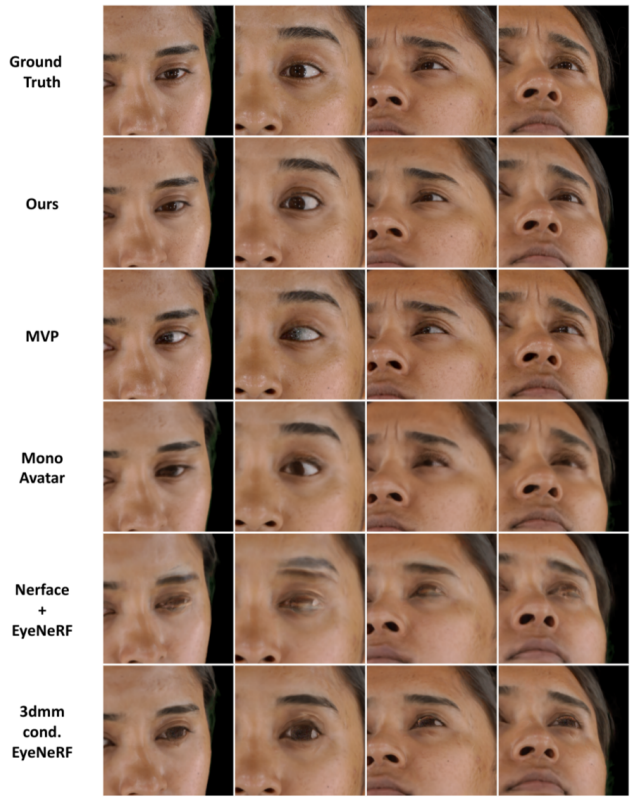


Figure 9: *Comparisons for regazing. Related works produce ghostly artifacts on the pupil. Even the best performing related work, MonoAvatar, struggles with the highly nonlinear deformations on the upper eyelid while ours consistently generates high-quality renderings.*

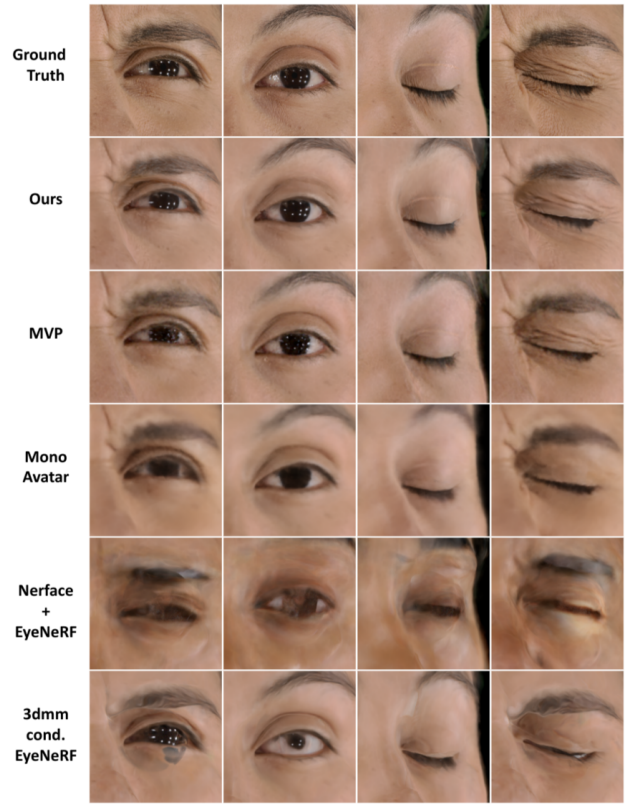


Figure 10: *Comparisons for novel view synthesis. Related works fail to synthesize details like specular reflections on the eye surface. Our approach maintains high fidelity and renders reflections of individual lights on the eyeball.*

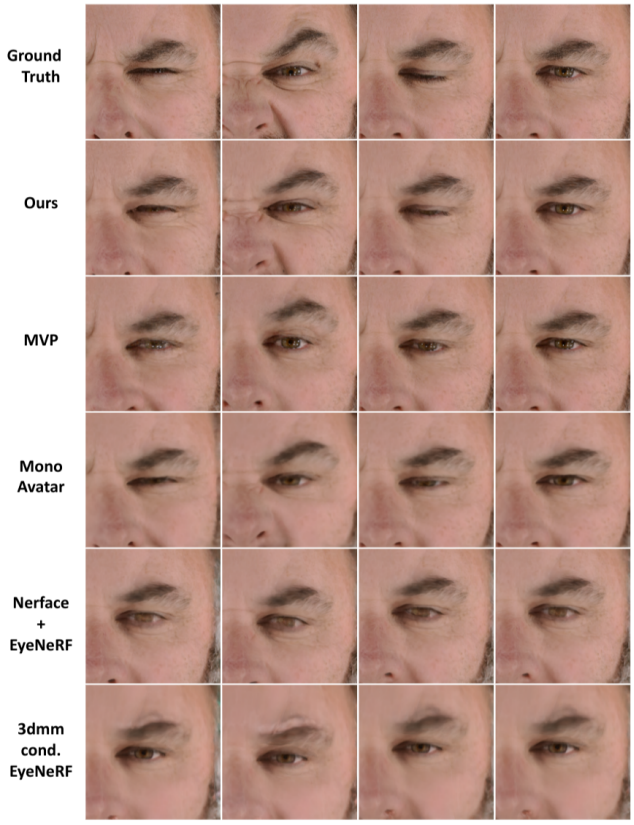


Figure 11: *Reenactment is a highly challenging task because the model has to synthesize expressions that were not seen during training. Note how the baselines syntheize images where the underlying expression seems perceivably different than the ground-truth. Our hybrid approach is more robust to out-of-distribution expressions and renders convincing novel expressions.*

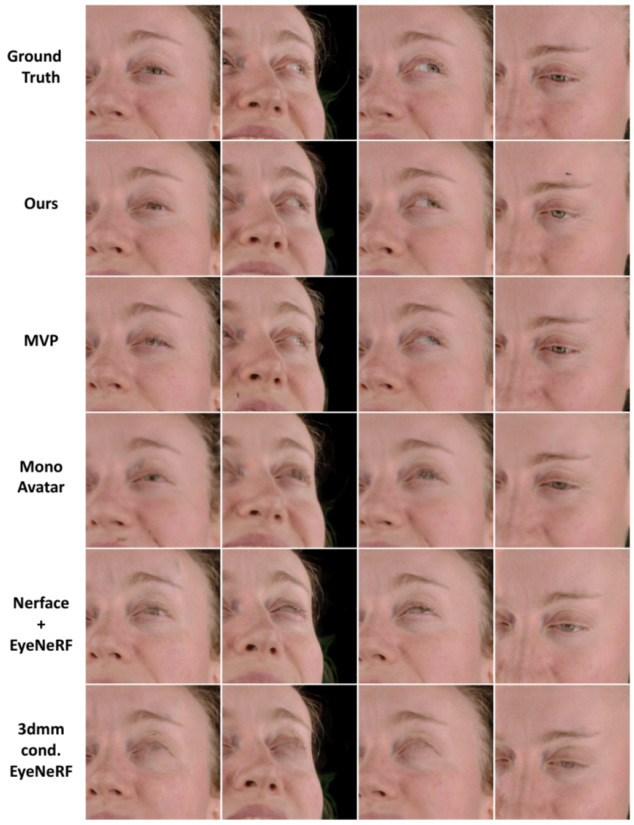


Figure 12: *Gazed reenactment. Our proposed approach disentangles eye gaze from expressions, which allows rendering the same expression with different gaze directions.*

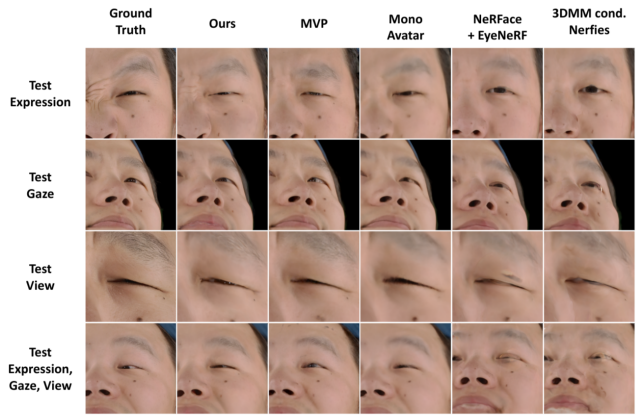


Figure 13: *We demonstrate the ability of our method to independently control and synthesize test gaze, expression and viewpoint from our captured dataset that were heldout during training. We outperform the SotA methods in all dimensions.*

[LSS*21], and MonoAvatar [BTH*23]. Furthermore, although EyeNeRF [LMM*22] is not directly animatable, we train a modified version of it in which the latent code is replaced with the 3DMM parameters, with an additional latent slack, similar to the conditioning used for the dynamic neural radiance field in NerFACE. Therefore, we use our own implementation of NerFACE and EyeNeRF, and additionally compare our method to a NerFACE conditioned warp field, as well as a full combination of NerFACE and Nerfies, using both a conditional warp and radiance field. Given their similarity, we further include the EyeNeRF eye model in the NerFACE evaluation for a more fair evaluation. To ensure a fair comparison, we also use our 3DMM model in the implementation of all SotA methods (where applicable). We note that MonoAvatar and MVP compare well with our method over many image quality metrics (see Table 1). While quantitatively our method does not obviously stand out, we encourage the reader to see the supplementary video results for the full qualitative evaluation. Our method achieves more accurate animation and view-synthesis, whereas all other methods struggle with gaze and expression changes and produce significant floaters for novel camera viewpoints. This seems

to indicate that image quality metrics do not capture the effects of spatio-temporal volumetric artifacts and may not be best suited for evaluation of such methods.

6.2. Evaluation settings

6.2.1. Novel View Synthesis

We demonstrate view consistency by showing novel view synthesis (see Fig. 10). During training, we hold out the 3 frontal clustered cameras, and use them as an evaluation set. Although the specular highlights shift and are slightly incorrect, we maintain the overall appearance, and maintain proper geometry. We are also able to synthesize approximate eyelashes from novel views, while SotA methods fail to generate the right wrinkles and frown lines.

6.2.2. Regazing

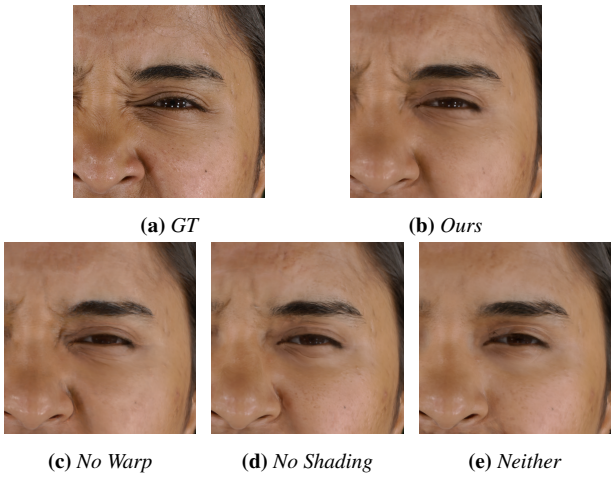
To evaluate regazing capabilities, during training, we hold out all images of the subject looking at one of the cameras as a test gaze. Similar to EyeNeRF [LMM*22], we can perform regazing by directly manipulating the eyeball pose. We demonstrate the ability to control the eyeball and achieve the desired gaze given the 3D gaze direction (computed from the known gaze target). While EyeNeRF can control the periocular region only by interpolating known gaze frames we achieve greater expression control by applying 3DMM parameters manually or from a known fit. In Fig. 9, we demonstrate generalization to new gaze targets using the corresponding 3DMM fits. Note that our method correctly disentangles periocular expression and eyeball pose and hence can synthesize an unseen gaze direction, whereas MonoAvatar [BTH*23] struggles to disentangle the two and tries to fall back to the nearest training expressions. NerFACE [GTZN21] and Nerfies [PSB*21] fail to synthesize the unseen eyeball poses.

6.2.3. Reenactment

We demonstrate the ability to reenact previously unseen expressions using held out test sequences consisting of the subject making various expressions for 10 seconds. Although the face pose may be slightly mismatched due to minor inaccuracies in 3DMM fits for the test sequence, we are still fully capable of reenacting a variety of expressions (see Fig. 11) and generating perceptually accurate wrinkles and eyelid positions. We also hold out the sequence of the subject looking at each camera while smiling during training. Using this data, we verify that our method can perform 'gazed-reenactment' – correctly model unseen expressions with unseen gaze directions as shown in Fig. 12.

6.3. User study

As both MonoAvatar and MVP compare well with our method over many image quality metrics, we further conducted a user study to evaluate these three methods. We asked 45 participants to choose which method performs best in terms of five tasks: regazing realism, regazing accuracy, reenactment accuracy, expression realism and novel view realism. As shown in Table 2, the vast majority of participants prefer our method over MonoAvatar and MVP across all categories.



(a) *GT* **(b)** *Ours* **(c)** *No Warp* **(d)** *No Shading* **(e)** *Neither*

Figure 14: *Ablation of the warp field. We train a version of the method without the warp field or shading decomposition conditioned on local geometry features described in Sec. 4.4 and Sec 4.6. We show comparisons on strongly deformed held out expressions. Note that only with both warp field and shading decomposition are all features preserved.*

To evaluate realism of regazing, expression and novel view synthesis, we showed each participant the results of the three methods side-by-side. Each result is a short video segment that transitions and interpolates through various (training) gazes, expressions and camera angles. We asked each participant to choose which of the three results is the most realistic. Please refer to the supplementary materials for examples, labelled as "Regazing", "Interpolating Expressions" and "Novel View Synthesis", respectively.

For regazing and reenactment accuracy, we showed each user a target novel gaze image or novel expression video as well as corresponding reconstructions of each subject, using each method. We then asked them to choose which method most accurately reconstructs that gaze direction. Examples are illustrated in Fig. 9 and in the supplementary materials.

6.4. Ablations

We perform ablations on our various design choices and settings to demonstrate their contribution to the method, see Table 3.

6.4.1. Warp Field and Shading

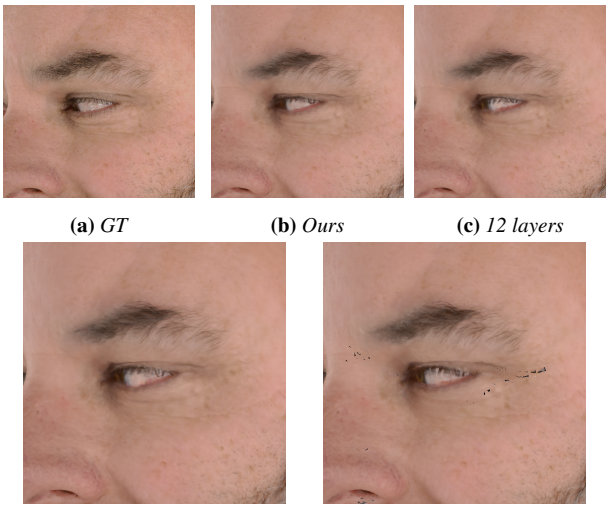
As warp field and shading are inherently ambiguous and are capable of partially compensating for each other, we evaluate them jointly. As described in Sec. 4.4, we model residual deformations like finer wrinkles using a warp field conditioned on local geometry features of the 3DMM mesh. We train a version of our method for a given subject *without* this warp field, see Fig. 14. This results in blurrier images. Without the shading decomposition, we are still able to partially reproduce shadows through use of the warp field, but this results in significant artifacts as the network attempts to use the warp field to change shading. When the warp field is also

Table 3: *Results from our quantitative ablation.*

	Regazing		Test View		Test Expression		Gaze+Expression	
	SSIM ↑	LPIPS ↓	SSIM ↑	LPIPS ↓	SSIM ↑	LPIPS ↓	SSIM ↑	LPIPS ↓
No Warp	0.7561	0.4140	0.6997	0.4610	0.7304	0.4361	0.7450	0.4148
No Shading	0.7516	0.4098	0.6919	0.4678	0.7286	0.4351	0.7406	0.4132
No Warp or Shading	0.7547	0.4192	0.5391	0.5262	0.7328	0.4419	0.7472	0.4223
No Eyelid Regularization	0.7519	0.3939	0.7081	0.4386	0.7226	0.4212	0.7401	0.3961
No Split Training	0.6731	0.4803	0.7248	0.4347	0.7368	0.4362	0.7475	0.4185
No Geometry Optimization	0.6896	0.4594	0.6803	0.4638	0.7227	0.4437	0.7330	0.4256
2 Shell Layers	0.7477	0.4224	0.6893	0.4677	0.7223	0.4432	0.7361	0.4245
6 Shell Layers	0.7129	0.4358	0.6539	0.4767	0.6962	0.4510	0.6998	0.4418
12 Shell Layers	0.7523	0.3980	0.7075	0.4417	0.7230	0.4243	0.7395	0.3999
20mm Outer Shell	0.7532	0.4030	0.7040	0.4508	0.7249	0.4287	0.7423	0.4049
6mm Outer Shell	0.7481	0.3863	0.7114	0.4306	0.7216	0.4135	0.7352	0.3886
3mm Inner Shell	0.7495	0.4087	0.6974	0.4581	0.7249	0.4319	0.7370	0.4115
ShellNeRF	0.7516	0.3924	0.7124	0.4358	0.7206	0.4198	0.7390	0.3941

(a) *GT* **(b)** *No Regularizer* **(c)** *Ours*

Figure 15: *Without the eyelid regularizer which ensure solidity of the eyelid and nearby region, the network is allowed to produce partially transparent skin which become visible at novel views, revealing the EyeNeRF eye model underneath.*



(a) *GT* **(b)** *Ours* **(c)** *12 layers*

(d) *6 layers* **(e)** *2 layers*

Figure 16: *As described in Section 4.3, an insufficient number of layers leads to visible artifacts, as can be seen in the images showing our method trained using 2 and 6 layers.*

removed, the network is unable to adapt, it is unable to model wrinkles and the overall quality suffers dramatically.

6.4.2. Eyelid Regularization

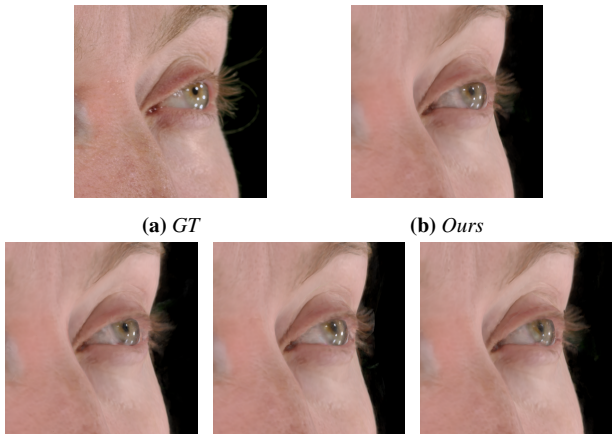
Without the eyelid regularizer, certain areas of the face remain transparent, allowing the eyeball model boundary to be visible. Our eyelid regularizer prevents this from happening by encouraging the network to make the eyeball completely invisible when the eyelid is closed. See Fig. 15

6.4.3. Shell Layering

For our final model, we use a total of 20 shell layers, resulting in 19 stacks of wedges. We also test our model with 2, 6, and 12 shell layers. As can be seen in Fig. 16, both with 2 and 6 layers, artifacts are clearly noticeable. Between 12 and 20 layers, we notice barely noticeable improvement. Since the drop in compute efficiency is not significant, we have opted for 20 layers as the default for generating our other results.

6.4.4. Shell Extrusion

In our default model, the outermost shell is extruded 12 mm outwards, and innermost 1 mm inwards. We perform an ablation by changing this range to 6 mm outwards and 20 mm outwards, as well as by 3 mm inwards. As can be see in Fig. 17, extruding the model further outwards effects slightly blurrier results, while 6 mm appears to just barely clip the eyelashes of certain subjects. We therefore chose 12mm as a conservative threshold. Extruding the shell by 3mm inwards on the other hand results in significant artifacts on certain expressions. This is likely due to the fact that the eyelids, which are very thin structures, would have their inner shells extended past the other side of the lid, resulting in strong inversions in a significant part of the face. As mentioned in Sec. 4.3, although layering *localizes* the effects of inversions which otherwise primarily occur above the face and outside of solid geometry, it does not solve it near the actual inversion location, which in this case is on top of or within the solid eyelid.



(a) *GT* **(b)** *Ours* **(c)** *20mm outer* **(d)** *6mm outer* **(e)** *3mm inner*

Figure 17: *We show an ablation with extruding our shell inwards and outwards at different distances. When extruding only 6mm, certain subjects' lashes extend beyond the shell and are clipped. 20mm makes the image marginally blurrier for no further benefits. When extruding 3mm inwards, artifacts appear near the eyelids.*

(a) *GT* **(b)** *No Geom. Optim.* **(c)** *Ours*

Figure 18: *We show an ablation showing the necessity of differentiable geometry. When attempting to render a novel expression, using the initial 3DMM fit results in significant artifacts near the eye.*

6.4.5. No Geometry Optimization

We also ablate our method by dropping the joint optimization of the 3DMM model parameters during the first phase of training, see Fig. 18. We note that this joint optimization is crucial to capture smaller geometry details that are usually not captured by traditional 3DMM fitting such as position of eyelids.

7. Discussion & Limitations

While our method achieves state-of-the-art quality in synthesis of the periocular face region, we note that the joint optimization of the 3DMM parameters during training may theoretically introduce biases due to the inherent shape-radiance ambiguity. Such biases may be exaggerated due to accumulation of features across multiple frames/expressions and could cause issues during reenactment if the optimized 3DMM geometry differs significantly from the ground-truth geometry. In practice, we do not notice any significant bias, possibly due to good initialization of the 3DMM parameters.

We also note that although the eyeball model improves the quality of the eye and allows for explicit control of the gaze, this poses an additional requirement for reenactment that the exact 3D gaze must be known and is not directly predicted, as done by some other methods [CSK*22, SWW*20].

8. Conclusion

We introduce *ShellNeRF*, a novel hybrid method that combines explicit face and eyeball models with a canonical neural volumetric representation to provide a holistic pipeline for animation and synthesis of the eye and periocular region. We demonstrate significant improvements in the quality of synthesized details over state-of-the-art approaches for various applications like expression reenactment, novel view synthesis, and regazing which are crucial for enabling the next generation human 3D telepresence.

References

[AXS*22] ATHAR S., XU Z., SUNKAVALLI K., SHECHTMAN E., SHU Z.: Rignerf: Fully controllable neural 3d portraits. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2022), pp. 20364–20373. 2

[BBGB16] BÉRARD P., BRADLEY D., GROSS M., BEELER T.: Lightweight eye capture using a parametric model. *ACM Trans. Graph. 35*, 4 (2016). URL: http://dx.doi.org/10.1145/2897824.2925962, doi:2897824.2925962. 2

[BBGB19] BÉRARD P., BRADLEY D., GROSS M., BEELER T.: Practical person-specific eye rigging. *Computer Graphics Forum 38* (05 2019), 441–454. doi:10.1111/cgf.13650. 2

[BBK*15] BERMANO A., BEELER T., KOZLOV Y., BRADLEY D., BICKEL B., GROSS M.: Detailed spatio-temporal reconstruction of eyelids. *ACM Trans. Graph. 34*, 4 (jul 2015). URL: https://doi.org/10.1145/2766924, doi:10.1145/2766924. 2

[BBN*14] BÉRARD P., BRADLEY D., NITTI M., BEELER T., GROSS M.: High-quality capture of eyes. *ACM Trans. Graph. 33*, 6 (nov 2014). URL: https://doi.org/10.1145/2661229.2661285, doi:10.1145/2661229.2661285. 2

[BMV*22] BARRON J. T., MILDENHALL B., VERBIN D., SRINIVASAN P. P., HEDMAN P.: Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2022), pp. 5470–5479. 8

[BTH*23] BAI Z., TAN F., HUANG Z., SARKAR K., TANG D., QIU D., MEKA A., DU R., DOU M., ORTS-ESCOLANO S., ET AL.: Learning personalized high quality volumetric head avatars from monocular rgb videos. *arXiv preprint arXiv:2304.01436* (2023). 2, 8, 9, 10

[BV99] BLANZ V., VETTER T.: A morphable model for the synthesis of 3d faces. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques* (1999), pp. 187–194. 3

[CFHT23] CHEN Z., FUNKHOUSER T., HEDMAN P., TAGLIASACCHI A.: Mobilenerf: Exploiting the polygon rasterization pipeline for efficient neural field rendering on mobile architectures. In *The Conference on Computer Vision and Pattern Recognition (CVPR)* (2023). 3

[CH19] CAÑIGUERAL R., HAMILTON A. F. D. C.: The role of eye gaze during natural social interactions in typical and autistic people. *Frontiers in Psychology 10* (2019). URL: https://www.frontiersin.org/articles/10.3389/fpsyg.2019.00560, doi:10.3389/fpsyg.2019.00560. 1

[CSK*22] CAO C., SIMON T., KIM J. K., SCHWARTZ G., ZOLLHOEFER M., SAITO S.-S., LOMBARDI S., WEI S.-E., BELKO D., YU S.-I., SHEIKH Y., SARAGIH J.: Authentic volumetric avatars from a phone scan. *ACM Trans. Graph. 41*, 4 (jul 2022). URL: https://doi.org/10.1145/3528223.3530143, doi:10.1145/3528223.3530143. 2, 13

[DJJ22] DONG Y., JÖRG S., JAIN E.: Is the avatar scared? pupil as a perceptual cue. *Computer Animation and Virtual Worlds 33*, 2 (2022), e2040. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/cav.2040, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/cav.2040, doi:https://doi.org/10.1002/cav.2040. 1

[DYXT22] DENG Y., YANG J., XIANG J., TONG X.: Gram: Generative radiance manifolds for 3d-aware image generation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2022). 3

[EP13] ENGELMANN J., POGOSYAN M.: Emotion perception across cultures: the role of cognitive mechanisms. *Frontiers in Psychology 4* (2013). URL: https://www.frontiersin.org/articles/10.3389/fpsyg.2013.00118, doi:10.3389/fpsyg.2013.00118. 1

[GKE*22] GARBIN S. J., KOWALSKI M., ESTELLERS V., SZYMANOWICZ S., REZAEIFAR S., SHEN J., JOHNSON M., VALENTIN J.: Voltemorph: Realtime, controllable and generalisable animation of volumetric representations. *arXiv preprint arXiv:2208.00949* (2022). 2, 3

[GMFB*18] GERIG T., MOREL-FORSTER A., BLUMER C., EGGER B., LUTHI M., SCHÖNBORN S., VETTER T.: Morphable face models-an open framework. In *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)* (2018), IEEE, pp. 75–82. 3

[GTZN21] GAFNI G., THIES J., ZOLLHÖFER M., NIESSNER M.: Dynamic neural radiance fields for monocular 4d facial avatar reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2021), pp. 8649–8658. 2, 8, 9, 10

[HPX*22] HONG Y., PENG B., XIAO H., LIU L., ZHANG J.: Headnerf: A real-time nerf-based parametric head model. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2022). 2

[JGY*12] JACK R. E., GARROD O. G. B., YU H., CALDARA R., SCHYNS P. G.: Facial expressions of emotion are not culturally universal. *Proceedings of the National Academy of Sciences of the United States of America 109*, 19 (2012), 7241–7244. URL: http://www.jstor.org/stable/41592999. 1

[KADM22] KERBIRIOU G., AVRIL Q., DANIEAU F., MARCHAL M.: Detailed eye region capture and animation. *Computer Graphics Forum 41*, 8 (2022), 279–282. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.14642, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.14642, doi:https://doi.org/10.1111/cgf.14642. 2

[KB14] KINGMA D. P., BA J.: Adam: A method for stochastic optimization. *CoRR abs/1412.6980* (2014). URL: https://api.semanticscholar.org/CorpusID:6628106. 8

[Kre15] KRET M. E.: Emotional expressions beyond facial muscle actions. a call for studying autonomic signals and their impact on social perception. *Frontiers in Psychology 6* (2015). URL: https://www.frontiersin.org/articles/10.3389/fpsyg.2015.00711, doi:10.3389/fpsyg.2015.00711. 1

[KRP*15] KLEHM O., ROUSSELLE F., PAPAS M., BRADLEY D., HERY C., BICKEL B., JAROSZ W., BEELER T.: Recent advances in facial appearance capture. *Computer Graphics Forum (Proceedings of Eurographics - State of the Art Reports) 34*, 2 (May 2015), 709–733. doi:10/f7mb4b. 2

[LA17] LEE D. H., ANDERSON A. K.: Reading what the mind thinks from how the eye sees. *Psychological Science 28*, 4 (2017), 494–503. PMID: 28406382. URL: https://doi.org/10.1177/0956797616687364, arXiv:https://doi.org/10.1177/0956797616687364, doi:10.1177/0956797616687364. 1

[LHR*21] LIU L., HABERMANN M., RUDNEV V., SARKAR K., GU J., THEOBALT C.: Neural actor: Neural free-view synthesis of human actors with pose control. *ACM Trans. Graph. 40*, 6 (dec 2021). URL: https://doi.org/10.1145/3478513.3480528, doi:10.1145/3478513.3480528. 2

[LMM*22] LI G., MEKA A., MUELLER F., BUEHLER M. C., HILLIGES O., BEELER T.: Eyenerf: a hybrid representation for photorealistic synthesis, animation and relighting of human eyes. *ACM Transactions on Graphics (TOG) 41*, 4 (2022), 1–16. 2, 3, 6, 7, 9

[LSS*21] LOMBARDI S., SIMON T., SCHWARTZ G., ZOLLHOEFER M., SHEIKH Y., SARAGIH J.: Mixture of volumetric primitives for efficient neural rendering. *ACM Trans. Graph. 40*, 4 (jul 2021). URL: https://doi.org/10.1145/3450626.3459863, doi:10.1145/3450626.3459863. 2, 9

[LW10] LOOSER C. E., WHEATLEY T.: The tipping point of animacy: How, when, and where we perceive life in a face. *Psychological Science 21*, 12 (2010), 1854–1862. URL: http://www.jstor.org/stable/40984586. 1

[MESK22] MÜLLER T., EVANS A., SCHIED C., KELLER A.: Instant neural graphics primitives with a multiresolution hash encoding. *arXiv:2201.05989* (Jan. 2022). 2, 3

[MST*20] MILDENHALL B., SRINIVASAN P. P., TANCIK M., BARRON J. T., RAMAMOORTHI R., NG R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision* (2020), Springer, pp. 405–421. 3

[PBFJ05] PORUMBESCU S. D., BUDGE B., FENG L., JOY K. I.: Shell maps. *ACM Transactions on Graphics (TOG) 24*, 3 (2005), 626–633. 2, 3

[PEL*21] PANDEY R., ESCOLANO S. O., LEGENDRE C., HAENE C., BOUAZIZ S., RHEMANN C., DEBEVEC P., FANELLO S.: Total relighting: learning to relight portraits for background replacement. *ACM Transactions on Graphics (TOG) 40*, 4 (2021), 1–21. 7

[Pol23] POLYWINK: Polywink blenshapes. http://www.polywink.com, 2023. Accessed: 2023-09-29. 3

[PSB*21] PARK K., SINHA U., BARRON J. T., BOUAZIZ S., GOLDMAN D. B., SEITZ S. M., MARTIN-BRUALLA R.: Nerfies: Deformable neural radiance fields. *ICCV* (2021). 2, 3, 5, 8, 10

[PSH*21] PARK K., SINHA U., HEDMAN P., BARRON J. T., BOUAZIZ S., GOLDMAN D. B., MARTIN-BRUALLA R., SEITZ S. M.: Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields. *ACM Trans. Graph. 40*, 6 (dec 2021). 2

[SWW*20] SCHWARTZ G., WEI S.-E., WANG T.-L., LOMBARDI S., SIMON T., SARAGIH J., SHEIKH Y.: The eyes have it: An integrated eye and face model for photorealistic facial animation. *ACM Transactions on Graphics (TOG) 39*, 4 (2020), 91–1. 2, 13

[WBM*16] WOOD E., BALTRUŠAITIS T., MORENCY L.-P., ROBINSON P., BULLING A.: A 3d morphable eye region model for gaze estimation. In *Computer Vision – ECCV 2016* (Cham, 2016), Leibe B., Matas J., Sebe N., Welling M., (Eds.), Springer International Publishing, pp. 297–313. 2

[WDY*22] WU Y., DENG Y., YANG J., WEI F., QIFENG C., TONG X.: Anifacegan: Animatable 3d-aware face image generation for video avatars. In *Advances in Neural Information Processing Systems* (2022). 3

[WXLY17] WEN Q., XU F., LU M., YONG J.-H.: Real-time 3d eyelids tracking from semantic edges. *ACM Trans. Graph. 36*, 6 (nov 2017). URL: https://doi.org/10.1145/3130800.3130837, doi:10.1145/3130800.3130837. 2

[XFM22] XU T., FUJITA Y., MATSUMOTO E.: Surface-aligned neural radiance fields for controllable 3d human synthesis. In *CVPR* (2022). 2

[ZBT22] ZIELONKA W., BOLKART T., THIES J.: Instant volumetric head avatars, 2022. arXiv:2211.12499. 3

[ZTB*18] ZOLLHÖFER M., THIES J., BRADLEY D., GARRIDO P., BEELER T., PÉEREZ P., STAMMINGER M., NIESSNER M., THEOBALT C.: State of the art on monocular 3d face reconstruction, tracking, and applications. 2