

Face2Face: Real-time Face Capture and Reenactment of RGB Videos

Justus Thies¹ Michael Zollhöfer² Marc Stamminger¹ Christian Theobalt² Matthias Nießner³

¹University of Erlangen-Nuremberg ²Max-Planck-Institute for Informatics ³Stanford University



Proposed online reenactment setup: a monocular target video sequence (e.g., from Youtube) is reenacted based on the expressions of a source actor who is recorded live with a commodity webcam.

Abstract

We present a novel approach for real-time facial reenactment of a monocular target video sequence (e.g., Youtube video). The source sequence is also a monocular video stream, captured live with a commodity webcam. Our goal is to animate the facial expressions of the target video by a source actor and re-render the manipulated output video in a photo-realistic fashion. To this end, we first address the under-constrained problem of facial identity recovery from monocular video by non-rigid model-based bundling. At run time, we track facial expressions of both source and target video using a dense photometric consistency measure. Reenactment is then achieved by fast and efficient deformation transfer between source and target. The mouth interior that best matches the re-targeted expression is retrieved from the target sequence and warped to produce an accurate fit. Finally, we convincingly re-render the synthesized target face on top of the corresponding video stream such that it seamlessly blends with the real-world illumination. We demonstrate our method in a live setup, where Youtube videos are reenacted in real time.

1. Introduction

In recent years, real-time markerless facial performance capture based on commodity sensors has been demonstrated. Impressive results have been achieved, both based

on RGB [8, 6] as well as RGB-D data [32, 10, 21, 4, 16]. These techniques have become increasingly popular for the animation of virtual CG avatars in video games and movies. It is now feasible to run these face capture and tracking algorithms from home, which is the foundation for many VR and AR applications, such as teleconferencing.

In this paper, we employ a new dense markerless facial performance capture method based on monocular RGB data, similar to state-of-the-art methods. However, instead of transferring facial expressions to virtual CG characters, our main contribution is monocular *facial reenactment* in real-time. In contrast to previous reenactment approaches that run offline [5, 11, 13], our goal is the *online* transfer of facial expressions of a source actor captured by an RGB sensor to a target actor. The target sequence can be any monocular video; e.g., legacy video footage downloaded from Youtube with a facial performance. We aim to modify the target video in a photo-realistic fashion, such that it is virtually impossible to notice the manipulations. Faithful photo-realistic facial reenactment is the foundation for a variety of applications; for instance, in video conferencing, the video feed can be adapted to match the face motion of a translator, or face videos can be convincingly dubbed to a foreign language.

In our method, we first reconstruct the shape identity of the target actor using a new global non-rigid model-based bundling approach based on a prerecorded training sequence. As this preprocess is performed globally on a set of training frames, we can resolve geometric ambiguities

common to monocular reconstruction. At runtime, we track both the expressions of the source and target actor’s video by a dense analysis-by-synthesis approach based on a statistical facial prior. We demonstrate that our RGB tracking accuracy is on par with the state of the art, even with online tracking methods relying on depth data. In order to transfer expressions from the source to the target actor in real-time, we propose a novel transfer functions that efficiently applies deformation transfer [27] directly in the used low-dimensional expression space. For final image synthesis, we re-render the target’s face with transferred expression coefficients and composite it with the target video’s background under consideration of the estimated environment lighting. Finally, we introduce a new image-based mouth synthesis approach that generates a realistic mouth interior by retrieving and warping best matching mouth shapes from the offline sample sequence. It is important to note that we maintain the appearance of the target mouth shape; in contrast, existing methods either copy the source mouth region onto the target [31, 11] or a generic teeth proxy is rendered [14, 29], both of which leads to inconsistent results. Fig. 1 shows an overview of our method.

We demonstrate highly-convincing transfer of facial expressions from a source to a target video in real time. We show results with a live setup where a source video stream, which is captured by a webcam, is used to manipulate a target YouTube video. In addition, we compare against state-of-the-art reenactment methods, which we outperform both in terms of resulting video quality and runtime (we are the first real-time RGB reenactment method). In summary, our key contributions are:

- dense, global non-rigid model-based bundling,
- accurate tracking, appearance, and lighting estimation in unconstrained live RGB video,
- person-dependent expression transfer using subspace deformations,
- and a novel mouth synthesis approach.

2. Related Work

Offline RGB Performance Capture Recent offline performance capture techniques approach the hard monocular reconstruction problem by fitting a blendshape [15] or a multi-linear face [26] model to the input video sequence. Even geometric fine-scale surface detail is extracted via inverse shading-based surface refinement. Ichim et al. [17] build a personalized face rig from just monocular input. They perform a structure-from-motion reconstruction of the static head from a specifically captured video, to which they fit an identity and expression model. Person-specific expressions are learned from a training sequence. Suwanjanakorn et al. [28] learn an identity model from a collection of images and track the facial animation based on a

model-to-image flow field. Shi et al. [26] achieve impressive results based on global energy optimization of a set of selected keyframes. Our model-based bundling formulation to recover actor identities is similar to their approach; however, we use robust and dense global photometric alignment, which we enforce with an efficient data-parallel optimization strategy on the GPU.

Online RGB-D Performance Capture Weise et al. [33] capture facial performances in real-time by fitting a parametric blendshape model to RGB-D data, but they require a professional, custom capture setup. The first real-time facial performance capture system based on a commodity depth sensor has been demonstrated by Weise et al. [32]. Follow up work [21, 4, 10, 16] focused on corrective shapes [4], dynamically adapting the blendshape basis [21], non-rigid mesh deformation [10], and robustness against occlusions [16]. These works achieve impressive results, but rely on depth data which is typically unavailable in most video footage.

Online RGB Performance Capture While many sparse real-time face trackers exist, e.g., [25], real-time dense monocular tracking is the basis of realistic online facial reenactment. Cao et al. [8] propose a real-time regression-based approach to infer 3D positions of facial landmarks which constrain a user-specific blendshape model. Follow-up work [6] also regresses fine-scale face wrinkles. These methods achieve impressive results, but are not directly applicable as a component in facial reenactment, since they do not facilitate dense, pixel-accurate tracking.

Offline Reenactment Vlasic et al. [31] perform facial reenactment by tracking a face template, which is re-rendered under different expression parameters on top of the target; the mouth interior is directly copied from the source video. Dale et al. [11] achieve impressive results using a parametric model, but they target face replacement and compose the source face over the target. Image-based offline mouth re-animation was shown in [5]. Garrido et al. [13] propose an automatic purely image-based approach to replace the entire face. These approaches merely enable self-reenactment; i.e., when source and target are the same person; in contrast, we perform reenactment of a different target actor. Recent work presents virtual dubbing [14], a problem similar to ours; however, the method runs at slow offline rates and relies on a generic teeth proxy for the mouth interior. Kemelmacher et al. [20] generate face animations from large image collections, but the obtained results lack temporal coherence. Li et al. [22] retrieve frames from a database based on a similarity metric. They use optical flow as appearance and velocity measure and search for the k -nearest neighbors based on time stamps and flow

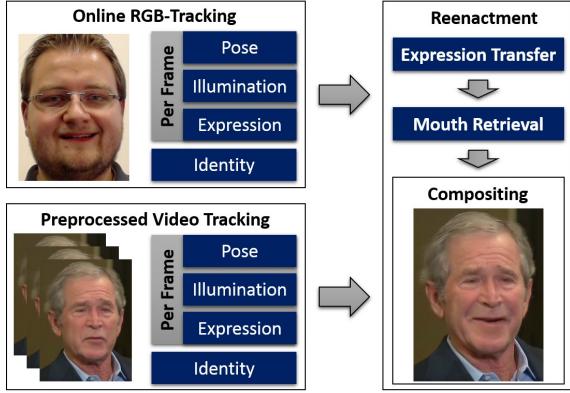


Figure 1: Method overview.

distance. Saragih et al. [25] present a real-time avatar animation system from a single image. Their approach is based on sparse landmark tracking, and the mouth of the source is copied to the target using texture warping. Berthouzoz et al. [2] find a flexible number of in-between frames for a video sequence using shortest path search on a graph that encodes frame similarity. Kawai et al. [18] re-synthesize the inner mouth for a given frontal 2D animation using a tooth and tongue image database; they are limited to frontal poses, and do not produce as realistic renderings as ours under general head motion.

Online Reenactment Recently, first online facial reenactment approaches based on RGB-(D) data have been proposed. Kemelmacher-Shlizerman et al. [19] enable image-based puppetry by querying similar images from a database. They employ an appearance cost metric and consider rotation angular distance, which is similar to Kemelmacher et al. [20]. While they achieve impressive results, the retrieved stream of faces is not temporally coherent. Thies et al. [29] show the first online reenactment system; however, they rely on depth data and use a generic teeth proxy for the mouth region. In this paper, we address both shortcomings: 1) our method is the first real-time RGB-only reenactment technique; 2) we synthesize the mouth regions exclusively from the target sequence (no need for a teeth proxy or direct source-to-target copy).

3. Synthesis of Facial Imagery

We use a multi-linear PCA model based on [3, 1, 9]. The first two dimensions represent facial identity – i.e., geometric shape and skin reflectance – and the third dimension controls the facial expression. Hence, we parametrize a face as:

$$\mathcal{M}_{\text{geo}}(\boldsymbol{\alpha}, \boldsymbol{\delta}) = \mathbf{a}_{\text{id}} + E_{\text{id}} \cdot \boldsymbol{\alpha} + E_{\text{exp}} \cdot \boldsymbol{\delta}, \quad (1)$$

$$\mathcal{M}_{\text{alb}}(\boldsymbol{\beta}) = \mathbf{a}_{\text{alb}} + E_{\text{alb}} \cdot \boldsymbol{\beta}. \quad (2)$$

This prior assumes a multivariate normal probability distribution of shape and reflectance around the average shape $\mathbf{a}_{\text{id}} \in \mathbb{R}^{3n}$ and reflectance $\mathbf{a}_{\text{alb}} \in \mathbb{R}^{3n}$. The shape $E_{\text{id}} \in \mathbb{R}^{3n \times 80}$, reflectance $E_{\text{alb}} \in \mathbb{R}^{3n \times 80}$, and expression $E_{\text{exp}} \in \mathbb{R}^{3n \times 76}$ basis and the corresponding standard deviations $\sigma_{\text{id}} \in \mathbb{R}^{80}$, $\sigma_{\text{alb}} \in \mathbb{R}^{80}$, and $\sigma_{\text{exp}} \in \mathbb{R}^{76}$ are given. The model has 53K vertices and 106K faces. A synthesized image C_S is generated through rasterization of the model under a rigid model transformation $\Phi(\mathbf{v})$ and the full perspective transformation $\Pi(\mathbf{v})$. Illumination is approximated by the first three bands of Spherical Harmonics (SH) [23] basis functions, assuming Labertian surfaces and smooth distant illumination, neglecting self-shadowing.

Synthesis is dependent on the face model parameters $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, $\boldsymbol{\delta}$, the illumination parameters γ , the rigid transformation \mathbf{R} , \mathbf{t} , and the camera parameters κ defining Π . The vector of unknowns \mathcal{P} is the union of these parameters.

4. Energy Formulation

Given a monocular input sequence, we reconstruct all unknown parameters \mathcal{P} jointly with a robust variational optimization. The proposed objective is highly non-linear in the unknowns and has the following components:

$$E(\mathcal{P}) = \underbrace{w_{\text{col}} E_{\text{col}}(\mathcal{P}) + w_{\text{lan}} E_{\text{lan}}(\mathcal{P})}_{\text{data}} + \underbrace{w_{\text{reg}} E_{\text{reg}}(\mathcal{P})}_{\text{prior}}. \quad (3)$$

The data term measures the similarity between the synthesized imagery and the input data in terms of photo-consistency E_{col} and facial feature alignment E_{lan} . The likelihood of a given parameter vector \mathcal{P} is taken into account by the statistical regularizer E_{reg} . The weights w_{col} , w_{lan} , and w_{reg} balance the three different sub-objectives. In all of our experiments, we set $w_{\text{col}} = 1$, $w_{\text{lan}} = 10$, and $w_{\text{reg}} = 2.5 \cdot 10^{-5}$. In the following, we introduce the different sub-objectives.

Photo-Consistency In order to quantify how well the input data is explained by a synthesized image, we measure the photo-metric alignment error on pixel level:

$$E_{\text{col}}(\mathcal{P}) = \frac{1}{|\mathcal{V}|} \sum_{\mathbf{p} \in \mathcal{V}} \|C_S(\mathbf{p}) - C_I(\mathbf{p})\|_2, \quad (4)$$

where C_S is the synthesized image, C_I is the input RGB image, and $\mathbf{p} \in \mathcal{V}$ denote all visible pixel positions in C_S . We use the $\ell_{2,1}$ -norm [12] instead of a least-squares formulation to be robust against outliers. In our scenario, distance in color space is based on ℓ_2 , while in the summation over all pixels an ℓ_1 -norm is used to enforce sparsity.

Feature Alignment In addition, we enforce feature similarity between a set of salient facial feature point pairs de-

tected in the RGB stream:

$$E_{\text{lan}}(\mathcal{P}) = \frac{1}{|\mathcal{F}|} \sum_{f_j \in \mathcal{F}} w_{\text{conf},j} \|f_j - \Pi(\Phi(v_j))\|_2^2. \quad (5)$$

To this end, we employ a state-of-the-art facial landmark tracking algorithm by [24]. Each feature point $f_j \in \mathcal{F} \subset \mathbb{R}^2$ comes with a detection confidence $w_{\text{conf},j}$ and corresponds to a unique vertex $v_j = \mathcal{M}_{\text{geo}}(\alpha, \delta) \in \mathbb{R}^3$ of our face prior. This helps avoiding local minima in the highly-complex energy landscape of $E_{\text{col}}(\mathcal{P})$.

Statistical Regularization We enforce plausibility of the synthesized faces based on the assumption of a normal distributed population. To this end, we enforce the parameters to stay statistically close to the mean:

$$E_{\text{reg}}(\mathcal{P}) = \sum_{i=1}^{80} \left[\left(\frac{\alpha_i}{\sigma_{\text{id},i}} \right)^2 + \left(\frac{\beta_i}{\sigma_{\text{alb},i}} \right)^2 \right] + \sum_{i=1}^{76} \left(\frac{\delta_i}{\sigma_{\text{exp},i}} \right)^2. \quad (6)$$

This commonly-used regularization strategy prevents degenerations of the facial geometry and reflectance, and guides the optimization strategy out of local minima [3].

5. Data-parallel Optimization Strategy

The proposed robust tracking objective is a general unconstrained non-linear optimization problem. We minimize this objective in real-time using a novel data-parallel GPU-based *Iteratively Reweighted Least Squares* (IRLS) solver. The key idea of IRLS is to transform the problem, in each iteration, to a non-linear least-squares problem by splitting the norm in two components:

$$\|r(\mathcal{P})\|_2 = \underbrace{(\|r(\mathcal{P}_{\text{old}})\|_2)^{-1}}_{\text{constant}} \cdot \|r(\mathcal{P})\|_2^2.$$

Here, $r(\cdot)$ is a general residual and \mathcal{P}_{old} is the solution computed in the last iteration. Thus, the first part is kept constant during one iteration and updated afterwards. Close in spirit to [29], each single iteration step is implemented using the Gauss-Newton approach. We take a single GN step in every IRLS iteration and solve the corresponding system of normal equations $\mathbf{J}^T \mathbf{J} \delta^* = -\mathbf{J}^T \mathbf{F}$ based on PCG to obtain an optimal linear parameter update δ^* . The Jacobian \mathbf{J} and the systems' right hand side $-\mathbf{J}^T \mathbf{F}$ are precomputed and stored in device memory for later processing as proposed by Thies et al. [29]. As suggested by [34, 29], we split up the multiplication of the old descent direction d with the system matrix $\mathbf{J}^T \mathbf{J}$ in the PCG solver into two successive matrix-vector products. Additional details regarding the optimization framework are provided in the supplemental material.

6. Non-Rigid Model-Based Bundling

To estimate the identity of the actors in the heavily under-constrained scenario of monocular reconstruction, we introduce a non-rigid model-based bundling approach. Based

on the proposed objective, we jointly estimate all parameters over k key-frames of the input video sequence. The estimated unknowns are the global identity $\{\alpha, \beta\}$ and intrinsics κ as well as the unknown per-frame pose $\{\delta^k, \mathbf{R}^k, \mathbf{t}^k\}_k$ and illumination parameters $\{\gamma^k\}_k$. We use a similar data-parallel optimization strategy as proposed for model-to-frame tracking, but jointly solve the normal equations for the entire keyframe set. For our non-rigid model-based bundling problem, the non-zero structure of the corresponding Jacobian is block dense. Our PCG solver exploits the non-zero structure for increased performance (see additional document). Since all keyframes observe the same face identity under potentially varying illumination, expression, and viewing angle, we can robustly separate identity from all other problem dimensions. Note that we also solve for the intrinsic camera parameters of Π , thus being able to process uncalibrated video footage.

7. Expression Transfer

To transfer the expression changes from the source to the target actor while preserving person-specificness in each actor's expressions, we propose a sub-space deformation transfer technique. We are inspired by the deformation transfer energy of Sumner et al. [27], but operate directly in the space spanned by the expression blendshapes. This not only allows for the precomputation of the pseudo-inverse of the system matrix, but also drastically reduces the dimensionality of the optimization problem allowing for fast real-time transfer rates. Assuming source identity α^S and target identity α^T fixed, transfer takes as input the neutral δ_N^S , deformed source δ^S , and the neutral target δ_N^T expression. Output is the transferred facial expression δ^T directly in the reduced sub-space of the parametric prior.

As proposed by [27], we first compute the source deformation gradients $\mathbf{A}_i \in \mathbb{R}^{3 \times 3}$ that transform the source triangles from neutral to deformed. The deformed target $\hat{v}_i = \mathbf{M}_i(\alpha^T, \delta^T)$ is then found based on the undeformed state $v_i = \mathbf{M}_i(\alpha^T, \delta_N^T)$ by solving a linear least-squares problem. Let (i_0, i_1, i_2) be the vertex indices of the i -th triangle, $\mathbf{V} = [v_{i_1} - v_{i_0}, v_{i_2} - v_{i_0}]$ and $\hat{\mathbf{V}} = [\hat{v}_{i_1} - \hat{v}_{i_0}, \hat{v}_{i_2} - \hat{v}_{i_0}]$, then the optimal unknown target deformation δ^T is the minimizer of:

$$E(\delta^T) = \sum_{i=1}^{|F|} \left\| \mathbf{A}_i \mathbf{V} - \hat{\mathbf{V}} \right\|_F^2. \quad (7)$$

This problem can be rewritten in the canonical least-squares form by substitution:

$$E(\delta^T) = \left\| \mathbf{A} \delta^T - b \right\|_2^2. \quad (8)$$

The matrix $\mathbf{A} \in \mathbb{R}^{6|F| \times 76}$ is constant and contains the edge information of the template mesh projected to the expression sub-space. Edge information of the target in neutral

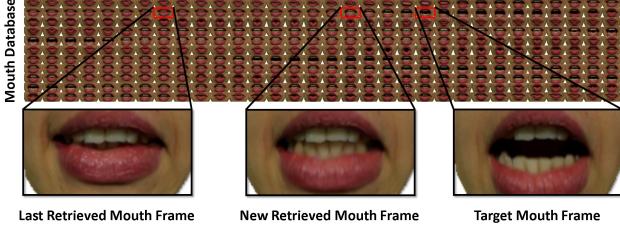


Figure 2: Mouth Retrieval: we use an appearance graph to retrieve new mouth frames. In order to select a frame, we enforce similarity to the previously-retrieved frame while minimizing the distance to the target expression.

expression is included in the right-hand side $\mathbf{b} \in \mathbb{R}^{6|\mathcal{F}|}$. \mathbf{b} varies with δ^S and is computed on the GPU for each new input frame. The minimizer of the quadratic energy can be computed by solving the corresponding normal equations. Since the system matrix is constant, we can precompute its *Pseudo Inverse* using a Singular Value Decomposition (SVD). Later, the small 76×76 linear system is solved in real-time. No additional smoothness term as in [27, 4] is needed, since the blendshape model implicitly restricts the result to plausible shapes and guarantees smoothness.

8. Mouth Retrieval

For a given transferred facial expression, we need to synthesize a realistic target mouth region. To this end, we retrieve and warp the best matching mouth image from the target actor sequence. We assume that sufficient mouth variation is available in the target video. It is also important to note that we maintain the appearance of the target mouth. This leads to much more realistic results than either copying the source mouth region [31, 11] or using a generic 3D teeth proxy [14, 29].

Our approach first finds the best fitting target mouth frame based on a frame-to-cluster matching strategy with a novel feature similarity metric. To enforce temporal coherence, we use a dense appearance graph to find a compromise between the last retrieved mouth frame and the target mouth frame (cf. Fig. 2). We detail all steps in the following.

Similarity Metric Our similarity metric is based on geometric and photometric features. The used descriptor $\mathcal{K} = \{\mathbf{R}, \boldsymbol{\delta}, \mathcal{F}, \mathcal{L}\}$ of a frame is composed of the rotation \mathbf{R} , expression parameters $\boldsymbol{\delta}$, landmarks \mathcal{F} , and a Local Binary Pattern (LBP) \mathcal{L} . We compute these descriptors \mathcal{K}^S for every frame in the training sequence. The target descriptor \mathcal{K}^T consists of the result of the expression transfer and the LBP of the frame of the driving actor. We measure the distance between a source and a target descriptor as follows:

$$D(\mathcal{K}^T, \mathcal{K}_t^S, t) = D_p(\mathcal{K}^T, \mathcal{K}_t^S) + D_m(\mathcal{K}^T, \mathcal{K}_t^S) + D_a(\mathcal{K}^T, \mathcal{K}_t^S, t).$$

The first term D_p measures the distance in parameter space:

$$D_p(\mathcal{K}^T, \mathcal{K}_t^S) = \|\boldsymbol{\delta}^T - \boldsymbol{\delta}_t^S\|_2^2 + \|\mathbf{R}^T - \mathbf{R}_t^S\|_F^2.$$

The second term D_m measures the differential compatibility of the sparse facial landmarks:

$$D_m(\mathcal{K}^T, \mathcal{K}_t^S) = \sum_{(i,j) \in \Omega} (\|\mathcal{F}_i^T - \mathcal{F}_j^T\|_2 - \|\mathcal{F}_{t,i}^S - \mathcal{F}_{t,j}^S\|_2)^2.$$

Here, Ω is a set of predefined landmark pairs, defining distances such as between the upper and lower lip or between the left and right corner of the mouth. The last term D_a is an appearance measurement term composed of two parts:

$$D_a(\mathcal{K}^T, \mathcal{K}_t^S, t) = D_l(\mathcal{K}^T, \mathcal{K}_t^S) + w_c(\mathcal{K}^T, \mathcal{K}_t^S)D_c(\tau, t).$$

τ is the last retrieved frame index used for the reenactment in the previous frame. $D_l(\mathcal{K}^T, \mathcal{K}_t^S)$ measures the similarity based on LBPs that are compared via a *Chi Squared Distance* (for details see [13]). $D_c(\tau, t)$ measures the similarity between the last retrieved frame τ and the video frame t based on RGB cross-correlation of the normalized mouth frames. Note that the mouth frames are normalized based on the models texture parameterization (cf. Fig. 2). To facilitate fast frame jumps for expression changes, we incorporate the weight $w_c(\mathcal{K}^T, \mathcal{K}_t^S) = e^{-(D_m(\mathcal{K}^T, \mathcal{K}_t^S))^2}$. We apply this frame-to-frame distance measure in a frame-to-cluster matching strategy, which enables real-time rates and mitigates high-frequency jumps between mouth frames.

Frame-to-Cluster Matching Utilizing the proposed similarity metric, we cluster the target actor sequence into $k = 10$ clusters using a modified k-means algorithm that is based on the pairwise distance function D . For every cluster, we select the frame with the minimal distance to all other frames within that cluster as a representative. During runtime, we measure the distances between the target descriptor \mathcal{K}^T and the descriptors of cluster representatives, and choose the cluster whose representative frame has the minimal distance as the new target frame.

Appearance Graph We improve temporal coherence by building a fully-connected appearance graph of all video frames. The edge weights are based on the RGB cross-correlation between the normalized mouth frames, the distance in parameter space D_p , and the distance of the landmarks D_m . The graph enables us to find an inbetween frame that is both similar to the last retrieved frame and the retrieved target frame (see Fig. 2). We compute this perfect match by finding the frame of the training sequence that minimizes the sum of the edge weights to the last retrieved and current target frame. We blend between the previously-retrieved frame and the newly-retrieved frame in texture

| CPU | | GPU | | | FPS |
|----------|---------|---------|----------|---------|---------------|
| SparseFT | MouthRT | DenseFT | DeformTF | Synth | |
| 5.97ms | 1.90ms | 22.06ms | 3.98ms | 10.19ms | 27.6Hz |
| 4.85ms | 1.50ms | 21.27ms | 4.01ms | 10.31ms | 28.1Hz |
| 5.57ms | 1.78ms | 20.97ms | 3.95ms | 10.32ms | 28.4Hz |

Table 1: Avg. run times for the three sequences of Fig. 8, from top to bottom. Standard deviations w.r.t. the final frame rate are 0.51, 0.56, and 0.59 fps, respectively. Note that CPU and GPU stages run in parallel.

space on a pixel level after optic flow alignment. Before blending, we apply an illumination correction that considers the estimated Spherical Harmonic illumination parameters of the retrieved frames and the current video frame. Finally, we composite the new output frame by alpha blending between the original video frame, the illumination-corrected, projected mouth frame, and the rendered face model.

9. Results

Live Reenactment Setup Our live reenactment setup consists of standard consumer-level hardware. We capture a live video with a commodity webcam (source), and download monocular video clips from Youtube (target). In our experiments, we use a *Logitech HD Pro C920* camera running at 30Hz in a resolution of 640×480 ; although our approach is applicable to any consumer RGB camera. Overall, we show highly-realistic reenactment examples of our algorithm on a variety of target Youtube videos at a resolution of 1280×720 . The videos show different subjects in different scenes filmed from varying camera angles; each video is reenacted by several volunteers as source actors. Reenactment results are generated at a resolution of 1280×720 . We show real-time reenactment results in Fig. 8 and in the accompanying video.

Runtime For all experiments, we use three hierarchy levels for tracking (source and target). In pose optimization, we only consider the second and third level, where we run one and seven Gauss-Newton steps, respectively. Within a Gauss-Newton step, we always run four PCG steps. In addition to tracking, our reenactment pipeline has additional stages whose timings are listed in Table 1. Our method runs in real-time on a commodity desktop computer with an NVIDIA Titan X and an Intel Core i7-4770.

Tracking Comparison to Previous Work Face tracking alone is not the main focus of our work, but the following comparisons show that our tracking is on par with or exceeds the state of the art.

Shi et al. 2014 [26]: They capture face performances offline from monocular unconstrained RGB video. The close-ups in Fig. 4 show that our online approach yields a closer

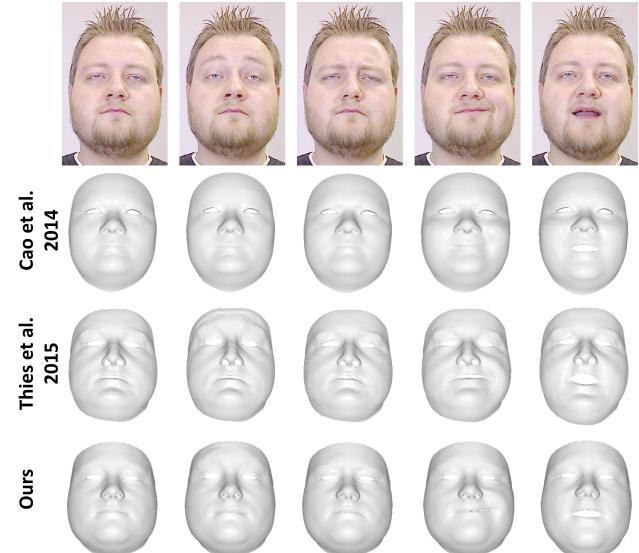


Figure 3: Comparison of our RGB tracking to Cao et al. [7], and to RGB-D tracking by Thies et al. [29].



Figure 4: Comparison of our tracking to Shi et al. [26]. From left to right: RGB input, reconstructed model, overlay with input, close-ups on eye and cheek. Note that Shi et al. perform shape-from-shading in a post process.

face fit, particularly visible at the silhouette of the input face. We believe that our new dense non-rigid bundle adjustment leads to a better shape identity estimate than their sparse approach.

Cao et al. 2014 [7]: They capture face performance from monocular RGB in real-time. In most cases, our and their method produce similar high-quality results (see Fig. 3); our identity and expression estimates are slightly more accurate though.

Thies et al. 2015 [29]: Their approach captures face performance in real-time from RGB-D, Fig. 3. Results of both approaches are similarly accurate; but our approach does not require depth data.



Figure 5: Comparison against *FaceShift* RGB-D tracking.



Figure 6: Dubbing: Comparison to Garrido et al. [14].

FaceShift 2014: We compare our tracker to the commercial real-time RGB-D tracker from *FaceShift*, which is based on the work of Weise et al. [32]. Fig. 5 shows that we obtain similar results from RGB only.

Reenactment Evaluation In Fig. 6, we compare our approach against state-of-the-art reenactment by Garrido et al. [14]. Both methods provide highly-realistic reenactment results; however, their method is fundamentally offline, as they require all frames of a sequence to be present at any time. In addition, they rely on a generic geometric teeth proxy which in some frames makes reenactment less convincing. In Fig. 7, we compare against the work by Thies et al. [29]. Runtime and visual quality are similar for both approaches; however, their geometric teeth proxy leads to undesired appearance changes in the reenacted mouth. Moreover, Thies et al. use an RGB-D camera, which limits the application range; they cannot reenact YouTube videos. We show additional comparisons in the supplemental material against Dale et al. [11] and Garrido et al. [13].

10. Limitations

The assumption of Lambertian surfaces and smooth illumination is limiting, and may lead to artifacts in the presence of hard shadows or specular highlights; a limitation

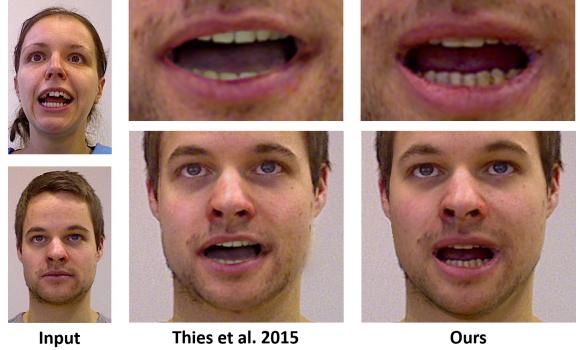


Figure 7: Comparison of the proposed RGB reenactment to the RGB-D reenactment of Thies et al. [29].

shared by most state-of-the-art methods. Scenes with face occlusions by long hair and a beard are challenging. Furthermore, we only reconstruct and track a low-dimensional blendshape model (76 expression coefficients), which omits fine-scale static and transient surface details. Our retrieval-based mouth synthesis assumes sufficient visible expression variation in the target sequence. On a too short sequence, or when the target remains static, we cannot learn the person-specific mouth behavior. In this case, temporal aliasing can be observed, as the target space of the retrieved mouth samples is too sparse. Another limitation is caused by our hardware setup (webcam, USB, and PCI), which introduces a small delay of ≈ 3 frames. Specialized hardware could resolve this, but our aim is a setup with commodity hardware.

11. Conclusion

The presented approach is the first real-time facial reenactment system that requires just monocular RGB input. Our live setup enables the animation of legacy video footage – e.g., from YouTube – in real time. Overall, we believe our system will pave the way for many new and exciting applications in the fields of VR/AR, teleconferencing, or on-the-fly dubbing of videos with translated audio.

Acknowledgements

We would like to thank Chen Cao and Kun Zhou for the blendshape models and comparison data, as well as Volker Blanz, Thomas Vetter, and Oleg Alexander for the provided face data. The facial landmark tracker was kindly provided by TrueVisionSolution. We thank Angela Dai for the video voice over and Daniel Ritchie for video reenactment. This research is funded by the German Research Foundation (DFG), grant GRK-1773 Heterogeneous Image Systems, the ERC Starting Grant 335545 CapReal, and the Max Planck Center for Visual Computing and Communications (MPC-VCC). We also gratefully acknowledge the support from NVIDIA Corporation for hardware donations.



Figure 8: Results of our reenactment system. Corresponding run times are listed in Table 1. The length of the source and resulting output sequences is 965, 1436, and 1791 frames, respectively; the length of the input target sequences is 431, 286, and 392 frames, respectively.

This is a preprint of the accepted version of the following CVPR2016 article: "Face2Face: Real-time Face Capture and Reenactment of RGB Videos".

References

- [1] O. Alexander, M. Rogers, W. Lambeth, M. Chiang, and P.Debevec. The Digital Emily Project: photoreal facial modeling and animation. In *ACM SIGGRAPH Courses*, pages 12:1–12:15. ACM, 2009.
- [2] F. Berthouzoz, W. Li, and M. Agrawala. Tools for placing cuts and transitions in interview video. *ACM TOG*, 31(4):67, 2012.
- [3] V. Blanz and T. Vetter. A morphable model for the synthesis of 3d faces. In *Proc. SIGGRAPH*, pages 187–194. ACM Press/Addison-Wesley Publishing Co., 1999.
- [4] S. Bouaziz, Y. Wang, and M. Pauly. Online modeling for realtime facial animation. *ACM TOG*, 32(4):40, 2013.
- [5] C. Bregler, M. Covell, and M. Slaney. Video rewrite: Driving visual speech with audio. In *Proc. SIGGRAPH*, pages 353–360. ACM Press/Addison-Wesley Publishing Co., 1997.
- [6] C. Cao, D. Bradley, K. Zhou, and T. Beeler. Real-time high-fidelity facial performance capture. *ACM TOG*, 34(4):46:1–46:9, 2015.
- [7] C. Cao, Q. Hou, and K. Zhou. Displaced dynamic expression regression for real-time facial tracking and animation. *ACM TOG*, 33(4):43, 2014.
- [8] C. Cao, Y. Weng, S. Lin, and K. Zhou. 3D shape regression for real-time facial animation. *ACM TOG*, 32(4):41, 2013.
- [9] C. Cao, Y. Weng, S. Zhou, Y. Tong, and K. Zhou. Faceware-house: A 3D facial expression database for visual computing. *IEEE TVCG*, 20(3):413–425, 2014.
- [10] Y.-L. Chen, H.-T. Wu, F. Shi, X. Tong, and J. Chai. Accurate and robust 3d facial capture using a single rgbd camera. *Proc. ICCV*, pages 3615–3622, 2013.
- [11] K. Dale, K. Sunkavalli, M. K. Johnson, D. Vlasic, W. Matusik, and H. Pfister. Video face replacement. *ACM TOG*, 30(6):130, 2011.
- [12] C. H. Q. Ding, D. Zhou, X. He, and H. Zha. R1-pca: rotational invariant 11-norm principal component analysis for robust subspace factorization. In W. W. Cohen and A. Moore, editors, *ICML*, volume 148 of *ACM International Conference Proceeding Series*, pages 281–288. ACM, 2006.
- [13] P. Garrido, L. Valgaerts, O. Rehmsen, T. Thormaehlen, P. Perez, and C. Theobalt. Automatic face reenactment. In *Proc. CVPR*, 2014.
- [14] P. Garrido, L. Valgaerts, H. Sarmadi, I. Steiner, K. Varanasi, P. Perez, and C. Theobalt. Vdub: Modifying face video of actors for plausible visual alignment to a dubbed audio track. In *Computer Graphics Forum*. Wiley-Blackwell, 2015.
- [15] P. Garrido, L. Valgaerts, C. Wu, and C. Theobalt. Reconstructing detailed dynamic face geometry from monocular video. *ACM TOG*, 32(6):158, 2013.
- [16] P.-L. Hsieh, C. Ma, J. Yu, and H. Li. Unconstrained realtime facial performance capture. In *Computer Vision and Pattern Recognition (CVPR)*, 2015.
- [17] A. E. Ichim, S. Bouaziz, and M. Pauly. Dynamic 3d avatar creation from hand-held video input. *ACM TOG*, 34(4):45:1–45:14, 2015.
- [18] M. Kawai, T. Iwao, D. Mima, A. Maejima, and S. Morishima. Data-driven speech animation synthesis focusing on realistic inside of the mouth. *Journal of Information Processing*, 22(2):401–409, 2014.
- [19] I. Kemelmacher-Shlizerman, A. Sankar, E. Shechtman, and S. M. Seitz. Being john malkovich. In *Computer Vision - ECCV 2010, 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part I*, pages 341–353, 2010.
- [20] I. Kemelmacher-Shlizerman, E. Shechtman, R. Garg, and S. M. Seitz. Exploring photobios. *ACM TOG*, 30(4):61, 2011.
- [21] H. Li, J. Yu, Y. Ye, and C. Bregler. Realtime facial animation with on-the-fly correctives. *ACM TOG*, 32(4):42, 2013.
- [22] K. Li, F. Xu, J. Wang, Q. Dai, and Y. Liu. A data-driven approach for facial expression synthesis in video. In *Proc. CVPR*, pages 57–64, 2012.
- [23] R. Ramamoorthi and P. Hanrahan. A signal-processing framework for inverse rendering. In *Proc. SIGGRAPH*, pages 117–128. ACM, 2001.
- [24] J. M. Saragih, S. Lucey, and J. F. Cohn. Deformable model fitting by regularized landmark mean-shift. *IJCV*, 91(2):200–215, 2011.
- [25] J. M. Saragih, S. Lucey, and J. F. Cohn. Real-time avatar animation from a single image. In *Automatic Face and Gesture Recognition Workshops*, pages 213–220, 2011.
- [26] F. Shi, H.-T. Wu, X. Tong, and J. Chai. Automatic acquisition of high-fidelity facial performances using monocular videos. *ACM TOG*, 33(6):222, 2014.
- [27] R. W. Sumner and J. Popović. Deformation transfer for triangle meshes. *ACM TOG*, 23(3):399–405, 2004.
- [28] S. Suwanjanakorn, I. Kemelmacher-Shlizerman, and S. M. Seitz. Total moving face reconstruction. In *Proc. ECCV*, pages 796–812, 2014.
- [29] J. Thies, M. Zollhöfer, M. Nießner, L. Valgaerts, M. Stamminger, and C. Theobalt. Real-time expression transfer for facial reenactment. *ACM Transactions on Graphics (TOG)*, 34(6), 2015.
- [30] J. Thies, M. Zollhöfer, M. Stamminger, C. Theobalt, and M. Nießner. Face2Face: Real-time Face Capture and Reenactment of RGB Videos. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2016.
- [31] D. Vlasic, M. Brand, H. Pfister, and J. Popović. Face transfer with multilinear models. *ACM TOG*, 24(3):426–433, 2005.
- [32] T. Weise, S. Bouaziz, H. Li, and M. Pauly. Realtime performance-based facial animation. 30(4):77, 2011.
- [33] T. Weise, H. Li, L. V. Gool, and M. Pauly. Face/off: Live facial puppetry. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer animation (Proc. SCA'09)*, ETH Zurich, August 2009. Eurographics Association.
- [34] M. Zollhöfer, M. Nießner, S. Izadi, C. Rehmann, C. Zach, M. Fisher, C. Wu, A. Fitzgibbon, C. Loop, C. Theobalt, and M. Stamminger. Real-time Non-rigid Reconstruction using an RGB-D Camera. *ACM TOG*, 33(4):156, 2014.

A. Optimization Framework

Our Gauss-Newton optimization framework is based on the work of Thies et al. [29]. Our aim is to include every visible pixel $p \in \mathcal{V}$ in C_S in the optimization process. To this end, we gather all visible pixels in the synthesized image using a parallel prefix scan. The computation of the Jacobian J of the residual vector F and the gradient $J^T F$ of the energy function are then parallelized across all GPU processors. This parallelization is feasible since all partial derivatives and gradient entries with respect to a variable can be computed independently. During evaluation of the gradient, all components of the Jacobian are computed and stored in global memory. In order to evaluate the gradient, we use a two-stage reduction to sum-up all local per pixel gradients. Finally, we add the regularizer and the sparse feature term to the Jacobian and the gradient.

Using the computed Jacobian J and the gradient $J^T F$, we solve the corresponding normal equation $J^T J \Delta x = -J^T F$ for the parameter update Δx using a preconditioned conjugate gradient (PCG) method. We apply a Jacobi preconditioner that is precomputed during the evaluation of the gradient. To avoid the high computational cost of $J^T J$, our GPU-based PCG method splits up the computation of $J^T J p$ into two successive matrix-vector products.

In order to increase convergence speed and to avoid local minima, we use a coarse-to-fine hierarchical optimization strategy. During online tracking, we only consider the second and third level, where we run one and seven Gauss-Newton steps on the respective level. Within a Gauss-Newton step, we always run four PCG iterations.

Our complete framework is implemented using DirectX for rendering and DirectCompute for optimization. The joint graphics and compute capability of DirectX11 enables the processing of rendered images by the graphics pipeline without resource mapping overhead. In the case of an *analysis-by-synthesis approach* like ours, this is essential to runtime performance, since many rendering-to-compute switches are required.

B. Non-rigid Bundling

For our non-rigid model-based bundling problem, the non-zero structure of the corresponding Jacobian is block dense. We visualize its non-zero structure, which we exploit during optimization, in Fig. 9. In order to leverage the sparse structure of the Jacobian, we adopt the Gauss-Newton framework as follows: we modify the computation of the gradient $J^T(\mathbf{P}) \cdot F(\mathbf{P})$ and the matrix vector product $J^T(\mathbf{P}) \cdot J(\mathbf{P}) \cdot \mathbf{x}$ that is used in the PCG method. To this end, we define a promoter function $\Psi_f : \mathbb{R}^{|\mathcal{P}_{global}|+|\mathcal{P}_{local}|} \rightarrow \mathbb{R}^{|\mathcal{P}_{global}|+k \cdot |\mathcal{P}_{local}|}$ that lifts a per frame parameter vector to the parameter vector space of all frames (Ψ_f^{-1} is the inverse of this promoter function).

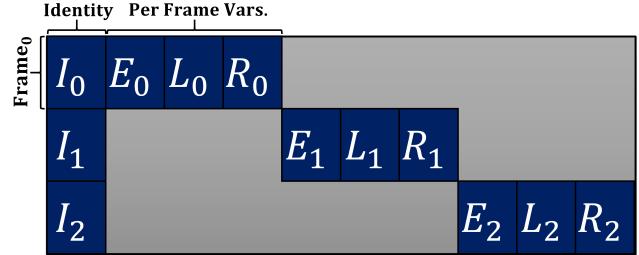


Figure 9: Non-zero structure of the Jacobian matrix of our non-rigid model-based bundling approach for three key-frames. Where I_i, E_i, L_i, R_i are the i -th per frame Jacobian matrices of the identity, expression, illumination, and rigid pose parameters.

\mathcal{P}_{global} are the global parameters that are shared over all frames, such as the identity parameters of the face model and the camera parameters. \mathcal{P}_{local} are the local parameters that are only valid for one specific frame (i.e., facial expression, rigid pose and illumination parameters). Using the promoter function Ψ_f the gradient is given as

$$J^T(\mathbf{P}) \cdot F(\mathbf{P}) = \sum_{f=1}^k \Psi_f(J_f^T(\Psi_f^{-1}(\mathbf{P})) \cdot F_f(\Psi_f^{-1}(\mathbf{P}))),$$

where J_f is the per-frame Jacobian matrix and F_f the corresponding residual vector.

As for the parameter space, we introduce another promoter function $\hat{\Psi}_f$ that lifts a local residual vector to the global residual vector. In contrast to the parameter promoter function, this function varies in every Gauss-Newton iteration since the number of residuals might change. As proposed in [34, 29], we split up the computation of $J^T(\mathbf{P}) \cdot J(\mathbf{P}) \cdot \mathbf{x}$ into two successive matrix vector products, where the second multiplication is analogue to the computation of the gradient. The first multiplication is as follows:

$$J(\mathbf{P}) \cdot \mathbf{x} = \sum_{f=1}^k \hat{\Psi}_f \left(J_f(\Psi_f^{-1}(\mathbf{P})) \cdot \Psi_f^{-1}(\mathbf{x}) \right)$$

Using this scheme, we are able to efficiently solve the normal equations.

The Gauss-Newton framework is embedded in a hierarchical solution strategy (see Fig. 10). This hierarchy allows to prevent convergence to local minima. We start optimizing on a coarse level and propagate the solution to the next finer level using the parametric face model. In our experiments we used three levels with 25, 5, and 1 Gauss-Newton iterations for the coarsest, the medium and the finest level respectively, each with 4 PCG steps. Our implementation is not restricted to the number k of used keyframes. The processing time is linear in the number of keyframes. In

our experiments we used $k = 6$ keyframes to estimate the identity parameters resulting in a processing time of a few seconds ($\sim 20s$).

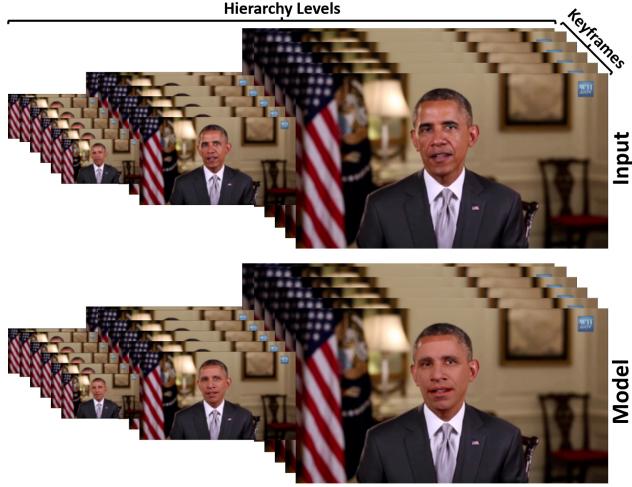


Figure 10: Non-rigid model-based bundling hierarchy: the top row shows the hierarchy of the input video and the second row the overlaid face model.

C. Reenactment Evaluation

In addition to the results in the main paper [30], we compare our method to other existing reenactment pipelines. Fig. 11 shows a self-reenactment scenario (i.e., the source and the target actor is the same person) in comparison to Garrido et al. [13]. Our online approach is able to achieve similar or better quality as the offline approach of Garrido et al. [13]. In Fig. 12, we show a comparisons to Dale et al.



Figure 11: Self-Reenactment comparison to Garrido et al. [13]. The expression of the actress is transferred to a recorded video of herself.

[11] and Garrido et al. [13]. Note that both methods do not preserve the identity of the target actor outside of the self-reenactment scenario. In contrast, our method preserves the identity and alters the expression with respect to the source actor, which enables more plausible results.

We evaluate the presented reenactment method by measuring the photometric error between the input sequence



Figure 12: Comparison to Dale et al. [11] and Garrido et al. [13]. The expression of the left input actor is transferred to the right input actor without changing the person’s identity.

and the self-reenactment of an actor using cross-validation (see Fig. 13). The first 1093 frames of the video are used to retrieve mouth interiors (training data). Thus, self-reenactment of the first half results in a small mean photometric error of 0.33 pixels (0.157px std.Dev.) measured via optical flow. In the second half (frames 1093-2186) of the video, the photometric error increases to a mean value of 0.42 pixels (0.17px std.Dev.).

| |
|---|
| Obama - Celebrating Independence Day https://www.youtube.com/watch?v=d-VaUaTF3.k |
| Donald Trump - Interview: 'I Love China' - Morning Joe - MSNBC https://www.youtube.com/watch?v=Tsh_V3U7EfU |
| Daniel Craig - Interview on the new James Bond Movie https://www.youtube.com/watch?v=8ZbCf7szjXg |
| Putin - New Year's Address to the Nation https://www.youtube.com/watch?v=8_JxKKY7l_Y |
| Arnold Schwarzenegger - Terminator: Genisys https://www.youtube.com/watch?v=p6CJx_ZbaG4 |
| Vocal Coach Ken Taylor - How to Sing Well https://www.youtube.com/watch?v=KDYaACGUt3k |

Table 2: Youtube Video References.



Figure 13: Self-Reenactment / Cross-Validation; from left to right: input frame (ground truth), resulting self-reenactment, and the photometric error.

D. List of Mathematical Symbols

| Symbol | Description |
|--|--|
| \mathcal{K} | feature descriptor |
| \mathcal{L} | Local Binary Pattern |
| t | timestep |
| $D(\mathcal{K}^T, \mathcal{K}^S, t)$ | distance measure |
| $D_p(\mathcal{K}^T, \mathcal{K}_t^S)$ | distance measure in parameter space |
| $D_m(\mathcal{K}^T, \mathcal{K}_t^S)$ | distance measure of facial landmarks |
| $D_a(\mathcal{K}^T, \mathcal{K}_t^S, t)$ | distance measure of appearance |
| $D_l(\mathcal{K}^T, \mathcal{K}_t^S)$ | Chi Squared Distance of LBPs |
| $D_c(\tau, t)$ | cross-correlation between frame τ and t |
| τ_k | k -th previous retrieved frame index |
| $w_c(\mathcal{K}^T, \mathcal{K}_t^S)$ | frame weight |
| $\Phi(\mathcal{P})$ | parameter promoter function |
| $\hat{\Phi}(F(\mathcal{P}))$ | residual promoter function |

| Symbol | Description |
|--|--------------------------------------|
| α, β, δ | shape, albedo, expression parameters |
| $\mathcal{M}_{\text{geo}}, \mathcal{M}_{\text{alb}}$ | parametric face model |
| $a_{\text{id}}, a_{\text{alb}}$ | average shape, albedo |
| $E_{\text{id}}, E_{\text{alb}}, E_{\text{exp}}$ | shape, albedo, expression basis |
| $\sigma_{\text{id}}, \sigma_{\text{alb}}, \sigma_{\text{exp}}$ | std. dev. shape, albedo, expression |
| F | triangle set of the model |
| n | number of vertices |
| v_j | vertex of the face model |
| γ | illumination parameters |
| $\Phi(v)$ | model-to-world transformation |
| R | rotation |
| t | translation |
| $\Pi(v)$ | full perspective projection |
| κ | camera parameters defining $\Pi(v)$ |
| \mathcal{P} | vector of all parameters |
| C_I | input color |
| C_S | synth. color |
| \mathcal{V} | set of valid pixels |
| p | integer pixel location |
| \mathcal{F} | set of detected features |
| f_j | j -th feature point |
| $w_{\text{conf}, j}$ | confidence of j -th feature point |
| $E(\mathcal{P})$ | energy function |
| $E_{\text{col}}(\mathcal{P})$ | photo-consistency term |
| $E_{\text{lan}}(\mathcal{P})$ | feature alignment term |
| $E_{\text{reg}}(\mathcal{P})$ | statistical regularization |
| $w_{\text{col}}, w_{\text{lan}}, w_{\text{reg}}$ | energy term weights |
| $r(\mathcal{P})$ | a general residual vector |
| $J(\mathcal{P})$ | jacobian matrix |
| $F(\mathcal{P})$ | residual vector |
| A_i | deformation gradient of triangle i |
| \hat{v}_i | deformed vertex |
| V | triangle spanning vectors |
| \hat{V} | deformed triangle spanning vectors |
| $E(\delta^T)$ | deformation transfer energy |
| A | system matrix of the transfer energy |
| b | rhs of the transfer energy |