Supplemental Material for "Face2Face: Real-time Face Capture and Reenactment of RGB Videos"

Justus Thies[1] Michael Zollhöfer[2] Marc Stamminger[1] Christian Theobalt[2] Matthias Nießner[3]
[1]University of Erlangen-Nuremberg [2]Max-Planck-Institute for Informatics [3]Stanford University

In this document, we provide supplementary information to the method by Thies et al. [4]. More specifically, we include additional detail about our optimization framework (see Section 1 and 2), and we show further comparisons against other methods (see Section 3). We also evaluate the reconstruction error in a self-reenactment scenario. In Section 4, a list of used mathematical symbols is given. The used video sources are listed in Table 1.

1. Optimization Framework

Our Gauss-Newton optimization framework is based on the work of Thies et al. [3]. Our aim is to include every visible pixel $\boldsymbol{p} \in \mathcal{V}$ 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.

2. 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. 1. In order to leverage

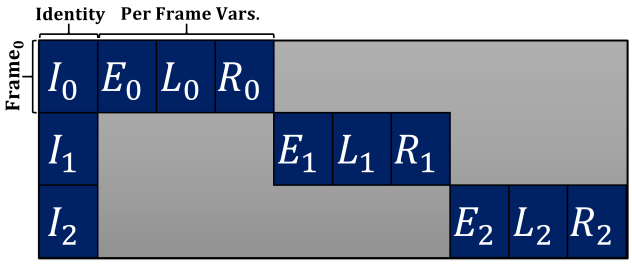

Figure 1: Non-zero structure of the Jacobian matrix of our non-rigid model-based bundling approach for three keyframes. Where I_i, E_i, L_i, R_i are the i-th per frame Jacobian matrices of the identity, expression, illumination, and rigid pose parameters.

the sparse structure of the Jacobian, we adopt the Gauss-Newton framework as follows: we modify the computation of the gradient $J^T(\boldsymbol{\mathcal{P}}) \cdot F(\boldsymbol{\mathcal{P}})$ and the matrix vector product $J^T(\boldsymbol{\mathcal{P}}) \cdot J(\boldsymbol{\mathcal{P}}) \cdot \boldsymbol{x}$ 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). $\mathcal{P}_{global}$ 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

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

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 [5, 3], we split up the computation of $J^T(\boldsymbol{\mathcal{P}}) \cdot J(\boldsymbol{\mathcal{P}}) \cdot \boldsymbol{x}$ into two successive matrix vector products, where the second multiplication is analogue to the computation of the gradient. The first multiplication is as follows:

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

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. 2). 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 ($\sim 20s$).

3. Reenactment Evaluation

In addition to the results in the main paper [4], we compare our method to other existing reenactment pipelines. Fig. 3 shows a self-reenactment scenario (i.e., the source and the target actor is the same person) in comparison to Garrido et al. [2]. Our online approach is able to achieve similar or better quality as the offline approach of Garrido et al. [2]. In Fig. 4, we show a comparisons to Dale et al. [1] and Garrido et al. [2]. 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 and the self-reenactment of an actor using cross-validation (see Fig. 5). 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.).

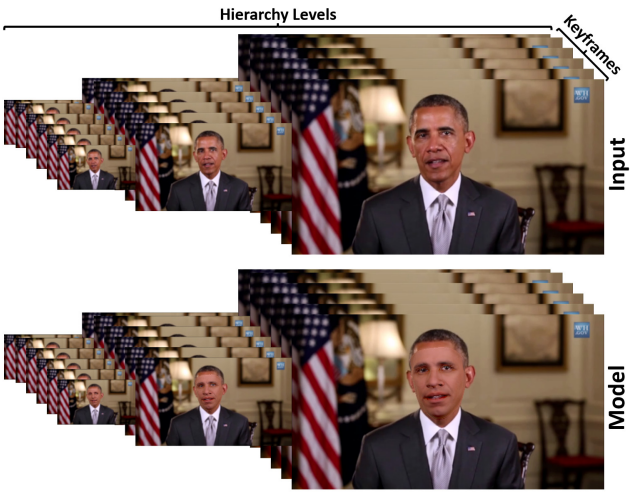

Figure 2: Non-rigid model-based bundling hierarchy: the top row shows the hierarchy of the input video and the second row the overlaid face model.

Figure 3: Self-Reenactment comparison to Garrido et al. [2]. The expression of the actress is transferred to a recorded video of herself.

Figure 4: Comparison to Dale et al. [1] and Garrido et al. [2]. The expression of the left input actor is transferred to the right input actor without changing the person's identity.

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_JxKKY7I_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 1: Youtube Video References.

Figure 5: Self-Reenactment / Cross-Validation; from left to right: input frame (ground truth), resulting self-reenactment, and the photometric error.

4. 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
$D_p(\mathcal{K}^T, \mathcal{K}^S_t)$	distance measure in parameter space
$D_m(\mathcal{K}^T, \mathcal{K}^S_t)$	distance measure of facial landmarks
$D_a(\mathcal{K}^T, \mathcal{K}^S_t, t)$	distance measure of appearance
$D_l(\mathcal{K}^T, \mathcal{K}^S_t)$	Chi Squared Distance of LBPs
$D_c(\tau, t)$	cross-correlation between frame τ and t
τ_k	k-th previous retrieved frame index
$w_c(\mathcal{K}^T, \mathcal{K}^S_t)$	frame weight
$\Phi(\boldsymbol{\mathcal{P}})$	parameter promoter function
$\hat{\Phi}(F(\boldsymbol{\mathcal{P}}))$	residual promoter function

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

References

[1] K. Dale, K. Sunkavalli, M. K. Johnson, D. Vlasic, W. Matusik, and H. Pfister. Video face replacement. *ACM TOG*, 30(6):130, 2011. 2

[2] P. Garrido, L. Valgaerts, O. Rehmsen, T. Thormaehlen, P. Perez, and C. Theobalt. Automatic face reenactment. In *Proc. CVPR*, 2014. 2

[3] 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. 1, 2

[4] 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. 1, 2

[5] 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. 2