RGB Face Reconstruction - Project Report

MUSTAFA IŞIK, Technical University of Munich, Germany
PATRICK RADNER, Technical University of Munich, Germany
WOJCIECH ZIELONKA, Technical University of Munich, Germany

This is the final report for "RGB Face Reconstruction" project for "Master Praktikum: 3D Scanning & Spatial Learning". Our project aims to reconstruct faces extracted from from RGB video sequences using a parametric face model.

ACM Reference Format:
Mustafa Işık, Patrick Radner, and Wojciech Zielonka. 2020. RGB Face Reconstruction - Project Report . *ACM Trans. Graph.* 1, 1, Article 1 (February 2020), 4 pages. https://doi.org/0

1 INTRODUCTION

Real-time markerless facial tracking has been well studied in the past couple of years. One of the papers on this track was Face2Face [Thies et al. 2016] which became very popular as it presented a way to transfer facial expressions to a target face in real-time. To do that, they fit the parametric face model to target videos offline to get high quality tracking and reconstruction. Then, they also process low-resolution webcam frames in real-time and re-enact the target face.

In our project, we managed to fit the parametric face model offline, which is the first part of Face2Face as explained above. In order to do that, we make use of analysis-by-synthesis approach where we optimize for face parameters to make it look like target face. We run all the computational parts of our pipeline on GPU using CUDA and OpenGL. Our energy function consists of a sparse landmark term, a dense photometric term and a regularizer term. This energy function is minimized using iteratively reweighted least squares method (IRLS). Each Gauss-Newton update is solved using preconditioned conjugate gradients method (PCG).

2 PARAMETRIC FACE MODEL

Matching a 3D surface to a given face image is a highly ill-posed problem. One of the methods to deal with it is using parametric models [Blanz and Vetter 1999], which not only allow to generate new faces, but also constrain the solution to stay within the vector space spanned by the database. Those models are usually crafted using high quality scans of real faces or different body parts such as eyes, teeth etc., depending on a given reconstruction problem. The database used in this project was taken from [Blanz and Vetter 1999], where the authors used 200 scanned heads of young adults to create a statistical face model by using Principal Component Analysis[1]. The final parametric face model looks as follows:

$$\begin{aligned} M_{geo}(\boldsymbol{\alpha}, \boldsymbol{\delta}) &= \boldsymbol{a}_{id} + E_{id} \cdot \boldsymbol{\alpha} + E_{exp} \cdot \boldsymbol{\delta}, \\ M_{alb}(\boldsymbol{\beta}) &= \boldsymbol{a}_{alb} + E_{alb} \cdot \boldsymbol{\beta} \end{aligned} \tag{1}$$

Where $\boldsymbol{a}_{id} \in \mathbb{R}^{3n}$ and $\boldsymbol{a}_{alb} \in \mathbb{R}^{3n}$ describe the average face shape and albedo. The Eigen basis are, $\boldsymbol{E}_{id} \in \mathbb{R}^{3n \times 80}$ for shape, $\boldsymbol{E}_{exp} \in \mathbb{R}^{3n \times 76}$ for expression and $\boldsymbol{E}_{alb} \in \mathbb{R}^{3n \times 80}$ for albedo. The coefficients $(\boldsymbol{\alpha}, \boldsymbol{\delta}, \boldsymbol{\beta})$ describe the deviation from the average model and can be used to generate new faces or optimized to fit the model to a given target actor. The final face generator configuration vector is defined as follows: $\boldsymbol{\mathcal{P}} = (\boldsymbol{\alpha}, \boldsymbol{\delta}, \boldsymbol{\beta}, \Phi, \gamma, \kappa)$, where Φ is pose matrix with rotation and translation, γ is illumination parameters and virtual camera field of view is defined as κ.

3 ENERGY FORMULATION

To find the best vector $\mathcal{P}$ for the face model, robust variational optimization is used. The goal is to minimize the highly non-linear objective function, composed of the following components:

$$E(\boldsymbol{\mathcal{P}}) = w_{col}E_{col}(\boldsymbol{\mathcal{P}}) + w_{lan}E_{lan}(\boldsymbol{\mathcal{P}}) + w_{reg}E_{reg}(\boldsymbol{\mathcal{P}}) \tag{2}$$

The first two terms $w_{col}E_{col}(\boldsymbol{\mathcal{P}})$ and $w_{lan}E_{lan}(\boldsymbol{\mathcal{P}})$ measure the distance between the synthetic image created by the program and the target video image considering photo-consistency and facial feature alignment, respectively. Additionally, a statistical regularizer E_{reg} is incorporated into the energy function. This is based on the assumption, that faces in the database are normally distributed and the solution should not deviate too far from the mean face.

$$E_{reg}(\boldsymbol{\mathcal{P}}) = \sum_{i=1}^{80} \left[\left(\frac{\boldsymbol{\alpha}_i}{\sigma_{id,i}} \right)^2 + \left(\frac{\boldsymbol{\beta}_i}{\sigma_{alb,i}} \right)^2 \right] + \sum_{i=1}^{76} \left(\frac{\boldsymbol{\delta}_i}{\sigma_{exp,i}} \right)^2 \tag{3}$$

This commonly used regularization strategy prevents degenerations of the facial geometry and color, and guides the optimization strategy out of local minima [Thies et al. 2016].

3.1 Sparse Features

In order to provide a good initialization and stabilize our reconstruction, sparse features alignment was incorporated to the energy function. Sparse features are salient, easily detectable landmarks of a face — for instance corners of eyes or lips. To find those landmarks, a tracker based on machine learning was used [Kazemi and Sullivan 2014]. The energy term is measured in screen space and the formulation is as follows:

$$E_{lan}(\boldsymbol{\mathcal{P}}) = \frac{1}{|F|} \sum_{f_j \in F} ||f_j - \pi(\Phi(\boldsymbol{v}_j))||_2^2 \tag{4}$$

Authors' addresses: Mustafa Işık, Technical University of Munich, Germany, XXX; Patrick Radner, Technical University of Munich, Germany, patrick.radner@tum.de; Wojciech Zielonka, Technical University of Munich, Germany, wojciech.zielonka@tum.de.

[1]Principal Component Analysis is a dimensionality reduction technique that can be used to reduce a large set of data points to a small set that still contains most of the information in the large set.

Where $\pi(\Phi(\boldsymbol{v}_j))$ describes the vertex transformation from local coordinate space to NDC[2]. The function π applies the projection matrix, and performs the perspective division and the matrix Φ describes the rigid pose matrix. This term helps to avoid local minima in the complex energy landscape generated by the photo-consistency error [Thies et al. 2016].

In order to minimize the energy function, the sparse term has to be derived w.r.t our parameter vector. Thus, we should obtain the Jacobian matrix for expression, shape, pose and field of view for an OpenGL virtual camera. Since the projection is composed of many functions, the chain rule has to be used. The derivation w.r.t α, which is responsible for shape morphing, looks as follows:

$$\frac{\partial E_{lan}}{\partial \alpha} = \frac{\partial \pi(\Phi(M_{geo}(\boldsymbol{\alpha}, \boldsymbol{\delta})))}{\partial \Phi(M_{geo}(\boldsymbol{\alpha}, \boldsymbol{\delta}))} \frac{\partial \Phi(M_{geo}(\boldsymbol{\alpha}, \boldsymbol{\delta}))}{\partial M_{geo}(\boldsymbol{\alpha}, \boldsymbol{\delta})} \frac{\partial M_{geo}(\boldsymbol{\alpha}, \boldsymbol{\delta})}{\partial \alpha} \tag{5}$$

The formula for the expression vector $\boldsymbol{\delta}$ works analogous. Moreover, the pose matrix Φ has to be also derived in this chain, therefore, we can branch out in the computation graph from the second step and obtain:

$$\frac{\partial E_{lan}}{\partial \Phi} = \frac{\partial \pi(\Phi(\boldsymbol{R}, \boldsymbol{T}))}{\partial \Phi(\boldsymbol{R}, \boldsymbol{T})} \frac{\partial \Phi(\boldsymbol{R}, \boldsymbol{T})}{\partial \boldsymbol{R}, \boldsymbol{T}} \tag{6}$$

Where $\boldsymbol{R}, \boldsymbol{T}$ describe rotation and translation. The virtual camera field of view parameter (FoV) is derived from the projection matrix. We define perspective division as ψ and the projection matrix from world space to screen space as Π. In this way π is decomposed into $\pi = \psi(\Pi(FoV))$ and the derivation can be denoted as:

$$\frac{\partial E_{lan}}{\partial FoV} = \frac{\partial \pi}{\partial FoV} = \frac{\partial \psi(\Pi(FoV))}{\partial \Pi} \frac{\partial \Pi}{\partial FoV} \tag{7}$$

3.2 Dense Features

Dense features are the most important term in the optimization process, as they allow us to reconstruct color, light and fine geometric details. They are based on a per-pixel photometric alignment error.

$$E_{col}(\boldsymbol{\mathcal{P}}) = \frac{1}{|V|} \sum_{\boldsymbol{p} \in V} ||C_S(\boldsymbol{p}) - C_I(\boldsymbol{p})||_2 \tag{8}$$

Here C_S is the synthesized image, C_I is the input RGB image and $\boldsymbol{p} \in V$ denotes all pixels covered by our rendered face model. To reduce the influence of outliers, a $L_{2,1}$ norm is used instead of a least-squares formulation. The computation graph starts with the two composed functions:

$$\begin{matrix} \frac{\partial C_S}{\partial \boldsymbol{\mathcal{P}}} \\ -\frac{\partial C_I}{\partial \boldsymbol{\mathcal{P}}} \end{matrix} \tag{9}$$

The dense term contributes with all parameter derivatives of the vector $\boldsymbol{\mathcal{P}} = (\boldsymbol{\alpha}, \boldsymbol{\delta}, \boldsymbol{\beta}, \Phi, \gamma, \kappa)$ to the Jacobian matrix, whereas the sparse term was not derived w.r.t the illumination model parameters and albedo. Ultimately, for every pixel the Jacobian entry w.r.t vector $\boldsymbol{\mathcal{P}}$ is calculated. The photo-consistency term basically derives the entire rendering pipeline, since it is defined on the pixel level. Therefore, it is necessary to take a closer look into pixel formation during rasterization-based rendering.

[2]Normalized Device Coordinates is a space obtained by applying perspective projection and it is usually represented by a unit cube which ranges between -1 and 1.

4 FORWARD RENDERING

The parametric face model is rendered using standard, rasterization-based rendering. As graphics framework, the OpenGL API was chosen. The approach of analysis-by-synthesis expects to gather some intermediate information from the rendering pipeline. For each pixel in the final image, barycentric coordinates and vertex identifiers of the corresponding triangle have to be stored, because the position of a vertex is a function of the parametric face model. This is necessary, as during rasterization process the pixels are computed by barycentric interpolation of the triangles per-vertex attributes, such as color and normals. Barycentric coordinates reflect the center of mass which is our pixel (P) and are calculated as a ratio of the areas of PBC, PCA and PAB to the area of the entire triangle ABC. Therefore, all those steps have to be included in the chain rule derivations. In order to store this per pixel information, additional render targets are used. For each optimization step 3 textures are rendered, containing the aforementioned parameters and the actual color image. They are then used in the analysis/differentiable rendering step to compute the Jacobian matrix and residuals.

We would like to emphasize a point about barycentric interpolation which is done automatically in the OpenGL pipeline. Since interpolation in 3D world space and in 2D image space is not the same, OpenGL applies perspectively correct linear interpolation. This makes interpolation in the image space effectively the same as interpolation in 3D world space. Thus, one should be very careful to account for this perspective correction, which is implicitly employed in the pipeline, when going backwards to calculate the jacobian matrices.

4.1 Differentiable Rendering

Differentiable rendering allows us to compute the gradients of our final pixels w.r.t. the input vertices. In particular gradients have to flow through the fragment shader, the rasterization stage (barycentric interpolation and perspective projection) and the vertex shader. The model is a vector-valued function, thus, the change is reflected by a Jacobian matrix. In our case the vertex shader applies the pose matrix Φ and the fragment shader computes Equation 10.

$$color = light(normal) * albedo \tag{10}$$

Where *light* is approximated by the first 3 bands of spherical harmonics[3] and calculated based on the normal for a given fragment. To compute the Jacobian matrix of the model, *color* has to be derived w.r.t all model parameters: albedo, shape, expression, FoV, pose, and light. This task requires derivation of the whole shader pipeline, starting with fragment shader.

In summary, the vertex position is a function of the parametric model and the normal used for shading is a function of the corresponding face, which is again described by 3 vertices.[4]

5 OPTIMISATION

Once all gradients of final image w.r.t. the model parameters are calculated, meaning, the Jacobian matrix is built, the loss function is

[3]Light transport involves many quantities defined over the spherical and hemispherical domains, making spherical harmonics a natural basis for representing these functions.
[4]Given face defined as tuple of 3 vertices (a, b, c) then $normal = (b - a) \times (c - a)$.

minimized to fit the model. Therefore, this section elaborates details on how the energy function is minimized.

5.1 Iteratively Reweighted Least squares (IRLS)

Iteratively Reweighted Least Squares is used to optimize objective functions that take the form of a p-norm[5]. This is done by using Equation 11, which transforms the problem into a weighted least squares problem at each iteration, where the weight is defined as $w = \|r(\boldsymbol{\mathcal{P}}_{old})\|^{p-2}$. Finally, an optimization update step is solved using the Gauss-Newton (GN) method, Equation 12. During development two different p-norms were tested, namely, 1-norm and 2-norm. The 1-norm turned out to be more robust and delivered better visual results.

$$\|r(\boldsymbol{\mathcal{P}})\|_p = \|r(\boldsymbol{\mathcal{P}}_{old})\|^{p-2} \cdot \|r(\boldsymbol{\mathcal{P}})\|_2^2 \tag{11}$$

$$\begin{aligned} \Delta\boldsymbol{\mathcal{P}}_k &= -(\boldsymbol{J}^T\boldsymbol{W}\boldsymbol{J})^{-1}\boldsymbol{J}^T\boldsymbol{W}\boldsymbol{f} \\ \boldsymbol{\mathcal{P}}_{k+1} &= \boldsymbol{\mathcal{P}}_k + \Delta\boldsymbol{\mathcal{P}}_k \end{aligned} \tag{12}$$

In the equations above $\boldsymbol{J}$ denotes Jacobian matrix, $\boldsymbol{f}$ is the residual vector, $\boldsymbol{W}$ is the weight matrix and $\boldsymbol{\mathcal{P}}$ is the face model parameters vector.

Additionally, we use a coarse to fine approach to optimize our energy function. These approaches are known for helping to avoid local minima. They also allow us to speed up computation, by reducing the number of equations to solve per iteration. The OpenCV library was used to create a Gaussian image pyramid[6]. The final pyramid consists of three levels. We run 25 IRLS iterations on the coarsest level, 5 on the middle one and 1 iteration on the finest level.

5.2 Preconditioned Conjugate Gradients (PCG)

The standard form of the Gauss-Newton update for a weighted least squares problem ,shown in (Equation 12), involves the inversion of the matrix $\boldsymbol{J}^T\boldsymbol{J}$. In addition to this being numerically unstable, the runtime complexity of matrix inversion lies in $O(n^3)$ for Gauss–Jordan elimination method. To avoid this, Equation 12 can be reformulated as $\boldsymbol{J}^T\boldsymbol{J}\boldsymbol{\delta} = -\boldsymbol{J}^T\boldsymbol{f}$. This can then be solved using, for example Conjugate Gradients. Furthermore the system can be preconditioned [Kaasschieter 1988] to allow for faster convergence of this iterative method. The preconditioned system is defined as follows:

$$\boldsymbol{M}^{-1}\boldsymbol{J}^T\boldsymbol{J}\boldsymbol{\delta} = -\boldsymbol{M}^{-1}\boldsymbol{J}^T\boldsymbol{f} \tag{13}$$

The symmetric positive definite matrix $\boldsymbol{M}$ must be chosen in such a way that the system $\boldsymbol{M}\boldsymbol{z} = \boldsymbol{r}$ can be solved with less computational work than the original system $\boldsymbol{J}^T\boldsymbol{J}\boldsymbol{\delta} = -\boldsymbol{J}^T\boldsymbol{f}$ for every vector $\boldsymbol{r}$ on the right-hand side. For this purpose a Jacobi preconditioner given as $\boldsymbol{M} = diag(\boldsymbol{J}^T\boldsymbol{J})^{-1}$ was selected, which fulfills the requirement, since the inverse of a diagonal matrix $\boldsymbol{D} = diag(\boldsymbol{J}^T\boldsymbol{J})$ is obtained by replacing each element in the diagonal with its reciprocal.

Figure 1 shows a general form of the PCG algorithm, where the matrix $\boldsymbol{M}$ is the preconditioner. $\boldsymbol{A}$ is the system matrix, in our case this is equal to $\boldsymbol{J}^T\boldsymbol{J}$, and $\boldsymbol{x}$ corresponds to our update vector $\boldsymbol{\delta}$. Note, that we do not compute $\boldsymbol{J}^T\boldsymbol{J}$ explicitly, but instead apply $\boldsymbol{J}$ and $\boldsymbol{J}^T$ consecutively to our solution vector $\boldsymbol{x}$ at each iteration. Finally, for each Gauss-Newton step, 5 iterations of PCG are computed, before applying the update.

$i \Leftarrow 0$
$r \Leftarrow b - Ax$
$d \Leftarrow M^{-1}r$
$\delta_{new} \Leftarrow r^T d$
$\delta_0 \Leftarrow \delta_{new}$
While $i < i_{max}$ and $\delta_{new} > \varepsilon^2\delta_0$ do
    $q \Leftarrow Ad$
    $\alpha \Leftarrow \frac{\delta_{new}}{d^T q}$
    $x \Leftarrow x + \alpha d$
    If i is divisible by 50
        $r \Leftarrow b - Ax$
    else
        $r \Leftarrow r - \alpha q$
    $s \Leftarrow M^{-1}r$
    $\delta_{old} \Leftarrow \delta_{new}$
    $\delta_{new} \Leftarrow r^T s$
    $\beta \Leftarrow \frac{\delta_{new}}{\delta_{old}}$
    $d \Leftarrow s + \beta d$
    $i \Leftarrow i + 1$

Fig. 1. Preconditioned conjugate gradients algorithm [Shewchuk 1994].

6 RESULTS

In this section some of the final results are presented. For more results and videos please refer to our presentation. The best performance of our model is achieved for scenes where the head movement is smooth and there are no face occlusions such as glasses or facial hair. Therefore, a good source of target videos is politicians giving interviews or public speeches.

Fig. 2. Showcase of our face reconstruction implementation based on Hilary Clinton.

Figure 2 shows 3 different frames of Hilary Clinton during a public speech. The face models quite nicely reflects her physique, but it is still easy to spot some subtle difference. The problem is caused by the age of the target which is not reflected in the model basis (only young adults faces).

[5]Let $x = (x_i) \in \mathbb{R}^n$ and $p \geqslant 1$, the p-norm is defined as $||x_p|| := \left(\sum_{i=1}^n |x_i|^p\right)^{1/p}$
[6]Subsequent images are weighted down using a Gaussian average (Gaussian blur) and scaled down.

Fig. 3. Showcase of our face reconstruction implementation based on Justin Trudeau.

One of the best results is presented in Figure 3, both, in the case of quality and expression tracking. Because, the target actor fits in the range of the model basis, the optimization is able to almost perfectly find the vector $\mathcal{P}$ for the generator. This example shows, how important it is, that the target lies within the space described by the statistical model. If the model database were to contain more diverse faces (not only young adults), the greater variety of target actors could be reconstructed.

6.1 Failure Cases

Figure 4 and Figure 5 show cases in which the quality of our reconstruction severely suffers, due to features, that cannot be explained by the model. The presence of facial hair poses a significant difficulty for facial reconstruction. Glasses also cause problems, as they occlude parts of the face and distort the region around the eyes. Furthermore, since our model is only based on young people, some features of older people cannot be explained and end up degrading the overall reconstruction quality.

Fig. 4. Facial hair and glasses cause the model to fail.

Fig. 5. The model cannot explain facial features that occur with advanced age.

7 CONCLUSION

The approach used in this project is based on non-linear optimization techniques and the analysis-by-synthesis concept. It is an old method, already used by Blanz and Vetter more than 20 years ago, however, it still produces very plausible results by today's standards. Nowadays, more sophisticated methods can be used to create digital faces. Usually, those methods are based on neural rendering and generative adversarial networks [Thies et al. 2019]. A model for general face reconstruction requires a database much bigger and more diverse than 200 scans of young people. For instance, faces of old people are impossible to capture because they simply don't exist in our face generator basis space. Another problem is facial hair, which also introduces reconstruction problems and requires a different approach. Therefore, this method works very good for faces of young people.

7.1 Contributions

Wojciech: sparse terms, OpenGL, CUDA
Mustafa: dense terms, OpenGL, CUDA
Patrick: optimization, OpenGL, CUDA

8 CODE REPOSITORY

Sources code can be reached from github.com/isikmustafa/face-tracking.

REFERENCES

Volker Blanz and Thomas Vetter. 1999. A Morphable Model for the Synthesis of 3D Faces. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '99)*. ACM Press/Addison-Wesley Publishing Co., USA, 187–194. https://doi.org/10.1145/311535.311556

E.F. Kaasschieter. 1988. Preconditioned conjugate gradients for solving singular systems. *J. Comput. Appl. Math.* 24, 1 (1988), 265 – 275. https://doi.org/10.1016/0377-0427(88)90358-5

Vahid Kazemi and Josephine Sullivan. 2014. One Millisecond Face Alignment with an Ensemble of Regression Trees. (06 2014). https://doi.org/10.13140/2.1.1212.2243

Jonathan R Shewchuk. 1994. An Introduction to the Conjugate Gradient Method Without the Agonizing Pain. (1994).

Justus Thies, Michael Zollhöfer, and Matthias Nießner. 2019. Deferred Neural Rendering: Image Synthesis using Neural Textures. *ACM Transactions on Graphics 2019 (TOG)* (2019).

J. Thies, M. Zollhöfer, M. Stamminger, C. Theobalt, and M. Nießner. 2016. Face2Face: Real-Time Face Capture and Reenactment of RGB Videos. (June 2016), 2387–2395. https://doi.org/10.1109/CVPR.2016.262