Performance-driven Facial Animation

Garoe Dorta-Perez, Ieva Kazlauskaite, Richard Shaw

CM50247 - Visual Effects

Unit Leader: Dr Darren Cosker

University of Bath

May 2015

Signature of Author . . . .

Garoe Dorta-Perez, Ieva Kazlauskaite, Richard Shaw

Chapter 1

Introduction

"*Birds are the most accomplished aeronauts the world has ever seen. They fly high and low, at great speed, and very slowly. And always with extraordinary precision and control.*" [**?**]

A talented painter or sculptor is able to imagine and reproduce the subtle details of a human face. Hours of training and endless manual adjustments are required before an arbitrary shape resembles a facial expression. Furthermore, a human eye is trained to notice subtle changes in the expression of another human being, which also applies to animated humans, thus even the smallest discrepancies in an animated model are easily detected. Though artists are able to produce good quality and appealing results, the limitations in budget and time so prominent in the entertainment industry motivate the development of more automated, faster and cheaper models.

The naive approach that allows for faster performance is the keyframe animation; though it is still based on manual input, keyframe animation requires fewer frames as the intermediate expressions are interpolated over. However, this approach is still very laborious and time consuming. With the emergence of software and systems for motion capture, development of mathods for performance-driven animation has received significant attention from both the industry as well as the academia.

Chapter 2

Previous Work

2.1 Data Capture

2.2 Sparse Reconstruction

2.3 Facial Animation

The first attempts at using performance-driven facial animation in academia date back to late 20^{th} century while in industry was first used for guidance rather than production, i.e. the facial motion of Gollum in the *Lord of the Rings* was based on the motion capture data of the performance of the role actor Andy Serkis [PSS02]. Another notable example is the pipeline used in the making of *the Polar Express*; the motion capture data was used to construct a multilayer facial rig that was used by an artist to create the final animation [Ben05]. Though many other examples exist, often the exact technology and methods used in production are not disclosed.

In the last two decades, nearly two hundred academic papers that include words *face*, *animation* and *motion capture* were published in computer graphics and vision journals [Sco]. The increase in the quality of the results is explained by the advancements in both the capturing technology and the computational methods. We shall concentrate on the latter; for a brief discussion of the capturing methods see Sec. 3.1.

The work of Guenter et al. provides the first detailed and unified system for capturing and reconstructing facial performance [GGW*98]. Their work is mostly focused on motion tracking and production of labelled three-dimensional motion of the dots on the actors face. Once a sequence of moving dots is acquired, the authors scan the actors face to get a polygonal mesh that corresponds to the geometry of the face. The motion of the sparse dots is described in terms of offsets from the neutral position. Then the vertices in the mesh are moved by calculating a linear combination of the offsets of the nearest dots, i.e. in each frame a weighted sum of the change in the dot position is calculated and added to the neutral position of the given vertex. The weights are chosen to be zero everywhere except in the one-ring neighbourhood where the weights depend on the distance of the vertex to the dots; the weights must sum to one. An additional stationary ring of dots is added on the edge of the face to ensure there is no motion outside the face. The algorithm suffers from noise introduced during in the tracking, the reconstruction and the three-dimensional scanning. Moreover, the choice of the nearest dots for each vertex in the mesh is not trivial due to the irregular distribution of the tracked dots. The method that assigns the blends has a number of manually adjusted parameters and does not guarantee to find the best set of reference dots. Additionally, the areas around the eyes and lips require a special treatment; the dots above and below the problematic areas are marked and are not allowed to be blended. The authors point out that a number of artefacts are visible in the resulting animation; some of these flaws attributed to the poor quality of the facial scan, and the fact that the reconstruction method is not robust to jitter and incorrect placement of the tracked data.

At around the same time Pinghin et al. proposed a method for synthesizing facial expressions form photographs [PHL*98]. The authors concentrate their attention on the capture and reconstruction part of the process. First, a three-dimensional model of the face pulling a number of different expressions is constructed, then the animation is created by morphing these meshes and producing the intermediate animation. The meshes are topologically consistent and the features are marked manually resulting in matching feature sets. Then linear interpolation between corresponding vertices in the different meshes is used to get a smooth morphing. Compared to the method proposed by Guenter et al., the approach of Pinghin et al. puts more emphasis on the capture, reconstruction, and creation of good quality meshes for different expressions resulting in an almost trivial animation production process.

Noh and Neumann addressed the problem of having to create a new model for a new animation by exploiting an existing database of high quality animations [NN01]. Given a new target model, dense correspondence between the source model from the database and the target is estimated using the a set of manually selected matching points. The volume morphing is performed using a weighted linear combination of Radial Basis Functions (RBF). However, some manual input is required when matching the features; for this authors use a set of heuristic rules. Then a cylindrical projection is used to transfer the vertices from the source onto the target model, thus the source vertices are embedded in the target surface. The animation is created by applying the adjusted target motion vector onto the source. The adjustments involve scaling, and change of direction according to the curvature of the source surface. Moreover, if the two models have very different geometries, then small neighbourhoods are used to determine the local changes that were imposed during the morphing of the source surface. As in most facial models, special attention is paid to the area around the mouth; the edges of the lips in the two models are aligned, and the motion transformations take into account the motion of both lips to ensure that the duplicated vertices move consistently. The presented approach has a number of limitations, the model requires manual matching, the eyelids, teeth and the tongue are not incorporated in the model, and it is only able to reproduce the motion of the source model and cannot produce novel movements.

Building on previous work on synthesis of facial expressions, by Joshi et al. address the problem of automating the creation of blend shapes [JTDP03]. The proposed method segments the face into characteristic parts that can then be modified independently to make a desired expression thus simplifying the process of creating new blend shapes. The authors argue that one of the advantages of controlling small regions is the increased number of expressions that may be produced using the blend shapes. Each frame in the keyframe animation is produced by calculating the linear combination of blend shapes in each region. A linear elasticity model is used to deform the surfaces. A related method, proposed by Pyun et al. generates a new facial expression by combining emotional and verbal key expressions [PKC*03]. The main limitation of this model is that a set of key expressions have to be designed by an artist.

In some situations the sets of blend shapes is only available for the target model. Choe and Ko propose a method that constructs a set of source key shapes given the target key shapes [CK05]. First, a mapping between the source and the target coordinate systems is constructed. Then the method iteratively refines the set of shapes using the captured source data. The authors build on their previous work, and use a muscle actuation basis as opposed to the surface-based key shapes. The elements in the actuation basis are linearly independent and they span the corresponding space, forming a complete basis for the facial expressions [CLK01]. However, the actuation basis is much harder to construct in comparison to the surface-based set of shapes and the relation between the motion of facial muscles and the resulting motion of the tissues is not straight-forward.

Vlasic et al. proposed an novel approach to facial animation that is based on a multilinear model [VBPP05]. The work offers an alternative approach to blend shape models, that involves creating a large dataset that encodes various features, for instance the identities, expressions and location of vertices. Each dimension of the data tensor corresponds to a unique feature, and due to the independence of modes, each feature may be varied without affecting the others. Such model can be extended to include an arbitrary number of features, and may be used for motion retargeting or actor replacement, when a database of three-dimensional scans is available. Moreover, the probabilistic interpretation of the model is related to probabilistic principal component analysis, and is able to deal with missing data.

A number of dimensionality reduction techniques have been used to produce facial animation; for example Blanz and Vetter apply principal component analysis (PCA) to create a controllable low dimensional model [BV99]. Deng et al. use PCA to find a correspondence between the motion capture data are the manually tuned blend shape models [DCFN06]. An extension to nonlinear dimensionality reduction techniques was presented by Wang et al. [WHL*04]. The proposed method separates the time-varying facial expression and the individual style associated with the performer. Then the locally linear embedding (LLE) framework is used to find a nonlinear mapping from the high dimensional space onto a low dimensional manifold. The LLE method is based on an assumption that small neighbourhoods around each data point may be treated as linear patches. Using the resulting generative model, new expressions or an animation may be produced by sampling from the low dimensional manifold. Dimensionality reduction is used when no blend shape model is available, or when different features of the data need to be extracted.

Most of the previously described methods are unable to capture and reproduce the fine details on the actor's face; Bicket et al. introduced a method that is able to create detailed wrinkles on the target model [BBA*07]. The approach is based on decomposition of the model into coarse features from the motion capture sequence, and fine features from the accompanying video. The novel part of their method relies on capturing, analysing and reproducing the wrinkled surface. In the tracking stage, uniform B-Splines are fitted to each fold of the skin. Then the shape of each fold is estimated by exploiting the self-shadowing effect, and finding the gradient that describes the shape of each fold. Moreover, the coupling behaviour is modelled separately. New wrinkles are synthesised by minimising the associated non-linear shell energy; the geometrical model, proposed by Grinspun et al. estimates the local curvature of a surface, and is able to capture the behaviour of thin flexible structures [GHDS03]. The facial model with detailed wrinkles relies on the quality of the captured data; in particular, the areas that are prone to small-scale deformations have to be painted in a way that minimises diffuse reflection.

One of the most successful attempts to produce a photo-realistic facial animation is known as the Digital Emily project [ARL*09]. The proposed method involves collecting high resolution data, building a detailed facial rig of the actor's face, producing an animation from the video data, and reproducing the high quality results. The blend shape model was created using approximately 30 three-dimensional scans; since each scan contained more than one discrete shape, a splitting algorithm had to be used to obtain larger set of controllable shapes. Additional subtle effects, such as the motion of the deformation of the eyelid when the eyes are closed were included to increase the realism of the results. The animation is produced using a number of example poses, and generating predictions for the required pose of the digital model. The entire production process is very time consuming and requires manual input from skilful artists and animators. The authors do not provide a detailed description of the method but their controllable facial rig is publicly available.

One recent development was proposed by Bouaziz et al., who develop a method for real-time facial animation [BWP13]. The new method does not rely on the construction of a three-dimensional expression model associated with the actor prior to the face animation stage. The presented method uses a consumer-level device that captures both the intensity and the depth; previous research by Baltrušaitis et al. indicates that the use of multimodal data improves the quality of feature tracking [BRM12]. To achieve real-time results, Bouaziz et al. use a template blend shape model that is constantly updated to better match the face of the performer. This update includes two steps. Firstly, PCA is used to construct a low dimensional representation of a large set of different facial meshes, then any new face is constructed by merging the average face with a linear combination of the orthonormal basis vectors produced by PCA. Secondly, surface deformation field is used to further refinement the blend shape model. Then the tracking, retargeting and production of an animation is performed in parallel with the optimisation algorithm that aims to personalise the blend shape model.

Concurrently, Li et al. combined blend shape models with facial tracking to animate a target character face [LYYB13]. The authors employ per frame correctives to achieve run time shape correction. The data is captured using RGB-D (intensity and depth) cameras. The initial state of this method consists of capturing a three-dimensional model of an actor's face, and PCA and a large database of captured facial data is used to construct a generic face. Then a blend shape model is constructed using the deformation transfer algorithm, proposed by Sumner and Popović [SP04]. Then the motion is produced using blend shapes, and it is refined using an orthogonal adaptive basis. This basis is constructed using PCA, and it contains the initial blend shapes as well as a set of corrective shapes; these additional basis elements correspond to expressions that are not contained in the original shape set. The corrective shapes capture the fine-scale details. Due to the non-linear nature of the resulting blend shape model, Laplacian deformations are used to align the tracked data to the three-dimensional model. Then the expectation-maximisation algorithm is used in the adaptive PCA space to iterative improve the space of the corrective shapes. Specifically, given a number of sufficiently diverse input samples and the initial guess of the corrective space, the algorithm estimates the coefficients of the corrective space. Then, the algorithm finds the model that best explains the samples given the corrective coefficients. After each two cycles of the algorithm, the newly acquired corrective space is orthonormalised, and used to decrease the error during the tracking. This method is capable of producing appealing visual results in real-time. However, it requires a scan of the actors face, and the corrective shapes are not directly used during the retargeting.

A different direction was chosen by Garrido et al., who aim to reconstruct high quality three-dimensional models using data only from a monocular video [GVWT13]. The method relies on a pre-designed blend shape model, and uses optical flow to produce three-dimensional motion. Meanwhile, Xu et al. improved the retargeting of facial animation from an actor onto a digital model [XCLT14]. The authors use a multi-scale approach, where a targeted optimisation algorithm is used to achieve best results at the course and the fine scale. Moreover, the proposed method provides the user with a set of control tools that are used to alter the automatically produced results. This blend shape model produces high-quality visual results but it still requires significant input from the user.

2.4 Skin Rendering

Rendering realistic skin is a challenging task. As social beings we interact with interact with other individuals on a daily basis, which has made human perception quite sensitive to skin appearance, even more so with human faces. Skin is composed of several layers with different properties, to accurately simulate skin the light transport between this layers has to be simulated. The full effect of light scattering between two points on the surface can be modelled using a Bidirectional Surface-Scattering Distribution Function (BSSRDF).

Weyrich et al [WMP*06] proposed a two-layer model for skin rendering, the outer layer simulates the air-oil interface and the inner layer models the subsurface scattering in the skin. The authors considered the scattering to be homogeneous, with this assumption they measured the skin BRDF of several subjects in a light dome, while the scattering was sampled at three points in the face with a custom made sensor. The BRDF data was fitted to a Blinn-Phong and a Torrance-Sparrow isotropic models, and the scattering was fitted with a single transport coefficient. Donner et al [DWd*08] also proposed a two-layer model, however the authors allow for the layers to be heterogeneous. With this addition they are able to introduce the effects of haemoglobin, veins and tattoos. Emotional induced haemoglobin variations have also been explored [JSB*10]. The authors measured the haemoglobin distributions of several subjects in different poses, then a linear combination of the captured data would determine the final haemoglobin distribution for a new sequence. Recently, Iglesias et al [IGAJG15] introduced a five-layer model to handle skin ageing. Haemoglobin, collagen and fat changes with age are modelled using the different layers.

Normal maps are use to alter the normals of the scene objects during rendering. This technique is used to add geometric detail to an object at rendering time without actually changing the geometry. Normal maps for skin rendering are usually captured using expensive light domes with a number of synchronized cameras [GTB*13, WMP*06].

Another technique to increase the quality of a face render is to scale the resolution of the textures being used. Ashikhmin et al [Ash01] presented a method to generate new textures using a goal image by greedily extending existing patches whenever possible. Hertzmann et al [HJO*01] extended Ashikhmin et al [Ash01] method by adding a second example image and using more complex distance metric to choose the next synthesized pixel. Graham et al [GTB*13] applied Hertzmann et al [HJO*01] example-based filter to generate bump maps with increased quality for skin rendering. An alternative approach using a dictionary of samples was presented by Jianchao et al [YWHM10]. This method is restricted to generating super-resolution images, however, the previous methods support a wide variety of filter effects. For an in depth analysis of super-resolution techniques, we refer the readers to Tian et al [TM11] survey.

Chapter 3

Methodology

3.1 Data Capture

3.2 Sparse Reconstruction

3.3 Blendshape Optimisation

In this section we introduce the techniques used to warp the meshes, and the optimisation methods.

3.3.1 Thin Plate Splines

Thin plate theory addresses problems that commonly arise in areas of natural sciences and engineering when trying to model the behaviour of a thin sheet of some material. The possible processes include but are not limited to stretching, bending, crumpling, buckling, shrinking, straining and tearing. The corresponding mathematical model is based on ideas from differential geometry as well as continuum mechanics, and the set of equations describing the aforementioned phenomena are often notoriously difficult to solve. Therefore, in Computer Graphics, as well as other fields, a number of simplifying assumptions are made when constructing a thin plate model.

A thin plate is considered to be a two dimensional object, i.e. it is assumed that the thickness is infinitesimal. The geometry of the object is often simplified to reduce the computational cost. Thin plate splines (TPS) are a two-dimensional counterpart of the cubic spline [SS91]. TPS are a deformation method based on the assumption that a thin surface deforms in a way that minimises the surface bending energy. The bending energy is proportional to the change in the second fundamental form. Specifically, given two corresponding sets of point $\{(x_i, y_i\}_i^N$ and

$\{v_i\}_i^N$ there exists a height field mapping between the two, $f : \mathbb{R}^2 \to \mathbb{R}$. The bending energy corresponding to this mapping is proportional to the second order derivatives of the mapping:

$$E_{bend}(f) = \lambda \iint \left(\left(\frac{\delta^2 f}{\delta x^2} \right)^2 + 2 \left(\frac{\delta^2 f}{\delta xy} \right)^2 + \left(\frac{\delta^2 f}{\delta y^2} \right)^2 \right) \mathrm{d}x \, \mathrm{d}y, \tag{3.1}$$

where λ is smoothing parameter, which balances the quality of fit and the amount of bending, i.e. the wiggliness of the function. TPS finds the transformation that fits the data while minimising the bending energy. Note that TPS may also be defined in terms of the radial basis functions that are used for smooth scattered data fitting. The RBF solution for thin plate splines is:

$$f(x, y) = \sum_{i=1}^{N} \omega_i \, \phi(\|(x, y) - (x_i, y_i)\|), \quad \text{where } \phi(r) = r^2 \log(r). \tag{3.2}$$

To ensure that the function f has square-integrable second derivatives, the following conditions are imposed:

$$\sum_{i=1}^{N} \omega_i = \sum_{i=1}^{N} \omega_i x_i = \sum_{i=1}^{N} \omega_i y_i = 0. \tag{3.3}$$

These conditions and the data fitting requirement may be combined into a linear system of equations:

$$\begin{bmatrix} K & P \\ P^T & 0 \end{bmatrix} \begin{bmatrix} \omega \\ o \end{bmatrix} = \begin{bmatrix} v \\ o \end{bmatrix}, \tag{3.4}$$

where K encodes the relation between the data points, i.e. $K_{ij} = \phi(\|(x_i, y_i) - (x_j, y_j)\|)$, $P_{i\cdot} = (1, x_i, y_i)$ contains the variables from Eq. 3.3, ω contains the values of ω_i, and v contains the target function values v_i. The variables 0 and o denote the zero matrix and the zero column vector respectively. Solving this linear system of equations gives the desired transformation in two dimensions. The method extends naturally to three-dimensional problems by including a dependence on an additional variable in the calculations described above. Radial basis functions are commonly used in the field of performance-driven animation [JTDP03].

3.3.2 Non-Rigid ICP

Iterative Closest Point (ICP) is an algorithm used to align two partially overlapping point clouds by minimising the square error between corresponding points. The quality of the results produced by this algorithm is sensitive to the initial guess at a solution, i.e. ICP only refines the initial estimation. See Alg. 1 for the outline of the algorithm. Mean square error algorithm is used to calculate the average of the squared errors between the target and the transformed source points. Thus the objective function is a function of rotations and translations. The output of the algorithm is a transformation matrix that provides the optimal mapping between the two point clouds within a given threshold. The transformation matrix may be split into rotation and translation. The algorithm solves a linear system of equations where the unknowns are the coefficients in the rotation matrix and the translation vector and the known point cloud values are used as coefficients. The method is often used to find a transformation between a point cloud and a surface; in that case the surface normals are used as additional input.

Algorithm 1: Iterative Closest Point.

Data: source point cloud
target point cloud;
initial guess;
error threshold;

while *error > threshold* **do**
  for *point in source* **do**
    find closest point in target;
  end
  use mean squared error to find best transformation that aligns source to target ;
  apply transformations;
end

The original ICP algorithm is limited to rigid transformations, i.e. rotation and translation, which have only 6 degrees of freedom. However, in order to capture scaling, shearing and other more complicated transformation, an affine mapping should be used. Thus an extension of the ICP algorithm, called non-rigid ICP is used when additional degrees of freedom are present. Allen et al. proposed a non-rigid registration method that uses a numerically non-linear solver to find a smooth affine mapping [ACP03]. Additional constrained are imposed by manually matching some known marker locations in the two datasets. The method exhibits slow convergence and often leads to local minima; consequently, the authors use a multi-resolution approach where the optimisation is first performed on a low resolution mesh, and then optimised on the high resolution mesh. Amberg et al. combined the ICP algorithm with the method of Allen et al. to overcome the issues with convergence [ARV07].

3.3.3 Blendshape model

The aim of this project is to used captured data of a face to animate a given model. We use the digital Emily model that comes with 68 controllable blendshapes ranging from simple eyebrow movements to complicated lip corner pulls. Each blendshape has a weight w_i associated with it, and $w_i \in [0, 1]$. The model includes eyes and teeth. The aim is to find a combination of these blendshapes that produce a specified facial expression. In particular, we are interested in reproducing the captured sequence of expressions.

First attempt. We manually construct a sparse set of points that corresponds to the sparse source mesh Fig. **??**???. Then the neutral expression of the subject is matched with the neutral expression of Emily by transforming the sparse source mesh to the sparse target mesh. The transformation is then stored, and it is used to map all the expressions in the captured sequence. This produces a sparse Emily sequence; using TPS again we warp the dense Emily mesh according to the positions of the sparse points. The quality of the resulting animation is poor for a number of reasons, see Fig.**??**. Firstly, the sparse points are not able to constrain the dense mesh, especially since parts of the face, for example the eyelids and the lips are not tracked in the sparse mesh. Consequently, numerous artefacts appear on the dense mesh, and they are particularly noticeable around the lips as no boundary conditions are imposed. Secondly, though there is a clear correspondence between the generated Emily sequence and the source sequence, the generated expressions look unnatural. This may be explained by the differences in the geometry and anatomy of the two faces; for instance, the source may be able to pull expressions that may not be mimicked by the model. In addition, the errors in tracking, reconstruction, and manual selection of sparse points on Emily further reduce the quality of the animation.

Simplified mesh. Our initial attempt to improve the method involved simplifying the dense Emily model by removing the part of the mesh that corresponds to the lips. Though the simpler mesh exhibited slightly better behaviour, i.e. there were fewer visible artefacts, the sparse point cloud was still unable to constrain the dense mesh well enough to produce realistic results, see Fig. **??**.

Blend shapes. In an attempt to improve the visual quality of the generated animation, we decided to use a set of Emily key-shapes, that were readily available to us. The main argument for using these blend shapes is that in most situations a simple linear relation exists between an arbitrary facial expression and the set of key-shapes. There exist a number of different options for create a set of shapes; the shape may encode the entire geometry of the case, the displacement from the neutral face, or the local features of a face. Alternatively, a set of blend shapes may be producing using a dimensionality reduction technique. Such set of blend shapes is orthogonal, thus the features encoded by each shape are controlled independently; however, such shapes do not allow for intuitive facial transformations making it difficult to perform manual post-process editing.

Let us define a set of blend shapes $\mathbf{B} = [\mathbf{b}_0, \ldots, \mathbf{b}_n]$ where $\mathbf{b}_0$ is the mesh of a neutral face, and each of the $\mathbf{b}_i$ correspond to a basic facial expression, for example the raise of an eyebrow. Unless stated otherwise, we shall use 68 shapes provided with the digital Emily model. Then a new facial expression $\mathbf{F}(\mathbf{w})$ may be constructed by finding an appropriate linear combination of the basic shapes:

$$\mathbf{F}(\mathbf{w}) = \mathbf{b}_0 + \sum_{i=1}^{N} w_i \, |\mathbf{b}_i - \mathbf{b}_0|, \tag{3.5}$$

where $\mathbf{w} = [w_1, \ldots, w_n]$ is a set of weights that describe how much each of the basic shapes affect the new expression $\mathbf{F}(\mathbf{w})$. Typically, a convexity constraint is imposed on the weights; in our case the choice of constraints is influenced by the constraints present in our blend shape model, i.e. $w_i \in [0, 1]$. Eq. 3.5 may be written as a linear system of equations as follows:

$$\mathbf{F}(\mathbf{w}) - \mathbf{b}_0 = \hat{\mathbf{B}}\mathbf{w}, \tag{3.6}$$

where $\hat{\mathbf{B}}$ is the original set of blend shapes without the neutral expression.

A number of different numerical optimisation methods where tested when solving for the blend shape weights. An unconstrained solution may be calculated by inverting matrix $\hat{\mathbf{B}}$, and multiplying it from the left with the left hand side of Eq. 3.6. Instead we use Matlab's constrained non-negative least-squares solver (`lsqnonneg`) and constrained linear least-squares solver (`lsqlin`) [MAT13]. For the non-negative least-squares solver we formulate our problem as follows:

$$\min_{\mathbf{w}} \|\hat{\mathbf{B}}\mathbf{w} - (\mathbf{F}(\mathbf{w}) - \mathbf{b}_0)\|_2^2,\ \mathbf{w}_i \geq 0,\ i = 1, \ldots, n. \tag{3.7}$$

The solver uses an active set method, and iterative improves the active set of basis vectors; it converges in finite time. The same formulation is used for the linear least squares solver though it includes an additional upper limit on the argument. This solver uses a reflexive Newton method which is able to accurately locate the local minimisers of large systems, and exhibits global convergence. It is also able to maintain sparsity of matrices but is generally slower.

3.4 Skin Rendering

Our objective in skin rendering is to generate a face that would be indistinguishable from a real one. In our case, we have have taken a 3D scan of a subject and our aim is to improve the realism when rendering it. For this task we will mainly look at the techniques presented by Hertzmann et al [HJO*01] and Graham et al [GTB*13].

Before we begin, lets explain the Image Analogies framework [HJO*01] in more detail. Given three images A, A' and B, where A is an unfiltered example, A' is a filtered example, and B is an input image, the algorithm will generate an output image B' such that B' relates in the same way to B, as A' does to A. A k-d tree for an Approximate Nearest-Neighbour Search (ANN) is built using a feature vector from a neighbour pixel p in A and A'. The closest match for a neighbourhood in pixel q in B and B' is located in the tree. A detailed description of the algorithm in pseudo code is shown in Algorithm 2, where F is computed a weighted distance over the feature vectors using a Gaussian kernel and s is a data structure such that $s(q) = p$. Following Ashikhmin et al [Ash01] method, a match that is coherent to what has been already synthesized is computed as well. These two candidates are weighted and the best one is chosen. The whole process is carried in a multiresolution pyramid, as shown in Figure 3-1, where l indicates the current level, in essence this means that the neighbourhoods also include the previous level in the search. We found three Image Analogies implementations available [ImAa, ImAb, ImAc]. The first one is a simple single threaded implementation, the second one was done with CUDA, however the author's single threaded code produced overall better results.

As a first approach we tried to reproduce the results for bump mapping quality increase by Graham et al [GTB*13], results are shown in Figure 3-3. The authors add an extra parameter $\alpha \in \{0, \ldots, 1\}$ to control Hertzmann's image synthesis process. To be more precise, a match between A and $\{B, B'\}$ will be weighted by $1 - \alpha$, and a match between A' and $\{B, B'\}$ will be weighted by α. The logic behind this addition is to encourage more details of A' to be included in B'. The modifications required to include this addition begin by building two separated k-d trees for A and A', when choosing the best match both distances will be weighted as explained

Algorithm 2: Image Analogies

Data: A unfiltered example, A' filtered example, B unfiltered source, L number of levels, k coherence parameter, t neighbourhood size.
Result: B' filtered source image.
Compute Gaussian pyramids for A, A' and B;
Compute features for A, A' and B;
Build k-d tree for $\{A, A'\}$;
for *l = 0 to L* **do**
  for *each pixel $q \in B'_l$* **do**
    p_{app} = ANN search of q neighbourhood from $\{B, B'\}$;
    $r^* = \arg\min_{r \in N(q)} \|F_l(s(r) + q - r) - F_l(q)\|^2$;
    $p_{coh} = s(r^*) + (q - r^*)$;
    $d_{app} = \left\|F_l(p_{app}) - F_l(q)\right\|^2$;
    $d_{coh} = \|F_l(p_{coh}) - F_l(q)\|^2$;
    if $d_{coh} < d_{app}(1 + k2^{l-L})$ **then**
      $p = p_{coh}$
    else
      $p = p_{app}$
    end
    $B'_l(q) = A'_l(p)$;
    $s_l(q) = p$;
  end
end
return B'_L

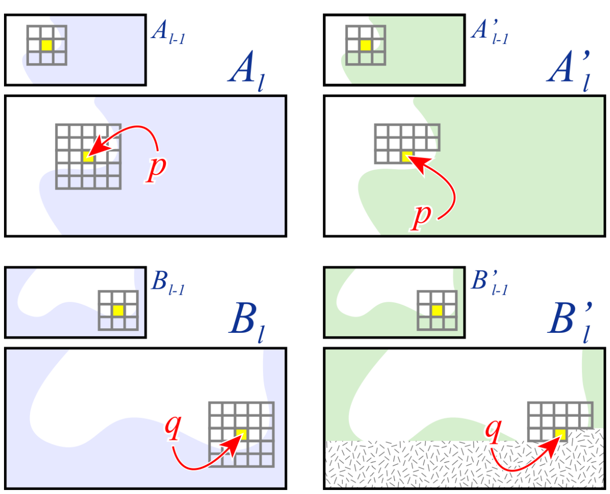

Figure 3-1: Neighbourhood matching for the Image Analogies framework, image taken from [Ash01].

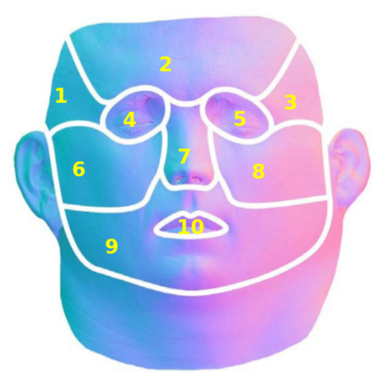

Figure 3-2: Texture segmentation in common coloured areas, image taken from [GTB*13].

above and the smaller one will be chosen. Also in the coherence match two searches will be done and weighted accordingly, and the final pixel will be chosen without further adjustments.

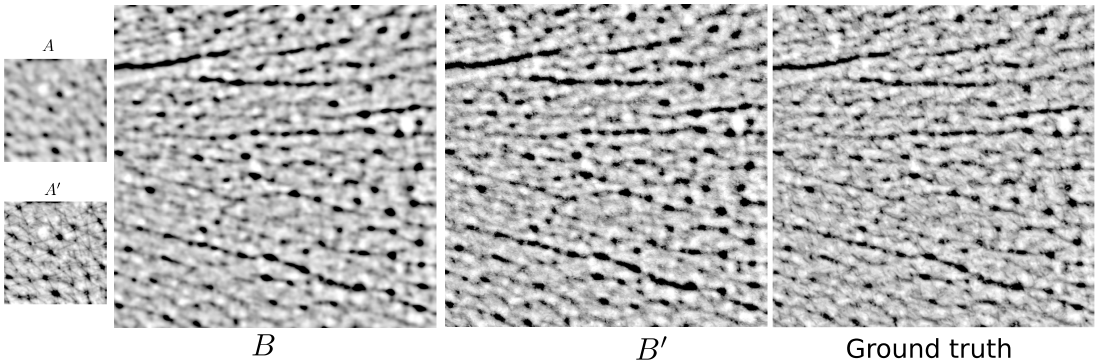

Figure 3-3: Bum map deblurring, A, A', B and ground truth images images taken from [GTB*13].

Another approach we tried was to use the Image Analogies filter to create increased quality textures. The idea is to improve a low quality texture from a 3D scan using pictures of the texture taken at a closer range. To achieve this we took a close up high quality sample A', to generate A, the sample A' was blurred using a Gaussian kernel until it look qualitative similar to the 3D scan texture B, with this three inputs we generated a picture B' of higher quality. Since faces have differentiated areas, this process was done separately for each of them, the generated patches are stitched together using linear interpolation. The texture segmentation is shown in Figure 3-2 and results are shown in Figure 3-4.

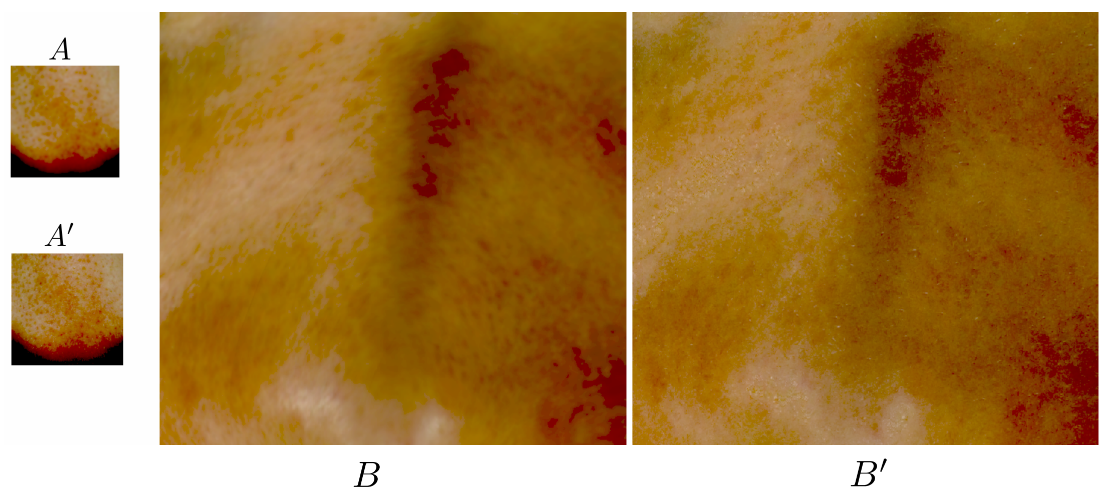

Figure 3-4: Texture quality increase using image analogies, the texture is false-coloured to highlight the differences.

Another option for increasing the quality of the texture is to use image super-resolution, we used Jianchao et al [YWHM10] work for this purpose. The idea is that by doubling the resolution of the original texture, yet avoiding blur by adding information from a dictionary of high resolution images, the final rendering quality of the face will increase. Results for this approach are shown in Figure 3-5.

Figure 3-5: Super resolution example, left original texture, left-centre face rendered with original texture, right-centre super-resolution texture, right face rendered with super-resolution texture, original data from [Fac].

Generating normals maps using Image Analogies is another interesting area, as it could provide an alternative to the costly standard capture methods. For this we tried to generate a normal maps from albedo images and from bump maps generated from the previous 3D scan texture, results are shown in Figure 3-6. To create the bump maps, the textures were transformed to gray scale and a histogram equalization was applied. In order to improve the quality of the bump maps, we also applied the Image Analogies filter to them using a known good bump map as a filtered example, however the quality did not improve significantly.

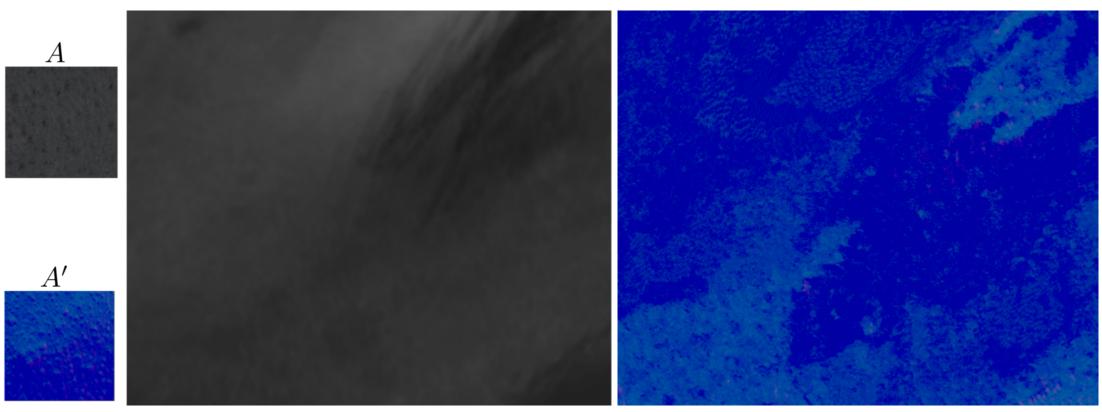

Figure 3-6: Normal synthesis from albedo image.

Chapter 4

Implementation details

Talk about Maya plugins and any other implementation stuff worth talking about.

Chapter 5

Results

Chapter 6

Conclusions

Bibliography

[ACP03] ALLEN B., CURLESS B., POPOVIĆ Z.: The space of human body shapes: Reconstruction and parameterization from range scans. *ACM Trans. Graph. 22*, 3 (July 2003), 587–594.

[ARL*09] ALEXANDER O., ROGERS M., LAMBETH W., CHIANG M., DEBEVEC P.: The digital emily project: Photoreal facial modeling and animation. In *ACM SIGGRAPH 2009 Courses* (New York, NY, USA, 2009), SIGGRAPH '09, ACM, pp. 12:1–12:15.

[ARV07] AMBERG B., ROMDHANI S., VETTER T.: Optimal step nonrigid icp algorithms for surface registration. cited By 37.

[Ash01] ASHIKHMIN M.: Synthesizing natural textures. In *Proceedings of the 2001 Symposium on Interactive 3D Graphics* (New York, NY, USA, 2001), I3D '01, ACM, pp. 217–226.

[BBA*07] BICKEL B., BOTSCH M., ANGST R., MATUSIK W., OTADUY M., PFISTER H., GROSS M.: Multi-scale capture of facial geometry and motion. *ACM Trans. Graph. 26*, 3 (July 2007).

[Ben05] BENNETT D.: The faces of "the polar express". In *ACM SIGGRAPH 2005 Courses* (New York, NY, USA, 2005), SIGGRAPH '05, ACM.

[BRM12] BALTRUŠAITIS T., ROBINSON P., MORENCY L.-P.: 3d constrained local model for rigid and non-rigid facial tracking. pp. 2610–2617. cited By 14.

[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* (New York, NY, USA, 1999), SIGGRAPH '99, ACM Press/Addison-Wesley Publishing Co., pp. 187–194.

[BWP13] BOUAZIZ S., WANG Y., PAULY M.: Online modeling for realtime facial animation. *ACM Trans. Graph. 32*, 4 (July 2013), 40:1–40:10.

[CK05] CHOE B., KO H.-S.: Analysis and synthesis of facial expressions with hand-generated muscle actuation basis. In *ACM SIGGRAPH 2005 Courses* (New York, NY, USA, 2005), SIGGRAPH '05, ACM.

[CLK01] Choe B., Lee H., Ko H.-S.: Performance-driven muscle-based facial animation. *Journal of Visualization and Computer Animation 12*, 2 (2001), 67–79. cited By 55.

[DCFN06] Deng Z., Chiang P.-Y., Fox P., Neumann U.: Animating blendshape faces by cross-mapping motion capture data. In *Proceedings of the 2006 Symposium on Interactive 3D Graphics and Games* (New York, NY, USA, 2006), I3D '06, ACM, pp. 43–48.

[DWd*08] Donner C., Weyrich T., d'Eon E., Ramamoorthi R., Rusinkiewicz S.: A layered, heterogeneous reflectance model for acquiring and rendering human skin. *ACM Trans. Graph. 27*, 5 (Dec. 2008), 140:1–140:12.

[Fac] Faceware Tech. http://facewaretech.com/. Accessed: 2015-05-19.

[GGW*98] Guenter B., Grimm C., Wood D., Malvar H., Pighin F.: Making faces. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1998), SIGGRAPH '98, ACM, pp. 55–66.

[GHDS03] Grinspun E., Hirani A. N., Desbrun M., Schröder P.: Discrete shells. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Aire-la-Ville, Switzerland, Switzerland, 2003), SCA '03, Eurographics Association, pp. 62–67.

[GTB*13] Graham P., Tunwattanapong B., Busch J., Yu X., Jones A., Debevec P., Ghosh A.: Measurement-based synthesis of facial microgeometry. *Computer Graphics Forum 32*, 2pt3 (2013), 335–344.

[GVWT13] Garrido P., Valgaert L., Wu C., Theobalt C.: Reconstructing detailed dynamic face geometry from monocular video. *ACM Trans. Graph. 32*, 6 (Nov. 2013), 158:1–158:10.

[HJO*01] Hertzmann A., Jacobs C. E., Oliver N., Curless B., Salesin D. H.: Image analogies. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 2001), SIGGRAPH '01, ACM, pp. 327–340.

[IGAJG15] Iglesias-Guitian J. A., Aliaga C., Jarabo A., Gutierrez D.: A biophysically-based model of the optical properties of skin aging. *Computer Graphics Forum (EUROGRAPHICS 2015) To Appear* (2015).

[ImAa] Image Analogies: C++ single threaded code. http://vis.berkeley.edu/courses/cs294-69-fa11/wiki/index.php/FP-JeffDonahue. Accessed: 2015-05-19.

[ImAb] Image Analogies: CUDA code. https://sites.google.com/a/college.harvard.edu/yaoyu-imageanalogies/parallel. Accessed: 2015-05-19.

[ImAc] Image Analogies: Hertzmann's code. http://www.mrl.nyu.edu/projects/image-analogies/. Accessed: 2015-05-19.

[JSB*10] Jimenez J., Scully T., Barbosa N., Donner C., Alvarez X., Vieira T., Matts P., Orvalho V., Gutierrez D., Weyrich T.: A practical appearance model for dynamic facial color. In *ACM SIGGRAPH Asia 2010 Papers* (New York, NY, USA, 2010), SIGGRAPH ASIA '10, ACM, pp. 141:1–141:10.

[JTDP03] Joshi P., Tien W. C., Desbrun M., Pighin F.: Learning controls for blend shape based realistic facial animation. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Aire-la-Ville, Switzerland, Switzerland, 2003), SCA '03, Eurographics Association, pp. 187–192.

[LYYB13] Li H., Yu J., Ye Y., Bregler C.: Realtime facial animation with on-the-fly correctives. *ACM Trans. Graph. 32*, 4 (July 2013), 42:1–42:10.

[MAT13] MATLAB: *version 8.1.0.604 (R2013a)*. The MathWorks Inc., Natick, Massachusetts, 2013.

[NN01] Noh J.-y., Neumann U.: Expression cloning. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 2001), SIGGRAPH '01, ACM, pp. 277–288.

[PHL*98] Pighin F., Hecker J., Lischinski D., Szeliski R., Salesin D. H.: Synthesizing realistic facial expressions from photographs. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1998), SIGGRAPH '98, ACM, pp. 75–84.

[PKC*03] Pyun H., Kim Y., Chae W., Kang H. W., Shin S. Y.: An example-based approach for facial expression cloning. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Aire-la-Ville, Switzerland, Switzerland, 2003), SCA '03, Eurographics Association, pp. 167–176.

[PSS02] Pighin F., Szeliski R., Salesin D.: Modeling and animating realistic faces from images. *International Journal of Computer Vision 50*, 2 (2002), 143–169. cited By 48.

[Sco] Scopus. http://www.scopus.com/. Accessed: 2015-05-21.

[SP04] Sumner R. W., Popović J.: Deformation transfer for triangle meshes. *ACM Trans. Graph. 23*, 3 (Aug. 2004), 399–405.

[SS91] Sibson R., Stone G.: Computation of thin-plate splines. *SIAM J. Sci. Stat. Comput. 12*, 6 (Sept. 1991), 1304–1313.

[TM11] Tian J., Ma K.-K.: A survey on super-resolution imaging. *Signal, Image and Video Processing 5*, 3 (2011), 329–342.

[VBPP05] Vlasic D., Brand M., Pfister H., Popović J.: Face transfer with multilinear models. *ACM Trans. Graph. 24*, 3 (July 2005), 426–433.

[WHL*04] Wang Y., Huang X., Lee C.-S., Zhang S., Li Z., Samaras D., Metaxas D., Elgammal A., Huang P.: High resolution acquisition, learning and transfer of dynamic 3-d facial expressions. *Computer Graphics Forum 23*, 3 SPEC. ISS. (2004), 677–686. cited By 60.

[WMP*06] Weyrich T., Matusik W., Pfister H., Bickel B., Donner C., Tu C., McAndless J., Lee J., Ngan A., Jensen H. W., Gross M.: Analysis of human faces using a measurement-based skin reflectance model. In *ACM SIGGRAPH 2006 Papers* (New York, NY, USA, 2006), SIGGRAPH '06, ACM, pp. 1013–1024.

[XCLT14] Xu F., Chai J., Liu Y., Tong X.: Controllable high-fidelity facial performance transfer. *ACM Trans. Graph. 33*, 4 (July 2014), 42:1–42:11.

[YWHM10] Yang J., Wright J., Huang T., Ma Y.: Image super-resolution via sparse representation. *Image Processing, IEEE Transactions on 19*, 11 (Nov 2010), 2861–2873.