Visual Effects

submitted by

Ieva Kazlauskaite, Garoe Dorta-Perez, Richard Shaw

unit

Unit Leader: Dr Darren Cosker

University of Bath

Department of Computer Sciences

EngD in Digital Entertainment

May 2015

Signature of Author . . . . . .

Ieva Kazlauskaite, Garoe Dorta-Perez, Richard Shaw

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.*" [**?**]

2 Previous Work

2.1 Data Capture

2.2 Sparse Reconstruction

2.3 Blendshape Optimisation

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.

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 deals with 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, 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:

$$E_{bend}(f) = \lambda \iint \left(\left(\frac{\delta^2 f}{\delta x^2} \right)^2 + 2 \left(\frac{\delta^2 f}{\delta xy} \right)^2 + \left(\frac{\delta^2 f}{\delta y^2} \right)^2 \right) \mathrm{d}x \, \mathrm{d}y, \tag{1}$$

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:

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

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

$$\sum_{i=1}^{N} \omega_i = \sum_{i=1}^{N} \omega_i x_i = \sum_{i=1}^{N} \omega_i y_i = 0. \tag{3}$$

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

$$\begin{bmatrix} K & P \\ P^T & 0 \end{bmatrix} \begin{bmatrix} \omega \\ o \end{bmatrix} = \begin{bmatrix} v \\ o \end{bmatrix}, \tag{4}$$

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, 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.

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. 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. **??**.

Introduction of blenshapes. In an attempt to improve the visual appeal 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

A number of different numerical optimisation methods where tested when solving for the blendshape weights.

3.4 Skin Rendering

Our objective in skin rendering is to generate a face that would indistinguishable from a real one. To be more specific, we have have taken a 3D scan of a subject and our aim is to improve the realism of 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, as shown in Figure 1. 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. 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 1, where l indicates the current level, in essence what this means is that the neighbourhoods also include the previous level in the search.

As a first approach we tried to reproduce the results in bump mapping quality increase by Graham et al [GTB*13]. The authors add am extra $alpha \in \{0, \ldots, 1\}$ parameter to control Hertzmann's image synthesis process. In more detail, a match between A and $\{B, B'\}$ will be weighted by $1 - \alpha$, while 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

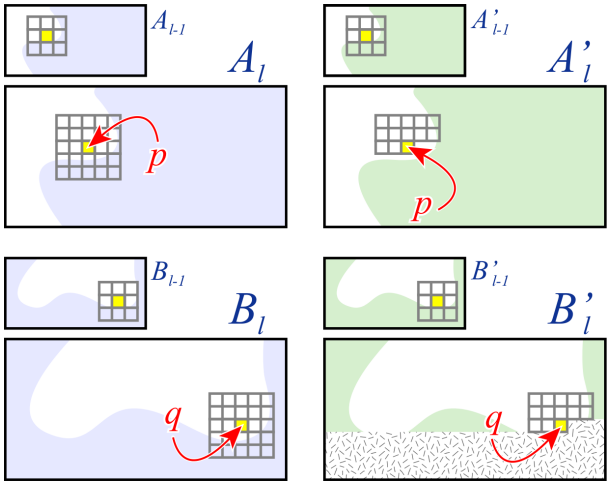

Figure 1: Neighbourhood matching for the Image Analogies framework, image taken from [Ash01].

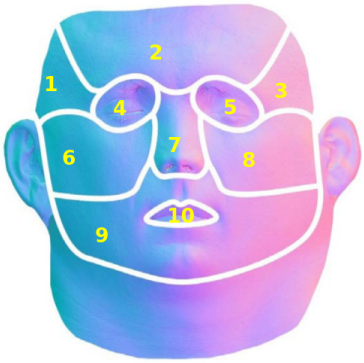

Figure 2: Texture segmentation in common coloured areas, image taken from [GTB*13].

include building two separated k-d trees for A and A' and then doing the appropriate weighting when choosing the pixel position. Also in the coherence match two searches will be done and weighted accordingly, and the final pixel will be chosen without further adjustments. ADD BUMP MAP IMAGE OR NOT IF THERE IS NOT ENOUGH SPACE

Another approach we tried was to use the Image Analogies filter to create increase quality textures. We found three implementations available [ImAa, ImAb, ImAc]. The second one was done with CUDA, however the author's single threaded code gave better results. The idea in this approach is to use a low quality texture from a 3D scan and improve using pictures of the texture at a closer range. To achieve this we took a close up high quality sample A', which was blurred using a Gaussian kernel until it look qualitative similar to the 3D scan texture B. The blurred image is A and then 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 then stitched together using linear interpolation. The texture segmentation is shown in Figure 2 and results are shown in Figure 3.

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 adding information from a dictionary of high resolution images we will be able to increase the final rendering quality of the face. Results for this are shown in Figure 4.

Generating normals maps using Image Analogies is another interesting area, as it could provide a way to avoid he costly standard capture methods. For this we tried to generate a normal map from an albedo images and from bump maps generated from the previous 3D scan texture. Results for this are shown in Figure 5.

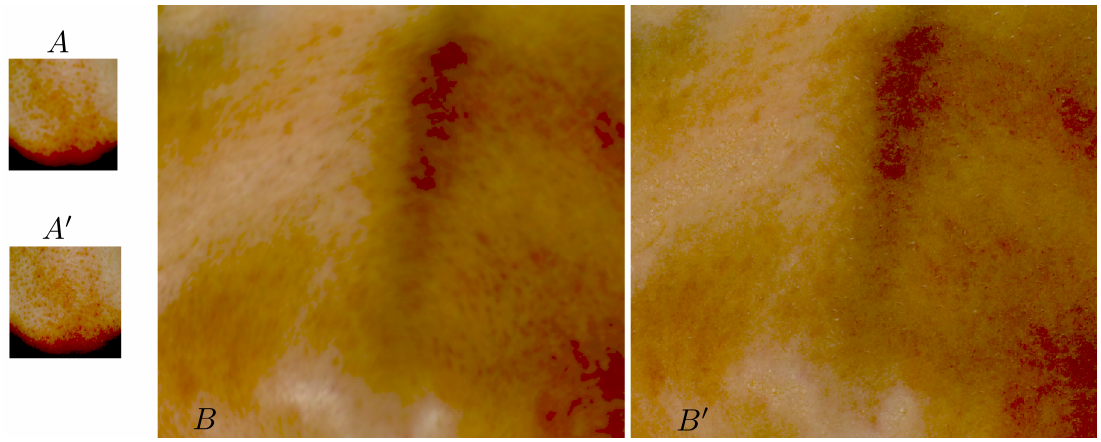

Figure 3: Texture quality increase using image analogies, the texture is false-coloured to highlight the differences.

Figure 4: Super resolution example, top-left original texture, top-right face rendered with original texture, bottom-left super-resolution texture, bottom-right face rendered with super-resolution texture, original data from CITE EMILY DATA.

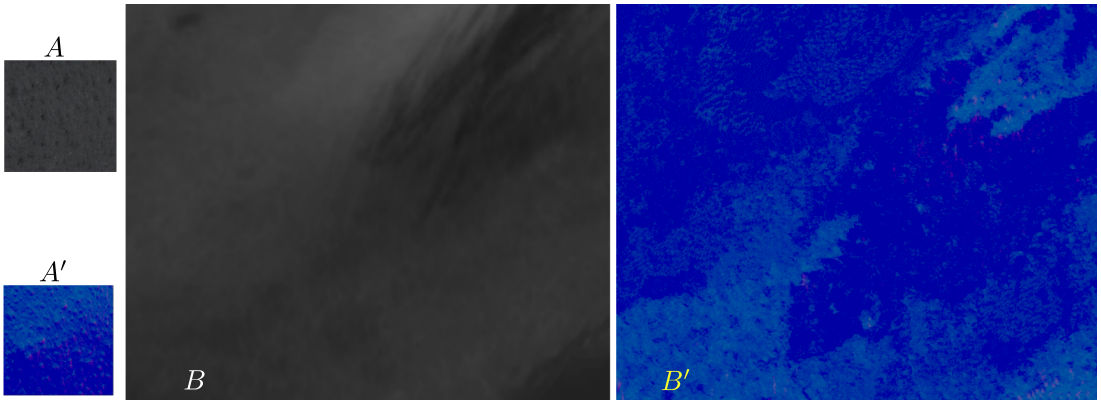

Figure 5: Normal synthesis from albedo image.

4 Results

5 Conclusions

References

[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.

[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.

[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.

[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.

[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.

[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.

[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.

[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.