Face Capturing and Calculation of Average 3D Face Models

Marcus Richter†, Daniel Stengel‡, and Tim Zeiß∗

Abstract—Average faces are used in multiple areas including biometric and social studies as well as facial recognition software. In the past average faces were computed in 2D images but these are limited if a 3D representation is needed. 3D representations usually were not textured, therefore we decided to create a fully colored 3D Mesh. We divided the averaging process into averaging mesh and averaging textures. While averaging the mesh is fairly simple, we also found a good and fully automated way of averaging the textures. We created a single application that implements the complete process of scanning faces, managing them and calculating average faces. Our result is a textured 3D representation which is more realistic than previous methods.

Index Terms—3D Face Scanning, Calculation of Average Faces, Microsoft Kinect.

1 INTRODUCTION

The computation of average faces has gained high relevance in many fields, including sociology and face recognition. It has become essential for attractiveness studies and is often used as a base for digital face modelling, like creating virtual faces. Therefore, computing 2-dimensional average faces has become quite popular and there are already many approaches that give good results. There also exist methods for creating 3D faces which are mostly used in facial recognition software. These methods usually work on point clouds or depth maps and completely omit colors. Up to this point, there are no methods that compute colored and realistic 3D average faces. That’s why we decided to create a software that is able to capture 3D faces, including a 3D mesh and a corresponding texture, and compute an average face with these information.

First we give an overview of existing methods (Sec. 2). After this, we explain the method that we used to create 3-dimensional textured average faces (Sec. 3). Then we give a deeper view on the implementation of said method (Sec. 4), divided in averaging the mesh and the texture, where we describe the algorithms we used to calculate an average face and show the problems we encountered. We also created a Software Manual that tells the user how to capture new faces and calculate average faces with our software (Sec. 5). Finally, we give a small discussion about the output and performance of our software (Sec. 6) and show some possible improvements for the future, as well as possible uses for our program (Sec. 7).

2 RELATED WORK

In the past different methods have been proposed for creating average faces, since average faces can exist in different forms. The different types can be roughly divided into two categories: 2D images and 3D meshes or point clouds. Here we want to shortly introduce to these two types and name examples implementing or proposing methods to calculate average faces in the corresponding form.

The 2D image variant is the more commonly used and therefore multiple implementations are available. Because humans are used to viewing 2D images, the representation of a face as a 2D image is very natural to humans. Hence popular application areas include studies about biometrics and social studies. As many tools using this method are available for free, many people also like to use them just for fun to see how a mix of them and their friends would look like or how their future child could look like. Compared to the 3D variant 2D images are good especially because the input is easy to acquire. Only a single image taken from a standard camera is needed. Also the process can be fully automated after the input is given to the algorithm which makes it as simple as it can be for the user. On the other hand 2D images carry no 3D information. So if a 3D representation is needed, obviously a 3D method must be used. So the input for 2D color image Methods are two images of two different faces and the output is an image of both faces merged into one. For the merging step the two input images need to be blended together in a way that the facial features are lying on top of each other. To achieve this, facial features are recognized in both input images and the input images are morphed in shape. Then a simple blending can be used to create the final image. A popular example that implements this method is the Psychomorph software by Dr Bernie Tiddeman [4]. Psychomorph can be used for a wide variety of areas involving feature detection. It is lightweight, has plugin support and is available for free. There are also online examples that are even easier to use since you only need to start your browser and upload the images. Examples are [5] which is an online implementation of Psychomorph.
There also exist several less serious online implementations like [2] that are meant for quick and fun use.

Another method to create an average face is by using 3D meshes of the face. This can be useful to analyse the shape of different faces, because meshes make it easy to calculate shape derivations between different faces. With this, face anomalies can be detected and analysed. To capture 3D meshes you need additional hardware that is not commonly available. Another advantage of 3D meshes is that they can be animated, which can be pretty useful for sociology and psychology studies, for example by showing different emotions.

The most obvious disadvantage with this method is the lack of a texture, which highly restricts their use for sociology and psychology studies, since untexturized faces dont feel natural. An example for the use of average 3D meshes is [3]. There, 2 3D cameras have been used to get face meshes of 72 children that were merged into an average face. These information were used to calculate the derivation of each point in a mesh compared to the average mesh, which shows facial anomalies of other children.

3 Method

In contrast to the aforementioned methods to create average faces, our method creates a fully color textured 3D mesh. As a result, compared to the 2D and the depth image methods, our calculated average faces can be represented in a more natural and realistic way. This makes our results more suitable for psychology studies, where a realistic representation is important so that probants react correctly.

We implemented all the needed functionalities for scanning and managing scanned faces, searching and filtering for specific subsets as well as calculating and managing average faces, all in one user-friendly application. This way we keep the software setup for the user to a minimum as he only needs our application for the process of calculating average faces. For 3D scanning we wanted to use a depth camera to yield good accuracy. We use the Microsoft Kinect which is easy to setup, easy to acquire and relatively cheap so everybody can afford it or may even already have one at home.

To scan faces we are using the High Definition Face class [1] which was added to the version 2 interface of the Kinect. It creates a high resolution 3D triangle mesh by sampling multiple depth images via the Kinect Sensor. The created mesh consists of 1347 vertices and the topology of the mesh is always the same. Every vertex has a special meaning in terms of face features and therefore by itself offers a high resolution result for facial feature detection. This saves us a lot of work since facial feature detection is essential for calculating average faces. The process for the sampling is fully automated and already implemented in the Kinect. While it samples images it gives instructions over a feedback stream to tell the user turn the face in different directions. Because it fully tracks the face in the procedure, the timing, speed and angle of the turns may vary slightly and the Kinect still knows what to do with the images. If a turn for example was too fast for the Kinect to recognize it will tell via the feedback stream that more samples from the missing view are needed. More details on the exact procedure can be found in section 5.1.

For creating the textures of the scanned faces we use the output from the color camera of the Microsoft Kinect. We take a single image and use planar mapping to project the texture onto the face. We wanted the scanning process to be very fast and easy to be able to scan as many probants as possible in a short period of time, which is often helpful in large scale studies.

All scanned faces are then conveniently stored in a database inside the application where subsets can be selected via a filter function. To calculate an average face we wanted it to be as easy as clicking one button. The calculation does not require any additional user inputs and should be fast enough to handle a couple hundred faces on an average desktop pc. All 3D meshes stored by the application are stored in the .obj format and inside a simple folder structure. This way all scanned and calculated faces can be easily extracted from the software and used in different applications. It also allows the user to scan faces on different computers and merge the results into a common database, making it possible for multiple users to work simultaneously and exchange their data. This makes it easier to create a large database from various locations all over the world to get results that are more representative.

4 Implementation Details

4.1 Averaging the mesh

The Microsoft Kinect uses the same topology on every scanned face. This way an average mesh can be easily achieved by calculating the arithmetic mean for every vertex separately:

\[ v_{i}^{\text{average}} = \frac{\sum_{j=0}^{n} (v_{j}^{i})}{n} \]

\( v_{i}^{j} \) is the corresponding vertex in the mesh \( j \). Note that we divide inside of the sum which is less efficient, but is needed since otherwise we would lose accuracy with increasing number of meshes due to the sum becoming a large number. We use a per mesh algorithm so that every mesh needs to be loaded only once and only one mesh needs to be in the memory at a time:

```plaintext
var averageMesh // average mesh to calculate meshes // input meshes
begin
for m in meshes do
begin
for i := 0 to 1347 do // all meshes consist of 1347 vertices
begin
averageMesh.vertices[i] += m.vertices[i] / meshes.length;
end;
end;
end;
```

Figure 2: The handmade uv unwrap we use for average faces

![Image](image.png)
4.2 Averaging the texture

Similar to the approach when averaging the mesh we use the fact that the Microsoft Kinect uses the same topology for all faces. As the resulting average face also has the same topology we can directly use the texture coordinates from the scanned faces to project the textures onto the average face. We use the same static handmade uv unwrap for every average face (Fig. 2). Since the mesh of faces are generally very similar, a static texture unwrap produces better results as any automated unwrap we could find. More importantly we can pre calculate large parts of the texture averaging process to significantly reduce calculation times.

Beginning the process we create an empty texture which will be the texture for the average face. To calculate the color of every pixel we first determine where on the mesh this pixel would end up on. To do so we search for the triangle in the mesh that the pixel is intersecting. Because in our unwrap no triangles share the same area on the texture we can be sure that there can be only up to one triangle the pixel is intersecting. After finding the triangle we calculate the exact position inside the triangle by calculating the barycentric coordinates of the pixel. With large texture sizes all the intersection tests can take a very long time. But as we use a static unwrap we can pre calculate the information for every pixel and ship the results for common texture sizes with the application. Now we use the texture coordinates from the input meshes and the previously calculated information where on the mesh the pixel is to calculate the uv coordinates on the input texture, that equates to the projection onto the pixel from the average texture. With the uv coordinates we can read the corresponding color value from the input texture. For every pixel on the average texture we do this for every input texture to calculate the arithmetic mean of the color value of that pixel. (Note that we divide inside of the sum and use a per texture algorithm for the same reasons mentioned in the mesh average process).

5 SOFTWARE MANUAL

Software requirements are Windows 8 or later version, Kinect SDK and .NET Framework

5.1 Scanning Process

To scan and add a new 3D Face to your gallery you press the Record New 3D Face Button in the Gallery Window (Fig.3) which opens the Scan Window (Fig.4) To record the facial contours of your proband click Start Capture Button. First the proband needs to be tracked by the Kinect if proband is not tracked by sitting in front of the camera you should tell him to stand up and go some steps behind. If the proband is still not tracked you should make sure that no other proband, user or object is in front of the camera which could falsely be tracked by the camera.

For optimal capturing conditions read Sec. 6.1.

5.1.1 Capturing Process

The proband first needs to look straight ahead into the camera, then the proband needs to turn slowly in 6 seconds left to a maximum of 45 degree then slowly back to the middle. Now the proband should slowly turn right to a maximum of 45 degree and again back to the middle. At least the proband needs to look up at a maximum of 30 degree and again back to the middle to be ready for texturize.

The feedback line tells you what samples are needed to finish the

Figure 3: The main window that opens on startup

Figure 4: The scan window where the full scanning process takes place

Figure 5: After scanning is complete, the result shows in this preview
mesh (Fig.4) like left views are needed, right views are needed, etc., so tell your proband what to do next until capturing is finished.

To create the appropriate textures make sure the environmental conditions are the same for every proband. Our proposal is to use two diffuse studio lights one on the right side of the proband and one on the left side to avoid shadows on the face. Apart from that, people with oily skin could cause shiny effects, if you want to avoid this you may need to powder probands with oily skin. To get the highest possible resolution make sure the proband is as close as possible in front of the camera (50 cm in front of the camera is the minimum distance) and looks neutral into the camera. Click the Texturize Button to take a snapshot of the proband as 2D texture. If you want to change the texture click the Texturize Button again or click on Save Button if the texture fits and you are satisfied with this texture. After this, the texture cannot be changed.

A click on the Save Button opens the Preview Window (Fig.5). Here you can see the 3D Face in detail. You can zoom in and out by scrolling with your mouse wheel, rotate with click and draw into a direction and move the face with the right mouse button. If you are dissatisfied with the 3D Face you can delete it by clicking the Delete Button and start the procedure with your proband again with a click on Record New 3D Face in the Gallery Window. If you are satisfied with the 3D Face you can add and edit appropriate attributes such as name, age, etc. After choosing all the attributes you want to save with this 3D Face click the Save Button to finish the scan procedure. The 3D Face is then saved and added to the Gallery Window.

5.2 Average Process

To create an average face you can select a group of 3D Faces by using the filter options. (Fig.5) For example if you want to search all 3D Faces which have the attributes between 20 and 40 years old, named Otto, male, all nationalities and any religion, you need to insert 20 to 40, write the name Otto, select male and use the default attributes for nationality and religion. If you want to search the 3D Faces with these attributes click the Apply Filter Button. To reset the filter and show all available 3D Faces click the Reset Filter Button. Before you create the average face of the selected group, you can change the resolution that the calculated average face should have. Texturize offers the four options 256x256 (default), 512x512, 1024x1024 and 2048x2048 but the higher the resolution the more time is needed to calculate the average face. To start the calculation click the Calculate Button. While the software is calculating a progress bar is shown that gives information about the overall progress and what the program is doing right now. After the calculation process is finished a Preview Window of the calculated average face will open (Fig.7). To see all average faces click the Show Average Faces Button in the Gallery Window which opens the Average Face Gallery (Fig.6). By double clicking on one average face the 3D Preview (Fig.7) of this average face will be shown together with all the 3D Faces which were merged to this result.

6 Results

The result of averaging three faces can be seen in (Fig.1). The depth sensor of the Kinect has a resolution of 1.5mm at the minimal distance of 50cm which is accurate enough for scanning faces to a visually accurate degree as seen in (Fig.1). The scanning process implemented with the Kinect that we use is extremely user friendly. Only a single camera setup is needed and the angles of the different samples can vary since the Kinect at the same time fully tracks the face and therefore knows the exact position to incorporate that in the 3D reconstruction algorithm. The output of the scan is a 3D mesh of just the face region consisting of 1347 vertices which is enough to represent all major details in a face. This resolution is not high enough for minor details in a face like facial wrinkles or impurities. But when averaging multiple faces, small details would be averaged out anyway due to noise in the input data averaging out. The texture size for faces is 1920x1080 which is far more than sufficient, but because the color camera of the Kinect has a fairly wide angle and there is a minimum distance you have to stand away for the depth sensor to work, the final resolution of only the face is just approximately 250x350. Again this seems to be a very low resolution but is sufficient for averaging. For the averaging step itself it is easy to visually verify the correctness in (Fig.1).

The performance of our method is good enough to easily handle datasets of a couple hundred faces on an average desktop pc, because we were able to keep everything in linear complexity. On our test system the averaging process for a set of one hundred faces took 8.35 seconds for the minimum output texture resolution of 256x256 pixels seconds and 36.71 seconds for the maximum output texture.
resolution of 2048x2048 pixels.

6.1 Capturing Conditions
To get the best results, the Kinect should be placed at face level so that the proband looks straight into the camera. He should be as close as possible to the camera to achieve the best scanning resolution. A near clipping plane indicates the closest possible position. Though the capturing process also works with sunlight, it is highly recommended to use artificial studio lighting to keep the lighting conditions stable for every proband and to prevent shadows on the face. The Kinect is able to track multiple objects, though it may occur that it loses track of the proband for a short moment because of interfering movements, in which case the capturing process has to be restarted. To avoid any interference with the tracking system, you should choose a location without any moving objects or people in the background. During the capturing process, the face of the proband should be fully visible. Everything that covers parts of the face like hats, glasses, jewelry or scarves should be removed. Very shiny skin should be powdered to avoid problems with depth sensor as well as highlights in the color texture. The scanning of all probands should be done in a similar environment to ensure the same conditions on all face scans and in turn to avoid errors in the average face.

7 Conclusion
Our software is successfully scanning faces and creating average faces in a high and realistic 3D quality. The resolution of both the resulting mesh and the texture is sufficient for visually correct results. The performance of our method is even better than we had expected. Since the core calculations can be very efficiently implemented in a compute shader, the performance could be increased even more to a point where datasets of thousands of faces could be easily handled by average desktop PCs.

7.1 Limitations
Our resulting 3D average faces are very realistic compared to previous methods and therefore can be very useful in sociology and psychology studies. But some applications may still require a more detailed representation. The 3D average faces that our software creates are limited most by the input faces. The Kinect offers 3D face meshes with 1347 vertices which is enough for most applications, but if you need more detailed shape feature you may want to use a better depth sensor. The color camera of the Kinect has a resolution of 1920x1080, but because the depth sensor needs a minimum distance of about 50cm to be able to track faces the face texture has an effective resolution of approximately 250x350. If you get closer to the camera to get a higher resolution, the Kinect loses track of the face and the texture cannot be mapped automatically. Since the texture is taken from a single image, it is restricted to the front view of a face. Though this already covers most parts of the face, some applications may require a more detailed view of the sides, in which case you would need more than one image to get a wider view.

7.2 Future Work
Since our resulting 3D average faces are highly dependant on the input faces, the accuracy could be greatly improved by using better input faces. Higher resolution depth sensors and more accurate reconstruction algorithms could be used to achieve more detailed meshes and a separated texturing process with a better camera and more accurate mapping methods could greatly improve the texture quality. With these improvements our method could be accurate enough for skin studies and other fields that require more accurate results.

References