Proceedings of the 9th Workshop on the Representation and Processing of Sign Languages, pages 107–112
Language Resources and Evaluation Conference (LREC 2020), Marseille, 11–16 May 2020

A survey of Shading Techniques for Facial Deformations on Sign Language Avatars

Ronan Johnson, Rosalee Wolfe
DePaul University, Chicago, IL, USA
sjohn165@depaul.edu, wolfe@cs.depaul.edu

Abstract

Of the five phonemic parameters in sign language (handshape, location, palm orientation, movement and nonmanual expressions), the one that still poses the most challenges for effective avatar display is nonmanual signals. Facial nonmanual signals carry a rich combination of linguistic and pragmatic information, but current techniques have yet to portray these in a satisfactory manner. Due to the complexity of facial movements, additional considerations must be taken into account for rendering in real time. Of particular interest is the shading areas of facial deformations to improve legibility. In contrast to more physically-based, compute-intensive techniques that more closely mimic nature, we propose using a simple, classic, Phong illumination model with a dynamically modified layered texture. To localize and control the desired shading, we utilize an opacity channel within the texture. The new approach, when applied to our avatar "Paula", results in much quicker render times than more sophisticated, computationally intensive techniques.

Keywords: Sign Language Synthesis, Nonmanuals, Illumination Models, Avatars

1. Introduction

In all humans, facial movements can convey important information including emotion, social cues, and a person's general demeanor. However, in signed languages, such facial movements are also used to convey important linguistic information. Notable examples include differences in eyebrow position when asking yes/no versus ”wh” questions, and nonmanual adjectives and adverbs that co-occur with their corresponding sign (Baker-Shenk, 1985; Reilly et al., 1990). Although a hearing person might initially try to understand sign language by following the movement of the hands, in fact, native signers focus primarily on a user's face (Siple, 1978). This is such an important component of comprehension that when viewing an avatar with displaying insufficient facial distinction, many signers become irritated or distracted. This can decrease their understanding of the utterances being portrayed (Kipp et al., 2011). To this end, accurate facial movements, clearly portrayed are of utmost importance when recreating signed language through an avatar.

Visual legibility is crucial to such portrayals. Subtle nuance and co-occurring actions in facial movements communicate important linguistic information. For example, the intensity of a nonmanual modifier to a verb corresponds to the intensity of the modification. Additionally, puffed cheeks in conjunction with a sign conveying a large object will indicate an extreme size difference. In English, we might interpret this as the difference between "large" and "huge" (Baker-Shenk, 1983).

Using avatars for linguistic research with Deaf participants requires making the movements and visual look of real life signers as closely as possible. However, one of the many challenges in rendering sign language avatars is properly gauging the trade-off between realism and computational efficiency. While it is important that digital visualizations mimic real life signers as closely as possible, one must also be cognizant of the technical challenges involved with rendering complicated avatars in real-time.

One such consideration is the lighting conditions used to illuminate an avatar. It has been observed that Deaf participants respond more positively to lighting conditions that clearly illuminate the hands and face. Furthermore, light and shadows that improve the realistic appearance of an avatar makes the avatar seem more like an interpreter (Kipp et al., 2011).

This paper presents a discussion of three primary illumination models in computer graphics and their potential application to realistic portrayal of sign language avatars. We conclude with an improved approach involving an implementation of layered textures to achieve a solid solution to the complexity/realism trade-off.

2. Illumination Models

In real life, the way light interacts with objects can be taken for granted. The same cannot be said for the world of computer graphics. In order for a computer generated object to be rendered visible on a screen, the computer must calculate the color of each pixel based on the conditions present in the 3D scene. Such conditions include the presence of any geometry, available light sources, and the surface properties of the geometry which determine how light will interact with any given object. Illumination models are algorithms designed to calculate light reflection from objects in a scene (Hall, 1986). There are three main types of illumination models, each with their own benefits and drawbacks. They are listed as follows:

- Local illumination
- Global illumination
- Semi-global illumination

2.1. Local Illumination – The Phong Illumination Model

Local illumination models empirically calculate the color of a point on a surface based on the properties of that surface and the properties of any light coming directly from a

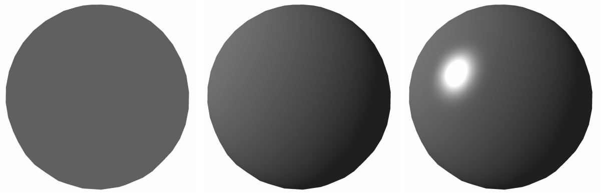

Figure 1: The ambient, diffuse, and specular components of the Phong illumination model.

light source that strikes the point (Phong, 1975). This set of algorithms is easy to compute, often requiring a minimal amount of coding. However, these models often prove insufficient for truly realistic displays of geometry, as they only a rough approximation of the natural effects of the light.

One of the most fundamental local illumination models is known as the Phong illumination model, named after its initial proposer Bui Tuong Phong in 1975. This algorithm computes the color of a point on an object by breaking the surface properties out into three components: the ambient light being reflected around the scene, the diffuse reflection of a direct light source, and the specular reflection from a direct light source. The algorithm then computes the sum of these components for each of the red, green, and blue color channels for each light present in the scene (Phong, 1975). This summation is the final result rendered to the screen. A breakdown of each of these components is shown in Figure 1.

Because local illumination only considers light coming directly from a light source, and ignores light from reflecting surfaces, the resulting shading can be very harsh and unnatural. To counteract this effect, illumination models add a constant lighting term called ambient light. It represents an amount of light present on an object in the absence of any direct light source.

In a local illumination model, diffuse reflection is the even scattering of a direct light according to Lambert's law. The strength of this reflection only depends on the strength of the incident light and the surface's light absorption properties. The viewing angle does not affect it (Cohen and Greenberg, 1985). The specular reflection of a surface is the amount of light reflected from the surface, based on the direction at which the object is viewed (Phong, 1975). It is responsible for the highlight or "shiny spot" seen in highly polished surfaces.

While this illumination model is simple to compute, it has several major drawbacks. For one, it is unable to render shadows cast by one object onto another object. We can observe this by placing another piece of geometry directly below the sphere in Figure 2. Although the sphere is physically touching the floor, the absence of shadow makes the sphere appear to float above the surface. We also see that this model also does not compute the effects of indirect light or bounced light, remaining reliant on direct light sources. In general, local illumination models such as Phong are unable to render the kind of complex detail we need for maximum legibility in sign language production. In order to achieve these properties, we must turn to more advanced illumination models.

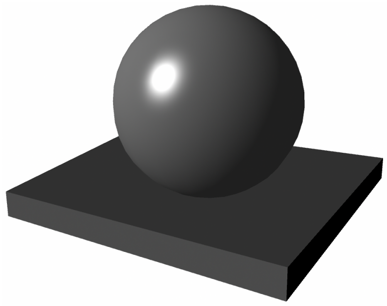

Figure 2: Phong illumination does not calculate cast shadows or indirect light.

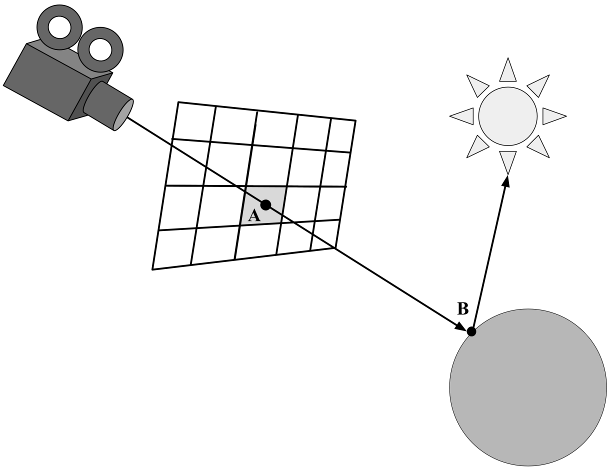

Figure 3: The calculation of the color of a pixel using ray tracing.

2.2. Global Illumination

2.2.1. Ray Tracing

Ray tracing was actually first described in 1532 by Albrecht Dürer, a renaissance artist who famously learned to draw by looking at his subjects through a grid (Hofmann, 1990). First described by Appel in 1967, this illumination technique uses the physical properties of light to simulate how photons would interact with a scene (Appel, 1967).

In the real world, light is emitted from a light source, then it bounces around a room until some of its photons reach a viewer's eye. This allows the viewer to see the object. Understandably, some photons may take more bounces than others. Others may never reach the viewer at all. It is a waste of processing time to compute light bounces that will never make it to a viewer. To avoid processing light that never enters the viewer's eye, ray tracing computes the light bounces backwards. A ray is begun at the center of each pixel in the computer generated camera's view. The algorithm then traces this ray across the scene until it intersects with some object. A final ray is then cast from this point towards the light source. The final color of the pixel is the result of the light and object properties that the ray encounters (Appel, 1967).

In 1979, this technique was expanded by Turner Whitted. His algorithm generated multiple new rays at the points of intersection, which could then bounce across the scene until they either hit a light source, or some maximum number of bounces was reached. The final color of the pixel is then back-computed from the combined information collected by each generated ray. In doing so, ray tracing became capable of yielding photorealistic images including cast shadows, reflections from shiny surfaces, and refraction through water and glass (Whitted, 2005). However, ray tracing's main drawback is its computational demands. At present, real-time ray tracing is a heated area of research, especially in the field of video games. The most successful techniques at time of writing are only achievable with dedicated graphics hardware and specialized APIs (Liu et al., 2019).

2.2.2. Radiosity

Of the illumination models surveyed, radiosity most closely simulates the actual physics of lighting. Radiosity, a form of global illumination, is similar to ray tracing, except that it tracks all the rays starting at the light source(s) and bounces them around the entire scene. As a white light ray bounces off a surface, it can change color, depending on the surface's properties. It treats the diffuse reflection of nearby objects as an additional ambient light source to simulate indirect lighting. Because of this, radiosity allows color bleeding from nearby objects to be visible. Because the ambient illumination is no longer a constant term, this technique is capable of rendering fine detail and soft, highly realistic shadows that deepen in corners (Cook and Torrance, 1981). This effect can be observed in Figure 5.

Of course, this comes at the cost of render time. Unlike ray tracing, radiosity is view-independent, meaning the lighting of the whole scene is computed, not just the light as seen from the camera (Cohen and Greenberg, 1985). Understandably, this would greatly increase the complexity of the computations. Compare this to ray tracing which reduces the complexity by calculating the illumination for only the geometry visible to the camera.

Figure 4 shows a diagram of the basic action of the light rays during radiosity calculations. The illumination of a given point C is calculated based on the sum of the illumination components of the rays leading from the light source to that point. Point A is where the direct light ray initially encounters geometry. The illumination at point B will then be calculated taking into consideration the properties of point A. Point C will be calculated with the properties of rays AB and BC.

2.3. Semi-Global with Ambient Occlusion

Semi-global methods attempt to produce the effects of global illumination but at a lower computational cost. For highly dynamic scenes, real-time radiosity is still untenable. However, it is possible to achieve some amount of the effect using approximations (Ritschel et al., 2009). One such approximation is ambient occlusion. This refers to the effect adjacent geometry has in blocking some sources of bounce light from reaching a specific point (Scherson and Caspary, 1987). True ambient occlusion is a by-product of radiosity, but it can be approximated using a variant of ray tracing and added on later using a render pass (Miller, 1994). The classic example is the way the corners of a room appear darker than the adjacent wall, as described in Figure 6.

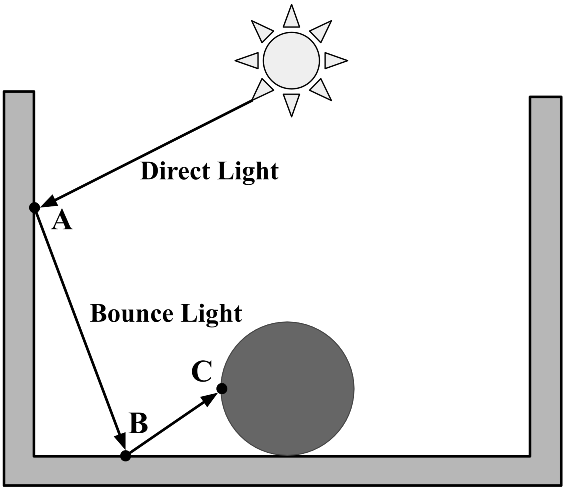

Figure 4: The calculation of radiosity.

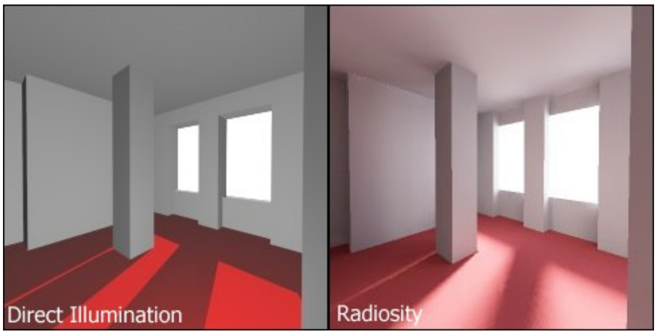

Figure 5: The shadows created by radiosity are softer and more realistic than direct illumination through ray tracing (Elias, 2006).

Ambient occlusion simplifies the calculations necessary for creating the soft shadows of radiosity. This greatly improves performance in real-time applications while providing significant improvements in image quality (Ritschel et

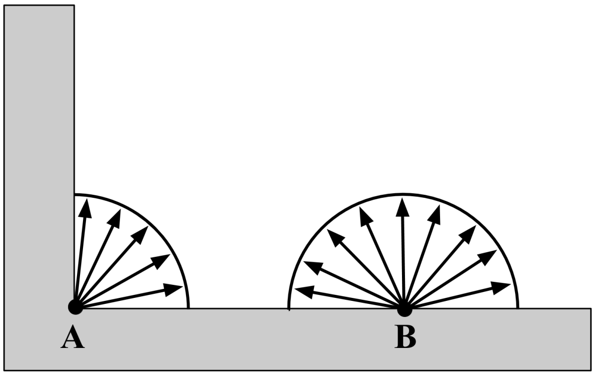

Figure 6: Point B can receive bounce light from any point in the scene. Point A can receive bounce light from fewer sources. This causes the point to be darker.

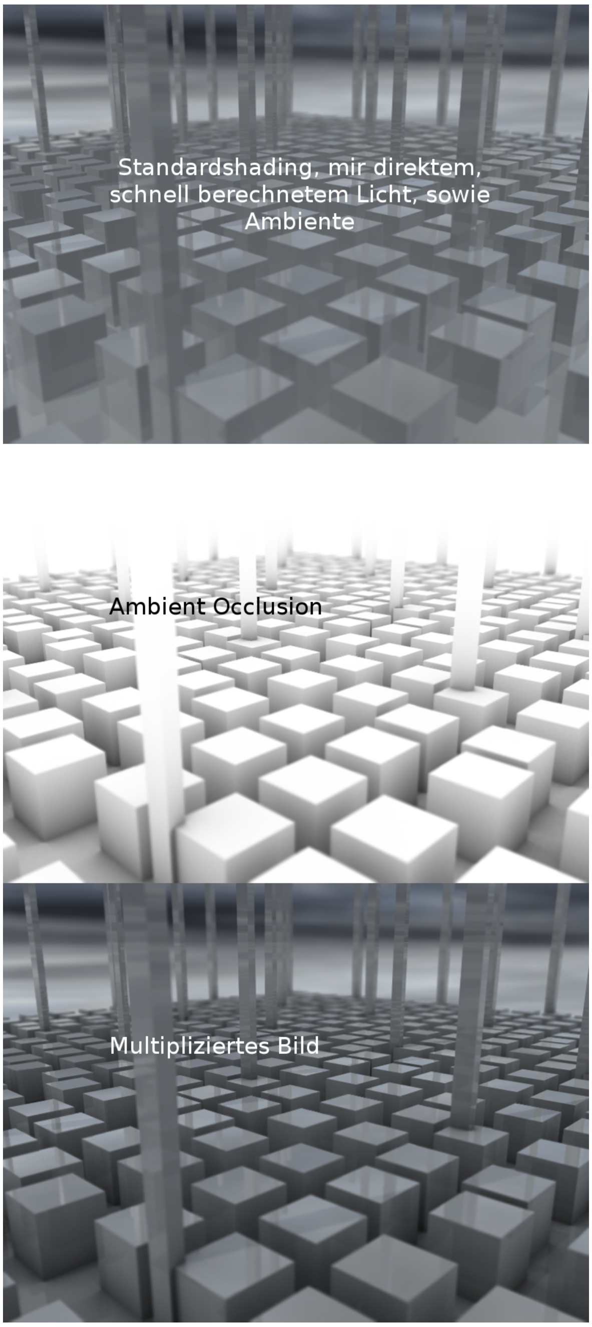
Standardshading, mir direktem, schnell berechnetem Licht, sowie Ambiente
Ambient Occlusion
Multipliziertes Bild

Figure 7: The addition of ambient occlusion heightens the realism of computer generated images (TheWusa, 2006).

al., 2009).

The sequence of images in Figure 7 shows the level of realism that can be achieved with ambient occlusion. The top image shows a scene rendered without ambient occlusion. The middle image shows the incident illumination computed by an ambient occlusion pass as an approximation of radiosity. The bottom image shows the results of combining this ambient occlusion with the initial rendering pass.

3. Trade-offs Among the Three Approaches

The Phong illumination model is easy to compute and yields fast results. However, the final renders often lack the amount of desired realism. In particular, it is incapable of rendering shadows. Ray tracing can compute these shadows, but they remain unrealistically harsh. Although soft shadows are possible with ray tracing, it comes at the cost of greatly increased render times (Scherson and Caspary, 1987). Radiosity provides the greatest level of realism, but the the most computationally expensive of all the options. The benefits of the soft shadows can be approximated with ambient occlusion. This is faster than true radiosity, it still represents a huge time cost. Optimizing ambient occlusion for real-time applications remains an ongoing area of research (Jiménez et al., 2016).

4. Applications to Sign Language Avatars

We require that our avatar be able to closely mimic reality as much as possible. To that end, the global and semi-global illumination models provide us the best opportunity for realistic rendering. However, their complexity makes them untenable for our real-time applications. We therefore turn to the Phong illumination model, which provides us the render times we need to implement our software on a variety of computers. In order to overcome the limitations in the level of detail this model can portray, we add pre-rendered texture maps to create shaded areas of fine detail. This gives us the speed of local illumination combined with the realism of radiosity.

4.1. A Layered Texture Approach

Previous work in applying the layered texture technique involved controlling dynamic textures to indicate facial deformations. There are several instances in portraying facial deformations on sign language avatars where the presence of shadows is necessary to clearly convey an appropriate amount of detail. One major example is the horizontal wrinkles that form on the forehead when the eyebrows are raised (Wolfe et al., 2011). Creating physical wrinkles requires the addition of significantly more geometry in that area. Our previous work in optimizing our avatar, "Paula" for real-time rendering describes the challenges associated with rendering large amounts of geometry (McDonald et al., 2016). In order to avoid major increases in render times, we elected to implement a layered texture that would fade in and out depending on the position of the eyebrows.

A further exploration of the semi-global illumination model of fast ambient occlusion yielded an insight into the problem of creating a more realistic, clearer face rendering that emphasized all necessary facial poses while achieving video rates while rendering. This approach pre-renders the shadows that appear in folds of the skin when a person extends or contracts sets of facial muscles. Additionally, the shading will become darker or lighter depending on the intensity of the deformation. For example, the higher the eyebrows are raised, the darker the forehead wrinkles become.

One hindrance of extending the previous approach was the opaque nature of the textures. Painting the wrinkle shadow directly onto the texture of the face and then turning that texture on and off precludes the introduction of additional wrinkles, which are necessary for producing co-occurring facial actions. In the previous approach it was possible to show the forehead wrinkles or enhanced eye creases, but to

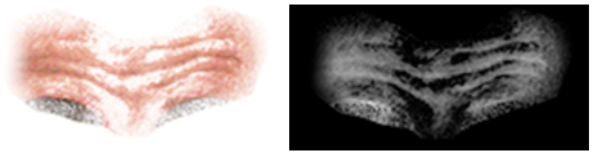

Figure 8: The wrinkled brows texture is isolated on the left. The right shows the alpha channel for this texture.

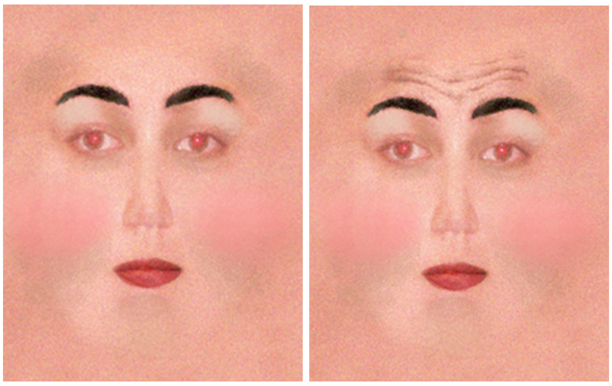

Figure 9: The left image is the base texture for our avatar's face. The right image shows the results of layering the wrinkled brows texture over this base using its alpha channel.

show both at the same time required a prohibitively complex masking function.

The question then was how to combine multiple pre-rendered textures. In addition to the classic red, blue, and green channels of an image, a new approach introduces an alpha channel in the pre-rendered textures. An alpha channel controls the visibility of the texture. The color black denotes the areas of an image that are transparent, as shown in Figure 8. Therefore, when layered atop one another, the base texture remains unchanged while the additional layers are able to be ”turned on and off” individually. Figure 9 shows the results of this layering. This provides complete control over selectively adding simulated ambient occlusion to localized areas. This approach removes the limitation of using only one texture map at a time. Not only are the render times unaffected by this addition, but it also allows greater freedom and flexibility in the final look of the renders. This is much faster than attempting to physically model and shade the smaller folds of the fast. Changes made by artists can be seen almost immediately.

This technique is also useful for facial areas where the geometry deformations do exist, but a conventional illumination model is not producing the level of shadows necessary for effective portrayal. The primary deformation that motivated this work was in the cheek puff action where an amount of air is pushed into the cheek, creating a rounded, protruding shape. However, the previous illumination model and lighting setup were inadequately rendering the shadows necessary to clearly portray the deformation. Figure 10 shows the original cheek puff deformation on the left and the same deformation with an added texture layer creating the ambient occlusion effect. With this alone, there is a perceivable increase in legibility of the cheek puffing action.

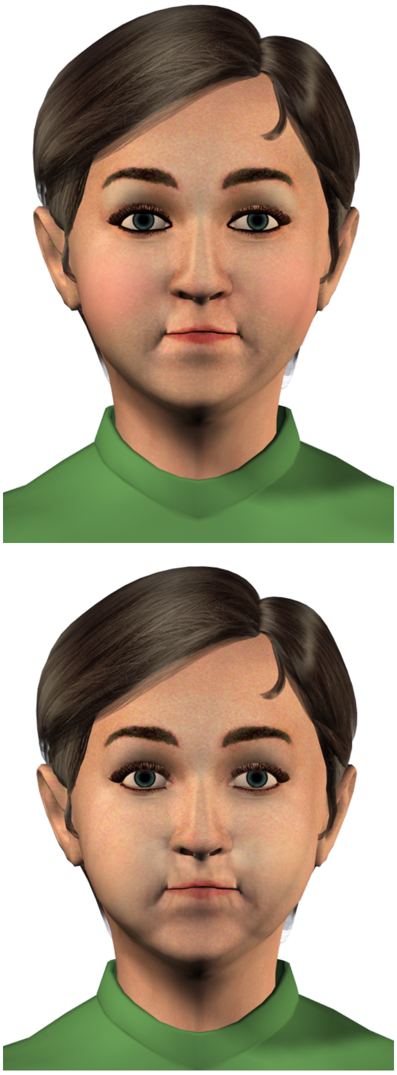

Figure 10: Paula displaying the puffing action with (bottom) and without (top) additional ambient occlusion.

As with the eyebrows, the intensity of the layered textures can be dynamically controlled with the intensity of the deformation. This creates a fading effect as the face animates so there is no distraction from the texture suddenly popping in and out. This is also important for portraying subtleties of the facial deformations denoting linguistic information. With variable intensity, Paula is capable of portraying the full variety of intensity modifiers to concurrent manual signs.

We are able to successfully implement this approximation based on two assumptions. The first assumption is the use of a static lighting setup, meaning there are limited changes in the ambient light as Paula's head turns relative to the viewer. Because there is limited motion relative to the lighting, the shadows remain consistent and predictable. The other assumption is the predictability of Paula's behavior. The location and intensity of the cheek puff deformation is within a strict range of parameters. This means we do not have to account for the general case of shading every possible position and deformation.

5. Conclusion and Future Work

A new technique for emphasizing facial deformations through texture mapping yields greater legibility while still maintaining the superior performance of local (simple) illumination models when rendering. This technique will be applicable to other areas of the face that experience similar deformations. One such example is the upper area of the cheek near the eyelids that pushes up and forward when expressing a smile or scrunching the face. Future work includes creating additional texture layers and continuing to compare render times with a conventional single-texture model to assess whether there is a maximum practical limit to the number of layers. This will maintain Paula's ability to be run on machines without high-end processors and graphics cards.

6. Acknowledgment

We would like to acknowledge Elena Jahn for her work on the makeup textures for Paula.

Bibliographical References

Appel, A. (1967). The notion of quantitative invisibility and the machine rendering of solids. In *Proceedings of the 1967 22nd national conference*, pages 387–393.

Baker-Shenk, C. (1983). A microanalysis of the nonmanual components of questions in american sign language.

Baker-Shenk, C. (1985). The facial behavior of deaf signers: Evidence of a complex language. *American Annals of the Deaf*, 130(4):297–304.

Cohen, M. F. and Greenberg, D. P. (1985). The hemi-cube: A radiosity solution for complex environments. *ACM Siggraph Computer Graphics*, 19(3):31–40.

Cook, R. L. and Torrance, K. E. (1981). A reflectance model for computer graphics. *ACM Siggraph Computer Graphics*, 15(3):307–316.

Elias, H. (2006). https://upload.wikimedia.org/wikipedia/commons/5/55/radiosity_comparison.jpg. licensed under the gnu free documentation license, version 1.2.

Hall, R. (1986). A characterization of illumination models and shading techniques. *The Visual Computer*, 2(5):268–277.

Hofmann, G. R. (1990). Who invented ray tracing? *The Visual Computer*, 6(3):120–124.

Jiménez, J., Wu, X., Pesce, A., and Jarabo, A. (2016). Practical real-time strategies for accurate indirect occlusion. *SIGGRAPH 2016 Courses: Physically Based Shading in Theory and Practice*.

Kipp, M., Heloir, A., and Nguyen, Q. (2011). Sign language avatars: Animation and comprehensibility. In *International Workshop on Intelligent Virtual Agents*, pages 113–126. Springer.

Liu, E., Llamas, I., Kelly, P., et al. (2019). Cinematic rendering in ue4 with real-time ray tracing and denoising. In *Ray Tracing Gems*, pages 289–319. Springer.

McDonald, J., Wolfe, R., Schnepp, J., Hochgesang, J., Jamrozik, D. G., Stumbo, M., Berke, L., Bialek, M., and Thomas, F. (2016). An automated technique for real-time production of lifelike animations of american sign language. *Universal Access in the Information Society*, 15(4):551–566.

Miller, G. (1994). Efficient algorithms for local and global accessibility shading. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 319–326.

Phong, B. T. (1975). Illumination for computer generated pictures. *Communications of the ACM*, 18(6):311–317.

Reilly, J. S., McIntire, M. L., and Bellugi, U. (1990). Faces: The relationship between language and affect. In *From gesture to language in hearing and deaf children*, pages 128–141. Springer.

Ritschel, T., Grosch, T., and Seidel, H.-P. (2009). Approximating dynamic global illumination in image space. In *Proceedings of the 2009 symposium on Interactive 3D graphics and games*, pages 75–82.

Scherson, I. D. and Caspary, E. (1987). Data structures and the time complexity of ray tracing. *The Visual Computer*, 3(4):201–213.

Siple, P. (1978). Visual constraints for sign language communication. *Sign Language Studies*, (19):95–110.

TheWusa. (2006). https://upload.wikimedia.org/wikipedia/commons/9/91/ambientocclusion_german.jpg licensed under the gnu free documentation license, version 1.2.

Whitted, T. (2005). An improved illumination model for shaded display. In *ACM Siggraph 2005 Courses*, pages 4–es.

Wolfe, R., Cook, P., McDonald, J. C., and Schnepp, J. (2011). Linguistics as structure in computer animation: Toward a more effective synthesis of brow motion in american sign language. *Sign Language & Linguistics*, 14(1):179–199.