Learning-by-Synthesis: Photorealistic Rendering of Eyes for Eye Registration and Gaze Estimation

removed
for
blind review

Abstract

High-quality images of the human eye are required in the study of several computer vision problems, such as facial feature detection and tracking, eye detection, or gaze estimation and correction. In particular recent large-scale supervised methods require tedious and time-consuming collection and typically manual annotation of large amounts of training images. We present a novel method to synthesise photorealistic close-up images of the human eye for arbitrary head poses, gaze directions, scales, and illumination conditions. At the core of our method is a dynamic eye-region model that we ***[briefly describe what this model involves and how it was generated]*** *The model is able to simulate the large variability of real eyes, including pupil dilation, eyelid motion and corresponding changes in its shape, as well as iris colour variations. We demonstrate the usefulness of our method on the sample problems of eyelid detection and appearance-based gaze estimation. We show that* ***[results]***

1. Introduction

Andreas: would be nice to show the process figure as a teaser above abstract as in the disney paper Erroll: Hmm I tried to abuse the title section to put the wide figure in, but this seems tricky. If you know anyone who's managed in MPII, let me know, I'm sure there's a way.

Machine learning methods that leverage large amounts of training data currently perform best for many problems in computer vision, such as object detection, scene recognition, or gaze estimation [1, 2, 3]. However, capturing or collecting large-scale training data can be extremely time-consuming, especially for new areas of research without pre-existing datasets. In addition, supervised learning methods require accurate ground truth annotation for each image. This annotation process can be expensive and tedious, and there is no guarantee that human-provided labels will be correct. Ground truth annotation is particularly challenging and error-prone for learning tasks that require highly fine-grained and accurate labels, such as of facial landmarks for facial expression analysis and gaze estimation **[REF]** or body joints for pose estimation and activity recognition **[REF]** .

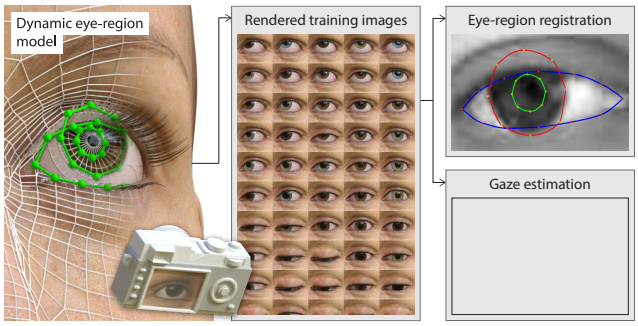


Figure 1: We render images.

To address these problems, a number of recent works explored the potential of synthesising training images at a large scale. The distinct advantages of this approach are that both data collection and annotation only require little human labour and image synthesis can be modified and geared to specific application scenarios. For example, **[add example works without face and eye]** One area of the human body largely neglected in the context of image synthesis so far is the face and particularly the eyes. This is partly due to these areas being particularly difficult to model accurately given the large amount of muscles and thus complex dynamic shape changes during different facial expressions and eyeball rotations, the large number of subtle details in appearance and texture around and in the eye, as well as the significant variation in general face and eye appearance across different people. **[final sentence on limitations of existing models in this area]**

In this paper we present a novel method for photorealistic rendering of full face and eye images at a large scale. Our method **[briefly summarise key characteristics**

of the method] We describe how we prepare a collection of dynamic eye-region models, and then our approach for generating large amounts of photorealistic training data. We then present and evaluate two separate systems trained on SynthesEyes: a novel eye-region specific deformable model and an appearance-based gaze estimator. These systems are case studies that show how we leverage the degrees of control made available by rendering our training data to easily and quickly generate high quality training datasets. We show that by rendering all our training data, we achieve state of the art performance...

The specific contributions of this work are threefold. First, **[...]**

2. Related Work

Our work is related to previous works on 1) learning using synthetic data as well as 2) computer generated eyes.

2.1. Learning Using Synthetic Data

[4] – relit 3d face scans to study the effect of illumination on automatic expression recognition.

[5] – learning by synthesis for 3D gaze estimation

[6]

[7] – train head pose estimator on only synthetic depth data.

2.2. Computer Generated Eyes

Andreas: the ruhland2014look reference contains lots of additional relevant references [8]

Andreas: could also cover geometric eye models, e.g. as used in eye tracking

The eyeballs are complex organs comprised of multiple layers of tissue, each with different reflectance properties and levels of transparency. Fortunately, given that rendering realistic eyes is important for many areas in computer graphics (CG), there is already a large body of previous work on modelling and rendering eyes (see [8] for a recent survey).

[9] – very early work on synthesising eye images

[10] – state of the art Disney approach

[11] – 3D biomechanical model of the human extra-ocular eye muscles

[12] – uses rendered videos of eyes to evaluate eye tracking algorithms.

3. Our Dynamic Eye-Region Model

[past tense] **Andreas: we have to explain what makes the model dynamic Erroll: Yes, I figure we should stress this more. I called it "dynamic" as previous work (Z-Face) has photorealistically rendered static scans for training (though doesn't discuss it in detail). We made our scans dynamic to capture important continuous variability in eye-region motion, rather than relying on discrete separate scans. I will work on this later today (16/04) Andreas: could be nice to show existing renderings and ours side by side, e.g. Leszeks and others (if any)**

(a) 3D eye model (b) Pupil dilation and iris color variation

Figure 4: Our realistic eye model is capable of expressing degrees of variability seen in real life.

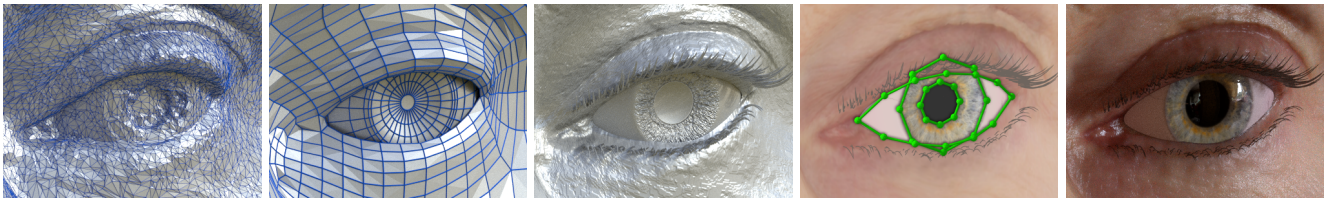
In this section we first present our anatomically inspired computer graphics eyeball model, and then explain our novel procedure for preparing a collection of 3D head scans for dynamic photorealistic labelled training data generation.

3.1. Simplified Eyeball Model

As shown in Figure 4a, our eye model consists of two parts. The outer part (red wireframe) approximates the eye's overall shape with two spheres ($r_1 = 12\text{mm}, r_2 = 8\text{mm}$ [8]), the latter representing the corneal bulge. To avoid a discontinuous seam between spheres, their meshes were joined, and the vertices along the seam where smoothed to minimize differences in face-angle. This outer part is transparent, refractive ($n = 1.376$), and partially reflective. The sclera's bumpy surface variations are modelled with smoothed solid noise functions, and applied using a *displacement map* – a 2D scalar function that shifts a surface in the direction of its normal [13]. The inner part (blue wireframe) is a flattened sphere with Lambertian material. The planar end represents the iris and pupil, and the rest represents the sclera – the white of the eye. There is a 0.5mm gap between the outer and inner parts which accounts for the thickness of the cornea. **Erroll: compare with recent Disney work**

Eyes exhibit variation in both shape (pupillary dilation) and texture (iris color and scleral veins). To model shape variation we use *blend shapes* – an animation technique where several different poses are created for the same topological mesh, and then interpolated between[14]. We created blend shapes representing dilated and constricted pupils, as well as large and small irises to account for a small amount (10%) of variation in iris size. Blend shapes are localized so can be mixed, so we can easily model an eye with a small pupil, and a large iris. We vary the texture of the eye by compositing images in three separate layers:

Figure 2: Our suite of female and male head models for rendering.



(a) Original 3D head scan data: 1.4 million polys (b) Retopologized head model: 9 thousand polys (c) Surface detail is stored with displacement maps (d) 3D iris and eyelid landmarks are annotated (e) The final render

Figure 3: Model preparation process

i) a *sclera* layer representing the tint of the sclera (white, pink, or yellow); *ii*) an *iris* layer with four photo-textures of different colored irises (amber, blue, brown, grey); and *iii*) a *veins* layer which varies between blood-shot and clear. We matched the sclera tint to each separate face model but uniformably randomly varied iris color.

3.2. 3D Head Scan Acquisition

For an eye-region rendering to be realistic, it must also feature realstic nearby face detail. While previous approaches used lifelike artist-created models, for example [12], we instead rely on high quality head scan data captured by a professional photogrammetry studio (10K diffuse color textures, 0.1mm resolution geometry) [15]. Nowadays it is possible to cheaply purchase such head scans online[1] (from $\sim$ \$15/scan), or use free or commercial photogrammetry software to generate facial geometry models in-house. A good set of trianing images should exhibit the types of appearance variability seen in real life, therefore our collection of head-models (see Figure 2) covers both genders with a variety of ethnicities and ages.

3.3. Eye-Region Geometry Preparation

As can be seen in Figure 3a, the cornea has been incorrectly reconstructed by the optical scanning process. This is because transparent surfaces are not directly visible, so cannot be reconstructed in the same way as diffuse surfaces, such as skin. Recent work used a hybrid reconstruction method to reconstruct the corneal surface separately, but requires additional hardware [10] – this level of detail was deemed unnecessary for our purposes. We wanted to render eye-region images representing a wide range of eye-gaze directions, so we needed to be able to pose the eyeball separately from the face geometry. We therefore removed the scanned eyeball from the mesh using boolean operations, and placed our own eyeball approximation in its place.

While the original head scan geometry is suitable for being rendered as a static model, its topology cannot easily represent dynamic changes in eye-region shape. Vertical saccades are always accompanied by eyelid motion [16], so we needed to be able to pose the eyelids according to the gaze vector. When preparing a mesh for facial animation, edge loops should flow along and around the natural contours of facial muscles. This leads to a more efficient (low-resolution) geometric representation of the face, and more realistic animation as mesh deformation matches that of actual skin tissue and muscles.

We therefore *retopologized* the face geometry into a more optimal form using a commercial semi-automatic system [17]. As can be seen in Figure 3b, edge loops now follow the *Orbicularis Oculi* muscle, allowing for realistic eye-region deformations. This retopologized low-poly mesh lacks the detail of the original scan (e.g. the crease above the eye), and has visible sharp edges. We therefore used it as the control mesh for a displaced subdivision surface [13], with displacement map computed from the scanned geometry. As can be seen in Figure 3c, skin surface detail like wrinkles and creases is restored. Although they are two separate organs, there is normally no visible gap between eyeball and skin. However, as a consequence of removing the eyeball from the original scan, the retopologized mesh will not necessarily meet the geometry of our eyeball model (see Figure 3b). To compensate for this, the face mesh's

[1] Ten24 3D Scan Store – http://www.3dscanstore.com/

Figure 5: Eyelids are posed using blend shapes linked to the gaze vector. Note how we simulate the folding of the skin above and below the eye.

eyelid vertices are displaced along their normals to their respective closest positions on the eyeball geometry (see Figure 3c). This automatic operation prevents unwanted gaps between the models, even after changes in pose [18].

3.4. Modelling Eyelid Motion

When someone looks up or down, their eyelids move accordingly [16]. To simulate this we created blend shapes for upwards-looking and downwards-looking eyelids, and interpolate between them based on the global pitch of the eyeball model. Manually defining blend shapes through vertex manipulation can be a difficult, repetitive, and time-consuming task. Fortunately, only two are required, and they are localized to the eye-region. As the tissue around the eye is compressed or stretched, skin details like wrinkles and folds are either attenuated or exaggerated (see Figure 5). We modeled this by using smoother color and displacement textures for downwards-looking eyelids, removing any wrinkles. The blend shape and texture modifications were carried out using photos of the same heads looking up and down as references. However, an alternative would be to purchase the appropriate corresponding head scans and match the blend shape to that geometry.

3.5. Modelling of Eyelashes

Eyelashes are short curved hairs that grow from the outer edges of the eyelids. These can occlude parts of the eye and affect eye tracking algorithms, so are simulated as part of our comprehensive model. We followed the approach of Świrski and Dodgson [12], and model eyelashes using directed hair particle effects. The hair particles were generated from a control surface manually placed below the eyelids. To make them curl, eyelash particles experience a slight amount of gravity during growth (negative gravity for the upper eyelash).

3.6. Eye-Region Landmark Annotation

As shown in Figure 3d, each 3D eye-region was annotated in 3D with 28 landmarks, corresponding to the eye corners (2), eyelids ($5+5$), iris boundary (8), and pupil boundary (8). The iris and pupil landmarks were defined as a subset of the eyeball geometry vertices, so deform automatically with changes in pupil and iris size. The eyelid and eye corner landmarks were manually labelled with a separate mesh that follows the seam where eyeball geometry meets skin geometry. This mesh is assigned shape keys and deforms automatically during eyelid motion. Whenever an image is rendered, the 2D image-space coordinates of these 3D landmarks are calculated using the camera projection matrix and saved.

4. Rendering Photo-Realistic Images

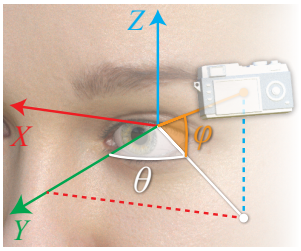
Andreas: past tense

In this section we first describe how we rendered our eye-region models, then we briefly describe how we use image-based lighting [19] to model a wide range of realistic lighting conditions, and then finally discuss the details of our rendering setup.

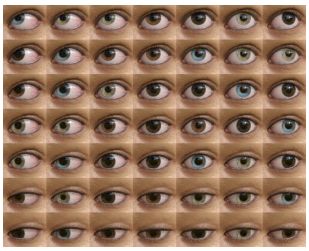
For a chosen eye-region model configuration, each rendered image is determined by parameters $(\mathbf{c}, \mathbf{g}, L)$: 3D camera position $\mathbf{c}$; 3D gaze vector $\mathbf{g}$; and lighting environment L. As shown in Figure 6, camera positions $\mathbf{c}$ were chosen by iterating over spherical coordinates (r, θ, ϕ), centered around the eyeball center. We used an orthographic camera, as this simulates an eye region-of-interest being cropped from a wide-angle camera image, so we set $r = 1$ for convenience. At each camera position $\mathbf{c}$, we rendered multiple images with different gaze vectors to simulate the eye looking in different directions. Examples with fixed L are shown in Figure 6b. Gaze vectors $\mathbf{g}$ were chosen by first pointing the eye directly at the camera (simulating eye-contact), and then modifying the eyeball's pitch (α) and yaw (β) angles over a chosen range. A consequence of this approach is the possibility of rendering "unhelpful" images that either simulate impossible scenarios or are not useful for training. To avoid violating anatomical constraints, we only rendered images for valid eyeball rotations $|\alpha| \leq 25°$ and $|\beta| \leq 35°$ [20]. Before rendering, we also verified that the projected 2D pupil center in the image was within the 2D polygon of the eyelid landmarks – this prevented us from rendering images where too little of the iris was visible.

In this section we first describe how we rendered our eye-region models, then we briefly describe how we use image-based lighting [19] to model a wide range of realistic lighting conditions, and then finally discuss the details of our rendering setup.

For a chosen eye-region model configuration, each rendered image is determined by parameters $(\mathbf{c}, \mathbf{g}, L)$: 3D camera position $\mathbf{c}$; 3D gaze vector $\mathbf{g}$; and lighting environment L. As shown in Figure 6, camera positions $\mathbf{c}$ were chosen by iterating over spherical coordinates (r, θ, ϕ), centered around the eyeball center. We used an orthographic camera, as this simulates an eye region-of-interest being cropped from a wide-angle camera image, so we set $r = 1$ for convenience. At each camera position $\mathbf{c}$, we rendered multiple images with different gaze vectors to simulate the eye look-



(a) The camera is positioned using spherical coordinates



(b) At each camera position, we render many gaze directions

Figure 6: 6a shows how we position the camera to simulate changes in head pose. At each camera position, we render many eye images (6b) by posing the eyeball model.

ing in different directions. Examples with fixed L are shown in Figure 6b. Gaze vectors $\mathbf{g}$ were chosen by first pointing the eye directly at the camera (simulating eye-contact), and then modifying the eyeball's pitch (α) and yaw (β) angles over a chosen range. A consequence of this approach is the possibility of rendering "unhelpful" images that either simulate impossible scenarios or are not useful for training. To avoid violating anatomical constraints, we only rendered images for valid eyeball rotations $|\alpha| \leq 25°$ and $|\beta| \leq 35°$ [20]. Before rendering, we also verified that the projected 2D pupil center in the image was within the 2D polygon of the eyelid landmarks – this prevented us from rendering images where too little of the iris was visible.

4.1. Lighting the Model

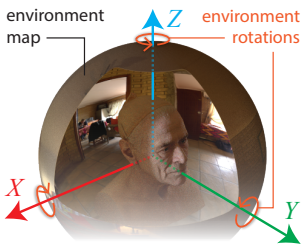
One of the main challenges in computer vision is illumination invariance – a good system should work under a range of real-life lighting conditions, even though appearance varies dramatically. We realistically illuminate our eye-model using *image-based lighting*, a technique where high dynamic range (HDR) panoramic images are used to provide light in a scene [19]. This works by photographically capturing omni-directional light information, storing it in a texture, and then projecting it onto a sphere around the object. When a ray hits that texture during rendering, it takes that texture's pixel value as light intensity. At render time we randomly chose one of four freely available HDR environment images to simulate a range of different lighting conditions (see Figure 7) [21]. Each environment image is then randomly rotated to simulate a continuous range of head-pose within each environment. As shown in Figure 7c, we can generate a wide range of different appearances with only a single HDR environment.

4.2. Computational Setup

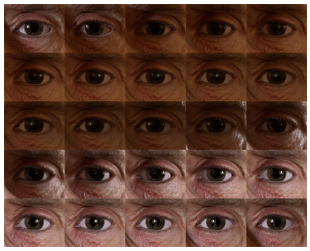
We can rapidly generate diverse datasets much faster than manual collection and annotation.

We use Blender's inbuilt Cycles path-tracing engine for

(a) The four HDR environment maps we use for realistic lighting: bright/cloudy outdoors, and bright/dark indoors



(b) The environment is rotated to simulate different head poses



(c) Renders using a single environment, rotated about Z

Figure 7: Appearance variation from lighting is modelled with poseable high dynamic range environment maps [19].

rendering **Erroll: CITE**. This physically correct Monte Carlo method integrates over light rays that hit a surface, and stochastically calculates how much of it reaches a pixel in the camera. Though it requires a large number of rays, a GPU solution ...

When rendering a 120×80 eye-region image, it took 5 seconds on average using a commodity GPU (Nvidia GTX660). A dataset for a single eye-region model can be rendered in several hours, so it can take several days to render the entire dataset of ten models. This is much faster than the months taken for traditional data collection approaches [3].

5. Experiments

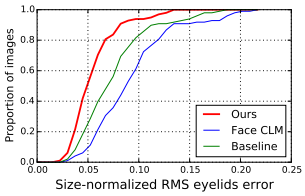
We evaluated the usefulness of our synthetic data generation method on two sample problems, eye registration and gaze estimation.

Andreas: briefly say something about significance/importance of both problems, more in corresponding subsections

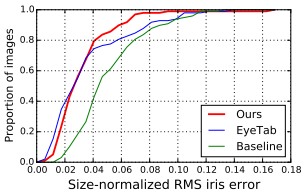
5.1. Eye Registration

Andreas: results look pretty good already, I suggest put them in asap so that we can completely draft this subsection. If results improve we can always update later but this way we have something to produce text and refine the story

- Evaluate eyelid landmark accuracy on LFW and M-PIE data, compare against several state-of-the-art CLM methods.



(a) Eyelids results



(b) Iris results

Figure 8: Results of annotated MPII experiment

While the in-the-wild images represent challenging conditions for eyelid registration, they are not representitive of typical webcam-style images and do not feature labels for the iris. We therefore annotated detailed eyelid and iris boundaries onto a subset of MPIIGaze ($n = 100$), a dataset collected during natural everday laptop use over several months [3]. We compared our eye-region CLNF with EyeTab [22], a state-of-the-art shape-based approach that robustly fits ellipses to the limbus using image-aware RANSAC [23]. We used the author's implementation, with their suggested modifications to improve performance: improved eyelid localization with a state-of-the-art facial landmark detector [24]. As a baseline, we used the mean position of all 28 eye-landmarks following model initialization. Eyelid errors were calculated as mean distances from the 12 eye corner and eyelid landmarks to ground truth eyelid boudnary. Iris errors were calculated by least-squares fitting an ellipse to the 8 iris landmarks, and then calculating mean distance between estimated and ground truth iris ellipses. The iris error distances were only calculated between the boundaries of the ground truth eyelids.

As shown in Figure 8, our approach performs comparably with state-of-the-art specialist algorithms for fitting ellipses to irises.

5.2. Gaze Estimation

We render a targeted dataset that matches MPII's gaze and pose distribution, with added 3D laptop screen emitting light. This shows how we can target specific scenarios like laptop-based gaze estimation, and render a suitable dataset within a day rather than take 3 months of data collection.

Using Xucong's CNN sytem, we train on targeted version of SynthesEyes, test on MPII. Show results are better than training on UT and testing on MPII. This shows that the range of lighting in SynthesEyes is important for better results.

6. Conclusion

References

[1] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning deep features for scene recognition using places database," in *Proc. NIPS*, 2014, pp. 487–495. 1

[2] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. CVPR*, 2014, pp. 580–587. 1

[3] X. Zhang, Y. Sugano, M. Fritz, and A. Bulling, "Appearance-based gaze estimation in the wild," in *Proc. CVPR*, 2015. 1, 5, 6

[4] G. Stratou, A. Ghosh, P. Debevec, and L. Morency, "Effect of illumination on automatic expression recognition: a novel 3d relightable facial database," in *Proc. FG*, 2011, pp. 611–618. 2

[5] Y. Sugano, Y. Matsushita, and Y. Sato, "Learning-by-synthesis for appearance-based 3d gaze estimation," in *Proc. CVPR*, 2014, pp. 1821–1828. 2

[6] F. Lu, Y. Sugano, T. Okabe, and Y. Sato, "Head pose-free appearance-based gaze sensing via eye image synthesis," in *Proc. ICPR*, 2012, pp. 1008–1011. 2

[7] G. Fanelli, J. Gall, and L. Van Gool, "Real time head pose estimation with random regression forests," in *Proc. CVPR*, 2011, pp. 617–624. 2

[8] K. Ruhland, S. Andrist, J. Badler, C. Peters, N. Badler, M. Gleicher, B. Mutlu, and R. Mcdonnell, "Look me in the eyes: A survey of eye and gaze animation for virtual agents and artificial systems," in *Proc. Eurographics*, 2014, pp. 69–91. 2

[9] G.-C. Feng and P. C. Yuen, "Variance projection function and its application to eye detection for human face recognition," *Pattern Recognition Letters*, vol. 19, no. 9, pp. 899–906, 1998. 2

[10] P. Bérard, D. Bradley, M. Nitti, T. Beeler, and M. Gross, "Highquality capture of eyes," *ACM Transactions on Graphics*, vol. 33, no. 6, p. 223, 2014. 2, 3

[11] A. Priamikov and J. Triesch, "Openeyesim - a platform for biomechanical modeling of oculomotor control," in *Proc. ICDL-Epirob*, Oct 2014, pp. 394–395. 2

[12] L. Świrski and N. Dodgson, "Rendering synthetic ground truth images for eye tracker evaluation," in *Proc. ETRA*, 2014, pp. 219–222. 2, 3, 4

[13] A. Lee, H. Moreton, and H. Hoppe, "Displaced subdivision surfaces," in *Proc. SIGGRAPH*, 2000, pp. 85–94. 2, 3

[14] V. Orvalho, P. Bastos, F. Parke, B. Oliveira, and X. Alvarez, "A facial rigging survey," in *Proc. Eurographics*, 2012, pp. 10–32. 2

[15] J. Busby and C. Rawlinson, "Face scanning," http://www.ten24.info/?page_id=299, 2015. 3

[16] S. Liversedge, I. Gilchrist, and S. Everling, *The Oxford handbook of eye movements*. Oxford University Press, 2011. 3, 4

[17] Pixologic, "Zbrush zremesher 2.0 automatic retopology taken to a new level," http://docs.pixologic.com/user-guide/3d-modeling/topology/zremesher/, 2015. 3

[18] The Blender Foundation, "The shrinkwrap modifier," http://wiki.blender.org/index.php/Doc:2.6/Manual/Modifiers/Deform/Shrinkwrap, 2015. 4

[19] P. Debevec, "Image-based lighting," *IEEE Computer Graphics and Applications*, vol. 22, no. 2, pp. 26–34, 2002. 4, 5

[20] *MIL-STD-1472G Design Criteria Standard: Human Engineering*, Department of Defence, USA, January 2012. 4, 5

[21] G. Zaal, "Hdr panoramas," http://adaptivesamples.com/category/hdr-panos/, 2015. 5

[22] E. Wood and A. Bulling, "Eyetab: Model-based gaze estimation on unmodified tablet computers," in *Proceedings of the Symposium on Eye Tracking Research and Applications*. ACM, 2014, pp. 207–210. 6

[23] L. Świrski, A. Bulling, and N. Dodgson, "Robust real-time pupil tracking in highly off-axis images," in *Proceedings of the Symposium on Eye Tracking Research and Applications*. ACM, 2012, pp. 173–176. 6

[24] T. Baltrusaitis, P. Robinson, and L.-P. Morency, "Constrained local neural fields for robust facial landmark detection in the wild," in *Computer Vision Workshops (ICCVW), 2013 IEEE International Conference on*. IEEE, 2013, pp. 354–361. 6