Synthetic Skies Using High Dynamic Range Images and Eigenskies

B. A. Olsson, A. Ynnerman and R. Lenz
Linköping University, Sweden

Abstract

This paper presents a method to render synthetic sky images corresponding to given weather conditions. The method combines artificial neural networks and principal component analysis to associate the appearance of the sky with the state of a weather parameter vector. This association is then used to generate artificial sky images corresponding to a given weather parameter vector. The proposed method has important applications for example in flight simulators and in the game industry.

The skies are represented by high-dynamic-range images which are able to store the dynamic properties of sky light. This representation can be used for global illumination in software packages such as Radiance to render scenes at arbitrary lighting conditions. The results show that, although the cloud details can not be represented by this method it is possible to distinguish between different weather states.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Color, shading and texture.

Keywords: Skylight, illumination and weather visualization.

1 Introduction

Lighting conditions and the appearance of the sky in outdoor scenes have a fundamental effect on the way human beings perceive an environment. Just by altering the light and appearance of the sky the same place can be perceived as frightening or comforting, cold or warm etc. In the fine arts and psychology literature the importance of the sky for human perception has long been recognized [DaVinci 1970]. This paper strives to encapsulate this importance in the context of computer graphics and presents a method to render synthetic sky images corresponding to given weather conditions. The method allows us to generate virtual worlds that exhibit a rich multitude of lighting conditions and sky backdrops. There are several possible applications of synthetic skies. One obvious application is flight simulators where it is important to generate realistic impressions of weather conditions. Another application is the entertainment industry where extreme weather conditions are frequently used in both computer games and in the film industry.

Our method of choice to represent sky light is based on high-dynamic-range images (HDR images) which can store the dynamic properties of the sky light [Debevec and Malik 1997]. The focus of the paper is to present a method to synthesize HDR images of the sky corresponding to a weather parameter vector. The method uses eigenskies, based on principal component analysis (PCA) [Joliffe 1980] of HDR photographs of a real sky, to compress the input data. Artificial neural networks (ANN) are then used to tie the values of a PCA coefficient vector to a vector of weather parameters measured at the same time as the HDR image was captured. The ANN can then, given any set of weather parameters as input, generate PCA components that are used to create a synthetic HDR image. If weather parameters from a forecast are used it is for instance possible to generate an HDR image corresponding to a future weather scenario.

The general field of weather visualization can be divided into two main categories. The first contains the more traditional approaches aimed at the creation of maps, in 2D and 3D, for weather forecasting. Trivis [Haase et al. 2000] which is used in the creation of maps for broadcast weather forecasts is one example of such a tool. To generate realistic images of an environment during given weather conditions is the task of the second approach. The analytic method of [Preetham et al. 1999] to render images of sky light is one example of this approach, which is able to render a general image of the sky using information about the viewer's location on earth, the position of the sun and irradiance values. The resulting image will be an average image for this time of the year, but no information about the daily weather is included and thus the dynamic change of the weather can not be seen in this image. Related methods using basis functions to fit simulation data have been developed by other authors. A series of Legendre basis functions was used by [Dobashi et al. 1995] and steerable basis functions were used by [Nimeroff and Rushmeier 1996] to fit sky luminance model data. Methods to synthesize clouds using real-time volume rendering taking subsurface scattering effects into account have been developed by [Riley et al. 2003]. The method described in this paper falls into this second category but will in contrast to other methods be able to render images which vary with the actual weather state.

To show the results of the method an application of the synthetic HDR images to define the lighting conditions in the rendering tool Radiance [Ward 1994] is described.

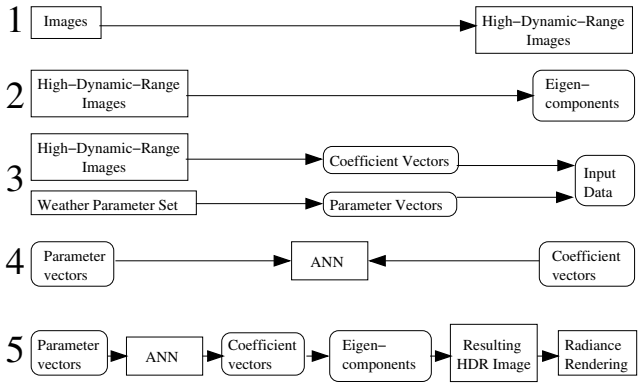

Figure 1: A diagrammatic representation of the main steps in the method. The method can be divided in five steps: Computation of HDR-images, computation of eigencomponents, data collection, training and synthesizing.

2 Method

Instead of explicitely defining the appearance of the sky in various weather conditions we collect images from different weather states. These images are then used to train the model. The method uses principal component analysis (PCA) to compress the data and learn the most important features of the images. The eigenface [Turk and Pentland 1991; McGuire and D'Eleuterio 2001] method inspired this work. The relation between the weather parameters and the image features is learned by ANN. We thus use ANN as a statistical method for function approximation. Our method is based on a combination of PCA and neural networks with supervised training.

In our approach we first capture a series of images with varying shutter speeds at each time interval (An image series can be seen in Figure 3). One series is used to construct one HDR image. The method used for computing HDR images is described in detail in [Debevec and Malik 1997]. The resulting fisheye images define the irradiance in every direction in Euclidean space.

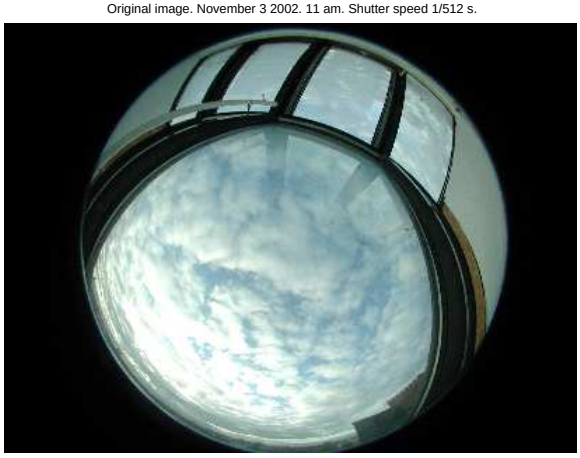

Figure 2: This is one example of the original fisheye images of size 1500x2000 pixels. The images used are captured of exactly the same scene by a camera located in a roof window. Some parts of the field of vision are obstructed. These parts are interpolated using radial symmetry.

2.1 Principal Component Analysis

Next we use PCA to decompose HDR images as follows: The pixels in an image are rearranged into a vector $\bar{x} = [x_1, x_2, \ldots, x_N]$ of length N. We assume that we have M such vectors. The transpose is denoted $\bar{x}^T$. The mean vector $\overline{m_x}$ is defined as $\overline{m_x} = E(\bar{x})$ and the covariance matrix is $\mathbf{C_{xx}} = E((\bar{x} - \overline{m_x}) * (\bar{x} - \overline{m_x})^T)$, where $E(.)$ means the expectation value of a stochastic variable. The eigenvalue problem is then defined $\mathbf{C_{xx}V} = \mathbf{VD}$, where $\mathbf{V}$ is a matrix with the eigenvectors as columns and $\mathbf{D}$ a diagonal matrix with the eigenvalues along the main diagonal.

For the resolution needed in this project the size of the covariance matrix becomes impractical. The size of the matrix can be decreased by using dimensionality reduction [Haykin 1999]. Let $\mathbf{Y}$ be the rectangular data matrix of size NxM. The matrix has the same height as the number of vectors (M). The eigenvalue problem, $\mathbf{C_{xx}V} = \mathbf{VD}$, can be written as

$$\mathbf{C_{xx}} = \frac{1}{N^2}\mathbf{Y^T Y} \Rightarrow \frac{1}{N^2}\mathbf{Y^T YV} = \mathbf{VD} \qquad (1)$$

The size of the covariance matrix is reduced from a size of NxN elements to a size of MxM elements by multiplying with $\mathbf{Y}$ from the left and changing variable $\mathbf{W} = \mathbf{YV}$.

$$\frac{1}{N^2}\mathbf{YY^T YV} = \mathbf{YVD} \Rightarrow \frac{1}{N^2}\mathbf{YY^T W} = \mathbf{WD}. \qquad (2)$$

The eigenvectors are the eigenskies. New sky images can be generated from them by linear combination. The weights are computed from weather parameters using ANN's.

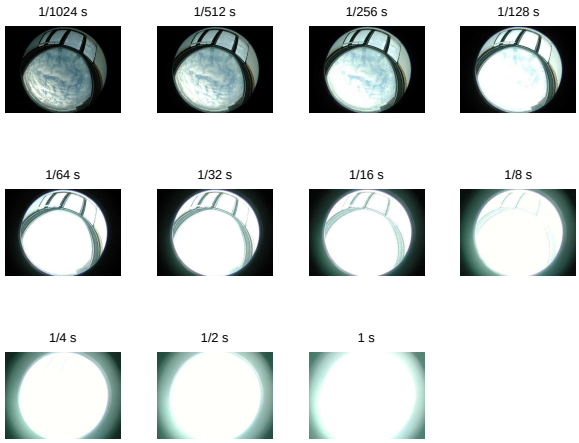

Figure 3: In order to compute a High-Dynamic-Range image a series of images is captured with varying shutter speeds. The first image is captured with a shutter speed of $1/1024$ s. For every image in the series the shutter speed is doubled until 1 s which is the last exposure.

2.2 Artificial Neural Networks

Artificial neural network is a collective term for a number of connected but in many cases very different methods. Neural networks originate from neuroscience as a historical method of simulating the brain neurons. This technology has found applications in diverse areas such as pattern recognition and signal processing. For a more detailed description of this subject see [Hagan et al. 1995] or [Haykin 1999].

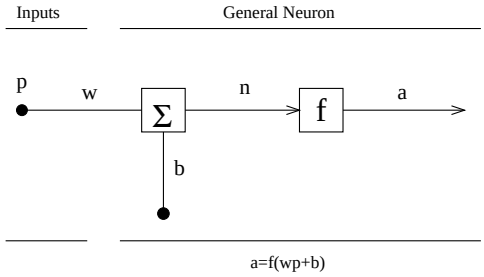

Figure 4: The single-input neuron. An activation function f is applied on the sum of the bias and the input p multiplies with a weight w.

ANN methods are used for function approximation in this paper. In the simplest form an ANN consists of one neuron (See Figure 4). The scalar input p is multiplied by the scalar weight w and then added to parameter b. The neuron output is calculated as $a = f(wp + b)$. Parameters w and b are adjusted by some learning rule. The activation function f may be linear or nonlinear. Examples of activation functions include the hard-limit activation function (also called the step function, $f(x) = \{0, x < 0; 1, x >= 0\}$),

the linear activation function and the log-sigmoid activation function ($f(x) = \frac{1}{1+e^{-x}}$). In a single layer network several neurons are all connected to the same input-signals. The mathematical formulation is identical to the single-neuron case, $\overline{a} = f(\mathbf{W}\overline{p} + \overline{b})$, but now $\mathbf{W}$ is a matrix and $\overline{a}$, $\overline{p}$ and $\overline{b}$ are vectors. A multiple layer network is built by connecting several single layer networks. In this paper we use two layers of the form

$$\overline{a} = f_2(\mathbf{W}_2 f_1(\mathbf{W}_1 \overline{p} + \overline{b_1}) + \overline{b_2}). \quad (3)$$

The first layer is called the hidden layer and the units of this layer are called the hidden units. The second layer is called the output layer. Matrices $\mathbf{W}_1$ and $\mathbf{W}_2$ and vectors $\overline{b_1}$ and $\overline{b_2}$ are the adjustable parameters in the two-layer case. Many different training methods exist but most are variations of the backpropagation algorithm [Rumelhart et al. 1986]. We used the Levenberg-Marquardt algorithm [Scales 1985], which is a variation of Newton's method.

The algorithm is provided with a set of examples of proper network behaviour:

$$(\overline{p_1}, \overline{t_1}), (\overline{p_2}, \overline{t_2}), \ldots, (\overline{p_x}, \overline{t_x}), \ldots, (\overline{p_n}, \overline{t_n}), \quad (4)$$

where $\overline{p_x}$ is an input to the network and $\overline{t_x}$ is the corresponding target output. The network output is compared to the target as each input is applied to the network. The network parameters are adjusted by the algorithm in order to minimize the mean squared error.

Data from two different types of data sets were used. *HDR fisheye images*, computed from eleven fisheye images with varying shutter speeds and *direct measurements of weather parameters*. Ten parameters were measured at the same time as an image was captured. The image database and the database of weather parameters were combined to train the model used to synthesize an accurate HDR fisheye image for a general set of weather parameters.

The method can be divided into five steps (See Figure 1). In the first step captured image series are used to compute high-dynamic-range representations of the sky light. The eigencomponents are computed from all the HDR-images in the second step. In order to compute well-defined eigenskies a large number of images at different weather conditions from a large time interval is needed. Every image is projected to the most important eigenskies and by this method a PCA-coefficient vector is computed. The vector represents a compressed version of the original image.

This vector is then combined with a weather parameter vector in the fourth step to form a data pair which is used to train the model. By using PCA the image data is compressed to a small number of coefficients. The first part of a data pair is an example vector of the input which in this case is a vector with the actual weather parameters at a certain time. The second part is the expected output at this input vector, a small number of PCA coefficients. We used twentyfour neural networks, one for each hour of the day, but the eigenskies were identical for all networks. In the fifth and final step, the synthesizing step, a weather parameter vector is sent to the neural network which generates a coefficient vector. This coefficient vector is used together with the eigenskies to synthesize a resulting image which is used in a Radiance rendering to define the sky light for this particular set of weather parameters.

3 Results

3.1 HDR images

An off-the-shelf digital camera (Nikon Powershot 990) has been mounted in a roof window. This camera captures an image series consisting of eleven images with varying shutter speeds (1/1024 s, 1/512 s, ..., 1/2 s, 1 s) every hour. An example image can be seen in Figure 2. Images from a time-period of seven months were used in this work. A C++ program to convert an image series to an HDR image has been developed. An image series of original HDR images is located in Figure 6. The original images were downscaled to a size of (150x200) pixels to lower the memory-demand. The ten most important eigenskies were used in the calculations.

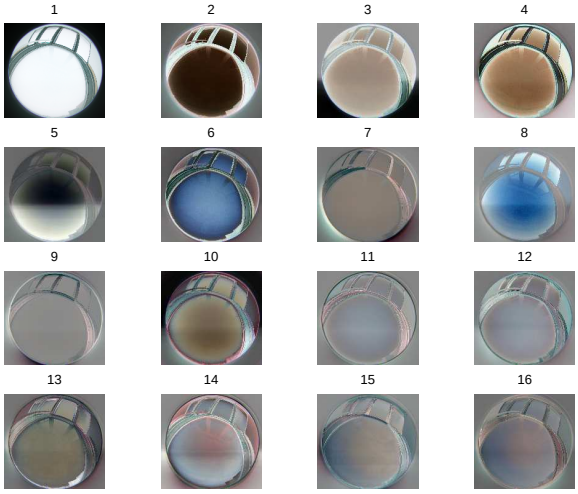

Figure 5: The sixteen most important eigencomponents - the eigenskies. Note the small artifacts in the middle of the images which are caused by reflections. The most important component, in the upper left corner, is a grey sky, which means that mostly the sky is grey.

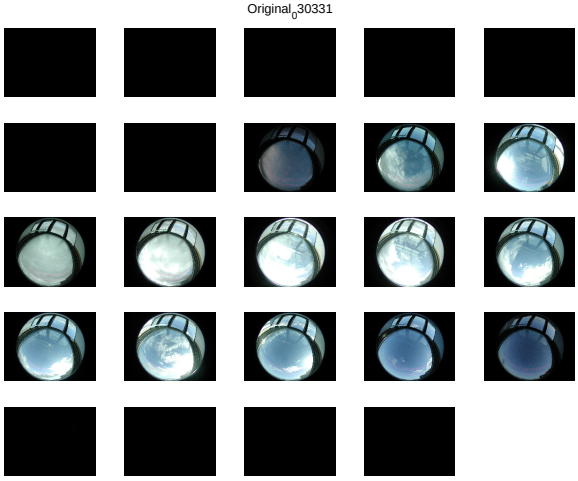

Figure 6: An image series of High-Dynamic-Range images. There is one image for each hour of the 24 hour day. This series was captured March 31, 2003.

3.2 Weather parameter vector

Direct measurements of the weather parameters were used for the training of the neural networks. These parameters were measured by an automatic weather station every hour at a position situated two kilometers from the camera. Ten parameters are measured of which temperature, pressure, date, time of the day, derivatives of pressure and temperature were used.

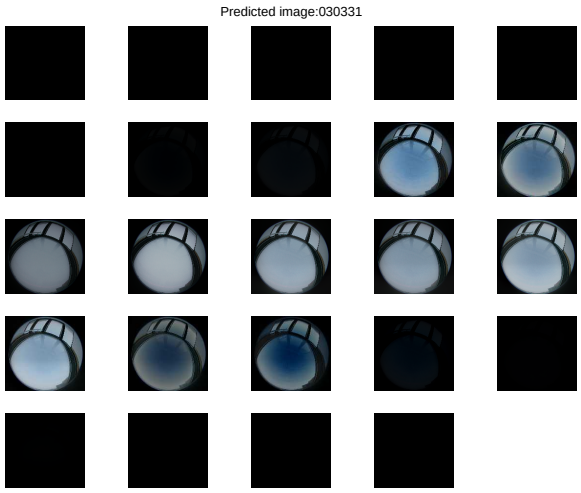

Figure 7: Images which have been synthesized using the method described in this paper. There is one image for each hour of the day. The neural networks were trained with seven months of image data and weather parameters. The weather parameters for March 31, 2003 were used to render these images. This day was not included in the training set.

3.3 The resulting images

All images were used to compute the eigencomponents (See Figure 5). The database of images was divided into twentyfour parts and each neural network was trained with data from one specified hour of the day. All networks were optimized using the available data. Most of the data was used, as a training set, to train the neural networks, but a number of data points were saved and used, as a test set, to verify the output result. The process to synthesize images was initiated by sending a weather parameter vector to the networks. A PCA-coefficient vector was predicted by one of the neural networks, depending on which time of the day the data point belonged to. The resulting image was computed by using this vector in combination with the eigenskies. This image represented a view of the weather at this specific data point. For every hour an ANN was trained to predict the eigensky coefficients from the temperature, pressure, time, time derivative of temperature, time derivative of pressure and date entries of the measurement vector. Twentyfour neural networks were used to simplify the training process of the individual networks. All images were transformed to coefficient vectors using the ten most important eigenskies. The two-level network used had a linear activation function on the hidden-layer and a log-sigmoid activation function on the second layer. All networks were trained until no further change could be seen. It was found that six hidden units gave a good compromise between flexibility and the risk of over-training. We have developed a software package in MATLAB [Olsson et al. 2003] that computes the eigenskies and trains an ANN to synthesize HDR-images. HDR images of the light field can then be synthesized from the artificial neural networks using weather data sets (See examples in Figure 7).

3.4 Results from Radiance

The synthesized HDR images can be used to define the light field in the *Radiance* [Ward 1994] system. HDR images of the sky light are synthesized by the model using weather parameter vectors as input. The resulting image is a map of the lighting conditions in the upper half sphere. Radiance is then used to render a general scene with the lighting conditions defined by the HDR-image. A resulting image series can be seen in Figure 8. The missing data was interpolated assuming radial symmetry.

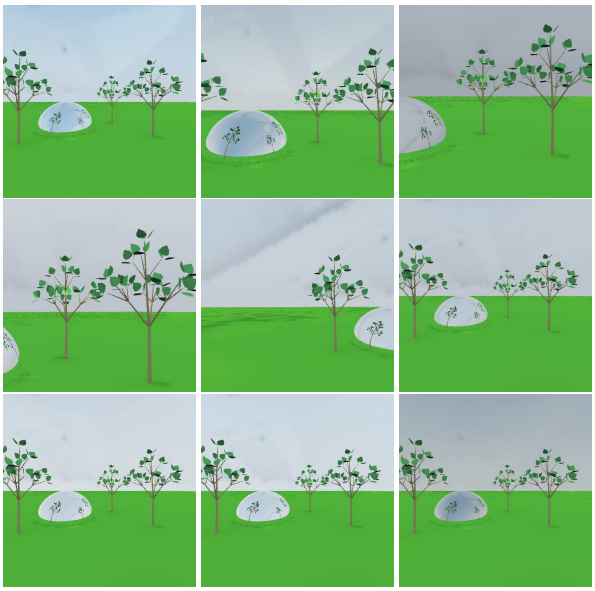

Figure 8: Examples of the result possible using the presented method in combination with Radiance. The High-Dynamic-Range fisheye images in Figure 7 were used to define the sky-light in a Radiance rendering. These images are the result from Radiance for 9 am until 5 pm. As can be seen the sky is varying during the day. Some small artifacts exist in the images due to internal reflections and interpolation errors.

4 Discussion

The synthesized HDR images have less detail than captured images due to the compression to a small number of coefficients. For example the cloud details are not represented correctly, but the main apperance of the sky is correct. The presented method should be seen as an alternative to the analytic method [Preetham et al. 1999] to synthesize sky light. While the analytic method of Preetham et al. is able to synthesize ab initio a realistic image based on average values of the irradiance depending on the position of the sun it is not able to produce an image varying with the weather conditions. The method presented in this paper is on the other hand able to synthesize an image, based on real photographs which varies depending on weather information for a specific instance in time. Although some differences in the intensity levels between the computed and the captured images exist, this method should prove to be a valuable aid in synthesizing sky images varying with the weather conditions. The obstructions in the images due to the camera location has been a major problem but this will be improved with an outdoor camera. A minor drawback is the loss of detail due to the time interval between the different exposures in the image series. No visible artifacts can be seen in the resulting images depending on this delay.

5 Future Work

Future work include increasing the number of parameters used in the neural network training to increase the precision of the prediction. Another future task is to generalize this method to synthesize sky images from parameters in weather forecast files and to combine this method with a method to compute realistic clouds from

parameters included in the forecast data sets.

6 Acknowledgements

The authors acknowledges the financial support of SMHI (Swedish Meteorological and Hydrological Institute) and NGSSC (National Graduate School in Scientific Computing). We thank SMHI for making the weather data available.

References

CATS, G., AND WOLTERS, L. 1997. The hirlam project. In *IEEE Computational Science and Engineering*, vol. 4, 22–31.

DAVINCI, L. 1970. *The Notebooks of Leonardo Da Vinci*. Dover.

DEBEVEC, P. E., AND MALIK, J. 1997. Recovering high dynamic range radiance maps from photographs. In *Proceedings of SIGGRAPH 97, Computer Graphics Proceedings*, 369–378.

DOBASHI, Y., NISHITA, T., KANEDA, T., AND YAMASHITA, H. 1995. Fast display method of sky color using basis functions. In *Pacific Graphics '95*.

HAASE, H., BOCK, M., HERGENROTHER, E., KNOPFLE, C., KOPPERT, H. J., SCHRODER, F., TREMBILSKI, A., AND WEIDENHAUSEN, J. 2000. Meteorology meets computer graphics - look at a wide range of weather visualizations for diverse audiances. *Computer & Graphics 24*, 391–397.

HAGAN, M. T., DEMUTH, H. B., AND BEALE, M. 1995. *Neural Network Design*. PWS Publishing Company.

HAYKIN, S. 1999. *Neural Networks*. Prentice Hall.

HIBBARD, A., AND SANTEK, D. 1990. The vis-5d system for easy interactive visualization. vol. 462, 28–35.

JOLIFFE, I. T. 1980. *Principal Component Analysis*. Springer.

MCGUIRE, P., AND D'ELEUTERIO, G. M. T. 2001. Eigenpaxels and a neural-network approach to image classification. *IEEE Transactions on neural networks 12*, 625–635.

NIMEROFF, J. DORSEY, J., AND RUSHMEIER, H. 1996. Implementation and analysis of an image-based global illumination framework for animated environments. *IEEE Transactions on Visualization and Computer Graphics 2*, 4.

OLSSON, B., YNNERMAN, A., AND LENZ, R. 2003. Skyvis, an application of matlab in meteorological visualization. In *Proceedings of Nordic MATLAB Conference 2003*, 295–300.

PREETHAM, A., SHIRLEY, P., AND SMITS, B. 1999. A practical analytic model for daylight. In *Proc. SIGGRAPH*.

RILEY, K., EBERT, D., HANSEN, C., AND LEVIT, J. 2003. Visually accurate multi-field weather visualization. In *Proceedings of IEEE Visualization 2003*.

RUMELHART, D. E., HINTON, G. E., AND WILLIAMS, R. J. 1986. Learning representations by back-propagating errors. *Nature 323*, 533–536.

SCALES, L. E. 1985. *Introduction to Non-Linear Optimization*. Springer-Verlag.

TURK, A., AND PENTLAND, A. P. 1991. Face recognition using eigenfaces. *J. Cognitive Neurosci. 3*, 71–86.

WARD, G. J. 1994. The radiance lighting simulation and rendering system. In *Proceedings of SIGGRAPH'94*, ACM Press / ACM SIGGRAPH, ACM, 459–472.