Deep Synthesis of Cloud Lighting

P. Satilmis
Birmingham City University, UK

D. Marnerides
University of Warwick, UK

K. Debattista
University of Warwick, UK

T. Bashford-Rogers
University of the West of England, UK

***Abstract*—Current appearance models for the sky are able to represent clear sky illumination to a high degree of accuracy. However, these models all lack a common feature of real-skies: clouds. These are an essential component for many applications which rely on realistic skies, such as image editing and synthesis. While clouds can be added to existing sky models through rendering, this is hard to achieve due to the difficulties of representing clouds and the complexities of volumetric light transport. In this work, an alternative approach to this problem is proposed whereby clouds are synthesized using a learned data-driven representation. This leverages a captured collection of High Dynamic Range cloudy sky imagery, and combines this dataset with clear sky models to produce plausible cloud appearance from a coarse representation of cloud positions. This representation is artist controllable, allowing for novel cloudscapes to be rapidly synthesized, and used for lighting virtual environments.**

Index Terms: Deep learning, Skies, Clouds.

REALISTIC whole sky appearance is crucial for many applications in image processing and computer vision, especially since the sky is the dominant light source in most outdoor environments. This is essential for creation of synthetic data for dataset generation for a wide range of imaging applications. Sky illumination can be represented in many ways, with the most common approach being environment maps [1] which are typically High Dynamic Range (HDR) images that represent distant lighting over the sphere of incoming directions in a scene. There has been extensive work on modelling clear sky appearance in a manner suitable for encoding into an environment map [2], however, these do not represent most skies observed in reality as they do not implicitly consider clouds.

Clouds can be rendered into environment maps by rendering clouds into these clear sky models, although this typically requires an expensive modelling and rendering process. The results of this are also often unrealistic due to important subtleties of realistic cloud structure [3] and light transport within clouds [4], [5], and slow to compute due to the requirements of a Monte Carlo simulation of light transport within and between clouds.

We propose a different approach for predict-

ing whole sky appearance that includes clouds by directly synthesising clouds into an existing clear sky environment map. This network encodes lighting both locally and globally over the sky via a convolutional neural network (CNN) termed CloudNet, and encodes the important dependence on sun direction on the appearance of clouds. Cloud positions are specified via a binary mask that additionally allows the positions and shape of clouds to be easily modified. This mask is pre-processed into a distance field before input into the network to efficiently encode the scale of the clouds. This network is trained on a captured dataset of 360 images consisting of a wide range of cloud configurations.

The method and an example of its use is illustrated in Figure 1. An artist sketches the location of clouds onto a black background (cloud map) and selects settings for the sky model, including sun location and atmospheric conditions. The sky model generates a sky map and a distance field map is generated from the cloud map. Both of these are input into CloudNet to produce the synthesised output environment map. This map can then be used to illuminate virtual environments.

We summarise our main contributions as follows:

- A novel approach for whole-sky cloud synthesis into environment maps.
- Artistic control of cloud placement while preserving realistic appearance.
- Results showing the ability of the proposed method to represent realistic local and global cloud illumination.

Background and Related Work

Synthesis of atmospheric phenomena have been tackled from several perspectives. These can be broadly classified into two categories: atmospheric light transport and lighting considering clouds. While these are closely related, and light transport models that are typically used in image synthesis applications apply both to atmospheres and clouds, most existing research has tackled these independently.

Clear Sky Lighting

Models for clear sky illumination consider light originating at the sun, then scattering through the atmosphere before eventually arriving at an observer based on a description of the constituents of the atmosphere. We briefly review these models as any of these can form one of the inputs to our approach.

An early model applied to computer graphics was introduced by Perez et al. [6] which can model different sky scenarios with several parameters. Further work improved the model to represent more complex light transport and wider range of skies. A comparison of these methods can be found in the research by Kider et al. [7] which evaluated measurements of real skies and showed that existing models are able to represent atmospheric lighting to a good level of accuracy, although all methods deviated from real-world values in different situations.

The state-of-the-art in sky models is the work by Wilkie et al. [8] who produced and validated a atmospheric light transport simulation over several atmospheric configurations and additionally produced a fitted sky radiance, polarization and transmittance model. An alternative approach for representing atmospheric lighting is to encode it into a neural network. Satilmis et al. [9] proposed a compact Multilayer Perceptron neural network which could represent both real captured skies and the result of evaluating sky models at a single point.

Rendered Cloud Synthesis

There are multiple methods for computing lighting from clouds for use in rendering imagery. These approaches all require either a geometric or volumetric representation of the structure of the clouds, then perform a light transport simulation of the light transport within, and between clouds. While these models allow the clouds to be rendered from any viewpoint, the resulting lighting can be “baked” into an environment map for subsequent use; this comes at a significant rendering cost.

When rendering using 3D volumetric cloud representations, the most common approach is to use Monte Carlo path tracing, see [10] for more details. Deep learning has been applied to cloud rendering by Kallweit et al. [4] who train a neural network to approximate light transport within a cloud volume. These rendering methods are further complicated by non-analytic phase

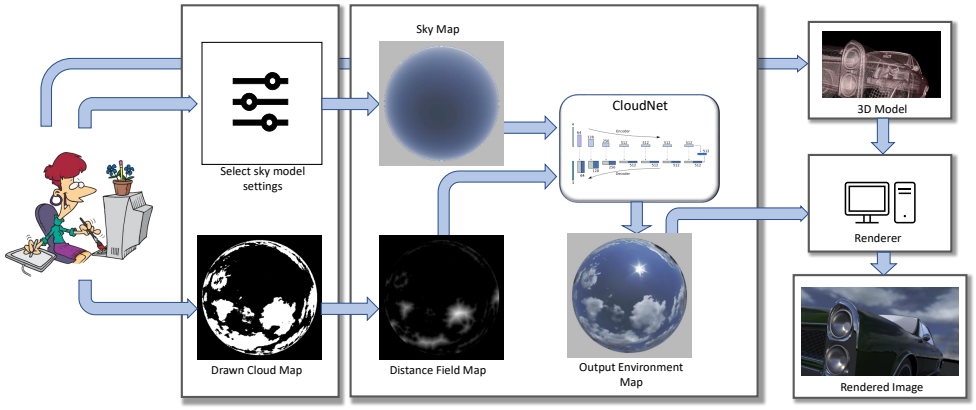

Figure 1. Framework overview.

functions, and non-exponential transport in clouds [5]. In contrast, our method avoids these issues by learning how to represent light transport across the whole sky in image space.

Deep Image Synthesis

There exist a variety of domain adaptation and texture synthesis methods that make use of deep neural networks and learn from data. Gatys et al. proposed a method for generating textures using CNNs [11] while Isola et al. [12] use Generative Adversarial Networks (GAN) [13] for image translation, adapting images from one domain to the another. This framework was later adapted for unpaired image translation [14]. Park et al. [15] introduced a spatially adaptive normalisation layer that helps generate more semantically consistent results again within the GAN framework and also allows for artist control by providing input guidance.

While some of these methods allow for artistic control, none of them is specifically designed for realistic cloud synthesis and production of environment maps. Often, a great part of the artistic control is left to the model, partially due to the use of the GAN framework, which in some cases produces results inconsistent with the artist's intention, altering the input shape in an unexpected manner. For example the work by Park et al. [15], while producing impressive results over a variety of input labels, does not reproduce outputs at the per-pixel granularity required for accurate cloud synthesis. In addition, these methods are not immediately applicable for computing images with environment lighting, as clouds are added to a single view of the skydome; applications such as rendering require lighting from the entire skydome. These methods also do not deal with HDR content, and the difficulty of directly extending these to HDR stems from requiring significant amounts of training data which is impractical to capture, and from changing the architecture of these networks to directly produce HDR values [16].

Our work differs from these approaches in that we develop a model that is purely focused on cloud and sky lighting, produces HDR results which are amenable for rendering applications, yet retains the ability for easy artistic control over cloud shapes.

Method

The overall goal of our approach is to generate whole sky HDR imagery containing realistic clouds. This section covers the design decisions and details behind the representation of input data, the structure of the CNN, and the acquisition and processing of the data required to train our system. Figure 1 shows a summary of our pipeline.

Cloud and Sky Representation

Our method requires two inputs: one being a representation of the clear sky illumination without clouds, and the second being a representation of the placement of clouds in the sky. We use the method proposed by Hosek and Wilkie [2] for the clear sky input, but any other clear sky model could be used. The method generates HDR input values, as required, and is able to accurately

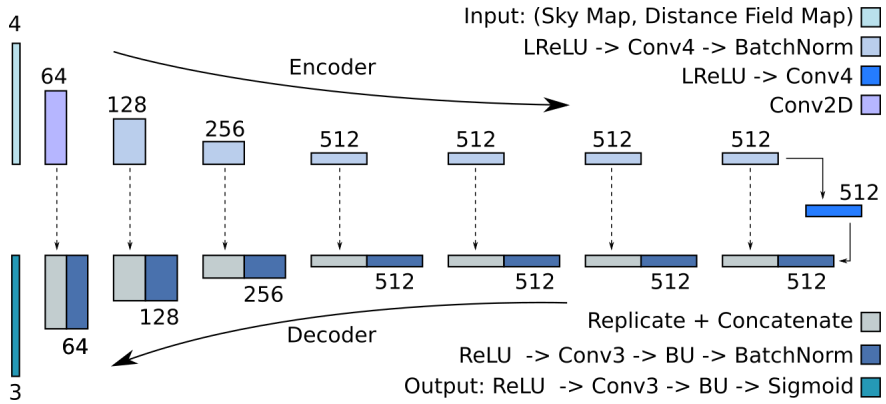

Figure 2. The CNN model used for the generation of the environment map.

represent a variety of lighting conditions present in real-world skies. Our method, however, is not limited to this method. In this work, we refer to clear sky lighting from the sky map as L_{CS}.

The description of cloud placement requires several design decisions. The first is the domain over which to represent clouds. Clouds are naturally represented on the sphere in $\mathbf{S}^2$; however to remain compatible with typical 2D CNN layers, we project from $\mathbf{S}^2$ to $\mathbf{R}^2$. While many projections for $\mathbf{S}^2 \mapsto \mathbf{R}^2$ are possible, such as latitude-longitude or cylindrical coordinates, we use a low distortion mapping by representing the cloud map as a fisheye image. This has the additional benefit of being an easy domain to make edits in, again due to the low distortion mapping between the fisheye image and the sphere.

In this domain, clouds are represented as a binary mask C, where white values represent clouds, and black represent clear sky. These masks can be generated through conventional image editing software to allow the placement of clouds to be directly specified. Although these masks could be used directly as input to the network, we found the use of binary values led to poor prediction performance, especially in the interiors of large clouds. To mitigate this issue, and to improve the ability of the network to predict smaller scale structures in the clouds, we provide the network with information about the extent of the clouds. This is efficiently represented by transforming the binary mask into a distance field D, where each pixel represents the distance from the pixel x to the edge of the cloud ∂C: $D(x) = \inf_{c \in \partial C} ||x - c||_2$. The distances are

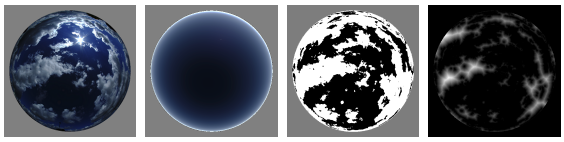

Figure 3. Generation of the dataset from 360 degree sky images. The figure from left to right shows a 360 degree cloudy image and the maps estimated from it: clear sky, mask map, distance field map (its values scaled up for visualisation purposes).

then normalised by the resolution of the input cloud map to ensure a consistent meaning for the distance field values.

Using the distance field also allows small scale cloud details to be either manually or procedurally added. One simple approach to add extra detail is to use a noise function to modify the distance field by a position dependent noise function $\xi(x) \in [0..1]$:

$$D'(X) = \lambda_n \xi(x) D(X), \qquad (1)$$

where $\lambda_n \in \mathbf{R}^+$ is a scaling term for the noise.

CNN Model

The CNN model used for the production of realistic sky environment maps is a UNet [17] architecture is chosen for its ability to process the input on multiple scales, having a large receptive field, but also being efficient due to most of the processing happening on lower resolutions.

The encoder layers of the UNet architecture downsample and produce a low resolution encoding which is then progressively upsampled in the decoder and combined with the encoder features

before producing the output environment map. The encoder uses 2D convolution layers (k $= 4$, s $= 2$, p $= 1$) which also downsample the feature maps. The encoder uses bilinear upsampling (BU) followed by 2D convolutions (k $= 3$, s $= 1$, p $= 1$). Bilinear upsampling is used inplace of the more commonly used transposed convolution due to the checkerboard artefacts that can be caused by the latter [16]. Similarly to the UNet used for image to image translation by Isola et al. [12], the model uses batch normalisation layers, along with the Leaky ReLU activation (slope=0.2) in the encoder, ReLU activation in the decoder and feature sizes 64-128-256-512-512-512-512-512. A diagram of the architecture is shown in Figure 2.

The input to the network is the single-channel distance field map, D, along with the three-channel clear sky model, L_{CS}, forming a four channel image, X, at a resolution of 512×512, given by:

$$X = f_{\text{TM}}(\text{Cat}(D, L_{CS})), \tag{2}$$

where f_{TM} is a tone mapping operator used to compress the high dynamic range of the input before its processed by the CNN:

$$f_{\text{TM}}(z) = \frac{1}{1+z}. \tag{3}$$

The model output, $\hat{S}_c$, is the predicted HDR environment map with compressed dynamic range, at the same resolution as X, containing RGB values. The output is mapped to the range $(0, 1)$ using the pixel-wise sigmoid function applied directly after the last network layer. The final prediction is formed by applying the inverse of f_{TM} on the network output:

$$f_{\text{TM}}^{-1}(z) = \frac{1}{z} - 1. \tag{4}$$

The whole model is given by:

$$\hat{S} = f_{\text{TM}}^{-1}(\text{CNN}(f_{\text{TM}}(D, L_{\text{HW}}))) \tag{5}$$

Loss Function

The loss function, $\mathcal{L} = \mathcal{L}(\hat{S}_c, S_c)$, between the compressed dynamic range prediction, $\hat{S}_c$, and target, S_c, is composed of multiple partial losses, and is given by:

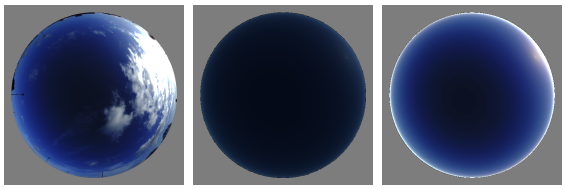

Figure 4. An example of histogram matching between the ground truth (left) and estimated Hosek-Wilkie sky map (middle). The image on the right shows the histogram matching result.

$$\mathcal{L} = \mathcal{L}_{\text{MSE}} + \lambda_{\text{VGG}}\mathcal{L}_{\text{VGG}} + \lambda_{\text{cos}}\mathcal{L}_{\text{cos}} + \lambda_{\text{R/B}}\mathcal{L}_{\text{R/B}}, \tag{6}$$

where $\mathcal{L}_{\text{MSE}}$ is the Mean Squared Error, $\mathcal{L}_{\text{VGG}}$ is a VGG [18] based perceptual loss [19], $\mathcal{L}_{\text{cos}}$ is a cosine dissimilarity loss and $\mathcal{L}_{\text{R/B}}$ is based on the Red over Blue channel ratio. The λ_{VGG}, λ_{cos} and $\lambda_{\text{R/B}}$ factors, scale the final magnitudes of the corresponding partial losses to equalise their contributions when training.

$\mathcal{L}_{\text{MSE}}$ accounts for the correct magnitude in the output regression per pixel. The perceptual loss, $\mathcal{L}_{\text{VGG}}$, accounts for inter-pixel correlations and thus helps produce improved textures, as shown by Johnson et al. [19]. It is defined as:

$$\mathcal{L}_{\text{VGG}}(\hat{S}_c, S_c) = \sum_{j=1}^{4} \left[\frac{1}{N_j} \sum_{i}^{N_j} |h_j(\hat{S}_c) - h_j(S_c)| \right], \tag{7}$$

where $h_{1,2,3,4}$ are the outputs from feature maps $2, 4, 7$ and 10 respectively, of a pretrained VGG16 network.

The cosine dissimilarity loss, $\mathcal{L}_{\text{cos}}$, is independent of the (R, G, B) vector magnitude and helps address the colour shift that is sometimes observed [16] on low light pixels when HDR content is regressed, due to the dominance that large values have on $\mathcal{L}_{\text{MSE}}$. $\mathcal{L}_{\text{cos}}$ is given by:

$$\mathcal{L}_{\text{cos}}(\hat{S}_c, S_c) = \frac{1}{2}\left[1 - \frac{1}{N}\sum_{i=1}^{N} \frac{\hat{S}_{c,i}\dot{S}_{c,i}}{||\hat{S}_{c,i}||_2||S_{c,i}||_2}\right], \tag{8}$$

where $\hat{S}_{c,i}$ and $S_{c,i}$ are the (R, G, B) vectors of the i^{th} pixel of $\hat{S}_c$ and S_c respectively.

Finally, the R/B loss, $\mathcal{L}_{\text{R/B}}$, is motivated from the cloud classification literature [20] and is used

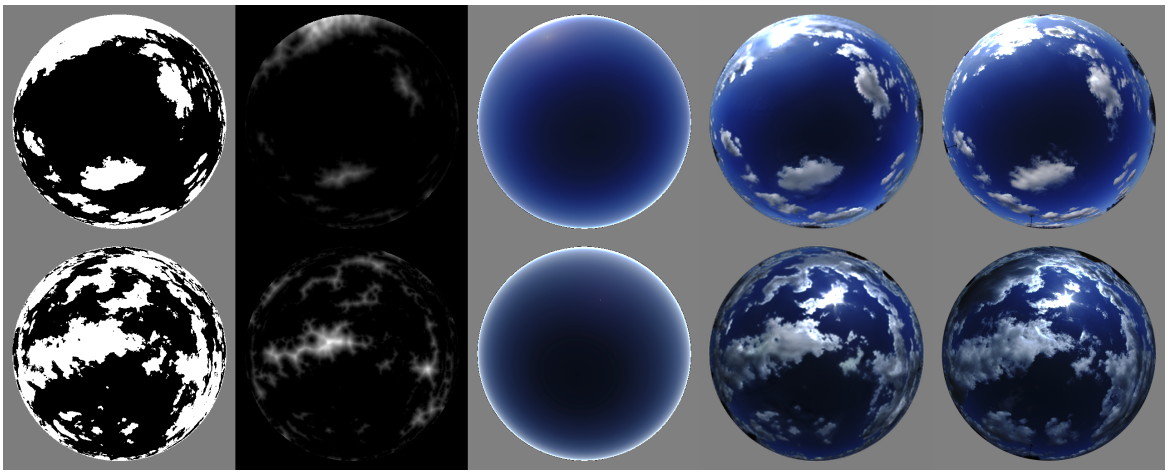

Figure 5. Results showing (from left to right): input cloud map, distance field map, sky map, CloudNet output and reference.

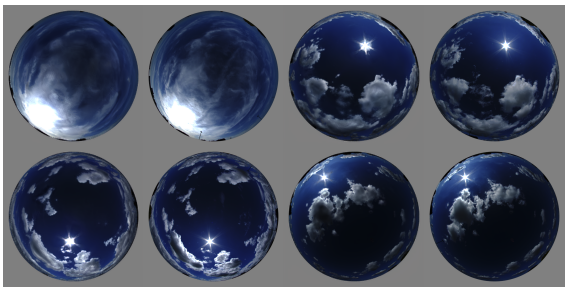

Figure 6. A selection of synthesised CloudNet images and their references. CloudNet outputs on the left and references on the right.

to weigh cloud consistency over sky pixels such that the result focuses more on the cloudy areas:

$$\mathcal{L}_{\text{R/B}}(\hat{S}_c, S_c) = \mathcal{L}_{\text{MSE}}\left(\frac{\hat{S}_c^R}{\hat{S}_c^B}, \frac{S_c^R}{S_c^B}\right), \quad (9)$$

where $\hat{S}_c^R$, $\hat{S}_c^B$, S_c^R and S_c^B are the red and blue channels of the prediction and target respectively. The performance of the R/B loss compared to other loss functions for cloud applications has been examined in [20] and we refer the reader to this work for a motivation for our approach.

Dataset

The dataset used to train the model consists of 629 (including 83 clear sky images) and the test set includes 135 panoramic 360 degree images, including 11 clear sky images. The images were captured using a Canon EOS 5D Mark 3 with a Sigma 8mm EX DG Fisheye Lens, an aperture of f3.5 and 7 bracketed exposures equally spaced from -8 to 8 and contain skies under varying sky and cloud conditions. The multiple exposures were fused into an HDR image using the method by Debevec et al. [21].

Once the HDR images were created, the input sky and cloud masks were required for each training example. Cloud masks were generated using the R/B ratio commonly used in cloud classification literature [20] which was able to automatically generate cloud masks as shown in Figure 3.

The input sky for each training example was also created by estimating the sun position in each image through finding the brightest pixel and converting this to azimuth and elevation of the sun. Next, the Hosek-Wilkie model was evaluated for all directions over the hemisphere above the camera leading to the input clear sky map. However, this map did not always correspond to the captured images due to colour and luminance differences (see Kider et al. [7]). Therefore, a histogram matching method was employed where an HDR cumulative histogram is computed for the same clear sky pixels in both the captured and Hosek-Wilkie sky map. The value for each pixel in the Hosek-Wilkie sky map is mapped to the corresponding value in the histogram of the captured pixels. An example of the estimated clear sky from cloudy images can be seen in the Figure 4.

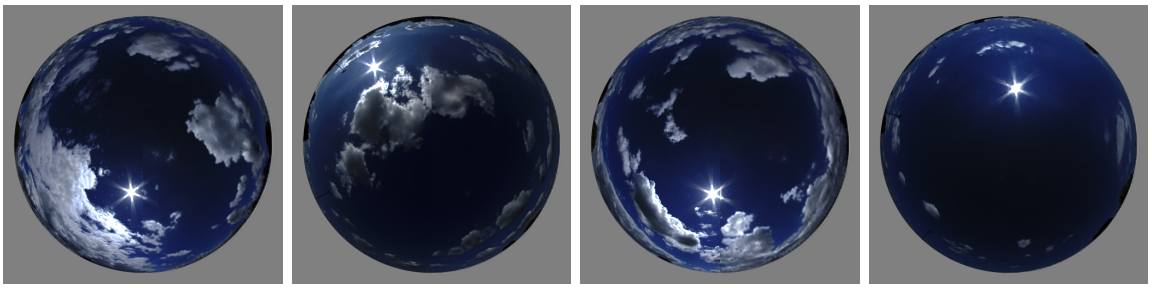

Figure 7. Juxtaposed images composed of the reference image on the left, and the CloudNet synthesised image on the right, split exactly across the middle horizontally.

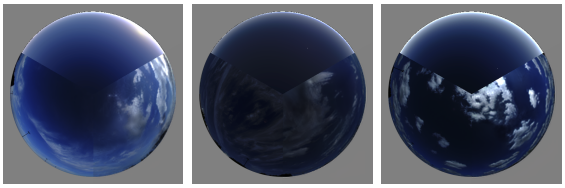

Figure 8. Split showing thirds with the sky map on the top, the ground truth on the bottom left and the CloudNet output in the bottom right.

Training

The network is trained using the training dataset of 629 captured HDR environment maps from which the clear sky inputs and distance fields are pre-computed. The inputs and targets are randomly rotated on-the-fly during training to provide data augmentation and avoid overfitting. The CNN is implemented and trained using the PyTorch framework [1] on an NVIDIA P100 GPU for 7,500 epochs. The Adam optimiser is used with a learning rate of 0.001 and the default parameters ($\beta_1 = 0.9$, $\beta_2 = 0.999$). The loss factors from Equation 6 are set to $\lambda_{\text{VGG}} = 0.001$, $\lambda_{\text{cos}} = 0.01$ and $\lambda_{\text{R/B}} = 0.001$.

Results

In this section results are presented. Initially results from a test set are presented, followed by manually crafted results and, finally, rendered images that use CloudNet output for illumination are presented. Inference time for the images was 14.2 ms on average.

All these results (unless noted) are generated with distance field map with added noise with $\lambda_n = 1.2$. For these results the addition of noise to the distance field, see Eq. 1, $\xi(x)$ is represented by single frequency Perlin noise. All displayed results were tone mapped from the original HDR in order to visualise the results.

Comparison to ground truth

CloudNet is evaluated on a test dataset of 135 captured environment maps that were not used for training. There are several possible methods of evaluation. The techniques described in the Related Work have the some limitations. Rendered images of clouds require prior knowledge of the ground truth cloud structure and scattering properties, something that is currently not feasible. Comparisons with deep learning approaches, which are not designed for the purpose of accurately generating clouds, would lead to unfairly poor performance by the other methods.

Therefore, comparisons against the ground truth images serve to highlight how close the quality of the output is to real-world captures. The training-testing split from the dataset was performed at random and the images were manually checked to ensure that no duplicates or near-duplicates / similarly looking environment maps were shared across between testing and training.

We show results for traditional MSE and RMSE, normalised by range, for evaluation over the entire set. The mean error across the 135 testing images compared to the ground truth is 1.647 for MSE and 0.012 for RMSE normalised by range, effectively producing an error of 1.2%.

Figure 5 demonstrates the results for two images from the test data set and includes the original cloud map, distance field map and sky map. Furthermore, Figure 6 show more pairs of CloudNet outputs and references. Note how with

[1] https://pytorch.org/

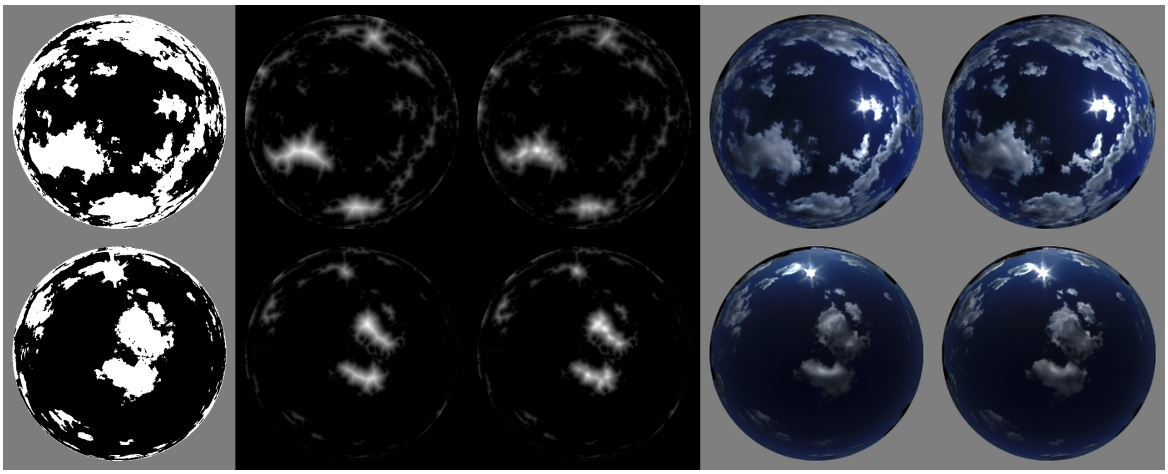

Figure 9. Images showing the effect of noise. From left to right: cloud map, distance field map (no noise), distance field map (noise), CloudNet output (no noise), CloudNet output (noise).

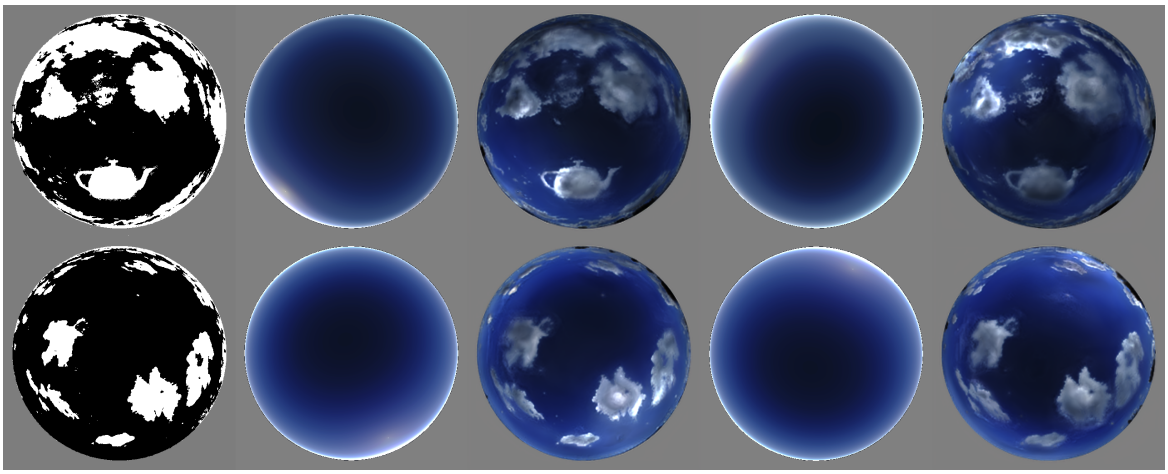

Figure 10. Results using hand drawn segmentation masks and different skies. Top row shows broadly different skies. The bottom row shows broadly similar skies but with the sun location changed. Clouds are consistent within the row.

these examples the illumination across the clouds is consistent between the ground truth and synthesised images. Figure 7, further demonstrates the consistency of the illumination across the sky, by shows composites of the ground truth and synthesised outputs split in the model of the horizontal axis.

Figure 8 shows three images composited, with the top third showing the sky map, the bottom left third the reference and the bottom right third the CloudNet output. These images show how CloudNet is adapting the output sky to be closer to the natural sky rather than just compositing clouds onto the sky map output from the sky model.

Figure 9 demonstrates the effect of applying Perlin noise to the distance field map providing further detail in the clouds.

Ablation Studies

Loss	MSE
$\mathcal{L}_{\text{MSE}} + \mathcal{L}_{\text{VGG}}$	1.904
$\mathcal{L}_{\text{MSE}} + \mathcal{L}_{\text{VGG}} + \mathcal{L}_{\text{cos}}$	1.858
$\mathcal{L}_{\text{MSE}} + \mathcal{L}_{\text{VGG}} + \mathcal{L}_{\text{R/B}}$	1.891
$\mathcal{L}_{\text{MSE}} + \mathcal{L}_{\text{VGG}} + \mathcal{L}_{\text{cos}} + \mathcal{L}_{\text{R/B}}$	1.819

Table 1. Ablation study for loss functions.

To test the performance of our combination of loss functions we performed an ablation study

to investigate the impact of each of the loss functions on the resulting error. We use the commonly used combination of MSE and the VGG based perceptual loss ($\mathcal{L}_{\text{MSE}} + \mathcal{L}_{\text{VGG}}$) as our baseline, and then add cosine loss ($\mathcal{L}_{\text{MSE}} + \mathcal{L}_{\text{VGG}} + \mathcal{L}_{\text{cos}}$) and R/B loss ($\mathcal{L}_{\text{MSE}} + \mathcal{L}_{\text{VGG}} + \mathcal{L}_{\text{R/B}}$) to this to examine the impact of each term. Finally, we show results for the combination used in this paper. These results were computed at 400 epochs, and are presented in Table 1. This shows that while the addition of both the cosine and R/B loss decrease MSE when used by themselves, and their combination produces a result with lower error as each term is able to optimize different aspects of the cloud lighting.

In order to demonstrate the suitability of the distance field as opposed to a binary mask, we ran an another ablation study which trained a network with the same structure and loss functions as described in the Method Section. The MSE when using the binary mask only was 1.905 while the distance field results were 1.819 (as above).

Generation

This section demonstrates the output of CloudNet independent of the dataset for manually crafted cloud maps and shows how illumination changes when sky settings are changed for the same set of clouds.

Figure 10 (top, showing the teapot shaped cloud) shows how the manually generated cloud map is illuminated by different sky model settings. The bottom row shows how the hand crafted map uses the same sky model but changes the sun location. This highlights how the sun's illumination and overall sky model settings affects the contribution to the clouds.

Rendering

Figure 11 shows results for rendered scenes lit by CloudNet environment maps and the same renderings without any cloud coverage and with ground truth sky maps. This shows that plausible illumination is generated by the network and can be used a light source for image synthesis. These renders were created using an in-house renderer using path tracing. Figure 12 shows renderings using synthetic cloud maps with a dog and teapot shaped cloud under different sun positions. Illumination changes in both the environment map and the scene are clearly visible.

Conclusion and Future Work

This paper proposed a data-driven approach to whole sky cloud synthesis. Based on a captured dataset of HDR sky images, we trained a CNN to be able to synthesise clouds into a clear sky model. This enables artistic control of cloud structure and placement while generating plausible lighting results. We show the results from our method are able to capture realistic lighting effects, and show the results in rendered images.

We intend to extend this work in several ways. Firstly, we are interested in replacing the use of a binary mask for cloud positions by a cloud type mask. This however is challenging due to the requirements of cloud type classification in order to generate training data. Secondly, we intend to capture images over a wider array of illumination conditions at differing times of day to further expand the dataset used to train our network. Thirdly, we plan to extend this work to dynamic clouds including artistic control, likely requiring a temporally consistent network. Finally, we believe our work is applicable to overcast skies via gathering of an overcast dataset and training a network to capture overcast features.

Acknowledgements

We would like to thank the reviewers for their valuable comments on our work. We also would acknowledge Blendswap users MrChimp2313 and thecali for the Victorian House model and the car respectively. We would also like to thank NoneCG for the city and Alim Zhilov for the skyscraper scenes. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research. The authors also would like to acknowledge the University of Warwick Scientific Computing Research Technology Platform for assistance in the research described in this paper.

REFERENCES

1. P. Debevec, "Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography," in *ACM SIGGRAPH 2008 classes*, 2008, pp. 1–10.

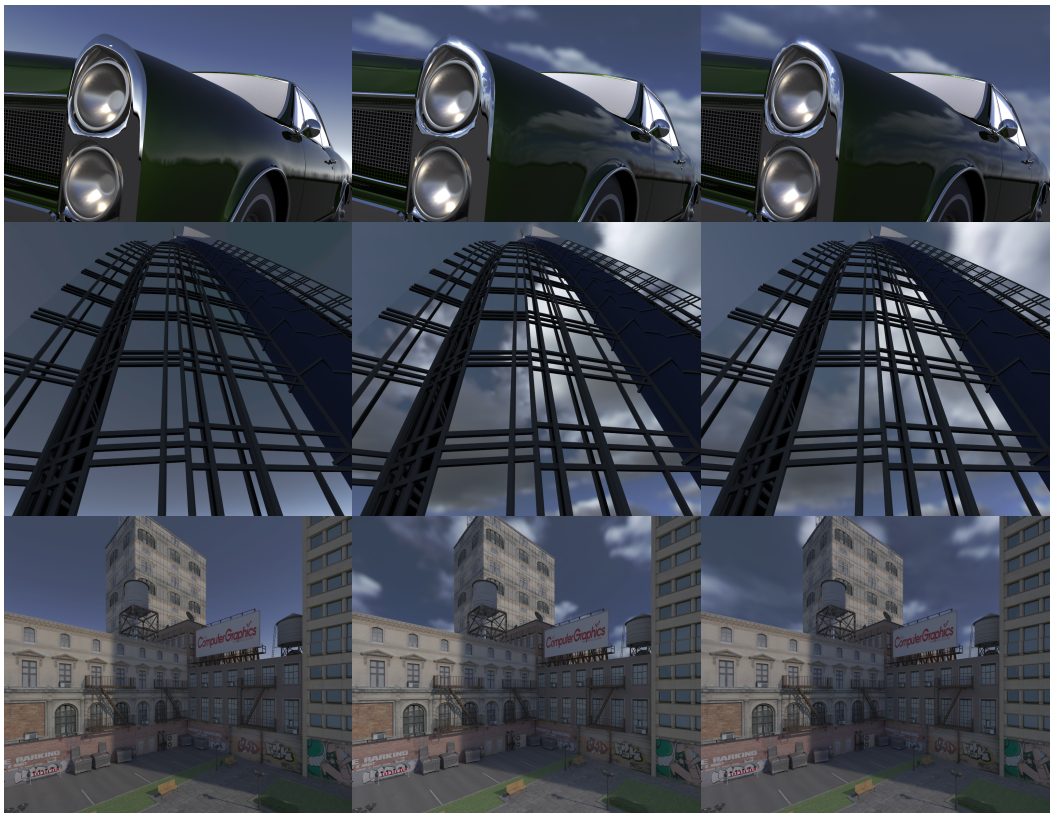

Figure 11. Rendered images for three scenes showing results for different illumination sources: Clear sky (left), ground truth sky (centre), CloudNet sky (right).

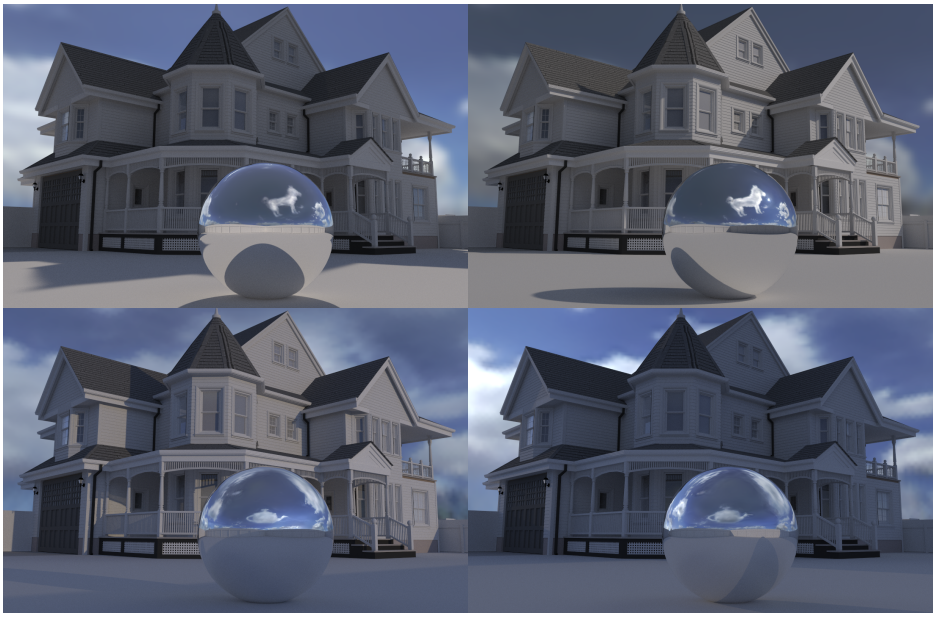

Figure 12. Rendered images for CloudNet generated environment maps using artist input showing the same scene under different illumination conditions. Across the rows the same cloud map is used with different sun position resulting in a different illumination across the image. The top row shows a cloud in the shape of a dog, then second in the shape of a teapot.

2. L. Hosek and A. Wilkie, "An analytic model for full spectral sky-dome radiance," *ACM Transactions on Graphics (TOG)*, vol. 31, pp. 1–9, 2012.
3. T. Hädrich, M. Makowski, W. Pałubicki, D. T. Banuti, S. Pirk, and D. L. Michels, "Stormscapes: Simulating cloud dynamics in the now," in *ACM Transactions on Graphics (TOG)*, vol. To appear. ACM New York, NY, USA, 2020, pp. 1–11.
4. S. Kallweit, T. Müller, B. Mcwilliams, M. Gross, and J. Novák, "Deep scattering: Rendering atmospheric clouds with radiance-predicting neural networks," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 6, pp. 1–11, 2017.
5. B. Bitterli, S. Ravichandran, T. Müller, M. Wrenninge, J. Novák, S. Marschner, and W. Jarosz, "A radiative transfer framework for non-exponential media," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 6, pp. 1–17, 2018.
6. R. Perez, R. Seals, and J. Michalsky, "All-weather model for sky luminance distribution—preliminary configuration and validation," *Solar energy*, vol. 50, no. 3, pp. 235–245, 1993.
7. J. T. Kider Jr, D. Knowlton, J. Newlin, Y. K. Li, and D. P. Greenberg, "A framework for the experimental comparison of solar and skydome illumination," *ACM Transactions on Graphics (TOG)*, vol. 33, no. 6, pp. 1–12, 2014.
8. A. Wilkie, P. Vevoda, T. Bashford-Rogers, L. Hošek, T. Iser, M. Kolářová, T. Rittig, and J. Křivánek, "A fitted radiance and attenuation model for realistic atmospheres," *ACM Transactions on Graphics (TOG)*, vol. 40, no. 4, pp. 1–14, 2021.
9. P. Satilmis, T. Bashford-Rogers, A. Chalmers, and K. Debattista, "A machine-learning-driven sky model," *IEEE computer graphics and applications*, vol. 37, no. 1, pp. 80–91, 2016.
10. J. Novák, I. Georgiev, J. Hanika, and W. Jarosz, "Monte carlo methods for volumetric light transport simulation," in *Computer Graphics Forum*, vol. 37. Wiley Online Library, 2018, pp. 551–576.
11. L. Gatys, A. S. Ecker, and M. Bethge, "Texture synthesis using convolutional neural networks," in *Advances in neural information processing systems*, 2015, pp. 262–270.
12. P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1125–1134.
13. I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.
14. J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2223–2232.
15. T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu, "Semantic image synthesis with spatially-adaptive normalization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2337–2346.
16. D. Marnerides, T. Bashford-Rogers, J. Hatchett, and K. Debattista, "Expandnet: A deep convolutional neural network for high dynamic range expansion from low dynamic range content," in *Computer Graphics Forum*, vol. 37. Wiley Online Library, 2018, pp. 37–49.
17. O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.
18. K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations*, 2015.
19. J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *European conference on computer vision*. Springer, 2016, pp. 694–711.
20. A. Heinle, A. Macke, and A. Srivastav, "Automatic cloud classification of whole sky images," *Atmospheric Measurement Techniques*, vol. 3, no. 3, pp. 557–567, 2010.
21. P. E. Debevec and J. Malik, "Recovering high dynamic range radiance maps from photographs," in *ACM SIGGRAPH 2008 classes*, 2008, pp. 1–10.

Pinar Satilmis, is a Research Fellow in Computer Graphics and Vision at Digital Media Technologies Lab, Birmingham City University. She has a PhD (2017) in Computer Science from the University of Warwick, UK, and a BSc (2009) and MSc (2012) in Mathematics from Hacettepe University, Turkey. Her research interests are Sky Lighting for Graphics, Machine Learning, Image Processing, Rendering and HDR imaging. Contact her at Pinar.Satilmis@bcu.ac.uk.

Demetris Marnerides, is an independent researcher. He has previously worked as a Research Fellow at the Warwick Manufacturing Group (WMG), University of Warwick. He holds a BA in Physics

(University of Cambridge, 2013), an MSc in Scientific Computing (University of Warwick, 2015), and a PhD in Engineering (University of Warwick, 2019). His research topics include Machine Learning, Computer Vision, Image Processing and HDR Imaging. Contact him at dmarnerides@gmail.com.

Kurt Debattista, is Professor at WMG, University of Warwick. He holds a PhD from the University of Bristol, an MSc in Computer Science, an MSc in Psychology and a BSc in Mathematics and Computer Science. His research has focused on high-fidelity rendering, high-dynamic range imaging, applications of vision, and applied perception. Contact him at K.Debattista@warwick.ac.uk.

Thomas Bashford-Rogers, is a Senior Lecturer at the University of the West of England. He holds a PhD from the University of Warwick and a BSc in Computer Science from the University of Bristol. His research interests include Rendering, MCMC, Machine Learning, High Dynamic Range Imaging, Quantum Computing, and Image Processing. Contact him at Tom.Bashford-Rogers@uwe.ac.uk.