Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

A Multi-timescale Image Space Model for Dynamic Cloud Illumination

Pinar Satilmis[a], Thomas Bashford-Rogers[b,*]

[a] *University of Warwick*
[b] *Birmingham City University*

ARTICLE INFO

Article history:
Received January 6, 2025

Keywords: Computers and Graphics, Formatting, Guidelines

ABSTRACT

Sky illumination is a core source of lighting in rendering, and a substantial amount of work has been developed to simulate lighting from clear skies. However, in reality, clouds substantially alter the appearance of the sky and subsequently change the scene illumination. While there have been recent advances in developing sky models which include clouds, these all neglect cloud movement which is a crucial component of cloudy sky appearance. In any sort of video or interactive environment, it can be expected that clouds will move, sometimes quite substantially in a short period of time. Our work proposes a solution to this which enables whole-sky dynamic cloud synthesis for the first time. We achieve this by proposing a multi-timescale sky appearance model which learns to predict the sky illumination over various timescales, and can be used to add dynamism to previous static, cloudy sky lighting approaches.

1. Introduction

The sky is a vital illumination source when rendering 3D scenes. A core component of real skies is clouds which can substantially alter the scene illumination compared to clear sky lighting, and as such, need to be represented when generating imagery. This typically takes the form of whole-sky illumination, which represents lighting information for the whole hemisphere of the sky above the scene.

Sky illumination can be computed using environment maps, analytical models, or by simulating the radiative transport equation. While simulating the radiative transport equation can generate highly realistic clouds, it is computationally demanding and requires 3D volumetric representations of complicated cloud structures, which are challenging to model or generate. Environment maps are High Dynamic Range (HDR) images that store far-field directional illumination and are used for image-based lighting, see Debevec [1]. Typically, they are obtained using specialized devices that capture 360° or fish-eye images of the upper hemisphere containing sky illumination. Although this method can represent lighting from cloudy skies, it is limited by the acquisition process to a fixed number of locations, times, and cloud types. Analytical models, on the other hand, can reproduce a wide range of sky scenarios and create dynamic lighting for different solar positions [2, 3, 4], but these models typically represent clear sky lighting and do not account for clouds.

To overcome these limitations, recently generative machine learning methods have been developed [5, 6] to generate cloudy sky environment maps. These typically are trained on images of real cloudy skies and are combined with an analytical sky model input to generate clouds either via a Generative Adversarial Network (GAN) as in the work by Mirbauer et al. [6], or via user or artist specified masks and a U-net as used in

*Corresponding author:
e-mail: thomas.bashford-rogers@warwick.ac.uk ()

Satilmis et al. [5]. These methods can generate realistic images of cloudy skies suitable for use as environment maps without the complexities of representing and rendering cloud volumes.

Despite the ability of these generative methods to synthesize cloudy sky imagery, these aforementioned methods still lack one important feature of clouds: dynamism. Clouds are not static and can significantly change their position in a relatively short period of time. Existing methods focus on the generation of a single frame of cloudy sky illumination, but when used in practice, the dynamism of cloud movement needs to be included in a generative model. Examples of this use case are clouds moving in a rendered environment for an animated sequence for film, or used to provide dynamic skies in interactive entertainment applications. In contrast to existing generative methods for synthesing video [7, 8], our work proposes an approach which can synthesize plausible *hemispherical* cloud movement over a short to medium term timescale given a single input image, either captured or generated by a generative machine learning model.

We achieve this by proposing a multi-timescale approach to cloud synthesis. At a longer timescale, a deep learning model predicts the larger non-linear changes in cloud positions and associated appearance. It achieves this via neural networks which predict a flow field of how clouds move across the sky, and based on this an illumination at the previous time step, predicts cloudy sky appearance at the next time step. Cloud movement at shorter timescales is predicted via a linear model of cloud movement conditioned on the nonlinear movement from the longer timescale and leads to the smooth movement of cloud position, shape, and illumination between the longer timesteps. Our method predicts plausible cloud appearance at these timescales based on a captured low dynamic range training set; new predicted frames are expanded to high dynamic range via inverse tone mapping [9].

Our approach is trained and validated using a database of recorded cloud movement sequences, and we illustrate how it can be applied to either a single captured image of a cloudy environment or an image generated by a cloud model [5, 6]. Our technique allows animation of artistically generated clouds and produces results which can be directly used in a rendering system for environment illumination.

To summarize, the main contributions of the paper are:

- A novel framework for synthesizing dynamic cloud lighting from a single input image, either from a hemispherical sky capture or from recent static generative methods.
- A multi-timescale approach which can generate longer timescale cloud movement while ensuring smooth, coherent movement at short timescales.
- Results demonstrating our approach can synthesize smooth cloud movement and show this applied in different rendering scenarios.

2. Related Work

In this section, we cover the main approaches to generating sky illumination. First, we discuss clear sky models which provide cloud-free sky illumination; then we cover the two main approaches to synthesize sky illumination with clouds: simulating cloud structure, dynamics, and lighting via numerical simulations and using deep image-generation techniques to generate cloudy sky illumination.

2.1. Clear Sky

Clear sky models are typically based on analytical or tabulated approximations of atmospheric light transport. These models combine sky specifications, for example, turbidity, solar position, and ground albedo, and predict incoming light from a given direction. An early model used in graphics was proposed by Perez [10]. When the parameters are carefully chosen, this five-parameter model can accurately describe low-turbidity skies. Later, models that were proposed based on fitting results to brute-force atmospheric light transport simulations, such as Nishita et al. [11, 12] Haber et al. [13], Preetham [14], Hosek and Wilkie [2] and Wilkie et al. [4]. These all increase the quality of the approximations of the clear sky illumination by including aspects such as multiple scattering in the atmosphere such as Hosek and Wilkie [2] and realistic profiles of scattering particles such as Wilkie et al. [4]. These methods also typically trade quality for memory usage, where previous models used fewer parameters which led to a limited range of representable phenomena, more recent models have generated more accurate sky illumination at a large memory cost.

2.2. Cloud Modeling

Modeling cloud structure is typically achieved via either procedural methods or physical simulation. Procedural methods include using implicit functions, Ebert [15]; fractals, Voss [16]; textured ellipsoids, Gardner [17]; spectral models, Sakas [18]; and implicit ellipsoids, Schpok et al. [19]. Physically based models are derived from images of clouds and different approaches have been investigated. Dobashi et al. [20] modeled clouds as metaballs from satellite images. The method by Wither et al. [21] generated a cloud mesh from a user-drawn sketch. Dobashi et al. [22] and Yuan et al. [23] focused on modeling clouds from a single input image.

Simulating clouds is also widely investigated in the literature, for example, Dobashi et al. [24], Harris and Lastra [25], and Dobashi et al. [26]. Recently, the complex nature of cloud formation was investigated by Hädrich et al. [27] who developed a novel framework to simulate physically accurate clouds. This led to plausible cloud dynamics in 3D, but it is computationally expensive and furthermore needs to be coupled with expensive light transport simulation to produce realistic imagery. For more details about approaches to representing clouds in computer graphics, please see the survey by Goswami [28].

2.3. Volumetric Cloud Rendering

Generating physically accurate cloud renderings is a computationally heavy task, especially if animations are required, or the clouds need to be synthesized into an environment map for use in conventional rendering software.

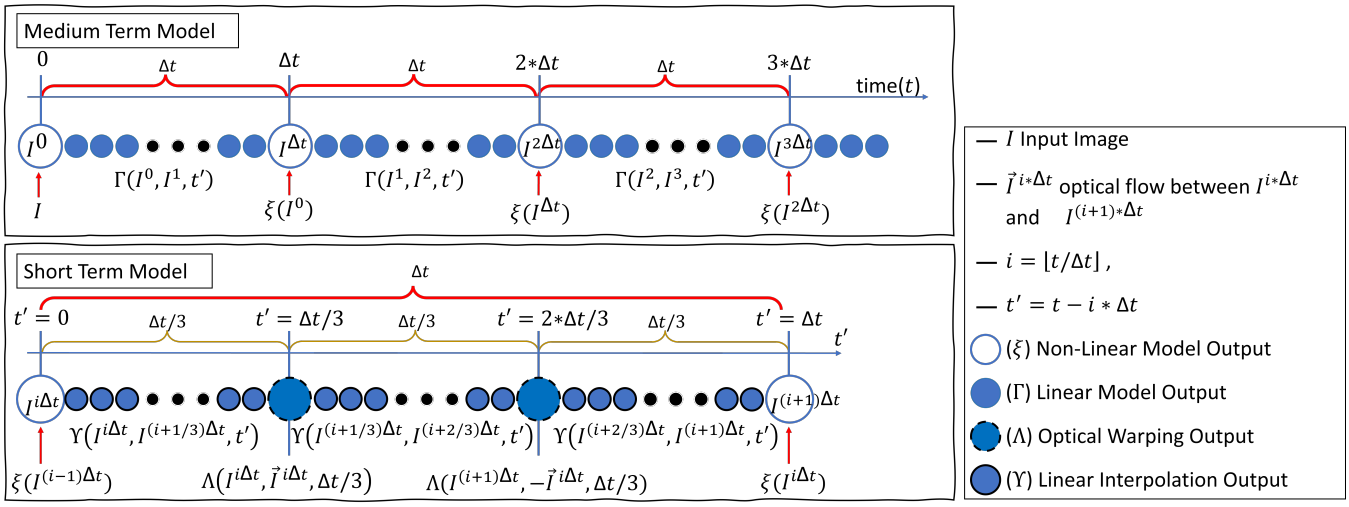

Fig. 1. An overview of deep dynamic cloud generation method. The method uses a non-linear model (ξ) and a linear model (Γ) to predict the sky lighting from a given input cloudy sky-image (I). ξ predicts the sky-image from its previous prediction for Δt time interval. Γ interpolates the sky lighting in between two sequential predictions of ξ. Γ is optical warping for $t' \in \{\Delta t/3, 2\Delta t/3\}$, and is linear interpolation for $t' \in (0, \Delta t) \setminus \{\Delta t/3, 2\Delta t/3\}$.

Motivated by this, several fast specialized, methods for cloud rendering have been proposed to generate plausible cloud appearance Harris and Lastra [25], Dobashi et al. [24], Riley et al. [29], Nishita et al. [12], Elek and Kmoch [30]. When higher levels of realism are required, physically based light transport methods can be used. These simulate light transport through the clouds and atmosphere. This is commonly achieved by Monte-Carlo volumetric path tracing; see Novák et al. [31] for a detailed explanation. Despite being physically accurate, this rendering technique is very computationally intensive, especially for sky and cloud rendering which has a very large spatial volume combined with cloud scattering parameters which lead to hundreds of scattering events for each path. To accelerate this process, approximation methods have been suggested, such as by Hillaire [32] who combined multiple approximation techniques into a technique suitable for real-time use, and Kallweit et al. [33] applied neural networks to improve the efficiency of Monte-Carlo rendering. Though these methods can achieve highly realistic cloud imagery, this does not take into account the subtleties of light transport in real clouds, for example, non-exponential free-flight distributions. This was addressed by Bitterli et al. [34] and Jarabo et al. [35] concurrently. These methods also provide little control over cloud shape and placement. A notable exception is Webanck et al. [36] which generated 3D animated cloudscapes by combining procedural techniques with Optimal Transport and considered both overall cloud shapes and smaller details using noise functions.

These methods can approximate cloudy sky illumination, but are very computationally expensive, and still can lead to non-photorealistic results.

2.4. Deep learning based methods

To avoid the complexities of simulating light transport when generating environment maps for later use in rendering, several deep learning based approaches for generating cloudy sky illumination have been proposed. These directly generate pixels in an environment map given a specification of the sky and cloud conditions.

Satilmis et al. [5] and Mirbauer et al. [6, 37] proposed methods to generate whole-sky cloudy lighting which is suitable for use as an environment map and therefore can be directly integrated into rendering systems. Satilmis et al. [5] used U-net structured autoencoders to generate cloudy sky lighting from encoded clear sky lighting and a cloud mask. GANs have also been used to synthesize clouds into environment maps. Mirbauer et al. [6, 37] used generative adversarial networks to synthesize cloudy skies conditioned on sun position and cloud coverage, and Valença et al. [38] who proposed a similar approach which synthesized clouds based on a set of sky parameters.

A different approach using transformers is developed by Chen et al. [39] to generate HDR panoramas from a given text description. Goswami et al. [40] produced a method that synthesized cloud animations in world space but relied on simplified volumetric cloud and lighting models.

Finally, there have been recent methods which generate dynamic outputs from large neural networks, typically conditioned via a Large Language Model. Work such as Blattmann et al. [41] and Brooks et al. [7] have generated impressive results for a wide range of inputs, for a summary see Cho et al. [8]. Our work differs in that it generates results over a longer timescale on a hemisphere and is specifically tailored to cloud movement.

2.5. Deep image synthesis and editing

There has also been work on generating or editing images with clouds in image space. This is different from our application as these approaches manipulate clouds in a perspective view, whereas we are focused on the full hemisphere of lighting. These approaches often utilize a semantic layout, for example, Chen and Koltun [42] used a single feedforward network to synthesize photographic images. Although it can generate high-resolution images, it lacks the high-frequency details needed to

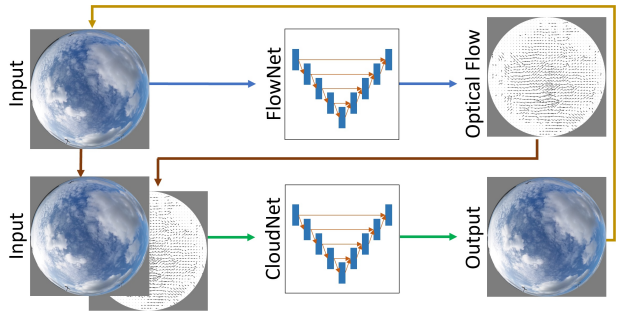

Fig. 2. The medium timescale Non-Linear model is a recursive method. ***CloudNet*** **takes two inputs: a sky image and its optical flow image.** ***FlowNet*** **is used to estimate the optical flow from the sky image. The output is the cloud image after Δt times and feedbacked to** ***CloudNet*** **to generate the next output. These recursive predictions create dynamic clouds.**

represent realistic clouds. Park et al. [43] used GANs to synthesize images with skies and clouds, enabling both semantic and style user control; see Tewari et al. [44] and Tewari et al. [45] for a wide discussion of techniques in neural rendering. Singer et al. [46] and Ho et al. [47] have proposed methods to generate videos from text descriptions, however, the outputs of these methods are not yet of high enough quality or resolution to be used for generating specific sky scenarios on the hemisphere or photorealistic lighting.

2.6. Summary

These methods propose photorealistic sky lighting which includes clouds, however are limited to generating one frame of illumination. As none of these approaches consider the temporal aspect of sky illumination, they cannot produce a coherent series of frames for cloud animation. Our work lifts this limitation by proposing a method for plausible cloud movement and illumination. We propose a method tailored to our application domain as existing methods do not produce imagery at an appropriate resolution or suitable for use in rendering applications.

3. Multi-timescale Sky Appearance Prediction

Static cloud synthesis on the hemisphere for use in rendering can be modeled by existing techniques such as Satilmis et al. [5] and Mirbauer et al. [6]. However in order to generate realistic cloud movement over multiple frames, we can start with these approaches or captured environment illumination and generate a sequence of images which contain cloud movement. As this needs to cover a longer timescale than a single frame, we need a method which provides smooth cloud movement frame-to-frame but coherent and plausible larger-scale changes to cloud shape, position, and illumination over a longer time period.

Motivated by this, we divide the timescales of cloud evolution into three: long, medium and short term. Long term here refers to the timescale in which the clouds or weather may change substantially and is typically modeled via weather forecasting. Medium term in the context of our work refers to tens of seconds, where the position and shape of the clouds may change substantially, but the types of clouds present are constant. Short term refers to the movement of clouds over a short timescale, in this work, this is less than 10 seconds, as discussed in Section 3.1.2.

As the focus of this work is on dynamic clouds, we focus on the short and medium timescales to capture movement. This faces the challenge that changes to cloud shape and position have to be smooth and coherent over hundreds of frames, while simultaneously predicting correct illumination. We also have to predict clouds moving at different speeds, due to factors such as different wind strengths for clouds at different altitudes [27]. While cloud evolution at these timescales can be predicted by physics-based models such as simulating fluid dynamics in the atmosphere, we operate in image space as this removes the need for complicated and expensive cloud, terrain, and atmospheric models.

However, simply predicting a subsequent frame of sky appearance given a previous image faces the issue that cloud appearance between frames which may be a fraction of a second apart, may change an extremely small amount, but over time the appearance change can be substantial. Therefore, a multi-timescale model is required. We take an approach which predicts the appearance of the sky at the medium scale, and uses a linear model to predict the short term movement conditioned on the current and next medium term prediction.

Specifically, the dynamic sky-lighting, represented as an image I^t, is modeled as a function of time $t \in \mathbb{R}^+$. Input and outputs are represented as an image I^t at a given time. The medium term, non-linear part $\xi(I^{i\Delta t})$ predicts the lighting at each Δt time interval and the short-term, linear part, $\Gamma(I^{i\Delta t}, I^{(i+1)\Delta t}, t')$ predicts cloudy sky illumination between each medium term output at time $t' \in (0, .., \Delta t)$. This can be summarized as:

$$I^t = \begin{cases} \xi(I^{i\Delta t}), & t \mod \Delta t = 0 \\ \Gamma(I^{i\Delta t}, I^{I(i+1)\Delta t}, t'), & otherwise \end{cases} \quad (1)$$

where $i = \lfloor t/\Delta t \rfloor$. The input, I^0, can be any hemispherical image of the sky, either captured or synthesized by a generative approach Satilmis et al. [5], Mirbauer et al. [6]. Figure 1, provides a summary of the method, and the functions $\xi(I^t)$ and $\Gamma(I^t, I^{t+\Delta t}, t')$ are explained in the following sections.

To achieve this in image space we need an invertible and low distortion mapping from the (hemi)sphere to image space $M : \mathbb{S}^2 \mapsto \mathbb{R}^2$. We use a fisheye mapping as used in previous work by Satilmis et al. [5], although our method can be re-trained to work with other mappings. We also need to take into account the dynamic range of real skies when applying our method, and as such propose to use inverse tone mapping to reconstruct HDR frames. This is discussed more in Section 3.3.

3.1. Medium Timescale Non-Linear Model

The medium timescale non-linear approach is composed of two Convolutional Neural Network (CNN) models, see Figure 2. The first model, *FlowNet* ($\mathcal{F}$), is trained to predict the movement and shape changes of the clouds at Δt intervals. To encode the movement of the clouds in this model, we propose to use a flow field $\in \mathbb{R}^3$ (two channels encode the angle, the other encodes the magnitude) to predict where each cloud pixel will

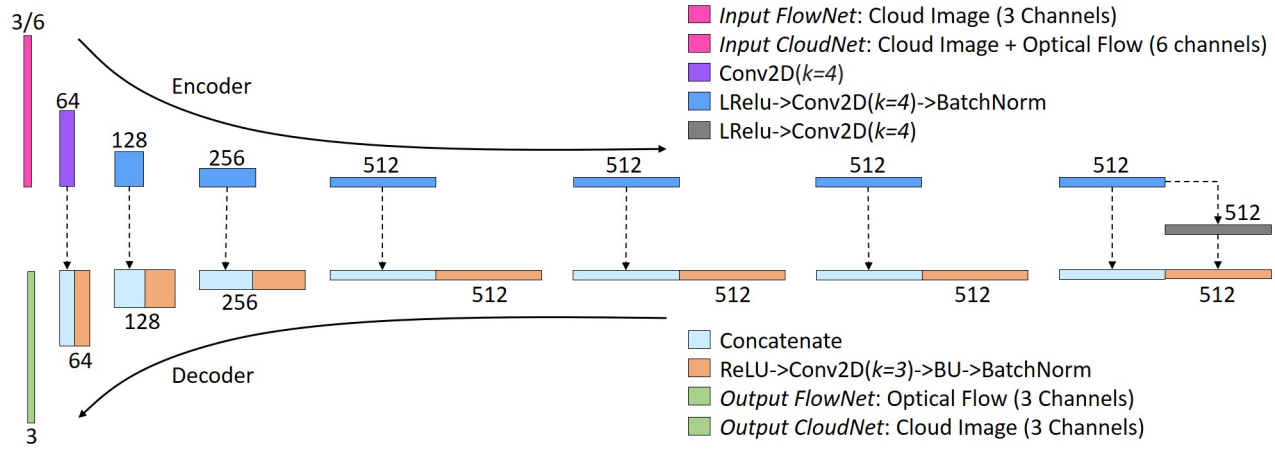

Fig. 3. The UNet architecture used in the framework. *CloudNet* **and the** *FlowNet* **use the same network structure except for the input layer.** *FlowNet* **takes a 3-channel input (cloud image), and** *CloudNet* **has a 6-channel input (concatenated cloud image and flow field). The encoder layers encode the inputs by applying eight sequential 4x4 convolutions, then the last seven convolutions come after a Leaky ReLU activation function and the six intermediate ones are followed by batch normalization. The decoder layer decodes the inputs by applying eight sequential ReLU, 3x3 convolutions, bilinear upsampling and batch normalization. All the convolutions use padding and stride of 1.**

move in the next Δt time. The second model, *CloudNet* ($\mathcal{C}$), is trained to predict the cloud illumination given a concatenation ($\oplus$) of the previous frame's illumination I^t and the flow field predicted by *FlowNet*. This also serves to reconstruct high-frequency details which may have been lost from naively applying a flow field. The overall non-linear model is defined as:

$$\xi(I^t) = \mathcal{C}(I^t \oplus \mathcal{F}(I^t)) \tag{2}$$

3.1.1. CNN Architecture

The models used in the framework are both UNets [49]. This model is chosen for its efficiency in recovering the low-level encoded data and also its success in generating cloud images [5].

FlowNet and *CloudNet* share the same network structure except for the input layer, see Figure 3. *FlowNet* takes a cloud image with 3 channels as input at $I^{i\Delta t}$ and outputs a predicted flow field $\vec{I}^{i\Delta t}$. *CloudNet* takes two inputs: the same cloud image at the previous timestep, $I^{i\Delta t}$ and the flow field $\vec{I}^{i\Delta t}$ resulting from *FlowNet*, which are concatenated to 6 channels. The encoder uses 2D convolutional layers with $kernelsize = 4$, $stride = 1$ and $padding = 1$ to downsample the images. The decoder uses bilinear upsampling, which doesn't suffer from checkerboard effects that might appear in the output, see Marnerides et al. [50]. This is followed by 2D convolutional layers with $k = 3, s = 1, p = 1$. Following a similar structure to Isola et al. [51], LRelu with slope 0.2 and ReLU are used as activation functions in the encoder and decoder, respectively. The networks have 64-128-256-512-512-512-512 features sequentially.

3.1.2. Flow Field Creation

The flow fields, $\vec{I}^{i\Delta t}$, in this work are generated by using optical flow, specifically the approach by Farnebäck et al. [48]. This produces an initial flow field $\vec{I}_F^{i\Delta t}$. As initial investigations showed that the optical flow method failed to estimate the flow if the time interval is too large between the images, we chose $\Delta t = 10$ seconds, which we found balanced between capturing enough movement and allowing the optical flow algorithm to produce valid results.

To ensure that optical flow is applied only to the cloud pixels, we estimate a binary cloud mask $I_{cm}^{i\Delta t}$ and elementwise multiply ($\odot$) the initial flow field by this mask to produce the final flow field:

$$\vec{I}^{i\Delta t} = \vec{I}_F^{i\Delta t} \odot I_{cm} \tag{3}$$

The cloud mask I_{cm} is computed for each pixel by:

$$I_{cm}^{i\Delta t} = \begin{cases} 1 & if \frac{I^{i\Delta t}(R)}{I^{i\Delta t}(B)} > 0.46 \\ 0 & otherwise, \end{cases} \tag{4}$$

where $I^{i\Delta t}(R)$ and $I^{i\Delta t}(B)$ are the red and blue color channels of $I^{i\Delta t}$ respectively. The thresholded ratio of these color channels $\frac{I^{i\Delta t}(R)}{I^{i\Delta t}(B)}$ is used for identifying pixels consisting of clouds. This is a common method used in cloud-sky segmentation [52]. We found the value of 0.46 worked well for thresholding cloud pixels from clear sky pixels in all of our dataset.

3.1.3. Dataset and Training

The dataset is composed of 3626 sequential LDR images captured every Δt seconds. 80% of the images were used in training and the rest were used for testing. The resolution of the images is 2048x2048. LDR images are used due to the difficulty of capturing HDR images of skies which do not include ghosting artifacts. The images were captured in the UK with a Ricoh Theta Z1 [53], see Figure 4. A flow field for each image was computed via the method outlined in Section 3.1.2 and used for training *FlowNet*.

During the training phase of *CloudNet* and *FlowNet*, all the input sky-images are taken from the dataset ($t = 0$). The optical flow inputs used for *CloudNet* training are generated by using the trained *FlowNet* model, see Figure 4 for an illustration of

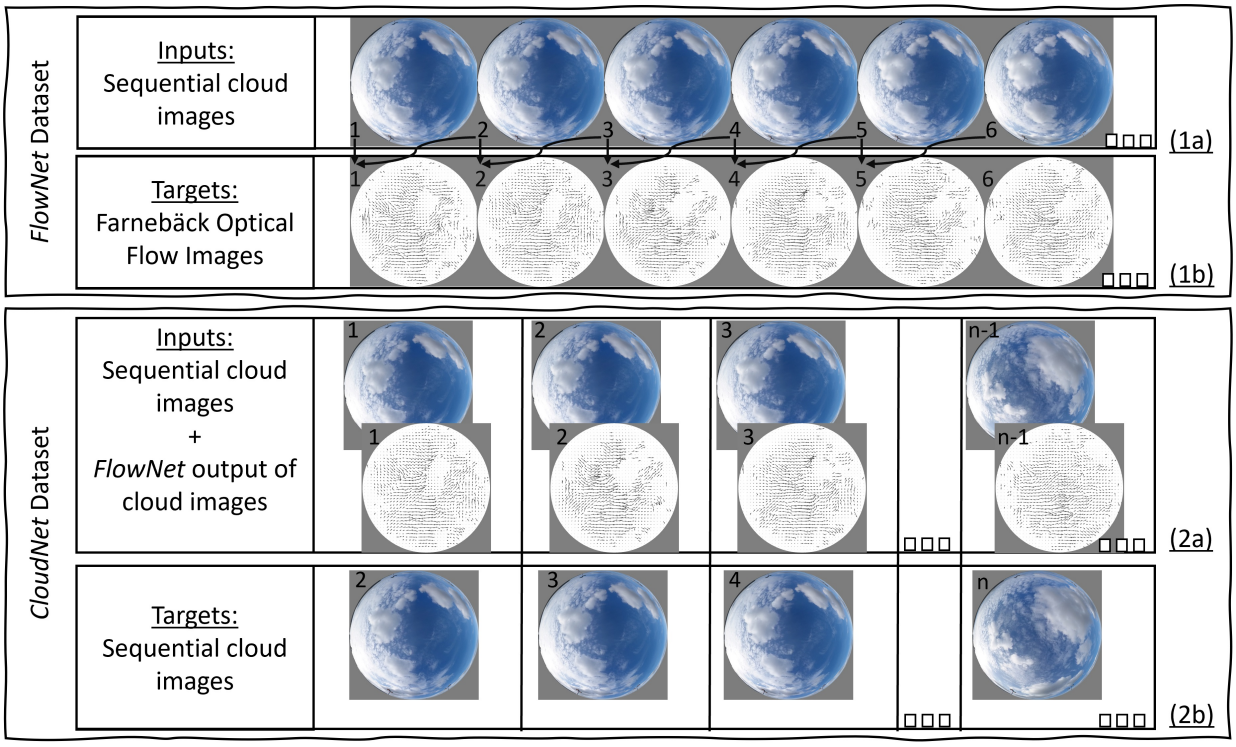

Fig. 4. Dataset for *FlowNet* and *CloudNet*. Our base dataset is composed of whole sky images captured at 10-second intervals (1a). For each image, a flow field is created using the method by Farnebäck et al. [48] (1b) to estimate the optical flow between each image and the consecutive image. Each sky image and its optical flow estimation forms the *FlowNet* dataset. The *CloudNet* input dataset is composed of the base pixel data and the corresponding output from *FlowNet* (2a). The target images are consecutive input sky images (2b).

this process. We trained the networks in PyTorch, utilizing the ADAM optimizer with a learning rate of 0.001.

3.1.4. Loss Function

For the loss function, the mean squared error (MSE) and cosine similarity loss functions were used. MSE is chosen as it is good at recovering the encoded data, but can lead to images which lack fine details. Furthermore, it provides low prediction accuracy in color distribution of pixel values [5]. Therefore, similar to Satilmis et al. [5], we included Cosine similarity due to its success in learning RGB values.

3.2. Short Timescale Linear Model

The short timescale model produces smooth cloud movement between every cloud image generated by the medium time non-linear model, i.e. the output of *CloudNet*. To ensure consistent and smooth cloud movement, the short timescale model must match the output of *CloudNet* at times $i\Delta t$ and $(i+1)\Delta t$. Furthermore, the flow fields at these times should also match to ensure consistent movement of these clouds. While it would be tempting to formulate this short-timescale model as a differential equation with Cauchy boundary conditions, in this case defined at times $i\Delta t$ and $(i+1)\Delta t$, the lack of uniqueness of the solution in this context limits practical application to this problem. Therefore, we propose a simple, fast-to-compute, deterministic model which respects the boundary conditions and leads to smooth cloud movement.

To achieve this, we divide the short timescale into three sub-intervals and model cloud movement using a piecewise linear approach. At times $(i+\frac{1}{3})\Delta t$ and $(i+\frac{2}{3})\Delta t$, we compute two intermediate frames by warping the clouds in image space. This is done by advancing the clouds forward in time from $i\Delta t$ and backward from $(i+1)\Delta t$ along their respective flow fields. We then linearly interpolate between these warped images to estimate cloud positions at the intermediate times.

We define a function $\Lambda(I, \vec{I}, t')$, which warps the cloud pixels in image I along the *flow field* $\vec{I}$ based on the time parameter t', where t' falls between $i\Delta t$ and $(i+1)\Delta t$. The flow field warps both images while trying to preserve structure. However, this does not handle the case where small scale structures change in the clouds. Our solution to this is to use linear interpolation between the images estimated with optical flow so that more fine structural changes are included in the frames. Therefore, we define another function, $\Upsilon(I^a, I^b, t')$, which is used to interpolate between two images taken at times a and b, with t' again representing the intermediate time.

This ensures the boundary conditions are respected which is automatically the case based on warping the clouds along the flow field forwards and backwards, and the interpolation in the middle step was sufficient to blend the outputs thus avoiding any discontinuity in cloud movement or appearance. To summarise, the short timescale model calculates cloud positions at different times using both warping (Λ) and interpolation (Υ):

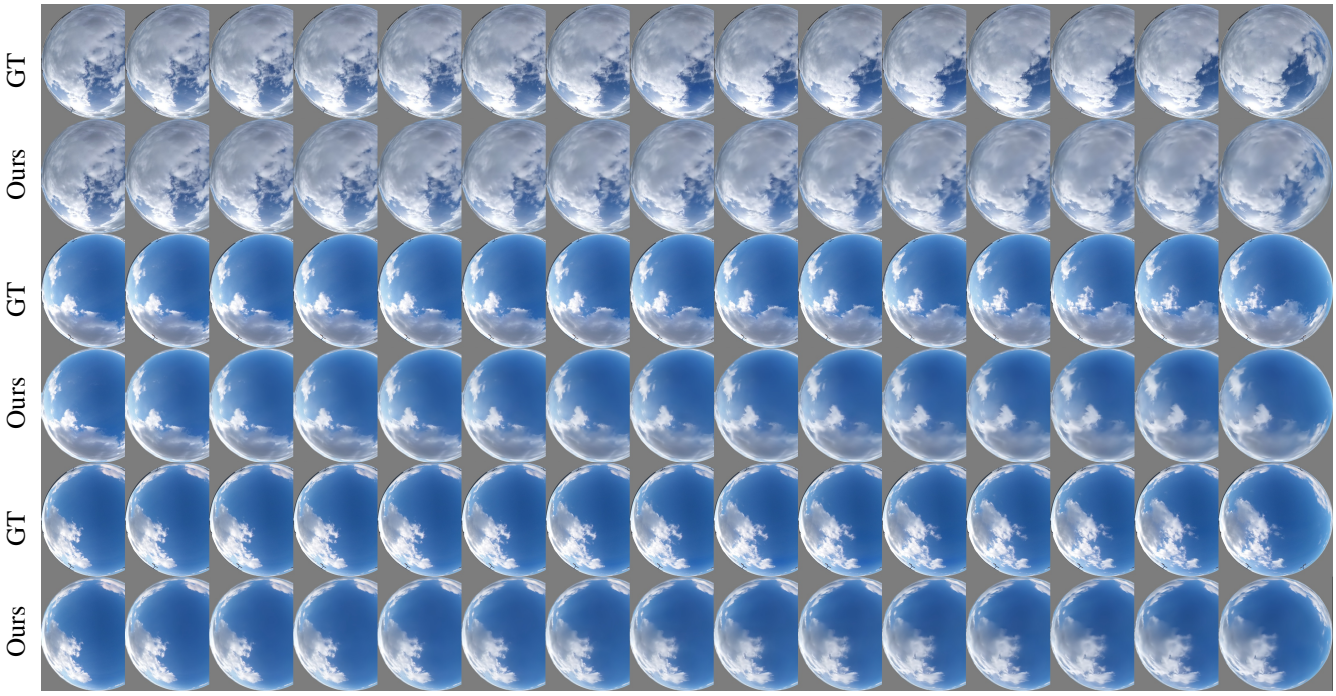

Fig. 5. Visual comparison of our method with a ground truth dataset. The first row is the test set (GT) and the second row are the predictions from our model (Ours). This shows that approaches generate similar predictions in terms of structure and illumination.

$$\Gamma(I^{i\Delta t}, I^{(i+1)\Delta t}, t') = \begin{cases} \Upsilon(I^{i\Delta t}, I^{(i+\frac{1}{3})\Delta t}, t'), & t' \in (0, \frac{\Delta t}{3}) \\ \Lambda(I^{i\Delta t}, \vec{I}^{i\Delta t}, \frac{\Delta t}{3}), & t' = \frac{\Delta t}{3} \\ \Upsilon(I^{(i+\frac{1}{3})\Delta t}, I^{(i+\frac{2}{3})\Delta t}, t'), & t' \in (\frac{\Delta t}{3}, \frac{2\Delta t}{3}) \\ \Lambda(I^{(i+1)\Delta t}, -\vec{I}^{i\Delta t}, \frac{\Delta t}{3}), & t' = \frac{2\Delta t}{3} \\ \Upsilon(I^{(i+\frac{2}{3})\Delta t}, I^{(i+1)\Delta t}, t'), & t' \in (\frac{2\Delta t}{3}, \Delta t) \end{cases} \quad (5)$$

where $I^{i\Delta t}$ and $I^{(i+1)\Delta t}$ are two sequential predictions of the *CloudNet* and $\vec{I}^{i\Delta t}$ is the optical flow between $I^{i\Delta t}$ and $I^{(i+1)\Delta t}$.

Moving clouds linearly along these vectors in image space does not take into account the curvature of the sphere. Therefore, the vectors in image space have to be converted to the sphere via the inverse mapping M^{-1}. Interpolation is performed along a great arc on the sphere, then mapped back to image space. In practice, we re-estimate the optical flow at timesteps $i\Delta t$ and $(i+1)\Delta t$ to incorporate the extra high frequency details which are added by *CloudNet* and use the method discussed in Section 3.1.2 to ensure that homogenous regions within the large clouds are filled with plausible values.

3.3. Inverse Tone Mapping

Similar to Chen et al. [39], we map the input High Dynamic Range image into a [0..1] domain, and apply an inverse tone mapper e.g. Zhang and Lalonde [9], Marnerides et al. [50], Khan et al. [54] to boost the final frame back to the appropriate dynamic range. It should be noted that this inverse tone mapping operator is not specific to our method; any inverse tone mapping operator can be used to expand the range of the resulting environment maps.

	ExpandNet	FHDR	HDRTVM
RMSE	0.201	0.267	0.322

Table 1. RMSE over common inverse tone mapping operators for reconstructing hemispherical cloudy skies against captured HDR data.

To choose an inverse tone mapping operator for our work, we evaluated some common inverse tone mapping methods against the set of captured ground truth images used in Satilmis et al. [5]. Table 1 shows the RMSE of the ExpandNet [50], FHDR [54], and HDRTVM [55] all of which were selected because they led to temporally stable reconstructions for our results. Based on this we used ExpandNet in our results, although this can easily be replaced with any other temporally stable inverse tone mapping operator.

All these inverse time mapping operators are not able to reconstruct the very high dynamic range of the sun, so we use the method by Hosek and Wilkie [56] to directly add the sun at the centre of the brightest region in the reconstructed HDR image. This has a limitation of not attenuating the sun if it is obscured by clouds, however in practice this leads to plausible lighting and shadows in the rendered images.

4. Results

In this section, we present the results of our method. First, we show qualitative and quantitative results for the dataset described in Section 3.1.3. We compare against real-world data, as to the best of our knowledge, there is no similar approach for generating whole sky dynamic clouds in the literature. Then, we provide examples of temporal clouds generated with synthesized inputs created with an existing generative cloud model.

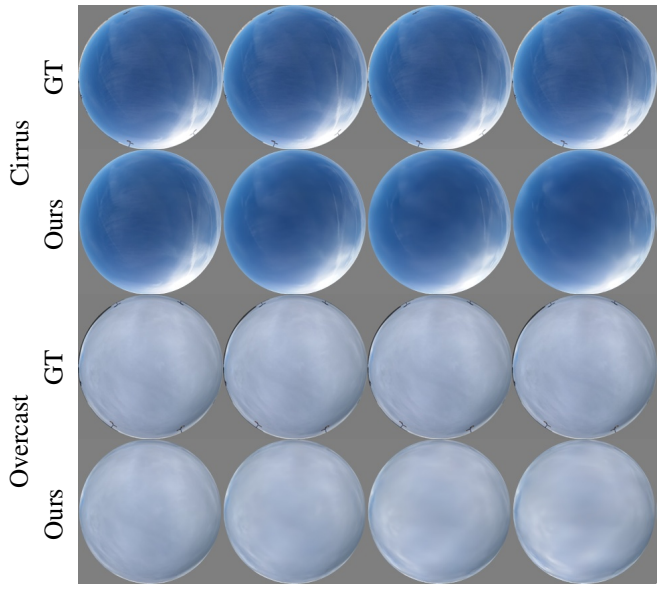

Fig. 6. Results showing out method applied to cloudy skies at two extremes. The top two rows shows our method applied to barely visible cirrus clouds, and at the other extreme the bottom two rows shows the results of our approach on overcast skies. The top images are ground truth captured images (GT), and the bottom are images generated with our method (Ours).

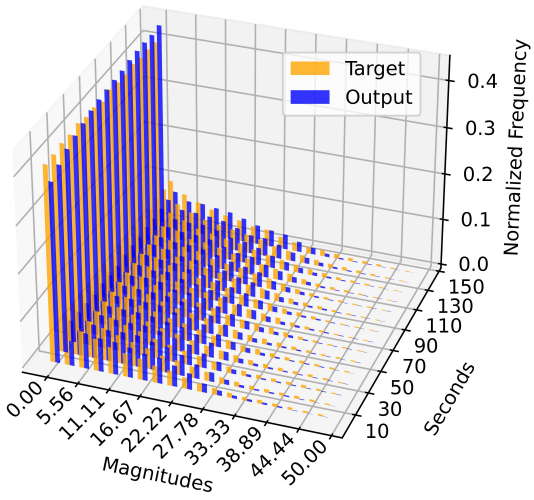

Fig. 7. The normalized distribution of the magnitude of the flow field in image space for each medium-term non-linear prediction per frame. These magnitudes correspond to angles on the sphere via the mapping M^{-1}. This compares the Target real images (orange) and the Output of our model (blue) and is calculated over all the test images in dataset. This shows that our approach follows a similar distribution to real-world cloud flow, and most of the errors of our approach are concentrated in sub-pixel movements which are less important than large-scale movement.

Lastly, we demonstrate results showing the main use case of our method which is rendering 3D scenes with generated cloudy sky animations. Further results generated with our method can be seen in the video in the supplementary material.

4.1. Real-world results

The accuracy of the *FlowNet* and *CloudNet* is evaluated through the test set of 20% of the dataset (725 frames). The structure of this set is described in Section 3.1.3. We present objective metrics averaged over the sequences in the test set. We show three metrics: MSE and PSNR measure pixel differences of the reconstruction, and SSIM measures differences in structure. In this evaluation for each test image, its next image is compared with *CloudNet* output. This shows the success of the model in prediction with an error of less than 0.5%. Table 2 summarises these results.

	MSE	PSNR	SSIM
Error	0.00536	23.673	0.904

Table 2. Summary of performance of our method averaged over the test set of sequences.

We visually compare our predictions with the ground truth captures in Figure 5. The top row shows ground truth captures, and the bottom row shows our method. This shows our method can generate plausible evolution of the clouds with time and broadly similar illumination generated with *CloudNet*.

We also show results where our method can be used to predict barely visible light clouds, such as cirrus with low coverage, or overcast skies at the other extreme. Examples of these types of skies generated with our method can be seen in Figure 6.

While the previous results assess the visual quality of our results, we also need to evaluate the temporal quality of our method. Our method produces plausible clouds, but the directional movement may not exactly match real data. Minor differences early on in a sequence can lead to exponential differences in cloud positions in later frames, although illumination and shape remain plausible. This is expected behavior, but direct pixel-to-pixel comparisons between frames would lead to meaningless comparisons. However, the magnitudes of the cloud movement frame-to-frame, in our case the magnitude of the flow field, can be expected to be similar between real and synthesized data as this captures how much the clouds move with time.

To assess this, we create a histogram of the magnitudes of cloud movement estimated with optical flow from both the captured real data and the output of our model. Each bin of the histogram stores a range of estimated flow values, from much less than a pixel to multiple pixels[1], and we plot this with respect to time. This is shown in Figure 7 which demonstrates that our model can approximate the amount of flow observed in real clouds as is shown by the similar distribution between real-world flow data (orange) and our predictions (blue). Most of the error between the method corresponds to cloud movements which are much less than a pixel, which typically corresponds to a fraction of a degree on the hemisphere. This shows the success of our model in predicting cloud movement over time.

In Figure 8, we show the results from both the medium timescale model, i.e. FlowNet and CloudNet which are the first and last images, and the short timescale model in the middle. These show results interpolated using optical flow and linear

[1]This also corresponds to movement over a great arc on the hemisphere

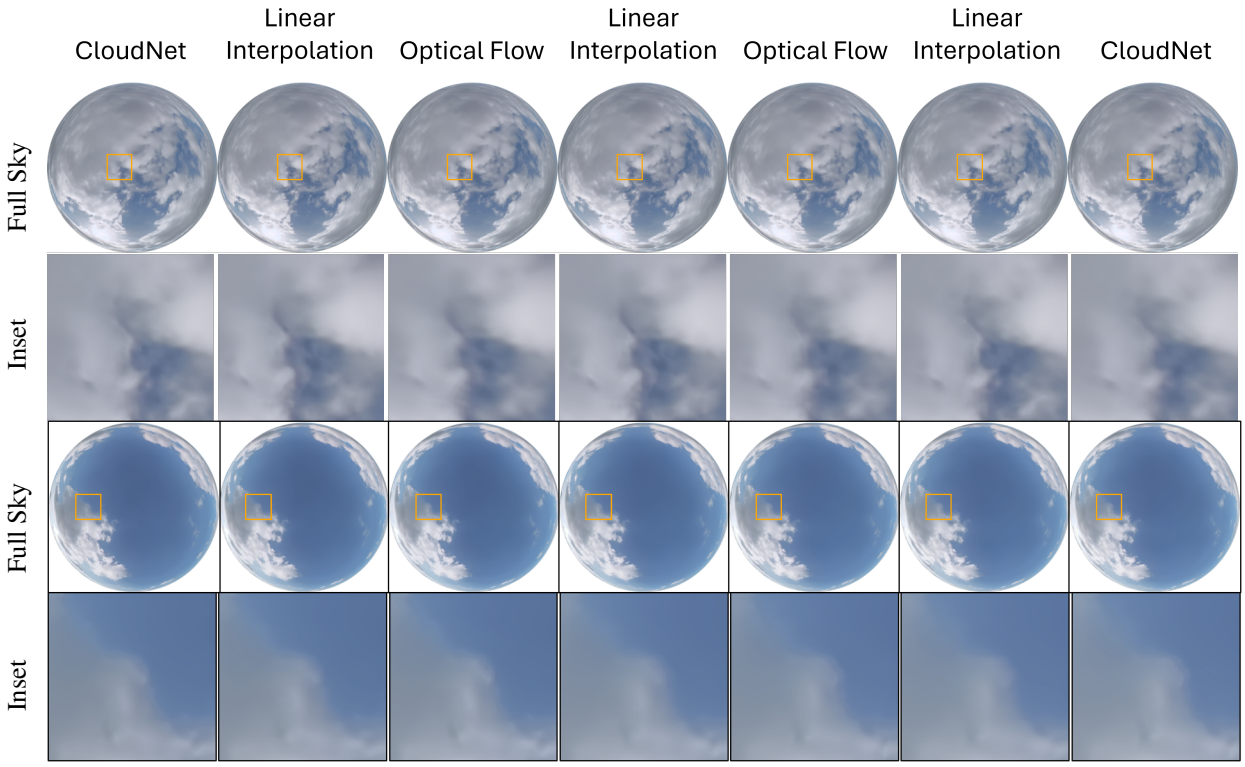

Fig. 8. Image showing the outputs of our method. The first and last images are results from FlowNet and CloudNet (medium timescale model), while the middle images use optical flow and linear interpolation.

interpolation. The shifts in the clouds are relatively subtle between timesteps, but natural cloud movement and illumination can be seen in the fisheye views and insets.

Finally, we report time for inference for our model. Our unoptimized code took on average 3 seconds to generate each frame, which includes running both *FlowNet* and *CloudNet* on the GPU, inverse tone mapping and conversion from fish eye to latitude longitude for rendering.

4.2. Deep Synthesized Inputs

Our model can also be used to predict cloud movement from static images synthesized by previous methods of generating clouds. This uses the output of these methods as the first frame and then generates cloud movement over time.

We show results from the method proposed by Satilmis et al. [5] as their method allowed the use of masks for generating clouds, and thus we can view the evolution of artistically controlled clouds. Figure 9 shows three examples of sequences of clouds starting from a generated cloudy sky including clouds in the shape of a "thought bubble", a teapot, and a dog.

This shows that our method can generate plausible cloud dynamics and illumination, even when the initial cloudy sky was implausible and artist specified.

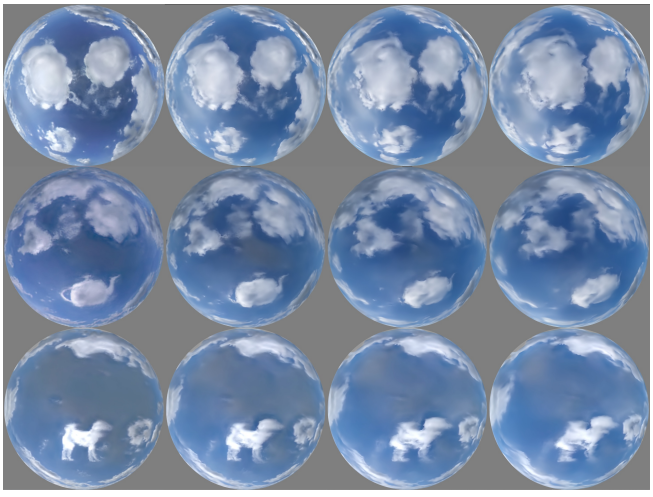

Fig. 9. This figure illustrates the prediction of cloud movements by using the inputs generated with the deep learning cloud synthesis method proposed by Satilmis et al. [5]. The first images are the inputs, and others illustrate the movement of the clouds after 50 seconds.

4.3. Rendering

We also show proof of concept results for sky models being used for lighting virtual environments which is their expected use case. This shows that our method can produce sky imagery at a high enough resolution (2048 × 2048 pixels) to be used

for practical rendering purposes. Figure 10 shows *tone mapped* stills from a variety of 3D scenes showing a direct view of part of the sky ("Tower" and "House" scenes), or reflections of a significant portion of the sky hemisphere (the "House", "JazzyPicnic" and "Observatory" scenes) being illuminated with the dynamic cloud model proposed in this work (the lighting is computed using a path tracer). Please see the supplementary material for the full videos.

Figure 11 shows tone mapped frames of an animation using the output of a generative cloudy sky model in the "JazzyPicnic", "House" and "Trophy Stadium" scenes, corresponding to the cloud masks shown in Figure 9. This again shows the realistic movement of clouds in a 3D environment, including when the initial clouds are artist controlled.

4.4. Comparison to Frame Interpolation

Our method is based on learning the flow of cloud movement at the various timescales proposed in Section 3. Another option for generating dynamic cloud movement is to use a generative model to generate two frames of cloudy sky images and use frame interpolation (see [58] for a survey of these types of approaches) to generate a smooth sequence of images between these two frames. Deep learning frame interpolation has been shown to be very effective in generating inbetween frames even under the presence of large movement. Therefore we perform two comparisons with a recent frame interpolation technique by Reda et al. [57].

The first applies frame interpolation to two synthetic start and end frames generated by Satilmis et al. [5], and we show results in Figure 12. While this generates smooth cloud movement, it does not synthesize natural cloud movement or lighting, i.e. clouds move in arbitrary directions or rapidly appear or disappear as is shown in the zoomed in insets. The second approach is to use our method for generating medium timescale images, and using frame interpolation to interpolate between these images. Figure 13 shows results for this scenario where again frame interpolation produces implausible cloud movement and lighting, whereas our method leads to smooth cloud movement and lighting.

This shows that our use of the flow field for medium timescale cloud movement constrains the clouds to be in plausible positions allowing for easier short timescale interpolation, which also uses the flow field to constrain cloud movement to expected directions. Our method is also designed to predict movement in large homogenous regions of the clouds by considering movement at the boundaries which current frame interpolation methods struggle with. Furthermore, our method also considers the non-linear motion of clouds on the hemisphere, whereas existing frame interpolation methods operate in 2D image space, so naive application of these methods to cloud movement is likely to result in artifacts, as is shown in Figures 12 and 13.

4.5. Impact of loss functions

We also conducted an ablation study to assess the use of MSE verses MSE+Cosine loss functions for training the network in Section 3.1. The results can be seen in Table 3. The results are obtained through training 500 epochs of *FlowNet* and *CloudNet* and shows an average MSE for the test set when trained with the two loss functions. It can be seen that using the combination of MSE and Cosine loss functions together provides substantially higher accuracy than only using MSE.

	MSE	MSE+Cos
Error	0.0085	0.0009

Table 3. Ablation study for loss functions used in this work. This shows that using MSE + Cosine Similarity leads to substantially less error than MSE.

4.6. Limitations

Our method produces smooth cloud movement and lighting, but there are occasional rapid changes generated by the medium timescale model which the short timescale model interpolates too quickly leading to rapid changes in cloud appearance. Furthermore, a slight loss of high frequency details can occur with our method. The first reason is that the use of optical flow can cause pixel values to be averaged in the medium timescale model which results on gradual loss of high frequency detail. The second reason is during the linear interpolation stage in the short timescale model the pixel values are directly interpolated in image space which is fine for small movements, but if these movements are slightly larger then the higher frequencies are temporarily shifted to lower frequencies during the interpolation. We intend to further explore deep learning short timescale interpolation models to handle these cases. Our loss is based on MSE, but other losses such as the perceptual loss may improve the preservation of high frequency details. We also captured our data at a single location at a similar time of year. This means that there is some similarity between our training and test data, and we intend to solve this by extending our dataset to include data captured in more geographical locations and at different times of the year and to include rarer cloud types. Finally, our method is designed for lighting from distant environment maps, but if closer 3D views of clouds are required, then approaches such as Webanck et al. [36] can be used.

5. Conclusion

This work has proposed a multi-timescale sky appearance model which adds dynamic cloud movement to static hemispherical cloud images. We show that our approach can synthesize plausible cloud movement for a wide range of captured skies and can also add dynamism to prior work which generates static sky imagery. We achieved this via a multi-scale approach which uses neural networks and flow fields to predict sky illumination at fixed intervals, and proposed a principled interpolation approach at short time-scales.

In the future, we aim to extend our approach to integrate longer term weather forecasting, possibly via graph neural networks combined with generative models, or to develop conditional video diffusion models extended to the hemisphere. We also intend to perform a user study to assess the perceived quality of static and dynamic generative cloud models and add artistic control to cloud movement by allowing a user to specify

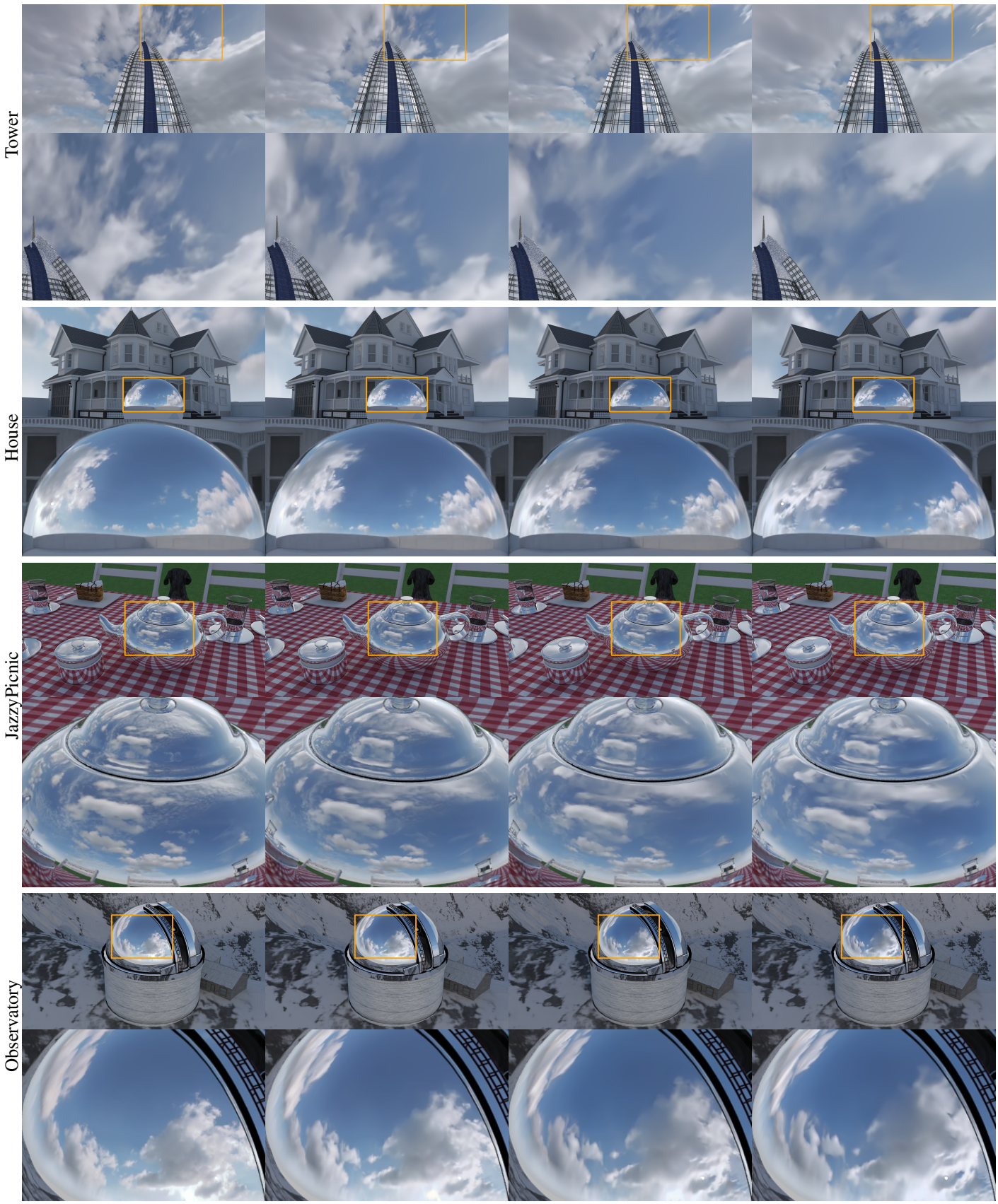

Fig. 10. Renderings showing our method being used to light a variety of scenes with different clouds over time. These tone mapped frames are captured every 45 seconds and show our method can synthesize plausible cloud appearance over time.

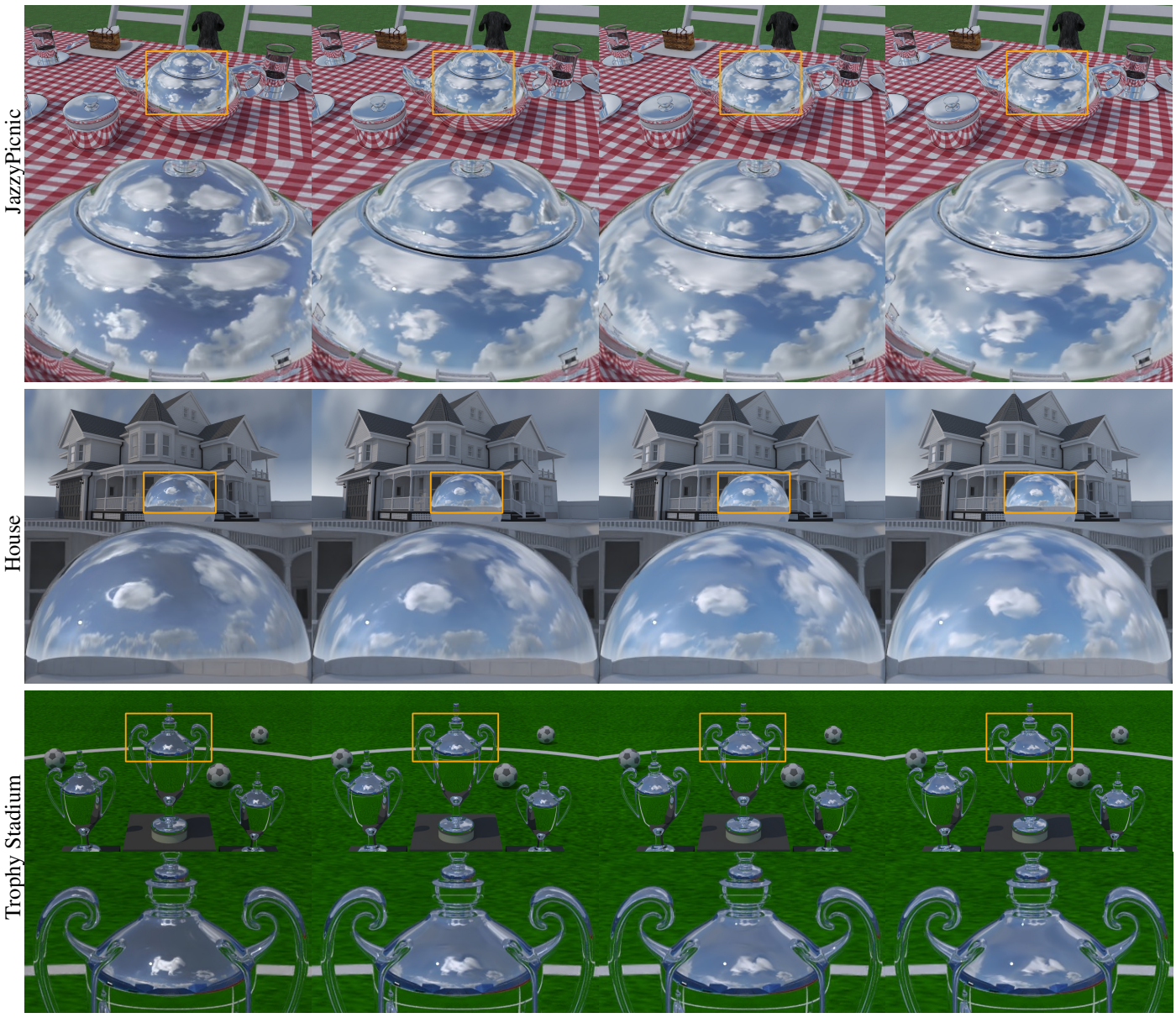

Fig. 11. Renderings showing the use of outputs from a deep learning based cloud ([5]) synthesis model, in this case showing clouds generated from a mask of a "Thought Bubbles" (top), Teapot (middle), and Dog (bottom). Our method provides photorealistic cloud animation even in this very artificial scenario.

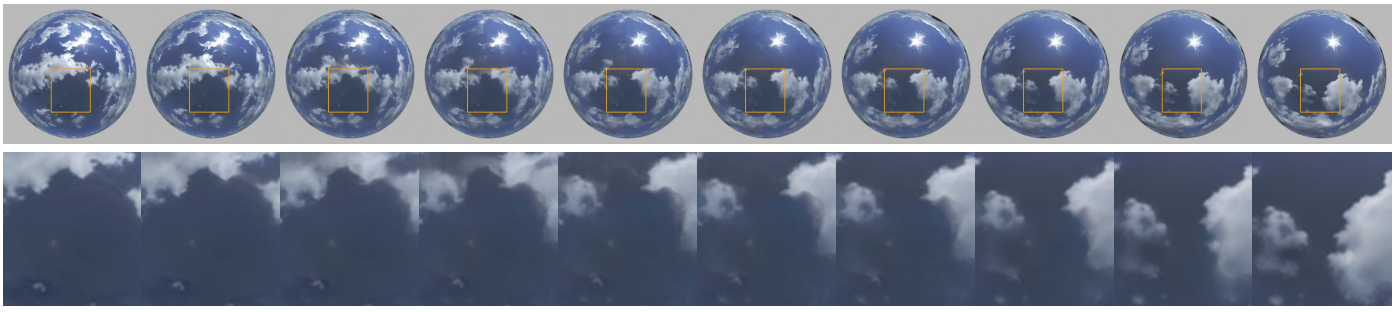

Fig. 12. Example of using a frame interpolation method ([57]) applied to two frames generated by a generative cloud method. While the motion is smooth, the predicted cloud movement is frequently not across the sky, but instead clouds disappear from one location and appear in another as can be seen in the inset images.

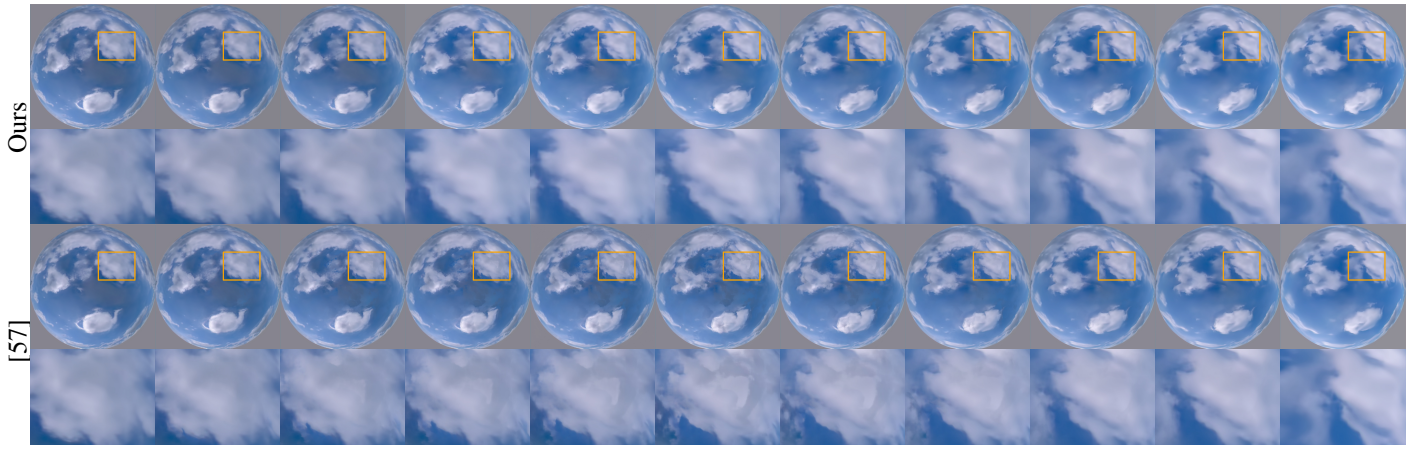

Fig. 13. Results comparing our approach (top) with frame interpolation (bottom) between two medium timescale images. Our approach leads to smooth cloud movement over the hemisphere, whereas frame interpolation can lead to artifacts as is shown in the insets.

a flow field. We also intend to extend the CloudNet network to enhance temporal consistency by incorporating information such as histograms from multiple preceding frames. Finally, we are interested in combining our image based method with light transport methods to additionally synthesize 3D volumetric clouds.

References

[1] Debevec, P. 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, p. 1–10.

[2] Hosek, L, Wilkie, A. An analytic model for full spectral sky-dome radiance. ACM Transactions on Graphics (TOG) 2012;31(4):1–9.

[3] Satilmis, P, Bashford-Rogers, T, Chalmers, A, Debattista, K. A machine-learning-driven sky model. IEEE computer graphics and applications 2016;37(1):80–91.

[4] Wilkie, A, Vevoda, P, Bashford-Rogers, T, Hošek, L, Iser, T, Kolářová, M, et al. A fitted radiance and attenuation model for realistic atmospheres. ACM Transactions on Graphics (TOG) 2021;40(4):1–14.

[5] Satilmis, P, Marnerides, D, Debattista, K, Bashford-Rogers, T. Deep synthesis of cloud lighting. IEEE Computer Graphics and Applications 2022;.

[6] Mirbauer, M, Rittig, T, Iser, T, Kriváneк, J, Šikudová, E. SkyGAN: Towards realistic cloud imagery for image based lighting. In: Eurographics Symposium on Rendering. The Eurographics Association. 2022,.

[7] Brooks, T, Peebles, B, Holmes, C, DePue, W, Guo, Y, Jing, L, et al. Video generation models as world simulators 2024;URL: `https://openai.com/research/video-generation-models-as-world-simulators`.

[8] Cho, J, Puspitasari, FD, Zheng, S, Zheng, J, Lee, LH, Kim, TH, et al. Sora as an AGI world model? a complete survey on text-to-video generation. arXiv preprint arXiv:240305131 2024;.

[9] Zhang, J, Lalonde, JF. Learning high dynamic range from outdoor panoramas. In: Proceedings of the IEEE International Conference on Computer Vision. 2017, p. 4519–4528.

[10] Perez, R, Seals, R, Michalsky, J. All-weather model for sky luminance distribution—preliminary configuration and validation. Solar energy 1993;50(3):235–245.

[11] Nishita, T, Sirai, T, Tadamura, K, Nakamae, E. Display of the earth taking into account atmospheric scattering. In: Proceedings of the 20th annual conference on Computer graphics and interactive techniques. 1993, p. 175–182.

[12] Nishita, T, Dobashi, Y, Nakamae, E. Display of clouds taking into account multiple anisotropic scattering and sky light. In: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques. 1996, p. 379–386.

[13] Haber, J, Magnor, M, Seidel, HP. Physically-based simulation of twilight phenomena. ACM Transactions on Graphics (TOG) 2005;24(4):1353–1373.

[14] Preetham, AJ, Shirley, P, Smits, B. A practical analytic model for daylight. In: Proceedings of the 26th annual conference on Computer graphics and interactive techniques. 1999, p. 91–100.

[15] Ebert, DS. Volumetric modeling with implicit functions: a cloud is born. In: SIGGRAPH Visual Proceedings. 1997, p. 147.

[16] Voss, R. Fourier synthesis of gaussian fractals: 1/f noises, landscapes, and flakes. Tutorial on State of the Art Image Synthesis, SIGGRAPH'83 1983;.

[17] Gardner, GY. Visual simulation of clouds. In: Proceedings of the 12th annual conference on Computer graphics and interactive techniques. 1985, p. 297–304.

[18] Sakas, G. Modeling and animating turbulent gaseous phenomena using spectral synthesis. The Visual Computer 1993;9:200–212.

[19] Schpok, J, Simons, J, Ebert, DS, Hansen, C. A real-time cloud modeling, rendering, and animation system. In: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation. 2003, p. 160–166.

[20] Dobashi, Y, Nishita, T, Yamashita, H, Okita, T. Using metaballs to modeling and animate clouds from satellite images. The Visual Computer 1999;15:471–482.

[21] Wither, J, Bouthors, A, Cani, MP. Rapid sketch modeling of clouds. In: Eurographics Workshop on Sketch-Based Interfaces and Modeling (SBIM). Eurographics Association; 2008, p. 113–118.

[22] Dobashi, Y, Shinzo, Y, Yamamoto, T. Modeling of clouds from a single photograph. In: Computer Graphics Forum; vol. 29. Wiley Online Library; 2010, p. 2083–2090.

[23] Yuan, C, Liang, X, Hao, S, Qi, Y, Zhao, Q. Modelling cumulus cloud shape from a single image. In: Computer Graphics Forum; vol. 33. Wiley Online Library; 2014, p. 288–297.

[24] Dobashi, Y, Kaneda, K, Yamashita, H, Okita, T, Nishita, T. A simple, efficient method for realistic animation of clouds. In: Proceedings of the 27th annual conference on Computer graphics and interactive techniques. 2000, p. 19–28.

[25] Harris, MJ, Lastra, A. Real-time cloud rendering. In: Computer graphics forum; vol. 20. Wiley Online Library; 2001, p. 76–85.

[26] Dobashi, Y, Kusumoto, K, Nishita, T, Yamamoto, T. Feedback control of cumuliform cloud formation based on computational fluid dynamics. ACM Transactions on Graphics (TOG) 2008;27(3):1–8.

[27] Hädrich, T, Makowski, M, Pałubicki, W, Banuti, DT, Pirk, S, Michels, DL. Stormscapes: Simulating cloud dynamics in the now. ACM Transactions on Graphics (TOG) 2020;39(6):1–16.

[28] Goswami, P. A survey of modeling, rendering and animation of clouds in computer graphics. The Visual Computer 2021;37(7):1931–1948.

[29] Riley, K, Ebert, DS, Kraus, M, Tessendorf, J, Hansen, CD. Efficient rendering of atmospheric phenomena. Rendering Techniques 2004;4:374–386.

[30] Elek, O, Kmoch, P. Real-time spectral scattering in large-scale natural participating media. In: Proceedings of the 26th Spring Conference on Computer Graphics. 2010, p. 77–84.

[31] Novák, J, Georgiev, I, Hanika, J, Jarosz, W. Monte Carlo methods for volumetric light transport simulation. In: Computer Graphics Forum; vol. 37. Wiley Online Library; 2018, p. 551–576.

[32] Hillaire, S. Physically based sky, atmosphere and cloud rendering in frostbite. In: ACM SIGGRAPH. 2016, p. 1–62.

[33] Kallweit, S, Müller, T, Mcwilliams, B, Gross, M, Novák, J. Deep scattering: Rendering atmospheric clouds with radiance-predicting neural networks. ACM Transactions on Graphics (TOG) 2017;36(6):1–11.

[34] Bitterli, B, Ravichandran, S, Müller, T, Wrenninge, M, Novák, J, Marschner, S, et al. A radiative transfer framework for non-exponential media 2018;.

[35] Jarabo, A, Aliaga, C, Gutierrez, D. A radiative transfer framework for spatially-correlated materials. ACM Transactions on Graphics 2018;37(4).

[36] Webanck, A, Cortial, Y, Guérin, E, Galin, E. Procedural cloudscapes. In: Computer Graphics Forum; vol. 37. Wiley Online Library; 2018, p. 431–442.

[37] Mirbauer, M, Rittig, T, Iser, T, Křivánek, J, Šikudová, E. SkyGAN: Realistic Cloud Imagery for Image-based Lighting. In: Computer Graphics Forum; vol. 43. Wiley Online Library; 2024, p. e14990.

[38] Valença, L, Maquignaz, I, Moazen, H, Madan, R, Hold-Geoffroy, Y, Lalonde, JF. LM-GAN: A photorealistic all-weather parametric sky model. arXiv preprint arXiv:230200087 2023;.

[39] Chen, Z, Wang, G, Liu, Z. Text2light: Zero-shot text-driven HDR panorama generation. ACM Transactions on Graphics (TOG) 2022;41(6):1–16.

[40] Goswami, P, Cheddad, A, Junede, F, Asp, S. Interactive landscape–scale cloud animation using dcgan. Frontiers in Computer Science 2023;5.

[41] Blattmann, A, Rombach, R, Ling, H, Dockhorn, T, Kim, SW, Fidler, S, et al. Align your latents: High-resolution video synthesis with latent diffusion models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2023, p. 22563–22575.

[42] Chen, Q, Koltun, V. Photographic image synthesis with cascaded refinement networks. In: Proceedings of the IEEE international conference on computer vision. 2017, p. 1511–1520.

[43] Park, T, Liu, MY, Wang, TC, Zhu, JY. Semantic image synthesis with spatially-adaptive normalization. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019, p. 2337–2346.

[44] Tewari, A, Fried, O, Thies, J, Sitzmann, V, Lombardi, S, Sunkavalli, K, et al. State of the art on neural rendering. In: Computer Graphics Forum; vol. 39. Wiley Online Library; 2020, p. 701–727.

[45] Tewari, A, Thies, J, Mildenhall, B, Srinivasan, P, Tretschk, E, Yifan, W, et al. Advances in neural rendering. In: Computer Graphics Forum; vol. 41. Wiley Online Library; 2022, p. 703–735.

[46] Singer, U, Polyak, A, Hayes, T, Yin, X, An, J, Zhang, S, et al. Make-a-video: Text-to-video generation without text-video data. arXiv preprint arXiv:220914792 2022;.

[47] Ho, J, Chan, W, Saharia, C, Whang, J, Gao, R, Gritsenko, A, et al. Imagen video: High definition video generation with diffusion models. arXiv preprint arXiv:221002303 2022;.

[48] Farnebäck, G. Two-frame motion estimation based on polynomial expansion. In: Scandinavian conference on Image analysis. Springer; 2003, p. 363–370.

[49] Ronneberger, O, Fischer, P, Brox, T. U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention. Springer; 2015, p. 234–241.

[50] Marnerides, D, Bashford-Rogers, T, Hatchett, J, Debattista, K. 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, p. 37–49.

[51] Isola, P, Zhu, JY, Zhou, T, Efros, AA. Image-to-image translation with conditional adversarial networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2017, p. 1125–1134.

[52] Johnson, R, Hering, WS. Automated cloud cover measurements with a solid-state imaging system. In: Proceedings of the Cloud Impacts on DOD Operations and Systems—1987, Workshop. 1987, p. 59–69.

[53] Ricoh Theta Z1. 2022. URL: `https://theta360.com/en/about/theta/z1.html`; accessed: 23-12-2022.

[54] Khan, Z, Khanna, M, Raman, S. FHDR: HDR image reconstruction from a single LDR image using feedback network. In: 2019 IEEE Global Conference on Signal and Information Processing (GlobalSIP). 2019, p. 1–5. doi:`10.1109/GlobalSIP45357.2019.8969167`.

[55] Guo, C, Fan, L, Xue, Z, Jiang, X. Learning a practical sdr-to-hdrtv up-conversion using new dataset and degradation models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2023, p. 22231–22241.

[56] Hošek, L, Wilkie, A. Adding a solar-radiance function to the hošek-wilkie skylight model. IEEE computer graphics and applications 2013;33(3):44–52.

[57] Reda, F, Kontkanen, J, Tabellion, E, Sun, D, Pantofaru, C, Curless, B. Film: Frame interpolation for large motion. In: European Conference on Computer Vision. Springer; 2022, p. 250–266.

[58] Parihar, AS, Varshney, D, Pandya, K, Aggarwal, A. A comprehensive survey on video frame interpolation techniques. The Visual Computer 2022;:1–25.