HAL
open science

Interactive Meso-scale Simulation of Skyscapes

Ulysse Vimont, James Gain, Maud Lastic, Guillaume Cordonnier, Babatunde Abiodun, Marie-Paule Cani

To cite this version:

Ulysse Vimont, James Gain, Maud Lastic, Guillaume Cordonnier, Babatunde Abiodun, et al.. Interactive Meso-scale Simulation of Skyscapes. Computer Graphics Forum, 2020, 39 (2). hal-02517935

HAL Id: hal-02517935

https://hal.science/hal-02517935v1

Submitted on 24 Mar 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

EUROGRAPHICS 2020 / U. Assarsson and D. Panozzo
(Guest Editors)

Volume 39 (2020), Number 2

Interactive Meso-scale Simulation of Skyscapes

Ulysse Vimont[1], James Gain[1], Maud Lastic[2], Guillaume Cordonnier[3], Babatunde Abiodun[4], Marie-Paule Cani[2]

[1]Computer Science Department, University of Cape Town
[2] LIX, École Polytechnique, CNRS [3] ETH Zürich
[4] Environmental and Geographical Sciences Department, University of Cape Town

Abstract

Although an important component of natural scenes, the representation of skyscapes is often relatively simplistic. This can be largely attributed to the complexity of the thermodynamics underpinning cloud evolution and wind dynamics, which make interactive simulation challenging. We address this problem by introducing a novel layered model that encompasses both terrain and atmosphere, and supports efficient meteorological simulations. The vertical and horizontal layer resolutions can be tuned independently, while maintaining crucial inter-layer thermodynamics, such as convective circulation and land-air transfers of heat and moisture. In addition, we introduce a cloud-form taxonomy for clustering, classifying and upsampling simulation cells to enable visually plausible, finely-sampled volumetric rendering.
As our results demonstrate, this pipeline allows interactive simulation followed by up-sampled rendering of extensive skyscapes with dynamic clouds driven by consistent wind patterns. We validate our method by reproducing characteristic phenomena such as diurnal shore breezes, convective cells that contribute to cumulus cloud formation, and orographic effects from moist air driven upslope.

Keywords: Eulerian simulation, procedural modeling

1. Introduction

Weather, and its impact on the skyscape, plays a key role in the digital depiction of outdoor landscapes. Both the sweep of the sky and the way it scatters light and shadow across natural elements are crucial to the mood of scenes in games and feature films. Moreover, wind and clouds are even more critical to specific applications, such as flight simulators.

Although extensively investigated in Computer Graphics, physically-realistic clouds cannot yet be generated and animated at interactive rates. Despite advances in the modeling of cloud-like shapes, automatic detail enhancement and volumetric rendering, no solution exists for the efficient and detailed simulation of wind effects and attendant cloud evolution. Procedural methods are prevalent in practice and rely on user specified key-frames with inbetweening through linear blending or optimal transport. Despite recent improvements, these techniques are unable to generate the full variety and complexity of dynamic skies observable in nature.

Cloud development is governed by specific physical laws and processes [WH06]. Clouds are a physical manifestation of condensation of water vapour in air. They form when a moist air parcel reaches a saturation point, either by cooling to its dew point temperature or by evaporating water. Cloud formation and type depend on atmospheric conditions, as well as the structure of the underlying terrain. For instance, atmospheric temperature and moisture profiles, evaporation from water bodies, and interaction between the wind and mountain slopes, dictate the formation and shape of clouds over time. To fully simulate these phenomena is a daunting prospect. The physical processes involved, including thermal and fluid dynamics, are complex and tightly coupled. The simulation domain needs to be both large in extent and finely sampled to adequately capture cloud structure and interactivity is difficult to achieve given the computational demands.

The key insight of this work is to decouple large-scale, versus small-scale features. We model the former using a coarse physically-based simulation, which yields plausible wind and cloud shapes while remaining tractable. In contrast, small-scale features such as precise cloud appearance, are generated procedurally based on the output of the simulation. This enables us to maintain a coherent appearance across scales, while achieving inexpensive computation compared to a simulation of equivalent resolution.

More specifically, we propose a new layered atmospheric model that supports efficient simulation, combined with a mechanism for consistent rendering through detail amplification based on cloud-form classification. Although our layered representation is vertically sparse, the attendant simulation captures the underlying physics behind wind and cloud formation, such as interactions between atmosphere and terrain. Our framework also yields visually plausible results thanks to a consistent, procedural volumetric up-

© 2020 The Author(s)
Computer Graphics Forum © 2020 The Eurographics Association and John Wiley & Sons Ltd. Published by John Wiley & Sons Ltd.

Figure 1: *A $50 \times 50 \times 10$km skyscape obtained from our simulation, showing multiple layers with different cloudforms.*

sampling mechanism: simulation outputs such as wind velocity, convection and altitude are used to detect cloud-type, enabling the derivation of consistent noise parameters for cloud density. Finally, volumetric rendering can be applied directly to the up-sampled density datasets.

Our framework not only captures the variety of cloud types that are visually salient in nature, but also incorporates other important meteorological effects, such as wind. This is achieved at meso-scales (up to $50 \times \times 50 \times 10$km) with a level of resolvable cloud detail suitable for virtual environments. The simulation is run at near interactive rates ($< 10s$ per frame) using accelerated time scales (where each time-step represents from 30 seconds to 1 hour of the final real-speed animation), enabling the user to iteratively author a variety of meteorological scenarios at coarse scales. The results can then be refined both spatially and temporally for final rendering. As our results show, the simulation seamlessly captures both stratiform and convective cloud formations (see Figures 1 and 2). Furthermore, our model accounts for physical effects, such as phase changes (evaporation and condensation), latent heat exchange between ground and atmosphere, and orographic uplift, where mountains direct the air upwards.

In summary, our technical contributions include:

- A layered model of the atmosphere enabling the independent tuning of horizontal and vertical resolutions, and an associated fluid thermodynamics simulation method, which efficiently captures both intra-layer advection and inter-layer convection;
- A practical solution for modeling meteorological interaction with terrains of varying topography and surface coverage;
- A mechanism for classifying clouds based on a standard taxonomy that enables the coherent up-scaling of our simulation results.

2. Related Work

Being able to effectively model, animate and render clouds is essential to the visual portrayal of skyscapes. In particular, the interaction of light with clouds is essential to their appearance. This includes diffuse light transport [NDN96], self-shadowing [HBSL03], anisotropic scattering [BNL06, BNM*08] and lightning strike effects [DEYN07]. Most recently, deep neural networks have seen use as an alternative solution for cloud rendering [KMM*17]. However, our focus is on modeling and animation and so we employ an existing rendering engine for clouds [Pla19].

There is a broad divide in the cloudscape modeling and animation literature between procedural methods, where generative algorithms are designed using a non-physical phenomenological approach, and more physical simulation methods, which adapt fluid dynamics with either a grid (Eulerian) or particle (Lagrangian) formulation. There are also some procedural-simulation hybrids, such as ours, which employ simulation for broad-scale effects and procedural amplification for fine-scale detail.

Procedural methods typically represent stratiform clouds using slabs of multiresolution noise [BNL06, LLC*10], which are amenable to locally consistent animation, and convective clouds using a scaffold of implicit shapes, such as proto-particles [Ney97], warped blobs [BN04], or ellipsoids [Gar85, SSEH03, LJWcH07] that can be blended to define a volumetric density field, with an overlay of procedural noise for high-frequency detail.

Such approaches have the advantage of real-time performance, due to the efficiency of noise functions and the compact support of implicit shapes. Moreover, they support user control through sketching [WBC08, SBRS10], painting [WGM14] and keyframing [WCGG18] interfaces. It is even possible to roughly shape clouds to fit a target, supplied as a mesh or photograph [DSY10, CMCM11, YLH*14]. Unfortunately, realism remains an issue. With sufficient care on the part of an artist a convincing static cloudscape can be designed but physical plausibility often breaks down during animation. Even with specialised morphing schemes, such as optimal mass transport along anisotropic shortest paths [WCGG18] that account for input winds and terrain, the lack of physical representation makes it difficult to capture certain interactions, such as the complex behavior of clouds surrounding mountains.

Simulation methods take a physical approach to cloudscape evolution by considering the interplay between heat, water vapor, and wind. This tends to be more involved than fluid or smoke simulation because state changes through precipitation, and additional thermodynamics, such as terrestrial radiation, must be accounted for. Such approaches can be further categorised as Lagrangian (particle-based) or Eulerian (grid-based).

While Lagrangian approaches may seem better suited in theory to large volumes since clear skies could be sampled using fewer particles, they have trouble capturing the full range of cloud types and predominantly focus on cumuliform clouds. An early example [Ney97] represents air parcels using bubble-like particles. More recently, Welton et al. [WBDY15] and Barbosa et al. [FBDY15] incorporate dynamic splitting and merging strategies to cope with variable density over time. Duarte and Gomes [DG17] take a data-driven approach, using sounding curves that provide atmospheric measurements, such as horizontal wind direction, pressure, and temperature, sampled along a vertical profile. This allows a simplified and accelerated simulation with independent particles agglomerating around a sounding curve to form individual cumulus

© 2020 The Author(s)
Computer Graphics Forum © 2020 The Eurographics Association and John Wiley & Sons Ltd.

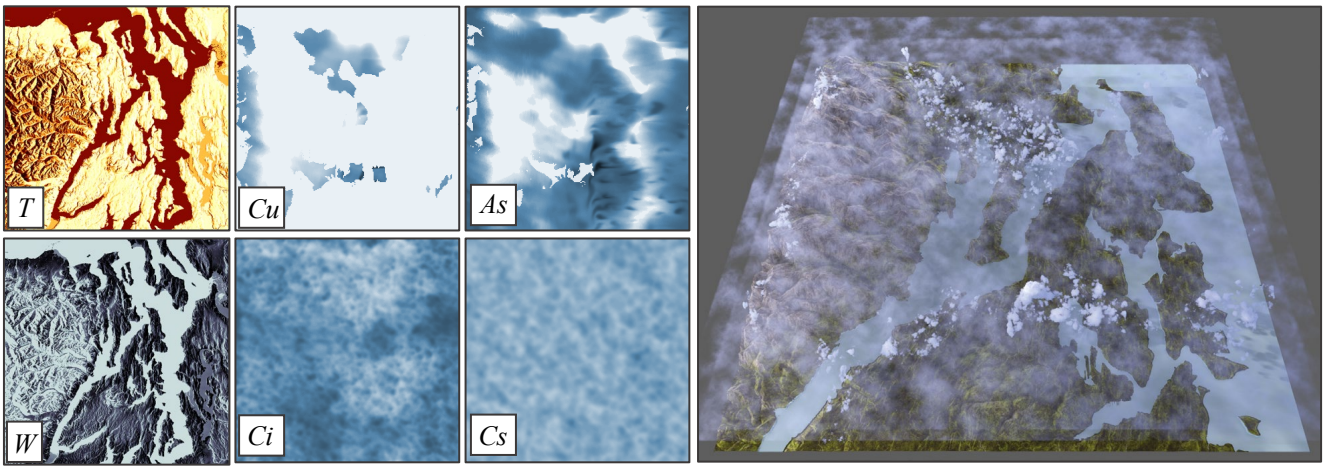

Figure 2: *(Left) Our meso-scale simulation engine generates temperature (T) and water (W) terrain data, as well as cumulus (Cu), altostratus (As), cirrus (Ci) and cirrostratus (Cs) cloud layers. (Right) These sparse outputs are procedurally amplified to enable the rendering of a detailed volumetric skyscape.*

Category	Method	Cloud Type	Scale	References
Procedural	Noise	Stratiform	Meso	[BNL06]
	Implicit Shapes	Convective	Micro, Meso	[Ney97, BN04, LJW06, YLH*14]
	Noise & Implicit Shapes	Mixed	Meso	[Gar85, WCGG18, SSEH03]
Simulation	Eulerian	Mixed	Micro	[DKY*00, MYND01, MIY02]
	Eulerian	Mixed	Meso	[GDAB*17]
	Lagrangian	Convective	Micro, Meso	[Ney97, WBDY15, FBDY15, DG17]
Hybrid	Lagrangian & Noise	Convective	Meso	[GN17]
	Eulerian & Noise	**Mixed**	**Meso**	**Ours**

Table 1: *A comparison of cloudscape approaches, comparing their underlying method, achievable cloud type (stratiform, convective or a combination of both), simulation scale (micro up to 1km, meso up to 100km and synoptic scales beyond) and research representative of each category.*

clouds. On the other hand, it leans heavily on the input data and the advective wind patterns are oversimplified.

In contrast, Eulerian approaches enable more accurate simulation mechanisms. They include the use of real-valued cellular automata [DKY*00, MYND01], and voxel grids encoding wind velocity, vapor density, pressure and temperature [MIY02, HBSL03, DYN06]. In theory the underlying grid makes these approaches well suited to GPU implementation. Unfortunately, in practice the extremely large volumetric domain generally forces a coarse sampling on such GPU implementations. For instance, Garcia-Dorado *et al.* [GDAB*17] adapt and simplify the Weather Forecasting Model (WFM) from meteorology to simulate wind effects and the formation of stratiform and convective clouds over cities, using a meso-scale domain (up to 100km $\times$ 100 $\times$ 10km at approximately 1km $\times$ 1km horizontal resolution per cell). We exploit a similar grid-like structure to capture changes in irradiation, evaporation, condensation, and precipitation, but seek to provide clouds with finer detail over more general landscapes.

Hybrid Schemes overcome the realism issues of procedural methods and the scale issues of simulation, by coupling coarse-scale simulation with detail amplification using volumetric noise. Unlike the previous hybrid scheme [GN17], which uses Lagrangian simulation of air parcels with noise-based procedural detail for convective cloud simulation, we are the first to hybridise with an Eulerian simulation, making our system capable of combining different cloud types (both convective and stratiform) within the same framework. Crucially, unlike most Eulerian schemes, we do not use a regular sampling of the volumetric domain, but instead employ a stack of irregularly-distributed Eulerian layers. This better captures the atmospheric physics, is more compact and therefore efficient, and allows us to extract convection and use it to differentiate between cloudforms during amplification.

3. Meteorological Framework

3.1. A Primer on Meso-scale Meteorology

A skyscape (also sometime called a cloudscape) is the vista that can be seen from a given viewpoint, often located on the ground. Its visual appearance can undergo dramatic change over the course of a single day due to day-night cycles and meteorological phenomena designated as *weather*.

© 2020 The Author(s)
Computer Graphics Forum © 2020 The Eurographics Association and John Wiley & Sons Ltd.

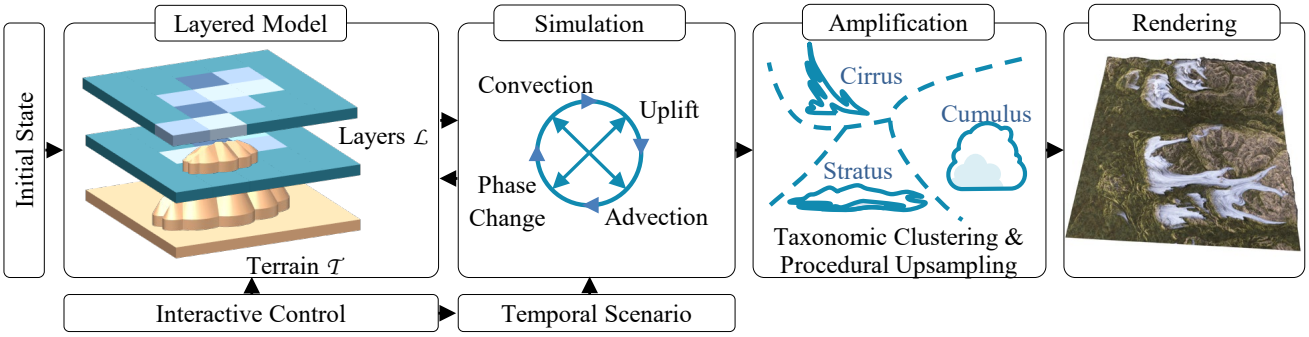

Figure 3: *Overview: A large-scale layered atmospheric model is updated at interactive rates by a meteorological simulation. The later can be used to feed a fine-scale procedural volumetric amplification process, used for the offline rendering of animated skyscapes.*

Weather takes place in the troposphere, the lowest 10km of the atmosphere that contains more than 75% of its mass. The upper boundary of the troposphere is the tropopause, where a strong increase in temperature with height prevents any ascending air movement. The lower boundary of the troposphere is the earth's surface (ground or water body) with which the troposphere exchanges heat, moisture and momentum. Most of the energy that drives weather is from the sun and reaches the atmosphere and earth surface in the form of radiation. The ground receives almost all the solar energy and transfers it back to the atmosphere through radiation as well as conduction to the lowest layer of air. In addition to heat, the ground also exchanges moisture through evaporation and condensation. However, these transfers induce energy discrepancies between air parcels at different locations, which in turn fuels complex air movements in both horizontal (wind) and vertical directions (thermals). Air further from the surface is cooler due to less pressure being exerted by the remaining air above it. Warm and cold air are hardly miscible, and the differences in temperature lead to a stratified structure in the atmospheric temperature profile. Present in most circumstances, this horizontal stratification of air becomes more obvious when a sea of clouds or a low-altitude smog trap appear due to the presence of an inversion layer (an air layer in which temperature increases with altitude).

When energy discrepancies are sufficient to force moist air to rise above its lifting condensation level (the altitude where relative humidity reaches 100%), vapor will condense and clouds appear. The surface of a cloud follows the general air motion and is therefore a good indicator of the origin of the rising air. Convective motion is very turbulent due to the internal and boundary velocity discrepancies and tends to create cumuliform clouds, whereas synoptic wind forcing air above a relief or a colder air mass tends to create stratiform clouds.

Water droplets tend to coalesce, increasing their weight to surface ratio until vertical air motion is too low to maintain them in the air: this is precipitation. This motion of the water being evaporated, transported, and precipitated is known as the water cycle. It is more complex that the explanation given here, but out of the scope of this paper. For further details on meso-scale atmospheric physics, we refer the reader to works by Barry *et al.* [Bar92, BC09] and Pielke *et al.* [PSA*98].

In summary, generating a descernible skyscape where the different cloudforms are visually distinguishable requires capturing fine-scale (~10m) cloud detail in an horizontal span up to 50 km^2, with a vertical extent of the entire troposphere (10 km). The resulting scale ratio (~10^{10}) makes this ill suited to computer memory storage and processing. In the subsequent section, we present a strategy for dramatically reducing the required memory footprint of a skyscape representation, while still allowing a physically-based simulation of its temporal evolution.

3.2. Method Overview

Our general approach (as outlined in Figure 3) is to apply meteorological simulation over a sparse set of atmospheric layers at meso-scales (approximately 200×200m horizontal resolution per cell) followed by noise-based upsampling keyed to quantifiable attributes, such as convection, advection, and altitude, in order to achieve micro-scale volumetric rendering.

Our layered atmospheric model and matching simulation (Section 4) are motivated by the stratification evident in meso-scale meteorology. For a given skyscape scenario a sparse irregular stack of horizontal layers is constructed, with each layer structured as a 2D regular grid of meteorological data. This serves the needs of both modelling and implementation: layers can be situated to best capture meteorological features, while their overall sparsity improves simulation efficiency and the individual layers themselves are a good fit for GPU texture memory.

We tailor our meteorological simulation to this novel representation, in order to consistently evolve layer data over time at interactive rates. Motion within layers is handled through an extension of standard 2D Eulerian divergence-free computational fluid dynamics [And18] to encompass atmospheric quantities, such as temperature, absolute humidity, and water content. Motion between layers is divided into two categories: convective motion modeled trough a difference in moist static energy and non-convective motions due to the orographic effects of terrain.

The handling of terrain requires special consideration (Section 5). Not only are there various heat and moisture fluxes between land and air, such as irradiation, evaporation and condensation, but topography also shapes winds and introduces orographic

© 2020 The Author(s)
Computer Graphics Forum © 2020 The Eurographics Association and John Wiley & Sons Ltd.

effects. We represent terrain as a height-field that may cut through one or more atmospheric layers, with horizontal advection and vertical convection being adapted accordingly.

As a last step we apply procedural amplification to increase volumetric detail preparatory to rendering (Section 6). Cloud forms are governed primarily by a combination of altitude, turbulence due to convection, and wind strength. These quantities are readily available in our simulation and can be used to assign a cloud type according to the standard taxonomy (see Figure 5). Then simulation cells are upsampled using harmonic noise with cloud-type-specific parameters. For example, regions with sufficient condensation, strong directional advection and low overall convection, will be populated by relatively low-frequency wispy noise expected in stratiform clouds. Finally, at each simulation step, the scene is rendered using ray-marching of the procedurally-amplified volume.

4. Efficient Atmospheric Simulation

4.1. Layered Model

As already stressed, clouds in nature are structured in horizontal layers and atmospheric quantities, such as temperature, do not decay linearly with altitude. Thus, even if a regular vertical sampling (and the attendant $3D$ voxel grid) did not suffer from serious memory overheads, it would still be a poor fit to the meteorological context. This was already recognized by Garcia-Dorado *et al.* [GDAB*17], who uses voxels of exponentially increasing thicknesses. Furthermore, the merits of such uneven atmospheric sampling are recognized in the meteorological simulation literature [RBD*18], albeit in a different application context. We generalize this by representing the atmosphere as any irregular stack of horizontal $2D$ layers.

A benefit of our approach is that we can pack each layer into a texture and exploit GPU acceleration to perform $2D$ within-layer advection of temperature and moisture. Vertical between-layer transfers must account for the uneven spacing between layers and are handled separately.

In our method, the user specifies the number of atmospheric layers (n), the layer resolution, and the altitude of each layer. For simulation purposes, each of the atmospheric layers $\mathcal{L}_i$, $i = 0, \ldots, n-1$, is then encoded as a grid with the following data fields (see Table 2): *temperature* T_i in Kelvin reflecting the mean temperature of a cell; *absolute humidity* H_i in g/m^3, which is the quantity of water vapour (uncondensed moisture) present in a cell; *water content* W_i indicating the quantity of condensed (liquid) water in g/m^3 and *2D velocity* $\mathbf{V}_i \in \mathbb{R}^2$, which is a measure of the local planar wind direction and strength and is used to advect the other quantities.

4.2. Meteorological Simulation Engine

Our simulation engine is built on the advection of atmospheric quantities using standard computational fluid dynamics (CFD), with two significant amendments. First, we account for phase changes between the liquid and vapour states of water, including the attendant release and absorption of latent heat. Second, we transfer quantities vertically between adjacent layers in order to simulate convective and dynamic uplift arising from vertical gradients in temperature, moisture, and pressure.

Category	Notation	Quantity	Units
Atmospheric layer $\mathcal{L}_i$	Θ_i	altitude (constant)	m
	T_i	temperature	K
	H_i	absolute humidity	g/m^3
	W_i	water content	g/m^3
	$\mathbf{V}_i$	2D velocity	m/s (2D)
	U_i	updraft velocity	m/s
	C_i	convection velocity	m/s
	D_i	vertical velocity	m/s
	P_i	dynamic pressure	Pa
	Π_i	atmospheric pressure	Pa
	σ_i	orographic velocity damping	none
Terrain $\mathcal{T}$	$\Theta_\mathcal{T}$	altitude	m
	$\Lambda_\mathcal{T}$	heat capacity	$J/kg/K$
	$T_\mathcal{T}$	temperature	K
	$M_\mathcal{T}$	water content	g/m^2
	ε	emissivity	none
	B	albedo	none
Coefficients	λ	convective coupling	$m.s^{-1}.J^{-1}$
	ν	non convective coupling	$m.s^{-1}.Pa^{-1}$
	μ	pressure coupling	$Pa.s$
	γ	humidity transfer rate	s^{-1}
	κ	heat transfer rate	$K.J^{-1}$
	Δt	time step	s

Table 2: *Symbols used in this paper.*

At each simulation timestep we apply $2D$ Eulerian CFD [And18] to each atmospheric layer $\mathcal{L}_i$, by computing a dynamic pressure field P_i using the Poisson equation:

$$\nabla^2 P_i = \nabla \cdot \mathbf{V}_i. \tag{1}$$

This is solved in the classical manner using a multi-grid method. Next, the pressure field is used to make the velocity field divergence free (∇P_i is subtracted from $\mathbf{V}_i$). In typical CFD, the velocity and a generic concentration field would then be advected. Instead of working on anonymous concentration we advect temperature T_i , absolute humidity H_i , and water content W_i in addition to velocity $\mathbf{V}_i$. This is not a significant point of departure: advection of such quantities is not uncommon in Eulerian cloud simulations [MYND01, MIY02, GDAB*17].

Phase changes: Our first substantive addition to the Eulerian scheme is to incorporate evaporation and condensation within layers. These occur when the saturation humidity H_i' for a given cell is above or below, respectively, the absolute humidity H_i . The latter is already stored as a layer quantity and subject to advection. What remains is to calculate the saturation humidity for a cell.

First, for each layer $\mathcal{L}_i$ at altitude Θ_i , we compute the average atmospheric pressure using the barometric formula [BC09]:

$$\Pi_i = \Pi_{sea} \cdot \exp\left[\frac{-g \cdot M_{air} \cdot T_{sea}}{R_0 \cdot T_0}\right], \tag{2}$$

© 2020 The Author(s)
Computer Graphics Forum © 2020 The Eurographics Association and John Wiley & Sons Ltd.

where Π_{sea} and T_{sea} are pressure and temperature at sea-level, g is gravity, M_{air} is the molar mass of air, and R_0 is the Gas constant.

Then, for each grid cell p within each layer i we compute the saturated vapor pressure Π'_i using Teten's formula (the temperature is given in Celsius in this equation, while we use Kelvins elsewhere) [BC09]:

$$\Pi'_i(p) = 0.61078 \exp\left(\frac{17.27 T_i(p)}{T_i(p) + 237.3}\right). \tag{3}$$

This equation holds for positive temperatures, and a variant exists for the negative case. In practice, we switch between them as appropriate. Finally, we are able to calculate the saturation humidity as [BC09]:

$$H'_i(p) = \frac{M_{water}}{M_{air}} \cdot \frac{\Pi'(p)}{\Pi_i - \Pi'(p)}, \tag{4}$$

with M_{water} and M_{air} as the molar mass of water and dry air, respectively.

We then compare the saturated and absolute humidity and restore equilibrium by transferring $\Delta H_i = \gamma(H'_i - H_i)\Delta t$ from W_i to H_i at each time step according to a rate parameter γ (with $\gamma = 0.1 s^{-1}$ in our simulation). This has the effect of decreasing water content and increasing absolute humidity during evaporation ($H'_i > H_i$), and vice versa for condensation ($H'_i < H_i$).

Inter-layer exchanges: Atmospheric layers that are entirely independent fail to capture uplift phenomena, such as convection, where relatively warm, moist parcels of air rise while cooler dryer parcels descend.

To account for such vertical transfers of temperature and moisture we incorporate a first exchange field, the *Convection factor*, into each atmospheric layer. This field depends on the interaction between the current layer $\mathcal{L}_i$ and the one above $\mathcal{L}_{i+1}$.

Convection is proportional to the disequilibrium in Mean Static Energy between layers since it is this that causes air parcels to rise or fall: $C_i = \lambda(MSE_{i+1} - MSE_i)$, where λ is the convective coupling coefficient. The standard form for MSE is [BC09]:

$$MSE_i = C_p \cdot T_i + g \cdot \Theta_i + L_v \cdot Q_i, \tag{5}$$

expressed in J/kg, and where L_v is the latent heat of vaporization, Q_i is the specific humidity of the air layer (and can be derived from H_i and Θ_i), and other quantities have already been defined. This serves to combine temperature, altitude and humidity into a single equation.

Vertical air motion can also be non-convective, solely and directly due to dynamic pressure differences between air layers. We express the local pressure difference as $U_i = \nu \cdot (P_{i+1} - P_i)$, where ν is the non-convective coupling coefficient. We then compute the total vertical velocity as $D_i = C_i + U_i$.

The vertical velocity field D_i is used to vertically transport T_i , H_i , and W_i upwards or downwards between layers towards a restoration of the MSE and pressure equilibria. To do so we start by computing the quantity to be exchanged between layers i and $i+1$:

$$\Delta K_i = |K_{i+1} - K_i| \cdot \left(1 - \exp\left(-\frac{|D_i| \cdot \Delta t}{\Delta \Theta_i}\right)\right) \tag{6}$$

where K represents either T_i , H_i , or W_i , and $\Delta\Theta_i$ is the altitude difference between layers L_i and L_{i+1}. If D_i is positive, this quantity is transferred from K_i to K_{i+1}, and conversely if it is negative, with the proviso that neither K_i nor K_{i+1} can become negative.

Vertical displacements take place because the air itself moves. In order to account for such physical transport, we modify the corresponding pressure fields as follows:

$$P_i^+ = P_i^- + \mu D_i \tag{7}$$

where μ is the pressure coupling factor.

While this might seem non-physical, this solution is effective in practice. In particular, it is preferred to an advection scheme because of the extreme discretization of our problem: vertical transport occurs between layer pairs, which makes it ill suited to a continuous displacement approach.

Note that these inter-layer exchanges mean that each layer is no longer divergence-free. The risk here is that high vertical velocities may induce extreme pressure that would in turn tip the simulation into instability. While this might occur in theory, we have found that velocities remain bounded in practice.

Boundary conditions: The handling of boundary conditions in simulations is a perennial issue. We adopt the popular mechanism of wrapping the edges of our simulation to define a toroidal topology. In order to emulate a larger domain field changes are fed in at the boundary. For instance, a prevailing wind can be created by setting the velocities along one of the boundary. Similarly, introducing patterns of water vapour, condensation and temperature along an edge can enable off scene clouds to be advected into the scene. To help hide the boundary wrapping from a viewer, we often extract clouds from a subset of the simulation domain.

5. Terrain Encoding

In this section we consider the modifications to our layered simulation necessary to incorporate terrain-atmosphere interaction, including temperature and moisture transfers and the redirection of wind through orographic lift, expressed as a second inter-layer exchange term.

Orographic effect: Incorporating non-planar terrain topography first requires a choice in air-layer representation: should layer shape conform to the terrain or remain flat and fixed at a constant altitude? The first option could seem simpler as it obviates the need to process the intersection of atmosphere and terrain, but ends up significantly complicating advection because of the impact of within-layer variations in altitude on pressure.

Therefore, we choose the second option: we retain an atmospheric structure consisting of planar layers with constant altitude Θ_i , and set up specific boundary conditions and inter-layer interactions where the terrain impinges on a layer. To cope with this we borrow methods for obstacle handling from computational fluid dynamics and adapt them to account for airflow that veers either sideways (barrier jets) or upwards (orographic lift) depending on local topography.

© 2020 The Author(s)
Computer Graphics Forum © 2020 The Eurographics Association and John Wiley & Sons Ltd.

To account for orographic lift we introduce the notion of orographic velocity damping $\sigma_i(p)$ based on how the terrain impinges on an air-layer $\mathcal{L}_i$ at position p. For a terrain with height $\Theta_{\mathcal{T}}(p)$ at p:

$$\sigma_i(p) = 1 - \min\left(1, \max\left(0, \frac{\Theta_{\mathcal{T}}(p) - \Theta_i}{\Delta\Theta_i}\right)\right). \tag{8}$$

This σ factor is used for damping the velocity field around obstacles. As depicted in Figure 4 it acts to deflect incident air upwards, producing non-convective vertical motion as explained in Section 4.2.

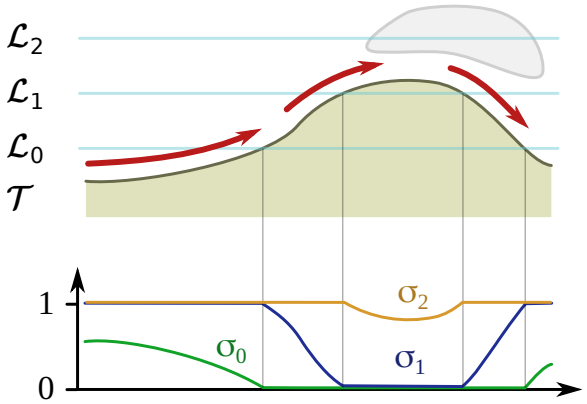

Figure 4: *We use terrain topography $\mathcal{T}$ to compute the orographic velocity damping coefficients σ_i per layer $\mathcal{L}_i$. Damping velocity creates high and low pressure areas that transform into updraft and downdraft velocity components, which accounts for orographic effects, such as the lenticular cloud depicted here.*

Heat balance: To account for heat fluxes in the terrain layer during the diurnal cycle and measure changes in heat and moisture due to interactions with the air, the following quantities are stored in terrain-specific fields: heat capacity $C_p(p)$ in Joules per kilogram per Kelvin, which is the amount of heat needed to raise up by 1K the temperature of one kilogram of a given material, determined by the land cover, be it open water, vegetation, grassland, or exposed ground; temperature $T_{\mathcal{T}}$ in Kelvin, indicating the current stored heat of the terrain cell, and water content $M_{\mathcal{T}}$ in g/m^2. The water content, in particular, is crucial to the cycle of evaporation and condensation.

We now turn to the process of heat absorption and release from the land surface. For terrain irradiation (and consequent heat absorption), the sun's position is tracked with a classical celestial model, and ray-marching used to compute the solar energy received by each parcel of terrain while accounting for self- and cloud- shadowing. Terrain temperature is updated according to the following irradiation equation [BC09]:

$$\Delta T_{\mathcal{T}}(p)^{absorbed} = \frac{F_s \cdot (1 - B(p)) \cdot \cos(\theta) \cdot \Delta t}{D(\theta) \cdot C_p(p)}, \tag{9}$$

where $F_s = 340\ W/m^2$ is the sun constant, B is surface albedo (we use 0.15 for ground and 0.05 for water), θ is the incident angle of the sun, and $D(\theta)$ is the traversed atmospheric thickness from the law of cosines.

For simulating heat release into the atmosphere we employ a standard grey body model:

$$\Delta T_{\mathcal{T}}(p)^{emitted} = \varepsilon \cdot \sigma_B \cdot \Delta t \cdot \frac{T_{\mathcal{T}}(p)^8}{C_p(p)}, \tag{10}$$

with $\varepsilon = 0.9$ being surface emissivity and σ_B the Stefan-Boltzmann constant. Strictly speaking, ε should depend on terrain type, but, except for snow, these values do not vary significantly.

Atmospheric heat absorption: The final stage is to connect the ground with relevant atmospheric layers. We note $\ell(p) = \arg\min_{\Theta_i > \Theta_{\mathcal{T}}(P)} i$ as the lowest air layer that does not intersect the terrain at position p. There are then two categories of transfer:

1. Heat exchange: air temperature rises or falls towards a match with the heat released from the ground. We model this as a standard conduction phenomenon:

$$T^+_{\ell(p)} = T^-_{\ell(p)} + \Delta T_{\mathcal{T}}(p)^{emitted} \cdot \kappa \tag{11}$$

   where κ is the heat transfer coefficient proportional to air heat capacity and absorbance.
2. Water exchange: ground water evaporates into the air and atmospheric moisture condenses on the ground. This is exactly the same process as in intra-layer phase changes (see Section 4.2). When a phase change takes place, a quanta of energy is either released or absorbed affecting a change in the temperature of the ground, causing oceans to cool down due to evaporation. For a given transfer ΔH_i the corresponding change in temperature is:

$$\Delta T_i = \frac{\Delta H \cdot Q^{evap}}{C_p}, \tag{12}$$

   where Q^{evap} is the latent heat of evaporation and C_p is the specific heat capacity of the terrain (water or ground).

6. Procedural Amplification

Skyscapes manifest to an observer primarily through the presence of liquid water suspended as clouds. This is encoded in our simulation as the water content field W_i of individual $2D$ atmospheric layers $\mathcal{L}_i$ at a relatively coarse sampling scale, but such a format is ill-suited to volumetric rendering and requires restructuring as a regular $3D$ grid, along with procedural upsampling to provide resolvable cloud detail.

To begin, a volumetric grid with regular voxel dimensions is derived by linearly interpolating corresponding elements of the water field between adjacent layers. This piecewise linear representation of vertical gradients is not uncommon in meteorology , where soundings are typically taken at different elevations using weather balloons [SKD*08]. However, it is important to capture local minima and maxima, which is why the number of layers and their specific altitudes are configurable during setup. In particular, in order to separate clouds into distinguishable bands it is necessary to add sandwiching layers of clear air, corresponding to inversions in the temperature profile. Since the user sets the initial temperature and altitude of these layers it is possible for them to exert some control over the range of possible outcomes.

© 2020 The Author(s)
Computer Graphics Forum © 2020 The Eurographics Association and John Wiley & Sons Ltd.

The output, per time step, is a resampled $3D$ cloud density volume $\mathcal{W}$ with a typical resolution of $200 \times 200 \times 200$m per cell. We next apply a vertical upsampling operator that preserves the natural appearance of clouds. Clouds usually lie around an isobar (which we approximate as an iso-altitude). They form above, below or around the isobar, and some clouds exhibit flat bottoms or tops. We vertically displace and scale a cloud cell depending on the cloud taxonomy and the associated clear air layer, so that it respects these geometric observations and the volume constraint given by the previously computed density. In particular, stratiform clouds are centered at the altitude of the simulated layer (used as the isobar in our model); flat-bottomed cumuliform clouds are sampled above the layer, and high altitude clouds such as cirrostratus, which are naturally flat-topped, are sampled below the layer.

Since this spatial upsampling mechanism would only be sufficient for the most amorphous cloud structures, we also procedurally upsample cloud density using a time-varying $3D$ harmonic noise field $\mathcal{N}$ whose local spectral parameters depend on cloud type identified on a per-cell basis. For instance, fluffy cumuliform clouds require noise of higher amplitude and frequency than more wispy stratiform clouds.

Finally, if the animation is to be rendered at a finer time-step compared to the simulation, for instance to provide an animated background in a film or game, intermediate frames are also computed. Using linear interpolation between densities at consecutive simulated frames would decrease the quality and coherence of cloud motion. Instead, we assume the wind velocity to be constant in time between these frame, and we advect the density for N intermediate frames using a semi-lagrangian scheme and a rescaled time-step $\Delta t/N$.

The method we just described heavily depends on our ability to classify the type of a cloud form the information within a simulated cell. The standard taxonomy of major cloudforms (see Figure 5 and Table 3) is based on distinctions in appearance arising primarily from the altitude at which a cloud appears and turbulence caused by convectivity. This provides a convenient basis for classification because altitude Θ_i and convection C_i are readily available from our simulation. The cloudforms in Table 3 are separable on the basis of altitude and convection alone, except for two cases (*nimbostratus* and *stratocumulus*) where thickness can be used as an additional discriminator. Unlike altitude and convection, thickness is not a local cell-specific property. We derive it by counting the number of vertically connected non-empty cells, which establishes the vertical extent of a column of cloud encompassing the current cell.

One issue is that such hard classification boundaries can degrade spatial and temporal coherence. For example, there might be cells of one type scattered through a mass of a different type, or an advected cell may transition irregularly between types across frames. We solve this by grouping cells using k-means clustering of $5D$ cell vectors (incorporating position, time, convection, and altitude). This is seeded using the k-means++ algorithm [AV07] and the resulting cluster means (along with mean thickness) provide a consistent classification of the entire space-time cluster. Importantly, this is only applied in cases where a first classification pass reveals fragmentation since the process is computationally expensive. For example, there is no point in performing clustering in a scene with no intermixing of cloudforms within layers.

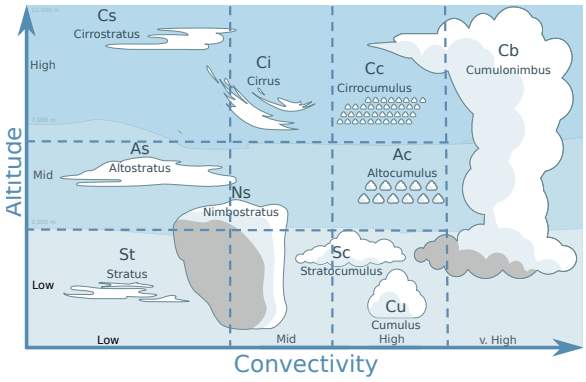

Figure 5: *Our cloud classification scheme. The dashed lines indicate classification boundaries between cloudforms derived from Table 3.*

It is worth noting that a particular advantage of a simulation strategy such as ours lies in its support for weather-consistent winds. By exporting the velocity field $\mathbf{V}_i$, particularly in atmospheric layers overlaying or intersecting the terrain, animated effects such as wind-swept trees, whirling detritus, or rippling water surfaces can be incorporated into natural scenes.

7. Results and discussion

We implemented our method in C++. Experiments were performed on a desktop computer equipped with an Intel® Xeon, clocked at 2.10GHz with 31.3GB of RAM and 32 cores. The output of our system was exported Terragen4® for photorealistic rendering.

7.1. Authoring

Authoring simulation-driven content is known to be a challenge, primarily because controls tend to be indirect, such as setting and tuning initial conditions and simulation parameters. In addition to providing interactive visual feedback on the simulation (see Figure 6), which is essential to make indirect control possible, we assist users by providing three categories of authoring tools:

- **A direct painting interface.** The user is able to pause the simulation and paint directly into the temperature, humidity, water content and velocity fields of any layer, with immediate effect (as is done in Figure 6). We have found that this is less jarring in practice when used on secondary (temperature and water vapour) rather than primary fields (velocity and water content).
- **Scenario initialization.** This allows a user to configure the initial conditions on a per layer basis, including establishing a prevailing wind and seeding layer quantities either with noise or saved state from previous simulations. In order to separate clouds into bands a user must incorporate warmer inversion layers, which are a quite common phenomenon in nature. While

© 2020 The Author(s)
Computer Graphics Forum © 2020 The Eurographics Association and John Wiley & Sons Ltd.

Type	Altitude (m)	Thickness (m)	Convection	Appearance
Stratus	100 – 2000	100 – 1000	low	pervasive and blanketing
Altostratus	2000 – 6000	1000 – 5000	low	sheet-like or wavy and possibly fragmented
Cirrostratus	> 6000	1000 – 5000	low	thin sheet, possibly with undulations
Nimbostratus	100 – 7000	2000 – 4000	low-med	thick, opaque, and featureless
Cirrus	> 6000	1000 – 5000	med	thin, wispy strands
Cumulus	1000 – 2000	400 – 1500	high	individual puffy clouds
Stratocumulus	200 – 2000	200 – 400	med-high	puffy and cohering
Altocumulus	2000 – 6000	500 – 1000	high	a puffy layer
Cirrocumulus	> 6000	500 – 1000	high	sheet with high-frequency structure
Cumulonimbus	> 200	2000 – 12000	very high	towering and lumpy

Table 3: *Typical properties of different cloud forms.*

this does require some domain knowledge, there is, fortunately, a direct correspondence between sandwich layers and cloud separation, and the correct behaviour naturally emerges from the simulation.

- **Timeline events.** A scenario timeline can be constructed with wind and field events triggered along the domain boundary at specific times and for a set duration. For instance, the velocity along a boundary edge can be altered to gradually shift the prevailing wind direction or introduce turbulence. Similarly, simplex noise or saved state can be introduced into selected layer fields in a strip along the boundary and then advected into the core domain. This simulates the impact of the larger world on the simulation domain over time.

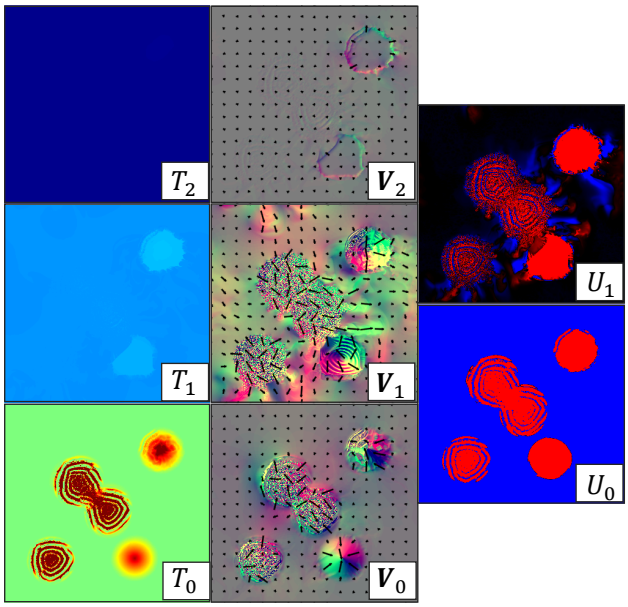

Figure 6: *Convection cells form in a 3-layer simulation after the user paints hot-spots in the lowest temperature field. From left to right: temperature, 2D velocity visualized using both arrows and RGB colouring, and convection with a red (uplift) to blue (downdraft) heatmap. Successive layers appear from top to bottom with the convection field straddling layers.*

7.2. Validation

In this section we present a set of scenarios (summarised in Table 4) that validate our simulation against expected meteorological behaviour, including the formation of convective cells, changes due to diurnal cycles, and the impact of terrain. We conclude with a scenario that showcases the interaction of different cloud types. For all scenarios, we set the timestep to $\Delta t = 30s$, although anything in the range $10s$ to $240s$ works well for high-velocity scenes.

Our first and most straightforward experiment (see Figure 6) confirms that vertically-circulating convection cycles develop between layers ($n = 3$) in the presence of differential temperatures, as would be expected from uneven heat exchanges with the terrain. These are reminiscent of the Bernard cells that form in more constrained circumstances, such as a simmering liquid. Here, the user has directly painted hot-spots into the lowermost air-layer.

Our second scenario builds on the initial convection experiment to demonstrate the formation of cumulus clouds over an archipelago (see Figure 7) induced by heat transferred from exposed ground heated throughout the day. In this scenario air with entrained water vapour is fed landward, warmed by heat conducted from the sun-baked island surface and lifts, cools and condenses to form turbulent cumulus clouds.

We use a diurnal cycle for a shoreline landscape to further test the influence of terrain type on irradiation and heat conduction (see Figure 8). The expected and evidenced behaviour is that a daytime onshore breeze will shift offshore during the night, due to different rates of absorption and release of heat between land and sea. A rendered image of the shoreline is shown in Figure 1.

To demonstrate the influence of topology on cloud formation we include an orographic lift scenario (see Figure 9) in which a moisture-laden wind is driven up a mountain slope to condense at cooler altitudes into stratus clouds. Such terrain interactions are a key contribution of our framework.

Finally, we showcase a combination of weather effects within a single scenario (see Figure 10). This encompasses the simultaneous evolution of different cloudforms within and between layers (l), including cumulus ($l = [1,3]$), altostratus ($l = 3$), cirrus ($l = 5$) and cirrostratus ($l = 7$). At higher altitudes we use inversion layers ($l = 3,5$) to create separating bands of clear air. Layers 3, 5, and 6 are initialized by the author with a prevailing wind direction and

© 2020 The Author(s)
Computer Graphics Forum © 2020 The Eurographics Association and John Wiley & Sons Ltd.

Scene	Characteristics						Compute
	Layers	Altitudes (m)	Grid size	Timestep (s)	Duration (s)	Authoring	per frame (s)
Convection cells (Fig. 6)	3	10, 800, 1600	512^2	30	328	init, painting	2.73
Island convection (Fig. 7)	3	10, 800, 1600	512^2	30	600	init	3
Diurnal cycles (Fig. 8)	3	10, 800, 1600	256^2	30	3692	init	1.3
Orographic lift (Fig. 9)	3	10, 800, 1600	512^2	30	947	init	3.4
Mixed cloudforms (Fig. 10)	7	10, 2000, 4000, 5000, 7000, 8000, 9000	512^2	30	1350	init, timeline	6.75

Table 4: *Main features -number of layers, layer heights, layer resolution, time step increment, overall scene duration, and authoring mechanisms- and performance (average computation time per frame) for our validation scenarios.*

Figure 7: *Cumulus clouds forming above a pair of islands in a 3-layer simulation. An aerial view is shown inset in the bottom right of each image.*

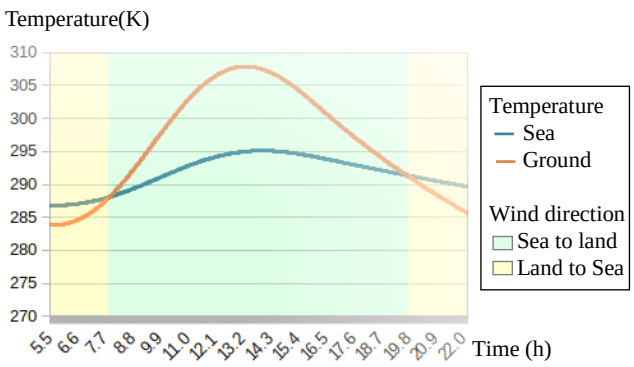

Figure 8: *Diurnal cycles leading to a shift in wind direction at the interface between land and sea from onshore during the day to offshore at night.*

simplex noise in the H_i field. We also employ a timeline event to change the direction of the wind and introduce turbulence in layer 2 after a 30 minute interval.

7.3. Performance

Table 4 reports timings and statistics for the validation scenarios presented in this paper. The simulation performance of our CPU implementation is sufficient for authoring purposes at an average of 4.3s per frame for a 512^2 domain. While it is not possible to make definitive claims, we anticipate that this could be reduced to under a second for an optimized GPU implementation, given the known performance of multigrid methods on such an architecture [And18]. In general, computation cost is roughly linear in the total number number of cells across all layers, as expected.

7.4. Limitations

Our hybrid approach has certain restrictions: some are inherent to a simulation strategy, but others could be solved through extensions to the framework.

Currently, the user is required to pre-set the layer parameters and finding an ideal configuration often requires trial-and-error adjustment. An auto-tuning framework, capable of automatically seeking and setting the optimal number and altitude of layers, would improve authoring. In addition, the planar resolution must be uniform and consistent across all layers. In theory, decoupling layer resolutions is possible, but would require changing the one-to-one mapping of updraft and convection between cells in vertically adjacent layers.

While we provide a variety of authoring tools, as with most simulations our controls cannot achieve the precision of procedural methods without degrading plausibility. For instance, a user can paint a shape directly into the water content field but this would force an unrealistic transition even if blended over several frames.

Currently, our skyscapes end abruptly at the 50×50km margin. In many cases this is not noticeable from a ground-based vantage. Nevertheless, it would be useful to couple our method with a synoptic-scale weather simulation [DYN06]) in order to provide coarser but still realistic boundary conditions for our simulation.

Finally, in this paper we focus on wind and cloud formation.

© 2020 The Author(s)
Computer Graphics Forum © 2020 The Eurographics Association and John Wiley & Sons Ltd.

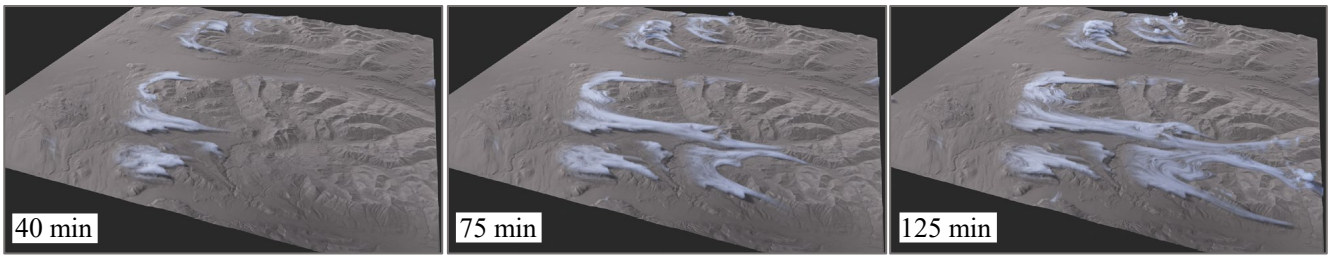

Figure 9: *Orographic lift causes upslope cloud formation, showing one of the effects of terrain in our simulation framework.*

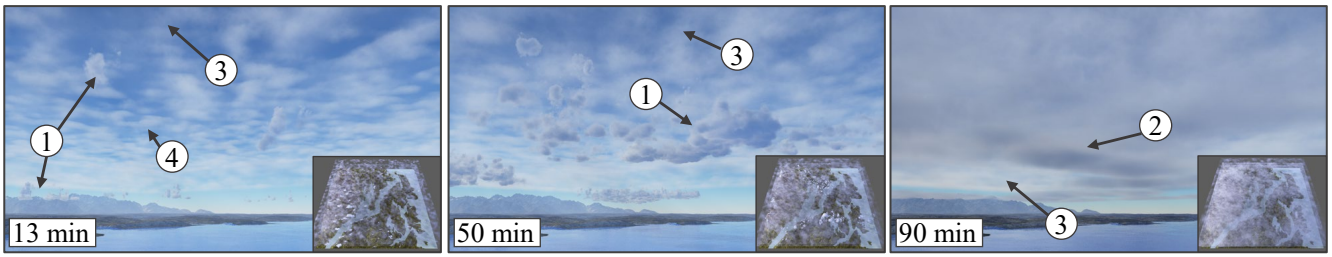

Figure 10: *Multiple cloudforms within a single scene, including cumulus (1), altostratus (2), cirrus (3), and cirrostratus (4) clouds.*

Nevertheless, modeling general skyscapes would require accounting for other phenomena, such as precipitation, which also play a key role in the dynamic behaviour of clouds and have visual significance in their own right, for instance through rain-darkened skies. This would require a significant but achievable extension of our framework.

8. Conclusion

In this work, we have shown that plausible and diverse animated wind and clouds can be generated at interactive rates, and then further enriched with consistent procedural details at the rendering stage. Our layered physically-based model provides a unique way to efficiently simulate the formation and evolution of wind and clouds over time, thanks to its ability to balance sparse vertical detail with a higher level of horizontal detail, enabling capture of the most relevant meteorological effects. This simulation allows the user to interactively launch and tune weather scenarios, before the offline rendering of an animated skyscape.

Our method is the first in Computer Graphics, to the best of our knowledge, to take the presence of water bodies and the topology of the terrain into account. Moreover, our taxonomy for cloud types enable us to compute consistent up-sampling of our simulation data, leading to the visual complexity required by games and feature films.

A first extension to tackle in future work would be the inclusion of more phenomena, starting with rain but also extending to thunder, hail, snow, and other cloud sub-types. This could be done using stochastic, procedural models controlled via probability heatmaps generated through simulation. Enabling the inverse modeling of precise events (e.g., “rain begins at 8 am") would also ease authoring of complex and precise scenarios.

Lastly, our model could be coupled with other natural phenomena frameworks to improve consistency in virtual landscape modelling: run-off and erosion simulation [CGG*17] as well as vegetation simulation [GLCC17] come immediately to mind, since these phenomena are heavily dependant on weather.

Acknowledgements

This work was partly funded by the chair between Google and École Polytechnique.

References

[And18] ANDO R.: Shiokaze Framework, 2018. http://research.nii.ac.jp/~rand/shiokaze-en/. 4, 5, 10

[AV07] ARTHUR D., VASSILVITSKII S.: k-means++: the advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms)* (2007), Society for Industrial and Applied Mathematics, pp. 1027–1035. 8

[Bar92] BARRY R.: *Mountain weather and climate*, 2 ed. Routledge, 1992. 4

[BC09] BARRY R., CHORLEY R.: *Atmosphere, Weather and Climate*, 9 ed. Routledge, 2009. 4, 5, 6, 7

[BN04] BOUTHORS A., NEYRET F.: Modeling Clouds Shape. In *Eurographics (short papers)* (Grenoble, France, Aug. 2004), Galin E., Alexa M., (Eds.), Eurographics Association, pp. –. 2, 3

[BNL06] BOUTHORS A., NEYRET F., LEFEBVRE S.: Real-time realistic illumination and shading of stratiform clouds. In *Eurographics Workshop on Natural Phenomena* (Vienne, Austria, Sept. 2006), Galin E., Chiba N., (Eds.), Eurographics. 2, 3

© 2020 The Author(s)
Computer Graphics Forum © 2020 The Eurographics Association and John Wiley & Sons Ltd.

[BNM*08] BOUTHORS A., NEYRET F., MAX N., BRUNETON E., CRASSIN C.: Interactive multiple anisotropic scattering in clouds. In *Proceedings of the 2008 symposium on Interactive 3D graphics and games* (2008), ACM, pp. 173–182. 2

[CGG*17] CORDONNIER G., GALIN E., GAIN J., BENES B., GUÉRIN E., PEYTAVIE A., CANI M.-P.: Authoring landscapes by combining ecosystem and terrain erosion simulation. *ACM Trans. Graph. 36*, 4 (July 2017), 134:1–134:12. 11

[CMCM11] CHUNG-MIN Y., CHUNG-MING W.: An effective framework for cloud modeling, rendering, and morphing. *Journal of Information Science and Engineering 27*, 3 (May 2011), 891–913. 2

[DEYN07] DOBASHI Y., ENJYO Y., YAMAMOTO T., NISHITA T.: A fast rendering method for clouds illuminated by lightning taking into account multiple scattering. *The Visual Computer 23*, 9-11 (2007), 697–705. 2

[DG17] DUARTE R. P., GOMES A. J.: Real-time simulation of cumulus clouds through skewt/logp diagrams. *Computers & Graphics 67* (2017), 103 – 114. 2, 3

[DKY*00] 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* (New York, NY, USA, 2000), SIGGRAPH '00, ACM Press/Addison-Wesley Publishing Co., pp. 19–28. 3

[DSY10] DOBASHI Y., SHINZO Y., YAMAMOTO T.: Modeling of clouds from a single photograph. *Computer Graphics Forum 29*, 7 (2010), 2083–2090. 2

[DYN06] DOBASHI Y., YAMAMOTO T., NISHITA T.: A controllable method for animation of earth-scale clouds. *Proc. of CASA* (2006), 43–52. 3, 10

[FBDY15] FERREIRA BARBOSA C. W., DOBASHI Y., YAMAMOTO T.: Adaptive cloud simulation using position based fluids. *Computer Animation and Virtual Worlds 26*, 3-4 (2015), 367–375. 2, 3

[Gar85] GARDNER G. Y.: Visual simulation of clouds. In *Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1985), SIGGRAPH '85, ACM, pp. 297–304. 2, 3

[GDAB*17] GARCIA-DORADO I., ALIAGA D. G., BHALACHANDRAN S., SCHMID P., NIYOGI D.: Fast weather simulation for inverse procedural design of 3d urban models. *ACM Trans. Graph. 36*, 4 (Apr. 2017). 3, 5

[GLCC17] GAIN J., LONG H., CORDONNIER G., CANI M.-P.: Ecobrush: Interactive control of visually consistent large-scale ecosystems. *Computer Graphics Forum 36*, 2 (2017), 63–73. 11

[GN17] GOSWAMI P., NEYRET F.: Real-time landscape-size convective clouds simulation and rendering. In *Proceedings of Workshop on Virtual Reality Interaction and Physical Simulation* (2017). 3

[HBSL03] HARRIS M. J., BAXTER W. V., SCHEUERMANN T., LASTRA A.: Simulation of cloud dynamics on graphics hardware. In *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS Conference on Graphics Hardware* (Aire-la-Ville, Switzerland, Switzerland, 2003), HWWS '03, Eurographics Association, pp. 92–101. 2, 3

[KMM*17] KALLWEIT S., MÜLLER T., MCWILLIAMS B., GROSS M., NOVÁK J.: Deep scattering: Rendering atmospheric clouds with radiance-predicting neural networks. *ACM Trans. Graph. 36*, 6 (Nov. 2017), 231:1–231:11. 2

[LJW06] LIU S., JIN X., WANG C. C. L.: Target shape controlled cloud animation. In *Advances in Computer Graphics* (Berlin, Heidelberg, 2006), Nishita T., Peng Q., Seidel H.-P., (Eds.), Springer Berlin Heidelberg, pp. 578–585. 3

[LJWcH07] LIU S., JIN X., WANG C. C., CHUEN HUI K.: Ellipsoidal-blob approximation of 3d models and its applications. *Computers & Graphics 31*, 2 (2007), 243 – 251. 2

[LLC*10] LAGAE A., LEFEBVRE S., COOK R., DEROSE T., DRETTAKIS G., EBERT D., LEWIS J., PERLIN K., ZWICKER M.: A survey of procedural noise functions. *Computer Graphics Forum 29*, 8 (2010), 2579–2600. 2

[MIY02] MIYAZAKI R.: Simulation of cumuliform clouds based on computational fluid dynamics. *EUROGRAPHICS 2002 Short Presentations* (2002), 405–410. 3, 5

[MYND01] MIYAZAKI R., YOSHIDA S., NISHITA T., DOBASHI Y.: A method for modeling clouds based on atmospheric fluid dynamics. In *Proceedings of the 9th Pacific Conference on Computer Graphics and Applications* (Washington, DC, USA, 2001), PG '01, IEEE Computer Society, pp. 363–. 3, 5

[NDN96] 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* (New York, NY, USA, 1996), SIGGRAPH '96, ACM, pp. 379–386. 2

[Ney97] NEYRET F.: Qualitative Simulation of Cloud Formation and Evolution. In *Eurographics Workshop on Computer Animation and Simulation* (Budapest, Hungary, 1997). 2, 3

[Pla19] PLANETSIDE SOFTWARE: Terragen 4. https://planetside.co.uk/, 2019. 2

[PSA*98] PIELKE R. A., SR ., AVISSAR R., RAUPACH M., DOLMAN A. J., ZENG X., DENNING A. S.: Interactions between the atmosphere and terrestrial ecosystems: influence on weather and climate. *Global Change Biology 4*, 5 (1998), 461–475. 4

[RBD*18] RANDALL D. A., BITZ C. M., DANABASOGLU G., DENNING A. S., GENT P. R., GETTELMAN A., GRIFFIES S. M., LYNCH P., MORRISON H., PINCUS R., THUBURN J.: 100 years of earth system model development. *Meteorological Monographs 59* (2018), 12.1–12.66. 5

[SBRS10] STIVER M., BAKER A., RUNIONS A., SAMAVATI F.: Sketch based volumetric clouds. In *Smart Graphics* (Berlin, Heidelberg, 2010), Taylor R., Boulanger P., Krüger A., Olivier P., (Eds.), Springer Berlin Heidelberg, pp. 1–12. 2

[SKD*08] SKAMAROCK W. C., KLEMP J. B., DUDHIA J., GILL D. O., BARKER D., DUDA M., ET AL.: A description of the advanced research (wrf) model, version 3. *Natl. Ctr. Atmos. Res., Boulder, CO* (2008). 7

[SSEH03] SCHPOK J., SIMONS J., EBERT D. S., HANSEN C.: A real-time cloud modeling, rendering, and animation system. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Aire-la-Ville, Switzerland, Switzerland, 2003), SCA '03, Eurographics Association, pp. 160–166. 2, 3

[WBC08] WITHER J., BOUTHORS A., CANI M.-P.: Rapid sketch modeling of clouds. In *Eurographics Workshop on Sketch-Based Interfaces and Modeling (SBIM)* (Annecy, France, June 2008), Alvarado C., Cani M.-P., (Eds.), Eurographics Association, pp. 113–118. 2

[WBDY15] WELTON C., BARBOSA F., DOBASHI Y., YAMAMOTO T.: Adaptive cloud simulation using position based fluids. *Computer Animation and Virtual Worlds 26*, 3-4 (2015), 367–375. 2, 3

[WCGG18] WEBANCK A., CORTIAL Y., GUÉRIN E., GALIN E.: Procedural Cloudscapes. *Computer Graphics Forum 37*, 2 (2018). 2, 3

[WGM14] WEI C., GAIN J., MARAIS P.: Interactive 3d cloud modelling with a brush painting interface. In *Proceedings of the 18th meeting of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (2014), ACM, pp. 160–160. 2

[WH06] WALLACE J. M., HOBBS P. V.: *Atmospheric science: an introductory survey*, vol. 92. Elsevier, 2006. 1

[YLH*14] YUAN C., LIANG X., HAO S., QI Y., ZHAO Q.: Modelling cumulus cloud shape from a single image. *Computer Graphics Forum 33*, 6 (2014), 288–297. 2, 3

© 2020 The Author(s)
Computer Graphics Forum © 2020 The Eurographics Association and John Wiley & Sons Ltd.