Exploiting Temporal Coherence in Forest Dynamics Simulation*

Pankaj K. Agarwal
Department of Computer Science
Duke University
pankaj@cs.duke.edu

Thomas Mølhave
Department of Computer Science
Duke University
thomasm@cs.duke.edu

Hai Yu
Google Inc.
New York, NY
fishhai@google.com

James S. Clark
Nicholas School of the Environment
Duke University
jimclark@duke.edu

ABSTRACT

Understanding the impact of climate and land-use on forest ecosystems involves modeling and simulating complex spatial interactions at many different scales. With this goal in mind, we have developed an individual-based, spatially explicit forest simulator, which incorporates fine-scale processes that influence forest dynamics. In this paper we present new, faster algorithms for computing understory light and for dispersal of seeds — the two most computationally intensive submodules in our simulator. By exploiting temporal coherence, we circumvent the problem of doing the entire simulation at each step. We provide experimental results that support the efficiency and efficacy of our approach.

Categories and Subject Descriptors

F2.2 [**Nonnumerical Algorithms and Problems**]: Geometrical problems and computations

General Terms

Algorithms, Experimentation

Keywords

Forest models, Simulator, Ecological forecasting, Graphics hardware, Quad-tree, Monopole approximation, Spatiotemporal

*This work is supported by NSF under grants CNS-05-40347, IIS-07-13498, CCF-09-40671, CCF-10-12254, and DDDAS 0540347, by ARO grants W911NF-07-1-0376 and W911NF-08-1-0452, by an NIH grant 1P50-GM-08183-01, by a grant from the U.S.–Israel Binational Science Foundation, by the Coweeta LTER grant, and by the US Forest Service.

SCG'11, June 13–15, 2011, Paris, France.

1. INTRODUCTION

Understanding how forest ecosystems respond to climate change and how this impacts biodiversity is central to global change research. Because these phenomena are so complex, computer based simulation is the only feasible approach to understand and predict them. Forest responses to climate change involve complex spatial interactions at many different scales, ranging from individual-tree scale to population scale [2, 12, 13, 14, 20]. At the individual-tree scale, growth and survival of an individual depends on interactions between its local environment and climate, as it competes for light and other resources with neighboring individuals. At the population scale, responses to climate change are determined by dispersal of seed to new regions as local climates become unsuitable for continued survival of the species. Operating at vastly different spatial scales—meters for light competition and kilometers for population migration—both processes are slow and interdependent. This necessitates simulating forest dynamics on a large area at a high resolution for a long period.

We have developed a *scalable landscape inference and prediction* (SLIP) model at the scales at which individual trees interact [7, 11, 16]. In SLIP model, trees grow, reach reproductive maturity, reproduce if mature, and survive; seeds are dispersed, and germination of new individuals can occur. A hierarchical Bayes model, SLIP includes known relationships among demographic rates, with parameters and unknown sources of variation that need inference. Using this model, we have also built a simulator that constructs predictive distributions of forest dynamics. Over forty species compete for light availability and respond to changes in climate. Estimates for individual growth, survival, maturation, fecundity, and dispersal, provide a full characterization of individual health and a basis for simulation. A straightforward implementation of this model is too slow to allow for the evaluation of meter to kilometer scale interactions over decades of forest change.

There are two computationally intense modules in SLIP: (i) The *seed-dispersal* module estimates the density of seeds at every location of the forest. For simplicity, a uniform grid is laid over the forest, and the (expected) seed density is computed in each grid cell by computing the intensity of seeds being dispersed from each tree to that cell. If the forest is composed of A grid cells and has n "mature"

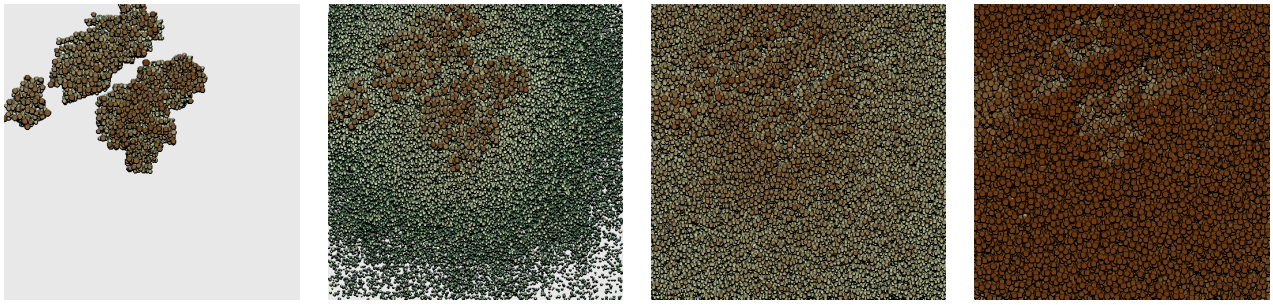


Figure 1. Canopy maps of the *Blackwood* site in Duke forest at four years of simulation on a 512m^2 site. The tallest trees (brown colors) are taller than 40 meters and the shortest trees included in the maps are 2 meters high (light green colors).

trees, the running time of the seed-dispersal algorithm is $O(nA)$; see below for further details. (ii) The *understory light* module estimates the total light reaching ground level, after being attenuated by tree canopies, in each cell. Although the growth of an adult tree is determined by the area of its canopy exposed to the sky, understory light affects the growth of a tree in its early stages. Computation of understory light involves solving several instances of a generalized hidden-surface-removal problem, and is thus quite expensive.

In [16] we proposed a quad-tree based approximation algorithm for computing seed density at each grid cell. It computes in $O((n + A/\mu^2) \log A)$ time a μ-approximation of seed density. As shown in [17], this approximation is reasonable because even for moderate values of μ (e.g. $\mu \leq 0.5$), the error is less than uncertainty/noise in the data and the process models. We also presented a GPU based algorithm for computing the understory light [16], which is about two orders of magnitude faster than a CPU-based algorithm. In our applications, we choose annual time step and run long-term simulations, ranging up to a few hundred years. In [16], the simulator performs the entire computation at each time step, making it too slow to run for a few hundred years on landscapes of area larger than a few sq. km. This raises the question whether the simulation can be expedited by exploiting *temporal coherence* and performing the computation locally at each step, only in those portions of the forest where the dynamics has changed significantly since the last year. For example, understory light in two successive years does not change in a "saturated" forest with no gap areas [10]. Similarly, if we ignore the effect of climate change in successive years, which happens at the regional scale and can be handled separately, the (expected) seed density in a cell changes significantly from the previous year only if a tree falls or becomes "reproductive" (begins producing seeds) in its neighborhood. In this paper, we describe dynamic algorithms for seed dispersal and understory light computation that exploit temporal coherence and update the information quickly. The paper contains two main results.

First, we describe a new algorithm, called the *source based algorithm*, for computing the (expected) seed density in each grid cell. Its running time is $O((n/\mu^2) \log A + A)$. We improve the running time further to $O((n/\mu^2) \log(A/n) + A)$. To ensure the desired accuracy, the size of grid cells is chosen to be sufficiently small; the value of $\log(A/\mu)$ varies in the range 2-6 in our experiments even for a dense forest, so these algorithms are faster in practice than the one in [16]. Even more importantly, unlike the previous algorithm, these algorithms are amenable to dynamic updates. More precisely, they approximate the (expected) seed density over the entire forest as a piecewise-linear function, which is represented as a hierarchy of piecewise-linear functions. The (expected) seed density in a grid cell can then be computed in $O(\log A)$ time. The hierarchical representation of the function can be updated in $O(\log(A)/\mu^2)$ whenever a tree dies or a new tree becomes reproductive.

Next, we define a *gap model* for understory light in a "saturated" forest. It exploits the facts that understory light is low and does not vary too much in non-gap areas in a saturated forest and has little impact on the growth of individuals [4, 5, 10], and that only a small portion of a saturated forest has gaps. We present efficient algorithms for keeping track of gaps in the forest and computing understory light inside the gap and the neighboring cells. The value of light in the non-gap areas of the forest is set using a pre-determined probabilistic distribution. The light-computation algorithm presented here uses the CUDA software library [22]. and is faster than the algorithm in [16].

The paper is organized as follows: Section 2 gives a brief overview of the model; Section 3 describes the seed-dispersal module and the algorithm for computing it; Section 4 describes the gap model, the algorithm for maintaining gaps, and the GPU algorithm for computing understory light. Section 5 discusses our experimental results.

2. MODEL OVERVIEW

The forest model consists of a landscape $\mathbb{L}$ and a population of trees. A planar region, $\mathbb{L}$ is discretized by bounding it with a square and overlaying a uniform grid $\mathbb{M}$. We assume $\mathbb{L} = [0, 2^L] \times [0, 2^L]$, for some positive integer L. $\mathbb{M}$ is a $2^L \times 2^L$ uniform grid, with each grid cell being $1m \times 1m$ large. The model uses a hybrid representation for trees — earlier stages (e.g. seeds and seedlings) are modeled as *densities* with no physical locations and attributes, and more advanced stages (e.g. saplings and adults) are modeled as *individuals* with physical locations and various physical and demographic attributes (e.g. diameter, height, crown area, fecundity). We use $\mathbb{T}$ to denote the set of individuals in the forest. For an individual $i \in \mathbb{T}$, let $\lambda(i) \in \mathbb{M}$ denote the cell that contains i. For a cell $j \in \mathbb{M}$, let $\mathbb{T}_j = \{i \in \mathbb{T} \mid \lambda(i) = j\}$

denote the set of individuals in cell j. Set $n = |\mathbb{T}|$ to be the number of trees in the forest and $A = 4^L$ to be the number of cells in $\mathbb{M}$. Although the model allows trees of different species, we assume for simplicity of the notation in this paper that all trees belong to the same species.

The landscape remains fixed over time, but the population changes over time. Figure 1 shows the result of simulating a site from the Duke forest forward in time by 100 years. Forest dynamics involve three main processes — reproduction, growth, mortality. New individuals enter the population via seed production and dispersal of seeds (see dispersal module). Based on a stochastic process, seeds die, remain dormant, or germinate. The germinated seeds start growing. The model assumes individuals are immature when small, growth occurs each year, and with increasing size, individuals make transition to maturity, after which they begin to reproduce. The growth of an individual depends on light, soil moisture, and other parameters. The growth of an adult tree depends on the area of its canopy exposed to the sky, called *exposed canopy area* (ECA), but the growth of a tree in its early stages depends on the light reaching the ground at its location, called *understory light*. The annual growth accumulates with an associated risk of death, with certain probability. We describe the seed-dispersal and light models in the next two sections. We refer the reader to [7] for details on other parts of our model, which are not modified in this paper.

3. SEED DISPERSAL

In this section we first describe the seed-dispersal and tree-fecundity models, then describe an efficient algorithm for computing seed dispersal over $\mathbb{L}$ by exploiting spatial coherence, and finally describe a dynamic algorithm that quickly updates seed dispersal at each time step using temporal coherence.

Dispersal and fecundity models. The seed-dispersal model estimates the number of seeds, produced by mature trees, that reach each cell of $\mathbb{M}$. Let $q_{j,t}$ and $s_{j,t}$ denote the expected seed density and the number of seeds dispersed, respectively, in cell $j \in \mathbb{M}$ at time t. The latter is computed from the former using a Poisson distribution

$$s_{j,t} = \text{Poisson}(q_{j,t}).$$

The expected seed density, $q_{j,t}$, is the total intensity of seeds that arrive at cell j from all (mature) individuals in $\mathbb{L}$, and it depends on two factors:

DISPERSAL KERNEL: spatial distribution of seeds produced by individual i, i.e., the probability of a seed being dispersed at a particular distance; denoted by K. Based on empirical analysis[9], it is chosen to be a 2-dimensional Student's t-distribution:

$$\mathsf{K}(r) = 1 \Big/ \pi u \left[1 + \frac{r^2}{u}\right]^2. \tag{1}$$

Here K is isotropic, but it may depend on the slope of $\mathbb{L}$ in mountainous regions.

FECUNDITY: volume of seeds produced by an individual i at time t, denoted by $f_{i,t}$. Based on empirical analysis [7], the functional form for $f_{i,t}$ is chosen to be

$$\begin{aligned}\ln f_{i,t} =& \beta_0 + \beta_1 \cdot \ln D_{i,t-1} + \beta_2 \cdot \ln^2 D_{i,t-1} + \\ & \beta_3 \cdot \ln \lambda_{i,t-1} + \beta_4 \cdot \ln d_{i,t-1} + \kappa_t + b_i + \varepsilon_{i,t},\end{aligned} \tag{2}$$

where $\beta_0, \ldots, \beta_4$ are constants that depend on the species of i; $D_{i,t-1}$ is the *diameter* of i at time $t-1$; $\lambda_{i,t-1}$ is the *exposed canopy area* (ECA) of i at time $t-1$, i.e., the portion of the canopy of i that is visible from $z = +\infty$; $d_{i,t-1} = D_{i,t-1} - D_{i,t-2}$ is the *growth* (diameter increment) of individual i at time $t-1$; κ_t, called *year effect* and drawn for each year from a probability distribution, models the inter-annual variation on a regional scale (e.g., climate variation); b_i, called *random individual effect* and drawn for each individual independently from a probability distribution (but does not change over time), models individual variation; $\varepsilon_{i,t}$, called *process error* and modeled as a normal distribution, depends on individual and changes every year. The seed density in cell j is

$$q_{j,t} = \sum_{\ell \in \mathbb{M}} \sum_{i \in \mathbb{T}_\ell} f_{i,t}\, \mathsf{K}(\|\ell - j\|). \tag{3}$$

Here $\|\ell - j\|$ is the distance between the centers of the cells ℓ and j. For simplicity of presentation, we use L_∞-metric in this paper, but the algorithm is implemented using Euclidean distance.

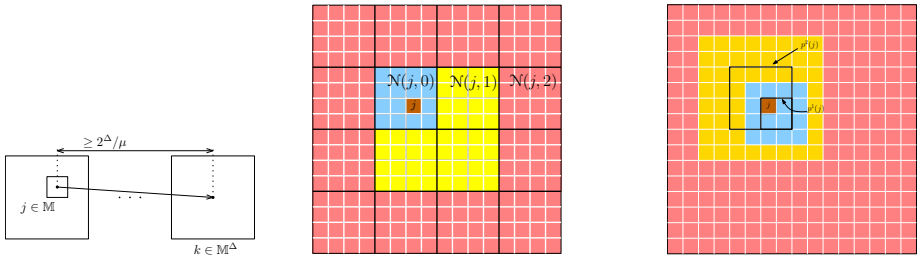
A straightforward implementation of computing $q_{j,t}$ for all cells in $\mathbb{M}$ takes $\Omega(nA)$ time at each time step t. This is computationally expensive and makes the simulations quite slow even on moderate size landscapes (e.g. 256m $\times$ 256m). In the next subsection we describe a faster algorithm that expedites the computation at a slight loss in accuracy.

Computing seed density. The improved algorithm relies on the observation that the contribution of an individual i far away from cell j to the quantity $q_{j,t}$ remains almost the same if $\lambda(i)$, the location of the cell containing i, varies a little. Similar ideas have been used in molecular dynamics, computational physics, approximate nearest-neighbor searching, and other geometric problems [1, 24, 18, 19], but new ideas are needed for our application.

We build a *quad-tree* $\mathcal{T}$ on $\mathbb{M}$. Recall that $\mathbb{M}$ is a $2^L \times 2^L$ grid with each cell size being $1m \times 1m$. The quad-tree has L levels, with $\mathbb{M}$ being level 0. For $0 \le i \le L$, let $\mathbb{M}^i$ denote the $2^{L-i} \times 2^{L-i}$ grid induced by the ith level of $\mathcal{T}$; each grid cell in $\mathbb{M}^i$ has size $2^i \times 2^i$. For a cell $j \in \mathbb{M}^i$, let $p(j)$ be the *parent* cell of j in $\mathbb{M}^{i+1}$, i.e., the cell of $\mathbb{M}^{i+1}$ that contains j. For $\Delta \ge i$, let $p^\Delta(j)$ denote the *ancestor* cell of j in $\mathbb{M}^\Delta$; j is called a *descendant* cell of $p^\Delta(j)$. For a cell $j \in \mathbb{M}^i$, let $\mathbb{T}_{j,t}$ denote the set of individuals in cell j at time t. For simplicity, if the time step t is not important or obvious from the context, we will drop the subscript t. We introduce a parameter μ, called *monopole coefficient*. Let k be a cell in $\mathbb{M}^\Delta$ and j a cell in $\mathbb{M}$. We say that k satisfies the *monopole condition* with respect to j if $\Delta = 0$ or $\|j - k\| \ge 2^\Delta/\mu$, where $\|j - k\|$ is the distance between the centers of cells j and k (in the L_∞-metric); see Figure 2 (a). If k satisfies the monopole condition, we approximate the distance between j and any point in k with $\|j - k\|$ which ranges between $(2^\Delta/\mu)(1 \pm \mu/2)$. We use this approximation in two ways:

(A1) We assume that the density of seed in cell $k \in \mathbb{M}^\Delta$ arriving from an individual i in cell j is uniform; for any descendant cell $\ell \in \mathbb{M}$ of k, the density is given by $f_i\, \mathsf{K}(\|j - k\|)$ (instead of $f_i\, \mathsf{K}(\|j - \ell\|)$). Using a Taylor series expansion of (1) it can be seen that this introduces a relative error of at most 2μ in the seed density computation.

(A2) To compute the contribution of all individuals in $\mathbb{T}_k$



(a) (b) (c)

Figure 2. (a) Monopole condition. (b) A cell j and its neighborhood at different levels: cells in $\mathcal{N}(j,2)$ (pink), $\mathcal{N}(j,1)$ (yellow), $\mathcal{N}(j,0)$ (blue); (c) $\mathcal{N}^{-1}(p^2(j))$ (pink), $\mathcal{N}^{-1}(p^1(j))$ (yellow), $\mathcal{N}^{-1}(p^0(j))$ (blue).

to the seed density q_j in grid cell j, we cluster all individuals in k as a single *super individual*, located at the center of k. The fecundity of this super individual, denoted φ_k, is $\varphi_k = \sum_{i\in\mathbb{T}_k} f_i$. The contribution of $\mathbb{T}_k$ to q_j is approximated by $\varphi_k\,\mathsf{K}(\|j-k\|)$ (instead of $\sum_{i\in\mathbb{T}_k} f_i\,\mathsf{K}(\|\lambda(i)-j\|)$; this also introduces a relative error of at most 2μ in the seed density computation.

For a cell $j \in \mathbb{M}$ and for $0 \leq \Delta < L$, let $\mathcal{N}(j,\Delta)$, called Δ-*neighbors*, be the set of cells $k \in \mathbb{M}^\Delta$ that satisfy the monopole condition but their parent cells do not satisfy the condition. That is, for $\Delta = 0$, $\mathcal{N}(j,\Delta)$ is the set of cells k for which $p(k)$ is within distance $2/\mu$ from j; for $\Delta > 0$,

$$\|k-j\| \geq 2^\Delta/\mu \quad \text{and} \quad \|p(k)-j\| < 2^{\Delta+1}/\mu.$$

See Figure 2 (b). The distance between j and all cells in $\mathcal{N}(j,\Delta)$ is at most $2^\Delta(1+2/\mu)$. Since the size of each cell in $\mathcal{N}(j,\Delta)$ is Δ, $|\mathcal{N}(j,\Delta)| = O(1/\mu^2)$. By construction, for each cell $\ell \in \mathbb{M}$, there is exactly one value of Δ such that $p^\Delta(\ell) \in \mathcal{N}(j,\Delta)$.

For a cell $k \in \mathbb{M}^\Delta$, let $\mathcal{N}^{-1}(k) = \{\ell \in \mathbb{M} \mid k \in \mathcal{N}(j,\Delta)\}$; see Figure 2 (c) and let

$$\eta(k) = \sum_{\ell\in\mathcal{N}^{-1}(k)} \sum_{i\in\mathbb{T}_\ell} f_i\,\mathsf{K}(\|\ell-k\|). \tag{4}$$

Since $\|k-\ell\| \geq 2^\Delta/\mu$ for all $\ell \in \mathcal{N}^{-1}k$, by (A1), we can assume that the seed density in k is uniform for any individual lying in a cell of $\mathcal{N}^{-1}(k)$. We can therefore rewrite (3) as

$$\begin{aligned} q_j &= \sum_{\Delta\geq 0} \sum_{\ell\in\mathcal{N}^{-1}(p^\Delta(j))} \sum_{i\in\mathbb{T}_\ell} f_i\,\mathsf{K}(\|\ell-p^\Delta(j)\|) \qquad (5)\\ &= \sum_{\Delta\geq 0} \eta(p^\Delta(j)). \end{aligned}$$

Intuitively, we cluster the cells of $\mathbb{M}$ so that $p^\Delta(j)$ is the highest ancestor of j that satisfies the monopole condition for all cells within each cluster. This immediately gives an algorithm, summarized in Figure 3, for computing the seed density of each cell: We first compute $\eta(\xi,\Delta)$ for all Δ and for all $\xi \in \mathbb{M}^\Delta$, and then compute q_j for all $j \in \mathbb{M}$ in a top-down manner. In order to implement the second step efficiently, we extend the notion of q_j to cells at higher levels of quad-tree: for a cell $j \in \mathbb{M}^\Delta$, define $q_j = 0$ if $\Delta = L$ and $q_j = q_{p(j)} + \eta(j)$ if $\Delta < L$. We compute the quantity q_j in decreasing order of j. Since $|\mathcal{N}(j,\Delta)| = O(1/\mu^2)$, the running time of the above algorithm is $O((n/\mu^2)\log A + A)$.

```
η(j) = 0   ∀Δ ≤ L ∀j ∈ M^Δ
for i ∈ T
  for Δ = L downto 0
    for ξ ∈ N(λ(i), Δ)
      η(ξ) = η(ξ) + f_i K(‖λ(i) − ξ‖)
forΔ = L downto 0
  for j ∈ M^Δ
    q_j = q_{p(j)} + η(j)
```

Figure 3. The first algorithm for computing seed density; $\lambda(i)$ is the cell of $\mathbb{M}$ containing individual i.

We next improve the running time further using (A2) and clustering individuals into super individuals. We now extend the definition of $\mathcal{N}(\cdot)$ to the cells at higher levels of $\mathcal{T}$. More precisely, for $\Delta > 0$ and for a cell $j \in \mathbb{M}^\Delta$, we now define

$$\begin{aligned} \mathcal{N}(j) = \{k \in \mathbb{M}^\Delta \mid &\|j-k\| \geq 2^{\Delta+1}/\mu \\ &\wedge \|p(k)-p(j)\| < 2^{\Delta+2}/\mu\}. \end{aligned}$$

For $j \in \mathbb{M}$ (i.e., $\Delta = 0$), $\mathcal{N}(j)$ is the set of cells $k \in \mathbb{M}\setminus\{j\}$ that are not covered any $\mathcal{N}(p^\Delta(j))$ for $\Delta > 0$, i.e., $\|p(j) - p(k)\| < 4/\mu$. Note that, the cells of $\mathcal{N}(j)$ now lie at the same level as j, and that if $a, b \in \mathbb{M}$ are descendants of j and of a cell in $\mathcal{N}(j)$, respectively, then $\|a-b\| \geq 2^\Delta/\mu$. We define the set $\mathcal{N}^{-1}(j)$ as earlier. For a cell $\xi \in \mathbb{M}^\Delta$, we set

$$\varphi(\xi) = \sum_{i\in\mathbb{T}_\xi} f_i, \qquad \eta(\xi) = \sum_{\ell\in\mathcal{N}^{-1}(\xi)} \varphi(\ell)\,\mathsf{K}(\|\ell-\xi\|). \tag{6}$$

(5) can now be rewritten as

$$\begin{aligned} q_j &= \sum_{\Delta\geq 0} \sum_{\ell\in\mathcal{N}^{-1}(p^\Delta(j))} \sum_{i\in\mathbb{T}_\ell} f_i\,\mathsf{K}(\|\ell-p^\Delta(j)\|)\\ &= \sum_{\Delta\geq 0} \sum_{\ell\in\mathcal{N}^{-1}(p^\Delta(j))} \varphi(\ell)\,\mathsf{K}(\|\ell-p^\Delta(j)\|) = \sum_{\Delta\geq 0} \eta(p^\Delta(j)). \end{aligned}$$

Figure 4 describes the improved algorithm. For a cell ξ, if the algorithm executes the step ($\star$) then one of the leaves in the sub-tree rooted at ξ contains at least one individual. Using these facts, we can now conclude that the running time of the algorithm is $O((n/\mu^2)\log(A/n) + A)$. The value of A/n ranges from 4 to 32 in our experiments.[1]

[1]The worst case running time occurs when the trees are sparse and uniformly distributed. In the scenarios we consider, the forest is either dense or the trees are clustered, so

```
η(j) = 0   ∀Δ ≤ L ∀j ∈ M^Δ
for Δ = 1 to L
  for ξ ∈ M^Δ
   φ(ξ) = Σ_{k=1}^{4} φ(ξ_i)
   if φ(ξ) > 0
    (⋆)for j ∈ N(ξ, Δ)
     η(j) = η(j) + φ(ξ) K(‖j − ξ‖)
for Δ = L downto 0
  for j ∈ M^Δ
   q_j = q_{p(j)} + η(j)
```

Figure 4. Improved algorithm for computing seed density; $\xi_1, \ldots, \xi_4$ are the children of ξ.

Exploiting temporal coherence. We can run the above seed dispersal algorithm at each time step to compute the seed density of each cell in $\mathbb{M}$. Unless new trees become mature or some tree dies in a cell j, the total fecundity of individuals in j does not change much, except possibly due to the year-effect term in (2). Since this term affects all individuals uniformly for a specific year, it can be handled separately. We now describe a data structure for maintaining the expected seed density at time t, $q_{j,t}$, approximately that updates the information locally, instead of re-computing it using the above algorithm at each time step. More precisely, we approximate $q_{j,t}$ by a linear function, which is represented hierarchically and updated periodically as needed. For simplicity, we first describe the data structure without the process-error term $\varepsilon_{j,t}$ in (2), and then describe how this error term is incorporated. Let $g_{i,t}$ denote the value of $f_{i,t}$ without the three error terms, i.e.,

$$\begin{aligned}\ln g_{i,t} =& \beta_0 + \beta_1 \cdot \ln D_{i,t-1} + \beta_2 \cdot \ln^2 D_{i,t-1} + \\ & \beta_3 \cdot \ln \lambda_{i,t-1} + \beta_4 \cdot \ln d_{i,t-1}.\end{aligned} \tag{7}$$

Including the individual and year effect terms (but still ignoring the process-error term), we obtain

$$f_{i,t} = g_{i,t} \exp(b_i + \kappa_t).$$

We approximate $f_{i,t}$ by a linear function using Taylor series with respect to a fixed time $t = t_0$:

$$f_{i,t} \approx (g_{i,t_0} + (t - t_0) g'_{i,t_0}) \exp(b_i) \exp(\kappa_t),$$

where $g'_{i,t_0} = \left.\frac{\mathrm{d}g_{i,t}}{\mathrm{d}t}\right|_{t=t_0}$. We estimate the function g'_{i,t_0} by approximating $D_{i,t-1}, \lambda_{i,t-1}$ as linear functions of t and replacing differentials with finite differences, i.e., setting

$$\begin{aligned}\frac{\mathrm{d}D_{i,t-1}}{\mathrm{d}t} &\approx D_{i,t-1} - D_{i,t-2} = d_{i,t-1}, \\ \frac{\mathrm{d}d_{i,t-1}}{\mathrm{d}t} &\approx 0, \\ \frac{\mathrm{d}\lambda_{i,t-1}}{\mathrm{d}t} &\approx \lambda_{i,t-1} - \lambda_{i,t-2}.\end{aligned}$$

We thus obtain

$$\frac{\mathrm{d}g_{i,t}}{\mathrm{d}t} \approx g_{i,t} \left(\frac{\beta_1 + 2\beta_2 \ln D_{i,t-1}}{D_{i,t-1}} d_{i,t-1} + \frac{\beta_3}{\lambda_{i,t-1}} \frac{\mathrm{d}\lambda_{i,t-1}}{\mathrm{d}t} \right).$$

the speedup by the improved algorithm is more significant than suggested by the above expression.

For a cell $\xi \in \mathbb{M}^\Delta$, we now define

$$a_0(\xi) = \sum_{k \in \mathcal{N}^{-1}(\xi)} \sum_{i \in \mathbb{T}_k} g_{i,t_0} \exp(b_i)\, \mathsf{K}(\|k - \xi\|), \tag{8}$$

$$a_1(\xi) = \sum_{k \in \mathcal{N}^{-1}(\xi)} \sum_{i \in \mathbb{T}_k} g'_{i,t_0} \exp(b_i)\, \mathsf{K}(\|k - \xi\|). \tag{9}$$

By plugging these values in (6), we obtain

$$\begin{aligned}\eta_t(\xi) =& \exp(\kappa_t)(a_0(\xi) + (t - t_0) a_1(\xi), \\ q_{j,t} =& \exp(\kappa_t) \sum_{\Delta \geq 0} a_0(p^\Delta(j)) + \\ & (t - t_0) \exp(\kappa_t) \sum_{\Delta \geq 0} a_1(p^\Delta(j))).\end{aligned} \tag{10}$$

As earlier, we define $q_{j,t}$ recursively: $q_{j,t} = q_{p(j),t} + \eta_t(j)$. We maintain the quantities $a_0(\xi)$ and $a_1(\xi)$ at each cell $\xi \in \mathbb{M}^\Delta$. We choose a time step τ. If $t \bmod \tau \equiv 0$ we recompute a_0, a_1 at time step t and set $t_0 = t$. For other values of t, we update them as follows. Suppose an individual i in a cell j becomes mature (i.e., starts producing seeds) at time step t_1. For all $\Delta \geq 0$ and for all cells $k \in \mathcal{N}(p^\Delta(j), \Delta)$, we set

$$\begin{aligned}a_0(k) =& a_0(k) + (g_{i,t_1} - (t_1 - t_0) g'_{i,t_1}) \exp(b_i)\, \mathsf{K}(\|k - p^\Delta(j)\|) \\ a_1(k) =& a_1(k) + g'_{i,t_1} \exp(b_i)\, \mathsf{K}(\|k - p^\Delta(j)\|)\end{aligned}$$

where t_0 is the last time step when a_0, a_1's were recomputed. We update the terms a_0, a_1 for appropriate cells if an individual dies at time step t_1. The total time spent in processing an individual is $O(\log(A)/\mu^2)$. Finally, we note that the process error term $\varepsilon_{i,t}$ for an individual i is drawn from a normal distribution, therefore each term in (8), (9) is multiplied by $\exp(\varepsilon_{i,t})$, which is a log-normal distribution. Since the weighted sum of a log-normal distribution can be approximated as a log-normal distribution (see e.g. [26]), we modify the two terms in (10) appropriately.

4. UNDERSTORY LIGHT

In this section we describe an improved model for computing understory light, the annual average sunlight from all directions that reaches each cell of $\mathbb{M}$ after being attenuated by tree canopies; roughly speaking each canopy attenuates the incoming light by a factor λ, which is chosen based on field data [10]. As mentioned earlier, understory light influences the growth of a tree in its early stages [8, 16]. We first briefly describe the basic understory light model and the GPU based algorithm for computing it, which was proposed in [15, 16]. We then focus on the improved gap model and the algorithm for maintaining the gap area.

Light model. We model the sky as a hemisphere $\mathbb{H}$. Since $\mathbb{L}$ spans only a few square kilometers, we assume that all grid cells in $\mathbb{L}$ receive the same intensity of light from a fixed direction; for larger landscapes (e.g. covering the entire eastern US), the intensity will depend on the latitude of a grid cell. The average annual sunlight intensity in each direction in $\mathbb{H}$ can be computed [3, 25]. We discretize the solar hemisphere by choosing a set $\Sigma = \{\sigma_1 \ldots, \sigma_u\} \subset \mathbb{H}$ of directions. Let $A_i \subseteq \mathbb{H}$ denote the set of directions for which σ_i is the nearest direction in Σ; σ_i is the representative direction for all directions in A_i. For each σ_i, we compute $E(\sigma)$, the total light energy emanating from the directions in

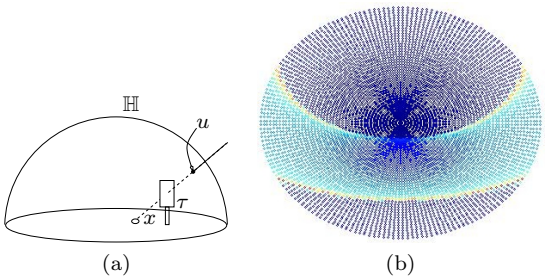


Figure 5. (a) Solar hemisphere, (b) intensity of light in each direction.

A_i;[2] set $E_0 = \sum_{\sigma\in\Sigma} E(\sigma)$. Σ is computed using an adaptive sampling method, so that the discretization error is as little as possible.

For a point $x \in \mathbb{L}$ and a direction $u \in \mathbb{H}$, let $\rho(x,u)$ denote the ray emanating from x in direction u. Let $\mathbb{T}(x,u)$ denote the set of individuals that intersect the ray $\rho(x,u)$. Each individual τ is modeled as an opaque trunk and a translucent cylindrical crown on top of the trunk. The light attenuation of the crown of τ is estimated by a species specific parameter λ_τ, which is determined by calibration of the light model to field estimates of light using canopy photography; see [10]. The fraction of light (intensity) reaching x in direction u is thus

$$\psi(x,u) = \begin{cases} 0 & \text{if } \rho(x,u) \text{ intersects a trunk,} \\ \prod_{\tau\in\mathbb{T}(x,u)} (1-\lambda_\tau) & \text{otherwise.} \end{cases} \tag{11}$$

The normalized light energy reaching a grid cell $j \in \mathbb{M}$ from direction u is now defined as

$$\ell_j^u = E(u)\psi(c_j,u)\cos\varphi, \tag{12}$$

where c_j is the center of the cell j and $\varphi \in [0,\pi/2]$ is the zenith angle of direction u (vertical=$\pi/2$, horizontal=0). The set $\mathcal{L}^u = \langle \ell_j^u \mid j \in \mathbb{M}\rangle$ is the *understory u-light map* of $\mathbb{L}$.

The total light for cell $j \in \mathbb{M}$ is thus:

$$\ell_j = \frac{1}{E_0}\sum_{\sigma\in\Sigma} I_j^\sigma, \tag{13}$$

and $\mathcal{L} = \langle \ell_j \mid j \in \mathbb{M}\rangle$ is the *understory light map* of $\mathbb{L}$.

Computing understory light. Computing the understory light directly using (11) and (13) is expensive, but a simple and efficient algorithm exists using GPUs available on modern PCs, as described in [15]. The algorithm first computes $\mathcal{L}^u$ for each direction $u \in \Sigma$ using the GPU, stores the data in the GPU memory during the computation, and transferrs to the main memory before the next direction is processed. The CPU is then used to compute the final understory light values using (13).

Unfortunately, transferring data between the GPU and main memory is slow and the transfer of the Σ buffers, each with $|\mathbb{M}| = A$ light values, becomes the bottleneck of the algorithm. In this paper we use a modified algorithm that handles the final computation of $\mathcal{L}$ on the GPU itself. This is facilitated by NVIDIAs CUDA software library [22], which simplifies the implementation of certain algorithms on the GPU.

We first initialize an empty *accumulation* buffer $\mathcal{A}$ of the same size as $\mathcal{L}$ in the GPU memory. We then compute the unidirectional light maps one at a time using the previous algorithm [15]. Following the computation of $\mathcal{L}^u$ for $u \in \Sigma$, the values of $\mathcal{A}$ are updated by adding $\mathcal{L}_j^u/E_0$ to each cell $\mathcal{A}_j$ of $\mathcal{A}$. When all the directions have been processed, the value of each cell $\mathcal{A}_j$ is $\mathcal{L}_l$ and $\mathcal{A}$ is now the final understory light map $\mathcal{L}$. Thus, we only have to save the final value of $\mathcal{A}$ in main memory. It follows that we only transfer A cells from the GPU to main memory, instead of the $|\Sigma|A$ cells transferred by the previous algorithm. Furthermore the value of $\mathcal{A}$ is updated in parallel on the GPU, removing the need for the sequential cell-by-cell evaluation of Equation (13) on the CPU.

The gap model. The light maps computed by our algorithm for saturated forests as well as experiments indicate that understory light values are relatively low and do not vary too much in *non-gap* areas of the forest [4, 5, 10, 15]. Furthermore it is not predicted well by our simplified local canopy architecture in non-gap areas because branching patterns adapt to the local light environment to fill space. Only a small portion of the forest has gaps, created by the demise of large individuals, where the light value is high and varies significantly. Since the density of the forest does not change each year except at a few locations and the growth is minimal in low-light areas, there is no need to compute understory light for the entire landscape at each time step. We propose the following *gap model* for computing understory light. First we need some notation. For a cell $j \in \mathbb{M}$ and a positive integer $r > 0$, let $\mathbb{B}_r(j)$ denote the $r \times r$ neighborhood around j, i.e., a square composed of $r \times r$ grid cells centered at j. For a set X of cells, let $\mathbb{B}_r(\mathrm{X}) = \bigcup_{j\in\mathrm{X}} \mathbb{B}_r(j)$.

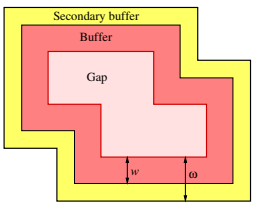
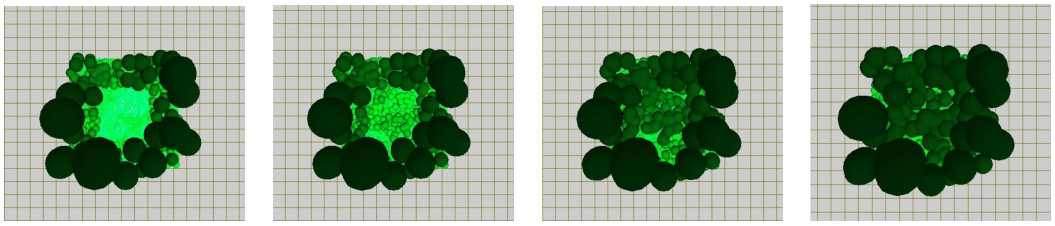


Figure 6. Structure of a gap.

We call a grid cell $j \in \mathbb{M}$ *exposed* if the height of all individuals in $\mathbb{B}_k(j)$ is at most h_0, where k and h_0 are empirically chosen parameters (e.g., we have chosen $k = 5$ and $h_0 = 2m$ in most of our experiments). The set of exposed cells is called the *gap area*. See Figure 6. Let $\mathcal{E}_t$ denote the set of exposed cells at time t. Cells in the neighborhood of $\mathcal{E}_t$ have comparatively higher light values than the average understory light (see Figure 8), so we include them as well in the gap area. More precisely, we fix a parameter called *buffer width* w and set $\mathcal{G}_t = \bigcup_{j\in\mathcal{E}_t} \mathbb{B}_w(j)$. The value of w depends on the density of the forest and the gap size, and it varies in the range 5–10m [4].

To calculate the light values in the cells of $\mathcal{G}_t$, we consider the individuals in the region surrounding $\mathcal{G}_t$ as well. This additional region is called *secondary buffer* (lightly shaded area in Figure 6). The secondary buffer region is only used for computing light in $\mathcal{G}_t$ and we regard it as non-gap area.

[2]Instead of computing the integral over A_i, we choose a dense grid over $\mathbb{H}$ and sum the energy along the grid points that lie in A_i.



(a) (b) (c) (d)

Figure 7. Snapshots from a gap simulation after x years: (a) $x = 2$, (b) $x = 4$, (c) $x = 6$, (d) $x = 8$ [4].

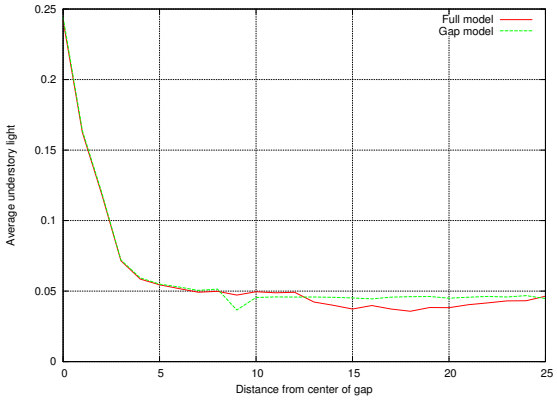


Figure 8. This graph shows the change in the light value vs distance from the gap center for a gap in the forest. It shows both the value computed by the gap (dashed line) and individual-tree (solid line) model.

The effect of the individuals beyond the secondary buffer region on the understory light of the buffer and gap regions is captured by drawing translucent walls around the secondary buffer, which are approximated by vertical rectangles.

Let ω denote the width of the buffer plus secondary buffer regions. We define $\mathcal{B}_t = \mathbb{B}_\omega(\mathcal{E}_t)$, and we set $\mathcal{W}_t = \partial\mathcal{B}_t$. For each cell $j \in \mathcal{G}_t$, its understory light value is the total light that reaches j through the walls erected on $\mathcal{W}_t$ and the set of trees in the cells of $\mathcal{B}_t$, namely, $\bigcup_{l\in\mathcal{B}_t} \mathbb{T}_l$; the light is computed as described in the previous section. For each grid cell in non-gap area, $\mathbb{M}\setminus\mathcal{G}_t$, we draw light values from a normal distribution with mean equal to an average non-gap background light value of a saturated forest. We assign a low variance to account for the low variability in this situation.

The dynamics of the gap model consists of three main parts at each time step: gap creation, gap deletion, and light update. That is, at each time step, we identify the new exposed cells (*gap creation*) and the cells that were exposed in the previous time step but no longer are exposed (*gap deletion*), and then we compute the light values in the updated gap region (*light update*). Figure 7 shows the process of filling of gaps with trees at different time periods after gap creation. These snapshots are taken from a simulation of the model.

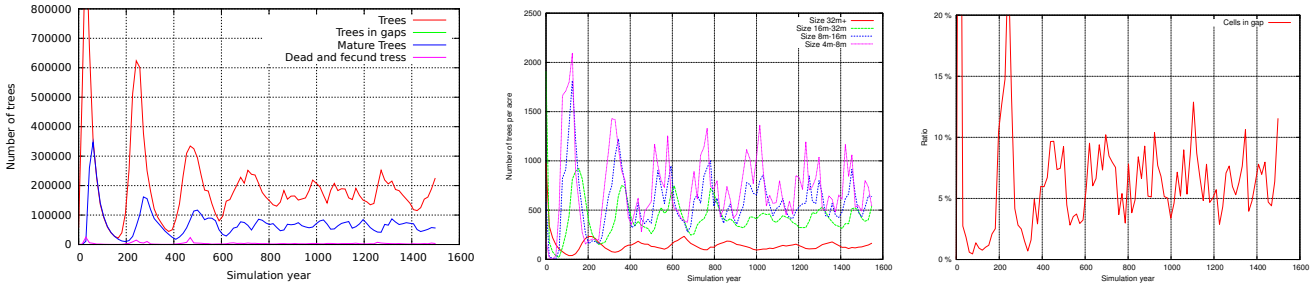
Maintaining gaps. At each time step t we maintain the sets $\mathcal{E}_t, \mathcal{G}_t, \mathcal{B}_t$, and $\mathcal{W}_t$. We maintain $\mathcal{E}_t$ as a list of cells, $\mathcal{G}_t, \mathcal{B}_t$ as bit maps, and $\mathcal{W}_t$ as a collection of orthogonal polygons. The maintenance of $\mathcal{G}_t$ is identical to that of $\mathcal{B}_t$, so we do not discuss it here. Instead of computing these sets anew at each time step, we update them dynamically. Let $\mathcal{I}_t$ denote the set of cells that were not exposed at time $t-1$ but became exposed at time t, and let $\mathcal{D}_t \subseteq \mathcal{E}_{t-1}$ denote the set of cells that were exposed at time $t-1$ but are no longer exposed at time t. Then $\mathcal{E}_t = (\mathcal{E}_{t-1} \setminus \mathcal{D}_t) \cup \mathcal{I}_t$.

Computation of $\mathcal{D}_t$. Let $\mathcal{E}^+ = \mathbb{B}_k(\mathcal{E}_{t-1})$. For each cell $\ell \in \mathcal{E}^+$, we compute its maximum canopy height at time t, i.e., the height of the tallest canopy that intersects j. Next, for each cell $j \in \mathcal{E}_{t-1}$, we check whether any cell of $\mathbb{B}_k(j)$ has maximum canopy height exceeding h_0. If so, j is no longer exposed at time t, and we add j to the set $\mathcal{D}_t$. Since each grid cell of $\mathbb{M}$ contains $O(1)$ adult individuals, the running time of this procedure is $O(k^2|\mathcal{E}_{t-1}|)$.

Computation of $\mathcal{I}_t$. Since tree-canopy heights increase monotonically over time, a cell j not exposed at time t can become exposed at time $t+1$ only if an individual $i \in \mathbb{T}$ whose canopy covers a cell of $\mathbb{B}_k(j)$ dies at time $t+1$ and reduces the exposed canopy height in $\mathbb{B}_k(j)$ to at most h_0. Thus, let $\mathbb{T}_t^\downarrow$ denote the set of deceased trees at time t. For each tree $i \in \mathbb{T}_t^\downarrow$, let $C_i \subseteq \mathbb{M}$ be the set of cells occupied by the canopy of i. For each cell $j \in C_i$, we recompute the exposed canopy height at j. If this values falls below h_0, we search within the $k \times k$ neighborhood of j and compute all cells that become exposed. We add these cells to $\mathcal{I}_t$. Processing each tree in $\mathbb{T}_t^\downarrow$ takes $O(k^2)$ time under the assumption that the canopy of each tree spans $O(1)$ cells, so the running time of this procedure is $O(k^2|\mathbb{T}_t^\downarrow|)$.

Computing $\mathcal{B}_t$ *and* $\mathcal{W}_t$. Recall that, to compute understory light in the gaps at time t, in addition to rendering all trees in the gap $\mathcal{G}_t$, we also need to compute trees in $\mathcal{B}_t$ and the translucent wall erected on $\mathcal{W}_t$. After having computed $\mathcal{E}_t$, $\mathcal{B}_t$ and its boundary $\mathcal{W}_t$ can be computed in a straightforward manner. Since $\mathcal{I}_t$ and $\mathcal{D}_t$ are typically small, we construct $\mathcal{B}_t$ from $\mathcal{B}_{t-1}$ as follows: For each cell $j \in \mathbb{M}$, we maintain the quantity β_j, the number of cells $\ell \in \mathcal{E}_t$ such that $j \in \mathbb{B}_\omega(\ell)$; $\mathcal{B}_t$ is the set of cells ℓ with $\beta_\ell > 0$. Let j be a cell in $\mathcal{D}_t \cup \mathcal{I}_t$. Then j affects $\mathcal{B}_t$ and $\mathcal{W}_t$ within $\mathbb{B}_\omega(j)$. We first process the cells of $\mathcal{D}_t$. Let j be a cell in $\mathcal{D}_t$. For each cell $\ell \in \mathbb{B}_\omega(j)$ we decrement the value of β_ℓ. If it becomes zero, then we also update $\mathcal{W}_t$ around ℓ. Next, let j be a cell in $\mathcal{I}_t$. For each cell $\ell \in \mathcal{I}_t$, we increment the value of β_ℓ. If it becomes 1, we update $\mathcal{W}_t$ around ℓ. The time spent in this step is $O(\omega^2(|\mathcal{I}_t| + |\mathcal{D}_t|))$.

Putting everything together, the total running time spent on updating $\mathcal{E}_t, \mathcal{G}_t, \mathcal{B}_t, \mathcal{W}_t$ is $O(k^2(|\mathcal{E}_{t-1}| + |\mathbb{T}_t^\downarrow|) + \omega^2(|\mathcal{I}_t| + |\mathcal{D}_t|))$.



(a) (b) (c)

Figure 9. (a) Tree statistics for 1500 year run: The total number of trees, the number of mature (taller than eight meters) trees, the number of trees that die, and the number of trees that become reproductive each year. The large oscillations in the early time steps disappear as the model becomes stable after a few generations. (b) This shows statistics similar to (a), but showing the number of individuals in different size classes, per acre of the forest. (c) Percentage of cells in gap areas for a 512m^2 forest.

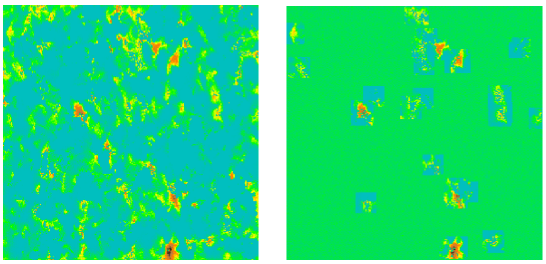
5. EXPERIMENTS

We have implemented both the understory-light and the seed-dispersal algorithms described in this paper and integrated them into the SLIP model. Our experimental results, described below, demonstrate that exploiting the temporal coherence leads to significant speed-up in the algorithm without significant loss of accuracy. We ran our experiments on an Intel Core2 Duo CPU E6850 at 3.00GHz with 4GB of internal memory. We used Ubuntu 10.4 and two 1TB SATA disk drives in a RAID0 configuration. Additionally, the machine contained a NVIDIA GeForce GTX 470 graphics card running CUDA 3.0. This card has 1.2 gigabytes of memory, 448 CUDA cores, and 14 multiprocessors. The algorithm was implemented in C++ using OpenGL to interact with the graphics card.

The experiments in this section were performed on a simulated forest consisting of trees of the species *Acer Rubrum* (red maple), which is one of the two dominant species in Duke forest (the other being *Liriodendron tulipifera*, yellow poplar). The parameters used in Equation (2) for computing the fecundity for the Acer Rubrum species were:

β_0	β_1	β_2	β_3	β_4
4.22	1.5	-0.11	0.388	-0.316

Light experiments. Recall that the main idea in the gap model for computing the understory light is to restrict the explicit computation of understory light in the vicinity of significant openings in the canopy. The parameters k, h_0, ω that determine the gap and buffer areas play an important role. Choosing them conservatively makes the algorithm slow — even slower than the individual-tree model because of overhead of maintaining gaps — and choosing them liberally deteriorates the accuracy. Based on empirical analysis, we choose $k = 5$, $h_0 = 2m$, and $\omega = 20m$ in our simulation. The time spent in rendering the gaps dwarfed the time in maintaining them in all of our tests, so use a more naive algorithm than the algorithm described in Section 4 for maintaining gaps. Figure 9(c) shows the percentage of cells in gap areas during a 1500-year run of a $512 \times 512\text{m}^2$ forest. Only 4-10% of the cells lie in gap areas most of the time, thereby significantly improving the running time — computing the understory light using the gap model took 2.1 seconds on average, including the time for maintaining the gaps, while average running time of the individual-tree model was 24.4 seconds.

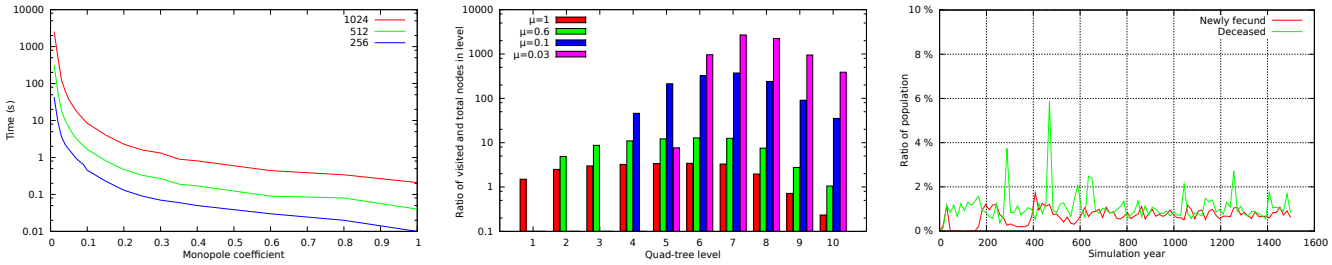


(a) (b)

Figure 10. (a) Light-map computed using the individual-tree model, and (b) using the gap-model. The color scheme maps from blue at 0% light intensity to red at 30%.

Figure 10 (a) shows an understory light-map computed using the individual-tree model, Figure 10 (b) shows the light-map for the following year computed using the gap model. Note that the high light values in the individual-tree model are also captured by the gap model, as desired. In general, light values in the individual-tree model and the gap model differ outside of the gap areas. This difference is an artifact of tree canopies being modeled as cylinders [16], thereby leaving small gaps between trees and causing non-uniform understory light, while in a real forest neighboring mature trees tend to have canopies that are "grown together", producing a uniform cover. The translucency parameter (λ) in the light model is empirically measured in real forests and thus not accurately modeled for trees with cylindrical canopies. In the gap model, this effect is naturally less pronounced.

Dispersal experiments. Figure 11(a) shows the results of running the dispersal algorithm with different monopole coefficients on different size forests; trees were almost uniformly distributed on the grid with an average of one tree for every four cells. The figure shows that the monopole coefficient has a significant impact on the running time of the dispersal algorithm; it becoming dramatically faster as μ increases. For instance, it takes about 1 second on a 1024×1024 forest with $\mu = 0.4$ while taking about 100



(a) (b) (c)

Figure 11. (a) Running time for a single dispersal step in a forest as a function of μ. (b) Number of nodes visited in each of the $\log_2 1024 = 10$ levels of the quad tree for different values of μ and relative to number of cells in the level. (c) Tree statistics for 1500 year run: the number of trees that become fecund or die each year as a percentage of the total number of trees at that year. The y-axis is on a log-scale in both (a) and (b).

seconds with $\mu = 0.05$. This speedup is explained by the histogram in Figure 11(b), which shows how many nodes of each level of the quad tree that are visited by Algorithm 3 relative to the number of cells in that level. For small values of μ, the algorithm performs most of the dispersal directly on the leaves of the quad tree, but as it increases more computation is performed at higher nodes of the tree. The bell shape of the histograms are caused by border effects because of the limited size of the landscape.

The effect of μ on the output is demonstrated in Figure 12. The output for high values of μ is coarse but as μ decreases, the resulting dispersal maps gets smoother. As evident from the figure (and other experiments we performed), decreasing the value of μ below 0.4 does not improve the accuracy much and it remains within uncertainty level of data and the process model, so we choose $\mu \approx 0.4$. The Student's t-distribution used to model the dispersal kernel is very heavy at small distances from the tree, therefore the distance approximation because of Assumptions (A1) and (A2) has significant impact on the approximated values of seed density in the neighborhood of mature trees, as evident in Figure 12.

As shown in Figure 11 (c), the number of trees dying or becoming reproductive each year is small compared to the total number of trees in the forest, thereby making the dynamic dispersal algorithm considerably faster. Over a long simulation like the one in Figure 11, the average time spent updating the temporal approximation is 0.1 seconds which is 10 times faster than running the full algorithm on a 512×512 forest. For a 512×512 forest this implies that the time spent updating every ten year about equal to the total time spent computing the temporal approximation in the nine preceding years.

6. DISCUSSION

In this paper we described new algorithms for computing seed dispersal and understory light in a forest, which exploit temporal coherence. This allows ecologists to simulate processes that span landscapes, but are still subject to interactions between individual trees. Without sacrificing the local details in dispersal and light competition, temporal coherence allows for landscape simulation, with acceptable and known approximation. The spatial detail needed to accurately capture competition within small canopy gaps is preserved by our methods, which is critical for recruitment of the next generation of trees. Dispersal accurately represents the tendency for most seed to fall near the parent trees, but also permits long-distance dispersal. We expect that application of these approaches will allow us to determine how dispersal and competition for light contribute to species coexistence and the diversity of forests.

We are currently refining the gap model and making the light-computation algorithm faster, by defining the notion of exposed cells more carefully. This will reduce the number of trees that are rendered at each time step. At the modeling level, building on SLIP and these algorithms, we are developing an emulator using statistical techniques, which will circumvent the need of maintaining individual trees at all locations in a forest. Such a "hybrid" technique is needed if we wish to perform simulations at a regional scale (e.g. entire North-Eastern US).

Acknowledgements

We thank Sean McMahon for his help with the parameter selection and overall tuning of the model. We also thank Sukhendu Chakraborty for his earlier work on the gap model, which led to some of the ideas of this paper.

7. REFERENCES

[1] S. Arya, T. Malamatos, and D. M. Mount. Space-time tradeoffs for approximate nearest neighbor searching. *J. ACM*, 57, 2009.

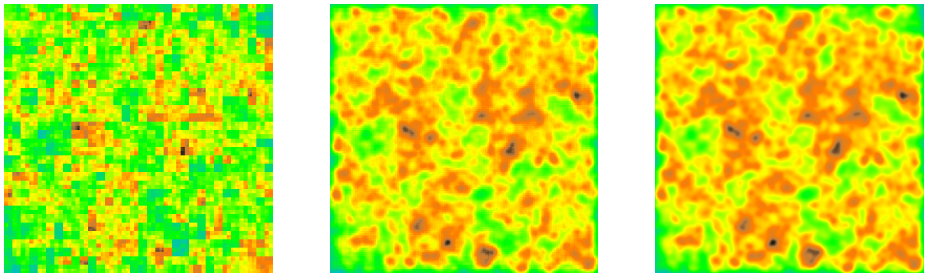
[2] V. A. Barber, G. P. Juday, and B. P. Finney. Reduced growth of Alaskan white spruce in the twentieth century from temperature-induced drought stress. *Nature*, 405:668–673, 2000.

[3] A. Cescatti. Modelling the radiative transfer in discontinuous canopies of asymetric crowns. I. model structure and algorithms. *Ecol. Modeling*, 101:263–274, 1997.

[4] S. Chakraborty. *Scalable Algorithms for a Forest Growth Model: The Understory Gap Light Model and Temporal Dispersal Model.* Master's Dissertation. Duke University, 2006.

[5] D. B. Clark, D. A. Clark, and P. M. Rich, Comparative analysis of microhabitat utilization by saplings of nin tree species in neotropical rain forest, *Biotropica*, 25:397–407, 1993.

[6] J. S. Clark. Individuals and the variation needed for



$\mu = 1.0$ $\mu = 0.4$ $\mu = 0.1$

Figure 12. These maps shows the dispersal of seeds in a single year of the simulation for different values of the monopole approximation coefficient μ. The range of the values is 500 seeds (blue colors) to 8000 seeds (black colors).

gigh species diversity in forest trees. *Science*, 327:1129–1132, 2010.

[7] J. S. Clark, D. Bell, C. Chu, B. Courbaud, M. Dietze, M. Hersh, J. HilleRisLambers, I. IbÃąÃśez, S. LaDeau, S. McMahon, J. Metcalf, J. Mohan, E. Moran, L. Pangle, S. Pearson, C. Salk, Z. Shen, D. Valle, and P. Wyckoff. High-dimensional coexistence based on individual variation: a synthesis of evidence. *Ecological Monographs*, 80:569–608, 2010.

[8] J. S. Clark, S. LaDeau, and I. Ibanez. Fecundity of trees and the colonization-competition hypothesis. *Ecological Monographs*, 74:415–442, 2004.

[9] J. S. Clark, M. Silman, R. Kern, E. Maclin, and J. HilleRisLambers. Seed dispersal near and far: patterns across temparate and tropical forests. *Ecology*, 80:1475–1494, 1999.

[10] M. Dietze. *Regeneration Dynamics in Large Forest Gaps*. Ph.D. Dissertation. Duke University, 2006.

[11] M. Dietze, M. Wolosin, and J. Clark. Tree allometries: capturing diversity using a hierarchical bayes approach. *Forest Ecology and Management*, 256:1939–1948, 2008.

[12] G. P. Elliott and W. L. Baker. Quaking aspen (Populus tremuloides Michx.) at treeline: a century of change in the San Juan Mountains, Colorado, USA. *Journal of Biogeography*, 31:733–745, 2004.

[13] C. B. Field, L. Mortsch, M. Brklacich, D. Forbes, P. Kovacs, J. A. Patz, S. Running, and M. Scott. North America. Climate Change 2007: Impacts, Adaptation and Vulnerability. Contribution of Working Group II to the Fourth Assessment Report of the Intergovernmental Panel on Climate Change. 619–640, 2007.

[14] I. Gamache and S. Payette. Height growth response of tree line black spruce to recent climate warming across the forest-tundra of eastern canada. *Journal of Ecology*, 92:835–845, 2004.

[15] S. Govindarajan, S. Chakraborty, M. Dietze, P. K. Agarwal, J. Clark, and M. Wolosin. A scalable algorithm for understory light computation. *unpublished manuscript*.

[16] S. Govindarajan, M. Dietze, P. K. Agarwal, and J. Clark. A scalable simulator for forest dynamics. In *Proc. 20th Annual Symposium on Computational Geometry*, 106–115, 2004.

[17] S. Govindarajan, M. Dietze, P. K. Agarwal, and J. Clark. A scalable algorithm for dispersing population. *Journal of Intelligent Information Systems*, 29:39–61, 2007.

[18] A. G. Gray and A. W. Moore. 'n-body' problems in statistical learning. In *Proc. 13th Advances in Neural Information Processing Systems*, 521–527, 2000.

[19] L. Greengard. *The Rapid Evaluation of Potential Fields in Particle Systems*. MIT Press, 1988.

[20] S. McMahon, G. Parker, and D. Miller. Evidence for a recent increase in forest growth. pages 3611–3615, 2010.

[21] P. R. Moorcroft, G. C. Hurtt, and S. W. Pacala. A Method for scaling vegetation dynamics: The ecosystem demography model (ED). *Ecological Monographs*, 71:557–585, 2001.

[22] NVIDIA. CUDA homepage. http://nvidia.com/cuda, 2010. 3.0.

[23] S. W. Pacala, C. D. Canham, J. Saponara, J. A. Silander, R. K. Kobe, and E. Ribbens. Forest models defined by field measurements: estimation, error analysis and dynamics. *Ecological Monographs*, 66:1–43, 1996.

[24] P. Ram, D. Lee, W. March, and A. Gray, Linear-time algorithms for pairwise statistical problems, *Proc. Neural Information Processing Systems*, 2009.

[25] P. M. Rich, J. Wood, D. A. Vieglais, K. Burek, and N. Webb. *Hemiview User's Manual, Version 2.1,* 1999.

[26] S. Schartz and Y. S. Yeh. On the distribution of function and moments of power sums with lognormal components. *Bell Syst. Tech. J.*, 61:1441–1462, 1982.

[27] H. Shugart. *A Theory of Forest Succession*. Springer Verlag, 1984.