2011-05-03

# Coherent Turbulence: Synthesizing tree motion in the wind using CFD and noise

Anthony Frank Selino
*Brigham Young University - Provo*

# Coherent Turbulence: Synthesizing tree motion in the wind using CFD and noise

Anthony F. Selino

A thesis submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Master of Science

Michael D. Jones, Chair
Parris K. Egbert
Scott N. Woodfield

Department of Computer Science

Brigham Young University

June 2011

ABSTRACT

Coherent Turbulence: Synthesizing tree motion in the wind using CFD and noise

Anthony F. Selino
Department of Computer Science, BYU
Master of Science

Animating trees in wind has long been a problem in computer graphics. Progress on this problem is important for both visual effects and biomechanics and may inform future work on two-way coupling between turbulent flows and deformable objects. Synthetic turbulence added to a coarse fluid simulation has been used to produce convincing animations of turbulent flows, but only considers one-way coupling between fluid and solid. We produce accurate animations of tree motion by creating a two-way coupling between synthetic turbulence and semipermeable proxy geometry. The resulting animations exhibit global wind sheltering effects and branch tips have motion paths which match paths collected from branch tips using motion capture.

Keywords: Animation, turbulence, solid-fluid interaction, stochastic modeling, tree motion

# ACKNOWLEDGMENTS

I would like to thank Seth Holladay and Side Effects for their help with implementation of these ideas; Wolfram Diestel for developing Arbaro. My advisor for helping me get this far and my dear wife for her support, encouragement, patience, and love. Finally, I would like to thank BYU and all the wonderful faculty and people here who I have had the pleasure of working with and learning from.

## Contents

## List of Figures

# Chapter 1

## Introduction

Plants and trees are nearly ubiquitous in visual storytelling. However, our ability to express plant motion for film and other visual mediums is limited. Current approaches are good at expressing plant motion in calm or violent winds, but not in medium winds where wind direction is visible on foliage during gusts and the tree's influence on the wind is significant. The scale of turbulent motion that one might observe in a tree is shown in Figure 1.1. Kinds of motion that can be generated using existing methods are highlighted in the red boxes. Animating trees on calm days is, as one might expect rather trivial as the tree is stationary. Animating a tree in a gentle breeze can be done by applying a low frequency low amplitude motion to the tree. This is done in several scenes in the movie "Monsters Inc." [Doc01] in which background trees often appear to move in a light breeze ( see Figure 1.2 ). Motion in gale force winds can be simulated by creating violent motion in the tree. Tree motion in gale force winds can be see in the film "Avatar" [Cam10] when various spacecraft appear to land in wooded areas.

In this thesis, we focus on the problem of animating trees in a strong breeze. This problem is particularly difficult because tree in strong breezes exhibit coherent motion that causes visually significant effects. Figure 1.3 is a photograph of a tree experiencing a strong breeze. Note that the tree crown exhibits more deformation on the windward side than the leeward side. In addition, the branches sweep back and this minimizes the profile of the crown and reduces aerodynamic drag.

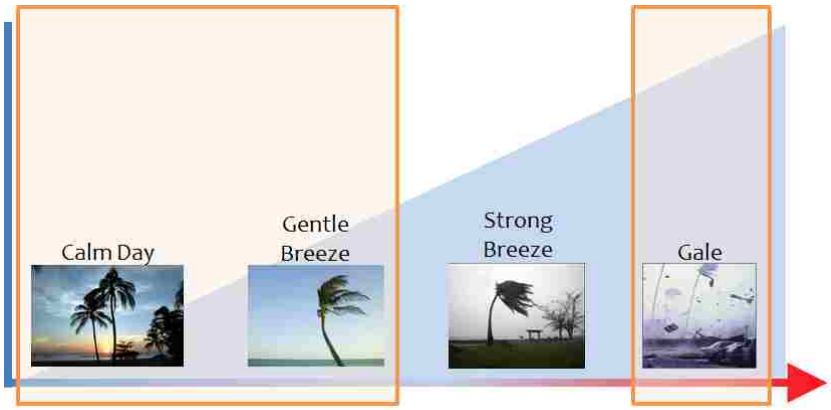


Figure 1.1: Using current techniques it is possible to represent plant motion in calm winds or strong gales, but not in moderate wind conditions such as a strong breeze.

Figure 1.2: Films like "Monster's, Inc." by Pixar use trees as background elements. In this scene the trees moves in a light breeze.

The problem of accurately modeling trees in wind is also important in forestry. Prior work on modeling trees in extreme gale force conditions is not suitable for this application as this work does not attempt to accurately model turbulent flows through forest canopies. Forest managers face the task of minimizing loss due to wind throw over the lifespan of a stand of trees. Depending on the species, the lifespan of a single stand may cover 100 years or more. There is a high probability that a stand will experience intense wind events over a 100 years span. Foresters must design pruning treatments which will withstand this rare but intense events. A better understanding of how tree interact with turbulent flows may yield insights into better pruning methodologies.

Figure 1.3: A real tree blowing in moderate wind before a storm that produced tornadoes. The tree exhibits both high-frequency flutter and it is possible to see visible gusts pass through the tree. Furthermore, the tree's windward side (left) blocks and slows the wind (wind sheltering) such that the leeward side (right) of the tree exhibits significantly less motion.

Our work focuses on improving the animation of trees in moderate wind. As shown in Figure 1.4, we consider the two-way interaction between trees and wind. We choose a simple

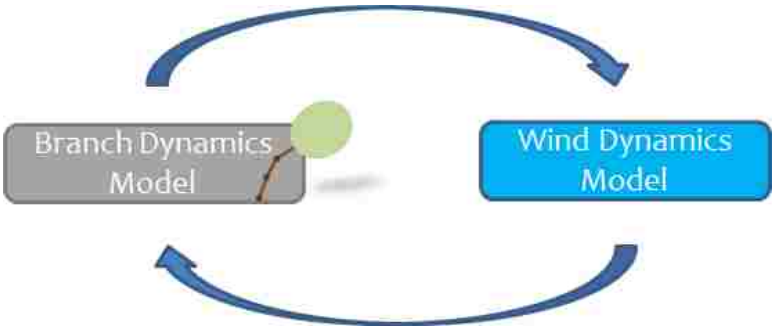


Figure 1.4: A simple model for plant motion. Almost all current approaches ignore the effect of the tree on the wind.

branch model and focus our research on improving the dynamic wind model and the two-way interaction between trees and wind.

Animating trees in wind is a difficult problem. Two-way interaction between wind and trees is challenging to model directly because trees contain many small, flexible moving parts which both generate and respond to both large and small turbulent eddies. In this paper, we address these issues by merging coarse fluid simulation with synthetic turbulence in the presence of two-way coupling between fluid and participating solids. This work is motived by motion capture data of trees moving in wind in which recorded branch tip motion paths include visually significant responses to large and small scale turbulence and exhibit asymmetric motion (across the streamwise flow direction) due to crown sheltering. Crown sheltering refers to the sheltering of leeward branches by windward branches which results in lower amplitude motion on the leeward side compared to the windward side.

Tree motion in wind should also exhibit crown streamlining. Crown streamlining occurs when branches in the tree crown sweep backward in response to wind. This sweeping effect reduces the profile presented to the wind and reduces the aerodynamic drag on the crown. Streamlining, like sheltering, is a globally coherent effect.

There is a growing body of work in computer graphics related to tree animation. This work can be split into two groups: spectral approximations that include small-scale turbulence effects, but omit crown sheltering and streamlining and fluid simulations that include crown sheltering but omit small-scale effects. Spectral approximations have received the most

attention recently but omit the creation and damping of turbulence by trees. We extend recent advances in combining fluid simulation with turbulence synthesis using two-equation turbulent kinetic energy (tke) budgets in one-way coupled flows [PTC+10] to achieve both small-scale turbulent effects with large scale sheltering effects in a two-way coupled simulation.

In this paper, we combine a Fluid-Implicit Particle (FLIP) [ZB05] simulation with a two-equation tke budget and semipermeable proxy geometry to obtain both small-scale turbulent effects and large scale sheltering effects in stands of up to 18 trees. We describe several developments which are new to the graphics literature and advance the state of the art in modeling trees in wind for forestry. Our main contributions are:

- a model in which tke is split into aeolian and arboreal components that are generated, advected and dissipated differently which allows turbulent energy to both move with the fluid and become entrapped in participating objects,
- an approach to two-way coupling in which semipermeable proxy geometry is used to abstract interactions between small flexible solids and fluid flow which allows us to mimic the effect of fluid flowing over a complex flexible object like a tree, and
- a model for generating tke in a FLIP simulation rather than a SMAC simulation which allows us to move from a purely Eulerian fluid simulation to a partially Lagrangian approach and this simplifies the use of proxy geometry because particle-sphere intersections are simpler than arbitrary cube-sphere intersections.

These contributions result in tree motion which exhibits global sheltering effects and which generates branch tip motion paths that are comparable to branch tip motion paths obtained from motion capture data recorded in a laboratory setting. The resulting motion can be controlled by adjusting the stiffness of the branches, the permeability of the proxy geometry and the evolution and dissipation of tke.

## 1.1 Thesis Statement

Splitting tke into aoelian and arboreal components in a two-way coupled FLIP fluid simulation with synthesized subgrid-scale turbulence results in tree motion which exhibits sheltering and branch tip motion paths with tortuosity similar to motion paths captured in 3D motion capture experiments.

# Chapter 2

# Related Work

Our work is related to work in the animation of tress for image synthesis and the use of synthetic turbulence in fluid flow simulation.

## 2.1 Animation of Trees

Animating trees in wind has a long history in graphics and can be split into one-way methods in which only the wind influences the trees and two-way methods in which the trees also influence the wind. Several methods for modeling branch dynamics have also been proposed and are discussed below.

In one-way methods, air flow influences trees, but trees do not influence wind. Because trees do not influence air flow, turbulence must be artificially added to the simulation. This is generally done using variations on the power spectrum of turbulent eddies created by wind passing through trees. [SF92] and [HKW09] used an empirically-derived formulation of the wind's power spectrum. Stam [Sta97] used filtered noise and Ota [OFT+03] used $1/f^{\beta}$ noise. Zhang [ZST+06] used the cross-power spectrum density matrix. These one-way methods do not require simulation, but cannot reproduce complex wind dynamics such as the effect of trees on wind. We build on these methods by using an empirically-derived wind noise together with a coarse fluid simulation to produce tree motion that includes these complex wind dynamics.

Akagi [AK06] describes the only two-way coupled fluid simulation for tree animation in computer graphics. Akagi's work uses a grid-based fluid solver to model two-way dynamics

of wind and trees. This method captures wind sheltering and redirection as seen in nature. However, it does not account for turbulence smaller than the grid will allow which, for a scene with four trees, they used a 50x20x50 grid with a cell size of .25m. Because they use a grid-based solver they must estimate tree volume occupying each cell. This involves generalized sphere or cylinder intersection tests with each cell for each frame or through some other approximation. We use a FLIP solver and calculate whether fluid particles are inside the tree's proxy geometry. This allows us to avoid any volume estimations while still modeling full two-way coupling between wind and trees. Moreover we add a sub-grid scale model to capture turbulence at scales below the grid resolution which allows us to represent turbulence on the tree at a finer detail than Akagi even when using significantly larger grid cells (1.6m).

Additional two-way coupled models appear in graphics and forestry ecology literature. In graphics, these approaches omit frequencies smaller than the grid size such as for bubbles [WZF+03] or deformable solids and shells [RMSG+08]. In forest ecology subgrid frequencies are considered [DGP+10], but a group of plants is treated as a continuous object which is not desirable for image synthesis.

The branch model is another issue in the animation of trees. We do not create a new branch model, but use a common mass-spring system similar to [SO99] where each branch is subdivided into several segments and connected by torsion springs. Our current model does not consider the transfer of force between different branch segments, branch twisting, or branch-branch collisions. Despite these limitations, we chose this approach because mass-spring systems are common in hair and tree animation and allow two-way coupling with wind and other objects in a scene. Furthermore, mass-spring systems can be adapted to handle transfer of forces [Web08] and torsion [Had06].

Other methods for animating trees do not discretize branches into several segments, but use modal analysis ( [SF92], [Sta97], [DRBR09] ) or the Euler-Bernoulli beam model ( [HKW09] ) to calculate deformation, but these approaches cannot be adapted for two-way

coupling because they either pre-compute motion or reactions to a specific instance of a wind field.

## 2.2 Synthetic Turbulence


Figure 2.1: Synthetic turbulence is used a post-process effect on coarse fluid simulations.

Synthesizing turbulence is one approach to overcome the limitations of low-resolution simulations in representing turbulent flow. Two-way coupled tree motion is especially well suited to using a low-resolution simulation together with synthetic turbulence because wind is invisible.

Previous methods improving low-resolution simulations modeled turbulence using the Kolmogorov spectrum, but did not account for the physically-based evolution of turbulence in a velocity field. Some papers which are representative of this work are [SE93], [SRF05], [ZYC10] ). More recent approaches, however, do account for the actual evolution and dissipation of turbulence as it corresponds to an evolving velocity field ( [KTJG08], [SB08], [NSCL08] ). [PTC+10] gets an accurate production term for turbulence by using the k-$\epsilon$ model while also accounting for anisotropic turbulence.

Similar to [PTC+10] we adapt the k-$\epsilon$ model to tree motion by evolving turbulent energy on particles, but, rather than using turbulent energy to improve the appearance of our fluid, we use turbulent energy to animate branch segments interacting with fluid.

Furthermore, we omit anisotropy, use a spectral approach rather than a curl-based method such as wavelet noise, and adapt the k-$\epsilon$ model to a particle method.

# Chapter 3

# Methods

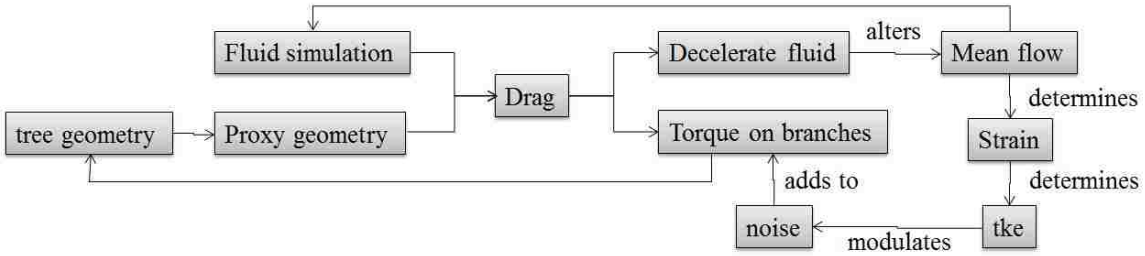


Figure 3.1: **Overview** A coarse wind-fluid simulation is initialized. For each frame we calculate the turbulent energy on the wind and use that information combined with the velocity to synthesize turbulent wind on branches. We move branches and update the tree's proxy geometry which, in turn, affects the fluid simulation.

Figure 3.1 outlines our method for animating trees moving in the wind. A coarse fluid simulation interacts with trees through proxy geometry which is attached to actual tree geometry. As fluid flows through proxy geometry, the proxy geometry exerts a drag on the fluid which causes deceleration of the fluid and creates torque on tree branches. Decelerating the fluid alters the mean flow. Mean flow in a region is used to determine strain and strain determines turbulent kinetic energy (tke). The tke term is used to modulate frequency-tuned noise which matches the frequency distribution of turbulent structures below the scale of the fluid simulation. The modulated noise adds to the torque on branches. Torque on branches moves the proxy geometry which then feeds into the next step of the simulation. In this section, we describe this process in detail.

Figure 3.2 illustrates the interaction between fluid particles as well as tke advected with particles and attached to branch segments. The diameters of both the particles represent

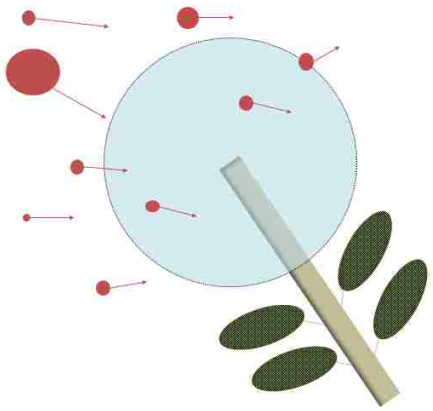

Figure 3.2: Proxy geometry (blue) and fluid particles (red) interacting for one branch segment. Fluid particle diameter represents the amount of tke advected with a particle.

the strength of the tke they carry with larger diameters representative of larger tke values. This figure will be used as a running example in the following sections.

## 3.1 Turbulent Kinetic Energy (tke)

A tke model based on fluid strain and passing tke between wind and trees is used to add physically-based, high frequency turbulence which is keyed to the low-frequency turbulence in the FLIP simulation. Tke is split into aeolian (particle tke) and arboreal (tree tke) components which are generated, advected, and dissipated differently. Tke values modulate noise textures which represent wind velocities due to turbulence at scales below the fluid simulation. Noise textures are Gaussian noise tuned in the frequency domain to match the power spectrum described by the Kolmogorov cascade. The method is sound because the strength of small turbulent eddies is proportional to the strength of large turbulent eddies. This relationship is described by the Kolmogorov cascade and large eddies are generated by the fluid simulation.

We use a FLIP simulation to model wind. This allows us to model turbulence up to the Nyquist limit of the simulation grid and proxy geometry as they interact. This lower-frequency turbulence is important because it includes sheltering effects that are ignored

in tree animation models based only on spectral synthesis of turbulence. However, much of the fluttery motion seen in real trees is developed from frequencies higher than the grid resolution. Motion at these frequencies is so important that it is the main feature of spectral synthesis approaches.

We also tried a purely Lagrangian approach using a Smoothed Particle Hydrodynamics (SPH) simulation rather than the hybrid FLIP simulation. As is common to other uses of SPH, fluid particles tended to separate, clump, and explode due to incorrect pressure estimations which results in an unstable simulation that is not suitable for modeling wind. FLIP, on the other hand, remains stable by equalizing pressure over a grid rather than using particle neighborhoods. This results in a stable wind simulation.

Similarly, we could have used a purely Eulerian simulation, such as simplified marker and cell (SMAC), rather than FLIP, as was done by Akagi. However, using SMAC would have required computing intersections between cubes and spheres and this is more expensive than computing intersections between spheres and points.

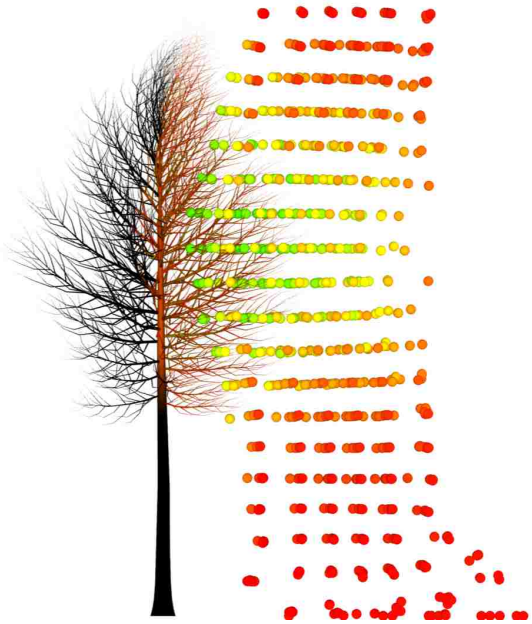

Figure 3.3: Without correction, high frequency branch motion disappears suddenly as tke-carrying FLIP particles leave a branch segment's sampling radius. Orange branches have high frequency motion whereas black branches have no high-frequency motion. To remove this artifact, tke is split into arboreal and aeolian components which are advected differently.

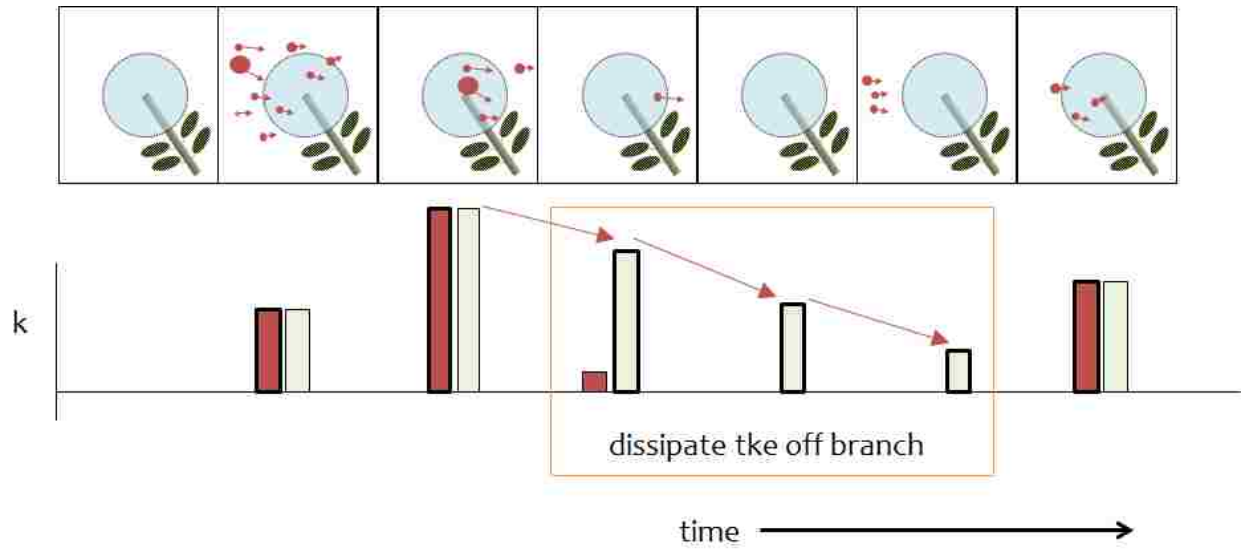


Figure 3.4: Treatment of aeolian tke (red bars) and arborial tke (tan bars) across several frames of simulation. Aeolian tke carried by fluid particles is represented by the diameter of the fluid particle. Tke values shown on the bottom with bold lines represent the tke used in a given frame. By using arboreal tke we can smoothly dissipate tke off branches.

We use a two-equation tke budget as in Pfaff [PTC+10] and Pope [Pop00] but split tke into aeolian and arboreal components which are advected and dissipated differently. Managing tke in two components results in smooth dissipation of tke. The tke advected on fluid particles influences motion until those particles exit the proxy geometry. Particles carrying tke may exit the proxy geometry with non-zero tke values. Visually, this results in abrupt changes in high-frequency turbulent tree motion. This is shown in Figure 3.3. In the figure, orange branches have tke-carrying FLIP particles in their proxy geometry while black branches do not. Hard lines between black and orange branches results in a sharp change in motion as tke drops to zero and creates a visual motion artifact. Attaching tke to branch end segments results in smoother animation and mimics the effects of small scale turbulence entrapped in participating foliage.

Figure 3.4 illustrates the relationship between aeolian and arboreal tke. Boxes at the top contain a single branch endpoint located in the middle of the box. Each box represents one frame of a fictitous simulation. The timeline at the bottom of the figure depicts the amount of aeolian and arboreal tke at the branch segment in each frame. Bars with a bold

outline represent the tke used to drive motion. In our approach, only the greater of the two tke values present, either aeolian or arboreal tke, drives motion in a given frame. This allows strong turbulent gusts to dissipate naturally on a segment even when new weaker gusts later enter the segment's proxy geometry. Aeolian tke, represented by the red bars, is averaged from particles that lie within the proxy geometry of the branch end segment (shown as a blue circle). When aeolian tke is present and greater than turbulence on the tree, the arboreal tke value is updated to match the sampled aeolian tke but is not used to drive motion.

### 3.1.1 Aeolian tke

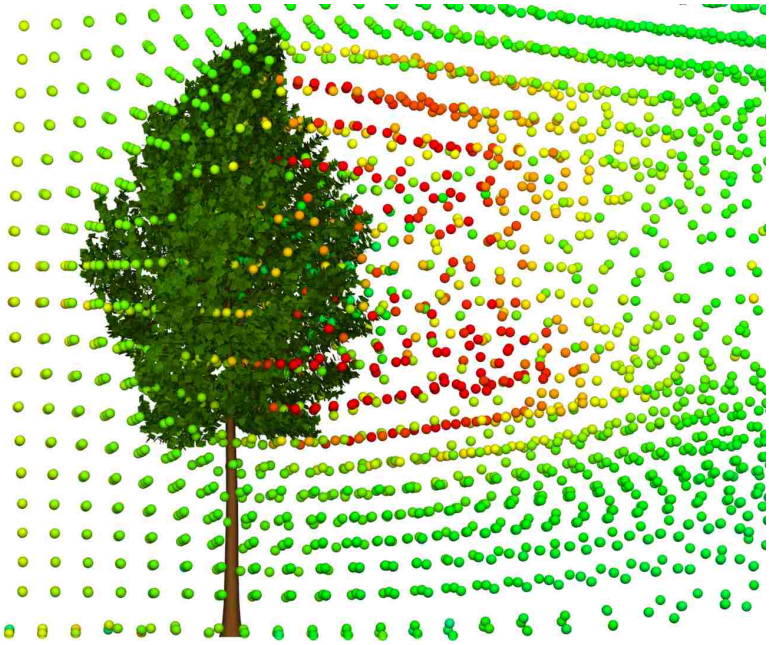

Figure 3.5: Aeolian Turbulent Kinetic Energy (tke) is stored and advected on wind particles. In this image red particles have large tke values and green particles have small tke values.

FLIP particles advect aeolian tke. Figure 3.5 shows an example of tke advection by fluid particles. In the figure, shaded spheres represent fluid particles and particles shaded red carry the most tke while particles shaded green carry the least. As expected, tke is present in the wake of the tree.

A simplified two-equation tke budget models aeolian tke values stored on FLIP particles. This is similar to Pfaff [PTC+10]. The $k - \epsilon$ model is an approach to estimating tke where $k$

is a measure of the turbulent energy at a location and $\epsilon$ is the dissipation rate of that energy. The evolution of $k$ and $\epsilon$ are given by

$$\frac{\partial k}{\partial t} = P - \epsilon \tag{3.1}$$

$$\frac{\partial \epsilon}{\partial t} = C_{\epsilon 1}\frac{P\epsilon}{k} - C_{\epsilon 2}\frac{\epsilon^2}{k} \tag{3.2}$$

where $P$ is the production of turbulence in the fluid and $C_{\epsilon 1} = 1.44$ and $C_{\epsilon 2} = 1.92$ are additional model constants. The production of turbulence, $P$, at a point is based on the strain $S_{ij}$ and the turbulent viscosity, $v_t = 0.09\,(k^2/\epsilon)$. $P$ is defined using

$$P = 2v_t \sum_{ij} S_{ij}^2 \tag{3.3}$$

where

$$\{(i,j) \in \{x,y,z\} | i \neq j\} \tag{3.4}$$

and

$$S_{ij} = \frac{1}{2}(\frac{\partial U_i}{\partial j} + \frac{\partial U_j}{\partial i}) \tag{3.5}$$

We adapt this equation to FLIP as shown in Figure 3.6. For each fluid particle, we sample velocities at six points of interest set at a distance $r$ from the particle in positive and negative $x$, $y$, and $z$ directions. To find the velocities we distance weight all particles within a sphere placed at each point of interest. The sphere's radius, $r$, is a function of the grid size such that the sphere contains a single grid cell. Since FLIP's resolution is effectively chosen by the grid size, choosing the correct size allows us to avoid undersampling the fluid and introducing artifacts and oversampling the fluid which would smooth out local velocities. Strain can then be calculated similar to Pfaff [PTC+10] using central differencing.

The update equations for $k$ and $\epsilon$ only apply when production is nonzero. When $P$ is zero both $k$ and $\epsilon$ dissipate where

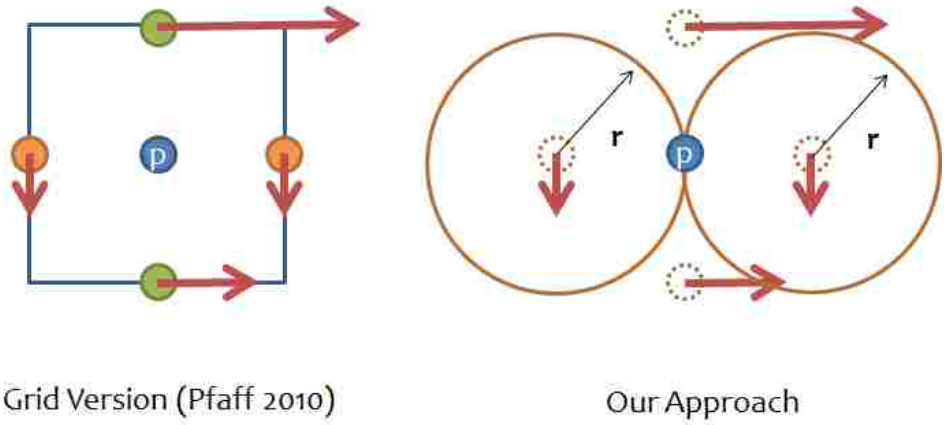


Figure 3.6: We calculate turbulence production, $P$, by averaging particles contained within spheres (orange) with a radius that is a function of the grid size such that each sphere can contain a single grid cell. In this figure only spheres on the $x$-axis are shown.

$$k(t) = k_0(\frac{t}{t_0})^{-n} \tag{3.6}$$

$$\epsilon(t) = \epsilon_0(\frac{t}{t_0})^{-(n+1)} \tag{3.7}$$

where $t$ is the current timestep, $k_0$ and $\epsilon_0$ are last values of $k$ and $\epsilon$ when turbulence production was nonzero. Furthermore the decay component $n = \frac{1}{C_{\epsilon 2}-1}$. Lastly, the reference time $t_0$ is given by

$$t_0 = n\frac{k_0}{\epsilon_0} \tag{3.8}$$

In case $k$ or $e$ are zero (such as before wind reaches our trees) we set $k$ and $\epsilon$ to a small constant.

### 3.1.2 Arboreal tke

For each branch segment we sample both tke and $\epsilon$ from nearby particles. We do not calculate the production of turbulence, $P$, or the evolution of $\epsilon$ on the branch segments as arboreal tke exists to capture turbulence that has already been produced by the change in fluid

flow. However, we do dissipate arboreal tke using Equation 3.6 though $n$ can be artistically modified to change the rate of turbulence dissipation on the tree.

If fluid particles later enter the proxy geometry, then aeolian tke is again sampled and—if greater than arboreal tke—used to drive motion and update arboreal tke values.

### 3.2 Forces Acting on Branch Segments

We decompose forces acting on branch segments into aerodynamic drag forces due to two kinds of flow: the mean flow and the turbulent flow. The velocity of the mean flow is sampled directly from the fluid simulation and an estimate of the aggregate turbulent flow is generated using tke to modulate a frequency-tuned noise field. The mean flow around a branch endpoint is calculated similar to how we sample the fluid to calculate strain in Section 3.1.1. We use a distance weighted average of the particle velocities within the proxy geometry of that endpoint where the proxy geometry's radius is scaled to match the size of the FLIP grid. The tke term scales the magnitude of a turbulence velocity vector which is taken from a noise field.

Rather than using tke to visualize turbulent flow directly, as in prior work on turbulence combined with fluids, we use tke to synthesize turbulent flow velocity acting on participating objects—tree branches, in this case. This is done by scaling synthesized noise by the tke. Instead of using band-limited wavelet noise we opted to use a spectral approach similar to [HKW09] because we're not visualizing the wind and Habel's noise is quicker to implement.

We synthesize a noise velocity texture, $\mathbf{V}$, by creating three complex 2d Gaussian random fields in the frequency domain, filtering them using the Kolmogorov scale shown in Equation 3.9,

$$P_v(f) = \frac{u_{mean}}{(1 + \kappa f / u_{mean})^{5/3}} \tag{3.9}$$

and then applying an inverse FFT operation to obtain the velocity texture. The Kolmogorov scale matches the power spectrum of wind from empirical measurements [SS78]. Because

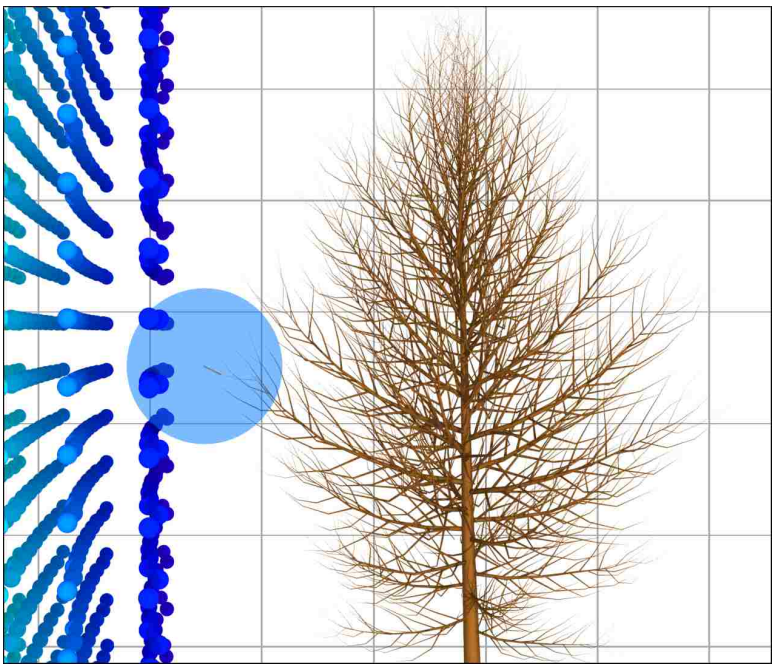

Figure 3.7: The blue sphere in this image is the proxy geometry for a single branch segment. Forces acting on each branch segment are calculated by sampling particles within a radius that is a function of the cell size of the grid.

we model velocity and not motion using the noise field, we do not convolve the noise with a damped mass-spring response as in Habel [HKW09].

Force due to the mean flow are computed directly using aerodynamic drag and velocity sampled from the fluid simulation. To obtain wind velocity samples for each branch segment in the tree model we sample nearby particles for velocity, $u$, as in Figure 3.7.

Once the neighboring flow is sampled, we synthesize a complete flow vector $U$ which represents one instant in a time-varying full-spectrum turbulent flow which is consistent with the mean flow velocity. The vector $U$ is defined as

$$U(t) = u(t) + \alpha\sqrt{k(t)}\mathbf{V(t)}$$

The noise texture $\mathbf{V}$ is sampled by assigning each branch segment sample point a unique position and trajectory in $\mathbf{V}$. For each time step, the segment's position in $\mathbf{V}$ is updated and $\mathbf{V}$ is re-sampled. The tke, $k$, is sampled from the fluid in the same manner as $u$, or is taken

from the arboreal tke if no aeolian tke is present, and is used to scale $\mathbf{V(t)}$. The parameter $\alpha$ is a tunable parameter used to vary the contribution of the scaled turbulence.

### 3.3 Forces Acting on FLIP Particles

Figure 3.8: Simple but abundant proxy geometry closely matches the tree's shape even under large deformations.

Proxy geometry allows us to quickly approximate the cumulative drag force exerted by small clumps of vegetation on wind. Proxy geometry consists of spheres centered on branch segments whose diameter is a function of segment length and sphere spacing. This allows us to avoid placing proxy geometry for each leaf because leaves are nearly covered by their parent branch's spheres. Proxy geometry could be placed per leaf rather than per branch to

potentially achieve greater realism. For tree models with many leaves, such as Shek [SLS+10], this could be intractable. In Figure 3.2, particles contained in the union of proxy geometry elements experience a drag force.

Particles experience the same drag force irrespective of how many proxy geometry elements contain the particle. This approach produces shielding and redirection that is visible on nearby branches. The drag force experienced by particles as they travel through proxy geometry could be adjusted to mimic changes in aerodynamic drag resulting from changes in tree shape. For example, crown streamlining reduces the frontal area of vegetation as the vegetation bends in the stream wise flow direction.

This approach does not enforce fluid-solid boundaries. Particles are free to flow through tree geometry. This saves considerable time by avoiding the computation of intersections between particles and tree geometry. Fortunately, fluid-solid penetration is irrelevant in this context because we are not interested in visualizing smoke or fluids as they interact with a tree. Furthermore, by not enforcing boundaries we allow wind to reach the inner branches and leaves despite using a coarse fluid simulation. If boundaries were enforced, particles would be trapped by the complex leaf geometry and their velocities excessively damped because the overall flow could not resolve the fine pressure details required for the flow to navigate through a densely-populated tree crown. As discussed above, synthesized turbulence compensates for the high frequency turbulence that would have been, but is not, generated from the abstracted collisions between fluid particles and actual geometry.

By varying the size and number of spheres we can increase the accuracy of turbulence produced by simulation of tree-to-wind interaction up to the Nyquist limit of the fluid simulation. Increasing the fidelity of the proxy geometry without increasing the number of particles (and the number of grid cells for hybrid methods) will yield limited visual improvement.

### 3.4 Implementation Details

We developed our approach as a Houdini extension and used tree models from Arbaro, an open-source program based on [WP95]. Similar to other approaches (Sakaguchi [SO99], Habel [HKW09] ) branches operate in a local coordinate system. Our branches are divided into several branch segments where the total force acting on a branch segment is the sum of the aerodynamic drag from the turbulent field as in 3.2 and the segment's own internal spring and damping forces. Angular acceleration is then calculated by dividing the various forces (torque) by the segment's moment of inertia.

One of the challenges of tree motion is avoiding manually setting these parameters for the thousands of non-uniform segments that compose a tree. We found it helpful to scale each segment's spring stiffness by its moment of inertia. This allowed our branch segments of various, unequal lengths to bend at similar frequencies. Furthermore, we model the branch elasticity by varying the moment of inertia and spring stiffness throughout the tree as a rough approximation of wood age [AK06].

We used the FLIP solver implemented in Houdini 11 [Sid11] to model wind. A ground plane and sky plane simulation atmospheric pressure. A restorative force accelerates slow particles back to the mean flow speed. This mimics the effect of large-scale pressure differentials over land which cause natural wind and prevents particles from being trapped in foliage.

# Chapter 4

## Results and Discussion

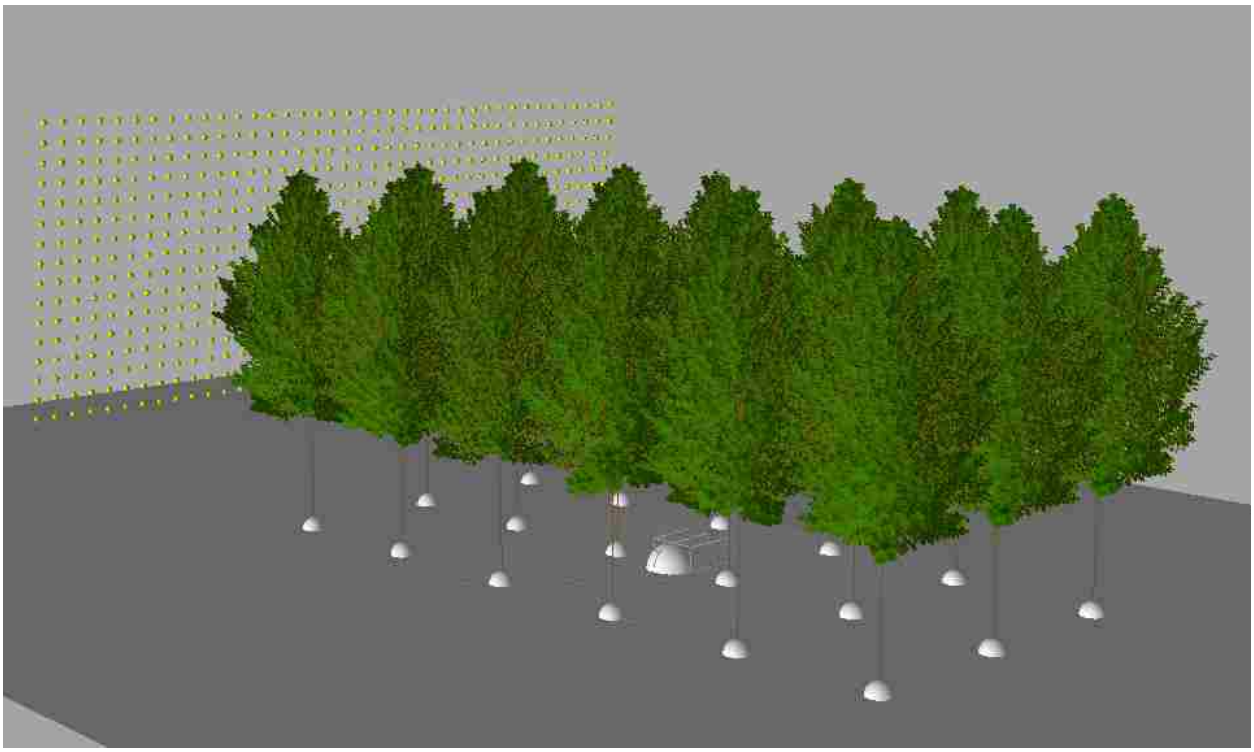

Figure 4.1: We simulate a stand of 18 trees in a wind event using Houdini.

Our approach combines both the turbulent motion of the branches and the global interactions between trees and wind. Proxy geometry closely matches the complex shape of a tree even under large deformations (Figure 3.8 ).

These contributions result in tree motion which exhibits global sheltering as seen in Figure 4.2. In this image we demonstrate the motion paths of six trees after a wind event where paths are colored by length. In this image red paths are the longest paths and violet-blue paths are the shortest paths. The left-most tree exhibits significantly more motion

Figure 4.2: 1200 branch motion paths on 6 trees (200 per tree) after a 12-15 m/s wind event using a 1.6m grid for fluid simulation and a turbulence model based on strain estimation and the Kolmogorov cascade.

and causes global sheltering which causes trees on the leeward side of the wind to move less (as shown by the many tiny blue paths). Furthermore, this sheltering causes changes in fluid velocity which result in a rolling cascade of turbulence on the tops of the trees. Thus the right-most trees have significantly more motion at their tops than on the left. While not as obvious from an image we believe that much of this motion is due to the artifact of our simple mass-spring model. As our model lacks two-way coupling the turbulent forces entering the top of the trees cannot dissipate down into larger branch segments and must be resolved by the small, thin segments at the top. We believe that with a more sophisticated model this would be corrected. To generate this image we simulated a stand of 18 trees ( as shown in Figure 4.1 ) contacting with a 12-15 m/s wind. Each tree uses the same parameters and is composed of 1,229 branches and 22,110 leaves. For illustrative purposes and to avoid additional complexity in rendering we rendered only the middle 6 trees.

As shown in Figure 4.3, our method can produce turbulent motion similar to real tree motion obtained through a passive motion capture system. For the motion capture data we placed markers on a small tree in a laboratory setting and used a large fan to apply a constant wind force to the tree. Similarly, in our test we applied a light constant wind to our tree (with and without turbulence) and captured the motion paths after the first 100

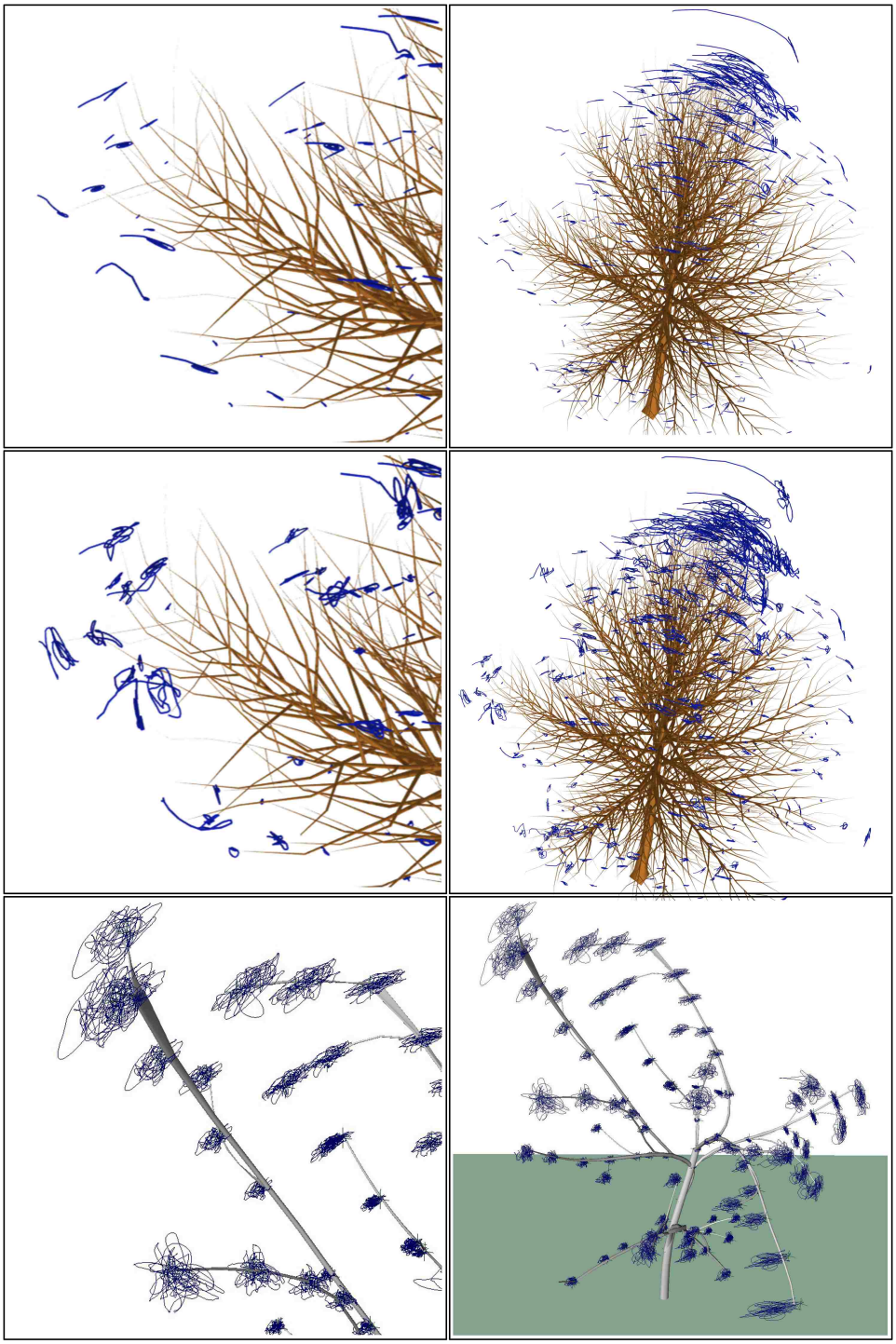

Figure 4.3: Motion Paths of a quaking aspen tree in light, constant wind of 3 m/s. The top image is motion from a fluid simulation without synthesizing turbulence. The middle image is the same simulation with our method. The bottom image is the motion paths of a real tree captured in light wind. Notice that the motion paths from our approach twist and bend significantly more than without turbulence. These twists and bends more closely match the motion capture data.

frames. We note that our method produces paths that bend and twist significantly more than approaches without considering subgrid turbulence. Tortuosity the measure of the amount of twisting. The Oxford English Dictionary defines tortuosity as "Full of twists, turns, or bends; twisted, winding, crooked, sinuous."

We find that our approach produces motion paths that closer approximate the tortuosity observed in the motion capture data than two-way coupled approaches without considering subgrid turbulence. We found that our method consumed an insignificant fraction of the processing time per frame. In Figure 4.2, total simulation time averaged 4 seconds per frame with 0.08 milliseconds per frame being used for the tree motion and turbulence calculation step.

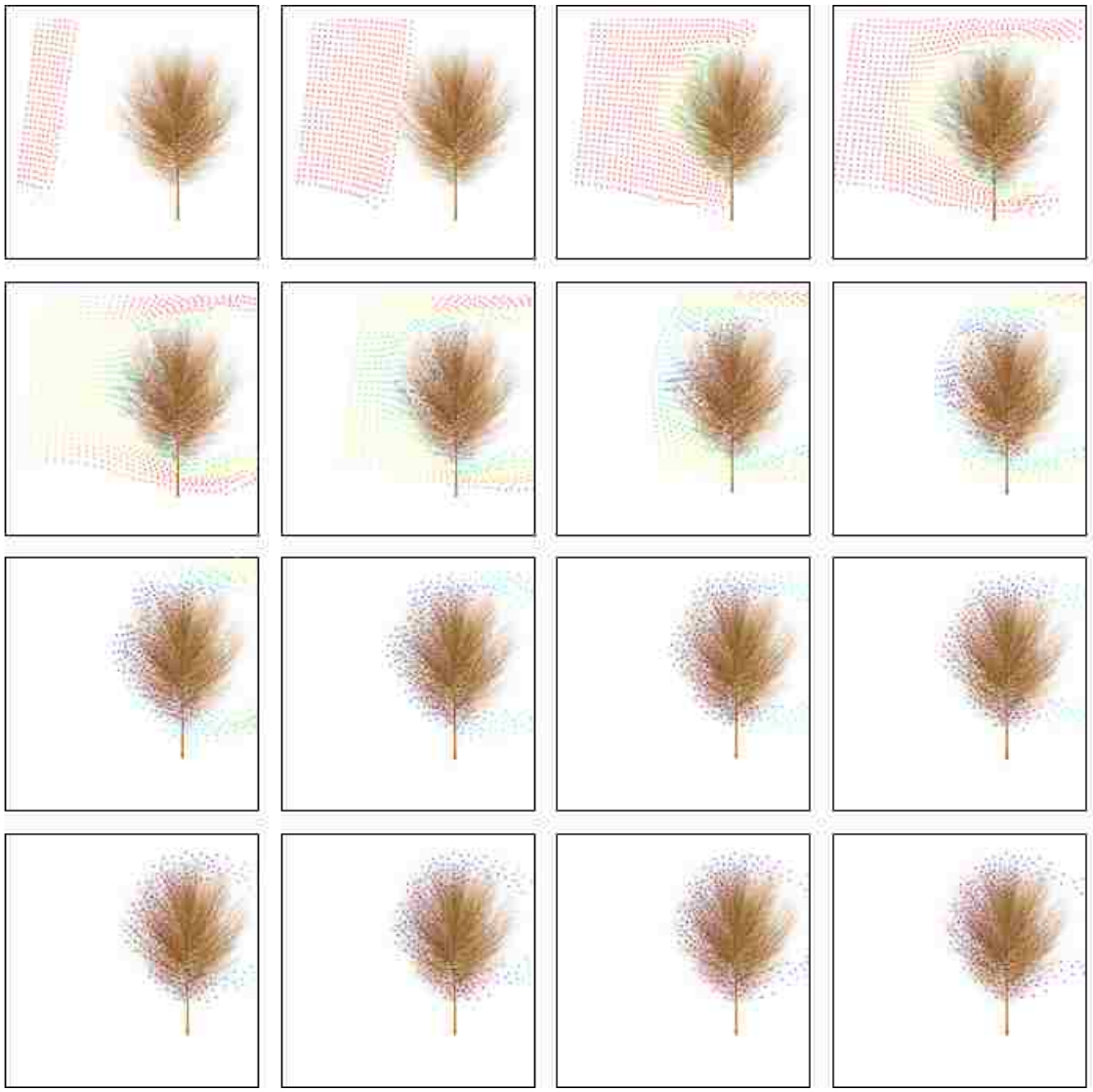

Figure 4.4: **16 frames taken from a 160 frame simulation.** In these images a large tree impacts with a moderate wind moving from 10-18 m/s. Fast moving particles are colored red; slow particles are colored blue. In frame 4, particles moving through foliage move much slower than particles outside and cause a build up of pressure. This pressure buildup causes newly emitted particles to be slowed (frame 5) and curl and wrap around the tree. The tree completely stops the particles in the crown and these particles continue to exert a damping force on all other particles.

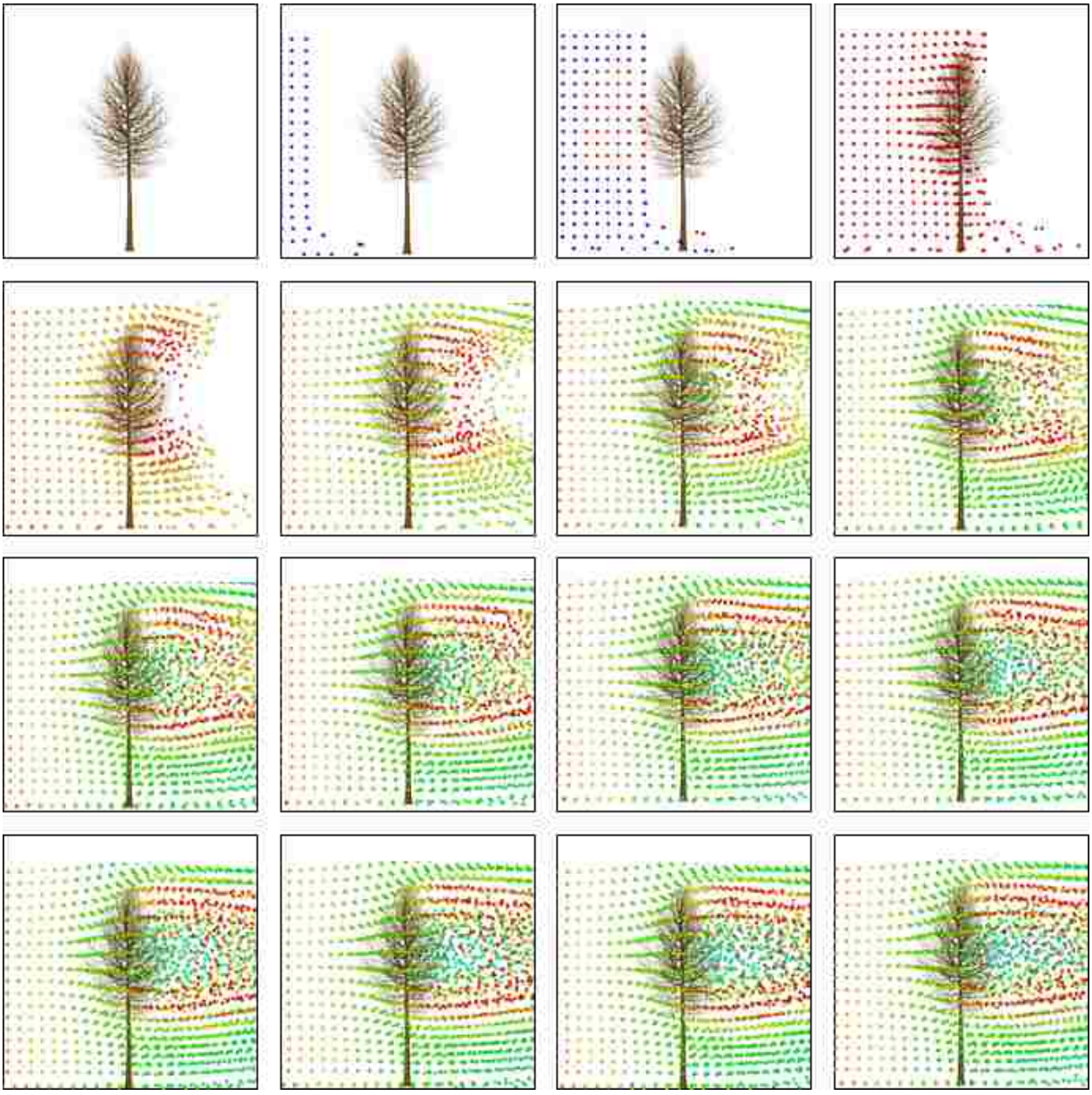

Figure 4.5: **Evolution of aeolian tke in a constant wind (10 m/s)**. The two-way coupling on the tree causes turbulence to be created on the wind particles and advected.

# Chapter 5

## Conclusion and Future Work

We have presented a method for animating highly-deformable objects moving in a turbulent medium such as wind. We have demonstrated that splitting tke into aoelian and arboreal components in a two-way coupled fluid simulation with synthesized subgrid-scale turbulence results in tree motion which exhibits sheltering and branch tip motion paths with tortuosity similar to motion paths captured in 3D motion capture experiments.

This thesis opens the path for new research into two-way coupling between synthesized turbulence in fluid simulation and participating objects. Work in this area may lead to better simulations for forestry micrometeorology as well as special effects for films and games.

Forestry micrometeorology is concerned with the problem of how turbulent flows affect the growth and longevity of trees in nature. This problem is particularly important if the frequency and duration of wind storm events increase due to global climate change. This is a difficult problem because observing turbulence in dense stands of tall trees is difficult. While it is possible to observe turbulence at a few selected points using sonic aneometers mounted on towers, it is not currently feasible to observe turbulence at a resolution of 1 meter or less. It is possible, however, to observe turbulence at that scale using the proposed simulation method. Predictive models of turbulence in tree canopies will need to be validated using data collected in the field.

Another avenue for future work involves the dynamics of tree growth in the presence of wind. Thigmomorphogenesis is the change in shape of a tree due to wind. Thigmomorphogenesis is difficult to study in the field because it requires observing the growth of an

individual tree over a long period of time. Just as simulation could be used to increase the spatial resolution of turbulence observations, simulation could also be used to increase the temporal duration of observations. Such observations would allow forestry ecologists to refine models of thigmomorphogenesis.

Since FLIP is a hybrid method we considered calculating $k$ on the grid and transferring it to and from particles for advection similar to other attributes. This would allow us to evaluate both $u$ and $k$ directly from the grid and runtime performance would be correlated to the size of the grid rather than the number of particles. In practice, though, this hasn't significantly affected our performance because of our acceleration structures and Houdini's adaptive FLIP grid implementation.

That said, our method suffers from a few weaknesses. Our simple tree model was unable to resolve the low-amplitude, high-frequency components of the wind-fluid simulation. This caused long, thin branch segments in highly turbulent flow to twitch back and forth excessively. Though we were able to ameliorate this condition somewhat by removing these components from our velocity texture, we believe the resulting tree motion would be even better if we could include them. We believe this artifact is due to the lack of coupling between branch segments and that—with a more sophisticated model such as Weber [Web08]—these components would propagate down to larger segments where they could be adequately resolved.

The tke term is used to scale the amplitude of turbulent flows but does not correlate the phase of turbulent flows. A more accurate method will include correct estimates of both phase and amplitudes across all turbulent eddies.

Another item we have not addressed is that branches just outside of fluid boundaries receive no wind contribution whatsoever where nearby branches may be in a strong wind. This is due to our atmospheric model which assumes that wherever there is no fluid there is no wind present. We leave this anamoly for future work.

Finally, our approach relies on a fluid simulation can only produce results that are physically-based and is not easily directable. We believe an interesting approach to look into would be adapting curl noise to two-way coupled flow. This could be done by creating and advecting collision information on the fluid particles and using that information—combined with artistic input—to generate a smooth potential field and, consequently, divergence-free flow.

# Appendix A

## Generating a 2D Velocity Texture

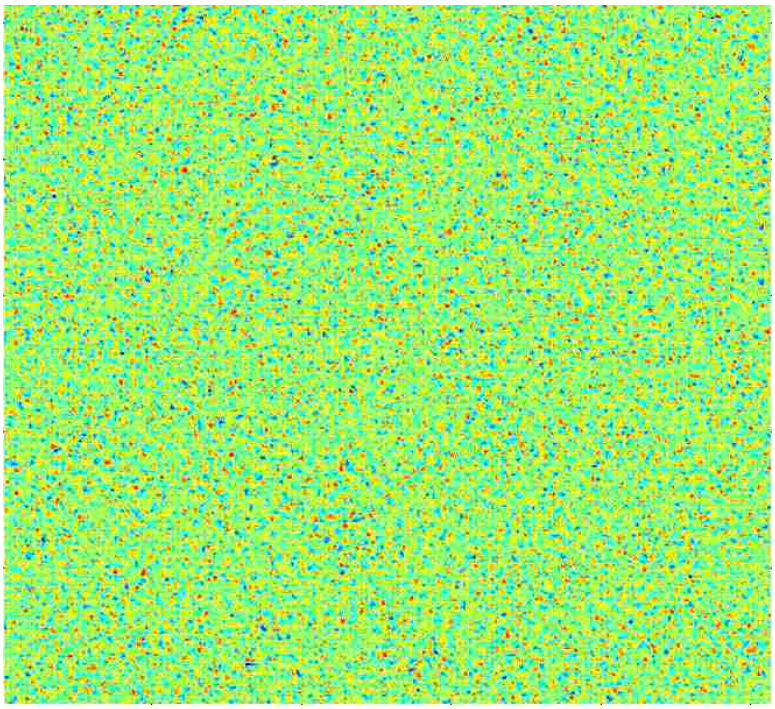

Figure A.1: An 2D velocity texture created in Matlab

The following is our Matlab code for generating a 2D velocity texture similar to Habel [HKW09] and Chuang [CGZ+05]. The basic idea is of these "spectral approaches" is to

1. Create a complex 2D Gaussian Random Field in the Frequency Domain
2. Filter the Random Field
3. Inverse FFT the Field into the Time Domain

[HKW09] adapted this approach to create a 2D texture which allows more variation. We follow Habel's 2D approach only that we do not combine velocity with the Euler-Bernouilli

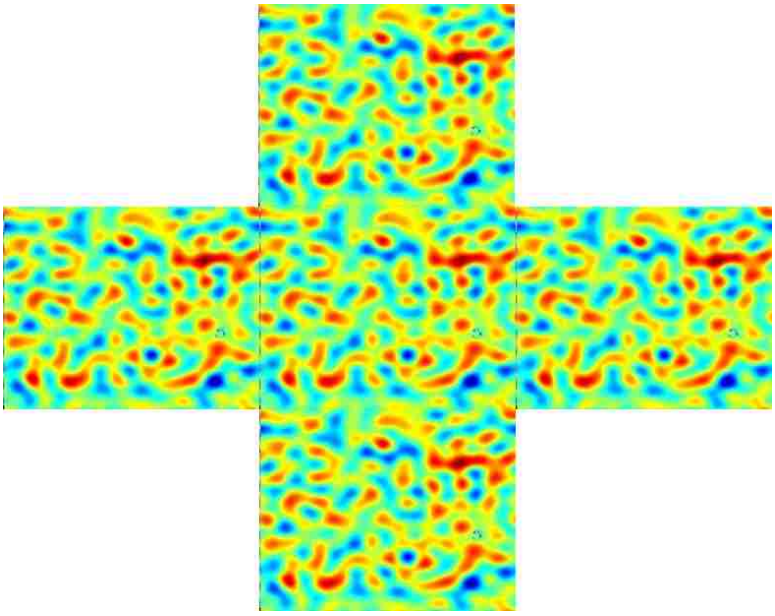

Figure A.2: An even lower-frequency velocity texture. This image demonstrates that the velocity textures we create are tileable.

beam equation in the frequency space. We just use velocity ( similar to Chuang [CGZ+05] ) and inverse FFT that to get a 512x512 velocity texture in the time domain which can be scaled by the amount of tke in our system. At initialization each branch segment is then randomly assigned a location and direction vector. For each frame of animation–even when the tree is not in wind–the branch segments follow their trajectory and wrap around when reaching a border. Because the 2D texture tiles ( see Figure A.2 ) there are no discontinuities in velocity. Since we have a finite number of samples our texture is periodic, but the period is very large and one order removed from motion so it is unlikely that during even a long simulation the user would notice. This is further helped by the fact that our texture is used only to augment low frequency fluid motion. In Figure A.1 the RGB components of the image represent the x,y, and z velocity components of the wind and visualized by calling the mesh function.

```
clear; close all;

FRAMES_PER_SECOND    = 23.9999;                %Sample frequency (Hz) (frames/second)
NYQUIST              = FRAMES_PER_SECOND/2.0;  %Nyquist frequency (frequencies above this are lost)
BETA                 = (5.0/3.0);              %Kolmogorov ``5/3'' spectrum
```

```
m                   = 512;                  %number of samples in each dimension (power of 2 is best)
MIN_FREQ            = 1.6;                  %FLIP takes care of frequencies lower than this
MAX_FREQ            = 7.0;                  %this keeps our tree stable for lack of two-way coupling

% GENERATE FREQUNCIES TO BE SAMPLED (MATCHING COMPLEX CONJUGATE RULES)
finc                = 2 * NYQUIST/m;
nyquistIndex        = m/2 + 1;
Fnormal1D           = [-NYQUIST:finc:NYQUIST-finc];
Fnormal2D           = ones(m,m);
for x = 1:m
    for y = 1:m
        Fnormal2D(x,y) = sqrt(Fnormal1D(x)^2 + Fnormal1D(y)^2);
    end
end
F               = ifftshift(Fnormal2D); %Not Centered

% GENERATE POWER SPECTRUM AND RESULTING NOISE TEXTURE
vm              = 1.0;                  %mean wind velocity
NoiseTexture    = ones(m,m,3);
realField       = 1.0 .* randn(m,m,3);      %normally-dist
complexField    = 1.0 .* randn(m,m,3);      %normally-dist

for dim=1:3
    % CREATE 2-D COMPLEX GAUSSIAN RANDOM FIELD (SPATIALLY CORRELATED RANDOM NUMBERS)
    %(Section 3.1 of Chuang "Animating Picturs with Stochastic Motion Textures")
    complexField(1,1,dim)                       = 0;
    complexField(1,nyquistIndex,dim)            = 0;
    complexField(nyquistIndex,1,dim)            = 0;
    complexField(nyquistIndex,nyquistIndex,dim) = 0;

    % MAKE OUR GAUSSIAN NOISE SYMMETRIC ACROSS ALL AXIS (1 QUADRANT HAS POSITIVE IMAGINARY NUMBERS)
    Gny = ones(m,m);
    for x=1:m
        for y = 1:m
            Gny(x,y) = realField(x,y,dim) + i * complexField(x,y,dim);
        end
    end

    %Mirror on nyquist and 0 axes
    for y=nyquistIndex+1:m %just the negative frequencies
        ydiff = y - nyquistIndex;
```

```
        yprime = nyquistIndex - ydiff;
        Gny(1,y) = conj(Gny(1,yprime));
        Gny(nyquistIndex,y) = conj(Gny(nyquistIndex,yprime));
    end

    for x=nyquistIndex+1:m %just the negative frequencies
        xdiff = x - nyquistIndex;
        xprime = nyquistIndex - xdiff;
        Gny(x,1) = conj(Gny(xprime, 1));
        Gny(x,nyquistIndex) = conj(Gny(xprime, nyquistIndex));
    end

    for x=nyquistIndex+1:m
        for y = 2:m
            if y~=nyquistIndex
                diffy = y - nyquistIndex; %y could be either
                yprime = nyquistIndex - diffy; %33 y - nyquijst
                diffx = x - nyquistIndex; %x is greater than nyquistIndex
                xprime = nyquistIndex - diffx;
                Gny(x,y) = conj(Gny(xprime, yprime));
            end
       end
    end

  % GENERATE POWER SPECTRUM AND RESULTING NOISE
    Pw      = ones(m,m);          %power spectrum of wind
    Noise   = ones(m,m);          %1-D noise texture
    for x=1:m
        for y = 1:m
          f          = F(x,y);
                                          if( f < MIN_FREQ | f > MAX_FREQ)
                Gny(x,y) = 0;
                                          end
          Pw(x,y)      = vm /((1.0 + f/vm)^BETA); %power spectrum of the wind (Simiu)
          Noise(x,y)   = Gny(x,y)* sqrt(Pw(x,y));
        end
    end

  %INVERSE FFT TO TIME DOMAIN
    NoiseTexture(:,:,dim)  = ifft2(Noise);
end
```

```
dlmwrite('treeTexture.txt', real(NoiseTexture), ' ');
```

## References


[AK06] Y. Akagi and K. Kitajima. Computer animation of swaying trees based on physical simulation. *Computers & Graphics*, 30(4):529–539, 2006.

[Cam10] James Cameron. "Avatar", 2010.

[CGZ+05] Yung-Yu Chuang, Dan B Goldman, Ke Colin Zheng, Brian Curless, David H. Salesin, and Richard Szeliski. Animating pictures with stochastic motion textures. In *ACM SIGGRAPH 2005 Papers*, SIGGRAPH '05, pages 853–860, New York, NY, USA, 2005. ACM.

[DGP+10] S. Dupont, F. Gosselin, C. PY, E. De Langre, P. Hemon, and Y. Brunet. Modelling waving crops using large-eddy simulation: comparison with experiments and a linear stability analysis. *Journal of Fluid Mechanics*, 652:5–44, 2010.

[Doc01] Pete Docter. "Monsters, Inc.", 2001.

[DRBR09] Julien Diener, Mathieu Rodriguez, Lionel Baboud, and Lionel Reveret. Wind projection basis for real-time animation of trees. *Computer Graphics Forum*, 28(2):533–540, 2009.

[Had06] Sunil Hadap. Oriented strands: dynamics of stiff multi-body system. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '06, pages 91–100, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.

[HKW09] Ralf Habel, Alexander Kusternig, and Michael Wimmer. Physically guided animation of trees. *Computer Graphics Forum*, pages 523–532, 2009.

[KTJG08] Theodore Kim, Nils Thürey, Doug James, and Markus Gross. Wavelet turbulence for fluid simulation. *ACM Transactions on Graphics*, 27:50:1–50:6, August 2008.

[NSCL08] Rahul Narain, Jason Sewall, Mark Carlson, and Ming C. Lin. Fast animation of turbulence using energy transport and procedural synthesis. In *ACM SIGGRAPH Asia 2008 papers*, SIGGRAPH Asia '08, pages 166:1–166:8, New York, NY, USA, 2008. ACM.

[OFT+03] Shin Ota, Tadahiro Fujimoto, Machiko Tamura, Kazunobu Muraoka, Kunihiko Fujita, and Norishige Chiba. $1/f^{\beta}$ noise-based real-time animation of trees swaying in wind fields. In *Computer Graphics International*, pages 52–59, 2003.

[Pop00] Stphen B. Pope. *Turbulent Flows*. Cornell University Press, New York, 2000.

[PTC+10] Tobias Pfaff, Nils Thuerey, Jonathan Cohen, Sarah Tariq, and Markus Gross. Scalable fluid simulation using anisotropic turbulence particles. *ACM Transactions on Graphics*, 29:174:1–174:8, December 2010.

[RMSG+08] Avi Robinson-Mosher, Tamar Shinar, Jon Gretarsson, Jonathan Su, and Ronald Fedkiw. Two-way coupling of fluids to rigid and deformable solids and shells. *ACM Transactions on Graphics*, 27:46:1–46:9, August 2008.

[SB08] H. Schechter and R. Bridson. Evolving sub-grid turbulence for smoke animation. In *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '08, pages 1–7, Aire-la-Ville, Switzerland, Switzerland, 2008. Eurographics Association.

[SE93] Jos Stam and Fiu Eugene. Turbulent wind fields for gaseous phenomena. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '93, pages 369–376, New York, NY, USA, 1993. ACM.

[SF92] Mikio Shinya and Alain Fournier. Stochastic motion-motion under the influence of wind. *Computer Graphics Forum*, 11(3):119–128, 1992.

[Sid11] Side Effects Software Inc. Houdini 11, 2011. 11.0.581.

[SLS+10] Arthur Shek, Dylan Lacewell, Andrew Selle, Daniel Teece, and Tom Thompson. Art-directing disney's tangled procedural trees. In *ACM SIGGRAPH 2010 Talks*, SIGGRAPH '10, pages 53:1–53:1, New York, NY, USA, 2010. ACM.

[SO99] Tatsumi Sakaguchi and Jun Ohya. Modeling and animation of botanical trees for interactive virtual environments. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*, VRST '99, pages 139–146, New York, NY, USA, 1999. ACM.

[SRF05] Andrew Selle, Nick Rasmussen, and Ronald Fedkiw. A vortex particle method for smoke, water and explosions. *ACM Transactions on Graphics*, 24:910–914, July 2005.

[SS78] Emil Simiu and Robert H Scanlan. *Wind Effects on Structures: An Introduction to Wind Engineering*. New York : Wiley, 1978.

[Sta97] Jos Stam. Stochastic dynamics: Simulating the effects of turbulence on flexible structures. *Computer Graphics Forum*, 16(3):159–164, 1997.

[Web08] Jason P. Weber. Fast simulation of realistic trees. *IEEE Computer Graphics and Applications*, 28:67–75, 2008.

[WP95] Jason Weber and Joseph Penn. Creation and rendering of realistic trees. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '95, pages 119–128, New York, NY, USA, 1995. ACM.

[WZF+03] Xiaoming Wei, Ye Zhao, Zhe Fan, Wei Li, Suzanne Yoakum-Stover, and Arie Kaufman. Blowing in the wind. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '03, pages 75–85, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[ZB05] Yongning Zhu and Robert Bridson. Animating sand as a fluid. *ACM Transactions on Graphics*, 24:965–972, July 2005.

[ZST+06] Long Zhang, Chengfang Song, Qifeng Tan, Wei Chen, and Qunsheng Peng. Quasi-physical simulation of large-scale dynamic forest scenes. In *Computer Graphics International*, pages 735–742, 2006.

[ZYC10] Ye Zhao, Zhi Yuan, and Fan Chen. Enhancing fluid animation with adaptive, controllable and intermittent turbulence. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '10, pages 75–84, Aire-la-Ville, Switzerland, Switzerland, 2010. Eurographics Association.