

A Study on Efficient Algorithms for Generating Virtual Wind Field Usable for Real-time Animation

Khorloo Oyundolgor

A thesis submitted for the degree of Doctor of Philosophy

Department of Design and Media Technology
Graduate School of Engineering
Iwate University

March 2012

Abstract

We all experience the wind from our everyday life. Wind is invisible. It has no shape, size, smell, taste or sound of its own. Thus, it is the most challenging natural phenomenon to predict accurately. In the field of computer animation, wind is an important element for creating visually plausible motion in natural outdoor scenery. Examples of such motions include the swaying of tree branches in the wind, field of grass blowing in the breeze, the swirling of smoke, the waving of a flag, the fluttering of cloth in the wind, etc. Without a turbulent wind field, animations would lack a natural chaotic look, which is an important characteristic of most natural phenomena.

A series of natural wind models have been developed since the 1960s to facilitate the study of wind engineering problems. In this thesis, we present a study on wind turbulence models from the field of engineering and their applicability in computer animations for simulating a variety of motions and behaviors of flexible objects in windy conditions.

At first, we describe some theoretical aspects of wind turbulence based on measurements of actual winds. Then we describe a simple yet visually convincing simulation of wind fields using our proposed wind models generated from actual wind data. The efficiency and practical utility of the wind field model is implemented in a simulation of grass blowing in the wind.

Then, we also present a simple method to generate three-dimensional frozen and non-frozen turbulent wind fields for use in the animation of wind-induced motion. Our approach uses $1/f^\beta$ noise to match the characteristics of natural wind. By employing a noise-based approach, the complexity as well as computational cost is reduced. Additionally, by considering key characteristics of actual wind that are applied in the engineering field, our proposed method is also able to produce believable motion results in outdoor wind field simulations. In this thesis, we describe the implementation results of our

proposed method and compare them with other existing approaches used to construct turbulent wind fields. The implementation and visualization are carried out for both two- and three-dimensional scenarios and compared with the results of other well-known methods.

Finally, we present applications of noise-based non-frozen wind fields for animating flexible structures such as grass blades, cloth like objects, fire, etc.

Acknowledgements

This thesis would not have been possible without the support of many people.

First of all, I would like to thank Prof. Norishige Chiba for introducing me to this wonderful field of Computer Graphics. This dissertation would not been possible without his guidance and support.

Dr. Gunjee Zorig and Dr. Sosorbaram Batjargal deserve special thanks for their support and helpful discussions during the initial phase of the research.

I am deeply indebted to Prof. Yuji Koui, Prof. Tadohiro Fujimoto, Prof. Kouichi Konno, all the past and present members of Chiba and Fujimoto CG Lab for their constructive comments and suggestions. I am also thankful to Mrs. Hikaru Kaketa for assistance with administrative work and to Mr. Norihiro Nasukawa for the computer related technical support.

Finally, I would like to thank my parents Dashzeveg Khorloo and Lhamzav Baigalmaa, my husband Altantsetseg Enkhbayar for their love, encouragement and support.

List of Publications

Refereed Publications

- Oyundolgor Khorloo, Zorig Gunjee, Batjargal Sosorbaram and Norishige Chiba. Wind Field Synthesis for Animating Wind-induced Vibration. *The International Journal of Virtual Reality*, 10(1), pp. 53–60, 2011.
- Oyundolgor Khorloo, Zorig Gunjee, Batjargal Sosorbaram and Norishige Chiba. A Study on Wind Turbulence Models for Animating Flexible Objects. *The Journal of the Society for Art and Science*, 10(3), pp. 118–128, 2011.

Non-refereed Publications

- Khorloo Oyundolgor, Sosorbaram Batjargal, Gunjee Zorig and Chiba Norishige. Fundamental study on wind models by experimental and theoretical approaches. *Information Processing Society of Japan – Tohoku branch*, Session 1, paper No. 2, Iwate University, December 2009.
- Oyundolgor Khorloo, Zorig Gunjee, Batjargal Sosorbaram and Norishige Chiba. Fundamental Study on Wind Models for Animating Flexible Objects. In Proceedings of *The 26th NICOGRAPH Paper Contest*, Session 6, VI-2, Morioka, Japan, September 2010.
- Gunjee Zorig, Khorloo Oyundolgor, Sosorbaram Batjargal and Chiba Norishige. Approximation Algorithms for Generating Wind Field Usable for Scenery Simulation. *Information Processing Society of Japan*

– *Tohoku branch*, Session 6, paper No. 30, Iwate Universtiy, December 2010.

- Oyundolgor Khorloo, Zorig Gunjee, Batjargal Sosorbaram and Norishige Chiba. Wind Field Synthesis for Animating Wind-induced Vibration. *The 24th International Conference on CASA*, China, May 2011.
- Oyundolgor Khorloo and Norishige Chiba. An Application of Noise-based Wind Field Generalization for Animating Fire. *The Society for Art and Science – Tohoku section*, paper No. 23–01–05, Iwate University, November 2011.

Contents

1	Introduction	1
1.1	Motivation	1
1.2	Prior works	2
1.3	Methodology	2
1.4	Objectives	3
1.5	Contributions	4
1.6	Thesis outline	4
2	Noise Functions	6
2.1	Intuitive definition of noise	8
2.2	$1/f^\beta$ noise	8
2.3	Spectral densities for fBm and the spectral exponent β	10
2.4	Generation technique of $1/f^\beta$ noise	10
2.5	Higher dimensional $1/f^\beta$ noise	12
2.6	$1/f^\beta$ noise by example	12
3	The Nature of Wind	16
3.1	What is wind?	16
3.2	The variable nature of wind	16
3.3	Variation of wind speed with height	17
3.4	Turbulence of wind	18
3.5	Wind modeling	19
4	The General Form of Wind Modeling	23
4.1	Background	23
4.2	Measurements of wind speed	24

4.3	Analysis on measured data	26
4.4	General form of wind spectrum	26
4.5	The proposed method	28
4.5.1	Wind simulation	28
4.5.2	Wind field construction	30
4.6	Animation of grass field	34
4.6.1	Grass model	34
4.6.2	Grass animation	35
4.6.3	Performance results	37
5	Noise-based Non-frozen Wind Field Generation	41
5.1	Background	41
5.2	Characteristics of wind	42
5.3	Wind field model	44
5.3.1	Comparison with well-known approaches	44
5.3.2	Important steps in constructing the wind field	45
5.4	Our approach	46
5.4.1	Consideration of power spectral density	47
5.4.2	Physical meanings of wind field model parameters	47
5.4.3	Implementation steps	49
5.5	Results	50
5.5.1	Wind field simulation and its visualization	51
5.5.2	Performance results	51
6	Applications of Noise-based Non-frozen Wind Field	55
6.1	Animation of grass field	55
6.2	Tracing particles	61
6.3	Animation of cloth in the wind	61
6.3.1	Verlet integration	62
6.3.2	Solving constraints by relaxation	62
6.3.3	Cloth simulation	63
6.4	Animation of fire	65
6.4.1	Introduction	65
6.4.2	Modeling and visualizing fire	68
6.4.2.1	A simple model of fire	69
6.4.2.2	Animating fire	71

6.4.3	Rendering fire	73
6.4.4	Implementation and results	74
6.4.5	Conclusion	74
7	Conclusion and Future Work	78
A	A hot wire anemometer (Model: AM-4204)	80
A.1	Features	80
A.2	RS232 PC Serial Interface	81
	Bibliography	87

Chapter 1

Introduction

1.1 Motivation

Wind is one of familiar natural phenomena that we could find easily everywhere. However, we cannot see the wind, nor hold it in our hands. Instead what we can see is the ripple upon the waters, the fallen leaves dancing in a circle, dust swirling across the path, the swaying of tree-branches in the breeze, etc. Thus, in the field of computer animation, wind is an important element for creating believable and visually pleasant motion in natural outdoor scenery. It is challenging however to mathematically model and to predict accurately these matters, since such natural phenomena have irregular shapes and change their shapes quickly. Therefore the problem of simulating such motions and behaviors due to winds is a complex, scientific topic.

Since wind exhibits a high degree of complexity, it is difficult to simulate using only physical models. On the other hand, in the computer animation industry, the ultimate goal for realistic animation is not a completely physics based reality, but rather a plausible reality. The success of simulating wind depends on many factors. An important visual fact of different types of winds is that they appear and behave differently at various scales. For practical reasons, smaller scales are modeled by a stochastic model. The idea of separating a wind phenomenon into a large scale smooth component and a small scale turbulent component has proven to be effective in both animation and rendering.

1.2 Prior works

In computer graphics, we have several research results on modeling winds. Most research efforts are focused on expressing them as natural phenomena [43], [39], [44], [52], [42], [51]. When wind blows, every object in the simulation environment is affected by the wind. Therefore, the natural wind has been expressed as velocity fields affecting on the whole environment. Wejchert et al. [52] introduced an aerodynamics model to simulate the motion of leaves blown by the wind. To depict animated trees and grass, Shinya et al. [39] constructed a complex flow field by modeling the wind field in Fourier space and then converting it to a time-varying force field. Stam et al. [44] decomposed the turbulent wind field into two components. The Kolmogorov spectrum was used to model the small-scale random vector field for the turbulent motion of gas. Ota et al. [32], Hu et al. [18] simulated the motion of branches and leaves swaying in a wind field by using noise functions. In the animations of waving phenomena, Fujimoto et al. [11], Qiang et al. [36] used fractional Brownian motion. Qiang et al. [36] expanded 2D simulation of breaking waves into 3D representation by giving motion variation using fBm.

Computational fluid dynamics methods for solving the Navier-Stoke's equations have led to significant developments in the visual simulation of fluids [43], [10]. Stam [43] proposed an unconditionally stable fluid solver using semi-Lagrangian advection schemes and implicit solving method. Wei et al. [51] presented an approach for modeling wind field affecting on lightweight deformable objects. They modeled the wind field by using the Lattice Boltzmann Model from computational fluid dynamics. In addition, many other papers, such as [46], [5], [15], are concerned with modeling the faithful motions to the influence of external forces such as caused by a wind.

1.3 Methodology

Approaches based on physical methods provide physical accuracy and visually convincing results, but their computational cost is quite high. Methods based on noise functions are simpler than abovementioned methods, but the simulated wind fields are not realistic enough because the key characteristics of

natural wind are not considered. To realistically model natural wind, experiments and statistics can be used from other fields, e.g. structural engineering or wind engineering. This observation forms the basis of our method and we derive a wind field model from the structural engineering. The key component of our model is a stochastic process that faithfully mimics the stochastic properties of the natural wind. This contributes to the visual and physical realism of the objects swaying in the wind. In this thesis, we will address the study on wind models from the field of engineering and indicate their applicability in computer animations for simulating a variety of motions and the behaviors of objects in windy conditions.

1.4 Objectives

The aim of this thesis is the identification of alternative approaches for simulating a wind field and the animation of dynamic natural scenes with reduced computational costs while simultaneously producing a realistic visual simulation. We use a noise-based approach to minimize computation time and stochastic approaches to approximate the complicated motions due to the winds. Fractional Brownian motion (fBm) is one useful technique to represent natural objects and phenomena with randomness. Up to now, fBm, also known as $1/f^\beta$ noise, has been mainly used for representing the shapes of static objects, such as mountainous terrains or clouds of fractal density distribution [25].

In our approach, we implement fBm functions using a simple spectrum synthesis technique based on Fourier transform [14]. It is also possible to use other techniques such as Perlin noise [33] and procedural noise functions [21]. Specifically, in order to create natural motions and behaviors, we use measured data of actual wind and employ theoretical approaches used in the structural engineering. Additionally, in computer animation, it requires to realistically simulating different types of winds such as breezes or gusts since wind can be destructive as easily as it can be soothing.

1.5 Contributions

Our contributions can be summarized in the following points.

- Our method is simple but can provide wind fields of almost same quality as those of well-known approaches.
- Wind fields simulated by our method can be described using relatively few parameters.
- Our wind model can simulate different kinds of wind conditions such as breezes or gusts by controlling parameters in wind models.
- We can simulate winds at different heights or on different terrains by changing the parameter values in wind models
- Our proposed model is applicable to a wide range of natural phenomena such as grass blowing in the wind, animation of fire simulation, animation of cloth in the wind, etc.

1.6 Thesis outline

This thesis is organized as follows. Chapter 2 gives a brief explanation on noise functions and a generation technique about $1/f^\beta$ noise. In this section, we define noise both intuitively and formally. We then examine important issues of modeling details with noise.

Chapter 3 describes natural features of wind. This section introduces the fundamental study of wind, descriptive terms of wind speed, and the relationship between wind speeds and the Beaufort force. Wind profiles parameters such as turbulence intensity, wind shear; and mathematical models of turbulence spectrums will be introduced on a basis of theoretical background.

Chapter 4 presents methods of wind modeling which have been used in the field of engineering. We start with the experimental analysis of measured wind speed data using theoretical approaches from the structural engineering field. In this section, we describe a general form of wind modeling and estimate parameters for our wind turbulence model using selected sample cases. Then, we present results of simulated winds determined with our proposed spectrum

and describe the implementation algorithm for constructing wind fields. At the end, we describe a simulation of grass fields utilizing our wind field method.

Chapter 5 describes an alternative method to construct 3D non-frozen wind fields based on $1/f^\beta$ noise. In order to illustrate the natural characteristics of wind predicted in an intuitive manner, we included some experimental data. We show that $1/f^\beta$ noise based wind field generation method is one useful technique to simulate different types of winds by controlling the parameters in wind field model. We compare our proposed two-dimensional and three-dimensional wind field models to other existing approaches. Implementation and visualization of wind fields will be followed.

Chapter 6 presents applications of noise-based non-frozen wind field generation. In this section, we describe the animation of complex behaviors of cloth-like object under the influence of the wind, animation of a simple grass field, animation of fire, etc.

Chapter 7 draws conclusions on this thesis at the end.

Chapter 2

Noise Functions

Procedural noise functions are widely used in computer graphics, from off-line rendering in the movie production to interactive video games. The ability to add complex and intricate details at low memory and cost is one of its main attractions. Ever since the first image of the marble vase, presented by Perlin [33] (Figures 2.1 and 2.2), Perlin noise has been widespread use in both in research and in industry. Noise has been used for a diverse and extensive range of purposes in procedural texturing, including clouds, waves, tornadoes, rocket trails, heat ripples, incidental motion of animated characters and so on.

Procedural noise has many advantages: it is typically very fast to evaluate, often allowing evaluation of complex and intricate patterns on-the-fly, and it has a very low memory footprint, making it an ideal candidate for compactly generating complex visual detail. In addition, with a suitable set of parameters, procedural noise can be used to easily generate a large number of different patterns. The most recent survey on noise is in the book of [9]. Since then there have been a multitude of recent research results in the domain such as [6], [3], [13], [22], as well as many others. In this section we provide more detail on fractional Brownian motion (fBm), also known as $1/f^\beta$ noise. We start from a high-level review and classification of existing procedural noise functions. Then we examine the important issues of modeling details with $1/f^\beta$ noise.

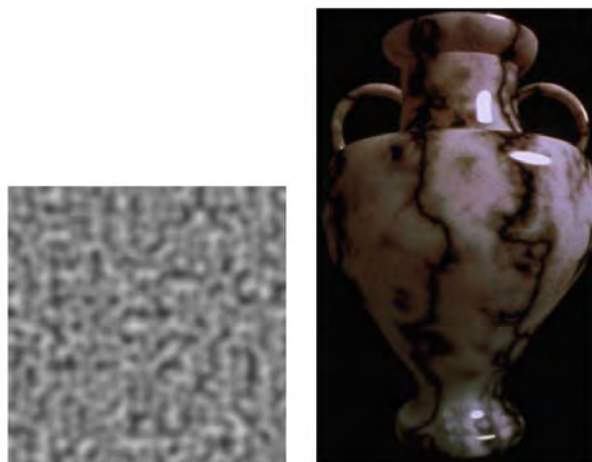


Figure 2.1: Perlin noise. (a) Perlin's noise function, the first procedural noise functions. (Figure from [34]) (b) Perlin's marble vase, one of the first procedural textures created using Perlin noise. (Figure from [33])

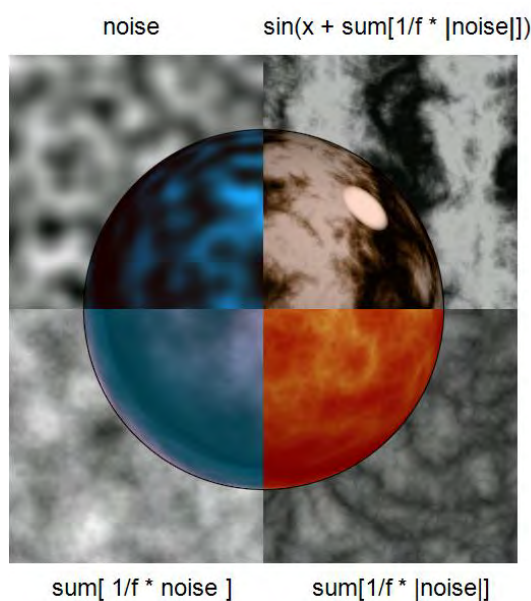


Figure 2.2: Image starting at the top-left (just noise) and going counter-clockwise, ending at the top right marble image. (a) worn metal, water wave; (b) turbulent flows, fire, marble; (c) turbulent flows, fire, marble, clouds; (d) rock, mountains, clouds (Making noise by Ken Perlin)

2.1 Intuitive definition of noise

Noise is the random number generator of computer graphics. It is a random and unstructured pattern, and is useful wherever there is a need for a source of extensive detail that is nevertheless lacking in evident structure. Random patterns are often described in the frequency domain. Whereas in the spatial domain, a signal is determined by specifying the value for every location in space, in the frequency domain, a signal is determined by specifying the amplitude and phase for every frequency. However, for unstructured patterns, the phase is random and does not contribute useful information. Therefore, noise is often described by its power spectrum, which specifies the magnitude (squared) of each frequency and ignores the phase. A high value of a specific frequency in the power spectrum corresponds to a high contribution of the corresponding feature size in the spatial domain. Noise is completely characterized by its power spectrum as explained in Chapter 2.2. Many tasks involving noise can be described as manipulations of the power spectrum of the noise, or spectral control. For example, modeling a noise corresponds to shaping its power spectrum, and filtering a noise corresponds to damping frequencies in the power spectrum that are too high.

White noise contains all frequencies in equal mixture and with random phase, so it provides the raw material to generate unstructured signals with any combination of frequencies. A band-limited power spectrum is non-zero only within a specific range of frequencies. So it can be used as a basis in the frequency domain to shape a desired power spectrum for modeling or filtering.

2.2 $1/f^\beta$ noise

The appearance of power laws in the theory of critical phenomena and all the work of Mandelbrot on fractals in the 1970s seemed to indicate that something deeper was hidden in those ubiquitous spectra.

To the physicist, unpredictable changes of any quantity V varying in time t are known as noise. Graphical samples of typical noises $V(t)$ are shown in Figure 2.3. To the left of each sample is a representation of its spectral densities. The spectral density, $S_V(f)$, gives an estimate of the mean square

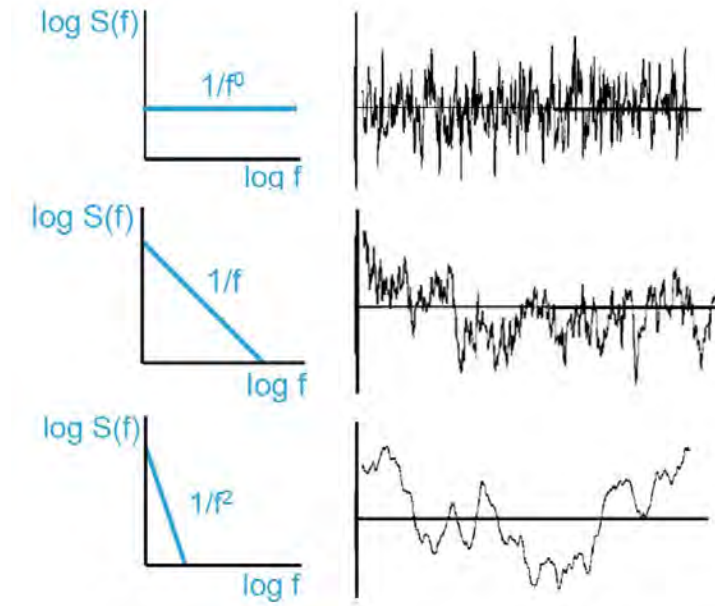


Figure 2.3: Samples of typical noises, $V(t)$, the random variations of a quantity in time. (a) White noise, the most random. (b) $1/f$ -noise, an intermediate but very commonly found type of fluctuation in nature. (c) Brownian motion or a random walk. To the left of each sample is a graphical representation of the spectral density, $S_V(f)$, a measurement characterizing the time correlations in the noise.

fluctuations at frequency f and, consequently, of the variations over a time scale of order $1/f$.

Power laws and $1/f$ spectra were found most unexpectedly in many different phenomena, and Figure 2.4 show how such spectra reproduced in a famous review paper by [1].

The work of Clarke and Voss on $1/f$ noise in resistors also spawned an interesting aside, a study of $1/f$ noise in music, which becomes widely known thanks to an excellent popularization made by Gardner in his Scientific American column [12]. Clarke and Voss found that both voice and music broadcasts have $1/f$ spectra (see Figure 2.5), and even devised an algorithm to compose “fractal” music [49].

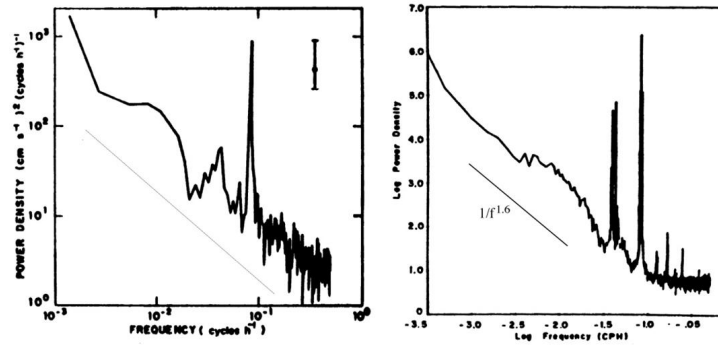


Figure 2.4: (a) Power spectrum of the east-west component of ocean current velocity; the straight line shows the slope of a $1/f$ spectrum. (b) Sea level at Bermuda: this is $1/f^\beta$ spectrum with $\beta = 1.6$. [4]

2.3 Spectral densities for fBm and the spectral exponent β

$1/f^\beta$ noise is a noise of which the spectral density is proportional to $1/f^\beta$. The value of β determines the correlation between noise values varying along the time axis t . As β decreases, the fluctuation of the noise increases. On the other hand, as β increases, the fluctuation decreases. The method proposed in this thesis is based on the fact that $1/f$ noise is observed in natural wind blowing. By then many physicists were convinced that there had to be a deep reason for the ubiquity of this kind of power-law noises, that there had to be something similar to the universality of exponents in critical phenomena, and therefore many people set out to find an all-encompassing explanation. We understand that the behaviors and motions due to the winds can be effectively represented using $1/f^\beta$ noise functions.

2.4 Generation technique of $1/f^\beta$ noise

In the field of computer graphics, in general, we know that “Perlin noise” is a useful technique for generating noise functions [33], [8]. Perlin noise is a general-purpose noise generator, and can be used for generating $1/f^\beta$ noise by giving proper weights to noise factors having different frequencies.

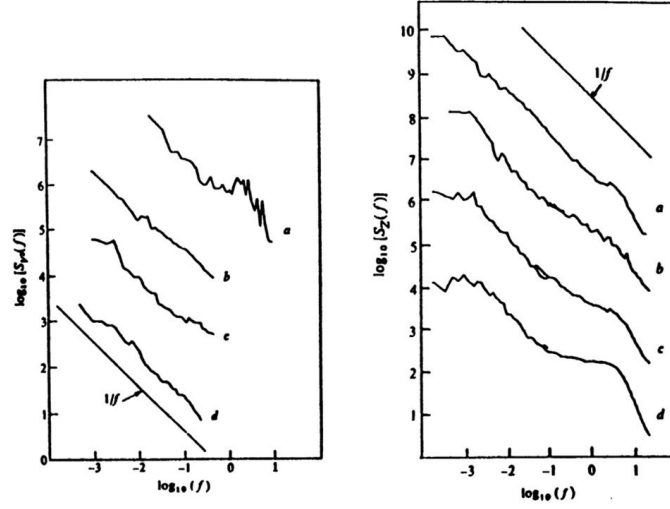


Figure 2.5: Loudness (left) and pitch (right) fluctuation spectra vs. frequency (Hz) (log-log scale), for (a) Scott Joplin piano rags; (b) classical radio station; (c) rock station; (d) news-and-talk station.

Nevertheless, in our method, we generate $1/f^\beta$ noise using “Fourier filtering” [49], since we focus on only $1/f^\beta$ noise, not general noise, and Fourier filtering is a technique that was specially developed for generating $1/f^\beta$ noise easily.

Using Fourier filtering, discrete $1/f^\beta$ noise values are generated. Then, these values are smoothly interpolated to define a continuous $1/f^\beta$ noise function $V(t)$. In this technique, first, the coefficients of the discrete Fourier transform are obtained in the frequency domain so as to meet the condition of $1/f^\beta$ noise by using a Gaussian random number generator. Then, the inverse Fourier transform of the coefficients is calculated to obtain discrete $1/f^\beta$ noise values in the time domain. The discrete noise values are normalized and stored in a one-dimensional array, which is named noise array. A noise value $V(t)$ at arbitrary time t can be obtained by interpolating the discrete noise values in the noise array near the time t . This makes the noise function $V(t)$ continuous along the time axis t .

2.5 Higher dimensional $1/f^\beta$ noise

In this section, we discuss how to generalize the spectral synthesis methods to two, three and higher dimensions.

The generalization of $1/f^\beta$ noise is straight forward. The random field V has stationary increments and is isotropic, i.e. all points (t_1, t_2, \dots, t_n) and all directions are statistically equivalent. In the frequency domain we have for the spectral density

$$S(f_1, \dots, f_n) = \frac{1}{\left(\sqrt{\sum_{i=1}^n f_i^2}\right)^{2H+n}} \quad (2.1)$$

where the parameter H satisfies $0 < H < 1$. The parameter H describes the “roughness” of the noise function at small scales.

This ensures that V restricted to any straight line will be $1/f^\beta$ noise corresponding to $2H = \beta - 1$. The fractal dimension of a sample $V(t_1, t_2, \dots, t_n)$ is

$$D = E + 1 - H = E + \frac{3 - \beta}{2} \quad (2.2)$$

where D spans the range $E < D < E + 1$, and $1 < \beta < 3$. The value $H = 0.8$ is empirically a good choice for many natural phenomena.

2.6 $1/f^\beta$ noise by example

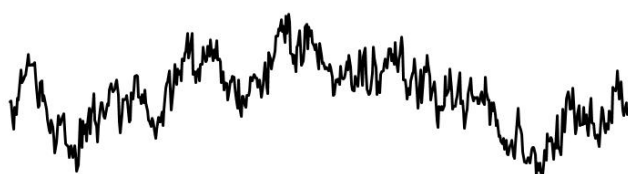
In this section we represent the visual connection between many, seemingly complex, shapes in the natural world and the relation between fractal dimension D , the parameter value of fBm, and the spectral density exponent β . We show the results of spectral synthesis method for producing a finite sample of fBm as a noise ($E = 1$), a landscape ($E = 2$), or a cloud ($E = 3$).

Creating visually rich and interesting content from noise is not an easy task, essentially because the random nature of noise makes it difficult to control and predict the result.

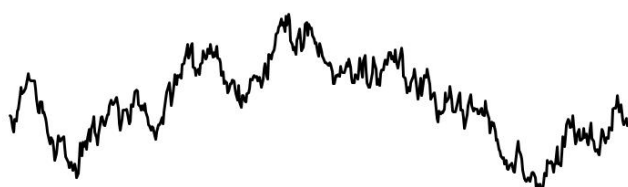
Figures 2.6, 2.7 and 2.8 show samples of fBm noise for different values of H and D .



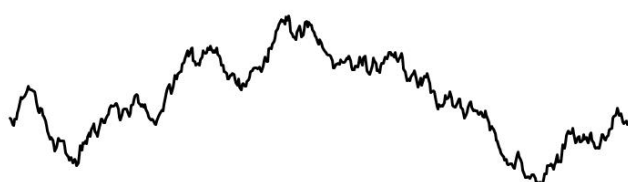
(a) $H = 0.1, \beta = 1.2, D = 1.9$



(b) $H = 0.3, \beta = 1.6, D = 1.7$



(c) $H = 0.5, \beta = 2.0, D = 1.5$

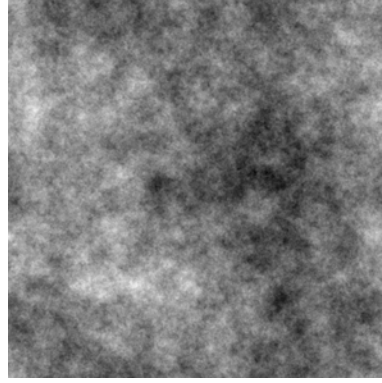


(d) $H = 0.7, \beta = 2.4, D = 1.3$

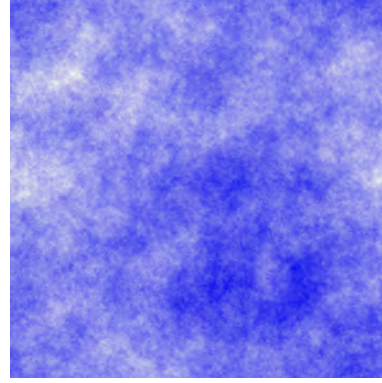


(e) $H = 0.9, \beta = 2.8, D = 1.1$

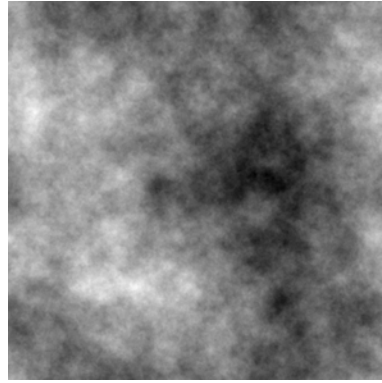
Figure 2.6: $1/f^\beta$ noise via spectral synthesis. The above curves correspond to spectral density functions of the form $1/f^\beta$ where $\beta = 2H + 1$, $D = 2 - H$.



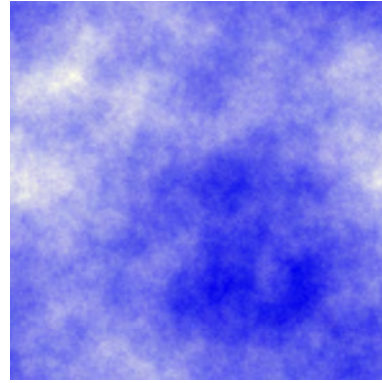
(a) $H = 0.2, D = 2.8$



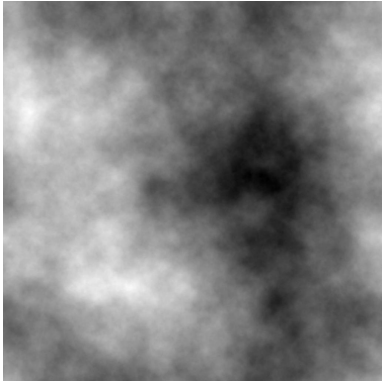
(b) $H = 0.2, D = 2.8$



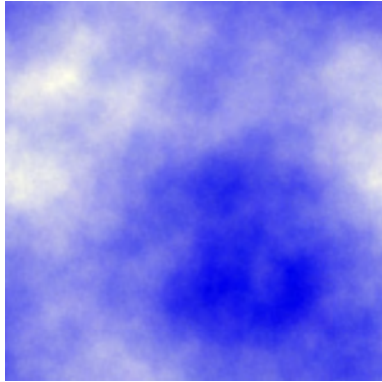
(c) $H = 0.5, D = 2.5$



(d) $H = 0.5, D = 2.5$

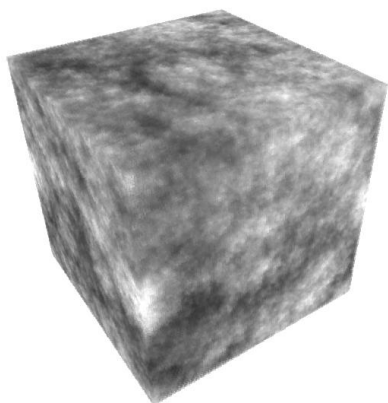


(e) $H = 0.8, D = 2.2$

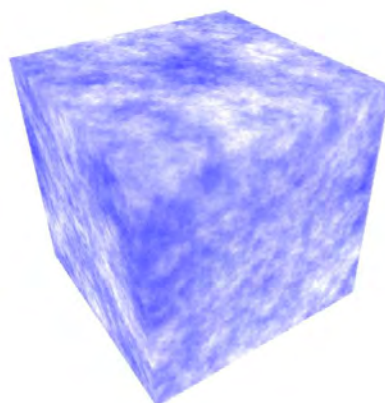


(f) $H = 0.8, D = 2.2$

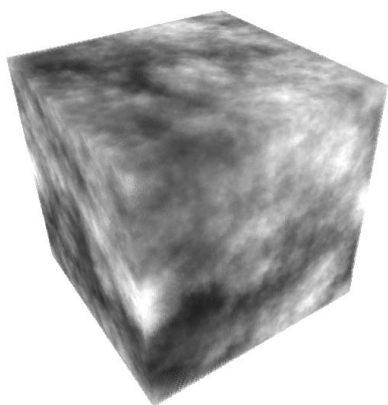
Figure 2.7: Fractal clouds with varying dimensions. The above clouds correspond to spectral density functions of the form $1/f^\beta$ where $\beta = 2H + 2$, $D = 3 - H$.



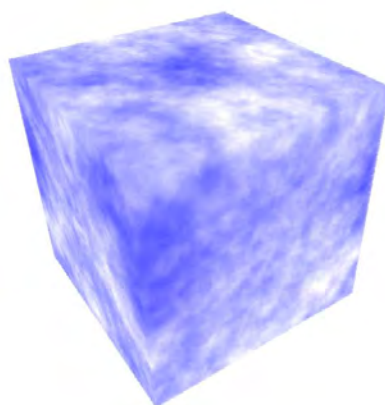
(a) $H = 0.2, D = 3.8$



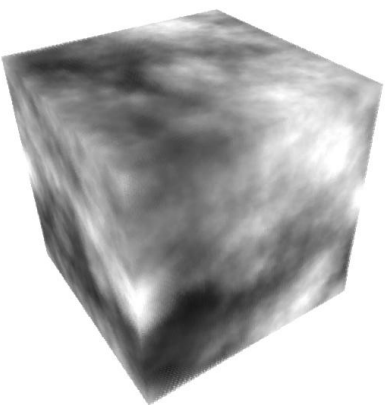
(b) $H = 0.2, D = 3.8$



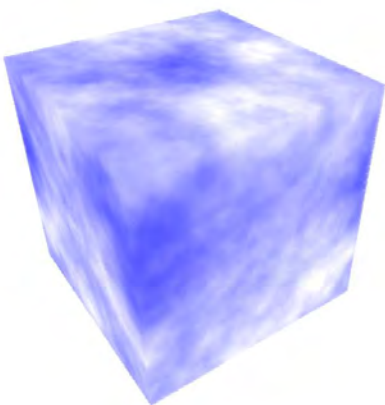
(c) $H = 0.5, D = 3.5$



(d) $H = 0.5, D = 3.5$



(e) $H = 0.8, D = 3.2$



(f) $H = 0.8, D = 3.2$

Figure 2.8: Fractal clouds with varying dimensions. The above clouds correspond to spectral density functions of the form $1/f^\beta$ where $\beta = 2H + 3$, $D = 4 - H$.

Chapter 3

The Nature of Wind

3.1 What is wind?

We feel winds because the air around us moves. Wind is movement of air. Wind occurs because of horizontal and vertical differences in atmospheric pressure. We describe wind by its direction and speed. It is usually expressed in terms of the point of compass. Air moving from east to west is called easterly wind. Wind speed is the speed of the air flow. It is usually expressed in kilometers per hour or as a force on the Beaufort scale. The scale was introduced by Sir Francis Beaufort of the British navy in the early 19th century. The Beaufort scale is used to measure and describe the effects of different wind velocities on objects on land or at sea. It is divided into 13 levels, from the calm wind of force 0 to hurricane wind of force 12. Table 3.1 illustrates the relationship between descriptive terms of wind speeds, the Beaufort force and wind speeds.

3.2 The variable nature of wind

The wind speed at a given location is continuously varying. There are changes in the annual mean wind speed from year to year, changes with season (*seasonal*), with passing weather system (*synoptic*), on a daily basis (*diurnal*) and from second to second (*turbulence*).

3.3 Variation of wind speed with height

Winds occur just above the surface or at high altitudes. The strongest, steadyest and most persistent winds occur in bands some 10 km above the earth's surface. The region below about 1 – 2 km, the wind is strongly affected by the surface, through friction, and hence wind speeds are lower.

In general, the wind speed is nominally zero at ground level and increases steadily with height. The change of wind speed with height is known as the wind shear.

The variation of wind speed with height can be represented mathematically by a logarithmic law.

$$U(z) = K \ln\left(\frac{z}{z_0}\right) \quad (3.1)$$

where $U(z)$ is the mean wind speed at height z , K is a factor which depends on the overall wind speed, and z_0 is known as the surface roughness length.

The surface roughness length characterizes the terrain and has a given value depending on the friction of the surface over which the wind passes. Typical values of z_0 are given in Table 3.2.

The mean wind speed profile for the atmospheric boundary layer is described using a power law [41], [17]:

$$U(z) = U(z_{ref}) \left(\frac{z}{z_{ref}}\right)^\alpha \quad (3.2)$$

where $U(z)$ is the mean wind speed at height z above the ground, z_{ref} is the reference height (normally taken to be 10 m), and α is the power law exponent.

An alternative description of the mean wind speed uses the logarithmic law:

$$U(z) = \frac{1}{k} u_* \ln\left(\frac{z}{z_0}\right) \quad (3.3)$$

where u_* is the friction velocity, k is von Karman's constant ($= 0.4$), and z_0 is the roughness length. The determination of u_* from Equation 3.3 requires $U(z)$, z , and z_0 . The typical values for α and z_0 are listed in Table 3.2.

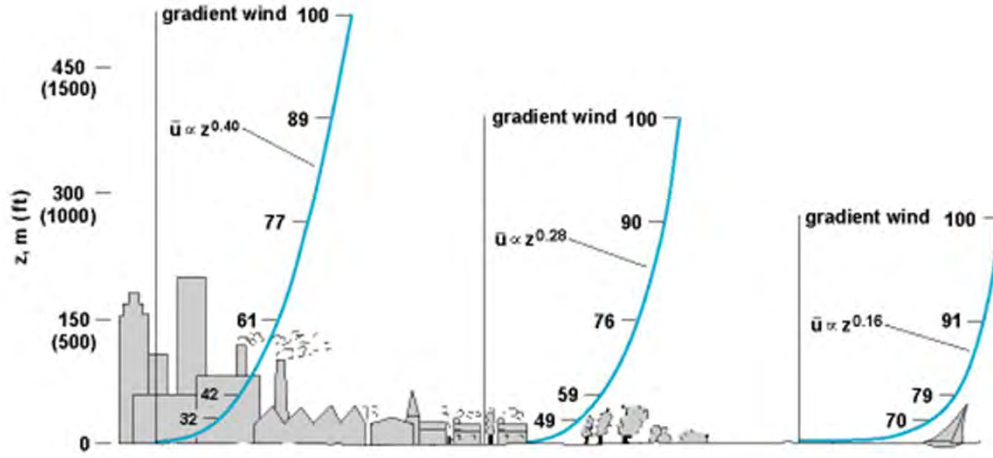


Figure 3.1: Atmospheric boundary-layer profiles (plots of average wind speed U versus height z) over different terrains. Wind speeds are expressed as percentages of the upper level wind (referred to as the gradient wind) above the boundary (or surface) layer.

Wind blowing over the ground surface is slowed when it encounters the friction of uneven ground, trees, and buildings. This roughness affects the thickness of the boundary layer and the power law exponent. The thickness of the boundary layer and the power law exponent both increase as a function of the surface roughness. Thus, the velocity at any height decreases as the surface roughness increases (See Figure 3.1). If the speed of wind for a particular terrain is known, Equation 3.2 and Table 3.2 can be used to calculate the wind speed at some other terrain.

3.4 Turbulence of wind

Short term variations in wind speed are also important in the design and evaluation of winds. For simplicity, we can regard the instantaneous wind speed $U(t)$ as quasi-steady component U and a turbulent fluctuations $u(t)$ about this mean wind value.

$$U(z, t) = U(z) + u(z, t) \quad (3.4)$$

The quasi-steady state value is given by averaging over an appropriate period of time. Over this same period we characterize the variability of the wind by the variance of $U(t)$, which is equal to the mean square value of $u(t)$ in Equation 3.4.

An indication of the gustiness at a given site is provided by a parameter, known as the turbulence intensity, denoted I . This is defined as

$$I(z) = \frac{\sigma(z)}{U(z)} \quad (3.5)$$

An overall measure of the intensity of turbulence is provided by the root mean square value. Thus, the longitudinal component of the turbulence is

$$\sigma(z) = \left[\frac{1}{T_0} \int_0^{T_0} u^2(z, t) dt \right]^{\frac{1}{2}} \quad (3.6)$$

where T_0 is the averaging period.

3.5 Wind modeling

There are many mathematical models of wind turbulence spectrum ([19], [16], [7], [20], [40]). Wind power spectrums are expressed in a non-dimensional form, so we computed normalized power spectral density $nS(n)/u_*^2$ and normalized frequency $nz/U(z)$.

In this section, we present a comparative study of these spectral densities. The formulae for the spectral densities are summarized as follows.

- Kolmogorov [41]:

$$\frac{nS_u(n)}{u_*^2} = 0.26f^{-2/3} \quad (3.7)$$

where n = the frequency in hertz,

u_* = the friction velocity,

$f = nz/U(z)$ is the reduced frequency,

$U(z)$ = the mean wind speed at height z .

- Davenport [7]:

$$\frac{nS_u(n)}{u_*^2} = \frac{4.0x^2}{(1+x^2)^{4/3}} \quad (3.8)$$

where $x = 1200n/U(z)$

- Kaimal [20] and Simiu [40]:

$$\frac{nS_u(n)}{u_*^2} = \frac{200f}{(1 + 50f)^{5/3}} \quad (3.9)$$

where $f = nz/U(z)$

The turbulence spectral density functions formulated to date have specific advantages and drawbacks. For example, Davenport's spectrum has no relationship with height above the ground, whereas the spectrum proposed by Kaimal and Simiu depends on the altitude and it can be applied to both low-frequency and high-frequency spectrum areas.

Table 3.1: Beaufort wind force scale.









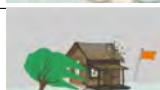

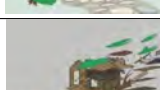
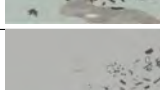
Beaufort Force	Description	Wind speed (km/h)	Illustration
Force 0	Calm	< 2	
Force 1	Light	$2 - 6$	
Force 2		$7 - 12$	
Force 3	Moderate	$13 - 19$	
Force 4		$20 - 30$	
Force 5	Fresh	$31 - 40$	
Force 6	Strong	$41 - 51$	
Force 7		$52 - 62$	
Force 8	Gale	$63 - 75$	
Force 9		$76 - 87$	
Force 10	Storm	$88 - 103$	
Force 11		$104 - 117$	
Force 12	Hurricane	≥ 118	

Table 3.2: Typical values of surface roughness length z_0 and power law exponent α , for various types of terrain.

Type of terrain	z_0	α
Mud flats, ice	10^{-5} to 3×10^{-5}	
Calm sea	2×10^{-4} to 3×10^{-4}	
Sand	2×10^{-4} to 10^{-3}	0.01
Mown grass	0.001 to 0.01	
Low grass	0.01 to 0.04	0.13
Fallow field	0.02 to 0.03	
High grass	0.04 to 0.1	0.19
Forest and woodland	0.1 to 1	
Built up area, suburb	1 to 2	0.32
City	1 to 4	

Chapter 4

The General Form of Wind Modeling

4.1 Background

Wind is a very complex phenomenon because many flow situations can arise from the interaction between wind and objects. Thus, it is challenging to mathematically model wind. However, wind can be considered to possess stationary characteristics that can be described using statistical terms. Computer graphics has produced several wind models for simulating the motion of natural objects in wind. Most studies [39], [44], [32], [18] typically make use of stochastic approximations in wind modeling and consider simulations only in the frequency domain. To realistically model wind, experimental data and statistics can be used from other fields, e.g., structural engineering and wind engineering.

One purpose of this thesis was to construct a wind model using our data measurements of actual wind and employing theoretical approaches used in structural engineering. Structural engineering involves the study of strong winds that might cause damage and destruction of structures. In computer graphics applications, we need to realistically simulate different kinds of wind conditions such as breezes or gusts by controlling specific parameters in wind models. We need also to describe a simple approach for constructing wind

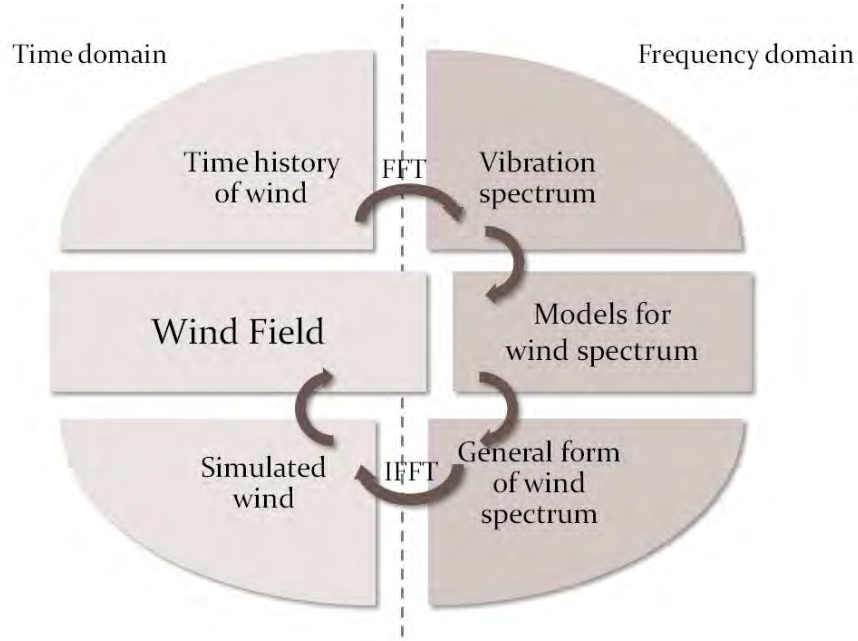


Figure 4.1: General flow diagram of our approach

fields with realistic turbulent winds that share similar characteristics with actual measured winds.

Figure 4.1 shows a general flow diagram of our proposed approach. This section includes time history data collected for the mean wind speed in the Morioka area of Japan. As shown in Figure 4.1, we formulate a suitable wind turbulence spectrum in the frequency domain using theoretical approaches for wind engineering. Wind velocity time series were generated using the Fourier transform approach in conjunction with the proposed wind spectrum. We were able to generate winds with the requisite characteristics for use in time-domain response prediction of flexible objects, by employing the best-fitted formulation for the wind power spectrum.

4.2 Measurements of wind speed

Mean wind speed time series were obtained for the Morioka area of Japan and used as an experimental basis for this fundamental study on wind modeling.

The wind speed data were collected with a hot-wire anemometer (see references in Appendix A1, “Lutron AM-4207”) at a rate of 1 Hz. We selected

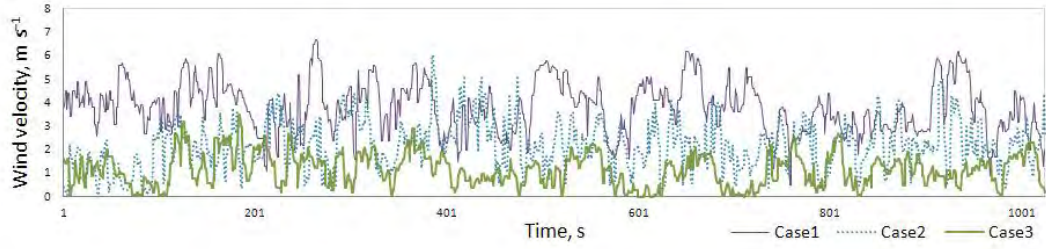


Figure 4.2: Time histories for sample cases

three sets of data from the numerous wind samples based on the criteria that the sample cases can depict different wind characteristics in computer animations.

Each dataset had a sampling interval of 1 s and contained 1024 data points. The first case (Case 1) was measured in the suburbs at the top of a small hill, whereas other two cases (Case 2 and Case 3) were recorded around Morioka City, Iwate Prefecture. Time histories for the selected three cases are shown in Figure 4.2. All three cases were measured at a height of 3 m above the ground. Cases 2 and 3 shared the same terrain roughness with tall buildings and trees.

The statistical measurements and estimates of the wind profile parameters for the selected cases are summarized in Table 4.1

Table 4.1: Wind data measurements for Morioka area of Japan

	Date/Time	Terrain	z	z_0	$U(z)$	$\hat{U}(z)$	σ_u	u_*	$I_u(z)$
			m	m	ms^{-1}	ms^{-1}	ms^{-1}	ms^{-1}	
Case 1	2009/07/06	Suburban	3	0.3	3.81	6.7	1.1	0.66	0.28
Case 2	2010/03/13	City	3	0.7	2.2	6.0	1.03	0.6	0.46
Case 3	2009/12/13	City	3	0.7	1.16	3.5	0.68	0.31	0.79

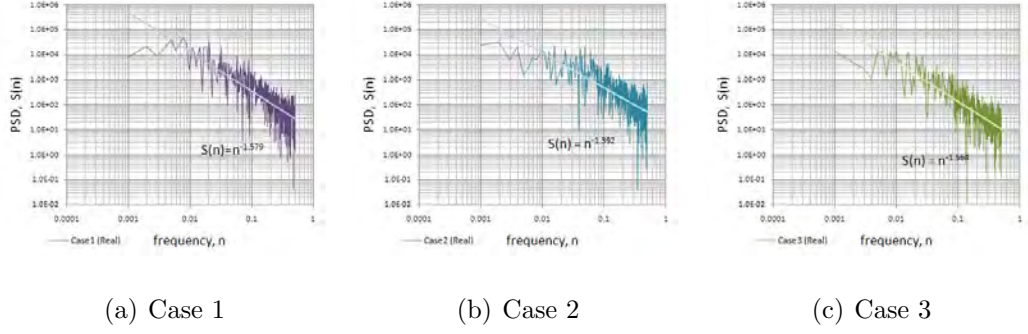


Figure 4.3: Power density spectra for sample cases

4.3 Analysis on measured data

For each group, Table 4.1 lists the following: the date and time, terrain in each case, measured height above the ground, roughness length, mean wind speed U , maximum wind speed \hat{U} , dispersion σ_u , friction velocity u_* , and wind turbulence intensity I .

The power spectral density (PSD) function, also referred to as a power spectrum, is another important function for describing the characteristics of wind. A spectral density function is denoted by $S_u(n)$, where the variable n is the frequency. In each case, we computed the square of the amplitude, $A^2(n)$, using the fast Fourier transform (FFT). The power spectral density of the wind velocity is calculated as $S(n) = A^2(n)/\Delta n$, where Δn is the sampling frequency f_s multiplied by the number of data points N used in the FFT.

Figure 4.3 shows the power spectra using the log-log method for our sample cases. As shown in Figure 4.3, the logarithmic slopes of the spectra are different, depending on the characteristics of the winds and terrains. The relationship between $S_u(n)$ and n appears as a straight line in this graph. In Case 1, $S_u(n)$ is proportional to $n^{-1.579}$ in the inertial subrange portion of the spectrum, whereas $S_u(n)$ is proportional to $n^{-1.392}$ in Case 2.

4.4 General form of wind spectrum

In order to develop a more suitable wind power spectrum for our measured data, we consider a more general form of spectrum model suggested by [30].

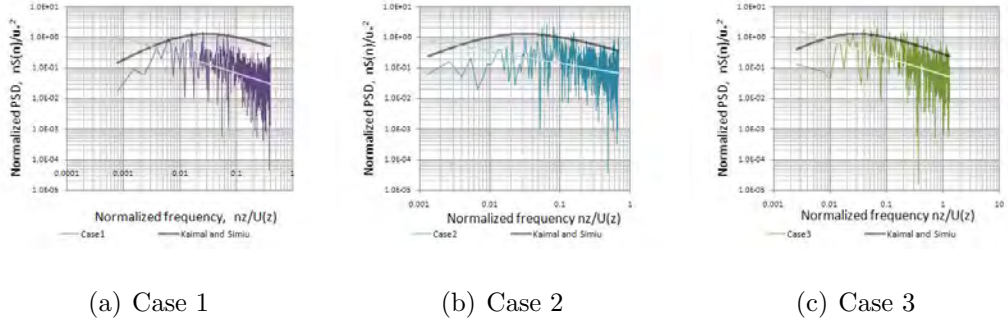


Figure 4.4: Normalized power spectra for sample cases compared with spectra of Kaimal and Simiu

$$\frac{nS_u(n)}{u_*^2} = \frac{Af^\gamma}{(1 + Bf^\alpha)^\beta} \quad (4.1)$$

To determine the unknown coefficients A , B , α , β , and γ , we applied the following four criteria from [30] to the model:

$$\gamma - \alpha\beta = -2/3$$

$$f_m = \left(\frac{1.5\gamma}{B} \right)^{1/\alpha}$$

$$G_m = A \left(\frac{1.5\gamma}{B} \right)^{\gamma/\alpha} (1 + 1.5\gamma)^{-\beta}$$

$$\frac{\sigma^2}{u_*^2} = \frac{1.5A}{B}$$

where f_m is the observed position of maximum and G_m is the observed value of the maximum.

The first criterion determines the slope of a high-frequency asymptote, according to Kolmogorov's law. The second and third criteria provide observed positions and observed values where the model maxima coincide. The fourth criterion is called the integral criterion. A and B determine the spectral frequency range, whereas α , β , and γ determine the spectral shape.

Using these criteria, a total of five unknowns are reduced to two unknowns. The remaining two unknowns are approximated using the Gauss-Newton iterative algorithm until the best agreement between the original and proposed

spectra is reached. The estimated values of parameters for our data cases are listed in Table 4.2.

Table 4.2: Estimated values of parameters.

	A	B	α	β	γ
Case1	83762	20941	1.78	1.57	2.1
Case 2	8283	2071	1.70	1.62	2.1
Case 3	5314	1329	1.77	1.56	2.1

Figure 4.5 shows the proposed spectrum estimated using the corresponding parameter values listed in Table 4.2. As shown in Figure 4.5, the proposed spectrum of wind turbulence for Case 1 has a peak value at a very low frequency of around 0.006 Hz, whereas the spectrum suggested by Kaimal and Simiu has a peak value at 0.02 Hz.

4.5 The proposed method

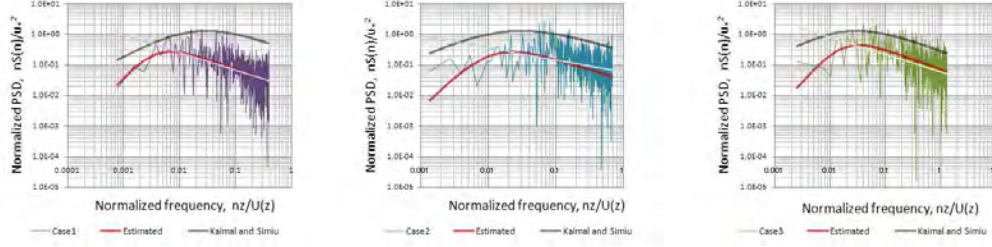
4.5.1 Wind simulation

The time series simulation of wind velocity fluctuations produced in our study uses the best-fitted wind power spectrum extracted from our measured data. Any random signal with varying frequency can be represented using the FFT method [25], [49]. The Fourier coefficients associated with this FFT are obtained using Gaussian random numbers, with zero mean and a specific standard deviation. We set the standard deviation of randomly generated numbers to the same as those found in the measured wind velocity data.

In the previous section, we formulated a wind power spectrum for a general case and estimated turbulence spectrum parameters. Hence, our generated wind velocity time series will be characterized by their spectral densities as

$$S_u(n) = \frac{u_*^2 A f^\gamma}{n (1 + B f^\alpha)^\beta} \quad (4.2)$$

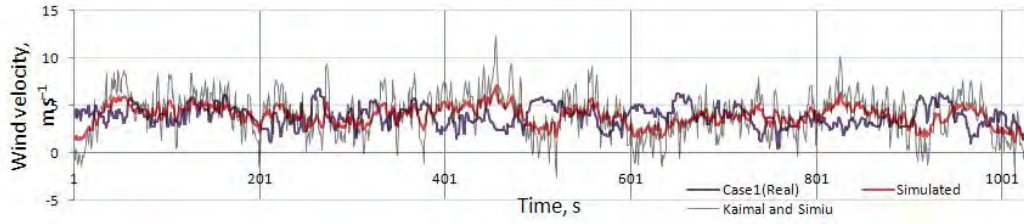
where $f = nz/U(z)$.



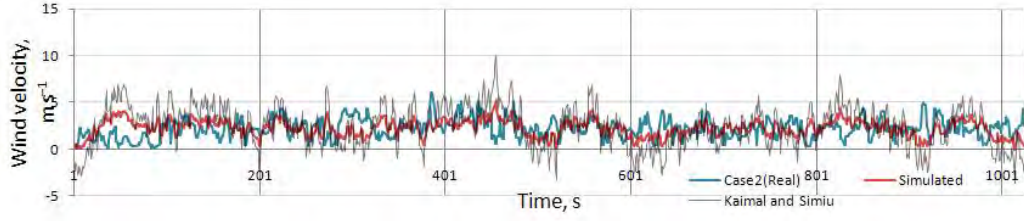
(a) Case 1

(b) Case 2

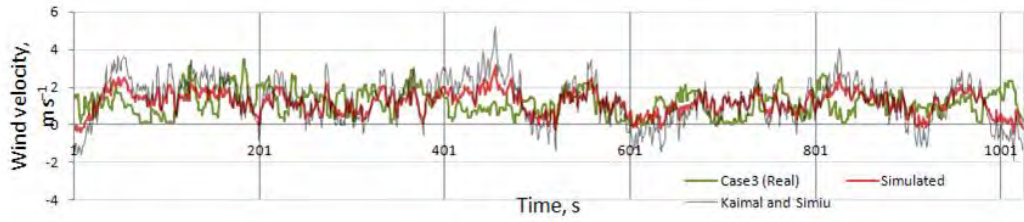
(c) Case 3



(d) Simulated winds for Case 1



(e) Simulated winds for Case 2



(f) Simulated winds for Case 3

Figure 4.5: (Left) Proposed spectrum with spectrum of Kaimal and Simiu and the original wind spectrum for each sample case; (Right) Simulated winds for each sample case

We added the steady mean wind component (slowly varying component) after calculating the longitudinal wind turbulence from the mean wind speed.

Figure 4.5 shows the simulated wind time series for the measured wind time histories. Figure 4.5 shows that the simulated wind constructed with the spectrum of Kaimal and Simiu ([20], [40]) has a larger variation from the measured data and our estimated data. The wind velocity simulation generated using our proposed estimation had similar wind characteristics to that found in the measured wind velocity data.

A consideration of wind profile parameters such as measured height above the ground, mean wind speed, roughness length, and frictional velocity in the estimation of wind turbulence spectrum provides a better approximation in the simulation of wind.

4.5.2 Wind field construction

Wind engineering commonly uses algorithms to simulate turbulent windy fields, which are based on a model of the spectra for atmospheric surface-layer turbulence during high winds. Generally, it is possible to simulate two- or three-dimensional fields using one, two, or three components by employing empirical forms of one-point spectra. The data analyzed for this part of the study consisted of velocity measures at a single point with a turbulent flow fluctuation.

We ignored spatial correlations between points in order to simplify the simulation of wind velocity fields. Overall, two-dimensional velocity fields were simplified into many one-dimensional velocity fields. Thus, our approach for constructing wind fields considered wind-speed time series at several points in a plane perpendicular to the mean wind direction, which propagated the time series in the mean wind direction at the mean wind speed (using Taylor's frozen turbulence hypothesis). For the sake of simplicity, we construct two-dimensional wind fields containing two components of the wind velocity fluctuations. For the longitudinal component of the wind turbulence, we applied the estimated wind power spectrum because it fits well with our measured data, as shown in the previous section.

Technical constraints meant that the spectrum of the horizontal-lateral component of wind turbulence, $v(t)$, used Equation 4.3, as suggested by [20] and slightly modified by [41].

$$\frac{nS_v(n)}{u_*^2} = \frac{15f}{(1 + 9.5f)^{5/3}} \quad (4.3)$$

where $f = nz/U(z)$.

Generally, any given wind field can be described using the long-term change and short-term change in the velocity field. The long-term change in velocity is determined by steady mean flow and long-term fluctuation of velocity. The short-term change in velocity is characterized using the turbulence spectrum of the wind. Thus, we investigated steady mean flow based on the short-term turbulence.

The long-term history of wind fluctuations can be set by the animators in a practical animation. In this study, we used a low-pass filter of the observed spectrum from our measured data to determine the long-term history of the wind. Short-term turbulence of the horizontal longitudinal component near the surface was computed using Equation 4.2 with the estimated parameters listed in Table 4.2.

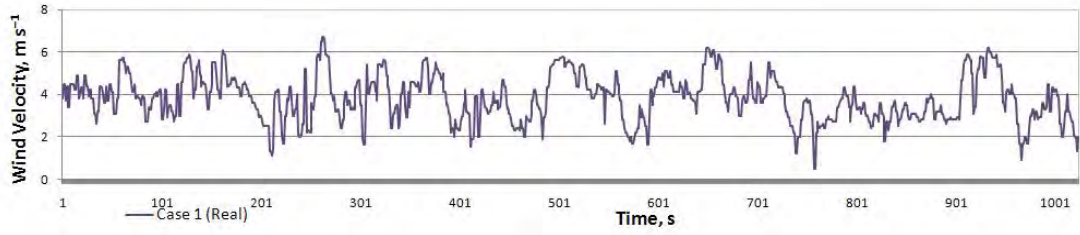
The implementation steps for the wind field simulation are described as follows.

- **Step 1:** Estimate the longitudinal velocity spectra from the measured data at a single point.
 - Initialize 1D data array using the measured wind speed data, with a mean of 0.
 - Evaluate PSD, $S_u(n)$ in the spectral domain by applying 1-D Fourier transform.
 - Calculate friction velocity u_* using Equation 3.3, with a given mean wind speed, roughness, and height measured above the ground.
 - Evaluate the normalized PSD, $nS_u(n)/u_*^2$.
 - Estimate parameters A , B , α , β , and γ in Equation 4.1, for the given criteria.
- **Step 2:** Evaluate the wind turbulence spectrum for each wind velocity component. For the 2D wind velocity field:

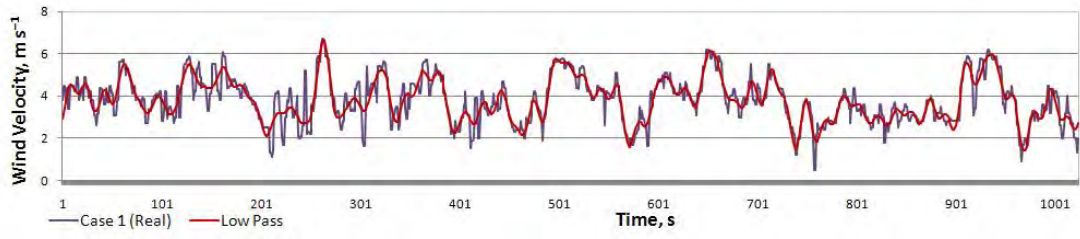
- Initialize the 2D array $S_u[M][N]$ with a longitudinal velocity spectrum given by Equation 4.2, to simulate a wind time series of length N at M different locations.
- Initialize the 2D array $S_v[M][N]$ with a horizontal lateral velocity spectrum given by Equation 4.3.
- Multiply random phase $\exp(\varphi)$ by both $S_u[M][N]$ and $S_v[M][N]$, where φ is a uniform random number from $-\pi$ to π .
- **Step 3:** Determine the long-term history of the fluctuation.
 - Apply the measured wind spectrum calculated in Step 1.3, to determine the long-term fluctuation of the wind history.
 - Replace the longitudinal velocity spectrum, S_u , with the measured wind spectrum for very low frequencies.
- **Step 4:** Apply 1-D inverse Fourier transform M times for each wind component:
 - $\text{IFFT1D}(S_u)$
 - $\text{IFFT1D}(S_v)$

Once we determine the long-term history from our measured data, our simulated field is strongly correlated in the low-frequency range and there are random fluctuations in the higher frequency range. The variation of the fluctuations will be very similar to the measured wind velocity. Simulated winds at two different locations are shown in Figure 4.6. We generated a correlated time series with a length of $N = 1024$ at two different locations in the simulation of wind fields for sample Case 1. The simulated wind field is shown in Figure 4.7. We assumed that the selected number of positions were perpendicular to the direction of the mean wind.

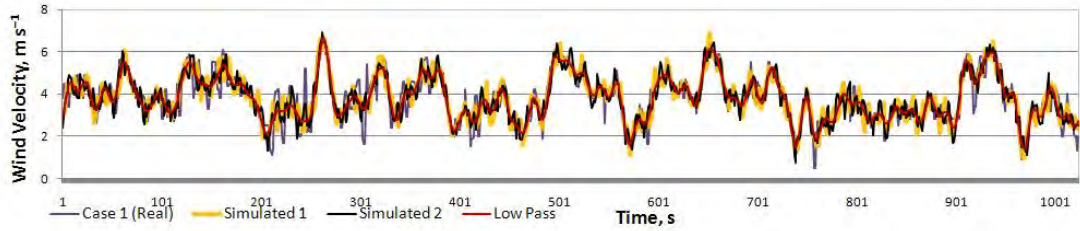
The time consumption for the overall steps is proportional to the total number of locations M in the simulation. The time required to perform the FFT is $O(N \log_2 N)$.



(a) Actual wind



(b) After low-pass filtering



(c) Simulated winds at two locations

Figure 4.6: Steps in wind field simulation: (Top) actual wind history, (Middle) result of low-pass filtering, and (Bottom) simulated winds at two different locations

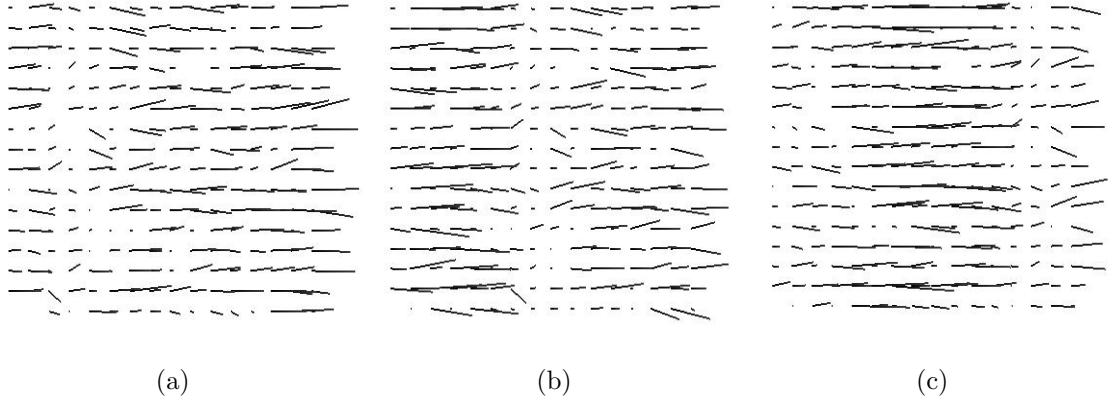


Figure 4.7: Simulated wind fields for sample Case 1 when $M = 16$ and $N = 16$ (Turbulence travels in the mean wind direction).

4.6 Animation of grass field

To simulate the effects of wind turbulence on flexible objects, we considered a field of grass blowing in the wind. Our purpose was to simulate grass that waves realistically in the wind.

4.6.1 Grass model

In this section, we describe a simple yet visually convincing grass simulation based on our wind field model described in the previous section. We considered detailed modeling of individual blades of grass in this study. For large meadows, this will of course require large number of polygons.

Here, we apply a simple but useful solution that meets the condition of representing the deformation caused by the influence of wind velocity. In our grass field simulation, grass blades were built using the Bantam3D Grass application [31]. This application easily creates ground cover and stores the data in an object file format.

A standard grass blade is copied many times to create the ground cover. We made the copies look slightly different from each other, in order to simulate nature. A grass blade is defined by

- the curve of the grass blade
- the size and rotation

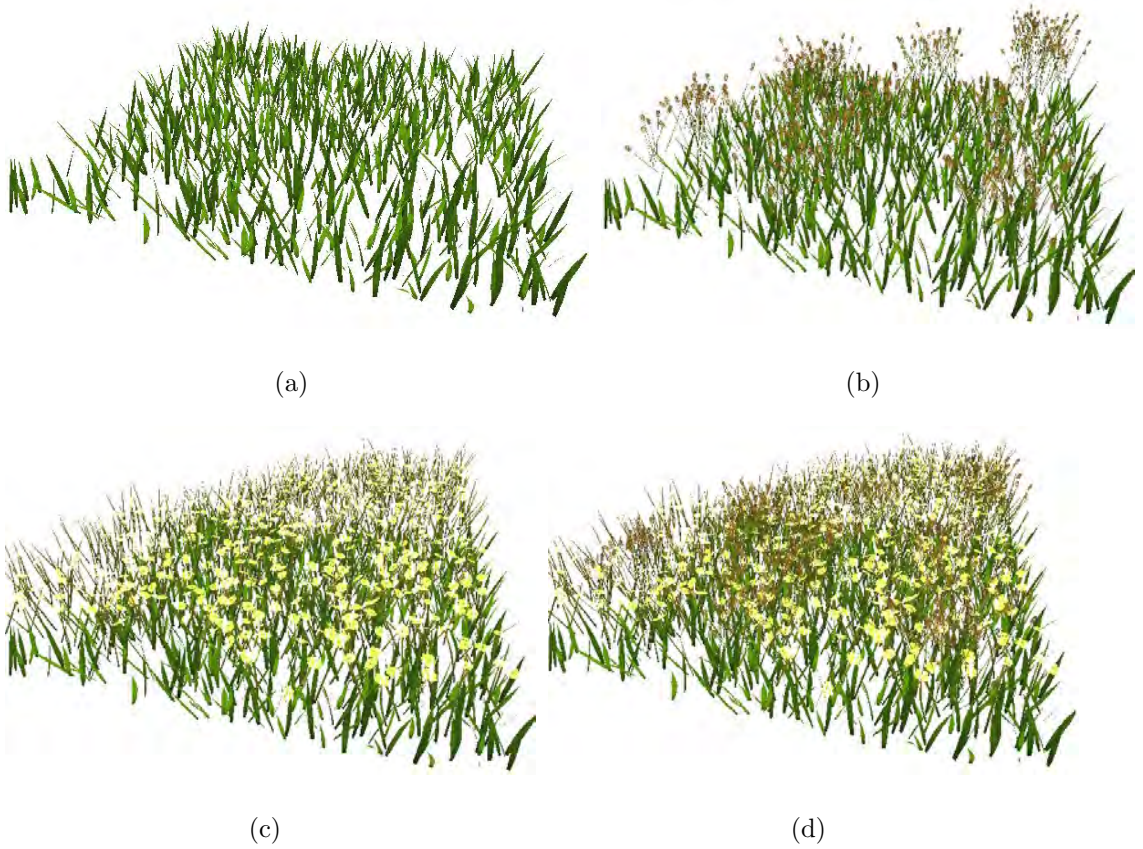


Figure 4.8: Ground covers created using Bantam3D Grass application.

- the color of the blade

Samples of ground cover created using the Bantam3D Grass application are shown in Figure 4.8.

The parameters of created ground covers for Figure 4.8 are summarized in Table 4.3.

4.6.2 Grass animation

In order to achieve a highly realistic animation, we used a calculation based on our simulated wind field. This calculation also took into account the position to be moved. The direction and strength of the prevailing wind were also important factors. Lower points of the blades are fixed at a given location, whereas the upper points are shifted in the mean wind direction by the estimated wind velocity.

Table 4.3: Parameters for Figure 4.8.

	Object type	Num. of nodes / obj.	Num. of objects
Figure 4.8(a)	Grass blade	6	1000
Figure 4.8(b)	Grass blade	6	1000
	Wheatgrass blade	12	500
Figure 4.8(c)	Grass blade	6	1000
	Daisy white flower	12	1000
Figure 4.8(d)	Grass blade	6	1000
	Wheatgrass blade	12	500
	Daisy white flower	12	1000

The middle points of the blades control the deflection and bending of the grass blade depending on the transition of the upper points. We wanted to draw numerous grass blades at one time, so the blades had to vary from time to time to make the grass blades look slightly different. Therefore, we applied different shades of green and yellow to provide a better differentiation of single blades.

A linear arrangement of grass blades would immediately make the structure recognizable. Therefore, we positioned the grass blades randomly. The length of the grass blades was also chosen randomly. In the simulation of grass animation, we did not consider stiffness and the clumping of grass blades for the sake of simplicity.

We use Taylor's frozen turbulence hypothesis to interpret a time series as a space series. Taylor's hypothesis is a widely accepted assumption and can be expressed for the two-dimensional case as follows:

$$u(x, y, t) = u(x - t \cdot U, y, 0) \quad (4.4)$$

The advantage of using this hypothesis is that the dimensionality of the data can be reduced from three to two.

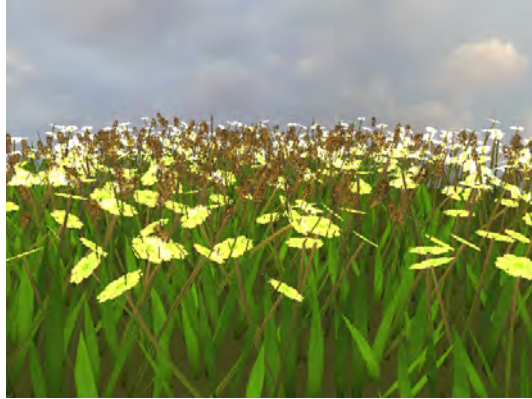
As stated in Chapter 4.3, our measured data was collected at a height of 3 m above the ground. Therefore, we decreased the wind velocity at the

height of grass based on Equation 3.2 in the grass animation. Animation frame sequences of the grass field for the sample Case 1 is shown in Figure 4.9. Figure 4.10 shows screenshots of grass fields from various view angles. Figure 4.11 shows the effects of wind turbulence travelling along the mean wind direction.

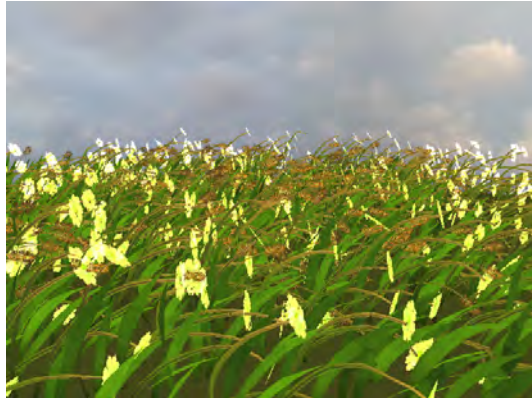
4.6.3 Performance results

We chose a grass field to visualize wind fields simulated by our approach. In our grass field animation, we considered three kinds of grass blades as shown in Figures 4.9 and 4.10. Figure 4.9 shows grass fields under different wind forces. Figure 4.10 shows screenshots of grass field animation showing the effects of wind turbulence traveling along the mean wind direction. From the experiments, our simulated grass field illustrates effects that are very similar to natural grass fields. Finally, in Figure 4.11, we show different grassland scenes.

Computational performance for Figure 4.10 is as follows: (a) 64.85, (b) 61.93, (c) 34.28, (d) 23.71 (fps). The computer environment used was a 3.0 GHz Intel Core 2 CPU, with a NVIDIA GeForce 9400 GT video card and 2GB of memory.



(a) No wind



(b) Gentle breeze



(c) Turbulent wind

Figure 4.9: Grass field under different wind forces.

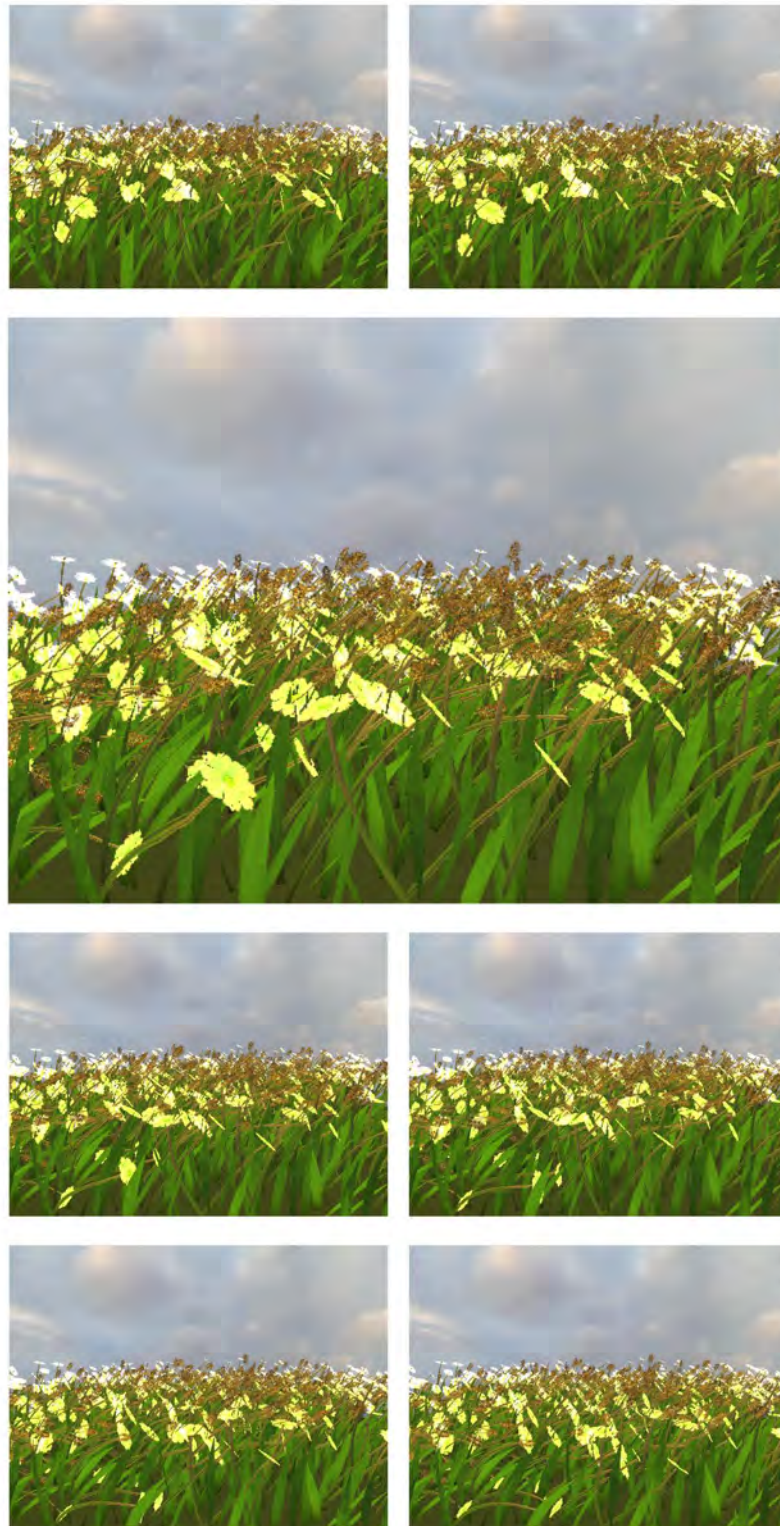


Figure 4.10: Screenshots of grass field animation showing the effects of wind turbulence traveling along in the mean wind direction.

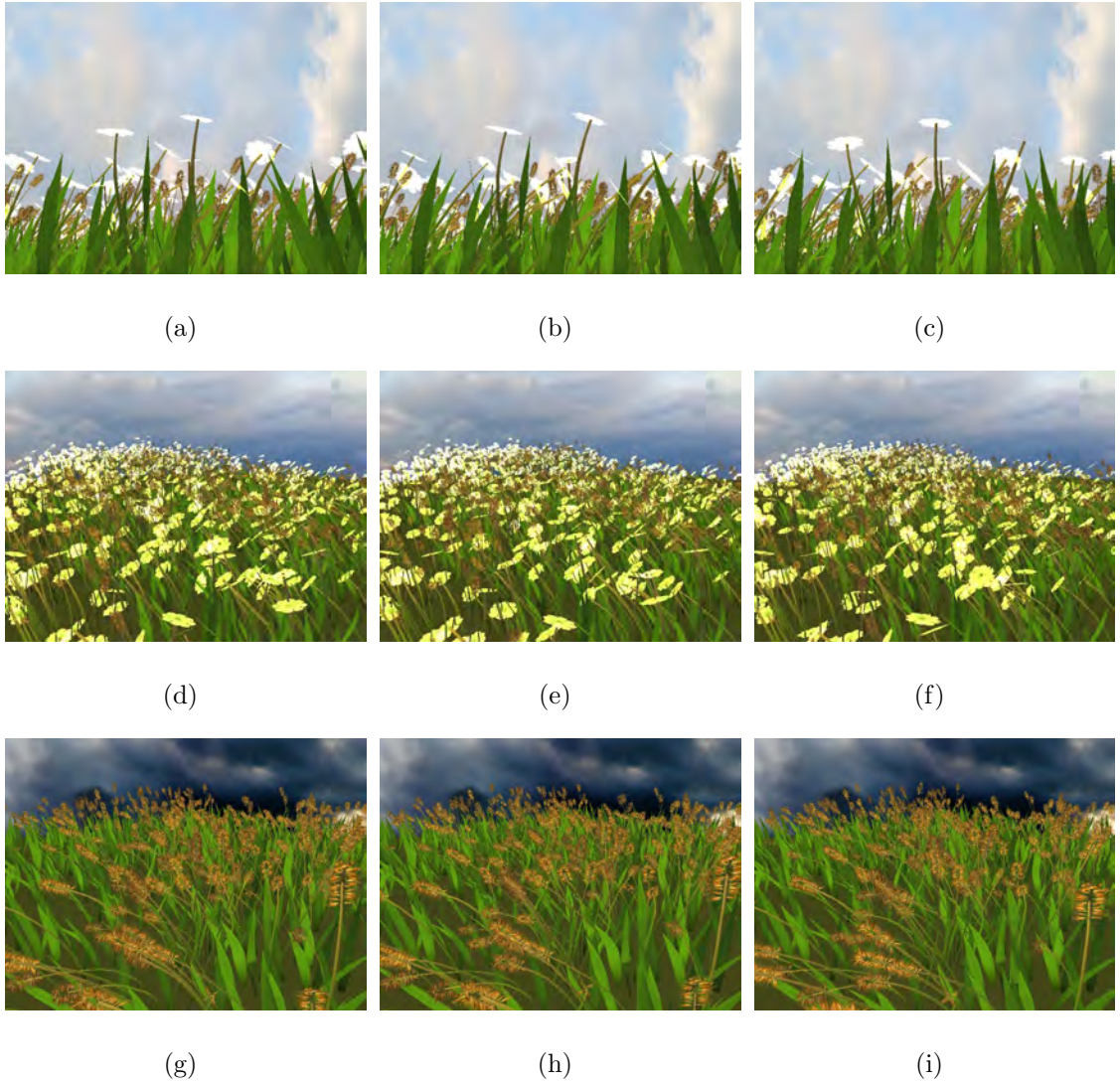


Figure 4.11: Different grassland scenes under wind effects.

Chapter 5

Noise-based Non-frozen Wind Field Generation

5.1 Background

In this section we present a simple method to generate three-dimensional frozen and non-frozen turbulent wind fields for use in the animation of wind-induced motion. Our approach uses $1/f^\beta$ noise to match the characteristics of natural wind. By employing a noise-based approach, the complexity as well as computational cost is reduced. Additionally, by considering key characteristics of actual wind that are applied in the structural engineering field, our proposed method is able to produce plausible results in outdoor wind field simulations. In this paper, we describe the implementation results of our proposed method and compare them with other existing approaches used to construct turbulent wind fields. The implementation and visualization are carried out for both two- and three-dimensional scenarios and compared with the results of other well-known methods.

The rest of the section is organized as follows: we start with discussion of the key characteristics of wind considered in the structural engineering field. In this section, some experimental data from previous section is included to illustrate the natural characteristics of wind predicted in an intuitive manner. Then, we introduce our proposed two-dimensional and three-dimensional wind

field model by comparing it to well-known methods. Finally, implementation and visualization results are described.

5.2 Characteristics of wind

Without a turbulent wind field, animations would lack a natural chaotic look, which is an important characteristic of most natural phenomena.

Here, we describe some fundamental characteristics and laws used as the theoretical basis of our research.

In general, the velocity of wind may be represented in vector form as

$$U(z, t) = U(z)i + u(z, t)i + v(z, t)j + w(z, t)k \quad (5.1)$$

where u , v , and w are the fluctuating components of gust along the x -, y -, and z -axes (longitudinal, lateral, and vertical axes) and $U(z)$ is the mean wind along the x -axis. The fluctuating component along the mean wind direction, u , is the largest, and it is therefore the most important parameter pertaining to wind.

To study turbulence using a continuous record of measurements at a single point, we need to assume that the turbulence is frozen at a point. Several mathematical models describe the wind power spectrum as shown in previous sections. Some of them have been suggested by von Karman, Kolmogorov, Davenport, Kaimal, etc. For example, the wind speed turbulence spectrum in the along-wind direction was described by Kolmogorov as:

$$\frac{nS_u(n)}{u_*^2} = Cf^{-2/3} \quad (5.2)$$

where n is the frequency in Hz, C is a constant, u_* is the friction velocity, $f = nz/U(z)$ is the normalized frequency, and $U(z)$ is the mean wind speed at height z .

A general form of the wind turbulence spectrum model was suggested by [37].

$$\frac{nS_u(n)}{u_*^2} = \frac{Af}{(1 + Bf^\alpha)^\beta} \quad (5.3)$$

The spatial description of turbulence is characterized by a coherence function [38], whereas the turbulence model for a single point in space can be

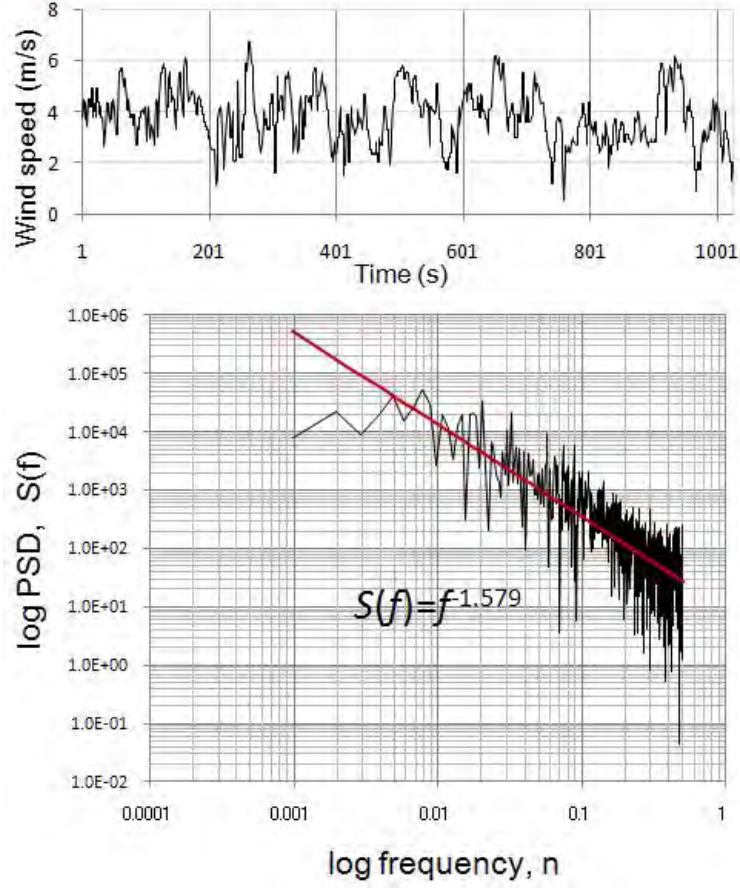


Figure 5.1: Measured wind speed history data in the along-mean-wind direction at a point in space and the corresponding power spectrum

described by Equations 5.2 or 5.3. Turbulence is often associated with wind shear. Wind shear refers to the change in the wind speed or direction with atmospheric elevation.

To describe the characteristics of natural wind, the time history data for the along-wind direction is discussed in this section. The wind speed data were recorded using a hot-wire anemometer at a rate of 1 Hz with a sampling interval of 1 s. The data contained 1024 data points. We measured the data at a height of 2 m above the ground in a suburban area. The mean wind speed from the collected data was found to be 3.81 m/s. That is, $U(2m) = 3.81$ m/s and $z_0 = 0.3$. Figure 5.1(a) shows the time history of the measured wind speed and Figure 5.1(b) illustrates its power spectrum in a log by log representation.

At first glance, the wind time-series resembles an irregular, noisy mess. However, as shown in Figure 5.1(b), the slope of the power spectrum for the actual wind data resembles $1/f^\beta$ -noise when $\beta = 1.58 (\approx 5.0/3.0)$.

Because “1/f-like” noises are quite prevalent in nature, we apply fBm noise while modeling our wind field (15). $1/f^\beta$ noise is an intermediate between white noise with no correlation in time and random walk (Brownian motion) noise with no correlation between increments. In general, a process with a spectral density proportional to $1/f^\beta$ corresponds to fBm for which $\beta = 2H + 1$.

$$S(f) = \frac{1}{f^\beta} \quad (5.4)$$

where the parameter H describes small-scale noise roughness.

In the next section, we describe our proposed approach in more detail for constructing a wind field based on fBm noise by considering the characteristics of actual wind.

5.3 Wind field model

5.3.1 Comparison with well-known approaches

Turbulent wind fields are modeled as stochastic processes. [39] employed a stochastic process and applied Fourier synthesis to derive a wind field in the spatiotemporal frequency domain and inverted the result to obtain a periodic spacetime wind field. [44] employed the same paradigm but they were mainly concerned with coupling the wind field model with microscopic interaction of gaseous and fluid phenomena. In contrast, [39] were mainly concerned with coupling the wind field model with macroscopic physical models of rigid or deformable objects. Both wind field models, however, can be applied to a wide range of phenomena.

Both the methods mentioned above adopted the wind field model used in the structural engineering field; this model considers the characteristics of wind. In [39], Shinya and Fournier used the experimental formulation of the power spectrum of wind in uniform fields and applied Davenports exponential coherence functions [38] to build a spatially correlated turbulent field. The

coherence function is used for all three wind directions. In [39], Shinya and Fournier selected power spectra for each of the following wind components: u , v , and w . The spectra were respectively referred to as S_u , S_v , and S_w , and are given by the following equations:

$$\begin{aligned} S_u &= C_0 200 (v/f) / (1 + 50v)^{5/3} \\ S_v &= C_0 15 (v/f) / (1 + 9.5v)^{5/3} \\ S_w &= C_0 3.36 (v/f) / (1 + 10v^{5/3}) \end{aligned} \tag{5.5}$$

where f is the temporal frequency in Hz, $v = fz/U(z)$, z is the height of the observation point from the ground, and $U(z)$ is the mean wind velocity at that point. To simplify the wind field model, Shinya and Fournier applied Taylor's frozen turbulence hypothesis.

Unlike [39], Stam divided the wind field into a large-scale component and a small-scale component. The small-scale term is modeled by random vector fields by considering the cross-correlation between different components of the velocity field at two different points in space-time. Additionally, the structure of a velocity field is entirely determined by Kolmogorov's energy spectrum. Moreover, instead of invoking Taylor's hypothesis, Stam and Fiume modeled the temporal frequency according to the energy spectrum function as well as the temporal spread function.

Additionally, the two methods mentioned above produce spatiotemporal wind velocity fields by accounting for spatiotemporal auto-correlation.

5.3.2 Important steps in constructing the wind field

To describe and compare our approach, we outlined the important steps and the differences between the abovementioned approaches [39] and [44].

- Stam and Fiume assume that a velocity field is homogenous in space and stationary in time, which implies that the corresponding cross correlation only depends on the difference between the two points and the difference between the two times.

- Stam and Fiume assume that the velocity field is spatially isotropic. Thus, the cross-correlation functions only depend on the distance between the points.
- The structure of a velocity field in [44] is determined by its energy spectrum function (via its cross-spectral density functions).
- The approach of Stam and Fiume uses the best-known Kolmogorov energy spectrum, whereas Shinya and Fournier use the wind turbulence spectrum model, which is determined using experimental data and given by Equation 5.5.
- Instead of invoking Taylor’s hypothesis in [39], Stam and Fiume model the temporal frequency dependence of the energy spectrum using a temporal spread function.
- Before computing the velocity field in the frequency domain, Shinya and Fournier implement Fourier transform to reduce spatial dependence, whereas Stam and Fiume compute transformed kernels in terms of cross-spectral density functions by inserting expressions for Fourier velocity components.
- The velocity field is then obtained by taking the inverse Fourier transform for each wind component.

5.4 Our approach

Our fBm-based wind field model does not cover cross-spectral density functions applied in the approaches [39], [44] mentioned in the previous section. We construct the wind field directly from the power spectral density functions by applying higher dimensional fBm noise. Therefore, our approach is quite simple as compared to the existing approaches. Figure 5.2 shows a graphical representation of our noise-based non-frozen wind field generation approach.

Additionally, we consider the key characteristics of natural wind employed in the structural engineering field in our wind field modeling approach. Our approach provides an intuitive description of wind parameters and allows animators to control the wind field by using those parameters. It also constructs

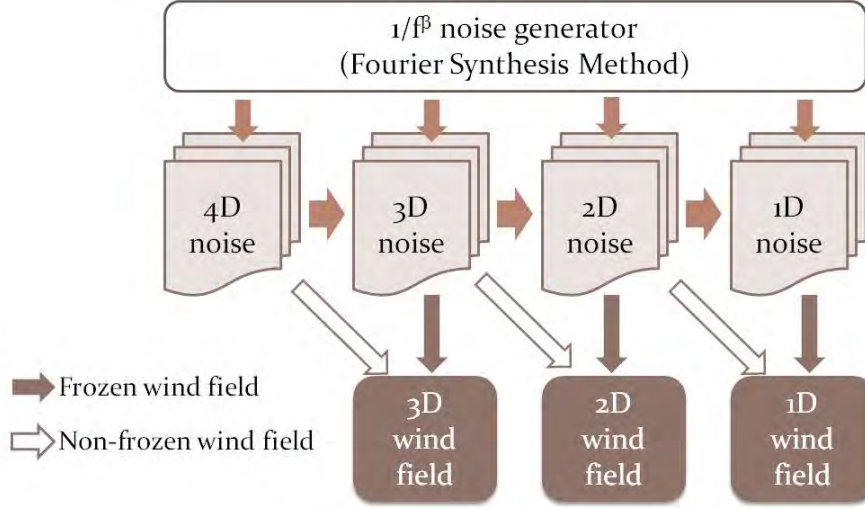


Figure 5.2: A graphical representation of our proposed approach

spatially as well as temporally varying 2D and 3D wind fields and shows a comparison of the visualization results of the two approaches mentioned above.

5.4.1 Consideration of power spectral density

Since we use higher dimensional fBm noise data, we employ the PSD function, which is given as follows.

$$S(f_1, \dots, f_n) = \frac{A}{\left(\sqrt{\sum_{i=1}^n f_i^2}\right)^{\beta+(n-1)}} \quad (5.6)$$

where n is the dimension of the wind field and β is the slope of the power spectrum. The algorithm developed by [49] is the main point of reference for the generation of $1/f$ noise.

5.4.2 Physical meanings of wind field model parameters

If we adopt the wind power spectrum model suggested by Kolmogorov and given by Equation 5.2, the intuitive meanings of parameters A and β can be derived from the following equation:

$$S_u(n) = u_*^2 \left(\frac{U(z)}{z} \right)^{\frac{2}{3}} \frac{C}{n^{5/3}} \quad (5.7)$$

In our model, we determined the parameter values of A and β with small modifications, given as follows:

$$\begin{aligned} A_u &= u_* \left(\frac{U(z)}{z} \right)^{\frac{2}{3}} \\ \beta &= 5.0/3.0 \end{aligned} \quad (5.8)$$

When we used the wind turbulence spectrum model, we assumed that the constant C is equal to 1 and omitted the degree of the power of friction velocity because our model was highly sensitive when we presumed the friction velocity as a degree of power, as defined in other wind field models. Usually wind spectrum models describe strong wind turbulence at high altitudes. However, in outdoor scenery simulations, we need to simulate winds closer to the ground. From Equation 5.8, we can describe the physical meaning of a parameter A in the along-mean-wind direction in terms of wind characteristics, where z is the observed height above the ground, u_* is the friction velocity, and $U(z)$ is the mean wind velocity at a height of z .

The lateral and vertical turbulence components are generally lower in magnitude than the corresponding longitudinal value. Thus, the following simple relationships are applied to the parameter value of A for the other two directions of wind velocity [17]:

$$\begin{aligned} A_v &= 0.88A_u \\ A_w &= 0.55A_u \end{aligned} \quad (5.9)$$

The parameter β can vary depending on the characteristics of wind as well as the velocity component of wind. Generally, the parameter β is selected as 5.0/3.0 according to Kolmogorov's law applied in the along-mean-wind direction. Also, the friction force u_* can be derived from Equation 3.3. To determine the friction force u_* we must know the mean wind speed $U(z)$ at a height of z above the ground and the terrain roughness z_0 . Typical values of terrain are provided in Table 3.2. Animators can use Table 3.2 as the reference to control the wind field while simulating a specific type of wind depending on the terrain roughness and the height of the observation point. In Chapter

5.4.4, we illustrate the influence of parameters on the simulated wind fields for various terrain conditions at different heights.

5.4.3 Implementation steps

We now describe the implementation steps involved in the calculations made in our proposed approach.

- Step 1: Determine characteristics of wind fields by defining wind parameters such as mean wind speed $U(z)$ at the given height z and the terrain roughness z_0 .
- Step 2: Calculate friction force using Equation 3.2.
- Step 3: Calculate parameter values of A_u , A_v , and A_w , as defined in Chapter 5.4.2, for each wind velocity component.
- Step 4: Select parameter β as 5.0/3.0 according to Kolmogorov's law applied in the along-mean-wind direction.
- Step 5: Construct wind power spectrums for each wind direction according to Equations 5.6, 5.8 and 5.9.

$$\begin{aligned}
 S_u(f_1, \dots, f_n) &= \frac{A_u}{\left(\sqrt{\sum_{i=1}^n f_i^2}\right)^{\beta_u + (n-1)}} \\
 S_v(f_1, \dots, f_n) &= \frac{A_v}{\left(\sqrt{\sum_{i=1}^n f_i^2}\right)^{\beta_v + (n-1)}} \\
 S_w(f_1, \dots, f_n) &= \frac{A_w}{\left(\sqrt{\sum_{i=1}^n f_i^2}\right)^{\beta_w + (n-1)}}
 \end{aligned} \tag{5.10}$$

- Step 6: Apply inverse Fourier transform based on the dimension of the simulated wind field for each wind component:
 - IFFTnD(S_u)
 - IFFTnD(S_v)
 - IFFTnD(S_w)
- Step 7: Employ Taylor's frozen turbulence hypothesis to generate frozen wind fields, if required.

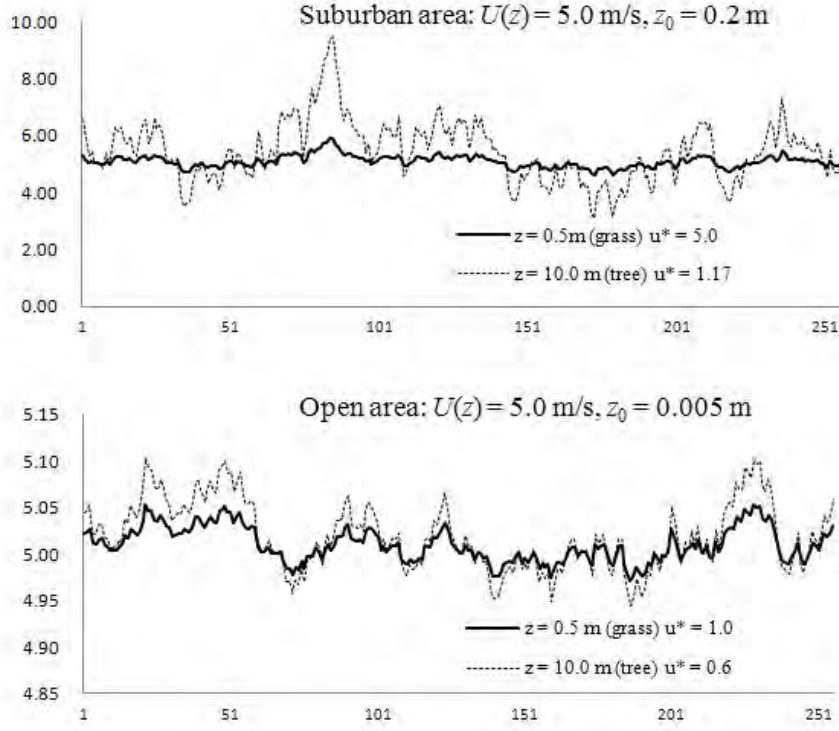


Figure 5.3: Simulated winds at different parameter values

5.5 Results

This section briefly describes the experimental results obtained by constructing specific types of wind fields according to the terrain and height pertaining to the selected parameter values that are listed in Table 3.2, as illustrated in Chapter 3. 3.

In our experiments, we simulated winds for different terrains. In the first case, we constructed winds in suburban area. In the second case we considered terrain as an open area. In our simulations we also examined the elevation of the observation points. This could be used in graphical outdoor scenery simulations. The simulated winds are illustrated in Figure 5.3 with the selected wind parameters. As shown in Figure 5.3, wind turbulence has a strong dependence on terrain roughness and the elevation of the observation point for a given mean wind speed. Wind is more turbulent in suburban areas than in open areas.

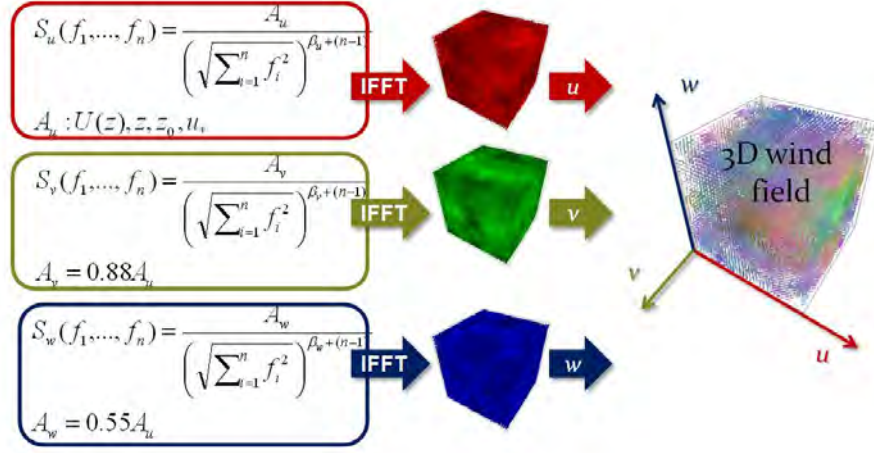


Figure 5.4: Algorithm overview of our proposed method for generating non-frozen 3D wind field.

5.5.1 Wind field simulation and its visualization

In this section, we illustrate the simulation of 3D non-frozen turbulence and 2D frozen turbulence wind fields and compare our results with those obtained by other well-known approaches. To demonstrate the turbulence of 3D velocity fields, we used an RGB color combination that corresponds to the magnitude of the velocity components u , v , and w . For example, the color value R determines the strength of the u component in the simulated wind field at a given point. Further, in our illustration, the direction as well as magnitude of velocity is depicted by a line (see Figure 5.4).

5.5.2 Performance results

The visualization results of simulated wind fields are shown in Figure 5.5 and Figure 5.6. Table 5.1 and Table 5.2 show a comparison between our approach and other approaches in terms of method complexity and computational cost.

Note that performance measurements were obtained using a computer with Intel Core (3.0GHz) processor and 2GB of RAM and measured in milliseconds.

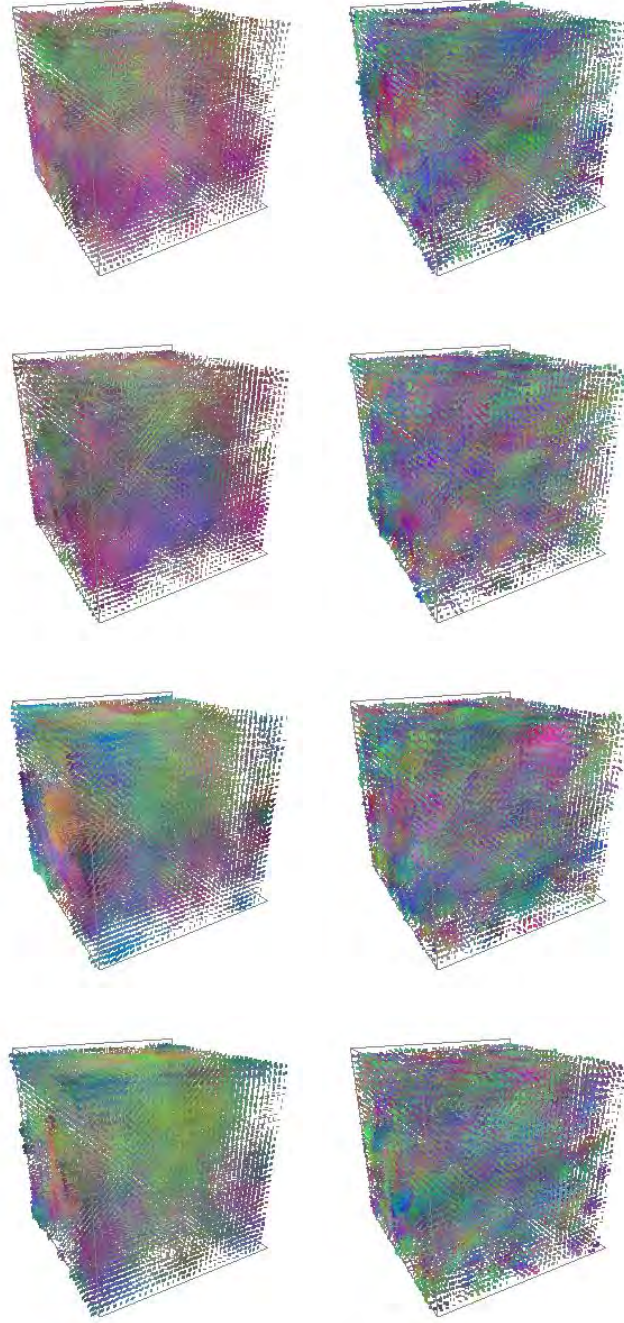


Figure 5.5: Three-dimensional non-frozen turbulent wind fields 32^4 . Screenshots of simulated three-dimensional wind velocity fields. (Left) by our approach. (Right) by Stam and Fume's approach.

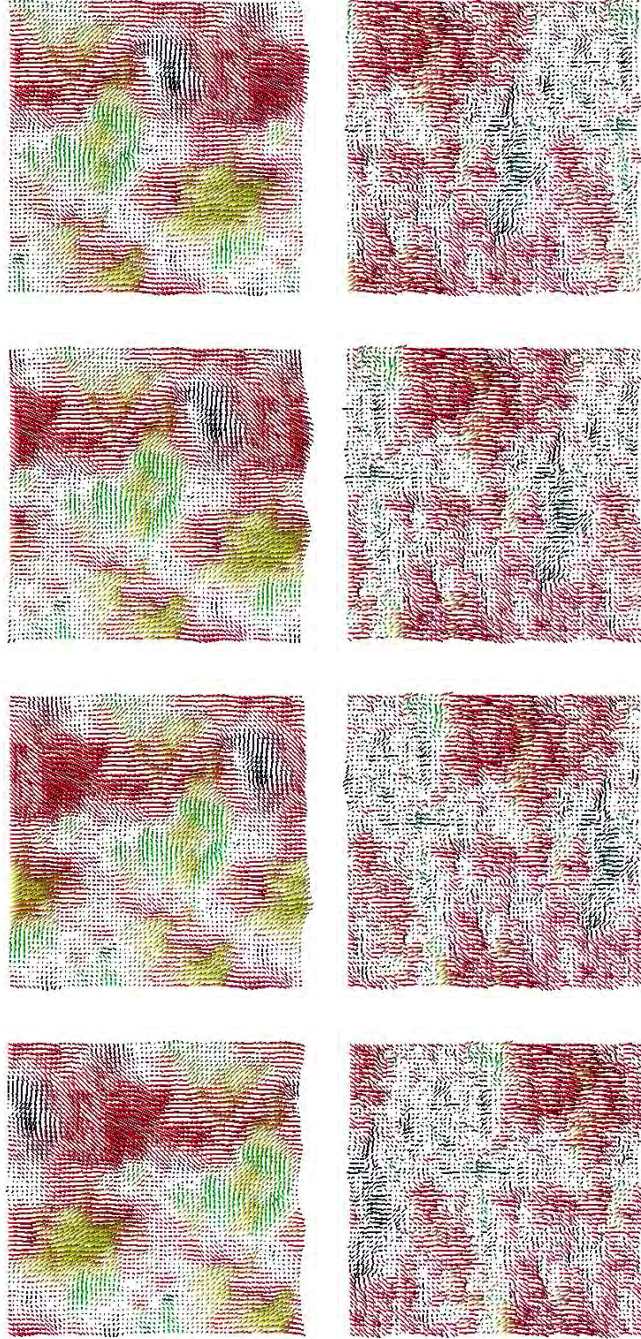


Figure 5.6: 2D frozen turbulent wind fields (64×64). (Left) Screenshots of simulated 2D wind velocity fields by our approach. (Right) Screenshots of simulated 2D wind velocity fields by Shinya and Fournier's approach.

Table 5.1: Performance results of 2D frozen turbulent wind fields.

	Size 512×512	Size 1024×1024
Our approach	359 ms	2438 ms
Shinya et al. [1]	493 ms	2969 ms

Table 5.2: Performance results of 3D non-frozen turbulent wind fields.

	$32 \times 32 \times 32 \times 32$	$64 \times 64 \times 64 \times 64$
Our approach	3250 ms	64297 ms
Stam et al. [1]	4188 ms	83158 ms

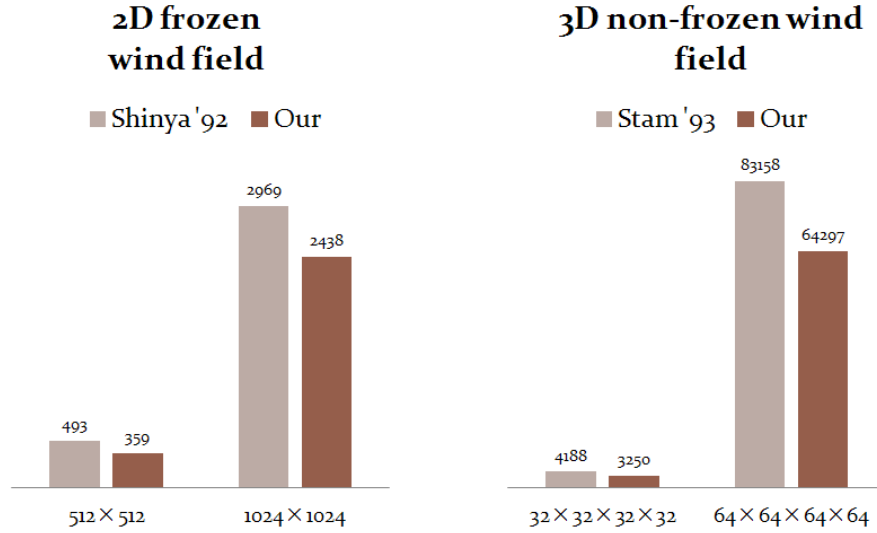


Figure 5.7: The running time of generating wind fields. Measured in milliseconds and obtained with Intel Core (3.0GHz)

Chapter 6

Applications of Noise-based Non-frozen Wind Field

Our noise-based wind fields can be applied to a wide range of phenomena. In this section, we show the effects of our noise-based non-frozen wind fields. Moreover, we propose a simple approach for visually simulating fire and modeling the effects of winds on cloth.

6.1 Animation of grass field

We chose a grass field to visualize two-dimensional wind fields. Our purpose is to simulate grass that wave realistically in the wind. We should consider a detailed modeling of the individual blades of grass to simulate a grass field. For larger meadows, it would require the number of polygons. Therefore we apply a simple but useful alternative that meets the condition of representing deformation caused by the influence of wind velocity.

Grass blade is seldom a straight line. In nature, each grass blade is curved slightly. So in our simulation, grass blades are built by Bezier curves. A typical grass blade will be copied many times to create the ground cover. We will make the copies look slightly different from each other in order to simulate nature. Sample curves of the typical grass blade are shown in the Figure 6.1.

In order to achieve a highly realistic animation, we use a calculation based on our simulated wind field. This calculation should take into account the

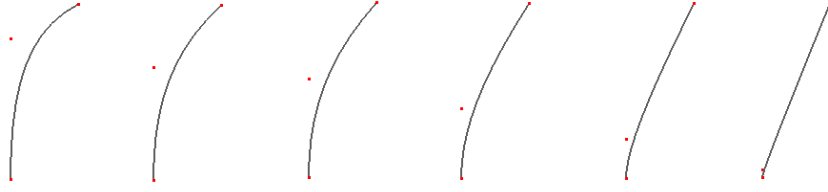


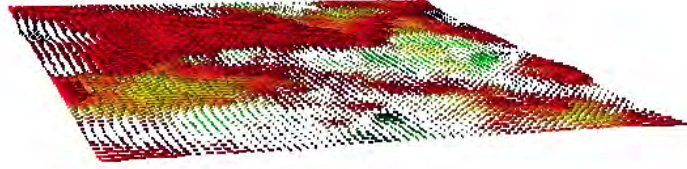
Figure 6.1: Sample curves of a blade of grass

position that has to be moved. Also, the direction and strength of the prevailing wind will be factors. The lower control point of the curve will be fixed at the given location while the upper control point is shifted along the mean wind direction by the estimated wind velocity. The middle control point of the curve manipulates the deflection and bending of the grass blade depending on the transition of the upper control point.

Since we want to draw numerous of grass blades at a time, the curves must vary from time to time to make the grass blades look slightly different. Therefore, we apply different shades of green and yellow to get a better differentiation of single blades. For the shading, we computed two-dimensional fBm noise to determine the color intensities of the grass blades (See Figure 6.2).

A linear arrangement of the grass blades would immediately make the structure recognizable. Thus, we position blades randomly. As well as the length of the grass blades is chosen randomly. In the simulation of grass animation, we did not consider the stiffness and the dumping of grass blades for the sake of simplicity.

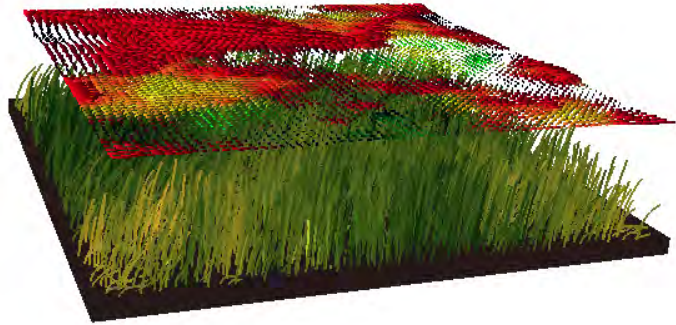
Figure 6.2 shows a procedure for animating a grass field by using our simulated turbulent wind fields. For large-scale scenery simulations such as those of grass fields, it is better to use Taylor's frozen turbulence hypothesis to decrease the dimensionality of space. We compared the result of our grass field simulation and its wind velocity field with the results obtained from Shinya and Fournier's approach (see Figure 6.3). From the experiments, our simulated grass field illustrates effects that are very similar to natural grass fields as well as the simulated fields obtained by Shinya's approach.



(a)



(b)



(c)

Figure 6.2: A process for simulating grass field animation. (a) 2D turbulent frozen wind field generated by our approach (b) grass field is built by a simple Bezier curve (c) grass field animation.

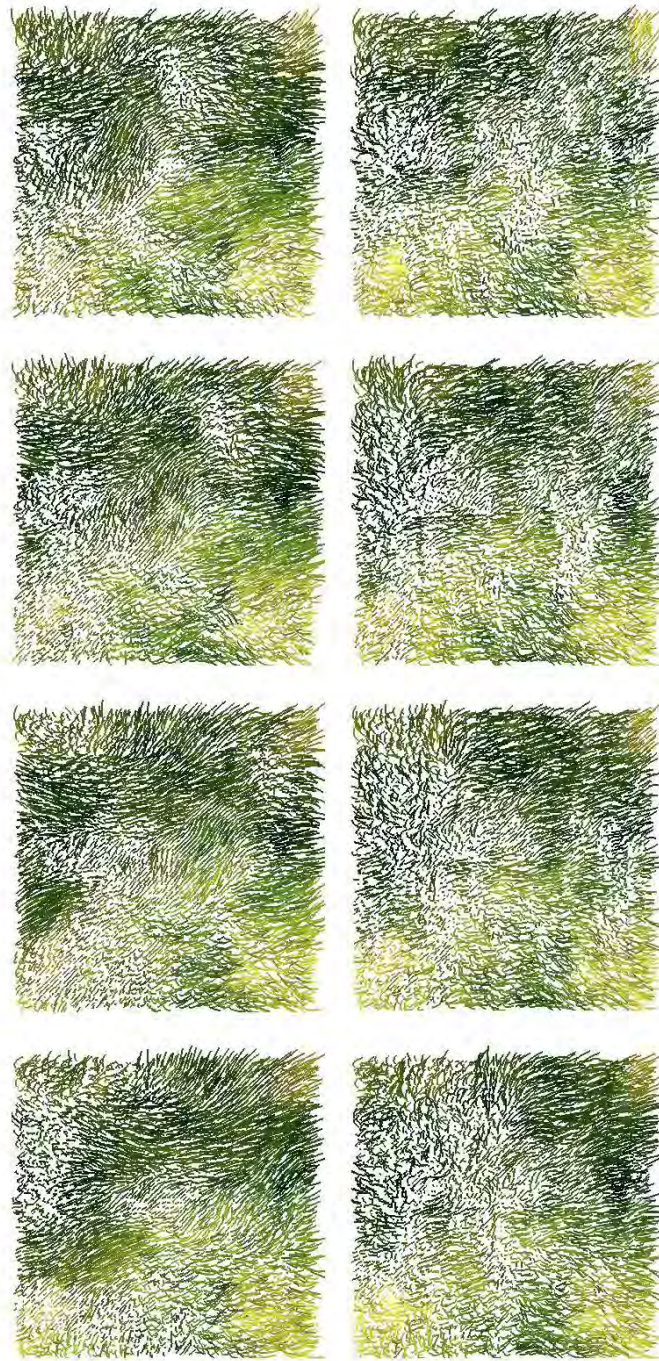


Figure 6.3: Top views of grass fields generated by the corresponding 2D frozen turbulent wind fields. (a) Grass field generated by our approach (b) Grass field generated by Shinya and Fournier's approach.

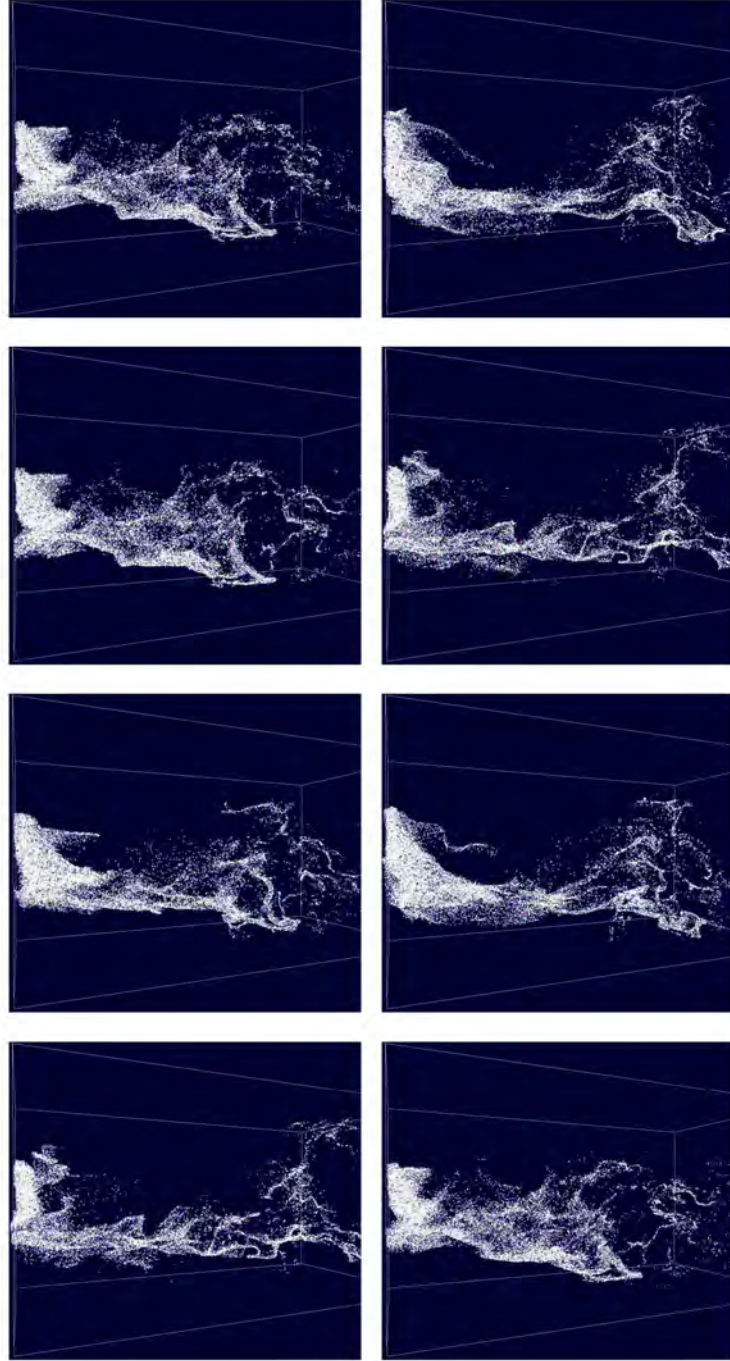


Figure 6.4: Three-dimensional non-frozen turbulent wind fields ($64 \times 32 \times 32 \times 64$). Visualization by our approach.

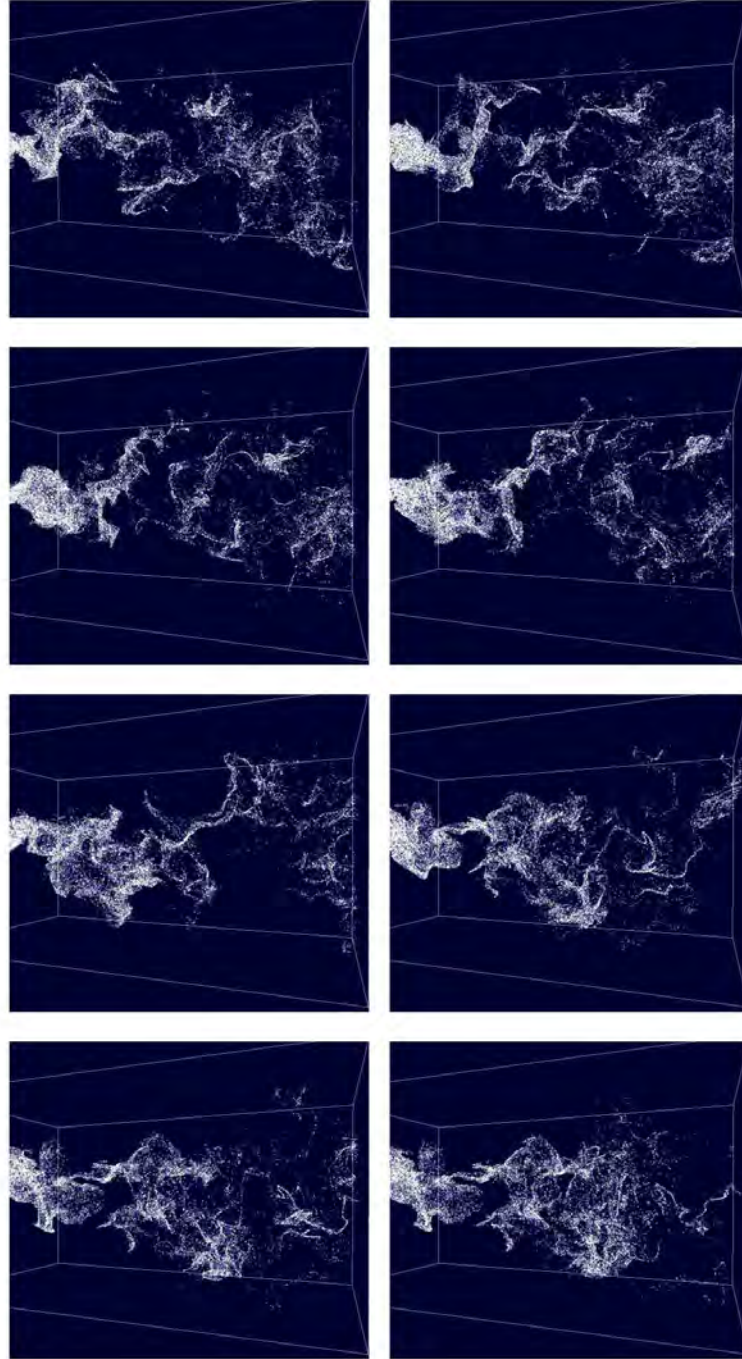


Figure 6.5: Three-dimensional non-frozen turbulent wind fields ($64 \times 32 \times 32 \times 64$). Visualization by Stam and Fiume's approach

6.2 Tracing particles

Particle systems can produce so much irregular, three-dimensional detail that exact shading and visible surface calculations become infeasible. Particle systems represent objects as clouds of primitive clouds that occupy their volumes, rather than using more classical surface-based representations such as polygons, patches, and quadric surfaces. A particle system is not a static entity, as its particles can move and change form with the passage of time. The position, orientation, attributes, and dynamics of each particle are defined by a set of constrained stochastic processes.

Therefore, we preferred tracing particles through the wind fields as a better visualization method for 3D wind fields. Figure 6.4 shows screenshots for particle tracing through the wind field simulated by our non-frozen 3D wind field. Figure 6.5 shows results for particle tracing obtained by Stam and Fiume’s approach.

The believability of a particle system depends on rendering as much as it depends on the actual simulation of the behavior. Given the tremendous amount of phenomena that can be modeled using particles, it shouldn’t be surprising that many different rendering approaches exist.

A total of 30,000 particles were traced through the wind field to show the effects of wind field turbulence. We used very simple shading calculations. Each particle is stochastically assigned an initial color that changes over time according to a simple relationship that simulates lifespan of the particle.

6.3 Animation of cloth in the wind

In this section, we describe an implementation of some of the basic features of a highly deformable object, such as a cloth-like object, in a wind. There are many ways of simulating cloth. Finite element methods offer a physically accurate approach that breaks the highly deformable object into elements, often tetrahedral, over which partial differential equations are solved for the stresses. Shape matching applies a penalty to parts of a model on the basis of the distance from some optimal position, with the effect of driving the object toward its original shape. Mass and spring models construct the highly deformable object from a set of masses connected by weightless springs that

apply forces to the masses in proportion to the compression or extension from rest length.

The simulation method used in our animation of cloth is similar to a mass and spring model, but rather than applying forces to the masses it applies position and velocity corrections based on the work of [47]. To model cloth, the particle system is modified with constraints. Nonetheless, the essence of the simulation is a particle system.

6.3.1 Verlet integration

Typically, in implementations of particle system, each particle has two main variables: its position $pos = (x, y, z)$ and its velocity $vel = (u, v, w)$. Then, in the time-stepping loop, the new position $posNew$ and velocity $velNew$ are often computed by applying the following rules:

$$\begin{aligned} posNew &= pos + vel\Delta t \\ velNew &= vel + a\Delta t \end{aligned}$$

where Δt is the time step, and a is the acceleration computed using Newton's law $F = m \cdot a$ (where F is the total force acting on the particle).

Here, however, instead of storing the position and velocity of each particle, we store the current position pos and the previous position $posPrev$. Keeping the time step fixed, the update rule is then as follows:

$$\begin{aligned} posNew &= 2.0 \cdot pos - posPrev + a \cdot \Delta t^2 \\ posPrev &= pos \end{aligned}$$

This is called Verlet integration [48] and is used intensely when simulating molecular dynamics. It is quite stable since the velocity is implicitly given. By lowering the value 2.0 to something like 1.99 a small amount of drag can be introduced into the system.

6.3.2 Solving constraints by relaxation

A common model for cloth consists of a simple system of interconnected springs and particles. However, it is not always trivial to solve the corresponding system of differential equations. It suffers from some of the same

problems as the penalty-based systems. Strong springs leads to stiff systems of equations that lead to bad performance, or even instability, and in turn disappointment if only simple integration techniques are used. Conversely, weak springs lead to cloth that looks elastic.

If we assume that our world is the inside of the cube $(0, 0, 0) - (100, 100, 100)$, we need to keep all positions inside the valid interval:

$$0 \leq x \leq 100$$

$$0 \leq y \leq 100$$

$$0 \leq z \leq 100$$

If we now extend the experiment to model a stick of length 10, then we require a distance between two individual particles (with positions pos_1 and pos_2) to be 10.

$$|pos_2 - pos_1| = 10$$

Although the particles may initially have been correctly placed, after one integration step the separation distance between them might have become invalid. In order to obtain the correct distance once again, we move the particles by projecting them onto the set of solutions described by a constraint. This is done by pushing the particles directly away from each other or by pulling them closer together.

6.3.3 Cloth simulation

In our cloth simulation, rectangular cloth nailed at its top corner starts from a vertical hanging position and flutters in a strong or light wind. Figure 6.6 shows the simulation results of cloth fluttering in the wind. The internal forces acting on the cloth derived from spring and bending, which each apply both elastic and damping forces. The external forces acting on the cloth are gravity and wind.

We simulated a cloth-like object fluttering in our wind field with resolution $[32 \times 32 \times 32 \times 32]$. The simulation results were obtained using a computer with an Intel Core (3.0 GHz) and NVIDIA GeForce 9400 GT video card. We achieved a realistic looking cloth animation running at around 65 fps using $[64 \times 64]$ particle nodes.

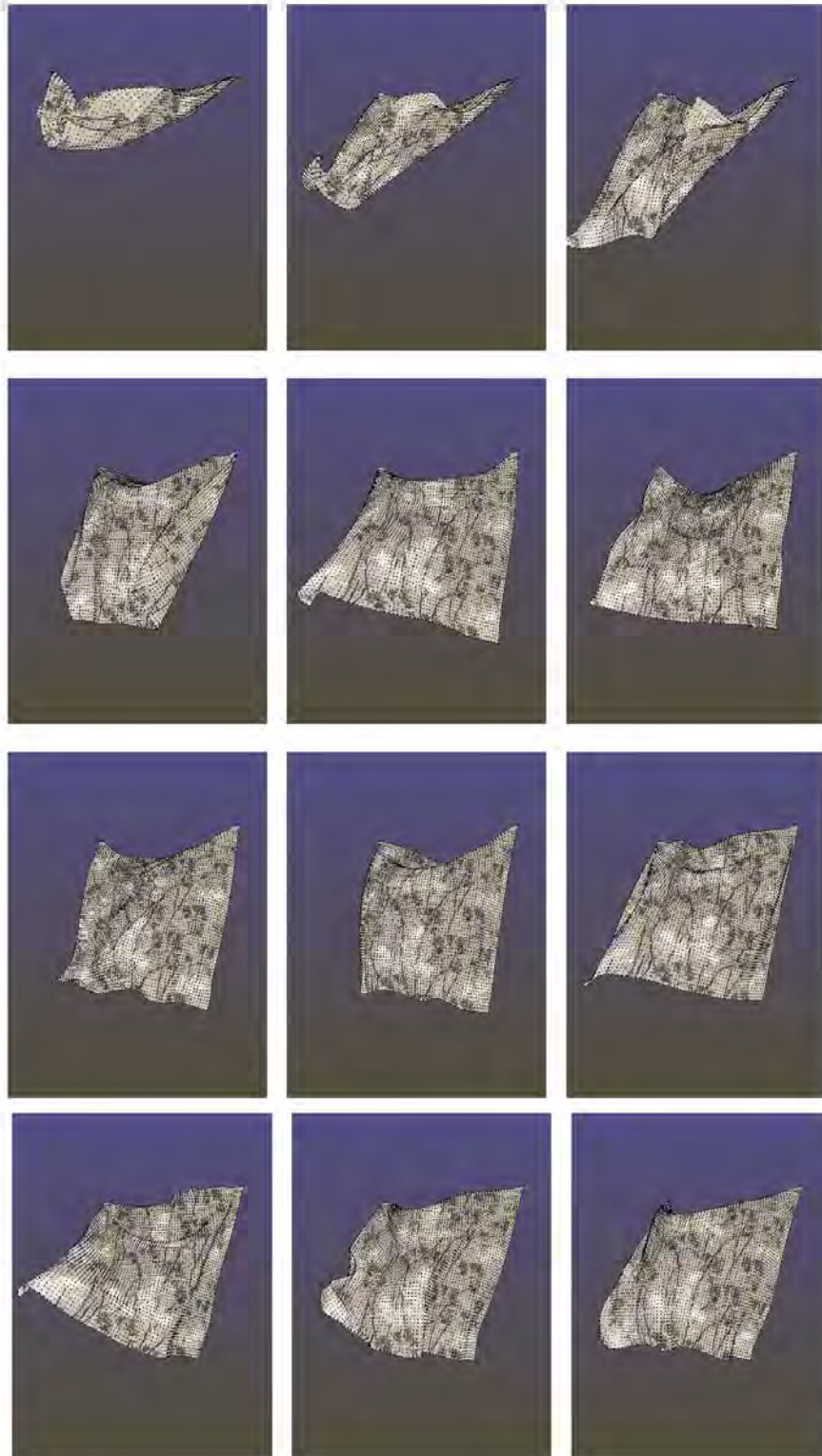


Figure 6.6: Animation sequences of cloth like objects in the 3D non-frozen wind field (sequences from top to bottom and left to right)

6.4 Animation of fire

In this section we present a simple technique for animating fire. Fire motion is determined by the wind field model which is based on fBm noise functions. Our method is very suitable for modeling the turbulent motion of an open fire such as one depicting campfire. We model and render the flame of fire and its spread on the basis of our wind field generation. For the visualization purposes a simple particle tracer will be applied. We also discuss how to use billboards to render flame of fire at interactive animation rates. Finally, some believable animation results are overviewed.

6.4.1 Introduction

Fire is a dramatic element that requires maintaining a believable appearance. Developing a visually convincing model of fire is among the most difficult and attractive problems in computer graphics. There are multitudes of fires that exist in the physical reality and have a wide range of visual representations. With the very first computerized graphical fire simulation, Reeves [37] introduced particle systems as a modeling, animation and rendering primitive. Due to the discrete nature of particles, an enormous amount of them were required to achieve good results. See Figure 6.7(a).

Nguyen et al. [27] present a purely physics-based method for modeling fire. The simulation uses the incompressible Navier-Stokes equations to model the hot gases, thereby also modeling the expansion effects caused by vaporization, and the effect of buoyancy in supporting the elevation of smoke and soot (See Figure 6.7(b)). Modeling fire in this manner is very complex and requires a large series of complex equations to be solved on a large scale. Purely physics-based simulations such as Nguyen et al. scale very poorly, thus making animation of large fires extremely slow and inefficient.

Lamorlette et al. [23] introduce an entirely different sort of pre-rendered fire simulator. In the simulation of animated motion picture movie, Shrek, they use physics-based wind fields (Figure 6.8 (a)). Stochastic models of flickering and buoyant diffusion reflect the appearance of the scene, and the simulation allows for noise to be composed over the wind fields for enhanced

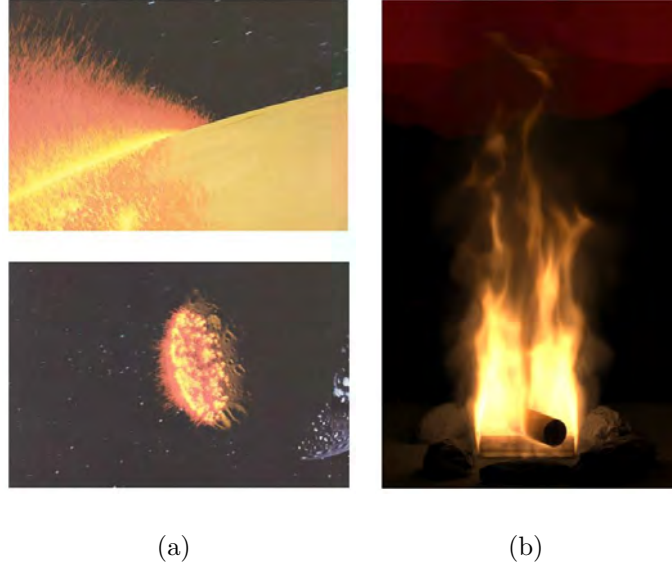


Figure 6.7: (a) Wall of fire from Star Trek II from Reeves et al. (b) The sample image from Nguyen et al.

control over motion and scale. Other aspects of the simulation, such as combustion spread, flickering, separation and merging are all manually controlled through procedural mechanisms.

3D real-time fire finds its application in interactive applications. Only in the past decade, however, has it become three-dimensionally feasible. Among interactive applications, computer games, having had a need to represent explosions and fire, have been using graphical fire animations. Since that time, hardware rendering speeds have increased exponentially, allowing for more and more detailed special effects. Independent of computer gaming, Wei et al. [50] proposed the use of texture splats (textures with a specified degree of transparency) as their basic display primitive for the real-time fire simulation (See Figure 6.8(b)). To achieve real-time performance, they utilized a model known as the Lattice Boltzmann Model to simulate physics-based equations describing fire evolution and its interaction with the environment (e.g. obstacles, wind and temperature).

Perry et al. [35], and more recently Beaudoin et al. [2], both worked to simulate the spread of fire on polygonal meshes. In their model, each flame is constructed out of a series of hexagon particles stacked and scaled above one another. The effect is very convincing on a small, candle-sized scale. The results on any larger scale, however, are not quite so convincing.

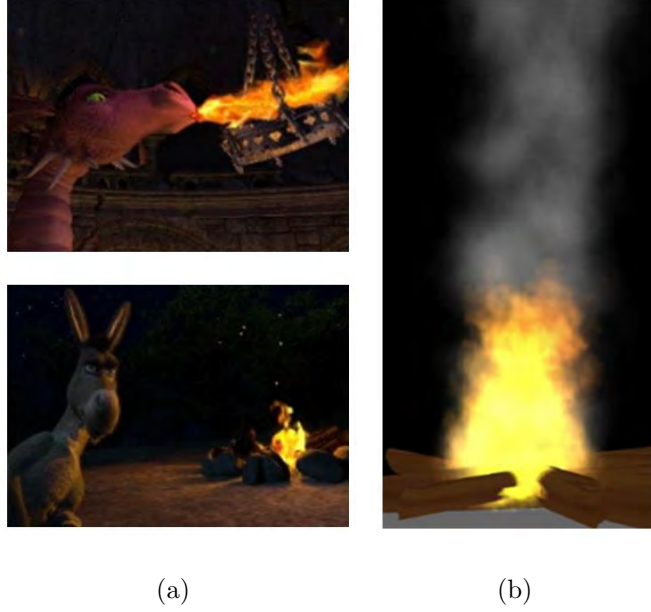


Figure 6.8: (a) Image from Shrek modeled by Lamorlette et al. (b) A simple campfire using texture splats from Wei et al.

Muller et al. [26] investigated real-time particle-based water rendering, proposing a new method of handling surface tensions. The method was based on Smoothed Particle Hydrodynamics and utilized Computational Fluid Dynamics for physics-based modeling. While fire and water are clearly not the same, they share certain visual properties that make this work worthy of comparison. The visualization method used in rendering the water particle system was a method known as point splatting. Using a volumetric approach such as the marching cubes algorithm is not effective when dealing with an intangible substance such as fire, which has no “surface”. Nevertheless the idea of point splatting is not an unrealistic one to apply to fire visualization; it had been applied with textures in the fire simulator created by Wei et al [50].

This section presents an alternative approach for simulating fire based on our wind field generation. Our fire simulation method by no means provides complete physical simulation. We expect fire to look complex and unpredictable, while at the same time having a recognizable structure that defines fire phenomena.

The simulation of fire can be broken into three non-overlapping parts: modeling, animation, and visualization [28]. Particle systems are the most widely used for fire model [29], [45]. Thus, we also use particle emitter sys-

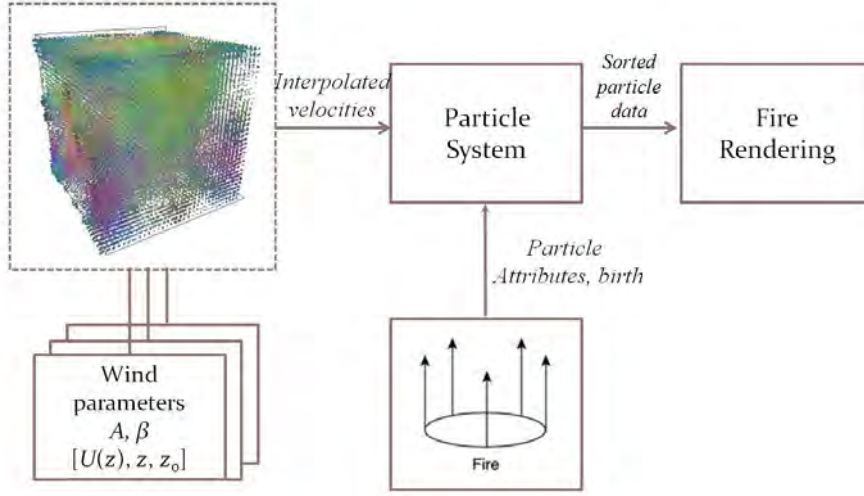


Figure 6.9: Basic idea of modeling fire

tems to create 3D fires. First, we adopt noise-based wind field generation model realistically depicting nature of wind. In our approach, we implement fractional Brownian motion (fBm) functions using a simple spectrum synthesis based on FFT. The basic idea behind fire animation is that the force of wind moves the fire blobs (particles) along the wind field to determine the effects of fire. On the other hand, in our model, the particles play the role of tracers in the field of turbulence for simulating fire. Finally, to complete the system, we need to define a number of custom controls to govern placement, intensity, lifespan, and evolution in shape, color, size, and behavior of the fire flames.

6.4.2 Modeling and visualizing fire

Our goal is to create a visual simulation of fire and to simulate the effects arising from the surrounding global winds. Figure 6.9 shows a schematic overview of our system, which consists of three components; a noise-based wind field generator, a dynamical particle system, and a rendering system. The earliest related work in computer graphics is by Reeves [37], in which particle systems were used to evoke a variety of visual effects, including fire. We also use particle-based simulation for the animation of fire. Particles are created at some kind of emitter. A variety of emitters have frequently used (see Figure 6.10). If we want our system to behave in any interesting way,

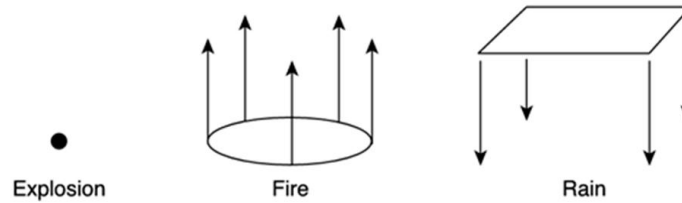


Figure 6.10: Emitters for different effects

the key is to generate each particle with slightly different initial values, so the behavioral rules make each one seem a unique element.

In the particle based simulation, it is important to solve problems on how to model the appearance and disappearance of particles, how to model the motion, how to construct environment and how to render particles.

6.4.2.1 A simple model of fire

The fire itself is modeled using a particle system. Each particle contains the following information regarding the modeling aspect of the fire system:

- *Position* - the 3D coordinate set of the particle at a given point in time.
- *Death* - the time at which the particle will die. Once this time expires, the death is reset to a new value and particle is reborn at the base of the flame
- *Current cell* - a pointer to the environment cell in which the particle is currently located
- *xOrigin, zOrigin* - the coordinate pair at which the particle originated. The position of particle is reset to this point at the base of the flame when the particle is reborn
- *cellOrigin* - a pointer to the environment cell in which the particle originated.

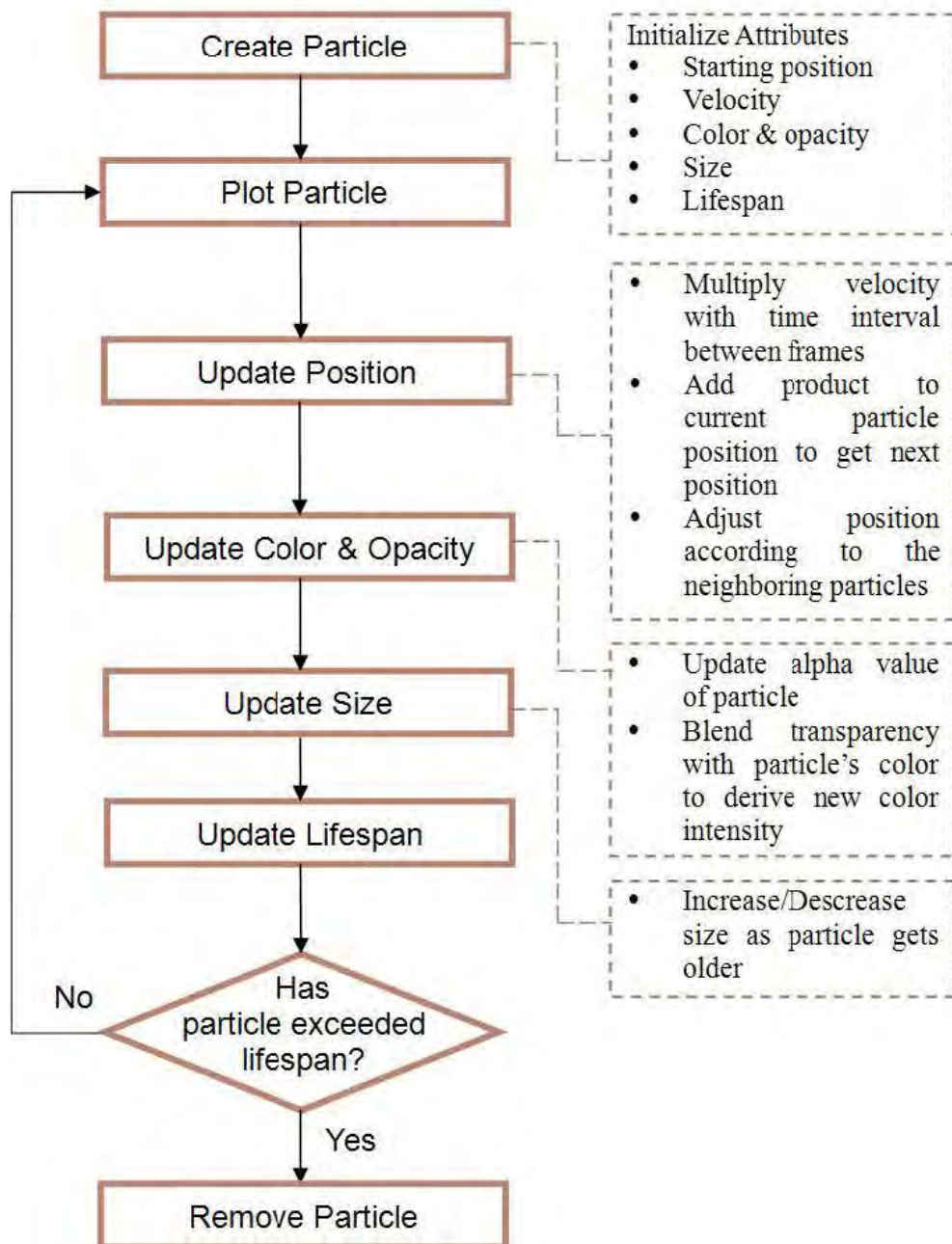


Figure 6.11: A typical life-cycle of a particle in a particle system.

6.4.2.2 Animating fire

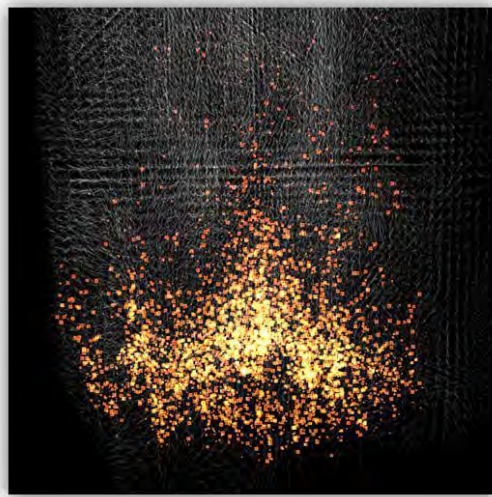
Figure 6.11 shows the flowchart and summarizes behavior of particles in a particle system. It illustrates a typical life-cycle of a particle in a particle system. Once a particle is created and assigned various attributes, it is plotted on the screen. Subsequently, all its attributes are modified and updated step-by-step and replotted on screen. If the particle exceeds its lifespan, it is removed from the system.

The particle dynamics of the fire are simple since each particle independently follows trajectories calculated from our wind field generalization. Moreover, in our fire simulation, wind field advects particles of fire over time.

An outline of an animation loop for creating particle motion for fire is as follows:

- Particles are born, or created, at the base of the cell grid system, within a certain radius of the grid system known as the “fuel radius”.
- Fire particles have a life, after which time they are “reborn”
- Fire particles are displaced away from its initial position using the wind velocity values generated from our noise-based wind field generalization.
- After the transformations, the fire particles are tested against the transformed profiles. If a particle is outside the region, then that particle is not rendered at all.
- The wind field ensures that merged flames behave in a locally similar fashion even though there is nothing in the explicit model to account for merging.
- The particles are then in their final positions, ready for rendering.

Figure 6.12 shows the three dimensional distributions of particles. We trace particles through the wind field to visualize fire.



(a) Points



(b) Lines



(c) Billboards



(d) After texturing

Figure 6.12: Three dimensional distribution of particles according to the wind field. Fire visualizations illustrated by different primitives: points, lines, camera-aligned billboards, billboards with attached texture.



Figure 6.13: Example of billboarding

6.4.3 Rendering fire

Particle system rendering methods usually splat particles onto the screen. Splatting substitutes particles with semi-transparent, camera-aligned rectangles, called billboards. Figure 6.13 illustrates an example for showing how the particles to be rendered.

A particle can be expanded into the quadrangle and regarded as a display primitive. Finally the display primitives seems like a spreading fire effect by attaching a texture on each of it and the blending effect can smooth the edge of the overlapping polygons.

As mentioned in [24], fire has unique visual signatures. Color, geometry, and motion of fire are all essential information for fire recognition. A region of fire can be defined in terms of spectral characteristics and the spatial structure defined by their spectral variation. The pixels in a fire region have different color spectra depending on its spatial locations. For color image, we might see the bright white color in the core, and yellow, orange and red away from the core outer region.

As mentioned earlier, all newly born particles have an initial value assigned to the lifespan parameter and the lifespan will decrease gradually until back to zero. Thus, when implementing the particle system, we also control the size and transparency of a flame particle by using simple rules that depend on the lifespan of the particle as follows:

$$particleSize = initialSize \cdot lifeSpan$$

$$alphaTransparency = 1.0 - lifeSpan$$

It means that both the size and alpha transparency of a particle will gradually decrease over the life of the particle.

6.4.4 Implementation and results

Figure 6.14 illustrates the use of our fire model. Images in this figure contain approximately 5000 particles. As shown in figure, we can generate various types of effects for fire simulation using our noise-based wind field generalization. The simulated results of fire were obtained using a computer with Intel Core (3.0GHz) processor and NVIDIA GeForce 9400 GT video card. It was possible to achieve a realistic looking fire running at around 40 frames per second using 5000 particles. Fewer particles were still capable of achieving realistic results, but only from a more remote distance.

6.4.5 Conclusion

In this section, we have presented a simple model that is able to produce interesting animations of fire. We presented that good results can be obtained without using any physics-based solutions. We developed simple wind field generalization based on fBm noise. The core of our model is applying particle system to the simulation of 3D fire. We think that there are some possibilities for further development. We only take into account the larger scale fire such as campfire or open wildfire. We think that smaller scale fire can be generalized too. Adding new features and parameters makes a better approximation of natural fire. In our work we tried to keep balance between complexity and efficiency. We believe that we managed to achieve it and our method can be attractive for the animation purposes.

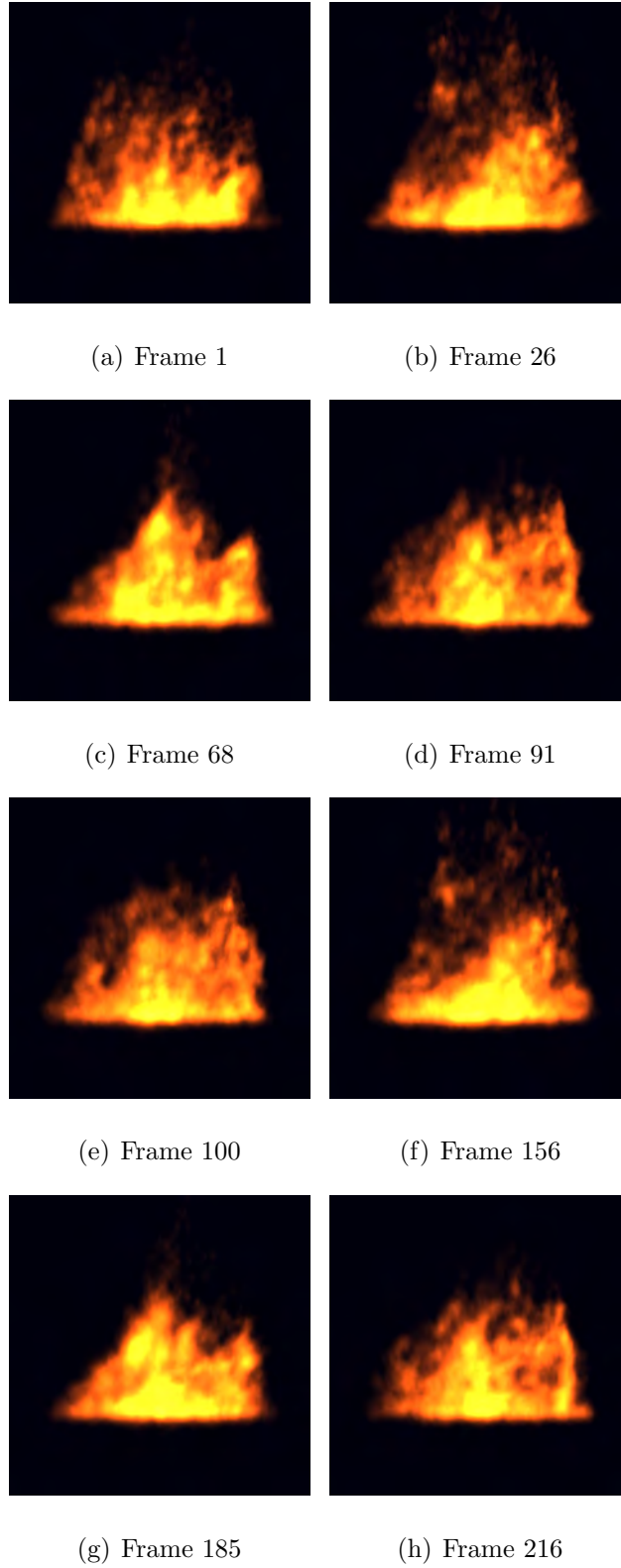
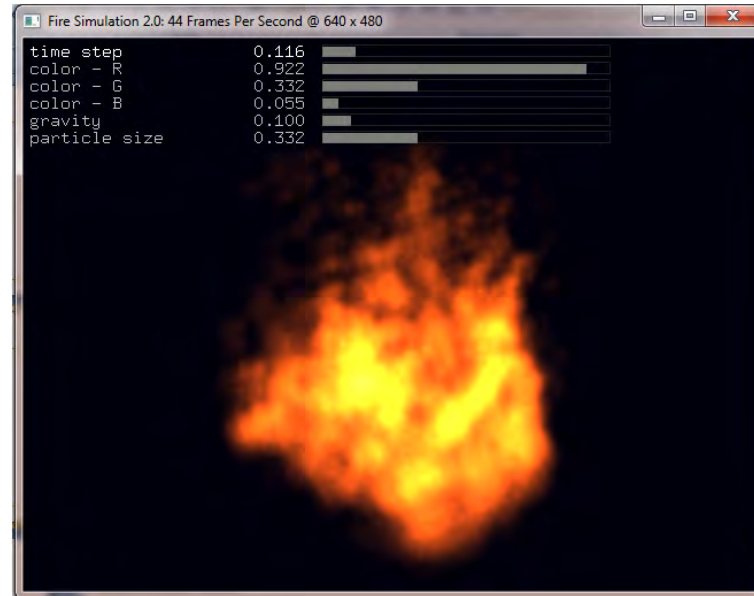
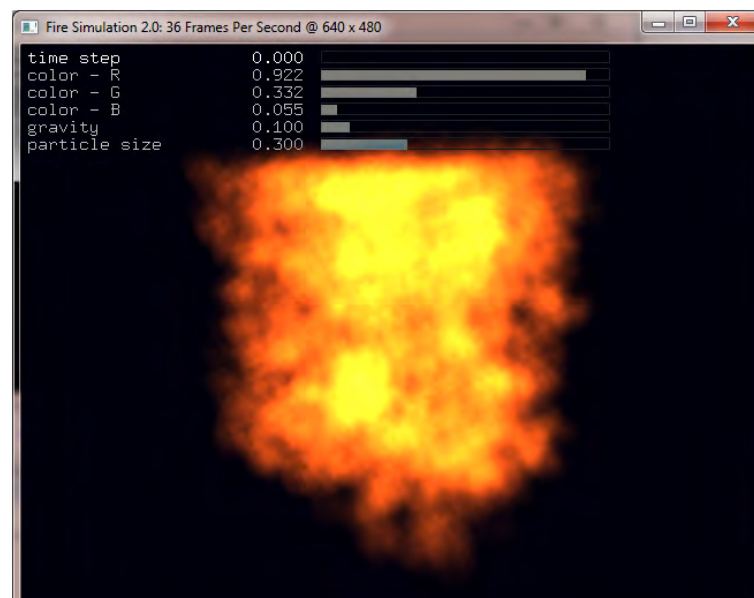


Figure 6.14: Animation sequences of fire simulation by the noise-based wind field generation (sequences are from left to right and top to bottom).



(a) Wind field advects flame particles



(b) With no wind influence

Figure 6.15: A graphical user interface for simulating fire. With or without influence of wind field.



(a)



(b)

Figure 6.16: Fire simulation for different parameter values.

Chapter 7

Conclusion and Future Work

- In computer animations, motion simulation in wind fields is constructed using stochastic approaches or statistical models applied in the field of structural engineering. This is not an ideal approximation of a specific wind with a certain profile because of the varying conditions of the atmosphere and terrain. Thus, better estimation of wind profile parameters is important in wind simulations.
- In this thesis we mainly concerned with the characteristics of wind velocity near the ground. We determined the wind turbulence of the longitudinal wind component based on an examination of the wind spectrum and explored theoretical approaches from structural engineering for evaluating experimental wind profile data. We constructed a wind field with the same characteristics as the real wind measurements that our wind turbulence model was based on, and simulated wind fields exhibiting the motion of grass waving in the wind.
- We also presented an alternative approach to simulate three-dimensional non-frozen turbulent wind fields. By using an fBm-noise based approach with the characteristics of natural wind, wind fields can be constructed in a simple manner with reasonable visual accuracy. Simulated wind fields are described by relatively few parameters. Our approach provides an intuitive description of wind parameters to control wind fields. The implementation results demonstrate that our proposed approach is

competitive with other well-known approaches through the comparison of implementation results and visualizations.

- We showed that our approach is efficient for calculating the forces applied to the objects affected by the wind.
- Finally, we applied our approaches to the animation of many different phenomena directly affected by wind.

Naturally, there are still many areas for future work, both to enhance the quality of the wind field construction and to extend the range of application of the approach with respect to modeling of natural phenomena. Subjects of future work may include the following:

- Flexible noise functions allowing the generation of an extremely varied range of controllable wind fields
- Effective combination of noise-based wind fields with other physically motivated models
- Application of the noise-based non-frozen wind field approach to other phenomena
- A user interface utilizing our wind can also be developed and it can be used in the field of virtual reality

Appendix A

A hot wire anemometer (Model: AM-4204)

A.1 Features

- Thermal anemometer, available for very low air velocity measurement.
- Slim probe, ideal for grilles and diffusers
- Combination of hot wire and standard thermostat, deliver rapid and precise measurements even at low air velocity value.
- Microprocessor circuit
- m/s, km/h, ft/min, knots. mile/h.
- Heavy duty and compact housing case
- Data hold, Memory (Max. and Min.)
- Auto shut off saves battery life.
- RS 232 PC serial interface
- Thermostat sensor for Temperature measurement, fast response time



Figure A.1: A hot wire anemometer.

- Applications: Environmental testing, Air conveyors, Flow hoods, Clean rooms, Air velocity, Air balancing, Fans/motors/blowers, Furnace velocity, Refrigerated case, Paint spray booths.

A.2 RS232 PC Serial Interface

The instrument features an RS232 output via 3.5mm “RS232 Output Socket”.

The connector output is a 16 digit data stream which can be utilized to the user’s specific application.

The 16 digit data stream will be displayed in the following format:

D15 D14 D13 D12 D11 D10 D9 D8 D7 D6 D5 D4 D3 D2 D1 D0

Table A.1: Each digit indicate the following status.

D0	End Word
D1 to D4	Upper Display reading, D1=LSD, D4=MSD
D5 to D8	Lower Display reading, D5=LSD, D8=MSD
D9	Decimal Point (DP) for Upper display. 0 = No DP, 1 = 1 DP, 2 = 2 DP, 3 = 3 DP
D10	Decimal Point (DP) for Lower display. 0 = No DP, 1 = 1 DP, 2 = 2 DP, 3 = 3 DP
D11 & D12	Anunuciator for Upper Display 00 = No Symbol 07 = mg/L 14 = mS 01 = $^{\circ}\text{C}$ 08 = m/s 15 = Lux 02 = $^{\circ}\text{F}$ 09 = Knots 16 = Ft - cd 03 = % 10 = Km/h 17 = dB 04 = % RH 11 = Ft/min 18 = mV 05 = % PH 12 = mile/h 06 = % O2 13 = uS
D13	Anunuciator for Lower Display 0 = No Symbol 1 = $^{\circ}\text{C}$ 2= $^{\circ}\text{F}$
D14	Reading Polarity for the Display 0 = Both upper & lower display value "+". 1 = Upper "-", Lower "+". 2 = Upper "+", Lower "-". 3 = Both upper & lower display value "-"
D15	Start Word

Bibliography

- [1] Procedural noise using sparse gabor convolution. *Comments on Astrophysics*, 7(103), 1978.
- [2] Philippe Beaudoin, Sbastien Paquet, and Pierre Poulin. Realistic and controllable fire simulation. In *Graphics Interface*, pages 159–166, 2001.
- [3] Robert Bridson, Jim Houriham, and Marcus Nordenstam. Curl-noise for procedural fluid flow. *ACM Trans. Graph.*, 26(3):46, 2007.
- [4] Wunsch C. Bermuda sea level in relation to tides, weather, and baroclinic fluctuations. *Rev. Geophys.*, 10:1–49, 1972.
- [5] Yung-Yu Chuang, Dan B. Goldman, Ke Colin Zheng, Brian Curless, David Salesin, and Richard Szeliski. Animating pictures with stochastic motion textures. *ACM Trans. Graph.*, 24(3):853–860, 2005.
- [6] Robert L. Cook and Tony DeRose. Wavelet noise. *ACM Trans. Graph.*, 24(3):803–811, 2005.
- [7] A. G. Davenport. The spectrum of horizontal gushiness near the ground in high winds. *Journal of the Royal Meteorological Society*, 87:194–211, 1961.
- [8] D. Ebert, K. Musgrave, P. Peachey, K. Perlin, and S. Worley. *Texturing and Modeling: A Procedural Approach*. AP Professional, 1994.
- [9] David S. Ebert. *Texturing and modeling: A procedural approach*. Morgan Kaufmann Publishers, Massachusetts, 2002.
- [10] Nick Foster and Dimitris N. Metaxas. Modeling the motion of a hot, turbulent gas. In *SIGGRAPH*, pages 181–188, 1997.

- [11] T. Fujimoto, S. Miyauchi, T. Suzuki, and N. Chiba. Noise-based animation of waving phenomena. In *Proc. IWAIT2005 / IEICE technical report. Image engineering.*, volume 104, pages 93–98, 2005.
- [12] Martin Gardner. White and brown music, fractal curves and one-over-f fluctuations. *Scientific American*, 238(4), 1978.
- [13] Alexander Goldberg, Matthias Zwicker, and Frédo Durand. Anisotropic noise. In *ACM SIGGRAPH 2008 papers*, SIGGRAPH '08, pages 54:1–54:8, New York, NY, USA, 2008. ACM.
- [14] Peitgen H. and Saupe D. *The Science of Fractal Images*. New York : Heinz-Otto and Dietmar Saupe, 1988.
- [15] Ralf Habel, Alexander Kusternig, and Michael Wimmer. Physically guided animation of trees. *Comput. Graph. Forum*, 28(2):523–532, 2009.
- [16] D. Hiriart, J. L. Ochoa, and B. Garcia. Wind power spectrum measuread at the san pedro martir sierra. *Revista Mexicana de Astonomia y Astrofisica*, 37:213–230, 2001.
- [17] J. D. Holmes. *Wind loading of structures*. Spon Press, London, 2001.
- [18] Shaojun Hu, Tadahiro Fujimoto, and Norishige Chiba. Pseudo-dynamics model of a cantilever beam for animating flexible leaves and branches in wind field. *Journal of Visualization and Computer Animation*, 20(2-3):279–287, 2009.
- [19] J. Jang and Y. Lee. A study of along wind speed power spectrum of taiwan area. *Journal of Marine Science and Technology*, 6(1):71–77, 1998.
- [20] J. C. Kaimal, J. C. Wyngaard, Y. Izumi, and O. R. Cote. Spectral characteristics of surface-layer turbulence. *Journal of the Royal Meteorological Society*, 98(417):563–589, 1972.
- [21] Ares Lagae, Sylvain Lefebvre, R. Cook, T. DeRose, George Drettakis, David S. Ebert, J. P. Lewis, K. Perlin, and Matthias Zwicker. A survey of procedural noise functions. *Comput. Graph. Forum*, 29(8):2579–2600, 2010.

- [22] Ares Lagae, Sylvain Lefebvre, George Drettakis, and Philip Dutré. Procedural noise using sparse gabor convolution. *ACM Trans. Graph.*, 28(3), 2009.
- [23] Arnauld Lamorlette and Nick Foster. Structural modeling of flames for a production environment. In Tom Appolloni, editor, *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 2002, San Antonio, Texas, USA, July 23-26, 2002*, pages 729–735. ACM, 2002.
- [24] Che-Bin Liu and Narendra Ahuja. Vision based fire detection. In *ICPR (4)*, pages 134–137, 2004.
- [25] Benoit Mandelbrot. *Fractal Geometry of Nature*. W. H. Freeman, 1977.
- [26] Matthias Müller, David Charypar, and Markus Gross. Particle-based fluid simulation for interactive applications. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation, SCA '03*, pages 154–159, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.
- [27] Duc Quang Nguyen, Ronald Fedkiw, and Henrik Wann Jensen. Physically based modeling and animation of fire. In Tom Appolloni, editor, *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 2002, San Antonio, Texas, USA, July 23-26, 2002*, pages 721–728. ACM, 2002.
- [28] T.E. Nielsen. Modeling, animation, and visualization of fire.
- [29] Tomoyuki Nishita and Yoshinori Dobashi. Modeling and rendering of various natural phenomena consisting of particles. In Horace Ho-Shing Ip, Nadia Magnenat-Thalmann, Rynson W. H. Lau, and Tat-Seng Chua, editors, *Computer Graphics International 2001 (CGI 01), July 3-6, 2001, Hong Kong, China, Proceedings*, pages 149–158. IEEE Computer Society, 2001.
- [30] H. R. Olesen, S. E. Larsen, and J. Hojstrup. Modelling velocity spectra in the lower part of the planetary boundary layer. *Boundary-Layer Meteorology*, 29(3):285–312, 1984.

- [31] Online. Bantam 3d grass application.
- [32] Shin Ota, Machiko Tamura, Tadahiro Fujimoto, Kazunobu Muraoka, and Norishige Chiba. A hybrid method for real-time animation of trees swaying in wind fields. *The Visual Computer*, 20(10):613–623, 2004.
- [33] Ken Perlin. An image synthesizer. In Pat Cole, Robert Heilman, and Brian A. Barsky, editors, *Proceedings of the 12st Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1985*, pages 287–296. ACM, 1985.
- [34] Ken Perlin. Improving noise. *ACM Trans. Graph.*, 21(3):681–682, 2002.
- [35] C. H. Perry and R. W. Picard. Synthesizing flames and their spreading. In *Proceedings of the Fifth Eurographics Workshop on Animation and Simulation*, pages 1–14, 1994.
- [36] Wang Qiang, Zheng Yao, Chen Chun, Fujimoto Tadahiro, and Chiba Norishige. Efficient rendering of breaking waves using mps method *, 2005.
- [37] William T. Reeves. Particle systems - a technique for modeling a class of fuzzy objects. *ACM Trans. Graph.*, 2(2):91–108, 1983.
- [38] K. Saranyasoontorn, L. Manuel, and P. S. Veers. A comparison of standard coherence models for inflow turbulence with estimates from field measurements. *Journal of Solar Energy Engineering*, 126:1069–1082, 2004.
- [39] Mikio Shinya and Alain Fournier. Stochastic motion-motion under the influence of wind. *Comput. Graph. Forum*, 11(3):119–128, 1992.
- [40] E. Simiu. Wind spectra and dynamic along-wind response. *Journal of the Structural Division*, 100(9):1897–1910, 1974.
- [41] E. Simiu and R. H. Scanlan. *Wind effects on structures*. John Wiley and Sons Inc., New York, 1986.
- [42] Jos Stam. Stochastic dynamics: Simulating the effects of turbulence on flexible structures. *Comput. Graph. Forum*, 16(3):159–164, 1997.

- [43] Jos Stam. Stable fluids. In *SIGGRAPH*, pages 121–128, 1999.
- [44] Jos Stam and Eugene Fiume. Turbulent wind fields for gaseous phenomena. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1993*, pages 369–376. ACM, 1993.
- [45] Jos Stam and Eugene Fiume. Depicting fire and other gaseous phenomena using diffusion processes. In *SIGGRAPH*, pages 129–136, 1995.
- [46] Meng Sun, Allan D. Jepson, and Eugene Fiume. Video input driven animation (vida). In *9th IEEE International Conference on Computer Vision (ICCV 2003), 14-17 October 2003, Nice, France*, pages 96–103. IEEE Computer Society, 2003.
- [47] J. Thomas. Advanced character physics.
- [48] L. Verlet. Computer experiments on classical fluids: Thermodynamical properties of lennard-jones molecules. *Phys. Rev.*, 159:98–103, 1967.
- [49] Richard F. Voss. *Fractals in nature: from characterization to simulation*, pages 21–70. Springer-Verlag New York, Inc., New York, NY, USA, 1988.
- [50] Xiaoming Wei, Wei Li, Klaus Mueller, and Arie E. Kaufman. Simulating fire with texture splats. In *IEEE Visualization*, 2002.
- [51] 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.
- [52] Jakub Wejchert and David R. Haumann. Animation aerodynamics. In James J. Thomas, editor, *Proceedings of the 18th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1991*, pages 19–22. ACM, 1991.

List of Figures

2.1	Perlin noise. (a) Perlin's noise function, the first procedural noise functions. (Figure from [34]) (b) Perlin's marble vase, one of the first procedural textures created using Perlin noise. (Figure from [33])	7
2.2	Image starting at the top-left (just noise) and going counter-clockwise, ending at the top right marble image. (a) worn metal, water wave; (b) turbulent flows, fire, marble; (c) turbulent flows, fire, marble, clouds; (d) rock, mountains, clouds (Making noise by Ken Perlin)	7
2.3	Samples of typical noises, $V(t)$, the random variations of a quantity in time. (a) White noise, the most random. (b) $1/f$ -noise, an intermediate but very commonly found type of fluctuation in nature. (c) Brownian motion or a random walk. To the left of each sample is a graphical representation of the spectral density, $S_V(f)$, a measurement characterizing the time correlations in the noise.	9
2.4	(a) Power spectrum of the east-west component of ocean current velocity; the straight line shows the slope of a $1/f$ spectrum. (b) Sea level at Bermuda: this is $1/f^\beta$ spectrum with $\beta = 1.6$. [4]	10
2.5	Loudness (left) and pitch (right) fluctuation spectra vs. frequency (Hz) (log-log scale), for (a) Scott Joplin piano rags; (b) classical radio station; (c) rock station; (d) news-and-talk station.	11
2.6	$1/f^\beta$ noise via spectral synthesis. The above curves correspond to spectral density functions of the form $1/f^\beta$ where $\beta = 2H+1$, $D = 2 - H$	13

2.7	Fractal clouds with varying dimensions. The above clouds correspond to spectral density functions of the form $1/f^\beta$ where $\beta = 2H + 2$, $D = 3 - H$	14
2.8	Fractal clouds with varying dimensions. The above clouds correspond to spectral density functions of the form $1/f^\beta$ where $\beta = 2H + 3$, $D = 4 - H$	15
3.1	Atmospheric boundary-layer profiles (plots of average wind speed U versus height z) over different terrains. Wind speeds are expressed as percentages of the upper level wind (referred to as the gradient wind) above the boundary (or surface) layer. . .	18
4.1	General flow diagram of our approach	24
4.2	Time histories for sample cases	25
4.3	Power density spectra for sample cases	26
4.4	Normalized power spectra for sample cases compared with spectra of Kaimal and Simiu	27
4.5	(Left) Proposed spectrum with spectrum of Kaimal and Simiu and the original wind spectrum for each sample case; (Right) Simulated winds for each sample case	29
4.6	Steps in wind field simulation: (Top) actual wind history, (Middle) result of low-pass filtering, and (Bottom) simulated winds at two different locations	33
4.7	Simulated wind fields for sample Case 1 when $M = 16$ and $N = 16$ (Turbulence travels in the mean wind direction). . . .	34
4.8	Ground covers created using Bantam3D Grass application. . .	35
4.9	Grass field under different wind forces.	38
4.10	Screenshots of grass field animation showing the effects of wind turbulence traveling along in the mean wind direction.	39
4.11	Different grassland scenes under wind effects.	40
5.1	Measured wind speed history data in the along-mean-wind direction at a point in space and the corresponding power spectrum	43
5.2	A graphical representation of our proposed approach	47
5.3	Simulated winds at different parameter values	50
5.4	Algorithm overview of our proposed method for generating non-frozen 3D wind field.	51

5.5	Three-dimensional non-frozen turbulent wind fields 32^4 . Screenshots of simulated three-dimensional wind velocity fields. (Left) by our approach. (Right) by Stam and Fiume's approach. . . .	52
5.6	2D frozen turbulent wind fields (64×64). (Left) Screenshots of simulated 2D wind velocity fields by our approach. (Right) Screenshots of simulated 2D wind velocity fields by Shinya and Fournier's approach.	53
5.7	The running time of generating wind fields. Measured in milliseconds and obtained with Intel Core (3.0GHz)	54
6.1	Sample curves of a blade of grass	56
6.2	A process for simulating grass field animation. (a) 2D turbulent frozen wind field generated by our approach (b) grass field is built by a simple Bezier curve (c) grass field animation.	57
6.3	Top views of grass fields generated by the corresponding 2D frozen turbulent wind fields. (a) Grass field generated by our approach (b) Grass field generated by Shinya and Fournier's approach.	58
6.4	Three-dimensional non-frozen turbulent wind fields ($64 \times 32 \times 32 \times 64$). Visualization by our approach.	59
6.5	Three-dimensional non-frozen turbulent wind fields ($64 \times 32 \times 32 \times 64$). Visualization by Stam and Fiume's approach	60
6.6	Animation sequences of cloth like objects in the 3D non-frozen wind field (sequences from top to bottom and left to right) . .	64
6.7	(a) Wall of fire from Star Trek II from Reeves et al. (b) The sample image from Nguyen et al.	66
6.8	(a) Image from Shrek modeled by Lamorlette et al. (b) A simple campfire using texture splats from Wei et al.	67
6.9	Basic idea of modeling fire	68
6.10	Emitters for different effects	69
6.11	A typical life-cycle of a particle in a particle system.	70
6.12	Three dimensional distribution of particles according to the wind field. Fire visualizations illustrated by different primitives: points, lines, camera-aligned billboards, billboards with attached texture.	72
6.13	Example of billboarding	73

6.14	Animation sequences of fire simulation by the noise-based wind field generation (sequences are from left to right and top to bottom).	75
6.15	A graphical user interface for simulating fire. With or without influence of wind field.	76
6.16	Fire simulation for different parameter values.	77
A.1	A hot wire anemometer.	81

List of Tables

3.1	Beaufort wind force scale.	21
3.2	Typical values of surface roughness length z_0 and power law exponent α , for various types of terrain.	22
4.1	Wind data measurements for Morioka area of Japan	25
4.2	Estimated values of parameters.	28
4.3	Parameters for Figure 4.8.	36
5.1	Performance results of 2D frozen turbulent wind fields.	54
5.2	Performance results of 3D non-frozen turbulent wind fields.	54
A.1	Each digit indicate the following status.	82