EUROGRAPHICS '97 / D. Fellner and L. Szirmay-Kalos
(Guest Editors)

Volume 16, (*1997*), *Number 3*

Stochastic Dynamics: Simulating the Effects of Turbulence on Flexible Structures

Jos Stam†

INRIA, Rocquencourt
Le Chesnay, France

VTT, Information Technology
Espoo, Finland

Alias—Wavefront
Seattle, Washington

Abstract

This paper addresses the problem of realistically simulating the motion of tree-branches subjected to turbulence. Since the resulting motion is random in nature, we model it as a stochastic process. We synthesize this process directly by filtering a white noise in the Fourier domain. The filter is constructed by performing a modal analysis of the tree. We use a sophisticated numerical technique which is able to compute the first few significant modes of large trees. The main advantage of our technique over previous methods is that we are able to compute complicated motions without the necessity of integrating dynamical equations over time. Consequently, a user can view and manipulate tree-motions in real-time Our technique can be further extended to other flexible structures such as two-dimensional plates.

1. Introduction

Many motions encountered in Nature are caused by atmospheric turbulence. Everyday examples of such motions include the swirling behaviour of rising smoke and the swaying of tree-branches in the wind. This paper will address the problem of modelling the latter effect. Since this phenomenon is difficult to key frame directly, researchers usually resort to physics-based methods[9]. In these methods, the motion of the branches is computed by integrating the relevant dynamical equations over time. Since the forces driving these equations result from turbulent wind fields, we observe that the corresponding motions of the trees are random in nature. This observation forms the basis of our method, that of modelling the movement of the tree as a stochastic process. This process is synthesized directly by filtering an uncorrelated noise in the frequency domain. A filter is constructed by computing a small number of deformations characterizing the tree. This is done using a technique known in engineering as *modal analysis* and was used by Pentland and Williams to animate and control simple flexible objects[3]. Our method however differs from their work since we consider arbitrary large objects which are driven by a random process.

The closest model to ours is that of Shinya and Fournier[7]. But a fundamental difference is that they must integrate dynamical equations over time. Where they synthesize the wind field to drive the motion, we synthesize the motion directly, bypassing the need to integrate. Simply put, where Shinya and Fournier model the cause, we model the effect stochastically. By using a spectral synthesis algorithm, we can precompute periodic motions for a given class of trees. These motions therefore are defined for any instant of time and can be simulated by a straightforward table lookup. This has several advantages. Firstly, the motion of the tree can be manipulated in real-time by an animator. Secondly, specific frames of an animation can be computed directly without the necessity of resimulating everything from the beginning. Thirdly, a single motion can be multiplied to similar trees to create a "wind in the forest" effect without having to solve for each individual tree. In short, our method enables an an-

† This research was conducted while the author was an ERCIM postdoctoral fellow at INRIA and VTT. Author's current address: Alias—Wavefront, 1218 3rd Avenue, Ste 800, Seattle, WA 98101, U.S.A.

imator to almost effortlessly add realistic motion to many trees.

However, we would like to emphasize the value of Shinya and Fournier's approach[7] in that they addressed a wider range of problems pertaining to the realistic simulation of trees, particularly the rendering and animation of foliage. This method has been improved recently by Weber and Penn[10]. As well, we do not address the problem of synthesizing the geometry of various trees, since this has been fully explored by de Reffye et al and Prusinkiewicz, among others[1, 5].

In order to shorten the presentation of the material, we use a compact vector notation throughout the paper. Boldface lowercase letters refer to vectors while boldface uppercase refer to matrices. The components of a vector are identified by their corresponding letter: the i-th component of a vector $\mathbf{u}$ is denoted by u_i, which should not be confused with an indexed vector: $\mathbf{u}_i$. The $n \times n$ identity matrix is denoted by $\mathbf{I}_n$. Transposition is denoted by appending a "^{T}" symbol, while the "*" symbol is reserved for transposition and conjugation.

2. Dynamics of Trees

We discretize our trees into a set of N branches. Each branch at rest is defined by both a line segment and by its radii at each endpoint. To generate such descriptions, we employ simple L-systems[5]. Fig. 3 to 6 depict trees synthesized in this manner. By considering elastic deformations only, we can model the motion of each branch over time. The corresponding deviation from the branch segment is described by a cubic curve, whose shape depends on the six displacements at each of the end points of the branch. These displacements result from translational and rotational forces. We denote the displacements at a node i of the tree by a six-dimensional vector $\mathbf{u}_i$. The displacement associated with each component

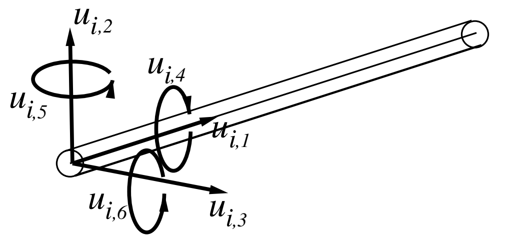

Figure 1: *The six displacements at the end-point of a branch.*

of $\mathbf{u}_i$ is depicted in Fig. 1. The actual shape of a branch, labelled k, is then given by the following cubic curve:

$$\mathbf{x}_k(s) = \mathbf{x}_k^0(s) + \mathbf{A}_k(s) \begin{pmatrix} \mathbf{u}_{i_k} \\ \mathbf{u}_{j_k} \end{pmatrix}, \quad s \in [0,1], \qquad (1)$$

where $\mathbf{x}_k^0$ denotes the branch segment at rest. The 12×3 matrix $\mathbf{A}_k(s)$ is provided in Appendix A. The integers i_k and j_k are used to index the displacements corresponding to the end points and are values between 0 and N. The displacement $\mathbf{u}_0$ represents the root of the tree and is therefore zero.

By grouping all of the displacements into a $6N$-dimensional vector $\mathbf{u}$, the evolution of the displacements can be condensed into the following dynamical equation[8]:

$$\mathbf{M}\ddot{\mathbf{u}}(t) + \mathbf{C}\dot{\mathbf{u}}(t) + \mathbf{K}\mathbf{u}(t) = \mathbf{f}(t), \qquad (2)$$

where $\mathbf{M}$, $\mathbf{C}$ and $\mathbf{K}$ are the mass, damping and stiffness matrices respectively. The force $\mathbf{f}$ accounts for the external loading due to wind. The dampness matrix is often assumed to be proportional to the mass matrix via a damping coefficient α, i.e., $\mathbf{C} = \alpha\mathbf{M}$[8]. In Appendix A, stiffness and mass matrices are given for a single branch. The global matrices are

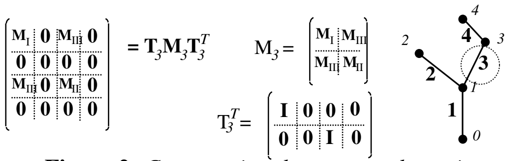

Figure 2: *Constructing the structural matrices.*

built from those of each individual branch by adding the four 6×6 subblocks to the corresponding ones in the global matrix. Fig. 2 illustrates how this is achieved for a simple tree. More precisely, let $\mathbf{T}_k$ be a $12 \times 6N$ matrix which is zero except for two 6×6 subblocks that are determined by i_k and j_k and are equal to the identity matrix. If $\mathbf{K}_k$ denotes the stiffness matrix for the k-h branch, then the global stiffness matrix is equal to[6]

$$\mathbf{K} = \sum_{k=1}^{N} \mathbf{T}_k \mathbf{K}_k \mathbf{T}_k^T.$$

We proceed similarly for the mass matrix.

The force vector is constructed in the same manner from the loads $\mathbf{f}_k$ on each individual branch caused by a wind field $\mathbf{v}(\mathbf{x},t)$. Indeed, the pressure $\mathbf{p}_k(t,s)$ acting on a single branch k is proportional to the square of the magnitude of the velocity. For small displacements, it is often assumed that this relationship is linear[8]:

$$\mathbf{p}_k(t,s) \approx \rho C_D r_k(s) \mathbf{P}_k \mathbf{v}(\mathbf{x}_k^0(s),t), \qquad (3)$$

where $\mathbf{P}_k$ is an operator which projects the velocity onto the plane normal to the direction of the branch, and where $r_k(s)$ is the thickness along the branch. The density of the air, ρ, is approximately equal to $1.2kg/m^3$ and C_D denotes the lateral drag coefficient. The force acting on a branch is obtained by inverting Eq. 1 and integrating along the branch:

$$\mathbf{f}_k(t) = \int_0^1 \mathbf{A}_k(s)^T \mathbf{p}_k(t,s)\, ds \approx \mathbf{B}_k \mathbf{p}_k(t,0.5) \qquad (4)$$

where $\mathbf{B}_k$ is a 12×3 matrix equal to the transpose of $\mathbf{A}$ integrated over the length of the branch (see Appendix A). The global force is calculated by adding up the contributions from each individual branch:

$$\mathbf{f}(t) = \rho C_D \sum_{k=1}^{N} r_k(0.5)\mathbf{T}_k\mathbf{B}_k\mathbf{P}_k\mathbf{v}_k(t), \qquad (5)$$

where $\mathbf{v}_k(t) = \mathbf{v}(\mathbf{x}_k(0.5),t)$. Now that we have shown how all the terms appearing in Eq. 2 are related to the geometry of the tree, we investigate its solution.

3. Modal Analysis

Instead of integrating Eq. 2 over time, we transform this Equation into the frequency domain (note that all quantities become complex valued in the Fourier domain.):

$$\mathbf{L}(\omega)\hat{\mathbf{u}}(\omega) = \hat{\mathbf{f}}(\omega), \qquad (6)$$

where "$\hat{}$" denotes the corresponding Fourier transform and the linear operator

$$\mathbf{L}(\omega) = (-\omega^2 + \mathrm{i}\alpha\omega)\mathbf{M} + \mathbf{K}$$

($\mathrm{i} = \sqrt{-1}$). From this equation it is evident that Eq. 2 can be solved directly if the matrix $\mathbf{L}$ is invertible for any ω. Usually this inverse is difficult to obtain for trees since the size of $\mathbf{L}$ can become very large (six times the number of branches).

A convenient way of solving such equations for flexible structures is to compute the modal frequencies and shapes associated with the tree. A flexible structure, in general, displays certain characteristic shapes when subjected to a force vibrating at one of its modal frequencies. The shape resulting from an arbitrary force can be regarded as a superposition of the modal shapes. In practice, only a small number of the modal shapes corresponding to the smallest modal frequencies contribute to the motion of the tree. Mathematically, the modal frequencies and shapes are calculated by solving the following eigenproblem[8]:

$$\mathbf{MQD} = \mathbf{KQ},$$

where $\mathbf{D}$ is a diagonal matrix whose non-zero elements ω_i^2 are the modal frequencies of the tree, while the columns of the matrix $\mathbf{Q}$ correspond to the modal shapes. Since the characteristics of the modal shapes are left invariant under rotations and scalings, it is possible to require that they are mass-orthogonal: $\mathbf{Q}^T\mathbf{MQ} = \mathbf{I}_{6N}$. Consequently, the operator $\mathbf{L}$ becomes diagonal in the modal basis:

$$\mathbf{QLQ}^T = (-\omega^2 + \mathrm{i}\alpha\omega)\mathbf{I}_{6\mathrm{N}} + \mathbf{D}.$$

Let us denote the displacement and the force in this new basis by $\underline{\hat{\mathbf{u}}}$ and $\underline{\hat{\mathbf{f}}}$, respectively. The i-th displacement of this new basis can then be computed directly by inverting the diagonal system:

$$\underline{\hat{u}}_i(\omega) = \frac{1}{(\omega_i^2 - \omega^2) + \mathrm{i}\alpha\omega}\,\underline{\hat{f}}_i(\omega), \quad i = 1,\cdots,6N. \qquad (7)$$

Specifically, the contribution of each mode is inversely proportional to the value of each modal frequency. Thus only a small number $m \ll N$ of modal frequencies is sufficient to characterize the resulting deformations. Assuming that the diagonal elements of $\mathbf{D}$ are sorted in increasing order, then in practice we only have to compute the first m columns $\mathbf{Q}_m$ of $\mathbf{Q}$. Once the displacements $\underline{\hat{\mathbf{u}}}$ in the modal basis have been calculated, we perform m inverse Fourier transforms in order to obtain the displacements $\underline{\mathbf{u}}$ in the temporal domain. Finally, we retransform this vector back to get the final displacements which will be used in the simulation: $\mathbf{u} = \mathbf{Q}_m\underline{\mathbf{u}}$.

4. Solving the Eigenproblem

We observe that for this method to work, we require an efficient algorithm that solves the eigenproblem. We have experimented with two different algorithms. The first one is from the Numerical Recipes library: routines `tred2` and `tqli`[4]. As explained on p. 462 of the reference, the eigenproblem is first transformed into a standard eigenproblem by using a Choleski decomposition of the mass matrix. The standard problem is then solved by tridiagonalizing the matrix of the system. Then a routine dedicated to the solution of such systems is called. Because this method calculates all the modal frequencies and shapes, it is expensive: $O(N^3)$. This is an overkill. In practice, we find that using only ten or twenty modal frequencies is sufficient.

Fortunately, there exist many other methods which iteratively calculate the modal frequencies in ascending order, taking advantage of the sparsity of the mass and stiffness matrices. We chose the *Lanczos* method because of recommendations from many sources[8, 2]. The Lanczos method iteratively computes only the $m \times m$ upper left block of the tridiagonal system and in fact, this upper block contains all the corresponding lowest modal frequencies. The eigenvalues of the truncated tridiagonal matrix are computed with a routine similar to `tqli`. Many implementations of the Lanczos algorithm are available in the public domain. We implemented the "lanz" package written by Jones and Patrick available from `netlib` (http://www.netlib.org/lanz/index.html). This package is written in FORTRAN but was easy to link with the rest of our code which is written in C. The routine has many parameters which must be set appropriately in order to obtain good results. A very helpful option is to consider only a sparse (approximate) Choleski decomposition of the mass matrix. This option greatly decreased computing time. The package also prints out the corresponding error bounds on each eigenvalue. By asking for about double the amount of modal frequencies than is required, we can get a precision up to machine accuracy. These findings are consistent with numerical experiments[2].

Knowing that we can invert Eq. 6 efficiently, it remains to compute the Fourier transform of the force on the right hand side of Eq. 6. This is the topic of the subsequent section.

5. Stochastic Dynamics

If the Fourier transform of the force is known exactly, we could solve Eq. 6 directly. In reality, however, only the aggregate behaviour of atmospheric turbulence is known. We therefore must consider all the functions in the previous section to be stochastic processes. For *stationary* processes, i.e., processes whose statistics are time shift invariant, spectral representations exist. Therefore, all the results of the previous section apply. In practice, we approximate the Fourier transform of the generalized force by a (truncated) Fourier series of M terms. The (vector) coefficients of this series are computed from the velocity at each branch via Eq. 5. This requires us in turn to have a realization of a random wind field. One solution would be to generate a random vector independently at each branch. Real wind fields, which are albeit random, exhibit spatial structure. So, we expect that nearby branches in the tree should receive similar loads. We achieve spatial correlations by filtering a set of uncorrelated complex random vectors, $\mathbf{w}_1, \cdots, \mathbf{w}_N$, by a set of filters $\mathbf{H}_{kl}$:

$$\hat{\mathbf{v}}_k(\omega) = \sum_{l=1}^{N} \mathbf{H}_{kl}(\omega)\mathbf{w}_l(\omega), \quad k = 1, \cdots, N. \tag{8}$$

Since a wind field has finite energy and is usually coherent over time, we expect these filters to decay as the frequency increases. To account for spatial correlations, we make the magnitude of the filter inversely proportional to the distance, d_{kl}, between branches k and l. A simple formula which includes both behaviours is given by:

$$\mathbf{H}_{kl}(\omega) = \exp\left(-t_c\omega\frac{d_{kl}}{d_c}\right)\mathbf{I}_3,$$

where t_c and d_c represent typical temporal and spatial correlations of the wind field. The multiplication by the identity matrix reflects that the components of the wind field are uncorrelated amongst themselves. For small correlation lengths, typically comparable to the size of a branch, most filters are zero. Consequently, the sum in Eq. 8 usually has a number of non-zero terms which are independent of the size of the tree. The cost of computing the turbulent wind field at the branches is therefore essentially linear in N.

The force is now determined in the frequency domain, since the force is related to the velocity just computed via Eq. 5. Finally, the displacements are computed in the modal basis via Eq. 7.

6. Putting it all Together

We now give the complete algorithm which generates the stochastic displacements from the previous considerations. On input, the user provides the length T of the periodic time interval, the number M of discrete frequencies included in the Fourier series, the correlation parameters t_c and d_c of the filters, the damping factor α and the geometry of the tree.

Compute Matrices:
  $\mathbf{K} = \mathbf{M} = \mathbf{0}$
  for $k \in \{1, \cdots, N\}$ do
    Compute $\mathbf{K}_k$ and $\mathbf{M}_k$ (see Appendix A)
    $\mathbf{K} = \mathbf{K} + \mathbf{T}_k^T\mathbf{K}_k\mathbf{T}_k$ and $\mathbf{M} = \mathbf{M} + \mathbf{T}_k^T\mathbf{M}_k\mathbf{T}_k$
  end do
Solve Eigenproblem:
  Call `lanz` $\mathbf{K},\mathbf{M} \longrightarrow \mathbf{Q}_m$ and $\omega_1^2, \cdots, \omega_m^2$
Compute Force:
  for $p \in \{0, \cdots, M/2\}$ do
    $\omega = \frac{p}{T}$
    $\hat{\mathbf{f}} = \mathbf{0}$
    generate N random vectors $\mathbf{w}_l(\omega)$
    for $k \in \{1, \cdots, N\}$ do
      compute $\hat{\mathbf{v}}_k(\omega)$ from Eq. 8
      $\hat{\mathbf{f}} = \hat{\mathbf{f}} + \rho C_D r_k(0.5)\mathbf{T}_k\mathbf{B}\mathbf{P}_k\hat{\mathbf{v}}_k(\omega)$
    end
    $\underline{\hat{\mathbf{f}}} = \mathbf{Q}_m^T\hat{\mathbf{f}}$
    Compute $\underline{\hat{\mathbf{u}}}_p$ using Eq. 7
    if $p \neq 0$ then $\underline{\hat{\mathbf{u}}}_{M-p} = \underline{\hat{\mathbf{u}}}_p^*$
  end do
  $\mathrm{Im}\{\underline{\hat{\mathbf{u}}}_0\} = \mathrm{Im}\{\underline{\hat{\mathbf{u}}}_{M/2}\} = \mathbf{0}$
  $\{\underline{\mathbf{u}}_0, \cdots, \underline{\mathbf{u}}_{M-1}\} = \mathrm{invFFT}\{\underline{\hat{\mathbf{u}}}_0, \cdots, \underline{\hat{\mathbf{u}}}_{M-1}\}$
  $\mathbf{u}_p = \mathbf{Q}_m\underline{\mathbf{u}}_p, \quad p = 0, \cdots, M-1$

The bottleneck of the algorithm is the solution to the eigenproblem having a theoretical complexity of $O(N^3)$. However, in our simulations the time required to solve the eigenproblem was much smaller than the time required to compute the force. The computation of the latter is $O(mNM) = O(MN)$, since $m \ll N, M$ and as noted above, the sum in Eq. 8 has a constant number of non-zero terms.

7. Results

We have written a simulator which, when given a precomputed displacement table and a geometric description of a tree, displays the evolution of the tree in real-time. Displacements for times falling in between the time samples computed by the fast Fourier transform are linearly interpolated from neighbouring samples. We can either display the trees by drawing the curves defined by Eq. 1 or generate cones which can then be hardware-shaded on an Iris Indigo. For our simulations, we used a time resolution of $M = 256$ and a time interval of $T = 256$ seconds. We employed the Numerical Recipes routines on small matrices to determine how many model frequencies are sufficient. For small trees (less than 50 branches), retaining only $m = 10$ modes results in simulations which are visually indistinguishable from simulations of the same tree including all modes.

When using the Lanczos method, we usually requested 40 modes and employed only 20, which were all accurate up to machine precision. Table 1 shows the computation times on an Iris Indigo with a 150 MHz R4400 processor for the trees depicted in Fig. 3. As is evident, the computation of the force is of an order of magnitude higher than the eigenproblem solution. To remedy this problem, we attempted to generate the

Nb. of Branches	2257	692	212	62	17
`lanz` (sec.)	48.8	10.1	5.5	0.5	0.2
Gen. noise (sec.)	451.4	145.3	37.5	3.4	0.3

Table 1: *Computation times for trees with different numbers of branches.*

m modal forces $\hat{\underline{\mathbf{f}}}$ directly, since in the case of low damping, Eq. 7 essentially selects frequencies close to the modal ones only. In practice, this has turned out to be effective only for small trees. For larger trees, unnatural displacements along the tree's length occurred. Therefore, in this case, it is generally necessary to go through the entire force computation.

In Fig. 3 through 6 we show frames of animations which were generated using our method. Fig. 4 is an animation of a ray-traced tree with 55 branches. In Fig. 5, we used the same displacement computation to animate several similar trees. By using a different time-shift we can "desynch" the motion of each tree. This permits us to animate large amounts of trees using virtually no extra computation time. Fig. 6 depicts an underwater scene with algae-like structures. We increased the damping in order to simulate the characteristic motion of underwater plants. Please view the animations on the CDROM accompanying this paper for a better appreciation of the results.

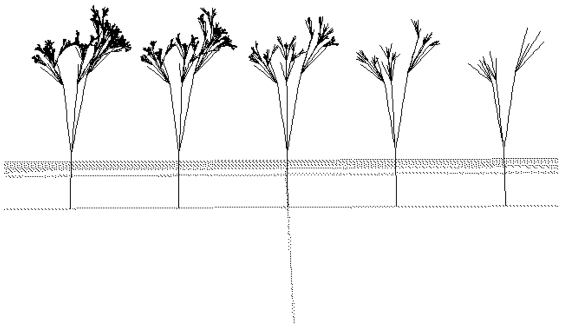

Figure 3: *The five trees used for the timings given in Table 1.*

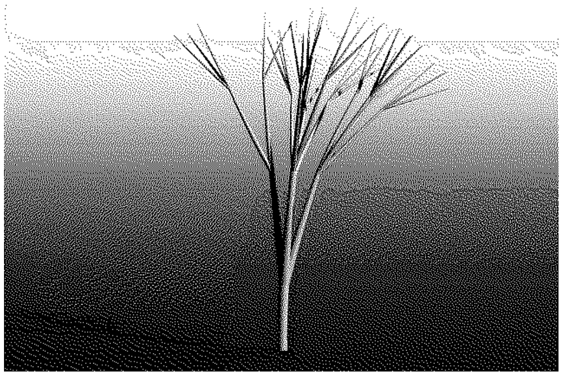

Figure 4: *One frame from an animation of a small tree swaying in the wind.*

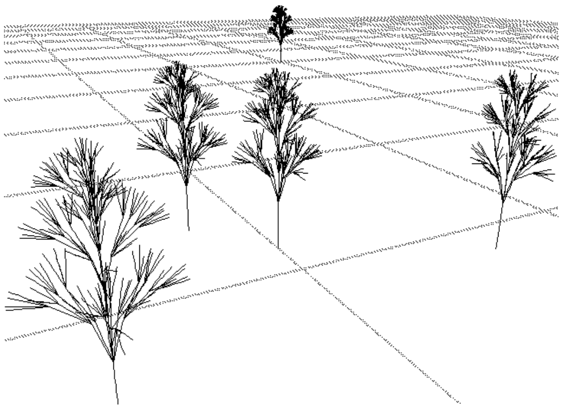

Figure 5: *Forest: the motion computed for a single tree is applied to many trees.*

8. Conclusions

The results demonstrate that despite the assumption of small displacements required to maintain a linear description, we are able to synthesize convincing motions of tree-branches swaying in the wind. The same methodology can in fact be applied to other flexible elements. Refer to Fig. 7, four frames of an animation of a flag waving in the wind, which was generated using the same methodology. The corresponding structural elements and displacements can be found in[6]. However, once we encounter larger deflections, a method based solely on elastic deformations breaks down, and geometrical deformations take over. Larger deformations therefore require more expensive techniques.

9. Acknowledgements

Thanks to the European Research Consortium for Informatics and Mathematics (ERCIM) for their financial support. Thanks to the Projet Syntim at INRIA for their stimulating research environment where most of this research was conducted. Finally, thanks to Pamela Jackson for improving the writeup of this paper.

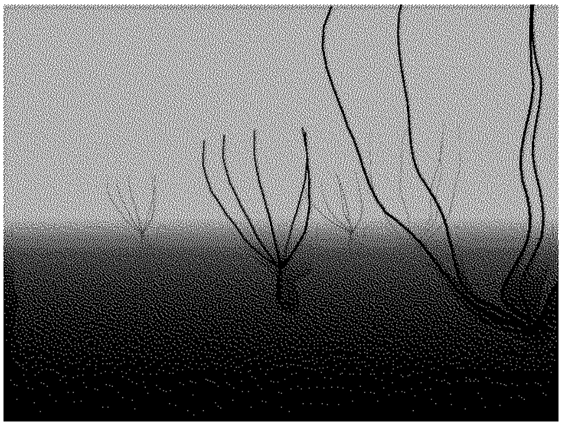

Figure 6: *By increasing the aerodynamical damping we can model the movements of underwater plants.*

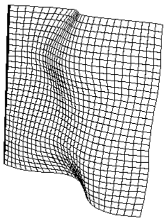

Figure 7: *The methodology described in this paper can also be applied to other flexible structures. A "flag-like" structure, for example.*

Appendix A: Structural Elements

In this appendix we present the exact form of the matrices pertaining to a single branch k modelled as a flexible beam. These matrices are usually given in a local coordinate system where the x-axis coincides with the direction of the branch. The matrices must therefore be transferred back into global coordinates. Let $\mathbf{r}_k$ be the 3×3 matrix which performs the transformation of local to global coordinates. From this matrix we construct the 12×12 transformation, whose diagonal blocks are equal to $\mathbf{r}_k$:

$$\mathbf{R}_k = \begin{pmatrix} \mathbf{r}_k & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{r}_k & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{r}_k & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{r}_k \end{pmatrix}.$$

Using this transformation, the structural matrices in global coordinates are

$$\mathbf{K}_k = \mathbf{R}_k \bar{\mathbf{K}} \mathbf{R}_k^T \quad \mathbf{M}_k = \mathbf{R}_k \bar{\mathbf{M}} \mathbf{R}_k^T$$
$$\mathbf{A}_k(s) = \mathbf{r}_k \bar{\mathbf{A}}(s) \mathbf{R}_k^T \quad \mathbf{B}_k = \mathbf{R}_k \bar{\mathbf{B}} \mathbf{r}_k^T.$$

The matrices $\bar{\mathbf{K}}$, $\bar{\mathbf{M}}$ and $\bar{\mathbf{A}}$ can be found in reference[6]. The stiffness and inertia matrices depend on the following physical parameters: Young's modulus $E = 10^8 N/m^2$, shear modulus $G = 10^5 N/m^2$ and the mass density $\rho = 10^3 kg/m^3$, while the following quantities are a function of the beam's average width r of the branch: area $A = \pi r^2$, moment of inertia $I = \pi r^4$ and moment of gyration $J = \frac{1}{2}\pi r^4$. Since the matrix $\bar{\mathbf{B}}$ is not given explicitly in[6], it is provided next

$$\bar{\mathbf{B}} = (\mathbf{B}_1 \ \ \mathbf{B}_2 \ \ \mathbf{B}_1 \ \ -\mathbf{B}_2)^T,$$

where (l = length of the branch)

$$\mathbf{B}_1 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & l/2 & 0 \\ 0 & 0 & l/2 \end{pmatrix} \quad \mathbf{B}_2 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & l^2/12 \\ 0 & -l^2/12 & 0 \end{pmatrix}.$$

References

1. P. de Reffye, C. Edelin, J. Francon, M. Jaeger, and C. Puech. "Plant Models Faithful to Botanical Structure and Development". *ACM Computer Graphics (SIGGRAPH '88)*, 22(4):151–158, August 1988.
2. G. Gambolati. *Solution of Large Eigenvalue Problems*. John Wiley & Sons Ltd, 1993.
3. A. Pentland and J. Williams. "Good Vibrations: Modal Dynamics for Graphics and Animation". *ACM Computer Graphics (SIGGRAPH '89)*, 23(3):215–222, July 1989.
4. W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes in C. The Art of Scientific Computing*. Cambridge University Press, Cambridge, 1988.
5. P. Prusinkiewicz, A. Lindenmayer, and J. Hanan. "Developmental Models of Herbaceous Plants for Computer Imagery Purposes". *ACM Computer Graphics (SIGGRAPH '88)*, 22(4):141–150, August 1988.
6. J. S. Przemieniecki. *Theory of Matrix Structural Analysis*. McGraw-Hill Book Company, New York, 1968.
7. M. Shinya and A. Fournier. "Stochastic Motion - Motion Under the Influence of Wind". In *Proceedings of Eurographics '92*, pages 119–128, September 1992.
8. J. W. Smith. *Vibration of Structures. Applications in Civil Engineering Design*. Chapman and Hall, London, England, 1988.
9. D. Terzopoulos, J. Platt, and K. Fleisher. "Elastically Deformable Models". *ACM Computer Graphics (SIGGRAPH '87)*, 21(3):205–214, July 1987.
10. J. Weber and J. Penn. "Creation and Rendering of Realistic Trees". In *Computer Graphics Proceedings, Annual Conference Series, 1995*, pages 119–128, August 1995.