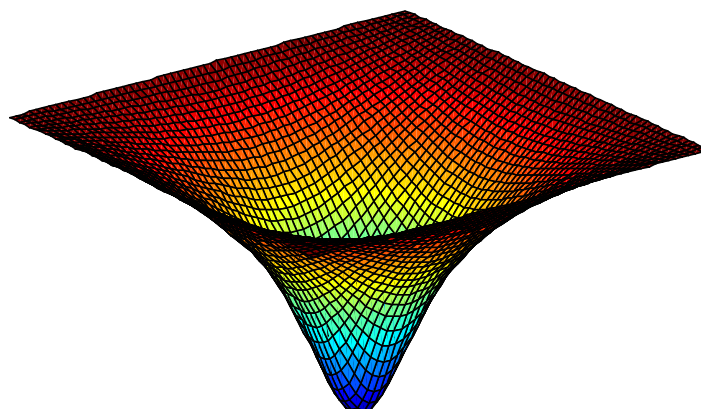


Vibrating membranes 7

A project in the course acoustics, TNM027,
at Linköpings Technical University.
Year 2004.



Authors: Strandell, Ebbe
Uddholm, Mikael

Examiner: Miklavcic, Stan

Executive summary

The aim of this report is to provide the reader a solution to the wave equation for free vibrations of square membranes. This solution can be used to simulate vibrating membranes such as percussions or loudspeaker cones. Part of the project was to simulate vibrating membranes, included in this report are discussions about three possible ways to simulate vibrations of membranes and each methods advantages and disadvantages.

Table of contents

1. Introduction	3
2. Method	3
2.1 The wave equation	3
2.2 Boundary conditions	5
2.3 The wave equation of free vibration	5
2.4 Displacement of the surface	6
2.5 Fourier series	7
2.6 Dynamic simulation	9
2.7 Transmission line matrix	11
2.8 Spring and mass approximation	12
3. Discussion	14
4. Conclusion.....	14

1. Introduction

This report is a result of a project in the course Acoustics, TNM027, at Linköpings Technical University.

The goal of the project is to find a way to simulate the free vibrations of a membrane. In reality the membrane could be a percussion that is made of a thin membrane which is stretched over a resonator. When the membrane is hit by a drumstick it will start to vibrate and a sound is produced as soon as the vibrations are passed on to the air. The knowledge of how a membrane acts during stress is important if one is to either simulate or construct a drum or a loudspeaker cone.

To make these simulations the wave equation for free vibrations of a square membrane has to be solved. Here it is done using the separation of variables method. The result of that equation is used in three different approaches to simulate the membrane.

Note: The project was handed to us as four questions, here called Q1-Q4. Each question, except the first, is dependent on the question before. Question 1 concern the solution of the wave equation while Q2 concern implementation of the same equation. Q2, Q3 and Q4 involves simulation problems, the three questions focus on the same problem but reaches solutions in different ways.

2. Method

2.1 The wave equation

To be able to plot the free vibrations of a square membrane the wave equation must be solved. Here the separation of variables method is used. With the result one can determine the height of each point in a surface at any time. The wave equation below, equation 1a, is given in Q1.

$$\frac{\partial^2 z}{\partial t^2} - c^2 \nabla^2 z = \frac{\partial^2 z}{\partial t^2} - c^2 \left(\frac{\partial^2 z}{\partial x^2} + \frac{\partial^2 z}{\partial y^2} \right) = 0 \quad (1a)$$

In the end a solution to this equation gives the height of each point of the membrane at any time. Later on in this report it is shown how one can use the solution of the wave equation to simulate the vibration of a square membrane. That being one way to simulate such a system the above is also used in the second simulation to set the initial displacement. Equation 1a implies equation 1b as follows.

$$z(x, y, t) = \frac{\partial^2 z}{\partial t^2} - c^2 \left(\frac{\partial^2 z}{\partial x^2} + \frac{\partial^2 z}{\partial y^2} \right) \quad (1b)$$

Here one can assume the partial solution

$$z(x, y, t) = \psi(x, y) e^{i\omega t}$$

Combined with equation 1b this gives the following:

Vibration of Membranes 7

$$\begin{aligned}
 z(x, y, t) &= \frac{\partial^2 \psi e^{i\omega t}}{\partial t^2} - c^2 \left(\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} \right) e^{i\omega t} \Leftrightarrow \\
 z(x, y, t) &= \frac{i\omega \partial \psi e^{i\omega t}}{\partial t} - c^2 \left(\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} \right) e^{i\omega t} \Leftrightarrow \\
 z(x, y, t) &= \left(-\omega^2 \psi - c^2 \left(\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} \right) \right) e^{i\omega t} \tag{2}
 \end{aligned}$$

The constant K can be derived from starting and boundary conditions (which are explained in chapter 2.2) K can also be expressed as:

$$K = \frac{\omega}{c} \tag{3}$$

Equation 2 divided by $-c^2$ and combined with equation 3 gives:

$$z(x, y, t) = \left(K^2 \psi + \frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} \right) e^{i\omega t} \tag{4}$$

Here, a separation of variables comes handy.

$$K^2 \psi + \frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} \tag{4a}$$

Equation 4a implies the partial solution $\psi(x, y) = X(x)Y(y)$.
Combined with equation 4a, this gives

$$\begin{aligned}
 &K^2 X(x)Y(y) + \frac{d^2 X(x)Y(y)}{dx^2} + \frac{d^2 X(x)Y(y)}{dy^2} \\
 &\frac{K^2 X(x)Y(y) + \frac{d^2 X(x)Y(y)}{dx^2} + \frac{d^2 X(x)Y(y)}{dy^2}}{X(x)Y(y)} \Rightarrow \\
 &K^2 + \frac{1}{X(x)} \frac{d^2 X(x)}{dx^2} + \frac{1}{Y(y)} \frac{d^2 Y(y)}{dy^2} \tag{4b}
 \end{aligned}$$

Inserted into to the wave equation, equation 4, gives

$$z(x, y, t) = \left(K^2 + \frac{1}{X(x)} \frac{d^2 X(x)}{dx^2} + \frac{1}{Y(y)} \frac{d^2 Y(y)}{dy^2} \right) e^{i\omega t} \tag{5}$$

2.2 Boundary conditions

While the membrane is clamped at its rim the equation must be zero at its edges.

Mathematically this could be described as:

$$z(0, y, t) = z(a, y, t) = z(x, 0, t) = z(x, a, t) = 0$$

$e^{i\omega t}$ in equation 5 will not be 0 for all t , therefore $K^2 + \frac{1}{X(x)} \frac{d^2 X(x)}{dx^2} + \frac{1}{Y(y)} \frac{d^2 Y(y)}{dy^2}$ has to match the conditions above. This results in the following equations:

$$\frac{d^2 X(x)}{dx^2} + k_x^2 X(x) = 0$$

$$\frac{d^2 Y(y)}{dy^2} + k_y^2 Y(y) = 0$$

$$k_x^2 + k_y^2 = K^2$$

The solution to these differential equations is

$$z(x, y, t) = (\sin(k_x x + \varphi_x) \sin(k_y y + \varphi_y)) e^{i\omega t}$$

φ_x and φ_y are due to the boundary conditions 0.

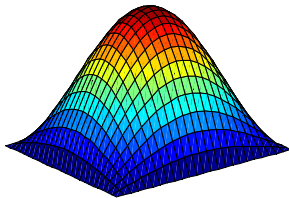
k_x and k_y are also determined by the boundaries and given by $k_x = \frac{n\pi}{a}$ and $k_y = \frac{m\pi}{a}$.

2.3 The wave equation of free vibration

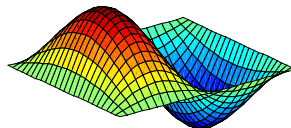
The result of these calculations is the following expression, equation 6. This equation is the answer to Q1. In Q2 this equation will be used to displace the points in a surface along its Z-axis.

$$z(x, y, t) = \sum_{n,m=0}^{\infty} (A_{nm} \cos(\omega_{nm} t) + B_{nm} \sin(\omega_{nm} t)) \sin\left(\frac{n\pi x}{a}\right) \sin\left(\frac{m\pi y}{a}\right) \quad (6)$$

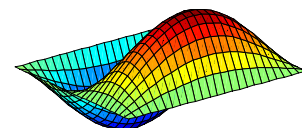
The wave equation is used to plot the first 6 free modes of vibration. n and m below represent the mode number.



$n=1, m=1$

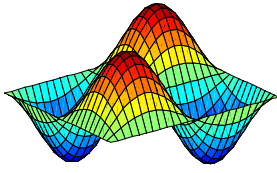


$n=1, m=2$
nodal line: a/m

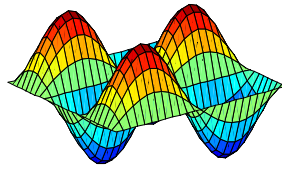


$n=2, m=1$
nodal line: a/n

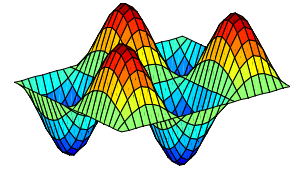
Vibration of Membranes 7



$n=2, m=2$
nodal lines: $a/m, a/n$



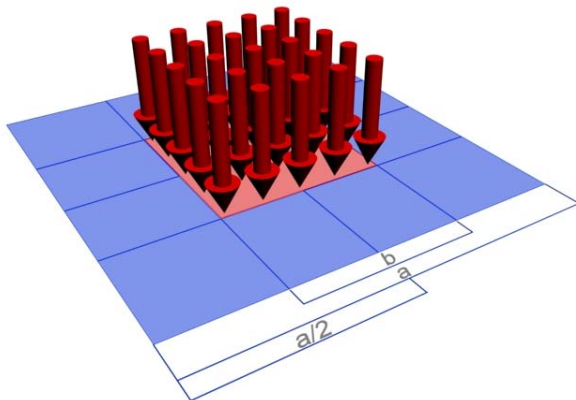
$n=3, m=2$
nodal lines: $a/m, a/n, 2*a/n$



$n=2, m=3$
nodal lines: $a/m, 2*a/m, a/m$

2.4 Displacement of the surface

In Q2 a surface, which boundaries and conditions are equal to surface discussed above, is displaced from equilibrium by a force. Here is the wave equation derived in the last chapter used to calculate the displacement along the Z-axis for each point of the surface.



The force that acts on the surface is described as:

$$f_0(x, y) = \begin{cases} \frac{F}{b^2}, & (a-b/2 \leq x, y \leq a-b/2) \\ 0 & \text{for all other } x, y \end{cases}$$

Depending on the constants (a, n, m) the force will shape the surface in a certain way. The membrane is released from rest at $t = 0$. The tension in the membrane will force the membrane to equilibrium as the time goes to infinity.

The force acting on the surface.

As the membrane is at rest before time 0, $\frac{\partial^2 z}{\partial t^2} \rightarrow 0$

$$z(x, y, 0) = \frac{\partial^2 z}{\partial t^2} - c\nabla^2 z \Rightarrow \rho \frac{\partial^2 z}{\partial t^2} = T\nabla^2 z + f(x, y) = 0$$

At time 0 the wave equation can be simplified to

$$z(x, y, 0) = \sum_{n,m=0}^{\infty} A_{nm} \sin\left(\frac{n\pi x}{a}\right) \sin\left(\frac{m\pi y}{a}\right) \text{ because } (A_{nm} \cos(\omega_{nm} t) + B_{nm} \sin(\omega_{nm} t)) = A_{nm}$$

$$T\nabla^2 z = -f(x, y) \Rightarrow$$

Vibration of Membranes 7

$$-T\nabla^2 z(x, y, 0) = -T\nabla^2 \sum_{n,m=0}^{\infty} A_{nm} \sin\left(\frac{n\pi x}{a}\right) \sin\left(\frac{m\pi y}{a}\right) = f_0(x, y)$$

The second derivative of this equation is described in equation 7.

$$-Tz(x, y, 0) = -T\left(\left(\frac{m\pi}{a}\right)^2 + \left(\frac{n\pi}{a}\right)^2\right) \sum_{n,m=0}^{\infty} A_{nm} \sin\left(\frac{n\pi x}{a}\right) \sin\left(\frac{m\pi y}{a}\right) \quad (7)$$

All constants in equation 7 are substituted into D_{nm} to ease up the equation.

$$-Tz(x, y, 0) = \sum_{n,m=0}^{\infty} D_{nm} \sin\left(\frac{n\pi x}{a}\right) \sin\left(\frac{m\pi y}{a}\right) \quad (8)$$

2.5 Fourier series

The goal is to find A_{nm} , the amplitude for each mode of the free vibration of the surface. Since D_{nm} was derived from A_{nm} (equation 7 and 8) one must be able to find a solution for A_{nm} if the solution for D_{nm} is found. To find our vibration modes we want to describe D_{nm} as a Fourier series of sinus curves. We start of with the part depending on n. Below it is proven that n must equal p , otherwise the sum is 0.

$$\sum_{p,q=0}^{\infty} \int D_{pq} \sin\left(\frac{n\pi x}{a}\right) \sin\left(\frac{p\pi x}{a}\right)$$

$$n = p \Rightarrow \sum_{p,q=0}^{\infty} \int D_{pq} \sin^2\left(\frac{p\pi x}{a}\right) = \left[\frac{x}{2} - \frac{1}{4} \sin\left(\frac{2\pi xp}{a}\right)\right]_0^a = \frac{a}{2} - 0 - (0 - 0) = \frac{a}{2}$$

$$n \neq p \Rightarrow \sum_{p,q=0}^{\infty} \int D_{pq} \sin\left(\frac{n\pi x}{a}\right) \sin\left(\frac{p\pi x}{a}\right) \Rightarrow \dots \Rightarrow \left[\frac{\sin\left(\frac{\pi x}{a}(p \pm q)\right)}{2\left(\frac{n\pi}{a} \pm \frac{p\pi}{a}\right)}\right]_0^a = 0$$

$$\sum_{m=0}^{\infty} D_{pm} \frac{a}{2} \sin\left(\frac{m\pi y}{a}\right)$$

The same procedure is used for m and thereafter we follow up with the rest of our original equation.

$$D_{pq} \frac{a^2}{4} = \int_0^a \int_0^a f(x, y) \sin\left(\frac{p\pi x}{a}\right) \sin\left(\frac{q\pi y}{a}\right) dx dy \Rightarrow$$

$$D_{pq} = \frac{4F}{a^2 b^2} \left[-\left(\frac{a}{p\pi}\right) \cos\left(\frac{p\pi x}{a}\right) \right]_{\frac{a-b}{2}}^{\frac{a+b}{2}} \left[-\left(\frac{a}{q\pi}\right) \cos\left(\frac{q\pi y}{a}\right) \right]_{\frac{a-b}{2}}^{\frac{a+b}{2}}$$

Vibration of Membranes 7

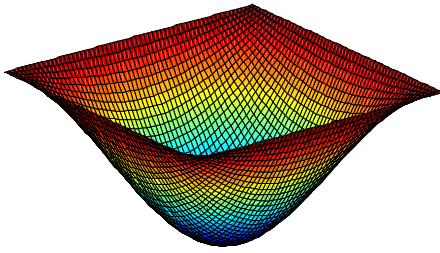
$$D_{pq} = \frac{4F}{b^2 pq \pi^2} \left[\cos\left(\frac{p\pi x}{a}\right) \right]_{\frac{a-b}{2}}^{\frac{a+b}{2}} \left[\cos\left(\frac{q\pi y}{a}\right) \right]_{\frac{a-b}{2}}^{\frac{a+b}{2}}$$

$$= \frac{4F}{b^2 pq \pi^2} \left(-\cos\left(\frac{p\pi(a+b)}{2a}\right) + \cos\left(\frac{p\pi(a-b)}{2a}\right) \right) \left(-\cos\left(\frac{q\pi(a+b)}{2a}\right) + \cos\left(\frac{q\pi(a-b)}{2a}\right) \right)$$

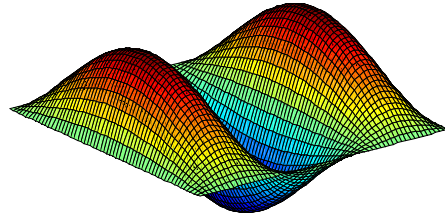
As D_{nm} was derived from A_{nm} in equation 7 and 8 A_{nm} is, in reversed order, derived from D_{nm} .

$$A_{nm} = \frac{4Fa^2 \left(-\cos\left(\frac{n\pi(a+b)}{2a}\right) + \cos\left(\frac{n\pi(a-b)}{2a}\right) \right) \left(-\cos\left(\frac{m\pi(a+b)}{2a}\right) + \cos\left(\frac{m\pi(a-b)}{2a}\right) \right)}{(-T)b^2 nm \pi^4 (n^2 + m^2)} \quad (9)$$

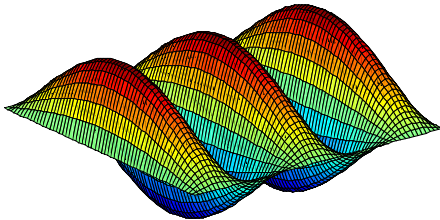
Below are the first modes of vibrations followed by linear combinations of 3 and 9 modes.



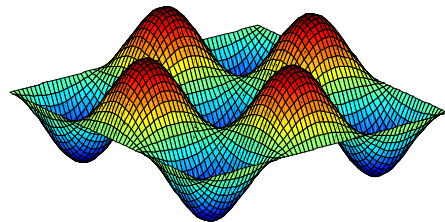
$n=1, m=1$



$n=2, n=1 / n=1, n=2$

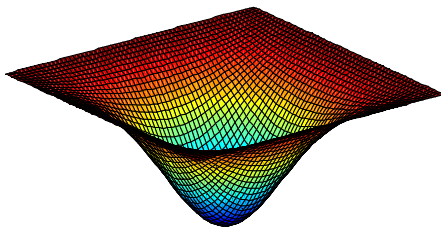


$n=3, m=1 / n=1, m=3$

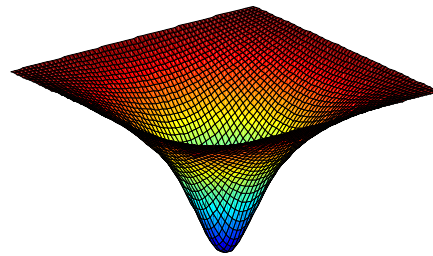


$n=3, m=3$

Vibration of Membranes 7



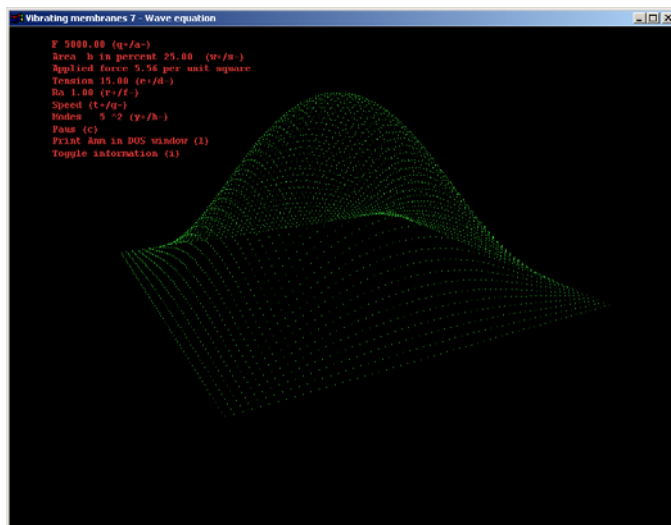
Linear combination n,m=3



Linear combination n,m=9

2.6 Dynamic simulation

Wave equation simulation



The wave equation simulator is a dynamic simulation built on the wave equation for this membrane.

The simulator allows the user to vary variables as:

Tension

ρ

Force

Affected area

The number of modes calculated.

and can also print the current A_{nm}

The simulation is written in C++ and based on three equations

$$\omega_{nm} = \pi c \sqrt{(n/a)^2 + (m/a)^2}$$

$$A_{nm} = \frac{4Fa^2 \left(-\cos\left(\frac{n\pi(a+b)}{2a}\right) + \cos\left(\frac{n\pi(a-b)}{2a}\right) \right) \left(-\cos\left(\frac{m\pi(a+b)}{2a}\right) + \cos\left(\frac{m\pi(a-b)}{2a}\right) \right)}{(-T)b^2 nm \pi^4 (n^2 + m^2)}$$

Vibration of Membranes 7

$$z(x, y, t) = \sum_{n,m=0}^{\infty} (A_{nm} \cos(\omega_{nm}t)) \sin\left(\frac{n\pi x}{a}\right) \sin\left(\frac{m\pi y}{a}\right)$$

The program first calculates A_{nm} and ω_{nm} . Every iteration the program calculates each nodes position by summing allowed modes and frequencies. The time is increased by a fixed number each frame.

Advantages

- Gives a exact mathematical simulation of the wave
- Good scalability
- Valid for all x,y,t within boundaries
- Continuous for all x,y,t within boundaries

Limitations

- Very computational heavy for large number of modes
- Small number of modes result in stiff simulation

Summary of functions

int buildANM(int nr) line 157

Is called every time A_{nm} needs to be modified for example when the program is started or some variable is changed. Calculation is done by the formula described above. Returns the current number of modes.

void buildWNM() line 180

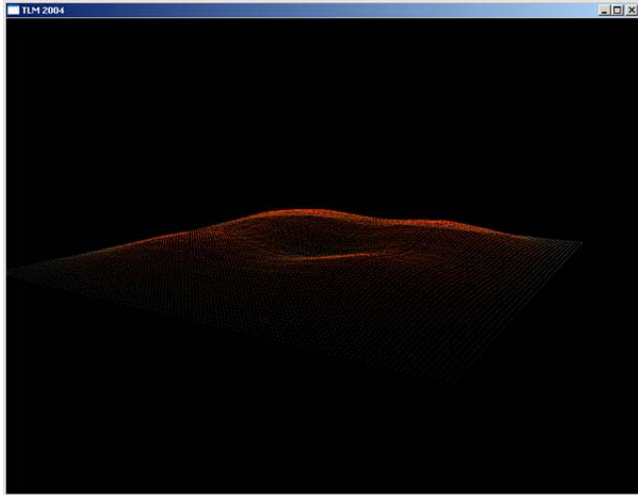
Calculates the current ω_{nm} .

void calculate() line 189

Is called every iteration. It calculates every nodes position by summing the contribution of every mode at the current time.

The OpenGL framework is built on a version written by Jeff Molofee 2000 (nehe.gamedev.net)

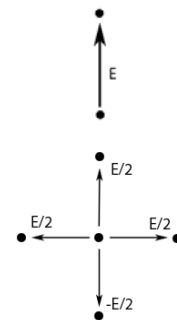
2.7 Transmission line matrix



The strategy of transmission line matrix, TLM, is that every point (here called node) in a mesh is dependent on the energy of its four neighbors. The TLM method works for all kinds of energies; here it is potential energy that is spread between the nodes. This energy is created with the wave equation that displaces the mesh from equilibrium. The tension of the membrane will try to drag each point back into equilibrium.

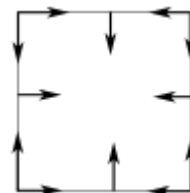
The transmission of energy is divided into two steps.
In the first step energy from one node affects another node.

In the second step the energy is distributed to the neighboring nodes. To keep the energy level constant before and after step 2 it is necessary to distribute the energy according to the matrix below.



$$E_{ij} = \frac{1}{2} \begin{bmatrix} -1 & 1 & 1 & 1 \\ 1 & -1 & 1 & 1 \\ 1 & 1 & -1 & 1 \\ 1 & 1 & 1 & -1 \end{bmatrix}$$

Since the membrane is clamped at its rim the energy has to go back into the membrane when it reaches the rim. In this simulation the system is completely free from resistance and therefore all energy that reach the rim goes back into the membrane. This special case is not taken care of in the matrix above and therefore one must handle this situation separately.



Advantages

- Possibility to displace the membrane as one would like. Here the wave equation is used to set the initial energy for each node but one could easily set the initial energy to something else.
- Fast calculations which allow high resolution.

Disadvantages

- Difficulties to change the properties of the membrane while running the simulation. This is due to that the tension, for example, is only used when the initial energy is set, during the simulation a change of the tension would not have any effect at all.

Summary of functions in tlm.cpp

void initHeight(void), line 92

This function calculates An_m according to the wave equation, equation 9. This function is called when the membrane is released from rest, line 677. At line 690 the initial energy for each node is calculated using An_m .

void Vr(int i, int j), line 111

This function spreads the energy from *last iteration* between the nodes according to the matrix above. This function is called one time for each node in the mesh each iteration.

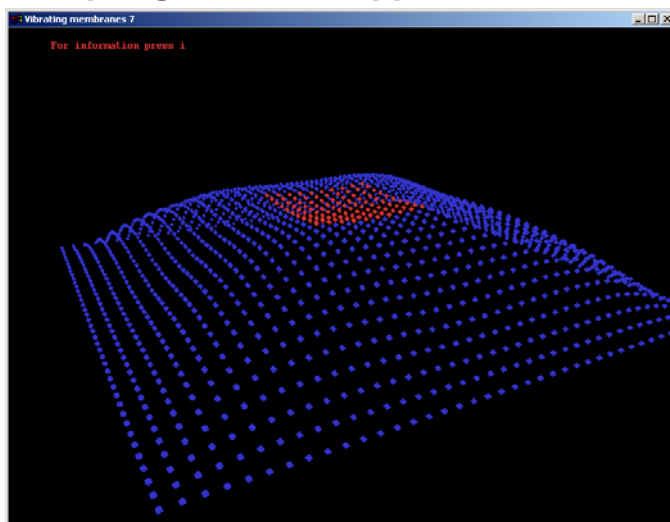
void Vi(int i, int j), line 155

This function sums up the energy each node has at the *current iteration*. This function is called one time for each node in the mesh each iteration.

int DrawGLScene(GLvoid), line 203

This function takes care of all the drawing and makes calls to the other functions that need to be called each iteration. Lines 225-238 make calls to V_r and V_i and finally assign the y-value (height) of each node to the sum of the current energy of the node times c , the speed of the wave, and t , the time since last iteration.

2.8 Spring and mass approximation



The wave equation simulator is a dynamic simulation built of a grid of masses attached to each other by springs.

The simulator allows the user to vary variables as:

- Mass
- Spring constant
- Pre tension of the springs
- Affected area
- Apply a force to selected masses

Vibration of Membranes 7

The simulation is written in C++. Each iteration every nodes height is compared to its four neighbors except the ones in the outer lines that are fixed, and will only affect the others. The calculations are made as follows:

d distance between nodes in xy plane

h current nodes height

h_x neighbor

$$\text{Stretch} = \sqrt{(d)^2 + (h - h_x)^2} - d \quad \text{Tension} = \text{stretch} + \text{pretension}$$

$$\text{Force} = \sin\left(\arctan\left(\frac{(h - h_x)}{d}\right)\right) \cdot \text{stretch}$$

The forces are summed and the new position is calculated in an Euler function with fixed time. As we have the Euler function we implemented the ability to add a force over a time to the masses instead of the mathematical displacement in the other two simulations. To make the masses stop at a displaced position while affected by a force we also added a friction that only is active when the force is.

Advantages

- The ability to easily add other forces
- Small number of computations
- Can take the shape of all modes

Limitations

- Non continuous equations result in some amount of noise
- Small number of nodes result in low resolution

Summary of functions

float spring(float node, float neighbor) line 176

Returns the tension between two nodes using the formula described above.

void calcForce(int node) line 182

Sums up the forces acting on “node” by calling *spring* for each neighbor.

void calculate() line 190

This function is called every iteration. It will call *calcForce* for each node and then add the forces applied by the user over a fixed time.

The OpenGL framework is built on a version written by Jeff Molofee 2000
(nehe.gamedev.net)

3. Discussion

Its time to core the best method for simulation; dynamic simulation, TLM or spring and mass approximation. Since the result of all three methods was satisfying we must make the following judgments based on our thoughts of what else these methods can be used to. The first method that continuously calculated the linear combination for all times was easy to write and in this case the result was good. However it is very hard to implement another force in this method. This is much easier to do with the spring and mass approximation-method. The TLM method was maybe a little trickier to program, it was hard to keep all the energy in the membrane, and even though this program is not prepared for external forces it would be fairly easy to implement. Since we ran into problems with increased stiffness of the membrane when the number of nodes increased our recommendation about simulation methods for thin membranes is transmission line matrix.

4. Conclusion

The computers today are powerfull enough to calculate large amount of data in real time. As the speed of computers increase its possible to increase the accuracy of the simulations as well. Combined with advanced math one can easily generate visual simulations to gain grater understanding of how various systems behave.

The methods discussed in this report are valid for not only square membranes but also for circular membranes. The wave equation for circular membranes is not exactly the same as for square membranes. However the way of solving that equation should be similar to what is described above.