Synthetic thunder

(/user/mnemonic)
mnemonic (/user/mnemonic)

Hey all,

I've succeeded in generating thunder using a quasilinear model and MATLAB, and hopefully will soon be able to find an algorithm that will allow me to create entire storms in realtime in Pd (right now it takes a few hours to calculate the thunder signature of a 6-km lightning stroke).

In the meantime, I thought you guys might enjoy the product of today's computations, the thunder signature of a 3km lightning stroke 100m away from the observer. The dry output is very synthetic sounding, but with the simple addition of some reverb it really comes to life!

(I apologize for the quick & dirty cooledit 'verb on the example)

http://www.pdpatchrepo.info/hurleur/ThunderGenerator.zip (http://www.pdpatchrepo.info/hurleur/ThunderGenerator.zip)

0

mnemonic (/user/mnemonic) posted 10 years ago

Posts **7** | Views **4.1k**

Log in to reply (/login)

(/user/obiwannabe)
obiwannabe (/user/obiwannab...

Wonderful stuff mnemonic. When you say a quasi-linear model do you mean a finite element model? If so, what are you modelling at each point, pressure due to thermal expansion? How many elements
are in it?

Would love to see the MATLAB code mate 😀

Try convolving it with a very sparse spiky noise to mimic trees and buildings instead of a normal reverb and you will hear HUGE difference in how it comes to life!

nice one!

a.

Use the Source.

· 0

● **obiwannabe (/user/obiwannabe)** posted 10 years ago

(/user /mnemonic)
mnemonic (/user /mnemonic)

Hey obi,

thanks for the verb suggestion, I'll have to look into that soon... but first I have to figure out an efficient way of using pd to make the generator real-time(ish).

I was thinking that (due to the way the N-waves are formed mathematically) I could populate two wavetables based on the given parameters -- one for the positive pulse and one for the negative pulse (see attached paper). Then I could have each lightning element just bang out the pulses at the appropriate time. It's still thousands of such elements doing it though :\

As for the quasilinearity, the lightning strike is approximated by many linear segments, each of which emits one pressure wave when the strike occurs.

I've attached the matlab code and the theory paper for your enjoyment (it could probably use some further optimization but it's come a long way).

http://www.pdpatchrepo.info/hurleur/ThunderGen_Theory.zip (http://www.pdpatchrepo.info/hurleur/ThunderGen_Theory.zip)

· 0

● **mnemonic (/user/mnemonic)** posted 10 years ago

Thanks again. I PM'd you with a an email. This afternnon I had a play with a recursive way of adding N-wave segments. It's very crude and

(/user/obiwannabe) obiwannabe (/user/obiwannab...

unfortunately the random process still adds too much colour to the result, but it might give you ideas on a possible real-time solution.

http://www.pdpatchrepo.info/hurleur/thunder-nwave-experiment1.pd (http://www.pdpatchrepo.info/hurleur/thunder-nwave-experiment1.pd)

Use the Source.

⚙ · ▲ 0 ▼

● **obiwannabe (/user/obiwannabe)** posted 10 years ago

(/user/mnemonic) mnemonic (/user/mnemonic)

Hey for a quick job that sounds pretty good obi 😀

A bit short on the documentation side of things though... I'm having trouble figuring out the process you used to shape the waves. Are all the N-waves that you generate assumed to emit perpendicularly from the lightning segment (ie. do they all have the "N" shape)?

E: i sent a pm email to you as well since yours hasn't seemed to come through yet

⚙ · ▲ 0 ▼

● **mnemonic (/user/mnemonic)** posted 10 years ago

(/user/obiwannabe) obiwannabe (/user/obiwannab...

I don't understand enough about the geometry to visualise the distortions and phase changes that occur off the perpendicular axis. I can see you deal with this in the matlab code. Is that a re-implementation of Ribner and Roys equations? It's hard enough to imagine in a 2D plane, nevermind 3D 😀

All I did with Pd is make a recirculating buffer that has a filter to approximate propagation losses and add N-waves created by the vline segment randomly to it. Sometimes the superposition does create thunder like effects, but that's more luck than judgement it seems.

I reckon to get the right effect you either need several n-wave generators in parallel, or several parallel delays operating on one wave.

As the text says, it's equivalent to a convolution of the n-wave with a set of points that are distances to corners in the tortuous line, it's the hypotenuse of a right angle triange if we assume the bolt comes straight down that's the square root of the horizontal distance squared plus the height squared, and with c as a constant then time correlates directly to distance.

In a way it's granular synthesis, so the density must be calculated. I think you could reduce the whole caper to two variables, density and off-axis phase shift. It's the propagation time divided by the period of one n-wave I think. At ground level the observer is perpendicular so it's very dense, but as you move up the lightning bolt the subtended angle increases and so does the propagation time, so density tails off.

I'm really looking forward to hearing where this takes you next...

a.

Use the Source.

0

● **obiwannabe (/user/obiwannabe)** posted 10 years ago

(/user/mnemonic)

mnemonic (/user/mnemonic)

The matlab code is more or less a direct implementation of the equations contained in the paper (I'm also only doing this in 2D for the moment). This is, as you point out, the flaw in your implementation since the shape of each pressure wave is dependent upon the relative positions of two parabolic pulses, p+ and p-. When these pulses are close together their addition forms the characteristic "N" shape, however the further off-axis it is observed the further apart these pulses become, until they are two distinct bulges separated by some empty space. This is why I think it might work to have two wavetable lookups for each N-wave, where the off-axis shaping can be controlled by indexing p- behind p+ by a number of samples determined by the off-axis angle (phi).

I agree that this is very similar to granular synthesis, with a decaying density as you describe. It's quite an interesting puzzle for real-time implementation and as soon as I've got some new progress this forum will be the first to know.

And after that, I'm really excited about the musical prospects of creating "stylized" lightning for recreatable timbres, or using different-shaped pulses for the N-wave building blocks.... the storm's the limit 😉

0

● **mnemonic (/user/mnemonic)** posted 10 years ago

Posts **7** | Views **4.1k**

Log in to reply (/login)