Research into Real-time Audio Processing

Michael Xing, under Prof. Walker White

Fall 2020

1 Introduction

The simulation of audio reverberations in real-time has traditionally been limited to simple processing effects. These effects, however, must be manually tuned for the location being simulated, and are limited in their ability to simulate complex spaces, especially when combined with moving sources / listeners.

The goal of this research was to explore more accurate ways of simulating sound propagation in an environment while still maintaining the ability to run in real-time applications such as game engines.

2 Literature

The use of a digital waveguide to simulate the propagation of sound waves through a medium is well documented in the literature. [5] By linking together scattering junctions with delay lines, a discrete approximation of the actual medium a wave would travel through can be made. In the limit where the number of scattering junctions in the waveguide becomes large, the waveguide serves as an accurate reproduction of the true propagation of the wave. [7] The literature suggests that, especially for simple models, a waveguide mesh should be capable of being run in real-time. [5]

Alternative algorithms have also been proposed. One approach has been to use ray-tracing to draw rays from the listener outwards, tracking distance and also reflections, to calculate how the sound should be perceived. [3] This approach runs close to existing patents, [4] though the patent notably traces rays from the source to the listener.

A hybrid approach dubbed "mesh tracing" has also been proposed, where the space is first filled with randomly generated nodes, delaunay triangulated into a graph, which is then traversed with a BFS to build a lookup table of impulse response functions. [2] This approach has been cited by others in the literature, [1] and seemed the most promising.

3 Approach

An attempt was made to implement the mesh-tracing algorithm as described in a 2-dimensional square room. The four vertices representing the square were filled with 100 randomly distributed points, and the entire set was delaunay triangulated before the algorithm was run. When running the BFS, the total distance travelled from the source as well as the power distribution was tracked in order to adjust the volume and delay of the sound during playback from the listener's perspective.

The literature on mesh tracing unfortunately does not go into specifics about how to compute the impulse response function based on the result of the graph traversal. The original paper states simply to "calculate the incoming power by applying a propagation function F to the vertices" [2] without further explanation. For the demo, the amount of power at the source was set to 100%, and then equally divided based on the percentage of the arc-length spanned by the bisectors of the angles formed by each outgoing edge versus its adjacent planar neighbors.

When conducting mesh tracing, each node on average had around three neighbors that the power should propagate to at each step. Since the vertices were randomly distributed, the lengths of the paths are irrational and thus every unique path has a unique length. As more nodes are propagated, the number of unique paths exploded exponentially and became intractable to compute and store. To alleviate this problem, the length of each path was bucketed, and the total number of buckets was limited.

A second simulation was created for comparison using a standard waveguide mesh. The algorithm used was as described in standard textbooks, [6] written in C++ and run on audio samples at 48kHz.

4 Results

4.1 Mesh Tracing

Using mesh tracing, even with bucketing, the frequency of dropped power due to the number of buckets at a node being full became significant. Widening the buckets could alleviate this, but at the cost of greatly impacting the accuracy of the final computation. In addition, with power exponentially dropping off after each step, floating point accuracy became nontrivial. All of these factors combined resulted in sound that was not only abnormally quiet, but sporadic and dependent on the exact placement of the randomly distributed nodes within the the space.

Consequently, the demo, which allowed the user to walk around the space and listen to the changes in audio, was not particularly realistic, though it did run at framerate. Moving the listener through the space would result in large changes in how the sound was perceived, both in volume and in reverberation, as the viewer switched from node to node. Rerunning the initial computation would result in drastic changes even at the same position.

Further, while the Voronoi diagram could be used to allow the listener to move around, there was no easy way to move the source without rerunning the entire graph traversal from the new source location. This limitation makes this method rather limiting, as with fixed source locations, the impulse responses at various locations throughout a space can be precomputed before runtime, rendering real-time rendering less necessary.

4.2 Square Waveguide Mesh

The waveguide mesh demonstrated a reasonably good ability to simulate propagation of sound. Unfortunately, the results showed that the high computational cost to step the mesh made it impractical for simulations of any realistic size that would be useful in a game engine.

At 48kHz, or $48,000$ samples per second, because the mesh must be stepped after every sample, the amount of time spent stepping the mesh is strictly upper bounded at $20,833.\bar{3}$ nanoseconds.

Unfortunately, while the step time for the mesh is expected to scale linearly with the number of nodes, filling a 2-dimensional space necessitates quadratic scaling in the number of nodes with respect to the length of the edge in a square room.

Running some timing experiments, the theoretical scaling is observed. The data was collected by stepping a mesh of size $n \times n$ a total of 20 times and averaging to find the nanoseconds per step. Meshes from $n = 1$ to $n = 100$ were tested. The resulting times were plotted in figure 1. Figure 2 shows the time plotted against the total number of nodes within a network, n^2. The graphs show quadratic and linear scaling, as expected.

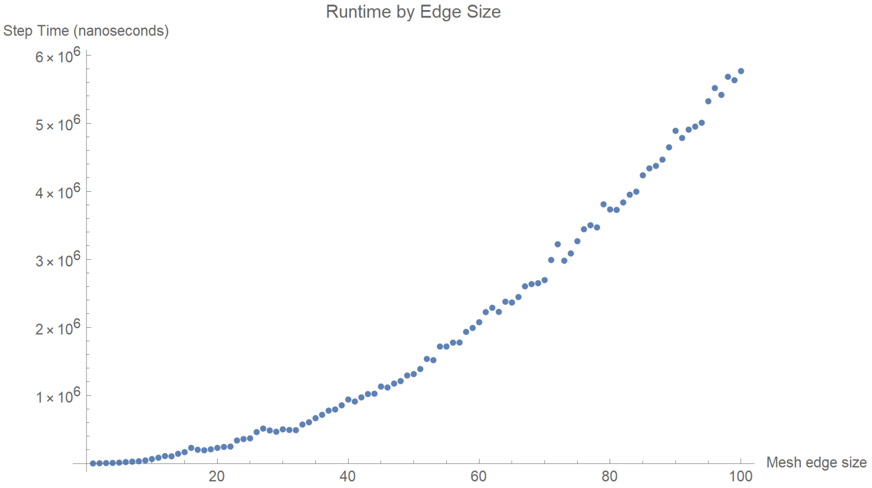

Figure 1: Time per step plotted against n

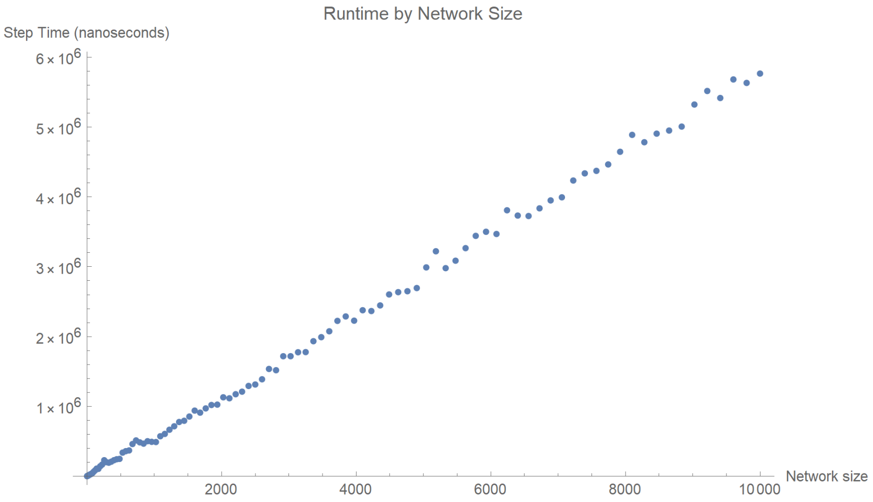

Figure 2: Time per step plotted against n^2

To confirm this, the same data is shown on a log-log plot in figure 3 and figure 4. Running a linear regression on figure 3 gives the equation $6.57692 + 1.9441x$, whose slope of ≈ 2 confirms that the scaling is quadratic. Likewise, figure 4 had a linear regression that came out to $6.57692 + 0.972049x$, again with an expected slope of ≈ 1.

Looking at the timings shown in figure 1, a 5×5 mesh with 25 nodes is around the largest size one can reasonably expect to run without noticeable clicking in the audio. Unfortunately, as a mesh waveguide requires a large number of nodes to accurately approximate the medium it aims to simulate, this unfortunately also appears to preclude using a mesh waveguide in real-time contexts.

5 Next Steps

Based on the results shown, it appears that the mesh-tracing algorithm as described in the literature remains relatively underdeveloped for real-time use. At the same time, simulating an entire waveguide mesh is computationally intractable for all but the most uninteresting scenarios. As such, ray-tracing remains a potentially viable avenue to proceed for computing accurate real-time reflections and reverberations.

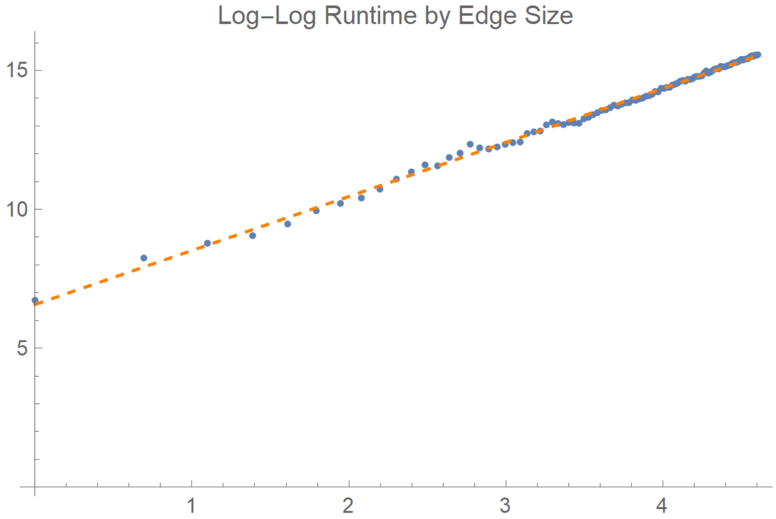

Figure 3: The data from figure 1 on a log-log plot.

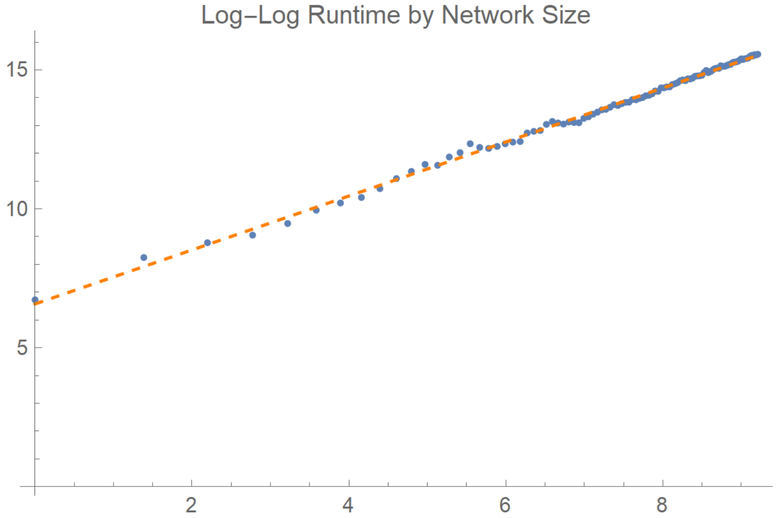

Figure 4: The data from figure 2 on a log-log plot.

References

[1] Patty Huang, Matti Karjalainen, and Julius O Smith. Digital waveguide networks for room response modeling and synthesis. In *Audio Engineering Society Convention 118*. Audio Engineering Society, 2005.

[2] Bert Schiettecatte, Axel Nackaerts, Bart De Moor, et al. Real-time acoustics simulation using mesh-tracing. In *ICMC*. Citeseer, 2003.

[3] Carl Schissler and Dinesh Manocha. Gsound: Interactive sound propagation for games. In *Audio Engineering Society Conference: 41st International Conference: Audio for Games*. Audio Engineering Society, 2011.

[4] Robert Allen Shearer. Using ray tracing for real time audio synthesis, March 20 2012. US Patent 8,139,780.

[5] Julius O Smith. Physical modeling using digital waveguides. *Computer music journal*, 16(4):74–91, 1992.

[6] Julius O. Smith. *The Rectilinear 2D Mesh.* http://ccrma.stanford.edu/~jos/pasp/Rectilinear_2D_Mesh.html, accessed December 16, 2020. online book.

[7] Scott A Van Duyne and Julius O Smith. Physical modeling with the 2-d digital waveguide mesh. In *Proceedings of the International Computer Music Conference*, pages 40–40. INTERNATIONAL COMPUTER MUSIC ACCOCIATION, 1993.