ARTIFACT-FREE ASYNCHRONOUS GEOMETRY-BASED AUDIO RENDERING

Nicolas Tsingos

Bell Laboratories - Lucent Technologies*

ABSTRACT

Most audio rendering systems include a geometry-based simulation engine which computes and updates sound propagation paths and an auralization engine which renders audible the resulting sound field. In the case of dynamic environments the attributes of the sound paths vary over time, which can cause severe artifacts in the auralized sound signal. To avoid this problem, geometrical calculations have to be conducted synchronously with the signal processing tasks and the necessary high update rates can be difficult to achieve in complex environments. In this paper, we describe a technique which allows for conducting geometrical calculations completely asynchronously from the signal processing. In particular, we introduce a prediction mechanism of path attributes which ensures that artifact free signal processing can be achieved even when geometrical information is updated at low rates. This technique can be applied to any geometry-based simulations, regardless of the technique used for finding the sound propagation paths.

1. INTRODUCTION

Virtual acoustics aims at simulating the sound field created by virtual sound sources in a virtual environment and rendering it audible to a user/listener (see [1, 2, 3, 4, 5] for extensive introduction, system descriptions and bibliography). Most virtual acoustics systems use geometrical calculations to build early reverberation paths from a 3D model of the environment. This can be achieved using a variety of techniques such as ray, cone or beam tracing. Each path corresponds to a sequence of propagation events (reflection, transmission, diffraction,...) and its contribution to the impulse response of the environment can be modeled by a secondary *image* source from where the sound emanates.

In the case of interactive dynamic environments, the impulse response is continuously changing and it should be updated at the audio sampling rate to avoid any artifacts in the generated sound. One possible solution could be to morph between impulse responses at successive time-steps. However, correct filter interpolation would be difficult to achieve. Thus, sound paths are usually individually rendered, their attributes being updated over time based on geometrical calculations [6, 7, 3]. Such attributes typically include: 1) the length of the path which relates to the propagation delay and attenuation, 2) the visibility/validity of the path relative to the listener (since surfaces are of finite extend, contributions of many image sources get discarded since the corresponding paths run outside the reflecting polygons), 3) the incident direction on the listener which is used for 3D localization cues and 4) filtering parameters which are used to model frequency dependent effects.

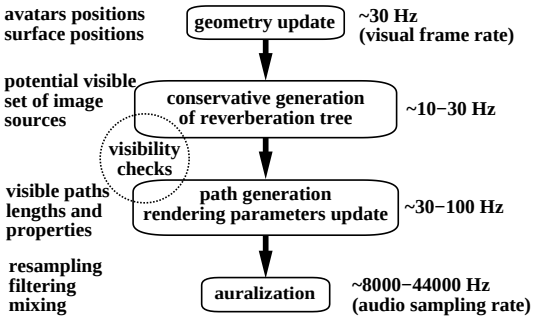


Fig. 1. Flowchart of a typical asynchronous virtual acoustics application: part of the geometrical simulation is running at low rates while valid path construction and auralization run at high rates.

For efficiency reasons, a common approach is to process the signal by blocks of several samples and interpolate propagation delay and other parameters across this time-step (the validity/visibility attribute might require special diffraction treatment to avoid discontinuities in the signal [7, 8]). Hence, most applications [7, 3] perform synchronous geometrical calculations and signal processing at typical rates of 20/30Hz (which also corresponds to typical video frame-rates). Recent developments using beam-tracing [5] have shown that it is possible to update high order reflection paths in complex architectural environments at rates up to 100Hz while performing most of the geometrical calculations at 10Hz. In that case, an asynchronous system is used as shown in Figure 1. However, the system as proposed in [5] does not allow for consistent update of dynamic paths' attributes, since

*Visual Communication Research Dpt.,
Room 2D-509, 600-700 Mountain Avenue, Murray Hill, NJ 07974, USA.
tsingos@research.bell-labs.com
http://www.bell-labs.com/org/1133/Research/VirtualAcoustics

the reverberation tree and path data are discarded and recreated from scratch after each environment modification.

Our work is motivated by two aspects. First, although most of the computing resources are usually used by the auralization stage, the cost of performing geometrical calculations can quickly become prohibitive, especially for interactive applications in large dynamic environments. More complex phenomena, such as diffraction, also make path generation more costly since diffracted propagation paths can be much more difficult to construct than simple specularly reflected paths [8]. In such cases, path generation rate can easily drop below 10 frames per second which is not compatible with the high rates needed for properly auralizing the sound. In the case of a networked application, transmission latency can also cause the parameters to be updated at low rates. More generally, being able to perform auralization and parameter update at two different and uncorrelated rates would allow for designing more flexible processing pipelines.

Second, it can be interesting to make the update rate of the geometrical simulation as low as possible to leave more resources to the signal processing tasks or to better control load balancing in a multi-processor system. In this paper, we present a technique which allows to totally uncouple geometrical calculations from auralization processing while maintaining artifact-free sound. In particular, we introduce an extrapolation mechanism which allows for refreshing at high rates the attributes necessary for proper auralization.

2. UPDATE AND QUERY OF PATH ATTRIBUTES

We assume an architecture similar to the one in Figure 1, where a reverberation tree, encoding all possible propagation sequences in the environment, is continuously updated. For example, this can be achieved through ray or beam tracing. However, we would like to allow the path generation process to run at low rates (below 20Hz) and maintain high refresh rates only in the auralization process. We now describe how this can be achieved.

2.1. Consistent update of the paths

Contrary to the solution proposed in [5], we do not discard the previously computed reverberation tree to rebuild it from scratch every time the environment is modified. Similarly, we do not discard previously existing paths, since it makes it impossible to track the value of their attributes over time. We keep on expanding the same tree and path list as the environment is modified, adding new nodes and creating new paths as they appear. Each node in the reverberation tree is tagged with an *event identifier* (reflection, diffraction, transmission, etc.) and a pointer to the surface (or edge) where the event occurred. One problem with such approach can be high memory requirements. Hence, nodes in the reverberation tree are also time-stamped when the corresponding event occurs. Thus, events which did not occur for a "long-time" are likely to correspond to dead branches of the tree and can be discarded. A *garbage collector* process can be used to regularly check for unused nodes and free memory. Paths are tagged in the same way by the sequence of events and surface identifiers and time-stamped according to the last time they were valid. To update the attributes of the propagation paths, we traverse all possible sequences in the current reverberation tree. The identifiers are used to check if a path already existed for this sequence or must be created. Paths can thus be updated in a consistent way over time. The auralization stage skips all invalid paths and, as for the reverberation tree nodes, uses time-stamps to discard unused paths and free memory. We will discuss in the next section how discarding the paths can affect the auralization.

2.2. Extrapolating attributes of paths

Our goal is to update attributes of paths at low rates, while maintaining high rate queries from the auralization process. Extrapolation is one solution to this problem. In this section, we focus primarily on the extrapolation of the propagation delay and incident direction of the sound paths. The same extrapolation scheme could, of course, be applied to other attributes such as filter parameters. We chose to directly extrapolate the location of the image source corresponding to every path, which allows us to retrieve both delay and incident direction on the listener.

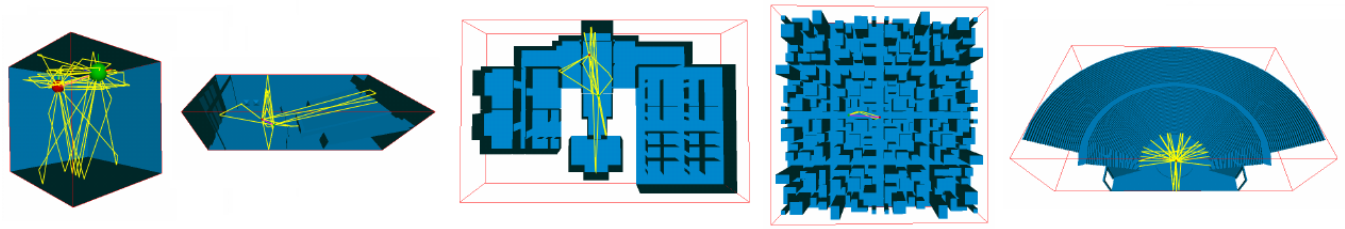
Linear extrapolation has been used for similar problems to uncouple visual and haptics simulations in the context of virtual surgery (contact forces must be updated at more than 1kHz to provide realistic interaction with rigid surfaces) [9]. However, a simple linear extrapolation might prove insufficient at low update rates. A common robust prediction method is Kalman filtering [10] and it could probably be used here, but as discussed in [9], a simpler technique can be used in non-noisy situations. Polynomial extrapolation suffers from divergence problems in case of abrupt or irregular motion (e.g. mouse-controlled) and non-uniform updates makes it even less robust. We thus experimented with a rational function:

$$\pi_{\mathbf{n}}(t) = \frac{\mathbf{A_n}(t-t_n)^3+\mathbf{B_n}(t-t_n)^2+\mathbf{C_n}(t-t_n)+\mathbf{D_n}}{1+(t-t_n)^2}, \quad t_n < t < t_{n+1}. \tag{1}$$

Every time a new position $\mathbf{P_n}$ is specified for an image-source at time t_n, the parameters $\mathbf{A_n}$, $\mathbf{B_n}$, $\mathbf{C_n}$ and $\mathbf{D_n}$ must be updated. We impose C1 continuity for the predicted position $\pi_{\mathbf{n}}(t)$ which gives

$$\mathbf{D_n} = \pi_{\mathbf{n-1}}(t_n) \text{ and } \mathbf{C_n} = \pi'_{\mathbf{n-1}}(t_n). \tag{2}$$

$\mathbf{A_n}$ and $\mathbf{B_n}$ can be determined by choosing a predicted value for $\pi_{\mathbf{n}}$ and $\pi'_{\mathbf{n}}$ at a future time $t_n + \Delta t_n$. However, to



cube O. Messiaen hall office city Epidaurus

Fig. 2. Example early reverberation paths in different environments of increasing complexity. Polygon count for the different models from left to right: 6, 92, 401, 1041, 2060.

reduce the computations, we chose to directly impose

$$\mathbf{A_n} = (\mathbf{P_n} - \mathbf{P_{n-1}})/(t_n - t_{n-1}) \quad (3)$$

and $\mathbf{B_n}$ is then computed so that

$$\pi_{\mathbf{n}}(t_n + \Delta t_n) = \mathbf{P_n} + (\mathbf{P_n} - \mathbf{P_{n-1}})\Delta t_n, \quad (4)$$

where $\Delta t_n = k(t_n - t_{n-1})$, $k > 1$.

Parameter k controls the behavior of the scheme. Low values of parameter k will cause the prediction function to oscillate around the target position (Figure 3). Increasing k smoothes the oscillations and makes the scheme more robust but relaxes the constraint for the extrapolated position to quickly match any change in the trajectory. Using $k = 4$ provides a good tradeoff between robustness and damping of the scheme even in the case of abrupt motion using the mouse. Finally, we assume a minimum delay between two consecutive updates of the parameters to further reduce the risk of numerical instability. Using a minimum delay $t_n - t_{n-1} > 1/100^{th}$ s. and double precision floating point calculations provides good results.

Concurrently to the path update process (which will use Equ.(2),(3),(4)), the auralization process can now use Equ.(1) to recompute at the necessary rate the image-source location and deduce the resulting incoming direction and delay. Note that only the extrapolated positions are then used by the auralization process.

In order for the extrapolation scheme to work well, all attributes of paths should always be updated, regardless of whether the path is valid/visible or not. One particular case where this cannot be achieved perfectly is when a path first appears or re-appears after being deleted. In that case, oscillations can occur in the predicted trajectory, which might impact the resulting signal. If sufficient memory is available, all nodes in the reverberation tree and paths can be kept once they are created. Otherwise, the use of time-stamps, as suggested in section 2.1, limits this problem.

3. RESULTS AND DISCUSSION

We implemented the described techniques in the context of a real-time auralization system based on dynamic Monte-Carlo path-tracing. The system supports multi-processing and different update rates for the three stages of the simulation: construction of the potential valid set of image sources (or sequences), update of the attributes of paths and auralization.

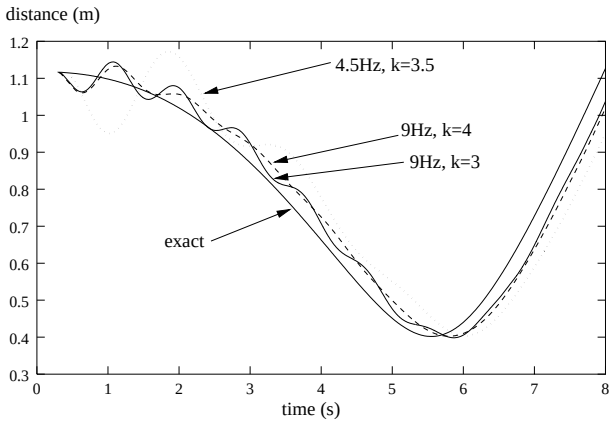


Fig. 3. Comparison of exact source to receiver distances over time with the values obtained from extrapolated positions using different parameter values and (low) path update rates. Note the oscillations as k is decreased and the improved convergence as the update rate increases.

Table 1 shows timing results in several environments of increasing complexity (see corresponding models in Figure 2). In particular, we experimented with both interior and exterior environments, with slow and fast moving sources and receivers. All values are averaged over 1s. time-steps. All data was collected on a SGI Octane workstation with one R10k processor 225MHz/256Mb of RAM and a SGI Onyx2 with 4 R10k 250MHz/1Gb of RAM (denoted by (*) in the table). Statistics have been collected by extrapolating positions of image sources at display frame rate (30-70Hz) as a visualization of time-varying echograms was generated. We used image-sources to receiver distance to evaluate the error caused by the extrapolation. Table 1 shows the number of valid paths and potential sequences we considered for all situations. Path generation considers every sequence and the path generation rate includes visibility calculation cost. We used a regular 3D grid as a spatial data structure

Table 1: Comparative execution results in several environments with variable source speed

environment	valid paths/sequences	cpu load	extrapol. rate/max.	path update rate	max. error/avg. path length	avg. speed
cube (6x6x6 m)	374/737	92 %	26/70 Hz	1 Hz	0.4/14 m	0.25 m.s $^{-1}$
cube (6x6x6 m)	129/201	50 %	68/330 Hz	2.5 Hz	0.4/14 m	0.25 m.s $^{-1}$
cube (6x6x6 m)	129/201	85 %	47/330 Hz	9.5 Hz	0.2/14 m	0.25 m.s $^{-1}$
cube (6x6x6 m)	63/87	65 %	37/820 Hz	22 Hz	0.09/8 m	0.25 m.s $^{-1}$
cube (3x3x3 m)	129/259	60 %	67/280 Hz	2.5 Hz	0.4/7 m	0.25 m.s $^{-1}$
O. Messiaen Hall	50/107	60 %	76/650 Hz	2 Hz	0.4/20 m	0.25 m.s $^{-1}$
office (*)	10/100	33.5 %	59/550 Hz	13 Hz	5 / 150 m	3 m.s $^{-1}$
city (*)	7/40	40 %	60/1340 Hz	11.5 Hz	10 / 150 m	18 m.s $^{-1}$
city (*)	5/25	25 %	60/2000 Hz	20 Hz	2.5 / 40 m	18 m.s $^{-1}$
Epidaurus (*)	26/36	37.5 %	53/1440 Hz	7.5 Hz	10 / 130 m	2 m.s $^{-1}$

to speed-up visibility checks. The maximum frame rates for the extrapolation scheme represent the absolute update speed for all the sequences (no video synchronization). The *average speed* column indicates the average speed of the sound source, the receiver remaining still. The *maximum error* column represents the maximum difference between predicted and actual position of the image sources. For reference, we also show the average paths lengths for which this maximum difference occurs. Our extrapolation scheme proved stable even when the paths update rate is as low as 2Hz. On the Octane 225 MHz platform, it allows for extrapolating 60000 3D vectors per second. As expected, the extrapolation process introduces a positional error for the image-sources. The resulting predicted trajectory usually corresponds to the exact trajectory but with an additional latency since the extrapolation induces some filtering. For the cube example, we measured a global latency of 0.2 s. for 9Hz updates and 0.5 s. for 4.5Hz updates. This means that the extrapolation process can cause auditory information to be desynchronized with the visual information if the "real" positions are used for display purposes. This latency increases with the speed of the source (relative to the listener). Thus, for fast moving sources, it is necessary to maintain higher update rates. On the Onyx2 platform, we were able to add simple auralization (no filtering) of up to the 25 shortest paths (in the case of the cube) at CD quality while maintaining artifact-free sound. An 8-speaker array was used for reproduction and the signal was processed by blocks of 1024 samples (extrapolation was thus performed at 43Hz).

4. CONCLUSION

We proposed an original approach to asynchronous update of sound propagation paths and auralization, in the context of interactive geometrical acoustics. We showed that extrapolation can be used for this purpose and presented a non-linear extrapolation scheme which is robust and can predict efficiently the value of any attribute from sparse, non-uniform updates in time. This technique leads to more efficient and flexible processing pipelines for virtual audio. If the geometrical simulation cannot be performed at high rates or if some delay occurs in the simulation, the auralization engine will rely on predicted values to output artifact-free sound. Moreover, geometrical updates can be slowed down to give more processing power to the auralization stage and better balance computational load.

5. ACKNOWLEDGMENTS

The author would like to thank Jean-Paul Vian, Jean-Dominique Gascuel and Tom Funkhouser for providing or creating the 3D models used in the tests.

6. REFERENCES

[1] H. Lehnert and J. Blauert, "Principles of binaural room simulation," *Applied Acoustics*, vol. 36, pp. 259–291, 1992.

[2] M. Kleiner, B.I. Dalenbäk, and P. Svensson, "Auralization - an overview," *J. of the Audio Engineering Society*, vol. 41, no. 11, pp. 861–875, Nov. 1993.

[3] L. Savioja, J. Huopaniemi, T. Lokki, and R. Väänänen, "Creating interactive virtual acoustic environments," *J. of the Audio Engineering Society*, vol. 47, no. 9, pp. 675–705, Sept. 1999.

[4] Jean-Marc Jot, "Real-time spatial processing of sounds for music, multimedia and interactive human-computer interfaces," *Multimedia Systems*, vol. 7, no. 1, pp. 55–69, 1999.

[5] T. Funkhouser, P. Min, and I. Carlbom, "Real-time acoustic modeling for distributed virtual environments," *ACM Computer Graphics, SIGGRAPH'99 Proceedings*, pp. 365–374, Aug. 1999.

[6] F.R. Moore, "A general model for spatial processing of sounds," *Computer Music Journal*, vol. 7, no. 3, pp. 6–15, Fall 1983.

[7] Nicolas Tsingos and Jean-Dominique Gascuel, "Soundtracks for computer animation: sound rendering in dynamic environments with occlusions," *Proceedings of Graphics Interface'97*, pp. 9–16, May 1997.

[8] Nicolas Tsingos, Thomas Funkhouser, Addy Ngan, and Ingrid Carlbom, "Geometrical theory of diffraction for modeling acoustics in virtual environments," Tech. Rep. 10009662-000802-03TM, Bell Laboratories, Jan. 2000.

[9] G. Picinbono and J.C. Lombardo, "Extrapolation: a solution for force feedback ?," *Proceedings of International Scientific Workshop on Virtual Reality, Laval, France*, June 1999.

[10] C.K. Chui and G.Chen, *Kalman Filtering with real-time applications, 2nd ed.*, Springer-Verlag, 1991.