Realtime audiovisual rendering and contemporary audiovisual art*

TAPIO LOKKI, JARMO HIIPAKKA, RAMI HÄNNINEN, TOMMI ILMONEN, LAURI SAVIOJA and TAPIO TAKALA

Helsinki University of Technology, Telecommunication Software and Multimedia Laboratory, PO Box 5400, FIN-02015 HUT, Finland
E-mail: Tapio.Lokki@hut.fi
URL: http://www.tcm.hut.fi/Research/DIVA/

Visual rendering is the process of creating synthetic images of digital models. The modelling of sound synthesis and propagation in a virtual space is called sound rendering. In this article we review different audiovisual rendering techniques suitable for realtime rendering of three-dimensional virtual worlds. Virtual environments are useful in various application areas, for example in architectural visualisation. With audiovisual rendering, lighting and acoustics of a modelled concert hall can be experienced early in the design stage of the building. In this article we demonstrate an interactive audiovisual rendering system where an animated virtual orchestra plays in a modelled concert hall. Virtual musicians are conducted by a real conductor who wears a wired data dress suit and a baton. The conductor and the audience hear the music rendered according to the acoustics of the virtual concert hall, creating a lifelike experience.

1. INTRODUCTION

While sitting in a concert hall and listening to a symphony orchestra playing, have you ever imagined being the conductor? As the conductor you could control the whole orchestra and the interpretation of the music would be in your hands. You could make ritenutos and accelerandos whenever you wanted and the whole orchestra would follow the movements of your baton. Unfortunately, symphony orchestras are not available to everyone. If you had an opportunity to try a real orchestra the musicians might not be that willing to play as a rehearsal orchestra. But, if the concert hall and musicians were not real, they could play any time you wished. This kind of orchestra could be encountered in virtual reality.

To be able to conduct virtual musicians in a virtual concert hall and at the same time see and hear them playing requires, among other things, realtime three-dimensional (3D) audiovisual rendering. Methods exist for creating photorealistic computer graphics as do methods to simulate the concert hall acoustics. Modern sound synthesis methods realistically produce the sound of musical instruments and the human–computer interaction is getting more and more usable. The animation of human models already resembles the real movements of human beings.

All of the above-mentioned research fields are needed for interactive (realtime) audiovisual rendering. Unfortunately, the computational requirements of sophisticated virtual reality methods are prohibitive. Some compromises must therefore be made. One example of interactive audiovisual rendering is the virtual orchestra created by the Digital Interactive Virtual Acoustics (DIVA) research group. In this article we present the techniques with which this kind of virtual orchestra can be created.

Realtime audiovisual rendering can be used in contemporary multimedia performances. Typically in such performances the tempo of the musical piece has to be determined in advance so that the music and the animated computer graphics can be synchronised. However, this *a priori* determined tempo limits the interpretation of the conductor. With the DIVA conductor following and animation system presented in this paper, such tempo limitation is avoided. The integrated conductor following and animation process described here takes into account the irregularities in tempo imparted by a conductor.

Similar problems occur in the field of contemporary music where there can be MIDI-driven electronic instruments in the orchestra. Whereas previously these compositions would have to be performed at a predetermined tempo, to suit the fixed tempo of the MIDI system, the DIVA system's conductor following process is able to direct the tempo of the MIDI instruments.

Another aspect of the DIVA system is the ability to produce spatialised sound with multiple loudspeakers. The system enables composers to place virtual musicians and sound sources around the concert hall and thus increase the immersion of listeners in the soundfield, such that the listener feels that s/he is sitting within the orchestra.

* The authors would like to thank the personnel of the Acoustics Laboratory at Helsinki University of Technology for their support in the project. The DIVA group has been partially financed by the Academy of Finland (Tacit Knowledge and VSOP projects) and the Technology Development Centre, Finland.

https://doi.org/10.1017/S1355771898003069 Published online by Cambridge University Press

The DIVA system has been used in several audiovisual presentations. For example, in the summer of 1998, an 'augmented orchestra' gave a concert in the Museum of Contemporary Art in Finland. The augmented orchestra consisted of a real string quintet, a virtual flautist, a guitarist and bass player. The virtual musicians followed the conductor's tempi successfully in the concert. The virtual flautist has also played as a soloist in two SYNK performançes organised by Finnish composers Ilkka Niemelainen and Otto Romanowski.

The article is organised as follows. In section 2 the principles and techniques of audiovisual rendering are discussed. The DIVA system is overviewed in section 3 along with the presentation of each subpart of the system. The system configuration of DIVA is presented in section 4 and section 5 concludes the article.

2. AUDIOVISUAL RENDERING

Audiovisual rendering is necessary to see and hear virtual spaces. Visual rendering is the process of creating artificial images of data models. The quality of visualisation must be balanced with available computational resources and accuracy of the visual model. With accurate but slow offline rendering it is possible to reach photorealistic results. In realtime rendering all shadows are typically missing and models cannot be as complex and large as in non-realtime rendering. Examples of scenes rendered in nonreal time and real time are presented in figures 1 and 2, respectively. The non-realtime model consists of 565,481 polygons while the realtime model has only 481 polygons.

In sound rendering the behaviour of sound in a space is modelled. The goal is to create a soundscape so realistic that the user cannot tell if it is artificially created or not. A rendered scene can be reproduced either with loudspeakers or with headphones. Unfortunately, as in visual rendering, realtime rendering seldom produces a perfectly natural soundscape.

2.1. Modelling of the virtual space

In the first stage of creating a virtual space, a 3D model of the space must be designed. Typically this consists of flat polygons that represent the surfaces of the model. Other model types are also possible, e.g. volume models or models with curved surfaces. However, typical rendering methods are best suited to polygonal models where curved surfaces are approximated with many flat polygons. The number of polygons depends on the curvature of the surfaces and on the available computational capacity.

Visual and acoustical parameters can be associated with the surfaces of a model to recover visual and acoustical properties necessary for creating more realistic virtual environments. Typical graphical parameters include textures and colour attributes while acoustical parameters specify absorption and diffusion coefficients for various frequency bands.

2.2. Physical similarities of light and sound

Both light and sound sources emit waves which reflect from surfaces eventually reaching a viewing and listening point, i.e. the observer. The essence of virtual space is that these wave propagations are modelled and reproduced artificially.

To understand the rendering process better, let us first consider the similarities of light and sound (Takala and Hahn 1992, Astheimer 1995). In particular, both are created by a source that radiates waves, and the waves are reflected from walls and other obstacles. The edges of obstacles cause diffraction while refraction occurs whenever medium density changes. However, their different propagation velocity (sound, $340\,\mathrm{m\,s^{-1}}$; light, $3\times10^{8}\,\mathrm{m\,s^{-1}}$) and average wavelength cause essential practical differences. To be more specific, light does not manifest noticeable propagation delay, and its wave nature appears only when it interacts with structures of microscopic size. On the other hand, slow propagation of sound causes noticeable delays that are perceived as echoes and reverberation. The wavelength of sound (from 0.02 to 17 m) is also so long that diffraction occurs with normal human-size objects.

2.3. Visual rendering

Visual rendering methods can be divided into local and global illumination models (Foley, van Dam, Feiner and Hughes 1990, Glassner 1995). In local methods the intensity of light on a surface depends only on its location and orientation with respect to light sources. Various shading models provide examples of such techniques. Flat shading is the simplest and yields constant intensity over the whole surface of a flat polygon (see, for example, figure 2 which was produced with flat shading). The Gouraud shading method introduces linear interpolation across polygon surfaces so that adjacent polygons appear as a continuous smooth surface. Phong shading provides specular highlight effects. These shading methods are computationally easy and they are suitable for realtime rendering, but they do not produce very realistic results.

In the global models, such as ray-tracing and radiosity, there is interaction between objects such that the intensity of light and colour of an object depend on the other objects surrounding it, in addition to light sources acting upon it. These techniques can produce highly realistic scenes having shadow effects, for example. Ray-tracing best suits

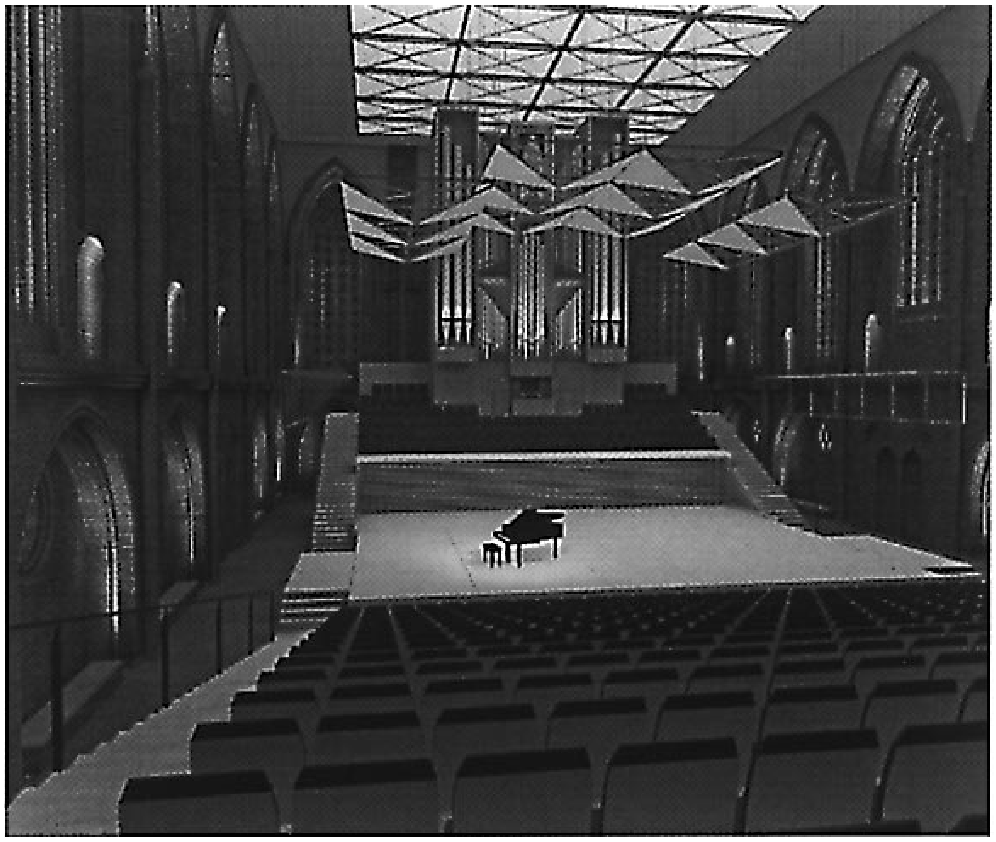

Figure 1. Example of non-realtime visual rendering (global illumination model with accurate lighting) © 1998 Erkki Rousku.

scenes consisting of shiny objects, whereas radiosity is optimal for diffuse reflections. An example of global illumination is presented in figure 1, which is created with a combination of radiosity and ray-tracing techniques. Unfortunately, these methods are computationally so complex that the illumination calculation is always a time-consuming offline task. A useful combination of local and global techniques is such that the illumination is calculated beforehand with the radiosity method and the result is presented as textures projected onto surfaces of the model, thus making quite realistic realtime rendering possible.

2.4. Sound rendering

When we are enjoying a musical performance in a concert hall, the sounds we hear are invariably accompanied by thousands of delayed reflections from many different directions. If the reflections occur soon after the initial sound, the result is not perceived as separate sound events. Instead, the reflections modify the perception of the sound, changing the loudness, timbre, and most importantly, the spatial characteristics of the sound (Gardner 1998).

To obtain a natural-sounding virtual environment, the reflections and their incoming directions have to be simulated. Fortunately, only direct sound and the first reflections have to be created precisely because our perception makes the decision of the position of sound source mainly based on these. The late reverberation gives information of the size of the space and is often approximated with exponentially decaying noise.

In the first part of 3D sound rendering, the sound reflections are calculated. This part is called room acoustic modelling. In the auralisation part of sound rendering all the simulated reflections and especially their incoming directions have to be reproduced for the listener. Auralisation principles are presented in more detail by Kleiner, Dalenback and Svensson (1993) and Begault (1994).

2.4.1. Room acoustic modelling

The sound rendering methods can be roughly divided into two categories according to their functional principles (Kuttruff 1995). In the wave-based methods, such as the finite element, boundary element and finite difference methods, the wave equation is

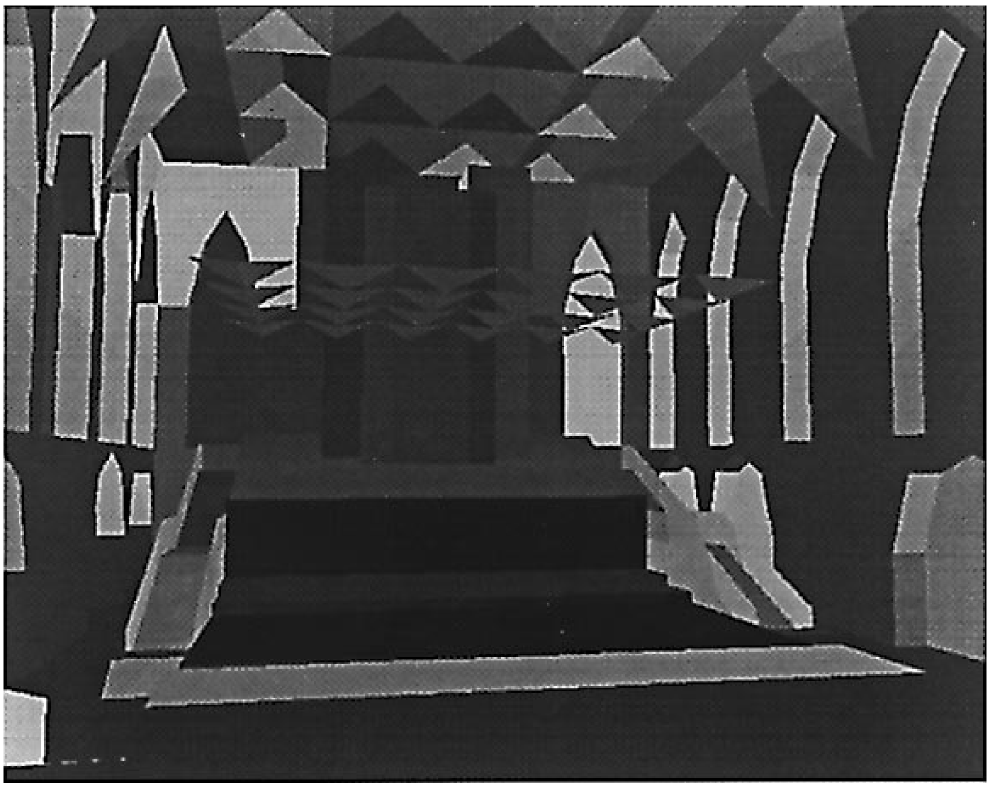

Figure 2. Example of realtime visual rendering (local illumination with flat shading and without textures).

numerically solved in a modelled space. These methods take into account the diffractions and interferences and give physically correct results, but they are far from realtime implementations. Also, the large memory requirements make these wave-based methods impractical in typical virtual reality applications. Nevertheless, in some areas these techniques are applied, for example in the modelling of automotive acoustics (Granier, Kleiner, Dalenback and Svensson 1996).

The other category of sound rendering involves ray-based methods. In these methods, such as ray-tracing (Krokstad, Strom and Sorsdal 1968) and the image–source method (Allen and Berkley 1979, Borish 1984), the wave nature of the sound is neglected and the sound is supposed to act like rays. In this case all the wave phenomena, like diffraction and interference, are excluded. In spite of these simplifications, the ray-based methods give natural-sounding results even in real time. These techniques, especially ray-tracing, are very close to ray-tracing in visual rendering.

2.4.2. Auralisation and reproduction techniques

One of the most important things in sound rendering is that the sound events (direct sound and reflections) have to be reproduced so that the listener hears the sounds coming from certain directions. With normal stereo panning this is not possible.

A brute-force reproduction technique is to put one loudspeaker in every possible location where a sound event is intended to happen. Of course this is very expensive and impractical. A better solution is to produce virtual sources with two or more loudspeakers. The virtual source is a source in listener's auditory space that does not correspond to any physical sound source.

With virtual sources the number of loudspeakers can be reduced so that the accuracy of the directional perception is not ruined. Pulkki (1997) has presented a vector base amplitude panning (VBAP) method which allows the arbitrary positioning of virtual sources with a small number of loudspeakers. In VBAP at least three loudspeakers are positioned around the listener. The VBAP method is based on normal amplitude panning in three dimensions.

In many situations, the required loudspeakers for VBAP reproduction are not available and the reproduction has to be handled with a pair of loudspeakers or with headphones. To be able to reproduce the direction of sounds with this kind of equipment, modelling of the spatial hearing is needed. This modelling is usually done with head-related transfer functions (HRTFs). These model the transfer functions from all directions in space to the listener's ear canal (Moller 1992).

The principle in HRTF reproduction techniques is to control as accurately as possible the sound signal at the entrance of the listener's ear canal. In other words, both ears should get exactly the same sound signal that they would receive in a real situation, when the sound comes from a certain direction. By controlling the sound in the ear canal (not in the loudspeaker), virtual sources outside the loudspeaker aperture can also be created. Three-dimensional sound reproduction techniques with a single loudspeaker pair are extensively discussed by Gardner (1997).

The HRTFs are laboriously measured in an anechoic room with expensive equipment (Moller, Hammershoi and Jensen 1995). The results of the measurements, namely the HRTF functions, have complex magnitude and phase characteristics. This means that the filter design implementing the HRTF is a difficult problem (Huopaniemi, Zacharov and Karjalainen 1999) and when the filters are applied to a virtual acoustics system the computational requirements are high.

Despite the computational complexity, HRTF techniques are common in auralisation systems. One reason for this is that it is possible to use headphones in reproduction. With headphones, the acoustics of the listening room does not affect the perceived sound and a controlled sound signal to each ear canal is guaranteed. One feature that can be implemented only with HRTFs is that in headphone reproduction the virtual sources can be localised outside the head, not only between the ears.

2.5. Established tools for producing virtual worlds

The Virtual Reality Modelling Language (VRML) (JTC/SC24 1997) was an early standard which allowed interaction between user and virtual worlds. It is a file format for describing interactive 3D objects and virtual worlds. VRML is designed to be used in Internet applications where a Web browser takes care of audiovisual rendering. In VRML the main emphasis has been in visual rendering. It does not offer high-quality sound rendering.

Other programming interfaces that offer tools for audiovisual rendering are Java 3D (Java3D 1998) and MPEG-4 (JTC1/SC29/WG11 1999). The new MPEG-4 standard is also intended to include VRML.

3. THE VIRTUAL ORCHESTRA

This paper presents the DIVA system (Takala, Hanninen, Valimaki, Savioja, Huopaniemi, Huotilainen and Karjalainen 1996) as an application of realtime 3D audiovisual rendering. The DIVA research group studies the integration of visual and aural elements in interactive and dynamic virtual environments.

A virtual orchestra is a demanding realtime application because live music performance is an art form where timing and cooperation of musicians must be smooth and accurate. The concept of a virtual orchestra binds together many fields of science, ranging from acoustics and music theory to computer graphics and human behaviour.

In our present orchestra, four virtual musicians play their virtual instruments in a modelled concert hall. The animated players play their instruments with realistic fingerings in time with the rhythm of baton movements and produce sounds that are generated with physical models of musical instruments. The players stand on the stage of a virtual concert hall, with acoustics simulated and auralised to the user. In this section the four subparts of the DIVA system (see figure 3) are presented: conductor following, animation, modelling of musical instruments and sound rendering.

3.1. Conductor following

A conductor follower interprets the movements of a conductor and shares the interpretation with other

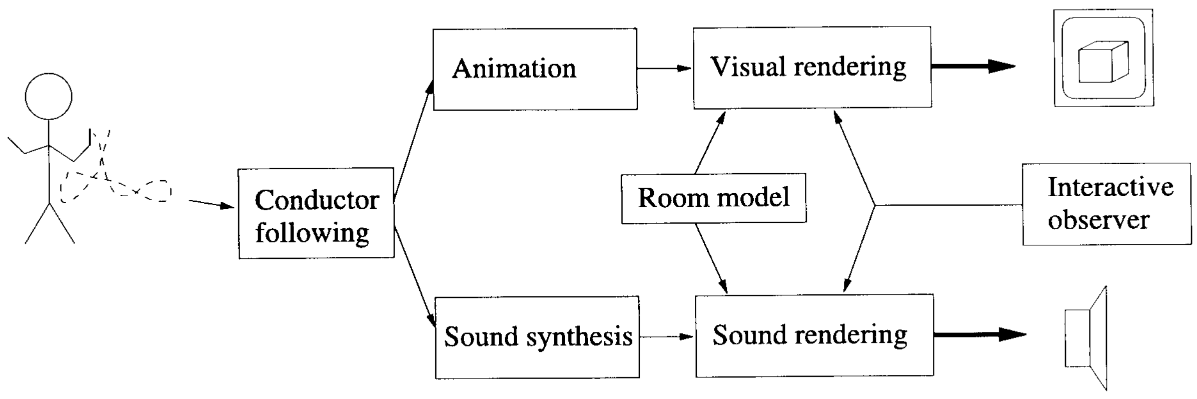

Figure 3. The main components of the DIVA system. The conductor follower sends synchronisation signals to both the animation and sound synthesis modules which provide input to rendering processes. The other user, the observer, may move around the virtual space and watch and listen to the virtual orchestra in a virtual space.

components of the DIVA system. This interpretation is expressed by synthesising the instrument sounds and animating the musicians. The aim is to simulate a musician (or group of musicians) playing a piece. This involves interpreting the gestures (movements) the conductor makes and interpreting the music accordingly. Following the rhythm of the conductor has been our main research goal. The simulated musician cannot cope with all possible conducting styles, rather it works with reasonably standard conducting technique (see, for example, McElheran 1989).

The starting point of the computation is tracking the conductor motion. We use a magnetic tracker as a motion capture tool. Small sensors are attached to the conductor's clothing and the tracker measures the position and orientation of these sensors. The mounting mechanism is shown in figure 4.

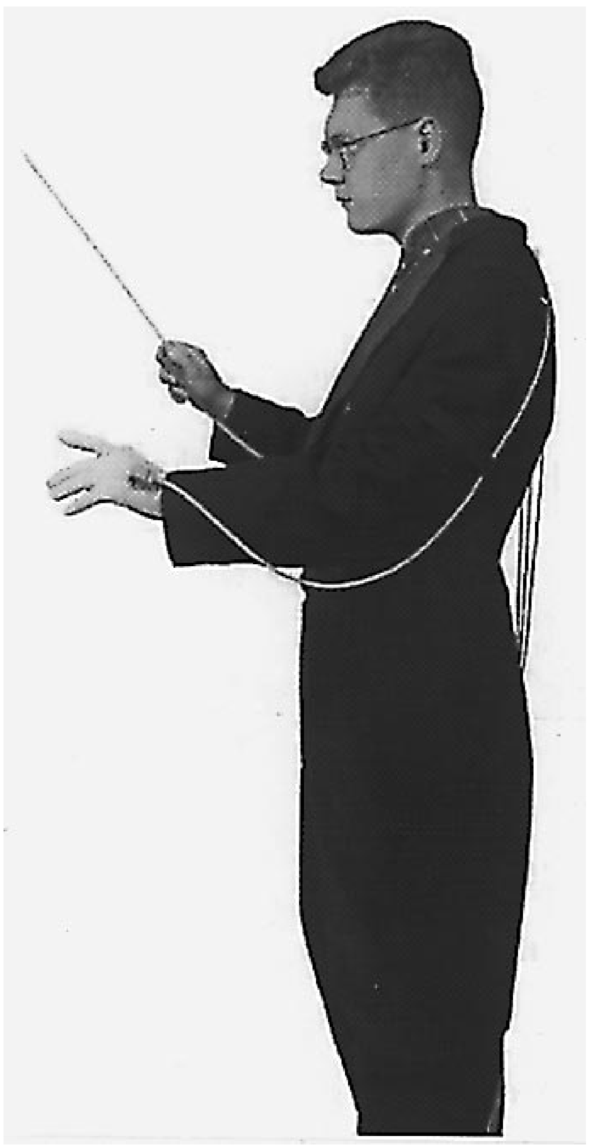

Figure 4. Conductor in a data dress suit.

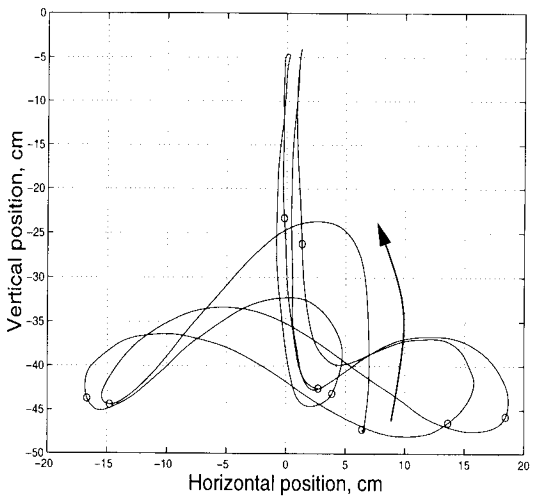

Figure 5. Measured baton tip motion for the time signature 4/4.

The rhythm is expressed mostly with the beats of the baton. A great deal of information can be gained by observing only the vertical motion. Figure 5 shows a detailed measurement of the conductor's baton tip motion. In our implementation, while the horizontal movement is important for time signature recognition, the vertical motion alone indicates the tempo.

A down-beat indicates a strong part of a bar. The tempo of the piece can be estimated by measuring the time between successive down-beats. Finding down-beats is easy enough: As the baton tip velocity changes from downward to upward, a beat is detected. The down-beat method is a direct way to calculate the tempo. We use it in conjunction with more elaborate methods to overcome its limitations.

When one uses down-beats alone for tempo tracking, the tempo cannot be adjusted unless a new beat takes place. During quick tempo changes one needs to react more quickly, even before the beat is detected. Our system tracks the cyclic conductor motion phase continuously and estimates when the next beat will take place. We use an artificial neural network (ANN) (see, for example, Rojas 1996) to estimate current motion phase.

The follower updates its interpretation of the rhythm continuously. This gives it the possibility to react very quickly to tempo changes. The speed of reaction can be adjusted as necessary to allow different interpretation of styles. The follower can predict when the next beat will take place. This capability is often useful when countering delays in data processing. Without this prediction process, the music would lag behind the conductor regardless of how s/he conducts.

The follower can also detect other gestures from the motion. For example, heuristic rules have been used to track dynamics gestures (louder and softer). In these cases the gesture is detected if the motion fits certain criteria – fast enough, persistent enough, etc. A more comprehensive presentation of the DIVA conductor follower can be found in Ilmonen (1999).

The gestures have a context – the music. The follower must know the piece to utilise fully the detected gestures – just as real musicians have their scores. Otherwise there would be a lot of uncertainty regarding the way in which the system should react. The follower is given musical hints about the piece before the playing starts; expected tempo, time signature, dynamics and nuances. Using this knowledge the virtual musician can understand detected gestures with greater confidence.

Creating musical expression – modelling the musician playing the piece – is a problem with few solutions today. How exactly does a musician play the music? Mimicking a musician playing a piece is difficult if there is no comprehensive model of how a musician plays in an orchestra. Using scientific guesses about the processes in a musician's mind, one can create algorithms that produce expression. Whether this computational expression is good or not depends on the quality of the heuristic rules that control the playing. In practice, different rules are tried out and the best-sounding ones are used.

How well does the simulated orchestra fare? A number of people have conducted our virtual orchestra – from ordinary folk to professional conductors. Of these the conductors are the most demanding users. Others do not realise how the reaction of the virtual orchestra differs from a real orchestra. The current heuristics mimicking a player can react quickly to changes in conducting style, but the results can be unmusical. Since the follower tracks the conductor constantly, detecting his movements over thirty times per second, the follower can react quickly. In comparison, a normal musician can spend more than half of the time looking at the score. Unmusical output is caused by imperfect player modelling. When changing tempo, the player model can make computationally very subtle, but audible mistakes in note timing.

3.2. Animation

Fluent and convincing character animation is a significant factor in scene realism, more significant in fact than detailed models or brilliant rendering. Human motion animation in particular must be executed carefully as we are so accustomed to seeing natural human movement, that even the smallest motion irregularities are easy to spot.

Rendering detailed human body models with complete bone and muscle structure covered by elastic skin is computationally very expensive. To decrease the rendering load to a level more suitable for realtime animation, human body models must be made as simple as possible. A 3D cartoon puppet made of solid nonelastic components with rotational joints provides such a simple model.

To make a virtual musician's behaviour believable, virtual instruments must be operated with sufficient detail and accuracy. In particular, delicate control of the musician's hands and fingers is required. Furthermore, respiration and other secondary actions should also agree with the music. Such animation can be composed by hand by a skilful animation artist, but this work is tedious and slow. An automatic system that could generate musician animation directly from music notation would be greatly appreciated (Lytle 1990). Such a task can be broken into two phases, where the first phase extracts instrument fingering information from music notation, and the second generates an animation sequence from that data. Both phases can be carried out before the virtual performance is actually committed, thus relaxing realtime computational load. Related but separate realtime tasks that still remain are how to synchronise animation playback with live conductor tempo, and how to combine the precomputed animation with dynamic events that may happen during the show.

3.2.1. From a MIDI file to fingering

Musicians operate their instruments with grips that dictate the actions they must take in order to produce any given sound from an instrument. A grip is composed of *grip components* that describe where relevant musician body parts must reside at the time the grip is executed. A grip component may be static, requiring, for example, that a finger just rests on a vent, or it may be dynamic, involving movement like slapping a percussion instrument or a guitar string. Dynamic grip components therefore contain several timed events that are expected to happen in sequence before and after the actual sound commences.

The goal of an automatic fingering resolver is to find a sensible sequence of instrument grips that will guide an animated virtual musician through a music piece. Our fingering resolver implementation does this. It operates on MIDI note on and note off events, taken from a standard MIDI file. Aside from MIDI file input, a target instrument description that describes the instrument controls and musical capabilities is also needed.

The fingering resolver steps through MIDI events, and reacts whenever a new sound begins or an old one ends. At each reaction, a search is made in the target instrument database to find a grip that matches with the given set of sustained and awakened notes. Sound pitch, velocity and expected duration are the search factors used.

The fact that many instruments feature several alternate fingerings for a given set of notes complicates the grip selection process, as one query may well result in several alternative matches. The best match in any given situation will depend on the context of other grips occurring immediately before and after the grip. This context remains partially unknown until all grip candidates have been found, and therefore the final grip selection is postponed until the resolver has proceeded through all the events.

After the resolver has completed the MIDI event sweep, it has collected a complete list of candidate grip sets. A sequence of grips that match each other well must be selected next. One way to do this is to define a distance function that measures the distance between any two grips. With such a distance function, the length of any grip sequence can be computed. By using a technique called critical path analysis, the grip sequence with minimum length can then be searched for. The resulting sequence represents the optimal set of grips with respect to the criteria set by the distance function.

There is no clear answer to what kind of distance function would be the best to measure distances between grips. Clearly the function should depend on the target instrument. Our general solution is to use the squared sum of the distances the musician's fingers must move to reach one grip from another. Such distances depend on the locations of the target instrument controls. Fortunately, this information can be easily stored into the instrument control and musical capability database.

Finally, it must be emphasised that for the algorithm to work successfully, the whole MIDI sequence must be known in advance. Usually this is not a problem, since it is not unreasonable to assume that musicians know in detail what they are going to play. However, improvisation is a completely different matter, and is not solvable with this method.

3.2.2. From fingering to animation

Once the fingering sequence has been solved, it is possible to transform it into virtual musician animation. As mentioned earlier, each grip is associated with data that describes where the musician's body part must reside at the time of the execution of the grip. Taken together, this data describes a sequence of key-frames. A key-frame is a fundamental animation term that specifies the posture of an animated character at a given time. Given two key-frames, the character is assumed to be moving from the first posture to the next with natural smoothness and within the time interval between the two key-frames.

The details of the posture an animated character must take in each key-frame is resolved with a process called *inverse kinematics*. Inverse kinematics computes the joint angles necessary for a mechanical arm fixed at one end to extend and touch with the other end a specific point somewhere within its reach. In this case, the process is applied to the arms and fingers of our animated puppet musician. The actual inverse kinematic process is a complex and computationally expensive task, in particular because the mechanics of human hands are very complex. Furthermore, there is usually no single best solution to the way in which a musician's arms and fingers are placed on given instrument controls, but instead an infinite number of closely related alternatives. Fortunately, all this time-consuming inverse kinematics can be executed in advance, without overloading the later realtime performance act.

After individual key-frames have been computed, displaying them in sequence immediately results in a crude animation of a musician playing an instrument. However, such an animation will probably suffer from some jerkiness, as transitions from one grip to another happen instantly. To further enhance the animation, smooth transitions between key-frames must be computed by filling the gaps between key-frames with intermediate animation frames that are interpolated from the key-frames.

3.2.3. Other movements

Musicians sway, dance and gesture while playing. They also respire, look around and blink their eyes. These movements are of secondary importance with respect to the actual instrument manipulation, but for a realistic virtual musician display, they are absolutely essential. Unfortunately, MIDI files contain little or no information to guide such additional actions.

Autonomous character movement animation is still under intense research. Various rule-based approaches in particular have demonstrated some encouraging results (Masataka, Hidaka, Matsumoto, Yosuke and Yoichi 1996, Perlin and Goldberg 1996). Our musicians currently manifest some simple behaviour, too, that allows them to look at an approaching observer, and blink their eyes. These simple control processes already make the musicians feel more real, but more advanced dynamic animation is still required. Our current opinion is that realistic animation will arise from combining several simpler behaviour models together. This will most probably require some physical modelling as well, that will take into account the masses and balance of the virtual musicians.

3.3. Physical models of musical instruments

Physical modelling of musical instruments is an exciting paradigm in digital sound synthesis. While earlier synthesis methods try to imitate the properties of the sound signal, the physical modelling approach focuses on the imitation of the sound source itself.

Indeed, the basic idea behind physical modelling is that when the vibrating structure is modelled with sufficient accuracy, the resulting sound will automatically be identical with the real sound of the modelled physical structure.

Valimaki and Takala (1996) divide the physical modelling techniques into five categories:

(1) source–filter modelling,
(2) numerical solution of partial differential equations,
(3) vibrating mass–spring networks,
(4) modal synthesis, and
(5) waveguide synthesis.

In this article we concentrate only on waveguide modelling, which is the most important and widely used physical modelling method both in academic and commercial applications.

3.3.1. Digital waveguide models

The theory of digital waveguides has primarily been developed by Smith (1987, 1998), but the method was first applied to artificial reverberation (Smith 1985). The method suits the simulation of one-dimensional resonators particularly well (vibrating strings, acoustic tubes, thin bars), which are found in many families of musical instruments. The method has also been generalised to two (VanDuyne and Smith 1993) and more dimensions. These generalisations allow the method to be used, for example, in the modelling of vibrating membranes and in simulating room acoustics (Savioja, Rinne and Takala 1994).

A well-known forerunner of the waveguide synthesis model is the Karplus–Strong algorithm developed for the synthesis of plucked string instrument sounds (Karplus and Strong 1983). The block diagram of the original Karplus–Strong model is presented in figure 6. The delay line is initialised with

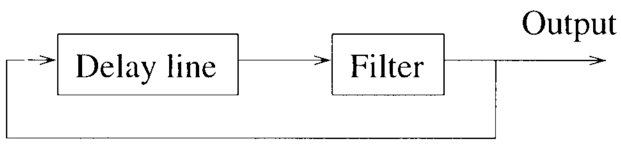

Figure 6. The original Karplus–Strong string model.

white noise and the output of the delay line is fed to a digital low-pass filter called the loop filter. The output of the Karplus–Strong model is the filtering result, and this output signal is also fed back into the delay line. The comb-filter type structure of the Karplus–Strong model selectively attenuates the frequency components not coinciding with the resonances of the delay line loop, and the signal in the delay line turns into a quasi-periodic signal with a clearly perceivable pitch. In the original paper by Karplus and Strong (1983), the use of a two-point averaging filter as a loop filter was proposed, but most implementations now use a more sophisticated filter structure.

The original Karplus–Strong model has been further extended by Jaffe and Smith (1983), and has later led to a number of detailed models of string instruments (Karjalainen and Valimaki 1993, Smith 1993). A recent plucked string instrument synthesis model by Valimaki and Tolonen (1998) includes the two polarisations of the string vibration as well as the sympathetic couplings between the strings. The derivation of the more advanced waveguide string instrument models and the relation between Karplus–Strong and waveguide formulations is fully explained in Karjalainen, Valimaki and Tolonen (1998).

Waveguide modelling has been successfully applied also to the synthesis of wind instrument sounds, e.g. clarinet (Smith 1986, Valimaki, Karjalainen, Janosy and Laine 1992) and flute (Karjalainen, Laine, Laakso and Valimaki 1991). The woodwind bore models have later been extended to contain models of the finger holes (Valimaki, Karjalainen and Laakso 1993). Cook (1991) has introduced a physical model capable of producing brass instrument sounds.

3.3.2. Realtime implementation

Realtime implementation emphasises some aspects of the instrument models. First of all, the models have to be as simple as possible without excessively compromising the sound quality. Of course the consideration of the simplicity vs sound quality depends on the specific application; as the computer resources become more affordable over time, more complex models become usable in realtime applications. Also, the actual implementation of the instrument model must be carried out with great care. This includes selecting the algorithms so that realtime operation is feasible and also considering the trade-offs between memory and CPU requirements appropriate to a given target platform. When designing waveguide instrument models, the most critical components from the implementation point of view are the delay lines and digital filters used by the instruments.

The implementation of the parameter control can also cause problems for realtime instrument models. Physical models typically provide very flexible control over the expressive qualities of the synthesised sound, but controlling the model parameters in a realtime environment is far from simple. The current standard in synthesizer control is the MIDI standard, which is rather limited and overly keyboard-oriented to be a universal control scheme for different types of synthesizers. In practice, MIDI is still the only method commonly available for musical instrument control, and for the synthesizer to be usable the MIDI standard has to be supported. The expression control of the different types of physical instrument model has been considered by Valimaki, Huopaniemi, Karjalainen and Janosy (1996), Jaffe and Smith (1995) and Scavone (1996).

In the DIVA system, three instrument models are used: a flute, an acoustic guitar and a double bass. The flute model currently in use is a slightly modified version of the flute proposed by Karjalainen *et al.* (1991). The guitar model is the advanced model by Tolonen (1998) and the double bass is a simpler plucked string instrument model. All the instruments in the DIVA system are controlled via a standard MIDI interface so that any other synthesizer can be easily used as a sound source in the distributed DIVA system.

3.4. Sound rendering in the DIVA system

The realtime room acoustic simulation method used in the DIVA system is called a parametric room impulse response rendering method. In this method the actual rendering process is done in several parts (see figure 7). The first part consists of direct sound and early reflections, both of which are time- and place-variant. The latter part of the rendering process represents the diffuse reverberant field, which is treated as an invariant process.

As illustrated in figure 7, the direct sound and early reflections are modelled with the image–source method (Allen and Berkley 1979, Borish 1984) and the late reverberation is created with an efficient recursive digital filter structure (Vaananen, Valimaki and Huopaniemi 1997). The image–source method is a ray-based method in which the sound is considered to propagate as rays. This is quite a raw model of sound propagation, but works surprisingly well. The image–source method is used to obtain the locations and orientations of a sound source together with image sources that represent reflections from the surfaces of the modelled space (see figure 8). From the locations of the sound source and image sources the delay, gain and direction relative to the listening point can be calculated. The modifications to the frequency response of the reflections caused by wall materials and air absorption can be taken into account by adding digital filters after the propagation delay. More information about the reflection and absorption filters used is presented by Huopaniemi, Savioja and Karjalainen (1997).

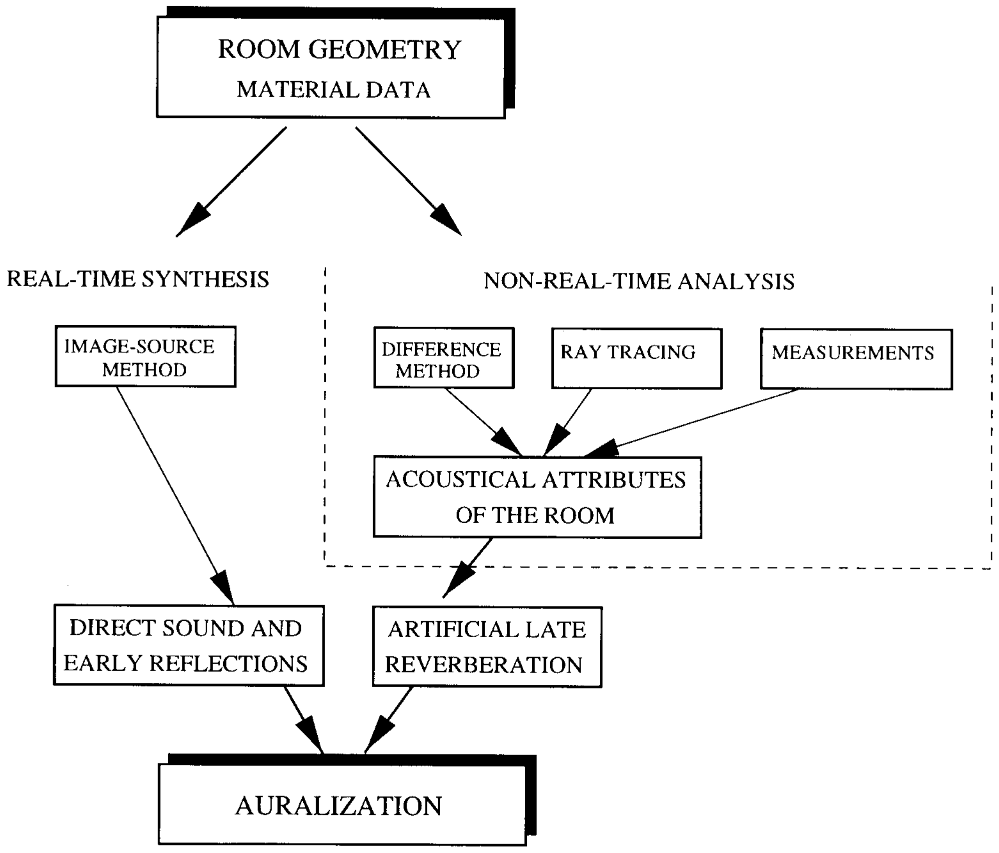

Figure 7. The sound rendering method used in the DIVA system. The model is a combination of the realtime image–source method and artificial late reverberation which is parametrised according to room acoustical parameters obtained by simulation or measurements.

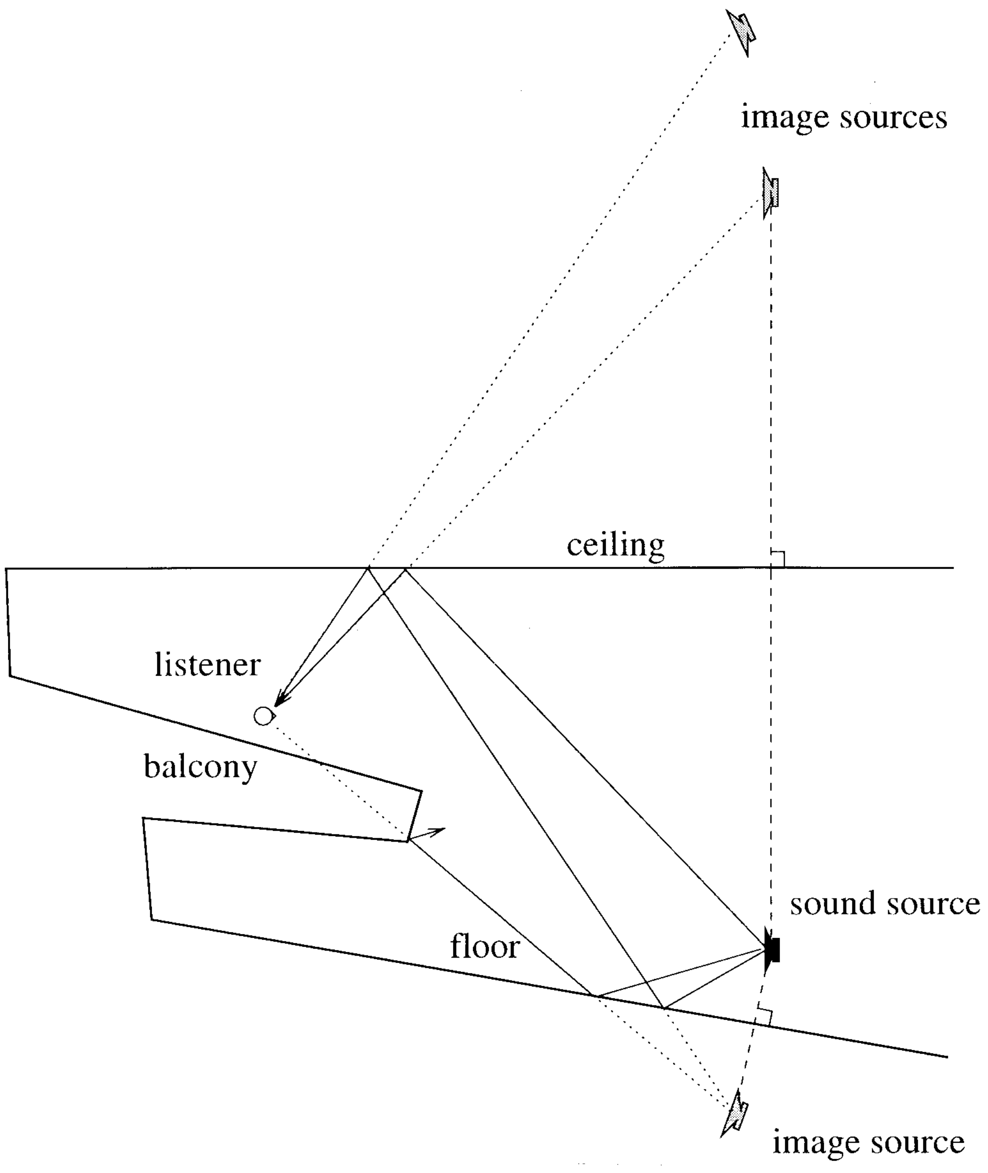

Figure 8. In the image–source method the sound source is mirrored against all surfaces to produce image sources which represent the corresponding reflection paths.

The late reverberation algorithm used in realtime sound rendering has to fulfil the requirements of natural-sounding reverberation: exponentially decaying reverberation, a high modal density, a high reflection density and frequency-dependent reverberation time (due to air absorption). Many different algorithms have been invented to simulate the late reverberation efficiently. A good overview of these algorithms is presented by Gardner (1998). Our late reverberation algorithm (Vaananen *et al.* 1997) fulfils all these requirements and produces a good spatial impression.

In an interactive system where the listener can move freely in a modelled space, the properties of direct sound (delay and directions, etc.) and early reflections may change continuously. To make the changes smooth, different kinds of interpolation technique for delays and digital filters are used. The techniques used in the DIVA system are beyond the scope of this article. They are presented in Savioja, Huopaniemi, Lokki and Vaananen (1999).

The DIVA system is capable of both multichannel reproduction with VBAP and two-channel reproduction using HRTFs either with headphones or a pair of loudspeakers.

4. SYSTEM CONFIGURATION OF THE VIRTUAL ORCHESTRA

The complete DIVA system was demonstrated for the first time as an interactive installation in the SIGGRAPH'97 conference (Hiipakka, Hanninen,

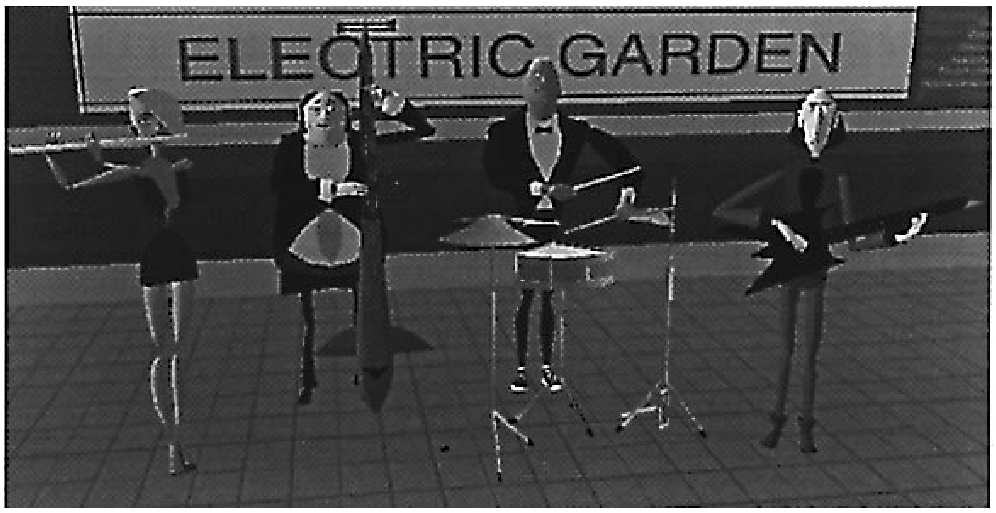

Figure 9. The DIVA virtual orchestra consisting of a flautist, a guitarist, a bass player and a drummer playing in a virtual metro station.

Ilmonen, Napari, Lokki, Savioja, Huopaniemi, Karjalainen, Tolonen, Valimaki, Valimaki and Takala 1997). During the conference over 700 attendees tried conducting the virtual orchestra and many more visitors listened to the auralised performance with headphones. All experimenters were enthusiastic and gave positive feedback. The orchestra consisted of four virtual musicians: a flautist, guitarist, bass player and a drummer. Figure 9 is a screenshot of the orchestra playing in a virtual metro station.

The hardware setup for presentation consisted of three workstations (two Silicon Graphics Octanes and one Silicon Graphics O2) and a magnetic tracking device (Ascension Motion Star) connected in an Ethernet network (figure 10). The software was written using the C++ programming language.

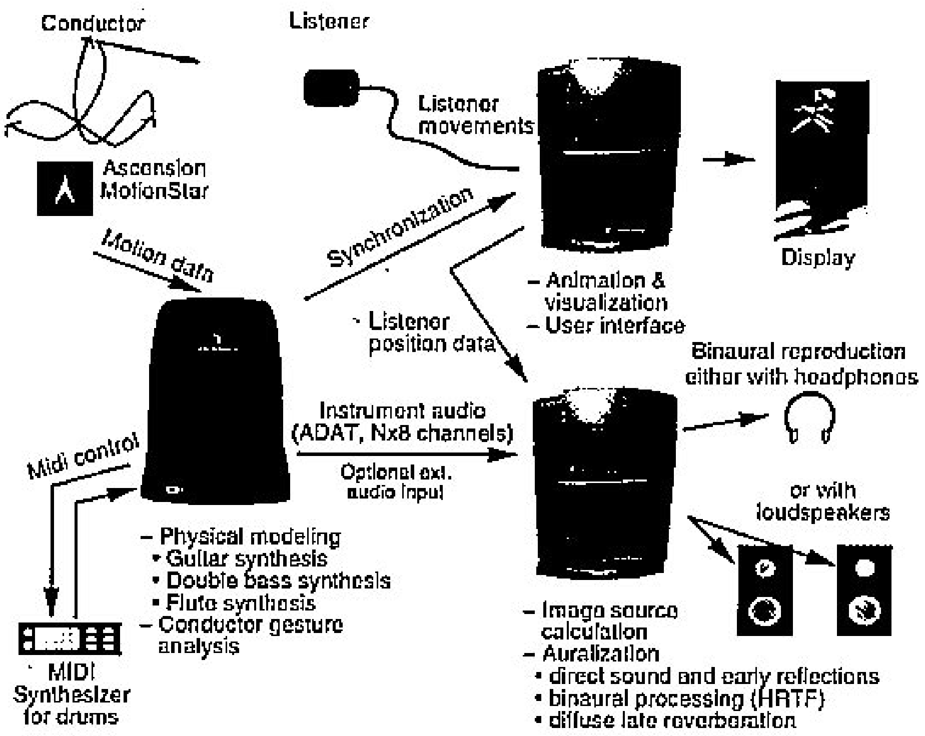
Conductor
Listener
Listener movements
Ascension MotionStar
Motion data
Synchronization
Display
- Animation & visualization
- User interface
Listener position data
Binaural reproduction either with headphones
Instrument audio (ADAT, Nx8 channels)
Optional ext. audio input
Midi control
or with loudspeakers
- Physical modeling
• Guitar synthesis
• Double bass synthesis
• Flute synthesis
- Conductor gesture analysis
MIDI Synthesizer for drums
- Image source calculation
- Auralization
• direct sound and early reflections
• binaural processing (HRTF)
• diffuse late reverberation

Figure 10. Configuration of the DIVA system. The processing is distributed across several workstations to gain enough computational power, but it is still possible to run a single workstation version with a reduced quality rendering.

4.1. Latency

It is impossible to completely eradicate transmission delays. With some dedicated hardware, this *latency* time can be made smaller up to a point. The latency time can be related to the distance that sound signals propagate in air. In 50 ms the sound signal propagates about 17 m. This is approximately the distance between the backmost first violin and the backmost cellist in a symphony orchestra. Without seeing each other they seldom succeed to play in the same tempo. This 50 ms is often considered to be the longest acceptable latency in interactive virtual reality systems (Ellis, Breant, Menges, Jacoby and Adelstein 1997).

Observed latency can be decreased if a synchronisation signal source could foresee the near future in some way. The signal source can then compensate excess latency by sending synchronisation events in advance. A timer with a constant tempo is a trivial example of such a case, but more interesting predictors can also be constructed. In particular, orchestra conductor gestures have been specifically designed so that they carry a strong anticipation about near-future timing. As mentioned in section 3.1, our conductor following module is able to predict the tempo. With this prediction we can compensate for the complete latency of the DIVA system, which is about 200 ms.

5. CONCLUSIONS

A realtime audiovisual rendering system requires sophisticated and efficient methods. As an example we have demonstrated the DIVA system. Different rendering methods were reviewed by pointing out realtime implementation issues. In audiovisual rendering, global light and sound propagation models are desirable, but the realtime requirements enforce the use of simplified models, which assume sound waves to propagate like rays and accurately takes into account only the first few reflections.

The virtual orchestra demonstrates realtime audiovisual rendering in a most fascinating way. Several techniques required in implementing the virtual orchestra were discussed. The conductor follower must predict the tempo, thus being able to compensate for the system latency. The animation sequences are calculated in advance and only the playback is synchronised to the desired tempo. Secondary movements of musicians – blinking, tapping of foot – are added automatically during performance, spontaneously or as a response to environment. The sounds of the orchestra are created with physical models of musical instruments and the interactive three-dimensional soundscape is created with a parametrised sound-rendering process.

While available computational capacity limits the performance of our system, still better algorithms are needed. New methods for character animation, visualisation, modelled instrument control and conductor motion analysis are under development.

REFERENCES

Allen, J. B., and Berkley, D. A. 1979. Image method for efficiently simulating small-room acoustics. *Journal of the Acoustical Society of America* **65**(4): 943–50.

Astheimer, P. 1995. Acoustic simulation for visualization and virtual reality. In R. Veltkamp (ed.) *Eurographics State-of-the-Art Reports*, pp. 1–23. Maastricht, Netherlands.

Begault, D. 1994. *3-D Sound for Virtual Reality and Multimedia*. Cambridge, MA: Academic Press.

Borish, J. 1984. Extension of the image model to arbitrary polyhedra. *Journal of the Acoustical Society of America* **75**(6): 1,827–36.

Cook, P. R. 1991. Tbone: an interactive waveguide brass instruments synthesis workbench for the NeXT machine. *Proc. Int. Computer Music Conf.*, pp. 297–9. Montreal, Canada.

Ellis, S. R., Breant, F., Menges, B., Jacoby, R., and Adelstein, B. D. 1997. Factors influencing operator interaction with virtual objects viewed via head-mounted see-through displays: viewing conditions and rendering latency. *Proc. Virtual Reality Annual Internation Symp.* (VRAIS'97), pp. 138–45.

Foley, J., van Dam, A., Feiner, S., and Hughes, J. 1990. *Computer Graphics*, *Principles and Practice*, 2nd edn. Reading, MA: Addison-Wesley.

Gardner, W. G. 1997. *3-D Audio Using Loudspeakers,* PhD Thesis, MIT Media Lab.

Gardner, W. G. 1998. Reverberation algorithms. In M. Kahrs and K. Brandenburg (eds.) *Applications of Digital Signal Processing to Audio and Acoustics*, Chap. 3, pp. 85–131. Boston, MA: Kluwer Academic Publishers.

Glassner, A. 1995. *Principles of Digital Image Synthesis*, Vol. 2. San Francisco: Morgan Kaufmann.

Granier, E., Kleiner, M., Dalenback, B.-I., and Svensson, P. 1996. Experimental auralization of car audio installations. *Journal of the Audio Engineering Society* **44**(10): 835–49.

Hiipakka, J., Hanninen, R., Ilmonen, T., Napari, H., Lokki, T., Savioja, L., Huopaniemi, H., Karjalainen, M., Tolonen, T., Valimaki, V., Valimaki, S., and Takala, T. 1997. Virtual orchestra performance. *Visual Proceedings of SIGGRAPH'97*, p. 81. ACM SIGGRAPH, Los Angeles.

Huopaniemi, J., Zacharov, N., and Karjalainen, M. 1999. Objective and subjective evaluation of head-related transfer function filter design. *Journal of the Audio Engineering Society* **47**(4): 218–39.

Huopaniemi, J., Savioja, L., and Karjalainen, M. 1997. Modeling of reflections and air absorption in acoustical spaces – a digital filter design approach. *Proc. of IEEE 1997 Workshop on Applications of Signal Processing to Audio and Acoustics*. Mohonk, New Paltz, New York.

Ilmonen, T. 1999. *Tracking Conductor of an Orchestra using Artificial Neural Networks*. Master's Thesis, Telecommunications Software and Multimedia Laboratory, Helsinki University of Technology.

Jaffe, D., and Smith, J. O. 1983. Extensions of the Karplus–Strong plucked string algorithm. *Computer Music Journal* **7**(2): 56–69. Reprinted in C. Roads (ed.) *The Music Machine*, pp. 481–94. Cambridge, MA: MIT Press, 1989.

Jaffe, D., and Smith, J. O. 1995. Performance expression in commuted waveguide synthesis of bowed strings. *Proc. Int. Computer Music Conf.*, pp. 343–6.

Java 3D. 1998. SUN, Inc. JAVA 3D API Specification 1.1. URL: http://www.javasoft.com/products/java-media/3D/

JTC1/SC29/WG11. 1999. ISO/IEC JTC1/SC29/WG11 IS 14496 (MPEG-4).

JTC/SC24. 1997. ISO/IEC JTC/SC24 IS 14772–1. *Information Technology – Computer Graphics and Image Processing – The Virtual Reality Modeling Language (VRML97).* April 1997. URL: http://www.vrml.org/Specifications/VRML97/

Karjalainen, M., Laine, U. K., Laakso, T. I., and Välimäki, V. 1991. Transmission-line modeling and real-time synthesis of string and wind instruments. *Proc. Int. Computer Music Conf.*, pp. 293–6. Montreal, Quebec, Canada.

Karjalainen, M., and Välimäki, V. 1993. Model-based analysis/synthesis of the acoustic guitar. *Proc. Stockholm Music Acoustics Conf. (SMAC'93)*, pp. 443–7. Stockholm, Sweden, published in 1994. Sound examples included on the SMAC'93 CD.

Karjalainen, M., Valimaki, V., and Tolonen, T. 1998. Plucked string models – from Karplus–Strong algorithm to digital waveguides and beyond. *Computer Music Journal* **22**(3): 17–23.

Karplus, K., and Strong, A. 1983. Digital synthesis of plucked string and drum timbres. *Computer Music Journal* **7**(2): 43–55. Reprinted in C. Roads (ed.) *The Music Machine*. Cambridge, MA: MIT Press, 1989.

Kleiner, M., Dalenback, B.-I., and Svensson, P. 1993. Auralization – an overview. *Journal of the Audio Engineering Society* **41**(11): 861–75.

Krokstad, A., Strom, S., and Sorsdal, S. 1968. Calculating the acoustical room response by the use of a ray tracing technique. *Journal of Sound and Vibration* **8**(1): 118–25.

Kuttruff, H. 1995. Sound field prediction in rooms. *Proc. 15th Int. Congr. Acoust.,* Vol. 2, pp. 545–52. Trondheim, Norway.

Lytle, W. 1990. More bells and whistles [video]. SIGGRAPH'90 Electronic Theater (1990). Technically described in J. Hahn (ed.) *Sound Synchronization and Synthesis for Computer Animation and Virtual Reality*, Course Note 12 (SIGGRAPH'94 Conf., Orlando, FL, 1994). Also described in *IEEE Computer* **24**(7): 4 and cover (July 1991).

Masataka, G., Hidaka, I., Matsumoto, H., Yosuke, K., and Yoichi, M. 1996. A jazz session system for interplay among all players. In *Proc. of the Int. Computer Music Conf.*, pp. 346–9, Hong Kong.

McElheran, B. 1989. *Conducting Technique for Beginners and Professionals*. Oxford/New York: Oxford University Press.

Möller, H. 1992. Fundamentals of binaural technology. *Applied Acoustics* **36**(3–4): 171–218.

Moller, H., Hammershoi, M. S. D., and Jensen, C. 1995. Head-related transfer functions of human subjects. *Journal of the Audio Engineering Society* **43**(5): 300–21.

Perlin, K., and Goldberg, A. 1996. IMPROV: a system for scripting interactive actors in virtual worlds. *Proc. of SIGGRAPH'96*, pp. 205–16. ACM SIGGRAPH, New Orleans, LA.

Pulkki, V. 1997. Virtual sound source positioning using vector base amplitude panning. *Journal of the Audio Engineering Society* **45**(6): 456–66.

Rojas, R. 1996. *Neural Networks: A Systematic Introduction*, p. 502. Berlin: Springer Verlag.

Savioja, L., Huopaniemi, J., Lokki, T., and Väänänen, R. 1999. Creating interactive virtual acoustic environments. *Journal of the Audio Engineering Society* **47**(9): 675–705.

Savioja, L., Rinne, T., and Takala, T. 1994. Simulation of room acoustics with a 3-D finite difference mesh. *Proc. Int. Computer Music Conf.*, pp. 463–6. Arhus, Denmark.

Scavone, G. P. 1996. Modeling and control of performance expression in digital waveguide models of woodwind instruments. *Proc. Int. Computer Music Conf.*, pp. 224–7. Hong Kong.

Smith, J. O. 1998. Principles of digital waveguide models of musical instruments. In M. Kahrs and K. Brandenburg (eds.) *Applications of Digital Signal Processing to Audio and Acoustics*, Chap. 10, pp. 417–66. Boston, MA: Kluwer Academic Publishers.

Smith, J. O. 1985. A new approach to digital reverberation using closed waveguide networks. *Proc. Int. Computer Music Conf.*, pp. 47–53. Vancouver, BC, Canada.

Smith, J. O. 1986. Efficient simulation of the reed-bore and bow-string mechanisms. *Proc. Int. Computer Music Conf.*, pp. 275–80. The Hague, The Netherlands.

Smith, J. O. 1987. Music applications of digital waveguides. *Technical Report STAN-M-39*. Department of Music, CCRMA, Stanford University, Stanford, CA.

Smith, J. O. 1993. Efficient synthesis of stringed musical instruments. *Proc. Int. Computer Music Conf.*, pp. 64–71. Tokyo, Japan.

Takala, T., and Hahn, J. 1992. Sound rendering. *Computer Graphics* **26**(2): 211–20. Proc. SIGGRAPH'92.

Takala, T., Hanninen, R., Valimaki, V., Savioja, L., Huopaniemi, J., Huotilainen, T., and Karjalainen, M. 1996. An integrated system for virtual audio reality. *100th Audio Engineering Society (AES) Convention Preprint no. 4229.* Copenhagen, Denmark.

Tolonen, T. 1998. *Model-Based Analysis and Resynthesis of Acoustic Guitar Tones*. Master's Thesis, Helsinki University of Technology, Laboratory of Acoustics and Audio Signal Processing, Espoo, Finland.

Vaananen, R., Valimaki, V., and Huopaniemi, J. 1997. Efficient and parametric reverberator for room acoustics modeling. *Proc. Int. Computer Music Conf.*, pp. 200–3. Thessaloniki, Greece.

Valimaki, V., Huopaniemi, J., Karjalainen, M., and Janosy, Z. 1996. Physical modeling of plucked string instruments with application to realtime sound synthesis. *Journal of Audio Engineering Society* **44**(5): 331–53.

Välimäki, V., Karjalainen, M., Janosy, Z., and Laine, U. K. 1992. A real-time DSP implementation of a flute model. *Proc. 1992 IEEE Int. Conf. on Acoustics, Speech, and Signal Processing,* Vol. 2, pp. 249–52. San Francisco, CA.

Valimaki, V., Karjalainen, M., and Laakso, T. I. 1993. Modeling of woodwind bores with finger holes. *Proc. Int. Computer Music Conf.*, pp. 32–9. Tokyo, Japan.

Valimaki, V., and Takala, T. 1996. Virtual musical instruments – natural sound using physical models. *Organised Sound* **1**(2): 75–86.

Valimaki, V., and Tolonen, T. 1998. Development and calibration of a guitar synthesizer. *Journal of the Audio Engineering Society* **46**(9): 766–78.