Chapter 1

Introduction

"All the heavens, seem to twinkle
With a crystalline delight;
Keeping time, time, time,
In a sort of Runic rhyme,
To the tintinnabulation that so musically wells
From the bells, bells, bells, bells..."

—Edgar Allan Poe

When QuickTime 1.0 debuted at Apple's Worldwide Developer Conference in December 1991, it was seen as a major technological breakthrough – a postage stamp sized video played back in real-time! Technology has advanced significantly since then; computers today are capable of not only real-time playback of high definition audio and video material, but can also support complex chains of processing on this material.

Modern multimedia interfaces continue to use 1960s tape recorder metaphors.

Unfortunately, techniques for interacting with time-based media such as audio and video have not changed significantly since the 1960s – the same "play", "fast-forward", and "rewind" buttons on a 1966 tape recorder remain the primary playback controls on the latest versions of QuickTime Player and Windows Media Player (see Figure 1.1). Most players do not even offer an option to play audio and video at speeds other than at its nominal speed. Software applications that do support this functionality usually offer a simple rate slider as an advanced setting hidden away in the user interface (see Figure 1.2).

Explicit control over time is often taken for granted when interacting with many non-computer-based media, and the limited control over time in digital audio and video directly contradicts this assumption.

Figure 1.1: A comparison of a 1966 RCA YHS-18J tape recorder and current versions of Windows Media Player and QuickTime Player. The fundamental interaction metaphors have not changed, and modern media players continue to use the decades-old "play", "pause", "fast-forward" and "rewind" controls as the primary means for controlling media playback.

Figure 1.2: Recent versions of Windows Media Player and QuickTime Player allow media playback at rates other than one. However, the range of allowed rates is limited, and this control is offered as an advanced setting in the interface.

1.1 Malleability of Time

Imagine a world where reading a book is limited to a constant rate of 100 words per minute (wpm). While pages can be flipped, chapters can be skipped, and reading may start and stop at arbitrary locations in the text, actual consumption of the written content is limited to this one rate, regardless of whether the text is read for comprehension, or simply scanned for key phrases. It would be hard to imagine such a scenario – it is natural for us to freely control our reading rate, which we may slow down to 50 wpm for more difficult or important passages, or speed up to several hundred wpm when skimming. However, this is the situation that is present today when, for example, listening to a podcast or other recorded media program.

The temporal nature of audio and video also makes control over the time

axis much more significant than for traditional "spatial" media such as text. As Hürst and Stiegeler [2002] observe, only the smallest unit of a continuous audio and video stream, such as a single video frame or audio sample, can be conveyed to the user at any moment in time. In contrast, for spatial media such as a text document, many lines of text can be displayed to the user at the same time. Moreover, temporal media must often be perceived over time. While a single time instant of video can be interpreted as a still image, a single time instant of audio has no meaningful interpretation.

Control over time is key in musical expressiveness.

Explicit control over time is also a key component of musical expression [Dobrian and Koppelman, 2006]. Many computer music applications today continue to use synthesized music, where the audio is rendered using MIDI (Musical Instrument Digital Interface) or wavetable synthesis. Synthesized music has the advantage of using an event-based time model, which gives system designers complete control over when and how certain events, such as notes or beats, are triggered – that is, time is malleable by controlling when these events occur. Digital audio and video streams, however, can offer a higher degree of fidelity and realism compared to synthesized audio, and despite continuing advances in synthesizing technology and physical instrument modelling, it is still not possible to reproduce, for example, the unique character of the Vienna Philharmonic playing in their Golden Hall of Vienna's *Musikverein*. Digital audio and video streams, however, have a different time model – as the word "stream" implies, these media types consider time as a continuous flow, and simply shifting audio samples around like one would with MIDI note events creates unpleasant pop and click artifacts.

Previous work supports the importance of temporal interaction. One example is our previous work in interactive conducting systems, which allow users to control the speed, volume, and instrument emphasis of a digital audio and video recording. [Borchers et al., 2004] found that users most easily identified the interaction with music tempo: in a particular evaluation session where users were silently observed and then interviewed, 93% of the users recognized that they could control tempo by moving the baton faster or slower, 77% realized that they could control volume by making larger or smaller gestures with the baton, and 37% realized that they could control the instrument emphasis by conducting towards different sections of the orchestra shown on the large display.

1.2 Scope and Context of this Thesis

Interactive media systems are becoming increasingly complex.

Computers and processing capacity continue to advance at rates that exceed Moore's original prediction [1965]. Certain types of processing, such as using the phase vocoder for changing the speed of an audio recording while maintaining its original pitch [Flanagan and Golden, 1966] that were once a fantasy can now run in real-time [Karrer et al., 2006]. Similarly, continuing research in computer vision enables new interaction methods [Camurri et al., 2003]. These research areas form the basic building blocks

that, when combined, form an interactive media system. A discussion of methods to design such systems, and of the challenges of integrating these various components into a single system, is thus needed. And as interactive media systems become increasingly complex, this discussion likewise becomes increasingly important, since it is no longer possible for a single person to understand at depth the intricacies of all aspects of such systems.

Multimedia frameworks have, for a long time, been a part of software engineering research. As François [2004] writes, many existing multimedia frameworks focus on media "storage, retrieval, transmission and presentation". Some of these frameworks include the *Berkeley Continuous Media Toolkit* [Mayer-Patel and Rowe, 1997], and *VuSystem* [Lindblad and Tennenhouse, 1996]; the two main aspects that make such frameworks unsuitable for building interactive media systems is the missing ability to insert custom processing, and lack of support for interactivity. Apple's QuickTime media framework [Apple, 2006b] also falls into this category – while it is relatively simple to open movies from a variety of sources and present them to a display, it is extremely difficult, if not impossible, to perform functions that reach beyond the ecosystem provided by the framework. For example, it is extremely difficult (if not impossible) to use an external, user-controlled clock to control a QuickTime movie's playback position and rate. François' own work on a software architecture for immersipresence (SAI) is one of the few that considers interactivity; however, even SAI treats time as discrete "pulses" of data and input events, rather than continuous streams, which we argue is both more general and more suitable for time-based media.

A specialized category of multimedia frameworks are those designed for computer music applications. Many of these frameworks focus on audio synthesis, including the Synthesis ToolKit (STK) [Cook and Scavone, 1999], and the C++ Library for Audio and Music (CLAM) [Amatriain, 2004]. However, since performance is an integral part of music (and thus, computer music), many of these frameworks include capabilities for real-time interaction – some examples include Max/MSP [Puckette, 2002] and SuperCollider [McCartney, 1996]. Computer music frameworks however, are often tied to the music model of time, and are thus difficult to generalize beyond music to other multimedia domains.

One aspect of system design that remains to be addressed is the *temporal* aspect – the design of systems that respond to continuous input from the user, and, in response, continuously adjust the timebase of multimedia streams. These systems also pose a wide range of problems, from issues of audio time-stretching, to synchronization of multiple timebases, to interpretation of user input in relation to the media.

The work performed in this thesis originated from work on interactive conducting systems. The family of interactive conducting systems created by members of the Media Computing Group at RWTH Aachen University is known as *Personal Orchestra*, and primary responsibility of the *Personal Orchestra* project fell to the author beginning with the development of the

second system, *Personal Orchestra 2* (also known *You're the Conductor* [2004]). *You're the Conductor*'s key accomplishment was the ability to offer arbitrary control over the speed of a synchronized digital audio and video recording – *The Virtual Conductor* (*Personal Orchestra 1*) offered only a limited range of playback speeds due to the way time-stretching was performed. Indeed, *Personal Orchestra* remains, today, one of the few systems that guarantees *synchronous* playback of digital audio and video media at arbitrary rates – similar systems, such as Kolesnik's conducting system [2004] or the *Virtual Symphony Orchestra* [Brügge, 2005] employ digital audio and video media, but no attempt is made to ensure synchronicity between the audio and video streams is maintained.

This thesis was inspired by our ongoing work in interactive conducting systems.

The challenges we faced in designing a system that supports synchronous audio and video playback at arbitrary speeds motivated us to develop the Semantic Time Framework, a software library for these types of applications [Lee et al., 2006b]. While the initial focus was on interactive conducting systems, the framework evolved to support an increasing number of other multimedia applications that allow users to freely manipulate the temporal dimension of time-based media. We developed additional applications that, for example, allow users to skim and search through audio [Lee and Borchers, 2006b], or perform common audio editing tasks [Lee et al., 2006c]. These applications are not only interesting by themselves, but have also created opportunities for additional research in audio navigation techniques [Lee, 2007b].

The additional experience obtained from developing these interactive media systems was then incorporated into a second iteration of the Semantic Time Framework. This second iteration not only offers support for a more generic class of multimedia systems, but it also includes a discussion of *semantic time*, a theory we developed for representing time and temporal transformations [Lee and Borchers, 2006a]. The combined theory and software implementation allows us to simplify the design process and facilitate code reuse across multiple systems, and versions of our previous systems were re-implemented using this second version of the Semantic Time Framework to demonstrate this.

1.3 Contributions

The primary goal of this thesis is to facilitate the design and construction of interactive media systems where manipulating the timebase of the media is a key component of the interaction. This temporal aspect is the main focus, and the main contributions include:

- The development of a time-design space for interactive media systems that is an extension/refinement of a well-established classification space for general human-computer interaction

- A discussion of the challenges of designing interactive media systems – in particular, the issues a system designer may encounter when interpreting and mapping continuous temporal input from the user, or applying timeline changes to digital time-based media such as audio and video
- A method for representing time and temporal transformations in interactive media systems
- A software framework that incorporates the above aspects, and offers a *low threshold* for designers wishing to build new applications while still supporting a *high ceiling* of potential functionality

1.4 Structure

The remainder of this thesis will be divided as follows:

Chapter 2 introduces our time-design space for interactive media systems; this time-design space consists of three domains: user, medium, and technology. The scope and areas of research that fall within each of these domains is described.

Chapter 3 talks about the problems when mapping time across these domains. Interpretation of both latency in response to user input, and processing latency in audio time-stretching algorithms will be discussed, and synchronization methods to address these latencies will be described.

Chapter 4 introduces the concept of *semantic time*, a common means of representing time and temporal transformations at the system level.

Chapter 5 describes the *Semantic Time Framework*, a software framework we created for constructing interactive media systems. The *Semantic Time Framework* realizes the ideas presented in the previous chapters in software, and we will also describe how the Semantic Time Framework evolved from a framework for interactive conducting systems to one that supports the more general class of interactive media systems. Two sample programs will also be described in detail to illustrate how the Semantic Time Framework allows designers to easily solve common problems (*low threshold*) when working with time-based media.

Chapter 6 presents three more systems that use the Semantic Time Framework as their foundation. These systems are considerably more complex than the sample programs described in Chapter 5, and demonstrate how the Semantic Time Framework can be used to develop systems with a *high ceiling* of functionality.

Chapter 7 provides an outlook to future challenges, and **Chapter 8** summarizes the work presented.

Finally, supporting material is provided in the appendices of this thesis: **Appendix A** introduces the principles of sampling and quantization, **Appendix B** is a very brief overview of Fourier Theory, and **Appendix C** provides the full source code listings of the programs discussed in Chapter 5.