# Accelerated Playback of Meeting Recordings


***by***

***Gerrit Hendrikus van Doorn***


***July 2007©***

**A thesis submitted in partial fulfillment**
**of the requirements for the degree of**
**Master of Science**




**Graduation committee**
Prof. dr. ir. Anton Nijholt[1]
Dr. Roeland Ordelman[1]
Mike Flynn[2]
Dr. Pierre Wellner[2]

University of Twente
*Enschede - The Netherlands*

---


[1] Chair Human Media Interaction, Dept. of EE-Math-CS, University of Twente, Enschede, Overijssel, The Netherlands
[2] IDIAP Research Institute, Rue du Simplon 4, 1920 Martigny, Switzerland

# Acknowledgements

I would like to thank the following people, without whom this thesis would not have been possible:

My advisors Roeland Ordelman and Anton Nijholt for their advice, encouragement and patience.

My advisors Mike Flynn and Pierre Wellner for their advice, encouragement, and work on the Speed/Accuracy trade-off model.

Simon Tucker from the University of Sheffield for letting me use his Mach1 Matlab implementation.

Ferran Galán and Eileen Lew for participating in one of the user tests and providing me with feedback.

Natalie Solomonoff for her encouragement and enormous belief in me.

# Table of Contents

# List of figures

# List of tables

# Abstract

Current technology allows us to record and store multimodal recordings of meetings. These recordings can function like an archive, as a replacement for minutes. Although recording these meetings is straightforward, finding specific information in them is not. In order to make it easier to find information in meeting recordings, meeting browsers are developed. A meeting browser may consist of many different components, each making it possible to browse or search the recorded media in a different manner. Meeting recordings consist of video and audio recordings. Other modalities, such as speech recognition transcripts and speech segmentations can be extracted from these sources and used to make browsing and searching the meeting more efficient in the sense that information can be found easier and quicker.
This report describes two novel approaches for presenting recorded meeting audio in an interactive meeting browser: time-compressed audio and simultaneous playback using binaural audio. Both methods reduce the playback time of the audio without discarding any of the spoken content from the audio, resulting in the same information intake in less time. User tests were carried out to see which approach yields a better performance.

# 1 Introduction

The goal of this research is to explore more efficient ways of browsing archived meeting recordings. This research focuses specifically on methods to present audio efficiently.

## 1.1 Motivation

Meetings are an important part of our working lives. No matter what company branch you work in or weekend sports club you are part of, decisions have to be made and information has to be exchanged at any level of the hierarchy. The acts or processes of coming together to communicate, discuss issues, set priorities, and make decisions are called meetings. In order for participants of a meeting to look back on what happened during a meeting, minutes are made of the meeting. These minutes are usually an abstract of the meeting and contain only the main ideas and decisions.
The increase of low-cost disk space and improved multimedia encoding techniques, have made multimodal recordings of meetings possible. Having complete recordings of meetings available makes it possible to recall exactly what happened during a meeting.
People that could not attend a meeting or participants who would like to recall what happened during a meeting can playback the meeting as if they were there.
The duration of meetings can vary from very short to hours and sometimes multiple days. Although a meeting might take up a lot of time, the information contained in a meeting is often scarce. People might want to recall a certain topic, the decisions made during a meeting, or look at the process that led to a certain decision. Having audio/video recordings of your meetings thus can have great benefits. However, *although it is easier to speak than to write, it is slower to listen than to read* (Arons, 1997). Therefore, having to playback one or more meeting recordings to find some piece of information is a slow process and in this respect inefficient. A more flexible method of playing back audio is through browsing. We define playback of audio as *"a method of reproducing sound recordings"* and browsing as *"to scan through in order to find items of interest, especially without knowledge of what to look for beforehand"*. Browsing, in the case of a media recording, adds an interactive element to playback as it allows skipping content that is potentially irrelevant. Although playing back a meeting is straightforward, browsing a meeting is much more laborious. Creating new technologies to enhance browsing of recorded meetings has therefore become an active area of research.

## 1.2 Problem description

Meeting recordings may consist of audio alone or audiovisual data. If one wants to be informed on the discussions that took place during a meeting, the audio recording is a very important information carrier as it contains information like speech, prosody and back channeling. Speech is expressive and contains various cues that express the current emotional state of a speaker. Such information is difficult to capture in a textual or graphical form. The waveform or spectrogram of the speech might be used to make the speech graphical, but this still does not show what is said and how it is being said. The only way to make full use of the context and audio cues is to play the speech.

Locating a piece of information in a meeting recording can be a difficult task. Playing back a meeting completely to find a specific piece of information is not very efficient. At best this information is located at the beginning, at worst at the end. When the meeting recording consists of just audio and video, the timeline of a general-purpose media-player can be used to browse through the meeting and skip irrelevant information, but this too is not very efficient and laborious. Adding additional information, such as slides, can make it easier to locate specific information in the audiovisual recording. Slides are structured and contain headings and other content. Locating specific audiovisual content can be done more easily by linking presentation slides to their corresponding audiovisual content, than by browsing just the audiovisual content.
The use of presentation slides makes answering certain questions easier, for example:

- "What material did Christine prefer the remote control to be made of?"
- "What was the result of the discussion about which materials to use?"
- "Why was wood chosen as the material for the remote control?

It is not always the case that presentation slides give an indication of where to look for the information, as is the case with questions like:

- "What was Christine's role in this meeting?"
- "What did Christine say about the previous meeting?"

The first question is general while the second question is very specific. Therefore it is not clear if they belong to a certain topic or part of the meeting. In order to answer these questions you often need to listen to or skim through a large part of the media.

The problem with meeting recordings and audiovisual (A/V) recordings in general, is that they have a sequential/transient nature (Ranjan, 2005); there is no natural way for humans to skim speech information as the ear cannot skim in the temporal domain the way the eyes can browse in the spatial domain (Arons, 1997).

Unlike text documents, where you can skim through the content of the document, you need to play back the complete recording in order to know the structure of the content. You can skim through the recording but still you need to listen for at least several seconds to know what it is about at a specific time in the recording. If we can shorten the playback time of a recording, browsing the recording would become more efficient since the same amount of information would be processed in a shorter timeframe. Of course, while doing this we do not wish to remove content from the recording that might be of interest to a user.
The combination of not removing content from media recordings and their sequential nature of media recordings leaves open only two options for shortening playback length:

- Speedup audio playback
- Present multiple streams of audio simultaneously

The simultaneous presentation of multiple audio streams is based on the idea that people are able to switch attention between audio streams, making it

possible to browse through more audio in the same amount of time. These techniques will be discussed in more detail in Chapter 2.

## 1.3 Hypothesis

The hypothesis we set out to test is:

*"Accelerated playback will improve meeting comprehension in a limited time situation"*

When the length of a meeting is shortened users will have less meeting to skim through.
We will be investigating techniques to shorten the playback time of meeting recordings with the use of sped up speech and simultaneous playback of speech in interactive meeting browsers.

These meeting browsers will be designed to make browsing meetings more efficient, in comparison to using a general-purpose meeting browser, by decreasing the playback length of the meeting.
User tests will be conducted on 3 browsers to see what the effects of the implemented techniques are on browsing meetings, and to compare the performance of the different browsers.

We expect that decreasing the playback time of a meeting increases the efficiency of browsing. We will prove this expectation false or true by comparing the number of questions users can answer in a limited time situation using sped up and overlapped speech and speech played back at a normal rate.

## 1.4 Overview of document

The thesis is organized as follows:

In Chapter 2 we review previous work from which our research draws. Time-compression and binaural audio algorithms and implementations are discussed to give an understanding of these techniques and the way they are used in the implementation of our meeting browsers. Also, some related systems are discussed.

Chapter 3 discusses the setup of the user test that was used to compare the use of time-compression and binaural audio techniques in a meeting browser, using an objective measure.

Chapter 4 describes the design of the implemented meeting browsers, based on literature and observations from informal tests.

Chapter 5 gives a summary of the results collected during the user test described in chapter three.

Finally, conclusions will be drawn in Chapter 6 based on the results. Recommendations for future research are given based on these conclusions.

# 2 Background

The following sections will give background on techniques to speed-up audio or enhance the ability of people to listen to multiple sound sources at once.

## 2.1 Time-compressed speech

Time compression technologies allow the playback speed of a recording to be increased and therefore decrease the time needed to listen to a complete recording. Time compressed speech is also referred to as accelerated, compressed, time-scale modified, sped-up, rate-converted, or time altered speech (Arons, 1994).

### 2.1.1 Perception of time-compressed speech

Time compressed speech can be used to increase the amount of information a person can perceive per time-unit and increase storage capacity of speech recordings.
Research by Sticht has shown that listening twice to teaching materials that are sped-up twice, is more effective than listening to the same recording once at normal speed (Arons, 1994).

According to Arons (1997), Beasley and Maki informally reported that, following a 30-minute exposure to time-compressed speech, listeners became uncomfortable if they were forced to return to the normal rate of presentation. They also observed that subject's listening-rate preference shifted to faster rates after exposure to compressed speech. According to Janse (2003), this adaptation is not permanent. The type of adjustment is not like learning a 'trick' that is stored in long-term memory.

### 2.1.2 General techniques

There are many ways of compressing speech. Some general techniques are:

**Changing the playback rate**
When playing back speech, the playback rate can be changed to slow it down or speed it up. This is similar to playing a tape faster or slower or playing a 45rpm record on a 78rpm record player. When playing back a speech recording at 44 KHz that was recorded with a sampling rate of 22 KHz, the playback time will be halved, but an unwanted pitch-shifting effect will occur. The frequency shift is proportional to the change in playback speed; when the speech is sped up twice, the frequency will be doubled and the speech will sound like chipmunks. Preserving pitch and timbre is important to maximize intelligibility and quality of the listening experience (Omoigui, He, Gupta, Grudin, & Sanocki, 1999; Verhelst, 2000).

**Speaking faster**
When people speak fast, they unintentionally change attributes of their speech such as pause durations, consonant and vowel durations (Covell, Withgott, & Slaney, 1998a). Although there are exceptions, people usually are only capable of compressing their own speech up to 70% (Arons, 1994, p. 59).

**Speech synthesis**
Text-to-speech systems make use of speech synthesis to generate artificial human speech. These speech synthesizers are capable of reducing phoneme and silence durations in order to produce speech with different word rates. This technique is particularly helpful as a technical aid for visually impaired people but is irrelevant for recorded speech (Arons, 1992b).

**Pause removal**
Since silences do not contain any lexical information, they are redundant. Most spontaneous speech contains pauses and hesitations. By removing these silences the playback time of the recording can be reduced. Removing pauses will be experienced as disruptive when they occur within clauses, making it important to distinguish between different pauses. The duration of a hesitation is often smaller than the duration of a grammatical pause. Recent research has shown that a distinction between these two can not be made solely based on duration (O'Shaughnessy, 1992). Because pauses are not uniformly distributed over a recording and differ in length, removal or shortening of pauses can be seen as non-linear time-compression, making it difficult to obtain a pre-determined compression ratio.

### 2.1.3 Sampling
Much of the research on time-compressed speech was influenced by Miller and Licklider (1950), who demonstrated the temporal redundancy of speech. Their research shows that large parts of a speech recording can be removed without affecting intelligibility, by discarding interrupted speech segments when interruptions were made at frequent intervals. This technique is referred to as Fairbanks sampling. An implementation of this technique is shown in Figure 1.

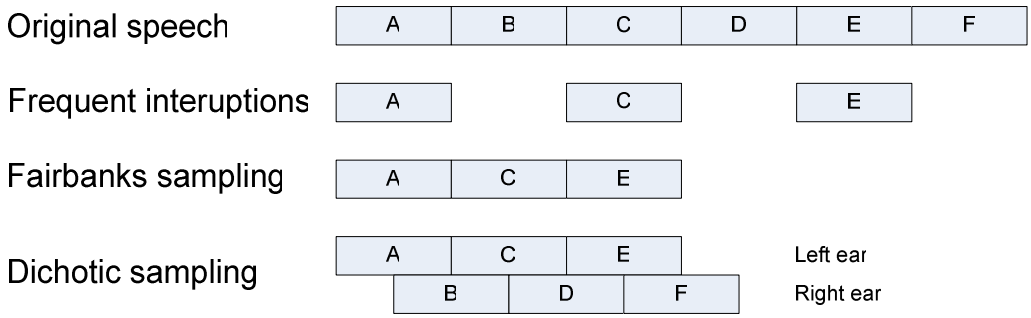


**Figure 1. Sampling techniques**

The Fairbanks sampling from Figure 1 compresses the original speech signal by 50%. By adjusting the frequency and interval size, the amount of compression can be adjusted. However, the sampling interval should be at least as long as one pitch period and not longer than the length of a phoneme (Arons, 1994). Although this technique is computationally simple and very easy to implement, it introduces discontinuities at the interval boundaries and produces clicks and other signal distortions.

By taking advantage of the auditory system's ability to integrate information from both ears, both intelligibility and comprehension can be increased. Dichotic sampling is a technique that accomplishes this by playing the standard sampled

signal on one ear, while playing the discarded intervals to the other ear. Information is lost when the compression ratio is greater than 50%.

### 2.1.4 SOLA

The Synchronized OverLap Add (SOLA) method is a variant of the sampling method and was first introduced by Roucos and Wilgus (1985). SOLA does not discard intervals

SOLA uses correlation techniques to perform time-scaling speech. It is a very popular time-scaling technique as it is computationally simple yet generates good quality sped up speech, and therefore can be used for real-time applications.
The speech signal is broken up into fixed sized segments that are shifted according to the time-scaling factor $\alpha$. Then, for each segment, the discrete time lag factor $\Delta t_n$ is determined using cross correlation of the overlapping parts to find the point of maximum similarity. The overlapping parts are added together using a fade-in/fade-out-function, producing less artifacts than traditional sampling-techniques (Zölzer et al., 2002). SOLA operates in the time-domain, without making use of Fourier Transforms, making it very computational efficient.

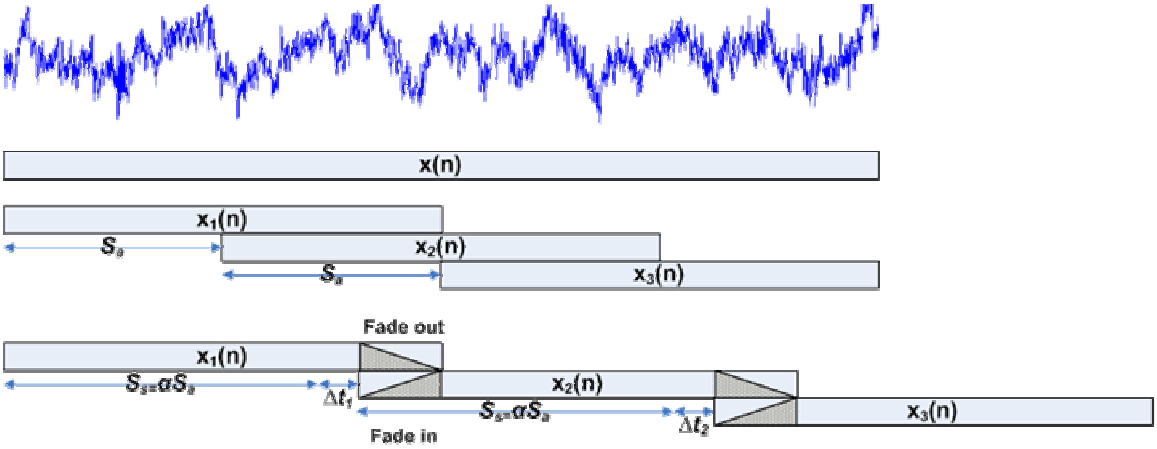


**Figure 2. SOLA example of time stretching.**

Figure 2 gives an example of time-stretching using SOLA. Time-compression is achieved in a similar matter, but with a different $\alpha$ value.

Algorithm:

1. Segment the input signal into blocks of N with a time shift Sa
2. Reposition the blocks with the time scale factor to $S_s = \alpha S_a$
3. Compute the cross correlation of the overlapping parts
4. Extract the discrete time lag $\Delta t_n$ where the cross correlation has its maximum value
5. Using this $\Delta t_n$, fade out the first segment $x_1(n)$ and fade in the second segment $x_2(n)$
6. Add $x_1(n)$ and $x_2(n)$ to form the output signal

High quality time-compressed speech is achieved using the SOLA technique, and speech that is processed with SOLA does not result in speech that is pitch-shifted.

### 2.1.5 PSOLA

A variation on SOLA is Pitch-Synchronous OverLap-Add (PSOLA). PSOLA was proposed by Moulines and Charpentier (Zölzer et al., 2002). PSOLA makes use of the pitch to correctly synchronize time segments, avoiding pitch discontinuities. By duplicating or eliminating segments of the speech signal at a pitch-synchronous rate, voiced speech can be time-compressed or time-stretched. As with SOLA, duration manipulations obtained with PSOLA are applied directly to the signal itself. First a pitch detector places markers at each pitch period. At the pitch-mark locations, the signal is decomposed into separate but overlapping windows, with the use of Hanning windows that are usually twice the fundamental frequency duration. When the speech signal is time-compressed, pitch periods are deleted from the signal, as much as is necessary to realize the wanted duration. The resulting signal contains less pitch periods but still has many of the brief acoustic events important to speech perception. However, when compressing the speech signal to 50% of its original duration, neighboring pitch-periods will be removed.

### 2.1.6 WSOLA

Another variant of the SOLA algorithm is the Waveform-similarity-based Synchronous Overlap and Add algorithm (WSOLA) (Verhelst, 2000; Verhelst & Roelands, 1993). WSOLA uses an asynchronous segmentation technique with a fixed length window in combination with regularly spaced synthesis. It is computationally and algorithmically more efficient than SOLA and PSOLA.

### 2.1.7 Mach1

Covell, Withgott and Slaney (1998a; 1998b) propose a new approach to non-uniform time compression. Mach1 tries to mimic the natural timing of speech that is used by humans when they talk fast. When speaking fast, human speakers compress silences and pauses the most, stressed vowels the least and unstressed vowels intermediate. The compression of consonants is based on their neighboring vowels. On average, consonants are more compressed than vowels. Covell et al. (1998a; Covell et al., 1998b) also avoided over-compressing already rapid sections of speech. To achieve this, Mach1 tries to estimate something they call the *Audio Tension*, which is the degree to which the local speech segments resist changes in rate. High-tension segments are compressed less than low-tension segments. The Audio Tension is constructed from two estimated continuous-valued measurements of the speech signal: *local emphasis* and *relative speaking rate*. The local emphasis measure is used to distinguish among silence, unstressed syllables, and stressed syllables. Although emphasis in speech correlates with relative loudness, pitch variations and duration, only the relative loudness is used to estimate emphasis. The relative speaking rate is estimated to prevent over-compressing already rapid speech segments. The relative acoustic variability is used to estimate the phoneme-transition rate, which is used as a measure for the speaking rate. The local target rates are calculated from the audio tension together with the desired global compression rate. However, there is no guarantee that the desired global compression rate will be achieved. The

local target rates are used as input for a standard time-compression algorithm like SOLA.
User tests were conducted to compare user's comprehensibility and preference between Mach1 and SOLA. The tests showed significant improvements in comprehension on the behalf of Mach1, especially at high compression rates. Users preferred Mach1 95% of the time on average over SOLA, increasing with compression rate. Comprehension tests were conducted on short dialogues, long dialogues and monologues. Mach1 performed better on the short dialogues and monologues but showed no statistical significant comprehension improvement on the long dialogues. This could be due to confusing interactions between Mach1 and turn-taking techniques in the conversations.
Non-linear time-compressed audio (e.g. Mach1) is more comprehensible than linear time-compressed audio (like SOLA) at higher speeds. There is no significant difference in comprehension at lower speeds (He & Gupta, 2001).

Although Mach1 is perhaps the most sophisticated and most promising algorithm for time-compressed speech playback, there are some drawbacks:

- The Mach1 algorithm is an open loop algorithm; the eventual global compression rate can be different from the desired global compression rate

- Because of the high computational load, it can not be used in real-time

### 2.1.8 SpeechSkimmer

Arons (1994; 1997) describes the SpeechSkimmer system for interactively skimming recorded speech. SpeechSkimmer uses speech-processing techniques in order to make the user hear the recorded speech at a higher speed and at different levels of detail. With the use of a physical input device, based on a touch pad, the user can control the speed and level of detail in real-time. Because of the 2 dimensional nature of the touch pad grid, the user can control both the speed and the level of detail at the same time. Arons mentions a variety of techniques that can be used to speed up the speech: pause removal, pause shortening, Fairbanks sampling, dichotic sampling, SOLA, combined time-compression techniques (e.g. SOLA with pause removal) and backward sampling (for rewinding). It also mentions some skimming techniques that can be used: backward skimming, isochronous skimming (equal time intervals), synchronous skimming based on pauses, pitch, energy, speaker identification, word spotting, user selected segments and combinations of these techniques. In addition a hierarchical representation ("Fish ear") of audio information was made. Four distinct skimming levels were implemented. The first level is the original speech played back at normal speed. Level 2 removes pauses smaller than 500ms and shortens pauses that are larger than 500ms to 500ms. Level 3 includes skimming based on juncture pauses. When for example, a pause of 750ms is detected the subsequent 5 seconds of speech are played with shortened pauses. Level 4 is similar to level 3 in that it presents segments of speech that are highlights of the recording. Level 4 uses pitch to indicate these highlights and is based on the fact that there tends to be an increase in pitch range when a speaker introduces a new topic. Because it is very hard to know when a new segment is started in the highest skimming levels, a 600ms pause is inserted between segments. Alternative skimming levels based on speaker identification were also implemented. Level 2 played only speech from one speaker while level 3 played

speech from the other speaker. The speed of the segments at skimming levels 2, 3 and 4 can be adjusted continuously. A usability test where users have to think aloud was conducted on the system. The test looks at the interface and skimming of speech. No tests were done to see which time compression technique performs the best (e.g. has good comprehension at high compression rates). Some users thought there was a major improvement when using headphones. One user was amazed at how much better the comprehension was when using dichotic time-compressed speech instead of hearing monaural speech presented over a speaker. An interesting thing to note is that most users skimmed (level 3) a recording to find the general topic of interest after which they switched to level 1 (playing) or level 2 ( pauses removed), usually with time compression.

## 2.2 Binaural audio

### 2.2.1 The cocktail party effect

Most people have experienced it before; you are at a busy party listening to a friend talking, but are also able to switch to another conversation. This phenomenon of selective attention is often referred to as "the cocktail party effect". The ability to selectively attend to a single talker or stream of audio among a cacophony of others or background noise has been studied by many, an extensive overview can be found in Arons (1992a) and Stifelman (1994). Early research made use of dichotic audio to study the phenomena of selective attention. With dichotic audio, 2 messages are played back over headphones, each message on a different ear. Subjects usually had to shadow (i.e. repeat the speech out loud word by word) a primary message, while a secondary message was played in the other ear. Subjects did not recall the secondary message, except when it was the subject's name or some other material of "importance". When subjects needed to detect target words from either the primary or secondary message, while shadowing the primary message, detection of words in the primary message was higher (87%) than in the secondary message (9%). The high cognitive load of shadowing task interferes with the transfer from short-term memory to long-term memory, but non-attended messages do get into short-term memory.

### 2.2.2 Stream segregation

Treisman (1964) studied the effect of irrelevant messages on selective attention (Stifelman, 1994). He found that interference with the primary shadowed messages occurred when the irrelevant messages (0 to 2) where distinguished by voice (male versus female) and spatial location (left, center, or right). When all irrelevant messages had the same voice and location, performance improved. This effect is due to *auditory stream segregation*. When two sound sources have similar acoustic features (like pitch and location), they are perceived as one single sound source. When the irrelevant messages had different acoustic features, performance decreased because the number of interfering channels was higher than when their acoustic features were similar.
Multiple factors exist that can enhance stream segregation, like:

- Difference in pitch (male, female)
- Word transition probabilities (different subject matter)
- Spatial location

Yost (1994) showed that subjects could detect more words, numbers, and letters from three simultaneous sound sources when they were presented binaurally than when presented monaurally. When the number of sound sources increased, subject's performance decreased (Stifelman, 1994).

### 2.2.3 What is binaural audio?

Binaural literally means "having or relating to two ears". Humans hear with two ears. Our auditory system allows us, with only two ears, to perceive sounds from all direction. With these two ears, people can distinguish what direction sounds come from, approximate at what distance the sound source is located and approximate how big the sound source is. There are several perceptual cues we can perceive that make it possible to do this:

- The amplitude of the sound at each ear
- The time difference of the sound at each ear
- The frequency spectrum of the sound at each ear

By applying these perceptual cues to a monaural audio stream, the monaural audio stream can be perceived as being at a specific spatial location. This is best perceived when using headphones.

### 2.2.4 HRTF and HRIR

HRTF stands for Head Related Transfer Function and is a function that describes how a sound wave is modified by the diffraction and reflection caused by the human torso, shoulders, head and pinnae (the visible part of the ears) in the frequency domain. The time domain equivalent of the HRTF is called the Head Related Impulse Response (HRIR). The HRTF varies in a complex way with frequency, elevation, azimuth, range. It also varies significantly per person as it depends on a person's body shape.

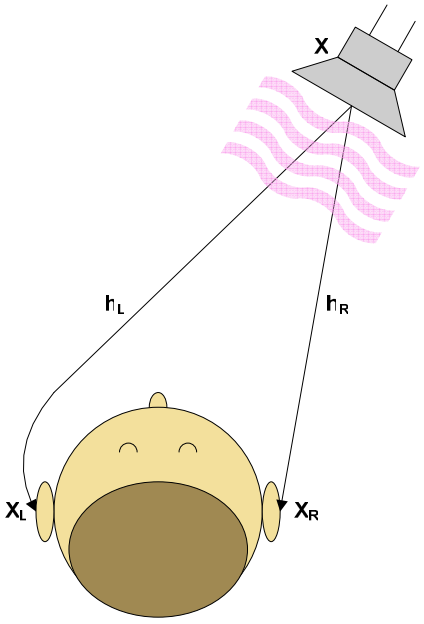


**Figure 3. HRTFs for the left and right ears as HRIRs.**

In Figure 3 the sound $x$ is filtered by $h_r$ and $h_l$ before it arrives at the inner ear as $x_r$ and $x_l$. By applying the HRTF of the person from Figure 3 to a monaural stream, this person perceives the audio stream as coming from a specific direction when heard through headphones. HRTF's can either be experimentally measured, or they can be approximated using a model.

### 2.2.5 Measured HRTF

One way to obtain the HTRF for a given source location is to measure the HRIR of an impulse at the ear drum of a person or dummy. This is usually done in an anechoic room. An anechoic is a room that is isolated from external sound or electromagnetic radiation sources, and prevents against reverberation. The room is completely covered with anechoic tiles, which absorb the sound waves in order to minimize reflections of the sound. In this way, the effect of the surroundings is minimized when measuring the HRIR, and thus only captures the influence of head, pinnae, and/or torso.

**Figure 4. HRIR measurements using KEMAR[1]**

HRIR measurements are done by placing microphones in a human or dummy ear and measuring the response to an impulse $\delta(t)$. The person or dummy is usually located in the centre of the room and is surrounded by loud speakers. The HRIR is measured for a fixed number of elevation and azimuth angles. As an HRIR is person dependent, these measurements sometimes are carried out for different individuals.

[1] http://www.lpi.tel.uva.es/~nacho/docencia/ing_ond_1/trabajos_03_04/Csound/44.htm

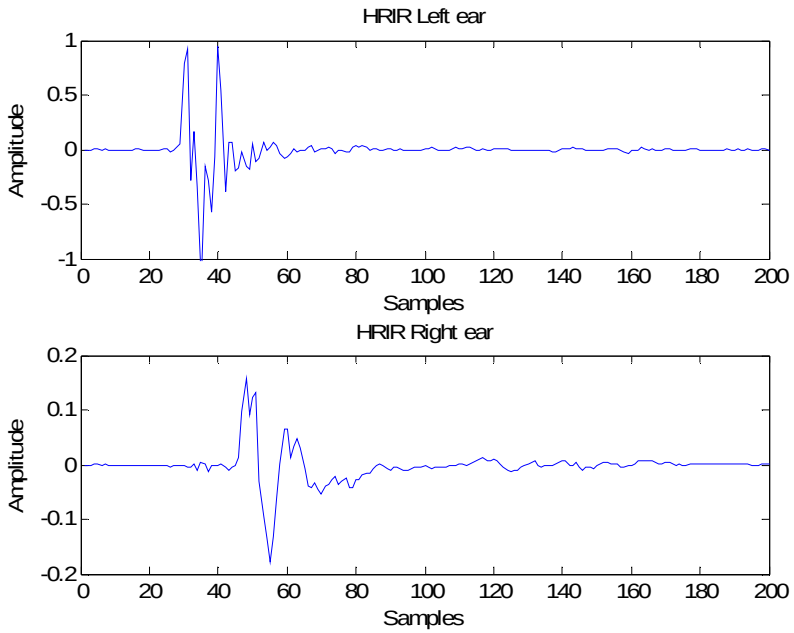


**Figure 5 HRIR for left and right ears at an elevation of 0 degrees and azimuth of -45 degrees**

When the HRIR for a location is captured, a monaural signal *x(t)* can be perceived binaurally by convolving the monaural sound $x(t)$ with the HRIRs $h_l(t,\theta,\phi)$ and $h_r(t,\theta,\phi)$ and presenting the result $y_n(t,\theta,\phi) = h_n(t,\theta,\phi) * x(t)$ on headphones. By interpolating HRIR measurements, HRIRs of locations not included in a HRIR database can be created. In Figure 5 we can see that the HRIR of the right ear is delayed and the amplitude is smaller than that of the left ear. This makes sense as the sound has to travel around the head towards the right ear as is illustrated in Figure 3.

There are several public domain HRIR databases available online, such as the Listen HRIR database (Eckel, 2001), the CIPIC HRIR database (Algazi, Duda, Thompson, & Avendano, 2001), and MIT's KEMAR database (Gardner & Martin, 1995).

### 2.2.6 Model based HRTF

Many systems for spatial sound synthesis make use of measured HRTF's. However, measuring an HTRF accurately is time consuming and experimentally difficult. Low-frequency measurements are specifically difficult, as large loudspeakers are needed during the HTRF measurements. This can be problematic considering the setup of such measuring environments, as can be seen in Figure 4.
Because HRTFs are very person specific, measuring HRTFs is a time consuming and costly task. This problem can be tackled by creating simplified parameterized models that approximate experimentally measured HRIRs. A complete HRTF model that can approximate an experimentally measured HRIR is comprised of different components that account for azimuth, elevation and range. These features are described by the Interaural Time Difference (ITD), Interaural

Intensity Difference (IID) and pinna. Other sub-models, such as a shoulder model, can be incorporated to enhance the approximation.
The model proposed by Brown and Duda (1997) is comprised of the following sub-models:

- Head model
  *describing the ITD and IID as primary cues for azimuth*
- Pinna model
  *the pinna (visible part of the ears) is an important cue for elevation*
- Room model
  *to create an externalization effect using room reverberation and range*

The ITD describes the delay occurring when sound arrives at the ear furthest from the sound source compared to the closer ear. The IID describes the change in intensity caused by the head shadow.

Woodworth and Schlosberg (1954) proposed the following formula to model the ITD (Viste & Evangelista, 2004).

$$\Delta T(\theta) = \frac{r(\sin\theta + \theta)}{c}$$

**Equation 1 Interaural Time Delay model**

The *c* is the wave propagation speed, *r* is the radius of the head, and $\theta$ is the azimuth between sound source and listener. This model was also used by Brown and Duda (1997). The model is based on simple geometric considerations, assuming the head is spherical. In reality the ITD is slightly larger and differs at low frequencies. Other models assume different geometric models like the adaptable ellipsoidal head model (Duda, Avendano, & Algazi, 1999).

The IID is often modeled by a pole filter. The following filter was used by Brown an Duda (1997).

$$H(s,\theta) = \frac{\alpha(\theta)s + \beta}{s + \beta}, \quad \text{where } \beta = \frac{2c}{r}$$

**Equation 2 Interaural Intensity Difference model**

In the model proposed by Brown and Duda (1997), these two models are combined to create the head model. This IID model, unlike the ITD model, does take frequency into account. By adding a pinna model, elevation can be included. The pinna is considered to be a primary source for elevation.

By adjusting the parameters of these HRTF models to the needs of the user, convincing spatially placed audio can be created without having to measure this user's personal HRIRs.

### 2.2.7 Dynamic Soundscape

Dynamic Soundscape was proposed by Kobayashi and Schmandt (1997) and made it possible to browse audio data spatially by exploiting the cocktail party effect.
Browsing audio is not as easy as browsing printed data. We can rapidly skim through printed data to understand the structure of the document and use our

visual spatial memory to recall topics; this is not possible with audio due to the temporal nature of sound. In order to reliably know the topics in an audio stream, we need to listen to the complete stream. Dynamic Soundscape tries to overcome this limitation by simultaneous presentation and mapping of media time to a spatial location. A sound recording is presented to the listener by so called speakers that orbit around the listener. A speaker is a sound stream that is orbiting around the listener's head (with 4.8 degrees per second) by using binaural audio playback through headphones. The position of a speaker is related to the time in the audio recording, thus mapping time to space. There can be 1 to 4 speakers at once orbiting the listener, each playing back a different part of the recording. When the system starts, one speaker is orbiting the listener. When a listener wants to go back or skip ahead in the recording it can point (using a touch-pad) to a previous or future location. A new speaker will be created that starts playing at the indicated location. The listener can focus on one sound source while he/she can still hear the other sound sources in the background and switch to one of these sound sources when something of interest comes up. A rough initial implementation was first made that showed that slow continuous motion made listeners remember more topics and their location. Increasing the loudness of the speaker, towards which the listener is leaning to, enhances selective listening, especially when multiple speakers are talking with the same announcer voice. In order to know towards which speaker a listener was leaning to, a Polhemus sensor was attached to the headset.

### 2.2.8 AudioStreamer

Another non-visual user interface for simultaneous presentation of sound sources is AudioStreamer (Mullins, 1996; Schmandt & Mullins, 1995). Users of AudioStreamer listened to three simultaneous sound sources, segregated by exploiting the cocktail party effect. Users were able to enhance their ability to selectively attend to a single audio stream by bringing this stream into "focus" using head pointing. Subjects could make one of the streams acoustically prominent by moving their head in the direction of the audio streams. By auditory alerting the user of salient events (acoustic cues) the user's ability to perceive events on channels that are out of focus was augmented.
Although some users were positive about the system, no formal user study was carried out to give an objective measure on how well the interface compares to other interfaces. Other users found the interface confusing, tiring or difficult to understand.

# 3 User tests

## 3.1 Browser Evaluation Test

In order to build better meeting browsers we need to be able to evaluate a browser's performance using objective metrics. First we need to define what this objective metric should be:

> *The task of browsing a meeting recording is*
> *an attempt to find a maximum number of* ***observations of interest*** *in a minimum amount of time.*

The above definition was stated by Wellner et al. (2005) who developed the BET (Browser Evaluation Test). The BET aims to be:

a) An objective measure of the browser effectiveness based on user performance rather than satisfaction
b) Independent of experimenter perception
c) Produce numeric scores that can be compared
d) Replicable

To measure a browser's effectiveness, user tests are conducted. In these tests, subjects are exposed to statements about the meeting in pairs of two: a true statement and a false statement.

| | |
|---|---|
| True statement | The group decided to show The Big Lebowski |
| False statement | The group decided to show Saving Private Ryan |

**Table 1. A BET observation**

These statement pairs are called observations. The test subjects have to use a meeting browser to browse through a meeting in order to find out which statement of the presented observations is true. The test subjects get a limited amount of time; half the duration of the actual meeting. Their goal is to answers as many questions as possible in a limited amount of time.
The BET aims to be objective and independent from the experimenters in that the observations presented to the test subjects were collected by observers who are neither experimenters nor meeting participants and thus have no prior knowledge and bias in favor of a certain browser or meeting.

An overview of the BET process is shown in Figure 6.

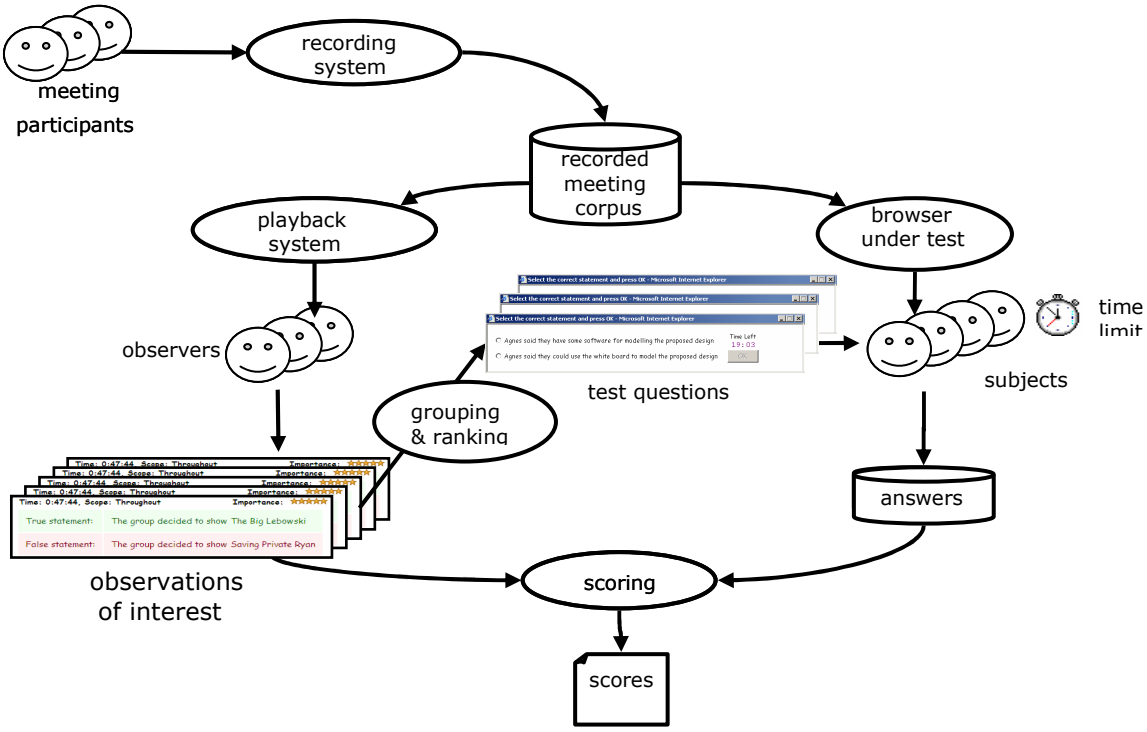


**Figure 6. The BET process as described in Wellner et al. (2005)**

The BET process displayed in Figure 6 can be summarized as follows:

1. Record meeting (of meeting participants)
2. Collect observations (done by observers)
3. Group and rank the observations (done by experimenters)
4. Answer the test questions using a browser (done by test subjects)
5. Compare browser scores

### 3.1.1 Collecting observations

The observations or observations of interest, presented to the users, are collected by so called observers. Observers are neither experimenters nor do they participate in the user tests. Unlike test subjects, the observers are able to see the full recordings without any time limit. Each observer is instructed to produce observations that the meeting participants appear to consider interesting. The observations should be difficult to guess without prior knowledge of the meeting. Observers create the observations in two steps:

1. Create a list of true statements
2. Rate the importance of the statements and create a matching false statement for each true statement

Figure 7 shows the interface the observers use to create the true statements. They can play back and browse through the meeting while submitting observations the participants might find interesting.

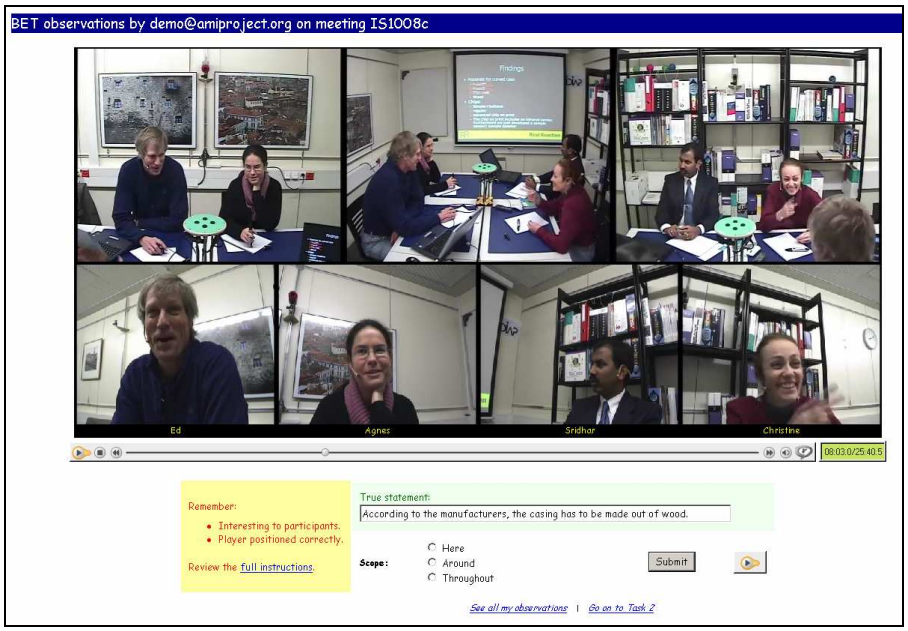


**Figure 7. Creating observations**

After all true statements are generated, the observers use the interface shown in Figure 8 to give each observation an importance value and to create a false statement.

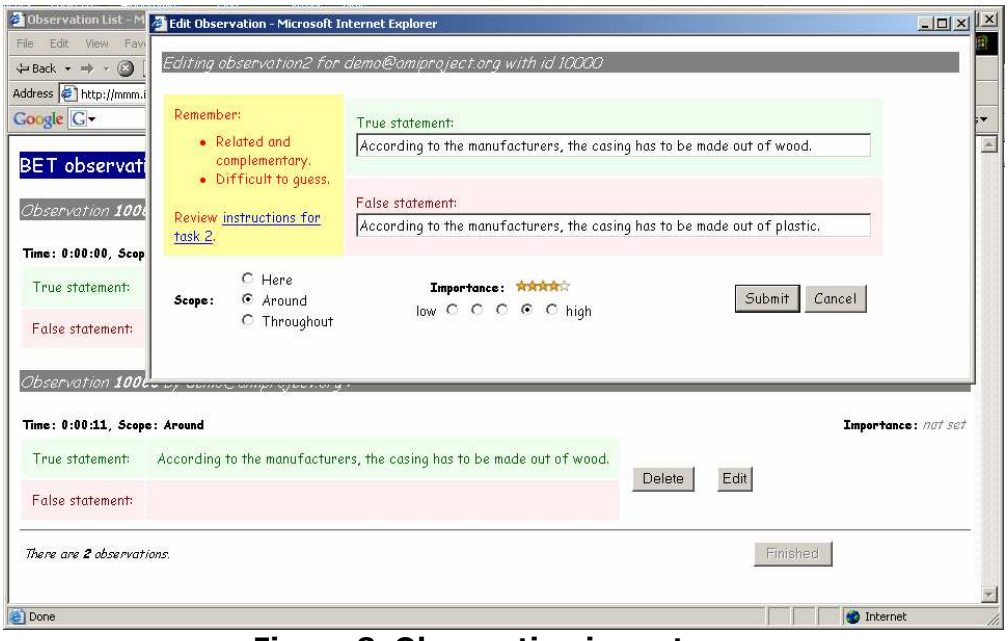


**Figure 8. Observation importance**

After the statements have been collected, observers have indicated how important they think each statement is, and matching false statements have been added, the observations undergo a procedure to group, cleanup and order the observations.

### 3.1.2 Observation grouping

Because there are multiple observers producing observations, many of the observations are the same or similar, e.g. when the true statements of multiple observations are very similar or overlap. These observations are grouped. The media-times of these observations usually are very close. From a group of observations, one representative observation is chosen. These group representatives are the observations that potentially, depending on the order, will be shown to the test subjects.

### 3.1.3 Observation cleaning

Cleaning up of the observations consists of rejecting observations when:

- True statements are not always true, or false statements are not always false
- Statements are incomprehensible
- The true statement can be guessed without prior knowledge of the meeting
- The true and false statement are not parallel enough
- Observations that are not group representatives

One of the goals of the BET is to have experimenter neutral observations to prevent a bias in favor of a certain browser design. The initial idea was to keep all observations as they were written down by the observers. But because observations can contain spelling or grammatical errors, are too long or not explicit enough, or the statements are too complementary to each other, the observations were edited. To prevent a browser specific bias in the observations due to editing, browser development teams could discuss and comment on the decisions on collaborative web pages.

### 3.1.4 Observation ordering

Observations are ordered by group size because this is an indication that many people have noticed this event and find it important. When group sizes are equal, the observations are sorted by *median adjusted importance*. The adjusted importance is the difference between the importance of an observation and the median importance of all observations its observer made. In this way, a high importance rating from someone who usually gives a high importance to observations has less impact than someone who does not give high importance ratings very often. Observations with equal group size and median adjusted importance are sorted by mean adjusted importance and finally by media time.

## 3.2 Test setup

The meeting recordings used for this user test are in English. To prevent variance in the test results due to insufficient English skills, user tests were carried out with native English speakers only. The tests were carried out at the University of Sheffield in Great Britain. A total of 46 people participated in the user test, of which 29 where women and 17 where men.

### 3.2.1 Meeting data

Four meeting browsers were tested using three different meeting recordings. These recordings are part of the AMI Corpus (Carletta et al., 2005). The AMI corpus is a multimodal data set that contains over hundreds of hours of recorded meetings. These meetings are either real, scripted or scenario based. Real meetings concern real projects or issues. Scripted meetings are held according to a pre-specified sequence of actions. Most of these meetings are scenario-based meetings; these meetings are motivated by a scenario or situation given to the participants before the meeting to guide their behavior. The meetings used in this user test were scenario-based meetings.

The following table shows the meetings that were used to test the meeting browsers.

| Meeting | Duration (mm:ss) | BET time | Description / Scenario |
|---|---|---|---|
| ISSCO-Meeting_024 | 47:50 | 20:00 | choosing furniture for the reading room |
| IS1008c | 25:46 | 12:53 | conceptual design of a new remote control |
| IB4010 | 49:20 | 24:40 | English movie club of Montreux |

**Table 2. BET meetings**

These meetings were chosen because the observations for these meetings were already collected. The recorded data of each of these meetings consists of:

- 12 microphone array audio recordings
- **4 headset audio recordings (one for each meeting participant)**
- 4 lapel audio recordings (one for each meeting participant)
- Audio mix of all headset recordings
- Audio mix of all lapel recordings
- **Close-up videos of each participant**
- Center, left and right room-overview videos
- Logitech I/O digital pen recordings (XML, images and video)
- Whiteboard drawings (XML output)
- **Slides (as recorded from the screen, including timestamps)**
- Original files of all presentations
- **Speaker segmentations**
- Speech recognition transcript

The data printed in bold were used in the user tests. For the ISSCO-Meeting_024 meeting the lapel audio recordings were used, since headset recordings were unavailable.

### 3.2.2 Test conditions

The user tests consisted of four different test conditions. Each test subject participated in three test conditions.

| Condition/task | Browser | Meeting | Description |
|---|---|---|---|
| Calibration | General meeting browser | ISSCO-Meeting_024 | Basic browser, with video, audio and speaker segmentations |
| SpeedupBase | Baseline | IS1008c or IB4010 | Like Speedup, but without speed control, playing at normal (1x) rate |
| Speedup | Speedup | IS1008c or IB4010 | Browser that makes use of time-compressed audio. Listen to accelerated playback, between 1.5 and 3 times normal speed, with speaker segmentation but no video |
| Overlap | Overlap | IS1008c or IB4010 | Browser that makes use of binaural audio. Exploits the cocktail party effect to listen to two halves of a meeting simultaneously |

**Table 3. Overview of test conditions and meeting recordings**

These browsers are described in detail in Chapter 4.

All test subjects performed a calibration task to make test subjects familiar with the concept of meeting browsers and in the hope we could use their performance on this task to normalize their results on the actual browser tests. The calibration task made use of both a different browser and a different meeting than the actual browser tests. The actual experimental browser tests were done using the Baseline, Speedup and Overlap browsers.

### 3.2.3 Test structure

Each test subject proceeds through the following five steps:

1. Test with basic meeting browser (20 min.)
2. Tutorial on experimental meeting browser (~10 min.)
3. First test of experimental meeting browser on first meeting (12:53 or 24:40 min.)
4. Second test of experimental meeting browser on second meeting (12:53 or 24:40 min.)
5. Final questionnaire (~10 min.)

Users were free to take a short break in between steps but not during steps, as their performance is being timed.

The first task involves the calibration task, which all test subjects performed. The test subjects were presented with the calibration meeting and general meeting browser that was not used during the actual experimental browser tests. Users

were asked to answer as many questions as possible within the limited time they had.

After the calibration task, each test subject received a written tutorial about the browser they were going to test. The tutorial described every browser control and let the test subjects perform some simple tasks. Each task involved a different browser control.

When the test subjects completed the tutorial they began the test with the experimental browser. Each subject only tested one experimental browser and each browser was tested with two different meetings. The order of the meetings, which meeting is first and which is second, was counter balanced; one half of the test subjects started the test with meeting IS1008c while the other half started with meeting IB4010. Observations differed for each meeting and were presented to the test subjects in a predefined order as is described in section 3.1.4.

Finally, the test subjects were asked to fill in a questionnaire.

### 3.2.4 Data collection

During the BET, the answers and timing of these answers are collected and stored in a database. The answer of each observation is stored as a Boolean telling if the test subject answered correctly or not. For each answer the time remaining till the end of the BET is recorded (time left).
The questionnaire at the end of the BET is designed to allow the gathering of data for both qualitative and quantitative analysis and contain both open-ended questions and Likert scales. Likert scale answers are stored as integers, where -2 represents disagreement and 2 represents agreement.
Each browser also stores a raw message log that contains all the messages that are passed over the ether (see section 4.1).

# 4 Meeting browser design

Three meeting browsers were created and tested with the BET, one for each BET condition (see chapter 3.2.2):

- Baseline browser (see chapter 4.2)
  *A basic audio browser with slides, speech segmentation and timeline*
- Speedup browser (see chapter 4.3)
  *Similar to the baseline browser but with additional speed-slider and time-compressed audio instead of monaural audio*
- Overlap browser (see chapter 4.4)
  *Similar to the baseline browser but plays back binaural instead of monaural audio and with additional balance control*

This chapter describes the choices that were made, based on requirements, literature study and informal tests, and how we came to the design of the meeting browsers that were used for the BET. First we discuss the application framework that was used to develop the browsers followed by the design and implementation of each meeting browser is discussed.

## 4.1 JFerret

A requirement was that the meeting browsers would be implemented using the JFerret framework so existing technologies could be used and development would not need to start from scratch. JFerret is a multimedia browser architecture developed for the AMI project and provides a plug-in API. The JFerret framework[2] is written entirely in Java, making it platform independent. New plug-ins can be easily created and added. By making use of the JFerret framework, a developer can make use of plug-ins that are available and extend the framework for his or her needs. Each plug-in is designed to handle a specific task. A meeting browser is constructed by combining different plug-ins. These plug-ins can communicate with each other using messages. An XML configuration file is used to describe a meeting browser by describing which plug-ins are loaded, the parameters of the plug-ins, to what messages the plug-ins should listen and what message names the plug-ins send.

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<Frame xmlns="ch.idiap.jferret.plugin" title="Test">

  <Audio id="2" name="media2" master="true" src="1.wav" mute="true" volume="1.0"/>
  <Panel>
    <Scroll>
      <Column attribute="time" name="media2" width="180">
        <Slide attribute="move" src="1.jpg" target="media2" value="0918000"/>
        <Slide attribute="move" src="2.jpg" target="media2" value="0968000"/>
      </Column>
    </Scroll>
    <Button name="play" target="media2" attribute="state" value="play"/>
  <Panel>

</Frame>
```

[2] Mike Flynn, IDIAP, Switzerland

The above XML example shows how plug-ins are combined to form a very simple browser. The Frame and Panel plug-ins together create the application window. Slides and a button are displayed within this window. Audio content is also loaded but not displayed. The button, audio and slides listen and/or send to the same target group "media2". The audio starts playing when the play button is pressed. The slides are placed within a scroll plug-in and scrollbars appear when the slides do not fit on the screen. Each slide has a time associated with it. When clicking on a slide a message is sent to "media2" and the media-time of the audio is adjusted to the time associated to this slide.

Figure 23, Figure 12 and Figure 11 give and idea of what a meeting browser can look like in the JFerret framework. These are the meeting browsers used for the BET.
These browsers contain plug-ins such as:

- Frame
  *Acts as the top-level window of an application*
- Column
  *A container that displays its children in a vertical column*
- Row
  *A container that displays its children in a horizontal row*
- Audio
  *A synchronized audio player that makes use of the Java Media Framework (JMF). The source can be a file or a URL using HTTP or streamed using RTSP*
- Video
  *A synchronized video player that makes use of the Java Media Framework (JMF). The source can be a file or a URL using HTTP or streamed using RTSP*
- MediaGroup
  *Represents a synchronized group of media players (Audio and Video)*
- Img
  *Displays an image*
- Panel
  *A generic container that displays its children using a Flow Layout. When provided with a title, the title is displayed together with a border*
- Split
  *A split pane container that displays its children as a row or column separated by narrow, movable divides*
- Scroll
  *A generic scrolling container that displays all its children using a FlowLayout and automatically displays scrollbars when the container is smaller than its children require*
- Text
  *Displays a text label with a single line of text in the given font, size and color*
- Border
  *Similar to the Panel plug-in, but with a border of which the color and thickness can be adjusted. Messages can be sent to a Border plug-in to turn the border on and off at runtime*
- Cards

*A generic stack-of-cards container that only displays one of its children and listens to messages to switch between its children*

- TimeGraph
  *A vertical timeline that displays its children on top of a grid, children being Interval plug-ins. When clicking on the background the represented time is sent to the specified destination*
- Interval
  *A simple Panel that is a visual representation of a labeled time interval. When clicked, the start time of the interval is send to a specified destination*
- TimeScale
  *Inherits from TimeGraph and displays an additional timescale*
- Cursor
  *A line representing the current media time in a TimeGraph or TimeScale*
- Button
  *A generic button plug-in that can embed text or an image. A button can be disabled/enabled by other plug-ins using messages. A button can also send messages.*
- Slide
  *A clickable image that sends a message when clicked on*

The above plug-ins can be combined to create browsers that can display or play back multimodal information such as:

- Speech segmentation
  *Speech segments of each speaker are represented by Interval plug-ins in a TimeGraph with Cursor and TimeScale with a Cursor placed in a Scroll plug-in*
- Slides
  *Images of presentation sheets captured during a meeting can be displayed using the Slide plugin*
- Audio
  *The Audio plug-in plays back the recorded audio*
- Video
  *The Audio plug-in plays back the recorded audio*

Besides being able to display this kind of information, these plug-ins also make it possible to browse through the multimodal recording because of the plug-in message system. Plug-ins can pass messages through the so-called ether. Other plug-ins can listen to the ether and respond to messages for which the plug-in is specifically listening.

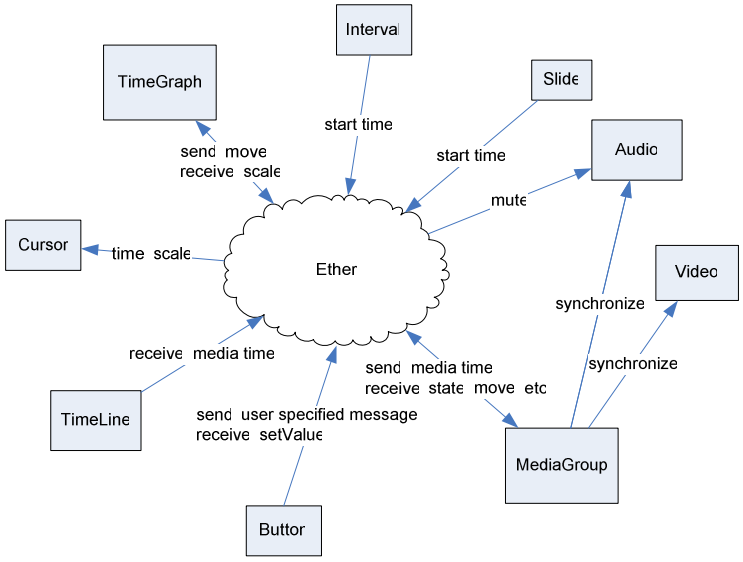


**Figure 9. Plug-ins passing messages through the ether**

In the above figure a user can push a button that sends the state "play" to the MediaGroup. The MediaGroup starts the Audio and Video media players. A user can also click on one of the Slides, which sends a time to the MediaGroup telling it to update the media time of the media players. The MediaGroup also broadcasts the media time over the ether and thus the time Cursor is set to the correct media time. This ether message system makes it possible to add functionality to the framework without changing the framework.
For example: a plug-in can be added that indicates the end of the media recording by playing a sound. This plug-in listens to the media-time broadcasts of the MediaGroup and compares it with the end time of the audio or video.

At the bottom of these browsers a small input form is displayed which shows the false and true statements the test subjects need to choose from and the amount of time that is left for the test. These are not necessarily part of a meeting browser but were used to gather user data during the BET.

## 4.2 Baseline browser

The speedup and overlap browsers are compared with the baseline browser. The baseline browser is similar to the speedup and overlap browsers but without the binaural audio and sped up audio. The baseline browser is basically a fancy audio player that includes meeting related components. These extra components are included to make browsing a meeting easier for the user, thereby making it possible to answer more questions in a fixed amount of time than when a basic audio player, such as in Figure 10, would be used. Test results can be more reliable when users are able to answer more questions.


**Figure 10. Basic audio player**

The components shared by the baseline browser (Figure 11) and a basic audio player (e.g. Figure 10) are:

- Audio player
- Timeline
- Play button

The meeting related components of the baseline browser are:

- Slides
- Speech segmentation
- Speech segmentation viewport indication
- Meeting participant names
- Meeting participant pictures
- Colored picture frames (light up when participants speak)

The slides represent the slides presented during the meeting on a projector and are tagged with the time the slide appeared in the meeting. When clicking a slide the media time of the audio is set to the time at which the slide was displayed during the meeting.

The speech segmentation shows the speech segments of each individual speaker indicated by the colored blobs. The user can click on the colored blobs to set the media time of the audio to the begin time of the blob or click on the background to go to that specific time. The speech segmentation also acts like a timeline; the vertical axis corresponds to the progress of the meeting.

The names and photos of the participants are placed above the speech segmentations to make navigation easier.

For instance, when a participant refers to another participant name, the user can use the speech segmentation to directly navigate to the response of the participant that was referred to by clicking on the corresponding speech segment. When a participant is speaking, a colored box corresponding to the color of the speech segmentation appears around that participant's picture.

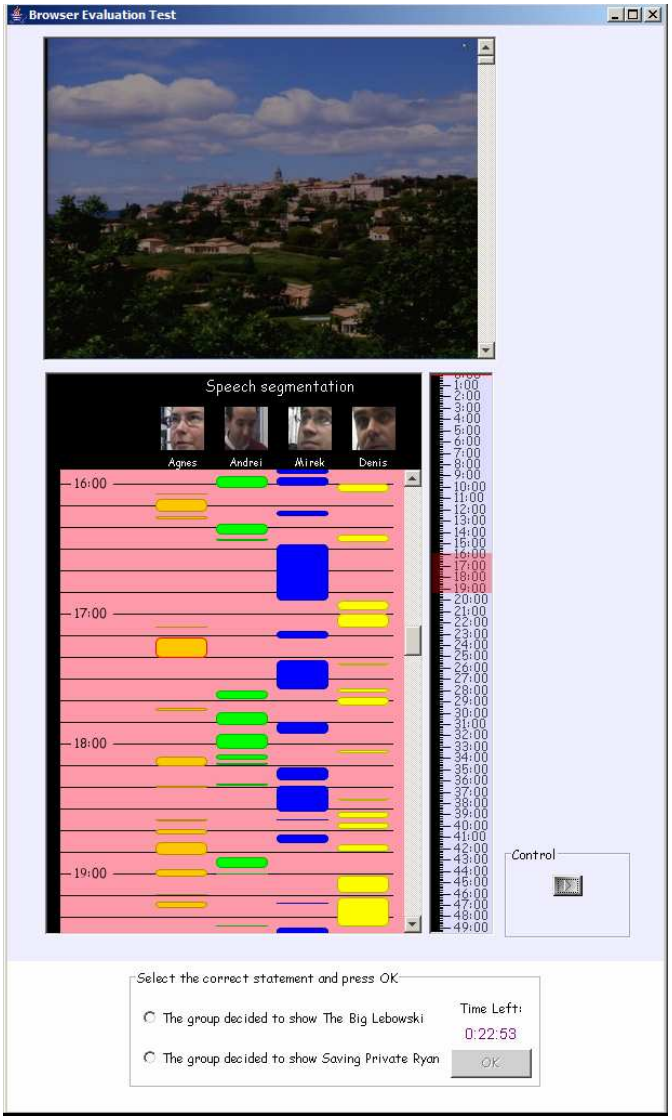


**Figure 11. Baseline browser**

The speech segmentation can be used as a timeline but does not give the user the possibility to navigate from one end of the meeting to the other end of the meeting by just one click. Only a small part of the meeting is visible to make the speech segments large enough to click on. An additional timeline was added to make it possible for the user to navigate to any point in the meeting with one click, as is the case with the basic audio player from Figure 10. Both the speech segmentation and the timeline contain a time cursor. This cursor is displayed that the current media time. The extra timeline contains another cursor. The cursor is displayed in pink in Figure 11 and represents the area that is visible in the speech segmentation viewport. This makes it easier for a user to navigate back to the current media time when the time cursor is no longer visible in the speech

segmentation viewport. This viewport cursor scrolls along with the speech segmentation viewport.

## 4.3 Speedup browser

### 4.3.1 The speedup browser implementation

The speedup browser is similar to the baseline browser except that it provides accelerated audio playback with a user-controlled speed.

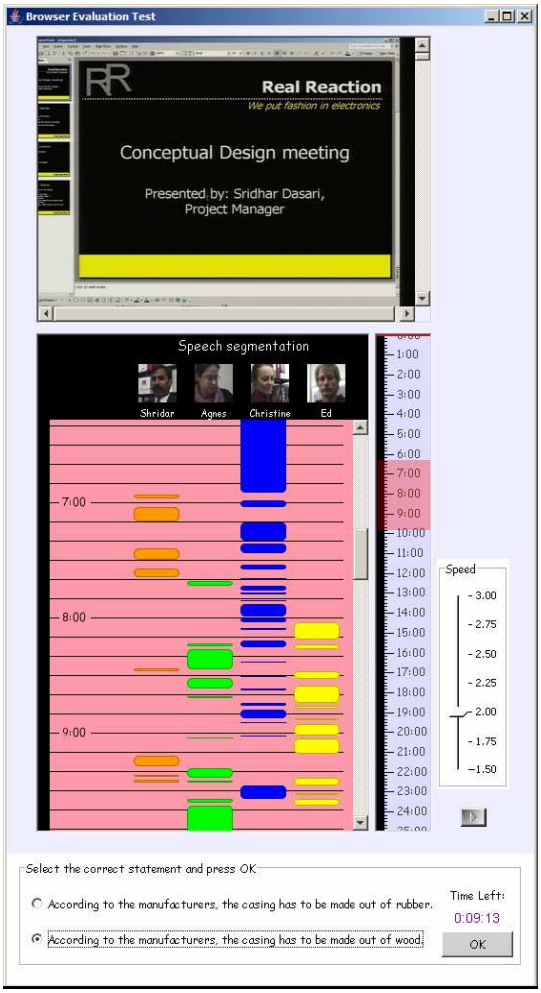


**Figure 12. Speedup browser**

As we can see in Figure 12, the only visible addition to this meeting browser is the extra speed slider control.

The speedup browser does not make use of real-time time-compression but uses preprocessed audio files to provide different speedup levels. The original meeting audio is preprocessed using the PSOLA implementation of PRAAT (Boersma & Paul, 2005). When the meeting is played back, all audio files are played back synchronously (see Figure 13).

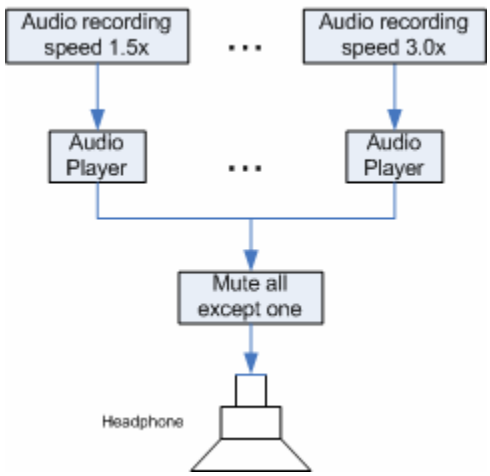


**Figure 13. Selecting the speedup factor**

All but one audio player are muted. The one audio player that is not muted is the audio player that is playing the audio file selected with the "Speed" slider. When the user changes the speed from 1.5 to 2.0, the audio player that is playing the recording with speedup factor 2.0 is unmuted after which the other players are muted. When switching to another audio player the time also has to be changed as they are all playing on the same timeline (see Figure 14). If, for example, the 1.0 times original speed audio is played back, and the media time is 10 minutes. If the user changes the speed to 2.0 times the original speed, the time needs to be adjusted to:
*(currentTime * currentSpeed) / newSpeed = (10 x 1.0) / 2 = 5 minutes*

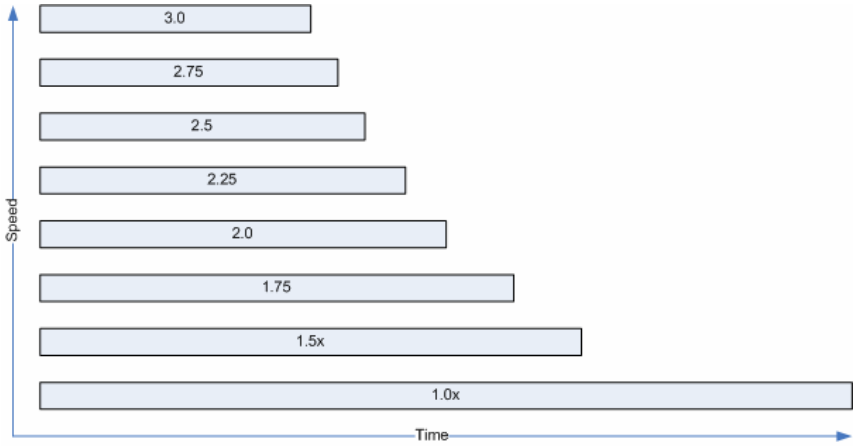


**Figure 14. Speedup browser media timeline**

In total there are 7 audio players playing at the same time but just one can be heard, with speedup factors: 1.5, 1.75, 2.0, 2.25, 2.5, 3.0. It was decided to not let the speedup browser play audio at the original speed to prevent users from ignoring the sped-up speech. The lowest speed the user could select was 1.5 times the original speed.

### 4.3.2 Which time-compression algorithm to use?

To get reliable test results and keep variability to the minimum, the most intelligible time-compression implementation needs to be used. Some publicly

available implementations were tested in a subjective user test. A website was created where samples created using the collected time-compression implementations could be compared to each other (see Figure 15).

The following implementations were tested:

- Phase vocoder matlab implementation (with different window sizes)(Ellis, 2002)
- Solafs matlab implementation (with different window sizes)(Ellis, 2006)
- S.O.X. Sound eXchange – stretch (cross fade) implementation
- Windows Media Encoder – time compression
- MFFM WSOLA v3.9[3]
- Mach1 matlab implementation by Simon Tucker[4]
- PSOLA (Boersma & Paul, 2005)

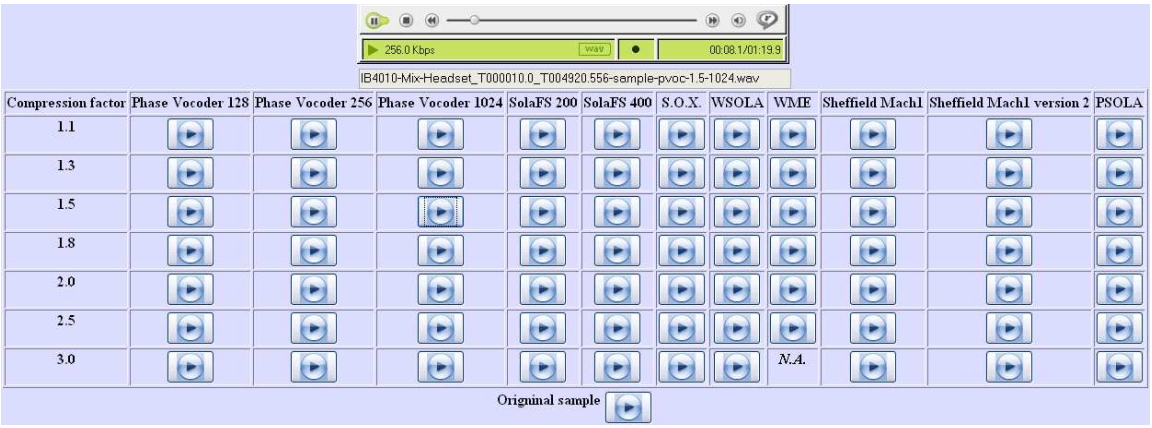


**Figure 15. Time-compression implementation test grid**

Several participants listened to the samples and judged the PSOLA implementation as being the most intelligible implementation. Other implementations became more quickly unintelligible at higher speedup factors compared to the PSOLA implementation. Some people found the Mach1 implementation more tiring to listen to as they needed to concentrate more due to the changing compression rate.
The matlab implementations were, besides being less intelligible at higher speeds, not usable for relatively large audio recordings due to their memory requirements.
The original Mach1 implementation by Covell et al. (1998a; 1998b) showed good results on the samples provided by the authors[5] but unfortunately was not publicly available. A Matlab implementation[6] derived from the original paper by Covell et al. (1998a; 1998b) was tested. This implementation had problems with speeds larger than 2.0 times the original speed. The sped up output sounded like their samples were shifted past the start of their preceding samples, making them sound like reversed speech. Some of the parameters from the script where adjusted to change this behavior. The adjusted script was an improvement but still showed problems with larger audio samples and compression larger than 2.0,

[3] http://sourceforge.net/projects/mffmtimescale/
[4] http://ir.shef.ac.uk/simon/
[5] http://cobweb.ecn.purdue.edu/~malcolm/interval/1997-061/
[6] Simon Tucker, University of Sheffield

while the PSOLA implementation sounded consistent when used on both small and large samples and all compression rates.

## 4.4 Overlap browser

Before discussing the actual user interface of the overlap browser, we first need to look at how the audio playback is handled by the overlap browser.

### 4.4.1 Interaural Time Difference

Different methods, both experimentally measured HRTFs and HRTF models, were tested to create binaural audio to see which method would be most useful for our Overlap Browser implementation. It was decided was to use the spherical head ITD model as described by Equation 1 in section 2.2.6 on the following observations:

- The spherical head ITD model is very simple, yet powerful
- The ITD is the main primary cue for azimuth
- Measured HRTFs cause a drop in the low frequencies due to the use of small speakers during the measurements, causing the audio to sound processed and less realistic
- We do not need binaural audio that is located at exact locations, we only need spatial segregation of the audio streams

An azimuth of -90 degrees would mean that the sound source is located left of the listener, 0 degrees is right in front of the listener and +90° is on the right side of the listener. When the sound source is placed at +45° of the listener, a wave propagation time of 343 m/s and head radius of 0.0875 meters are assumed, and the left channel of the stereo audio stream is delayed by 0.38ms compared to the right channel. With a 16 KHz recording this would mean that the left stereo channel starts playing the same monaural recording $0.00038 \cdot 16000 \approx 6$ samples later than the right stereo channel. When the sound source is located at -45° azimuth, the right channel is delayed by 6 samples.

### 4.4.2 What to overlap?

We will refer to a binaural audio stream as overlapped audio, since the final multiple monaural audio streams are played back in parallel (see Figure 17). These monaural audio streams are not just mixed together but are first given a special location using the ITD to enhance the segregation of the audio streams in order to increase intelligibility.

When we want to reduce the playback time of an audio recording by simultaneously playing back parts of a recording using binaural audio, or "overlap" as we called it, different strategies can be chosen:

- Overlap speakers and remove pauses
- Overlap parts of a recording where there are not many speakers speaking simultaneously
- Overlap fixed meeting parts without looking at the content

Since we would like to create spatial separation, hearing the same speaker simultaneously should be avoided. Having the same speaker positioned at the

same location is even worse. Audio streams blend together to a single auditory stream when their acoustic features, pitch and location, are similar (Stifelman, 1994).

The following image shows how a binaural audio stream is created out of *n* monaural audio streams.

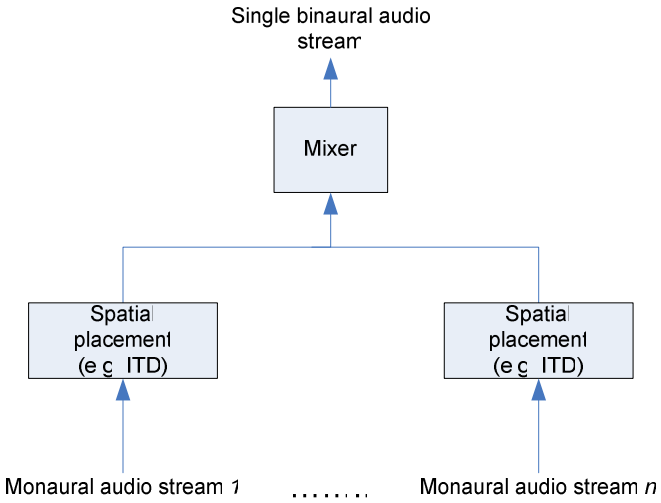


**Figure 16. Schematic presentation of how a binaural audio stream is created from *n* monaural audio streams**

The output of the spatial placement in the case of ITD is a stereo signal. The mix of the *n* processed audio streams also is a stereo signal and is presented to the listener.

The question that remains is how audio can be best overlapped in order to reduce the playback time of the audio recording while not removing any audio and maintaining intelligible audio.

### 4.4.3 Overlap speakers and remove pauses

One way to make sure that a speaker is not overlapped with him or her-self is to overlap different speakers. But it is still possible that two different speakers have similar pitch. Pitch shifting of similar voices increases stream segregation (Arons, 1992a; Hawley, Litovsky, & Culling, 2004; Stifelman, 1994). As there are separate headphone and lapel microphone recordings of each individual meeting participant available for most meetings, we have the possibility of overlapping the participant's speech. If all meeting participants would overlap with each other the playback time of the meeting would not be reduced. When removing or reducing the pauses in each participant's speech, the playback time of the meeting would be as long as the participant whose speech added together was longest. The following figure shows an example of a meeting or conversation with 2 people.

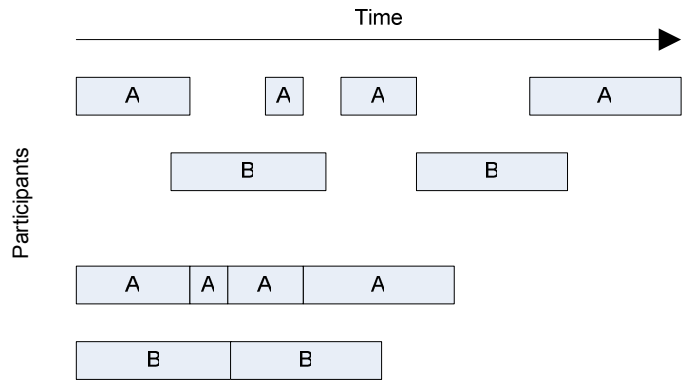


**Figure 17. Overlap speakers and pause removal. Each block represents a segment of speech**

In Figure 17 the pauses are removed while the speakers are completely overlapping each other. In this case the playback time was reduced by 37.5%.
A big problem with overlapping speakers and removing pauses is that there is no interaction left between the speakers. Meetings can be seen as dialogues. Dialogues exhibit anaphora, discourse structure and coherence (Jurafsky & Martin, 2000). Removing or reducing pauses for each individual participant extensively damages the discourse structure. The time at which participants speak is adjusted and this removes the relationship of statements and the response to these statements. Participant A might be talking about topic 1 while participant B might be responding to a statement of topic 3. The number of participants is also an issue. The meetings we used all contain 4 meeting participants; in many real meetings this number can be even higher. Stifelman (1994) reports a strong decrease in performance when more than 2 streams are presented to users, so even though we would like to use simultaneous playback of audio to decrease the playback time, using too many simultaneous streams makes the speech unintelligible.

### 4.4.4 Overlap parts with least amount of overlap

Another option might be to overlap parts of a meeting where mainly one participant is speaking. Figure 18 and Figure 20 show the moving average of the percentage of time each participant is speaking at each moment of the meeting using a moving average window size of 300 seconds. We can see multiple peaks of different meeting participants in the graph. When we look at the content of the meeting we see that these peaks correspond to the mini presentations the participants gave during the meeting. The meetings, displayed in Figure 18, start with an introduction, followed by several short presentations by a different participants, and end with a discussion.

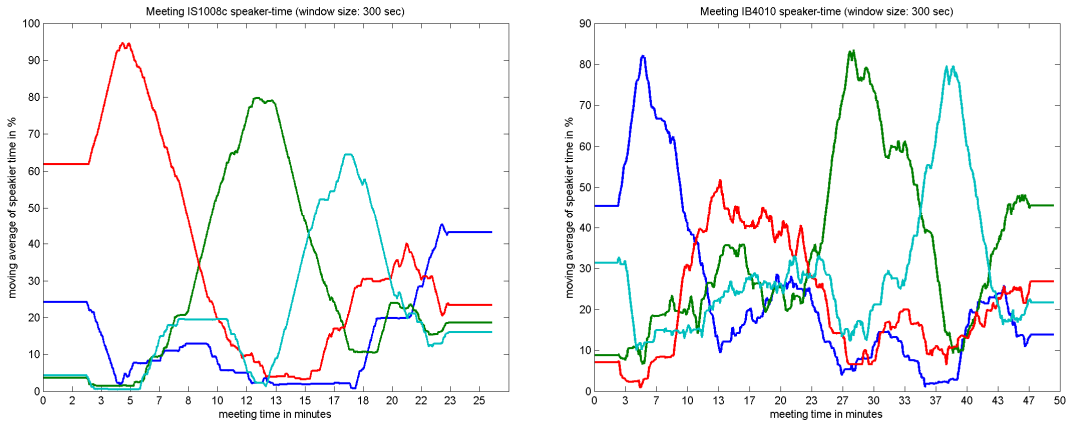


**Figure 18. Moving average of the per person speaking time during meetings IS1008c and IB4010**

These peaks can potentially be used to overlap with other peaks to prevent too much simultaneous speech while preserving the conversation structure during these peaks. It can also prevent overlapping the same speaker. As people usually do not interrupt while someone else is speaking and the duration of these peaks is reasonably long, overlapping these peaks would result in 2 streams where the majority of the simultaneous speech comes from 2 different speakers. If, for instance, we have a look at the green peak of meeting IS1008c in Figure 18, we see that the speaker is only overlapping with someone else for 3.5% of this segment. There is 15.6% no speech and 80.8% of the time one speaker is speaking. The start and end-times for this segment were determined by manually choosing the start and end-time of the "mini-presentation" that is represented by this green peak.

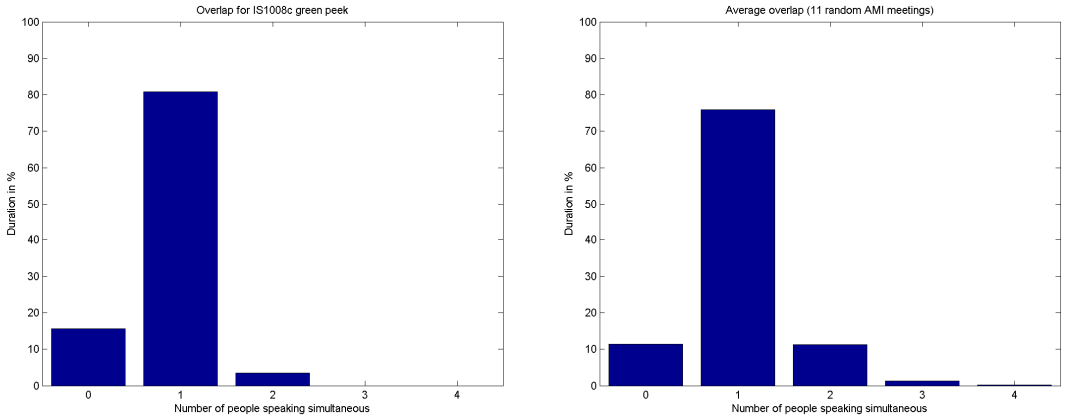


**Figure 19. Percentages of overlap of one specific time segment and the average of 11 randomly chosen meetings**

Taking 11 random meetings from the AMI corpus we can see in Figure 19 that on average 75.9% of the time only one person is speaking. On average only 12.75% of the time two to four people are speaking at the same time and on average 11.34% of the time nobody is speaking. The average duration of the overlapping speech is only 0.4 seconds whereas the average duration of speech for a single person is 2.32 seconds. The overlapping speech might be for a large part due to small overlap near turn taking events. The segments that do have multiple

participants speaking at the same time often exist of laughter and contain little speech information but can interfere with the another stream when overlapped with it.

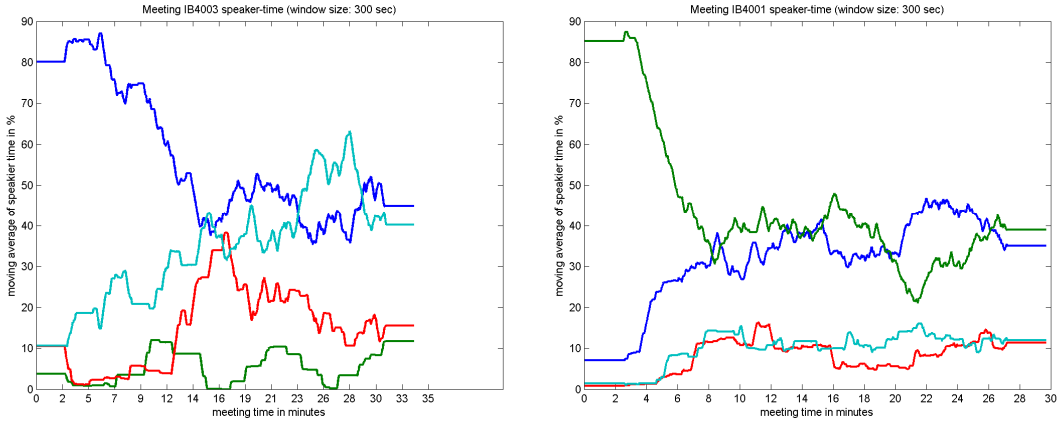


**Figure 20. Moving average of the per person speaking time during meetings IB4001 and IB4003**

When we look at Figure 20 we see that not all meetings follow the structure as shown in Figure 18. The meetings in Figure 20 do not contain peaks that indicate one meeting participant is dominating during a certain time frame. Many meetings differ in structure and thus overlapping peaks is not an option that can be applied to every meeting.

### 4.4.5 Overlap fixed parts

As the intelligibility decreases significantly when more than 2 speakers are overlapped and there is a reasonable low percentage of overlapping speech during meetings (see Figure 19), the decision was made to cut meeting recordings in half and to overlap these two halves with each other. This method does still allow overlapping a speaker with him/herself but we assumed that meetings are not long monologues and that the percentage of overlap of the same speaker is reasonably low.

Like the speedup browser, the overlap browser is quite similar to the baseline browser except that the overlap browser plays back two meeting halves simultaneously.
The first stream (audio file) contains the first half of a meeting and the second stream the second half of a meeting. The first stream is positioned at -45 degrees azimuth of the listener and the second stream at 45 degrees azimuth of the listener. These angles were chosen to limit the time a subject needs to switch attention between audio streams, while having enough spatial segregation.
When a meeting is played back, both streams are played simultaneously, like playing two media players at the same time. For the listener this sounds similar to listening to two speakers at the same time at a different position. From each stream there is also a version available where the pitch is shifted up by one semitone. When the meeting browser detects that the same speaker is speaking on both streams, one of the streams switches to its pitch-shifted version, to prevent the streams from blending together and becoming unintelligible.

### 4.4.6 Pitch shifting

To reduce interference caused by having the same speaker speaking on both streams simultaneously, pitch shifting is performed on one of the streams. Whenever speaker X is speaking on one stream and this same speaker starts speaking on the other stream, the pitch of the stream where speaker *X* started speaking last is raised by one semitone.

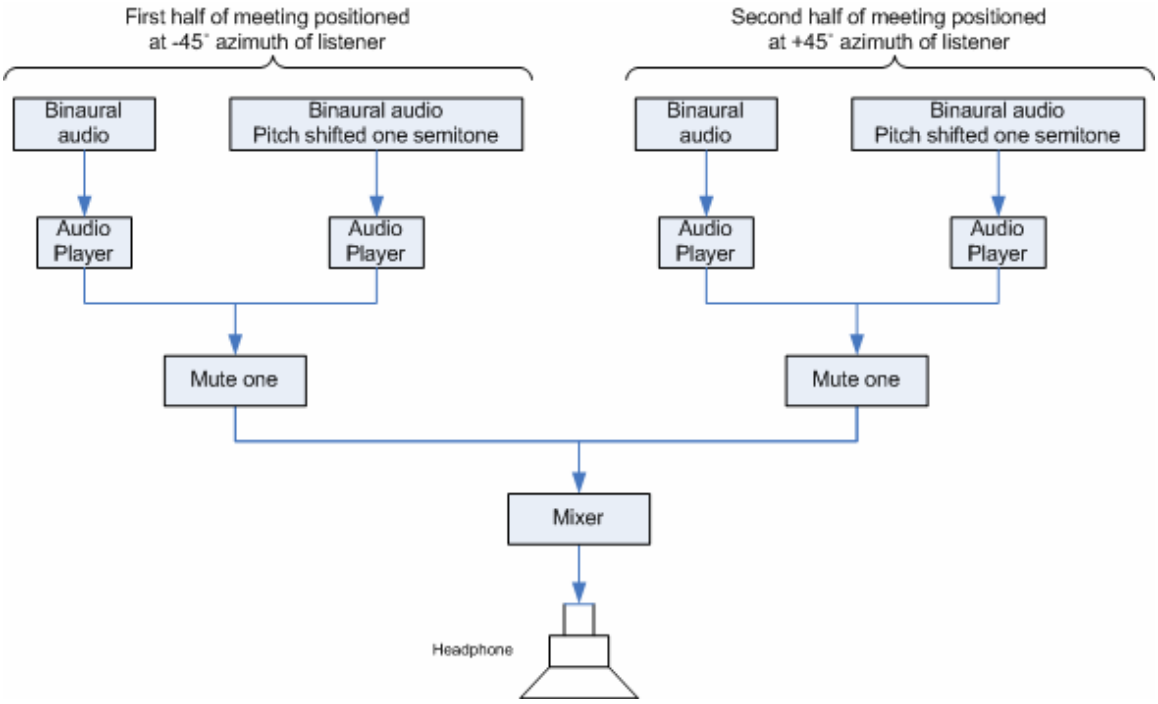


**Figure 21. Pitch shifting the binaural audio up one semitone**

Two streams are played back simultaneously, each representing a different location and meeting half. In reality, four audio players are playing back the recorded meeting halves, for each meeting half a pitch-shifted version is played back synchronously with the original audio. These pitch-shifted versions are muted. When the pitch of one side needs to be raised, the original audio file is muted and the pitch-shifted version is un-muted (see Figure 21). The speech segmentation is used to detect when the same speaker is speaking in both meeting halves at the same time. The pitch shifted audio streams were raised by only one semitone to prevent users from getting confused by who suddenly started speaking, but still making it possible for the user to segregate the streams.

The speech segmentation cannot be used to detect when people with similar pitched voices are speaking simultaneously. To overcome this each participant's voice should be analyzed to see which participants have similar pitched voices and use this information to change the pitch of one of the two audio streams. This was beyond the scope of this project.

### 4.4.7 The initial overlap browser implementation

An initial implementation was made of the overlap browser (see Figure 22). The overlap browser should be similar to the baseline browser but with addition of overlapped meeting playback and related controls, if needed. Because the

overlap browser is playing back two meeting halves, the user interface basically exists out of two baseline meeting browsers.

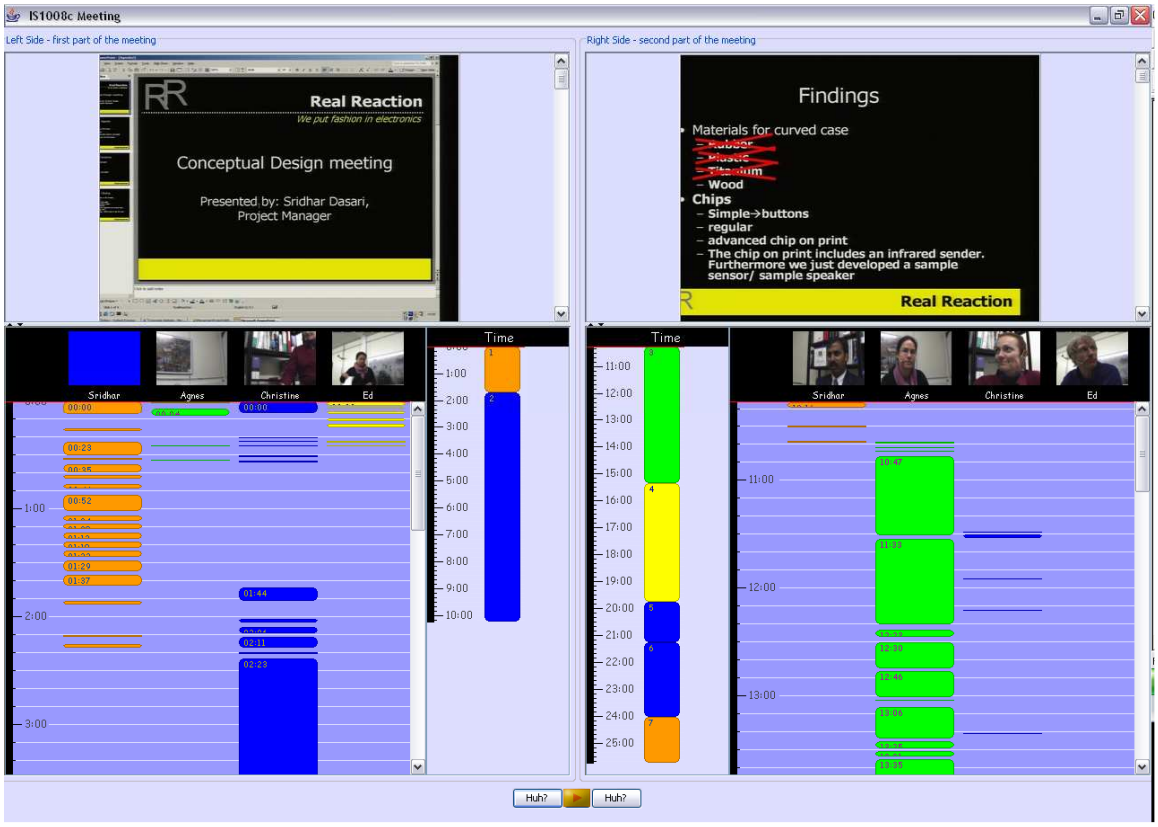


**Figure 22. Initial overlap-browser implementation**

This initial implementation has the views and controls of the first half of the meeting on the left side and the views and controls of the second half on the right side. As we can see, instead of dividing the meeting exactly half way, the meeting was divided at a point where a different speaker started presenting his or her presentation. This was done to prevent users from suddenly starting in the middle of a conversation during the second half of the meeting.

A difference from the baseline and speedup browsers is the substitution of the participant photos by close-up videos of the meeting participants. This change was made to make it easier for the user to see who is speaking on which side and possibly make use of lip movements to enhance the intelligibility of the speech, as is suggested by Cherry (Arons, 1992a; Mullins, 1996). The video option was not used in the speedup browser because the videos would need to be sped up. This was technically unfeasible. Two additions to this browser are the buttons left and right of the play button. These buttons are labeled "Huh?" and where added to give users the ability to repeat the previously played 5 seconds of the meeting while muting either the left or right side. Users can use this option, e.g. when speech from one meeting half gets unintelligible due to speech from the other meeting half.

The overlap browser is not just two baseline players in one window but is one browser that has views and controls for both meeting halves. The difference with just having two baseline players next to each other is that the timelines are linked to each other, and thus move simultaneously. For example, when a user moves 10 minutes into the first halve of the meeting, the media time on the right

side moves 10 minutes into the second half of the meeting. This was done to make sure each user is presented with the same overlapped speech parts, even when their browsing behavior differs.

### 4.4.8 Initial overlap browser user test

A subjective user test was carried out on an initial overlap browser-design to see how users would respond to this browser and the overlapped audio. This user test was only carried out for the overlap browser as we expected this browser to be more confusing than the speedup browser.

Two users were given 4 randomly chosen observations, each existing of a complementary pair of statements, one true and one false. They were asked to use the overlap browser to find out which statement was the true statement. The users were asked to speak aloud while using the browser, to describe their immediate reaction on using the interface components and why they were using them. No time limit was given; the user just had to answer the 4 questions.

The most relevant findings were:

- Users need a better explanation of the browser and browser components; expectations differed
- Male speakers were difficult to understand when a female speaker was speaking on the other side
- Users found the two timelines confusing
- Users wanted more control on the volume of both sides
- Users wanted more control on what parts to overlap

**Conclusions**
Both users stated that high pitched voices overrule lower pitched voices. Interference also occurs when similar pitched voices or multiple people speak at the same time. A balance control to change the volume of the right and left channels can help reduce these problems. It also gives the user more control on which meeting half to focus on. A user might want to focus more on the right channel by reducing the volume on the left side. By not muting the left side completely, the user can still listen to the left side and increase the volume again when the user thinks the left channel might be of interest or when the speech on the left channel interferes less with the right channel.
One user was mainly interested in having more control on what was being overlapped. As we are trying to test if overlapped speech can increase the browsing efficiency we will stick to having both timelines on each side fixed to each other, so as to not increase variability. Giving users more control over which speech is overlapped might increase the variance as some users might overlap audio parts of importance with interfering speech while other users do not. A more detailed overview can be found in Appendix A -.

### 4.4.9 Final overlap browser

Changes were made to the overlap browser based on the input from the test subjects (see Figure 23).

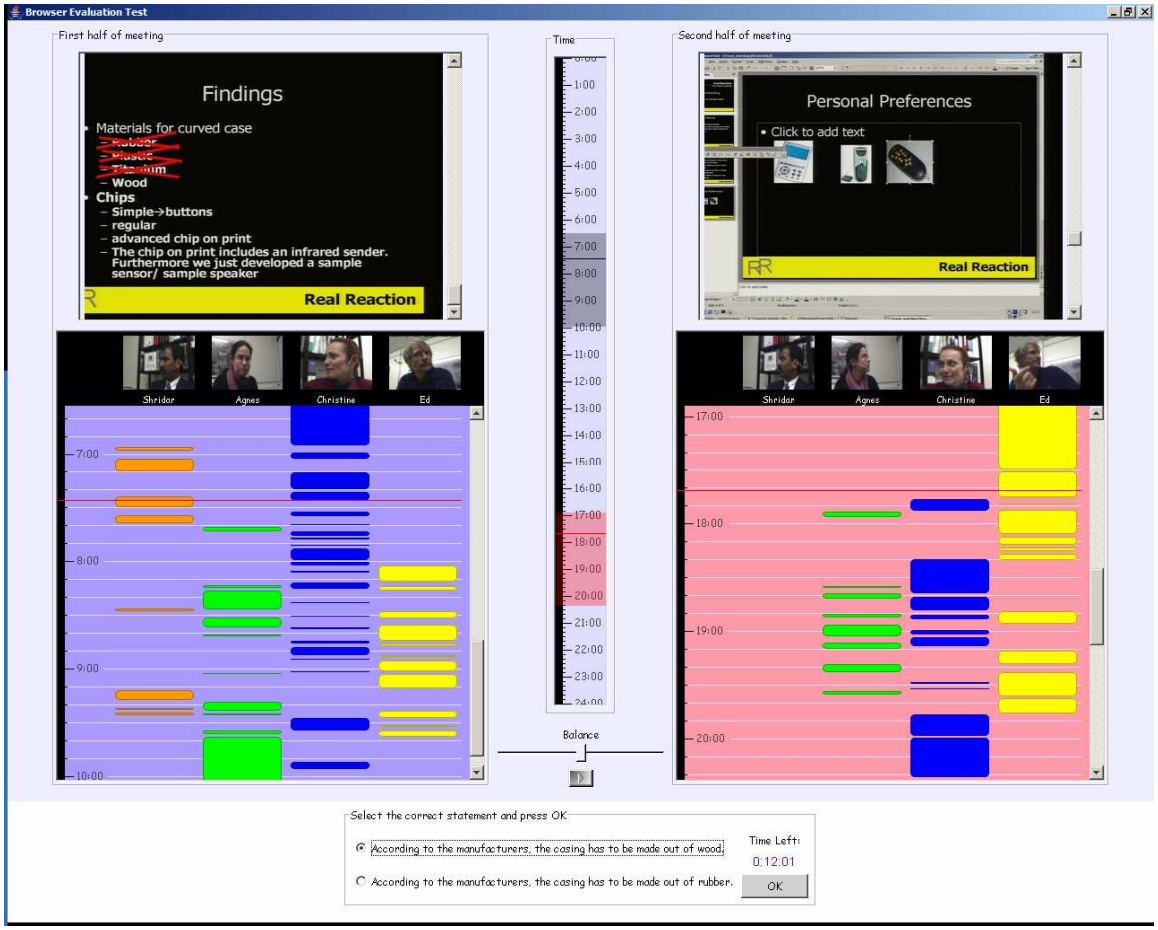


**Figure 23. Overlap browser**

To clarify that the overlap browser is actually one browser with additional controls and not two separate browsers, the two timelines were combined into one timeline. This might also promote the feeling that users are listening to one meeting.

The different speech-segment viewports were given distinctive colors that match the background color of the corresponding viewport. An important change was the replacement of the "Huh?" buttons by a balance control. When this balance control is centered, the volume on both sides is set to the maximum. When the balance control is moved for example to the left, the volume of the right half is lowered. The volume cannot be turned down completely as to force users to always try to listen to both meeting halves.

# 5 Results

## 5.1 The subjects

As a result from malfunctioning meeting browsers, 7 out of 46 subjects were not included in the analysis. The 7 subjects were part of the group in the Speedup condition.
Subjects on average make use of computers 25.5 hours per week ($\sigma$ = 15.6); 3 of the 37 people who filled in the questionnaire indicated their computer usage as 30+, 30-ish or 40-50 hours, these were interpreted as 35 or 45, respectively.
The average age of the subjects was 34.7 years ($\sigma$ = 13.9, median = 30, mode = 26). From the 37 subjects who filled in the questionnaire, there were 21 females and 16 males.

## 5.2 The data

When looking at data of each individual subject we notice that subjects exhibit different behavior. There is a huge variation in how much time users spend per question. Subjects tended to spend more time on the first few questions, after which their behavior changed.
A few subjects did not change their behavior and kept searching until they found the correct answer, resulting in a longer time-spent-per-question and thus fewer questions answered. Some subjects started off slowly but seemed to start guessing after a few questions, their time to answer questions dropped dramatically. Other users constantly changed the speed with which they answered questions. This might be due to the difficulty of the question or because the user might assume he will not find the question anyway and chooses to guess it. Within these groups there is also a lot of variation in accuracy. A few subjects that spent above average time per question still had low accuracy, some scoring even lower than random guessing (Wellner et al., 2005). However, other subjects that were patiently answering questions did have very high accuracy. Also, some of the subjects that were able to answer many questions scored better than some slower subjects.
None of the subjects from the speedup condition scored lower than random guessing. However, some subjects from the overlap and baseline conditions did score worse than random guessing.

The raw speed and accuracy scores are shown in Figure 24, Figure 25 and Figure 26, each dot representing a user. The numbers on the x-axis show the number of questions answered per minute. The y-axis show the percentage of questions each subject answered correctly.

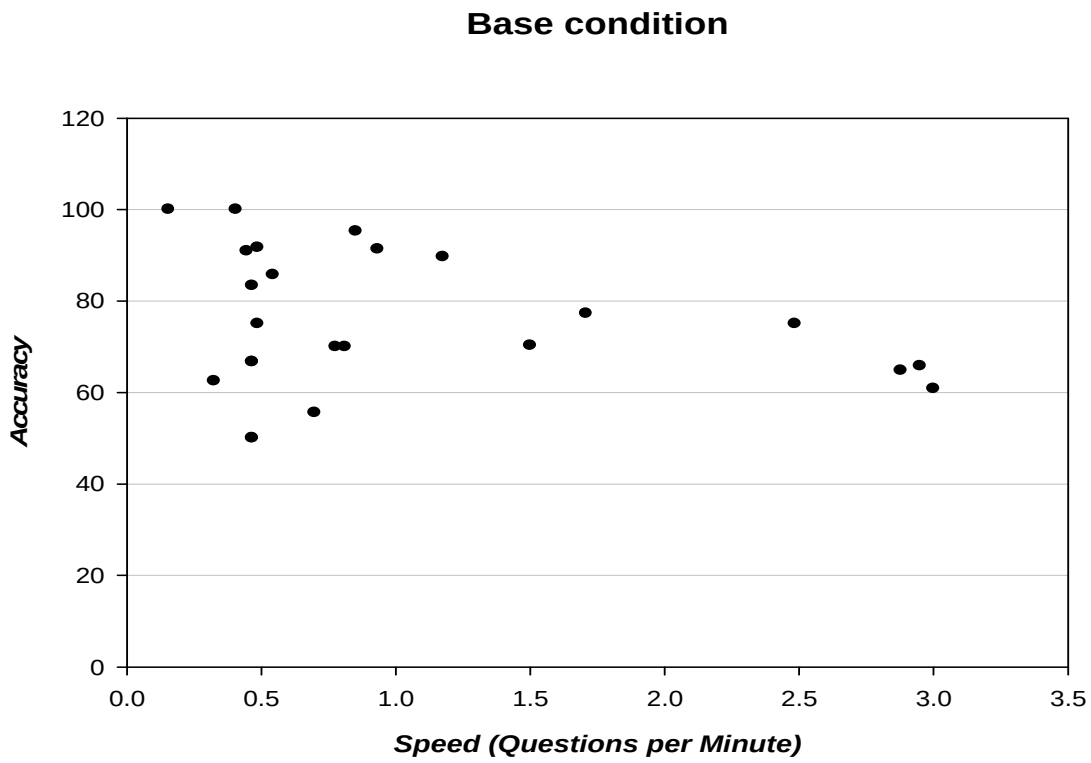


**Figure 24. Raw test scores base condition**

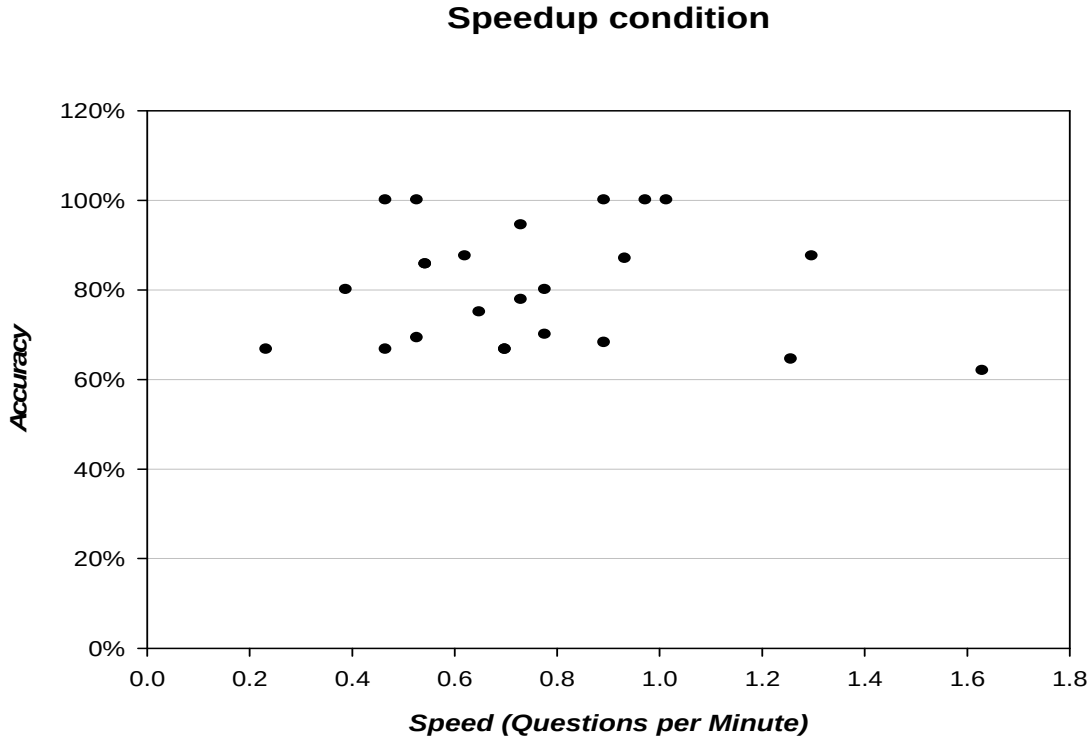


**Figure 25. Raw test scores speedup condition**

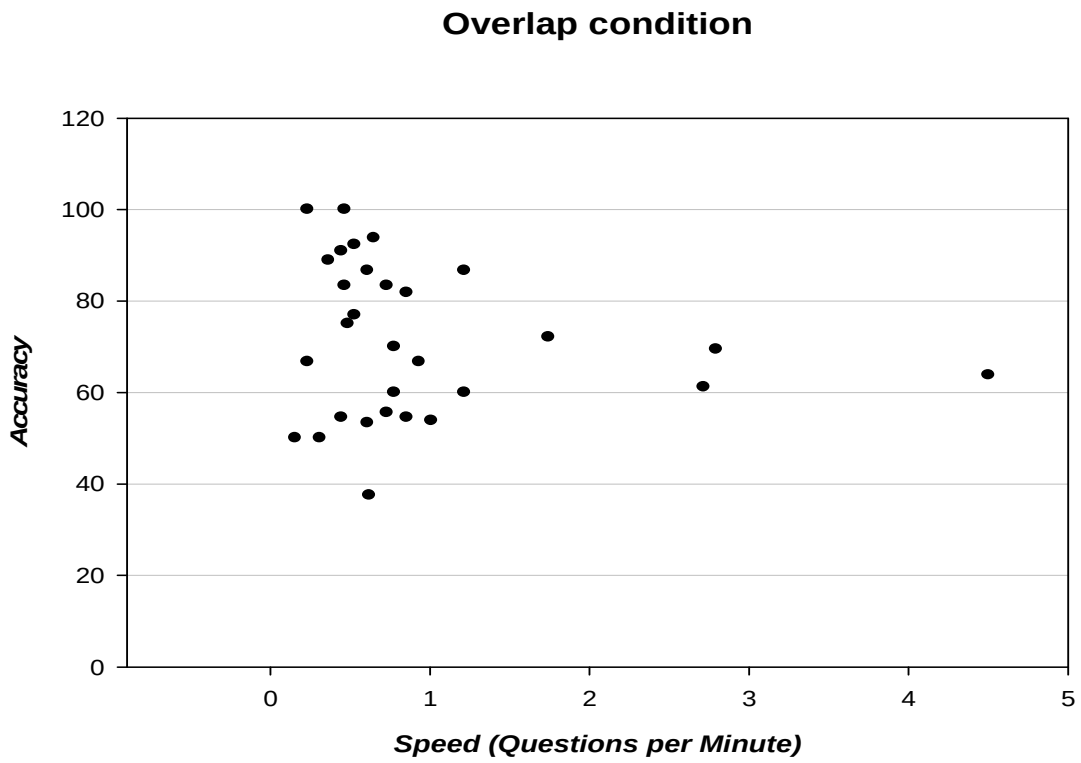


**Figure 26. Raw test scores overlap condition**

All three figures show a lot of variance. Subjects from the baseline and overlap conditions did answer questions at a higher speed than the speedup condition. The baseline and overlap browsers seem to follow a rough reciprocal curve, while the speedup browser does not. We suspect that as users answer at a higher speed, the speedup browser also follows a rough reciprocal curve. Although the results contain a lot of variation, we suspect that the maximum accuracy of each condition follows a rough reciprocal curve.

## 5.3 BET Scores

We assumed that a meeting browser, with which users can answer more questions in a limited amount of time, is more efficient than a meeting browser with which users can answer fewer questions in the same amount of time. To compare the browsers, the mean accuracy and mean speed values were calculated.

The following table shows the results per browser (condition), per meeting, and the average of both meetings.

| Condition | Number of subjects | Mean Accuracy | | Mean speed (questions per minute) | |
|---|---|---|---|---|---|
| | | *IS1008c* | *IB4010* | *IS1008c* | *IB4010* |
| Base | 12 | 70.5% | 80.2% | 1.20 | 1.37 |
| | | **75%** | | **1.3** | |
| Speedup | 12 | 76.5% | 85.3% | 0.65 | 0.87 |
| | | **81%** | | **0.8** | |
| Overlap | 15 | 67.2% | 75.4% | 1.00 | 0.87 |
| | | **71%** | | **0.9** | |

**Table 4. BET results: mean accuracies and speeds, average of each meeting average**

The mean values in Table 4 are calculated by taking the average of the average of each individual meeting. Another possibility of calculating the average accuracy and speed for each test condition is by combining the results of both meetings and taking the average of all results, making it more robust against outliers.

| Condition | Number of subjects | Mean Accuracy | Mean speed (questions per minute) |
|---|---|---|---|
| Base | 12 | 77% | 1.2 |
| Speedup | 12 | 83% | 0.9 |
| Overlap | 15 | 74% | 1.0 |

**Table 5. BET results: mean accuracies and speeds, average of all observations individually**

There is no significant difference between both results. Both tables show an increase in accuracy for the Speedup condition compared to the Base condition, but a decrease for the Overlap condition. When we compare the means of the speeds we see a decrease in speed for both tested browsers compared to the Base condition. It is difficult to draw clear conclusions from these results. The results also contain a lot of variance, like in the first BET trial (Wellner et al., 2005).

| Condition | Standard Deviation Accuracy | Standard Deviation Speed |
|---|---|---|
| Base | 13% | 0.89 |
| Speedup | 12% | 0.26 |
| Overlap | 13% | 0.8 |

**Table 6. Standard deviation of raw test scores**

The results from Table 4 and Table 5 are non-normalized scores. The data was not normalized using the data from the calibration condition, as there was little correlation between the test meetings and the calibration meeting.

| | IS1008c | IB4010 |
|---|---|---|
| **Accuracy** | 0.18 | 0.28 |
| **Speed** | 0.37 | 0.42 |

**Table 7. Correlation (Pearson's r) between calibration condition and the test condition meetings**

The correlation coefficients of the calibration condition and the meetings IS1008c and IB4010 show there is very little correlation, suggesting that users showed different browsing behavior in the test conditions and that the use of the calibration condition is questionable. The correlation of speed between the two test conditions was much higher (r=0.84).

## 5.4 Speed/Accuracy trade-off model

Subjects have a fixed time to complete as many questions as possible and as correctly as possible. However, where the balance lies between speed and accuracy is open to the subjects. Any individual subject might perform with huge variation.

The speed and accuracy values presented in Table 4 and Table 5 do not give intuitive insight into whether one browser is better than the other. Even if the results contain little variance, it is not possible to say that a browser with an average accuracy of 75% at 0.5 questions per minute is better than a browser that has an average accuracy of 50% at 1.5 questions per minute. It would be more intuitive to express the efficiency of a browser with just one value.

When pressed on accuracy, a subject might spend significant time searching for an answer. But when a subject is pressed on time, he or she might guess the answer based on the smallest clue found. A subject's behavior can vary widely, ranging from slow but accurate, to fast but inaccurate, even within the same test.

Any browser might be used by a subject, willing to spend much time answering questions, to obtain good accuracy. But the same browser in the hands of a less patient subject might yield poor accuracy. We claim that the superiority of a good browser lies in its ability to impart a greater speed for the diligent, but greater accuracy for the impatient.

We suspect that the location of a subject in the speed/accuracy trade-off spectrum depends on the browser and that the shape of this speed/accuracy trade-off curve can act as a measure to compare the browsers.

### 5.4.1 Method

Each subject is looking for the answer to some questions using a browser. The answer lies somewhere in the recording, which we assume is at one specific instance. The subject chooses to play a specific part of the meeting and may or may not find the answer to his question. When the answer is not found, the subject can either try again or guess the answer.
Without the help of a browser, the probability that a subject finds the answer to a question is just random chance. However, the use of a particular browser might be considered to boost this probability, whereas an ideal browser could take the subject directly to the answer every time.

The BET data may be analyzed as follows:

1. Each subject is given a browser, a recording, and a list of questions to answer.

2. We assume that the questions are independent and that subjects do not accumulate knowledge of the media.
3. The answer to a question is assumed to lie at one particular point in the media.
4. For a given question, a subject uses the browser to select a specific segment of the recording to play. The probability of finding the correct answer to the given question in one attempt is:

$$P_1\left(answer = correct\right) = \frac{Q \cdot W}{L}$$

where W is the length of the segment played and L is the length of the meeting. The factor Q is used to model if the browser hinders or aids finding the answer. When Q is 1, the probability of finding the correct answer in one try would be that of random chance, assuming W is known. We assume Q > 1 for most browsers.

5. When subjects find the answer to the question, they move on to the next question. However, when the answer is not found, they repeatedly use the browser to find and play segments of the recording. The probability of not finding the correct answer in a given number of tries *i* is:

$$\begin{aligned} P_i(answer = incorrect) &= P_{i-1}(answer = incorrect) \cdot P_1(answer = incorrect) \\ &= P_1(answer = incorrect)^i \end{aligned}$$

assuming the segments may or may not overlap, and the probability of finding the right answer after *i* tries is:

$$P_i(answer = correct) = 1 - P_i(answer = incorrect)$$

6. When a subject cannot find the answer but wants to move on, the subject guesses the answer. The average time a subject spends on a questions is:

$$t = \frac{L}{2N}$$

where L is the length of the meeting and N is the number of questions answered. The divisor of 2 is due to the fact that BET tests are half the length of the original recording. Assuming that a subject spends time *X* between playing segments, and the segments are each of time *W*, a subject has time for *k* tries:

$$k = \frac{t}{W + X}$$

7. When this time runs out, we assume the answer is guessed. The probability of it being correct, $P_{guess}$ is given by the known likelihood of tests without any media or browser (Wellner et al., 2005). This is known

to be 56.7%. During these experiments, subjects were only given questions, which they had to answer without the use of a browser and/or meeting recording.

8. Overall, an answer may be correct from either finding it or from guessing it. The probability of finding the correct answer is thus:

$$P(answer = correct) = P_k(answer = correct) + P_{guess} \cdot P(answer = incorrect)$$

9. Using the above described model, we can predict the accuracy of each subject given the number of questions answered during the BET test and the quality factor Q. Since the accuracy and number of questions answered of each subject is known, we can determine the quality factor Q by fitting the model to the actual results of the test using least squares. The other constants, segment length *W* and time overhead between segments *X*, are also estimated using least squares.

### 5.4.2 Results

The model was first fit to the calibration condition to obtain estimates for the segment length *W*, overhead length *X* and quality parameter Q:

| Condition | W (sec.) | X (sec.) | Q |
|---|---|---|---|
| Calibration | 25.05 | 4.76 | 8.0 |

**Table 8. Parameters estimated from calibration condition**

Then the parameters *W* and *X* are fixed while determining the quality factor Q for the other test conditions:

| Condition | Quality factor Q |
|---|---|
| Base | 14.5 |
| Speedup | 23.1 |
| Overlap | 10.9 |

**Table 9. Speed/accuracy trade-off model quality factors**

When plotting the model-predicted accuracy of each user against their average answering speed, the speed/accuracy trade-off curve for each test condition becomes visible (Figure 27).

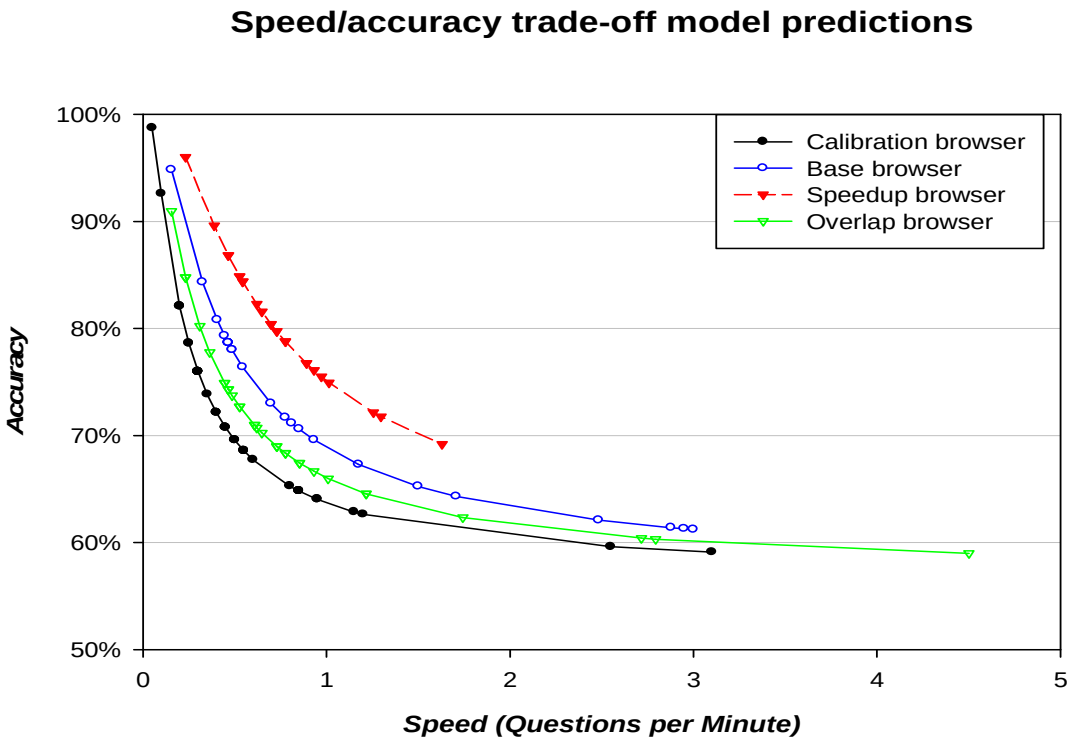


**Figure 27. Speed/accuracy trade-off curves**

Figure 27 shows that the accuracy in the speedup condition degrades slower than the accuracy in the other test conditions and the mean accuracy is higher. The quality factors from Table 9 give a more intuitive ranking of the quality of the browsers than the mean values of accuracy and speed (Table 4 and Table 5). The quality scores show that the speedup browser scores better than the baseline browser, and the overlap browser scores worse.

### 5.4.3 Model validity

The data collected for the BET has such high variability that it does not fit the trade-off model better than simply using mean accuracy and speed scores.

| Model | Average RMS error |
|---|---|
| Mean | 14.02% |
| Speed/accuracy trade-off model | 15.84% |

**Table 10. RMS error for mean model and trade-off model**

The RMS error for actual accuracy compared to the speed/accuracy trade-off model predicted accuracy is slightly higher than the RMS error for actual accuracy compared to the mean.

## 5.5 Questionnaire

After the tests with the browsers were conducted, all users were asked to fill in a questionnaire. Users could express their agreement with the statements from the questionnaire on a 5 point Likert scale. The subjects were able to explain their answer for most questions.

### 5.5.1 Base condition

The following table summarizes the questionnaire results for the base condition. The letters D, MD, N, MA and A are the 5 choices from the Likert scale and stand for Disagree, Mild Disagree, Neutral, Mild Agree and Agree. Under these choices the number of users who choose that choice is reported. Table 11 shows the distribution of the answers and their mode.

| Question | D | MD | N | MA | A | Mode |
|---|---|---|---|---|---|---|
| This meeting browser helped me answer questions correctly and quickly | 1 | 1 | 4 | 1 | 4 | Neutral / Agree |
| This meeting browser helped me get the gist of a meeting quickly | 0 | 0 | 0 | 3 | 8 | Agree |
| This meeting browser helped me find specific details about the meeting | 1 | 0 | 3 | 4 | 3 | Mild Agree |
| I enjoyed using this software | 1 | 1 | 5 | 0 | 4 | Neutral |
| The tutorial was helpful | 0 | 1 | 2 | 3 | 5 | Agree |
| The coloured speaker segmentation bars were helpful | 0 | 0 | 0 | 0 | 11 | Agree |
| The slides were helpful | 0 | 1 | 0 | 1 | 9 | Agree |
| The timeline was helpful | 0 | 0 | 4 | 4 | 3 | Neutral / Mild Agree |
| The names and photos with coloured frames were helpful | 0 | 1 | 3 | 3 | 5 | Agree |

**Table 11. Questionnaire Likert scale results for base conditon**

Subjects felt no need for software like this and found it both boring and amusing to see how much time is wasted during meetings. One subject suggested that being able to speed-up the speech could be helpful.

### 5.5.2 Speedup condition

The following table summarizes the questionnaire results for the Speedup condition.

| Question | D | MD | N | MA | A | Mode |
|---|---|---|---|---|---|---|
| This meeting browser helped me answer questions correctly and quickly | 0 | 3 | 3 | 3 | 3 | Mild Disagree / Neutral / Mild Agree / Agree |
| This meeting browser helped me get the gist of a meeting quickly | 0 | 1 | 0 | 5 | 6 | Agree |
| This meeting browser helped me find specific details about the meeting | 2 | 1 | 4 | 3 | 2 | Neutral |
| I enjoyed using this software | 0 | 2 | 3 | 6 | 1 | Mild Agree |
| The tutorial was helpful | 0 | 0 | 4 | 3 | 5 | Agree |
| I could understand what people were saying at high playback speed | 0 | 2 | 2 | 5 | 3 | Mild Agree |
| The coloured speaker segmentation bars were helpful | 0 | 0 | 4 | 3 | 5 | Agree |
| The speed control was helpful | 0 | 0 | 0 | 3 | 9 | Agree |
| The slides were helpful | 1 | 1 | 0 | 5 | 5 | Agree |
| The names and photos with coloured frames were helpful | 0 | 1 | 3 | 3 | 5 | Agree |
| The timeline was helpful | 0 | 1 | 3 | 2 | 6 | Agree |

**Table 12. Questionnaire Likert scale results for speedup condition**

Most people enjoyed using this browser and were still able to understand speech at a high playback rate. The maximum speed subjects still thought was intelligible was on average 2.34 ($\sigma$ = 0.26). We did not measure the amount of time subjects were actually playing back the recording at a certain speed; subjects might have overestimated their own capabilities.

### 5.5.3 Overlap condition

The following table summarizes the questionnaire results for the Overlap condition.

| Question | D | MD | N | MA | A | Mode |
|---|---|---|---|---|---|---|
| This meeting browser helped me answer questions correctly and quickly | 2 | 3 | 6 | 4 | 2 | Neutral |
| This meeting browser helped me get the gist of a meeting quickly | 1 | 0 | 0 | 6 | 10 | Agree |
| This meeting browser helped me find specific details about the meeting | 2 | 1 | 4 | 6 | 4 | Mild Agree |
| I enjoyed using this software | 2 | 1 | 8 | 3 | 3 | Neutral |
| The tutorial was helpful | 0 | 0 | 3 | 7 | 7 | Agree |
| I only listened to one side at a time and ignored the other side | 4 | 3 | 3 | 1 | 6 | Agree |
| I completely followed both sides at the same time | 8 | 7 | 1 | 1 | 0 | Disagree |
| The coloured speaker segmentation bars were helpful | 0 | 0 | 0 | 1 | 14 | Agree |
| The slides were helpful | 0 | 0 | 2 | 4 | 9 | Agree |
| The names and videos with coloured frames were helpful | 0 | 2 | 0 | 3 | 9 | Agree |
| The timeline was helpful | 1 | 2 | 3 | 4 | 5 | Agree |
| The volume balance control was helpful | 0 | 0 | 2 | 4 | 8 | Agree |
| The lack of independent control for each side was frustrating | 3 | 0 | 6 | 0 | 6 | Neutral / Agree |

**Table 13. Questionnaire Likert scale results for overlap condition**

In Table 13 we can see that many subjects only listened to one side of the meeting and ignored the other side. However, we can infer that many subjects tried to listen to both sides at once. The subjects strongly agree that they were not able to follow both sides at the same time.
Even though the Overlap browser had the lowest quality score (Table 9) and subjects complained that the simultaneous speech was tiring and took a lot of concentration, some subjects still thought they could pick up more information this way and that the simultaneous streams were "surprisingly easy to separate".
Subjects found the simultaneous speech helpful when they were searching for a specific piece of speech. They tended to concentrate on one side, but found it useful to have the other side playing so they could sometimes still pick out specific words or topics.

### 5.5.4 General remarks

When asked if the browser was helpful, useful to find specific details or get the gist of the meeting, subjects most often mentioned that the speech segmentation and slides were helpful. Subjects often mentioned that being able to browse through an individual speaker using the speech segmentation was very useful. Some subjects suggested incorporating keyword searching.

# 6 Conclusion

The Speedup browser scores better than the Baseline browser and the Overlap browser scores worse. Based on this ranking we can conclude that reducing the playback time of a meeting recording with the use of time-compression helps users find information faster compared to using speech played back at a normal rate. However, we do not have high confidence in the ranking due to the variability of the collected BET data.

Overlapping meeting parts using binaural audio, as was done with the Overlap Browser, showed to be to difficult to work with for most users. We assume this is due to an increase in cognitive load. Subjects needed to concentrate a lot when using the overlap browser and found this browser tiring, which is in agreement with previous studies on binaural audio. The use of binaural audio in meeting browsers to simultaneously play back parts of a meeting does not seem usable. However, binaural audio can still be used in meeting browsers to enhance spatial segregation between meeting participants by giving each meeting participants a different spatial location. This enhances the listening experience and can potentially improve detection of back channels or other overlapping speech.

Time-compression seems to be a promising technique to reduce listening time, as its use is easy to understand by users. Future research on using time-compressed speech in interfaces might include the use of a brain-computer interface to detect if a user can still understand speech at a certain compression rate. Current research suggest that it could be possible to detect high level cognitive and emotional states such as confusion and attention (Ferrez & Millan, 2005). Being able to detect whether a user is unable to understand the time-compressed speech at a certain compression level, will make it possible to maximize the amount of information a user can process while listening to time-compressed speech.

The additional browser components such as slides and speech segmentation were added to increase the subjects answering speed. These additional browsing components can also be an extra source of variance as users might use different browsing behavior. By limiting the number of navigational options, e.g. only use a timeline to browse through the meeting, users have to rely more on the audio to improve their performance. This could also make users focus more on the use of the audio component instead of on the different browser controls. However, we wanted to test the use of time-compressed speech and binaural audio in a fully interactive meeting browser in order to validate the use of the BET.

Currently, approximately 1 question per minute is collected from the subjects, some of these being guesses. Because there is so little data collected, a small difference can have a large impact. Variability can be reduced by collecting more information from the subjects. One way in which this can be done, is by collecting more information per question answered, e.g. for each answer asking why they chose that answer, or letting subjects indicate their level of confidence in their answers. Another way to collect more information is to stimulate subjects to answer more questions.

The collected BET data currently shows a large amount of variability at the left end of the speed/accuracy curve (Figure 24, Figure 25, and Figure 26) because that is where most information is collected. The data collected for the speedup condition does not seem to follow any curve, most likely because none of the subjects tried to answer questions faster than 1.6 questions per minute.
By forcing people to answer after a fixed time-period and repeating this for different time-frames, people are forced to give up their own browsing behavior. By limiting subjects in the amount of time they get to answer questions, subjects that usually take more time to answer questions are now forced to make more use of a browser's capabilities. We assume that subjects will follow the speed/accuracy trade-off model more closely.

# Bibliography


Algazi, V. R., Duda, R. O., Thompson, D. M., & Avendano, C. (2001). The CIPIC HRTF database. *Applications of Signal Processing to Audio and Acoustics, 2001 IEEE Workshop on the*, 99-102.

Arons, B. (1992a). A Review of the Cocktail Party Effect. *Journal of the American Voice I/O Society, 12*(7), 35–50.

Arons, B. (1992b). Techniques, Perception, and Applications of Time-Compressed Speech. *Proceedings of 1992 Conference*, 169–177.

Arons, B. (1994). *Interactively Skimming Recorded Speech.* Massachusetts Institute of Technology.

Arons, B. (1997). SpeechSkimmer: a system for interactively skimming recorded speech. *ACM Transactions on Computer-Human Interaction (TOCHI), 4*(1), 3-38.

Boersma, & Paul. (2005). Praat: doing phonethics by computer (Version 4.4): http://www.praat.org.

Brown, C. P., & Duda, R. O. (1997). An efficient HRTF model for 3-D sound. *Applications of Signal Processing to Audio and Acoustics, 1997. 1997 IEEE ASSP Workshop on*, 4.

Carletta, J., Ashby, S., Bourban, S., Flynn, M., Guillemot, M., Hain, T., et al. (2005). The AMI meeting corpus: A pre-announcement. *Proc. Workshop on Machine Learning for Multimodal Interaction (MLMI), Edinburgh, Jul*.

Covell, M., Withgott, M., & Slaney, M. (1998a). *Mach1 for Nonuniform Time-Scale Modification of Speech: Theory, Technique and Comparisons.* Paper presented at the Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing.

Covell, M., Withgott, M., & Slaney, M. (1998b). MACH1: nonuniform time-scale modification of speech. *Acoustics, Speech, and Signal Processing, 1998. ICASSP'98. Proceedings of the 1998 IEEE International Conference on, 1*.

Duda, R. O., Avendano, C., & Algazi, V. R. (1999). Adaptable ellipsoidal head model for the interaural time difference. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing- Proceedings, 2*, 965-968.

Eckel, G. (2001). Immersive audio-augmented environments-the LISTEN project. *Proc. of the 5th International Conference on Information Visualization (IV2001)*, 571.

Ellis, D. P. W. (2002). *A Phase Vocoder in Matlab*.

Ellis, D. P. W. (2006). SOLAFS in Matlab: http://www.ee.columbia.edu/~dpwe/resources/matlab/solafs-matlab.html.

Ferrez, P., & Millan, J. R. (2005). You Are Wrong!—Automatic Detection of Interaction Errors from Brain Waves. *Proc. 19th Int. Joint Conf. Artificial Intelligence*.

Gardner, W. G., & Martin, K. D. (1995). HRTF measurements of a KEMAR. *The Journal of the Acoustical Society of America, 97*, 3907.

Hawley, M. L., Litovsky, R. Y., & Culling, J. F. (2004). The benefit of binaural hearing in a cocktail party: Effect of location and type of interferer. *The Journal of the Acoustical Society of America, 115*, 833.

He, L., & Gupta, A. (2001). Exploring benefits of non-linear time compression. *Proceedings of the ninth ACM international conference on Multimedia*, 382-391.

Janse, E. (2003). Production and perception of fast speech. *Utrecht, The Netherlands: Landelijke Onderzoekschool Taalwetenshap*.
Jurafsky, D., & Martin, J. H. (2000). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*: MIT Press.
Kobayashi, M., & Schmandt, C. (1997). Dynamic Soundscape: mapping time to space for audio browsing. *Proceedings of the SIGCHI conference on Human factors in computing systems*, 194-201.
Miller, G. A., & Licklider, J. C. R. (1950). The Intelligibility of Interrupted Speech. *The Journal of the Acoustical Society of America, 22*, 167.
Mullins, A. T. (1996). *Audiostreamer: Leveraging The Cocktail Party Effect for Efficient Listening.* Massachusetts Institute of Technology.
O'Shaughnessy, D. (1992). Recognition of Hesitations in Spontaneous Speech. *Proc. ICASSP*.
Omoigui, N., He, L., Gupta, A., Grudin, J., & Sanocki, E. (1999). Time-compression: systems concerns, usage, and benefits. *Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit*, 136-143.
Ranjan, A. (2005). *Browsing Archived Meeting Audio and Time-Synchronized Data.* University of Toronto.
Roucos, S., & Wilgus, A. (1985). High quality time-scale modification for speech. *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'85., 10*.
Schmandt, C., & Mullins, A. (1995). AudioStreamer: exploiting simultaneity for listening. *Conference on Human Factors in Computing Systems*, 218-219.
Stifelman, L. J. (1994). The Cocktail Party Effect in Auditory Interfaces: A Study of Simultaneous Presentation: MIT Media Lab Technical Report, September.
Verhelst, W. (2000). Overlap-add methods for time-scaling of speech. *Speech Communication, 30*(4), 207-221.
Verhelst, W., & Roelands, M. (1993). An overlap-add technique based on waveform similarity (WSOLA) forhigh quality time-scale modification of speech. *Acoustics, Speech, and Signal Processing, 1993. ICASSP-93., 1993 IEEE International Conference on, 2*.
Viste, H., & Evangelista, G. (2004, 2004). *Binaural Source Localization.* Paper presented at the Proc. 7th International Conference on Digital Audio Effects (DAFx-04), invited paper, Naples, Italy.
Wellner, P., Flynn, M., Tucker, S., & Whittaker, S. (2005). A meeting browser evaluation test. *Conference on Human Factors in Computing Systems*, 2021-2024.
Zölzer, U., Amatriain, X., Arfib, D., Evangelista, G., Keiler, F., Loscos, A., et al. (2002). *DAFX-Digital Audio Effects*: John Wiley and Sons.

# Appendix

## Appendix A - User test initial overlap browser

**User 1**

The first user was given a few minutes to play around with the interface to explore each interface component's function. The user was allowed to ask questions about the interface. When the user felt ready, the observations were presented and the test could begin.

*Observations before the test*

The user needed a lot of explanation about the interface, especially about the "Huh?" button. It took the user a few looks through the browser before it became clear that the meeting was cut in half and each half was presented on a different side. It also was not clear to the user that most GUI components were clickable and altered the media-time.

*Observations during the test*

When the test started, the user initially went through all the slides to see if the questions could be answered from the slides. The "Huh?" button was used a few times when there were a lot of people talking or the same person was talking on both sides.

*Comments by user after the test*

The user asked why the mute time for the "Huh?" buttons was only 5 seconds, the user wanted it to be customizable (i.e. be able to control how long one side is muted). The user thought that one side sometimes was too loud and that the high frequencies were overruling the low frequencies. It was hard to focus on a male speaker when a female speaker was talking on the other side, as the women sounded "much brighter". The user suggested a volume control for both sides so when, for exaple, a man is speaking, the user can adjust the volume as necessary, or to completely turn down one side. The user felt like "watching 2 TVs at the same time".

The user said he was only able to focus on one side but still had the impression to catch the gist of the other side.

**User 2**

Because self exploration of the browser left many questions open for user 1, user 2 received a 2 page manual explaining each individual GUI component and a short explanation of the goal of the browser. After reading the manual, the user had some time to get familiar with the browser by exploring it.

*Observations before the test*

The user expressed concerns over not having adequate English language skills.

*Observations during the test*

The user still needed a lot of explanation on how certain things really work. His expectations were different from how it really worked.

This user made much less use of the slides, maybe because he didn't understand how to operate them. He thought that the time would change by using the

scrollbar of the slides (even thought the manual describes that clicking on a slide alters the media time). The user still needed a lot of explanation, especially on the timelines. He tried to scroll the timeline that contains the speech segmentations; this did not work because the cursor is being tracked in this timeline, i.e. the viewport of this timeline moves along with the time cursor. The user wanted to be able to browse the timeline like a website, meaning the viewport should be adjustable by using the scrollbars of the speaker segmentation.
The user said he could only focus on one side of the meeting at a time. The female speakers were much more dominant because of their high pitch voice.

*Comments by user after the test*
The user wanted to be able to only browse a specific user because sometimes the others made it more confusing while he was really searching for an answer of this specific user. He also suggested splitting the meeting in more parts than just two (according to the meeting structure like: opening, presentations, conclusions). This way he could listen to both the opening and the conclusions at once. He also wanted to be able to have complete control over both sides; be able to listen to the opening and also to the conclusions. He thought this way he could answer more questions because sometimes you know one answer is in the beginning and another is at the end of the meeting, with the current browser he felt limited to focusing on one side and not taking advantage of the other side because both times are linked.
He said it felt unnatural to make use of the timeline that shows the complete time view when there already is another timeline for that part.