User Benefits of Non-Linear Time Compression

Liwei He & Anoop Gupta

September 21st, 2000

Technical Report
MSR-TR-2000-96

Microsoft Research
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052

User Benefits of Non-Linear Time Compression

Liwei He & Anoop Gupta
Microsoft Research
One Microsoft Way, Redmond, WA 98052
+1 (425) 703-6259
{lhe, anoop}@microsoft.com

ABSTRACT

In comparison to text, audio-video content is much more challenging to browse. *Time-compression* has been suggested as a key technology that can support browsing – time compression speeds-up the playback of audio-video content without causing the pitch to change. Simple forms of time-compression are starting to appear in commercial streaming-media products from Microsoft and Real Networks.

In this paper we explore the potential benefits of more recent and advanced types of time compression, called *non-linear time compression*. The most advanced of these algorithms exploit fine-grain structure of human speech (e.g., phonemes) to differentially speed-up segments of speech, so that the overall speed-up can be higher. In this paper we explore what are the actual gains achieved by end-users from these advanced algorithms, and whether the gains are worth the additional systems complexity. Our results indicate that the gains today are actually quite small and may not be worth the additional complexity.

Keywords: Time compression, Digital library, Multimedia browsing, User evaluation

1 INTRODUCTION

Digital multimedia information on the Internet is growing at an increasing rate – corporations are posting their training materials and talks online [13], universities are putting up their videotaped courses online [23], news organizations are making newscasts available online. While the network bandwidth is somewhat of a bottleneck today, this is rapidly getting addressed with the new broadband infrastructure being put in place. The eventual bottleneck really is the limited human time.

With so much content available, it is highly desirable to have technologies that let people browse audio-video quickly. The impact of even a 10% increase in browsing speed can be large, if one considers the vast number of people that will end-up saving time. Just as a person may read text at different rates depending on the situation (e.g., when reading a deep technical article vs. skimming a magazine) or different people may have different reading rates, we will like to provide people the ability to speed-up or slow-down audio-video content based on their preferences.

In this paper we focus on technologies that allow such speed-up and slow-down of speech content. While the video-portion of audio-video content is also important, it is easier to handle than speech and is considered elsewhere [24]. Also, we focus on informational content with speech (e.g., talks, lectures, and news) rather than entertainment content (e.g., music videos, soap operas), as previous work has shown that people are less likely to speed-up the latter [16].

The core technology that supports such speed-up or slow-down of speech is called *time-compression* [6, 9, 12, 14, 20]. Simple forms of time-compression have been used before in hardware device contexts [1] and telephone voicemail systems [17]. Within the last few months, we have also seen basic support for time-compression in major streaming-media products from Microsoft and Real Networks [5, 18].

Most systems today use *linear time-compression*, where the speech content is uniformly time compressed, e.g., every 100ms chunk of speech is shortened to 75ms. Using linear time compression, previous user studies show [11, 16, 19] that participants achieve steady-state speed-up factors of ~1.4. With that speed-up, users can save more than 15 minutes on a one-hour lecture.

In this paper, we explore how much additional benefit can be achieved from *non-linear time-compression* techniques. We consider two such algorithms. The first, simpler algorithm combines pause-removal with linear time compression. It first detects pauses (silence intervals) in the speech, then shortens or removes the pauses. Such a procedure can remove 10-25% from normal speech [8]. It then performs linear time compression on the remaining speech.

The second non-linear algorithm we consider is much more sophisticated. It is based on the recently proposed Mach1 algorithm [3], the best such algorithm known to us. It tries to mimic the compression strategies that people use when they talk fast in natural settings, and it tries to adapt the compression rate at a fine granularity based on low-level features (e.g., phonemes) of human speech.

As we will elaborate later, the non-linear algorithms, while offering the potential for higher speed-ups, require significantly more compute (CPU) cycles, cause increased complexity in client-server systems for streaming media, and may result in a jerky video portion. So the core questions we address in this paper are the following:

1. What are the additional benefits of the two non-linear algorithms over the simple linear time-compression algorithm implemented in products today? While inventors of Mach1 present some user-study data about benefits, their results correspond to very high

speed-up factors (2.6 to 4.2 fold speedup), where only a subset of speech is understood. Most people will not listen to speech at such fast rates. We are interested in understanding people's preference at more comfortable and sustainable speed-up rates. Only if the difference at sustainable speed is large will it be worthwhile to implement these algorithms in products.

2. How much better is the more sophisticated algorithm over the simpler non-linear algorithm? The magnitude of differences will again guide our implementation strategy in products.

At a high level, our results show that for speed-up factors most likely to be used by people, the benefits of the more sophisticated non-linear time compression algorithms are quite small. Consequently, given the substantial complexity associated with these algorithms, we may not see them adopted in the near future.

The paper is organized as follows: Section 2 reviews various time-compression algorithms evaluated in this paper and associated systems implications. Section 3 presents our user-study goals, Section 4 the experimental method, and Section 5 the results of the study. We discuss results and present related work in Section 6 and conclude in Section 7.

2 TIME-COMPRESSION ALGORITHMS USED AND SYSTEMS IMPLICATIONS

In this section, we briefly discuss the three classes of algorithms we consider in this paper, and systems implications for incorporating them in client-server delivery systems.

2.1 Linear Time Compression (Linear)

In this class of algorithms, time-compression is applied consistently across the entire audio stream with a given speed-up rate, without regard to the audio information contained therein. The most basic technique for achieving time-compressed speech involves taking short fixed-length speech segments (e.g., 100ms), and discarding portions of these segments (e.g., dropping 33ms segment to get 1.5-fold compression), and abutting the retained segments [6].

Discarding segments and abutting the remnants, however, produces discontinuities at the interval boundaries and produces audible clicks and other forms of signal distortion. To improve the quality of the output signal, a windowing function or smoothing filter–such as a cross-fade–can be applied at the junctions of the abutted segments [20]. A technique called *Overlap Add (OLA)* yields good quality (Figure 1). Further improvements to OLA are made in *Synchronized OLA (SOLA)* [21] and *Pitch-Synchronized OLA* [10].

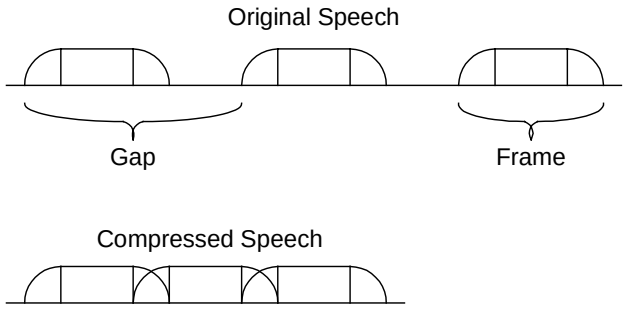

Figure 1: An illustration of Overlap Add algorithm.

The technique used in this study is SOLA, first described by Roucos and Wilgus [21]. It consists of shifting the beginning of a new speech segment over the end of the preceding segment to find the point of highest waveform similarity. This is usually accomplished by a cross-correlation computation. Once this point is found, the frames are overlapped and averaged together, as in OLA. SOLA provides a locally optimal match between successive frames and mitigates the reverberations sometimes introduced by OLA. The SOLA algorithm is labeled "Linear" in our user studies.

2.2 Pause Removal plus Linear Time Compression (PR-Lin)

Non-linear time compression is an improvement on linear compression: the content of the audio stream is analyzed, and compression rates may vary from one point in time to another. Typically, non-linear time compression involves compressing redundancies, such as pauses or elongated vowels, more aggressively. The PR-Lin algorithm we use here first detects pauses using the algorithm described below. It leaves pauses below 150ms untouched, and shortens longer pauses to 150ms. It then applies linear time-compression as described in the previous subsection.

Pause detection algorithms have been published extensively. A variety of measures can be used for detecting pauses even under noisy conditions [2]. Our algorithm uses "Energy" and "Zero crossing rate (ZCR)" features. In order to adjust changes in the background noise level, a dynamic energy threshold is used. We use a fixed ZCR threshold of 0.4 in this study.

If the energy of a frame is below the dynamic threshold and its ZCR is under the fixed threshold, the frame is categorized as a potential-pause frame, otherwise it is labeled as a speech frame. Contiguous potential-pause frames are marked as real-pause frames when they exceed 150ms.

Pause removal typically shortens the speech by 10-25% before linear time-compression is applied.

2.3 Adaptive Time Compression (Adapt)

A variety of more sophisticated algorithms have been proposed for non-linear adaptive time compression. For example, Lee and Kim [15] try to preserve the phoneme transitions in the compressed audio to improve understandability. Audio spectrum is computed first for

audio frames of 10ms. If the magnitude of the spectrum difference between two successive frames is above a threshold, they are considered as a phoneme transition and not compressed.

Mach1 [3] makes further improvements and tries to mimic the compression that takes place when people talk fast in natural settings. These strategies come from the linguistic studies of natural speech [25, 26] and are listed as follows:

- Pauses and silences are compressed the most
- Stressed vowels are compressed the least
- Schwas and other unstressed vowels are compressed by an intermediate amount
- Consonants are compressed based on the stress level of the neighboring vowels
- On average, consonants are compressed more than vowels

Mach1 estimates continuous-valued measures of local emphasis and relative speaking rate. Together, these two sequences estimate the audio tension: the degree to which the local speech segments resist changes in rate. High-tension regions are compressed less and low-tension regions are compressed more aggressively. Based on the audio tension, the local target compression rates are computed and used to drive a standard time-scale modification algorithm, such as SOLA.

Since Mach1 is, to our knowledge, the best adaptive time compression technique, the algorithm used in our adaptive time compression condition is based on it. The Mach1 executable was not available to us, so we could not use it directly. Furthermore, the original Mach1 algorithm cannot guarantee a specified speedup rate (it is an "open loop" algorithm). In order to compare audio clips compressed using different algorithms, we required precise speedup as specified by the user. We made modifications to the algorithm so the achieved speedup rate is always as the same as specified.

We wanted to ensure that our revised algorithm (Adapt) was comparable in quality to Mach1 algorithm. A preference study was run to compare our adaptive algorithm against the original Mach1 algorithm. Without indication of the sources, 12 colleagues were asked to compare 3 time-compressed speech files published on Mach1's web site and the same source files compressed using our implementation. Out of the total 36 comparisons, our algorithm was preferred 9 times, Mach1 was preferred 12 times, and they were found to be equal 15 times. A one-sample Chi-square test was conducted to assess whether the participants preferred the results from our algorithm, the published Mach1 results, or had no preference. The results of the test were non significant: Chi^2 (2,N=36) = 1.5, p=0.472, indicating our technique is comparable to the Mach1 algorithm.

2.4 Systems Implications of Algorithms

In deciding between these three algorithms for inclusion in products, there are two considerations: 1) what are the relative benefits (e.g. speed-up rates) achievable, and 2) what are the costs (e.g. implementation challenges). We explore the former in the User Study section. Here we briefly discuss the latter.

The first issue is computational complexity or CPU requirements. The first two algorithms, Linear and PR-Lin, are easily executed in real-time on any Pentium-class machine using only a small fraction of the CPU. The Adapt algorithm, in contrast, has 10+ times higher CPU requirements, although it can be executed in real-time on modern desktop CPUs.

The second issue is complexity of client-server implementations. We assume people will like the time-compression feature to be available with streaming-media clients where they can just turn a virtual knob to adjust speed-up. While there are numerous issues [19], a key issue has to do with buffer management and flow-control between the client and server. The Linear algorithm has the simplest requirements, where the server simply needs to speed-up its delivery at the same rate at which time-compression is requested by client. The nonlinear algorithms (both PR-Lin and Adapt) have much more complex requirements due to the uneven rate of data consumption at the client – e.g., if a 2 second pause is removed, then associated data is instantaneously consumed by the client and the server will have to compensate.

The third issue is audio-video synchronization quality. (This issue is obviously not present when considering speech-only content.) With the Linear algorithm, the rendering of video frames is speeded up at the same rate as the speed-up for speech. While everything happens at higher speed, the video remains smooth and perfect lip-synchronization between audio and video can be maintained. This task is much more difficult with non-linear algorithms (PR-Lin and Adapt). As an example, consider removal of a 2-second pause from the audio track. Option-1 is to also remove the video frames corresponding to those 2 seconds. In this case the video will appear jerky to the end-user, although we will retain lip synchronization between audio and video for subsequent speech. Option-2 is to make the video transition smoother by keeping some of the video frames from that 2-second interval and removing some later ones, but now we will loose the lip synchronization for subsequent speech. There is no perfect solution.

The bottom line is that non-linear algorithms add significant complexity to the implementer's task. We would like to know if there are significant user benefits.

3 USER STUDY GOALS

There are multiple dimensions along which we will like to understand users' reactions to the three algorithms presented above. We used the following four metrics:

1. **Highest intelligible speed**. What is the highest speed-up factor at which the user still understands the majority of the content? This metric tells us which algorithms perform best when the end-user is pushing the limits of time-compression technology for short segments of speech.
2. **Comprehension**. Given the same fixed speed-up factor for all algorithms, what is a user's relative comprehension? This metric is indicative of the relative quality of speech produced by algorithms. When observed for multiple speed-up factors, it also indicates when we are driving users beyond sustainable speed.
3. **Subjective preference**. When given the same audio clip compressed using two different techniques at the same speed-up factor, which one does a user prefer? This metric is directly indicative of the relative quality of speech produced by the algorithms. Since people are very sensitive to subtle distortions that are not computationally understood, this is the only way to understand quality issues.
4. **Sustainable speed**. What is the speed-up factor that end-users will settle on when listening to long pieces of content (e.g., a lecture), still assuming some time pressure? We believe this metric is the most indicative of benefits that will accrue to users in natural settings.

4 EXPERIMENTAL METHOD

24 people participated our study in exchange for a gratuity. They came with a variety of background from professionals in local firms to retirees to homemakers. All of them had some computer experience and some used computers on a daily basis. The subjects were invited to our usability lab to take the test.

The listener study was Web based. All the instructions were presented to the subjects via web pages. The study consisted of four tasks corresponding to the four goals outlined in the previous section (see Table 1).

Highest Intelligible Speed Task: The subjects were given 3 clips time-compressed by Linear, PR-Lin, and Adapt and were asked to find the fastest speed at which the audio was still intelligible. For each algorithm, short segments of a clip were presented to the subjects in sequence. The subjects used five speed-control buttons (much-faster, faster, same, slower, much-slower) to control the speed at which the next segment was played. The speed control buttons increased or decreased the speed by a discrete level of either 0.1 or 0.3. The subjects clicked the Done button when they found their highest intelligible speed for the clip. We asked the subjects to choose their own definition of what intelligible meant, e.g. understanding 90-95% of words in the audio, as long as they were consistent with their definition throughout the task.

Table 1: Information about tasks and test materials.

	Task	Audio source	WPM	Approx. Length
1	Highest intelligible speed	3 technical talks	99-169	In 10 sec segments
2	Comprehension	6 conversations from Kaplan's TOEFL program	185-204	28-50 sec
3	Preference	3 clips from an ACM'97 talk by Brenda Laurel	178	30 sec
4	Sustainable speed	3 clips from "Don't know much about geography"	169	8 min

The audio clips used in this task were from 3 talks. The natural speech speed, as measured by words per minute (WPM), had a fairly wide range among the chosen clips (see Table 1). The WPM of the fastest speaker is 71% greater than the slowest speaker. However, the experiments were all counterbalanced among subjects, as we will discuss later.

Comprehension Task: We gave each subject 6 clips of conversations time-compressed by the three algorithms at 1.5x and at 2.5x. The subjects listened to each conversation once (repeats were not allowed) and then answered four multiple-choice questions about the conversation. The conversation clips were taken from the audio CDs from Kaplan's TOEFL (Test of English as Foreign Language) study program [22]. The subjects were encouraged to guess if they were not sure of the answer. We note that the two chosen speed-up factors, 1.5x and 2.5x, represent points on each side of the sustained speed-up factor for users.

Subjective Preference Task: The subjects were instructed to compare 6 pairs of clips time-compressed by the three algorithms at 1.5x and at 2.5x and indicate their preference on a three-point scale: prefer clip 1, no preference, prefer clip 2. The audio clips in this task were captured live from an ACM'97 talk given by Brenda Laurel.

Sustainable Speed Task: We gave the subjects 3 clips time-compressed by the three algorithms and asked them to imagine that they were in a hurry, but still wanted to listen to the clips. They adjusted the speed control buttons to find a maximum speed for each clip that was

sustainable for the duration of the clips, which were about 8 minutes uncompressed. They were required to write 4-5 sentences to summarize what they just heard upon the completion each clip, though the textual summaries were used only to motivate the subjects to pay more attention but not as part of the actual measurement. The audio clips in this task were taken from the audio CD book "Don't Know Much About Geography" [4].

Within each task, we used a repeated measures design in a 3x3 Latin Square configuration to counterbalance against ordering effects, i.e., the order in which users experienced the time compression methods. The task list for a typical subject is listed in Table 2.

Table 2: The task list for a typical subject.

	Task	Condition	TC factor
1	Highest intelligible speed	Linear	(User adjusted)
		PR-Lin	
		Adapt	
2	Comprehension	Linear	1.5
		PR-Lin	
		Adapt	
		Linear	2.5
		PR-Lin	
		Adapt	
3	Preference	Linear vs. Adapt	1.5
		PR-Lin vs. Linear	
		Adapt vs. PR-Lin	
		Linear vs. Adapt	2.5
		PR-Lin vs. Linear	
		Adapt vs. PR-Lin	
4	Sustainable speed	Linear	(User adjusted)
		PR-Lin	
		Adapt	

5 LISTENER STUDY RESULTS

As stated in the Introduction section, for each of the metrics we would first like to understand the benefits of the non-linear algorithms (PR-Lin and Adapt together) over the simpler Linear algorithm. Second, if the non-linear algorithms are indeed better, we would like to differentiate between the simpler PR-Lin and the more complex Adapt algorithm.

5.1 Highest Intelligible Speed

The first task measures the highest speed at which the clips are still intelligible. As one would expect, we see that the non-linear algorithms do significantly better than Linear – combined average speed-up of 2.05 vs. 1.76 (see Tables 3 and 4). This is also true when listening speed is measured as words per minute (WPM).

Comparing the two non-linear algorithms, we find that PR-Lin does significantly better than Adapt when using speed-up factor as metric, but not when using WPM as a metric (see Table 4). The result is somewhat contradictory to our expectations, as we would have expected the more sophisticated Adapt algorithm to do better. On the other hand, one possible explanation may be as follows. PR-Lin is more aggressive as it totally eliminates pauses, while Adapt is gentler when shortening pauses and as a result it has to compress the audible speech more to reach the same speed-up as PR-Lin.

Table 3: Highest intelligible speed task. WPM numbers converted from raw speed are also listed. The standard deviations are in the parenthesis.

Condition	Speed (StDev)	WPM (StDev)
Linear	1.76 (0.29)	246 (64)
PR-Lin	2.15 (0.45)	296 (67)
Adapt	1.94 (0.36)	271 (75)
Average	1.95 (0.40)	271 (71)

Table 4: The results of the one-way within-subject ANOVA contrast test for highest intelligible speed task.

Contrast test	F	P
Linear vs. (Adapt & PR-Lin) in speed	44.910	.000
Adapt vs. PR-Lin in speed	8.137	.009
Linear vs. (Adapt & PR-Lin) in WPM	5.362	.030
Adapt vs. PR-Lin in WPM	1.885	.183

5.2 Comprehension Task

In this task, listener comprehension was tested under different algorithms at the speed-up factors of 1.5x and 2.5x. We expected Adapt to do best, followed by PR-Lin and Linear, and the comprehension differences to increase at the higher speed-up factor. Note that 1.5x and 2.5x represent points on the two sides of the highest intelligible speed-up factor for users.

The quiz scores from the comprehension task are listed in Table 5. At 1.5x speed-up, the average score of Linear actually came out on top, although there is no significant difference between Linear and the other two conditions (see Table 6). In essence, the data simply say that at 1.5x the content is well understood across all conditions.

At 2.5x speed-up, we see that the two non-linear algorithms do significantly better than Linear (see Table 6, row 3). This is not very surprising, since the non-linear algorithms need to compress the audible portions of speech much less than the Linear algorithm (since the pauses are compressed much higher than target rate by PR-Lin and Adapt).

Comparing PR-Lin and Adapt at 2.5x, there is no significant difference at $p < .05$ level. There does seem to be a trend in favor of Adapt though, given that $p = 0.083$

(Table 6, row 4). We reflect on this trend in the discussion section.

Table 5: Quiz score results from the comprehension task.

Condition	1.5x (%)	2.5x (%)	Overall (%)
Linear	84	49	67
PR-Lin	78	61	70
Adapt	82	74	78
Average	82	61	72

Table 6: The results of the one-way within-subject ANOVA contrast test for the comprehension task.

Contrast test	F	P
Linear vs. (Adapt & PR-Lin) at 1.5x	.754	.394
Adapt vs. PR-Lin at 1.5x	.324	.575
Linear vs. (Adapt & PR-Lin) at 2.5x	8.507	.008
Adapt vs. PR-Lin at 2.5x	3.286	.083

5.3 Preference Task

In this task, subjective preference was tested under different algorithms at the speed-up factors of 1.5x and 2.5x. The motivation was that minor artifacts caused by time compression which might not affect comprehension may still change a listener's preference.

At 1.5x (see Tables 7 and 8), we see that people's preference is essentially the same for Linear and Adapt, although there is a slight preference for PR-Lin (p=.093).

Table 7: The preference counts from the preference task.

Condition	Preference	1.5x	2.5x	Overall
Linear vs. PR-Lin	Linear	6	2	8
	None	5	8	13
	PR-Lin	13	14	27
PR-Lin vs. Adapt	PR-Lin	13	4	17
	None	5	9	14
	Adapt	6	11	17
Adapt vs. Linear	Adapt	8	21	29
	None	8	3	11
	Linear	8	0	8

Table 8: Chi square test results on the preference task.

Condition	Chi^2	P
Linear vs. PR-Lin at 1.5x	4.750	.093
PR-Lin vs. Adapt at 1.5x	4.750	.093
Adapt vs. Linear at 1.5x	.000	1.000
Linear vs. PR-Lin at 2.5x	9.000	.011
PR-Lin vs. Adapt at 2.5x	3.250	.197
Adapt vs. Linear at 2.5x	13.500	.000

At 2.5x, as may be expected, both PR-Lin and Adapt do significantly better than Linear (p=.011 and p=.000 respectively). Comparing the two non-linear algorithms, there is slight but non-significant preference for Adapt over PR-Lin (p=.197), with 11 subjects preferring Adapt, 8 having no preference, and 4 preferring PR-Lin.

5.4 Sustainable Speed

This task tries to measure the highest speed at which a subject can listen to the audio for a sustained period of time. The average speed-up factors at which the listeners eventually settled are summarized in Table 9. The highest speed-up factor is with Adapt (8% better than Linear), followed by PR-Lin (4% better than Linear).

Again a one-way within-subject ANOVA was conducted. The contrast between PR-Lin and Adapt as a group vs. Linear is significant (see Table 10). There is no significant difference between Adapt and PR-Lin, though there is a trend in favor of Adapt. We comment on this trend in the discussion section.

Table 9: Sustainable speed by conditions. WPM numbers converted from raw speed are also listed. The standard deviations are in the parenthesis.

Condition	Speed (StDev)	WPM (StDev)
Linear	1.62 (0.28)	273 (46)
PR-Lin	1.69 (0.38)	286 (63)
Adapt	1.76 (0.40)	298 (68)
Average	1.69 (0.36)	286 (60)

Table 10: The results of the one-way within-subject ANOVA contrast test for sustainable speed task. The results for raw speed and WPM are the same because all three clips were from the same speaker and have almost identical WPM.

Contrast test	F	P
Linear vs. (Adapt & PR-Lin) in speed	9.414	.005
Adapt vs. PR-Lin in speed	2.181	.153
Linear vs. (Adapt & PR-Lin) in WPM	9.414	.005
Adapt vs. PR-Lin in WPM	2.181	.153

6 DISCUSSION AND RELATED WORK

Before discussing results from this paper we briefly summarize results from the Mach1 paper [3]. The user study reported in the Mach1 paper included listener comprehension and preference tasks comparing Mach1 and linear time compression algorithm. Clips of 2 to 15 sentences in length were compressed at a speedup factor of 2.6-4.2. These are very high speeds, as the resulting word rates are from 390 to an astonishing 673 wpm. Listener comprehension for Mach1 compressed speech was found to improve on average 17% over that for linear time compressed speech. In the preference test, Mach1 compressed speech was chosen 95% of the time. The

difference between Mach1 and linear time compression was found to increase with the speedup factor.

In attempting to benefit from Mach1's results, and our own results reported earlier, it is useful to segment the observations into two sets: a) for low-to-medium speed-up factors (e.g., 1.5x), and b) for high speed-up factors (e.g., 2.5x).

For low-to-medium speed-up factors, we have no data from Mach1 paper. Our own data for 1.5x – looking at comprehension and preference metrics – shows that there is no significant difference between Linear, PR-Lin, and Adapt. There is a slight trend in favor of PR-Lin ($p=.093$) in the preference task. Our speculation is that this is due to the fact that with removal of pauses (~15-20% time savings upfront), PR-Lin has to compress the audible speech much less than the other two algorithms, and the data seem to indicate that people do not care as much about pauses when listening to short speeded-up speech segments.

At high speed-up factors (e.g., 2.5x), our own data show that there is significant preference for the non-linear algorithms (PR-Lin and Adapt) over Linear ($p=.008$ for comprehension task and $p=.011$ and $p=.000$ for preference task). These are consistent with the Mach1 results, which compared at even higher speed-up factors (2.6 to 4.2). Comparing PR-Lin and Adapt, while we see no significant differences at $p < .05$, we see a slight trend in favor of Adapt. Our intuition is that as we go to much higher speed-up factors beyond 2.5, Adapt will likely be significantly better.

So what do the above results imply for a designer? The first question to ask is what will be the sweet-spot speedup factor where users will spend most of the time. Our data here on sustainable speed indicates around 1.6-1.7 when in a hurry. Past results from Harrigan [11], Omoigui et al [19] and Li et al [16] indicate comfortable speed-up factor of ~1.4. Results from Foulke and Sticht [7] indicate speedup of ~1.25 corresponding to a word rate of 212 WPM.

The above data indicate that low-to-medium speed-up factors will likely dominate users' viewing patterns. Consequently, for most purposes the Linear algorithm should suffice – as discussed in Section 2.4, it is computationally efficient, simpler for client-server systems, and there is no jerky video. More aggressive implementations can go to PR-Lin, while still having the benefit of being computationally simple. Algorithms like Adapt/Mach1 may only be suitable for very high speed-up factors, for example, when one is in fast-forward mode.

As we were wrapping up these studies – and thinking about the results showing no substantial benefits from sophisticated algorithms like Mach1/Adapt at sustainable speeds – we were left wondering whether it is the case that these state-of-the-art algorithms are still not so good or whether we are hitting some more inherent human limits. With even the best algorithms, participants reached a sustainable speed of only 1.76x. Is that limit due to the technology or to a human limitation on the parsing end? Assuming humans are most adept at parsing natural human speech, this can be tested by comparing naturally sped up speech with artificially compressed speech. We ran two such comparisons in a quick user study.

A colleague of ours with significant background in public speaking was asked to read 2 articles (each around 700 words) and 3 short sentences at two speeds. His fast speed was approximately 1.4 times the regular speed. Both the *slow readings* (SR) and *fast readings* (FR) were digitized and were time compressed using our Adapt algorithm. Fifteen colleagues participated in the web-based experiment.

In the first comparison, subjects compared the slow readings speeded-up by Adapt at 1.4x versus the fast readings (which were naturally 1.4 times faster than the slow reading). Out of 45 total comparisons (since there were 3 short clips) 19 preferred FR, 18 preferred speeded-up SR, and 8 expressed no preference.

Our second comparison was a sustainable speed test where subjects speeded-up both SRs and FRs until comfortable. If naturally sped up speech is qualitatively different from that generated by Adapt, we would expect the benefits of each to be somewhat additive. Using Adapt, participants should be able to speed up the FR clips to a speed faster than that of the SR clips. This was not the case. When normalized to the speech-rate of the slow readings, the sustainable speed-up for SR was 1.63 and 1.68 for FR. There were no statistical differences, suggesting that the algorithm is a reasonable substitute for natural human speech "compression."

The results from both tasks support the hypothesis that for low-to-medium speed-up factors – speeds that end-users feel comfortable with – end-users cannot distinguish between computer algorithms speeding-up speech and a human speaking faster. The results also indicate that the current crop of algorithms is indeed very good, effectively substitutable for natural speech speed-up. It may be the case that limits are on the human-listening side rather than on how we generate time-compressed speech.

7 CONCLUDING REMARKS

We are faced with an information glut, both of textual information and, increasingly, audio-visual information. The most precious commodity today is human attention and time. Time-compression in some sense is a magical technology that helps us generate extra time by allowing us to watch audio-visual content speeded-up. Simple forms of time-compression technology are already appearing in commercial streaming-media products from Microsoft and Real Networks. The question explored in this paper is whether the new advanced algorithms for time-compression have the potential of significantly

enhancing user benefits (time savings) and to develop an understanding of the associated implementation costs.

Our results show that for speed-up factors most likely to be used by people, the more sophisticated non-linear time compression algorithms do not offer a significant advantage. Consequently, given the substantial implementation complexity associated with these algorithms in client-server streaming-media systems, we may not see them adopted in the near future. Based on a preliminary study, we speculate the problem is not that the benefits are small because the sophisticated algorithms are not very good. In fact, end-users cannot distinguish between these algorithms speeding-up speech and a human speaking faster. Thus delivering significantly larger time-compression benefits to end-users remains an open challenge for researchers.

ACNOWLEDGMENTS

The authors would like to thank Scott LeeTiernan for his help in statistical analysis. Thanks also go to JJ Cadiz for his initial implementation of the experiment code and his voice for the fast and slow reading experiment and Marc Smith for his valuable comments on the paper.

REFERENCES

1. Arons, B. “Techniques, Perception, and Applications of Time-Compressed Speech.” In *Proceedings of 1992 Conference*, American Voice I/O Society, Sep. 1992, pp. 169-177.
2. Atal, B.S. & Rabiner, L.R. “A Pattern Recognition Approach to Voiced-Unvoiced-Silence Classification with Applications to Speech Recognition,” *IEEE International Conference on Acoustics, Speech, and Signal Processing*, ASSP-24, 3 (June 1976), 201-212.
3. Covell, M., Withgott, M., & Slaney, M. “Mach1: Nonuniform Time-Scale Modification of Speech,” *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*. Seattle, WA, May 12-15 1998.
4. Davis, K.C. “Don’t Know Much About Geography,” *Bantam Doubleday Dell Audio Publishing*, New York, 1992.
5. Enounce, 2xAV Plug-in for RealPlayer http://www.enounce.com/products/real/2xav/index.htm
6. Fairbanks, G., Everitt, W.L., & Jaeger, R.P. "Method for Time or Frequency Compression-Expansion of Speech." *Transactions of the Institute of Radio Engineers, Professional Group on Audio AU-2* (1954): 7-12. Reprinted in G. Fairbanks, Experimental Phonetics: Selected Articles, University of Illinois Press, 1966.
7. Foulke, W. & Sticht, T.G. “Review of research on the intelligibility and comprehension of accelerated speech.” *Psychological Bulletin*, 72: 50-62, 1969.
8. Gan, C.K. & Donaldson, R.W. Adaptive Silence Deletion for Speech Storage and Voice Mail Applications. *IEEE Transactions on Acoustics, Speech, and Signal Processing 36, 6 (Jun. 1988)*, pp 924-927.
9. Gerber, S.E. “Limits of speech time compression.” In S. Duker (Ed.), Time-Compressed Speech, 456-465. Scarecrow, 1974.
10. Griffin, D.W. & Lim, J.S. “Signal estimation from modified short-time fourier transform.” *IEEE* Transactions *on Acoustics, Speech, and Signal Processing*, ASSP-32 (2): 236-243, 1984.
11. Harrigan, K. “The SPECIAL System: Self-Paced Education with Compressed Interactive Audio Learning,” *Journal of Research on Computing in Education*, 27, 3, Spring 1995.
12. Harrigan, K.A. “Just Noticeable Difference and Effects of Searching of User-Controlled Time-Compressed Digital-Video. Ph.D. Thesis, University of Toronto, 1996.
13. He, L., Grudin J. & Gupta, A., 2000. “Designing Presentations for On-demand Viewing,” In *Proc.CSCW’00*. ACM.
14. Heiman, G.W., Leo, R.J., Leighbody, G., & Bowler, K. "Word Intelligibility Decrements and the Comprehension of Time-Compressed Speech." *Perception and Psychophysics 40*, 6 (1986): 407-411.
15. Lee, S. & Kim, H. “Variable Time-Scale Modification of Speech Using Transient Information,” *IEEE International Conference on Acoustics, Speech, and Signal Processing*, Vol. 2, pp 1319-1322, Munich, 1997.
16. Li, F.C., Gupta, A., Sanocki, E., He, L. & Rui Y. "Browsing digital video," *Proc. CHI’00, Pages 169 – 176*, ACM.
17. Maxemchuk, N. "An Experimental Speech Storage and Editing Facility." *Bell System Technical Journal 59*, 8 (1980): 1383-1395.
18. Microsoft Corporation, Windows Media Encoder 7.0 http://www.microsoft.com/windows/windowsmedia/en/wm7/Encoder.asp
19. Omoigui, N., He, L., Gupta, A., Grudin, J. & Sanocki, E. Time-compression: System Concerns, Usage, and Benefits. Proceedings of *ACM Conference on Computer Human Interaction*, 1999.
20. Quereshi, S.U.H. “Speech compression by computer.” In S. Duker (Ed.), *Time-Compressed Speech*, 618-623. Scarecrow, 1974.
21. Roucos, S. & Wilgus, A. "High Quality Time-Scale Modification for Speech," *IEEE International Conference on Acoustics, Speech, and Signal Processing*, Vol. 2, pp 493-496, Tampa, FL, 1985.
22. Rymniak, M., Kurlandski, G, et al. “The Essential Review: TOEFL (Test of English as a Foreign Language,” *Kaplan Educational Centers* and *Simon & Schuster*, New York.
23. Stanford Online: Masters in Electrical Engineering, 1998. http://scpd.stanford.edu/cee/telecom/onlinedegree.html
24. Tavanapong, W., Hua, K.A. & Wang J.Z. “A Framework for Supporting Previewing and VCR Operations in a Low Bandwidth Environment,” In *Proc. Multimedia’97*, 303-312, ACM.
25. van Santen, J. “Assignment of Segmental Duration in Text-to-Speech Synthesis,” *Computer Speech and Language*, 8(2): 95-128, 1994.
26. Withgott, M. & Chen, F. “Computational Models of American Speech,” CSLI Lecture Notes #32, *Center for the Study of Language and Information*, Stanford, CA.