Rethinking the Progress Bar

Chris Harrison[1,2] *Brian Amento*[2] *Stacey Kuznetsov*[3] *Robert Bell*[2]

[1]Human Computer Interaction Institute
Carnegie Mellon University
chrish@cmu.edu

[2]AT&T Labs-Research
Florham Park, NJ
{brian,rbell}@research.att.com

[3]Computer Science Department
New York University
stacey@nyu.edu

ABSTRACT
Progress bars are prevalent interface elements in modern software. Typically, a linear function is employed, in which the progress of the bar is directly proportional to how much work has been completed. However, numerous factors cause progress bars to proceed at non-linear rates. Additionally, humans perceive time in a non-linear way. This paper explores the impact of various progress bar behaviors on user perception of process duration. We present a comprehensive experiment that was devised and conducted to determine the user perception of different progress bar behaviors. The results are used to suggest several design considerations that can be applied to enhance progress bars and ultimately improve users' computing experience.

ACM Classification: H5.2 [Information interfaces and presentation]: User Interfaces. - Graphical user interfaces.

General terms: Design, Human Factors

Keywords: Progress bars, user interfaces, time perception, human-centric, peak-and-end, duration neglect.

INTRODUCTION
Most software packages employ progress bars to visualize the status of an ongoing process. Users rely on progress bars to verify that an operation is proceeding successfully and to estimate its completion time [10]. Typically, a linear function is applied such that the advancement of a progress bar is directly proportional to the amount of work that has been completed. However, estimating progress can be difficult for complex or multi-stage processes. Varying disk, memory, processor, bandwidth and other factors complicate this further. Consequently, progress bars often exhibit non-linear behaviors, such as acceleration, deceleration, and pauses.

Humans do not perceive the passage of time in a linear way [1,3,7]. This, coupled with the irregular behavior of progress bars, causes human perception of process duration to vary. An understanding of which behaviors perceptually shorten process duration can help engineer a progress bar that feels faster, even though the actual duration remains unchanged. This paper describes an experiment that aimed to identify the impacts of progress bar behaviors on user perception. The results are then analyzed to identify the behaviors that perceptually speed up and slow down process execution. We conclude with several design suggestions that rely on perceptually enhanced progress bars to contribute to an overall more responsive, pleasant and human-centric computing experience.

RELATED WORK
Myers investigates the impact of progress indicators on user experience in graphical user interfaces [10]. He concludes that users have a strong preference for progress indicators during long tasks, and, overall, found them useful. Conn explores the concept of time affordance in [4], which enumerates a series of properties of an ideal progress bar. The exemplar offers users an accurate and understandable method for gauging progress in interactive systems. Conn also defines another concept, the time tolerance window, which is the length of time a user is wiling to wait before deciding a task is not making adequate progress. Conn goes on to describe that predicative algorithms could be applied to set user expectations for longer waits, essentially reporting progress in a non-linear method to enhance the user experience.

Moreover, Fredrickson et al. [5] suggests that duration had little effect on how pleasant people rated an affective experience (duration neglect). Instead, perception is most heavily influenced by salient features (both good and bad) during the experience and at the conclusion of the experience (peak-and-end effects). This occurs because humans do not remember experiences in a consistent and linear way, but rather, recall events selectively and with various biases [1,7]. Duration neglect and peak-and-end effects can be seen a variety of domains, including medicine, economics, advertising and human computer interaction (e.g. [11], [9], [2] and [6] respectively).

EXPERIMENT
We identified and developed eight non-linear functions that embodied different progress behaviors. A linear function was included as a baseline for comparison. Table 1 and Figure 1 describe the progress behaviors for each function. To test the human perception of these functions, an experimental application was developed that simultaneously presented two progress bars to the user (Figure 2). The progress bars ran in series; when the first progress bar completed the second began automatically. The duration of each progress bar was kept at a constant 5.5 seconds to act as a control. Three response buttons were provided which allowed users select which progress bar they believed felt faster or if they equal in duration. Another button enabled the user to replay each trail before proceeding to the subsequent pair of progress bars. Once an answer was provided, the next trial was initiated. The response and replay buttons could be pressed at anytime.

Name	Description	Rate Trend	Acceleration	Function
Linear	Progresses linearly	Constant	None	f(x) = x
Early Pause	Almost linear; large pause around 20%	Speeds up	Unstable near beginning	f(x) = x+(1-sin(x*π*2+π/2))/-8
Late Pause	Almost linear; large pause around 80%	Slows down	Unstable near end	f(x) = x+(1-sin(x*π*2+π/2))/8
Slow Wavy	Three large steps separated by pauses	Constant	Highly unstable	f(x) = x+sin(x*π*5)/20
Fast Wavy	Increments in small, quick steps	Constant	Highly unstable	f(x) = x+sin(x*π*20)/80
Power	Accelerates	Speeds up	Constant	f(x) = $(x+(1-x)*0.3)^2$
Inverse Power	Decelerates	Slows down	Constant	f(x) = $1+(1-x)^{1.5}*-1$
Fast Power	Rapidly accelerates	Speeds up	Stable	f(x) = $(x+(1-x)*0.5)^8$
Inv. Fast Power	Rapidly decelerates	Slows down	Stable	f(x) = $1+(1-x)^3*-1$

Table 1: The nine experimental progress functions.

The java-based application ran on an Apple Laptop with a 12" display running at 1024x768. The progress bars were custom made using Java Graphics2D primitives and were 600x50 pixels in size (approximately 1.2cm x 14.3cm). A coloration and naming scheme was applied to better visually inform the user: a running progress bar was shown in blue and titled "running", while completed progress bars were colored green and titled "finished." A touchpad with an integrated, single mouse button provided the means for input.

Comparing all distinct ordered pairs of the 9 progress functions would have required 81 trials. Initial pilot testing showed that users found the task to be fairly tedious and began to lose interest after approximately 50 sets of progress bars. To maintain subject interest and integrity of the responses, we decided to present all unique pairings of the nine progress functions (36 trials) along with the functions paired with themselves (9 trials) for a total of 45 trials per user. This kept the total task time under 15 minutes. The order of presentation was counterbalanced in two ways. First, a sequence of 45 trials was randomly selected for each pair of users. Second, within each pair of users, the order of presentation was reversed for each trial (i.e. if the first user of the pair saw linear/power, the second would see power/linear).

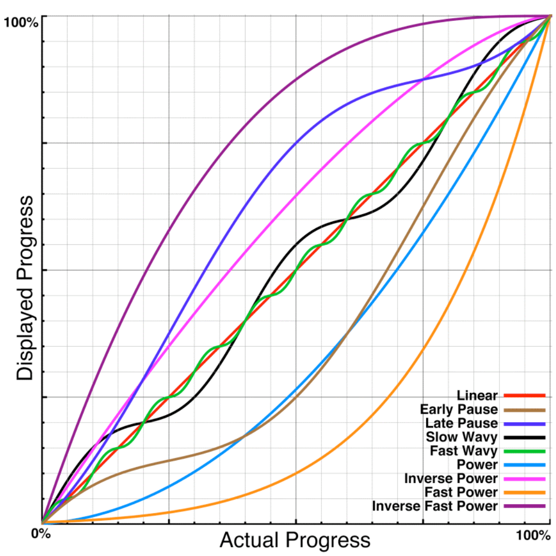

Figure 1: Graphs of the nine progress functions.

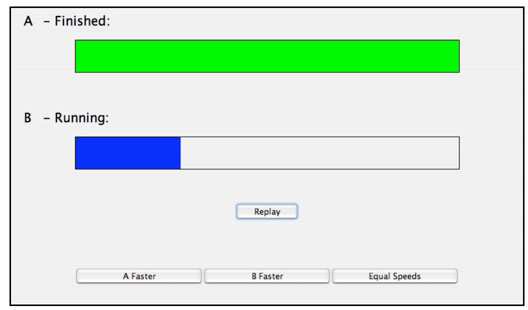

Figure 2: Experiment Interface.

We recruited 22 participants from two large computer research labs (14 male, 8 female) with a mean age of approximately 37. The experiment took place in the participants' offices. A brief verbal explanation of the simple comparison interface was given. Participants were told that progress bars may proceed at different rates and that they should select the one that they perceived as faster or equal if they appeared to be the same.

ANALYSIS AND RESULTS

Participants tended to prefer (i.e., perceive as faster) whichever function they saw first. Of the 990 paired comparisons, the first function was preferred 376 times (38%), the second 262 times (26%), with no preference 352 times (36%). This finding is supported by the results of a chi-square test, discussed subsequently.

Participants had strong preferences among the nine functions. For any paired comparison of functions, we assigned a preference score of +1 if the first function was preferred,

-1 if the second function was preferred, and 0 if the participant had no preference. Table 2 shows mean preference scores for each of the 36 function pairs. For example, in 22 comparisons of Slow Wavy with Fast Wavy (with each occurring first 11 times), 10 participants preferred Fast Wavy, 5 preferred Slow Wavy, and 7 rated the functions as equal. Consequently, the mean preference score is (10 – 5) / 22 = 0.23. The rows and columns of the table are ordered in terms of increasing overall preference. Bold values indicate statistical significance from 0 at the 0.05 level using a 2-sided sign test of the null hypothesis that each function was equally likely to be preferred.

Using the mean preferences scores in Table 2, we generated a rough ordering of preferences for the nine progress functions, shown in Figure 3.

To combine information efficiently across cells, while controlling for presentation order, we fit a logistic regression model [8] to the 638 cases where a preference was given. The probability of preferring Function *i* to Function *j* given that Function *i* was seen first was modeled as

$$P(i,\ j) = \frac{e^{\alpha+\beta_i}}{e^{\alpha+\beta_i} + e^{\beta_j}} = \frac{e^{\alpha+(\beta_i-\beta_j)}}{e^{\alpha+(\beta_i-\beta_j)} + 1}.$$

A Hosmer-Lemeshow chi-square test (8.87 with 7 d.f.) failed to show lack of fit of the model. The parameter α, estimated to be 0.42 with standard error 0.09, reflects the tendency for participants to prefer the first function they saw. The estimated α's measure the relative preferences among the functions. Because the probabilities only depend on differences between the α's, we fixed the estimated α for linear at 0 (Figure 4). Standard errors for differences between α's ranged between 0.28 and 0.37.

The nine functions clustered cleanly into three groups (Figure 4): three that were perceived as slower than linear, four that were perceived as near linear, and two which were perceived faster than linear. Differences between all three groups were significant but not necessarily significant within groups. The α for each function differs significantly at the 0.05 level from each function in any of the other clusters. The two functions that were perceived as faster than linear, power and fast power were both exponential functions with a speedup near the end of the progress bar, which users tend to prefer over all other functions.

Three general findings explain the pattern of estimates, which are inline with the peak-and-end effects mentioned previously. First, participants perceived progress bars with pauses as taking longer to complete (peak effect). Second, participants were more sensitive to the rate of progress near the end of the period than near the beginning (end effect). These two factors appear to combine in the case of Early Pause, making it essentially equivalently preferred to the linear function. Lastly, accelerating progress, especially near the conclusion of the process, was strongly favored.

	Slow Wavy	Late Pause	Inverse Fast Power	Inverse Power	Early Pause	Linear	Power	Fast Power
Fast Wavy	0.23	0.18	0.36	0.23	0.14	0.41	0.45	0.73
Slow Wavy		0.14	0.23	0.36	0.23	0.36	0.68	0.77
Late Pause			0.05	0.45	0.27	0.27	0.73	0.59
Inv. Fast Power				-0.14	0.00	-0.05	0.59	0.50
Inverse Power					0.41	-0.05	0.27	0.36
Early Pause						0.05	0.23	0.64
Linear							0.32	0.59
Power								0.00

Table 2: Preference score means for all pairs (orderings combined). Positive values indicate preference for the column label over the row label. Statistically significant results are bolded (p<.05)

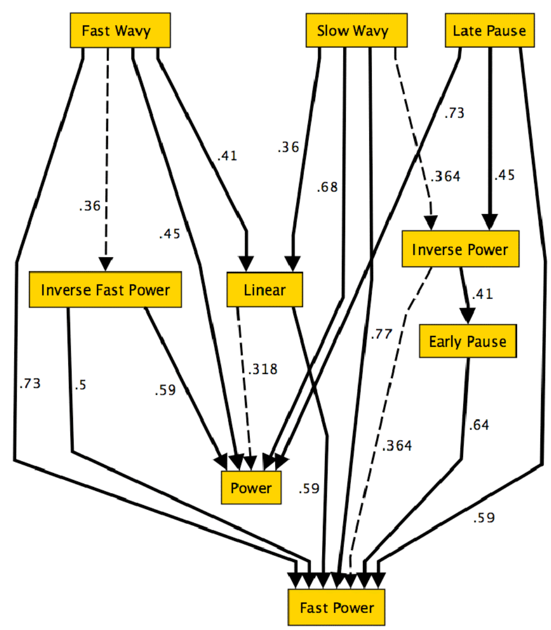

Figure 3: A rough hierarchy of the nine progress functions. Statistically significant edges are shown with solid lines (p<.05). Dashed edges show relationships approaching significance. Mean preference scores are labeled on the edges.

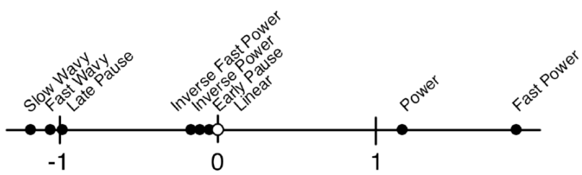

Figure 4: Number line showing relative distances from linear, which is centered at 0. Values generated from logistic regression model.

DISCUSSION

Although our results could be used to enhance progress bars system-wide, there are many cases where modifying progress behavior seems inappropriate. In general, processes with known, static completion conditions and stable progress are not good candidates – standard progress bars can visualize these effectively and accurately. In addition, these types of processes tend to be less effected by pauses or other negative progress behavior (sufficiently so that they are frequently accompanied by accurate time estimates). Examples of this type of process include copying a file to disk, encoding a video, scanning a photograph, or playing an audio file.

However, progress bars with dynamic completion conditions and roughly estimated durations (e.g. defragmenting a hard drive) can be augmented in two significant ways. First, users seem to have a strong aversion to pauses, especially towards the end of an operation. In response, it would be possible for an intelligent progress bar to cache progress when the operation is first starting to mitigate negative progress behaviors (e.g. pauses or slow-downs) later on. Secondly, progress could be downplayed in the beginning and accelerate towards the end, providing a sense of rapid process conclusion that was highly favored by users in the experiment.

The preferred progress behavior can also be integrated into the design of multi-stage processes such as the installation of software. Our results suggest that users are most willing to tolerate negative behavior (e.g. stalls and inconsistent progress) at the beginning of an operation. Hence, process stages can be arranged such that the slower operations are completed first. For example, if part of the installer requires fetching updates from a remote server and network connectivity could be irregular or unreliable, it may be best to run this stage early in the install sequence. The updates themselves can always be applied later, since they will run locally, with more predictable behavior.

A caveat to our findings is that perceptually enhancing progress bar behavior would most strongly benefit novice users. Unlike expert users, who are more likely to switch to another task, novices may monitor an extended operation until completion. Also, novice users are more likely to be frustrated and potentially frightened by irregular progress and pauses.

CONCLUSION

Different progress bar behaviors appear to have a significant effect on user perception of process duration. By minimizing negative behaviors and implementing positive behaviors, we can effectively make progress bars and their associated processes feel faster. Additionally, if elements of a multistage operation can be reordered, it may be possible to reorder the stages in a more pleasing and seemingly faster sequence.

FUTURE WORK

Testing new progress behaviors and durations is an obvious avenue. It may also be fruitful to run an adaptive experiment, in which individual progress bar durations are dynamically adjusted to a state where users find them all as having the same duration. This would allow the relative perceptive speed up and slow down to be evaluated. In this experiment, all progress bars completed in 5.5 seconds. However many operations, which employ progress bars to visualize their progress, are considerably longer in duration. It may be interesting to investigate if these general findings scale to other durations.

REFERENCES

1. Allan, L. G. The Perception of Time. (1979). *Perception and Psychophysics,* Vol. 26, pp. 340-354.
2. Baumgartner H., Sujan, M., and Padgett D. (1997). Patterns of Affective Reactions to Advertisements: The Integration of Moment-to-Moment Responses into Overall Judgments. *Journal of Marketing Research*, Vol. 34, No. 2, pp. 219-232.
3. Block, R. 1990. *Cognitive Models of Psychological Time,* Lawrence Erlbaum Associates, Hillsdale, NJ.
4. Conn, A. P. (1995). Time affordances: the time factor in diagnostic usability heuristics. In *Proceedings of the 1995 SIGCHI Conference on Human Factors in Computing Systems* (Denver, Colorado, May 1995). CHI '95. ACM Press, New York, NY, pp. 186-193.
5. Fredrickson, B. L. and Kahneman, D. (1993). Duration Neglect in Retrospective Evaluations of Affective Episodes. *Journal of Personality and Social Psychology*, Vol. 65, pp. 45-55.
6. Geelhoed, E., Toft, P., Roberts, S., and Hyland, P. (1995). To Influence Time Perception. In *Conference Companion on Human Factors in Computing Systems*. CHI '95. ACM Press, New York, NY, pp. 272-273.
7. Hogan, W. H. A Theoretical Reconciliation of Competing Views of Time Perception. (1978). *The American Journal of Psychology*, Vol. 91, No. 3, pp. 417-428.
8. Hosmer, D.W., and Lemeshow, S. 1989. *Applied Logistic Regression,* John Wiley & Sons, New York.
9. Langer, T., Sarin, R., and Weber, M. (2005). The Retrospective Evaluation of Payment Sequences: Duration Neglect and Peak-And-End Effects. *Journal of Economic Behavior & Organization,* Vol. 58, pp. 157-175.
10. Myers, B. A. (1985). The importance of percent-done progress indicators for computer-human interfaces. In *Proceedings of the 1985 SIGCHI Conference on Human*

Factors in Computing Systems (San Francisco, California). CHI '85. ACM Press, New York, NY, pp. 11-17.

11. Redelmeier, D., & Kahneman, D. (1996). Patients' Memories of Painful Medical Treatments: Real-time and Retrospective Evaluations of Two Minimally Invasive Procedures. *Pain*, Vol. 66, pp. 3–8.