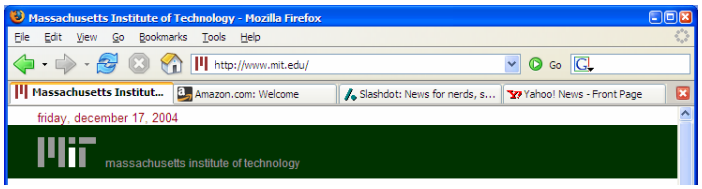
Lecture 4: Human Capabilities

UI Hall of Fame or Shame?



Fall 2004 6.831 UI Design and Implementation 2

Today's candidate for the User Interface Hall of Fame is **tabbed browsing**, a feature found in almost all web browsers (Mozilla, Firefox, Safari, Konqueror, Opera) except Internet Explorer. With tabbed browsing, multiple browser windows are grouped into a single top-level window and accessed by a row of tabs. You can open a hyperlink in a new tab by choosing that option from the right-click menu.

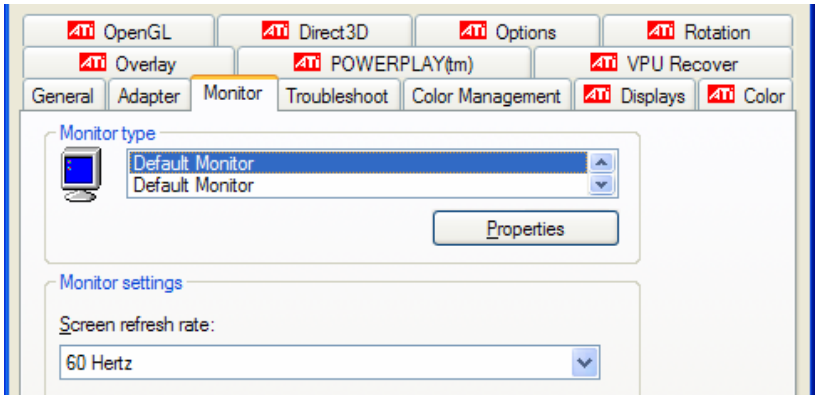
Tabbed browsing neatly solves a scaling problem in the Windows taskbar. If you accumulate several top-level Internet Explorer windows, they cease to be separately clickable buttons in the taskbar and merge together into a single Internet Explorer button with a popup menu. So your browser windows become **less visible** and **less efficient** to reach.

Tabbed browsing solves that by creating effectively a separate task bar specialized to the web browser. But it's even better than that: you can open multiple top-level browser windows, each with its own set of tabs. Each browser window can then be dedicated to a particular task, e.g. apartment hunting, airfare searching, programming documentation, web surfing. It's an easy and natural way for you to create task-specific groupings of your browser windows. That's what the Windows task bar tries to do when it groups windows from the same application together into a single popup menu, but that simplistic approach doesn't work at all because the Web is such a general-purpose platform. So tabbed browsing clearly wins on **task analysis**.

Another neat feature of tabbed browsing, at least in Mozilla, is that you can bookmark a set of tabs so you can recover them again later – a nice **shortcut** for task-oriented users.

What are the downsides of tabbed browsing? For one thing, you can't compare the contents of one tab with another. External windows would let you do this by resizing and repositioning the windows. Another problem is that, at least in Mozilla, tab groups can't be easily rearranged– moved to other windows, dragged out to start a new window.

Hall of Shame



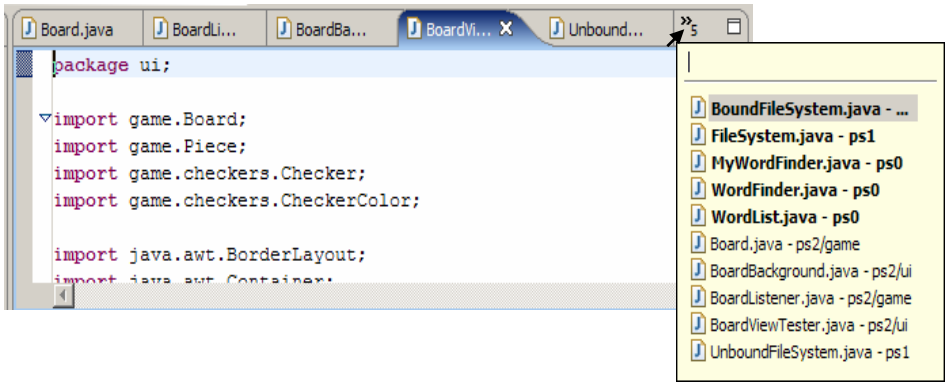
Fall 2004 6.831 UI Design and Implementation 3

Another problem is that tabs don't really scale up either – you can't have more than 5-10 without shrinking their labels so much that they're unreadable. Some designers have tried using **multiple rows of tabs**, but this turns out to be a horrible idea. Here's a typical example. Clicking on a tab in a back row (like OpenGL) has to move the whole row forward in order to maintain the tabbing metaphor. This is disorienting for two reasons: first, because the tab you clicked on has leaped out from under the mouse; and second, because other tabs you might have visited before are now in totally different places. Some plausible solutions to these problems were proposed in class – e.g., color-coding each row of tabs, or moving the front rows of tabs *below* the page. Animation might help too. All these ideas might reduce disorientation, but they involve tradeoffs like added visual complexity, greater demands on screen real estate, or having to move the page contents in addition to the tabs. And none of them prevent the tabs from jumping around, which is a basic problem with the approach.

As a rule of thumb, only one row of tabs really works, and the number of tabs you can fit in one row is constrained by the screen width and the tab label width. Most tabbing controls can scroll the tabs left to right, but scrolling tabs is definitely slower than picking from a popup menu.

In fact, the Windows task bar actually scales better than tabbing does, because it doesn't have to struggle to maintain a metaphor. The Windows task bar is just a row of buttons. Expanding the task bar to show two rows of buttons puts no strain on its usability, since the buttons don't have to jump around. Alas, you couldn't simply replace tabs with buttons in the dialog box shown here. (Why not?) Tabbed browsing probably couldn't use buttons either, without some careful graphic design to distinguish them from bookmark buttons.

Hall of Fame or Shame?



Here's how Eclipse 3.0 tries to address the tab scaling problem: it shows a few tabs, and the rest are found in a pulldown menu on the right end of the tab bar.

This menu has a couple of interesting features. First, it offers **incremental search**: typing into the first line of the menu will narrow the menu to tabs with matching titles. If you have a very large number of tabs, this could be a great shortcut. But it doesn't communicate its presence very well. I've been using Eclipse 3.0 for months, and I only noticed this feature when I started carefully exploring the tab interface.

Second, the menu tries to distinguish between the visible tabs and the hidden tabs using boldface. Quick, before studying the names of the tabs carefully -- which do you think is which? Was that a good decision?

Picking an item from the menu will make it appear as a tab – replacing one of the tabs that's currently showing. Which tab will get replaced? It's not immediately clear.

The key problem with this pulldown menu is that it completely disregards the **natural, spatial mapping** that tabs provide. The menu's order is unrelated to the order of the visible tabs; instead, the tabs are listed in the order they were last used. If you choose a hidden tab, it replaces the least recently used visible tab. LRU is a great policy for caches. Is it appropriate for frequently-accessed menus? No, because it interferes with users' spatial memory.

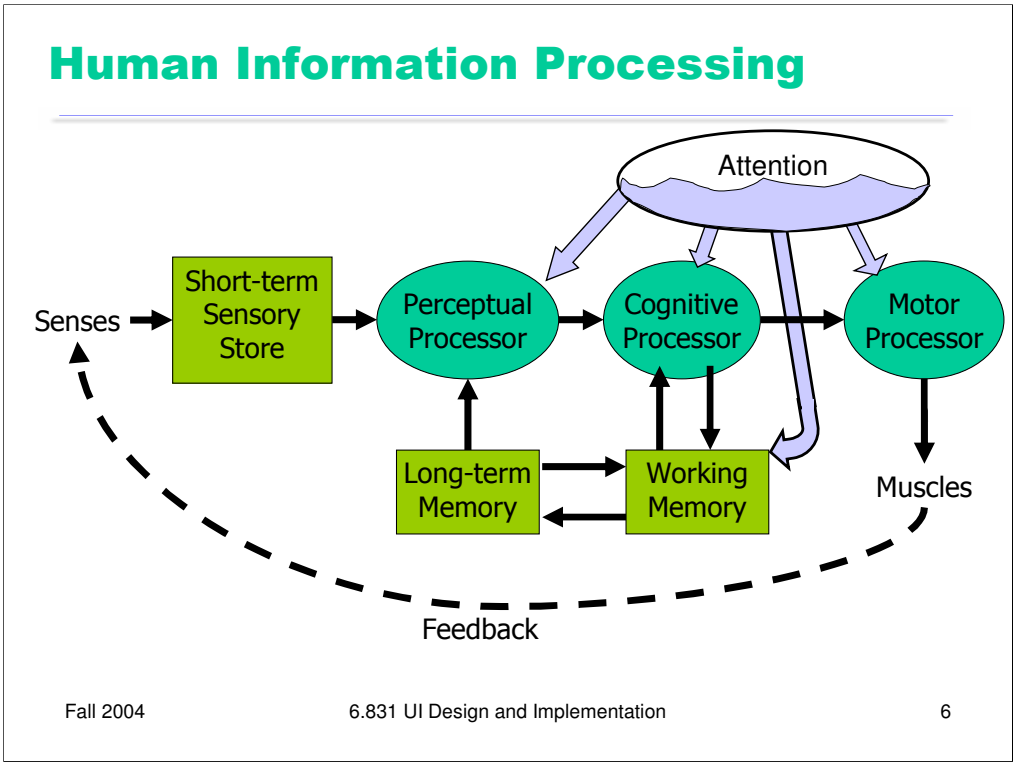
Today's Topics

- Human information processing
  - Perception
  - Motor skills
  - Memory
  - Decision making
  - Attention
  - Vision

This course is about building effective human-computer interfaces. Just as it helps to understand the properties of the computer system you're programming for – its processor speed, memory size, hard disk, operating system, and the interaction between these components – it's going to be important for us to understand some of the properties of the human that we're designing for.

We talked last week about user analysis, which collects information about high-level properties of our target users, particularly ways in which the target users are different from ourselves.

In today's lecture, we're going to look at low-level details: the processors, memories, and properties of the human cognitive apparatus. And we will largely concentrate on properties that most of us have in common (with some important exceptions when we look at color).



Here's a high-level look at the cognitive abilities of a human being -- really high level, like 30,000 feet. This is a version of the Model Human Processor was developed by Card, Moran, and Newell as a way to summarize decades of psychology research in an **engineering model**. (Card, Moran, Newell, *The Psychology of Human-Computer Interaction*, Lawrence Erlbaum Associates, 1983) This model is different from the original MHP; I've modified it to include a component representing the human's attention resources (Wickens, *Engineering Psychology and Human Performance*, Charles E. Merrill Publishing Company, 1984).

This model is an abstraction, of course. But it's an abstraction that actually gives us *numerical parameters* describing how we behave. Just as a computer has memory and processor, so does our model of a human. Actually, the model has several different kinds of memory, and several different processors.

Input from the eyes and ears is first stored in the **short-term sensory store**. As a computer hardware analogy, this memory is like a frame buffer, storing a single frame of perception.

The **perceptual processor** takes the stored sensory input and attempts to recognize *symbols* in it: letters, words, phonemes, icons. It is aided in this recognition by the **long-term memory**, which stores the symbols you know how to recognize.

The **cognitive processor** takes the symbols recognized by the perceptual processor and makes comparisons and decisions. It might also store and fetch symbols in **working memory** (which you might think of as RAM, although it's pretty small). The cognitive processor does most of the work that we think of as "thinking".

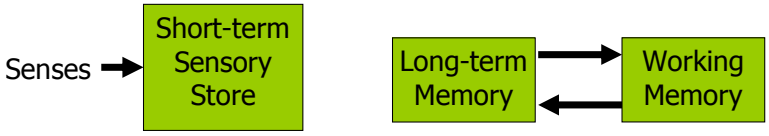
The **motor processor** receives an action from the cognitive processor and instructs the muscles to execute it. There's an implicit **feedback** loop here: the effect of the action (either on the position of your body or on the state of the world) can be observed by your senses, and used to correct the motion in a continuous process.

Finally, there is a component corresponding to your **attention**, which might be thought of like a thread of control in a computer system.

Note that this model isn't meant to reflect the anatomy of your nervous system. There probably isn't a single area in your brain corresponding to the perceptual processor, for example. But it's a useful abstraction nevertheless.

We'll look at each of these parts in more detail, starting with the processors.

Memories



- Memory properties
  - Encoding: type of things stored
  - Size: number of things stored
  - Decay time: how long memory lasts

Each component of our model has some properties. For example, memories are characterized by three properties: encoding, size, and decay time.

Short-Term Sensory Store

- Visual information store
  - encoded as physical image
  - size ~ 17 [7-17] letters
  - decay ~ 200 ms [70-1000 ms]
- Auditory information store
  - encoded as physical sound
  - size ~ 5 [4.4-6.2] letters
  - decay ~ 1500 ms [900-3500 ms]

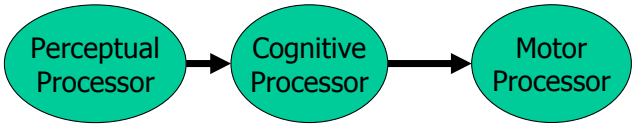
The **visual image store** is basically an image frame from the eyes. It isn't encoded as pixels, but as physical features of the image, such as curvature, length, edges. It retains physical features like intensity that may be discarded in higher-level memories (like the working memory). We measure its size in letters because psych studies have used letters as a convenient stimulus for measuring the properties of the VIS; this doesn't mean that letters are represented symbolically in the VIS. The VIS memory is fleeting, decaying in a few hundred milliseconds.

The **auditory image store** is a buffer for physical sound. Its size is much smaller than the VIS (in terms of letters), but lasts longer – seconds, rather than tenths of a second.

Both of these stores are **preattentional**; that is, they don't need the spotlight of attention to focus on them in order to be collected and stored. Attention can be focused on the visual or auditory stimulus after the fact. That accounts for phenomena like "What did you say? Oh yeah."

Processors

- Processors have a cycle time
  - T_p ~ 100ms [50-200 ms]
  - T_c ~ 70ms [30-100 ms]
  - T_m ~ 70ms [25-170 ms]



- Fastman may be 10x faster than Slowman

Turning to the processors, the main property of a processor is its **cycle time**, which is analogous to the cycle time of a computer processor. It's the time needed to accept one input and produce one output.

Like all parameters in the MHP, the cycle times shown above are derived from a survey of psychological studies. Each parameter is specified with a typical value and a range of reported values. For example, the typical cycle time for perceptual processor, T_p, is 100 milliseconds, but studies have reported between 50 and 200 milliseconds. The reason for the range is not only variance in individual humans; it is also varies with conditions. For example, the perceptual processor is faster (shorter cycle time) for more intense stimuli, and slower for weak stimuli. You can't read as fast in the dark. Similarly, your cognitive processor actually works faster under load! Consider how fast your mind works when you're driving or playing a video game, relative to sitting quietly and reading. The cognitive processor is also faster on practiced tasks.

It's reasonable, when we're making engineering decisions, to deal with this uncertainty by using all three numbers, not only the nominal value but also the range. Card, Moran, & Newell gave names to these three imaginary humans: Fastman, whose times are all as fast as possible; Slowman, whose times are all slow; and Middleman, whose times are all typical.

Perceptual Fusion

- Two stimuli within the same PP cycle (T_p ~ 100ms) appear **fused**
- Consequences
  - 1/ T_p frames/sec is enough to perceive a moving picture (10 fps OK, 20 fps smooth)
  - Computer response < T_p feels instantaneous
  - Causality is strongly influenced by fusion

One interesting effect of the perceptual processor is **perceptual fusion**. Here's an intuition for how fusion works. Every cycle, the perceptual processor grabs a frame (snaps a picture). Two events occurring less than the cycle time apart are likely to appear in the same frame. If the events are similar – e.g., Mickey Mouse appearing in one position, and then a short time later in another position – then the events tend to *fuse* into a single perceived event – a single Mickey Mouse, in motion.

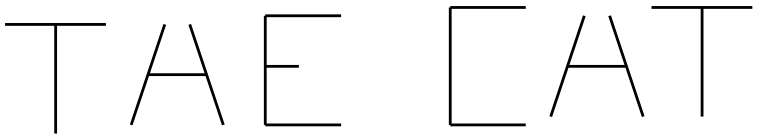
Perceptual fusion is responsible for the way we perceive a sequence of movie frames as a moving picture, so the parameters of the perceptual processor give us a lower bound on the frame rate for believable animation. 10 frames per second is good for Middleman, but 20 frames per second is better for Fastman (remember that Fastman's T_p = 50 ms represents not just the quickest humans, but also the most favorable conditions).

Perceptual fusion also gives an upper bound on good computer response time. If a computer responds to a user's action within T_p time, its response feels instantaneous with the action itself. Systems with that kind of response time tend to feel like extensions of the user's body. If you used a text editor that took longer than T_p response time to display each keystroke, you would notice.

Fusion also strongly affects our perception of causality. If one event is closely followed by another – e.g., pressing a key and seeing a change in the screen – and the interval separating the events is less than T_p, then we are more inclined to believe that the first event caused the second.

Bottom-up vs. Top-Down Perception

- Bottom-up uses features of stimulus
- Top-down uses context
  - temporal, spatial
  - draws on long-term memory



Perception is not an isolated process. It uses both **bottom-up** processing, in which the features of a stimulus are combined to identify it, and **top-down** processing, where the context of the stimulus contributes to its recognition. In visual perception, the context is usually *spatial*, i.e., what's around the stimulus. In auditory perception, the context is *temporal* -- what you heard before or after the stimulus.

Look at the two words drawn above. The middle letter of each word is exactly identical – halfway between H and A – but the effect of the context strongly influences how we see each letter.

We'll see more about this perceptual effect in a future lecture when we talk about Gestalt principles in graphic design.

Chunking

- "Chunk": unit of perception or memory
- Chunking depends on presentation and what you already know

  B M W R C A A O L I B M F B I

  MWR CAA OLI BMF BIB

  BMW RCA AOL IBM FBI
- 3-4 digit chunking is ideal for encoding unrelated digits

The elements of perception and memory are called **chunks**. In one sense, chunks are defined symbols; in another sense, a chunk represents the activation of past experience.

Our ability to form chunks in working memory depends strongly on how the information is presented – a sequence of individual letters tend to be chunked as letters, but a sequence of three-letter groups tend to be chunked as groups. It also depends on what we already know. If the three letter groups are well-known TLAs (three-letter acronyms) with well-established chunks in long-term memory, we are better able to retain them in working memory.

Chunking is illustrated well by a famous study of chess players. Novices and chess masters were asked to study chess board configurations and recreate them from memory. The novices could only remember the positions of a few pieces. Masters, on the other hand, could remember entire boards, but only when the pieces were arranged in *legal* configurations. When the pieces were arranged randomly, masters were no better than novices. The ability of a master to remember board configurations derives from their ability to **chunk** the board, recognizing patterns from their past experience of playing and studying games.

Attention and Perception

- Spotlight metaphor
  - Spotlight moves serially from one input channel to another
  - **Visual dominance**: easier to attend to visual channels than auditory channels
  - All stimuli within spotlighted channel are processed in parallel
    - Whether you want to or not

Let's look at how attention is involved with perception. The metaphor used by cognitive psychologists for how attention behaves in perception is the **spotlight**: you can focus your attention (and your perceptual processor) on only one input channel in your environment at a time. This input channel might be a location in your visual field, or it might be a location or voice in your auditory field. Humans are very visually-oriented, so it turns out to be easier to attend to visual channels than auditory channels.

Once you've focused your attention on a particular channel, all the stimuli within the area of the "spotlight" are then processed, whether you mean to or not. This can cause **interference**, as the next demonstration shows.

Here's a little demonstration of interference. Say the **colors** of each of these words aloud, and time yourself.

Now Do It Again

Green

Orange

Red

Black

Pink

Blue

Now do it again – say the **colors** of each word aloud. It's harder, and most people do it much slower than the previous task. Why? Because the word, which names a different color, interferes with the color we're trying to say. This is called the Stroop effect.

The lesson we should take away here is that we should choose the secondary characteristics of our displays – like the multiple dimensions of stimulus, or the context around the stimulus – to **reinforce** the message of the display, not **interfere** with it.

Cognitive Processing

- Cognitive processor
  - compares stimuli
  - selects a response
- Types of decision making
  - Skill-based
  - Rule-based
  - Knowledge-based

Let's say a little about the cognitive processor, which is responsible for making comparisons and decisions.

Cognition is a rich, complex process. The best-understood aspect of it is **skill-based** decision making. A skill is a procedure that has been learned thoroughly from practice; walking, talking, pointing, reading, driving, typing are skills most of us have learned well. Skill-based decisions are automatic responses that require little or no attention. Since skill-based decisions are very mechanical, they are easiest to describe in a mechanical model like the one we're discussing.

Two other kinds of decision making are **rule-based**, in which the human is consciously processing a set of rules of the form *if X, then do Y*; and **knowledge-based**, which involves much higher-level thinking and problem-solving. Rule-based decisions are typically made by novices at a task: when a student driver approaches an intersection, for example, they must think explicitly about what they need to do in response to each possible condition. Knowledge-based decision making is used to handle unfamiliar or unexpected problems, such as figuring out why your car won't start.

We'll focus on skill-based decision making for the purposes of this lecture, because it's well understood.

Hick-Hyman Law of Choice Reaction Time

- Reaction time depends on information content of stimulus

$$RT = c + d \log_2 1/Pr(stimulus)$$

  - e.g., for N equiprobable stimuli, each requiring a different response:

$$RT = c + d \log_2 N$$

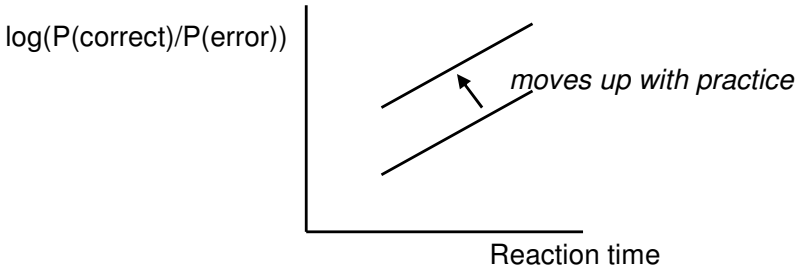
Simple reaction time – responding to a single stimulus with a single response – takes just one cycle of the human information processor, i.e. Tp+Tc+Tm.

But if the user must make a **choice** – choosing a different response for each stimulus – then the cognitive processor may have to do more work. The Hick-Hyman Law of Reaction Time shows that the number of cycles required by the cognitive processor is proportional to amount of **information** in the stimulus. For example, if there are N equally probable stimuli, each requiring a different response, then the cognitive processor needs log N cycles to decide which stimulus was actually seen and respond appropriately. So if you double the number of possible stimuli, a human's reaction time only increases by a constant.

Keep in mind that this law applies only to *skill-based* decision making; we assume that the user has practiced responding to the stimuli, and formed an internal model of the expected probability of the stimuli.

Speed-Accuracy Tradeoff

- Accuracy varies with reaction time
  - Can choose any point on curve
  - Can move curve with practice



Another important phenomenon of the cognitive processor is the fact that we can tune its performance to various points on a **speed-accuracy** tradeoff curve. We can force ourselves to make decisions faster (shorter reaction time) at the cost of making some of those decisions wrong. Conversely, we can slow down, take a longer time for each decision and improve accuracy. It turns out that for skill-based decision making, reaction time varies linearly with the log of odds of correctness; i.e., a constant increase in reaction time can double the odds of a correct decision.

The speed-accuracy curve isn't fixed; it can be moved up by practicing the task. Also, people have different curves for different tasks; a pro tennis player will have a high curve for tennis but a low one for surgery.

Divided Attention (Multitasking)

- Resource metaphor
  - Attention is a resource that can be divided among different tasks simultaneously
- Multitasking performance depends on:
  - Task structure
    - Modality: visual vs. auditory
    - Encoding: spatial vs. verbal
    - Component: perceptual/cognitive vs. motor vs. WM
  - Difficulty
    - Easy or well-practiced tasks are easier to share

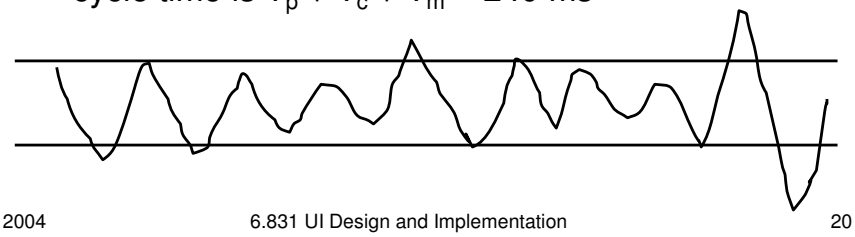
Earlier we saw the spotlight metaphor for attention. Now we'll refine it to account for our ability to handle multiple things at the same time. The **resource metaphor** regards attention as a limited resource that can be subdivided, under the human's control, among different tasks simultaneously.

Our ability to divide our attention among multiple tasks appears to depend on two things. First is the **structure** of the tasks that are trying to share our attention. Tasks with different characteristics are easier to share; tasks with similar characteristics tend to interfere. Important dimensions for task interference seem to be the **modality** of the task's input (visual or auditory), its **encoding** (e.g., spatial/graphical/sound encoding, vs. words), and the mental **components** required to perform it. For example, reading two things at the same time is much harder than reading and listening, because reading and listening use two different modalities.

The second key influence on multitasking performance is the difficulty of the task. Carrying on a conversation while driving a car is fairly effortless as long as the road is familiar and free of obstacles; when the driver must deal with traffic or navigation, conversation tends to slow down or even stop.

Motor Processing

- Open-loop control
  - Motor processor runs a program by itself
  - cycle time is T_m ~ 70 ms
- Closed-loop control
  - Muscle movements (or their effect on the world) are perceived and compared with desired result
  - cycle time is $T_p + T_c + T_m$ ~ 240 ms



The motor processor can operate in two ways. It can run autonomously, repeatedly issuing the same instructions to the muscles. This is "open-loop" control; the motor processor receives no feedback from the perceptual system about whether its instructions are correct. With open loop control, the maximum rate of operation is just T_m.

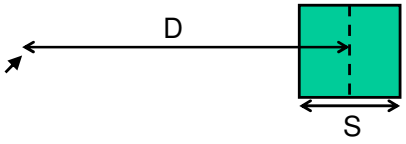
The other way is "closed-loop" control, which has a complete feedback loop. The perceptual system looks at what the motor processor did, and the cognitive system makes a decision about how to correct the movement, and then the motor system issues a new instruction. At best, the feedback loop needs one cycle of each processor to run, or $T_p + T_c + T_m$ ~ 240 ms.

Here's a simple but interesting experiment that you can try: take a sheet of lined paper and scribble a sawtooth wave back and forth between two lines, going as fast as you can but trying to hit the lines exactly on every peak and trough. Do it for 5 seconds. The frequency of the sawtooth carrier wave is dictated by open-loop control, so you can use it to derive your T_m. The frequency of the wave's **envelope**, the corrections you had to make to get your scribble back to the lines, is closed-loop control. You can use that to derive your value of $T_p + T_c$.

Fitts's Law

- Fitt's Law
  - Time T to move your hand to a target of size S at distance D away is:

$$T = RT + MT = a + b \log (2D/S)$$

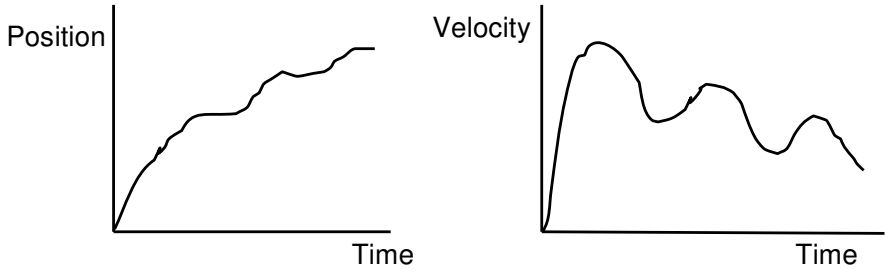


  - Depends only on *index of difficulty* log(2D/S)

Fitts's Law specifies how fast you can move your hand to a target of a certain size at a certain distance away (within arm's length, of course). It's a fundamental law of the human sensory-motor system, which has been replicated by numerous studies. Fitts's Law applies equally well to using a mouse to point at a target on a screen.

Explanation of Fitts's Law

- Moving your hand to a target is closed-loop control
- Each cycle covers remaining distance D with error εD



We can explain Fitts's Law by appealing to the human information processing model. Fitt's Law relies on closed-loop control. In each cycle, your motor system instructs your hand to move the entire remaining distance D. The accuracy of that motion is proportional to the distance moved, so your hand gets within some error εD of the target (possibly undershooting, possibly overshooting). Your perceptual and cognitive processors perceive where your hand arrived and compare it to the target, and then your motor system issues a correction to move the remaining distance εD – which it does, but again with proportional error, so your hand is now within $\varepsilon^2 D$. This process repeats, with the error decreasing geometrically, until *n* iterations have brought your hand within the target – i.e., $\varepsilon^n D \leq \frac{1}{2} S$. Solving for *n*, and letting the total time $T = n (T_p + T_c + T_m)$, we get:

$$T = a + b \log (2D/S)$$

where a is the reaction time for getting your hand moving, and $b = - (T_p + T_c + T_m)/\log \varepsilon$.

The graphs above show the typical trajectory of a person's hand, demonstrating this correction cycle in action. The position-time graph shows an alternating sequence of movements and plateaus; each one corresponds to one cycle. The velocity-time graph shows the same effect, and emphasizes that hand velocity of each subsequent cycle is smaller, since the motor processor must achieve more precision on each iteration.

Implications of Fitts's Law

- Targets at screen edge are easy to hit
  - Mac menubar beats Windows menubar
  - Unclickable margins are foolish
- Hierarchical menus are hard to hit
  - Gimp/GTK: instantly closes menu
  - Windows: .5 s timeout destroys causality
  - Mac does it right: triangular zone
- Linear popup menus vs. pie menus

Fitts's Law has some interesting implications:

•The edge of the screen stops the mouse pointer, so you don't need more than one correcting cycle to hit it. Essentially, the edge of the screen acts like a target with *infinite* size. (More precisely, the distance D to the center of the target is virtually equal to half the size S of the target, so T = a + b log (2D/S) solves to the minimum time T=a+b.) So edge-of-screen real estate is precious. The Macintosh menu bar, positioned at the top of the screen, is faster to use than a Windows menu bar (which, even when a window is maximized, is displaced by the title bar). Similarly, if you put controls at the edges of the screen, they should be active all the way to the edge to take advantage of this effect. Don't put an unclickable margin beside them.

•As we discussed last week, hierarchical submenus are hard to use, because of the correction cycles the user is forced to spend getting the mouse pointer carefully over into the submenu. Windows tries to solve this problem with a 500 ms timeout, and now we know another reason that this solution isn't ideal: it exceeds T_p (even for Slowman), so it destroys perceptual fusion and our sense of causality. Intentionally moving the mouse down to the next menu results in a noticeable delay. The Mac gets a Hall of Fame nod here, for doing it right with a triangular zone of activation for the submenu. The user can point straight to the submenu without unusual corrections, and without even noticing that there might be a problem. Hall of Fame interfaces are often invisible!

•Fitts's Law also explains why pie menus are faster to use than linear popup menus. With a pie menu, every menu item is a slice of a pie centered on the mouse pointer. As a result, each menu item is the same distance D away from the mouse pointer, and its size S (in the radial direction) is comparable to D. Contrast that with a linear menu, where items further down the menu have larger D, and all items have a small S (height).

Power Law of Practice

- Time T_n to do a task the *n*th time is:

$$T_n = T_1 \, n^{-\alpha}$$

α is typically 0.2-0.6

An important feature of the entire perceptual-cognitive-motor system is that the time to do a task decreases with practice. In particular, the time decreases according to the power law, shown above. The power law describes a linear curve on a log-log scale of time and number of trials.

In practice, the power law means that novices get rapidly better at a task with practice, but then their performance levels off to nearly flat (although still slowly improving).

Working Memory (WM)

- Small capacity: 7 ± 2 "chunks"
- Fast decay (7 [5-226] sec)
- **Maintenance rehearsal** fends off decay
- **Interference** causes faster decay

Working memory is where you do your conscious thinking. Working memory is where the cognitive processor gets its operands and drops its results. The currently favored model in cognitive science holds that working memory is not actually a separate place in the brain, but rather a pattern of **activation** of elements in the long-term memory.

A famous result, due to George Miller (unrelated), is that the capacity of working memory is roughly 7 ± 2 chunks.

Working memory decays in tens of seconds. **Maintenance rehearsal** – repeating the items to yourself – fends off this decay, much as DRAM must refresh itself. Maintenance rehearsal requires attentional resources. But distraction destroys working memory. A particularly strong kind of distraction is **interference** – stimuli that activate several conflicting chunks are much harder to retain. Recall the Stroop effect from earlier in this lecture: not only does it slow down perception, but it also inhibits our ability to retain the colors in working memory.

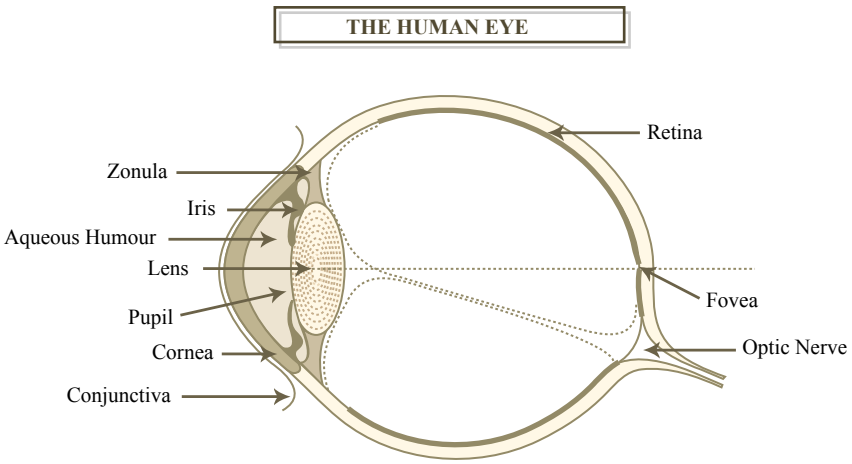
Long-term Memory (LTM)

- Huge capacity
- Little decay
- **Elaborative rehearsal** moves chunks from WM to LTM by making connections with other chunks

Long-term memory is probably the least understood part of human cognition. It contains the mass of our memories. Its capacity is huge, and it exhibits little decay. Long-term memories are apparently not intentionally erased; they just become inaccessible.

Maintenance rehearsal (repetition) appears to be useless for moving information into into long-term memory. Instead, the mechanism seems to be **elaborative rehearsal**, which seeks to make connections with existing chunks. Elaborative rehearsal lies behind the power of mnemonic techniques like associating things you need to remember with familiar places, like rooms in your childhood home. Elaborative rehearsal requires attention resources as well.

The Eye



Fall 2004 6.831 UI Design and Implementation 27

OK, we've looked at perception, cognition, motor skills, and memory. We'll conclude our discussion of the human machine today by considering the vision system in a little more detail, since vision is the primary way that a graphical user interface communicates to the user.

Here are key parts of the anatomy of the eye:

•The **cornea** is the transparent, curved membrane on the front of the eye.

•The **aqueous humor** fills the cavity between the cornea and the lens, and provides most of the optical power of the eye because of the large difference between its refractive index and the refractive index of the air outside the cornea.

•The **iris** is the colored part of the eye, which covers the lens. It is an opaque muscle, with a hole in the center called the **pupil** that lets light through to fall on the lens. The iris opens and closes the pupil depending on the intensity of light; it opens in dim light, and closes in bright light.

•The **lens** focuses light. Under muscle control, it can move forward and backward, and also get thinner or fatter to change its focal length.

•The **retina** is the surface of the inside of the eye, which is covered with light-sensitive receptor cells.

•The **fovea** is the spot where the optical axis (center of the lens) impinges on the retina. The highest density of photoreceptors can be found in the fovea; the fovea is the center of your visual field.

Photoreceptors

- Rods
  - Only one kind (peak response in green wavelengths)
  - Sensitive to low light ("scotopic vision")
    - Multiple nearby rods aggregated into a single nerve signal
  - Saturated at moderate light intensity ("photopic vision")
    - Cones do most of the vision under photopic conditions
- Cones
  - Operate in brighter light
  - Three kinds: S(hort), M(edium), L(ong)
  - S cones are very weak, centered in blue wavelengths
  - M and L cones are more powerful, overlapping
  - M centered in green, L in yellow (but called "red")

There are two kinds of photoreceptor cells in the retina. **Rods** operate under low-light conditions – night vision. There is only one kind of rod, with one frequency response curve centered in green wavelengths, so rods don't provide color vision. Rods saturate at moderate intensities of light, so they contribute little to daytime vision. **Cones** respond only in brighter light. There are three kinds of cones, called S, M, and L after the centers of their wavelength peaks. S cones have very weak frequency response centered in blue. M and L cones are two orders of magnitude stronger, and their frequency response curves nearly overlap.

Signals from Photoreceptors

- Brightness
  M + L + rods
- Red-green difference
  L - M
- Blue-yellow difference
  weighted sum of S, M, L

The rods and cones do not send their signals directly to the visual cortex; instead, the signals are recombined into three channels. One channel is **brightness**, produced by the M and L cones and the rods. This is the only channel really active at night. The other two channels convey color **differences**. High responses mean red, and low responses indicate green.

These difference channels drive the theory of **opponent colors**: red and green are good contrasting colors because they drive the red-green channel to opposite extremes. Similarly, black/white and blue/yellow are good contrasting pairs.

Color Blindness

- Red-green color blindness (protonopia & deuteranopia)
  - 8% of males
  - 0.4% of females
- Blue-yellow color blindness (tritanopia)
  - Far more rare
- Guideline: don't depend solely on color distinctions
  - use redundant signals: brightness, location, shape

Color deficiency ("color blindness") affects a significant fraction of human beings. An overwhelming number of them are male.

There are three kinds of color deficiency, which we can understand better now that we understand a little about the eye's anatomy:

•**Protonopia** is missing or bad L cones. The consequence is reduced sensitivity to red-green differences (the L-M channel is weaker), and reds are perceived as darker than normal.

•**Deuteranopia** is caused by missing or malfunctioning M cones. Red-green difference sensitivity is reduced, but reds do not appear darker.

•**Tritanopia** is caused by missing or malfunctioning S cones, and results in blue-yellow insensitivity.

Since color blindness affects so many people, it is essential to take it into account when you are deciding how to use color in a user interface. Don't depend solely on color distinctions, particularly red-green distinctions, for conveying information. Microsoft Office applications fail in this respect: red wavy underlines indicate spelling errors, while identical green wavy underlines indicate grammar errors.

Traffic lights are another source of problems. How do red-green color-blind people know whether the light is green or red? Fortunately, there's a spatial cue: red is always above (or to the right of) green. Protonopia sufferers (as opposed to deuteranopians) have an additional advantage: the red light looks darker than the green light.

Chromatic Aberration

- Different wavelengths focus differently
  - Highly separated wavelengths (red & blue) can't be focused simultaneously
- Guideline: don't use red-on-blue text
  - It looks fuzzy and hurts to read

The refractive index of the lens varies with the wavelength of the light passing through it; just like a prism, different wavelengths are bent at different angles. So your eye needs to focus differently on red features than it does on blue features.

As a result, an edge between widely-separated wavelengths – like blue and red – simply can't be focused. It always looks a little fuzzy. So blue-on-red or red-on-blue text is painful to read, and should be avoided at all costs.

Apple's ForceQuit tool in Mac OS X, which allows users to shut down misbehaving applications, unfortunately falls into this trap. In its dialog, unresponding applications are helpfully displayed in red. But the selection is a blue highlight. The result is incredibly hard to read.

Blue Details Are Hard to Resolve

- Fovea has no S cones
  - Can't resolve small blue features (unless they have high contrast with background)
- Lens and aqueous humor turn yellow with age
  - Blue wavelengths are filtered out
- Lens weakens with age
  - Blue is harder to focus
- Guideline: don't use blue against dark backgrounds where small details matter (text!)

A number of anatomical details conspire to make blue a bad color choice when small details matter.

First, the fovea has very few S cones, so you can't easily see blue features in the center of your vision (unless they have high contrast with the background, activating the M and L cones).

Second, older eyes are far less sensitive to blue, because the lens and aqueous humor slowly grow yellower, filtering out the blue wavelengths.

Finally, the lens gets weaker with age. Blue is at one extreme of its focusing range, so older eyes can't focus blue features as well.

As a result, avoid blue text, particularly small blue text.

Fovea Has No Rods

- Rods are more sensitive to dim light
- In scotopic conditions, peripheral vision (rod-rich) is better than foveal vision
  - Easier to see a dim star if you don't look directly at it

Incidentally, the fovea has no rods, either. That explains why it's easier to see a dim star if you don't look beside it, rather than directly at it.