

Cognitive Theories in HCI

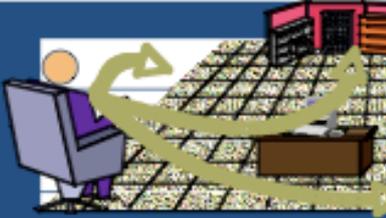
Cognitive Modeling

Cognitive approach to HCI attempts to predict user performance based on a model of cognition

Start with a model of how humans act

Use that model to predict how humans would complete tasks using a particular UI

Provided theoretical foundation of HCI in the 1970s and 1980s

10^7 (months) 10^6 (weeks) 10^5 (days)	SOCIAL	Social Behavior	
10^4 (hours) 10^3 10^2 (minutes)	RATIONAL	Adaptive Behavior	
10^1 10^0 (seconds) 10^{-1}	COGNITIVE	Immediate Behavior	
10^{-2} 10^{-3} (msec) 10^{-4}	BIOLOGICAL		

Cognitive Models are...

Abstract

Quantitative

Approximate

Estimated from experiments

Based on a theory of cognition

Advantages

Can predict without implement /
prototype

Don't need to test with real users

Theory has explanatory power

Provide a scientific foundation for
design, like other engineering fields

Cognitive Theories in HCI

Fitts' Law - Predicts how long it will take a user to select a target; used for evaluating device input

KLM (Keystroke-Level Model) - Description of user tasks based on low-level actions (keystrokes, etc.)

GOMS (Goals, Operators, Methods, Selectors) - Higher-level than KLM, with structure and hierarchy

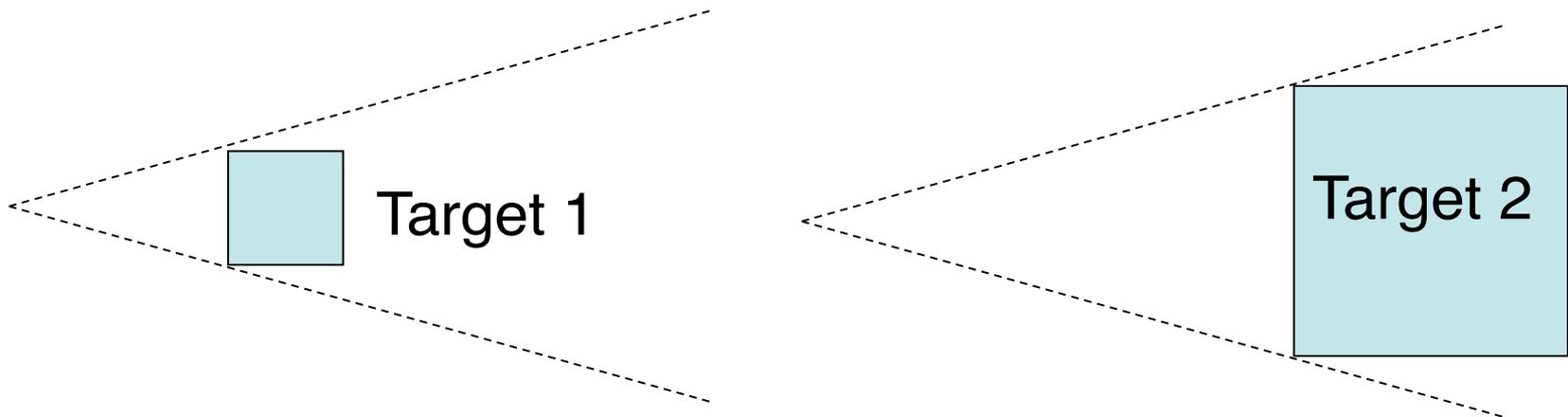
Model Human Processor (MHP) - Model of human cognition combining these theories

Fitts' Law Intuition

Time depends on *relative precision* (d/s)

Time is not limited by motor activity of moving your arm / hand, but rather by the cognitive activity of keeping on track

In below example, time will be the same because the ratio d/s is the same



Fitts' Law

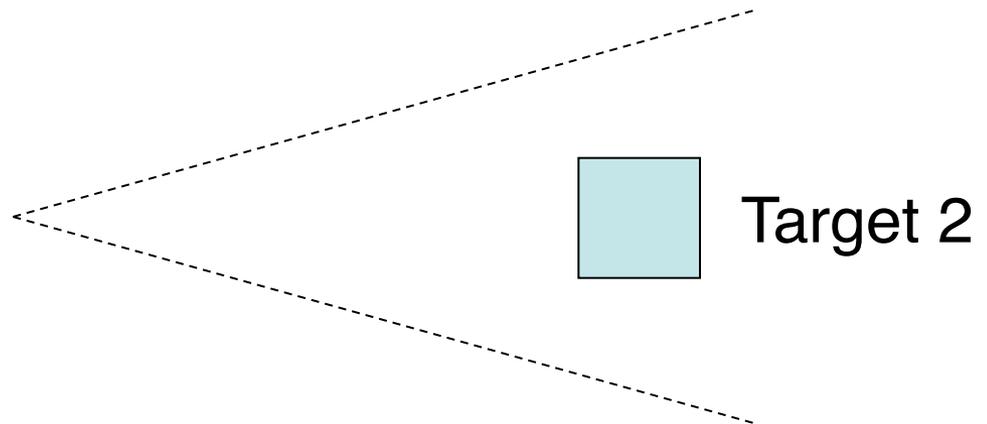
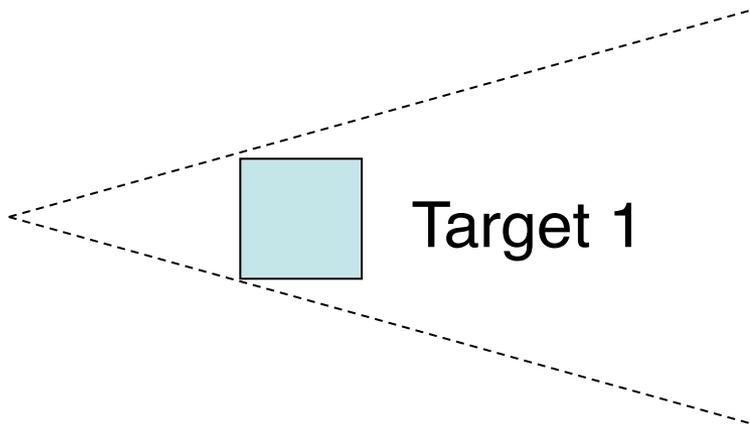
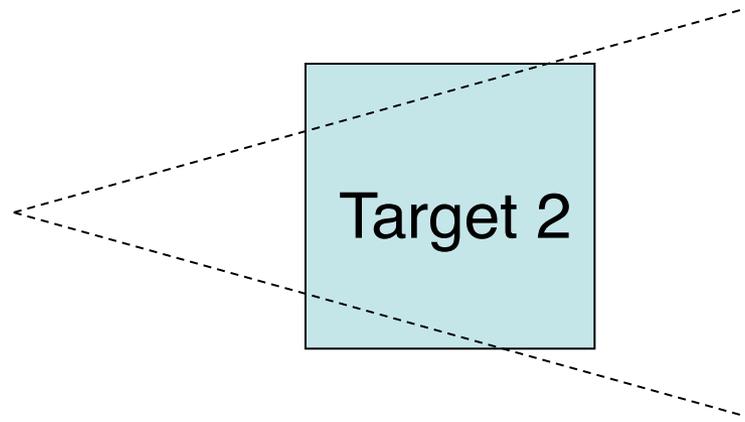
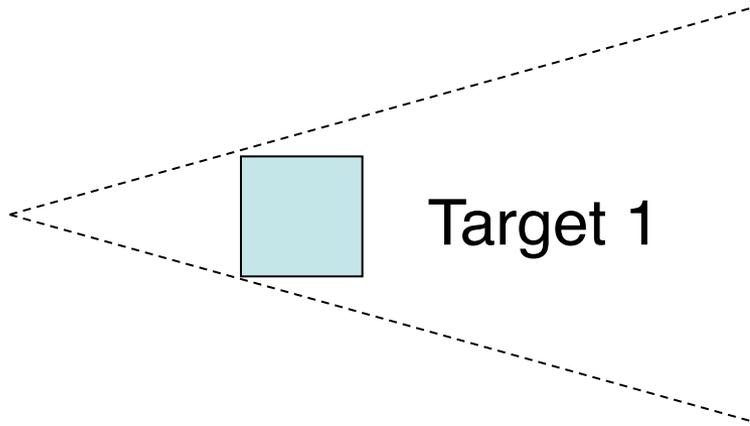
Models movement time for selection

Movement time for a rehearsed task

- Increases with distance to target (d)
- Decreases with width of target (s)
- Depends only on relative precision (d/s), assuming target is within arms reach

First demonstrated for tapping with finger (Fitts 1954), later extrapolated to mouse and other input devices

Fitts' Law Examples



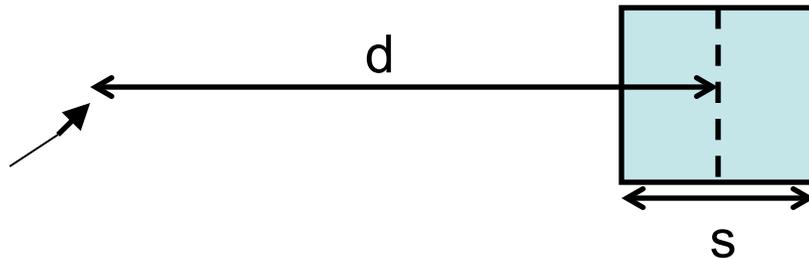
Fitts' Law Equation

$$T_{\text{msec}} = a + b \log_2 (d/s + 1)$$

a, b = empirically-derived constants

d = distance, s = width of target

ID (Index of Difficulty) = $\log_2 (d/s + 1)$



Determining a,b Constants

Conduct experiments varying d,s but keeping everything else the same

Measure execution time, error rate, accuracy

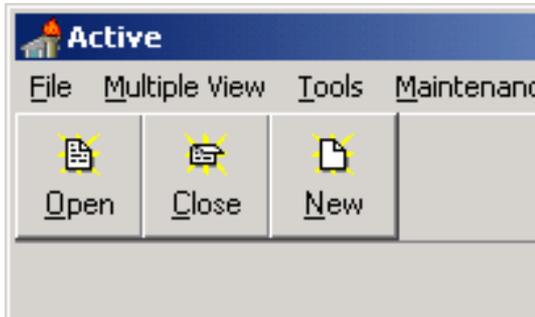
Exclude erroneous data

Perform linear regression

Fitts in Practice

Microsoft Toolbars allow you to either keep or remove the labels under Toolbar buttons

According to Fitts' Law, which is more efficient?



Fitts in Practice



You have a toolbar with 16 icons, each with dimensions of 16x16

Without moving the array from the left edge of the screen, or changing the size of the icons, how can you make this more efficient?



Fitts in Practice

Answer: Line up all 16 icons on the left hand edge of the screen

Make sure that each button can be activated up the last pixel on the left hand edge

Why? Because you cannot move your mouse off of the screen, the effective width s is infinite

Fitts in Practice

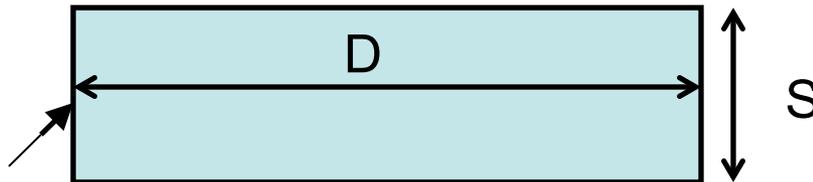


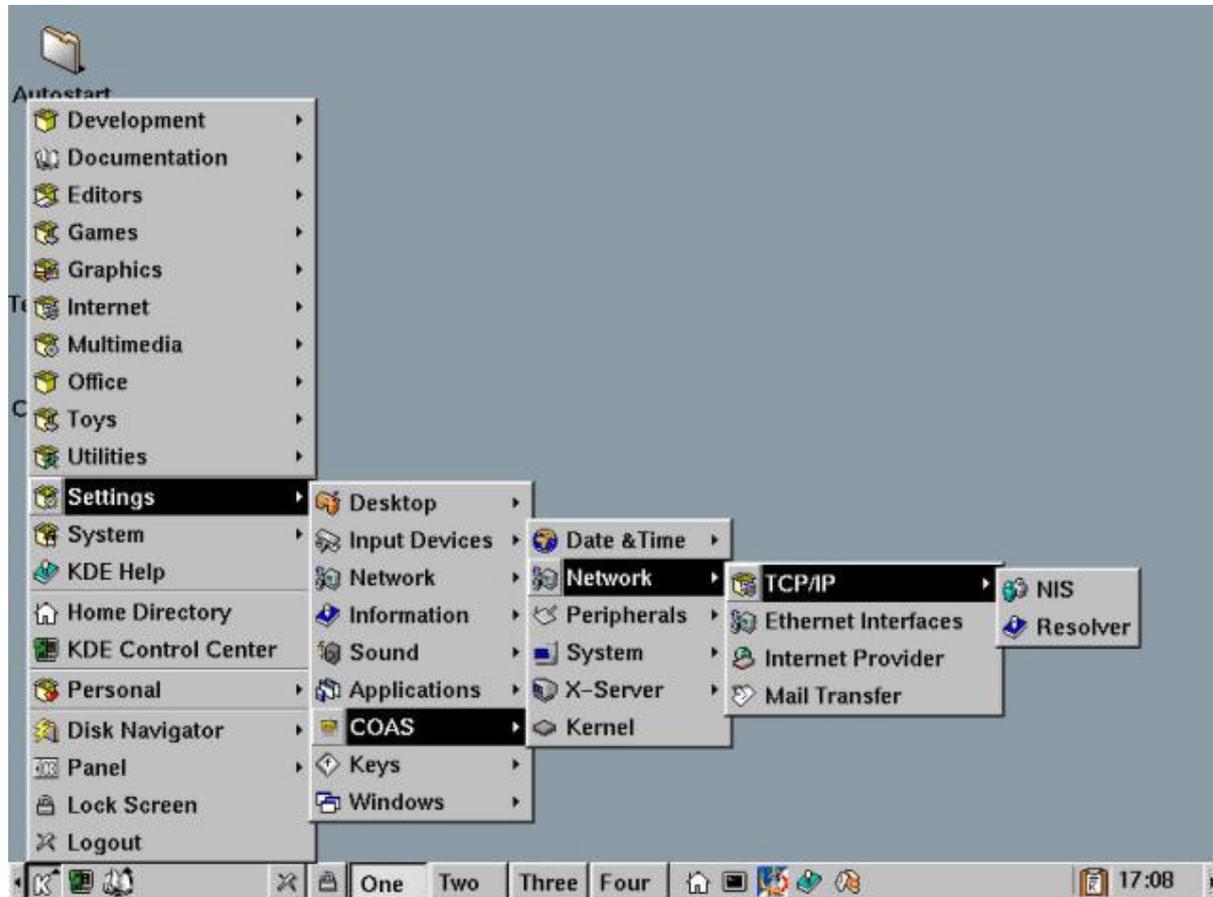
Steering Law

Applies same principles to *steering* through a tunnel (Accot, Zhai 1997)

Must keep the pointer within the boundaries throughout, not only at the target

Fitts' Law used for pointing, Steering Law used for drawing





Steering Law Equation

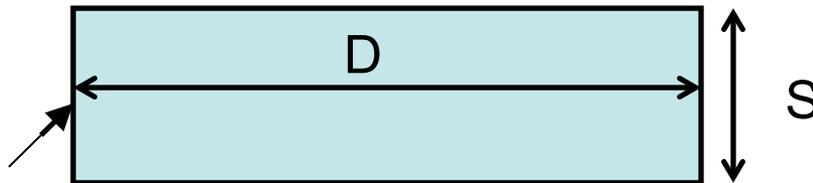
$$T_{\text{msec}} = a + b (d/s)$$

a, b = empirically-derived constants

d = distance, s = width of tunnel

ID (Index of Difficulty) = (d/s)

Index of Difficulty now *linear*, not logarithmic
(i.e. steering is more difficult than pointing)



Keystroke-Level Model (KLM)

Cognitive Analysis (KLM, GOMS, etc.)

Walk through a task, listing the actions needed to complete it

Use heuristics to insert “thinking” operators (for example, place M’ s in front of all P’ s that select a command)

- These can be different for different UI styles

Based on estimates for each operator, calculate the amount of time required to complete the task

KLM Operators

K Press a key or button

P Point to a target on the display

H Home hands on input device

D Draw a line segment

M Mentally prepare for an action

R (system response time)

Example: Replacing a Word

Description	Operation	Time (sec)
Reach for mouse	H[mouse]	0.40
Move pointer to "Replace" button	P[menu item]	1.10
Click on "Replace" command	K[mouse]	0.20
Home on keyboard	H[keyboard]	0.40
Specify word to be replaced	M4K[word]	2.15
Reach for mouse	H[mouse]	0.40
Point to correct field	P[field]	1.10
Click on field	K[mouse]	0.20
Home on keyboard	H[keyboard]	0.40
Type new word	M4K[word]	2.15
Reach for mouse	H[mouse]	0.40
Move pointer on Replace-all	P[replace-all]	1.10
Click on field	K[mouse]	0.20
Total		10.2

According to this KLM model, it takes 10.2 seconds to accomplish this task.

Operator Estimates

Keystroke determined by typing speed

0.28s for average typist (40 wpm), 0.08s for fast typist (155 wpm), 1.20s for worst typist

Pointing determined by Fitts' Law (or general approximation)

$$T = a + b \log (d/s + 1) \text{ OR}$$

$$T = 1.1s$$

Drawing determined by Steering Law

$$T = a + b (d/s)$$

Operator Estimates

Homing estimated by measurement

$T = 0.36\text{s}$ (between keyboard and mouse)

Mental prep estimated by measurement

$T = 1.35\text{s}$

(estimated by taking the total task time,
subtracting physical operator time, and dividing
by the number of M operations)

Heuristics for adding M' s

Basic idea: Put an **M** before each step requiring access of a “chunk” from long-term memory

Insert **M**' s before each sequence of Ks and P

$K \rightarrow MK; P \rightarrow MP$

Remove **M**' s in the middle of typing a word or string

$MKMKMK \rightarrow MKKK$

Delete **M**' s within repetitive composite actions (for example, point and click)

$MPMK \rightarrow MPK$

Example: Deleting a Word

Using Shift-Click

M

P [start of word]

K [click]

M

P [end of word]

K [shift]

K [click]

H [to keyboard]

M

K [Del]

Total: $3M + 2P + 4K$
= 7.37 sec

Using Delete

M

P [start of word]

K [click]

H

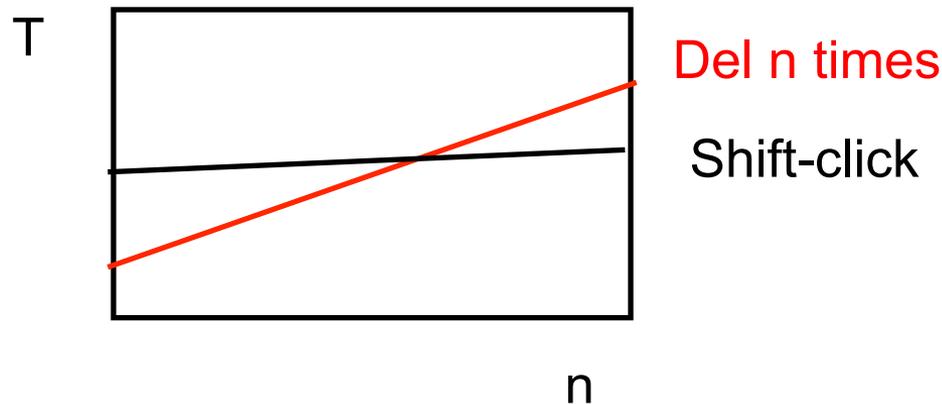
M

K [Del] x n [length of word]

Total: $2M + P + H + (n+1) K$
= 4.44 + 0.28n sec

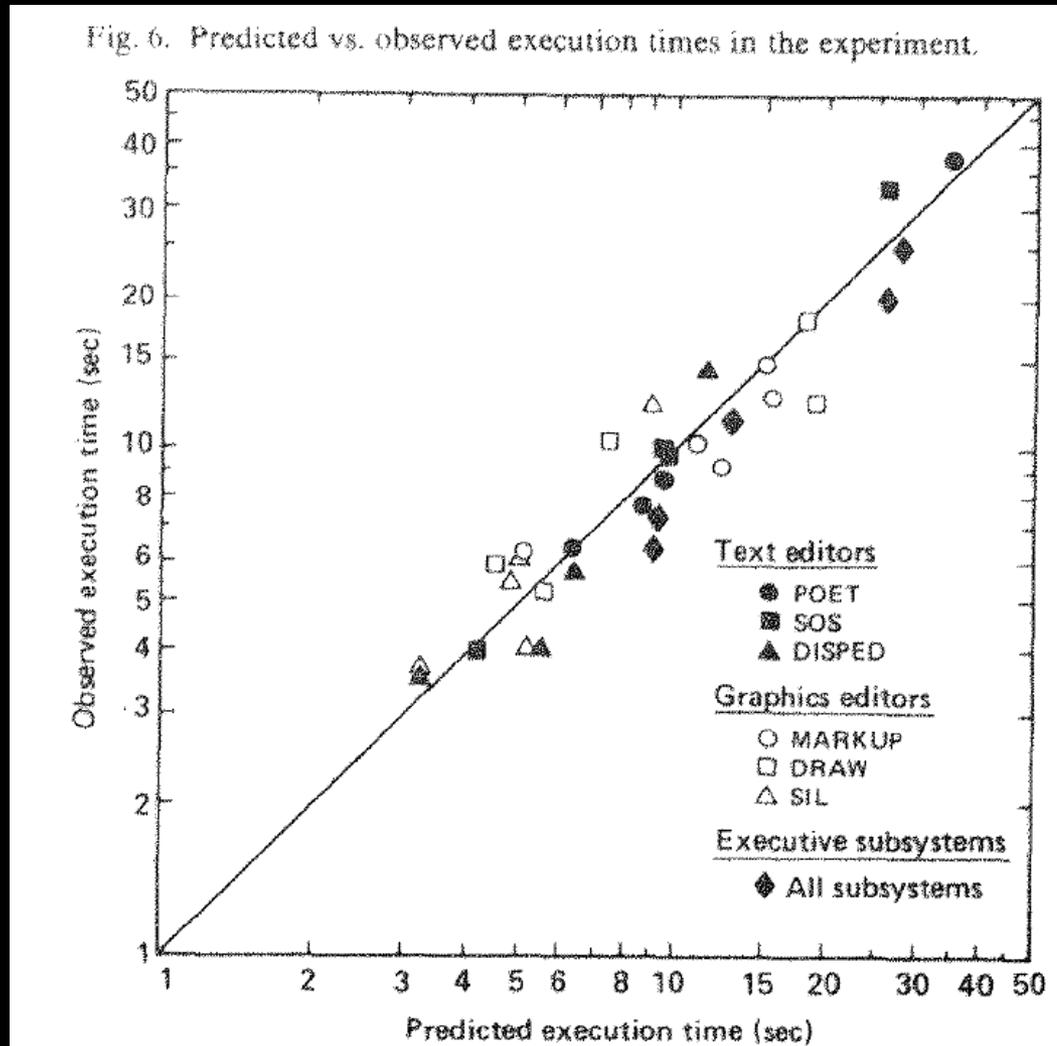
Using KLM

KLM can help evaluate UI designs, interaction methods and trade-offs, using parametric analysis



If common tasks take too long or consist of too many statements, can provide shortcuts

Empirical Validation of KLM



Limitations of KLM

Only applies to expert users doing routine (well-learned) tasks

Only predicts time - not error rate, memorizability, learnability, etc.

Impractical for all but the simplest tasks

Ignores

- Parallel processing, Multi-tasking
- Daydreaming
- Mental workload (working memory limits, fatigue)
- Planning and problem-solving (how to select a method?)

GOMS

GOMS provides a higher-level language for task analysis and UI modeling

Generates a set of quantitative and qualitative predictions based on description of the task and user interface

Provides a hierarchy of goals and methods to achieve them

Different GOMS variants use different terms, operate at various levels of abstraction, and make different simplifying assumptions

Model Human Processor

Model Human Processor

Model of human cognition useful for developing user interfaces

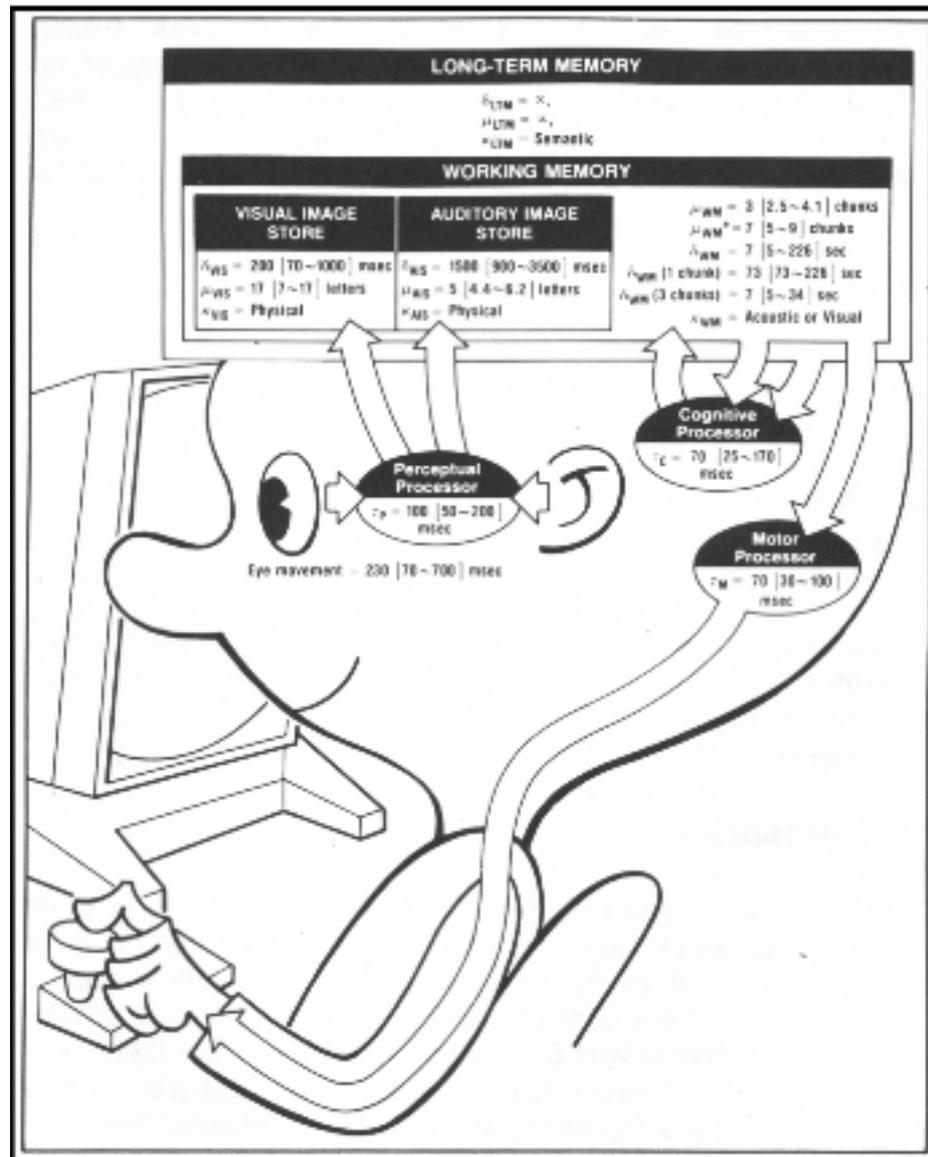
Summary of prior psychology research

Not an exact model of how the brain operates, but provides a useful approximation for understanding and estimating certain kinds of actions and reactions

The
Psychology
of
Human-Computer
Interaction

STUART K. CARD
THOMAS P. MORAN
ALLEN NEWELL

IEA



Model Human Processor

Processors

- Perceptual
- Cognitive
- Motor

Memories

- Sensory Image Store
- Working Memory
- Long-term Memory

Principles of Operation

Model Human Processor

“The perceptual system consists of sensors and associated buffer memories... The cognitive system receives symbolically coded information [from the perceptual system] in its working memory, and uses previously stored information from long-term memory to make decisions about how to respond. The motor system carries out the response”

Processors

Perceptual

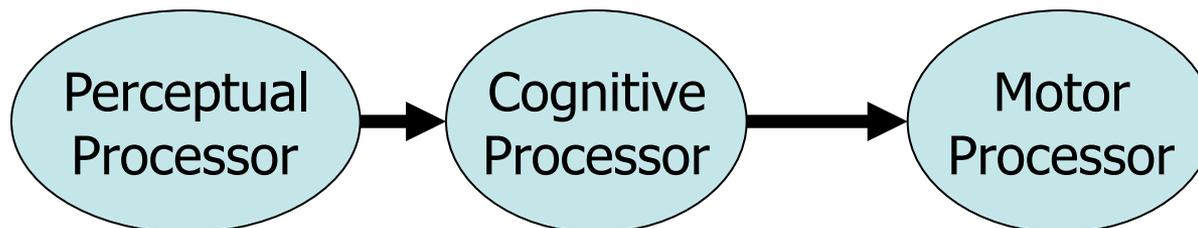
- Processes sensory input
- Populates sensory image store

Motor

- Execute physical actions
- Operates on working memory

Cognitive

- Connects perceptions to actions
- Operates on working and long-term memory



Cycle Time

Each processor has a cycle time

$T_p \sim 100\text{ms}$ [50-200 ms]

- Based on *unit impulse response*
- There is a *quantum* of experience
- Shorter for more intense stimuli

$T_m \sim 70\text{ms}$ [25-170 ms]

- Movement is also not continuous, but consists of a sequence of discrete movements (sometimes preprogrammed - talking, typing, etc.)

Cycle Time

$T_c \sim 70\text{ms}$ [30-100 ms]

- Based on *recognize-act* cycle
- Parallel recognition, serial action
- Can be shorter with lighter task / information loads, and practice

For each of the cycle times, there can be up to 10x difference between the fastest and slowest human beings - cycle times calculated both as nominal amounts and ranges

Power Law of Practice

The time to do a task decreases with practice

$$T_n = T_1 n^{-a}$$

T_n = time to do task on nth iteration

T_1 = time to do task on first iteration

A = constant (0.2 - 0.6)

Applies only to skilled behavior, not to knowledge stored in long-term memory

Memories

Properties of memories:

- Encoding: how things stored
- Size: number of things stored
- Decay time: how long memory lasts (measured as half-life)



Sensory Image 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]

Perceptual memory fades before all of it can be coded and transferred to working memory

Perceptual Fusion

Two stimuli within the same PP cycle
($T_p \sim 100\text{ms}$) appear fused

- Intuition: will be in the same SIS frame

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

Chunking

“Chunk”: unit of perception or memory

Chunking depends on presentation and what you already know

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

MWR CAA OLI BMF BIB

BMW RCA AOL IBM FBI

Working Memory

Holds intermediate products of thinking and coded representations produced by perceptual system

- encoded as acoustic or visual codes
- organized as “chunks” of information
- decay $\sim 7s$ [5-226s]
- decay rate is dependent on the number of chunks being recalled
- *Maintenance rehearsal* can keep chunks in working memory
- *Interference* between similarly coded (primarily acoustic) chunks can reduce chance of retrieval
- size ~ 7 [5-9] chunks

Long-term Memory

Holds the mass of the user's knowledge and experiences

Network of inter-linked chunks, accessed associatively from working memory

- primarily encoded as semantic links
- decay ~ infinite
- size ~ infinite
- fast-read, slow-write

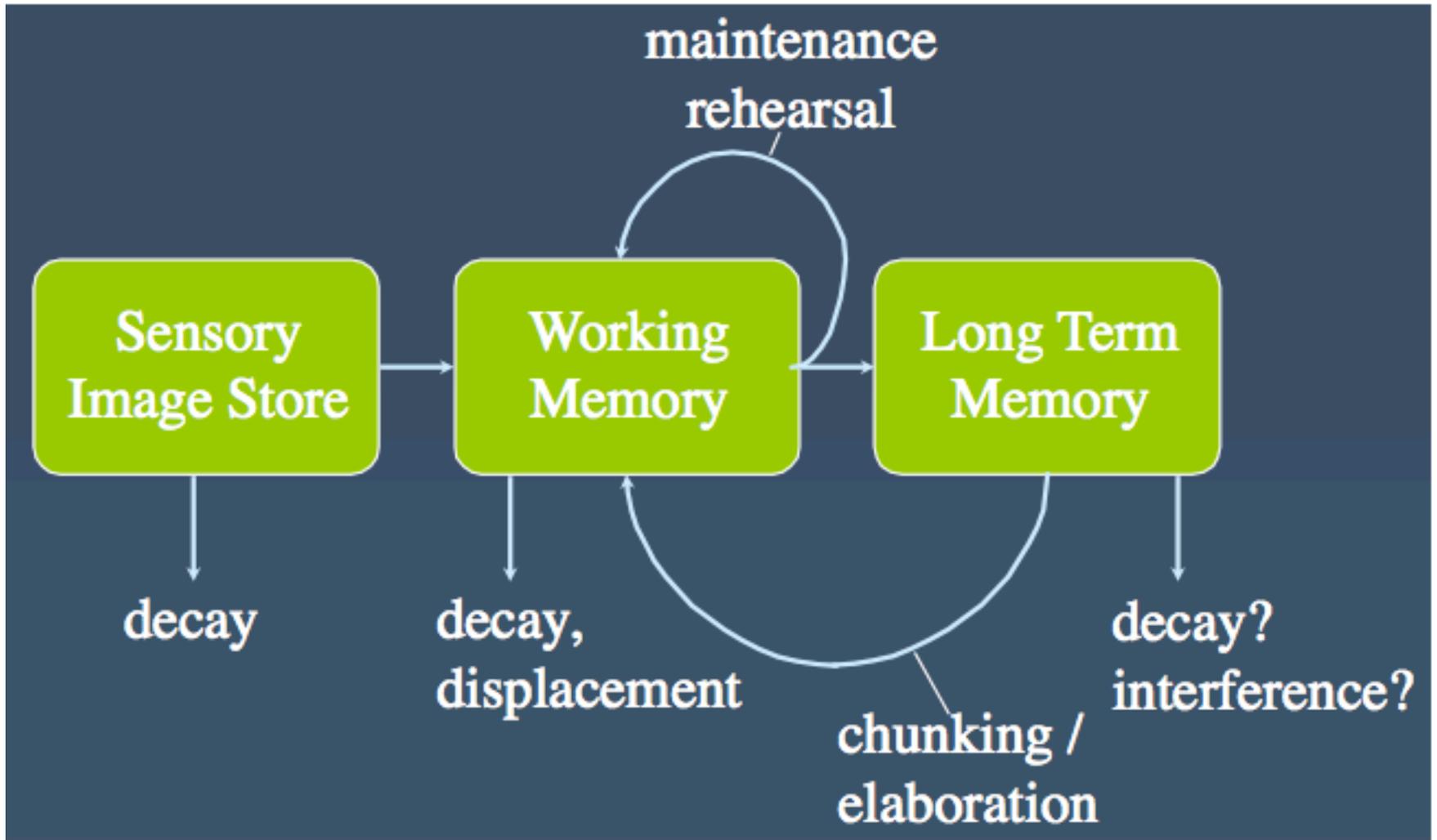
Working on complicated tasks means less time for transferring from working memory to long-term memory

Retrieval from LTM

Retrieval of LTM chunks is based on what other chunks it is associated with (retrieval cues)

Elaborative rehearsal can create more links, increasing chances of retrieval

Interference between similarly coded (semantically similar) can reduce chances of retrieval



maintenance rehearsal

Sensory Image Store

Working Memory

Long Term Memory

decay

decay, displacement

chunking / elaboration

decay? interference?

Uncertainty Principle

Response time RT increases with uncertainty about the judgment or decision to be made; proportionally to the information content of the stimuli

For example, for n equally probably stimuli, each requiring a different response

$$RT = c + d \log_2 (n + 1)$$

Where c, d are constants

For Next Time

Functional Prototype due on Thurs!

Readings / links on Reflective Design