Interaction Design

Chapter 6 (June 6th, 2011, 9am-12pm):
Laws of Interaction Design

Laws of Interaction Design

- Moore’s law
- Buxton’s law
- Fitts’ law
- Steering law
- Guiard’s Kinematic chain model
- Hick’s law
- Murphy’s law

Why laws?

I'll be presenting some of the laws you can rely on when designing interactive systems.

The point is not to follow them blindly but to know they exist and use them as framing for your designs

It's really like usability rules it's good to know them but sometimes you have to break them or just ignore them

Laws of Interaction Design

- Moore’s law
- Buxton’s law
- Fitts’ law
- Steering law
- Guiard’s Kinematic chain model
- Hick’s law
- Law of practice
- Murphy’s law

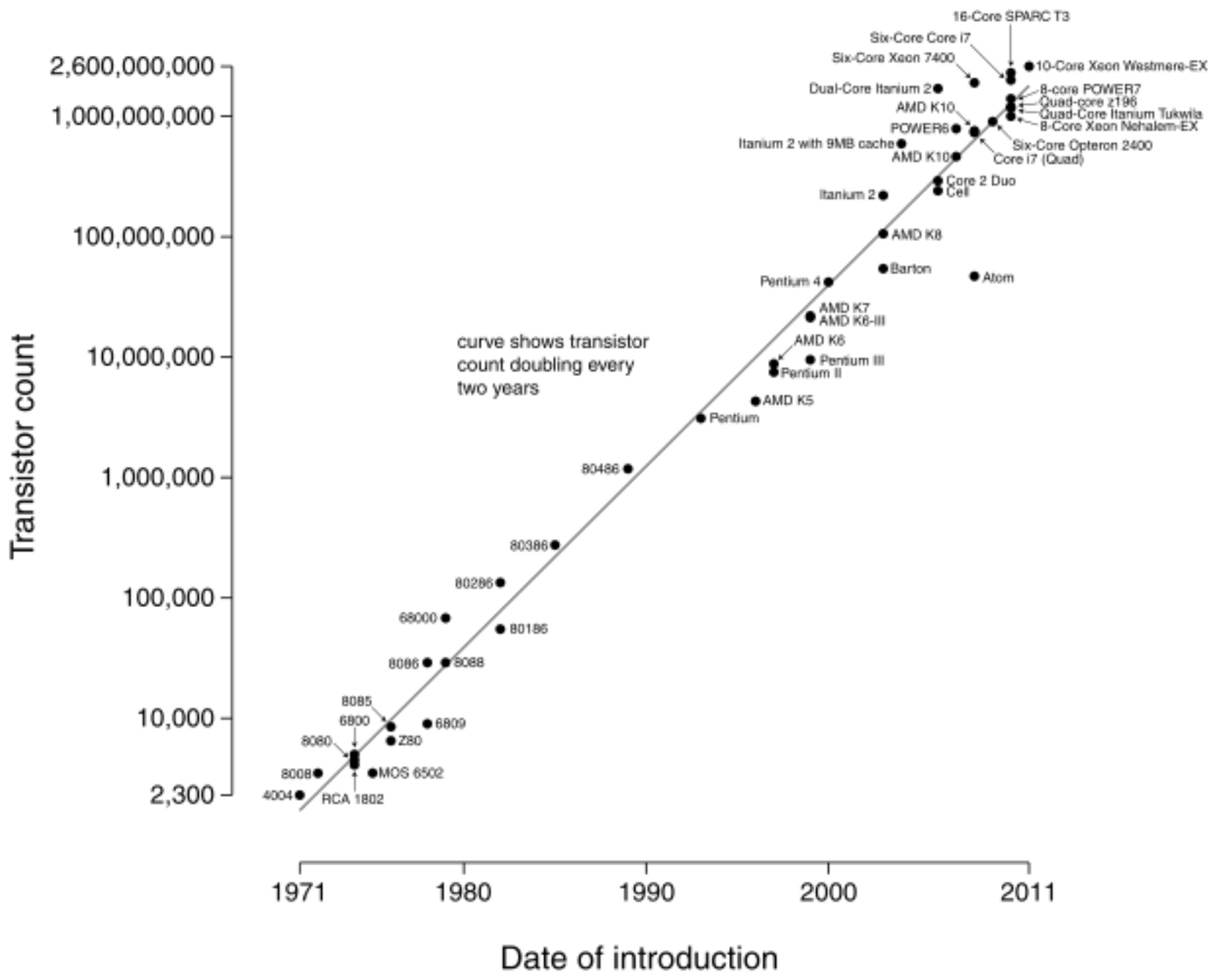
Moore’s law

“The complexity for minimum component costs has increased at a rate of roughly a **factor of two per year***… Certainly over the short term this rate can be expected to continue, if not to increase. Over the longer term, the rate of increase is a bit more uncertain, although there is no reason to believe it will not remain nearly constant for at least 10 years. That means by 1975, the number of components per integrated circuit for minimum cost will be 65,000. I believe that such a large circuit can be built on a single wafer.”*

[Moore, Gordon E. (1965). "Cramming more components onto integrated circuits". Electronics, Volume 38, Number 8, April 19, 1965.]

Source: http://en.wikipedia.org/wiki/File:Transistor_Count_and_Moore%27s_Law_-_2011.svg

Moore’s law illustration



Moore’s law implications

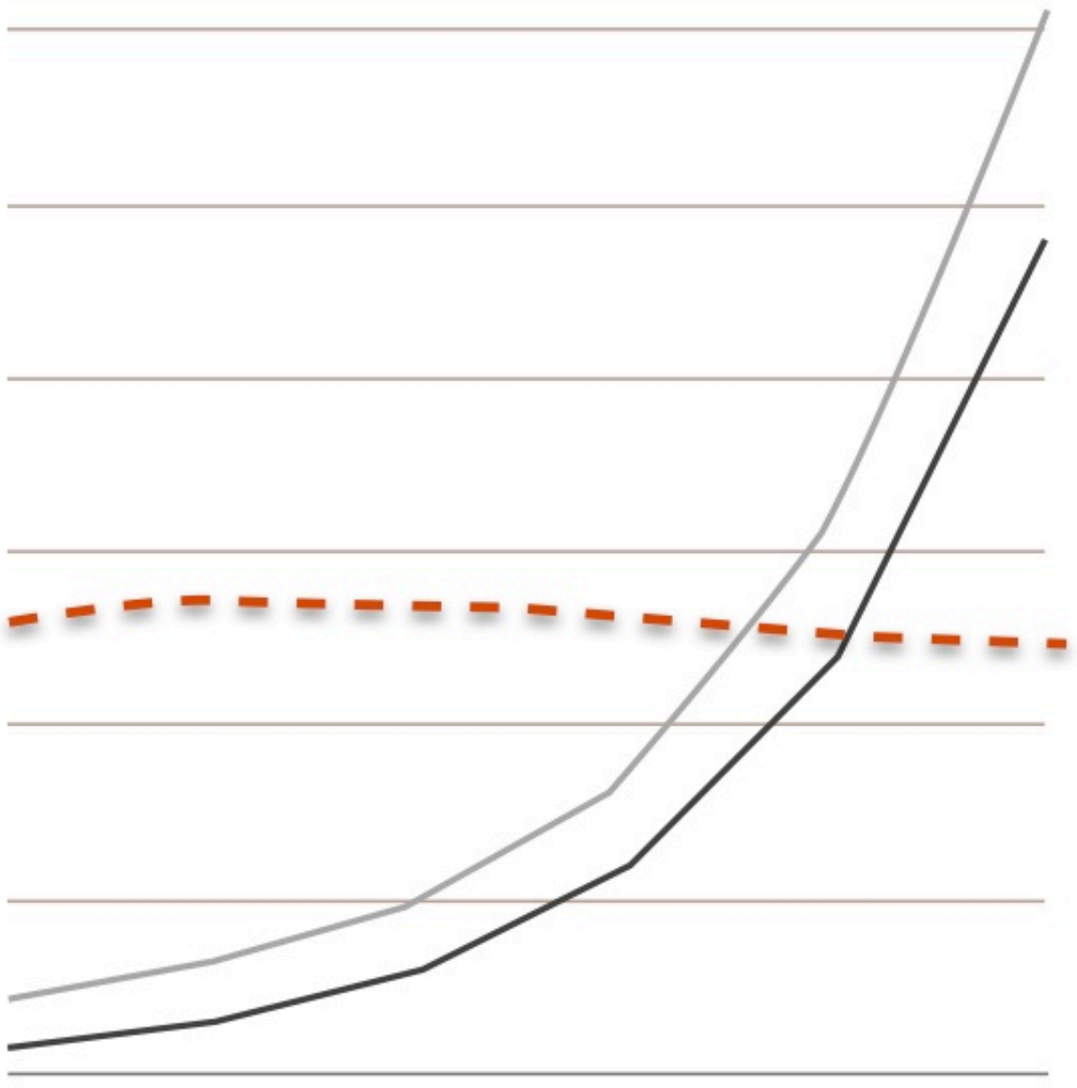
Don’t worry too much about:

- computing power
- storage capacity
- screen resolution
- device size
- weight
- battery life (?)

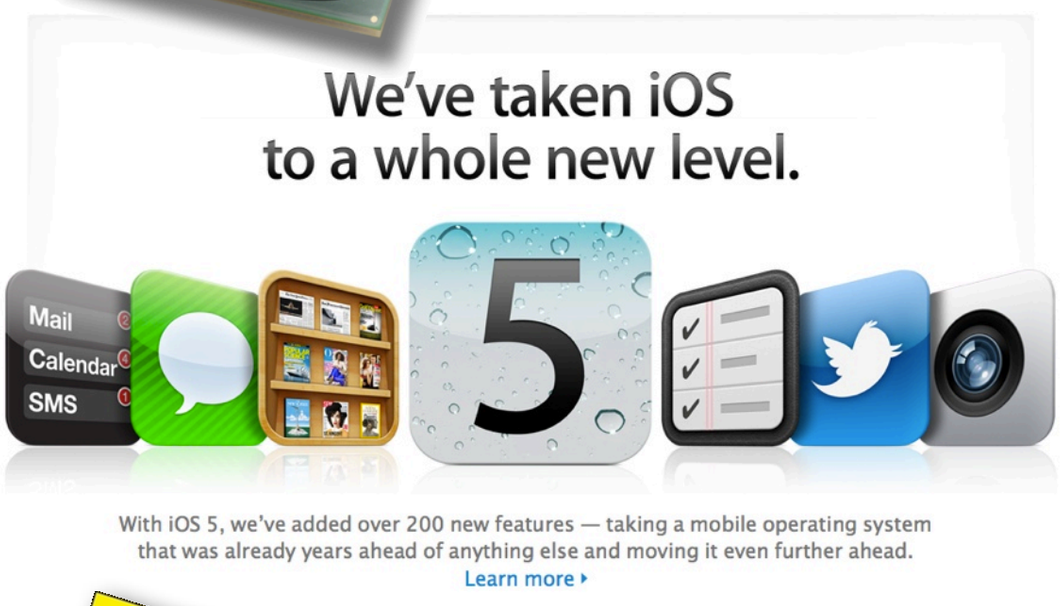
Laws of Interaction Design

- Moore’s law
- Buxton’s law
- Fitts’ law
- Steering law
- Guiard’s Kinematic chain model
- Hick’s law
- Law of practice
- Murphy’s law

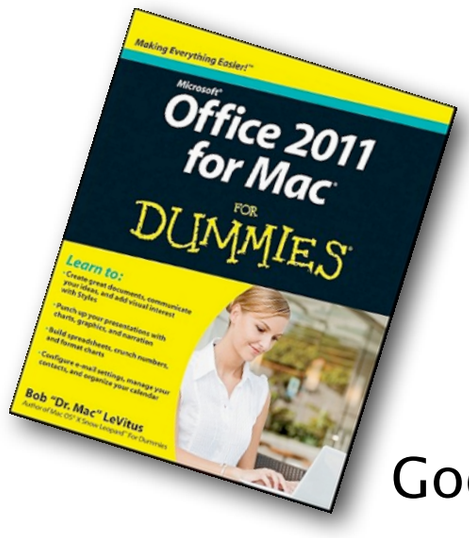
Buxton's law



Moore's law



Buxton's law



God's law

The first line (gray) shows Moore's law we've just talked about.
The second line shows the so called Buxton's law: the number of features increased at each software release.
The last line (dashed), called God's law by Bill Buxton shows the evolution of the human capabilities:

Our intellectual and physical capabilities did not change much in 50 years and may not change much in the years to come.

So how do we cope with this?

Laws of Interaction Design

- Moore’s law
- Buxton’s law
- Fitts’ law
- Steering law
- Guiard’s Kinematic chain model
- Hick’s law
- Law of practice
- Murphy’s law

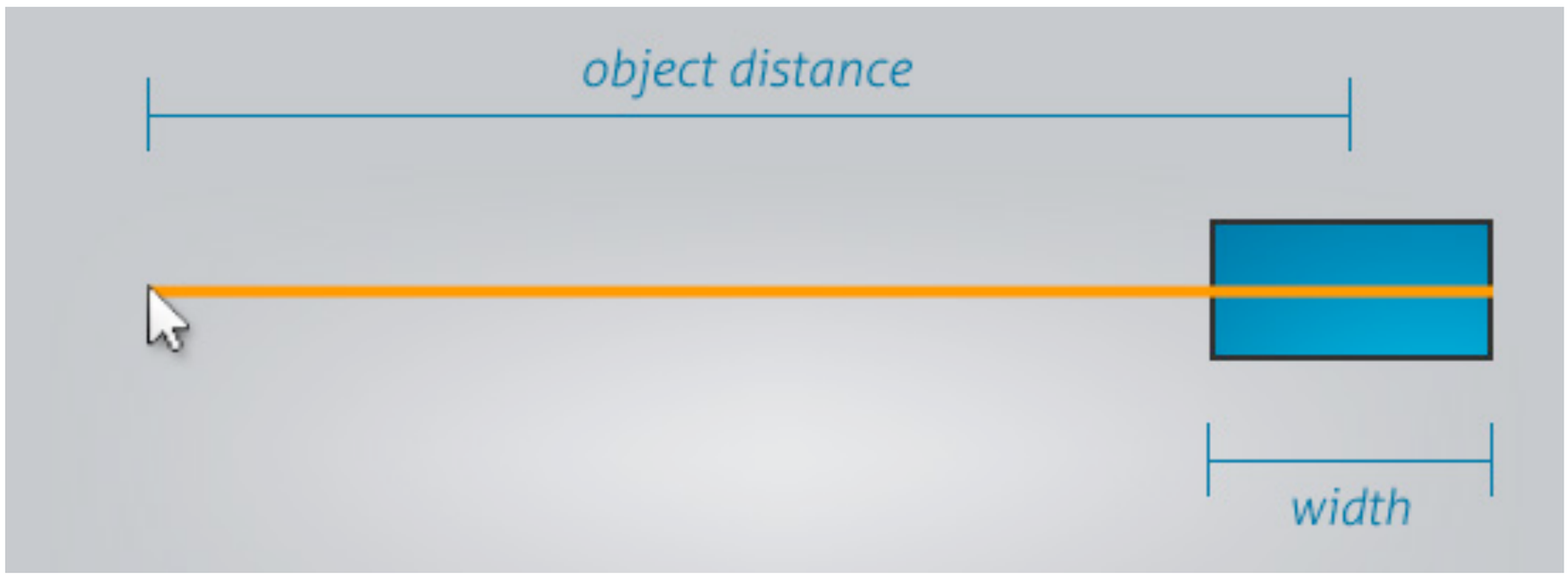
So let’s dig into the physical capabilities of humans

More from
http://sixrevisions.com/usabilityaccessibility/improving-usability-with-fitts-law/
+ from Andreas’ slides

Illustration from http://sixrevisions.com/usabilityaccessibility/improving-usability-with-fitts-law/

Fitts' law

The time to acquire a target is a function of the distance to and width of the target.



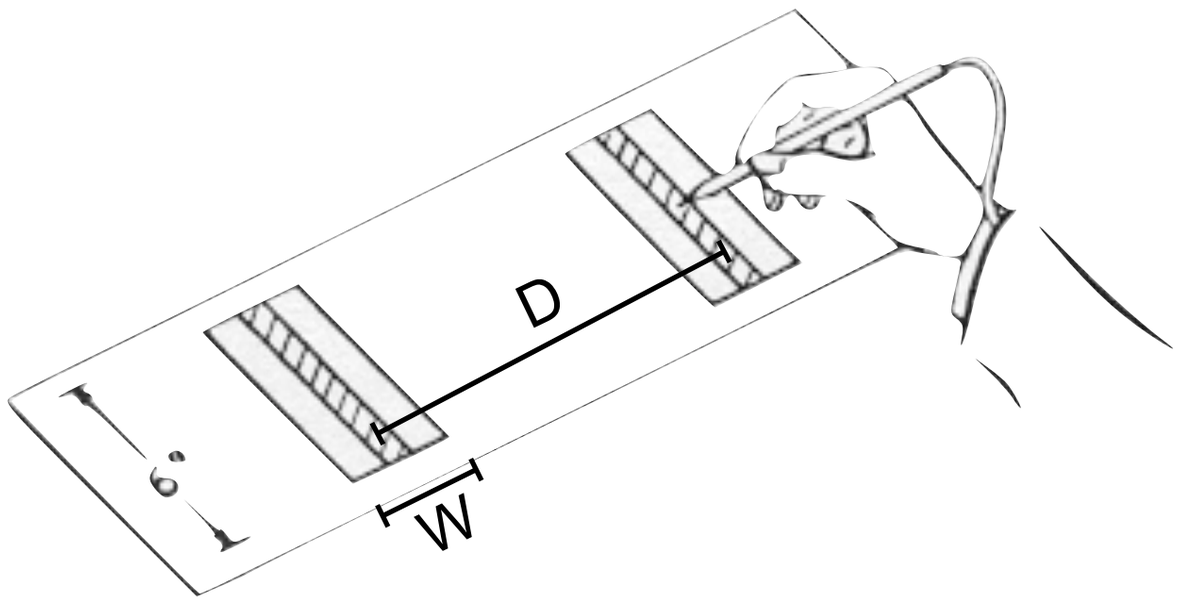
While at first glance, this law might seem patently obvious, it is one of the most ignored principles in design. Fitts' law (properly per American English rules, spelled "Fitts's Law," though rarely done so) dictates the Macintosh pull-down menu acquisition should be approximately five times faster than Windows menu acquisition, and this is proven out.

Fitts' law dictates that the windows task bar will constantly and unnecessarily get in people's way, and this is proven out. Fitts' law indicates that the most quickly accessed targets on any computer display are the four corners of the screen, because of their pinning action, and yet, for years, they seemed to be avoided at all costs by designers.

Use large objects for important functions (Big buttons are faster).

Use the pinning actions of the sides, bottom, top, and corners of your display: A single-row toolbar with tool icons that "bleed" into the edges of the display will be many times faster than a double row of icons with a carefully-applied one-pixel non-clickable edge between the tools and the side of the display.

Fitts' law



Time ⟶

$$T = a + b \cdot log_2 (2 \frac{D}{W})$$

Distance ↓ D

W ↑ Width

Coefficients
a: Intercept
b: Slope

Small story about Fitts and his experiments...

- **time** is the amount of time required to complete the movement
- **a** and **b** are empirically determined regression coefficients, which is basically a fancy way of stating they are values gained from direct observation that build a slope.
- **distance** is a measurement from the starting point to the end point (target object)
- **width** is the width of the target object

Fitts task:

http://www.youtube.com/watch?v=kly2QA1bFc8

Fitts' law

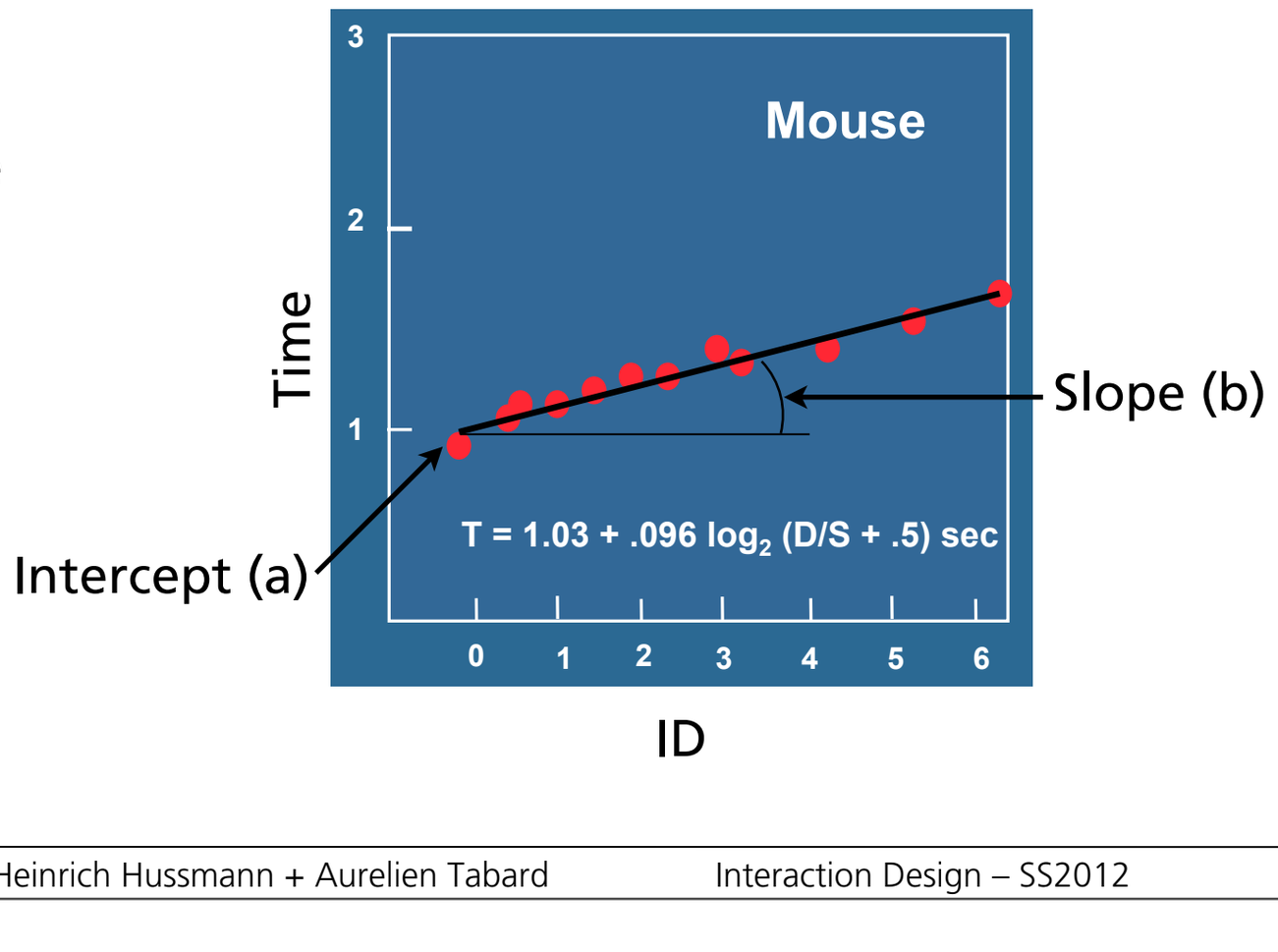
Movement Time (MT) ⟵ $MT = a + b \log_2 (\frac{D}{W} + 1)$ ⟶ Index of Difficulty (ID)

Task difficulty is analogous to information,

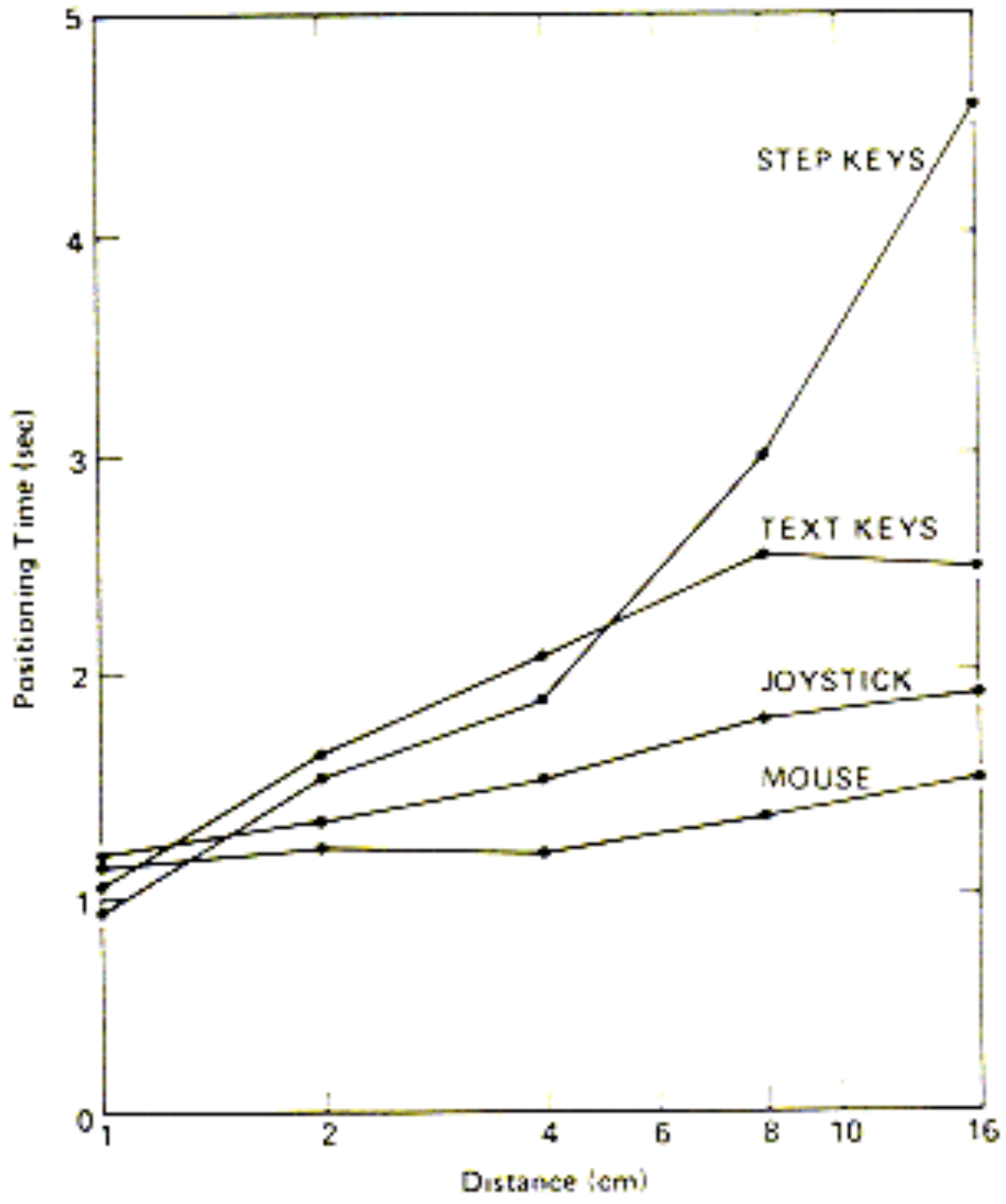
- Execution interpreted as human rate of information processing (cf Shannon inf. theory).

Index of Performance (IP) = ID/MT (bits/s)

- Time to position mouse proportional to Fitts' Index of Difficulty ID.
  [i.e. how well can the muscles direct the input device]
- Therefore speed limit is in the eye-hand system, not the mouse.



50 years of data



50 years of data

http://www.designinginteractions.com/interviews/StuCard

50 years of data

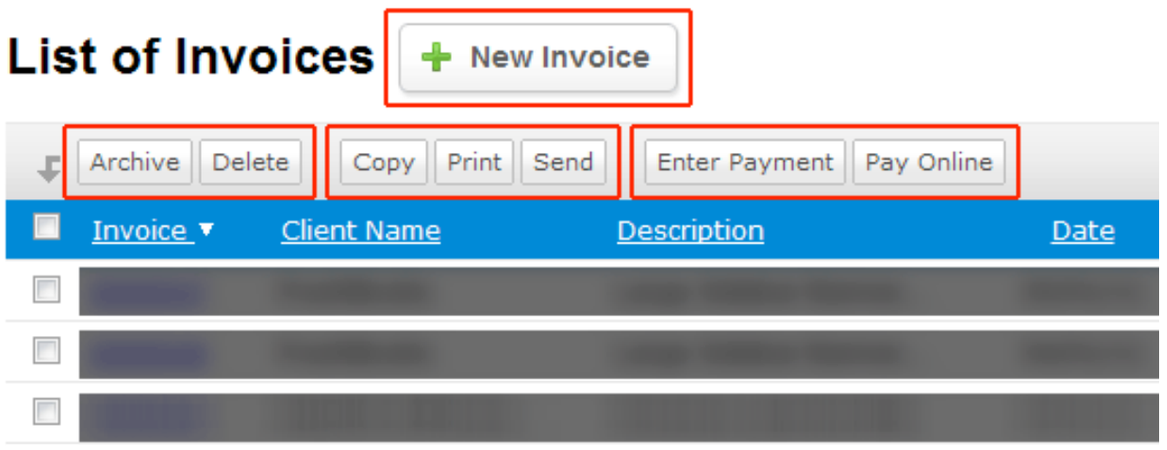
Device	Study	IP (bits/s)
Hand	Fitts (1954)	10.6
Mouse	Card, English, & Burr (1978)	10.4
Joystick	Card, English, & Burr (1978)	5.0
Trackball	Epps (1986)	2.9
Touchpad	Epps (1986)	1.6
Eyetracker	Ware & Mikaelian (1987)	13.7

[MacKenzie, 1992] Fitts' Law as a research and design tool in human computer interaction.

Illustration from http://particletree.com/features/visualizing-fittss-law/

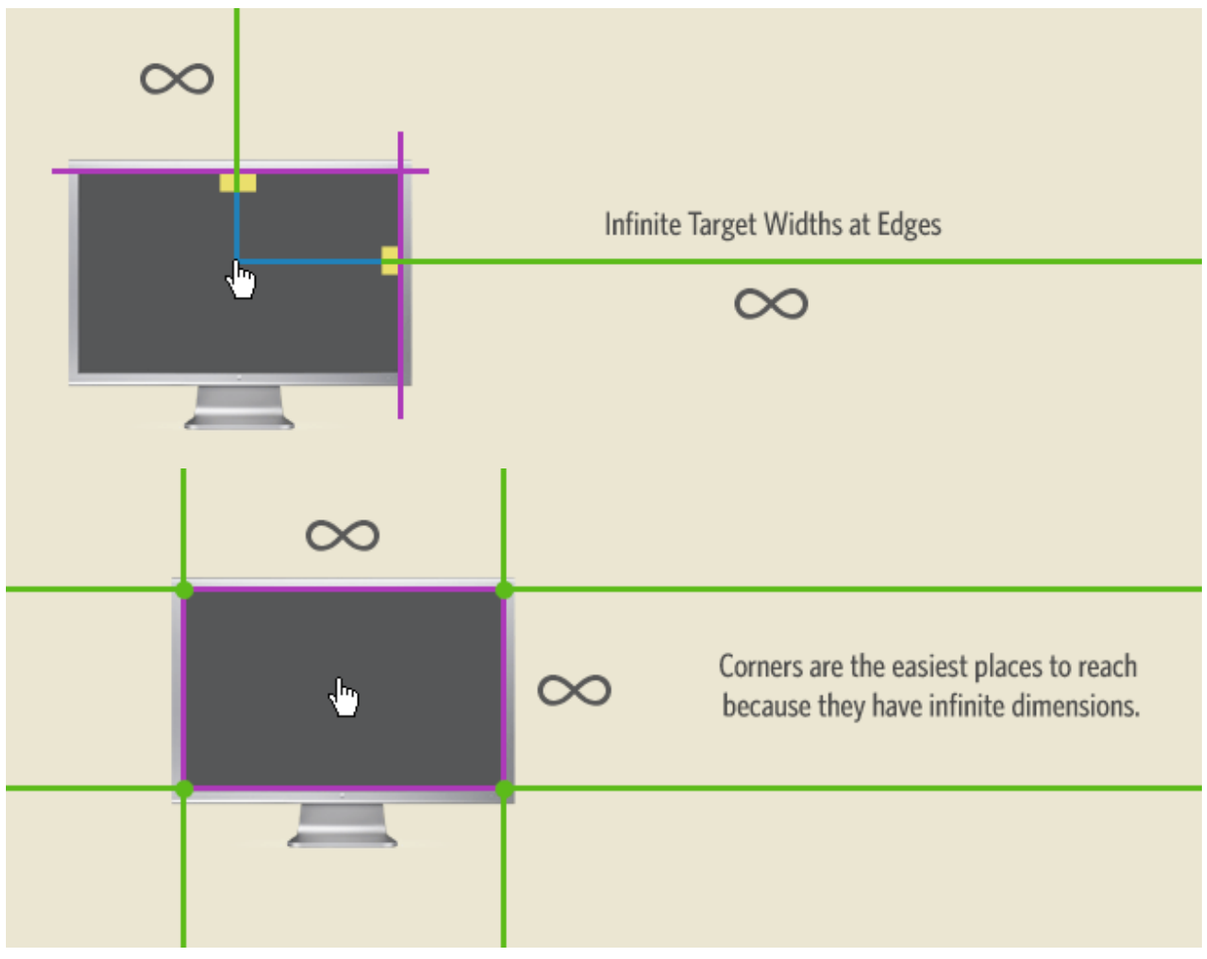
Implications of Fitts' law

Larger targets are easier to hit
-> maximize button size



Movement time increases (logarithmically) with distance
-> minimize distances
-> no movement is even better!

Infinite targets:
-> leverage screen borders
-> leverage corners



Leverage screen borders:

Mac's top menu,
Windows 95: Missed by a pixel
Windows XP: Good to the last drop

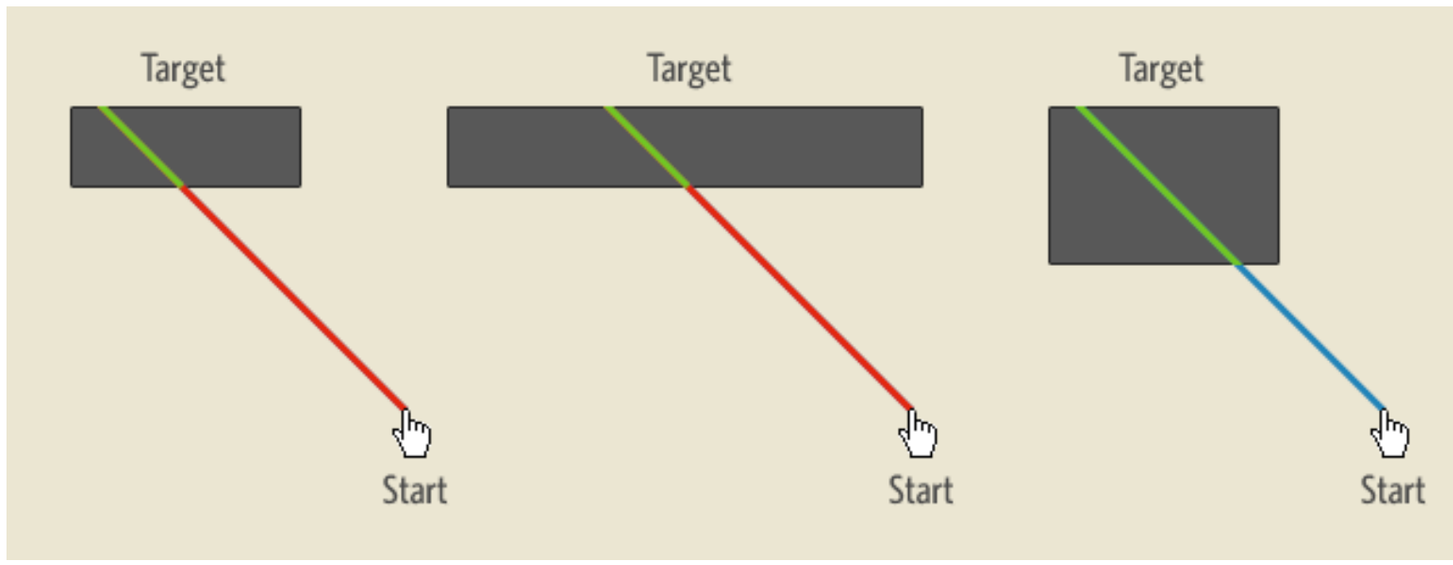
Leverage corners:

- Active corners on MacOSX + Apple menu
- Windows Start

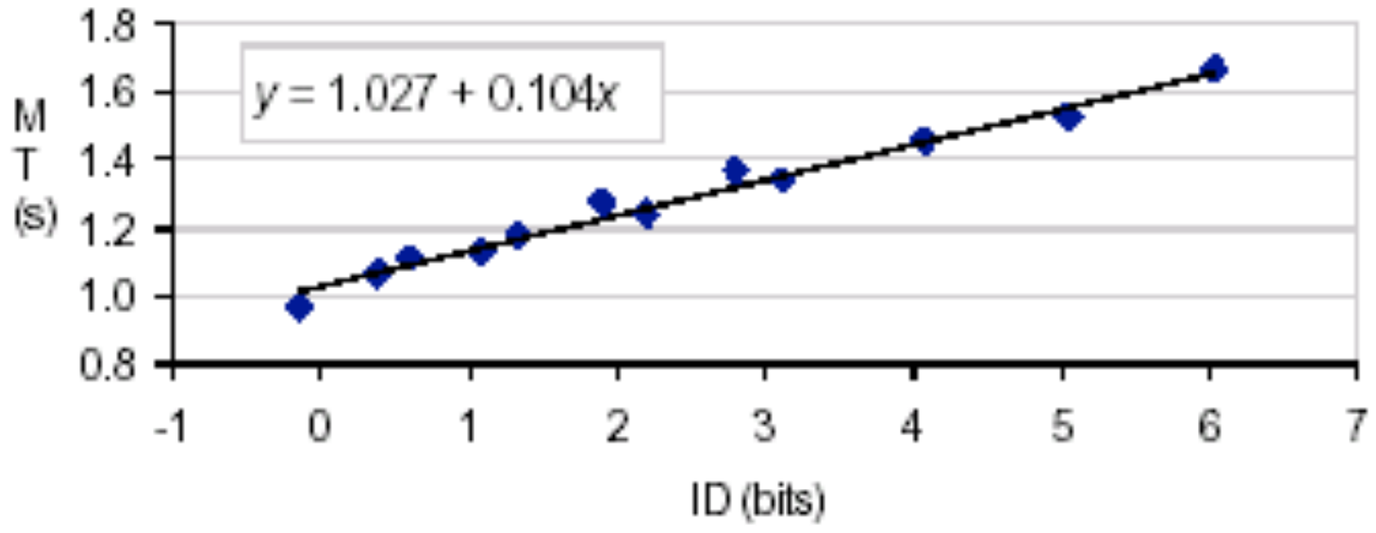
Illustration from http://particletree.com/features/visualizing-fittss-law/

Bigger Is Not Always Better

Movement direction to target



Logarithmic improvements with size



MacKenzie revaluation of Card's Fitts' Experiment for text selection

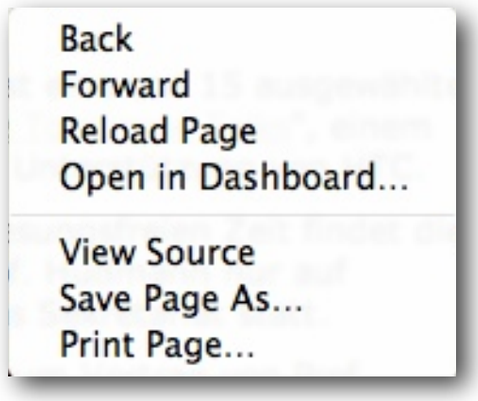
Fitts' law is a binary logarithm. This means that the predicted results of the usability of an object runs along a curve, not a straight line. At the scale of UI design, this means that a very small object will become significantly easier to click when given a 20% size increase, while a very large object will not share the same boost in usability when given the same 20% boost in size.

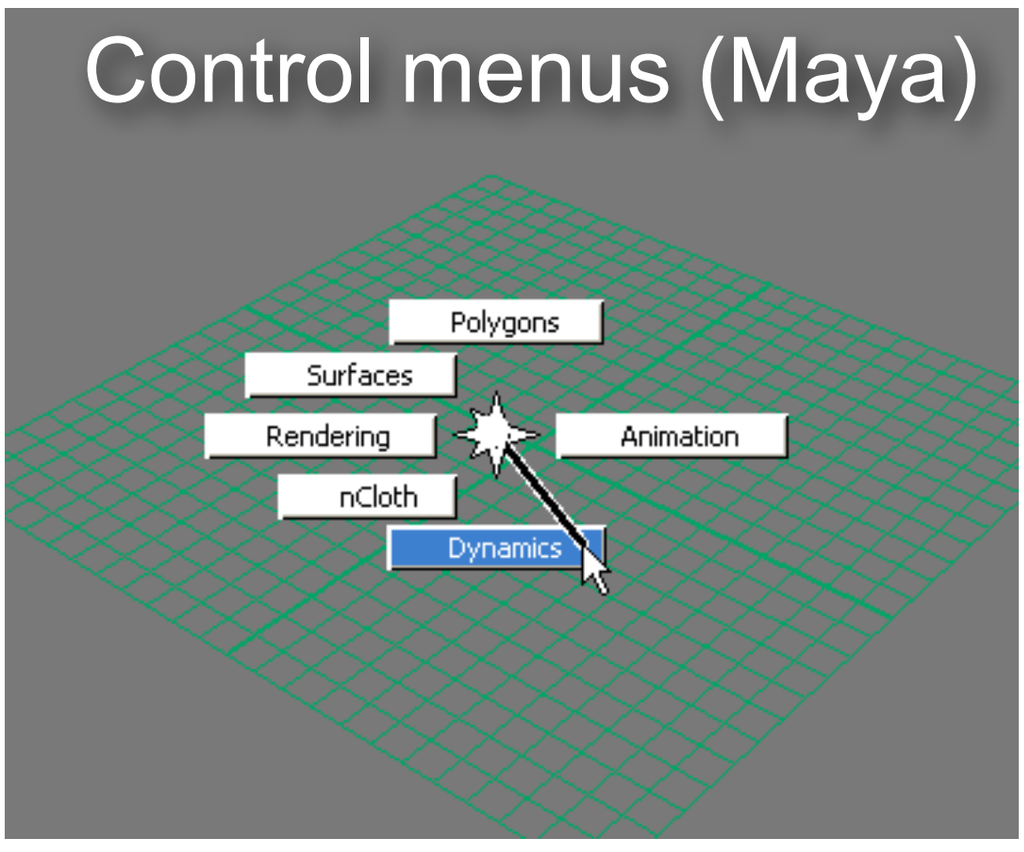
Fitts' law application to menu selection

Imagine a pop-up menu with 8 entries

Compare linear vs. pie menu

Selection time for each entry





Quiz

Microsoft Toolbars offer the user the option of displaying a label below each tool. Name at least one reason why labeled tools can be accessed faster. (Assume, for this, that the user knows the tool and does not need the label just simply to identify the tool.)

1. You have a palette of tools in a graphics application that consists of a matrix of 16x16-pixel icons laid out as a 2x8 array that lies along the left-hand edge of the screen. Without moving the array from the left-hand side of the screen or changing the size of the icons, what steps can you take to decrease the time necessary to access the average tool?

2. A right-handed user is known to be within 10 pixels of the exact center of a large, 1600 X 1200 screen. You will place a single-pixel target on the screen that the user must point to exactly. List the five pixel locations on the screen that the user can access fastest. For extra credit, list them in order from fastest to slowest access.

3. Microsoft offers a Taskbar which can be oriented along the top, side or bottom of the screen, enabling users to get to hidden windows and applications. This Taskbar may either be hidden or constantly displayed. Describe at least two reasons why the method of triggering an auto-hidden Microsoft Taskbar is grossly inefficient.

4. Explain why a Macintosh pull-down menu can be accessed at least five times faster than a typical Windows pull-down menu. For extra credit, suggest at least two reasons why Microsoft made such an apparently stupid decision.

5. What is the bottleneck in hierarchical menus and what techniques could make that bottleneck less of a problem?

6. Name at least one advantage circular popup menus have over standard, linear popup menus.

7. What can you do to linear popup menus to better balance access time for all items?

8. The industrial designers let loose on the Mac have screwed up most of the keyboards by cutting their function keys in half so the total depth of the keyboard was reduced by half a key. Why was this incredibly stupid?

The Answers

Let's start out with a preview of the answer to question 10, Fitts' law, since all the others revolve around it. From First Principles of Design:

- Fitts' Law: The time to acquire a target is a function of the distance to and size of the target.

This little bit of obviousness is so often ignored, I sometimes wonder if it is on purpose. Usually, though, it is merely a leading indicator of overall ignorance, amplified by superstition and unsullied by facts or study.

Fortunately, readers of AskTog, being as tenacious as they are perspicacious, either know exactly what Fitts' Law is or will before they go to bed tonight.

Fitts' Law (properly spelled "Fitts's," per the rules of American English, though rarely done so) is simple, absolute, and immutable. Let's see how it pertains to the questions:

Question 1

- Microsoft Toolbars offer the user the option of displaying a label below each tool. Name at least one reason why labeled tools can be accessed faster. (Assume, for this, that the user knows the tool and does not need the label just simply to identify the tool.)

Here are two answers. You may have more.

1. The label becomes part of the target. The target is therefore bigger. Bigger targets, all else being equal, can always be acccessed faster. Fitts' Law.
2. When labels are not used, the tool icons crowd together.

At first glance, it might appear advantageous to crowd the icons together, since it results in less distance among targets. However, the task here is not to hop from target to target. Instead, the point of origin when a user decides to access the toolbar will usually be somewhere in the content region, away from all the targets.

When the icons are spread apart, users have a "buffer zone" between icons, where an incorrect acquisition will result in no action. When the targets are crowded together, however, the user has more chance to initiate an unwanted action. To avoid this possibility, non-label users learn to slow way down. (Don't bother to ask them whether they've slowed down. They'll tell you it sped them up. Only the stopwatch knows for sure.)

Another way to make the targets bigger, of course, is to always choose large icons, rather than small. Pity the "power-user" with the 4x4-pixel icons who thinks he's going faster.

Question 2

- You have a palette of tools in a graphics application that consists of a matrix of 16x16-pixel icons laid out as a 2x8 array that lies along the left-hand edge of the screen. Without moving the array from the left-hand side of the screen or changing the size of the icons, what steps can you take to decrease the time necessary to access the average tool?

Two separate steps may be necessary to average tool access time. Both are important.

1. Change the array to 1X16, so all the tools lie along the edge of the screen.

2. Ensure that the user can click on the very first row of pixels along the edge of the screen to select a tool. There should be no buffer zone.

This second step is vital, and is so often ignored.

Remember that Fitts' Law states that access time is a function of distance and target size. If the target size is larger, then the time is reduced. It is reduced for a simple reason: the user need not slow down when approaching the target for fear of overshooting.

Now consider the screen edge. How deep is the target? If it were really only the one pixel it appears, it would be very hard to hit. However, the screen edge is, for all practical purposes, infinitely deep. It doesn't matter how fast that mouse is going when it hits the screen edge, that pointer absolutely will not overshoot. Having to hit a pixel two pixels in from the screen edge takes much longer than hitting the edge itself. Use that edge. It is your friend.

Question 3

- A right-handed user is known to be within 10 pixels of the exact center of a large, 1600 X 1200 screen. You will place a single-pixel target on the screen that the user must stop upon and point to exactly. List the five pixel locations on the screen that the user can access fastest. For extra credit, list them in order from fastest to slowest access.

No, this is not a trick question. And the first part should be immediately answerable by any interaction designer. The extra credit question is not quite as simple. But first, the locations of the five "magic pixels":

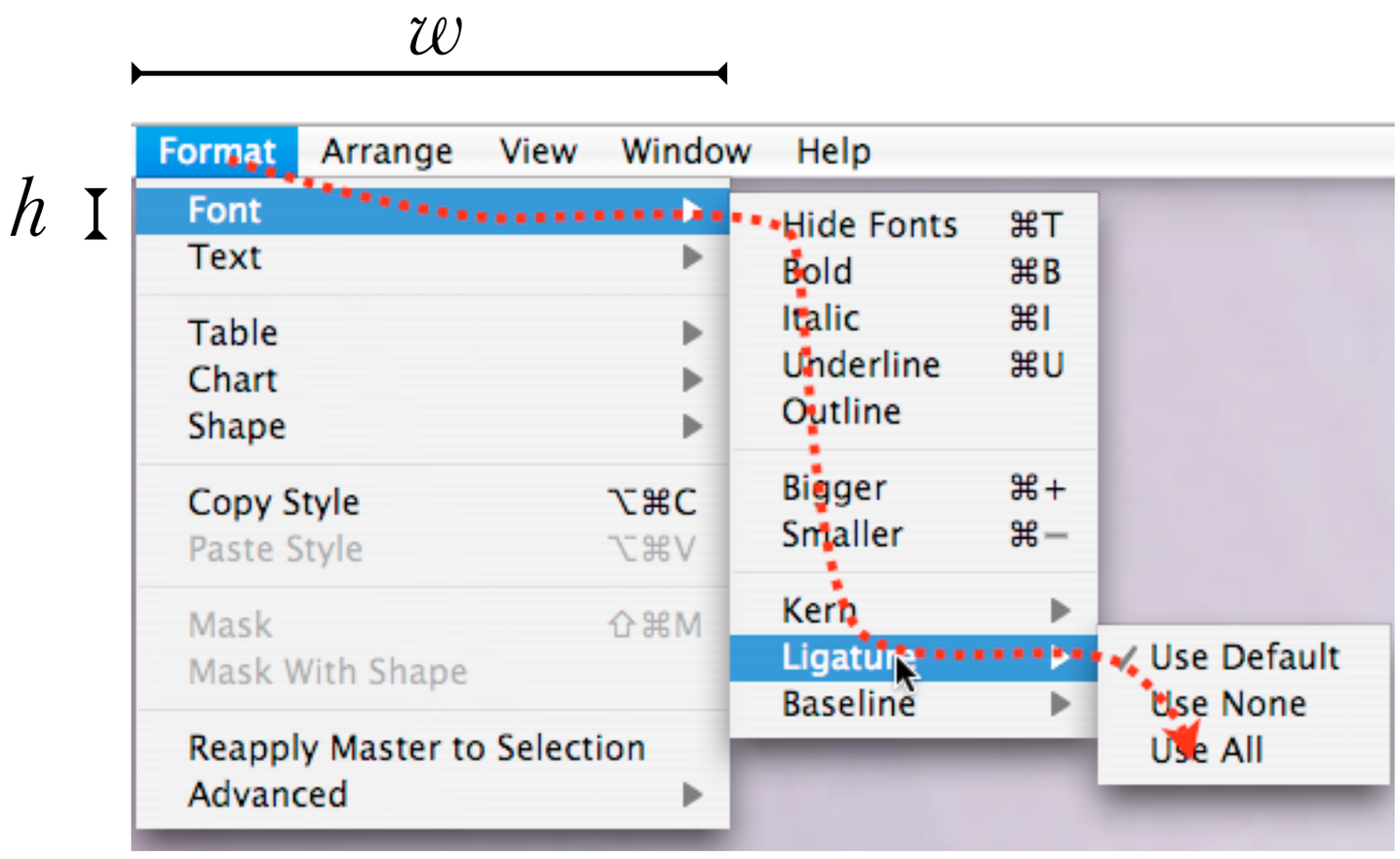
The prime pixel is located at the current location of the mouse pointer. Popup menus make use of this pixel, showing up relative to the mouse pointer, no matter where the user may have moved it. This pixel

Laws of Interaction Design

- Moore’s law
- Buxton’s law
- Fitts’ law
- Steering law
- Guiard’s Kinematic chain model
- Hick’s law
- Law of practice
- Murphy’s law

So we have seen Fitt’s law. Actually this law only applies for 1D movement. In the real life movements are not so straight. This is where Steering’s law come into action

Steering law



$$T_n = \overbrace{a + b\frac{nh}{w}}^{\text{Vertical}} + \overbrace{a + b\frac{w}{h}}^{\text{Horizontal}}$$
$$= 2a + b(\frac{n}{x} + x) \text{ with: } x = \frac{w}{h}$$

Then, a straight tunnel of length A and constant width W can be approximated as a sequence of N evenly spaced goals, each separated from its neighbours by a distance of A/N. We can let N grow arbitrarily large, making the distance between successive goals become infinitesimal.

The total time to navigate through all the goals, and thus through the tunnel, is the sum of the horizontal and vertical steering tasks.

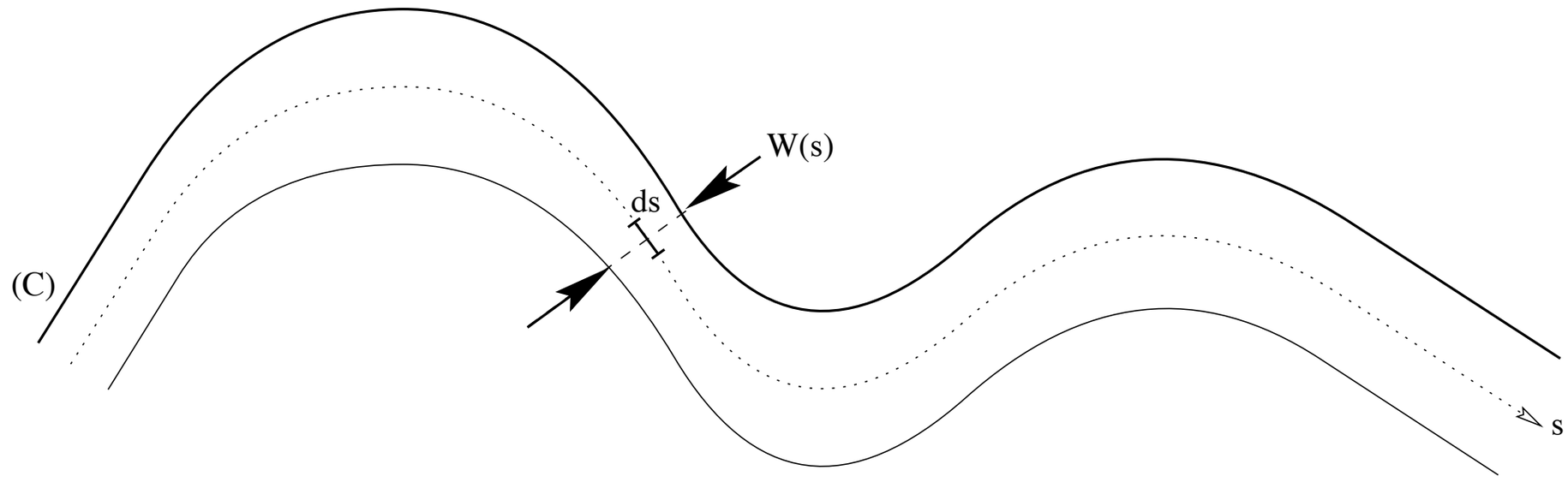
BUT Actually things are a bit different!

on the mac, someone gave it some thoughts and you are actually not forced to have follow a the menu item, you can go in diagonal!

Compare that to most webbased menus specially the ones made with CSS which can’t have any timeout and are only based on hover states.

Steering law on curved paths

C is the path parameterized by s:



average time to navigate through the path

$$T = a + b \int_C \frac{ds}{W(s)}$$

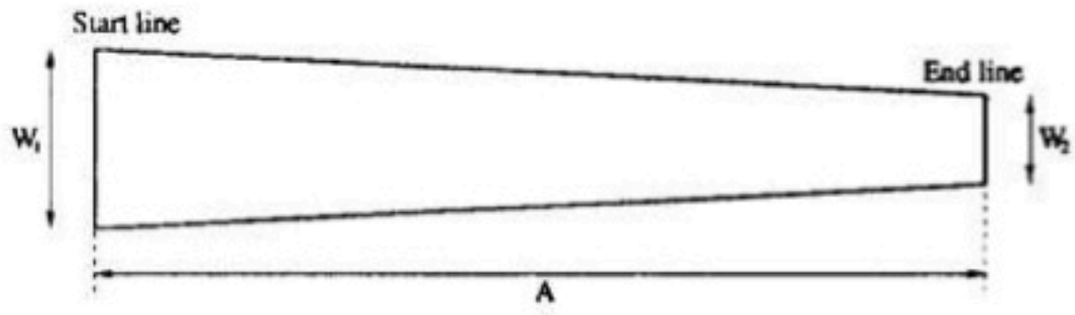
← width of the path at s

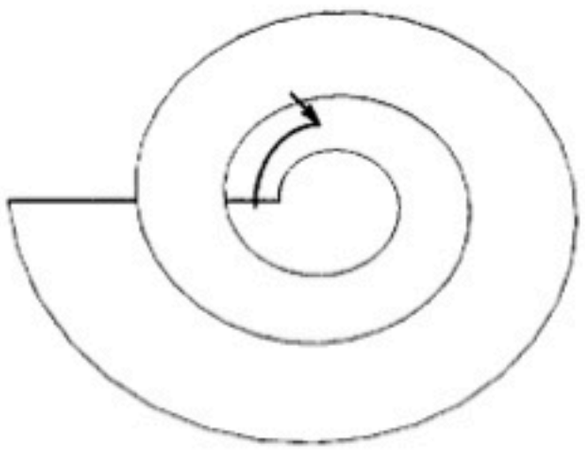
experimentally fitted constants

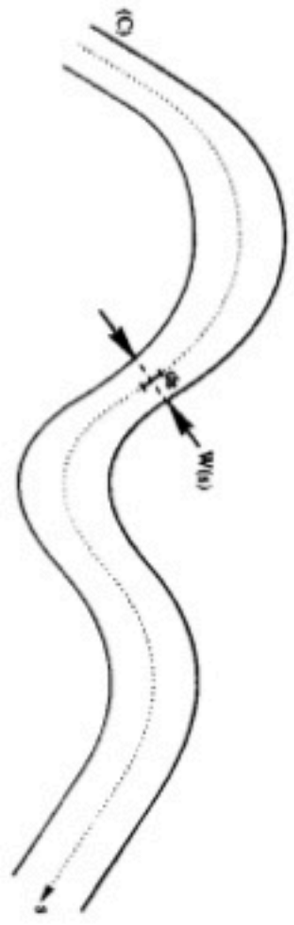
if is a curved path, we define the index of difficulty for steering through this path as the sum along the curve of the elementary indexes of difficulty.

Steering Law applications

- Early work focused on car driving scenarios and models with straight tunnels
- Various example tunnel shapes have been explored

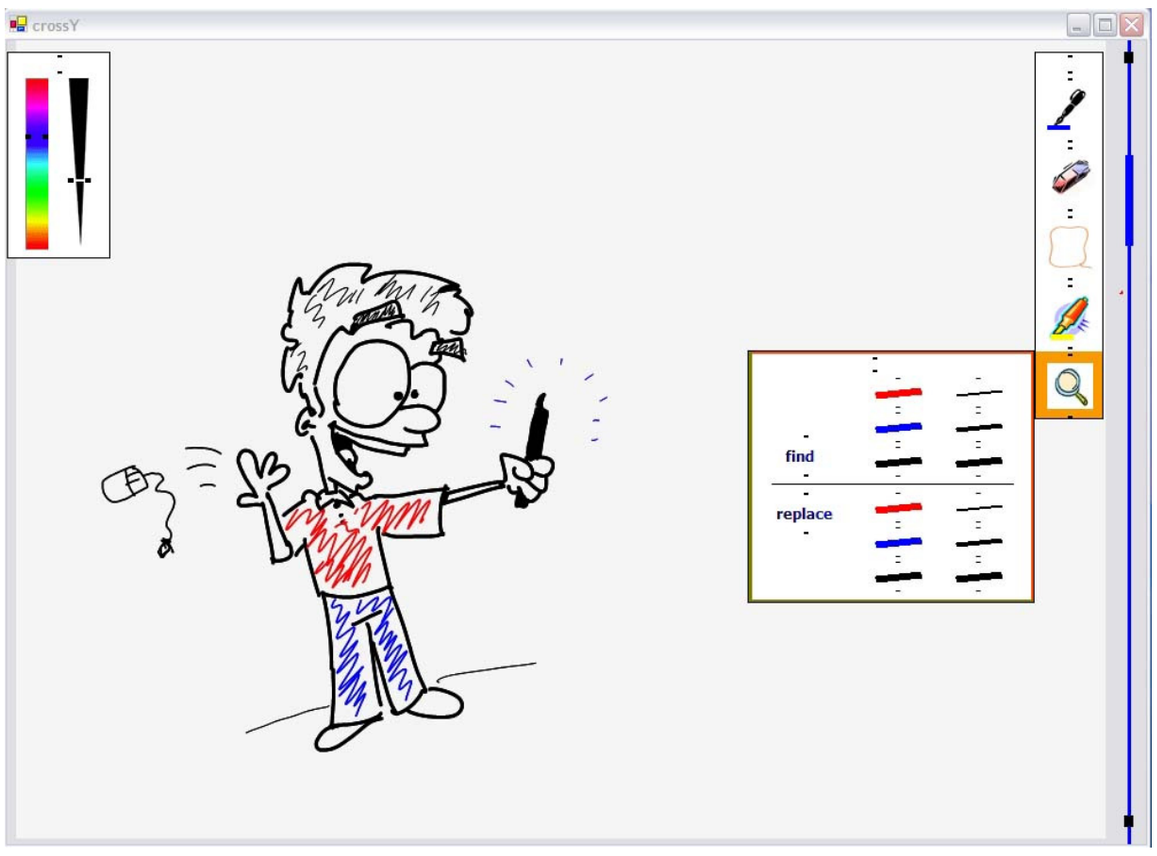


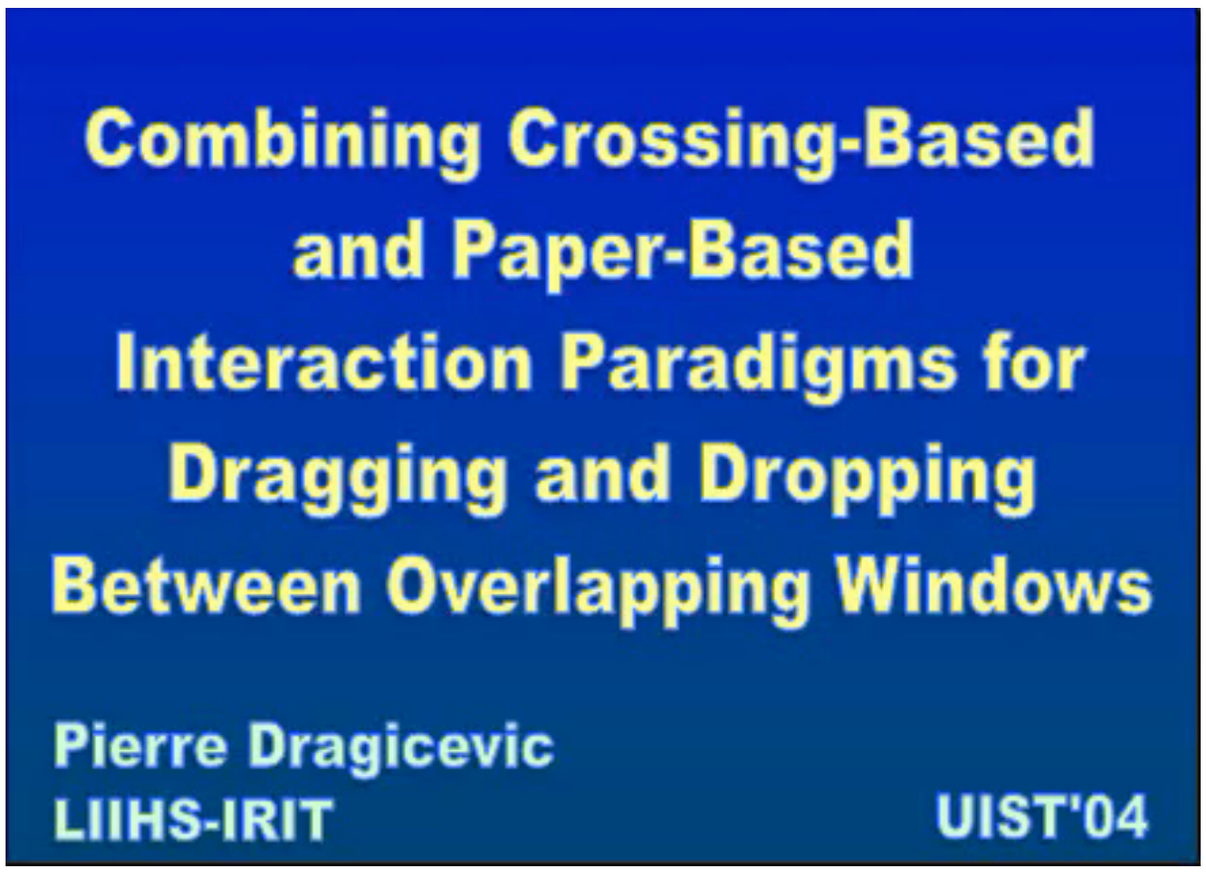




Applications

Steering tasks can be related to crossing tasks:



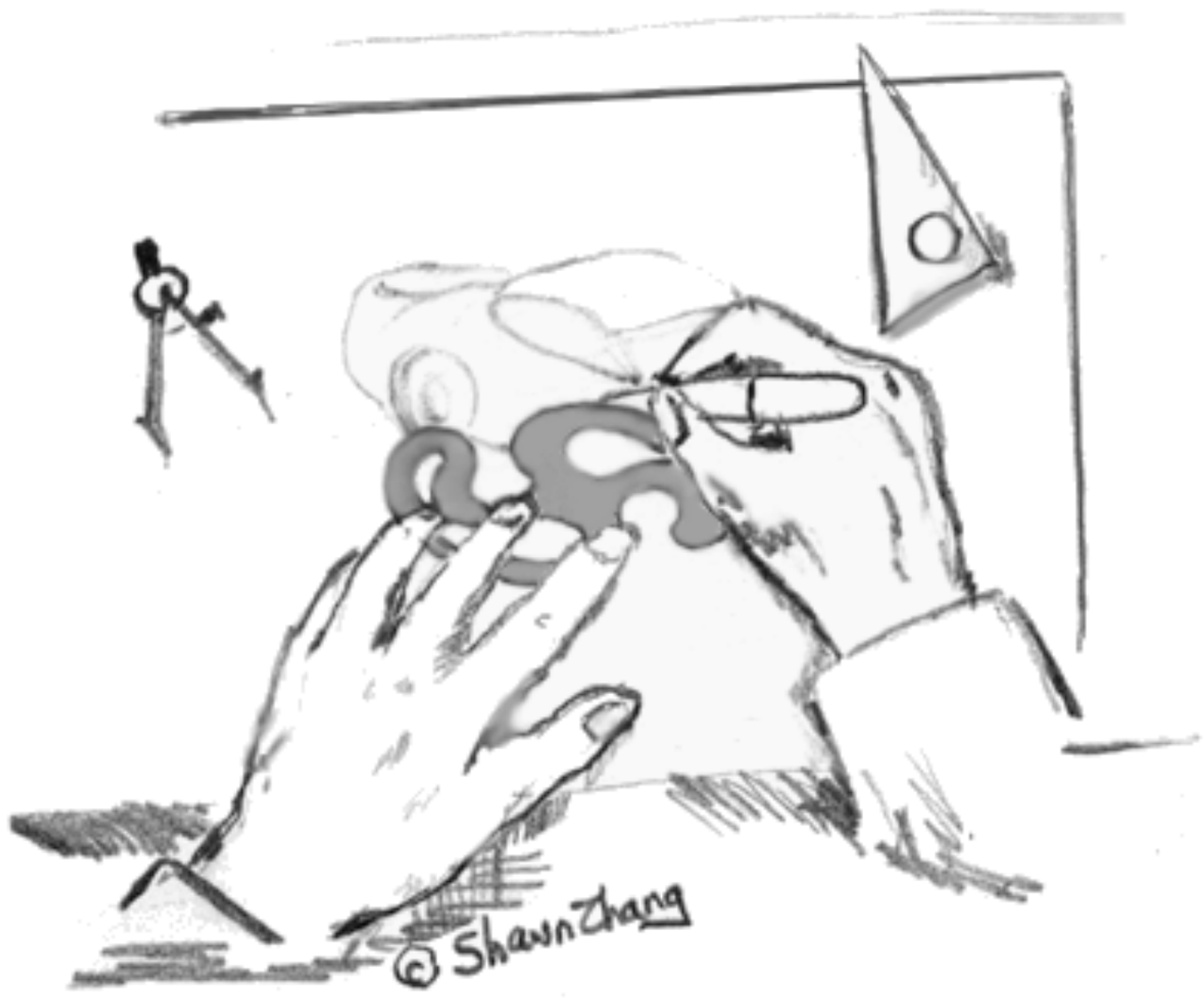


Laws of Interaction Design

- Moore’s law
- Buxton’s law
- Fitts’ law
- Steering law
- Guiard’s Kinematic chain
- Hick’s law
- Law of practice
- Murphy’s law

So we have seen Fitt’s law. Actually this law only applies for 1D movement. In the real life movements are not so straight. This is where Hick’s law come into action

A human capability



From The Two-Handed Desktop Interface: Are We There Yet? [MacKenzie & Guiard, 2001]

Humans are not only two-handed-they use their hands differently. Research on the between-hand division of labor in everyday tasks [3] and HCI [1] reveals that most tasks are asymmetric. Typically, the non-preferred (NP) hand leads, sets the frame of reference for the preferred (P) hand, and works at a relatively coarse level. The P hand follows, works within the frame of reference set by the NP hand, and acts at a finer level.

In this illustration you can see a right-handed artist is sketching the design of a new car. The artist acquires the template with her Left Hand (*NP hand goes first*); the template is manipulated over the workspace (*coarse movement, sets frame of reference*). The stylus is acquired in the RH (*P hand follows*) and brought into the vicinity of the template (*works within frame of reference set by NP hand*). Sketching takes place (*P hand makes precise movements*).

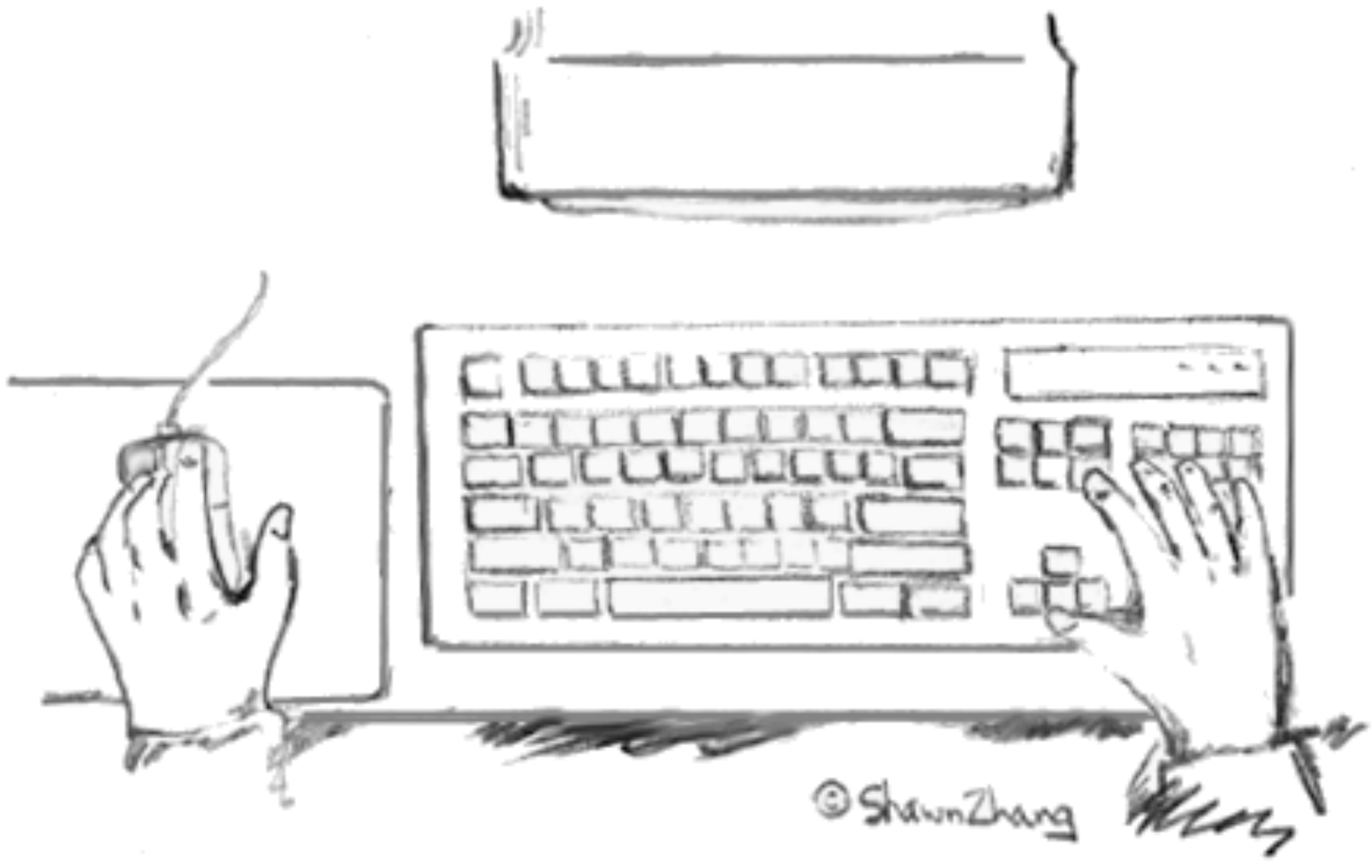
Guiard's Kinematic Chain

*"Under standard conditions, the spontaneous writing speed of adults is **reduced** by some **20%** when instructions **prevent the non-preferred hand** from manipulating the page"*

Non-dominant hand provides a frame of reference for the dominant hand

- Non-dominant hand operates at a coarse temporal and spatial scale;
- Dominant hand operates at a fine temporal and spatial scale

Two handed-interaction at the desktop

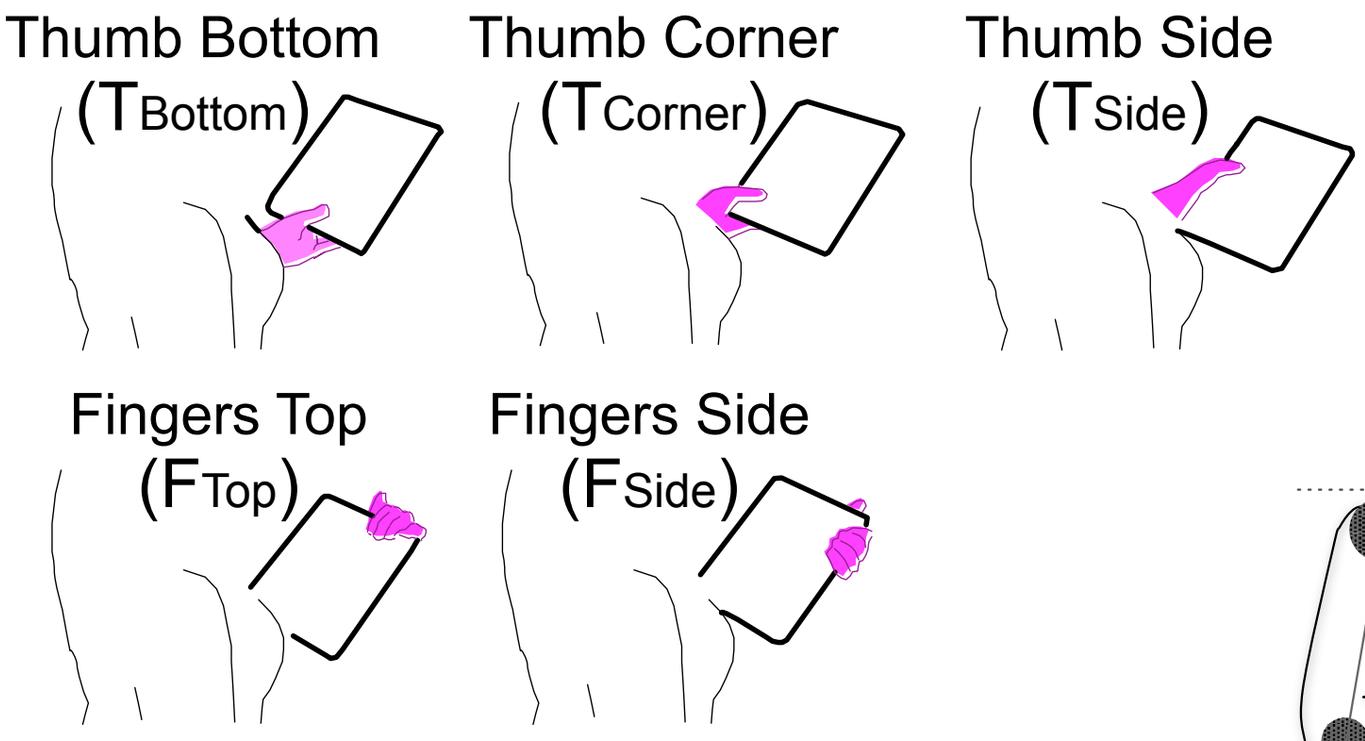


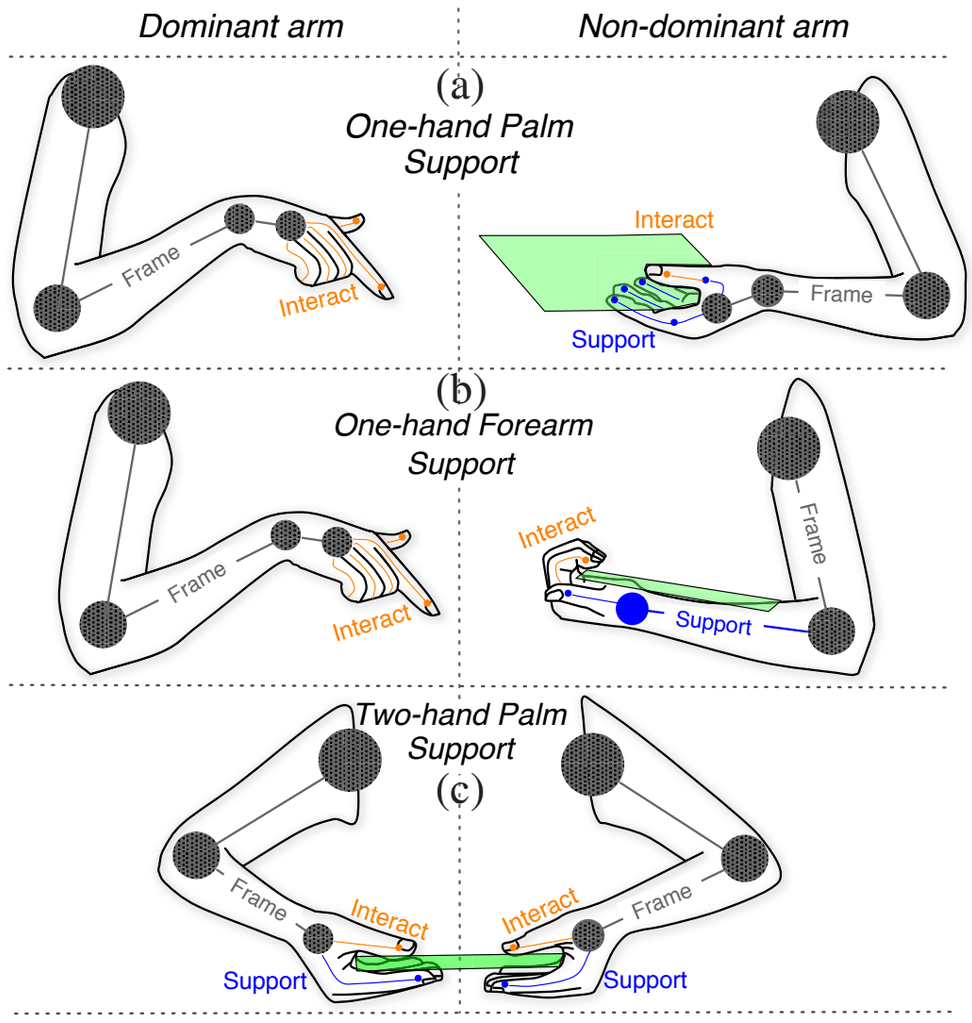
From The Two-Handed Desktop Interface: Are We There Yet? [MacKenzie & Guiard, 2001]

Toolglass palette...

Kinect etc.

Application - how do people hold tablets?





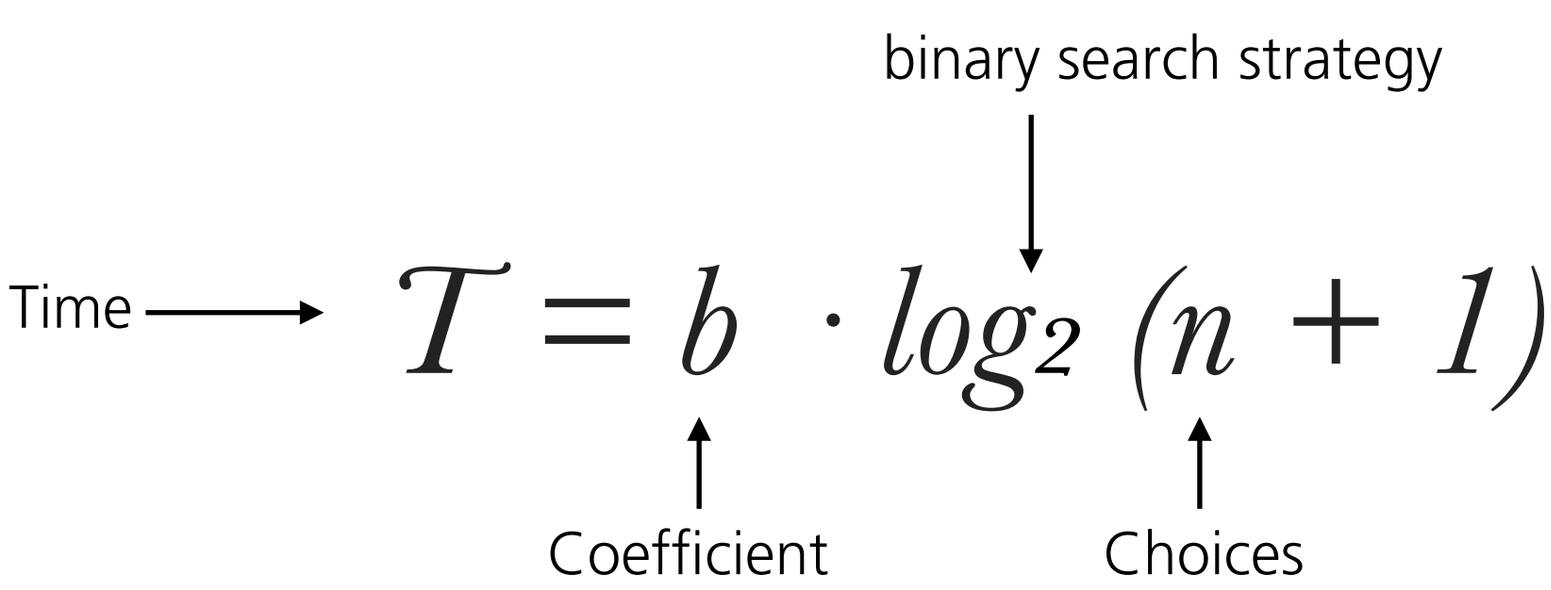
J. Wagner, S. Huot, W. E. Mackay. **BiTouch and BiPad: Designing Bimanual Interaction for Hand-held Tablets**. In *CHI'12: Proceedings of the 30th International Conference on Human Factors in Computing Systems*, ACM, May 2012.

Laws of Interaction Design

- Moore’s law
- Buxton’s law
- Fitts’ law
- Steering law
- Guiard’s Kinematic chain
- Hick’s law
- Law of practice
- Murphy’s law

Hick's law

*Given **n** equally **probable choices**,*
*the average reaction **time** T required*
***to choose among them** is:*



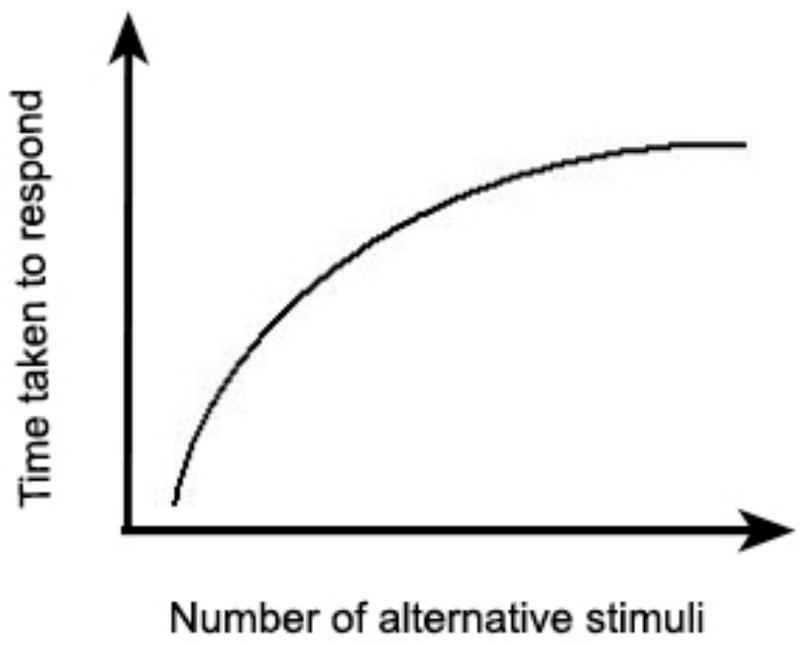
d = constant value, experimentally determined
n = number of choices

Hick Law Examples

http://www.hier-luebeck.de/wp-content/uploads/2010/09/StartMenueWindows7.jpg

http://www.photosophic.com/iphone_screen

Hick’s law



$$T = b \cdot log_2 (n + 1)$$

- Given n equally probable choices, the average reaction time T required to choose among them [read formula]

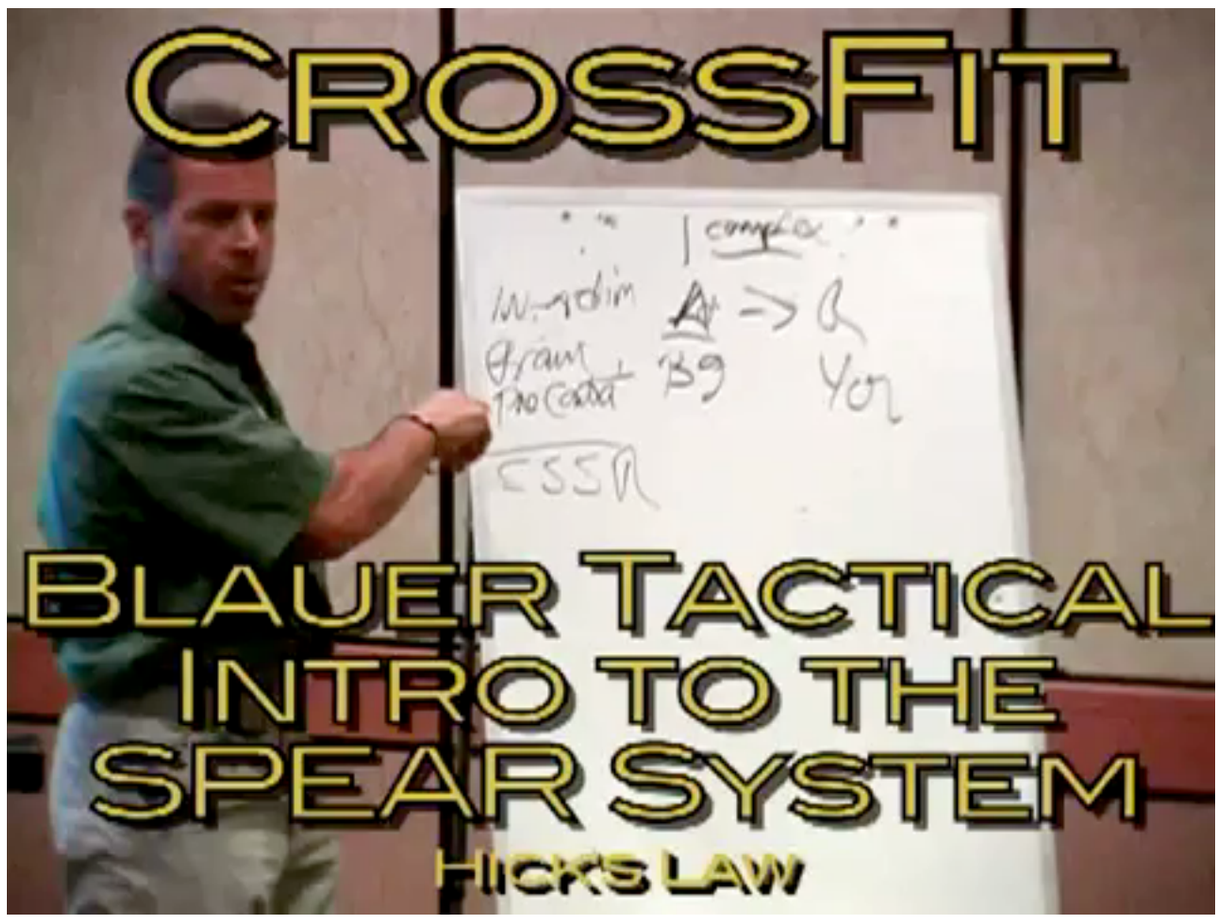
- where b is a constant that can be determined empirically by fitting a line to measured data. Operation of logarithm here expresses depth of "choice tree" hierarchy. Basically log2 means that you perform binary search. According to Card, Moran, and Newell (1983), the +1 is "because there is uncertainty about whether to respond or not, as well as about which response to make."

- Hick's Law is similar in form to Fitts's law. Intuitively, one can reason that Hick's Law has a logarithmic form because people subdivide the total collection of choices into categories, eliminating about half of the remaining choices at each step, rather than considering each and every choice one-by-one, requiring linear time.

- Hick's Law is sometimes cited to justify menu design decisions. However, applying the model to menus must be done with care. For example, to find a given word (e.g. the name of a command) in a randomly ordered word list (e.g. a menu), scanning of each word in the list is required, consuming linear time, so Hick's law does not apply. However, if the list is alphabetical and the user knows the name of the command, he or she may be able to use a subdividing strategy that works in logarithmic time.

http://www.youtube.com/watch?v=w0hJveJ8Hp0

In another context...



Laws of Interaction Design

- Moore’s law
- Buxton’s law
- Fitts’ law
- Steering law
- Guiard’s Kinematic chain
- Hick’s law
- Law of practice
- Murphy’s law

The Power Law of Practice

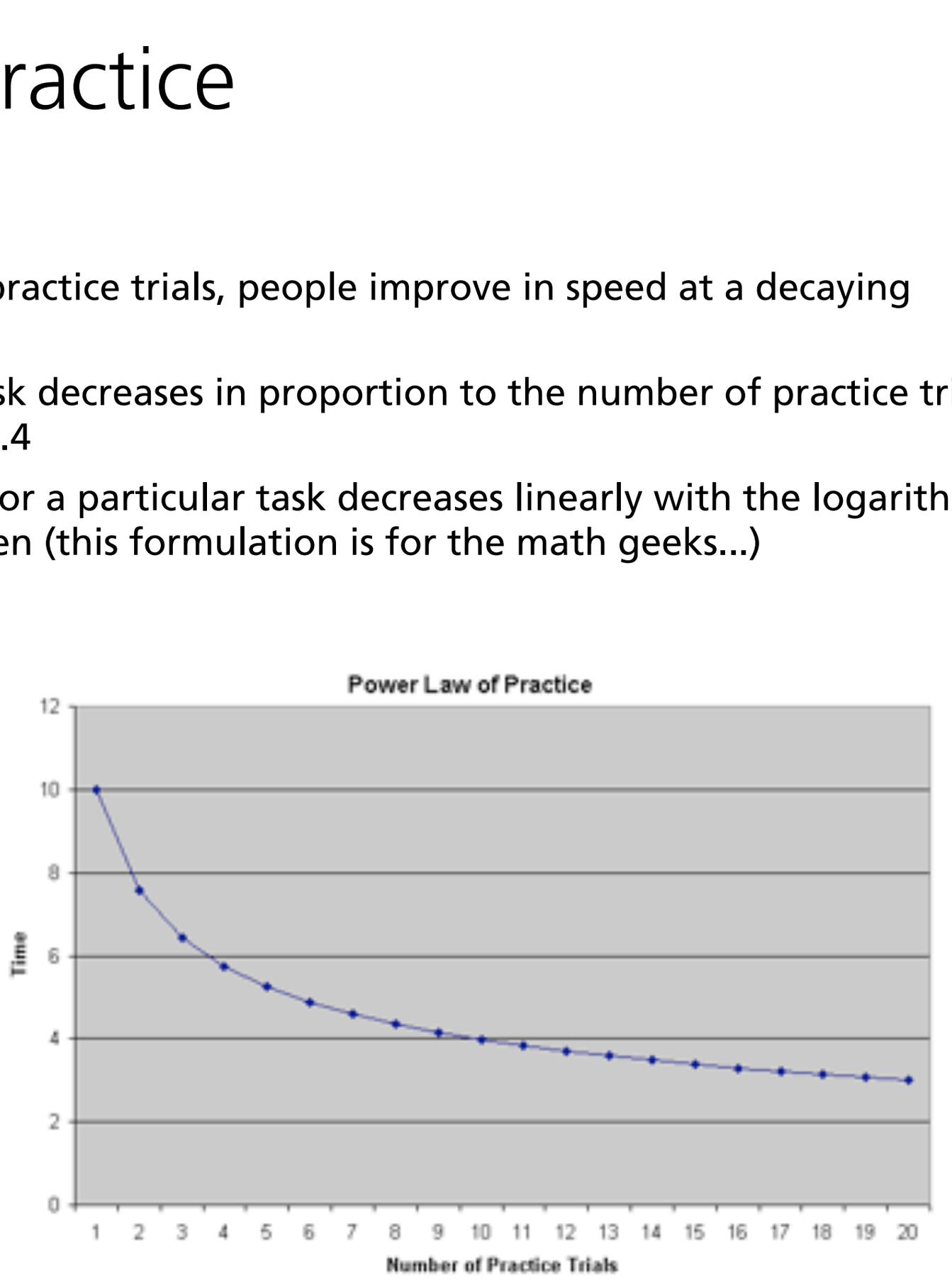
- When performing a task based on practice trials, people improve in speed at a decaying exponential rate.
- The time needed for a particular task decreases in proportion to the number of practice trials taken raised to a power of about -0.4
- The logarithm of the time needed for a particular task decreases linearly with the logarithm of the number of practice trials taken (this formulation is for the math geeks...)

Completion time for trial *n*

Constants

$$T(n) = T(1).n^{a} + c$$

Completion time for trial *1*



The quantitative statement of the power law of practice has been applied to a wide variety of different human behaviors: immediate response tasks, motor perceptual tasks, recall tests, text editing, and more high-level, deliberate tasks such as game playing (from University of Michigan, Artificial Intelligence Laboratory: Power law of Practice, adapted).

Because of the decay according to a power function, we can make two observations:

- The largest improvements in speed are made during the very first trials. Therefore, we should be careful with generalizing timing results from first-time users.
- The learning process lasts virtually endlessly. With workers who rolled cigars, small improvements were demonstrated even after tens of thousands of trials.

Laws of Interaction Design

- Moore’s law
- Buxton’s law
- Fitts’ law
- Steering law
- Hick’s law
- Guiard’s Kinematic chain
- Law of practice
- Murphy’s law

So we have seen Fitt’s law. Actually this law only applies for 1D movement. In the real life movements are not so straight. This is where Hick’s law come into action

Murphy’s law

“Whatever can go wrong, will go wrong.“

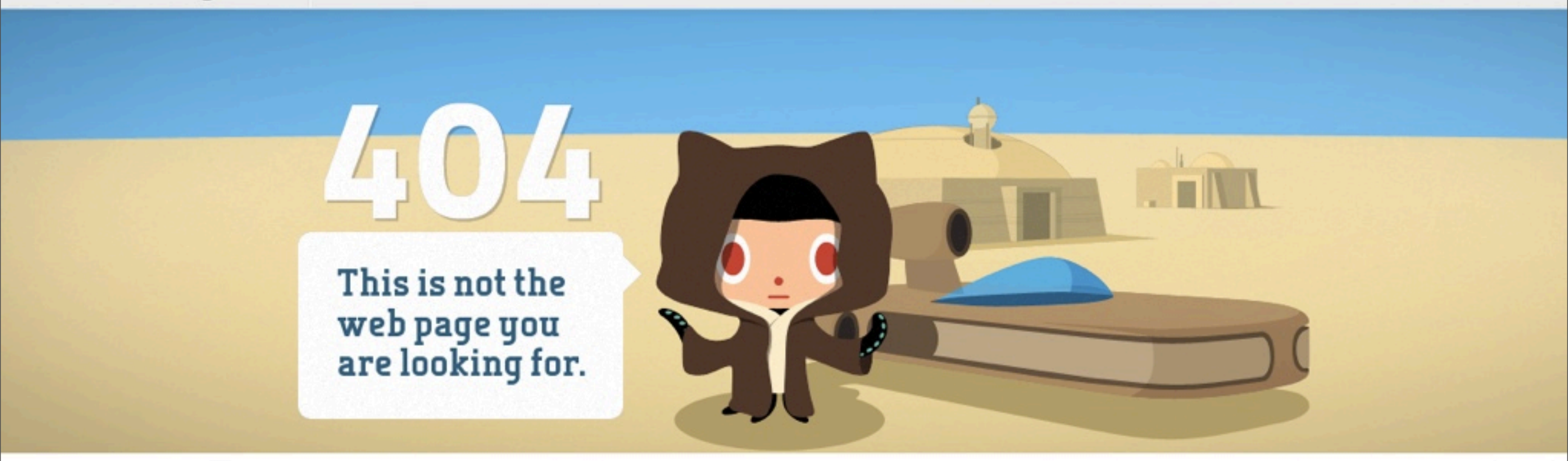
[Edward Aloysius Murphy Jr., 1949]

“If there's more than one possible outcome of a job or task, and one of those outcomes will result in disaster or an un-desirable consequence, then somebody will do it that way.“

Implications of Murphy's law

- Prepare for human errors, wrong input etc.
  - do sanity checks in dialogs
  - provide useful defaults
  - make serious mistakes hard

- When building stuff, provide extra time for:
  - mistakes in manufacturing
  - non-functioning tools
  - faulty material
  - misunderstandings



GitHub
About
Blog
Features
Contact & Support
Training
GitHub Enterprise
Site Status

Tools
Gauges: Analyze web traffic
Speaker Deck: Presentations
Gist: Code snippets
GitHub for Mac
GitHub for Windows
Issues for iPhone
Job Board

Extras
GitHub Shop
The Octodex

Documentation
GitHub Help
Developer API
GitHub Flavored Markdown
GitHub Pages

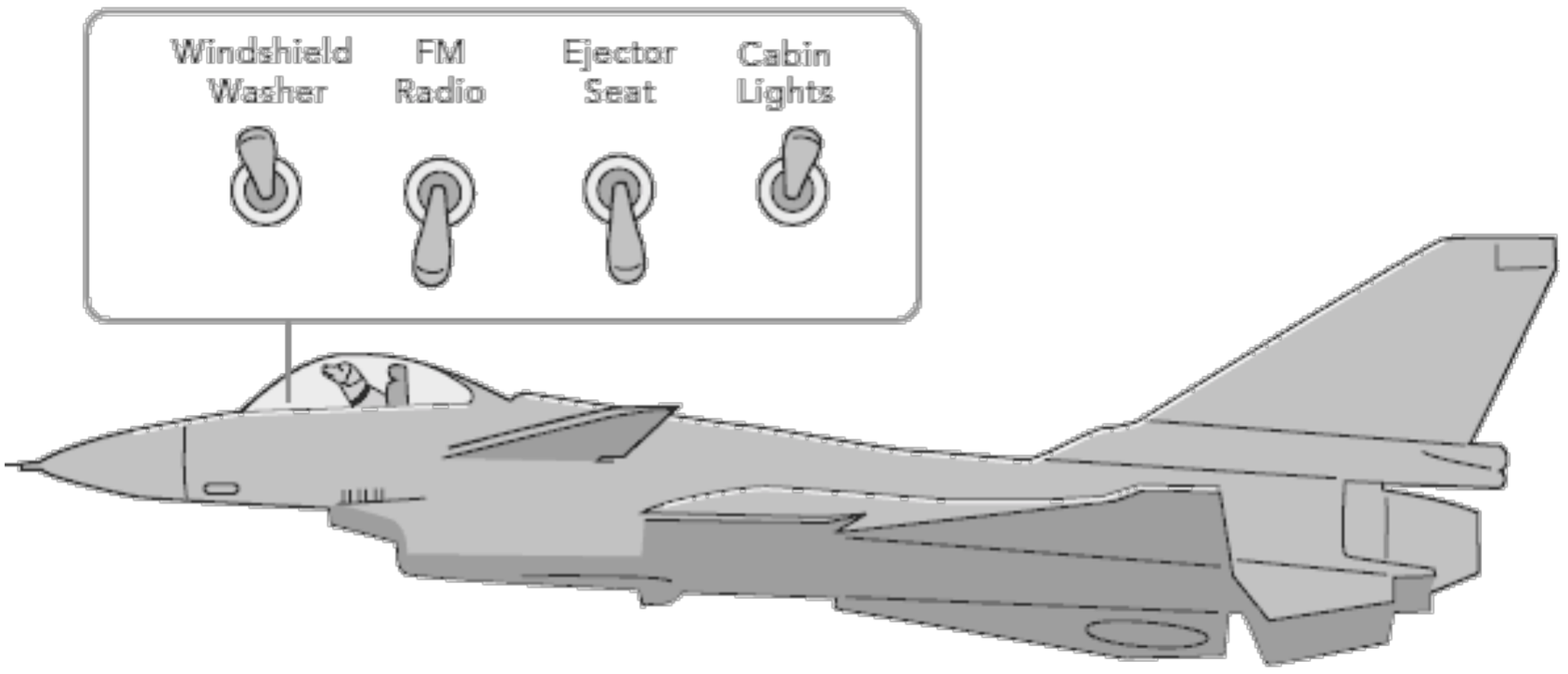
github SOCIAL CODING

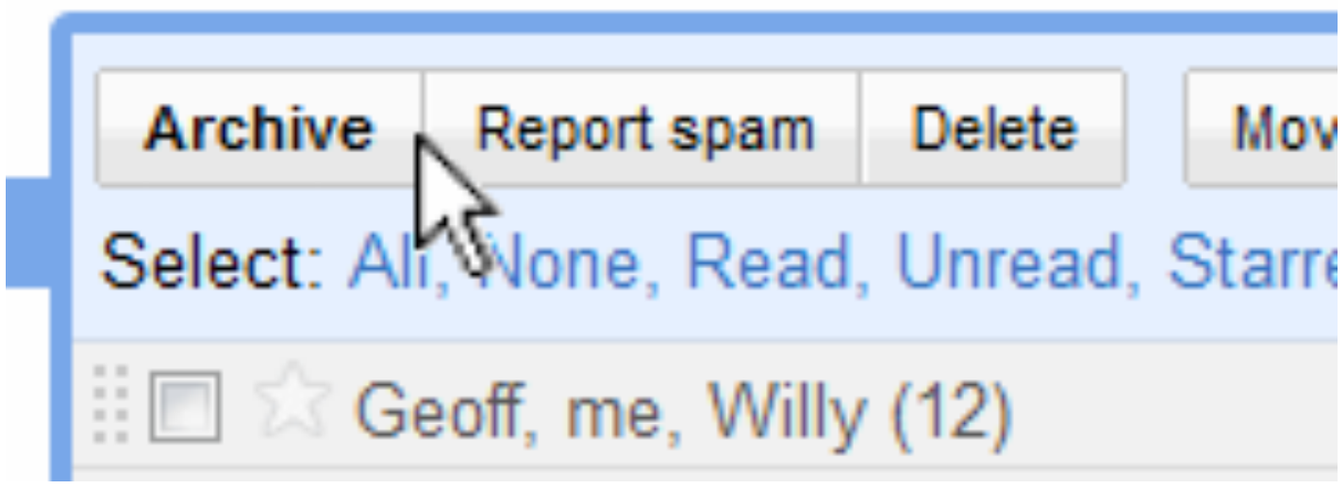
Terms of Service Privacy Security

Powered by the **Dedicated Servers** and **Cloud Computing** of Rackspace Hosting®

http://www.codinghorror.com/blog/2010/03/the-opposite-of-fitts-law.html

Anti Fitts law





In the cockpit of every jet fighter is a brightly painted lever that, when pulled, fires a small rocket engine underneath the pilot's seat, blowing the pilot, still in his seat, out of the aircraft to parachute safely to earth. Ejector seat levers can only be used once, and their consequences are significant and irreversible.
Applications must have ejector seat levers so that users canâ€”occasionallyâ€”move persistent objects in the interface, or dramatically (sometimes irreversibly) alter the function or behavior of the application. The one thing that must never happen is accidental deployment of the ejector seat.

http://www.codinghorror.com/.a/6a0120a85dcdae970b01310fd5e9f8970c-800wi

- Protection of the tabletop
- Rotation of the racks
- Tangible not used.

Breakoutsession No. 4

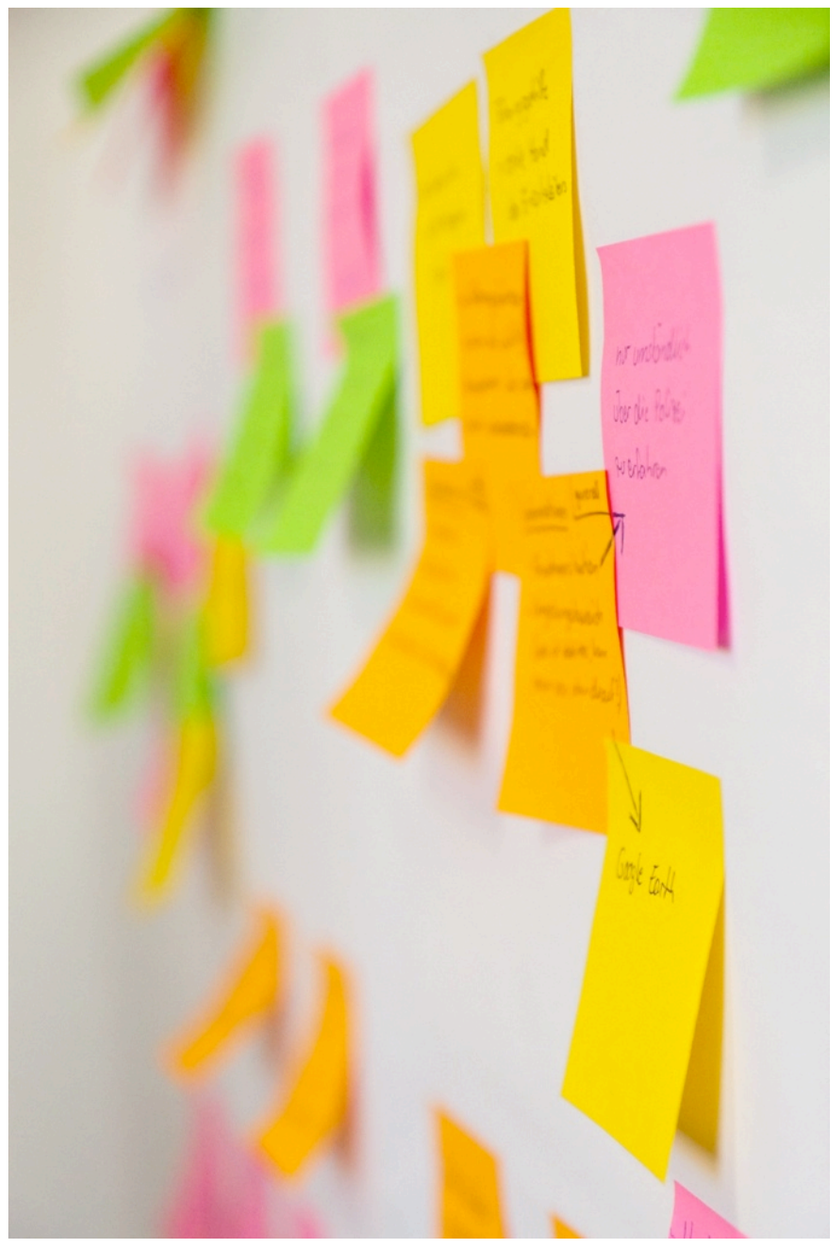
Brainstorming

Looking back… (Discussion)

- discuss experiences made during the interviews
  - What was you strategy to prepare the interview?
  - What was your target group?
  - Was it easy to find interviewees?
  - How did the interviews go?
  - Was the preparation sufficient?
  - Was it easy to receive the information you were looking for?
  - How long did the interview take?
  - How did you record the results?
  - What will you do different in your next interview?
  - Do you have any other tips for the others?

Affinity Diagram

- recap:
  - method for sorting and making sense out of data
  - helps to identify themes and discover unseen connections
  - data points can be recorded on sticky notes and sorted into logical groups/themes

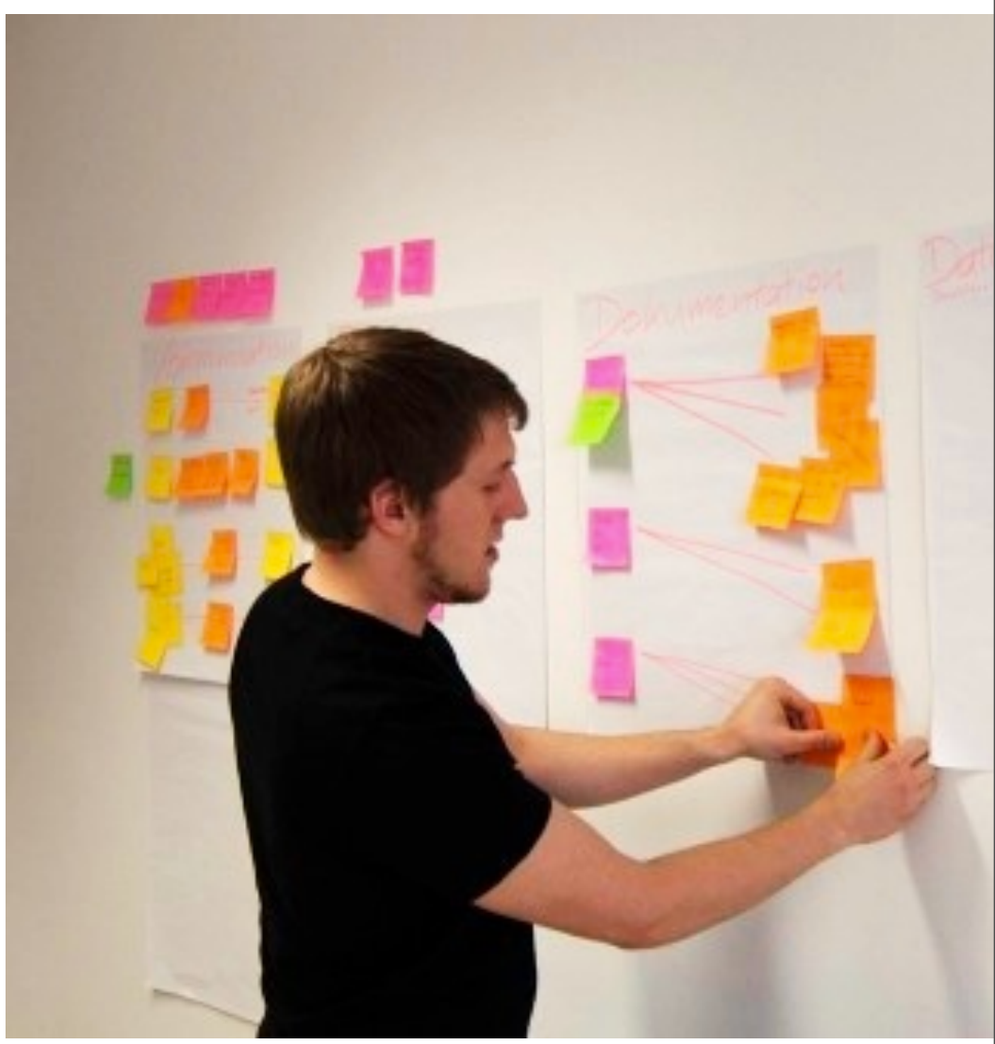


http://conceptdevelopmentlmu.wordpress.com/

Affinity Diagram

- process overview:

1. use recorded research data to identify ideas, aspects, issues
2. record each finding on post-it notes
3. look for related ideas
4. sort notes into groups until all cards have been used
5. repeat as many times as needed
6. add labels to themes if appropriate
7. draw connections between findings and themes



http://conceptdevelopmentlmu.wordpress.com/

Affinity Diagram

- guideline – how to cluster and model data:

  – everyone reads through the post-it's and arranges them
  – everyone is allowed to re-order
  – group post-it's into themes
  – name and discuss the themes
  – rate themes and ideas to weight your findings

http://conceptdevelopmentlmu.wordpress.com/

Brainstorming

http://conceptdevelopmentlmu.wordpress.com/

- rules :
  - avoid too early judgment
    - there are no bad ideas
  - bring in also crazy ideas
    - it’s the wild ideas that often provide the breakthroughs
  - place ideas on top of each other
    - think ‘and’ rather than ‘but’
  - keep the focus on the topic
    - you get better output if everybody is disciplined
  - one conversation at a time
    - that way all ideas can be heard and built upon
  - vote for the **best** ideas!

Task for Today

– affinity diagram
  - create an affinity diagram to analyse your data

– brainstorming
  - pick one theme
  - try to find a solution that solves the problem or improves the quality of the task
  - target device characteristics:
    - touch screen
    - mobile

– before you go
  - come to us and show your concept!

http://conceptdevelopmentlmu.wordpress.com/

Homework

– work on your concept

– create a visual presentation of your concept:

- containing sketches and annotations
- send it via email to sebastian.loehmann@ifi.lmu.de
- file format: PDF
- deadline: Sunday 27.05.2012 – 21:00