

Continguts

- Realitat Virtual
- **Augmented Reality**

Introduction to AR

- Augmented Reality is a combination of a **real scene** viewed by a user and a synthetic **virtual scene** that augments the scene with additional information.
- AR environments differ from VEs in that we have access to both real and virtual objects at the same time.



Goal of AR

- Goal: enhance user **performance** and **perception** of the world.
- Challenge: keep users from **perceiving the difference** between the real world and the virtual augmentation of it.



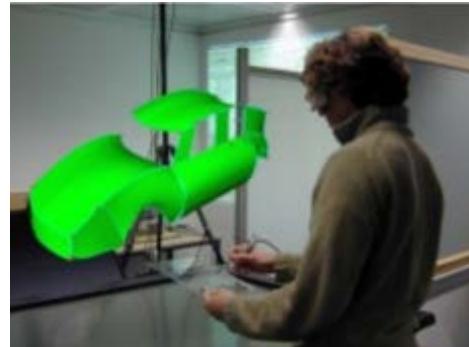
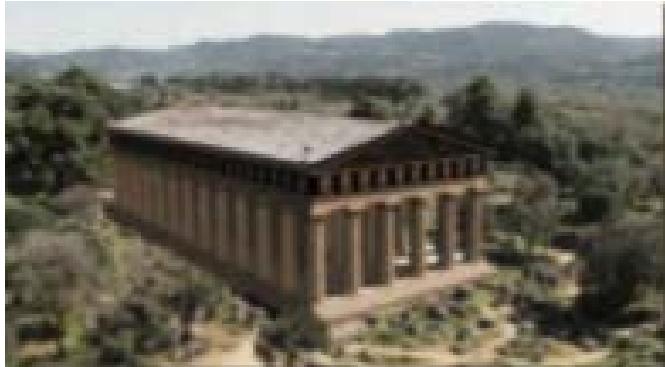
Augmented Reality

©2001 How Stuff Works



AR applications

- Archeology
- Entertainment
- Engineering design
- Consumer design



Augmented vs Virtual Reality

Augmented Reality

- System augments the real world scene
- User maintains a sense of presence in real world
- Needs a mechanism to combine virtual and real worlds

Virtual Reality

- Totally immersive environment
- Visual senses are under control of system (sometimes aural and proprioceptive senses too)

Augmented Reality

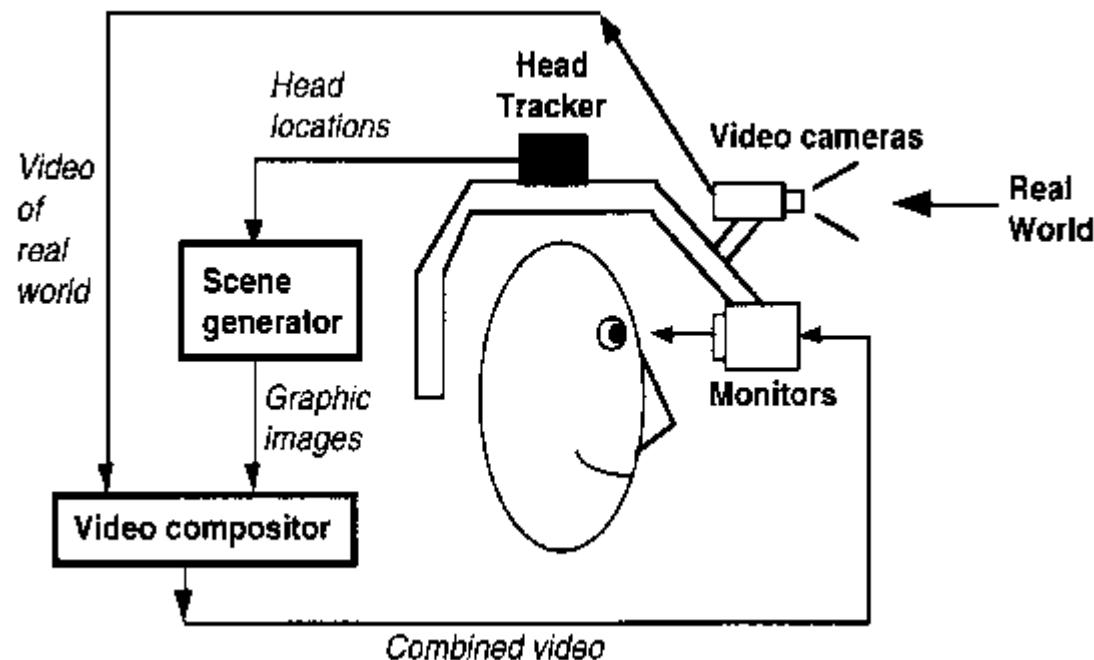
- The importance of object registration:
 - The computer generated virtual objects must be **accurately registered** with the real world in all dimensions.
 - Errors in this registration will prevent the user from seeing the real and virtual images as fused.
 - The **correct registration** must be maintained while the user moves about within the real environment.
 - Discrepancies or changes in the apparent registration will range from distracting (difficult to work with), to physically disturbing (unusable system).

Augmented Reality

- There are basically three ways to visually present an augmented reality.
 - **Video see-through:** the virtual environment is replaced by a video feed of reality and the AR is overlaid upon the digitised images
 - **Optical see-through:** Leaves the real-world perception alone but displays only the AR overlay by means of transparent mirrors and lenses.
 - **AR projection** onto real objects.

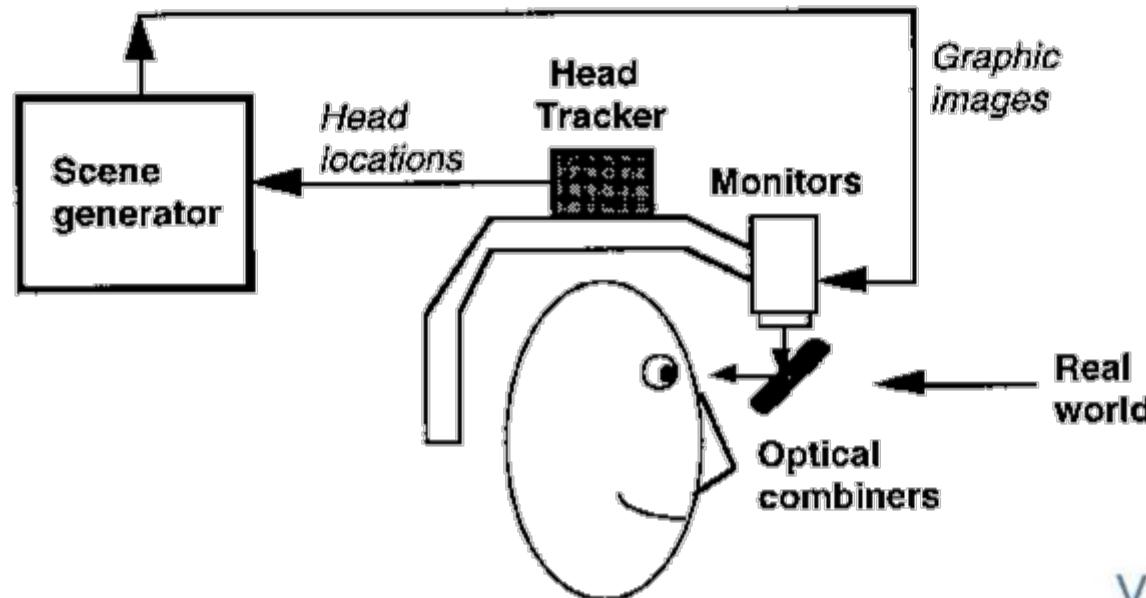
Video see-through HMDs

- Video see-through
 - Use closed-view HMDs.
 - Combine real-time video from head-mounted cameras with virtual imagery.



Optical see-through HMDs

- The user sees the real world **directly**
- Make use of optical combiners:
 - Half-silvered mirrors (partially transparent, partially reflective)
 - Transparent LCD



Optical see-through HMDs

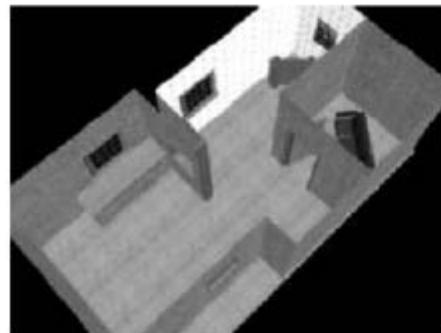


Augmented Reality



Projection-based spatial displays

- Images are projected directly into physical objects.
- Single static, single steerable or multiple projectors.



(a)



(b)



Augmented Reality

- Projective displays. Advantages:
 - They do not require special eye-wear
 - Eye accomodation not required
 - They can cover large surfaces for a wide field-of-view

Augmented Reality

- Projective displays. Disadvantages:
 - Projectors need to be calibrated each time the environment or the distance to the projection surface changes (crucial in mobile setups).
 - Fortunately, calibration may be automated
 - Limited to indoor use only due to low brightness and contrast of the projected images.
 - Occlusion or mediation of objects is also quite poor.

RA: Videos

- Robust high speed feature tracking:
[./RobustHighSpeedTracking_PC_v2.avi](#)
- <https://www.youtube.com/watch?v=UWXictuNowI>

Professors d'IDI - UPC

Interacció i Disseny d'Interfícies

Classe 7: contingut

- Realisme: Il·luminació (2)
 - Il·luminació en OpenGL 3.3 (1)
 - Càcul de color en vèrtexs (en el Vertex Shader)
 - Shading de polígons
 - Suavitzat d'arestes
 - Il·luminació en OpenGL 3.3 (2)
 - Càcul de color en fragments

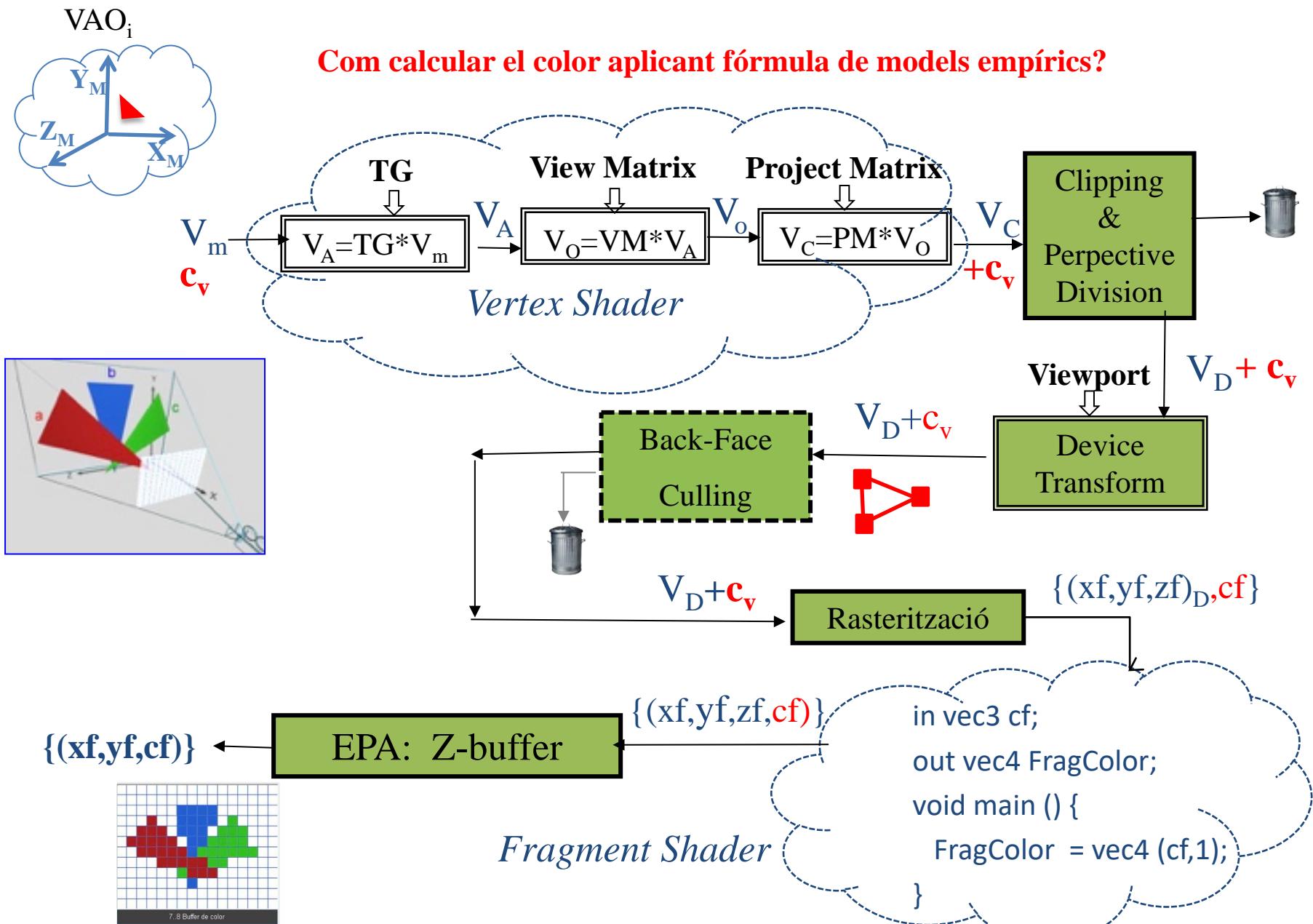
Recordem: Càcul color en un punt

$$I_{\lambda}(P) = I_{a\lambda} k_{a\lambda} + \sum_i (I_{f_i\lambda} k_{d\lambda} \cos(\Phi_i)) + \sum_i (I_{f_i\lambda} k_{s\lambda} \cos^n(\alpha_i))$$

Model Ambient

Model Difús (Lambert) **Model Especular (Phong)**

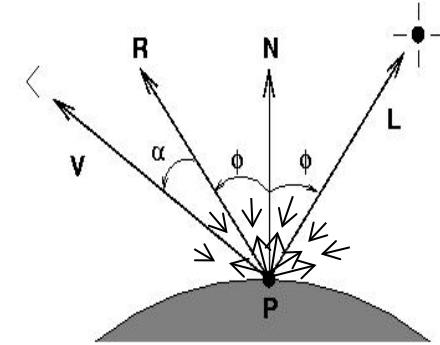
Procés de visualització



Càcul del color per vèrtex en el Vèrtex Shader (1)

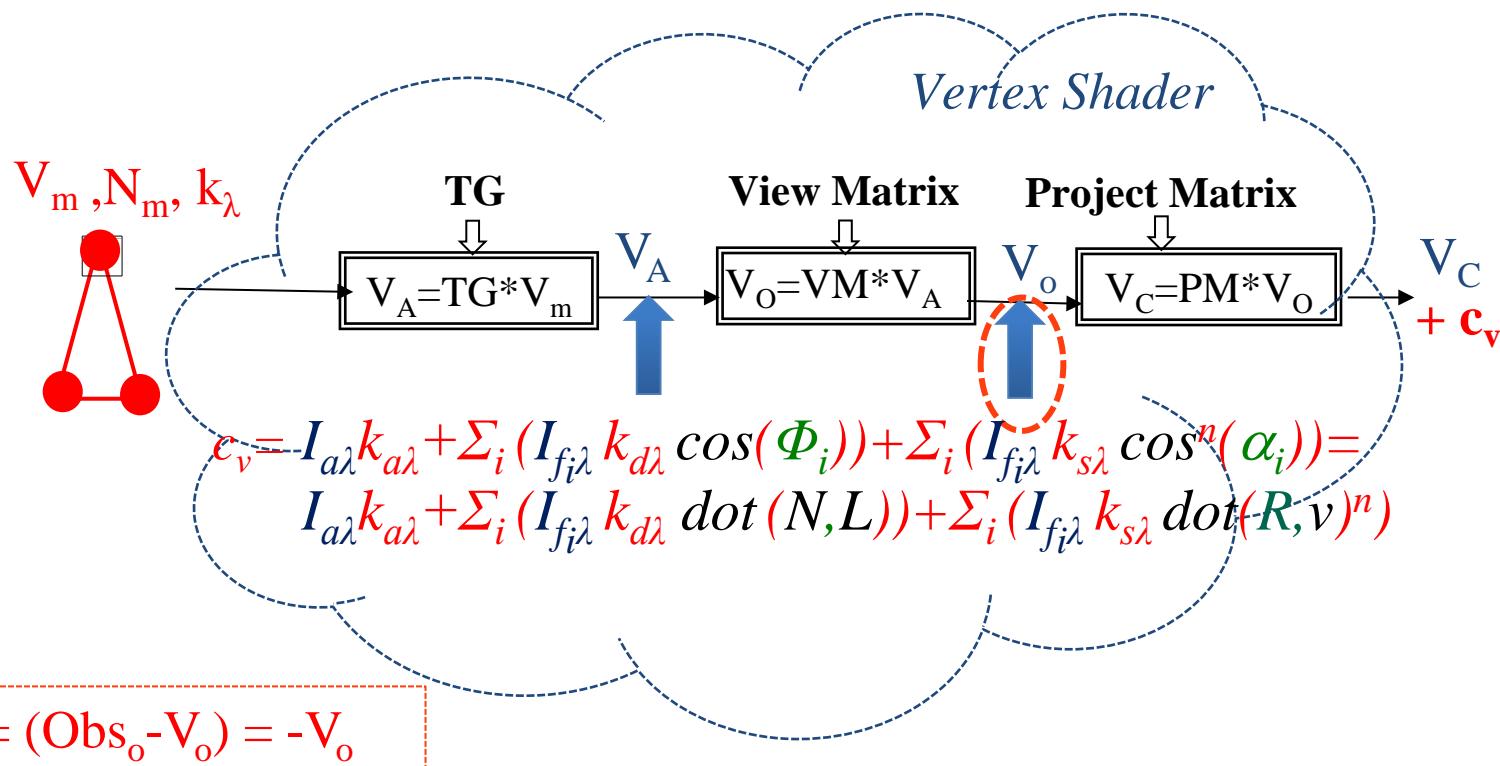
Atributs per cada model:

- Coordenades (V), normal (N) i constants de material (k_λ) per vèrtex en VBOs del seu VAO



Uniforms:

- Fonts de llum actives => color, posició
- Llum ambient



Càcul del color per vèrtex en el Vèrtex Shader (3)

El càcul el farem per cada vèrtex (al Vertex Shader)
I el farem en SCO, per tant:

- Cal passar la posició del vèrtex a SCO
 - multiplicant per (**view * TG**)
- Cal passar el vector normal a SCO
 - multiplicant per la matriu **inversa** de la **transposada de (view * TG)**, -li direm **NormalMatrix**-
*mat3 NormalMatrix = inverse (transpose (mat3 (view * TG)))*
- La posició del focus de llum també ha d'estar en SCO
 - Multiplicat per **view** (si no la tenim directament en SCO)

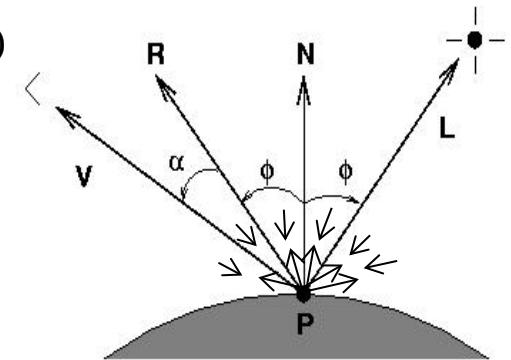
Càcul del color per vèrtex en el Vèrtex Shader (2)

Pseudocodi de Vertex Shader per calcular color en SCO:

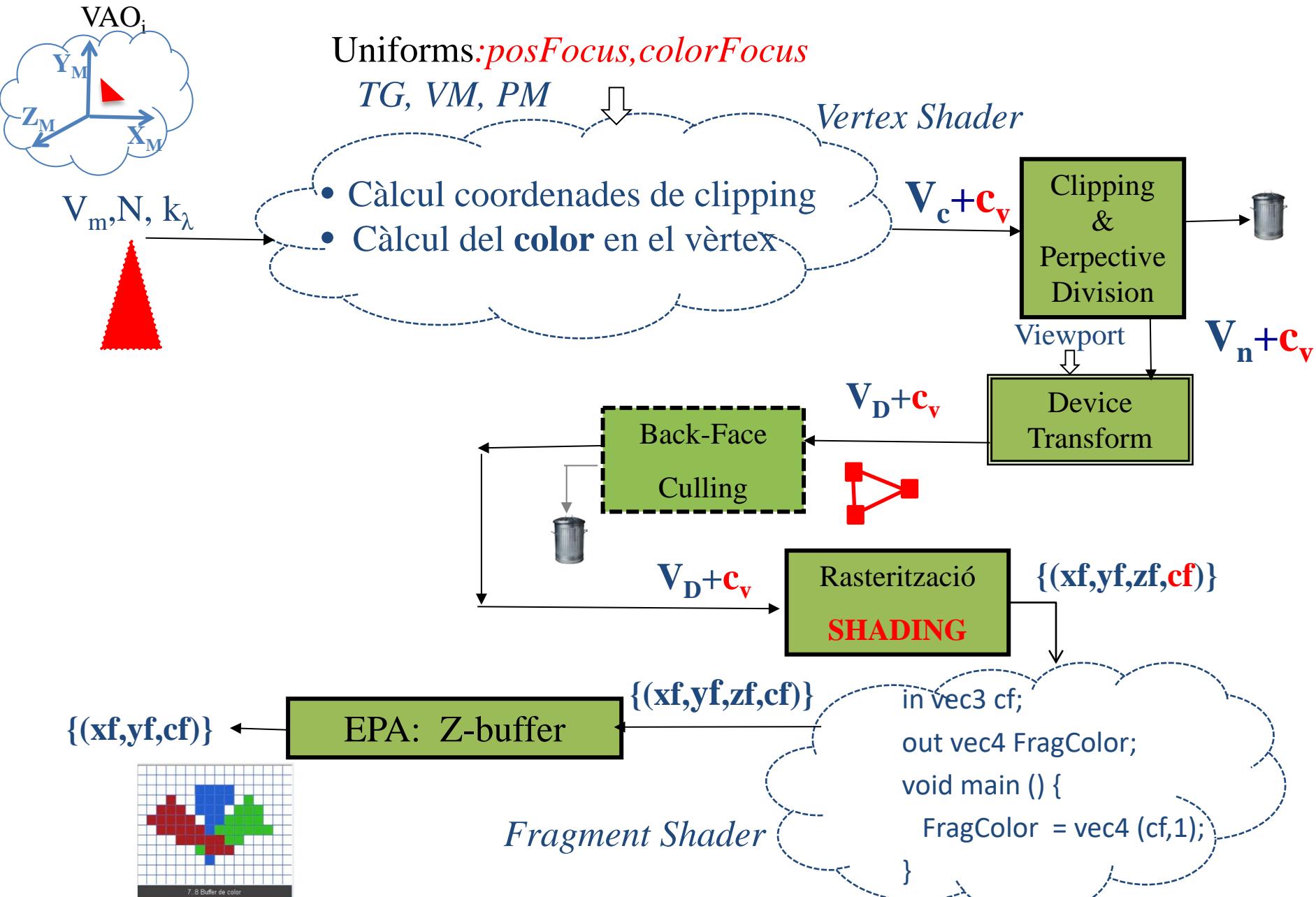
$$c_v = I_{a\lambda} k_{a\lambda} + \sum_i (I_{f_i\lambda} k_{d\lambda} \text{dot}(\mathbf{N}_o, \mathbf{L}_o)) + \sum_i (I_{f_i\lambda} k_{s\lambda} \text{dot}(\mathbf{R}_o, \mathbf{v}_o)^n)$$

```
in vec3 vertex, N;  
in vec3 matamb, matdiff, matspec;  
in float matshin;  
uniform mat4 proj, view, TG;  
uniform vec3 posFocus, Ia, If;  
out vec3 fcolor;
```

```
...  
void main()  
{  
    mat3 NormalMatrix = inverse (transpose (mat3 (view * TG))  
    vec3 NormSCO = normalize (NormalMatrix * N);  
    vec4 vertexSCO = view * TG * vec4 (vertex, 1.0);  
    vec4 focusSCO = view * vec4 (posFocus, 1.0);  
    vec3 LSCO = normalize (focusSCO.xyz - vertexSCO.xyz);  
    fcolor = color_vertex (NormSCO, LSCO, -vertexSCO);  
    gl_Position = proj * view * TG * vec4 (vertex, 1.0);  
}
```

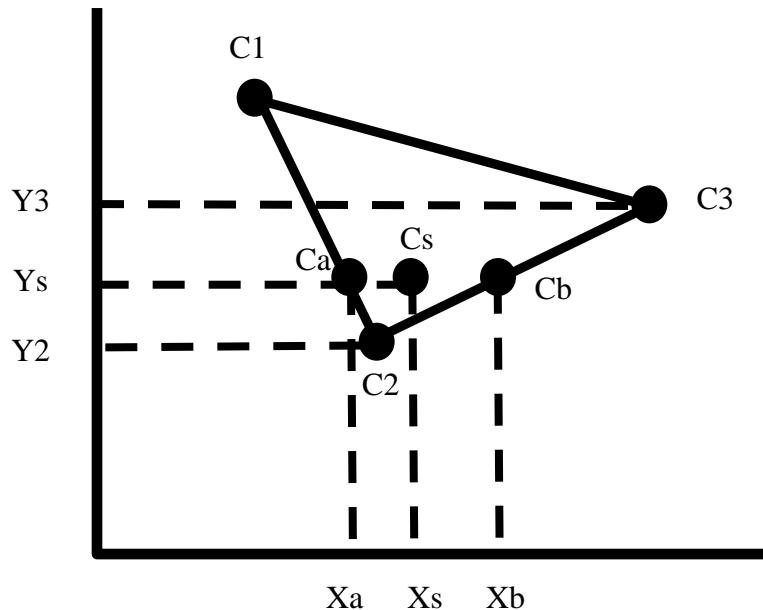


Càcul del color en el Vèrtex Shader + shading (1)



Recordem: Shading (colorat) de polígons

- Colorat Constant \equiv **Flat shading** $\rightarrow C_f = C1$
color uniforme per tot el polígon (funció del color calculat en un vèrtex); cada cara pot tenir diferent color.
- Colorat de Gouraud \equiv **Gouraud shading** \equiv **Smooth shading**

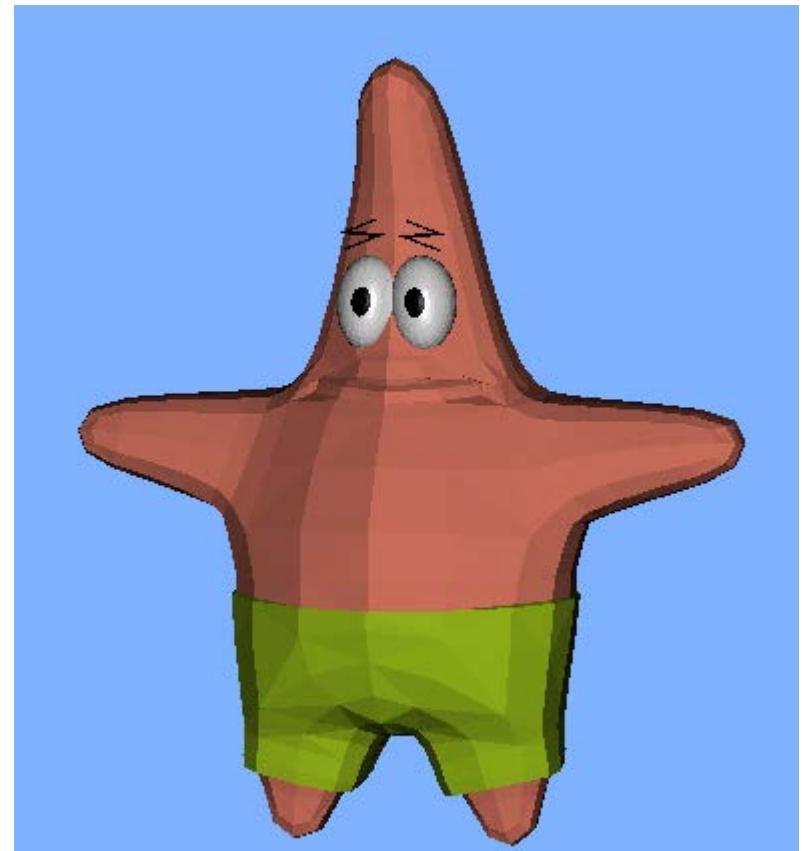
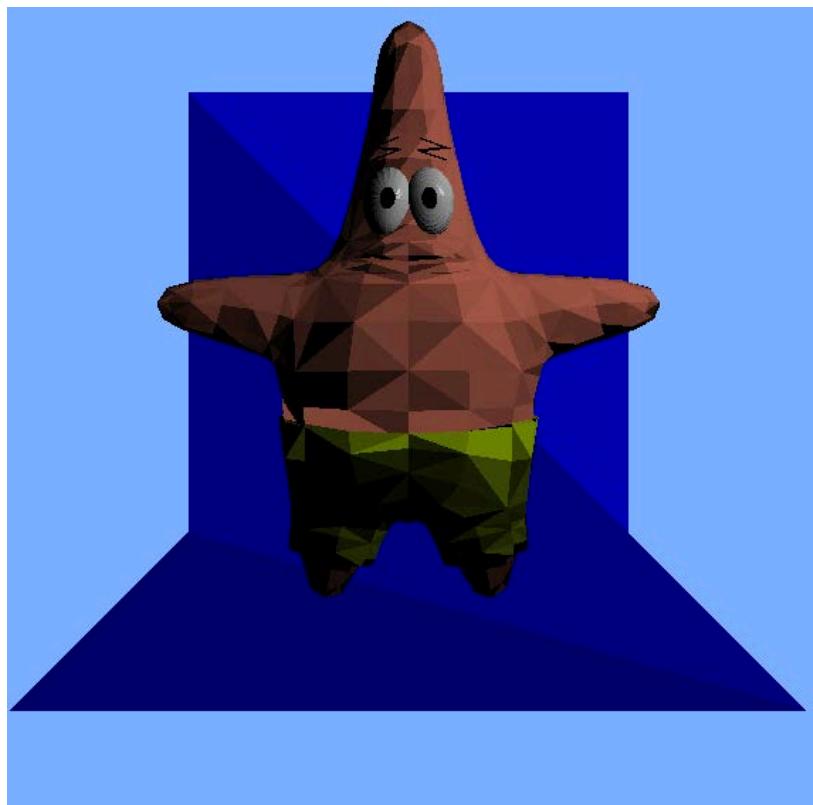


$$Ca = \frac{1}{Y1 - Y2} (C1(Ys - Y2) + C2(Y1 - Ys))$$

$$Cb = \frac{1}{Y3 - Y2} (C2(Y3 - Ys) + C3(Y1 - Ys))$$

$$Cs = \frac{1}{Xb - Xa} (Ca(Xb - Xs) + Cb(Xs - Xa))$$

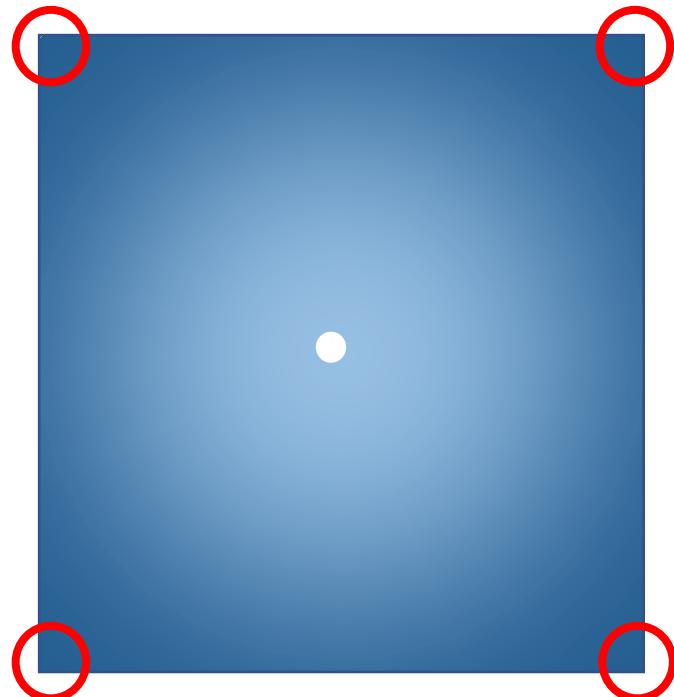
Flat versus Gouraud/smooth Shading



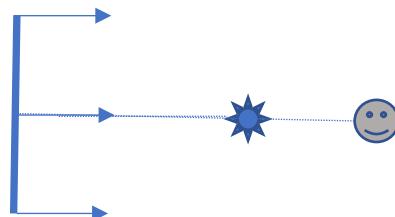
Càcul color en VS: limitacions

Efecte del càlcul de Lambert i Phong en Vertex Shader:

Com s'hauria de veure



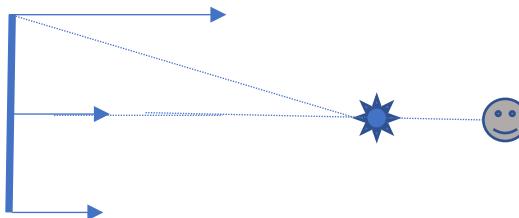
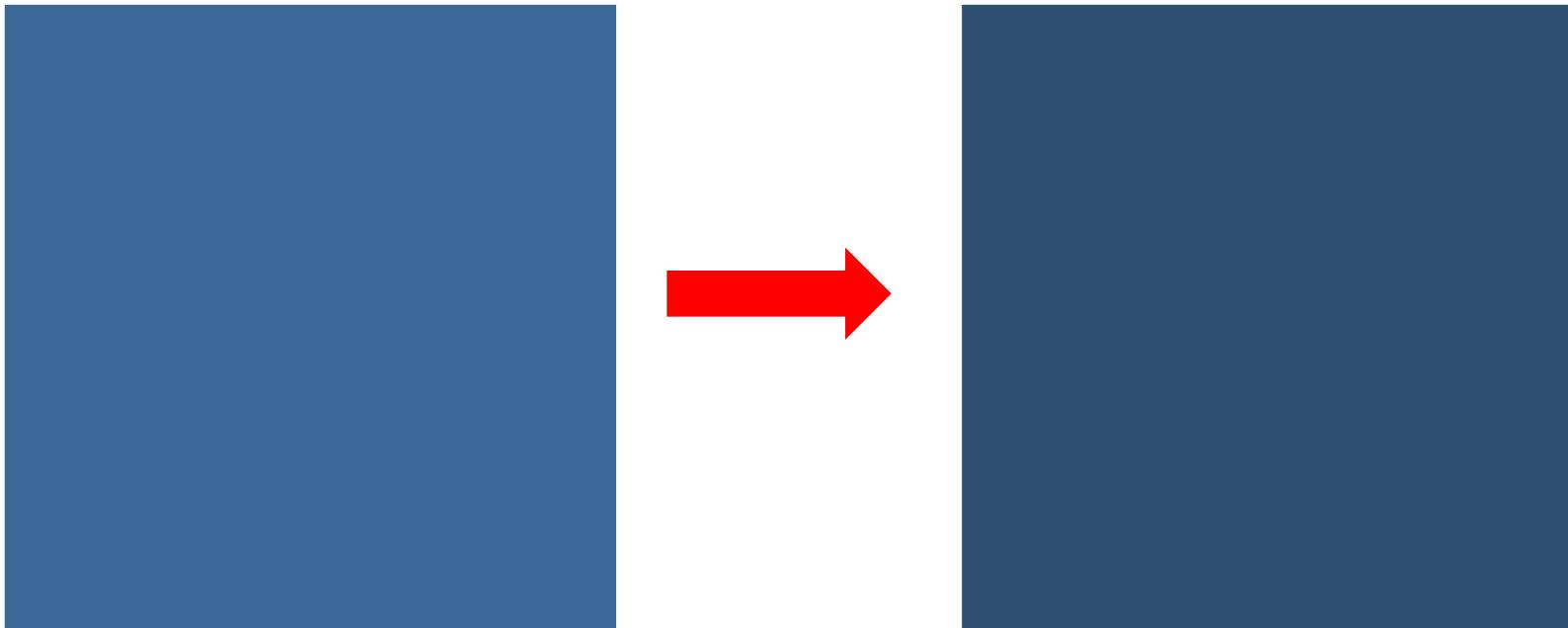
Com es veu



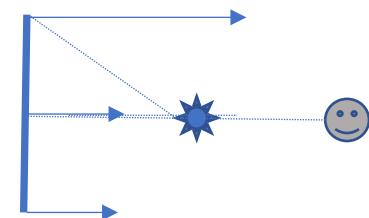
Possible solució: **Discretitzar més**

Càcul color en VS: limitacions

Què passa si aprotem el focus de llum al quadrat?



angle Φ creix



Classe 7: contingut

- Realisme: Il·luminació (2)
 - Il·luminació en OpenGL 3.3 (1)
 - Càcul de color en vèrtexs (en el Vertex Shader)
 - Shading de polígons
 - **Suavitzat d'arestes**
 - Il·luminació en OpenGL 3.3 (2)
 - Càcul de color en fragments

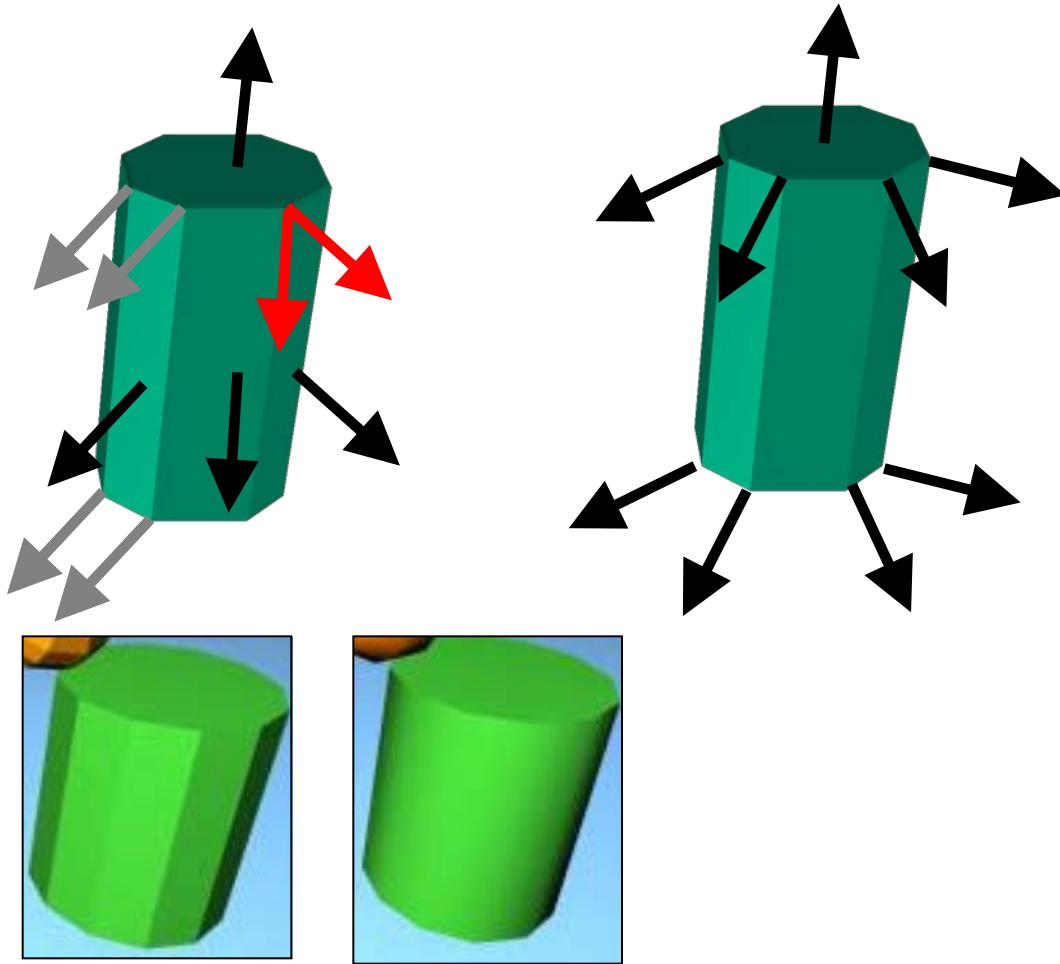
Suavitzat d'arestes (1)



**Quin model d'il·luminació i shading s'utilitza?
Per què no es veuen les arestes?
Noteu la forma de les siluetes**

Suavitzat d'arestes (2)

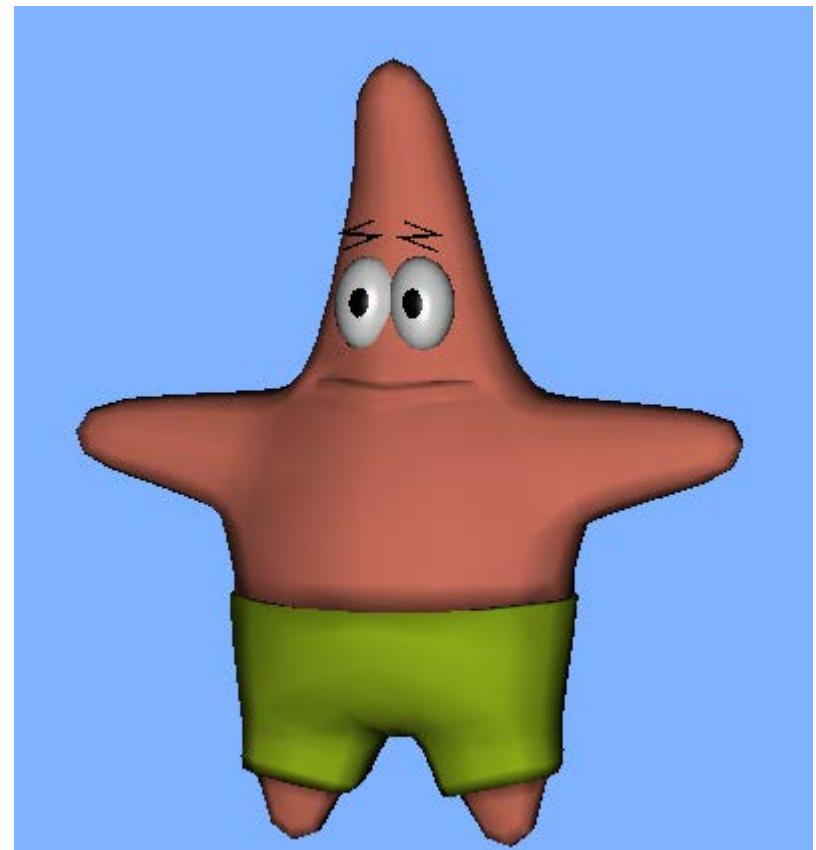
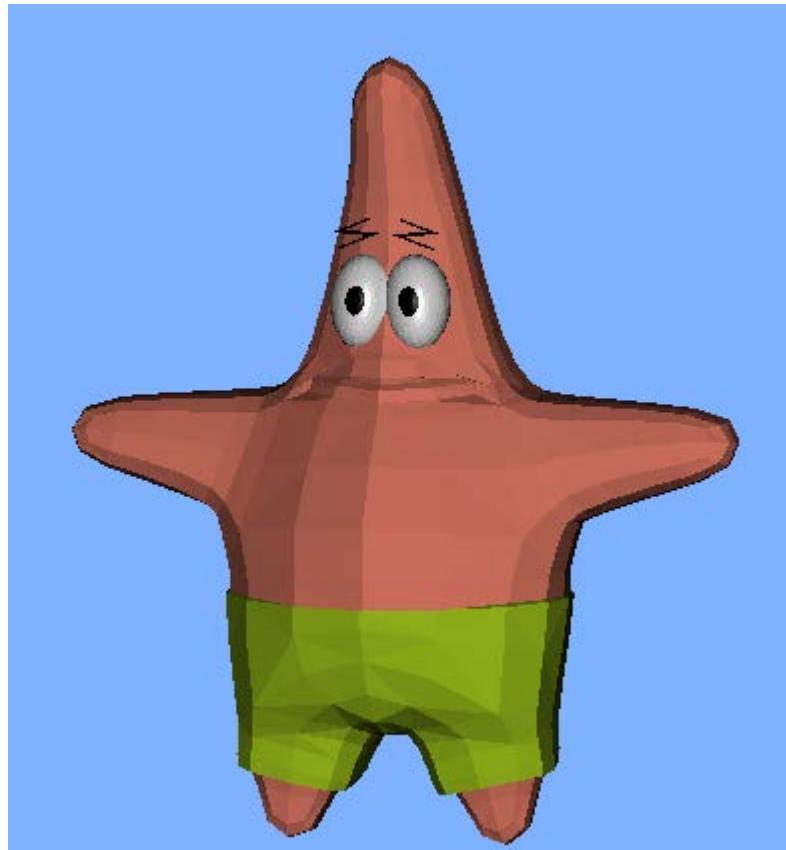
- Normal per cara vs normal per vèrtex



$$\overrightarrow{N_v} = \frac{\sum_i \overrightarrow{N_i}}{\left\| \sum_i \overrightarrow{N_i} \right\|}$$

Suavitzat d'arestes: exemple

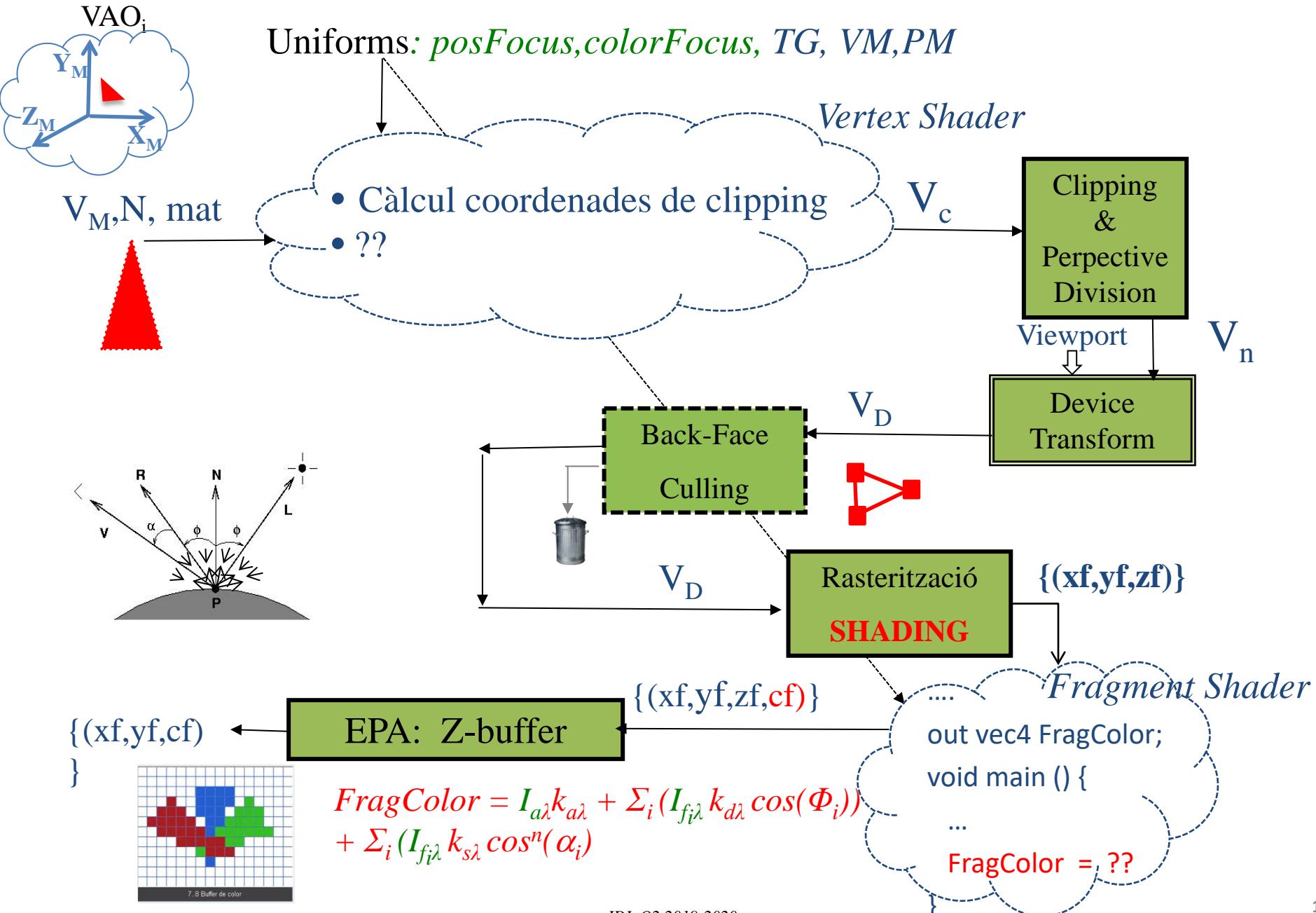
- Normal per cara vs normal per vèrtex



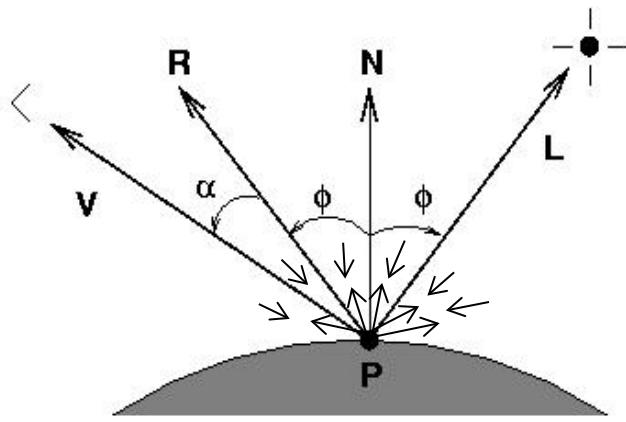
Classe 7: contingut

- Realisme: Il·luminació (2)
 - Il·luminació en OpenGL 3.3 (1)
 - Càcul de color en vèrtexs (en el Vertex Shader)
 - Shading de polígons
 - Suavitzat d'arestes
 - **Il·luminació en OpenGL 3.3 (2)**
 - Càcul de color en fragments

Càcul del color per fragment en un Fragment Shader (1)



Càcul del color per fragment en en Fragment Shader (2)



$$FragColor = I_{a\lambda} k_{a\lambda} + \sum_i (I_{f_i\lambda} k_{d\lambda} \cos(\Phi_i)) + \sum_i (I_{f_i\lambda} k_{s\lambda} \cos^n(\alpha_i))$$

$\cos(\Phi) \Rightarrow \text{dot}(L, N) \text{ en } SCO$

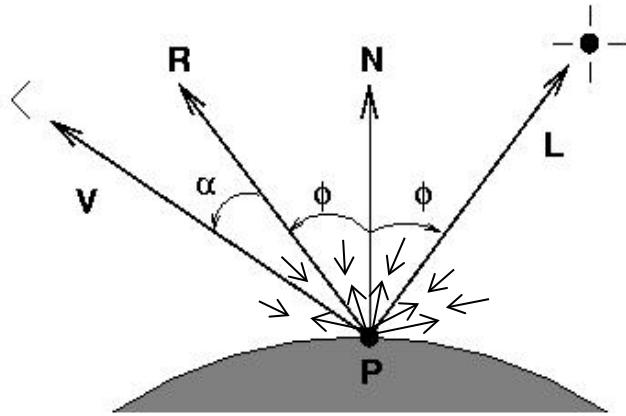
$\cos(\alpha) \Rightarrow \text{dot}(R, v) \text{ en } SCO$

Idea 1: Per cada píxel (fragment)

- ✓ Tenim la informació de les llums, són *uniforms*
- Requereix el vèrtex, altres vectors en SCO o SCA i les constants material
 - ✓ Tenim el fragment en SCD i matrius (uniforms) => *podríem calcular les coordenades del punt que es projecte en SCO o SCA aplicant inversa de les matrius.*
 - Procés costós
 - No tenim accés a la Normal ni a les constants empíriques del material (eren atributs/in en VS)

ALTRE IDEA?

Càcul del color per fragment en un Fragment Shader (3)



$$FragColor = I_{a\lambda}k_{a\lambda} + \sum_i (I_{f_i\lambda} k_{d\lambda} \cos(\Phi_i)) + \sum_i (I_{f_i\lambda} k_{s\lambda} \cos^n(\alpha_i))$$

$\cos(\Phi) \Rightarrow \text{dot}(L, N) \text{ en } SCO$

$\cos(\alpha) \Rightarrow \text{dot}(R, v) \text{ en } SCO$

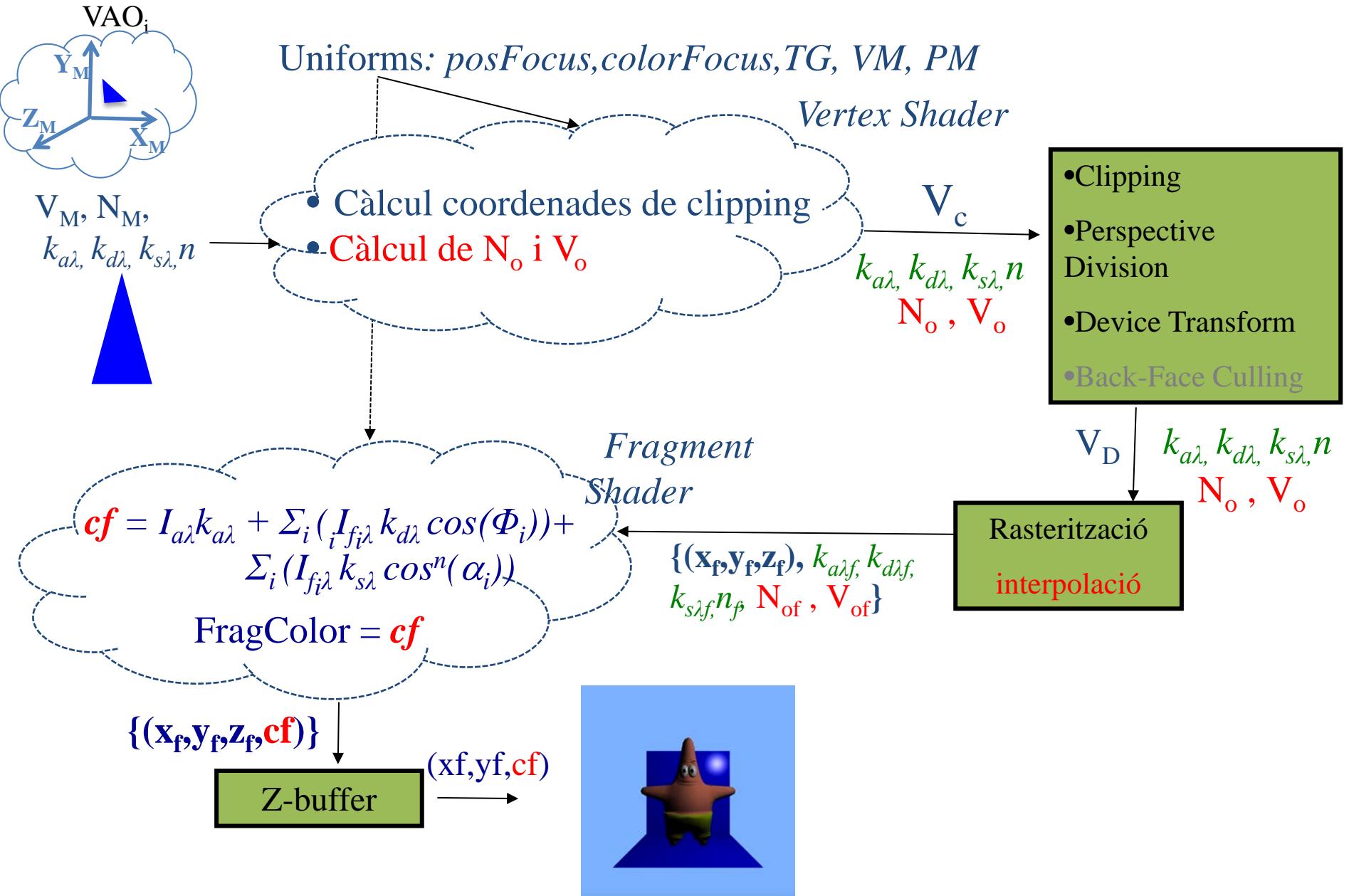
Proposta de solució:

- ✓ Tenim la informació de les llums, són *uniforms*
- ✓ Podem fer “out” en el VS dels atributs associats al vèrtex: N i V en SCO, i de les constants empíriques de material . La rasterització calcularà el seu valor pel fragment interpolant la informació dels vèrtexs del triangle → tindrem els seus valors en el FS com variables “in” 😊

Observació:

- La normal del fragment és una aproximació de la normal del punt del triangle que es projecta en el fragment. Aquesta interpolació de la normal es coneix com “shading de phong”

Càcul del color per fragment en en Fragment Shader (4)



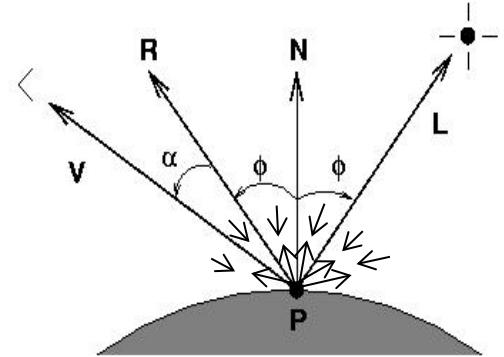
Càcul del color per vèrtex en el Vèrtex Shader (5)

Pseudocodi de Fragment Shader per calcular color en SCO:

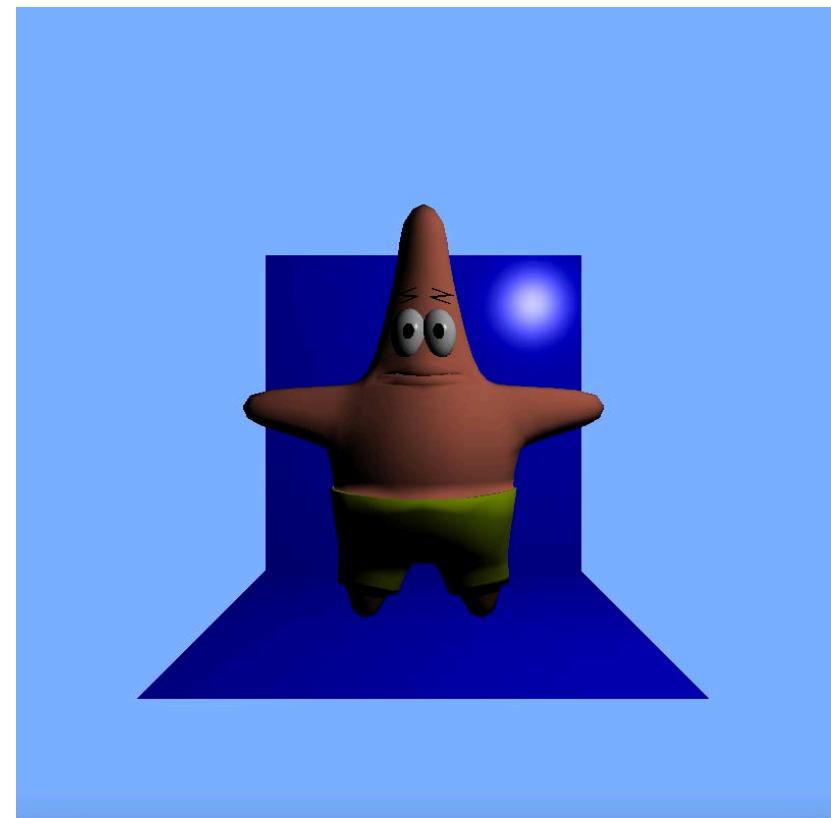
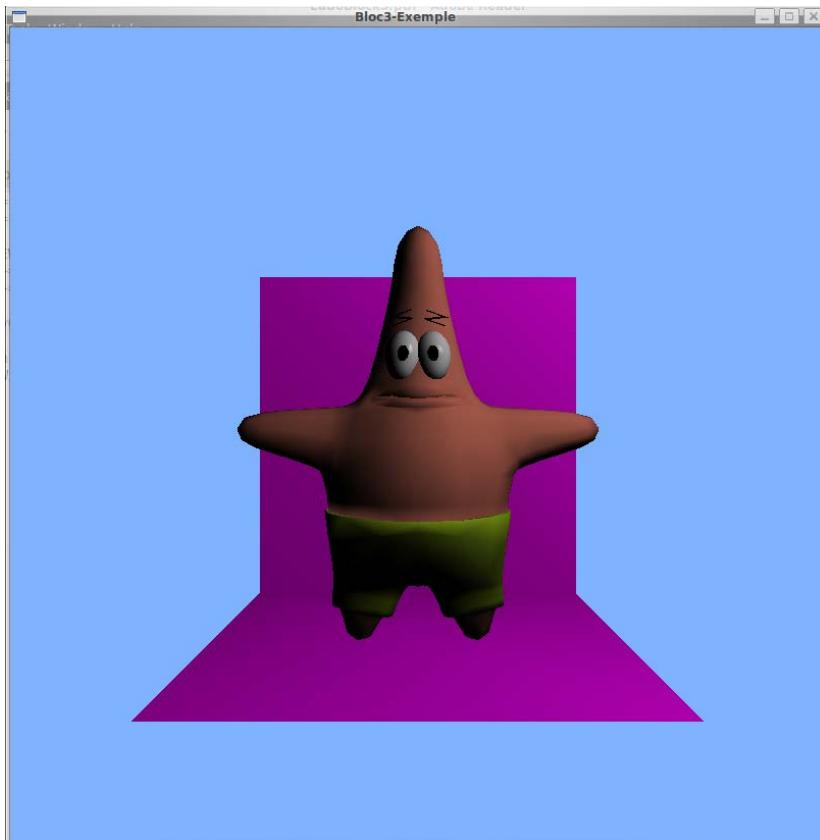
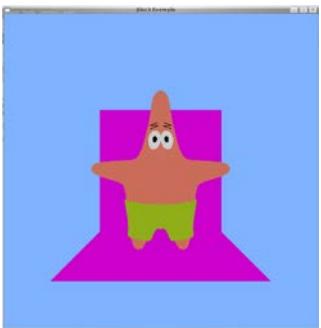
$$c_v = I_{a\lambda} k_{a\lambda} + \sum_i (I_{f_i\lambda} k_{d\lambda} \text{dot}(\mathbf{N}_o, \mathbf{L}_o)) + \sum_i (I_{f_i\lambda} k_{s\lambda} \text{dot}(\mathbf{R}_o, \mathbf{v}_o)^n)$$

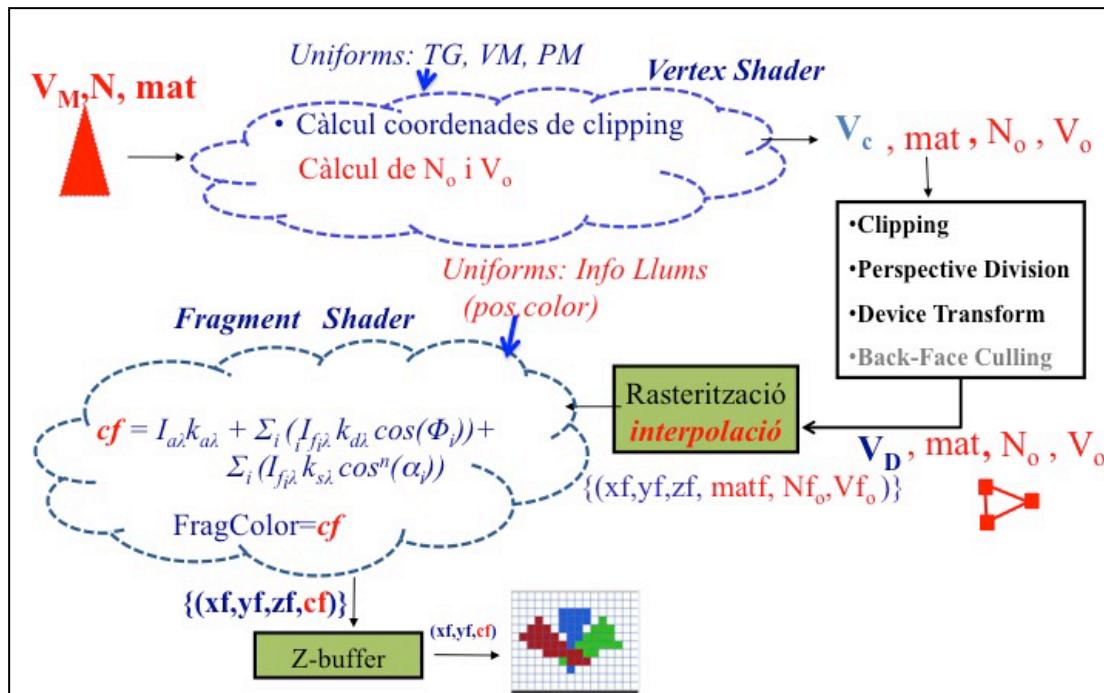
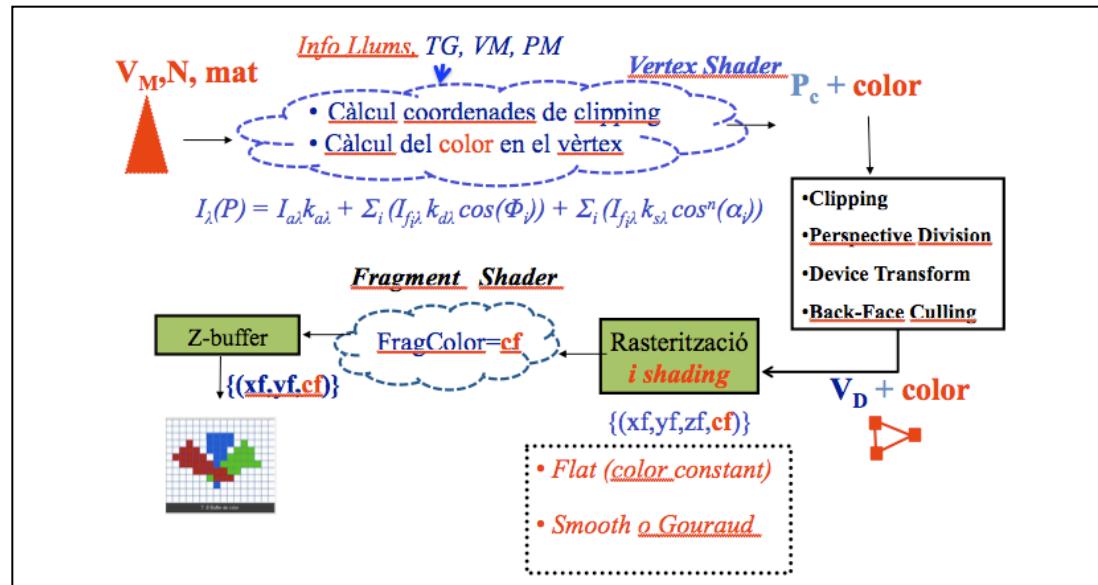
```
in vec3 NSCO, VSCO;  
in vec3 fmatamb, fmatdiff, fmatspec;  
in float fmatshin;  
uniform mat4 proj, view, TG;  
uniform vec3 posFocus, Ia, If;  
out vec4 FragColor;
```

```
...  
void main()  
{  
    vec4 focusSCO = view * vec4 (posFocus, 1.0);  
    vec3 NSCO_norm = normalize (NSCO);  
    vec3 L = normalize (focusSCO.xyz - VSCO.xyz);  
    FragColor = vec4 (color_vertex (NSCO_norm, L, -VSCO), 1);  
}
```



Exemple final: Sense il·luminació i Il·luminació en el VS versus FS





Conceptes i preguntes

- Càcul del color en el vèrtex shader. Quants VBOs requereix per VAO d'un model? Quina informació requereix? Cal fer algun càlcul en el fragment shader? La geometria en quin sistema de coordenades de referència ha d'estar?
- Limitacions del càlcul del color en el vèrtex shader.
- Suavitzat d'arestes. Quan cal fer-lo? Quina informació requereix? Degut a quin procés del procés de visualització es pot realitzar? Requereix algun càlcul extra en el vèrtex i/o en el fragment shader?
- Càcul del color en el fragment shader. Quina informació requereix? Com accedeix a aquesta informació?
- Perquè el càlcul de la il·luminació d'una escena és més costós si es realitza en el FS que en el VS (la fórmula és la mateixa)?
- Quines limitacions en el realisme d'una escena es resolen calculant la il·luminació en el FS enllloc del VS? Quin experiment faries per assegurar-te executant un programa si calcula la il·luminació en el VS o en FS?
- Fer els exercicis proposats de la col·lecció de problemes.

Exercici 5:

Una escena està formada per tres cubs d'aresta 2, centrats als punts $(-5, 0, 0)$, $(0, 0, 0)$ i $(5, 0, 0)$ i amb cares paral·leles als plans de coordenades. Els cubs són de color magenta mat.

Ubiquem un focus de llum blanca en la posició $(0, 0, 0)$. No hi ha llum ambient. De quin color s'observaran les cares dels cubs ubicades en $x=6$ i $x=-4$?

Observació: la ubicació de la càmera permet veure totes dues cares.

- a) Es veuran negres perquè el focus de llum està dins del cub central en $(0, 0, 0)$
- b) Si es té activat el *back-face culling*, es veuran les dues cares de color magenta, més fosca la de $x=6$ perquè està més lluny del focus
- c) Es veurà la cara en $x=6$ negra i la $x=-4$ de color magenta
- d) Si es té activat el *back-face culling*, es veuran les dues cares de color magenta, més fosca la de $x=-4$

Solució: c)

Justificació: El focus de llum està situat al centre del cub del mig, i per tant en la realitat hauria de quedar tot fosc, però els models que utilitzem recordem que són locals, i per tant no tenen en compte els altres objectes. És a dir, el focus de llum es troba, respecte al pla que determina la cara $x=-4$, en el semiespai on no es troba el seu cub (la cara “mira” cap al semiespai on es troba el focus de llum) i, per tant, com els altres objectes no compten, ningú la tapa i li li arriba la llum, per tant es veurà magenta. Per contra, respecte a la cara en $x=6$ el focus es troba en el mateix semiespai que es troba el sòlid del cub al que pertany (la cara no “mira” cap al semiespai on es troba el focus de llum), i per tant aquesta cara no es veurà il·luminada (es veurà negre).

Exercici 6:

Una escena està formada per dos cubs amb les cares paral·leles als plans de coordenades. El CUB1 té aresta 20, el centre de la seva base en (0,0,0) i és de color verd i mate; el CUB2 té aresta 20, centre de la seva base en (30,0,0) i és del mateix color verd però brillant. Il·luminem l'escena amb un focus groc situat en (50,10,0). L'observador es troba en una posició que pot veure les cares dels cubs ubicades en x=10 i x=40. Si es pinta l'escena amb OpenGL utilitzant model d'il·luminació de Phong en VS i Smooth shading (Gouraud Shading), de quin color es veuran aquestes cares? No hi ha llum ambient.

- a) La cara en x=10 és veurà de color verd constant, la cara en x=40 també és veurà de color constant però d'un verd més fosc.
- b) La cara en x=10 és veurà de color verd constant, la cara en x=40 també és veurà de color constant però d'un verd més clar.
- c) La cara en x=10 és veurà de color verd constant, la cara en x=40 també és veurà de color constant però d'un verd més clar i amb una taca espelular groga en mig de la cara.
- d) La cara en x=10 és veurà amb diferents tonalitats de verd, la cara en x=40 també és veurà amb diferents tonalitats de verd però més clars i amb una taca espelular groga en mig de la cara.

Solució: a)

Justificació: Les opcions c) i d) no poden ser perquè amb el càlcul de la il·luminació en el Vertex Shader no es poden veure taques espelulars en el mig d'una cara (això només pot ser si el càlcul es fa en el Fragment Shader). Les opcions a) i b) totes dues diuen que el color de les dues cares és constant, per tant és clar que l'observador no veu taca espelular en cap dels vèrtexs. Sabent que és així, i tenint en compte que la cara que està en x=40 està més aprop del focus de llum que la cara en x=10, està clar que la cara en x=10 (que està més lluny) està més il·luminada que la que està més aprop (perquè en els vèrtexs d'aquesta última l'angle entre N i L és més gran, per tant el cosinus de l'angle que intervé en el model difús és més petit).

Exercici 7:

Un cub amb constants de material $K_d=(0.8,0,0.8)$ i $K_s=(1,1,1)$ i $N=100$, és il·luminat amb un focus que emet llum de color $(1,1,0)$. No hi ha llum ambient. La càmera (correctament definida) és axonomètrica i l'observador i el focus estan a una distància 10 d'una cara (i mirant cap a ella) sobre una recta que és perpendicular a la cara i que passa pel seu centre. Indica, suposant càlcul d'il·luminació en el Vertex shader,

- quins colors observa l'observador en el cub si s'utilitza *FLAT shading* (colorat constant)? Indica els colors dels vèrtexs.
- quins colors observa l'observador en el cub si es pinta amb *(colorat de Gouraud)?*

Tant en apartat a) com en apartat b) la cara es veurà igual i de color constant en tots dos casos, tal i com està situat l'observador no pot veure cap reflexió especular en cap vèrtex i l'angle entre la normal N en cada vèrtex i la direcció d'incidència de la llum en ell és igual per a tots 4 vèrtexs.

El color vindrà donat per la component difusa del càlcul del color en un punt, i per tant els 4 vèrtexs tindran el mateix color i no importa com es calculi/interpoli el color en punts interiors (Flat o Smooth shading), tots seran del mateix color.

El color de tota la cara serà:

$$(0.8, 0, 0.8) * (1, 1, 0) * \cos(\text{angle}(N, L)) \rightarrow \text{Es veurà de color vermell no gaire intens.}$$

Exercici 8:

Volem il·luminar un polígon de 10x10 ubicat sobre el pla XZ i centrat en l'origen, amb un focus de llum blanca ubicat en la posició (0,2,0). No hi ha llum ambient. La normal del polígon és (0,1,0). Les constants de material del polígon són $K_d=(0,0,8,0)$, $K_s=(1,1,1)$ i $Shininess= 100$. Indica quina de les següents afirmacions és la correcta:

- a) Com la llum ha d'estar fixa en l'escena, el càlcul de la il·luminació s'ha de fer obligatòriament en el vèrtex shader per a cada vèrtex del polígon.
- b) Si el càlcul de la il·luminació es realitza en el fragment shader, cal passar la posició de la llum i la normal a coordenades de dispositiu.
- c) **Si el càlcul de la il·luminació es realitza en el vèrtex shader, cal que les posicions del vèrtex, del focus i la normal estiguin referenciades totes respecte al sistema de coordenades de l'aplicació o de l'observador.**
- d) La imatge -acoloriment- que s'obtindrà del polígon serà la mateixa tant si els càlculs es realitzen en el vertex com en el fragment shader; sempre que es realitzin en el sistema de coordenades adient.

Solució: c)

Exercici 9:

Una escena està formada per dos cubs d'aresta 2 amb cares paral·leles als plans coordenats i centres als punts $(0, 1, 0)$ i $(3, 1, 0)$. El primer és vermell i el segon verd. Ambdós són matis.

Per error s'ubica a l'usuari a la posició $(0, 1, 0)$ amb VRP al $(3, 1, 0)$. L'òptica és ortogonal amb un $window = (-4, 4, -4, 4)$, $zN = -1$, $zF = 6$. S'ubica una llum blanca a $(8, 1, 0)$. Si no hi ha llum ambient, i el *background* és blau, indica què es veurà en funció del mètode d'eliminació de parts amagades que s'utilitza:

- a) Si només s'empra *back-face culling*: un quadrat de color negre
- b) Si tenim *zbuffer* i *back-face culling* activats: un quadrat de color verd
- c) Si només tenim el *zbuffer* activat: un quadrat de color vermell
- d) Si només tenim el *back-face culling* activat: un quadrat de color verd

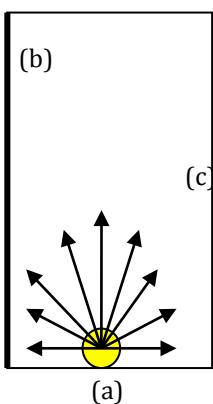
Solució: a)

Justificació: L'observador està situat en $(0, 1, 0)$ **PERO tenim una càmera ortogonal i zNear=-1**, per tant tenim que la direcció de visió és $(1, 0, 0)$ i **totes** les cares estan dins del volum de visió. La cara que s'hauria de veure, doncs, si fem només z-buffer, és la que està en $X=-1$ que com la llum li arriba per darrera és veurà negra, per tant la c) és falsa.

EL *back-face culling* eliminarà les cares que estan en $x=1$ i $x=4$ perquè la seva normal és $nz=(1, 0, 0)$ (no mira cap a la direcció de visió). Independentment de l'ordre en què s'enviïn a pintar els cubs (i tinguem o no activat el *z-buffer*), les dues cares que queden "potencialment visibles" (com a resultat del *back-face culling*), reben la llum per darrera i per tant es veuran negres.

Exercicis de realisme: el·limació parts amagades i il·luminació¹

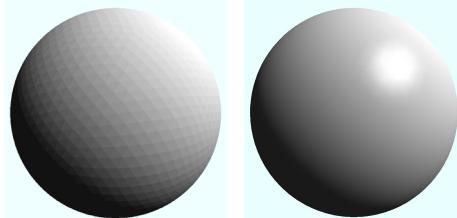
1. (2004-2005 1Q) Una esfera brillant de metall que es veu groga quan s'il·lumina amb llum blanca, la posem en una habitació que té llum ambient (.5, .5, .5) i un únic focus, de llum verda, situat 2 metres damunt de la càmera (en direcció de l'eix y). Quines zones distingirem en la visualització de l'esfera i de quins colors seran? Justifiqueu la resposta en relació a les propietats del material de l'esfera i les llums. Imagineu que es calcula el color en cada punt de l'esfera.
2. (2004-2005 1Q) Ens movem interactivament per una habitació que té llum ambient i un focus de llum a una de les seves parets (a). Veiem que una de les parets laterals (b) té una il·luminació que no canvia mentre ens movem, tot i que és més clara prop del focus i més fosca lluny del focus. En una altra de les parets (c) veiem perfectament el reflex del focus mentre ens anem movent. Justifiqueu el material de les parets i com ha de ser el seu model geomètric 3D.



3. (2004-2005 1Q) Quin efecte té en l'algorisme de z-buffer: (a) No efectuar culling. (b) Incrementar el nombre de bits de cada component del depth-buffer.
4. (2004-2005 2Q) Raona amb quins valors inicialitzaries les constants empíriques del material kd i ks d'un objecte que té el següent comportament: els reflexos especulars sempre es veuen del mateix color que la llum del focus i la resta de zones il·luminades pel focus es veuen de color groc si el focus és groc i del mateix color que les zones no il·luminades pel focus quan el focus és de color blau.
5. (2004-2005 2Q) La continuïtat visual de color en el veïnatge d'una aresta permet "suavitzar" les arestes d'objectes amb cares planes que aproximen cares corbades. Justifiqueu com s'aconsegueix l'esmentada continuïtat visual atenent a l'equació dels models empírics d'il·luminació i del shading (acoloriment) de polígons.
6. (2004-2005 2Q) Disposem de dos cubes amb les seves cares paral·leles als plans coordenats, longitud d'aresta igual a 2 i centres als punts (2,1,2) i (5,1,2) respectivament. Els cubes són de metall gris i s'il·luminen amb un focus de llum verda situat al punt (20,1,2). Com és possible que la cara del cub_1 situada en x=3 es vegi il·luminada si el cub_2 li fa ombra?. Quines altres cares es veuran il·luminades pel focus?
7. (2005-2006 1Q). Una escena està formada per dos objectes: una esfera de color groc d'aspecte mat i un cub amb les cares cobertes de miralls rosats. Especifiqueu quin podria ser el seu material (constants empíriques: k_a , k_d , k_s , n). Justifiqueu la resposta.

¹ Els exercicis estan basats en preguntes d'examen de l'assignatura VIG del pla 2003 de l'Enginyeria o de l'actual IDI (a partir del curs 2011-2012). La solució de quasi tots ells està en la web de la DAFIB i/o de la biblioteca. Millor comenceu pels exercicis relacionats amb IDI (els darrers).

8. (2005-2006 1Q) Si tant el *back-culling* com el *z-buffer* són algoritmes que permeten determinar les parts visibles d'una escena, ¿per què, usualment, s'apliquen conjuntament les dues tècniques en el procés de visualització? Raona la resposta.
9. (2005-2006 1Q). Les imatges (a) i (b) corresponen a dues visualitzacions d'una esfera aproximada per cares planes i han estat obtingudes utilitzant el procés de visualització amb *z-buffer* sense modificar ni la posició de la càmera, ni el focus de llum. Indica i raona el model d'il·luminació i la tècnica de shading (acoloriment) utilitzats per obtenir cada imatge.



10. (2005-2006 2Q) Tenim un objecte que es veu de color groc fosc quan no hi ha cap focus de llum i la llum ambient es $(0.2, 0.2, 0.2)$. L'objecte és mate. De quins colors el veurem quan s'il·lumina simultàniament amb dos Focus de llum: un de vermell i un de blau? (Cal suposar que el color dels focus no té component ambient i que estan situats en posicions diferents).
11. (2005-2006 2Q) Volem visualitzar una escena formada per dues esferes aproximades per una triangulació convexa. Quina diferència hi haurà entre pintar-la utilitzant com a tècnica de detecció de visibilitat *z-buffer* o només *back-face culling*? De què depèn? (Nota: qualsevol altra inicialització és la mateixa en ambdós casos).
12. (2005-2006 2Q) Una aplicació dibuixa una esfera brillant de color blanc girant sobre si mateixa. Amb una càmera donada, veiem en pantalla que les dues taques especulars produïdes en l'esfera per dos focus (un verd i un vermell) fluctuen de mida. A què es degut? Com reduiries el problema? (Nota: les llums estan fixes respecte a coordenades de l'observador).
13. (2006-2007 1Q) Estem visualitzant una esfera aproximada per una malla de triangles, i observem que la veiem tota suavitzada (no distingim els triangles) excepte un dels triangles, que el podem veure clarament al mig de la superfície suavitzada. Com es possible? Raona la resposta en base al model de colorejat ("shading") que creguis que s'ha utilitzat i com s'han enviat a pintar els triangles.
14. (2006-2007 1Q) Estem visualitzant un objecte curvat que rep llum ambient, llum d'un focus blanc i llum d'un focus verd. Els dos focus es troben en posicions fixes respecte al Sistema de Coordenades de l'Aplicació. Observem dos efectes: quan movem l'observador, ens canvia el color d'algunes zones de l'objecte; però en canvi, el color no canvia si encenem o apaguem el focus de color verd. Què podeu dir sobre el material d'aquest objecte ? Inicialitzeu els valors de les constants del material.
15. (2006-2007 2Q) Estem visualitzant una esfera en OpenGL, il·luminada per un focus de llum groga. La part no il·luminada pel focus es veu de color gris, la part il·luminada es veu amb una gradació de verd, i s'hi observa una taca espelcular de color groc. És possible? En cas afirmatiu, indica els paràmetres del material de l'esfera i de la llum ambient.
16. (2006-2007 2Q) Imaginem que tenim definit un laberint sobre una retícula de 10×10 amb cel·les buides i cel·les plenes (tipus paret). Situem 3 llums puntuals de colors vermell, verd i blau al punt mig del sostre de les cel·les de tipus "paret" (1,5), (5,5) i (9,5), respectivament. Les cel·les (3,5) i (7,5) no tenen "paret" (estan buides). Totes les cel·les es pinten havent definit un mateix material amb $k_{d\lambda}=k_{a\lambda}=(0.5, 0.5, 0.5)$ i $k_{s\lambda}=(0, 0, 0)$. Sense realitzar cap càcul explícit, raona de quin color veurà l'observador la cara superior de les cel·les (1,5), (3,5) i (7,5). Nota: la cel·la (X, Z) representa la cel·la que té les seves coordenades mínimes en $x=X$ i $z=Z$.
17. (2007-2008 1Q) Una escena està formada per un conjunt d'objectes. Hi ha dos focus de llum encesos. Un està ubicat en la posició $(F1.x, F1.y, F1.z)$ en coordenades de l'aplicació. L'altre es mou per l'escena i es disposa d'una funció `PosicioFocus(t, F2.x, F2.y, F2.z)` que retorna la seva

posició en coordenades de l'aplicació en un instant determinat t. La càmera està ubicada i orientada segons (OBS1,VRP1,up1). Podeu suposar que existeix una acció pinta_escena() encarregada de pintar l'escena. Escriviu i justifiqueu el tros de codi OpenGL requerit per a la ubicació de les posicions de les llums i de la càmera, així com la crida a l'acció de pintat. No cal que definiu el tipus de càmera.

18. (2007-2008 1Q) Suposeu que tenim un triangle de vèrtexs $V1=(2,0,0)$, $V2=(-2,0,-2)$ i $V3=(-2,0,2)$ i volem calcular la il·luminació que fa en ell un focus de llum puntual de color blanc situat a la posició $(2,4,0)$. Suposant que usem el model d'il·luminació de Phong, que l'observador es troba a la posició $(2,4,0)$ i que el material del triangle té propietats $ka=(0,0,0)$, $kd=(0,0,0.8)$, $ks=(0.8,0.8,0.8)$ i $n=100$, indiqueu, justificant la resposta, com es veurà pintat el triangle a la vista en els següents casos (no es necessari que realitzeu càlculs explícits, simplement raoneu el resultat):
 - a) La normal als tres vèrtexs és la normal del triangle. La cara mira cap al semiespai on es troba el focus de llum i s'usa colorat (shading) constant.
 - b) La normal és diferent per a cada vèrtex. La normal al vèrtex $V1$ és $(0,1,0)$. La normal als vèrtexs $V2$ i $V3$ és $(-1,1,0)$. S'usa colorat de Gouraud.
19. (2007-2008 1Q) Un estudiant ha implementat una aplicació en la que només hi ha un llum d'escena però per error l'ha col·locat DINS d'un edifici. Els edificis són objectes convexos tancats. En visualitzar l'escena des d'una càmera exterior a l'edifici, de quin color es veurà aquest edifici? De quin color es veuran la resta d'edificis?
20. (2007-2008 1Q) Els models empírics d'il·luminació d'OpenGL tenen certes limitacions quan calculen la il·luminació d'una escena. Per exemple, no es contemplenombres ni miralls. A què és degut? Tanmateix, en les visualitzacions podem observar algunes cares més fosques, és a dir, no els arriba la llum del focus. Com és possible tenir aquest efecte si s'utilitzen models empírics?
21. (2008-2009 1Q) Una escena amb dues esferes A i B (representades per un poliedre convex) es mira des d'una posició on l'esfera B queda davant de la A i la tapa parcialment. Usant per a l'eliminació de parts amagades l'algorisme de culling, ens trobem que la visualització obtinguda no és correcta perquè el tros de l'esfera A que ha de ser tapat per l'esfera B es veu en la imatge. Perquè no està funcionant correctament l'eliminació de parts amagades? Com ho solucionaries?
22. (2008-2009 1Q) Tenim una esfera el material de la qual està definit com: $Ka = (0.5,0.5,0)$, $Kd = (0,1,1)$, $Ks = (1,1,1)$. Suposant que està il·luminada per un únic focus de llum, indica els colors del focus de llum i de la llum ambient per a què a l'esfera es vegin els següents efectes: color verd fosc allà on no il·lumina el focus de llum i, en la zona il·luminada pel focus, una gradació de blaus i una taca clarament més lluminosa de color magenta (tot amb un fons verd fosc).
23. (2008-2009 1Q) Una escena està formada per diverses esferes de diferents grandàries però totes de mateix material. Observem en cadascuna d'elles 3 taques especulars blanques i ubicades, pràcticament, en la mateixa posició relativa en cada esfera.
 - Creus que és possible? En cas afirmatiu, què podries dir de l'entorn d'il·luminació (nombre de focus, colors, ubicació,...) i de la posició de l'observador?
 - Les taques tenen forma poligonal, per què?
24. (2008-2009 1Q) Tenim una escena formada per dos cubs i l'estem visualitzant amb un càmera perspectiva que està definida (matrius viewMatrix i projectionMatrix inicialitzades). La volem il·luminar per tres focus de llum: un ha d'estar situat al centre del segment que uneix el centre dels cubs (punts C1 i C2); altre sempre estarà ubicat 10 unitats per sobre de l'observador i el tercer el volem ubicar 5 unitats a la "dreta visual" del centre del segon cub (punt C2). Un cop ubicat aquest tercer focus, ha de quedar en una posició fixa respecte de l'escena, encara que es modifiqui la càmera. Indiqueu el codi OpenGL per a ubicar els tres focus. Observació: entenem per "dreta visual" que el focus estigui desplaçat de la projecció de C2 en el viewport sobre una recta que es veu paral·lela al costat horitzontal del viewport.

25. (2008-2009 2Q) Tenim un cub de costat 20 centrat a l'origen, i una càmera que té l'observador a (0,8,0), el vrp a (0,0,0) i el vector up (0,0,1) i amb un window de (-20, 20, -20, 20) i znear=0.1 i zfar=20. Si visualitzem aquesta escena amb el culling activat, el z-buffer activat i un focus de llum blanca situat a la posició de l'observador, veiem que la visualització no mostra res en la imatge al viewport. Per què?
26. (2008-2009 2Q) Un cub amb constants de material $K_d=(0.8,0,0.8)$ i $K_s=(1,1,1)$ i $N=100$, és il·luminat amb un focus que emet llum de color (1,1,0). No hi ha llum ambient. La càmera (correctament definida) és axonomètrica i l'observador i el focus estan a una distància 10 d'una cara (i mirant cap a ella) sobre una recta que és perpendicular a la cara i que passa pel seu centre. Indica:
- quins colors observa l'observador en el cub si s'utilitza *FLAT shading* (colorat constant)? Indica els colors dels vèrtexs en RGB i HSB.
 - quins colors observa l'observador en el cub si es pinta amb *(colorat de Gouraud)?*
27. (2008-2009 2Q) Volem visualitzar un cilindre aproximat per cares planes. Tenim un vector amb les coordenades de tots els vèrtexs i altre vector que ens permet coneixer, per a cada cara, la seva normal i els índexs dels seus vèrtexs. Sabem que les dues primeres cares de la llista de cares són les tapes del cilindre. Volem pintar el cilindre suavitzant només les arestes entre les cares que aproximen la part corbada del cilindre. Les arestes de les tapes del cilindre no s'han de suavitzar. Indica l'algorisme requerit per a fer el pintat de la geometria. Si s'escau, pots modificar/afegir informació a l'estructura de dades tot indicant com es calcula. Observació: No cal definir la càmera.
28. (2009-2010 1Q) Explica en què consisteix el shading de Gouraud, i en què el shading de Phong. Explica quin és el seu propòsit. Quin fa servir OpenGL?
29. (2009-2010 1Q) Amb la il·luminació apagada i fent servir OpenGL, dibuixem un triangle tal que els seus tres vèrtexs A, B i C cauen en coordenades de dispositiu (10; 10), (50; 50), (10; 100) respectivament, fent servir una càmera ortogonal. Com a resultat de la rasterització, el píxel que es troba a la posició (10; 55) canvia a un color magenta pur (0.5; 0; 0.5). Què ens permet deduir això del color dels tres vèrtexs A, B i C si el mode de pintat és GL_FLAT? I si és GL_SMOOTH? Raoneu les respuestes.
30. (2009-2010 1Q) Donada una esfera amb constants de material: $K_a = (0.2; 0.2; 0.2)$, $K_d = (0.8; 0; 0.8)$, $K_s = (1; 1; 0)$. Suposant que s'il_lumina _unicament per un focus de llum, indica, justificant, quins haurien de ser els paràmetres de llum del focus (posició i color, sense distingir entre components ambient, difusa i especular) per a poder observar els següents efectes en l'esfera (en cas que l'efecte no sigui possible també has d'indicar-ho).
- S'observa la silueta de l'esfera d'un color gris molt fosc i en la resta de l'esfera una gradació de colors magenta.
 - S'observa la silueta de l'esfera d'un color gris molt fosc i en la resta de l'esfera una gradació de colors vermells amb una taca més lluminosa de color groc.
31. (2009-2010 2Q) Tenim una paret il·luminada per un únic focus de llum. La il·luminació es calcula usant únicament el model d'il·luminació de Phong. La paret està pintada amb una pintura brillant. Si s'està dibuixant amb OpenGL amb colorat GL_SMOOTH, veurem diferències si la geometria de la paret és un únic polígon rectangular o si està formada per una malla densa de triangles? Dóna una explicació detallada de perquè.
32. (2009-2010 2Q) En un programa, l'autor ha definit, per error, dues llums idèntiques (mateix tipus, posició i colors). En visualitzar una escena amb el programa, tal que les llums la il·luminen correctament, hi haurà alguna diferència apreciable si està activada sols una, o si ho estan les dues? La teva resposta és independent dels paràmetres de color de les llums i de l'escena que visualitzem?
33. (2010-2011 1Q) Tenim un polígon gris mate amb els vèrtexs V1(-10,0,10), V2 (10,0,10), V3

(10,0,-10) i V4 (-10,0,-10); ubiquem sobre cada un dels seus vèrtexs i a una distància de 0.5, un llum puntual de color blau, blau, vermell i verd, respectivament.

- Indiqueu les constants del material del polígon.
- Si les llums estan totes enceses i pintem el polígon utilitzant OpenGL amb colorat (*shading*) de Gouraud (*Smooth*), quina seria la distribució de colors en el polígon? Justifica la resposta. No hi ha llum ambient.
- Apaguem els focus blaus i ubiquem dos cubs del mateix material que el polígon, amb longitud d'aresta 1, i amb el centre de les seves bases en (5,0,-10) i (-5,0,-10); de quin color quedaran cadascuna de les seves cares?

34. (2010-2011 1Q) Per què diem que utilitzem models d'il·luminació locals en OpenGL? Quines limitacions tenen en quant al realisme de l'escena?
35. (2011-2012 1Q) Un estudiant vol fer una aplicació que genera una imatge que visualitza una pilota d'or. Ha generat el model geomètric en base a una aproximació de l'esfera per cares planes. Ara es pregunta quines constants de material li hauria de posar i quin model d'il·luminació i shading hauria d'utilitzar per a què l'esfera es vegi el més realista possible.
a) Raona la solució que faries i escriu el tros de codi d'OpenGL que envia a pintar l'esfera (totes les seves cares són triangles). Suposa un focus de llum blanca ja inicialitzat.
b) Suposant l'esfera centrada a l'origen de radi 2, l'observador al punt (0, 0, 10) i el focus de llum al punt (0, 0, 10), com seria la imatge de l'esfera a la vista? És a dir, quins colors es veurien en l'esfera a la vista?
36. (2011-2012 1Q) Un joc per computador vol simular un helicòpter amb un llum que segueix un cotxe que es mou per una ciutat (el focus de l'helicòpter il·lumina sempre al cotxe). Suposant que tenim les crides següents: `posicio_cotxe(x,y,z)` que retorna la posició del cotxe en un instant; `posicio_llum_helicopter(xh,yh,zh)` que retorna la posició de l'helicòpter; `pinta_cotxe()` que pinta el cotxe centrat a l'origen de coordenades; `pinta_helicopter()` que pinta l'helicòpter centrat a l'origen de coordenades; i `setcamera()` que contínuament actualitza la càmera per veure correctament en tercera persona l'escena. Es demana que doneu el tros de codi que pinta l'helicópter i el cotxe fent que la llum de l'helicòpter sigui puntual i de tipus SPOT.
37. (2011-2012 2Q) Volem pintar una escena amb 3 objectes i un focus de llum SPOT. Els objectes es pinten amb les crides `PintaObj1()`, `PintaObj2()` i `PintaObj3()` respectivament (no cal fer cap transformació per a pintar els objectes). El primer objecte est_a centrat al punt C1, el segon al punt C2 i el tercer al punt C3. Es vol tenir un focus de llum puntual i de tipus SPOT que vagi donant voltes al voltant de l'objecte 2, a una al_cada de 10 unitats sobre el seu centre i describint un cercle de radi R al voltant de l'eix vertical que passa pel centre de l'objecte. El focus SPOT ha d'enfocar sempre al centre de l'objecte 2. Escriu el tros de codi que permet visualitzar aquesta escena amb el focus de llum SPOT tal i com es demana. Suposa que la càmera està ja definida i no es modifica (pots suposar un mètode "`set camera()`" que ho fa si el necessites).
38. (2011-2012 2Q) Tenim dues esferes, de radis 10 i 2, una a dins de l'altra, amb els seus centres situats al mateix punt. Suposant que tenim una font de llum a l'observador que emet una llum (1,0,1) i que tots dos, observador i focus, es troben situats fora de les dues esferes i mirant cap al seu centre. Com creus que estar_a il·luminada l'esfera de dins, si aquesta té constants de material $K_a=(0,0,0)$, $K_d=(0.8,0.8,0)$, $K_s=(1,1,1)$ i $N=100$?
39. (2011-2012 1Q) Tenim una escena formada per 3 cubs sòlids de longitud d'aresta 2, centres de les seves bases en els punts (2,0,0), (5,0,0) i (8,0,0), cares paral·leles als plans de coordenades i d'un material mate de color vermell, verd i blau respectivament. Situem un focus puntual blanc en el punt (8,1,0), l'observador en (5,1,0), el VRP en (0,1,0), up (0,1,0) i una càmera perspectiva amb relació d'aspecte 1, $zN=0.5$, $zF=6$ i $FOV=90^\circ$. Quines cares dels cubs seran visibles en el `viewport` i de quin color es veuran si es pinta l'escena utilitzant OpenGL si per eliminar parts amagades s'utilitza: a) Només el *back-face culling* b) Activat només el *zbuffer*.

40. (2011-2012 1Q) Quines constants de material definiries si es vol que un objecte sigui de plàstic polit de color vermell? Raona la resposta.
41. (2011-2012 1Q) Per què diem que utilitzem models d'il·luminació locals en OpenGL? Quines limitacions tenen en quant al realisme de l'escena?
42. (2011-2012 2QP) Si tant el *back-face culling* com el *z-buffer* són algoritmes que permeten determinar les parts visibles d'una escena, ¿per què, usualment, s'apliquen conjuntament les dues tècniques en el procés de visualització? Raona la resposta.
43. (2011-2012 2QP) Donada una esfera amb constants de material: $K_a = (0.2; 0.2; 0.2)$, $K_d = (0.8; 0; 0.8)$, $K_s = (1; 1; 0)$, $N=100$. Suposant que s'il·lumina únicament per un focus de llum, indica, justificadament, quins haurien de ser els paràmetres de llum del focus (posició i color) per a poder observar el següents efectes en l'esfera (en cas que l'efecte no sigui possible també has d'indicar-ho): S'observa la silueta de l'esfera d'un color gris molt fosc i en la resta de l'esfera una gradació de colors magenta amb una taca més lluminosa de color blanc.
44. (2011-2012 2Q) Un terra molt polit (brillant) de color blau es modela amb un polígon de coordenades $(0,0,10)$, $(10,0,10)$ $(10,0,0)$ i $(0,0,0)$ i normal $(0,1,0)$. Existeix un focus de llum blanc d'escena a la posició $(5,5,5)$. Es visualitza utilitzant un codi OpenGL que utilitza "smooth shading", una càmera ubicada en $(10,5,10)$ que mira cap al $(5,0,5)$ i que permet veure tot el polígon, indica i justifica:
- Les constants empíriques del material del terra.
 - La distribució de colors que veurem en el terra
- Si ara el terra es pinta amb 100 quadrats de $1x1$ que cobreixen la mateixa àrea que el polígon original:
- La distribució de colors que veurem en el terra
45. (2011-2012 2Q) Una escena amb dues esferes A i B (representades cadascuna per un poliedre convex) es mira des d'una posició on l'esfera B queda davant de la A i la tapa parcialment. Usant per a l'eliminació de parts amagades NOMÉS l'algorisme de *back-face culling*, ens trobem que la visualització obtinguda no és correcta perquè el tros de l'esfera A que ha de ser tapat per l'esfera B es veu en la imatge.
- Per què no està funcionant correctament l'eliminació de parts amagades?
 - Com ho solucionaries?
46. (2012-2013 1Q) S'aproxima un cilindre per un prisma amb bases poligonals de 20 costats. S'inicialitza una càmera en una posició arbitrària que permet veure el cilindre optimitzant la seva ocupació en el viewport. Es visualitza utilitzant OpenGL amb z-buffer, il·luminació i shading de Gouraud (Smooth). Quines diferències qualitatives observaries en la imatge resultant si s'envia a pintar la geometria amb normal per vèrtex o normal per cara?
47. (2012-2013 2Q) Una escena està formada per dos cubs amb les cares paral·leles als plans de coordenades. El CUB1 té aresta 20, el centre de la seva base en $(0,0,0)$ i és de color verd i mate; el CUB2 té aresta 20, centre de la seva base en $(30,0,0)$ i és del mateix color verd però brillant. Il·luminem l'escena amb un focus groc situat en $(50,10,0)$. L'observador es troba en una posició que pot veure les cares dels cubs ubicades en $x=10$ i $x=40$. Si es pinta l'escena amb OpenGL utilitzant model d'il·luminació de Phong i Smooth shading (Gouraud Shading), de quin color es veuran aquestes cares? No hi ha llum ambient.
- a. **La cara en $x=10$ és veurà de color verd constant, la cara en $x=40$ també és veurà de color constant però d'un verd més fosc.**
 - b. La cara en $x=10$ és veurà de color verd constant, la cara en $x=40$ també és veurà de color constant però d'un verd més clar.
 - c. La cara en $x=10$ és veurà de color verd constant, la cara en $x=40$ també és veurà de color constant però d'un verd més clar i amb una taca specular groga en mig de la cara.
 - d. La cara en $x=10$ és veurà amb diferents tonalitats de verd, la cara en $x=40$ també és veurà amb diferents tonalitats de verd però més clars i amb una taca specular groga en mig de la cara.

48. (2013-2014 1Q) Una escena molt simple ha d'estar formada per un "terra" quadrat de costat 10, centrat a l'origen de coordenades i ubicat en el pla X-Z i amb costats paral·lels als eixos X i Z. El terra és groc brillant (està molt polit). S'il·lumina amb un focus blanc ($If=(0.8,0.8,0.8)$) ubicat en la càmera. La càmera permet inspeccionar l'escena en mode tercera persona (angles Euler) mirant sempre al centre de l'escena.

Un estudiant representa el "terra" mitjançant un quadrat de costat 10, mentre que un altre estudiant el representa mitjançant 10x10 quadrats de costat 1."

- Indica les constants de material que posaries al "terra".
- Si la càmera està en (0,5,0) com quedaria la visualització del "terra" dels dos estudiants? I si la càmera està en (5,5,5)?

49. (2013-2014 1Q) Una escena està formada per tres cubs d'aresta 2, centrats als punts (-5,0,0), (0,0,0), (5,0,0) i amb cares paral·leles als plans de coordenades. Els cubs són de color magenta mat. Ubiquem un focus de llum blanca en la posició (0,0,0). No hi ha llum ambient. De quin color s'observaran les cares dels cubs ubicades en $x=6$ i $x=-4$?

Observació: la ubicació de la càmera permet veure les dues cares.

- a. Es veuran negres perquè el focus de llum està dins del cub centrat en (0,0,0).
- b. Si es té activat el *back-face culling*, es veuran les dues cares de color magenta, més fosca la de $x=6$ perquè està més lluny del focus.
- c. **Es veurà la cara en $x=6$ negra i la $x=-4$ de color magenta.**
- d. Si es té activat el *back-face culling*, es veuran les dues cares de color magenta, més fosca la de $x=-4$.

50. (2013-2014 2Q) Es vol visualitzar un terra vermell polit de 10x10, el centre del terra està en (0,0,0) i la seva normal (0,1,0). Les constants de material són $ka=(0,0,0)$, $kd=(1,0,0)$, $ks=(1,1,1)$ i $n=100$. S'il·lumina amb un focus de llum groga ubicat en (0,5,0), l'observador està en (5,5,0). Un estudiant (E1) modela el terra amb un sol quadrat, mentre que altre estudiant (E2) ho fa amb 100 quadrats de 1x1 tots amb el mateix material. Si el codi OpenGL de cada estudiant és correcte (càmera i il·luminació), i si utilitzen *smooth shading*. Selecciona la resposta correcta.

- a. **E2 veurà una zona de vermell brillant al centre que es degrada a vermell fosc cap a les cantonades del terra virtual; i una taca especial groga a un costat de la zona vermella brillant. E1 veurà tot el polígon d'un color vermell constant.**
- b. Si utilitzen normal per vèrtex, les imatges d'ambdós estudiants seran similars i es veurà una zona de vermell brillant al centre que es degrada a vermell fosc cap a les cantonades del terra; i una taca especial groga a un costat de la zona vermella brillant.
- c. Si utilitzen normal per vèrtex, les imatges d'ambdós estudiants seran similars i es veurà una zona de vermell brillant al centre que es degrada a vermell fosc cap a les cantonades del terra; i una taca especial blanca a un costat de la zona vermella brillant.
- d. Si utilitzen normal per cara, E1 veurà tot el polígon de color vermell constant amb una taca especial blanca en un vèrtex; E2 veurà una zona de vermell brillant al centre que es degrada a vermell fosc cap a les cantonades del terra; i una taca especial blanca a un costat de la zona vermella brillant.

51. (2013-2014 2Q) Una escena està formada per dos cubs d'aresta 2 amb cares paral·leles als plans coordinats i centres als punts (0,1,0) i (3,1,0). El primer és vermell i el segon verd. Ambdós són mat. Per error, s'ubica l'usuari a la posició (0,1,0) amb VRP al (3,1,0). L'òptica és axonomètrica amb un $window=(-4,4,-4,4)$, $ZN=-1$, $zF=6$. S'ubica una llum blanca a (8,1,0). Si no hi ha llum ambient, i el *background* és blau, indica què es veurà en funció del mètode d'eliminació de parts amagades que s'utilitzen:

- a. **Si només s'empra *back-face culling*: un quadrat de color negre**
- b. Si tenim *zbuffer* i *back-face culling* activats: un quadrat de color verd
- c. Si només tenim el *zbuffer* activat: un quadrat de color vermell
- d. Si només tenim *back-face culling* activat: un quadrat de color verd

52. (2014-2015 1Q P) Tenim una esfera de radi 3 centrada a l'origen de coordenades i un focus de llum situat a la posició (0, 3, 5) de color (1, 1, 0). No hi ha llum ambient. Un observador es mira aquesta escena des de la posició (0, 0, 5) i mirant cap al centre de l'esfera, i el que observa és una esfera que té una part propera a la silueta per la part de baix de l'esfera de color negre, un degradat de colors verds que són més clars per la part de dalt de l'esfera i més foscos per la part de baix i una taca de color groc capa la part del mig de la semiesfera superior. Quines constants de material de l'esfera permeten que es pugui veure aquesta escena de la forma descrita?
- a) $K_a = (0, 0.2, 0)$, $K_d = (0, 0.8, 0)$, $K_s = (0, 0, 0)$ i $N = 100$
 - b) $K_a = (0.2, 0.2, 0.2)$, $K_d = (0.8, 0.8, 0.8)$, $K_s = (1, 1, 1)$ i $N = 100$
 - c) $K_a = (0, 0.2, 0.2)$, $K_d = (0, 0.8, 0.8)$, $K_s = (1, 1, 1)$ i $N = 100$**
 - d) $K_a = (0, 0.2, 0.2)$, $K_d = (0, 0.8, 0.8)$, $K_s = (0, 1, 1)$ i $N = 100$
53. (2014-2015 1Q)) Es vol pintar una escena que conté diferents objectes il·luminada amb un sol focus de llum i sense llum ambient. La il·luminació es pot activar i desactivar, i el focus de llum encendre i apagar. Quina de les següents respostes és la correcta?
- a. Si totes les cares d'un objecte són del mateix material, cada cop que es pinta cal definir totes les constants de material però només cal fer-ho un cop per objecte.**
 - b. Si totes les cares d'un objecte són del mateix material, i l'objecte no és especular, no cal indicar la constant empírica especular quan es pinta.
 - c. Si un objecte és vermell, les seves constants empíriques especular i difusa són vermelles.
 - d. Cada cop que s'activa la il·luminació cal declarar els colors de la llum.
54. (2014-2015 2Q) Tenim una escena amb una esfera de radi 2 centrada a l'origen. L'observador es troba a la posició (0, 0, 6) mirant cap al centre de l'esfera i tenim un focus de llum magenta $If=(0.8, 0, 0.8)$ situat a la posició (0, 2, 6) i no tenim llum ambient $Ia=(0,0,0)$. Quines seran les constants del material Kd , Ks i Shininess (N) de l'esfera si la visualització que obté l'observador té les següents característiques:
- Una zona molt fosca a la part de baix de l'esfera, per sota de l'equador de l'esfera
 - Una taca magenta per damunt de l'equador de l'esfera
 - Un degradat de color blau a la resta de l'esfera, essent més intens el blau prop de la taca magenta i menys intens com més s'allunya d'aquesta.
55. (2014-2015 2Q) Volem il·luminar un polígon de 10x10 ubicat sobre el pla XZ i centrat en l'origen, amb un focus de llum blanca ubicat en la posició (0,2,0). No hi ha llum ambient. La normal del polígon és (0,1,0). Les constants de material del polígon són $Kd=(0,0.8,0)$, $Ks=(1,1,1)$ i Shininess= 100. Indica quina de les següents afirmacions és la correcta:
- a) Com la llum ha d'estar fixa en l'escena, el càlcul de la il·luminació s'ha de fer obligatòriament en el vèrtex shader per a cada vèrtex del polígon.
 - b) Si el càlcul de la il·luminació es realitza en el fragment shader, cal passar la posició de la llum i la normal a coordenades de dispositiu.
 - c) Si el càlcul de la il·luminació es realitza en el vèrtex shader, cal que les posicions del vèrtex, del focus i la normal estiguin referenciades totes respecte al sistema de coordenades de l'aplicació o de l'observador.**
 - d) La imatge -acoloriment- que s'obtindrà del polígon serà la mateixa tant si els càlculs es realitzen en el vertex com en el fragment shader; sempre que es realitzin en el sistema de coordenades adient.
56. (2015-2016 Q1) Per a poder veure les escenes amb un cert realisme s'activa el *z-buffer* i s'implementa il·luminació de Phong en el *Vertex Shader*. Per a què tot funcioni correctament quan es pinta l'escena:
- a) En fer el càlcul d'il·luminació cal que la normal i els vectors L, R i V estiguin normalitzats.**
 - b) En fer el càlcul d'il·luminació cal que la normal, el vector L i el vèrtex estiguin normalitzats.

- c) Si en comptes de *z-buffer* s'utilitza *back-face culling*, no cal que s'indiqui la normal.
d) Si no s'ha fet cap escalat als objectes, no cal que la normal estigui normalitzada.
57. (2015-2016 Q1) En un examen de laboratori d'OpenGL es demana posicionar un focus de llum d'escena. Com podrà el professor detectar que realment l'estudiant ha definit un llum d'escena i no de càmera?
a) Ho podrà detectar si modifica el FOV per a fer un zoom.
b) Ho podrà detectar si modifica el color del focus de llum.
c) Només ho podrà detectar si mira el codi del *Vertex Shader*.
d) Ho podrà detectar si mou la càmera al voltant de l'escena.
58. (2015-2016 Q1) Es vol definir el material d'un objecte de manera que sigui de plàstic de color vermell brillant. Quines constants de material et semblen més apropiades?
a) $K_d=(0.5, 0, 0)$, $K_s=(0, 0, 0)$, $n=100$.
b) $K_d=(0.5, 0, 0)$, $K_s=(0.5, 0, 0)$, $n=1$.
c) $K_d=(0.5, 0, 0)$, $K_s=(1, 1, 1)$, $n=100$.
d) No es pot definir ni la K_s ni la K_d si no sabem el color de la llum.
59. (2015-2016 Q2) Un estudiant vol implementar una escena il·luminada per un focus fix en l'escena en la posició (0,0,0); però la seva implementació fa que el focus sigui de càmera. Per solucionar el problema:
a) Si ha implementat el càlcul de la il·luminació en el Vèrtex Shader, ha de multiplicar la posició del focus per la View Matrix i la Model Matrix (view * TG).
b) Si ha implementat el càlcul de la il·luminació en el Fragment Shader, no te solució perquè no pot accedir a la informació requerida.
c) Si ha implementat el càlcul de la il·luminació en el Vèrtex Shader, ha de multiplicar la posició del focus per la View Matrix (view).
d) Cal que normalitzi els vectors L i N, és evident que no estan normalitzats.
60. (2015-2016 2Q) afirmació és correcta respecte als models d'il·luminació:
a) La posició de l'observador afecta al resultat en el model de Lambert mentre que no afecta en el de Phong.
b) La posició de l'observador afecta al resultat en el model de Phong mentre que no afecta en el de Lambert.
c) La posició de l'observador afecta al resultat en tots dos models: el de Phong i el de Lambert.
d) El model d'il·luminació de Phong només es percep si es calcula en el Fragment Shader.
61. (2015-2016 2Q) Una escena està formada per un cub de color vermell molt brillant centrat a l'origen amb cares paral·leles als plans de coordenades i longitud d'aresta 2. S'il·lumina amb un focus de llum blanca situat en el (10,0,0) i observador està en (5,0,0). El càlcul de la il·luminació es realitza correctament en el Fragment Shader utilitzant model de Phong. Si l'observador es mou en direcció cap al centre del cub (sense arribar a tocar la cara del cub):
a) La cara en X=1, s'anirà enfosquit.
b) la cara en X=1, s'anirà enfosquit però es continuará veient la taca espectral blanca al mig.
c) La cara en X=1, no es modificarà de color.
d) La cara en X=1, s'anirà veient cada cop amb un vermell més intents.
62. (2016-2017 1Q) Tenim una implementació correcta del càlcul de la il·luminació en el Fragment Shader i volem pintar un model d'una esfera de manera que es vegi l'esfera amb un degradat de color verd i amb una taca espectral de color groc. Suposant que la

posició del focus de llum i la de l'observador permeten veure l'esfera il·luminada pel focus i la taca espectral, i que el focus emet llum blanca, indica quins paràmetres del material de l'esfera farien possible aquesta visió que es vol aconseguir:

- a) $K_a = (0.2, 0.2, 0.0)$; $K_d = (0.8, 0.8, 0.0)$; $K_s = (1.0, 1.0, 1.0)$; $N = 100$;
- b) $K_a = (0.0, 0.2, 0.0)$; $K_d = (0.0, 0.8, 0.0)$; $K_s = (1.0, 1.0, 0.0)$; $N = 70$;**
- c) $K_a = (0.0, 0.2, 0.0)$; $K_d = (0.0, 0.8, 0.0)$; $K_s = (0.0, 1.0, 0.0)$; $N = 70$;
- d) Cap de les altres combinacions permet la visualització desitjada.

63. (2016-2017 1Q) Volem fer el càlcul de la il·luminació en el Vertex Shader amb un focus donat en coordenades de càmera (uniform amb nom focus). Quines són les instruccions que cal tenir en el Vertex Shader per a calcular les posicions del focus, observador i vertex en SCO (Sistema de Coordenades d'Observador)? Observació: suposem que tenim correctament definits els uniforms VM, TG, OBS i l'atribut (vertex) i que estan inicialitzats en SCA.

- a) $\text{Vertex_SCO} = \text{VM} * \text{TG} * \text{vec4}(\text{vertex}, 1)$; $\text{Focus_SCO} = \text{VM} * \text{vec4}(\text{focus}, 1)$; $\text{OBS_SCO} = \text{VM} * \text{vec4}(\text{OBS}, 1)$;
- b) $\text{Vertex_SCO} = \text{VM} * \text{TG} * \text{vec4}(\text{vertex}, 1)$; $\text{Focus_SCO} = \text{VM} * \text{TG} * \text{vec4}(\text{focus}, 1)$; $\text{OBS_SCO} = \text{vec4}(\text{OBS}, 1)$;
- c) $\text{Vertex_SCO} = \text{VM} * \text{TG} * \text{vec4}(\text{vertex}, 1)$; $\text{Focus_SCO} = \text{VM} * \text{vec4}(\text{focus}, 1)$; $\text{OBS_SCO} = (0, 0, 0, 1)$;
- d) $\text{Vertex_SCO} = \text{VM} * \text{TG} * \text{vec4}(\text{vertex}, 1)$; $\text{Focus_SCO} = \text{vec4}(\text{focus}, 1)$; $\text{OBS_SCO} = (0, 0, 0, 1)$;**

64. (2016-2017 1Q) Quina és la diferència entre fer el càlcul de la il·luminació en el Vertex Shader o fer-la en el Fragment Shader?

- a) Si fem el càlcul en el Fragment Shader aconseguim el model d'il·luminació de Phong, sinó no.
- b) Si fem el càlcul en el Vertex Shader aconseguim els resultats més realistes i és més eficient.
- c) Si fem el càlcul en el Fragment Shader aconseguim més realisme però a un cost d'eficiència més elevat.**
- d) Si fem el càlcul en el Vertex Shader aconseguim fer l'acoloriment (Shading) de Phong, sinó no.

65. (2017-2018P 2Q) Tenim una escena formada per dos cubs de costat 2, el cub1 es troba centrat en el punt $(0,0,0)$ i és de material magenta mat, i el cub2 es troba centrat en el punt $(0,0,3)$ i és de material magenta brillant. Els materials dels dos cubs estan inicialitzats com correspon i tots dos materials tenen la mateixa constant de reflexió difusa K_d . Suposant que tenim un focus de llum en el punt $(0,0,6)$ de color blanc, que l'observador es troba al punt $(0,2,6)$ i que la il_luminació s'ha calculat en el Fragment Shader, indica quina de les següents afirmacions és FALSA.

- a) La cara del cub1 en $Z=1$ es veurà il·luminada pel focus i sense cap taca espectral.
- b) En el cub2 l'observador veurà una taca espectral blanca al mig de l'aresta superior de la cara $Z=4$.
- c) El terme de reflexió espectral que calcula el model d'il·luminació de Phong només caldria aplicar-lo al cub2.
- d) L'observador no veurà taca espectral en cap cara dels dos cubs.

66. (2017-2018 2Q). Respecte als algoritmes d'eliminació de parts amagades, quina de les següents afirmacions és FALSA ?

- A.Podem usar indistintament només backface culling o només z-buffer perquè obtindrem el mateix resultat amb tots dos.
- B. L'algoritme de z-buffer necessita informació de la profunditat de cada fragment per a funcionar.
- C. L'algoritme de z-buffer s'executa en espai imatge, per a cada fragment.
- D. Si tots els objectes són tancats l'algorisme del backface culling no té cap efecte visible en la visualització de l'escena.
67. (2017-2018 2Q) Un estudiant ha fet el següent tros de codi per a calcular la il·luminació en el Vertex Shader. Tenint en compte que la funció Phong(...) és la que hem usat al laboratori, indica quina de les afirmacions següents és correcta.
- ```

1 mat3 normalMatrix = inverse(transpose(mat3 (view * TG)));
2 vec3 normalSCO = normalize(normalMatrix * normal);
3 vec4 vertexSCO = view * TG * vec4(vertex, 1);
4 vec3 posFocus = vec3(0,0,0);
5 vec3 L = normalize(posFocus - vertexSCO.xyz);
6 fcolor = Phong(normalSCO, L, vertexSCO);

```
- A. La crida a Phong de la línia 6 està malament perquè la variable vertexSCO no està normalitzada.
- B. La posició del focus de la variable posFocus és una posició respecte a l'escena.
- C. La multiplicació de les matrius en la línia 3 està malament, hauria de ser  $TG * view * vec4(vertex, 1)$
- D. La posició del focus de la variable posFocus és una posició respecte a la càmera.**
68. Tenim una escena amb un cub de costat 2 centrat a l'origen, orientat amb els eixos i de material verd mat ( $K_s=(0,0,0)$ ). L'observador es troba a la posició  $(0, 0, 2)$  mirant cap a l'origen i tenim un focus de llum blanca a la posició  $(0, 2, 2)$ . Suposant que calculem la il·luminació en el Fragment Shader, quina de les següents afirmacions serà certa respecte a la cara del cub que veu l'observador?
- A. Si usem el model d'il\_luminació de Lambert no podrem veure diferències entre els colors dels diferents punts de la cara.
- B. Si usem el model d'il\_luminació de Phong veurem una taca especial blanca al mig de l'aresta superior de la cara.
- C. Si usem el model d'il\_luminació de Phong no podem saber com es veurà perquè no coneixem el valor del shininess.
- D. El càlcul de la il\_luminació en la cara donarà el mateix resultat tant si usem el model de Phong com si usem el model de Lambert.**
69. (2018-2019 1Q) En OpenGL es poden habilitar dos mètodes d'eliminació de parts amagades: Depth-buffer i back-face culling. Tenim una escena formada per dos cubs d'aresta 2 amb cares paral·leles als plans de coordenades i centres en  $(0,0,0)$  i  $(3,0,0)$ . Un observador inspecciona l'escena amb una càmera en 3ra persona correctament definida.
- A. Les cares visibles que observarà seran les mateixes estiguin activats els dos o només depthbuffer.
- B. Com només hi ha dos objectes convexos i la càmera és en 3ra persona, podem no activar cap mètode i és veurà bé.
- C. Les cares visibles que observarà seran les mateixes estiguin activats els dos o només back-face culling.
- D. No poden estar activats els dos mètodes a la vegada.
70. (2018-2019 1Q) Si una escena la il\_luminem amb un focus de càmera vermell i calculem correctament la il·luminació en el Vertex Shader, les cares visibles per l'observador que són il\_luminades pel focus es veuran...

- A. totes vermelles únicament si utilitzem el model d'il\_luminació de Phong.  
 B. vermelles les cares amb  $K_s=(x,y,z)$  i  $x>0$ , independentment de la resta de constants empíriques.  
 C. vermelles les cares amb  $K_d=(x,y,z)$  i  $x>0$ , independentment de la resta de constants empíriques.  
 D. vermelles les cares de material mat (no brillant), en les cares brillants/especulars veurem una taca blanca.
71. (2018-2019 1Q) Una escena està formada per un cub vermell brillant d'aresta 2 amb centre a l'origen de coordenades i cares paral·leles als plans de coordenades. S'il·lumina amb focus blanc. El focus i l'observador estan ubicats en la posició (2, 0, 0) i VRP està en (0,0,0). Si es calcula la il·luminació en el Fragment Shader (FS) utilitzant el model de Lambert, quina diferència s'observarà en la visualització de l'escena respecte a si la il·luminació es calcula, usant també Lambert, en el Vertex Shader (VS)?  
 A. No ho podem predir, un focus d'escena no pot estar ubicat en la posició de l'observador.  
 B. Cap diferència perquè estem utilitzant Lambert que només té en compte la reflexió difusa.  
 C. Amb el càlcul al FS veurem una taca specular blanca en el mig de la cara i amb el càlcul al VS no.  
**D. Amb el càlcul al VS la cara es veurà de color constant/uniforme vermell, amb el càlcul al FS veurem una gradació de vermells.**
72. (2018-2019 2Q) Respecte als algoritmes d'eliminació de parts amagades, quina de les següents afirmacions és FALSA ?  
 A. Podem usar indistintament només backface culling o només z-buffer perquè obtindrem el mateix resultat amb tots dos.  
 B. L'algoritme de z-buffer necessita informació de la profunditat de cada fragment per a funcionar.  
 C. L'algoritme de z-buffer s'executa en espai imatge, per a cada fragment.  
 D. Si tots els objectes són tancats l'algorisme del backface culling no té cap efecte visible en la visualització de l'escena.
73. (2018-2019 2Q) Un estudiant ha fet el següent tros de codi per a calcular la il\_luminació en el Vertex Shader. Tenint en compte que la funció Phong(...) és la que hem usat al laboratori, indica quina de les afirmacions següents és correcta.
- ```

1 mat3 normalMatrix = inverse(transpose(mat3 (view * TG)));
2 vec3 normalSCO = normalize(normalMatrix * normal);
3 vec4 vertexSCO = view * TG * vec4(vertex, 1);
4 vec3 posFocus = vec3(0,0,0);
5 vec3 L = normalize(posFocus - vertexSCO.xyz);
6 fcolor = Phong(normalSCO, L, vertexSCO);

```
- A. La crida a Phong de la línia 6 està malament perquè la variable vertexSCO no està normalitzada.
 B. La posició del focus de la variable posFocus és una posició respecte a l'escena.
 C. La multiplicació de les matrius en la línia 3 està malament, hauria de ser: $TG * view * vec4(vertex, 1)$
 D. La posició del focus de la variable posFocus és una posició respecte a la càmera.
74. (2018-2019 2Q) Tenim una escena amb un cub de costat 2 centrat a l'origen, orientat amb els eixos i de material verd mat ($K_s=(0,0,0)$). L'observador es troba a la posició (0, 0, 2) mirant cap a l'origen i tenim un focus de llum blanca a la posició (0, 2, 2). Suposant que calculem la il_luminació en el Fragment Shader, quina de les següents afirmacions serà certa respecte a la cara del cub que veu l'observador?
 A. Si usem el model d'il_luminació de Lambert no podrem veure diferències entre els colors dels diferents punts de la cara.

- B. Si usem el model d'il_luminació de Phong veurem una taca espectral blanca al mig de l'aresta superior de la cara.
- C. Si usem el model d'il_luminació de Phong no podem saber com es veurà perquè no coneixem el valor del shininess.
- D. El càlcul de la il_luminació en la cara donarà el mateix resultat tant si usem el model de Phong com si usem el model de Lambert.

Solucions

- Com que l'esfera es veu groga quan s'il·lumina amb llum blanca, les constants de reflexió difusa del material han de ser del tipus $(a,a,0)$. Suposem que els colors ambient $(0.2,0.2,0)$, difós $(0.5,0.5,0)$. La constant specular pot ser $(1,1,1)$ i $N=50$, donat que és una esfera metàl·lica i, per tant, specular. Com la llum ambient és blanca, la contribució ambient és groga i tant la difosa com l'especular donen un color verd ja que és el producte de la intensitat del focus $(0., 1., 0.)$ per les constants de material. El que veiem es:
 - La zona inferior, a l'ombra, de color groc fosc
 - La zona superior, afectada pel terme de Lambert, de color verd amb un màxim a la direcció en que el vector normal a l'esfera es dirigeix cap al focus de llum
 - A més, veurem el reflex specular (terme de Phong) verd molt brillant en el punt que la reflexió specular tingui la direcció de visió del punt.

33.

- (a) Donat que el polígon és mate, no ha de reflectir llum specular: $ks=(0,0,0)$; com ha de ser de color gris, kd i ka han de ser $r=g=b$; per exemple: $kd=(0.8,0.8,0.8)$ i $ka=(0.2,0.2,0.2)$ per a donar menys pes a la reflexió ambient.
- (b) Donada la posició dels focus i les distàncies entre vèrtexs, cada vèrtex està il·luminat, pràcticament, només pel focus que té a sobre (la contribució dels altres serà practicament nul·la degut a l'angle entre la direcció de la llum incident i la normal en el vèrtex). A més, l'angle entre la direcció d'il·luminació i la normal de la cara en el vèrtex és de 0° ($\cos 0^\circ=1$). Cada vèrtex quedará del color del focus que té a sobre: $I_v(V) = I_f * kd$.
 El shading de Gouraud interpola el color en cada fragment de la rasterització en funció de la seva distància als vèrtexs del polígon. Concretament, l'aresta entre V1 i V2 quedará blava, entre V2 i V3 trobarem colors de blau a vermell passant per magenta, de V3 a V4 de vermell a verd passant per groc i de V4 a V1 de verd a blau passant per cyan; cap al centre del polígon trobarem un color funció de la seva distància als vèrtexs (arestes) i, per tant, hi podem trobar la suma de tots els colors que produirà un gris.
- (c) Les cares dels cubes amb normal segons els eixos Y i Z no reben llum, donat que la normal en els seus vèrtexs forma un angle superior a 90° amb la direcció d'incidència de la llum en ells; es veuran negres. La cares amb normal segons $+X$ ($-X$) es veuran verdes (vermelles) perque l'angle de la normal en els seus vèrtexs i la direcció de la llum del focus verd (vermell) és $<90^\circ$. Noteu que l'angle esmentat és el mateix per tots els vèrtex d'una cara; per tant, encara que es faci colorat de Gouraud, seran d'un color uniforme. A més, la cara verda (vermella) del cub més proper al focus verd (vermell) serà més fosca que la cara de l'altre cub del mateix color perque l'angle entre la normal als seus vèrtexs i la direcció d'incidència de la llum en ells és més gran (cosinus més petit).

47. Donat que el model d'il·luminació és Phong, es calcularà el color en els vèrtexs de les cares tenint en compte la reflexió difusa i l'especular.

$$I(V) = If kd \cos(\text{fit}a) + If ks \cos^n(\text{fi})$$

Fita= angle entre la normal en un vèrtex V i la direcció de la llum incident en ell.

Fi= angle entre el raig reflectit en el vèrtex V i la recta que uneix el vèrtex amb l'observador.

El color als punts interiors de les cares es calcularà interpolant el color dels seus vèrtexs (smooth shading).

El focus està situat sobre la recta que passa just pel mig de les cares $x=10$ i $x=40$; per tant, pels 4 vèrtexs d'una mateixa cara l'angle entre la direcció d'incidència de la llum i la seva normal és el mateix i, per tant, tots els vèrtexs tindran el mateix color difus => color de la cara constant. Podem desacartar la resposta d.

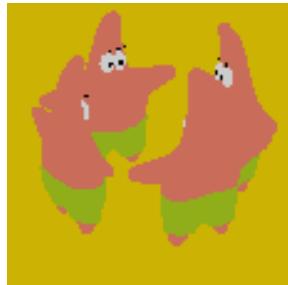
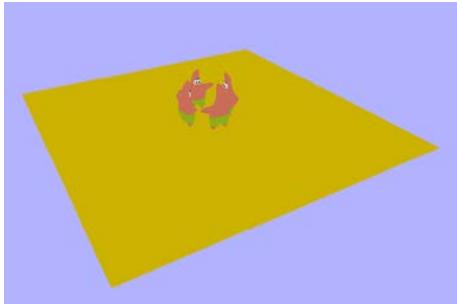


Com les dues cares tenen el mateix color verd, són il·luminades pel mateix focus groc (per exemple, $If=(1,1,0)$); però l'angle fita és més gran pels vèrtexs de la cara $x=40$ que per la cara $x=10 \Rightarrow$ la cara $x=40$ és veurà més fosca ($\cos(\text{fita})$ serà més petit pels seus vèrtexs). Per tant, podem descartar les respostes b i c.

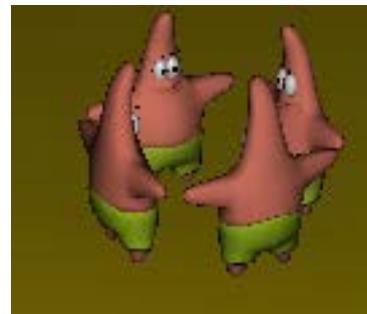
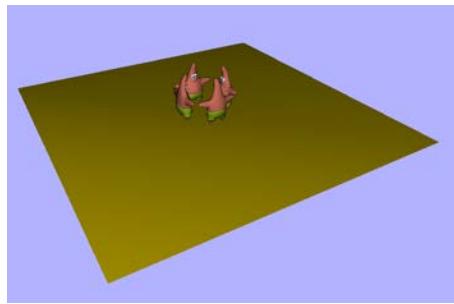
Comprovem que la responsta a és correcta. Noteu que és impossible que l'observador vegi una taca espectral al mig de la cara $x=40$ per molt espectral que sigui, perquè el color es calcula en els vèrtexs. Com a molt podria veure la taca espectral en un vèrtex (i aquesta resposta no hi és).

Classe 6: contingut

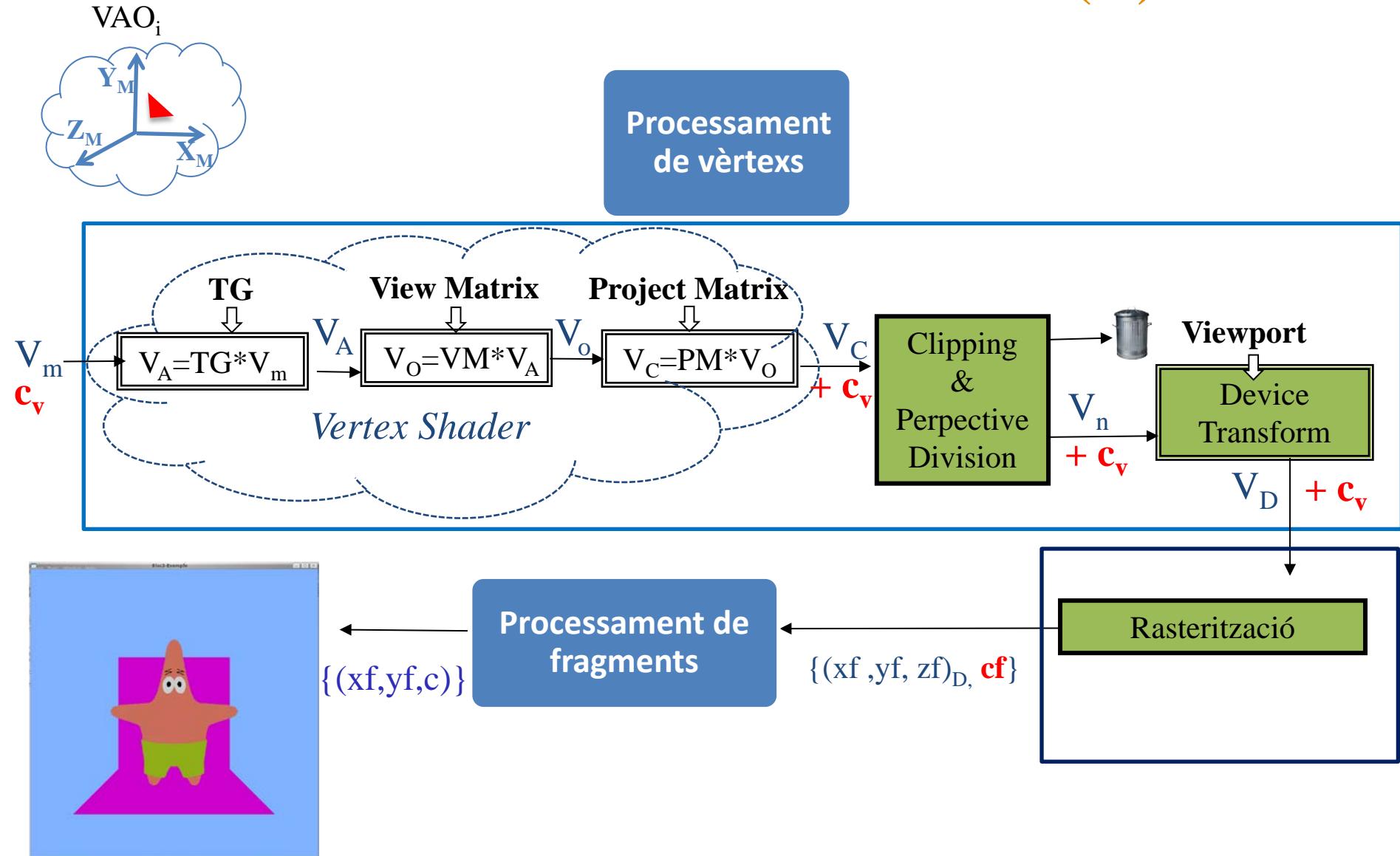
- Realisme: Eliminació de parts ocultes



- Realisme: models d'il·luminació

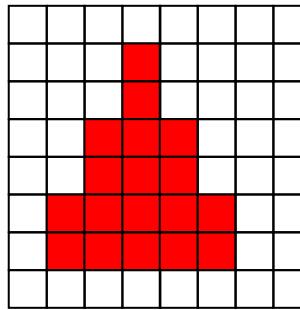
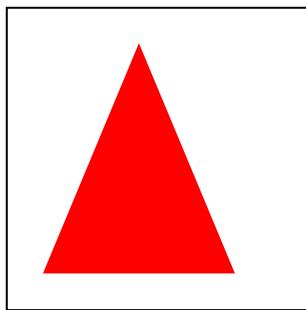
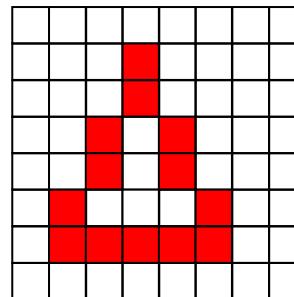
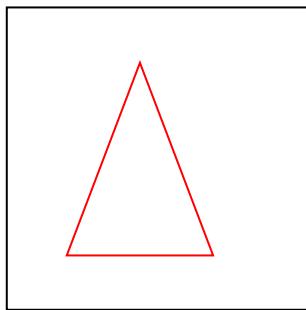
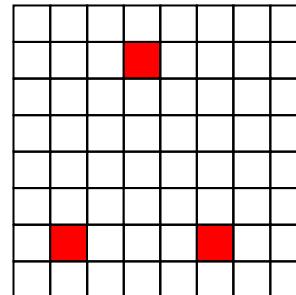
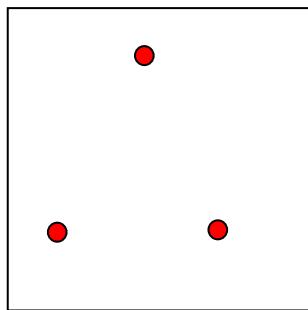


Procés de visualització(1)



Algorismes de rasterització

La discretització és diferent per a cada primitiva: punt, segment, polígon



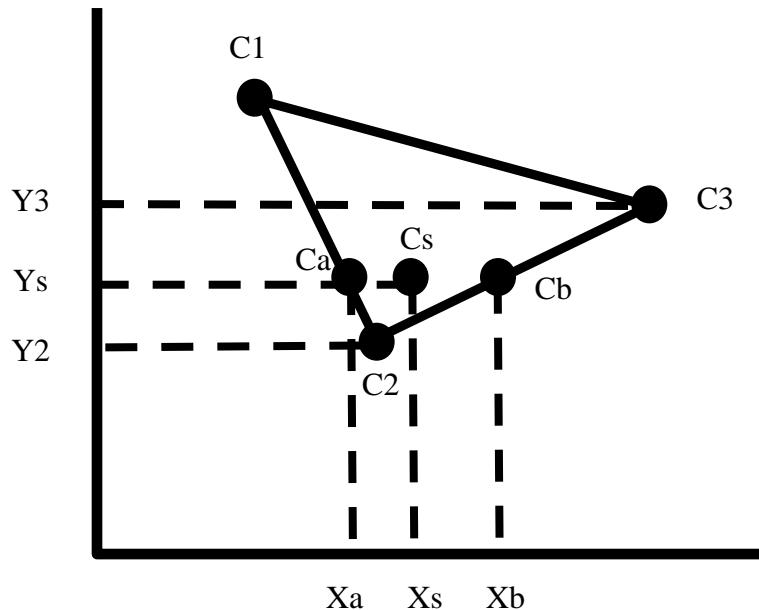
$$\mathbf{V}_D = (x_D, y_D, z_D) + \mathbf{c}_v$$



$$\{(x_f, y_f, z_f, c_f)\}$$

Shading (colorat) de polígons

- Colorat Constant \equiv Flat shading $\rightarrow C_f = C_1$
color uniforme per tot el polígon (funció del color calculat en un vèrtex); cada cara pot tenir diferent color.
- Colorat de Gouraud \equiv Gouraud shading \equiv Smooth shading

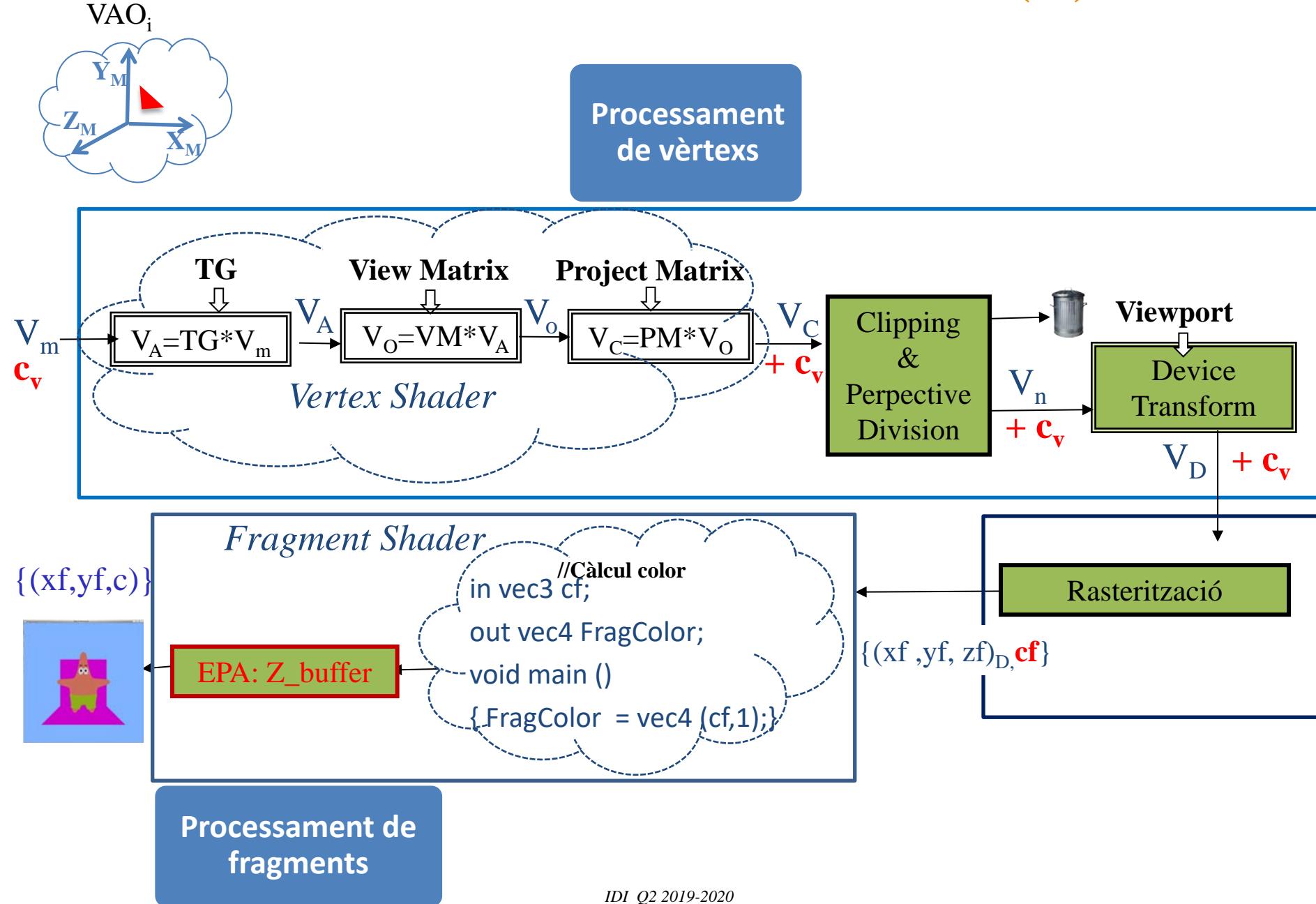


$$Ca = \frac{1}{Y_1 - Y_2} (C_1(Y_s - Y_2) + C_2(Y_1 - Y_s))$$

$$Cb = \frac{1}{Y_3 - Y_2} (C_2(Y_3 - Y_s) + C_3(Y_s - Y_2))$$

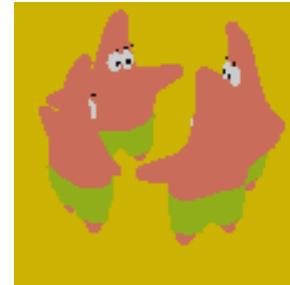
$$Cs = \frac{1}{X_b - X_a} (Ca(X_b - X_s) + Cb(X_s - X_a))$$

Procés de visualització(2)



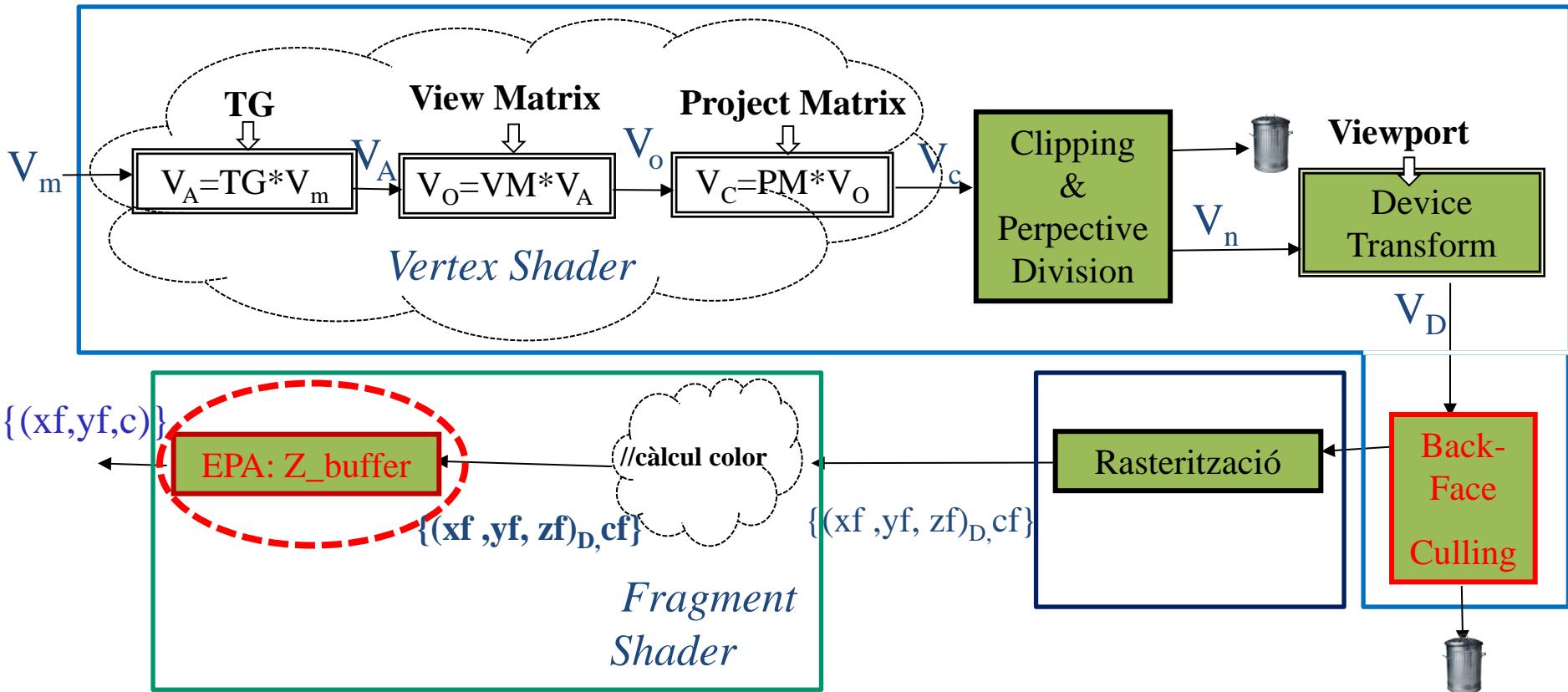
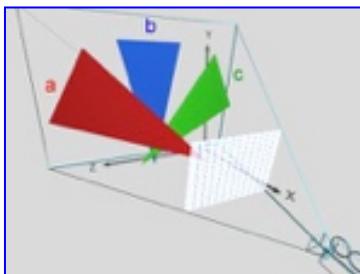
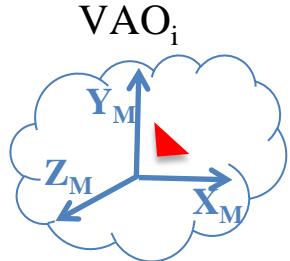
Classe 6: contingut

- **Realisme: Eliminació de parts ocultes**
 - Depth-buffer
 - Back-face culling
- Realisme: Il·luminació (1)
 - Càcul del color en un punt
 - Models d'il·luminació empírics



Bibliografia: capítol EPA&Il·luminació del llibre multimèdia

Procés de visualització: EPA (1)



Depth Buffer

- Mètode EPA en espai imatge (*a nivell de píxel/fragment*)
- Després de la **rasterització i del Fragment Shader**
- Requereix conèixer per a cada píxel, un valor (depth) que sigui proporcional a la distància a l'observador a la que es troba el polígon que es projecta en el píxel.
- No importa ordre en que s'enviïn a pintar els triangles (ordre en què estiguin en VBO)
- No requereix tenir el Back-face culling activat

Depth Buffer (z-buffer)

Dos buffers de la mateixa resolució que la pantalla

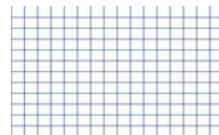
Buffer color (frame_buffer)

$$(r, g, b) \in [0, 2^n - 1]$$

Buffer profunditats (depth_buffer)

$$z \in [0, 2^{nz} - 1]$$

1. Inicialitzar al color de fons

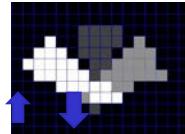


1. Inicialitzar al més lluny possible



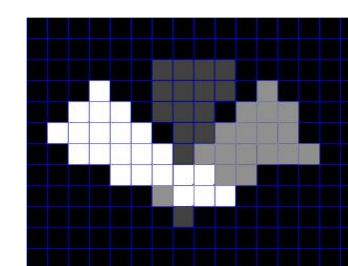
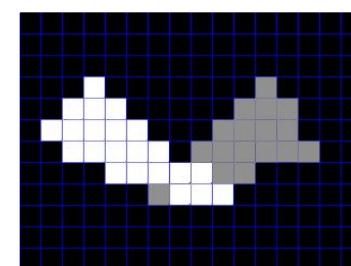
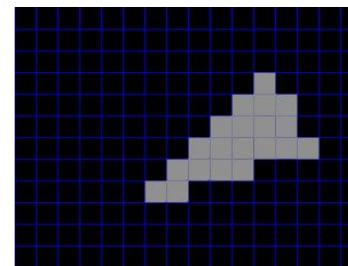
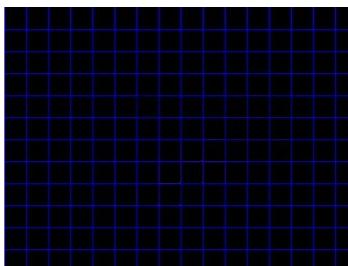
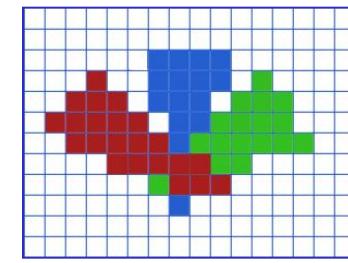
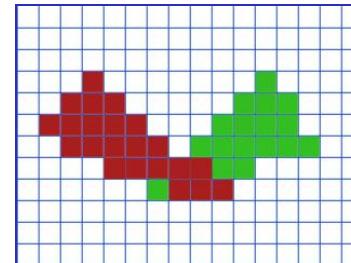
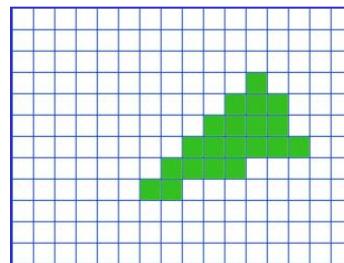
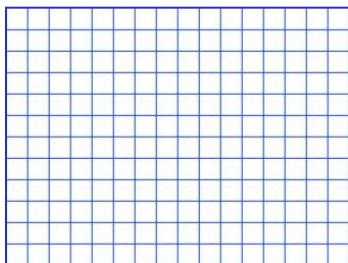
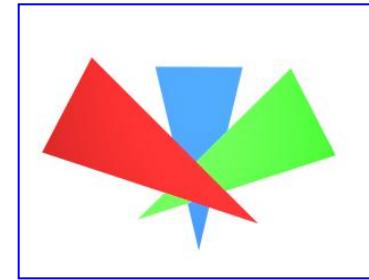
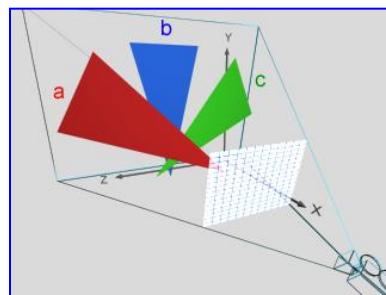
```
glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT)
```

2. Per a cada fragment

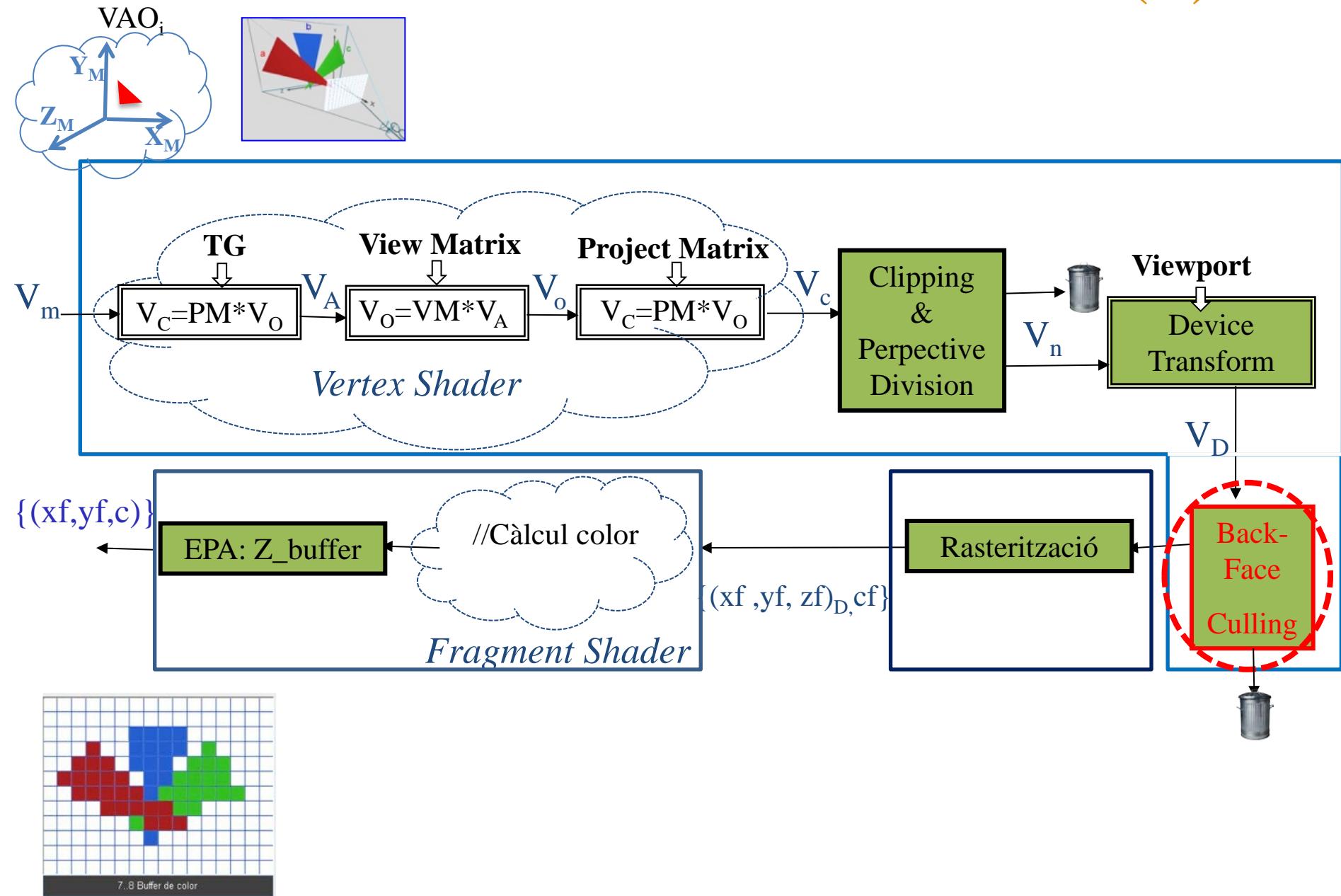


```
if (zf < depth_buffer[xf, yf]) {  
    depth_buffer [xf, yf] = zf;  
    color_buffer [xf, yf] = cf;  
}
```

Depth Buffer (z-buffer)

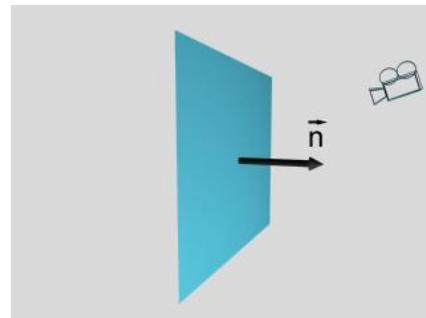


Procés de visualització: EPA (2)

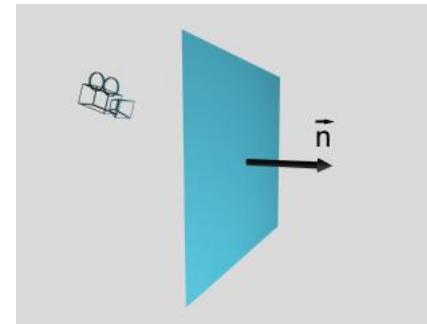


Back-face Culling

- Mètode EPA en espai *objecte* (*a nivell de triangle*)
- Requereix cares orientades, opaques, objectes tancats
- Considera escena formada només per la *cara* i l'*observador*
- És conservatiu (determina les cares que “segur” no són visibles)



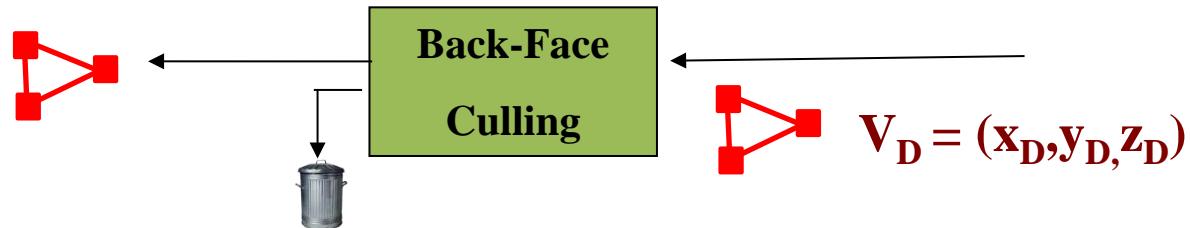
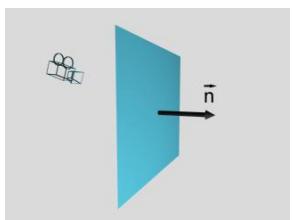
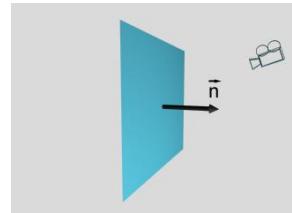
visible



no visible

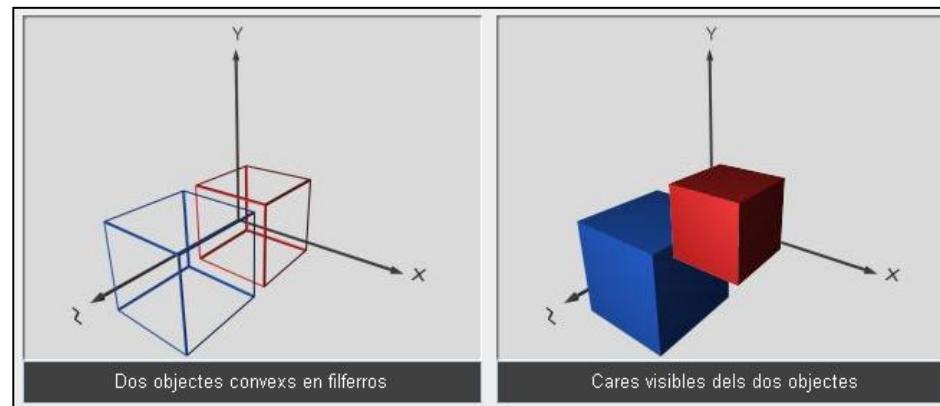
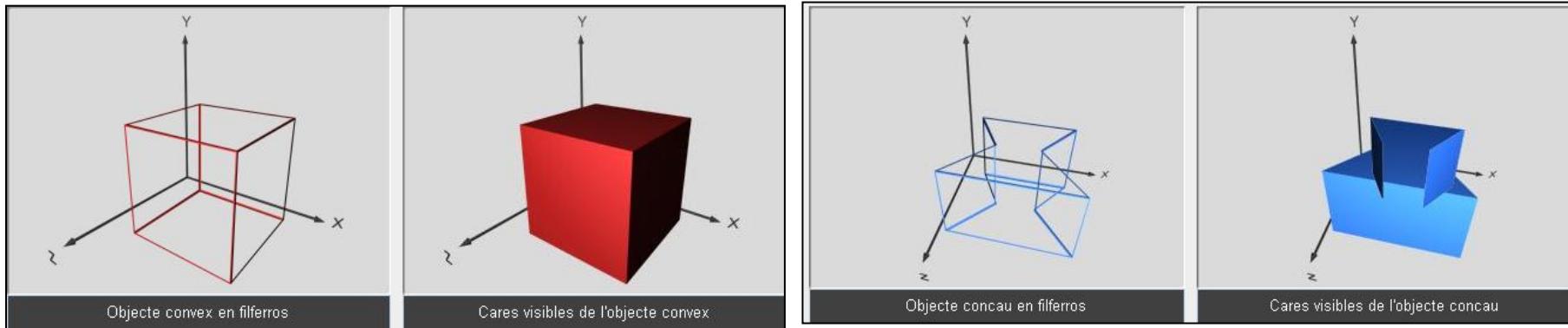
Back-face Culling

- OpenGL fa el càlcul en coord. dispositiu
 - direcció de visió $(0,0,-1)$
 - visibles les cares amb $n_z > 0$ (ordenació vèrtexs antihorari)
 - el càlcul de la normal de la cara el fa OpenGL a partir dels vèrtexs en coordenades de dispositiu => **importància ordenació vèrtexs.**



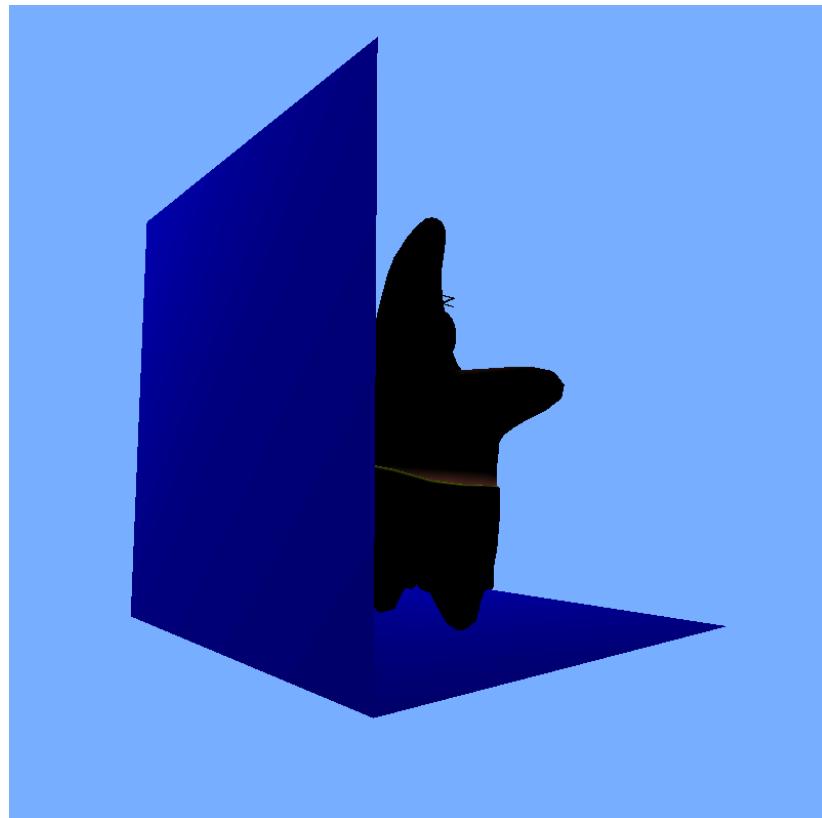
Back-face Culling

- Culling com EPA només si l'escena conté un únic objecte convex.

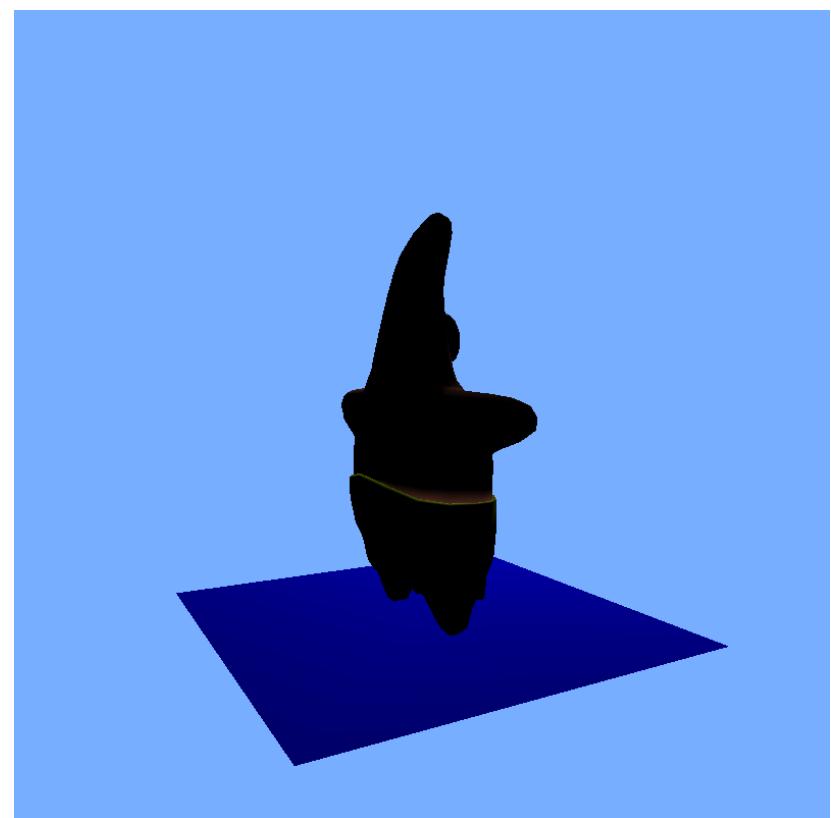


Exemple que podreu comprovar al laboratori

Sense back-face culling



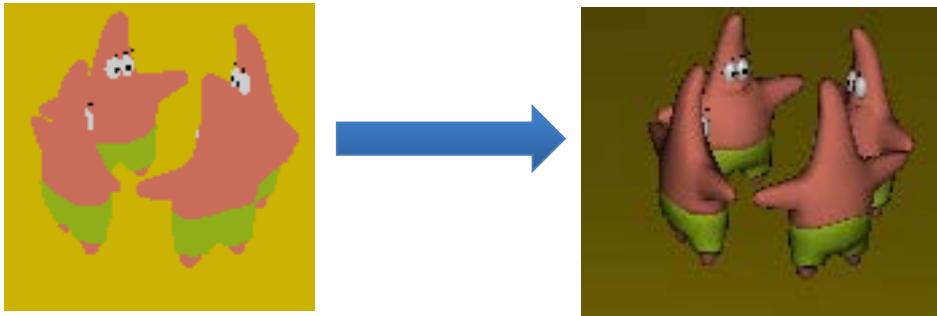
Amb back-face culling



`glEnable(GL_CULL_FACE);`

Classe 6: contingut

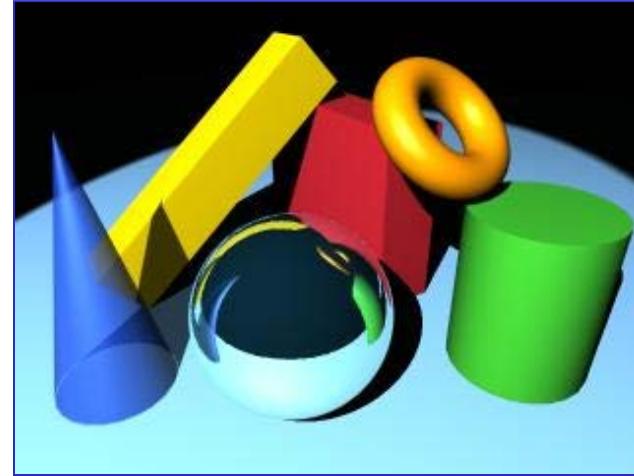
- Realisme: Eliminació de parts ocultes
 - Depth-buffer
 - Back-face culling
- **Realisme: Il·luminació (1)**
 - **Càcul del color en un punt**
 - Models d'il·luminació empírics



Bibliografia: capítol EPA&Il·luminació del llibre multimèdia

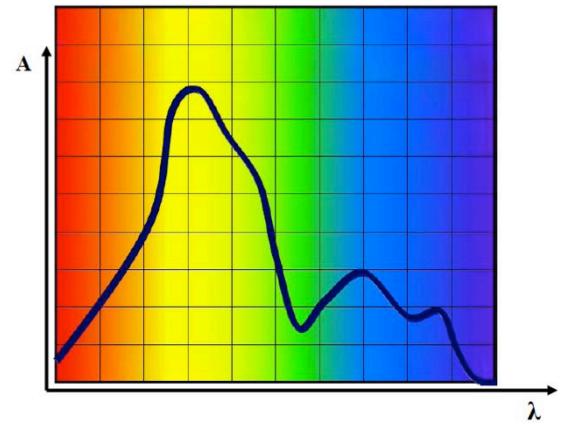
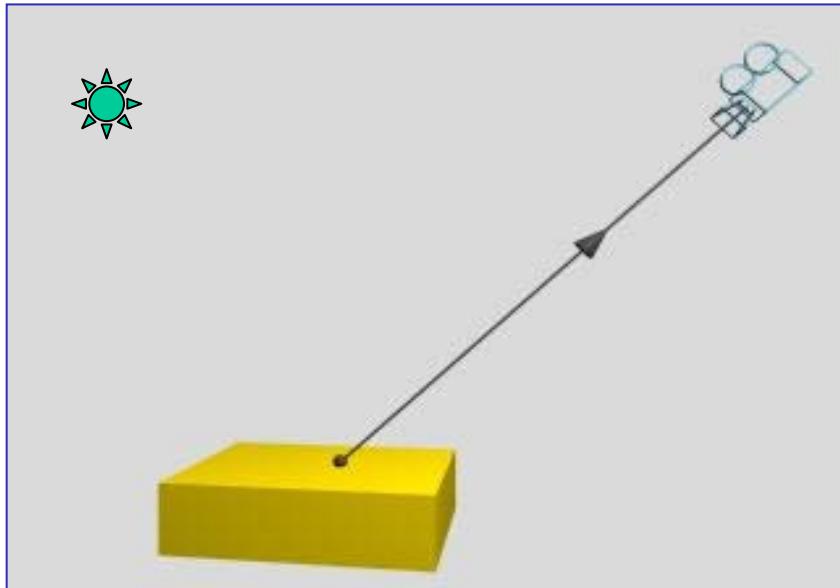
Introducció

- Els models d'il·luminació simulen el comportament de la llum per determinar el color d'un punt de l'escena.
- Permeten obtenir imatges molt més realistes que pintant cada objecte d'un color uniforme:



Color d'un punt

El color amb el que un Observador veu un punt P de l'escena és el color de la llum que arriba a l'Obs procedent de P: $I_\lambda(P \rightarrow Obs)$

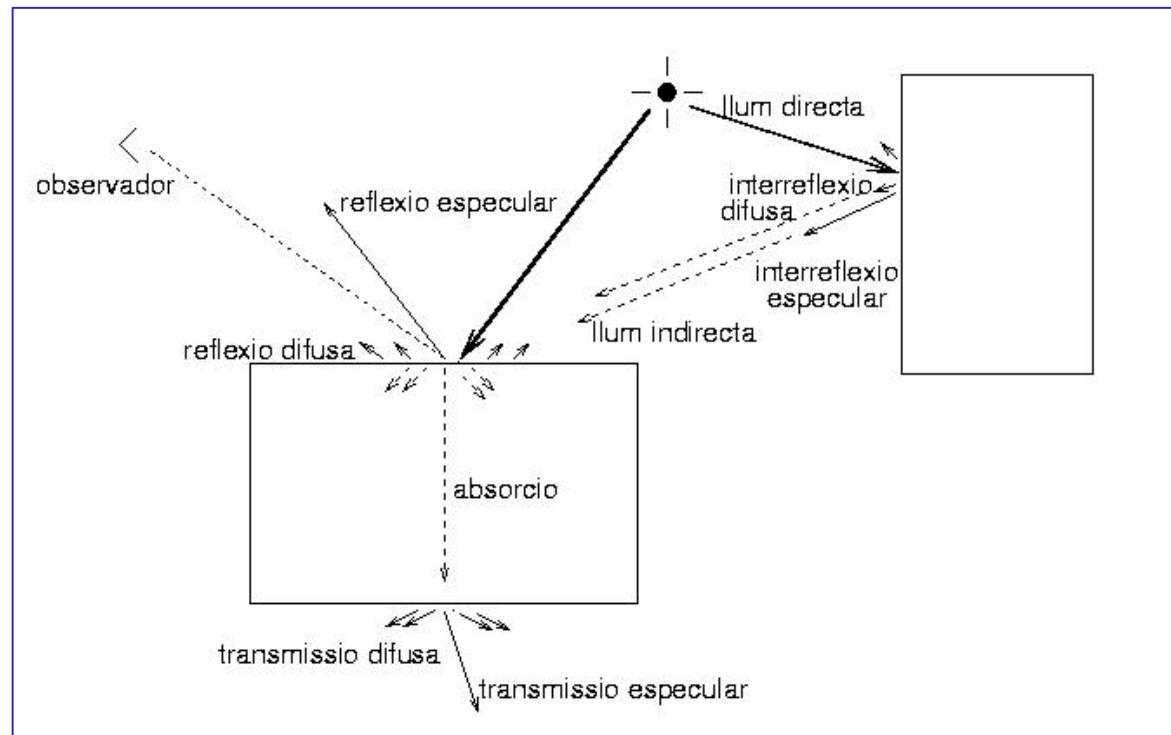


$$I_\lambda(P \rightarrow Obs) \quad \lambda \in \{r,g,b\}$$

Elements que intervenen

El color que arriba a l'Obs procedent de P, $I_\lambda(P \rightarrow Obs)$, funció de:

- Fonts de llum
- Materials
- Altres objectes
- Posició de l'observador
- Medi pel que es propaga



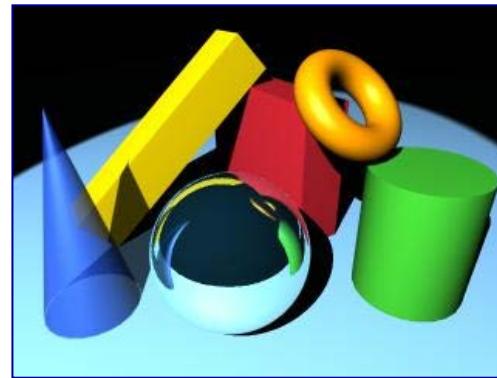
Models d'il·luminació

- Els models d'il·luminació simulen les lleis físiques que determinen el color d'un punt.
- El càlcul exacte és computacionalment inviable.
- Una primera simplificació és usar només les energies corresponents a les llums vermella, verda i blava.

$$I_\lambda(P \rightarrow Obs) \quad \lambda \in \{r,g,b\}$$

Models d'il·luminació: Classificació

- Models Locals o empírics
- Models Globals: traçat de raig, radiositat



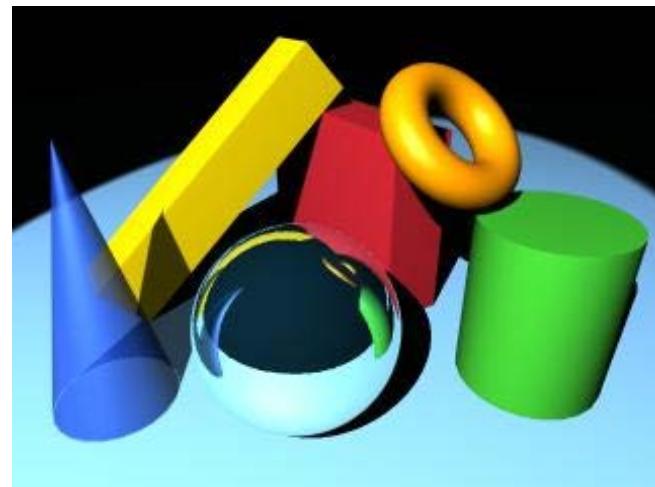
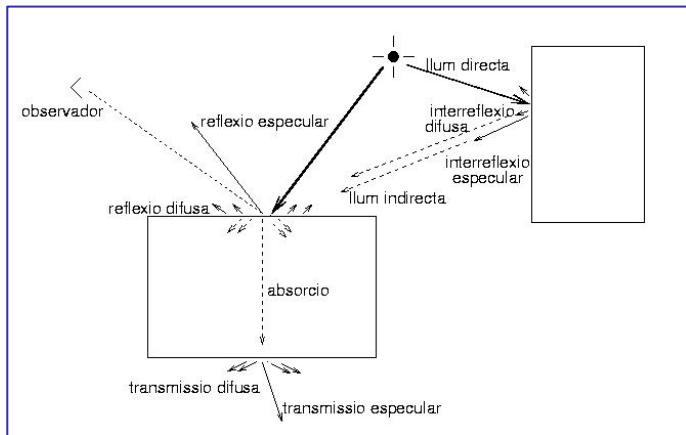
Models locals o empírics

- Només consideren per al càlcul del color: el punt **P** en què es calcula, els focus de llum (sempre puntuals) i la posició de l'observador.
- No consideren altres objectes de l'escena (noombres, nomiralls, no transparències).
- Aproximen la transmissió de la llum per fórmules empíriques i les propietats de reflexió dels materials per constants.



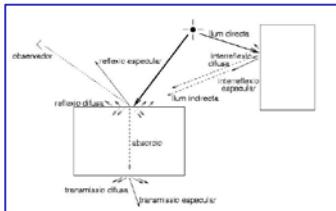
Models de traçat de raig

- Els models d'il·luminació de traçat de raig consideren:
 - Focus de llum puntuals
 - Altres objectes existents en l'escena però només transmissions especulars
- Permeten simular ombres, transparències i miralls.
- Són més costosos en càlcul .



Models de radiositat

- Consideren els focus de llum com un objecte qualsevol de l'escena.
- Els objectes només poden produir **reflexions difuses pures**.
- Com que totes les reflexions són difuses, la radiositat no considera la posició de l'observador.
- Poden **modelar ombres i penombres**, però no miralls ni **transparències**.
- Són els més costosos i es basen en l'anàlisi de l'intercanvi d'energia entre tots els objectes de l'escena.



Classe 6: contingut

- Realisme: Eliminació de parts ocultes
 - Depth-buffer
 - Back-face culling
- **Realisme: Il·luminació (1)**
 - Càcul del color en un punt
 - **Models d'il·luminació empírics**

Model empíric ambient

- No es consideren els focus de llum de l'escena.
- La llum ambient és deguda a reflexions difuses de llum entre objectes, per tant es considera que no prové de cap focus específic i no té cap direcció concreta.
- Tots els punts de l'escena reben la mateixa aportació de llum.
- S'observarà el mateix color en tots els punts d'un mateix objecte.
- Equació: $I_\lambda(P) = I_{a\lambda} k_{a\lambda}$
 - $I_{a\lambda}$: color de la llum ambient
 - $k_{a\lambda}$: coef. de reflexió ambient



Model empíric ambient

- Equació: $I_\lambda(P) = I_{a\lambda} k_{a\lambda}$

Exemple amb una esfera amb:

$$I_a = (1, 1, 1)$$

K_a també blanca amb intensitat variant entre 0.2, 0.4, 0.6, 0.8 i 1

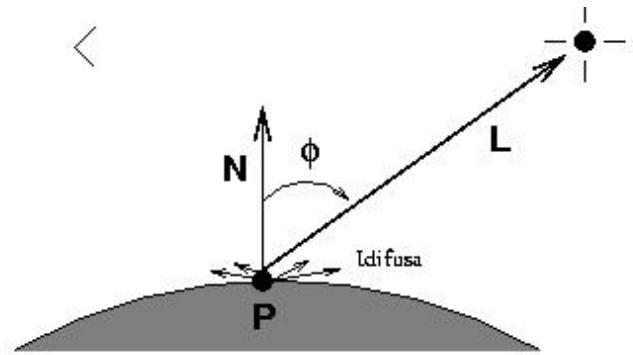


Model empíric difús (Lambert)

- Focus puntuals. Objectes només tenen reflexió difusa pura.
- Podem imaginar que el punt **P** irradia la mateixa llum en totes direccions i per tant el seu color no depèn de la direcció de visió.

$$I_{\lambda}(P) = I_{f\lambda} k_{d\lambda} \cos(\Phi) = I_{f\lambda} k_{d\lambda} \text{dot}(N, L)$$

si $|\Phi| < 90^\circ$



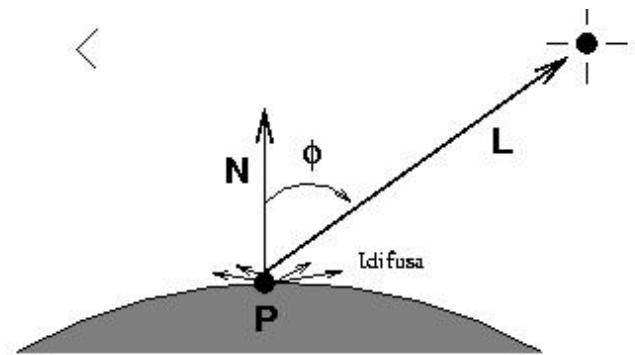
- $I_{f\lambda}$: color (r,g,b) de la llum del focus puntual f
 - $k_{d\lambda}$: coef. de reflexió difusa del material
 - $\cos(\Phi)$: cosinus de l'angle entre la llum incident i la normal a la superfície en el punt **P**.
- Pot calcular-se com el producte escalar entre **N** i **L** si estan normalitzats.*



Model empíric difús (Lambert)

$$I_\lambda(P) = I_{f\lambda} k_{d\lambda} \cos(\Phi) = I_{f\lambda} k_{d\lambda} \text{dot}(N, L)$$

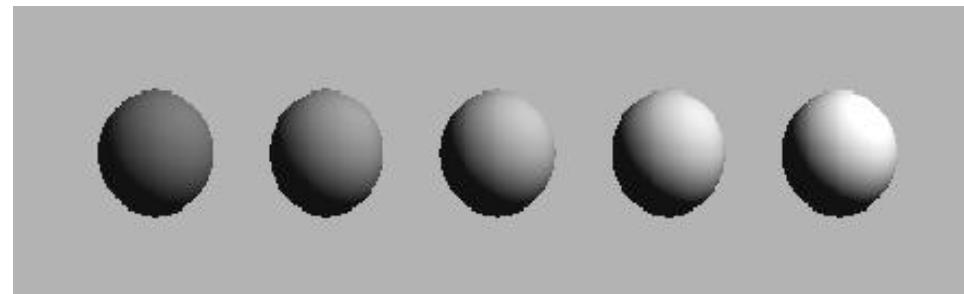
si $|\Phi| < 90^\circ$



Exemple amb una esfera amb:

$$I_f = (1, 1, 1)$$

K_d també blanca amb intensitat variant entre 0.4, 0.55, 0.7, 0.85 i 1



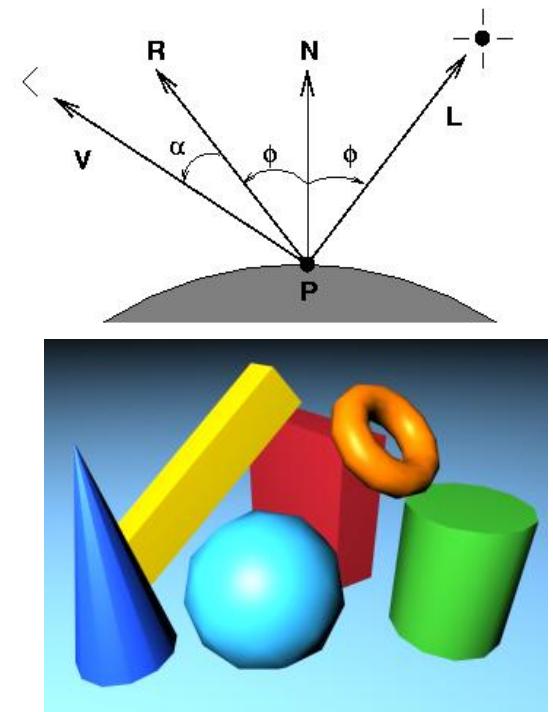
Model empíric especular (Phong)

- Focus de llum puntuals i objectes només reflexió especular.
- L'observador només podrà observar la reflexió especular en un punt si es troba en la direcció de la reflexió especular.
- La direcció d'especularitat és la simètrica de \mathbf{L} respecte \mathbf{N} i es pot calcular com: $\mathbf{R} = 2\mathbf{N}(\mathbf{N}^*\mathbf{L}) - \mathbf{L}$ si tots els vectors són normalitzats.

$$I_\lambda(P) = I_{f\lambda} k_{s\lambda} \cos^n(\alpha) = I_{f\lambda} k_{s\lambda} \operatorname{dot}(R, v)^n$$

si $|\Phi| < 90^\circ$

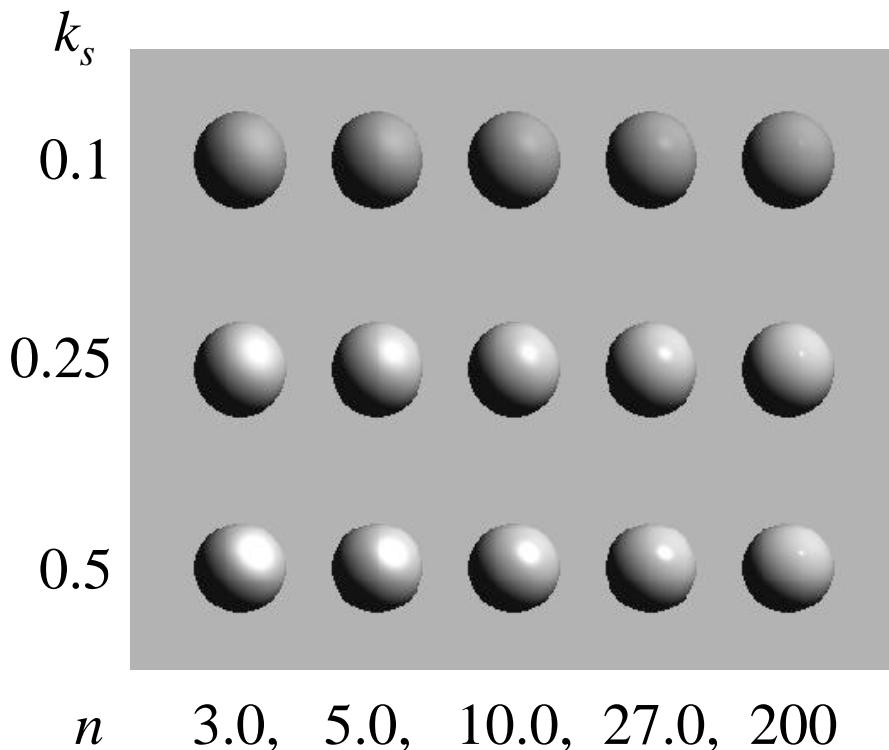
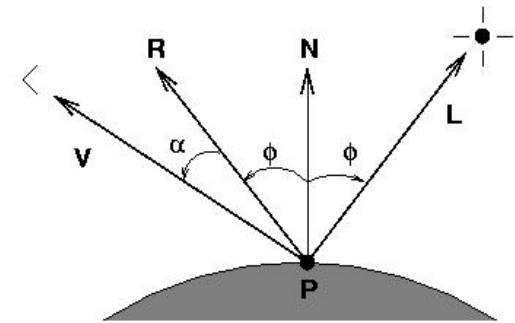
- $I_{f\lambda}$: color (r,g,b) del focus puntual f
- $k_{s\lambda}$: coef. de reflexió especular (x,x,x)
- n : exponent de reflexió especular
- v és vector normalitzat que uneix punt amb Obs



Model empíric especular (Phong)

$$I_\lambda(P) = I_{f\lambda} k_{s\lambda} \cos^n(\alpha) = I_{f\lambda} k_{s\lambda} \operatorname{dot}(R, v)^n$$

si $|\Phi| < 90^\circ$



Exemple d'una esfera amb:

$$I_f = (1, 1, 1)$$

k_d blanca amb intensitat 0.5

k_s blanca amb 0.1, 0.25 i 0.5

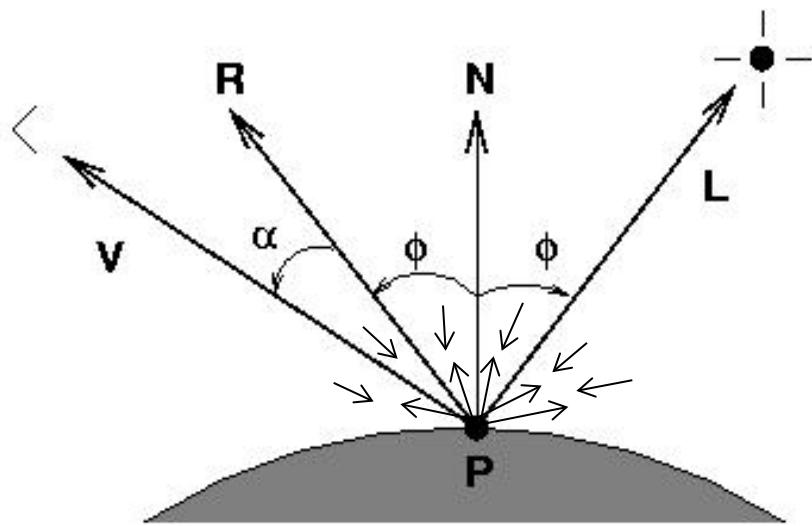
$n : 3.0, 5.0, 10.0, 27.0, 200$

Resum

Color d'un punt degut a...	Depèn de la normal?	Depèn de l'observador?	Exemple
Model ambient	No	No	
Model difús	Sí	No	
Model especular	Sí	Sí	

$$I_{\lambda}(P) = I_{a\lambda}k_{a\lambda} + \sum_i (I_{f_i\lambda} k_{d\lambda} \cos(\Phi_i)) + \sum_i (I_{f_i\lambda} k_{s\lambda} \cos^n(\alpha_i))$$

$$I_\lambda(P) = I_{a\lambda} k_{a\lambda} + \sum_i (I_{f_i\lambda} k_{d\lambda} \cos(\Phi_i)) + \sum_i (I_{f_i\lambda} k_{s\lambda} \cos^n(\alpha_i))$$



Conceptes i preguntes

- Determinació de visibilitat, eliminació de cares ocultes, algorismes en espai imatge i en espai objecte.
- Shading de polígons. Com i qui el realitza.
- L'algorisme Back-face culling: perquè és conservatiu? Quines són les seves limitacions?
- L'algorisme de depth-buffer: en quin espai treballa? Perquè no importa l'ordre de processament dels fragments? Requereix el back-face culling?
- Classificació dels models d'il·luminació.
- Models empírics o locals: limitacions
- Model ambient, difús/Lambert, espeular/Phong: què modelen? Quines restriccions?
- Interpretació de les constants empíriques. Què significa que un material sigui mat?
- Què significa el angle “fita” del model de Lambert?
- Què és la taca espeular? Perquè és produïx? Usualment, quin color té?
- Quina diferència de colorejat observarem en una esfera il·luminada per un focus de llum si només reflecteix llum difusa o si reflecteix difusa i espeular?
- El color “base” d'un objecte (color objecte il·luminat per llum blanca), en quina constant empírica queda reflectit?
- Fer els exercicis proposats.

Exercici 1:

Quines constants de material definiries si es vol que un objecte sigui de plàstic polit/brillant de color vermell?
Raona la resposta.

Exercici 2:

Una esfera brillant de metall que es veu groga quan s'il·lumina amb llum blanca, la posem en una habitació que té llum ambient (.5, .5, .5) i un únic focus, de llum verda, situat 2 metres damunt de la càmera (en direcció de l'eix y).

Quines zones distingirem en la visualització de l'esfera i de quins colors seran?

Justifiqueu la resposta en relació a les propietats del material de l'esfera i les llums. Imagineu que es calcula el color en cada punt de l'esfera.

Exercici 3:

Disposem de dos cubs amb les seves cares paral·leles als plans de coordenades, longitud d'aresta igual a 2 i centres als punts $(2,1,2)$ i $(5,1,2)$ respectivament. Els dos cubs són de metall gris i s'il·luminen amb un focus de llum verda situat al punt $(20,1,2)$.

Com és possible que la cara del cub_1 situada en $x=3$ es vegi il·luminada si el cub_2 li fa ombra?

Quines altres cares es veuran il·luminades pel focus?

Exercici 4:

Raona amb quins valors inicialitzaries les constants empíriques del material Kd i Ks d'un objecte que té el següent comportament: els reflexos especulars sempre es veuen del mateix color que la llum del focus i la resta de zones il·luminades pel focus es veuen de color groc si el focus és groc i del mateix color que les zones no il·luminades pel focus quan el focus és de color blau.

Penseu-lo vosaltres...

Exercici 5:

Una escena està formada per tres cubs d'aresta 2, centrats als punts $(-5, 0, 0)$, $(0, 0, 0)$ i $(5, 0, 0)$ i amb cares paral·leles als plans de coordenades. Els cubs són de color magenta mat.

Ubiquem un focus de llum blanca en la posició $(0, 0, 0)$. No hi ha llum ambient. De quin color s'observaran les cares dels cubs ubicades en $x=6$ i $x=-4$?

Observació: la ubicació de la càmera permet veure totes dues cares.

- a) Es veuran negres perquè el focus de llum està dins del cub central en $(0, 0, 0)$
- b) Si es té activat el *back-face culling*, es veuran les dues cares de color magenta, més fosca la de $x=6$ perquè està més lluny del focus
- c) Es veurà la cara en $x=6$ negra i la $x=-4$ de color magenta
- d) Si es té activat el *back-face culling*, es veuran les dues cares de color magenta, més fosca la de $x=-4$

Penseu-lo vosaltres...

Exercici 1:

Quines constants de material definiries si es vol que un objecte sigui de plàstic polit/brillant de color vermell?
Raona la resposta.

IDI Q2 2019-2020

Les constants del material són: K_a , K_d , K_s i n .

- K_a i K_d representen el “color” de l’objecte, per tant si diem que l’objecte és vermell, aquestes dues constants han de reflectir el color vermell i no el blau i el verd. Si no volem colors molt saturats, podem posar com a $K_d = (0.8, 0, 0)$ i com que K_a s’acostuma a posar de menys intensitat (però del mateix color perquè també està relacionada amb reflexió difusa del material) podem posar $K_a = (0.3, 0, 0)$.
- K_s i n són les constants que tenen a veure amb la reflexió especular, per tant, si tenim que l’objecte ha de ser brillant, vol dir que K_s no pot ser nul·la. Com que habitualment aquesta K_s és en forma (x, x, x) , direm que si volem que l’objecte sigui brillant posarem $K_s = (1, 1, 1)$. A més, si volem que la brillantor es vegi amb taques especulars distingibles (si la taca especular és massa gran es difumina i no es distingeix) hem de fer que la taca especular sigui petita, i per tant n ha de ser gran. Per exemple $n = 200$.

Exercici 2:

Una esfera brillant de metall que es veu groga quan s'il·lumina amb llum blanca, la posem en una habitació que té llum ambient (.5, .5, .5) i un únic focus, de llum verda, situat 2 metres damunt de la càmera (en direcció de l'eix y).

Quines zones distingirem en la visualització de l'esfera i de quins colors seran?

Justifiqueu la resposta en relació a les propietats del material de l'esfera i les llums. Imagineu que es calcula el color en cada punt de l'esfera.

IDI Q2 2019-2020

Si l'esfera es veu de color groc quan se la il·lumina amb llum blanca vol dir que l'esfera és “de color” groc (reflecteix de maner difusa el groc = R+G i absorbeix el blau), per tant les constants del material Ka i Kd tenen valor groc. Per exemple Ka=(0.3,0.3,0) i Kd=(0.8,0.8,0).

El focus de llum està situat damunt de l'observador, per tant hi ha una part de l'esfera que l'observador la veu però a la que no li arriba la llum del focus. En aquesta part no il·luminada pel focus de llum, només s'hi veurà la component ambient del càlcul de la il·luminació, per tant, com que la Ka és groc i la llum ambient és blanca (gris) aquesta zona inferior de l'esfera es veurà de color groc fosc.

La part més extensa de l'esfera estarà il·luminada pel focus (de color verd) i es veurà d'un degradat de color verd on el més intens estarà en aquella zona on la normal als punts de l'esfera tingui una direcció propera a la direcció en què els arriba la llum ($\cos(\theta)$ apropiat). També s'hi ha de sumar una mica de color groc fosc a tots aquests punts perquè la component ambient està en tota l'esfera.

Com que l'esfera és de metall, això vol dir que és brillant i per tant les constants de reflexió specular Ks i n són altes. Llavors, en una part mig-superior de l'esfera (punts on reflexió specular té una direcció propera a la direcció de visió dels punts per part de l'observador) es veurà una taca specular de color verd (perquè el focus és verd) més brillant que la resta de l'esfera.

Exercici 3:

Disposem de dos cubs amb les seves cares paral·leles als plans de coordenades, longitud d'aresta igual a 2 i centres als punts $(2,1,2)$ i $(5,1,2)$ respectivament. Els dos cubs són de metall gris i s'il·luminen amb un focus de llum verda situat al punt $(20,1,2)$.

Com és possible que la cara del cub_1 situada en $x=3$ es vegi il·luminada si el cub_2 li fa ombra?

Quines altres cares es veuran il·luminades pel focus?

IDI Q2 2019-2020

El cub_2 “hauria de” fer ombra al cub_1 en una escena real, però com que el càcul d'il·luminació amb models empírics és local, això vol dir que **no tenim en compte altres objectes** a l'hora de calcular el color dels punts d'un objecte, per tant quan calculem la il·luminació de la cara del cub_1 situada en $x=3$ aquesta cara “mira cap al focus de llum” (l'angle entre la seva normal i la direcció d'il·luminació és menor de 90°) i per tant es veurà il·luminada.

L'altra cara que també es veurà il·luminada pel focus de llum és la cara del cub_2 que està en $x=6$, que també “mira cap al focus de llum”.

Exercici 4:

Raona amb quins valors inicialitzaries les constants empíriques del material Kd i Ks d'un objecte que té el següent comportament: els reflexos especulars sempre es veuen del mateix color que la llum del focus i la resta de zones il·luminades pel focus es veuen de color groc si el focus és groc i del mateix color que les zones no il·luminades pel focus quan el focus és de color blau.

Penseu-lo vosaltres...

Exercici 5:

Una escena està formada per tres cubs d'aresta 2, centrats als punts $(-5, 0, 0)$, $(0, 0, 0)$ i $(5, 0, 0)$ i amb cares paral·leles als plans de coordenades. Els cubs són de color magenta mat.

Ubiquem un focus de llum blanca en la posició $(0, 0, 0)$. No hi ha llum ambient. De quin color s'observaran les cares dels cubs ubicades en $x=6$ i $x=-4$?

Observació: la ubicació de la càmera permet veure totes dues cares.

- a) Es veuran negres perquè el focus de llum està dins del cub centrat en $(0, 0, 0)$
- b) Si es té activat el *back-face culling*, es veuran les dues cares de color magenta, més fosca la de $x=6$ perquè està més lluny del focus
- c) Es veurà la cara en $x=6$ negra i la $x=-4$ de color magenta
- d) Si es té activat el *back-face culling*, es veuran les dues cares de color magenta, més fosca la de $x=-4$

Penseu-lo vosaltres...

Usability Testing

TR 29.3820
August 24, 2006

James R. Lewis

IBM Software Group

Boca Raton, Florida

Abstract

Usability testing is an essential skill for usability practitioners – professionals whose primary goal is to provide guidance to product developers for the purpose of improving the ease-of-use of their products. It is by no means the only skill with which usability practitioners must have proficiency, but it is an important one. A recent survey of experienced usability practitioners indicated that usability testing is a very frequently used method, second only to the use of iterative design. One goal of this chapter is to provide an introduction to the practice of usability testing. This includes some discussion of the concept of usability and the history of usability testing, various goals of usability testing, and running usability tests. A second goal is to cover more advanced topics, such as sample size estimation for usability tests, computation of confidence intervals, and the use of standardized usability questionnaires.

ITIRC Keywords

Usability evaluation
Usability testing
Formative
Summative
Sample size estimation
Confidence intervals
Standardized usability questionnaires

*NOTE: The contents of this technical report have been published as a chapter in the Handbook of Human Factors and Ergonomics (3rd Edition) – Lewis, J. R. (2006). Usability testing. In G. Salvendy (ed.), *Handbook of Human Factors and Ergonomics* (pp. 1275-1316). Hoboken, NJ: John Wiley. The most recent version of this technical report is available at <http://drjim.0catch.com>.*

Contents

INTRODUCTION	1
THE BASICS	1
What is Usability?	1
What is Usability Testing?.....	2
<i>Where Did Usability Testing Come From?</i>	4
<i>Is Usability Testing Effective?</i>	5
Goals of Usability Testing.....	6
<i>Problem Discovery Test</i>	7
<i>Measurement Test Type I: Comparison against Quantitative Objectives</i>	7
<i>Measurement Test Type II: Comparison of Products</i>	9
Variations on a Theme: Other Types of Usability Tests.....	10
<i>Think Aloud</i>	10
<i>Multiple Simultaneous Participants</i>	11
<i>Remote Evaluation</i>	11
Usability Laboratories	12
Test Roles	13
<i>Test Administrator</i>	14
<i>Briefer</i>	14
<i>Camera Operator</i>	14
<i>Data Recorder</i>	14
<i>Help Desk Operator</i>	14
<i>Product Expert</i>	14
<i>Statistician</i>	14
Planning the Test.....	15
<i>Purpose of Test</i>	15
<i>Participants</i>	15
<i>Test Task Scenarios</i>	19
<i>Procedure</i>	20
<i>Pilot Testing</i>	21
<i>Number of Iterations</i>	21
<i>Ethical Treatment of Test Participants</i>	21
Reporting Results	22
<i>Describing Usability Problems</i>	22
<i>Crafting Design Recommendations from Problem Descriptions</i>	23
<i>Prioritizing Problems</i>	23
<i>Working with Quantitative Measurements</i>	25
ADVANCED TOPICS	27
Sample Size Estimation	27
<i>Sample Size Estimation for Parameter Estimation and Comparative Studies</i>	27
<i>Example 1: Parameter estimation given estimate of variability and realistic criteria</i>	28
<i>Example 2: Parameter estimation given estimate of variability and unrealistic criteria</i>	29
<i>Example 3: Parameter estimation given no estimate of variability</i>	29
<i>Example 4: Comparing a parameter to a criterion</i>	30
<i>Example 5: Sample size for a paired t-test</i>	31
<i>Example 6: Sample size for a two-groups t-test</i>	31
<i>Example 7: Making power explicit in the sample size formula</i>	32
<i>Appropriate statistical criteria for industrial testing</i>	34
<i>Some tips on reducing variance</i>	35
<i>Some tips for estimating unknown variance</i>	36

<i>Sample Size Estimation for Problem-Discovery (Formative) Studies</i>	37
<i>Adjusting the initial estimate of p</i>	37
<i>Using the adjusted estimate of p</i>	38
<i>Examples of sample size estimation for problem-discovery (formative) studies.....</i>	42
<i>Evaluating sample size effectiveness given fixed n</i>	43
<i>Estimating the number of problems available for discovery.....</i>	44
<i>Some tips on managing p</i>	44
<i>Sample Sizes for Non-Traditional Areas of Usability Evaluation.....</i>	45
Confidence Intervals	45
<i>Intervals Based on t-Scores</i>	45
<i>Binomial Confidence Intervals</i>	46
Standardized Usability Questionnaires	48
<i>The QUIS</i>	48
<i>The CUSI and SUMI.....</i>	49
<i>The SUS</i>	49
<i>The PSSUQ and CSUQ</i>	49
<i>The ASQ.....</i>	53
WRAPPING UP	54
Getting More Information about Usability Testing	54
A Research Challenge: Improved Understanding of Usability Problem Detection.....	54
Usability Testing: Yesterday, Today, and Tomorrow.....	55
Acknowledgements	55
REFERENCES.....	56

INTRODUCTION

Usability testing is an essential skill for usability practitioners – professionals whose primary goal is to provide guidance to product developers for the purpose of improving the ease-of-use of their products. It is by no means the *only* skill with which usability practitioners must have proficiency, but it is an important one. A recent survey of experienced usability practitioners (Vredenburg et al., 2002) indicated that usability testing is a very frequently used method, second only to the use of iterative design.

One goal of this chapter is to provide an introduction to the practice of usability testing. This includes some discussion of the concept of usability and the history of usability testing, various goals of usability testing, and running usability tests. A second goal is to cover more advanced topics, such as sample size estimation for usability tests, computation of confidence intervals, and the use of standardized usability questionnaires.

THE BASICS

What is Usability?

The term ‘usability’ came into general use in the early 1980s. Related terms from that time were ‘user friendliness’ and ‘ease-of-use,’ which ‘usability’ has since displaced in professional and technical writing on the topic (Bevan et al., 1991). The earliest publication (of which I am aware) to include the word ‘usability’ in its title was Bennett (1979).

It is the nature of language that words come into use with fluid definitions. Ten years after the first use of the term ‘usability,’ Brian Shackel (1990) wrote, “one of the most important issues is that there is, as yet, no generally agreed definition of usability and its measurement.” (p. 31) As recently as 1998, Gray and Salzman stated, “Attempts to derive a clear and crisp definition of usability can be aptly compared to attempts to nail a blob of Jell-O to the wall.” (p. 242)

There are several reasons why it has been so difficult to define usability. Usability is not a property of a person or thing. There is no thermometer-like instrument that can provide an absolute measurement of the usability of a product (Dumas, 2003). Usability is an emergent property that depends on the interactions among users, products, tasks and environments.

Introducing a theme that will reappear in several parts of this chapter, there are two major conceptions of usability. These dual conceptions have contributed to the difficulty of achieving a single agreed upon definition. One conception is that the primary focus of usability should be on measurements related to the accomplishment of global task goals (summative, or measurement-based, evaluation). The other conception is that practitioners should focus on the detection and elimination of usability problems (formative, or diagnostic, evaluation).

The first conception has led to a variety of similar definitions of usability, some embodied in current standards (which, to date, have emphasized summative evaluation). For example:

- “The current MUSiC definition of usability is: the ease of use and acceptability of a system or product for a particular class of users carrying out specific tasks in a specific environment; where ‘ease of use’ affects user performance and satisfaction, and ‘acceptability’ affects whether or not the product is used.” (Bevan et al., 1991, p. 652)
- Usability is the “extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.” (ANSI, 2001, p. 3; ISO, 1998, p. 2)
- “To be useful, usability has to be specific. It must refer to particular tasks, particular environments and particular users.” (Alty, 1992, p. 105)

One of the earliest formative definitions of usability (ease-of-use) is from Chapanis (1981):

“Although it is not easy to measure ‘ease of use,’ it is easy to measure difficulties that people have in using something. Difficulties and errors can be identified, classified, counted, and measured. So my premise is that ease of use is inversely proportional to the number and severity of difficulties people have in using software. There are, of course, other measures that have been used to assess ease of use, but I think the weight of the evidence will support the conclusion that these other dependent measures are correlated with the number and severity of difficulties.” (p. 3)

Practitioners in industrial settings generally use both conceptualizations of usability during iterative design. Any iterative method must include a stopping rule to prevent infinite iterations. In the real world, resource constraints and deadlines can dictate the stopping rule (although this rule is valid only if there is a reasonable expectation that undiscovered problems will not lead to drastic consequences). In an ideal setting, the first conception of usability can act as a stopping rule for the second. Setting aside, for now, the question of where quantitative goals come from, the goals associated with the first conception of usability can define when to stop the iterative process of the discovery and resolution of usability problems. This combination is not a new concept. In one of the earliest published descriptions of iterative design, Al-Awar et al. (1981) wrote:

“Our methodology is strictly empirical. You write a program, test it on the target population, find out what’s wrong with it, and revise it. The cycle of test-rewrite is repeated over and over until a satisfactory level of performance is reached. Revisions are based on the performance, that is, the difficulties typical users have in going through the program.” (p. 31)

What is Usability Testing?

Imagine the two following scenarios.

Scenario 1: Mr. Smith is sitting next to Mr. Jones, watching him work with a high-fidelity prototype of a web browser for Personal Digital Assistants (PDAs). Mr. Jones is the third person that Mr. Smith has watched performing these tasks with this version of the prototype. Mr. Smith is not constantly reminding Mr. Jones to talk while he works, but is counting on his proximity to Mr. Jones to encourage verbal expressions when Mr. Jones encounters any difficulty in accomplishing his current task. Mr. Smith takes written notes whenever this happens, and also takes notes whenever he observes Mr. Jones faltering in his use of the application (for example, exploring menus in search of a desired function). Later that day he will use his notes to develop problem reports and, in consultation with the development team, will work on recommendations for product changes that should eliminate or reduce the impact of the reported problems. When a new version of the prototype is ready, he will resume testing.

Scenario 2: Dr. White is watching Mr. Adams work with a new version of a word processing application. Mr. Adams is working alone in a test cell that looks almost exactly like an office, except for the large mirror on one wall and the two video cameras overhead. He has access to a telephone and a number to call if he encounters a difficulty that he cannot overcome. If he places such a call, Dr. White will answer and provide help modeled on the types of help provided at the company’s call centers. Dr. White can see Mr. Adams through the one-way glass as she coordinates the test. She has one assistant working the video cameras for maximum effectiveness and another who is taking time-stamped notes on a computer (coordinated with the video time stamps) as different members of the team notice and describe different aspects of Mr. Adams’ task performance. Software monitors Mr. Adams’ computer, recording all keystrokes and mouse movements. Later that day, Dr. White and her associates will put together a summary of the task performance measurements for the tested version of the application, noting where the performance measurements do not meet the test criteria. They will also create a prioritized list of problems and

recommendations, along with video clips that illustrate key problems, for presentation to the development team at their weekly status meeting.

Both of these scenarios provide examples of usability testing. In Scenario 1, the emphasis is completely on usability problem discovery and resolution (formative, or diagnostic evaluation). In Scenario 2, the primary emphasis is on task performance measurement (summative, or measurement-focused evaluation), but there is also an effort to record and present usability problems to the product developers. Dr. White's team knows that they cannot determine if they've met the usability performance goals by examining a list of problems, but they also know that they cannot provide appropriate guidance to product development if they only present a list of global task measurements. The problems observed in the use of an application provide important clues for redesigning the product (Chapanis, 1981; Norman, 1983). Furthermore, as John Karat (1997, p. 693) observed, "The identification of usability problems in a prototype user interface (UI) is not the end goal of any evaluation. The end goal is a redesigned system that meets the usability objectives set for the system such that users are able to achieve their goals and are satisfied with the product."

These scenarios also illustrate the defining properties of a usability test. During a usability test, one or more observers watch one or more participants perform specified tasks with the product in a specified test environment (comp are this with the ISO/ANSI definition of usability presented earlier in this chapter). This is what makes usability testing different from other User-Centered Design (UCD) methods. In interviews (including the group interview known as a focus group), participants do not perform work-like tasks. Usability inspection methods (such as expert evaluations and heuristic evaluations), also do not include the observation of users or potential users performing work-like tasks. The same is true of techniques such as surveys and card-sorting. Field studies (including contextual inquiry) can involve the observation of users performing work-related tasks in target environments, but restrict the control that practitioners have over the target tasks and environments. Note that this is not necessarily a bad thing, but it is a defining difference between usability testing and field (ethnographic) studies.

This definition of usability testing permits a wide range of variation in technique (Wildman, 1995). Usability tests can be very informal (as in Scenario 1) or very formal (as in Scenario 2). The observer might sit next to the participant, watch through a one-way glass, or watch the on-screen behavior of a participant who is performing specified tasks at a location halfway around the world. Usability tests can be think-aloud (TA) tests, in which observers train participants to talk about what they're doing at each step of task completion and prompt participants to continue talking if they stop. Observers might watch one participant at a time, or might watch participants work in pairs. Practitioners can apply usability testing to the evaluation of low-fidelity prototypes (see Figure 1), Wizard of Oz (WOZ) prototypes (Kelley, 1985), high-fidelity prototypes, products under development, predecessor products, or competitive products.



Figure 1. Practitioner and participant engaging in an informal usability test with a pencil-and-paper prototype. (Photo courtesy of IBM.)

Where Did Usability Testing Come From?

The roots of usability testing lie firmly in the experimental methods of psychology (in particular, cognitive and applied psychology) and human factors engineering, and are strongly tied to the concept of iterative design. In a traditional experiment, the experimenter draws up a careful plan of study that includes the exact number of participants that the experimenter will expose to the different experimental treatments. The participants are members of the population to which the experimenter wants to generalize the results. The experimenter provides instructions and debriefs the participant, but at no time during a traditional experimental session does the experimenter interact with the participant (unless this interaction is part of the experimental treatment). The more formative (diagnostic, focused on problem discovery) the focus of a usability test, the less it is like a traditional experiment (although the requirements for sampling from a legitimate population of users, tasks, and environments still apply). Conversely, the more summative (focused on measurement) a usability test is, the more it should resemble the mechanics of a traditional experiment. Many of the principles of psychological experimentation that exist to protect experimenters from threats to reliability and validity (for example, the control of demand characteristics) carry over into usability testing (Holleran, 1991; Wenger and Spyridakis, 1989).

As far as I can tell, the earliest accounts of iterative usability testing applied to product design came from Alphonse Chapanis and his students (Al-Awar et al., 1981; Chapanis, 1981; Kelley, 1984), with almost immediate influence on product development practices at IBM (Kennedy, 1982; Lewis, 1982) and other companies, notably Xerox (Smith et al., 1982) and Apple (Williams, 1983). Shortly thereafter, John Gould and his associates at the IBM T. J. Watson Research Center began publishing influential papers on

usability testing and iterative design (Gould, 1988; Gould and Boies, 1983; Gould and Lewis, 1984; Gould et al., 1987).

The driving force that separated iterative usability testing from the standard protocols of experimental psychology was the need to modify early product designs as rapidly as possible (as opposed to the scientific goal of developing and testing competing theoretical hypotheses). As Al-Awar et al. (1981) reported, “Although this procedure [iterative usability test, redesign, and retest] may seem unsystematic and unstructured, our experience has been that there is a surprising amount of consistency in what subjects report. Difficulties are not random or whimsical. They do form patterns.” (p. 33)

When, during the early stages of iterative design, difficulties of use become apparent, it is hard to justify continuing to ask test participants to perform the test tasks. There are ethical concerns with intentionally frustrating participants who are using a product with known flaws that the design team can and will correct. There are economic concerns with the time wasted by watching participants who are encountering and recovering from known error-producing situations. Furthermore, any delay in updating the product delays the potential discovery of problems associated with the update or problems whose discovery was blocked by the presence of the known flaws. For these reasons, the earlier you are in the design cycle, the more rapidly you should iterate the cycles of test and design.

Is Usability Testing Effective?

The widespread use of usability testing is evidence that practitioners believe that usability testing is effective. Unfortunately, there are fields in which practitioners’ belief in the effectiveness of their methods does not appear to be warranted by those outside of the field (for example, the use of projective techniques such as the Rorschach test in psychotherapy, Lilienfeld et al., 2000). In our own field, a number of recently published papers have questioned the reliability of usability problem discovery (Kessner et al., 2001; Molich et al., 1998, 2004).

The common finding in these studies has been that observers (either individually or in teams across usability laboratories) who evaluated the same product produced markedly different sets of discovered problems. Molich et al. (1998) had four independent usability laboratories carry out inexpensive usability tests of a software application for new users. The four teams reported 141 different problems, with only one problem common among all four teams. Molich et al. (1998) attributed this inconsistency to variability in the approaches taken by the teams (task scenarios, level of problem reporting). Kessner et al. (2001) had six professional usability teams independently test an early prototype of a dialog box. None of the problems were detected by every team, and 18 problems were described by one team only. Molich et al. (2004) assessed the consistency of usability testing across nine independent organizations that evaluated the same website. They documented considerable variability in methodologies, resources applied, and problems reported. The total number of reported problems was 310, with only two problems reported by six or more organizations, and 232 problems uniquely reported. “Our main conclusion is that our simple assumption that we are all doing the same and getting the same results in a usability test is plainly wrong” (Molich et al., 2004, p. 65).

This is important and disturbing research, but there is a clear need for much more research in this area. A particularly important goal of future research should be to reconcile these studies with the documented reality of usability improvement achieved through iterative application of usability testing. For example, a limitation of research that stops with the comparison of problem lists is that it is not possible to assess the magnitude of the usability improvement (if any) that would result from product redesigns based on design recommendations derived from the problem lists (Wixon, 2003). When comparing problem lists from many labs, one aberrant set of results can have an extreme effect on measurements of consistency across labs, and the more labs that are involved, the more likely this is to happen.

The results of these studies (Kessner et al., 2001; Molich et al., 1998, 2004) stand in stark contrast to the published studies in which iterative usability tests (sometimes in combination with other UCD methods) have led to significantly improved products (Al-Awar et al., 1981; Bailey, 1993; Bailey et al., 1992;

Gould et al., 1987; Kelley, 1984; Kennedy, 1982; Lewis, 1982; Lewis, 1996b; Rutherford and Ramey, 2000). For example, in a paper describing their experiences in product development, Marshall et al. (1990) stated, “Human factors work can be reliable – different human factors engineers, using different human factors techniques at different stages of a product’s development, identified many of the same potential usability defects” (p. 243). Published cost-benefit analyses (Bias and Mayhew, 1994) have demonstrated the value of usability engineering processes that include usability testing, with cost-benefit ratios ranging from 1:2 for smaller projects to 1:100 for larger projects (C. Karat, 1997).

Most of the papers that describe the success of iterative usability testing are case studies (such as Marshall et al., 1990), but a few have described designed experiments. Bailey et al. (1992) compared two user interfaces derived from the same base interface – one modified via heuristic evaluation and the other modified via iterative usability testing (three iterations, five participants per iteration). They conducted this experiment with two interfaces, one character-based and the other a graphical user interface (GUI), with the same basic outcomes. The number of changes indicated by usability testing was much smaller than the number indicated by heuristic evaluation, but user performance was the same with both final versions of the interface. All designs after the first iteration produced faster performance than and, for the character-based interface, were preferred to, the original design. The time to complete the performance testing was about the same as that required for the completion of multi-reviewer heuristic evaluations.

Bailey (1993) provided additional experimental evidence that iterative design based on usability tests leads to measurable improvements in the usability of an application. In the experiment, he studied the designs of eight designers, four with at least four years of professional experience in interface design and four with at least five years of professional experience in computer programming. All designers used a prototyping tool to create a recipes application (eight applications in all). In the first wave of testing, Bailey videotaped participants performing tasks with the prototypes, three different participants per prototype. Each designer reviewed the videotapes of the people using his or her prototype, and used the observations to redesign his or her application. This process continued until each designer indicated that it was not possible to improve his or her application. All designers stopped after three to five iterations. Comparison of the first and last iterations indicated significant improvement in measurements such as number of tasks completed, task completion times, and repeated serious errors.

In conclusion, the results of the studies of Molich et al. (1998, 2004) and similar studies show that usability practitioners must conduct their usability tests as carefully as possible, document their methods completely, and show proper caution when interpreting their results. On the other hand, as Landauer stated in 1997, “There is ample evidence that expanded task analysis and formative evaluation can, and almost always do, bring substantial improvements in the effectiveness and desirability of systems” (p. 204). This is echoed by Desurvire et al. (1992, p. 98), “It is generally agreed that usability testing in both field and laboratory, is far and above the best method for acquiring data on usability.”

Goals of Usability Testing

The fundamental goal of usability testing is to help developers produce more usable products. The two conceptions of usability testing (formative and summative) lead to differences in the specification of goals in much the same way that they contribute to differences in fundamental definitions of usability (diagnostic problem discovery and measurement). Rubin (1994, p. 26) expressed the formative goal as, “The overall goal of usability testing is to identify and rectify usability deficiencies existing in computer-based and electronic equipment and their accompanying support materials prior to release.” Dumas and Redish (1999, p. 11) provided a more summative goal with, “A key component of usability engineering is setting specific, quantitative, usability goals for the product early in the process and then designing to meet those goals.”

These goals are not in direct conflict, but they do suggest different foci that can lead to differences in practice. For example, a focus on measurement typically leads to more formal testing (less interaction between observers and participants) whereas a focus on problem discovery typically leads to less formal testing (more interaction between observers and participants). In addition to the distinction between diagnostic problem discovery and measurement tests, there are two common types of measurement tests: (1) comparison against objectives and (2) comparison of products.

Problem Discovery Test

The primary activity in diagnostic problem discovery tests is the discovery, prioritization, and resolution of usability problems. The number of participants in each iteration of testing should be fairly small, but the overall test plan should be for multiple iterations, each with some variation in participants and tasks. When the focus is on problem discovery and resolution, the assumption is that more global measures of user performance and satisfaction will take care of themselves (Chapanis, 1981). The measurements associated with problem-discovery tests are focused on prioritizing problems, and include frequency of occurrence in the test, likelihood of occurrence during normal usage (taking into account the anticipated usage of the part of the product in which the problem occurred), and magnitude of impact on the participants who experienced the problem. Because the focus is not on precise measurement of the performance or attitudes of participants, problem discovery studies tend to be informal, with a considerable amount of interaction between observers and participants. Some typical stopping rules for iterations are a preplanned number of iterations or a specific problem discovery goal, such as “Identify 90% of the problems available for discovery for these types of participants, this set of tasks, and these conditions of use.” See the section below on sample size estimation and adequacy for more detailed information on setting and using these types of problem-discovery objectives.

Measurement Test Type I: Comparison against Quantitative Objectives

Studies that have a primary focus of comparison against quantitative objectives include two fundamental activities. The first is the development of the usability objectives. The second is iterative testing to determine if the product under test has met the objectives. A third activity (which can take place during iterative testing) is the enumeration and description of usability problems, but this activity is secondary to the collection of precise measurements.

The first step in developing quantitative usability objectives is to determine the appropriate variables to measure. Renger (1991), as part of the work done for the European MUSiC project (Measuring the Usability of Systems in Context) produced a list of potential usability measurements based on 87 papers out of a survey of 500 papers. He excluded purely diagnostic studies and also excluded papers if they did not provide measurements for the combined performance of a user and a system. He categorized the measurements into four classes:

- Class 1: Goal achievement indicators (such as success rate and accuracy)
- Class 2: Work rate indicators (such as speed and efficiency)
- Class 3: Operability indicators (such as error rate and function usage)
- Class 4: Knowledge acquisition indicators (such as learnability and learning rate)

In a later discussion of the MUSiC measures, Macleod et al. (1997) described measures of effectiveness (the level of correctness and completeness of goal achievement in context) and efficiency (effectiveness related to cost of performance – typically the effectiveness measure divided by task completion time). Optional measures were of productive time and unproductive time, with unproductive time consisting of help actions, search actions, and snag (negation, cancelled, or rejected) actions.

Their (Macleod et al., 1997) description of the measures of effectiveness and efficiency seem to have influenced the objectives expressed in ISO 9241-11 (1998, p. iv): “The objective of designing and evaluating visual display terminals for usability is to enable users to achieve goals and meet needs in a particular context of use. ISO 9241-11 explains the benefits of measuring usability in terms of user performance and satisfaction. These are measured by the extent to which the intended goals of use are

achieved, the resources that have to be expended to achieve the intended goals, and the extent to which the user finds the use of the product acceptable.”

In practice (and as recommended in the ANSI Common Industry Format for Usability Test Reports, 2001), the fundamental global measurements for usability tasks are successful task completion rates (for a measure of effectiveness), mean task completion times (for a measure of efficiency), and mean participant satisfaction ratings (either collected on a task-by-task basis or at the end of a test session – see the section below on standardized usability questionnaires for more information on measuring participant satisfaction). There are many other measurements that practitioners could consider (Dumas and Redish, 1999; Nielsen, 1997), including but not limited to:

1. The number of tasks completed within a specified time limit
2. The number of wrong menu choices
3. The number of user errors
4. The number of repeated errors (same user committing the same error more than once)

After determining the appropriate measurements, the next step is to set the goals. Ideally, the goals should have an objective basis and shared acceptance across the various stakeholders, such as Marketing, Development, and Test groups (Lewis, 1982). The best objective basis for measurement goals are data from previous usability studies of predecessor or competitive products. For maximum generalizability, the historical data should come from studies of similar types of participants completing the same tasks under the same conditions (Chapanis, 1988). If this information is not available, then an alternative is for the test designer to recommend objective goals and to negotiate with the other stakeholders to arrive at a set of shared goals.

“Defining usability objectives (and standards) isn’t easy, especially when you’re beginning a usability program. However, you’re not restricted to the first objective you set. The important thing is to establish some specific objectives immediately, so that you can measure improvement. If the objectives turn out to be unrealistic or inappropriate, you can revise them.” (Rosenbaum, 1989, p. 211) Such revisions, however, should take place only in the early stages of gaining experience and taking initial measurements with a product. It is important not to change reasonable goals to accommodate an unusable product.

When setting usability goals, it’s usually better to set goals that make reference to an average (mean) of a measurement than to a percentile. For example, set an objective such as “The mean time to complete Task 1 will be less than five minutes” rather than “95% of participants will complete Task 1 in less than ten minutes”. The statistical reason for this is that sample means drawn from a continuous distribution are less variable than sample medians (the 50th percentile of a sample), and measurements made away from the center of a distribution (for example, measurements made to attempt to characterize the value of the 95th percentile) are even more variable (Blalock, 1972). Cordes (1993) conducted a Monte Carlo study comparing means and medians as measurements of central tendency for time-on-task scores, and determined that the mean should be the preferred metric for usability studies (unless there is missing data due to participants failing to complete tasks, in which case the mean from the study will underestimate the population mean).

A practical reason to avoid percentile goals is that the goal can imply a sample size requirement that is unnecessarily large. For example, you can’t measure accurately at the 95th percentile unless there are at least twenty measurements (in fact, there must be many more than twenty measurements for accurate measurement). For more details, see the section below on sample size estimation for measurement.

An exception to this is the specification of successful task completions (or any other measurement that is based on counting events), which necessarily requires a percentile goal, usually set at or near 100% (unless there are historical data that indicate an acceptable lower level for a specific test). If ten out of ten participants complete a task successfully, the observed completion rate is 100%, but a 90% binomial confidence interval for this result ranges from 74% to 100%. In other words, even perfect performance for ten participants with this type of measure leaves open the possibility (with 90% confidence) that the true

completion rate could be as low as 75%. See the section below on binomial confidence intervals for more information on computing and using this information in usability tests.

After the usability goals have been established, the next step is to collect data to determine if the product has met its goals. Representative participants perform the target tasks in the specified environment as test observers record the target measurements and identify, to the extent possible within the constraints of a more formal testing protocol, details about any usability problems that occur. The usability team conducting the test provides information about goal achievement and prioritized problems to the development team, and a decision is made regarding whether or not there is sufficient evidence that the product has met its objectives. The ideal stopping rule for measurement-based iterations is to continue testing until the product has met its goals.

When there are only a few goals, then it is reasonable to expect to achieve all of them. When there are many goals (for example, five objectives per task multiplied by ten tasks for a total of fifty objectives), then it is more difficult to determine when to declare success and to stop testing. Thus, it is sometimes necessary to specify a meta-objective of the percentage of goals to achieve.

Despite the reluctance of some usability practitioners to conduct statistical tests to quantitatively assess the strength of the available evidence regarding whether or not a product has achieved a particular goal, the best practice is to conduct such tests. The best approach is to conduct multiple *t*-tests or nonparametric analogs of *t*-tests (Lewis, 1993) because this gives practitioners the level of detail that they require. There is a well-known prohibition against doing this because it can lead investigators to mistakenly accept as real that some differences that are due to chance (technically, alpha inflation). On the other hand, if this is the required level of information, then it is an appropriate method (Abelson, 1995). Furthermore, the practice of avoiding alpha inflation is a concern more related to scientific hypothesis testing than to usability testing (Wickens, 1998), although usability practitioners should be aware of its existence and take it into account when interpreting their statistical results. For example, if you compare two products by conducting fifty *t*-tests with alpha set to .10, and only five (10%) of the *t*-tests are significant (have *p* less than .10), then you should question whether or not to use those results as evidence of the superiority of one product over the other. On the other hand, if substantially more than five of the *t*-tests are significant, then you can be more confident that the indicated differences are real.

In addition to (or as an alternative to) conducting multiple *t*-tests, practitioners should compute confidence intervals for their measurements. This applies to the measurements made for the purpose of establishing test criteria (such as measurements made on predecessor versions of the target product or competitive products) and to the measurements made when testing the product under development. See the section below on confidence intervals for more details.

Measurement Test Type II: Comparison of Products

The second type of measurement test is to conduct usability tests for the purpose of directly comparing one product with another. As long as there is only one measurement that decision makers plan to consider, then a standard *t*-test (ideally, in combination with the computation of confidence intervals) will suffice for the purpose of determining which product is superior.

If decision makers care about multiple dependent measures, then standard multivariate statistical procedures (such as MANOVA or discriminant analysis) are not often helpful in guiding a decision about which of two products has superior usability. The statistical reason for this is that multivariate statistical procedures depend on the computation of centroids (a weighted average of multiple dependent measures) using a least-squares linear model that maximizes the difference between the centroids of the two products (Cliff, 1987). If the directions of the measurements are inconsistent (for example, a high task completion rate is desirable, but a high mean task completion time is not), then the resulting centroids are uninterpretable for the purpose of usability comparison. In some cases it is possible to recompute variables so they have consistent directions (for example, recomputing task completion rates as task failure rates). If this is not possible, then another approach is to convert measurements to ranks (Lewis, 1991a) or standardized (*Z*)

scores (Jeff Sauro, personal communication, March 1, 2004) for the purpose of principled combination of different types of measurements.

To help consumers compare the usability of different products, the American National Standards Institute (ANSI) has published the Common Industry Format (CIF) for usability test reports (ANSI, 2001). Originally developed at the National Institute of Standards and Technology (NIST), this test format requires measurement of effectiveness (accuracy and completeness – completion rates, errors, assists), efficiency (resources expended in relation to accuracy and completeness – task completion time), and satisfaction (freedom from discomfort, positive attitude toward use of the product – using any of a number of standardized satisfaction questionnaires). It also requires a complete description of participants and tasks.

Morse (2000) reviewed the NIST IUSR project conducted to pilot test the CIF. The purpose of the CIF is to make it easier for purchasers to compare the usability of different products. The pilot study ran into problems, such as inability to find a suitable software product for both supplier and consumer, reluctance to share information, and uncertainty about how to design a good usability study. To date, there has been little if any use (at least, no published use) of the CIF for its intended purpose.

Variations on a Theme: Other Types of Usability Tests

Think Aloud

In a standard, formal usability test, test participants perform tasks without necessarily speaking as they work. The defining characteristic of a Think Aloud (TA) study is the instruction to participants to talk about what they are doing as they do it (in other words, to produce verbal reports). If participants stop talking (as commonly happens when they become very engaged in a task), they are prompted to resume talking.

The most common theoretical justification for the use of TA is from the work in cognitive psychology (specifically, human problem solving) of Ericsson and Simon (1980). Responding to a review by Nisbett and Wilson (1977) that described various ways in which verbal reports were unreliable, Ericsson and Simon provided evidence that certain kinds of verbal reports could produce reliable data. They stated that reliable verbalizations are those that participants produce during task performance that do not require additional cognitive processing beyond the processing required for task performance and verbalization.

Some discussions of usability testing hold that the best practice in usability testing is to use the TA method in all usability testing. For example, Dumas (2003) encouraged the use of TA because (1) TA tests are more productive for finding usability problems (Virzi, Sorce, and Herbert, 1993) and (2) thinking aloud does not affect user ratings or performance (Bowers and Snyder, 1990). As the references indicate, there is some evidence in support of these statements, but the evidence is mixed.

Earlier prohibitions against the use of TA in measurement-based tests assumed that thinking aloud would cause slower task performance. Bowers and Snyder (1990), however, found no measurable task performance or preference differences between a test group that thought aloud and one that didn't. Surprisingly, there are some experiments in which the investigators reported better task performance when participants were thinking aloud. Berry and Broadbent (1990) provided evidence that the process of thinking aloud invoked cognitive processes that improved rather than degraded performance, but only if people were given (1) verbal instructions on how to perform the task and (2) the requirement to justify each action aloud. Wright and Converse (1992) compared silent with TA usability testing protocols. The results indicated that the think-aloud group committed fewer errors and completed tasks faster than the silent group, and the difference between the groups increased as a function of task difficulty.

Regarding the theoretical justification for and typical practice of TA, Boren and Ramey (2000) noted that TA practice in usability testing often does not conform to the theoretical basis most often cited for it (Ericsson and Simon, 1980). "If practitioners do not uniformly apply the same techniques in conducting thinking-aloud protocols, it becomes difficult to compare results between studies." (Boren and Ramey, 2000,

p. 261) In a review of publications of TA tests and field observations of practitioners running TA tests, they reported inconsistency in explanations to participants about how to think aloud, practice periods, styles of reminding participants to think aloud, prompting intervals, and styles of intervention. They suggest that rather than basing current practice on Ericsson and Simon, a better basis would be speech communication theory, with clearly defined communicative roles for the participant (in the role of domain expert or valued customer, making the participant the primary speaker) and the usability practitioner (the learner or listener, thus, a secondary speaker).

Based on this alternative perspective for the justification of TA, Boren and Ramey (2000) have provided guidance for many situations that are not relevant in a cognitive psychology experiment, but are in usability tests. For example, they recommend that usability practitioners running a TA test should continually use acknowledgement tokens that do not take speakership away from the participant, such as “mm hm?” and “uh-huh?” (with the interrogative intonation) to encourage the participant to keep talking. In normal communication, silence (as recommended by the Ericsson and Simon protocols) is not a nonresponse – the speaker interprets it in a primarily negative way as indicating aloofness or condescension. They avoided providing precise statements about how frequently to provide acknowledgments or somewhat more explicit reminders (such as “And now...?”) because the best cues come from the participants. Practitioners need to be sensitive to these cues as they run the test.

The evidence indicates that, relative to silent participation, TA can affect task performance. If the primary purpose of the test is problem discovery, then TA appears to have advantages over completely silent task completion. If the primary purpose of the test is task performance measurement, then the use of TA is somewhat more complicated. As long as all the tasks in the planned comparisons were completed under the same conditions, then performance comparisons should be legitimate. The use of TA almost certainly prevents generalization of task performance outside of the TA task, but there are many other factors that make it difficult to generalize specific task performance data collected in usability studies.

For example, Cordes (2001) demonstrated that participants assume that the tasks they are asked to perform in usability tests are possible (the “I know it can be done or you wouldn’t have asked me to do it” bias). Manipulations that bring this assumption into doubt can have a strong effect on quantitative usability performance measures, such as increasing the percentage of participants who give up on a task. If uncontrolled, this bias makes performance measures from usability studies unlikely to be representative of real-world performance when users are uncertain as to whether the product they are using can support the desired tasks.

Multiple Simultaneous Participants

Another way to encourage participants to talk during task completion is to have them work together (Wildman, 1995). This strategy is similar to TA in its strengths and limitations.

Hackman and Biers (1992) compared three think-aloud methods: thinking aloud alone (Single), thinking aloud in the presence of an observer (Observer), and verbalizations occurring in a two-person team (Team). They found no significant differences in performance or subjective measures. The Team condition produced more statements of value to designers than the other two conditions, but this was probably due to the differing number of participants producing statements in the different conditions. There were three groups, with 10 participants per group for Single and Observer, and 20 participants (10 two-person teams) for the Team condition. “The major result was that the team gave significantly more verbalizations of high value to designers and spent more time making high value comments. Although this can be reduced to the fact that the team spoke more overall and that there are two people talking rather than one, this finding is not trivial.” (p. 1208)

Remote Evaluation

Recent advances in the technology of collaborative software have made it easier to conduct remote software tests (tests in which the usability practitioner and the test participant are in different locations). This can be an economical alternative to bringing one or more users into a lab for face-to-face user testing.

A participant in a remote location can view the contents of the practitioner's screen, and in a typical system the practitioner can decide whether the participant can control the desktop. System performance is typically slower than that of a local test session.

Some of the advantages of remote testing are (1) access to participants who would otherwise be unable to participate (international, special needs, etc.), (2) the capability for participants to work in familiar surroundings, and (3) no need for either party to install or download additional software. Some of the disadvantages are (1) potential uncontrolled disruptions in the participant's workplace, (2) lack of visual feedback from the participant, and (3) the possibility of compromised security if the participant takes screen captures of confidential material. Despite these disadvantages, McFadden et al. (2002) reported data that indicated that remote testing was effective at improving product designs and that the test results were comparable to the results obtained with more traditional testing.

Usability Laboratories

A typical usability laboratory test suite is a set of soundproofed rooms with a participant area and observer area separated by a one-way glass, and with video cameras and microphones to capture the user experience (Marshall et al., 1990; Nielsen, 1997), possibly with an executive viewing area behind the primary observers' area. The advantages of this type of usability facility are quick setup, a place where designers can see people interacting with their products, videos to provide a historical record and backup for observers, and a professional appearance that raises awareness of usability and reassures customers about commitment to usability. In a survey of usability laboratories, Nielsen (1994) reported a median floor space of 63 m^2 (678 ft^2) for the observer room and 13 m^2 (144 ft^2) for test rooms. This type of laboratory (see Figure 2) is especially important if practitioners plan to conduct formal, summative usability tests.



Figure 2. View of a usability laboratory. (Photo courtesy of IBM.)

If the practitioner focus is on formative, diagnostic problem discovery, then this type of laboratory is not essential (although still convenient). “It is possible to convert a regular office temporarily into a usability laboratory, and it is possible to perform usability testing with no more equipment than a notepad.” (Nielsen, 1997, p. 1561) Making an even stronger statement against the perceived requirement for formal laboratories, Landauer (1997, p. 204) stated, “Many usability practitioners have demanded greater resources and more elaborate procedures than are strictly needed for effective guidance – such as expensive usability labs rather than natural settings for test and observations, time consuming videotaping and analysis where observation and note-taking would serve as well, and large groups of participants to achieve statistical significance when qualitative naturalistic observation of task goals and situations, or of disastrous interface or functionality flaws, would be more to the point.”

Test Roles

There are several ways to categorize the roles that testers need to play in the preparation and execution of a usability test (Dumas and Redish, 1999; Rubin, 1994). Most test teams will not have an individual assigned to each role, and most tests (especially informal problem discovery tests) do not require every role. The actual distribution of skills across a team might vary from these roles, but the standard roles help to organize the skills needed for effective usability testing.

Test Administrator

The test administrator is the usability test team leader. He or she designs the usability study, including the specification of the initial conditions for a test session and the codes to use for data logging. The test administrator's duties include conducting reviews with the rest of the test team, leading in the analysis of data, and putting together the final presentation or report. People in this role should have a solid understanding of the basics of usability engineering, ability to tolerate ambiguity, flexibility (knowing when to deviate from the plan), and good communication skills.

Briefer

The briefer is the person who interacts with the participants (briefing them at the start of the test, communicating with them as required during the test, and debriefing them at the end of the test sessions). On many teams, the same person takes the roles of administrator and briefer. In a think-aloud study, the briefer has the responsibility to keep the participant talking. The briefer needs to have sufficient familiarity with the product to be able to decide what to tell participants when they ask questions. People in this role need to be comfortable interacting with people, and need to be able to restrict their interactions to those that are consistent with the purposes of the test without any negative treatment of the participants.

Camera Operator

The camera operator is responsible for running the audio-visual equipment during the test. He or she must be skilled in the setup and operation of the equipment, and must be able to take directions quickly when it is necessary to change the focus of the camera (for example, from the keyboard to the user manual).

Data Recorder

The video record is useful as a data backup when things start happening quickly during the test, and as a source for video examples when documenting usability problems. The primary data source for a usability study, however, is the notes that the data recorder takes during a test session. There just isn't time to take notes from a more leisurely examination of the video record. Also, the camera doesn't necessarily catch the important action at every moment of a usability study.

For informal studies, the equipment used to record data might be nothing more than a notepad and pencil. Alternatively, the data recorder might use data-logging software to take coded notes (often time-stamped, possibly synchronized with the video). Before the test begins, the data recorder needs to prepare the data-logging software with the category codes defined by the test administrator. Taking notes with data-logging software is a very demanding skill, so the test administrator does not usually assign additional tasks to the person taking this role.

Help Desk Operator

The help desk operator takes calls from the participant if the user experiences enough difficulty to place the call. The operator should have some familiarity with the call-center procedures followed by the company that has designed the product under test, and must also have skills similar to those of the briefer.

Product Expert

The product expert maintains the product and offers technical guidance during the test. The product expert must have sufficient knowledge of the product to recover quickly from product failures and to help the other team members understand the system's actions during the test.

Statistician

A statistician has expertise in measurement and the statistical analysis of data. Practitioners with an educational background in experimental psychology typically have sufficient expertise to take the role of statistician for a usability test team. Informal tests rarely require the services of a statistician, but the team needs a statistician to extract the maximum amount of information from the data gathered during a formal test (especially if the purpose of the formal test was to compare two products using a battery of measurements).

Planning the Test

One of the first activities a test administrator must undertake is to develop a test plan. To do this, the administrator must understand the purpose of the product, the parts of the product that are ready for test, the types of people who will use the product, what they are likely to use the product for, and in what settings.

Purpose of Test

At the highest level, is the primary purpose of the test to identify usability problems or to gather usability measurements? The answer to this question provides guidance as to whether the most appropriate test is formal or informal, think-aloud or silent, problem discovery or quantitative measurement. After addressing this question, the next task is to define any more specific test objectives. For example, an objective for an interactive voice response system (IVR) might be to assess whether participants can accomplish key tasks without encountering significant problems. If data are available from a previous study of a similar IVR, an alternative objective might be to determine whether participants can complete key tasks reliably faster with the new IVR than they did with the previous IVR. Most usability tests will include several objectives.

If a key objective of the test is to compare two products, then an important decision is whether the test will be within-subjects or between-subjects. In a within-subjects test, every participant works with both products, with half of the participants using one product first and the other half using the other product first (a technique known as counterbalancing). In a between-subjects study, the test groups are completely independent. In general, a within-subjects test leads to more precise measurement of product differences (requiring a smaller number of participants for equal precision, primarily due to the reduction in variability that occurs when each participant acts as his or her own control) and the opportunity to get direct subjective product comparisons from participants. For a within-subjects test to be feasible, both products must be available and set up for use in the lab at the same time, and the amount of time needed to complete tasks with both products must not be excessive. If a within-subjects test is not possible, a between-subjects test is a perfectly valid alternative. Note that the statistical analyses appropriate for these two types of tests are different.

Participants

To determine who will participate in the test, the administrator needs to obtain or develop a user profile. A user profile is sometimes available from the marketing group, the product's functional specification, or other product planning documentation. It is important to keep in mind that the focus of a usability test is the end user of a product, not the expected product purchaser (unless the product will be purchased by end users). The most important participant characteristic is that the participant is representative of the population of end-users to whom the administrator wants to generalize the results of the test. Practitioners can obtain participants from employment agencies, internal sources if the participants meet the requirements of the user profile (but avoiding internal test groups), market research firms, existing customers, colleges, newspaper ads, and user groups.

To define representativeness, it is important to specify the characteristics that members of the target population share but are not characteristic of nonmembers. The administrator must do this for the target population at large and any defined subgroups. Within group definition constraints, administrators should seek heterogeneity in the final sample to maximize the generalizability of the results (Chapanis, 1988; Landauer, 1997) and to maximize the likelihood of problem discovery. It is true that performance measurements made with a homogeneous sample will almost always have greater precision than measurements made with a heterogeneous sample, but the cost of that increased precision is limited generalizability. This raises the issue of how to define homogeneity and heterogeneity of participants. After all, at the highest level of categorization, we are all humans, with similar general capabilities and limitations (physical and cognitive). At the other end of the spectrum, we are all individuals – no two alike.

One of the most important defining characteristics for a group in a usability test is specific relevant experience, both with the product and in the domain of interest (work experience, general product experience, specific product experience, experience with the product under test, and experience with similar products). One common categorization scheme is to consider people with less than three months experience as novices, with more than a year of experience as expert, and those in between as intermediate (Dumas and Redish, 1999). Other individual differences that practitioners routinely track and attempt to vary are education level, age, and sex.

When acquiring participants, how can practitioners define the similarity between the participants they can acquire and the target population? An initial step is to develop a taxonomy of the variables that affect human performance (where performance should include the behaviors of indicating preference and other choice behaviors). Gawron et al. (1989) produced a human performance taxonomy during the development of a human performance expert system. They reviewed existing taxonomies and filled in some missing pieces. They structured the taxonomy as having three top levels: environment, subject (person), and task. The resulting taxonomy took up 12 pages in their paper, and covered many areas which would normally not concern a usability practitioner working in the field of computer system usability (for example, ambient vapor pressure, gravity, acceleration, etc.). Some of the key human variables in the Gawron et al. (1989) taxonomy that could affect human performance with computer systems are:

- Physical Characteristics
 - * Age
 - * Agility
 - * Handedness
 - * Voice
 - * Fatigue
 - * Gender
 - * Body and body part size
- Mental State
 - * Attention span
 - * Use of drugs (both prescription and illicit)
 - * Long-term memory (includes previous experience)
 - * Short-term memory
 - * Personality traits
 - * Work schedule
- Senses
 - * Auditory acuity
 - * Tone perception
 - * Tactual
 - * Visual accommodation
 - * Visual acuity
 - * Color perception

These variables can guide practitioners as they attempt to describe how participants and target populations are similar or different. The Gawron et al. (1989) taxonomy does not provide much detail with regard to some individual differences that other researchers have hypothesized to affect human performance or preference with respect to the use of computer systems: personality traits and computer-specific experience.

Aykin and Aykin (1991) performed a comprehensive review of the published studies to that date that involved individual differences in human-computer interaction (HCI). Table 1 lists the individual differences that they found in published HCI studies, the method used to measure the individual difference, and whether there was any indication from the literature that manipulation of that individual difference led to a crossed interaction.

In statistical terminology, an interaction occurs whenever an experimental treatment has a different magnitude of effect depending on the level of a different, independent experimental treatment. A crossed interaction occurs when the magnitudes have different signs, indicating reversed directions of effects. As an example of an uncrossed interaction, consider the effect of turning off the lights on the typing throughput of blind and sighted typists. The performance of the sighted typists would probably be worse, but the presence or absence of light shouldn't affect the performance of the blind typists. As an extreme example of a crossed interaction, consider the effect of language on task completion for people fluent only in French or English. When reading French text, French speakers would outperform English speakers, and vice versa.

Table 1. Results of Aykin and Aykin (1991) review of individual differences in HCI

Individual Difference	Measurement Method	Crossed Interactions
<i>Level of experience</i>	Various methods	No
<i>Jungian personality types</i>	Myers-Briggs Type Indicator	No
<i>Field dependence/independence</i>	Embedded Figures Test	Yes – field dependent participants preferred organized sequential item number search mode, but field independent subjects preferred the less organized keyword search mode (Fowler et al., 1985)
<i>Locus of control</i>	Levenson test	No
<i>Imagery</i>	Individual Differences Questionnaire	No
<i>Spatial ability</i>	VZ-2	No
<i>Type A/Type B personality</i>	Jenkins Activity Survey	No
<i>Ambiguity tolerance</i>	Ambiguity Tolerance Scale	No
<i>Sex</i>	Unspecified	No
<i>Age</i>	Unspecified	No
<i>Other (reading speed and comprehension, intelligence, mathematical ability)</i>	Unspecified	No

For any of these individual differences, the lack of evidence for crossed interactions could be due to a paucity of research involving the individual difference or could reflect the probability that individual differences will not typically cause crossed interactions in HCI. In general, a change made to support a problem experienced by a person with a particular individual difference will either help other users or simply not affect their performance.

For example, John Black (personal communication, 1988) cited the difficulty that field dependent users had working with one-line editors at the time (decades ago) when that was the typical user interface to a mainframe computer. Switching to full-screen editing resulted in a performance improvement for both field dependent and independent users – an uncrossed interaction because both types of users improved, with the performance of field dependent users becoming equal to (thus improving more than) that of field independent users. Landauer (1997) cites another example of this, in which Greene et al. (1986) found that young people with high scores on logical reasoning tests could master database query languages such as SQL with little training, but older or less able people could hardly ever master these languages. They also determined that an alternative way of forming queries, selecting rows from a truth table, allowed almost everyone to make correct specification of queries, independent of their abilities. Because this redesign improved the performance of less able users without diminishing the performance of the more able, it was an uncrossed interaction. In a more recent study, Palmquist and Kim (2000) found that field dependence

affected the search performance of novices using a web browser (with field independent users searching more efficiently), but did not affect the performance of more experienced users.

If there is a reason to suspect that an individual difference will lead to a crossed interaction as a function of interface design, then it could make sense to invest the time (which can be considerable) to categorize users according to these dimensions. Another situation in which it could make sense to invest the time in categorization by individual difference would be if there were reasons to believe that a change in interface would greatly help one or more groups without adversely affecting other groups. (This is a strategy that one can employ when developing hypotheses about ways to improve user interfaces.) It always makes sense to keep track of user characteristics when categorization is easy (for example, age or sex). Another potential use of these types of variables is as covariates (used to reduce estimates of variability) in advanced statistical analyses (Cliff, 1987).

Aykin and Aykin (1991) reported effects of users' levels of experience, but did not report any crossed interactions related to this individual difference. They did report that interface differences tended to affect the performance of novices, but had little effect on the performance of experts. It appears that behavioral differences related to user interfaces (Aykin and Aykin, 1991) and cognitive style (Palmquist and Kim, 2000) tend to fade with practice. Nonetheless, user experience has been one of the few individual differences to receive considerable attention in HCI research (Fisher, 1991; Mayer, 1997; Miller et al., 1997; Smith et al., 1999).

According to Mayer (1997), relative to novices, experts have:

- better knowledge of syntax
- an integrated conceptual model of the system
- more categories for more types of routines
- higher level plans.

Fisher (1991) emphasized the importance of discriminating between computer experience (which he placed on a novice-experienced dimension) and domain expertise (which he placed on a naïve-expert dimension). LaLomia and Sidowski (1990) reviewed the scales and questionnaires developed to assess computer satisfaction, literacy and aptitudes. None of the instruments they surveyed specifically addressed measurement of computer experience. Miller et al. (1997) published the Windows Computer Experience Questionnaire (WCEQ), an instrument specifically designed to measure a person's experience with Windows 3.1. The questionnaire took about five minutes to complete and was reliable (coefficient alpha = .74; test-retest correlation = .97). They found that their questionnaire was sensitive to three experiential factors: general Windows experience, advanced Windows experience, and instruction. Smith et al. (1999) distinguished between subjective and objective computer experience. The paper was relatively theoretical and "challenges researchers to devise a reliable and valid measure" (p. 239) for subjective computer experience, but did not offer one.

One user characteristic not addressed in any of the cited literature is one that becomes very important when designing products for international use – cultural characteristics. For example, it is extremely important that in adapting an interface for use by members of another country that all text is accurately translated. It is also important to be sensitive to the possibility that these types of individual differences might be more likely than others to result in crossed interactions.

For comparison studies, having multiple groups (for example, males and females or experts and novices) allows the assessment of potential interactions that might otherwise go unnoticed. Ultimately, the decision for one or multiple groups must be based on expert judgment and a few guidelines. For example, practitioners should consider sampling from different groups if they have reason to believe:

- There are potential and important differences among groups on key measures (Dickens, 1987)
- There are potential interactions as a function of group (Aykin and Aykin, 1991)

- The variability of key measures differs as a function of group
- The cost of sampling differs significantly from group to group

Gordon and Langmaid (1988) recommended the following approach to defining groups:

1. Write down all the important variables.
2. If necessary, prioritize the list.
3. Design an ideal sample.
4. Apply common sense to collapse cells.

For example, suppose a practitioner starts with 24 cells, based on the factorial combination of six demographic locations, two levels of experience, and the two levels of gender. The practitioner should ask himself or herself whether there is a high likelihood of learning anything new and important after completing the first few cells, or would additional testing be wasteful? Can one learn just as much from having one or a few cells that are homogeneous within cells and heterogeneous between cells with respect to an important variable, but are heterogeneous within cells with regard to other, less important variables? For example, a practitioner might plan to (1) include equal numbers of males and females over and under 40 years of age in each cell, (2) have separate cells for novice and experienced users, and (3) drop intermediate users from the test. The resulting design requires testing only two cells (groups), but a design that did not combine genders and age groups in the cells would have required eight cells.

The final issue is the number of participants to include in the test. According to Dumas and Redish (1999), typical usability tests have 6 to 12 participants divided among two to three subgroups. For any given test, the required sample size depends on the number of subgroups, available resources (time/money), and the purpose of the test (for example, precise measurement versus problem discovery). It also depends on whether a study is single-shot (needing a larger sample size) or iterative (needing a smaller sample size per iteration, building up the total sample size over iterations). For a more detailed treatment of this topic, see the section below on sample size estimation.

Test Task Scenarios

As with participants, the most important consideration for test tasks is that they are representative of the types of tasks real users will perform with the product. For any product, there will be a core set of tasks that anyone using the product will perform. People who use barbecue grills use them to cook. People who use desktop speech dictation products use them to produce text. For usability tests, these are the most important tasks to test.

After defining these core tasks, the next step is to list any more peripheral tasks that the test should cover. If a barbecue grill has an external burner for heating pans, it might make sense to include a task that requires participants to work with that burner. If, in addition to the basic vocabulary in a speech dictation system, the program allows users to enable additional special topic vocabularies such as cooking or sports, then it might make sense to devise a task that requires participants to activate and use one of these topics. Practitioners should avoid frivolous or humorous tasks because what is humorous to one person might be offensive to another.

From the list of test tasks, create scenarios of use (with specific goals) that require participants to perform the identified tasks. Critical tasks can appear in more than one scenario. For repeated tasks, vary the task details to increase the generalizability of the results. When testing relatively complex systems, some scenarios should stay within specific parts of the system (for example, typing and formatting a document) and others should explore usage across different parts of the system (for example, creating a figure using a spreadsheet program, adding it to the document, attaching the document to a note, and sending it to a specified recipient).

The complete specification of a scenario should include several items. It is important to document (but not to share with the participant), the required initial conditions so it will be easy to determine before a

test session starts that the system is ready. The written description of the scenario (presented to the participant) should state what the participant is trying to achieve and why (the motivation), keeping the description of the scenario as short as possible to keep the test session moving quickly. The scenario should end with an instruction for the action the participant should take upon finishing the task (to make it easier to measure task completion times). The descriptions of the scenario's tasks should not typically provide step-by-step instructions on how to complete the task, but should include details (for example, actual names and data) rather than general statements. The order in which participants complete scenarios should reflect the way in which users would typically work and with the importance of the scenario, with important scenarios done first unless there are other less important scenarios that produce outputs that the important scenario requires as an initial condition. Not all participants need to receive the same scenarios, especially if there are different groups under study. The tasks performed by administrators of a web system that manages subscriptions will be different from the tasks performed by users who are requesting subscriptions.

Here are some examples of scenarios:

Example 1: "Frank Smith's business telephone number has changed to (896) 555-1234. Please change the appropriate address book entry so you have this new phone number available when you need it. When you have finished, please say 'I'm done.'"

Example 2: "You've just found out that you need to cancel a car reservation that you made for next Wednesday. Please call the system that you used to make the reservation (1-888-555-1234) and cancel it. When you have finished, please hang up the phone and say 'I'm done.'"

Procedure

The test plan should include a description of the procedures to follow when conducting a test session. Most test sessions include an introduction, task performance, post-task activities, and debriefing.

A common structure for the introduction is for the briefer (see the section above on testing roles) to start with the purpose of the test, emphasizing that its goal is to improve the product, not to test the participant. Participation is voluntary, and the participant can stop at any time without penalty. The briefer should inform the participant that all test results will be confidential. The participant should be aware of any planned audio or video recording. Finally, the briefer should provide any special instructions (for example, think-aloud instructions) and answer any other questions that the participant might have.

The participant should then complete any preliminary questionnaires and forms, such as a background questionnaire, an informed consent form (including consent for any recording, if applicable), and, if necessary, a confidential disclosure form. If the participant will be using a workstation, the briefer should help the participant make any necessary adjustments (unless, of course, the purpose of the test is to evaluate workstation adjustability). Finally, the participant should complete any prerequisite training. This can be especially important if the goal of the study is to investigate usability after some period of use (ease of use) rather than immediate usability (ease of learning).

The procedure section should indicate the order in which participants will complete task scenarios. For each participant, start with the first assigned task scenario and complete additional scenarios until the participant finishes (or runs out of time). The procedure section should specify when and how to interact with participants, according to the type of study. This section should also indicate when it is permissible to provide assistance to participants if they encounter difficulties in task performance.

Normally, practitioners should avoid offering assistance unless the participant is visibly distressed. When participants initially request help at a given step in a task, refer them to documentation or other supporting materials if available. If that doesn't help, then provide the minimal assistance required to keep the participant moving forward in the task, note the assistance, and score the task as failed. When participants ask questions, try to avoid direct answers, instead turning their attention back to the task and

encouraging them to take whatever action seems right at that time. When asking questions of participants, it is important to avoid biasing the participant's response. Try to avoid the use of loaded adjectives and adverbs in questions (Dumas and Redish, 1999). Instead of asking if a task was easy, ask the participant to describe what it was like performing the task. Give a short satisfaction questionnaire (such as the ASQ – see the section on questionnaires for details) at the end of each scenario.

After participants have finished the assigned scenarios, it is common to have them complete a final questionnaire, usually a standard questionnaire and any additional items required to cover other test- or product-specific issues. For standardized questionnaires, ISO lists the SUMI (Software Usability Measurement Inventory – Kirakowski, 1996; Kirakowski and Corbett, 1993) and PSSUQ (Post-Study System Usability Questionnaire – Lewis, 1995, 2002). In addition to the SUMI and PSSUQ, ANSI lists the QUIS (Questionnaire for User Interaction Satisfaction – Chin et al., 1988) and SUS (System Usability Scale – Brooke, 1996) as widely used questionnaires. After completing the final questionnaire, the briefer should debrief the participant. Toward the end of debriefing, the briefer should tell the participant that the test session has turned up several opportunities for product improvement (this is almost always true), and thank the participant for his or her contribution to product improvement. Finally, the briefer should discuss any questions the participant has about the test session, and then take care of any remaining activities, such as completing time cards. If there has been any deception employed in the test (which is rare, but can legitimately happen when conducting certain types of simulations), the briefer must inform the participant.

Pilot Testing

Practitioners should always plan for a pilot test before running a usability test. A usability test is a designed artifact, and like any other designed artifact needs at least some usability testing to find problems in the test procedures and materials. A common strategy is to have an initial walkthrough with a member of the usability test team or some other convenient participant. After making the appropriate adjustments, the next pilot participant should be a more representative participant. If there are no changes made to the design of the usability test after running this participant, then the second pilot participant can become the first real participant (but this is rare). Pilot testing should continue until the test procedures and materials have become stable.

Number of Iterations

It is better to run one usability test than not to run any at all. On the other hand, “usability testing is most powerful and most effective when implemented as part of an iterative product development process” (Rubin, 1994, p. 30). Ideally, usability testing should begin early and occur repeatedly throughout the development cycle. When development cycles are short, it is a common practice to run, at a minimum, exploratory usability tests on prototypes at the beginning of a project, to run a usability test on an early version of the product during the later part of functional testing, and then to run another during system testing. Once the final version of the product is available, some organizations run an additional usability test focused on the measurement of usability performance benchmarks. At this stage of development, it is too late to apply information about any problems discovered during the usability test to the soon-to-be-released version of the product, but the information can be useful as early input to a follow-on product if the organization plans to develop another version of the product.

Ethical Treatment of Test Participants

Usability testing always involves human participants, so usability practitioners must be aware of professional practices in the ethical treatment of test participants. Practitioners with professional education in experimental psychology are usually familiar with the guidelines of the American Psychology Association (APA, see <http://www.apa.org/ethics/>), and those with training in human factors engineering are usually familiar with the guidelines of the Human Factors and Ergonomics Society (HFES, see <http://www.hfes.org/About/Code.html>). It is particularly important (Dumas, 2003) to be aware of the concepts of informed consent (participants are aware of what will happen during the test, agree to participate, and can leave the test at any time without penalty) and minimal risk (participating in the test does not place participants at any greater risk of harm or discomfort than situations normally encountered in daily life). Most usability tests are consistent with guidelines for informed consent and minimal risk. Only

the test administrator should be able to match a participant's name and data, and the names of test participants should be confidential. Anyone interacting with a participant in a usability test has a responsibility to treat the participant with respect.

Usability practitioners rarely use deception in usability tests. One technique in which there is potential use of deception is the Wizard of Oz (WOZ) method (originally, the OZ Paradigm, Kelley, 1985, also see <http://www.musicman.net/oz.html>). In a test using the WOZ method, a human (the Wizard) plays the part of the system, remotely controlling what the participant sees happen in response to the participant's manipulations. This method is particularly effective in early tests of speech recognition interactive voice response (IVR) systems because all the Wizard needs is a script and a phone (Sadowski, 2001). Often, there is no compelling reason to deceive participants, so they know that the system they are working with is remotely controlled by another person for the purpose of early evaluation. If there is a compelling need for deception (for example, to manage the participant's expectations and encourage natural behaviors), then this deception must be revealed to the participant during debriefing.

Reporting Results

There are two broad classes of usability test results, problem reports and quantitative measurements. It is possible for a test report to contain one type exclusively (for example, the ANSI Common Industry Format has no provision for reporting problems), but most usability test reports will contain both types of results.

Describing Usability Problems

"We broadly define a usability defect as: Anything in the product that prevents a target user from achieving a target task with reasonable effort and within a reasonable time. ... Finding usability problems is relatively easy. However, it is much harder to agree on their importance, their causes and the changes that should be made to eliminate them (the fixes)." (Marshall et al., 1990, p. 245)

The best way to describe usability problems depends on the purpose of the descriptions. For usability practitioners, the goal should be to describe problems in such a way that the description leads logically to one or more potential interventions (recommendations). Ideally, the problem description should also include some indication of the importance of fixing the problem (most often referred to as problem severity). For more scientific investigations, there can be value in higher levels of problem description (Keenan et al., 1999), but developers rarely care about these levels of description. They just want to know what they need to do to make things better while also managing the cost (both monetary and time) of interventions (Gray and Salzman, 1998).

The problem description scheme of Lewis and Norman (1986) has both scientific and practical merit because their problem description categories indicate, at least roughly, an appropriate intervention. They stated (p. 413) that "although we do not believe it possible to design systems in which people do not make errors, we do believe that much can be done to minimize the incidence of error, to maximize the discovery of the error, and to make it easier to recover from the error." They separated errors into mistakes (errors due to incorrect intention) and slips (errors due to appropriate intention but incorrect action), further breaking slips down into mode errors (which indicate a need for better feedback or elimination of the mode), capture errors (which indicate a need for better feedback), and description errors (which indicate a need for better design consistency). In one study using this type of problem categorization, Prümper et al. (1992) found that expertise did not affect the raw number of errors made by participants in their study, but experts handled errors much more quickly than novices. The types of errors that experts made were different from those made by novices, with experts' errors primarily occurring at the level of slips rather than mistakes (knowledge errors).

Rasmussen (1986), using an approach similar to that of Lewis and Norman (1986), described three levels of errors: skill-based, rule-based, and knowledge-based. Two relatively new classification schemes are Structured Usability Problem EXtraction, or SUPEX (Cockton and Lavery, 1999) and the User Action

Framework, or UAF (Andre et al., 2000). The UAF requires a series of decisions, starting with an Interaction Cycle (Planning, Physical Actions, Assessment) based on the work of Norman (1986). Most classifications require four or five decisions, with inter-rater reliability (as measured with kappa) highest at the first step ($\kappa=.978$), but remaining high through the fourth and fifth steps ($\kappa>.7$).

Whether any of these classification schemes will see widespread use by usability practitioners is still unknown. There is considerable pressure on practitioners to produce results and recommendations as quickly as possible. Even if these classification schemes see little use by practitioners, effective problem classification is a very important problem to solve as usability researchers strive to compare and improve usability testing methods.

Crafting Design Recommendations from Problem Descriptions

As indicated by the title of this section, the development of recommendations from problem descriptions is a craft rather than a rote procedure. A well-written problem description will often strongly imply an intervention, but it is also often the case that there might be several ways to attack a problem. It can be helpful for practitioners to discuss problems and potential interventions with the other members of their team, and to get input from other stakeholders as necessary (especially, the developers of the product). This is especially important if the practitioner has observed problems but is uncertain as to the appropriate level of description of the problem.

For example, suppose you have written a problem description about a missing Help button in a software application. This could be a problem with the overall design of the software, or might be a problem isolated to one screen. You might be able to determine this by inspecting other screens in the software, but it could be faster to check with one of the developers.

The first recommendations to consider should be for interventions that will have the widest impact on the product. “Global changes affect everything and need to be considered first.” (Rubin, 1994, p. 285) After addressing global problems, continue working through the problem list until there is at least one recommendation for each problem. For each problem, start with interventions that would eliminate the problem, then follow, if necessary, with other less drastic (less expensive, more likely to be implemented) interventions that would reduce the severity of the remaining usability problem. When different interventions involve different tradeoffs, it is important to communicate this clearly in the recommendations. This approach can lead to two tiers of recommendations – those that will happen for the version of the product currently under development (short-term) and those that will happen for a future version of the product (long-term).

Prioritizing Problems

Because usability tests can reveal more problems than there are resources to address, it is important to have some means for prioritizing problems. There are two approaches to prioritization that have appeared in the usability testing literature: (1) judgment driven (Virzi, 1992) and (2) data driven (Dumas and Redish, 1999; Lewis et al., 1990; Rubin, 1994). The bases for judgment-driven prioritizations are the ratings of stakeholders in the project (such as usability practitioners and developers). The bases for data-driven prioritizations are the data associated with the problems, such as frequency, impact, ease of correction, and likelihood of usage of the portion of the product that was in use when the problem occurred. Of these, the most common measurements are frequency and impact (sometimes referred to as severity, although, strictly speaking, severity should include the effect of all of the types of data considered for prioritization). Hassenzahl (2000), in a study of the two approaches to prioritization, found a lack of correspondence between data-driven and judgment-driven severity estimates. This suggests that the preferred approach should be data-driven.

The usual method for measuring the frequency of occurrence of a problem is to divide the number of occurrences within participants by the number of participants. A common method (Dumas and Redish, 1999; Rubin, 1994) for assessing the impact of a problem is to assign impact scores according to whether the problem (1) prevents task completion, (2) causes a significant delay or frustration, (3) has a relatively minor

effect on task performance, or (4) is a suggestion. This is similar to the scheme of Lewis et al. (1990), in which the impact levels were (1) scenario failure or irretrievable data loss (for example, the participant required assistance to get past the problem or caused the participant to believe the scenario to be properly completed when it wasn't), (2) considerable recovery effort (recovery took more than one minute or the participant repeatedly experienced the problem within a scenario), (3) minor recovery effort (the problem occurred only once within a scenario with recovery time at or under one minute), or (4) inefficiency (a problem not meeting any of the other criteria).

When considering multiple types of data in a prioritization process, it is necessary to combine the data in some way. A graphical approach is to create a problem grid with frequency on one axis and impact on the other (see Figure 3). High-frequency high-impact problems would receive treatment before low-frequency low-impact problems. The relative treatment of high-frequency low-impact problems and low-frequency high-impact problems depends on practitioner judgment.

An alternative approach is to combine the data arithmetically. Rubin (1994) described a procedure for combining four levels of impact (using the criteria described above with 4 assigned to the most serious level) with four levels of frequency (4: frequency = 90%; 3: 51-89%; 2: 11-50%; 1: = 10%) by adding the scores. For example, if a problem had an observed frequency of occurrence of 80% and had a minor effect on performance, then its priority would be 5 (a frequency rating of 3 plus an impact rating of 2). With this approach, priority scores can range from a low of 2 to a high of 8. If information is available about the likelihood that a user would work with the part of the product that enables the problem, then this information would be used to adjust the frequency rating. Continuing the example, if the expectation is that only 10% of users would encounter the problem, then the priority would be 3 (a frequency rating of 1 for the 10% x 80%, or 8% likelihood of occurrence plus an impact rating of 2).

A similar strategy is to multiply the observed percentage frequency of occurrence by the impact score. The range of priorities depends on the values assigned to each impact level. Assigning 10 to the most serious impact level leads to a maximum priority (severity) score of 1000 (which can optionally be divided by 10 to create a scale that ranges from 1 to 100). Appropriate values for the remaining three impact categories depend on practitioner judgment, but a reasonable set is 5, 3, and 1. Using those values, the problem with an observed frequency of occurrence of 80% and a minor effect on performance would have a priority of 24 ($80 \times 3/10$). It is possible to extend this method to account for likelihood of use using the same procedure as that described by Rubin (1994), which in the example resulted in modifying the frequency measurement from 80% to 8%. Another way to extend the method is to categorize the likelihood of use with a set of categories such as very high likelihood (assigned a score of 10), high likelihood (assigned a score of 5), moderate likelihood (assigned a score of 3), and low likelihood (assigned a score of 1), and multiplying all three scores to get the final priority (severity) score (then optionally divide by 100 to create a scale that ranges from 1 to 100). Continuing the previous example with the assumption that the task in which the problem occurred has a high likelihood of occurrence, the problem's priority would be 12 ($5 \times 240/100$). In most cases, applying the different data-driven prioritization schemes to the same set of problems should result in a very similar prioritization.

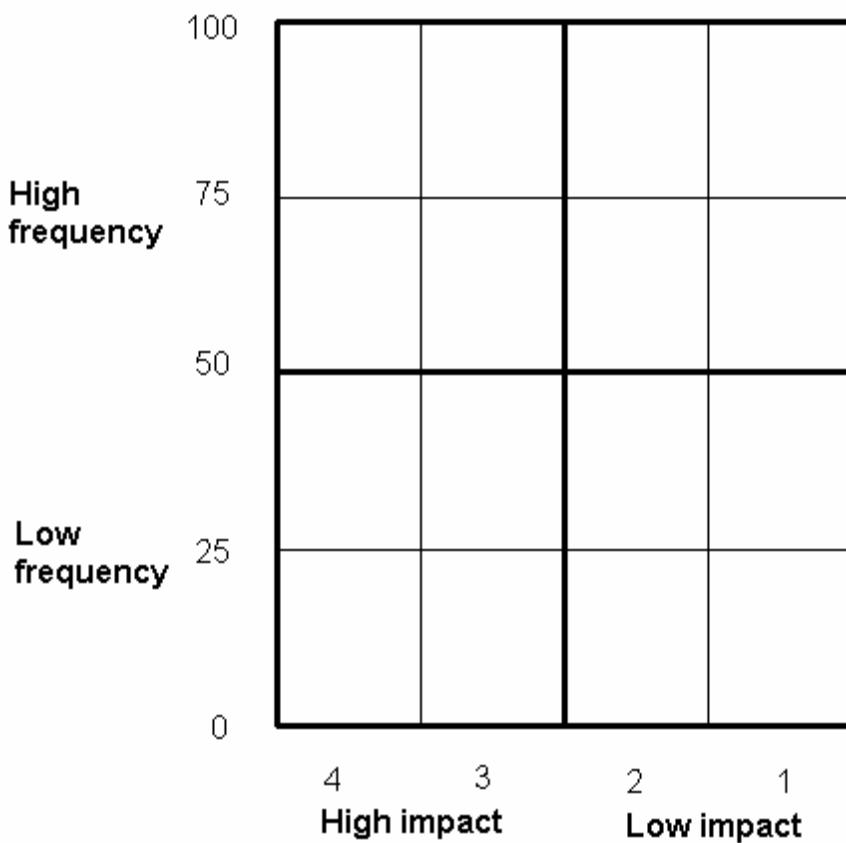


Figure 3. Sample problem grid.

Working with Quantitative Measurements

The most common use of quantitative measurements is to characterize performance and preference variables by computing means, standard deviations, and, ideally, confidence intervals. Practitioners use these results to compare observed to target measurements when targets are available. When targets are not available, the results can still be informative, for example, for use as future target measurements or as relatively gross diagnostic indicators.

The failure to meet targets is an obvious diagnostic cue. A less obvious cue is an unusually large standard deviation. Landauer (1997) describes a case in which the times to record an order were highly variable. The cause for the excessive variability was that a required phone number was sometimes, but not always, available, which turned out to be an easy problem to fix. Because the means and standard deviations of time scores tend to correlate, one way to detect an unusually large variance is to compute the coefficient of variation by dividing the standard deviation by the mean (Jeff Sauro, personal communication, April 26, 2004) or the normalized performance ratio by dividing the mean by the standard deviation (Moffat, 1990). Large coefficients of variation (or, correspondingly, small normalized performance ratios) are potentially indicative of the presence of usability problems.

ADVANCED TOPICS

This section covers more advanced topics in usability testing, including sample size estimation for problem discovery and measurement tests (both comparative and parameter estimation), confidence intervals based on t -scores and binomial confidence intervals, and standardized usability questionnaires. This chapter contains a considerable amount of information about statistical topics because statistical methods do not typically receive much attention in chapters on usability testing and, properly practiced, these techniques can be very valuable. On the other hand, practitioners should keep in mind that the most important factors that lead to successful usability evaluation are the appropriate selection of participants and tasks. No statistical analysis can repair a study in which you watch the wrong people doing the wrong activities.

Sample Size Estimation

The purpose of this section is to discuss the principles of sample size estimation for three types of usability test: population parameter estimation, comparative (also referred to as experimental), and problem-discovery (also referred to as diagnostic, observational, or formative). This section assumes some knowledge of introductory applied statistics, so if you're not comfortable with terms such as mean, variance, standard deviation, p , t -score, and Z-score, refer to an introductory statistics text such as Walpole (1976) for definitions of these and other fundamental terms.

Sample size estimation requires a blend of mathematics and judgment. The computations are straightforward, and it is possible to make reasoned judgments (for example, judgments about expected costs and precision requirements) for those values that the mathematics cannot determine.

Sample Size Estimation for Parameter Estimation and Comparative Studies

Traditional sample size estimation for population parameter estimation and comparative studies depends on having an estimate of the variance of the dependent measure(s) of interest and an idea of how precise (the magnitude of the critical difference and the statistical confidence level) the measurement must be (Walpole, 1976). Once you have that, the rest is mathematical mechanics (typically using the formula for the t statistic.)

You can (1) get an estimate of variance from previous studies using the same method (same or similar tasks and measures), (2) you can run a quick pilot study to get the estimate (for example, piloting with four participants should suffice to provide an initial estimate of variability), or (3) you can set the critical difference you are trying to detect to some fraction of the standard deviation (Diamond, 1981). (See the following examples for more details about these different methods).

Certainly, people prefer precise measurement to imprecise measurement, but all other things being equal, the more precise a measurement is, the more it will cost, and running more participants than necessary is wasteful of resources (Kraemer and Thiemann, 1987). The process of carrying out sample size estimation can also lead usability practitioners and their management to a realistic determination of how much precision they really need to make their required decisions.

Alreck and Settle (1985) recommend using a ‘what if’ approach to help decision makers determine their required precision. Start by asking the decision maker what would happen if the average value from the study was off the true value by one percent. Usually, the response would be that a difference that small wouldn’t matter. Then ask what would happen if the measurement were off by five percent. Continue until you determine the magnitude of the critical difference. Then start the process again, this time pinning down the required level of statistical confidence. Note that statistically unsophisticated decision makers are likely to start out by expecting 100% confidence (which is only possible by sampling every unit in the population). Presenting them with the sample sizes required to achieve different levels of confidence can help them settle in on a more realistic confidence level.

Example 1: Parameter estimation given estimate of variability and realistic criteria. The following example illustrates the process of computing the sample size requirement for the estimation of a population parameter given an existing estimate of variability and realistic measurement criteria. For speech recognition applications, the recognition accuracy is an important value to track due to the adverse effects misrecognitions have on product usability. Thus, part of the process of evaluating the usability of a speech recognition product is estimating its accuracy. For this example, suppose:

- Recognition variability (variance) from a previous similar evaluation = 6.35
- Critical difference (d) = 2.5%
- Desired level of confidence: 90%

The appropriate procedure for estimating a population parameter is to construct a confidence interval (Bradley, 1976). To determine the upper and lower limits of a confidence interval, add to and subtract the following from the observed mean:

$$[1] \text{sem} * t_{crit}$$

where sem is the standard error of the mean (the standard deviation, S , divided by the square root of the sample size, n) and t_{crit} is the t -value associated with the desired level of confidence (found in a t -table, available in most statistics texts). Setting the critical difference to 2.5 is the same as saying that the value of $\text{sem} * t_{crit}$ should be equal to 2.5. In other words, you don't want the upper or lower bound of the confidence interval to be more than 2.5 percentage points away from the observed mean, for a confidence interval width equal to 5.0.

Calculating the sem depends on knowing the sample size, and the value of t_{crit} also depends on the sample size, but you don't know the sample size yet. Iterate using the following method.

1. Start with the Z-score for the desired level of confidence in place of t_{crit} . For 90% confidence, this is 1.645. (By the way, if you actually **know** the true variability for the measurement rather than just having an estimate, you're done at this point because it's appropriate to use the Z-score rather than a t -score. However, you almost never know the true variability, but must work with estimates.)
2. Algebraic manipulations based on the formula $\text{sem} * Z = d$ results in $n = (Z^2 * S^2) / d^2$ which, for this example, is $n = (1.645^2 * 6.35) / 2.5^2$, which equals 2.7. Always round sample size estimates up to the next whole number, so this initial estimate is 3.
3. Now you need to adjust the estimate by replacing the Z-score with the t -score for a sample size of 3. For this estimate, the degrees of freedom (df) to use when looking up the value in a t table is $n-1$, or 2. This is important because the value of Z will always be smaller than the appropriate value of t , making the initial estimate smaller than it should be. For this example, t_{crit} is 2.92.
4. Recalculating for n using 2.92 in place of 1.645 produces 8.66, which rounds up to 9.
5. Because the appropriate value of t_{crit} is now a little smaller than 2.92 (because the estimated sample size is now larger, with 9-1 or 8 degrees of freedom), recalculate n again, using t_{crit} equal to 1.860. The new value for n is 3.515, which rounds up to 4.
6. Stop iterating when you get the same value for n on two iterations or you begin cycling between two values for n , in which case you should choose the larger value. Table 2 shows the full set of iterations for this example, which ends by estimating the appropriate sample size as 5.

Table 2. Full set of iterations for Example 1

	Iteration					
	Initial	1	2	3	4	5
t_{crit}	1.645	2.92	1.86	2.353	2.015	2.132
t_{crit}^2	2.71	8.53	3.46	5.54	4.06	4.55
S^2	6.35	6.35	6.35	6.35	6.35	6.35
d	2.5	2.5	2.5	2.5	2.5	2.5
Estimated n	2.7493	8.663	3.515	5.6252	4.1252	4.618
Rounded up	3	9	4	6	5	5
df	2	8	3	5	4	4

Note that there is nothing in these computations that makes reference to the size of the population. Unless the size of the sample is a significant percentage of the total population under study (which is rare, but correctable using a finite population correction), the size of the population is irrelevant. Alreck and Settle (1985) explain this with a soup-tasting analogy. Suppose you're cooking soup in a one-quart saucepan, and want to test if it's hot enough. You would stir it thoroughly, then taste one teaspoon. If it were a two-quart saucepan, you'd follow the same procedure – stir thoroughly, then taste one teaspoon.

Diamond (1981) points out that you can usually get by with an initial estimate and one iteration because most researchers don't mind having a sample size that's a little larger than necessary. If the cost of each sample is high, though, it makes sense to iterate until reaching one of the stopping criteria. Note that the initial estimate establishes the lower bound for the sample size (3 in this example), and the first iteration establishes the upper bound (9 in this example).

Example 2: Parameter estimation given estimate of variability and unrealistic criteria. The measurement criteria in Example 1 were reasonable – 90% confidence that the interval (limited to a total length of 5%) contains the true mean. The next example (Example 2) shows what happens when the measurement criteria are less realistic, illustrating the potential cost associated with high confidence and high measurement precision. Suppose the measurement criteria for the situation described in Example 1 were less realistic, with:

- Recognition variability from a previous similar evaluation = 6.35
- Critical difference (d) = .5%
- Desired level of confidence: 99%

In that case, the initial Z-score would be 2.576, and the initial estimate of n would be:

$$[2] n = (2.576^2 * 6.35) / .5^2 = 168.549 \text{ (which rounds up to 169).}$$

Recalculating n with t_{crit} equal to 2.605 (t with 168 degrees of freedom) results in n equal to 172.37, which rounds up to 173. (Rather than continuing to iterate, note that the final value for the sample size must lie between 169 and 173.) There might be some industrial environments in which usability investigators would consider 169 to 173 participants a reasonable and practical sample size, but they are rare. (On the other hand, collecting data from this number of participants or more in a mailed survey is common.)

Example 3: Parameter estimation given no estimate of variability. For both Examples 1 and 2, it doesn't matter if the estimate of variability came from a previous study or a quick pilot study. Suppose, however, that you don't have any idea what the measurement variability is, and it's too expensive to run a

pilot study to get an initial estimate. Example 3 illustrates a technique (from Diamond, 1981) for getting around this problem. To do this, though, you need to give up a definition of the critical difference (d) in terms of the variable of interest and replace it with a definition in terms of a fraction of the standard deviation.

In this example, the measurement variance is unknown. To get started, the testers have decided that, with 90% confidence, they do not want d to exceed half the value of the standard deviation. The measurement criteria are:

- Recognition variability from a previous similar evaluation = N/A
- Critical difference (d) = .5 S
- Desired level of confidence: 90%

The initial sample size estimate is:

$$[3] n = (1.645^2 * S^2) / (.5S)^2 = (1.645^2) / (.5)^2 = 10.824, \text{ which rounds up to 11.}$$

The result of the first iteration, replacing 1.645 with t_{crit} for 10 degrees of freedom (1.812), results in a sample size estimation of 13.13, which rounds up to 14. The appropriate sample size is therefore somewhere between 11 and 14, with the final estimate determined by completing the full set of iterations.

Example 4: Comparing a parameter to a criterion. For an example comparing a measured parameter to a criterion value, suppose that you have a product requirement that installation should take no more than 30 minutes. In a preliminary evaluation, participants needed an average of 45 minutes to complete installation. Development has fixed a number of usability problems found in that preliminary study, so you're ready to measure installation time again, using the following measurement criteria:

- Performance variability from the previous evaluation = 10.0
- Critical difference (d) = 3 minutes
- Desired level of confidence: 90%

The interpretation of these measurement criteria is that we want to be 90% confident that we can detect a difference as small as 3 minutes between the mean of the data we gather in the test and the criterion we're trying to beat. In other words, the installation will pass if the observed mean time is 27 minutes or less, because the sample size should guarantee an upper limit to the confidence interval that is no more than 3 minutes above the mean (as long as the observed variance is less than or equal to the initial estimate of the variance). The procedure for determining the sample size in this situation is the same as that of Example 1, shown in Table 3. The outcome of these iterations is a sample size requirement of 6 because the sample size estimates begin cycling between 5 and 6.

Table 3. Full set of iterations for Example 4

	Initial	1	2	3	4
t_{crit}	1.645	2.353	1.943	2.132	2.015
t_{crit}^2	2.706	5.537	3.775	4.545	4.060
s^2	10	10	10	10	10
d	3	3	3	3	3
d^2	9	9	9	9	9
<i>Estimated n</i>	3.006694	6.151788	4.194721	5.050471	4.511361
<i>Rounded up</i>	4	7	5	6	5
<i>df</i>	3	6	4	5	4

Example 5: Sample size for a paired t-test. When you obtain two comparable measurements from each participant in a test (a within-subjects design), you can assess the results using a paired *t*-test. Another name for a ““paired *t*-test” is a “difference score *t*-test”, because the measurements of concern are the mean and standard deviation of the set of difference scores rather than the raw scores. Suppose you plan to obtain recognition accuracy scores from participants who have dictated test texts into your product under development and a competitor’s product (following all the appropriate experimental design procedures such as counterbalancing the order of presentation of products to participants – see a text such as Myers, 1979 for guidance in experimental design), using the following criteria:

- Difference score variability from a previous evaluation = 5.0
- Critical difference (d) = 2%
- Desired level of confidence: 90%

This situation is similar to that of the previous example, because the typical goal of a difference scores *t*-test is to determine if the average difference between the scores is statistically significantly different from 0. Thus, the usability criterion in this case is 0, and we want to be 90% confident that if the true difference between the systems’ accuracies is 2% or more, then we will be able to detect it because the confidence interval for the difference scores will not contain 0. Table 4 shows the iterations for this situation, leading to a sample size estimate of 6.

Table 4. Full set of iterations for Example 5

	Initial	1	2	3	4
t_{crit}	1.645	2.353	1.943	2.132	2.015
t_{crit}^2	2.706	5.537	3.775	4.545	4.060
s^2	5	5	5	5	5
d	2	2	2	2	2
d^2	4	4	4	4	4
<i>Estimated n</i>	3.382531	6.920761	4.719061	5.68178	5.075281
<i>Rounded up</i>	4	7	5	6	6
<i>df</i>	3	6	4	5	5

Example 6: Sample size for a two-groups t-test. Up to this point, the examples have all involved one group of scores, and have been amenable to similar treatment. If you have a situation in which you plan

to compare scores from two independent groups, then things get a little more complicated. For one thing, you now have two sample sizes to consider – one for each group.

To simplify things in this example, assume that the groups are essentially equal (especially with regard to performance variability), which should be the case if the groups contain participants from a single population who have received random assignment to treatment conditions. In this case, it is reasonable to believe that the sample size for both groups will be equal, which simplifies things. For this situation, the formula for the initial estimate of the sample size for each group is:

$$[4] n = (2^*Z^*S^2)/d^2$$

Note that this is similar to the formula presented in Example 1, with the numerator multiplied by 2. After getting the initial estimate, begin iterating using the appropriate value for t_{crit} in place of Z. For example, suppose we needed to conduct the experiment described in Example 5 with independent groups of participants, keeping the measurement criteria the same:

- Estimate of variability from a previous evaluation = 5.0
- Critical difference (d) = 2%
- Desired level of confidence: 90%

In that case, iterations would converge on a sample size of 9 participants per group, for a total sample size of 18, as shown in Table 5.

Table 5. Full set of iterations for Example 6

	Initial	1	2	3
t_{crit}	1.645	1.943	1.833	1.86
t_{crit}^2	2.706	3.775	3.360	3.460
s^2	5	5	5	5
d	2	2	2	2
d^2	4	4	4	4
<i>Estimated n</i>	6.765	9.438	8.400	8.649
<i>Rounded up</i>	7	10	9	9
<i>df</i>	6	9	8	8

This illustrates the well-known measurement efficiency of experiments that produce difference scores (within-subjects designs) relative to experiments involving independent groups (between-subjects designs). For the same measurement precision, the estimated sample size for Example 5 was six participants, 1/3 the sample size requirement estimated for Example 6.

Doing this type of analysis gets more complicated if you have reason to believe that the groups are different, especially with regard to variability of performance. In that case, you would want to have a larger sample size for the group with greater performance variability in an attempt to obtain more equal precision of measurement for each group. Advanced market research texts (such as Brown, 1980) provide sample size formulas for these situations.

Example 7: Making power explicit in the sample size formula. The power of a procedure is not an issue when estimating the value of a parameter, but it is an issue when testing a hypothesis (as in Example 6). In traditional hypothesis testing, there is a null (H_0) and an alternative (H_a) hypothesis. The typical null hypothesis is that there is no difference between groups, and the typical alternative hypothesis is that the difference is greater than zero. When the alternative hypothesis is that the difference is nonzero, the test is

two-tailed because you can reject the null hypothesis with either a sufficiently positive or a sufficiently negative outcome. If you have reason to believe that you can predict the direction of the outcome, or if an outcome in only one direction is meaningful, you can construct an alternative hypothesis that considers only a sufficiently positive or a sufficiently negative outcome (a one-tailed test). For more information, see an introductory statistics text (such as Walpole, 1976).

When you test a hypothesis (for example, that the difference in recognition accuracy between two competitive dictation products is nonzero), there are two ways to make a correct decision and two ways to be wrong, as shown in Table 6.

Table 6. Possible outcomes of a hypothesis test

Decision	Reality	
	H_0 is true	H_0 is false
<i>Insufficient evidence to reject H_0</i>	Fail to reject H_0	Type II error
<i>Sufficient evidence to reject H_0</i>	Type I error	Reject H_0

Strictly speaking, you never accept the null hypothesis, because the failure to acquire sufficient evidence to reject the null hypothesis could be due to (1) no significant difference between groups or (2) a sample size too small to detect an existing difference. Rather than accepting the null hypothesis, you fail to reject it.

Returning to Table 6, the two ways to be right are (1) to fail to reject the null hypothesis (H_0) when it is true or (2) to reject the null hypothesis when it is false. The two ways to be wrong are (1) to reject the null hypothesis when it is true (Type I error) or (2) to fail to reject the null hypothesis when it is false (Type II error).

Table 7 shows the relationship between these concepts and their corresponding statistical testing terms:

Table 7. Statistical testing terms

Statistical Concept	Testing Term
Acceptable probability of a Type I error	Alpha
Acceptable probability of a Type II error	Beta
Confidence	1-alpha
Power	1-beta

The formula presented in Example 6 for an initial sample size estimate was:

$$[5] n = (2*Z^2*S^2)/d^2$$

In Example 6, the Z-score was set for 90% confidence (which means alpha = .10). To take power into account in this formula, you need to add another Z-score to the formula – the Z-score associated with the desired power of the test (as defined in Table 7). Thus, the formula becomes:

$$[6] n = (2*(Z_a + Z_b)^2 * S^2)/d^2$$

So, what was the value for power in Example 6? When beta equals .5 (in other words, when the power is 50%), the value of z_b is 0, so z_b disappears from the formula. Thus, in Example 6, the implicit power was 50%. Suppose you want to increase the power of the test to 80%, reducing beta to .2.

- Estimate of variability from a previous evaluation = 5.0

- Critical difference (d) = 2%
- Desired level of confidence: 90% ($Z_a=1.645$)
- Desired power: 80% ($Z_b=1.282$)

With this change, the iterations converge on a sample size of 24 participants per group, for a total sample size of 48, as shown in Table 8. To achieve the stated goal for power results in a considerably larger sample size.

Table 8. Full set of iterations for Example 7

	Initial	1	2	3
$t(\alpha)$	1.645	1.721	1.714	1.717
$t(\beta)$	1.282	1.323	1.319	1.321
$t(\text{total})$	2.927	3.044	3.033	3.038
$t(\text{total})^2$	8.567	9.266	9.199	9.229
S^2	5	5	5	5
d	2	2	2	2
d^2	4	4	4	4
<i>Estimated n</i>	21.418	23.165	22.998	23.074
<i>Rounded up</i>	22	24	23	24
<i>df</i>	21	23	22	23

Note that the stated power of a test is relative to the critical difference – the smallest effect worth finding. Either increasing the value of the critical difference or reducing the power of a test will result in a smaller required sample size.

Appropriate statistical criteria for industrial testing. In scientific publishing, the usual criterion for statistical significance is to set the permissible Type I error (α) equal to 0.05. This is equivalent to seeking to have 95% confidence that the effect is real rather than random, and is focused on controlling the Type I error (the likelihood that you decide that an effect is real when it's random). There is no corresponding scientific recommendation for the Type II error (β , the likelihood that you will conclude an effect is random when it's real), although some suggest setting it to .20 (Diamond, 1981). The rationale behind the emphasis on controlling the Type I error is that it is better to delay the introduction of good information into the scientific database (a Type II error) than to let erroneous information in (a Type I error).

In industrial evaluation, the appropriate values for Type I and Type II errors depend on the demands of the situation – whether the cost of a Type I or Type II error would be more damaging to the organization. Because we are often resource-constrained, especially with regard to making timely decisions to compete in dynamic marketplaces, this paper has used measurement criteria (such as 90% confidence rather than 95% confidence and fairly large values for d) that seek a greater balance between Type I and Type II errors than is typical in work designed to result in scientific publications. Nielsen (1997) has suggested that 80% confidence is appropriate for practical development purposes. For an excellent discussion of this topic for usability researchers, see Wickens (1998). For other technical issues and perspectives, see Landauer (1997).

Another way to look at the issue is to ask the question, “Am I typically interested in small high-variability effects or large low-variability effects?” The correct answer depends on the situation, but in usability testing, the emphasis is on the detection of large low-variability effects (either large performance effects or frequently-occurring problems). You shouldn’t need a large sample to verify the existence of large low-variability effects. Some writers equate sample size with population coverage, but this isn’t true. A small sample size drawn from the right population provides better coverage than a large sample size drawn

from the wrong population. The statistics involved in computing confidence intervals from small samples compensate for the potentially smaller variance in the small sample by forcing the confidence interval to be wider than that for a larger sample (specifically, the value of t is greater when samples are smaller).

Coming from a different tradition than usability research, many market research texts provide rules of thumb recommending large sample sizes. For example, Aaker and Day (1986) recommend a minimum of 100 per group, with 20-50 for subgroups. For national surveys with many subgroup analyses, the typical total sample size is 2500 (Sudman, 1976). These rules of thumb do not make any formal contact with statistical theory, and may in fact be excessive, depending on the goals of the study. Other market researchers (for example, Banks, 1965) do promote a careful evaluation of the goals of a study.

It is urged that instead of a policy of setting uniform requirements for type I and II errors, regardless of the economic consequences of the various decisions to be made from experimental data, a much more flexible approach be adopted. After all, if a researcher sets himself a policy of always choosing the apparently most effective of a group of alternative treatments on the basis of data from unbiased surveys or experiments and pursues this policy consistently, he will find that in the long run he will be better off than if he chose any other policy. This fact would hold even if none of the differences involved were statistically significant according to our usual standards or even at probability levels of 20 or 30 percent. (Banks, 1965, p. 252)

Finally, Alreck and Settle (1985) provide an excellent summary of the factors indicating appropriate use of large and small samples.

Use a large sample size when:

1. Decisions based on the data will have very serious or costly consequences
2. The sponsors (decision-makers) demand a high level of confidence
3. The important measures have high variance
4. Analyses will require the dividing of the total sample into small subsamples
5. Increasing the sample size has a negligible effect on the cost and timing of the study
6. Time and resources are available to cover the cost of data collection

Use a small sample size when:

1. The data will determine few major commitments or decisions
2. The sponsors (decision-makers) require only rough estimates
3. The important measures have low variance
4. Analyses will use the entire sample, or just a few relatively large subsamples
5. Costs increase dramatically with sample size
6. Budget constraints or time limitations limit the amount of data you can collect

Some tips on reducing variance. Because measurement variance is such an important factor in sample size estimation for these types of studies, it generally makes sense to attempt to manage variance (although in some situations, such management is out of a practitioner's control). Here are some ways to reduce variance:

- Make sure participants understand what they are supposed to do in the study. Unless potential participant confusion is part of the evaluation (and it sometimes is), it can only add to measurement variance.

- One way to accomplish this is through practice trials that allow participants to get used to the experimental situation without unduly revealing study-relevant information.
- If appropriate, use expert rather than novice participants. Almost by definition, expertise implies reduced performance variability (increased automaticity) (Mayer, 1997). With regard to reducing variance, the farther up the learning curve, the better.
- A corollary of this is that if you need to include both expert and novice users, you should be able to get equal measurement precision for both groups with unequal sample sizes (fewer experts required than novices – which is good, because experts are typically harder than novices to recruit as participants).
- If appropriate, study simple rather than complex tasks.
- Use data transformations for measurements that typically exhibit correlations between means and variances or standard deviations. For example, frequency counts often have proportional means and variances (treated with the square root transformation), and time scores often have proportional means and standard deviations (treated with the logarithmic transformation) (Myers, 1979).
- For comparative studies, use within-subjects designs rather than between-subjects designs whenever possible.
- Keep user groups as homogeneous as possible (but although this reduces variability, it can simultaneously pose a threat to a study's external validity if the test group is more homogenous than the population under study) (Campbell and Stanley, 1963).

Keep in mind that it is reasonable to use these tips only when their use does not adversely affect the validity and generalizability of the study. Having a valid and generalizable study is far more important than reducing variability.

Some tips for estimating unknown variance. Parasuraman (1986) described a method for estimating variability if you have an idea about the largest and smallest values for a population of measurements, but don't have the information you need to actually calculate the variability. Estimate the standard deviation (the square root of the variability) by dividing the difference between the largest and smallest values by 6. This technique assumes that the population distribution is normal, and then takes advantage of the fact that 99% of a normal distribution will lie in the range of plus or minus three standard deviations of the mean.

Nielsen (1997) surveyed 36 published usability studies, and found that the mean standard deviation for measures of expert performance was 33% of the mean value of the usability measure (in other words, if the mean completion time was 100 seconds, then the mean standard deviation was about 33 seconds). For novice-user learning the mean standard deviation was 46%, and for measures of error rates the value was 59%.

Churchill (1991) provided a list of typical variances for data obtained from rating scales. Because the number of items in the scale affects the possible variance (with more items leading to more variance), the table takes the number of items into account. For five-point scales, the typical variance is 1.2-2.0; for seven-point scales it is 2.4-4.0; and for ten-point scales it is 3.0-7.0. Because data obtained using rating scales tends to have a more uniform than normal distribution, he advises using a number nearer the high end of the listed range when estimating sample sizes.

Measurement theorists who agree with S. S. Steven's (1951) principle of invariance might yell 'foul' at this point because they believe it is not permissible to calculate averages or variances from rating scale data. There is considerable controversy on this point (for example, see Lord, 1953, Nunnally, 1976, or Harris, 1985). Data reported by Lewis (1993) indicate that taking averages and conducting *t*-tests on multipoint rating data provides far more interpretable and consistent results than the alternative of taking medians and conducting Mann-Whitney *U*-tests. You do have to be careful not to act as if rating scale data are interval data rather than ordinal data when you make claims about the meaning of the outcomes of your statistical tests. An average rating of 4 might be better than an average rating of 2, but you can't claim that it is twice as good (a ratio claim), nor can you claim that the difference between 4 and 2 is equal to the difference between 4 and 6 (an interval claim).

Sample Size Estimation for Problem-Discovery (Formative) Studies

“Having collected data from a few test subjects – and initially a few are all you need – you are ready for a revision of the text.” (Al-Awar et al., 1981, p. 34)

“This research does not mean that all of the *possible* problems with a product appear with 5 or 10 participants, but most of the problems that are going to show up with one sample of tasks and one group of participants will occur early.” (Dumas, 2003, p. 1098)

While these types of general guidelines have been helpful, it is possible to use more precise methods to estimate sample size requirements for problem-discovery usability tests. Estimating sample sizes for tests that have the primary purpose of discovering the problems in an interface depends on having an estimate of p , characterized as the average likelihood of problem occurrence or, alternatively, the problem discovery rate. As with comparative studies, this estimate can come from previous studies using the same method and similar system under evaluation, or can come from a pilot study. For standard scenario-based usability studies, the literature contains large-sample examples that show p ranging from .16 to .42 (Lewis, 1994). For heuristic evaluations, the reported value of p from large-sample studies ranges from .22 to .60 (Nielsen and Molich, 1990).

When estimating p from a small sample, it is important to adjust its initially estimated value because a small-sample estimate of p (for example, fewer than 20 participants) has a bias that results in potentially substantial overestimation of its value (Hertzum and Jacobsen, 2003). A series of Monte Carlo experiments (Lewis, 2001a) have demonstrated that a formula combining Good-Turing discounting with a normalization procedure provides a reasonably accurate adjustment of initial estimates of p (p_{est}), even when the sample size for that initial estimate has as few as two participants (preferably four participants, though, because the variability of estimates of p is greater for smaller samples, Faulkner, 2003; Lewis, 2001a). This formula for the adjustment of p is:

$$[7] p_{adj} = \frac{1}{2}[(p_{est} - 1/n)(1 - 1/n)] + \frac{1}{2}[p_{est}/(1+GT_{adj})]$$

where GT_{adj} is the Good-Turing adjustment to probability space (which is the proportion of the number of problems that occurred once divided by the total number of different problems). The $p_{est}/(1+GT_{adj})$ component in the equation produces the Good-Turing adjusted estimate of p by dividing the observed, unadjusted estimate of p (p_{est}) by the Good-Turing adjustment to probability space. The $(p_{est} - 1/n)(1 - 1/n)$ component in the equation produces the normalized estimate of p from the observed, unadjusted estimate of p and n (the sample size used to estimate p). The reason for averaging these two different estimates is that the Good-Turing estimator tends to overestimate the true value of p , and the normalization tends to underestimate it. For more details and experimental data supporting the use of this formula for estimates of p based on sample sizes from two to ten participants, see Lewis (2001a).

Adjusting the initial estimate of p . Because this is a new procedure, this section contains a detailed illustration of the steps used to adjust an initial estimate of p . To start with, organize the problem discovery data in a table (for example, Table 9) that shows which participants experienced which problems.

Table 9. Hypothetical results for a problem-discovery usability study

Participant	Prob 1	Prob 2	Prob 3	Prob 4	Prob 5	Prob 6	Prob 7	Prob 8	Count	Proportion
1	x		x		x		x	x	5	0.63
2	x	x			x		x		4	0.50
3	x		x	x	x				4	0.50
4	x	x				x			3	0.38
Count	4	2	2	1	3	1	2	1	16	
<i>Proportion</i>	1.00	0.50	0.50	0.25	0.75	0.25	0.50	0.25		0.50

With four participants and eight observed problems, there are 32 cells in the table. The total number of problem occurrences is 16, so the initial estimate of p (p_{est}) is .50 (16/32). Note that averaging the proportion of problem occurrence across participants or across problems also equals .50.

To apply the Good-Turing adjustment, count the number of problems that occurred with only one participant. In Table 9, this happened for three problems (Problems 4, 6, and 8) out of the eight unique problems listed in the table. Thus, the value of GT_{adj} is .375 (3/8), and the value of $p_{est}/(1+GT_{adj})$ is .36 (.5/1.375).

To apply the normalization adjustment, start by computing $1/n$, which in Table 9 is .25 (1/4). The value of $(p_{est} - 1/n)(1 - 1/n)$ is .19 (.25*.75).

The average of the two adjustments produces p_{adj} , which in this example equals .28 ((.36+.19)/2). In this example, the adjusted estimate of p is almost half of the initial estimate.

Using the adjusted estimate of p . Once you have an appropriate (adjusted) estimate for p , you can use the formula $1-(1-p)^n$ (derivable both from the binomial probability formula, Lewis, 1982, 1994, and from the Poisson probability formula, Nielsen and Landauer, 1993) for various values of n from, say, 1 to 20, to generate the curve of diminishing returns expected as a function of sample size. It is possible to get even more sophisticated, taking into account the fixed and variable costs of the evaluation (especially the variable costs associated with the study of additional participants) to estimate when running an additional participant will result in costs that exceed the value of the additional problems discovered (Lewis, 1994).

The Monte Carlo experiments reported in Lewis (2001a) demonstrated that an effective strategy for planning the sample size for a usability study is first to establish a problem discovery goal (for example, 90% or 95%). Run the first two participants and, based on those results, calculate the adjusted value of p using the equation in [7]. This provides an early indication of the likely required sample size, which might estimate the final sample size exactly or, more likely, underestimate by one or two participants (but will provide an early estimate of the required sample size). Collect data from two more participants (for a total of four). Recalculate the adjusted estimate of p using the equation in [7] and project the required sample size

using $1-(1-p)^n$. The estimated sample size requirement based on data from four participants will generally be highly accurate, allowing accurate planning for the remainder of the study. Practitioners should do this even if they have calculated a preliminary estimate of the required sample size from an adjusted value for p obtained from a previous study.

Figure 4 shows the predicted discovery rates for problems of differing likelihoods of observation during a usability study. Several independent studies have verified that these types of predictions fit observed data very closely for both usability and heuristic evaluations (Lewis, 1994; Nielsen and Landauer, 1993; Nielsen and Molich, 1990; Virzi, 1990, 1992; Wright and Monk, 1991). Furthermore, the predictions work both for predicting the discovery of individual problems with a given probability of detection and for modeling the discovery of members of sets of problems with a given mean probability of detection (Lewis, 1994). For usability studies, the sample size is the number of participants. For heuristic evaluations, the sample size is the number of evaluators.

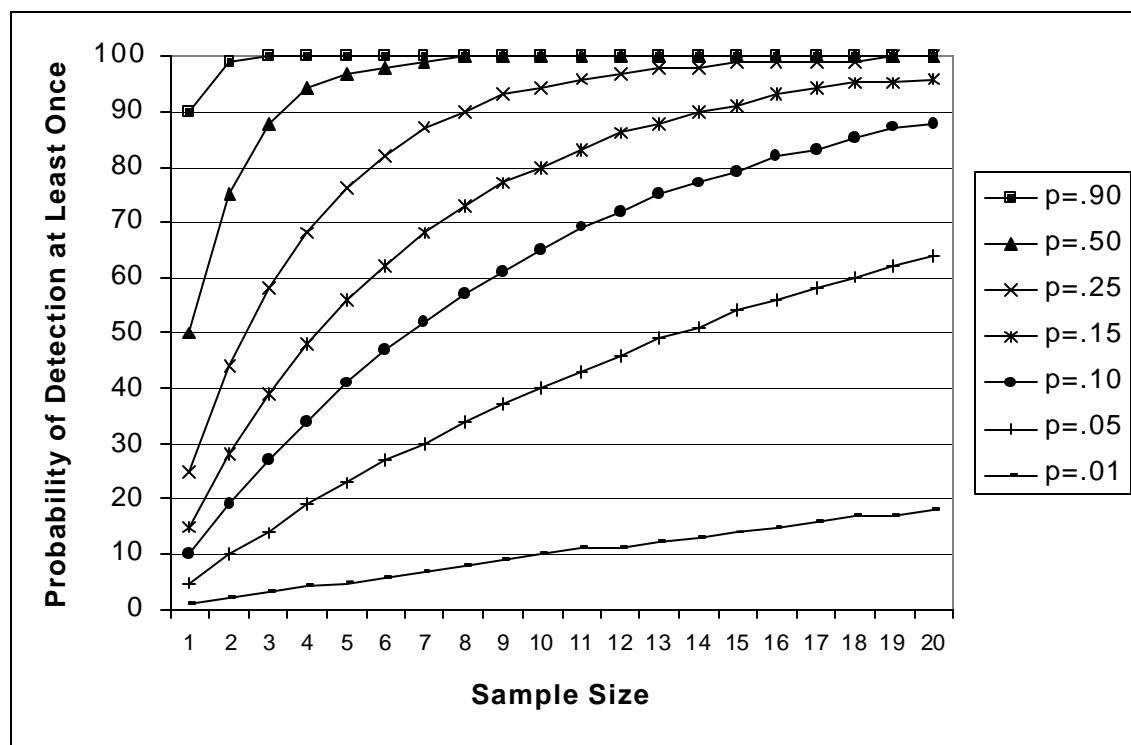


Figure 4. Predicted discovery as a function of problem likelihood

Table 10 (based on a table originally published in Lewis, 1994, but with updated computations to reduce the effect of round-off error) shows problem detection sample size requirements as a function of problem detection probability and the cumulative likelihood of detecting the problem at least once during the study. The required sample size for detecting the problem twice during a study appears in parentheses.

Table 10. Sample size requirements for problem discovery (formative) studies

<i>Problem Occurrence Probability</i>	Cumulative Likelihood of Detecting the Problem at Least Once (Twice)					
	0.50	0.75	0.85	0.90	0.95	0.99
<i>0.01</i>	69 (168)	138 (269)	189 (337)	230 (388)	299 (473)	459 (662)
<i>0.05</i>	14 (34)	28 (53)	37 (67)	45 (77)	59 (93)	90 (130)
<i>0.10</i>	7 (17)	14 (27)	19 (33)	22 (38)	29 (46)	44 (64)
<i>0.15</i>	5 (11)	9 (18)	12 (22)	15 (25)	19 (30)	29 (42)
<i>0.25</i>	3 (7)	5 (10)	7 (13)	9 (15)	11 (18)	17 (24)
<i>0.50</i>	1 (3)	2 (5)	3 (6)	4 (7)	5 (8)	7 (11)
<i>0.90</i>	1 (2)	1 (2)	1 (3)	1 (3)	2 (3)	2 (4)

To use this information to establish a usability sample size, you need to determine three things.

First, what is the average likelihood of problem detection probability (p)? This plays a role similar to the role of variance in the previous examples. If you don't know this value (from previous studies or a pilot study), then you need to decide on the lowest problem detection probability that you want to (or have the resources to) tackle. The smaller this number, the larger is the required sample size.

Second, you need to determine what proportion of the problems that exist at that level you need (or have the resources) to discover during the study (in other words, the cumulative likelihood of problem detection). The larger this number, the larger the required sample size.

Finally, you need to decide whether you are willing to take single occurrences of problems seriously or if problems must appear at least twice before receiving consideration. Requiring two occurrences results in a larger sample size.

For values of p or problem-discovery goals that are outside of tabled values, you can use the formula in [8] (derived algebraically from $Goal=1-(1-p)^n$) to directly compute the sample size required for a given problem discovery goal (taking single occurrences of problems seriously) and value of p .

$$[8] n = \log(1 - Goal)/\log(1 - p)$$

In the example from Table 9, the adjusted value of p was .28. Suppose the practitioner decided that the appropriate problem-discovery goal was to find 97% of the discoverable problems. The computed value of n is 10.6 ($\log(.03)/\log(.72)$, or $-1.522/-1.143$). The practitioner can either round the sample size up to 11 or adjust the problem-discovery goal down to 96.3% ($1-(1-.28)^{10}$).

Lewis (1994) created a return-on-investment (ROI) model to investigate appropriate cumulative problem detection goals. It turned out that the appropriate goal depended on the average problem detection probability in the evaluation – the same value that has a key role in determining the sample size. The model indicated that if the expected value of p was small (say, around 0.10), practitioners should plan to discover about 86% of the problems. If the expected value of p was larger (say, around .25 or .50), practitioners should plan to discover about 98% of the problems. For expected values of p between 0.10 and 0.25, practitioners should interpolate between 87 and 97% to determine an appropriate goal for the percentage of problems to discover.

The cost of an undiscovered problem had a strong effect on the magnitude of the maximum ROI, but contrary to expectation, it had a minor effect on sample size at maximum ROI (Lewis, 1994). Usability practitioners should be aware of these costs in their settings and their effect on ROI (Boehm, 1981), but these costs have relatively little effect on the appropriate sample size for a usability study.

In summary, there is compelling evidence that the law of diminishing returns, based on the cumulative binomial probability formula, applies to problem discovery studies. To use this formula to determine an appropriate sample size, practitioners must form an idea about the expected value of p (the average likelihood of problem detection) for the study and the percentage of problems that the study should uncover. Practitioners can use the ROI model from Lewis (1994) or their own ROI formulas to estimate an appropriate goal for the percentage of problems to discover and can examine data from their own or published usability studies to get an initial estimate of p (which published studies to date indicate can range at least from 0.16 to 0.60). With these two estimates, practitioners can use Table 10 (or, for computations outside of tabled values, the appropriate equations) to estimate appropriate sample sizes for their usability studies.

It is interesting to speculate that a new product that has not yet undergone any usability evaluation is likely to have a higher p than an established product that has gone through several development iterations (including usability testing). This suggests that it is easier (takes fewer participants) to improve a completely new product than to improve an existing product (as long as that existing product has benefited from previous usability evaluation). This is related to the idea that usability testing is a hill-climbing procedure, in which the results of a usability test are applied to a product to push its usability up the hill. The higher up the hill you go, the more difficult it becomes to go higher, because you have already weeded out the problems that were easy to find and fix.

Practitioners who wait to see a problem at least twice before giving it serious consideration can see from Table 10 the sample size implications of this strategy. Certainly, all other things being equal, it is more important to correct a problem that occurs frequently than one that occurs infrequently. However, it is unrealistic to assume that the frequency of detection of a problem is the only criterion to consider in the analysis of usability problems. The best strategy is to consider problem frequency and other problem data (such as severity and likelihood of use) simultaneously to determine which problems are most important to correct rather than establishing a cutoff rule such as “fix every problem that appears two or more times.”

Note that in contrast to the results reported by Virzi (1992), the results reported by Lewis (1994) did not indicate any consistent relationship between problem frequency and impact (severity). It is possible that this difference was due to the different methods used to assess severity (judgment-driven in Virzi, 1992; data-driven in Lewis, 1994). Thus, the safest strategy is for practitioners to assume independence of frequency and impact until further research resolves the discrepancy between the outcomes of these studies.

It is important for practitioners to consider the risks as well as the gains when using small samples for usability studies. Although the diminishing returns for inclusion of additional participants strongly suggest that the most efficient approach is to run a small sample (especially if p is high, if the study will be iterative, and if undiscovered problems will not have dangerous or expensive outcomes), human factors engineers and other usability practitioners must not become complacent regarding the risk of failing to detect low-frequency but important problems.

One could argue that the true number of possible usability problems in any interface is essentially infinite, with an essentially infinite number of problems with non-zero probabilities that are extremely close to zero. For the purposes of determining sample size, the p we are really dealing with is the p that represents the number of discovered problems divided by the number of discoverable problems, where the definition of a discoverable problem is vague, but almost certainly constrained by details of the experimental setting, such as the studied scenarios and tasks and the skill of the observer(s). Despite this vagueness and some

recent criticism of the use of p to model problem-discovery (Caulton, 2001; Woolrych and Cockton, 2001), these techniques seem to work reasonably well in practice (Turner et al., in press).

Examples of sample size estimation for problem-discovery (formative) studies. This section contains several examples illustrating the use of Table 10 as an aid in selecting an appropriate sample size for a problem-discovery study.

A. Given the following problem discovery criteria:

- Detect problems with average probability of: 0.25
- Minimum number of detections required: 1
- Planned proportion to discover: 0.90

The appropriate sample size is 9 participants.

B. Given the same discovery criteria, except the practitioner requires problems to be detected twice before receiving serious attention:

- Detect problems with average probability of: 0.25
- Minimum number of detections required: 2
- Planned proportion to discover: 0.90

The appropriate sample size would be 15 participants.

C. Returning to requiring a single detection, but increasing the planned proportion to discover to .99:

- Detect problems with average probability of: 0.25
- Minimum number of detections required: 1
- Planned proportion to discover: 0.99

The appropriate sample size would be 17 participants.

D. Given the following extremely stringent discovery criteria:

- Detect problems with average probability of: 0.01
- Minimum number of detections required: 1
- Planned proportion to discover: 0.99

The required sample size would be 459 participants (an unrealistic requirement in most settings, implying unrealistic study goals).

Note that there is no requirement to run the entire planned sample through the usability study before reporting clear problems to development and getting those problems fixed before continuing. These required sample sizes are total sample sizes, not sample sizes per iteration. The following testing strategy promotes efficient iterative problem discovery studies and is similar to strategies published by a number of usability specialists (Bailey et al., 1992; Fu et al., 2002; Kantner and Rosenbaum, 1997; Jeffries and Desurvire, 1992; Macleod et al., 1997; Nielsen, 1993; Rosenbaum, 1989).

1. Start with an expert (heuristic) evaluation or one-participant pilot study to uncover the obvious problems. Correct as many of these problems as possible before starting the iterative cycles with Step 2. List all unresolved problems and carry them to Step 2.
2. Watch a small sample of participants (for example, three or four) use the system. Record all observed usability problems. Calculate an adjusted estimate of p based on these results and re-estimate the required sample size.

3. Redesign based on the problems discovered. Focus on fixing high frequency and high impact problems. Fix as many of the remaining problems as possible. Record any outstanding problems so they can remain open for all following iterations.
4. Continue iterating until you have reached your sample size goal (or must stop for any other reason, such as you ran out of time).
5. Record any outstanding problems remaining at the end of testing and carry them over to the next product for which they are applicable.

This strategy blends the benefits of large and small sample studies. During each iteration, you observe only three participants before redesigning the system. Therefore, you can quickly identify and correct the most frequent problems (which means you waste less time watching the next set of participants encounter problems that you already know about). With five iterations, for example, the total sample size would be 15 participants. With several iterations you will identify and correct many less frequent problems because you record and track the uncorrected problems through all iterations.

Note that using this sort of iterative procedure affects estimates of p as you go along. The value of p in the system you end with should generally be lower than the p you started with (as long as the process of fixing problems doesn't create as many other problems). For this reason, it's a good idea to recompute the adjusted value of p after each iteration.

Evaluating sample size effectiveness given fixed n . Suppose you know you only have time to run a limited number of participants, are willing to treat a single occurrence of a problem seriously, and want to determine what you can expect to get out of a problem-discovery study with that number of participants. If that number were six, for example, examination of Table 10 indicates:

- You are almost certain to detect problems that have a .90 likelihood of occurrence (it only takes two participants to have a 99% cumulative likelihood of seeing the problem at least once).
- You are almost certain (between 95 and 99% likely) to detect problems that have a .50 likelihood of occurrence (for this likelihood of occurrence, the required sample size at 95% is 5, and at 99% is 7).
- You've got a reasonable chance (about 80% likely) of detecting problems that have a .25 likelihood of occurrence (for this likelihood of occurrence, the required sample size at 75% is 5, and at 85% is 7).
- You have a little better than even odds of detecting problems that have a .15 likelihood of occurrence (the required sample size at 50% is 5).
- You have a little less than even odds of detecting problems that have a .10 likelihood of occurrence (the required sample size at 50% is 7).
- You are not likely to detect many of the problems that have a likelihood of occurrence of .05 or .01 (for these likelihoods of occurrence, the required sample size at 50% is 14 and 69 respectively).

This analysis illustrates that although a problem-discovery study with a sample size of six participants will typically not discover problems with very low likelihoods of occurrence, the study is almost certainly worth conducting.

Applying this procedure to a number of different sample sizes produces Table 11. The cells in Table 11 are the probability of having a problem with a specified occurrence probability happen at least once during a usability study with the given sample size.

Table 11. Likelihood of discovering problems of probability p at least once in a study with sample size n

<i>Problem Occurrence Probability (p)</i>	<i>Sample Size (n)</i>						
	3	6	9	12	15	18	21
.01	0.03	0.06	0.09	0.11	0.14	0.17	0.19
.05	0.14	0.26	0.37	0.46	0.54	0.60	0.66
.10	0.27	0.47	0.61	0.72	0.79	0.85	0.89
.15	0.39	0.62	0.77	0.86	0.91	0.95	0.97
.25	0.58	0.82	0.92	0.97	0.99	0.99	1.00
.50	0.88	0.98	1.00	1.00	1.00	1.00	1.00
.90	1.00	1.00	1.00	1.00	1.00	1.00	1.00

Estimating the number of problems available for discovery. Another approach to assessing sample size effectiveness is to estimate the number of undiscovered problems. Returning to the situation illustrated in Table 9, the adjusted estimate of p is .28 with four participants and eight unique problems. The estimated proportion of problems discovered with those four participants is .73 ($1-(1-.28)^4$). If eight problems are about 73% of the total number of problems available for discovery, then the total number of problems available for discovery (given the constraints of the testing situation) is about 11 ($8/.73$). Thus, there appear to be about three undiscovered problems. With an estimate of only three undiscovered problems, the sample size of four is approaching adequacy.

Contrast this with the MACERR study described in Lewis (2001a) that had an estimated value of p of .16 with fifteen participants and 145 unique problems. For this study, the estimated proportion of discovered problems at the end of the test was .927 ($1-(1-.16)^{15}$). The estimate of the total number of problems available for discovery was about 156 ($145/.927$). With about 11 problems remaining available for discovery, it might be wise to run a few more participants.

On the other hand, with an estimated 92.7% of problems available for discovery extracted from the problem discovery space defined by the test conditions, it might make more sense to make changes to the test conditions (in particular, to make reasonable changes to the tasks) to create additional opportunities for problem discovery. This is one of many areas in which practitioners need to exercise professional judgment, using the available tables and formulas to guide that judgment.

Some tips on managing p . Because p (the average likelihood of problem discovery) is such an important factor in sample size estimation for usability tests, it generally makes sense to attempt to manage it (although in some situations, such management is out of a practitioner's control). Here are some ways to increase p :

- Use highly-skilled observers for usability studies.
- Use multiple observers rather than a single observer (Hertzum and Jacobsen, 2003).
- Focus evaluation on new products with newly-designed interfaces rather than older, more refined interfaces.
- Study less-skilled participants in usability studies (as long as they are appropriate participants).
- Make the user sample as heterogeneous as possible, within the bounds of the population to which you plan to generalize the results.
- Make the task sample as heterogeneous as possible.
- Emphasize complex rather than simple tasks.

- For heuristic evaluations, use examiners with usability and application domain expertise (double experts, Nielsen, 1992).
- For heuristic evaluations, if you must make a tradeoff between having a single evaluator spend a lot of time examining an interface versus having more examiners spend less time each examining an interface, choose the latter option (Dumas, Sorce, and Virzi, 1995; Virzi, 1997).

Note that some of the tips for increasing p are the opposite of those that reduce measurement variability.

Sample Sizes for Non-Traditional Areas of Usability Evaluation

Non-traditional areas of usability evaluation include activities such as the evaluation of visual design and marketing materials. As with traditional areas of evaluation, the first step is to determine if the evaluation is comparative/parameter estimation or problem-discovery.

Part of the problem with non-traditional areas is that there is less information regarding the values of the variables needed to estimate sample sizes. Another issue is whether these areas are inherently focused on detecting more subtle effects than is the norm in usability testing, which has a focus on large low-variability effects (and correspondingly small sample size requirements). Determining this requires the involvement of someone with domain expertise in these non-traditional areas. It seems, however, that even these non-traditional areas would benefit from focusing on the discovery of large low-variability effects. Only if there was a business case that the investment in a study to detect small, highly-variable effects would ultimately pay for itself should you conduct such a study.

For example, in *The Survey Research Handbook*, Alreck and Settle (1985) point out that the reason that survey samples rarely contain fewer than several hundred respondents is due to the cost structure of surveys. The fixed costs of the survey include activities such as determining information requirements, identifying survey topics, selecting a data collection method, writing questions, choosing scales, composing the questionnaire, etc. For this type of research, the additional or ‘marginal’ cost of including hundreds of additional respondents can be very small relative to the fixed costs. Contrast this with the cost (or feasibility) of adding participants to a usability study in which there might be as little as a week or two between the availability of testable software and the deadline for affecting the product, with resources limiting the observation of participants to one at a time and the test scenarios requiring two days to complete. The potentially high cost of observing participants in usability tests is one reason why usability researchers have devoted considerable attention to sample size estimation, despite some assertions that sample size estimation is relatively unimportant (Wixon, 2003).

“Since the numbers don’t know where they came from, they always behave just the same way, regardless.” (Lord, 1953, p. 751) What potentially differs for non-traditional areas of usability evaluation isn’t the behavior of numbers or statistical procedures, but the researchers’ goals and economic realities.

Confidence Intervals

A major trend in modern statistical evaluation has been a reduced focus on hypothesis testing and a move toward more informative analyses such as effect sizes and confidence intervals (Landauer, 1997). For most applied usability work, confidence intervals are more useful than effect sizes because they have the same units of measurement as the variables from which they are computed. Even when confidence intervals are very wide, they can still be informative, so practitioners should routinely report confidence intervals for their measurements. Although 95% confidence is a commonly used level, confidence as low as 80% will often be appropriate for applied usability measurements (Nielsen, 1997).

Intervals Based on t-Scores

The formulas for the computation of confidence intervals based on t -scores are algebraically equivalent to those used to estimate required sample sizes for measurement-based usability tests, but isolate the critical difference (d) instead of the sample size (n), as shown in [9].

$$[9] d = \text{sem} * t_{crit}$$

where sem is the standard error of the mean (the standard deviation, S , divided by the square root of the sample size, n) and t_{crit} is the t -value associated with the desired level of confidence (found in a t -table, available in most statistics texts). (Practitioners who are concerned about departures from normality can perform a logarithmic transformation on their raw data before computing the confidence interval, then transform the data back to report the mean and confidence interval limits.)

For example, suppose that a task in a usability test with seven participants has an average completion time of 5.4 minutes with a standard deviation of 2.2 minutes. The sem is .83 ($2.2/(7^{1/2})$). For 90% confidence and 6 ($n - 1$) degrees of freedom, the tabled value of t is 1.943. The computed value of d is 1.6 (.83 * 1.943), so the 90% confidence interval is 5.4 ± 1.6 minutes.

As a second example, suppose that the results of a within-subjects test of the time required for two installation procedures showed that the mean of the difference scores (Version A minus Version B) was 2 minutes with a standard deviation of 2 minutes for a sample size of 8 participants. The sem is .71 ($2/(8^{1/2})$). For 95% confidence and 7 ($n - 1$) degrees of freedom, the tabled value of t is 2.365. The computed value of d is 1.7 (.71 * 2.365), so the 95% confidence interval is 2.0 ± 1.7 minutes (ranging from 0.3 to 3.7 minutes). Because the confidence interval does not contain 0, this interval indicates that with alpha of .05 (where alpha is 1 minus the confidence expressed as a proportion rather than a percentage) you should reject the null hypothesis of no difference. The evidence indicates that Version A takes longer than Version B. The major advantage of a confidence interval over a significance test is that you also know with 95% confidence that the magnitude of the difference is probably no less than 0.3 minutes and no greater than 3.7 minutes. If the versions are otherwise equal, then Version B is the clear winner. If the cost of Version B is greater than the cost of Version A (for example, due to a need to license a new technology for Version B), then the decision about which version to implement is more difficult, but is certainly aided by having an estimate of the upper and lower limits of the difference between the two versions.

Binomial Confidence Intervals

As discussed above, confidence intervals constructed around a mean can be very useful. Many usability measurements, however, are proportions or percentages computed from count data rather than means. For example, a usability defect rate for a specific problem is the proportion computed by dividing the number of participants who experience the problem divided by the total number of participants.

The statistical term for a study designed to estimate proportions is a *binomial experiment*, because a given problem either will or will not occur for each trial (participant) in the experiment. For example, a participant either will or will not install an option correctly. The point estimate of the defect rate is the observed proportion of failures (p). However, the likelihood is very small that the point estimate from a study is exactly the same as the true percentage of failures, especially if the sample size is small (Walpole, 1976). To compensate for this, you can calculate interval estimates that have a known likelihood of containing the true proportion (Steele and Torrie, 1960). You can use these binomial confidence intervals to describe the proportion of usability defects effectively, often with only a small sample (Lewis, 1996a). Cordes and Lentz (1986) and Lewis (1996a) provided BASIC programs for the computation of binomial confidence intervals. There are similar programs available at the website of the Southwest Oncology Group Statistical Center (SOGSC, 2004 – http://www.swogstat.org/stat/public/binomial_conf.htm), the GraphPad website (GraphPad, 2004 – <http://graphpad.com/quickcalcs/ConfInterval1.cfm>), and the Measuring Usability website (Sauro, 2004 – http://www.measuringusability.com/conf_intervals.htm).

Some programs (Cordes and Lentz, 1986; Lewis, 1996a; SOGSC, 2004) produce binomial confidence intervals that always contain the exact binomial confidence interval. Other programs (GraphPad, 2004; Sauro, 2004) also produce a new type of interval called *approximate* binomial confidence intervals (Agresti and Coull, 1998). Exact and approximate binomial confidence intervals differ in a number of ways. An exact binomial confidence interval guarantees that the actual confidence is equal to or greater than the nominal

confidence. An approximate interval guarantees that the average of the actual confidence in the long run will be equal to the nominal confidence, but for any specific test, the actual confidence could be lower than the nominal confidence. On the other hand, approximate binomial confidence intervals tend to be narrower than exact intervals. When sample sizes are large ($n > 100$), the two types of intervals are virtually indistinguishable. When sample sizes are small, though, there can be a considerable difference in the width of the intervals, especially when the observed proportion is close to 0 or 1. The exact interval often has an actual confidence closer to 99% when the nominal confidence is 95%, making it too conservative.

Monte Carlo studies that have compared exact and approximate binomial confidence intervals using standard statistical distributions (Agresti and Coull, 1998) and data from usability studies (Sauro and Lewis, 2005) generally support the use of approximate rather than exact binomial confidence intervals. When the actual confidence of an approximate binomial confidence interval is below the nominal level, the actual level tends to be close to the nominal level (for example, Agresti and Coull, 1998, found that the actual level for 95% approximate binomial confidence intervals using the adjusted-Wald method was never less than 89%). “In forming a 95% confidence interval, is it better to use an approach that guarantees that the actual coverage probabilities are *at least* .95 yet typically achieves coverage probabilities of about .98 or .99, or an approach giving narrower intervals for which the actual coverage probability could be less than .95 but is usually quite *close* to .95? For most applications, we would prefer the latter” (Agresti and Coull, 1998, p. 125). This conclusion, that using approximate binomial confidence intervals will tend to produce superior decisions relative to the use of exact intervals, seems to apply to usability test data (Sauro and Lewis, 2005). If, however, it is critical for a specific test to achieve or exceed the nominal level of confidence, then it is reasonable to use an exact binomial confidence interval.

When using binomial confidence intervals, note that if the failure rate is fairly high, you do not need a very large sample to acquire convincing evidence of failure. In the first evaluation of a wordless graphic instruction (Lewis and Pallo, 1991), 9 of 11 installations (82%) were incorrect. The exact 90% binomial confidence interval for this outcome ranged from .53 to .97. This interval allowed us to argue that without intervention, the failure rate for installation would be at least 53% (and more likely closer to the observed 82%).

This suggests that a reasonable strategy for binomial experiments is to start with a small sample size and record the number of failures. From these results, compute a confidence interval. If the lower limit of the confidence interval indicates an unacceptably high failure rate, stop testing. Otherwise, continue testing and evaluating in increments until you reach a specified level of precision or you reach the maximum sample size allowed for the study.

This method can rapidly demonstrate with a small sample that a usability defect is unacceptably high if the criterion is low and the true defect rate is high. Although the confidence interval will be wide (50 percentage points in the graphic symbols example), the lower limit of the interval may be clearly unacceptable. When the true defect rate is low or the criterion is high, this procedure may not work without a large sample size. The decision to continue sampling or to stop the study should be determined by a reasonable business case that balances the cost of continued data collection against the potential cost of allowing defects to go uncorrected.

You cannot use this procedure with small samples to prove that a success rate is acceptably high. With small samples, even if the observed defect percentage is 0 or close to 0%, the interval will be wide, so it will probably include defect percentages that are unacceptable. For example, suppose you have run five participants through a task, and all five have completed the task successfully. The 90% confidence interval on the percentage of defects for these results ranges from 0 to 45%, with a 45% defect rate almost certainly unacceptable. If you had fifty out of fifty successful task completions, the 90% binomial confidence interval would range from 0 to 6%, which would indicate a greater likelihood of the true defect rate being close to 0%. The moral of the story is that it is relatively easy to prove (requires a small sample) that a product is unacceptable, but it is difficult to prove (requires a large sample) that a product is acceptable.

Standardized Usability Questionnaires

Standardized satisfaction measures offer many advantages to the usability practitioner. Specifically, standardized measurements provide objectivity, replicability, quantification, economy, communication, and scientific generalization (Nunnally, 1978). The first published standardized usability questionnaires appeared in the late 1980s (Chin et al., 1988; Kirakowski and Dillon, 1988). Questionnaires focused on the measurement of computer satisfaction preceded these questionnaires (for example, the Gallagher Value of MIS Reports Scale and the Hatcher and Diebert Computer Acceptance Scale – see LaLomia and Sidowski, 1990, for a review), but these questionnaires were not applicable to scenario-based usability tests.

The most widely used standardized usability questionnaires are the Questionnaire for User Interaction Satisfaction (QUIS, Chin et al., 1988), the Software Usability Measurement Inventory (SUMI, Kirakowski, 1996; Kirakowski and Corbett, 1993), the Post-Study System Usability Questionnaire (PSSUQ, Lewis, 1992, 1995, 2002), and the Software Usability Scale (SUS, Brooke, 1996). The most common application of these questionnaires is at the end of a test (after completing a series of test scenarios). The After-Scenario Questionnaire (ASQ, Lewis, 1991b) is a short three-item questionnaire designed for administration immediately following the completion of a test scenario. The ASQ takes less than a minute to complete. The longer standard questionnaires typically have completion times of less than ten minutes (Dumas, 2003).

The primary measures of standardized questionnaire quality are reliability (consistency of measurement) and validity (measurement of the intended attribute) (Nunnally, 1978). There are several ways to assess reliability, including test-retest and split-half reliability. The most common method for the assessment of reliability is coefficient alpha, a measurement of internal consistency. Coefficient alpha can range from 0 (no reliability) to 1 (perfect reliability). Measures that can affect an individual's future, such as IQ tests or college entrance exams should have a minimum reliability of .90 (preferably, reliability greater than .95). For other research or evaluation, measurement reliability in the range of .70 to .80 is acceptable (Landauer, 1997; Nunnally, 1978).

A questionnaire's validity is the extent to which it measures what it claims to measure. Researchers commonly use the Pearson correlation coefficient to assess criterion-related validity (the relationship between the measure of interest and a different concurrent or predictive measure). These correlations do not have to be large to provide evidence of validity. For example, personnel selection instruments with validities as low as .30 or .40 can be large enough to justify their use (Nunnally, 1978). Another approach to validity is content validity, typically assessed through the use of factor analysis (which also helps questionnaire developers discover or confirm clusters of related items that can form reasonable subscales).

Regarding the appropriate number of scale steps, it is true that more scale steps are better than fewer scale steps, but with rapidly diminishing returns. The reliability of individual items is a monotonically increasing function of the number of steps (Nunnally, 1978). As the number of scale steps increase from 2 to 20, the increase in reliability is very rapid at first, but tends to level off at about 7. After 11 steps there is little gain in reliability from increasing the number. The number of steps in an item is very important for measurements based on a single item, but is less important when computing measurements over a number of items (as in the computation of an overall or subscale score).

The QUIS

The QUIS (Chin et al., 1988; Shneiderman, 1987, see <http://lap.umd.edu/QUIS/>) is a product of the Human-Computer Interaction Lab at the University of Maryland. Its use requires the purchase of a license. Chin et al. (1988) evaluated several early versions of the QUIS (Versions 3 through 5). They reported an overall reliability (coefficient alpha) of .94, but did not report any subscale reliability.

The QUIS is currently at Version 7. This version includes demographic questions, an overall measure of system satisfaction, and 11 specific interface factors. The QUIS is available in two lengths, short

(26 items) and long (71 items). The items are nine-point scales anchored with opposing adjective phrases (such as “confusing” and “clear” for the item, “Messages which appear on screen”).

The CUSI and SUMI

The Human Factors Research Group (HFRG) at University College Cork published their first standardized questionnaire, the Computer Usability Satisfaction Inventory (CUSI), in 1988 (Kirakowski and Dillon). The CUSI was a 22-item questionnaire containing two subscales: Affect and Competence. Its overall reliability was .94, with .91 for Affect and .89 for Competence.

The HRFG replaced the CUSI with the SUMI (Kirakowski, 1996; Kirakowski and Corbett, 1993; see <http://www.ucc.ie/hfrg/questionnaires/sumi//index.html>), a questionnaire that has six subscales: Global, Efficiency, Affect, Helpfulness, Control, and Learnability. Its 50 items are statements (such as “The instructions and prompts are helpful.”) to which participants indicate that they agree, are undecided, or disagree. The SUMI has undergone a significant amount of psychometric development and evaluation to arrive at its current form. The results of several sensitivity studies that had significant main effects of system, SUMI scales, and their interaction (for example, McSweeney, 1992, and Wiethoff et al., 1992) support its validity.

The reported reliabilities of the six subscales (measured with coefficient alpha) are:

- Global: .92
- Efficiency: .81
- Affect: .85
- Helpfulness: .83
- Control: .71
- Learnability: .82

One of the greatest strengths of the SUMI is the database of results that is available for the construction of interpretive norms. This makes it possible for practitioners to compare their results with those of similar products (as long as there are similar products in the database). Another strength is that the SUMI is available in different languages (such as UK English, American English, Italian, Spanish, French, German, Dutch, Greek, and Swedish). Like the QUIS, practitioners planning to use SUMI must purchase a license for its use (which includes questionnaires and scoring software). For an additional fee, a trained psychometrician at the HRFG will score the results and produce a report.

The SUS

Usability practitioners at Digital Equipment Corporation (DEC) developed the SUS in the mid 1980s (Dumas, 2003). The ten 5-point items of the SUS provide a unidimensional (no subscales) usability measurement that ranges from 0 to 100. In the first published account of the SUS, Brooke (1996) stated that the SUS was robust, reliable, and valid, but did not publish the specific reliability or validity measurements. With regard to validity, “it correlates well with other subjective measures of usability (e.g., the general usability subscale of the SUMI)” (Brooke, 1996, p. 194). DEC has copyrighted the SUS, but according to Brooke (1996), “the only prerequisite for its use is that any published report should acknowledge the source of the measure” (p. 194).

The PSSUQ and CSUQ

The PSSUQ is a questionnaire designed for the purpose of assessing users’ perceived satisfaction with their computer systems. It has its origin in an internal IBM project called SUMS (System Usability MetricS), headed by Suzanne Henry in the late 1980s. A team of human factors engineers and usability specialists working on SUMS created a pool of 7-point scale items based on the work of Whiteside et al. (1988), and from that pool selected 18 items to use in the first version of the PSSUQ (Lewis, 1992). Each item was positively worded, with the scale anchors “Strongly Agree” at the first scale position (1) and “Strongly Disagree” at the last scale position (7). A Not Applicable (NA) choice and a comment area were available for each item (see Lewis, 1995 for examples of the appearance of the items).

The development of the Computer System Usability Questionnaire (CSUQ) followed the development of the first version of the PSSUQ. Its items are identical to those of the PSSUQ except that their wording is appropriate for use in field settings or surveys rather than in a scenario-based usability test, making it, essentially, an alternate form of the PSSUQ. For a discussion of CSUQ research and comparison of the PSSUQ and CSUQ items, see Lewis (1995).

An unrelated series of IBM investigations into customer perception of usability revealed a common set of five usability characteristics associated with usability by several different user groups (Doug Antonelli, personal communication, January 5, 1991). The 18-item version of the PSSUQ addressed four of these five characteristics (quick completion of work, ease of learning, high-quality documentation and online information, and functional adequacy), but did not address the fifth (rapid acquisition of productivity). The second version of the PSSUQ (Lewis, 1995) included an additional item to address this characteristic, bringing the total number of items up to 19.

Lewis (2002) conducted a psychometric evaluation of the PSSUQ using data from several years of usability studies (primarily studies of speech dictation systems, but including studies of other types of applications). The results of a factor analysis on these data were consistent with earlier factor analyses (Lewis, 1992, 1995) used to define three PSSUQ subscales: System Usefulness (SysUse), Information Quality (InfoQual), and Interface Quality (IntQual). Estimates of reliability were also consistent with those of earlier studies. Analyses of variance indicated that variables such as the specific study, developer, state of development, type of product, and type of evaluation significantly affected PSSUQ scores. Other variables, such as gender and completeness of responses to the questionnaire, did not. Norms derived from the new data correlated strongly with norms derived from earlier studies.

Significant correlation analyses indicated scale validity (Lewis, 1995). For a sample of 22 participants who completed all PSSUQ and ASQ items in a usability study (Lewis et al., 1990), the overall PSSUQ score correlated highly with the sum of the ASQ ratings that participants gave after completing each scenario ($r(20) = .80, p = .0001$). The overall PSSUQ score correlated significantly with the percentage of successful scenario completions ($r(29) = -.40, p = .026$). SysUse ($r(36) = -.40, p = .006$) and IntQual ($r(35) = -.29, p = .08$) also correlated with the percentage of successful scenario completions.

One potential criticism of the PSSUQ has been that some items seemed redundant, and that this redundancy might inflate estimates of reliability. Lewis (2002) investigated the effect of removing three items from the second version of the PSSUQ (Items 3, 5, and 13). With these items removed, the reliability of the overall PSSUQ score (using coefficient alpha) was .94 (remaining very high), and the reliabilities of the three subscales were:

- SysUse: .90
- InfoQual: .91
- IntQual: .83

All of the reliabilities exceeded .80, indicating sufficient reliability to be valuable as usability measurements (Anastasi, 1976; Landauer, 1997). Thus, the third (and current) version of the PSSUQ has 16 7-point scale items (see Table 12 for the items and their normative scores from Lewis, 2002).

Note that the scale construction is such that lower scores are better than higher scores, and that the means of the items and scales all fall below the scale midpoint of 4. With the exception of Item 7 ("The system gave error messages that clearly told me how to fix problems."), the upper limits of the confidence intervals are below 4. This shows that practitioners should not exclusively use the scale midpoint as a reference from which they would judge participants' perceptions of usability. Rather, they should also use the norms shown in Table 12 (and comparison with these norms is probably more meaningful than comparison with the scale midpoint).

Table 12. PSSUQ Version 3 items, scales, and normative scores (99% confidence intervals)

<u>Item/ Scale</u>	<u>Item Text/Scale Scoring Rule</u>	<u>Norms (99% CI)</u>		
		<u>Lower Limit</u>	<u>Mean</u>	<u>Upper Limit</u>
Q1	Overall, I am satisfied with how easy it is to use this system	2.60	2.85	3.09
Q2	It was simple to use this system.	2.45	2.69	2.93
Q3	I was able to complete the tasks and scenarios quickly using this system.	2.86	3.16	3.45
Q4	I felt comfortable using this system.	2.40	2.66	2.91
Q5	It was easy to learn to use this system.	2.07	2.27	2.48
Q6	I believe I could become productive quickly using this system.	2.54	2.86	3.17
Q7	The system gave error messages that clearly told me how to fix problems.	3.36	3.70	4.05
Q8	Whenever I made a mistake using the system, I could recover easily and quickly.	2.93	3.21	3.49
Q9	The information (such as on-line help, on-screen messages and other documentation) provided with this system was clear.	2.65	2.96	3.27
Q10	It was easy to find the information I needed.	2.79	3.09	3.38
Q11	The information was effective in helping me complete the tasks and scenarios.	2.46	2.74	3.01
Q12	The organization of information on the system screens was clear.	2.41	2.66	2.92
Note: The “interface” includes those items that you use to interact with the system. For example, some components of the interface are the keyboard, the mouse, the microphone, and the screens (including their graphics and language).				
Q13	The interface of this system was pleasant.	2.06	2.28	2.49
Q14	I liked using the interface of this system.	2.18	2.42	2.66
Q15	This system has all the functions and capabilities I expect it to have.	2.51	2.79	3.07
Q16	Overall, I am satisfied with this system.	2.55	2.82	3.09
SysUse	Average Items 1 through 6.	2.57	2.80	3.02
InfoQual	Average Items 7 through 12.	2.79	3.02	3.24
IntQual	Average Items 13 through 15.	2.28	2.49	2.71
Overall	Average Items 1 through 16.	2.62	2.82	3.02

Table notes: *SysUse* = system usefulness, *InfoQual* = information quality, *IntQual* = interface quality, *CI* = confidence interval. Means appear in bold face. Scores can range from 1 (strongly agree) to 7 (strongly disagree), with lower scores better than higher scores.

The way that Item 7 stands out from the others indicates:

- It should not surprise practitioners if they find this in their own data.
- It is a difficult task to provide usable error messages throughout a product.
- It may well be worth the effort to focus on providing usable error messages.
- If practitioners find the mean for this item to be equal to or less than the mean of the other items in InfoQual (assuming they are in line with the norms), they have been successful in creating better-than-average error messages.

The consistent pattern of relatively poor ratings for InfoQual versus IntQual (seen across all of the studies – for details and complete normative data, see Lewis, 2002) suggests that practitioners who find this pattern in their data should not conclude that they have poor documentation or a great interface. Suppose, however, that this pattern appeared in the first iteration of a usability evaluation and the developers decided to emphasize improvement to the quality of their information. Any subsequent decline in the difference between InfoQual and IntQual would be evidence of a successful intervention.

Another potential criticism of the PSSUQ is that the items do not follow the typical convention of varying the tone of the items so that half of the items elicit agreement and the other half elicit disagreement. The rationale for the decision to consistently align the items was to make it as easy as possible for participants to complete the questionnaire. With consistent item alignment, the proper way to mark responses on the items is clearer, potentially reducing response errors due to participant confusion. Also, the use of negatively worded items can produce a number of undesirable effects (Barnette, 2000; Ibrahim, 2001), including problems with internal consistency and factor structure. The setting in which balancing the tone of the items is likely to be of greatest value is when participants do not have a high degree of motivation for providing reasonable and honest responses (for example, in clinical and educational settings). Obtaining reasonable and honest responses is rarely a problem in most usability testing settings.

Additional key findings and conclusions from Lewis (2002) were:

- There was no evidence of response styles (especially, no evidence of extreme response style) in the PSSUQ data.
- Because there is a possibility of extreme response and acquiescence response styles in cross-cultural research (Baumgartner and Steenkamp, 2001; Clarke, 2001; Grimm and Church, 1999, van de Vijver and Leung, 2001), practitioners should avoid using questionnaires for cross-cultural comparison unless that use has been validated. Other types of group comparisons with the PSSUQ are valid because any effect of response style should cancel out across experimental conditions.
- Scale scores from incomplete PSSUQs were indistinguishable from those computed from complete PSSUQs. These data do not provide information concerning how many items a participant might ignore and still produce reliable scale scores. They do suggest that, in practice, participants typically complete enough items to produce reliable scale scores.

The similarity of psychometric properties across the various versions of the PSSUQ, despite the passage of time and differences in the types of systems studied, provides evidence of significant generalizability for the questionnaire, supporting its use by practitioners for measuring participant satisfaction with the usability of tested systems. Due to its generalizability, practitioners can confidently use the PSSUQ when evaluating different types of products and at different times during the development process. The PSSUQ can be especially useful in competitive evaluations (for an example, see Lewis, 1996b) or when tracking changes in usability as a function of design changes made during development. Practitioners and researchers are free to use the PSSUQ and CSUQ (no license fees), but anyone using them should cite the source.

The ASQ

The ASQ (Lewis, 1991b, 1995) is an extremely short questionnaire (three 7-point scale items using the same format as the PSSUQ). The items address three important aspects of user satisfaction with system usability: ease of task completion (“Overall, I am satisfied with the ease of completing the tasks in this scenario.”), time to complete a task (“Overall, I am satisfied with the amount of time it took to complete the tasks in this scenario.”), and adequacy of support information (“Overall, I am satisfied with the support information (on-line help, messages, documentation) when completing tasks.”) The overall ASQ score is the average of responses to these three items.

Because the questionnaire is short, it takes very little time for participants to complete, an important practical consideration for usability studies. Measurements of ASQ reliability (using coefficient alpha) have ranged from .90 to .96 (Lewis, 1995). A significant correlation between ASQ scores and successful scenario completion ($r(46) = -.40, p < .01$) in Lewis et al. (1990, analysis reported in Lewis, 1995) provided evidence of concurrent validity. Like the PSSUQ and CSUQ, the ASQ is available for free use by practitioners and researchers, but anyone using the ASQ should cite the source.

WRAPPING UP

Getting More Information about Usability Testing

This chapter has provided fundamental and some advanced information about usability testing, but there is only so much that you can cover in a single chapter. For additional chapter-length treatments of the basics of usability testing, see Nielsen (1997) and Dumas (2003). There are also two well-known books devoted to the topic of usability testing.

Dumas and Redish (1999) is one of these book-length treatments of usability testing. The content and references are somewhat dated. The 1999 copyright date is a bit misleading, as the body of the book has not changed since its 1993 edition. The 1999 edition does include a new preface and some updated reading recommendations, and provides an excellent coverage of the fundamentals of usability testing.

The other well-known usability testing book is Rubin (1994). Like Dumas and Redish (1999), the content and references are ten years out of date. It, too, covers the fundamentals of usability testing (which haven't really changed for 20 years) very well and contains many useful samples of a variety of testing-related forms and documents.

For late-breaking developments in usability research and practice, there are a number of annual conferences that have usability evaluation as a significant portion of their content. Companies making a sincere effort in the professional development of their usability practitioners should ensure that their personnel have access to the proceedings of these conferences and should support attendance at one or more of these conferences at least every few years. These major conferences are:

- Usability Professionals Association (see <http://www.upassoc.org/>)
- Human-Computer Interaction International (see <http://www.hci-international.org/>)
- ACM Special Interest Group in Computer-Human Interaction (<http://www.acm.org/sigchi/>)
- Human Factors and Ergonomics Society (see <http://hfes.org/>)
- INTERACT (held every two years, for example, see <http://www.interact2005.org/>)

A Research Challenge: Improved Understanding of Usability Problem Detection

The recently published papers that have questioned the reliability of usability problem discovery (Hertzum and Jacobsen, 2003; Kessner et al., 2001; Molich et al., 1998, 2004) have raised a number of questions. Dumas (2003, p. 1112) responded, "It is not clear why there is so little overlap in problems. Are slight variations in method the cause? Are the problems really the same but just described differently? We look to further research to sort out the possibilities." Hertzum and Jacobsen (2003) noted the following as potential causes of lack of reliability across evaluations:

- Vague goal analyses that lead to variability in task scenarios
- Vague evaluation procedures
- Vague problem criteria that lead to acceptance of anything as a usability problem

Developing a better understanding of why these studies produced their results, which are so at odds with the apparent success of usability testing (Al-Awar et al., 1981; Bailey, 1993; Bailey et al., 1992; Gould et al., 1987; Kelley, 1984; Kennedy, 1982; Lewis, 1982; Lewis, 1996b; Marshall et al., 1990; Rutherford and Ramey, 2000), should be one of the top usability research efforts of the coming decade. An improved understanding might provide guidance about how or whether practitioners should change the way they conduct usability tests. One of the most important components of this research effort should be to investigate the cognitive mechanisms that underlie the detection of usability problems. Even after over 20 years of professional practice with usability methods, there is still no general consensus on the boundaries of what constitutes a usability problem, or on the appropriate levels of description of usability problems

(Lewis, 2001b). Doctoral candidates should take note – this is a topic rich with theoretical and practical consequences!

For example, it seems reasonable that the task of the observer in a usability study involves classical signal-detection issues (Massaro, 1975). The observer monitors participant behavior, and at any given moment must decide whether that observed behavior is indicative of a usability problem. Thus, there are two ways for an observer to make correct decisions (rejecting non-problem behaviors correctly, identifying problem behaviors correctly) and two ways to make incorrect decisions (identifying non-problem behaviors as indicative of a usability problem, failing to identify problem behaviors as indicative of a usability problem). In signal detection terms, the names for these right and wrong decisions are Correct Rejection, Hit, False Alarm, and Miss. The rates for these types of decisions depend independently on both the skill and the bias of the observer. Applying signal detection theory to the assessment of the skill and bias of usability test observers is a potentially rich, but to date untapped, area of research, with potential application for both selection and training of observers.

Usability Testing: Yesterday, Today, and Tomorrow

It seems clear that usability testing (both summative and formative) is here to stay, and that its general form will remain similar to the forms that emerged in the late 1970s and early 1980s. The last 25 years have seen the introduction of more usability evaluation techniques and some consensus (and some continuing debate) on the conditions under which to use the various techniques (of which usability testing is a major one). In the last 15 years, usability researchers have made significant progress in the areas of standardized usability questionnaires and sample size estimation for formative usability tests. As we look to the future, usability practitioners should monitor the continuing research that will almost certainly take place in developing a better understanding of the cognitive mechanisms of usability problem discovery because such an understanding has the potential to increase the reliability of usability testing.

In the mean time, practitioners will continue to perform usability tests, exercising professional judgment as required. Usability testing is not a perfect usability evaluation method in the sense that it does not guarantee the discovery of all possible usability problems, but it doesn't have to be perfect to be useful and effective. It is, however, important to understand its strengths, limitations, and current best practices to ensure its proper (most effective) use.

Acknowledgements

I want to thank Gavriel Salvendy for giving me the opportunity and encouragement to write this chapter. I also want to express my deepest appreciation to the colleagues who took the time to review the first draft under a very tight deadline – Patrick Commarford, Richard Cordes, Barbara Millet, Jeff Sauro, and Wallace Sadowski.

REFERENCES

- Aaker, D. A., and Day, G. S. (1986). *Marketing research*. New York, NY: John Wiley.
- Abelson, R. P. (1995). *Statistics as principled argument*. Mahwah, NJ: Lawrence Erlbaum.
- Al-Awar, J., Chapanis, A., and Ford, R. (1981). Tutorials for the first-time computer user. *IEEE Transactions on Professional Communication*, 24, 30-37.
- Alreck, P. L., and Settle, R. B. (1985). *The survey research handbook*. Homewood, IL: Richard D. Irwin, Inc.
- Alty, J. L. (1992). Can we measure usability? In *Proceedings of Advanced Information Systems* (pp. 95-106). London, UK: Learned Information.
- Anastasi, A. (1976). *Psychological testing*. New York, NY: Macmillan.
- Andre, T. S., Belz, S. M, McCreary, F. A., and Hartson, H. R. (2000). Testing a framework for reliable classification of usability problems . In *Proceedings of the IEA 2000/HFES 2000 Congress* (pp. 573-576). Santa Monica: CA: Human Factors and Ergonomics Society.
- ANSI. (2001). *Common industry format for usability test reports* (ANSI-NCITS 354-2001). Washington, DC: American National Standards Institute.
- Aykin, N. M., and Aykin, T. (1991). Individual differences in human-computer interaction. *Computers and Industrial Engineering*, 20, 373-379.
- Bailey, G. (1993). Iterative methodology and designer training in human-computer interface design. In *INTERCHI '93 Conference Proceedings* (pp. 198-205). New York, NY: ACM.
- Bailey, R. W., Allan, R. W., and Raiello, P. (1992). Usability testing vs. heuristic evaluation: A head to head comparison. In *Proceedings of the Human Factors and Ergonomics Society 36th Annual Meeting* (pp. 409-413). Atlanta, GA: Human Factors and Ergonomics Society.
- Banks, S. (1965). *Experimentation in marketing*. New York, NY: McGraw-Hill.
- Barnette, J. J. (2000). Effects of stem and Likert response option reversals on survey internal consistency: If you feel the need, there is a better alternative to using those negatively worded stems. *Educational and Psychological Measurement*, 60, 361-370.
- Baumgartner, H., and Steenkamp, J. B. E. M. (2001). Response styles in marketing research: A cross-national investigation. *Journal of Marketing Research*, 38, 143-156.
- Bennett, J. L. (1979). The commercial impact of usability in interactive systems. *Infotech State of the Art Report: Man/Computer Communication*, 2, 289-297.
- Berry, D. C., and Broadbent, D. E. (1990). The role of instruction and verbalization in improving performance on complex search tasks. *Behaviour & Information Technology*, 9, 175-190.
- Bevan, N., Kirakowski, J., and Maissel, J. (1991). What is usability? In H. J. Bullinger (Ed.), *Human Aspects in Computing, Design and Use of Interactive Systems and Work with Terminals, Proceedings of the Fourth International Conference on Human Computer Interaction* (pp. 651-655). Stuttgart, Germany: Elsevier Science Publishers.

- Bias, R. G., and Mayhew, D. J. (1994). *Cost-justifying usability*. Boston, MA: Academic Press.
- Blalock, H. M. (1972). *Social statistics*. New York, NY: McGraw-Hill.
- Boehm, B. W. (1981). *Software engineering economics*. Englewood Cliffs, NJ: Prentice-Hall.
- Boren, T., and Ramey, J. (2000). Thinking aloud: Reconciling theory and practice. *IEEE Transactions on Professional Communications*, 43, 261-278.
- Bowers, V., and Snyder, H. (1990). Concurrent versus retrospective verbal protocols for comparing window usability. In *Proceedings of the Human Factors Society 34th Annual Meeting* (pp. 1270-1274). Santa Monica, CA: Human Factors Society.
- Bradley, J. V. (1976). *Probability; decision; statistics*. Englewood Cliffs, NJ: Prentice-Hall.
- Brooke, J. (1996). SUS: A “quick and dirty” usability scale. In P. Jordan, B. Thomas, and B. Weerdmeester (Eds.), *Usability Evaluation in Industry* (pp. 189-194). London, UK: Taylor and Francis.
- Brown, F. E. (1980). *Marketing research: A structure for decision making*. Reading, MA: Addis on-Wesley.
- Campbell, D. T., and Stanley, J. C. (1963). *Experimental and quasi-experimental designs for research*. Chicago, IL: Rand McNally.
- Caulton, D.A. (2001). Relaxing the homogeneity assumption in usability testing. *Behaviour & Information Technology*, 20, 1-7.
- Chapanis, A. (1981). *Evaluating ease of use*. Unpublished manuscript prepared for IBM, available from J. R. Lewis.
- Chapanis, A. (1988). Some generalizations about generalization. *Human Factors*, 30, 253-267.
- Chin, J. P., Diehl, V. A., and Norman, K. L. (1988). Development of an instrument measuring user satisfaction of the human-computer interface. In E. Soloway, D. Frye, and S. B. Sheppard (Eds.), *CHI '88 Conference Proceedings: Human Factors in Computing Systems* (pp. 213-218). Washington, D.C.: ACM.
- Churchill, Jr., G. A. (1991). *Marketing research: Methodological foundations*. Ft. Worth, TX: Dryden Press.
- Clarke, I. (2001). Extreme response style in cross-cultural research. *International Marketing Review*, 18, 301-324.
- Cliff, N. (1987). *Analyzing multivariate data*. San Diego: CA: Harcourt Brace Jovanovich.
- Cockton, G., and Lavery, D. (1999). A framework for usability problem extraction. In *Human Computer Interaction INTERACT '99* (pp. 344-352). Amsterdam, Netherlands: IOS Press.
- Cordes, R. E. (1993). The effects of running fewer subjects on time-on-task measures. *International Journal of Human-Computer Interaction*, 5, 393-403.
- Cordes, R. E. (2001). Task-selection bias: A case for user-defined tasks. *International Journal of Human-Computer Interaction*, 13, 411-419.

- Cordes, R. E., and Lentz, J. L. (1986). *Usability-claims evaluation using a binomial test* (Tech. Report 82.0267). Tucson, AZ: International Business Machines Corp.
- Desurvire, H. W., Kondziela, J. M., and Atwood, M. E. (1992). What is gained and lost when using evaluation methods other than empirical testing. In A. Monk, D. Diaper, and M. D. Harrison (Eds.), *Proceedings of HCI '92: People and Computers VII* (pp. 89-102). York, UK: Cambridge University Press.
- Diamond, W. J. (1981). *Practical experiment designs for engineers and scientists*. Belmont, CA: Lifetime Learning Publications.
- Dickens, J. (1987). The fresh cream cakes market: The use of qualitative research as part of a consumer research programme. In U. Bradley (ed.), *Applied Marketing and Social Research* (pp. 23-68). New York, NY: John Wiley.
- Dumas, J. S. (2003). User-based evaluations. In J. A. Jacko and A. Sears (Eds.), *The Human-Computer Interaction Handbook* (pp. 1093-1117). Mahwah, NJ: Lawrence Erlbaum.
- Dumas, J., and Redish, J. C. (1999). *A practical guide to usability testing*. Portland, OR: Intellect.
- Dumas, J., Sorce, J., and Virzi, R. (1995). *Expert reviews: How many experts is enough?* In Proceedings of the Human Factors and Ergonomics Society 39th Annual Meeting (pp. 228-232). Santa Monica, CA: Human Factors and Ergonomics Society.
- Ericsson, K. A., and Simon, H. A. (1980). Verbal reports as data. *Psychological Review*, 87, 215-251.
- Faulkner, L. (2003). Beyond the five-user assumption: Benefits of increased sample sizes in usability testing. *Behavior Research Methods, Instruments, & Computers*, 35, 379-383.
- Fisher, J. (1991). Defining the novice user. *Behaviour & Information Technology*, 10, 437-441.
- Fowler, C. J. H., Macaulay, L. A., and Fowler, J. F. (1985). The relationship between cognitive style and dialogue style: an exploratory study. In P. Johnson and S. Cook (eds.), *People and Computers: Designing the Interface* (pp. 186-198). Cambridge, UK: Cambridge University Press.
- Fu, L., Salvendy, G., and Turley, L. (2002). Effectiveness of user testing and heuristic evaluation as a function of performance classification. *Behaviour & Information Technology*, 21, 137-143.
- Gawron, V. J., Drury, C. G., Czaja, S. J., and Wilkins, D. M. (1989). A taxonomy of independent variables affecting human performance. *International Journal of Man-Machine Studies*, 31, 643-672.
- Gordon, W., and Langmaid, R. (1988). Qualitative market research: A practitioner's and buyer's guide. Aldershot, UK: Gower.
- Gould, J. D. (1988). How to design usable systems . In M. Helander (Ed.), *Handbook of Human-Computer Interaction* (pp. 757-789). Amsterdam, Netherlands: North-Holland.
- Gould, J. D., and Boies, S. J. (1983). Human factors challenges in creating a principal support office system – the speech filing system approach. *ACM Transactions on Information Systems*, 1, 273-298.
- Gould, J. D., and Lewis, C. (1984). *Designing for usability: Key principles and what designers think* (Tech. Report RC-10317). Yorktown Heights, NY: International Business Machines Corp.

- Gould, J. D., Boies, S. J., Levy, S., Richards, J. T., and Schoonard, J. (1987). The 1984 Olympic message system: A test of behavioral principles of system design. *Communications of the ACM*, 30, 758-769.
- GraphPad. (2004). Confidence interval of a proportion or count. Available at <http://graphpad.com/quickcalcs/ConfInterval1.cfm>.
- Gray, W. D., and Salzman, M. C. (1998). Damaged merchandise? A review of experiments that compare usability evaluation methods. *Human-Computer Interaction*, 13, 203-261.
- Greene, S. L., Gomez, L. M., and Devlin, S. J. (1986). A cognitive analysis of database query production. In *Proceedings of the 30th Annual Meeting of the Human Factors Society* (pp. 9-13). Santa Monica: CA: Human Factors Society.
- Grimm, S. D., and Church, A. T. (1999). A cross-cultural study of response biases in personality measures. *Journal of Research in Personality*, 33, 415-441.
- Hackman, G. S., and Biers, D. W. (1992). Team usability testing: Are two heads better than one? In *Proceedings of the Human Factors and Ergonomics Society 36th Annual Meeting* (pp. 1205-1209). Atlanta, GA: Human Factors and Ergonomics Society.
- Harris, R. J. (1985). *A primer of multivariate statistics*. Orlando, FL: Academic Press.
- Hassenzahl, M. (2000). Prioritizing usability problems: data driven and judgment driven severity estimates. *Behaviour & Information Technology*, 19, 29-42.
- Hertzum, M., and Jacobsen, N. J. (2003). The evaluator effect: A chilling fact about usability evaluation methods. *International Journal of Human-Computer Interaction*, 15, 183-204.
- Holleran, P. A. (1991). A methodological note on pitfalls in usability testing. *Behaviour & Information Technology*, 10, 345-357.
- Ibrahim, A. M. (2001). Differential responding to positive and negative items: The case of a negative item in a questionnaire for course and faculty evaluation. *Psychological Reports*, 88, 497-500.
- ISO. (1998). *Ergonomic requirements for office work with visual display terminals (VDTs) – Part 11: Guidance on usability* (ISO 9241-11:1998(E)). Geneva, Switzerland: Author.
- Jeffries, R., and Desurvire, H. (1992). Usability testing vs. heuristic evaluation: Was there a contest? *SIGCHI Bulletin*, 24, 39-41.
- Kantner, L. and Rosenbaum, S. (1997). Usability studies of WWW sites: Heuristic evaluation vs. laboratory testing." In *Proceedings of SIGDOC 1997* (pp. 153-160). Salt Lake City, UT: ACM.
- Karat, C. (1997). Cost-justifying usability engineering in the software life cycle. In M. Helander, T. K. Landauer, and P. Prabhu (Eds.), *Handbook of Human-Computer Interaction* (pp. 767-778). Amsterdam, Netherlands: Elsevier.
- Karat, J. (1997). User-centered software evaluation methodologies. In M. Helander, T. K. Landauer, and P. Prabhu (Eds.), *Handbook of Human-Computer Interaction* (pp. 689-704). Amsterdam, Netherlands: Elsevier.
- Keenan, S.L., Hartson, H.R., Kafura, D.G., and Schulman, R.S. (1999). The Usability Problem Taxonomy: A framework for classification and analysis . *Empirical Software Engineering*, 1, 71-104.

- Kelley, J. F. (1984). An iterative design methodology for user-friendly natural language office information applications. *ACM Transactions on Information Systems*, 2, 26-41.
- Kelley, J. F. (1985). CAL - A natural language program developed with the OZ Paradigm: Implications for supercomputing systems. In the *First International Conference on Supercomputing Systems* (pp. 238-248). New York: ACM.
- Kennedy, P. J. (1982). Development and testing of the operator training package for a small computer system. In *Proceedings of the Human Factors Society 26th Annual Meeting* (pp. 715-717). Santa Monica, CA: Human Factors Society.
- Kessner, M., Wood, J., Dillon, R. F., and West, R. L. (2001). On the reliability of usability testing. In Jacko, J. and Sears, A., Eds., *Conference on Human Factors in Computing Systems: CHI 2001 Extended Abstracts* (pp. 97-98). Seattle, WA: ACM Press.
- Kirakowski, J. (1996). The Software Usability Measurement Inventory: Background and usage. In P. Jordan, B. Thomas, and B. Weerdmeester (Eds.), *Usability Evaluation in Industry* (pp. 169-178). London, UK: Taylor and Francis. (Also, see <http://www.ucc.ie/hfrg/questionnaires/sumi/index.html>.)
- Kirakowski, J., and Corbett, M. (1993). SUMI: The Software Usability Measurement Inventory. *British Journal of Educational Technology*, 24, 210-212.
- Kirakowski, J., and Dillon, A. (1988). *The computer user satisfaction inventory (CUSI): Manual and scoring key*. Cork, Ireland: Human Factors Research Group, University College of Cork.
- Kraemer, H. C., and Thiemann, S. (1987). *How many subjects? Statistical power analysis in research*. Newbury Park, CA: Sage.
- LaLomia, M. J., and Sidowski, J. B. (1990). Measurements of computer satisfaction, literacy, and aptitudes: A review. *International Journal of Human-Computer Interaction*, 2, 231-253.
- Landauer, T. K. (1997). Behavioral research methods in human-computer interaction. In M. Helander, T. K. Landauer, and P. Prabhu (Eds.), *Handbook of Human-Computer Interaction* (pp. 203-227). Amsterdam, Netherlands: North Holland.
- Lewis, C., and Norman, D. (1986). Designing for error. In D. A. Norman and S. W. Draper (Eds.), *User Centered System Design: New Perspectives on Human-Computer Interaction* (pp. 411-432). Hillsdale, NJ: Lawrence Erlbaum.
- Lewis, J. R. (1982). Testing small system customer set-up. In *Proceedings of the Human Factors Society 26th Annual Meeting* (pp. 718-720). Santa Monica, CA: Human Factors Society.
- Lewis, J. R. (1991a). A rank-based method for the usability comparison of competing products. In *Proceedings of the Human Factors Society 35th Annual Meeting* (pp. 1312-1316). San Francisco, CA: Human Factors Society.
- Lewis, J. R. (1991b). Psychometric evaluation of an after-scenario questionnaire for computer usability studies: The ASQ. *SIGCHI Bulletin*, 23, 78-81.
- Lewis, J. R. (1992). Psychometric evaluation of the Post-Study System Usability Questionnaire: The PSSUQ. In *Proceedings of the Human Factors Society 36th Annual Meeting* (pp. 1259-1263). Atlanta, GA: Human Factors Society.

- Lewis, J. R. (1993). Multipoint scales: Mean and median differences and observed significance levels. *International Journal of Human-Computer Interaction*, 5, 382-392.
- Lewis, J.R. (1994). Sample sizes for usability studies: Additional considerations. *Human Factors*, 36, 368-378.
- Lewis, J. R. (1995). IBM computer usability satisfaction questionnaires: Psychometric evaluation and instructions for use. *International Journal of Human-Computer Interaction*, 7, 57-78.
- Lewis, J. R. (1996a). Binomial confidence intervals for small sample usability studies. In G. Salvendy and A. Ozok (eds.), *Advances in Applied Ergonomics: Proceedings of the 1st International Conference on Applied Ergonomics -- ICAE '96* (pp. 732-737). Istanbul, Turkey: USA Publishing.
- Lewis, J. R. (1996b). Reaping the benefits of modern usability evaluation: The Simon story. In G. Salvendy and A. Ozok (eds.), *Advances in Applied Ergonomics: Proceedings of the 1st International Conference on Applied Ergonomics -- ICAE '96* (pp. 752-757). Istanbul, Turkey: USA Publishing.
- Lewis, J. R. (2001a). Evaluation of procedures for adjusting problem-discovery rates estimated from small samples. *International Journal of Human-Computer Interaction*, 13, 445-479.
- Lewis, J. R. (2001b). Introduction: Current issues in usability evaluation. *International Journal of Human-Computer Interaction*, 13, 343-349.
- Lewis, J. R. (2002). Psychometric evaluation of the PSSUQ using data from five years of usability studies. *International Journal of Human-Computer Interaction*, 14, 463-488.
- Lewis, J. R., Henry, S. C., and Mack, R. L. (1990). Integrated office software benchmarks: A case study. In D. Diaper et al. (Eds.), *Human-Computer Interaction - INTERACT '90, Proceedings of the Third IFIP Conference on Human-Computer Interaction* (pp. 337-343). Cambridge, England: Elsevier Science Publishers.
- Lewis, J. R., and Pallo, S. (1991). *Evaluation of graphic symbols for phone and line* (Tech. Report 54.572). Boca Raton, FL: International Business Machines Corp.
- Lilienfeld, S. O., Wood, J. M., and Garb, H. N. (2000). The scientific status of projective techniques. *Psychological Science in the Public Interest*, 1, 27-66.
- Lord, F. M. (1953). On the statistical treatment of football numbers. *American Psychologist*, 8, 750-751.
- Macleod, M., Bowden, R., Bevan, N., and Curson, I. (1997). The MUSiC performance measurement method. *Behaviour & Information Technology*, 16, 279-293.
- Marshall, C., Brendan, M., and Prail, A. (1990). Usability of product X – lessons from a real product. *Behaviour & Information Technology*, 9, 243-253.
- Massaro, D. W. (1975). *Experimental psychology and information processing*. Chicago, IL: Rand McNally.
- Mayer, R. E. (1997). From novice to expert. In M. G. Helander, T. K. Landauer, and P. V. Prabhu (eds.), *Handbook of Human-Computer Interaction* (pp. 781-795). Amsterdam: Elsevier.
- McFadden, E., Hager, D. R., Elie, C. J., and Blackwell, J. M. (2002). Remote usability evaluation: overview and case studies. *International Journal of Human Computer Interaction*, 14, 489-502.

- McSweeney, R. (1992). *SUMI -- A psychometric approach to software evaluation*. Unpublished MA (Qual) thesis in Applied Psychology, University College Cork, Ireland.
- Miller, L. A., Stanney, K. M., and Wooten, W. (1997). Development and evaluation of the Windows Computer Experience Questionnaire (WCEQ). *International Journal of Human-Computer Interaction*, 9, 201-212.
- Moffat, B. (1990). Normalized performance ratio -- a measure of the degree to which a man-machine interface accomplishes its operational objective. *International Journal of Man-Machine Studies*, 32, 21-108.
- Molich, R., Bevan, N., Curson, I., Butler, S., Kindlund, E., Miller, D., and Kirakowski, J. (1998). Comparative evaluation of usability tests. In *Usability Professionals Association Annual Conference Proceedings* (pp. 189-200). Washington, DC: Usability Professionals Association.
- Molich, R., Ede, M. R., Kaasgaard, K., and Karyukin, B. (2004). Comparative usability evaluation. *Behaviour & Information Technology*, 23, 65-74.
- Morse, E. L. (2000). The IUSR project and the Common Industry Reporting Format. In *Proceedings of the Conference on Universal Usability* (pp. 155-156). Arlington, VA: ACM.
- Myers, J. L. (1979). *Fundamentals of experimental design*. Boston, MA: Allyn and Bacon.
- Nielsen, J. (1992). Finding usability problems through heuristic evaluation. In *CHI '92 Conference Proceedings* (pp. 373-380). Monterey, CA: ACM.
- Nielsen, J. (1993). *Usability engineering*. San Diego, CA: Academic Press.
- Nielsen, J. (1994). Usability laboratories. *Behaviour and Information Technology*, 13, 3-8.
- Nielsen, J. (1997). Usability testing. In G. Salvendy (Ed.), *Handbook of Human Factors and Ergonomics*. New York, NY: John Wiley.
- Nielsen, J., and Landauer, T.K. (1993). A mathematical model of the finding of usability problems. In *Proceedings of ACM INTERCHI'93 Conference* (pp. 206-213). Amsterdam, Netherlands: ACM Press.
- Nielsen, J., and Molich, R. (1990). Heuristic evaluation of user interfaces. In *Conference Proceedings on Human Factors in Computing Systems – CHI90* (pp. 249-256). New York, NY: ACM.
- Nisbett, R. E., and Wilson, T. D. (1977). Telling more than we can know: verbal reports on mental processes. *Psychological Review*, 84, 231-259.
- Norman, D. A. (1983). Design rules based on analyses of human error. *Communications of the ACM*, 26, 254-258.
- Norman, D. (1986). Cognitive engineering. In D. A. Norman and S. W. Draper (Eds.), *User centered system design: New Perspectives on human-computer interaction* (pp. 31-61). Hillsdale, NJ: Erlbaum.
- Nunnally, J.C. (1978). *Psychometric theory*. New York, NY: McGraw-Hill.
- Palmquist, R. A., and Kim, K. S. (2000). Cognitive style and on-line database search experience as predictors of web search performance. *Journal of the American Society for Information Science*, 51, 558-566.

- Parasuraman, A. (1986). Nonprobability sampling methods. In *Marketing Research* (pp. 498-516). Reading, MA: Addison-Wesley.
- Prümper, J., Zapf, D., Brodbeck, F. C., and Frese, M. (1992). Some surprising differences between novice and expert errors in computerized office work. *Behaviour & Information Technology*, 11, 319-328.
- Rasmussen, J. (1986). *Information processing and human-machine interaction: An approach to Cognitive Engineering*. New York, NY: Elsevier.
- Renger, R. (1991). Indicators of usability based on performance. In H. J. Bullinger (Ed.), *Human Aspects in Computing, Design and Use of Interactive Systems and Work with Terminals, Proceedings of the Fourth International Conference on Human Computer Interaction* (pp. 656-660). Stuttgart, Germany: Elsevier Science Publishers.
- Rosenbaum, S. (1989). Usability evaluations versus usability testing: When and why? *IEEE Transactions on Professional Communication*, 32, 210-216.
- Rubin, J. (1994). *Handbook of usability testing: How to plan, design, and conduct effective tests*. New York, NY: John Wiley.
- Rutherford, M. A., and Ramey, J. A. (2000). Design response to usability test findings: A case study based on artifacts and interviews. *IEEE International Professional Communication Conference* (pp. 315-323). Piscataway, NJ: IEEE.
- Sadowski, W. J. (2001). Capabilities and limitations of Wizard of Oz evaluations of speech user interfaces. In *Proceedings of HCI International 2001: Usability Evaluation and Interface Design* (pp. 139-143). Mahwah, NJ: Lawrence Erlbaum.
- Sauro, J. (2004). *Restoring confidence in usability results*. Available at http://www.measuringusability.com/conf_intervals.htm
- Shackel, B. (1990). Human factors and usability. In J. Preece and L. Keller (Eds.), *Human-Computer Interaction, Selected Readings* (pp. 27-41). Hemel Hempstead, UK: Prentice Hall International.
- Shneiderman, B. (1987). *Designing the user interface: Strategies for effective human-computer interaction*. Reading, MA: Addison-Wesley.
- Smith, B., Caputi, P., Crittenden, N., Jayasuriya, R., and Rawstorne, P. (1999). A review of the construct of computer experience. *Computers in Human Behavior*, 15, 227-242.
- Smith, D. C., Irby, C., Kimball, R., Verplank, B., and Harlem, E. (1982). Designing the Star user interface. *Byte*, 7(4), 242-282.
- SOGSC (Southwest Oncology Group Statistical Center). (2004). *Binomial confidence interval*. Available at http://www.swogstat.org/stat/public/binomial_conf.htm
- Steele, R. G. D., and Torrie, J. H. (1960). *Principles and procedures of statistics*. New York, NY: McGraw-Hill.
- Stevens, S. S. (1951). Mathematics, measurement, and psychophysics. In S. S. Stevens (Ed.), *Handbook of Experimental Psychology* (pp. 1-49). New York: John Wiley.
- Sudman, S. (1976). *Applied sampling*. New York: NY: Academic Press.

- Turner, C. W., Lewis, J. R., and Nielsen, J. (2006). Determining usability test sample size. In W. Karwowski (Ed.), *International Encyclopedia of Ergonomics and Human Factors* (pp. 3084-3088). Boca Raton, FL: CRC Press.
- van de Vijver, F. J. R., and Leung, K. (2001). Personality in cultural context: Methodological issues. *Journal of Personality*, 69, 1007-1031.
- Virzi, R. A. (1990). Streamlining the design process: Running fewer subjects. In *Proceedings of the Human Factors Society 34th Annual Meeting* (pp. 291-294). Santa Monica, CA: Human Factors Society.
- Virzi, R.A. (1992). Refining the test phase of usability evaluation: how many subjects is enough? *Human Factors*, 34, 457-468.
- Virzi, R. A. (1997). Usability inspection methods. In M. G. Helander, T. K. Landauer, and P. V. Prabhu (eds.), *Handbook of Human-Computer Interaction* (pp. 705-715). Amsterdam: Elsevier.
- Virzi, R. A., Sorce, J. F., and Herbert, L. B. (1993). A comparison of three usability evaluation methods: Heuristic, think-aloud, and performance testing. In *Proceedings of the Human Factors and Ergonomics Society 37th Annual Meeting* (pp. 309-313). Santa Monica, CA: Human Factors and Ergonomics Society.
- Vredenburg, K., Mao, J. Y., Smith, P. W., and Carey, T. (2002). A survey of user centered design practice. In *Proceedings of CHI 2002* (pp. 471-478). Minneapolis, MN: ACM.
- Walpole, R. E. (1976). *Elementary statistical concepts*. New York, NY: Macmillan.
- Wiethoff, M., Arnold, A., and Houwing, E. (1992). *Measures of cognitive workload* (MUSiC ESPRIT Project 5429 document code TUD/M3/TD/2).
- Wenger, M. J., and Spyridakis, J. H. (1989). The relevance of reliability and validity to usability testing. *IEEE Transactions on Professional Communication*, 32, 265-271.
- Whiteside, J., Bennett, J., and Holtzblatt, K. (1988). Usability engineering: Our experience and evolution. In M. Helander (Ed.), *Handbook of Human-Computer Interaction* (pp. 791-817). Amsterdam: North-Holland.
- Wickens, C. D. (1998). Commonsense statistics. *Ergonomics in Design*, 6(4), 18-22.
- Wildman, D. (1995). Getting the most from paired-user testing. *interactions*, 2(3), 21-27.
- Williams, G. (1983). The Lisa computer system. *Byte*, 8(2), 33-50.
- Wixon, D. (2003). Evaluating usability methods: Why the current literature fails the practitioner. *interactions*, 10(4), 28-34.
- Woolrych, A., and Cockton, G. (2001). Why and when five test users aren't enough. In Vanderdonckt, J., Blandford, A. and Derycke A., (Eds.), *Proceedings of IHM-HCI 2001 Conference, Vol. 2* (pp. 105-108). Toulouse, France: Cépadès Éditions.
- Wright, R. B., and Converse, S. A. (1992). Method bias and concurrent verbal protocol in software usability testing. In *Proceedings of the Human Factors and Ergonomics Society 36th Annual Meeting* (pp. 1220-1224). Atlanta, GA: Human Factors and Ergonomics Society.

Wright, P. C., and Monk, A. F. (1991). A cost-effective evaluation method for use by designers.
International Journal of Man-Machine Studies, 35, 891-912.

IDI – Usability use case – Depth perception in VR

Pere-Pau Vázquez – Dep. Computer Science, UPC

Contents

1 Introduction.....	1
1.1 Context.....	1
1.2 Goals of the study.....	1
2 Experiment setup.....	2
2.1 Shading techniques to compare	2
2.2 Methods and apparatus	3
2.3 Test preparation	5
2.4 Physical setup	6
2.5 Design and procedure	8
3. Implementation.....	8
3.1 Test performance	8
3.2 Data analysis.....	9
3.3 Results	9
References.....	12

1 Introduction

In these notes we will present a usability evaluation use case, more concretely, an experiment tailored to *measure* different performance aspects of depth perception in a virtual reality environment. As a consequence, the experiment was implemented as a **measurement-based usability testing**.

1.1 Context

Volume Rendering is a popular rendering technique that can be applied to many fields and types of data, such as medical visualization or flow rendering. The main feature of volume rendering comes from the fact that the data to visualize consists on a whole volume instead of having the geometry stored as a set of planar faces. The reason behind that is that we want to inspect what is *inside* the models, being able to see the whole volume, as in Figure 1, where a body that has been captured using a Computational Tomography (CT) is inspected using direct volume rendering.



Figure 1: Volume rendering of an anatomic model using the emission-absorption model with Phong Shading.

Volumetric rendering techniques are very important because they facilitate a type of exploration that cannot be done using other techniques. In order to produce such effects, it is necessary to strongly use semi-transparency and this sometimes limits the perception of depth: the ability of a person to classify the different distances at which the objects are placed in the real 3D data.

1.2 Goals of the study

The purpose of the study is to analyse whether the shading technique can influence the depth perception of volumetric models. More concretely, we want to evaluate if any of the

advanced rendering techniques that have been developed recently, produces better images in terms of depth perception in Virtual Reality environments [Grosset13].

The perception of depth in our brain depends basically on the fact that, since our eyes are separated by a certain distance, when we have both of them open, we see two slightly different images. In stereoscopic environments, we simulate this effect by generating two different images, one for each eye, and project them using some technique such as immersive systems (e.g. head mounted displays), or semi-immersive systems such as CAVEs, PowerWall, 3D TVs, etc.

2 Experiment setup

As we have already seen, the goal of a user study is to determine the response to certain stimuli. Therefore, we must set up the conditions such that we can make sure we are measuring/calculated what we actually intend to. Thus, we have to dive a bit into the human perception system to ensure that we know how to determine the influence of the shading technique in the ability of perceiving depth from other, confusion variables.

2.1 Shading techniques to compare

Many new advanced shading techniques have been developed rendering for volumetric models [Hernell09, Kniss03, Langer00, Schott09]. Testing all would make the experiment too long, and very difficult to evaluate: we cannot make the users test 20 different shading conditions for many models, since this would create obvious fatigue. Perception experiments should be carried out for a relatively low limited time, because the attention and vigilance capabilities of the human brain are quite limited.

As a result, we decided to test 4 conditions: two basic techniques, no shading and Phong shading, and two advanced shading methods, half-angle slicing and directional occlusion shading. Figure 2 shows an example of synthetic model with the four shading conditions. Note that some of the basic shading techniques seem to produce little or no depth cues, and therefore, one might think that the depth perception will likely be hampered. However, we do not know if the use of two images, one for each eye, will compensate. On the other hand, the advanced techniques presented in the two bottom images yield visual cues that encode the relative positions of the objects. However, we do not know if such shadows will make the users to perceive darker regions as further than they really are. All of this can only be determined by making a formal user study where the answers of the participants are gathered in such a way that there is no ambiguity on what they are answering. This does not mean that the users will all of them perceive all the scenes equally. We know that perception is variable among human beings. However, previous studies have demonstrated that low-level perception tasks, such as color identification or depth perception is very similar among persons with correct or corrected vision.

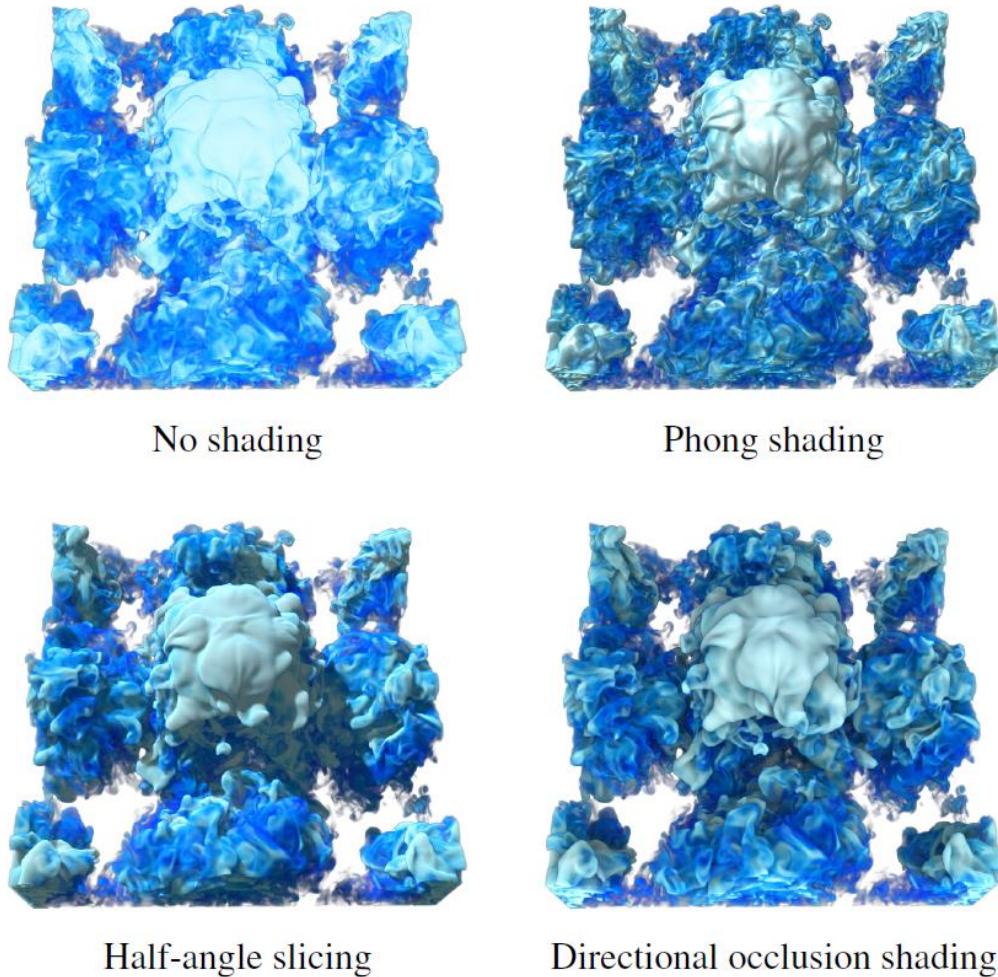


Figure 2: Volume rendering of a synthetic model using the four shading techniques: *no shading*, *Phong shading*, *directional occlusion shading*, and *half-angle slicing*. Some of them, especially the ones on the top, produce more *planar* images where depth seems more difficult to perceive. However, the shadows on the bottom images may sometimes mislead perception.

2.2 Methods and apparatus

There are many aspects that influence 3D perception. For example, the movement of our head, that causes the scene to be seen under different angles, causes parallax effects (small variations of what we see that depend on the distance and position of the point of view, that is, the elements in the background move slower than elements in the foreground). This is a very important factor in the perception of 3D, besides the two different images that we see with each eye.

As a result, if we set up an experiment in which the users can move freely around the scene, it will be very difficult if not impossible, to separate the parallax effect in the 3D perception from the variables we want to test: illumination algorithms. This has to be taken into account when performing a measuring experiment.

As a result, we will consider a stereoscopic environment where the users cannot move freely, so we will use a 3D TV with a fixed position for the participants. They will be seated at a fixed distance and no head tracking will be used to avoid parallax effects influence the results.

The tasks we will ask the users will consist on classifying two points, selecting the one that appears to be closer to the observer. These points will be randomly selected in the scene, but all the points will be the same for different users, so that we have data to compare with. Moreover, we will ensure that the distribution of the real depths of points will be also relatively uniform along the depth values of the scene: some point pairs will have similar distances, some of them will have medium distances, and some of them will have great depth separation. Also, for the points, some of the closer points will be at the left part of the image or the right. Moreover, the horizontal separation of the points will also vary quite uniformly, to test whether the users distinguish better points that are closer from points that are far away from each other. Generating such a set of points is quite time consuming. Testing absolutely random points per each user makes the statistical analysis more difficult, or even impossible unless we have thousands of samples, which is very complicated, due to the fact that the test must be carried out in a controlled environment.

Concerning the datasets, we need to provide examples where the user cannot get an idea of the relative position of the points just because the 3D models are known to the user. For example, we cannot use the model in Figure 3 and ask the users to classify points denoted by the blue circle and square because any participant will know that a point in our lip will be closer than a point in the neck.

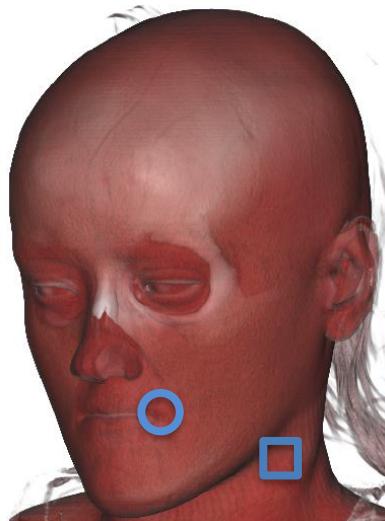


Figure 3: Volume model of a geometry the user will know and that would yield misleading results: the user can answer properly independently of the shading function used.

As a result, we will use either models that are not known by the vast majority of the potential participants, and in case of doubt, such as the *backpack* model, that is used in volume rendering research, we will generate view positions that make classification of points according to aspects that are not just the shading, more difficult (see Figure 4).

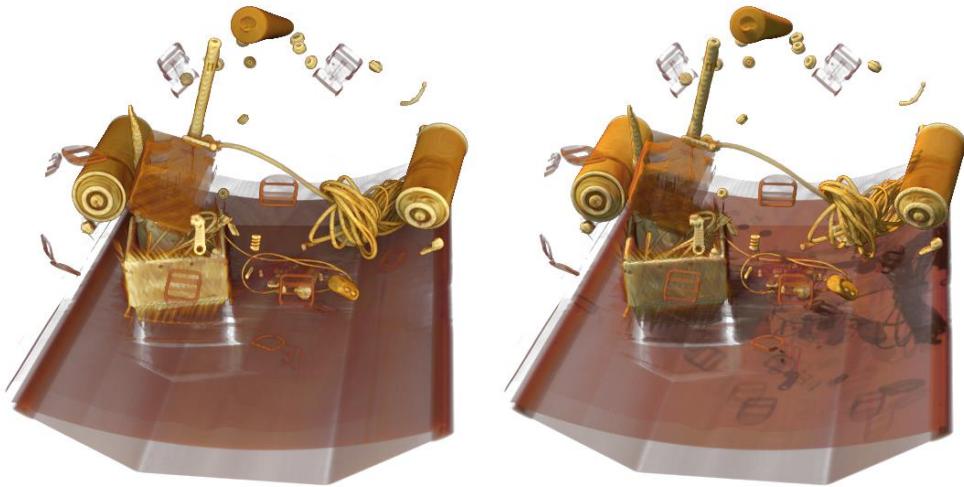


Figure 4: Volume rendering of the *backpack* model using two different shading techniques: *directional occlusion shading* (left) and *half-angle slicing* (right), at viewpoints that make very difficult to relatively place the objects with respect to the viewer.

2.3 Test preparation

Once we have decided the number of shading methods (4), and that we are going to use models that are unknown, we need to select the number of participants. This will depend on the nature of the problem we deal with. Since low-level perception is not very variable among humans, we need a sample group that is larger than 10 to have significant results.

Since we have 4 shading conditions, and we need enough data for the statistical analysis, we used 6 different modes in the study. These were a mixture of CT models and synthetic data sets. In total, users were exposed to six different models, half of them were CTs and the other half were synthetic. The generated stereo images are representative of volume visualization, since they present cluttered regions with complex shapes and combine semi-transparent with opaque layers. Note that, as commented previously, we used specific data sets and added arbitrary rotations to them to reduce the possibility that a previous knowledge of the data or too evident shapes facilitate the recognition of depths aside from the proper perception. We also avoided models that represent well-known parts of the human body, for example.

The participants were exposed to the different models where the images were systematically sorted based on the shading technique using Latin squares: We have 4 shading conditions and 6 models, so we can use a 4x4 Latin squares model. The 18 images of each technique (variations of models and views) can then be presented randomly, to avoid learning and fatigue effects. We need therefore a number of participants that can be divided by 4, to repeat all the Latin squares conditions the same number of times. That is why we used 16 participants for each task. In our case, we changed the participants for each task, to avoid learning effects. But due to the nature of the tasks, this would not have been necessary.

To be able to perform a proper data analysis, we will capture the correctness to the answers, as well as the time devoted to answer. This was necessary because it will be interesting to analyze whether a higher amount of correct answers can be due to a larger amount of time devoted to solve the questions. Note that in this case this implies to modify the software for the experiment, since accurate data of time answering can only be obtained by measuring the time between clicks.

2.4 Physical setup

For the experiment, we used a passive stereo system consisting on a 46" JVC 3D TV (GD-463D10 model) with polarized glasses. Moreover, we added a lamp placed top-left and some objects around the TV (two balls, a plastic glass and a couple of books) so that the lamp generates shadows of different sizes inside the participants' Field of Regard (at least the peripheral vision was aware of such shadows, see Figure 5). The lamp position was chosen to be coherent with the virtual light source that is used in half-angle slicing and Phong shading (the position of the virtual light source is changed in task 3 to make it non-coherent with the lamp position). Users sat during the study (2 m from the screen) to avoid movements and thus, limiting the impact of the perspective distortion.

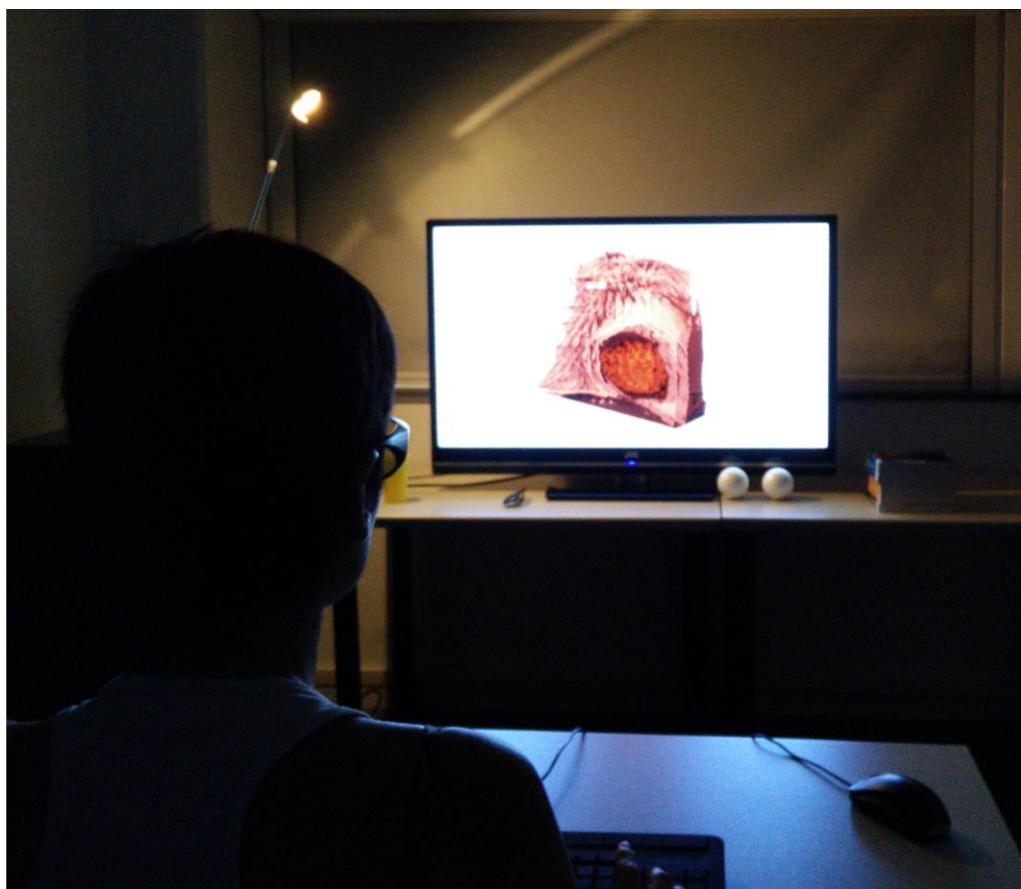


Figure 5: Physical setup.

The application shows static stereoscopic images to avoid depth inferring via other elements such as the model motion. We created a small application that shows two markers in each image, and lets the users select the one which is placed closer to the observer. The markers are designed as small windows with two shapes: circular and square. They indicate the points of the model to be classified by the users. The markers are placed at the same distance from the observer, and users were instructed to classify the point that the markers show, not the markers themselves. These markers pop up three times, and then disappear. If necessary, the users may request the application to show the markers again. In Figure 6 we show one of the used images.

The selection falls into the category of two-alternative forced choice, which means that the users have only two options and cannot answer “do not know”. This eases the statistical analysis, but may have some limitations if our goal is also to measure how sure or unsure a participant is of the given answer.

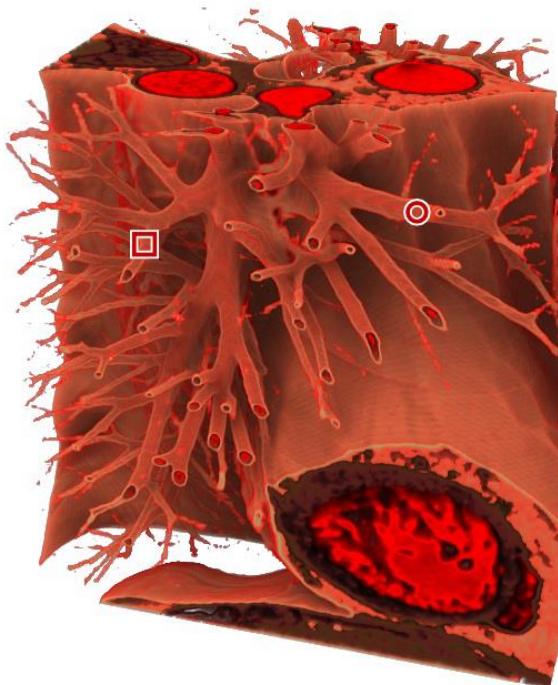


Figure 6: Users were shown an image with two markers that identify two points in the model that must be classified. The markers disappear after popping up for three times and the users must determine which of the points is closer.

In order to facilitate the tasks, we customized a keyboard by painting all the keys in black and putting some stickers to mark the necessary keys, as shown in Figure 7.

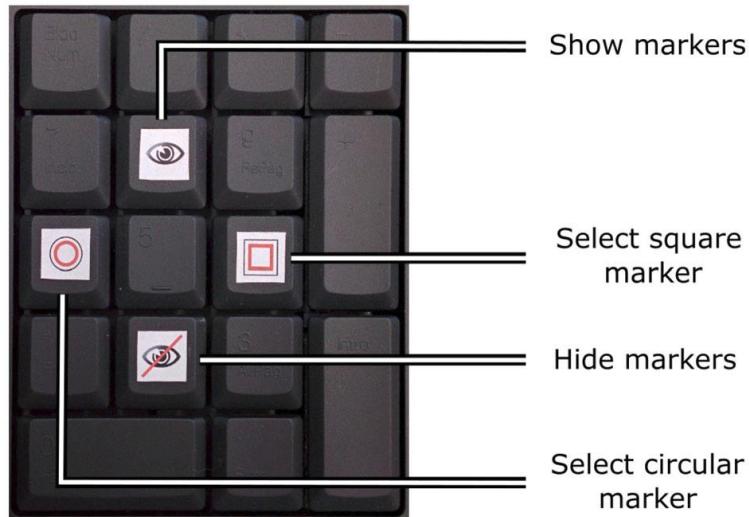


Figure 7: The modified keyboard to facilitate the selection. By painting the keys in black, we eliminate distractions, and the white labels facilitate the identification of the proper keys during the experiment, since the light is highly dimmed.

2.5 Design and procedure

The study comprises three different tasks. The first one evaluates the perception of depth using different shading techniques. Then, the second task evaluates the same methods in the presence of a controlled external light source (we use a lamp that casts obvious shadows inside the field of regard of the user). This task is tailored to determine if real illumination has any influence on the responses. Apart from the presence of the lamp, the experiment and the setup in both tasks are the same. Finally, the third task evaluates the influence on depth perception of having coherent real and virtual light directions. This task was meant to be done for those shading models that allow to change the virtual light position, only if the analysis of the previous tasks showed a real influence of external illumination on depth perception.

3. Implementation

3.1 Test performance

Participants were shown 72 images covering three different points of view, four different techniques, and six different models. Since the variable to analyze was the shading model, the images were systematically sorted based on the shading technique using Latin squares, and the 18 images of each technique (variations of models and views) were then presented randomly. The users were introduced the task with a first example that had to be solved before the experiment effectively started. Everybody showed a proper understanding of the procedure and all the users completed their tasks to their satisfaction (as proved by post hoc questionnaires). After each choice, the users had a neutral screen to let them recover from fatigue if necessary. Participants were instructed to determine the closer point from a pair indicated by two markers and asked to take as much time as necessary. Markers

popped up three times before the actual selection might start. In case of necessity, the users could make the markers visible again. The test was previously checked with two extra users whose results were not included in the data analysis. Throughout the experiment, we measured the correctness of the answers and the time spent in each choice.

They had to fill in two different questionnaires: one before the task with personal information (age, gender, quality of eyesight, experience with VREs, etc.) and another after the task, asking about a subjective evaluation of the performed activity. After analyzing the post hoc questionnaires, we found that all of them understood their task properly and found it easy to achieve. Therefore, there was no need of discarding any user.

3.2 Data analysis

The mean correctness of the answers for each task was analyzed by using a one-way analysis of variance (ANOVA) with a significance level of $\alpha = 0.05$. When significant differences between the means were found, we used a post hoc Bonferroni's pairwise test with the same significance level ($\alpha = 0.05$). The same analysis was performed with the means of the time spent by the users to complete each task.

To test the linear correlation between the measured variables, we used the Pearson's r statistic and assessed the linear model testing the regression coefficient β_1 with $\alpha = 0.05$. Finally, the Chi-square test of association with a significance level of $\alpha = 0.05$ was used to analyze categorical variables.

The most relevant elements we analyzed with the gathered data were:

- Effect of marker position: is there any difference in depth perception if the points are close or far from each other?
- Effect of left and right eye: is there any difference if the closer point appears in the left eye or the right one?
- Effect of the luminance: has the luminance of the pixels around a point to classify any effect on the correctness of the answer?
- Had time any effect on the correctness?

3.3 Results

In this experiment we found that the shading effect had effective influence on the correctness of the points classification. Directional Occlusion Shading performed significantly better than no shading, and marginally better than the other techniques, though no statistically significantly better. Half-angle slicing was also better than Phong Shading and no shading, but with no statistical significant level. The results can be seen in Figure 8.

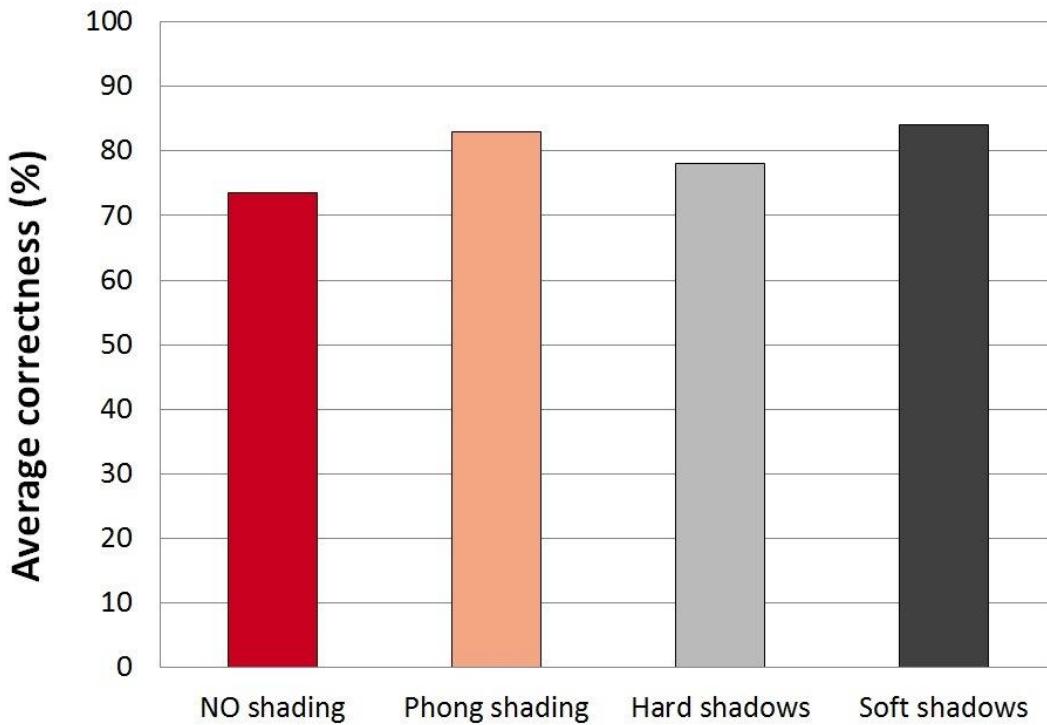


Figure 8: Average correctness for each shading technique.

The average correctness can also be analyzed per tasks. For task 1 and task 2, the results are summarized in Figure 9.

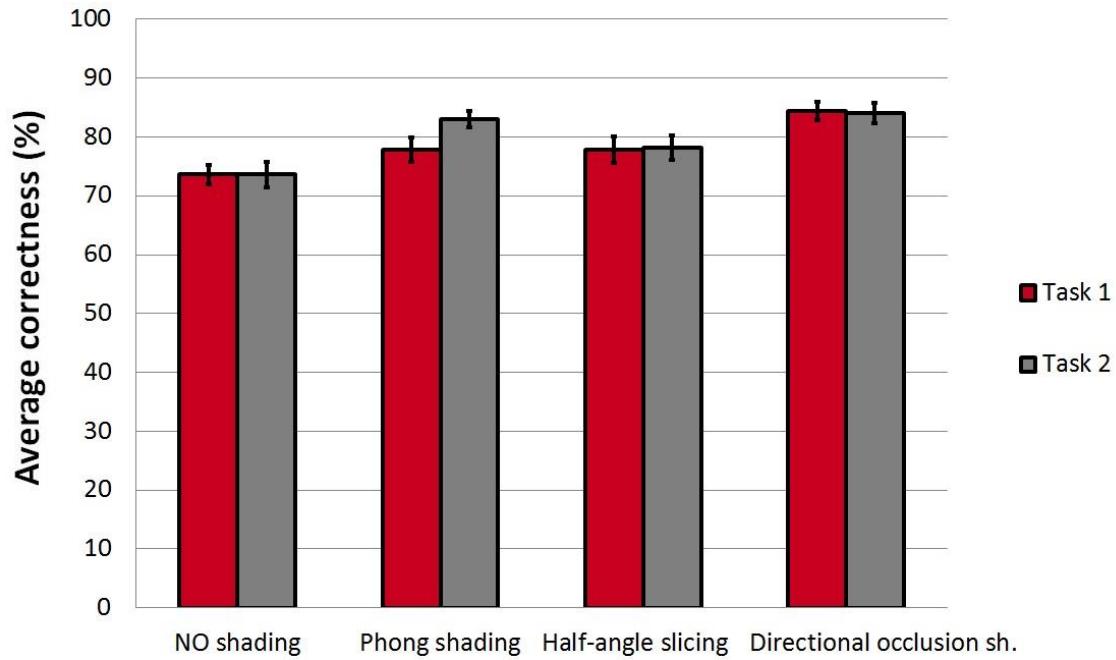


Figure 8: Average correctness for each shading technique, separated per tasks.

No significant differences seem to be related with the time elapsed to answer the questions and the different shading techniques. There seems to be a weak linear relationship between correctness and time spent. However, no significant enough.

We can conclude, after the experiment, that advanced volumetric shading techniques provide a better advantage at perceiving depth in virtual reality environments. There was also no detected correlation between the X/Y position of the markers and the correctness in the answers. Furthermore, real illumination around the scenario was not influencing in the answers.

Further details on the experiment and the results can be found in [Díaz17].

References

- [Díaz17] Díaz, J., Ropinski, T., Navazo, I., Gobetti, E., Vázquez, P. *An experimental study on the effects of shading in 3D perception of volumetric models*, The Visual Computer 33(1), 47-61 (2017)
- [Grosset13] Grosset, A., Schott, M., Bonneau, G.P., Hansen, C.: Evaluation of depth of field for depth perception in dvr. In: Proceedings of the 2013 IEEE Pacific Visualization Symposium (PacificVis), pp. 81–88 (2013)
- [Hernell09] Hernell, F., Ljung, P., Ynnerman, A.: Local ambient occlusion in direct volume rendering. IEEE Trans.Vis. Comput.Gr. 15(2), 548–559 (2009)
- [Kniss03] Kniss, J., Premoze, S., Hansen, C., Shirley, P., McPherson, A.: A model for volume lighting and modeling. IEEE Transactions on Visualization and Computer Graphics 9(2), 150–162 (2003)
- [Langer00] Langer, M.S., Bülthoff, H.H.: Depth discrimination from shading under diffuse lighting. Perception 29(6), 649–660 (2000)
- [Schott09] Schott, M., Pegoraro, V., Hansen, C., Boulanger, K., Bouatouch, K.: A directional occlusion shading model for interactive direct volume rendering. In: Proceedings of EuroVis'09, pp. 855–862 (2009)

IDI - Exercises

Professors IDI – Dep. Computer Science

Expanding targets

- Els *expanding targets*:
 - a. Es basen en la llei de Hick-Hyman.
 - b. Pretenen reduir el temps d'accés als elements basant-se en el fet que, segons la llei de Fitts, el temps d'accés es redueix si s'augmenta la longitud del desplaçament.
 - c. Si es combinen amb el moviment dels objectius poden causar confusió a l'usuari.
 - d. Cap de les anteriors.

Steering law

- **La llei de steering:**

- No es pot derivar a partir de la llei de *crossing*.
- Serveix per a modelar el temps necessari per a recórrer un camí de forma arbitrària.
- Diu que hi ha una relació logarítmica entre l'índex de dificultat de creuar un objectiu i el temps que requerit per a fer-ho.
- Diu que l'índex de dificultat de creuar un objectiu és D/W.

Interface design

Ens han encarregat fer un disseny d'una interfície per a un sistema tipus desktop en la qual hi haurà botons i menús drop-down.

- Podem predir la dificultat d'accedir als botons utilitzant la llei de Fitts i la dificultat de recórrer els menús amb la llei de crossing.
- Podem analitzar el nombre d'elements a posar en un menú utilitzant la llei de steering i en funció dels digrams.
- Podem analitzar el nombre d'elements a posar en un menú utilitzant la llei de Fitts.
- Podem analitzar la dificultat de recórrer els menús utilitzant la llei de steering.

Interface design

Els pop-up menus:

- A. Són el resultat de l'aplicació de la Llei de Crossing al disseny d'interfícies.
- B. En base a la Llei de Fitts, disminueixen l'índex de dificultat del pointing.
- C. Mai han de contenir més de 5 opcions segons Hick-Hyman.
- D. Només és recomanable utilitzar-los en dispositius tàctils.

Input devices

El ratolí...

- A. Si és molt precís, no cal que vagi acompanyat de tècniques d'acceleració.
- B. Cap de les altres respostes.
- C. És menys eficient que el *lightpen* perquè aquest darrer és més lleuger i en conseqüència causa menys.
- D. És un sistema d'assenyalament directe.

Keyboards

L'estratègia *land-on*:

- a. S'utilitza en dispositius d'assenyalament indirecte.
- b. És més ràpida que la tècnica *lift-off per a entrar text*.
- c. És l'única que s'utilitza per entrar text quan s'utilitzen teclats虚拟.
- d. Es combina amb la tècnica *pinch-to-zoom per a posar símbols que no són de l'alfabet*.

Keyboards

Per analitzar el comportament dels teclats virtuals...

- a. És molt senzill utilitzar usuaris perquè se'ls pot entrenar a teclejar de forma eficient en pocs minuts gràcies al seu coneixement previ dels teclats QWERTY.
- b. Es pot modelar el temps que cal per teclejar utilitzant aquesta fórmula: $on\ Wij = \sum Dij$ té a veure amb l'amplada de cada tecla i Dij és la distància que separa dues tecles consecutives.
- c. Es pot modelar sense tenir en compte l'espai que separa dues tecles perquè el que importa és quines tecles es cliquen de forma consecutiva.
- d. Es pot modelar sense fer servir usuaris, però no es pot fer de forma independent de l'idioma.

Keyboards

Els teclats per a dispositius mòbils:

- a. No poden dissenyar-se amb una distribució de tecles diferent a la QWERTY perquè és la que els usuaris coneixen.
- b. Es poden avaluar utilitzant un model teòric de llenguatge que contingui els digrams menys comuns per a reforçar el rendiment en aquests casos.
- c. Són difícils d'utilitzar perquè les funcionalitats estan amagades.
- d. Poden avaluar-se de forma teòrica i de forma empírica.

Mobiles

Per tal que la interfície d'usuari (UI) en un dispositiu mòbil estalviï espai:

- A. En els dispositius actuals, amb pantalles tan grans, no solem tenir problemes d'espai.
- B. Podem emprar la llei de Fitts, que ens diu on podem posar més elements de la interfície.
- C. Podem utilitzar la tècnica de *progressive disclosure*.
- D. Cap de les altres respostes és correcta.

Virtual Reality

Les tècniques de hand extension...

- a. Són tècniques de selecció que mapen la posició de la mà a una posició en un espai 3D.
- b. Són tècniques de selecció que estenen la posició de la mà llençant un raig a partir de la posició de la mà.
- c. Permeten interactuar amb models 3D en entorns desktop amb un ratolí.
- d. Construeixen un raig a partir de la posició de la mà o de l'ull i la direcció del raig es calcula a partir de la orientació del canell.

Virtual Reality

Els tres eixos de la Realitat Virtual són:

- A. Visualització interactiva, models 3D i immersió.
- B. Visualització interactiva, interacció implícita i models 3D.
- C. Immersió en 3D, interacció implícita i àptics.
- D. Visualització interactiva, immersió en 3D, i interacció implícita.

Data presentation

Quan vulguem mostrar moltes dades en una aplicació.

- a. És aconsellable organitzar la informació seguint algun dels criteris del LATCH.
- b. Organitzarem la informació utilitzant alguna categoria de les definides del garbage-in/garbage-out.
- c. Cal que les organitzem tenint en compte la llei de Prägnanz.
- d. Les ordenarem i organitzarem segons el criteri signal to noise ratio.

Visual design

Els estudis demostren que percebem els objectes del nostre entorn com a una composició de formes simples, encara que no ho siguin. Respecte a aquesta afirmació:

- a. L'affirmació és falsa, no hi ha estudis que demostrin això.
- b. Això és el que enuncia la llei de Prägnanz, o llei de la bona figura.
- c. Precisament això és el que enuncia la llei de Hick-Hyman.
- d. L'affirmació parla de la llei de destí comú.

Data Presentation

La tècnica de *chunking* consisteix en:

- a. En una web, posar un titular amb una pregunta perquè es cliqui a la notícia per a buscar la resposta.
- b. Agrupar els elements de la interfície per semblança en la seva forma o color.
- c. Escriure el contingut d'un article amb una estructura on primer hi ha el titular, el resum, després les conclusions i al final els detalls.
- d. Cap de les altres.

Fitts' Law

- Dos elements T1 i T2 a distàncies D1 = 10 cm i D2 = 8 cm en direcció horitzontal i d'amplades 5 cm i 2 cm, respectivament. Per a T1 emprem un dispositiu amb $a_1 = 200$ ms i $b_1 = 200$ ms/bit. Per a T2 utilitzem un dispositiu amb $a_2 = 200$ ms i $b_2 = 100$ ms/bit. Assumint la formulació original de la llei de Fitts:
 - a. $ID_1 > ID_2$.
 - b. $ID_1 = ID_2$.
 - c. $MT_1 = MT_2$.
 - d. $MT_2 < MT_1$.

IDI - Exercises

Professors IDI – Dep. Computer Science

Test Usabilitat

Segons la definició d'Usabilitat (ISO 9241) indica quina de les següents sentències és certa:

- A. Usabilitat sempre fa referència a un grup concret d'usuaris i a un entorn específic.
- B. La usabilitat té a veure únicament amb aspectes d'informàtica, disseny de software, i arts gràfiques.
- C. Per a que un programa sigui usable, només cal assegurar-se que sigui eficient des del punt de vista de la implementació.
- D. Usabilitat es defineix de forma general per a qualsevol usuari d'un producte.

Test Usabilitat

Per a realitzar un estudi d'usabilitat:

- a. El primer pas es l'elaboració dels perfils d'usuari i la selecció dels participants per a l'estudi.
- b. És necessari que algú de l'equip tingui coneixements d'estadística.
- c. S'ha de realitzar, preferiblement, utilitzant desenvolupadors i membres de l'equip de disseny de l'empresa que desenvolupa el producte.
- d. Només hauria d'emprar a participants que no coneguin ni el producte ni versions anteriors del mateix.

Un informe d'usabilitat

- a. Ha d'enumerar els problemes de forma prioritizada i ha de portar com a mínim una recomanació de solució per a cada problema.
- b. Només ha d'enumerar els problemes, la seva gravetat i la seva prioritat, però no ha de proposar solucions perquè els membres de l'equip no són els desenvolupadors.
- c. Ha d'ordenar els problemes en funció de la freqüència, ja que voldrem sempre arreglar els problemes que afecten a més gent.
- d. Ha d'estar escrit seguint el mètode de la cascada, perquè els lectors entenguin de seguida les conclusions i només calgui llegir fins al final si hem de trobar alguns detalls.

Test Usabilitat

Quina d'aquestes tasques és responsabilitat *del briefer* en un estudi d'usabilitat:

- a. Fer l'anàlisi estadística de les dades
- b. Gravar per vídeo el que fan els usuaris
- c. Dirigir la confecció de l'informe final
- d. Explicar als usuaris l'objectiu de l'estudi

Test Usabilitat

La tècnica d'avaluar la usabilitat de "*guerrilla testing*" :

- A. No requereix administrador, només *briefer*.
- B. En acabar cal prioritzar els errors i indicar una recomanació de solució.
- C. El participant pot estar en un bar però en silenci.
- D. Només serveix per avaluar aplicacions que s'executen en un mòbil o tablet.

Test Usabilitat

Els tests d'usabilitat remots:

- A. Són fiables perquè soLEN ser de més duració que els normals.
- B. Poden ser moderats, però fer-los així té més inconvenients que avantatges.
- C. No es poden realitzar per Skype.
- D. Si es moderen, s'han de dedicar esforços per a reclutar els participants i escollir el programari adequat.

IDI - Exercises

Professors IDI – Dep. Computer Science

Professors IDI

IDI – Usability Testing

2

Outline

- Concepts
- Usability testing
- Formal usability tests
- Simplified usability tests
- Use cases
- Exercises



3

Concepts

- Usability:

- Ease of use and acceptability of a system or product for a particular class of users carrying out specific tasks in a specific environment.
 - Where “Ease of use” affects user performance (efficacy, efficiency), satisfaction (comfort).
 - And “Acceptability” affects whether or not the product is used.



4

Concepts

- Usability:

- The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use.
- To be useful, usability has to be specific. It must refer to particular tasks, particular environments and particular users.
 - **So has to be its testing!**



5

Concepts

- How to test?
 - Ease of use is inversely proportional to the number and severity of difficulties people have in using software.
 - Let's examine the difficulties!!!



6

Outline

- Concepts
- Usability testing
- Formal usability tests
- Simplified usability tests
- Use cases
- Exercises



7

Usability testing

- Two major families by goals:

- **Determine usability problems** (*i.e. text editor*):
 - Discovery, prioritization, and resolution of usability problems
 - Iterative testing

- **Measure task performance** (*i.e. 3D selection*).

Include two fundamental tasks:

- The development of the usability objectives
- Iterative testing to determine if the product under test has met the objectives



8

Usability testing

- Great variety of usability tests:

- Can be very informal or very formal
- Often use think-aloud (TA)
 - more reliable than posterior interviews
 - doesn't affect efficiency
 - better for problem discovery than measurement
- Remote or local
- Evaluated software can be varied:
 - Prototypes, under development, competitive products...



9

Usability testing

- Testing techniques:
 - “Formal” usability tests
 - Remote testing
 - Guerrilla usability testing
 - Steve Krug’s “usability testing on 10 cents a day”
 - Heuristic/expert evaluation



10

Outline

- *Concepts*
- *Usability testing*
- **Formal usability tests**
 - Environment
 - Tasks & roles
 - Development
 - Reporting
- Simplified usability tests
- Use cases
- Exercises



11

Formal usability tests. Environment

- “Formal” usability tests require a controlled environment
 - Inside a room, outside...
 - Illumination conditions (useful for perception studies)
 - Devices used (e.g. computer with Internet connection and a browser, or a mobile...)
 - Other conditions (e.g. connection quality...)
- Usability lab ☺



12

Formal usability tests. Environment



13

Formal usability tests. Environment

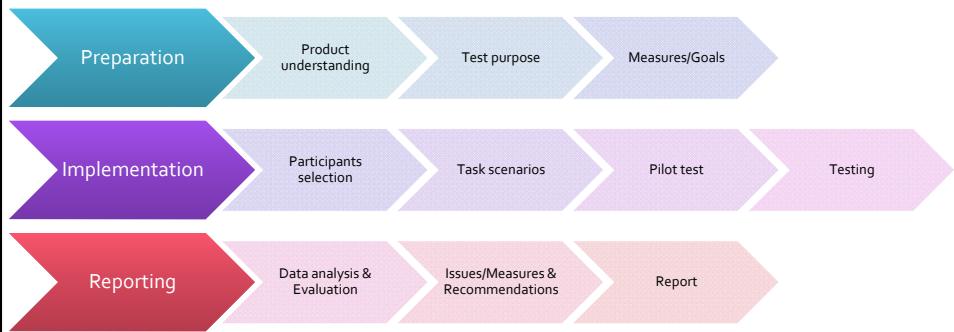
- Set of soundproofed rooms
 - Proper recording and avoiding distractions to participants
- Different areas and equipment
 - Participant area (where the experiment is carried out)
 - Observer area with one-way glass
 - Executive viewing area behind the primary observer area
 - Video cameras and microphones, telephone...



14

Formal usability tests. Tasks and roles

- Usability test workflow:



15

Formal usability tests. Tasks and roles

- Usability test roles:
 - **A:** Test administrator
 - **B:** Briefer
 - **CO:** Camera Operator
 - **DR:** Data Recorder
 - **HD:** Help Desk Operator
 - **PE:** Product Expert
 - **S:** Statistician



16

Formal usability tests. Preparation

- Preparation (A):
 - Product understanding: Purpose of the product, parts ready to test, type of users...: A, PE
 - Test purpose: Product comparison, within/between subjects...: A, S
 - Measures/Goals: Number of iterations, counting mistakes/errors, timings...: A, S



17

Formal usability tests. Preparation (1/3)

■ Product understanding (A + PE):

1. Understand the purpose of the product
2. Parts of the product are ready for testing
3. Types of people who will use the product
4. Determine the use given to the product
5. Conditions of usage of the product



18

Formal usability tests. Preparation (2/3)

■ Preparation:

- Product understanding: Purpose of the product, parts ready to test, type of users...: A, PE
- **Test purpose:** Product comparison, within/between subjects...: A, S
- Measures/Goals: Number of iterations, counting mistakes/errors, timings...: A, S



19

Formal usability tests. Preparation (3/3)

■ Preparation:

- Product understanding: Purpose of the product, parts ready to test, type of users...: A, PE
- Test purpose: Product comparison, within/between subjects...: A, S
- **Measures/Goals:** Number of iterations, counting mistakes/errors, timings....: A, S



20

Formal usability tests. Implementation

■ Implementation:

- Participants' selection: A
- Task scenarios: initial conditions, steps: A
- Pilot test: Members of the team: A, B, CO, DR, HD
- Testing, A, B, CO, DR, HD

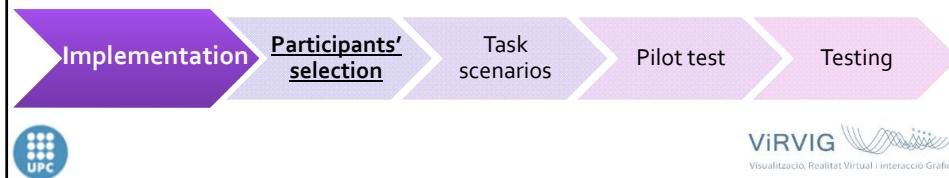


21

Formal usability tests. Implementation (1)

■ Participants' selection

- It's complicated.
- Should be representative
 - People that could be real users
 - E.g. no other managers!!!
 - Specialized recruiting agencies are a possibility
- No-show rates above 10%

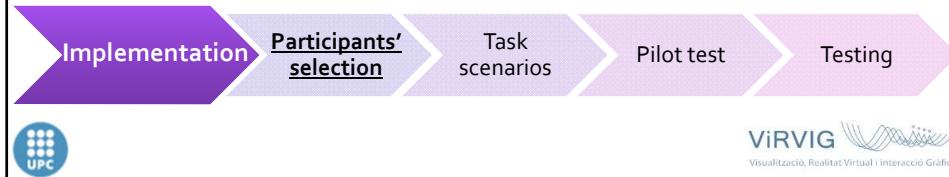


22

Formal usability tests. Implementation (1)

■ Participants' selection

- Should be payed. Factors to consider:
 - Time needed, qualification of the participant, would be the person be payed for her time?
 - Non-monetary for internal participants that are already being payed for their time
 - Always offer food and beverages!!!

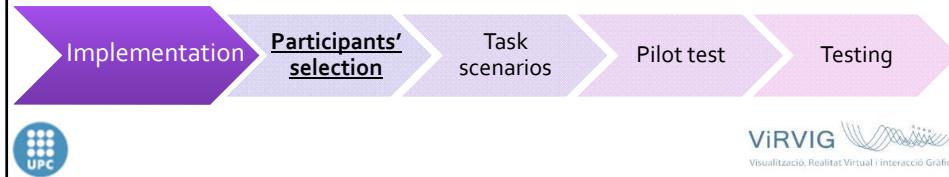


23

Formal usability tests. Implementation (1)

■ Participants' selection

- For statistical significance: 10-12 participants
- Less formal: 4-5 participants per user group
- Ensure recruiting criteria reflects user characteristics
 - E.g. for a website, ensure participants have prior experience browsing

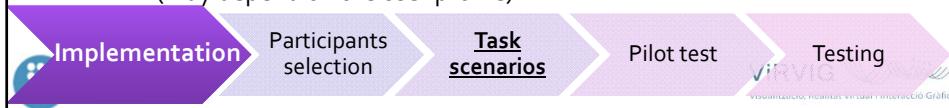


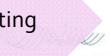
ViRVIG 
Visualització, Realitat Virtual i interacció Gràfica

Formal usability tests. Implementation (2)

■ Test task & scenarios:

- Tasks must be representative
 - Core tasks: Features that everybody uses (write a text)
 - Peripheral tasks: Features used less often (table insertion)
- Scenarios must be determined
 - Define initial conditions
 - Description of the scenario: what to do and why
 - Should not provide step-by-step instructions but should include details
 - Some action must be taken on finish
 - Not all users must be provided with the same scenarios (may depend on the user profile)



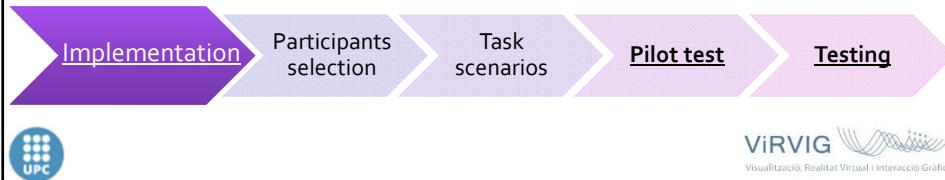
ViRVIG 
Visualització, Realitat Virtual i interacció Gràfica

25

Formal usability tests. Implementation (3)

■ **Testing (A, B, CO, DR, [HD]):**

- Brief participants: **B**
- Initial questionnaire: **B**
- Develop tasks: **B, CO, DR, [HD], A**
- Debrief: **B**
- Final questionnaires: **B**



26

Formal usability tests. Reporting

■ **Reporting (whole team):**

- Data Analysis & Evaluation: **A, S**
- Issues/Measures & **Recommendations**: **A, S, team**
- Report: **A, S, team**
 - Describe & prioritize the usability problems
 - Present quantitative measurements



27

Formal usability tests. Reporting (1)

Data analysis & evaluation:

- Frequency: Number of users that find a problem divided by the number of users testing the app or web
 - Easy (objective) to evaluate
- Severity: Importance of the problem
 - Might be completely catastrophic or simply cosmetic
 - Difficult (more subjective) to evaluate



28

Formal usability tests. Reporting (1)

- **Usability problems:**
 - Should indicate the importance: severity
 - Can be classified:
 - Mistakes: Errors due to incorrect intention
 - Slips: Errors due to appropriate intention but incorrect action
 - Expertise does not affect on the number of errors
 - But affects how fast they are handled



30

Formal usability tests. Reporting (1)

- Problem evaluation. **Dumas and Redish:**
 - **Level 1:** Prevents Task Completion
 - **Level 2:** Creates significant delay and frustration
 - **Level 3:** Problems have a minor effect on usability
 - **Level 4:** Subtle and possible enhancements/suggestions



31

Formal usability tests. Reporting (2)

- **Recommendations:**
 - Create a problem grid: frequency/impact
 - Global changes (prevent task completion) first
 - A *missing help* may be a global problem or something related with a concrete UI
 - Try to give at least one recommendation for each problem
 - Present the different trade-offs clearly

Reporting

Data analysis & Evaluation

Issues/Measures & Recommendations

Report



32

Formal usability tests. Reporting (3)

■ Problem evaluation. Conclusions

- Do not use a large number of categories
 - Do not get obsessed by the number of categories either
- Different evaluators may disagree on some problems' severity
- Treat frequency separately from severity
- Do not forget to point out positive findings



33

Outline

- Concepts
- Usability testing
- Formal usability tests
- Simplified usability tests
 - Guerrilla usability testing
 - Steve Krug's "usability testing on 10 cents a day"
 - Remote testing
 - Heuristic/expert evaluation
- Use cases
- Exercises



34

Simplified tests

- Testing just a single person early is much better than 50 near the end
- The point of testing is to inform your judgment



35

Simplified tests

- Guerrilla usability testing
 - Take someone in a coffee or public space and ask her to use a website for a couple of minutes
 - Observe users
 - Ask open-ended questions such as "What would you do here?"
 - Get to know them a bit
 - Offer coffee or bagels
 - Analyse captured data
 - Considering your audience



36

Simplified tests

- **“Usability testing on 10 cents a day”**

- Prepare some tasks to evaluate
- Grab somebody from the company as user
- Gather stakeholders in an observing room
- Let the user do a set of tasks
- Capture gestures, mouse, record...
- Discuss over lunch (order pizza for everybody)
- Report



37

Simplified tests

- **Remote testing**

- Like traditional tests but participant and facilitator are in different physical locations
 - Participants can do the test at home
 - Facilitator watches remotely



38

Simplified tests.

■ Remote testing

- Advantages
 - Cheaper and easier test setup
 - Usually faster (in terms of allocating/securing facilities travel...)
 - Can get geographically dispersed users

- Disadvantages
 - Cannot read body language.
 - Difficult to decide when to talk/interact
 - Variability in participants' motivation
 - No-show rates higher than in-person studies



39

Simplified tests

■ Remote testing. Two types:

- Unmoderated:
 - Users do the task completely alone

- Moderated:
 - Users have access to a facilitator



40

Simplified tests

- **Unmoderated Remote testing**
 - Users don't have real-time support
 - Don't get any clue on how the session went
 - No opportunity to ask detailed questions
 - Sometimes the software allows to have some of them predefined
 - Preferable to work only on a few specific elements than a broad view of a product
 - Good for tight timeframes



41

Simplified tests

- **Moderated Remote testing**
 - Facilitator can change or reorder tasks as needed
 - Facilitator can ask follow-up questions or clarifications
 - Participant is less likely to spend time on tasks not related to the test
 - Test sessions can be longer (usually about an hour)
 - Can perform more in-depth tests
 - The team can watch the test and discuss afterwards



42

Simplified tests

■ Heuristic evaluation:

- 3-5 usability experts evaluate an app or UI
- Use pre-defined principles (heuristics)
- Can highlight usability issues before user testing



43

Simplified tests

■ Heuristic evaluation. Advantages

- Can be quick and cost effective
 - If we have internal resources
- Can be used early in the design process
- Can give a comprehensive usability status of a product's UI
- Is compatible with other usability testing methods



44

Simplified tests

- **Heuristic evaluation.** Process:

- Collect the UI
- Understand the business and users' needs
- Understand user motivations and tasks to accomplish
- Define the heuristics
- Use a minimum of 3 experts
- Set up a consistent evaluation system
- Highlight problem(s) and its rating
- Compare and analyse the results of multiple experts



45

Outline

- Concepts
- Usability testing
- Formal usability tests
- Simplified usability tests
- Use cases (*Read & Study the document uploaded to the "racó"*)
 - Guerrilla testing: WhatsApp web app
 - Measure test: Depth perception in VR
- Exercises



[uxdesign.cc](https://uxdesign.cc/a-usability-test-on-what...)

A usability test on WhatsApp Web – UX Collective

Saadia Minhas

5-7 minutos

How to solve pain points identified in the whatsapp experience.



WhatsApp

WhatsApp is a messaging service for mobile phones and web that allows you to send text messages, images, audio and video through internet. WhatsApp is being used worldwide and is the most popular online messaging app.

WhatsApp desktop version is also very convenient to use and send messages to your contacts while working on your system. However, when tested with users it reveals few pain points that they face while using certain features.

This article presents the details and results of a Guerrilla usability test along with few recommendations.

Objective

To identify the users' pain points on WhatsApp web.

Testing Parameters

- *What is tested:* Two most commonly used tasks in WhatsApp web
- *With whom it is tested:* 3 users with 2 new users of WhatsApp web
- *How it is tested:* Through observations and interviews

Process Followed

The process of [Guerilla usability testing](#) is followed to test the identified tasks on WhatsApp. This involves following stages.

Usability Testing Stages

1. Identify User Tasks

A list of tasks is created that a user is able to perform using WhatsApp web.

2. Prioritize Tasks

Tasks are prioritized based on how frequently a user will perform them.

Each task is marked with a point from 1 to 3.

Most commonly used tasks are given 3 points, tasks that are performed occasionally are marked with 2 points, and

tasks that are done in a while are given 1 point.

Two tasks are identified as the most commonly used, and given 3 points.

1. Send message to your friend
2. Share photos with your friend

3. Perform Testing

The selected tasks are given to users along with instructions. Two methods are followed to collect their feedback.

- Users are observed while performing actions
- They were talked about their experience when they perform certain tasks

Below is the user flow of each task along with an indication of their expressions at that point.

Task 1: Send message to your friend

User flow for Task 1: Send Message

Task 2: Share photos with your friend

User flow for Task 2: Send Photos

4. Analyze Pain Points and Recommend Solutions

Pain Point 1: Finding a Contact

There are two ways to start a new chat: (i) Search within Chats list, (ii) Go to New Chat icon on top and search

contact. The user was not clear to differentiate between these two options.

Searching in Chats list gives an assumption that search will run through the Chats list only, and in fact it works for both Chats and Contacts. On the other side, New Chat option also provides a list of Chats as well as Contacts. This requires some kind of clarity.

Current WhatsApp image to start a new chat

Recommendation:

A clear separation between Chats and Contacts needs to provide. This can be done by giving a filter option in Contact list, or a single list can be sorted based on Recent Chats or Contact names.

First way provides a filter at bottom of Contact list

Another option is to provide filters in a drop-down on top of Contact list

Pain Point 2: Viewing Message Info

In Message Info pane, the area showing message status is merged with the Message pane. Also, it is not clear that user is viewing status of which message. Also, it took time for the user to find Close icon on top on Message Info pane.

Recommendation:

The area of Message info pane and Message pane needs

to differentiate clearly. Since this is desktop version and Message area is still visible when Info pane is opened, the link between message and its info could be made more prominent. The selected message is shown as highlighted.

Recommended Message Info pane

Pain Point 3: Using Attach Option

The Attach menu and tooltips do not match with the UI. It gives a totally different theme and experience in current screen.

Current WhatsApp Attach option

Recommendation:

Menu placement and theme is made consistent with UI. Instead of Tooltips, the option names along with icons seem more helpful.

Attachment icon is recommended to display closer to Message box

The Attach menu with a more user-friendly layout

Pain Point 4: Attaching Photos

Close icon with Preview title is confusing. The user clicked it just to close the preview of selected photos, but it discards all the selected photos.

Adding more files option is not clear. The Attach icon still displays on top, but it is not functional. The user clicked on

that icon first.

It is difficult to navigate large number of selected files.

Current WhatsApp image to view selected Photos

Recommendation:

Preview area is renamed to Attachments to avoid any confusion for the user. Scrolling is provided in thumbnails area. User can add more files by clicking the Add icon with Caption box.

Recommended changes in Photo Attachment feature

Few More Observations

- Using a scrollbar requires high accuracy to hold the bar and scroll it. The cursor got changed to resize when user tried to scroll Message pane. Scrolling cannot be done using keyboard in Contacts pane and Contact/Group Info pane.
- Notifications in response to an action display on bottom left corner of Contact list where user skipped it multiple times.
- Status cannot be updated on desktop version, neither user can delete his status. Also, user is not able to see the viewers of status.

Final Thoughts...

The purpose of this article is to provide user feedback about WhatsApp Web. There is no questions on the innovative user experience that WhatsApp is providing to its users,

and I hope this exercise will help to further enhance it.

Happy WhatsApping !!

45

Outline

- Concepts
- Usability testing
- Formal usability tests
- Simplified usability tests
- **Use cases** (*Read & Study the document uploaded to the "racó"*)
 - Guerrilla testing: WhatsApp web app
 - Measure test: Depth perception in VR
- Exercises



46

Use case. WhatsApp web app

- Web application



47

Use case. WhatsApp web app

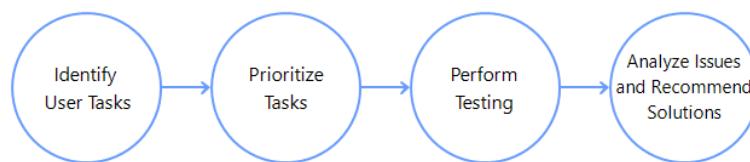
- Type of usability test: Guerrilla
- Objective
 - Identify common problems on WhatsApp web
- Testing parameters
 - What is tested: Just two common tasks
 - Participants: 3 users, 2 never used it previously
 - Test procedure: Observation + interview



48

Use case. WhatsApp web app

- Test process

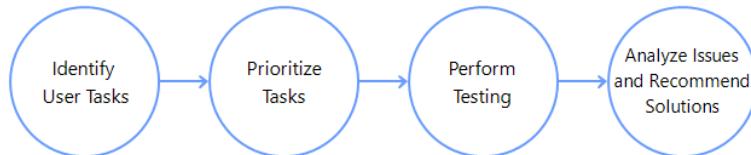


49

Use case. WhatsApp web app

■ User tasks

- Send a message to a friend
- Share photos with a friend



50

Use case. WhatsApp web app

■ Development (perform testing)

- Give the instructions to the users
 - Users are observed with performing actions
 - Asked about the experience on certain subtasks



51

Use case. WhatsApp web app

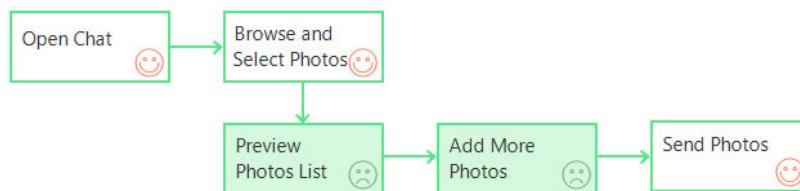
- Development of task 1



52

Use case. WhatsApp web app

- Development of task 2



53

Use case. WhatsApp web app

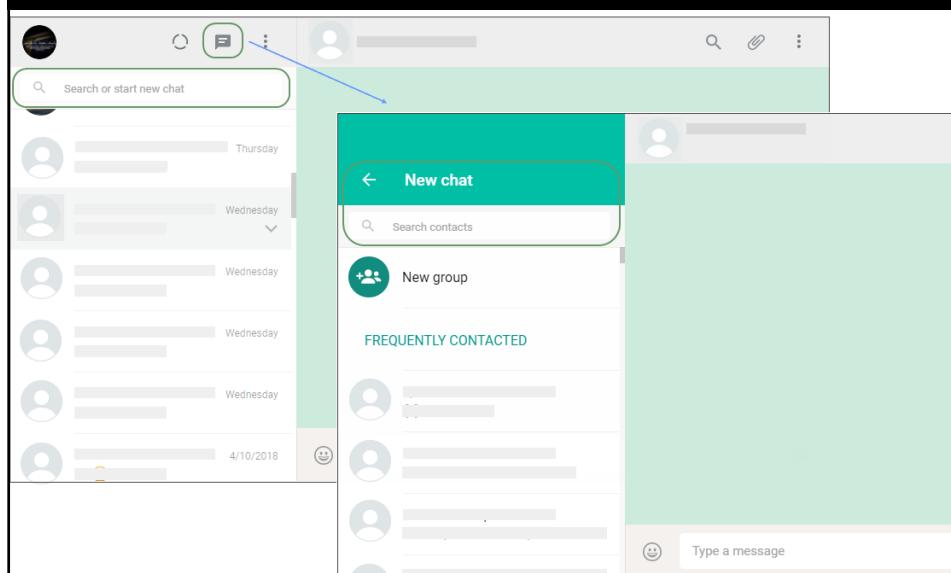
■ Analyse problems. Finding a contact:

- There are two ways to start a new chat:
 - (i) Search within Chats list
 - (ii) Go to New Chat icon on top and search contact
- The user was not clear to differentiate between these two options



54

Use case. WhatsApp web app



55

Use case. WhatsApp web app

■ Recommendation:

- A clear separation between Chats and Contacts is needed
 - Can be done by giving a filter option in Contact list, or a single list can be sorted based on Recent Chats or Contact names.



56

Use case. WhatsApp web app

Three filter options display in a drop-down

The screenshot shows the WhatsApp Web interface. On the left, there is a sidebar with a list of contacts. A callout box highlights the 'Recent' dropdown menu, which contains three filter options: 'Recent', 'Starred', and 'Online'. The contact 'Miyoko Brendle' is listed at the top of the list, followed by 'Justin Rasor', 'Lavada Schade', 'Ria Morales', 'Mathew Weddle', 'Ricky Alexis', and 'Sima Hafer'. Each contact entry includes a small profile picture and a checkmark icon. On the right side of the screen, the main chat area is visible, showing the name 'Miyoko Brendle' and a search bar. At the bottom, there is a message input field with a placeholder 'Type a Message' and a send button icon.



57

Use case. WhatsApp web app

■ Analyse problems. Viewing Message Receipt:

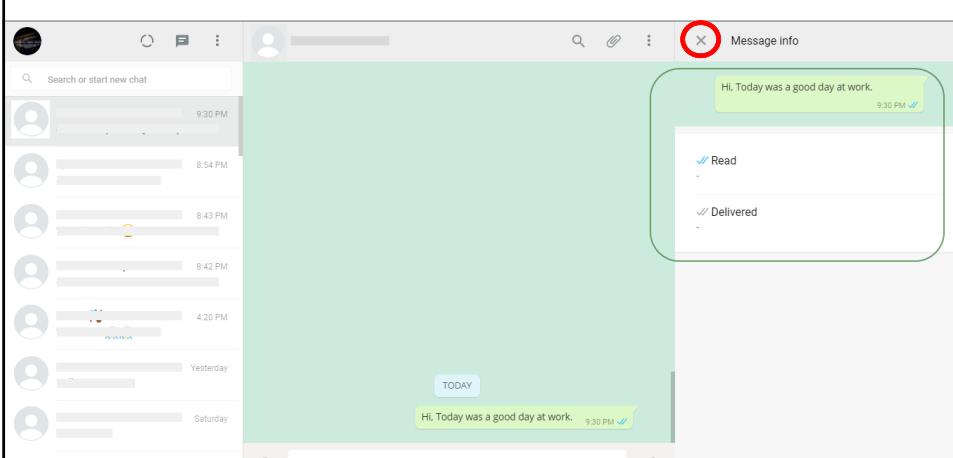
- In Message Info pane, the area showing message status is merged with the Message pane
- Also, it is not clear that user is viewing status of which message
- Also, it took time for the user to find Close icon on top on Message Info pane



ViRVIG 
Visualització, Realitat Virtual i Interacció Gràfica

58

Use case. WhatsApp web app



ViRVIG 
Visualització, Realitat Virtual i Interacció Gràfica

59

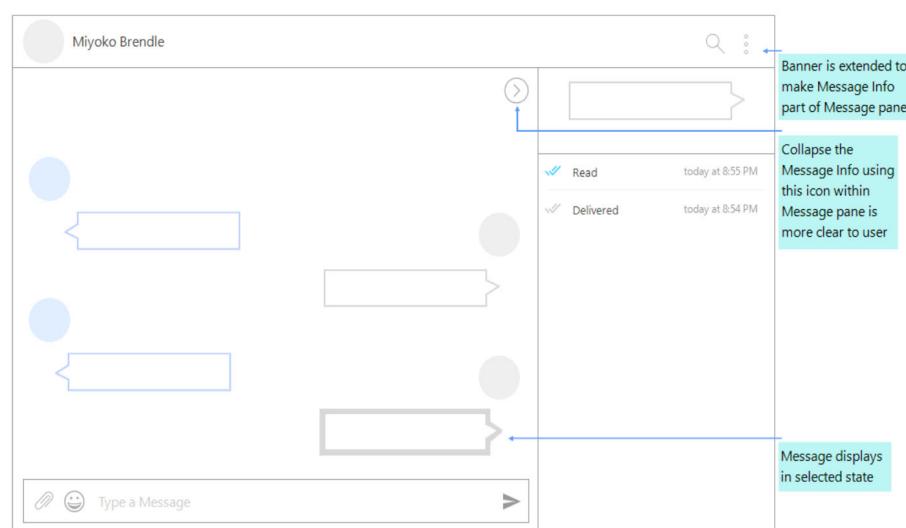
Use case. WhatsApp web app

- **Recommendation:** The area of Message info pane and Message pane needs to differentiate clearly
 - Since this is desktop version and Message area is still visible when Info pane is opened, the link between message and its info could be made more prominent



60

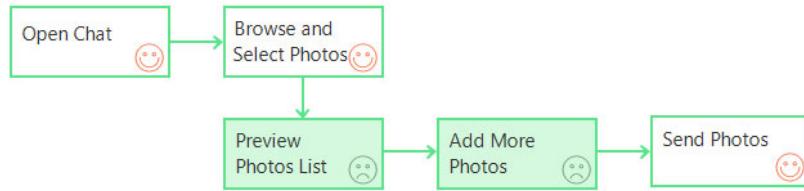
Use case. WhatsApp web app



61

Use case. WhatsApp web app

- Development of task 2

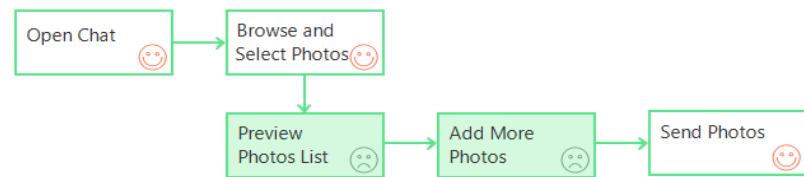


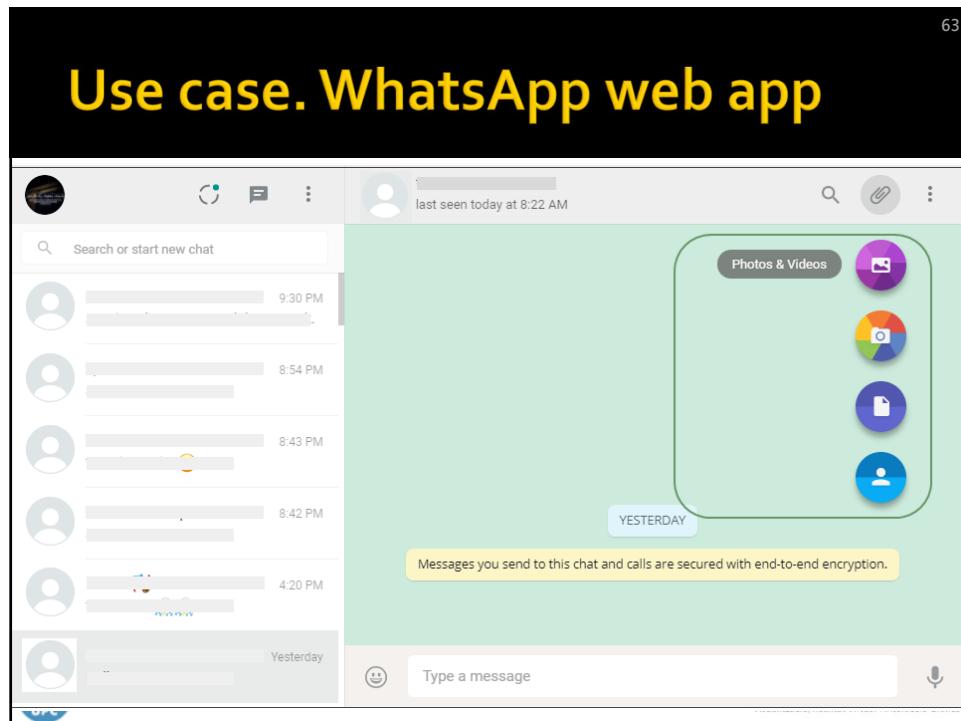
62

Use case. WhatsApp web app

- Analyse problems. Using attach

- The Attach menu and tooltips do not match with the UI
- Shows a totally different theme and experience in current screen





64

Use case. WhatsApp web app

- **Recommendation:** Make menu placement and theme consistent with UI.
- Instead of Tooltips, the option names along with icons seems more helpful.

The image contains two side-by-side screenshots of the WhatsApp Web interface. Both screenshots show a list of recent contacts on the left and a detailed chat window on the right. In the first screenshot, a tooltip at the bottom of the message input field says 'Attach icon is placed along with Message box'. In the second screenshot, a tooltip at the bottom of the message input field says 'The menu contains Attach options with titles'. The second screenshot also shows a different placement of the menu icons at the top of the message input field.

65

Use case. WhatsApp web app

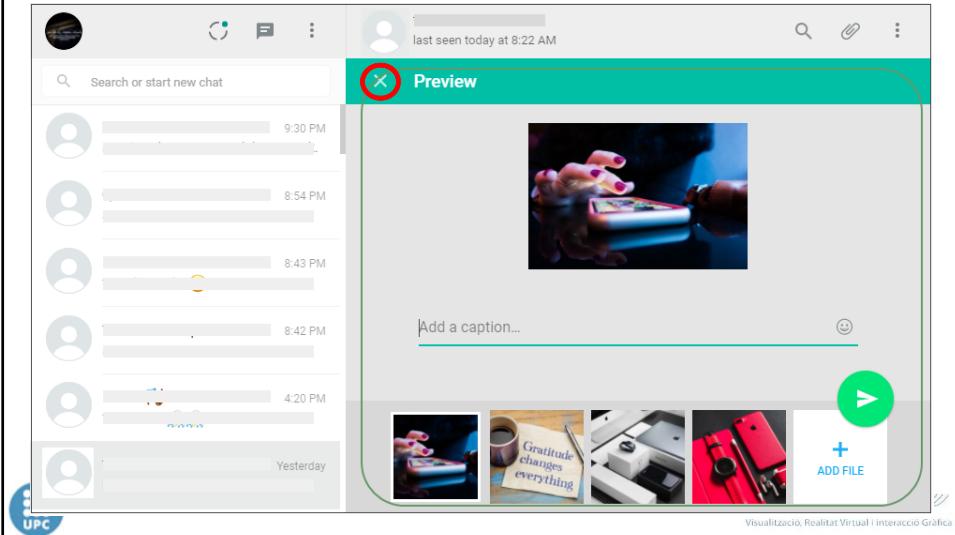
■ Analyse problems. Attaching photos:

- Close icon with Preview title is confusing.
 - The user clicked it just to close the preview of selected photos, but it discards all the selected photos.
- Adding more files option is not clear.
 - The Attach icon still displays on top, but it is not functional. The user clicked on that icon first.
- It is difficult to navigate large number of selected files.



66

Use case. WhatsApp web app



67

Use case. WhatsApp web app

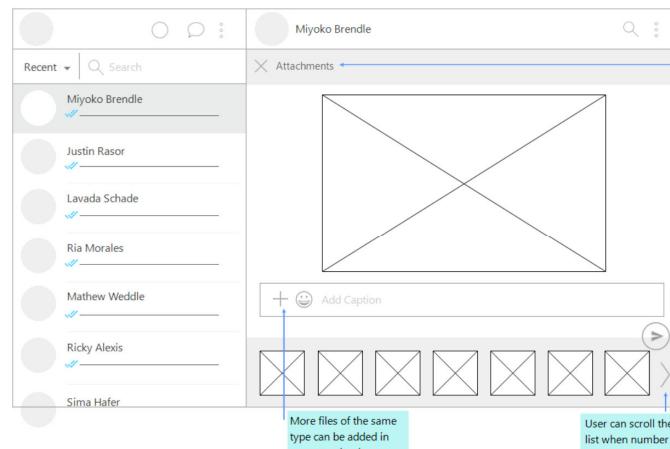
■ Recommendation:

- Rename preview area to Attachments to avoid any confusion for the user.
- Scrolling in thumbnails area
- User should be able to add more files by clicking an Add icon with caption



68

Use case. WhatsApp web app



69

Use case. WhatsApp web app

- More Observations

- Using a scrollbar requires high accuracy to hold the bar and scroll it
 - Cursor is changed to resize when user tries to scroll Message pane
 - No keyboard scroll allowed in Contacts & Contact/Group Info
- Little visibility of actions' visual feedback (bottom left)
 - Were skipped multiple times
- Status cannot be updated on desktop version
 - Users cannot see others' status



70

Outline

- Concepts
- Usability testing
- Formal usability tests
- Simplified usability tests
- Use cases
- Exercises



Professors IDI

IDI – Usability Testing

72

Use case. Depth perception in VR

- Goal:

- Evaluate performance of shading technique in VR environments

- Context:

- Perception of complex, volume datasets is difficult in VR
 - Shading techniques may enhance shape and depth perception



73

Use case. Depth perception in VR

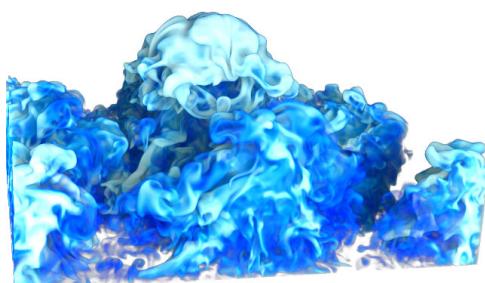
- Purpose of the test:
 - Analyze whether shading techniques influence the perception of shapes and depth in VR
- Methodology:
 - Provide images under different shading conditions
 - Ask the users to classify two points of the scene placed at different depths
 - Analyze the results obtained



74

Use case. Depth perception in VR

- Sample images:



75

Use case. Depth perception in VR

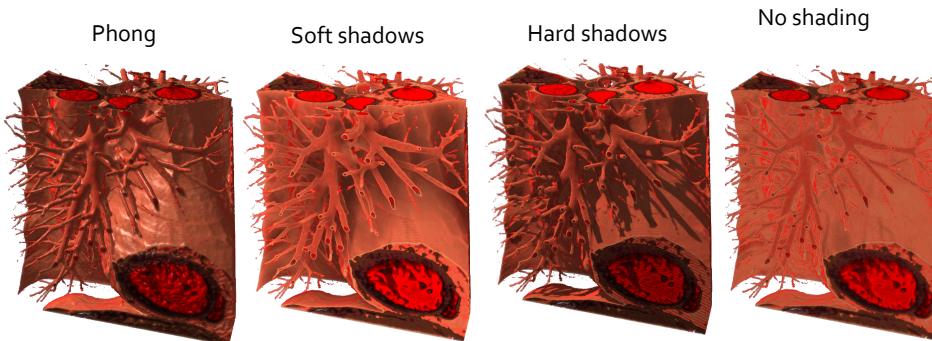
- Test preparation:
 - Select shading models (4)
 - Select models (likely unknown to users)
 - Determine number of participants, iterations
 - Low level perception problem -> should be > 10
 - Latin squares balance results -> 16 per experiment
 - Two tasks



76

Use case. Depth perception in VR

- Shading techniques:



77

Use case. Depth perception in VR

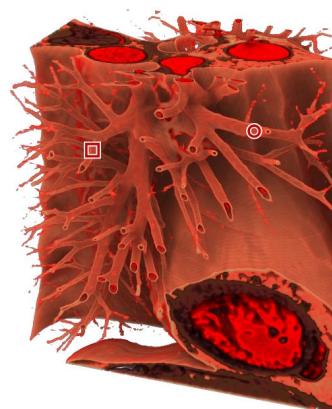
- Images selection:
 - Select models likely unknown
 - Avoid previous knowledge
 - Random shading sorting
 - Avoid learning (shading)
 - Random model sorting
 - Avoid learning (model)
 - Latin squares
 - Avoid fatigue and learning (within users)



78

Use case. Depth perception in VR

- Task: Select the closer point. 2-alternative forced choice (2AFC)



79

Use case. Depth perception in VR

- Measures (what we measure in the test):
 - Time to answer
 - Correctness



80

Use case. Depth perception in VR

- Variables to include in the analysis (to discard confounding or correlating variables)
 - Shading technique
 - Depth values
 - May analyze if absolute difference correlates with correctness
 - Previous VR background
 - Information of images for left and right eye
 - Luminance of the points' environment
 - Correlation between depth and shading maps



81

Use case. Depth perception in VR

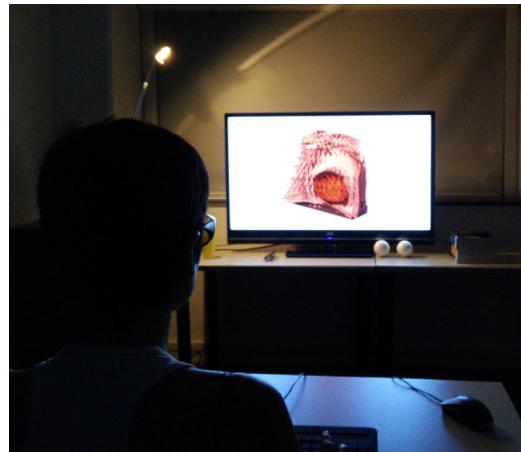
- Experiment setup:
 - 3D TV
 - Users placed at fixed distance
 - Chair to reduce movements
 - Avoid parallax as confounding variable
 - Dark room
 - External light (for virtual light source consistency analysis)



82

Use case. Depth perception in VR

- Experiment setup:



83

Use case. Depth perception in VR

- Experiment setup:
 - Modified keyboard to facilitate entry
 - Will compute timings



Visualització, Realitat Virtual i Interacció Gràfica



84

Use case. Depth perception in VR

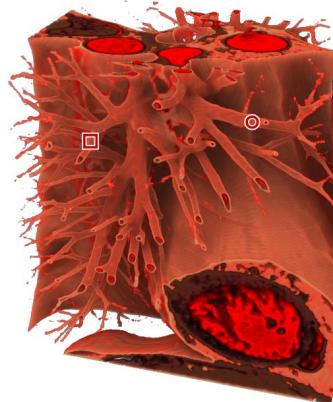
- Experiment setup:
 - Initial questionnaire (background, VR exposition...)
 - Initial training
 - Tasks
 - May rest between tasks
 - Post questionnaires



85

Use case. Depth perception in VR

- Task: Select the closer point. 2-alternative forced choice (2AFC)



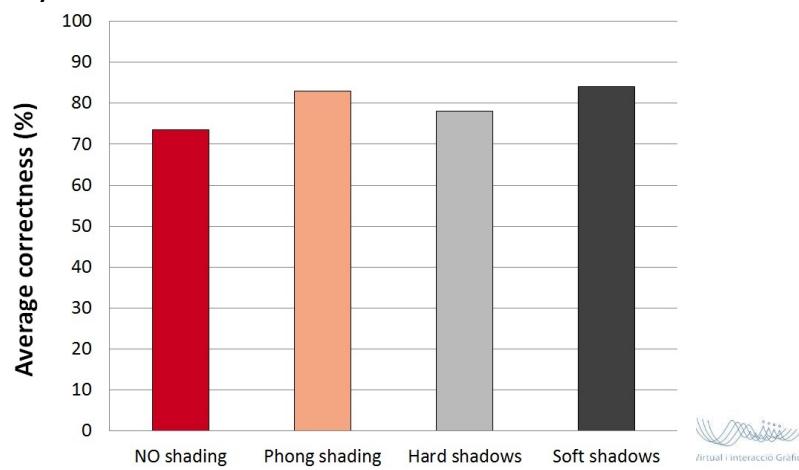
ViRVIG 
Visualització, Realitat Virtual i interacció Gràfica



86

Use case. Depth perception in VR

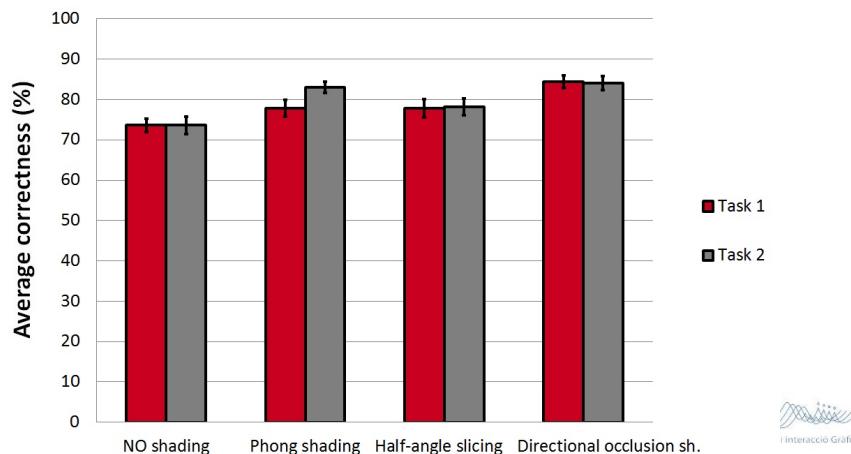
- Analysis of results



87

Use case. Depth perception in VR

■ Analysis of results



88

Use case. Depth perception in VR

■ Statistical analysis:

- **ANOVA test:** One-way analysis of variance to reject the null hypothesis that all correctness means are equal between shading techniques.
- For a significance level of $\alpha = 0.05$, a Bonferroni post-hoc test with the same acceptance level to reveal differences between the individual shading techniques
- **Result:** reject the null hypothesis when $p < 0.05$



89

Use case. Depth perception in VR

■ Statistical analysis.

- Chi-square test of association for the categorical variables relative depth and users' answers from tasks 1 and 2

Variables	χ^2	p value	Correct answers for each depth category
T1: relative depth vs. users' answers	5.991	<0.0001	<0.05: 66 % 0.05–0.1: 88 % >0.1: 86 %
T2: relative depth vs. users' answers	5.991	<0.0001	<0.05: 63 % 0.05–0.1: 86 % >0.1: 87 %

90

Use case. Depth perception in VR

■ Statistical analysis:

- The ANOVA analysis ($\alpha = 0.05$, $p < 0.0001$) of the NMI values shows that there is a significant difference between the images shaded with DOS with respect to the images shaded using HA or PH. A further Bonferroni's test revealed that DOS provides a significantly higher NMI (average NMI = 3.327) than HA (average NMI = 1.84) and PH (average NMI = 1.88).
 - Instead, there is no significant difference between the NMI means of HA and PH.



91

Use case. Depth perception in VR

- Guidelines and recommendations
 - Using advanced volumetric shading improves depth perception
 - Among the tested shading models the simulation of soft shadows by using directional occlusion shading for desktop-based VR seem to yield better results



92

Use case. Depth perception in VR

- Guidelines and recommendations
 - Real illumination does not affect depth perception when using advanced volume illumination techniques
 - External lighting may be carefully controlled to provide a pleasant environment
 - Specular highlights on the screen, reflections, or over-illuminated areas will certainly affect the correct perception of the data



93

Use case. Depth perception in VR

- Guidelines and recommendations
 - When trying to judge depth in volume models, the X/Y relative position of the markers or the luminance of the points to classify seems to have no importance



94

Usability. Test Planning: Measures

- For problem discovery:
 - Focus on prioritizing problems
 - Include frequency of occurrence
 - Likelihood of occurrence in normal usage
 - Magnitude of impact on the participants
 - Pre-planned number of iterations
 - Number of participants small, but multiple iterations,...



95

Usability. Test Planning: Measures

- For measurement tests:

- Categories
 - Goal achievement indicators (success rate and accuracy)
 - Work rate indicators (speed and efficiency)
 - Operability indicators (error rate and function usage)
 - Knowledge acquisition indicators (learnability and learning rate)



96

Usability. Test Planning: Measures

- For measurement tests:

- Fundamental global Measures
 - Successful task completion rates
 - Mean task completion times
 - Mean participant satisfaction ratings (on a task-by-task basis)
 - There are standardized questionnaires for this
 - Other measurements could be:
 - Number of tasks completed within a specified time limit, number of wrong menu choices, number of user errors, number of repeated errors (same user)



Usability. Test Planning

- After measurements choice, usability objective can be determined
 - It's usually better to set goals that make reference to an average (mean) than to a percentile
 - Sample means drawn from a continuous distribution are less variable than sample medians
 - Unless there is missing data due to participants failing to complete tasks
 - Percentile goals require large sample sizes
 - You can't measure accurately at the 95 percentile unless there are at least twenty measurements



IDI – Mobile Interaction Design

Pere-Pau Vázquez – Dep. Computer Science, UPC

Contents

1 Introduction.....	1
2 User interface and interaction design guidelines	2
2.1 Keep navigation simple	2
2.2 Design finger-friendly tap-targets.....	3
2.3 Progressive Disclosure & minimizing cognitive load.....	3
2.4 Make text legible	4
2.5 Provide feedback on interactions.....	5
2.6 Keep content to a minimum & reduce clutter	5
2.7 Reduce the inputs required from users & offload tasks	6
2.8 Don't make users wait for content.....	7
2.9 Gestures.....	7
2.10 Generate continuous integrated experiences.....	8
2.11 Don't replicate the web experience on apps	8
References.....	9

1 Introduction

Mobiles and, to a minor extent, wearables, have become an integral part of our lives. They have shaped the way we interact with other people, and are changing how we interact with businesses too.

Most of the general UX and UI guidelines are valid for mobile, but there are many aspects that are particularly different, and user interfaces and interaction design must be changed dramatically in most cases.

Mobile devices and wearables have different requirements when we design applications or services from the point of view of user experience. Mobiles (and to a greater extent wearables), place unique requirements on the design of the user experience. The main limitation is the screen size, which makes difficult the display of information as well as the interaction with elements. As a result, most of the user experience design has to focus in two important aspects *efficiency* and *discoverability*.

But there are other aspects, that do not depend on its form factor, but on their use that make them also quite particular:

- Mobile phones are much more personal: We use mobiles in a much broader set of situations than a regular computer. We will ask them to guide us in a foreign country, or to find good restaurant recommendations on the way, or we may want to pay a certain transaction.
- Entering data is much more complicated. Though we have better ways than in a regular laptop (we can upload images and videos, we can get position and movement tracking...), traditional ways of entering data, e.g. typing, are painful, as compared to using a regular keyboard.

The main reason we want to use it everywhere is because of its portability. Obviously, if we have a computer in our pocket everywhere and the whole time with us, what better way to take benefit of it than using it as much as possible?

The fact that most mobile devices do not sport a regular physical keyboard does not only depend from their form factor. Blackberries have had physical keyboards for many years, but it is also a trend (more screen to watch content), than a real necessity. This makes input awkward, time consuming, and prone to errors.

As a consequence, there are a lot of aspects that come from the physical and technological limitations of mobile devices, but also on how we use them that make them so special. So great care has to be taken in order to design user interfaces and interaction to ease the use of applications in mobile devices. The environment in which users use their mobiles competes for their attention. For example, users may need to focus on driving at the same time than getting directions from the mobile. So facilitating the users' tasks in the application is a must, since the users may not be fully able to focus on it.

As a result, when designing an application, the users should be able to:

- Quickly find what they intend to.
- Interact with the application with the minimum additional cognitive load.
- Be able to process the information with easy-to-digest chunks.

In this document, we provide some guidelines, most of them indicating what to do and what not to do when designing applications for the mobile environments. Note that some of them are just more restricted versions of well-known UX design guidelines.

2 User interface and interaction design guidelines

The following rules will help you design more usable mobile interfaces.

2.1 Keep navigation simple

It is a very obvious concept, but it is of uttermost importance in mobile. And it should be of top priority for every mobile app. After all, if our application provides awesome features that are difficult or impossible to reach, these would end up being useless. Since there will be little room to add navigation support elements such as breadcrumbs, and due to the impossibility of having long menus, we have to think navigation carefully. Take advantage of elements such as tabs, and make the navigation as self-evident as possible.

Some elements to take into account:

- **Ensure navigation feels familiar to users:** Users will feel more comfortable when the application meets their expectations. Do not use navigation patterns that are unfamiliar to the target audience. Be prudent with gestures, since these typically do not come with visual cues to indicate them, and the users may be less familiar with it.
- **Design good information architecture:** Information architecture is about organization of information in a logical and clear way. In order to facilitate the navigation, the user should spend the minimum number of steps to reach their destination.
- **Navigation should not grab more user attention than necessary:** The important element in your app is the content, not the navigation. Ensure you do not prioritize visual representation of the navigation over the content.
- **Ensure the users know their location:** This is a pretty basic rule from UX design. People should always know where they are in your application so that they can navigate successfully.
- **Strive for consistency:** Another basic rule of UX design, the visual elements for navigation should stay consistent for all the application. And if possible, with the OS widgets. Never forget that the users will spend more time using other applications than yours, so use their previous knowledge in your advantage.

- **Provide a clear path:** If 90% of your users will look for a certain element in your application, do not bury it under multiple other options. Amazon's users are going to shop most of the time, Amazon's app will not make the principal focus of attention other services such as Prime reading or Amazon Web Services.
- **Design with a clear visual hierarchy:** Use of font sizes, as well as layout may help you direct the users' attention to the right elements.

2.2 Design finger-friendly tap-targets

Probably everybody is familiar now with the so-called *fat finger problem*. Since we interact with mobile screens with our fingers instead of a stylus nowadays, the need of targets with a size big enough for us to tap is prevalent (see Figure 1).



Figure 1: Target size of 10x10mm is a good size to facilitate easy tapping.

Taking into account Fitts' law, the larger the elements, the easier to reach. However, we are very often limited by the available space in mobile. Opposite to desktop, where mouse pointing is highly precise, and thus targets can be reduced significantly, on mobile reducing the size is prone to errors and missclicks. In addition to this, the space between targets needs also be large enough to avoid clicking on the wrong element.

2.3 Progressive Disclosure & minimizing cognitive load

Cognitive load refers to the amount of brain power required to use the app. The human brain has a limited amount of processing power, and when an app provides too much information at once, it might overwhelm the user and make them abandon the task.

As a result, and in line to what we already mentioned in the first case, since it may be overwhelming (and it is not desirable) to show as much information as possible,

a good practice is to use the technique called progressive disclosure. The rationale behind that is that users do not need to have information or action visible before needing it. Progressive disclosure is an opportunity to reduce cognitive load and to improve the comprehension of the interface.

2.4 Make text legible

In many cases, most of your application contents will be either text or images. In the case of text, you should ensure it is legible and understandable for your users. You have to take into account font types and sizes. Some recommendations:

- Choose a typeface that works well in multiple sizes and weights to maintain readability and usability in every size. Typically, platforms will offer a default font (e.g. Google offers Roboto and Noto, while Apple uses San Francisco family).
- Use legible font sizes: For instance, text should be at least 11 points so that users can read it at a typical viewing distance without zooming (see Figure 2).

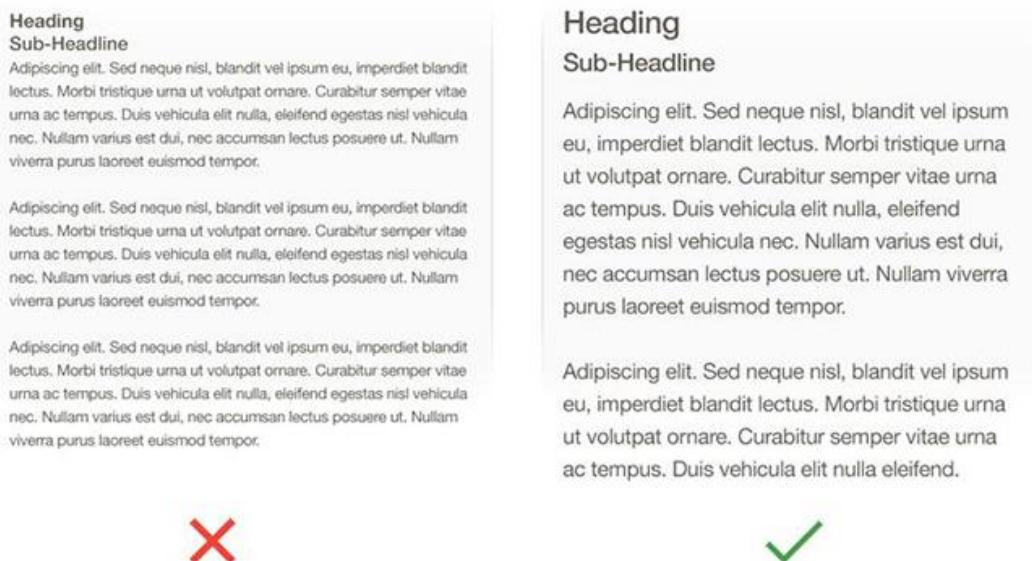


Figure 2: Small font size, like on the left, causes eye strain.

Besides font sizes, contrast is also important. Insufficient contrast makes text blend in with the background.

The vocabulary is also important using jargon, acronyms, brand-specific terms, cultural-specific axioms or technical terminology can make people not understand your text. It is better to use simple and direct language to maximize clarity.

2.5 Provide feedback on interactions

Whenever we interact with UI widgets, two elements come into play: user input, and computer reaction to it. Some tasks require more time than others, but users should never be left guessing whether the interaction was received by the computer. So it is essential to provide some sort of feedback in response to every user action. Nowadays, most platforms will help you use the so-called microinteractions. Microinteractions are animated responses to user's gestures. Accompanying these microinteractions with other visuals such as loading indicators helps providing the feeling that the application is responsive, and also informs the user that something in response to their action is being performed.

2.6 Keep content to a minimum & reduce clutter

Reducing the clutter is one of the major recommendations for mobile. Clutter is one of the worst enemies of good design. By cluttering your interface, you overload users with too much information. Clutter is bad on desktop, but is a much more important problem in mobile, due to the reduced space, and the need of larger targets for us to tap without mistakes. To reduce clutter one must make an important prioritizing effort:

- Keep content to a minimum: Present the user with only what they need to know.
- Keep interface elements to a minimum: Simple designs will simplify the learning curve (see Figure 3).
- As an alternative to provide more features, use progressive disclosure.

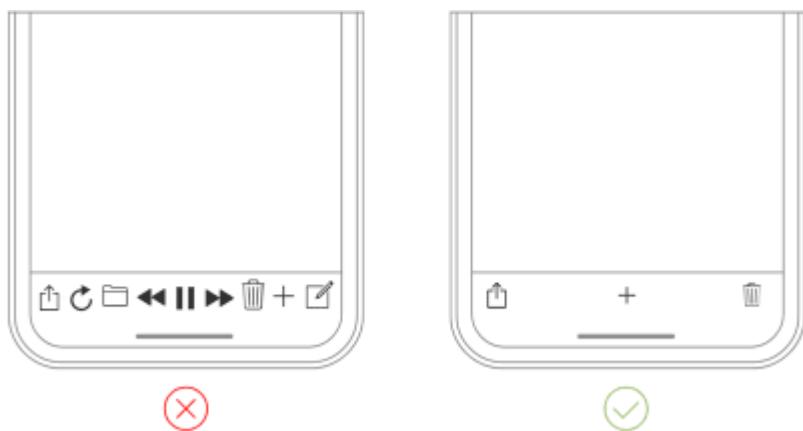


Figure 3: The bottom menu bar on the left leaves no breathing room, and makes interaction more difficult.

Of course, keeping the content to the minimum somewhat competes with the need of your application to provide a higher number of features. The latter will make your application more attractive to users, unfortunately this may affect user experience. A couple of ideas to keep in mind are:

- Strive for minimalism: Focus on the content that is valuable for your users and remove unnecessary elements that do not support user tasks. Minimal use of decorative elements such as gradients and drop shadows will help keeping the interface light and simple.

- **Prioritize one primary action per screen:** Try to design each screen for one thing and one thing only.

Note that, as in a presentation, you need to define clearly what message you want to communicate at each point.

2.7 Reduce the inputs required from users & offload tasks

Many applications will require more or less demanding input from the user. From complex onboarding processes where the user needs to provide all sorts of personal information such as in medical applications, to other shorter, but also painful tasks such as logging on any platform.

In China for example, where most users have accounts associated with mobile phone numbers instead of e-mails, the use of techniques such as SMS sending a one-time password (OTP), greatly facilitates logging in many services. Another very popular alternative is the use of QR codes. Since most of the users in China have WeChat tool installed and it incorporates QR recognition, it is also a simple to use alternative to requiring the memorization of dozens of different passwords.

In these and other cases, it is better to look for anything in the design that requires user effort and look for alternatives. From default values to previously entered data anything may be useful to reduce user's effort. In contrast to desktop, where we type with efficiency, in mobile this is still a painful experience, so tasks such as filling forms are undesirable in mobile.

Some recommendations:

- Keep forms as short as possible by removing any unnecessary fields. The app should ask for only the bare minimum of information from the user.
- Provide input masks. Field masking is a technique that helps users format inputted text. A mask appears once a user focuses on a field, and it formats the text automatically as the field is being filled out, helping users to focus on the required data and to more easily notice errors.
- Use smart features such as autocomplete.
- Dynamically validate field values. It is frustrating when, after submitting data, you have to go back and correct mistakes. Whenever possible, make the checks as soon as possible so that the user is able to correct them right away.
- Customize the keyboard for the type of query. Display a numeric keyboard when asking for phone number, and include the @ button when asking for an email address. Ensure that this feature is implemented consistently throughout the app, rather than only for certain forms.
- When possible, present choices instead of input fields, since choosing may be simpler than typing.

If a task contains a lot of steps and actions required from the user's side, such as onboarding processes, it's better to divide such tasks into a number of subtasks. This

principle is extremely important in mobile design because you don't want to create too much complexity for the user at one time. This is good both from the data entry point of view, and from the cognitive load reduction point of view.

2.8 Don't make users wait for content

Mobile connections are not stable as hardwired ones, and are not as fast. Therefore, you should always provide content as soon as possible. Bringing the users to a blank screen, shown when content is loading, can make it seem like your app is frozen, resulting in confusion and frustration. In order to avoid this, you should use skeletons, placeholders, etc. There is a vast literature on optimizing image loading on web that you can use to get ideas from in order to avoid this problem.

2.9 Gestures

We interact with mobile devices in different ways than with desktop, we use touch, gestures and voice. A great advantage, from the visual design point of view, of gestures is the lack of physical space to place an input widget. As a result, mobile designers have flirted with the idea of using gestures to increase functionalities without the cost of visual space.

But gestures come at a price: **Gestures are hard to remember and use.** There are several elements that we must take into account when using gestures to interact with the device. Typically, three attributes are used:

- Location where the gesture is initiated.
- Length of the swipe.
- Direction of the swipe.

UI changes are always unpopular with existing users. Thus, moving to a gesture-based interaction control has its dangers. However, big companies such as Apple have seen this as a golden opportunity to maximize the real estate devoted to contents. Eventually, the users will adapt to the new features, and end up learning how to interact with those devices. However, the way that those gestures have been incorporated in OS such Apple's iOS make the learning process slower than needed. For instance, iPhone X and iPhone 8 and older have different meanings for the same gesture (note that iPhone X has no home button and iPhone 8 and older ones still have it). For example swiping down from the top right corner will invoke the Control Center in iPhone X while it will show notifications in iPhone 8.

As a result, using gestures will increase the cognitive load, and make learning process as well as interaction typically slower.

Therefore, if using gestures in your app, try to keep them standard, that is, use the standard ones for analogous tasks in your application. And in the case of providing

non-standard actions, avoid using standard gestures, since this would lead to confusion and increase of complexity.

2.10 Generate continuous integrated experiences

In general, we can see mobile interaction as a part of a more complete experience with an application or service. A typical user will probably interact with the same applications in different environments, and we should design taking this into account. Some tasks such as synchronizing user's current progress may be highly desirable. For example, if a user has started a shopping process on one device, being able to finish it in another is good to provide a seamless experience.

2.11 Don't replicate the web experience on apps

Mobile devices and interfaces have evolved differently than web-based interfaces. As a consequence, users expect certain interaction patterns and interface elements in mobile apps. Do not try to replicate the same web experiences in mobile, not only because they will have usability problems, but also because we have to be consistent with users' expectations.

For the same reason, it is also almost never a savvy option to bring the user to a browser to complete some tasks. If necessary, use an in-app browser. Otherwise, the user may never return to the app.

References

This list of references has been used to generate this document, and provides many important knowledge for mobile UX:

- *10 Do's and Don'ts of Mobile UX Design*, Nick Babich, Adobe Blog, February 2018: <https://theblog.adobe.com/10-dos-donts-mobile-ux-design/>
- *11 Things every designer Needs To Know About Mobile App Interaction*, J. Vino, Medium, <https://medium.muz.li/11-things-every-designer-needs-to-know-about-mobile-app-interaction-a22c635527b3>
- *A Comprehensive Guide To Mobile App Design*, Smashing Magazine, N. Babich, <https://www.smashingmagazine.com/2018/02/comprehensive-guide-to-mobile-app-design/>
- *All You Need to Read to Get Started on Mobile UX Design*, V. Varhan, Medium: <https://medium.muz.li/all-you-need-to-read-to-get-started-on-mobile-ux-design-b482a0d0e7ae>
- *iPhone X: The Rise of Gestures*, R. Budiu, Nielsen Norman Group, <https://www.nngroup.com/articles/iphone-x/>
- *Mobile Login Methods Help Chinese Users Avoid Password Roadblocks*, X. Chen and Y.. Zhou, Nielsen and Norman Group, <https://www.nngroup.com/articles/mobile-login-china/>
- *Mobile User Experience (UX) Design*, Interaction Design Foundation, <https://www.interaction-design.org/literature/topics/mobile-ux-design>
- *The Guide to Mobile App Design: Best Practices for 2018 and Beyond*, N. Babich, UXpin.com, <https://www.uxpin.com/studio/blog/guide-mobile-app-design-best-practices-2018-beyond/>

INTERACTION DESIGN.

SESSION 2

Dept. Computer Science – UPC

OUTLINE

Session 1:

- *Understanding the fundamentals of basic interaction in UI*
 - *Background (Information Theory)*
 - *Hick-Hyman Law: Measuring Choice-Reaction Time*
 - *Fitts' Law: Measuring Pointing Time*
 - *Crossing and Steering Laws: Continuous Gestures*
- *Fitts' Law in UI Design*
 - *Applications in UI Design*
 - *Accelerating Target Acquisition*
- *Exercises*

Session 2:

- Pointing Devices
- Typing & Keyboards
- Mobile Interaction Design
- Exercises

POINTING DEVICES

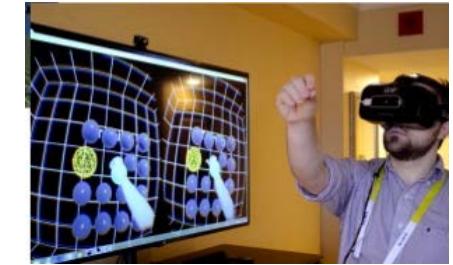
- Direct-control devices:

- Work directly on the surface of the screen
- Direct “touch” in VR



- Indirect-control devices:

- Work away from the surface
- Mapping of the user movement to a pointing element (cursor/ray).



POINTING DEVICES

- **Direct-control devices:**

- Old
 - Lightpen worked back in 1976
- May produce fatigue:
 - Moving the lightpen on the screen required much effort
 - Should have a surface to rest the arm



POINTING DEVICES

- **Direct-control devices. Issues:**

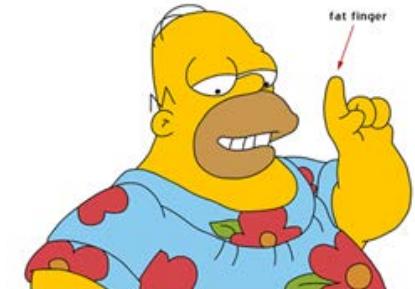
- Imprecision in pointing. Many factors:

- *Quality of the screen:*

- Capacitive screens less precise than resistive

- *Size of the pointer*

- Fat and not-so-fat fingers*

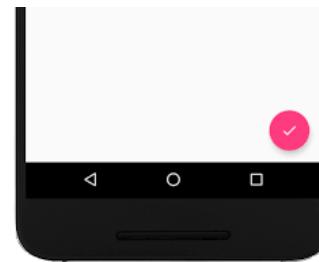


POINTING DEVICES

▪ Direct-control devices. Issues:

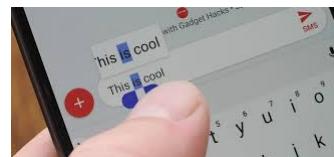
▪ Land-on strategy:

- Select on clicking point
- Faster feedback
- Prone to errors



▪ Lift-off strategy:

- Initial click creates “cursor”, dragging used for precision pointing, lift-off selects
- More time consuming



POINTING DEVICES

- **Direct-control devices.** Advantages:
 - Touch screens can be designed with no moving parts
 - Durable
 - Only device that has survived Walt Disney's theme parks
 - Multi-touch allows for complex data entry or manipulation
 - Pinch-to-zoom gestures



POINTING DEVICES

- **Direct-control devices.** Other issues:
 - Pens may be more suitable for some tasks
 - Reduce occlusion
 - Familiar to users
 - But require to be picked up and put down
 - Pens are more accurate than fingers
 - Fingers are less precise than wrist-based movement

POINTING DEVICES

▪ Indirect-control devices.

- Examples:
 - Mouse, trackball, joystick, graphics tablets...

▪ Issues:

- Alleviate hand fatigue
- Eliminate screen occlusion
- Mouse is the clear king
 - Cost-effective
 - Precise
 - Hand has a surface to rest on
 - Buttons easy to press
 - Long movements require to pick up mouse and replace
 - May be improved using accelerated moves



OUTLINE

Session 1:

- *Understanding the fundamentals of basic interaction in UI*
 - *Background (Information Theory)*
 - *Hick-Hyman Law: Measuring Choice-Reaction Time*
 - *Fitts' Law: Measuring Pointing Time*
 - *Crossing and Steering Laws: Continuous Gestures*
- *Fitts' Law in UI Design*
 - *Applications in UI Design*
 - *Accelerating Target Acquisition*
- *Exercises*

Session 2:

- **Pointing Devices**
- *Typing & Keyboards*
- *Mobile Interaction Design*
- *Exercises*

TYPING & KEYBOARDS. LAYOUTS

- **QWERTY keyboard layout:**

- Design by Christopher Latham Shole.
- The placement of the keys reduces key jams.
- Keys commonly typed together are placed at large physical distance
 - In a typing machine
 - Changing hands
 - Assuming language is English
- Does not make sense with computers
- Not everybody writes in English



https://www.youtube.com/watch?time_continue=3&v=WEyC1NkkR-Q&feature=emb_logo

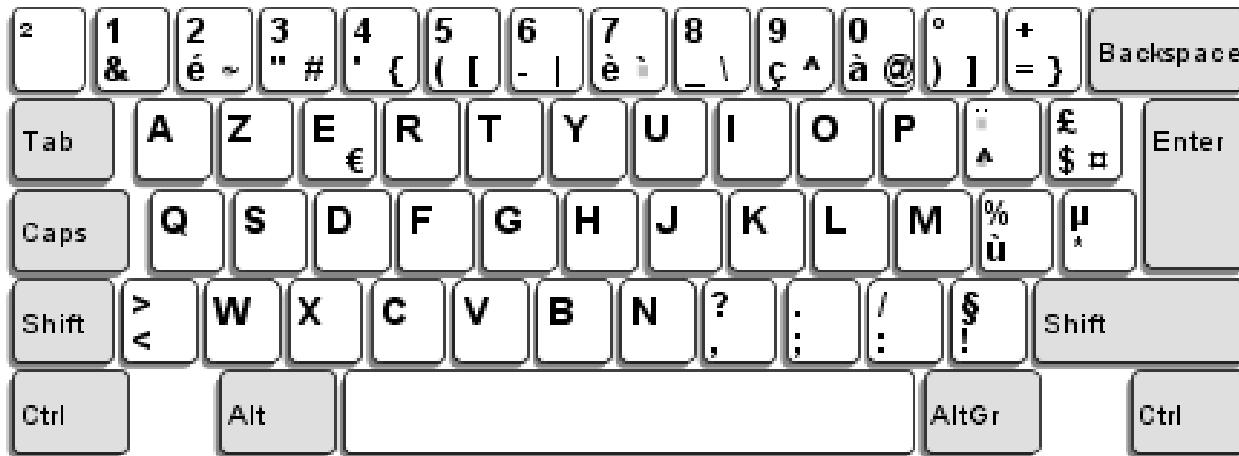
TYPING & KEYBOARDS. LAYOUTS

- QWERTY keyboard layout:



TYPING & KEYBOARDS. LAYOUTS

- Other ergonomic layouts: **AZERTY**

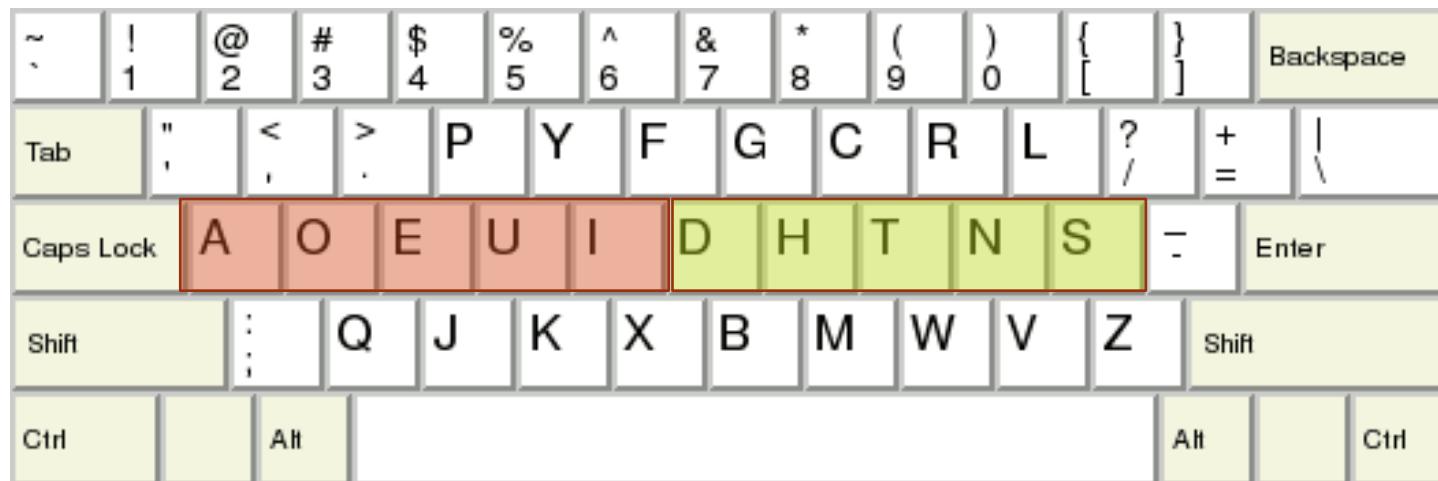


optimized for French

TYPING & KEYBOARDS. LAYOUTS

■ Dvorak layout:

- Vowels in one hand
 - Combinations with consonants impose hand change
- Most common letters at the places the fingers rest on the keyboard



Dvorak

22%

70%

8%

~	!	@	#	\$	%	^	&	*	()	{	}	←	Backspace	
~	1	2	3	4	5	6	7	8	9	0	[]	←	Backspace	
Tab ↪	"	<	>	P	Y	F	G	C	R	L	?	/	+		
,	:	.		E	U	I	D	H	T	N	S	-	=	\	
Caps Lock	A	O											Enter		
Shift ↑			Q	J	K	X	B	M	W	V	Z	Shift ↑			
Ctrl	Win Key	Alt										Alt Gr	Win Key	Menu	Ctrl

QWERTY

52%

32%

16%

~	!	@	#	\$	%	^	&	*	()	-	+	←	Backspace	
~	1	2	3	4	5	6	7	8	9	0	-	=	←	Backspace	
Tab ↪	Q	W	E	R	T	Y	U	I	O	P	{	}			
,	A	S	D	F	G	H	J	K	L	:	"	'	←	Enter	
Caps Lock			Z	X	C	V	B	N	M	<	>	?	Shift ↑		
Shift ↑										,	.	/			
Ctrl	Win Key	Alt										Alt	Win Key	Menu	Ctrl

TECH
QUERO



TYPING & KEYBOARDS. LAYOUTS

■ Dvorak layout:

- Invented with the objective of reducing travel distances
 - 10-finger typing
- Improvements of up to 30%
 - Other researchers say 5-10%
 - Typing Guinness world record held by a Barbara Blackburn with a DVORAK keyboard in a typewriter for many years
 - 150 wpm for 50 minutes
- Less errors
- Also optimized for English
- Low level of acceptance

TYPING & KEYBOARDS. LAYOUTS

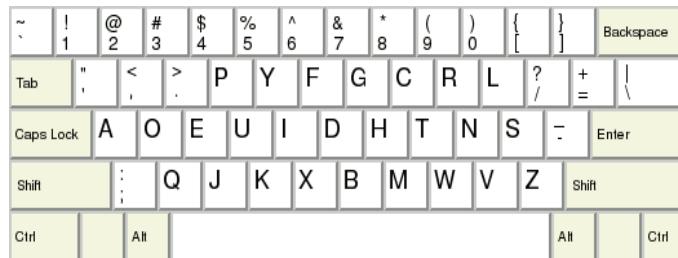
- Keyboard layouts
 - Improves posture and reduces tension
 - No proven advantage



TYPING & KEYBOARDS.

Keyboard arrangements should be designed so that:

1. Balance the loads on the right and left hands
2. Maximize the load on the home row
3. Maximize the frequency of alternating hand sequences
 - Alternating fingers avoids the need to wait for the end of the movement of the first finger before starting the second movement.
4. Minimizing the frequency of same finger typing



Especially good job: 1 & 2



Especially good job: 3

TYPING & KEYBOARDS. PRACTICAL ISSUES

- Experiment with keyboards layouts is difficult
 - Users get their proficiency for practice
 - It requires months of training in any layout
 - The same people would require to be training back to original arrangement for starting a new experiment
- It is commonly accepted formal results based as **predictive human performance model** rather than user testing for evaluation

TYPING & KEYBOARDS. PRACTICAL ISSUES



Source: <http://minuum.com/>

- Touchable layouts (some issues)
 - Size depends on screen size
 - Limited and occluded text
 - Require significant visual attention
 - No physical feedback. Sometimes sound
 - Distance from the keyboard to the insertion point
 - Especially on larger form factors
 - Errors: accidentally touching the screen
 - Touch and stylus based may be a good combined with stroke gestures or other ideas...

TYPING & KEYBOARDS. PRACTICAL ISSUES

- Expert typing model [Bi2013]:
 - Based on Fitts' Law
 - Time to move the tapping device with a single finger from one key (i) to another (j) depends on the distance and the width of the keys:

$$MT_{ij} = a + b \log_2 \left(\frac{D_{ij}}{W_{ij}} + 1 \right)$$

- D_{ij} is the distance between keys i and j ,
- W_{ij} is the width of each key
- Bi et al. also use the effective width

TYPING & KEYBOARDS. PRACTICAL ISSUES

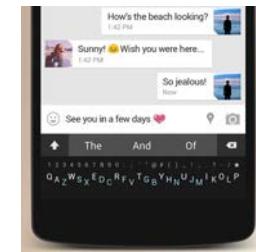
Fitts Law accurately predicts pointing movement

- If improvement required, it can help us modify our UI
 - Change target width:
 - Increase size for faster reach
 - Change distance:
 - Move targets closer to reduce movement time
 - Change pointer movement:
 - Increase speed

$$MT_{ij} = a + b \log_2 \left(\frac{D_{ij}}{W_{ij}} + 1 \right)$$

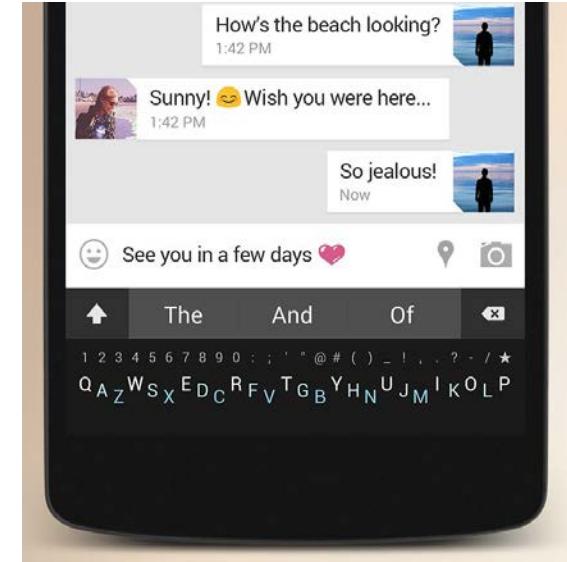
TYPING & KEYBOARDS. PRACTICAL ISSUES

- Improving mobile layouts:
 - Different parameters to take into account:
 - 10-finger typing? As of tablets
 - 2-thumb typing? Mobiles/tablets.
 - 1-finger typing? Most commonly mobile
- Optimize for the number of fingers
 - Tactile screen form factor
 - Maybe hand positions too

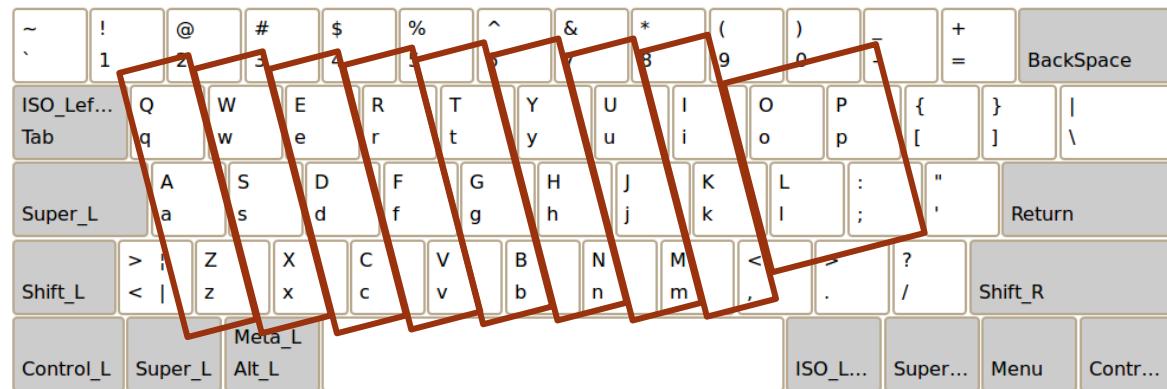


TYPING & KEYBOARDS. MOBILE LAYOUTS

- Proposed mobile layouts. **Minuum:**
 - **Two or one finger typing**
 - Compressing the three key rows into one
 - Reduction of distances (in vertical)
 - Larger targets
(the whole region of e. g. QAZ)
 - Proficient word prediction/correction is required
- More room in your screen



A new (old) idea?



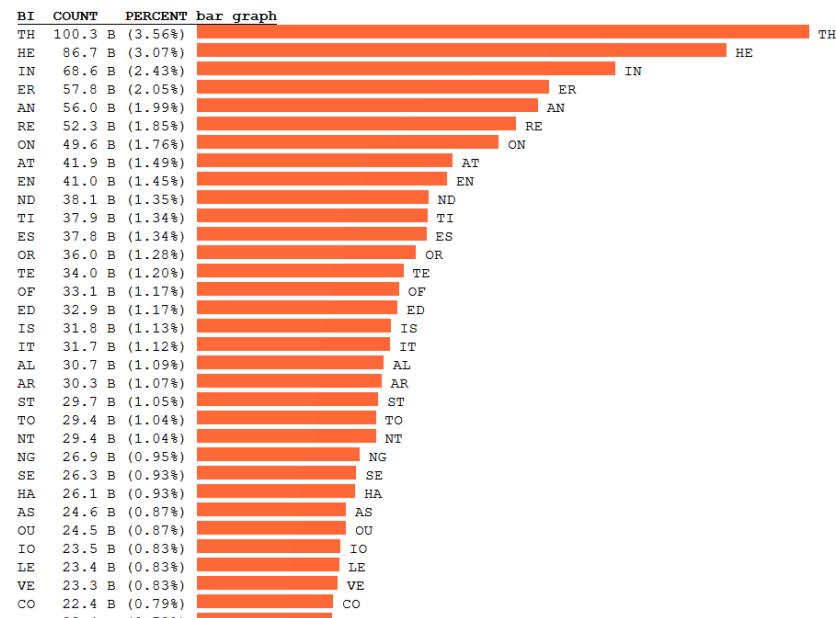
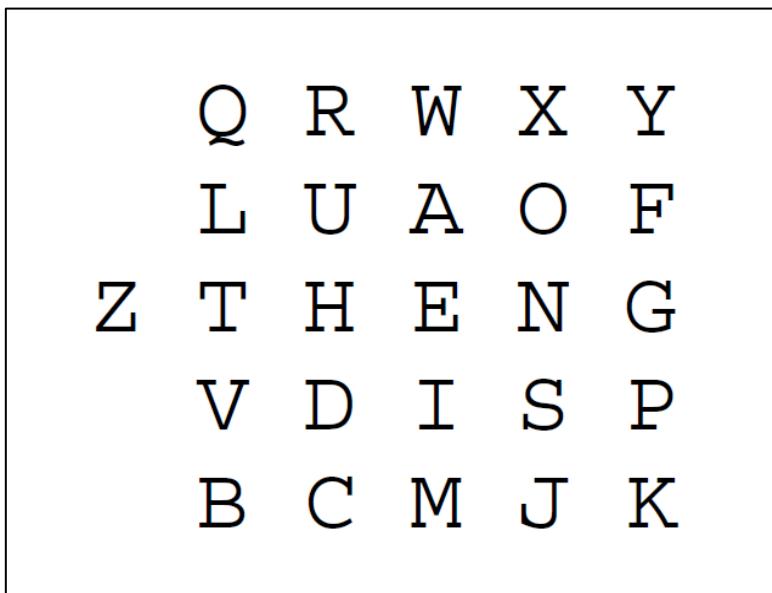
TYPING & KEYBOARDS. MOBILE LAYOUTS

- Minuum is intended to type everywhere:



TYPING & KEYBOARDS. MOBILE LAYOUTS

- Diagram-based layout for single-finger typing [Lewis99]:
 - Optimized distances
 - Up to 25 wpm (over the typical 20 wpm on a complete QWERTY)

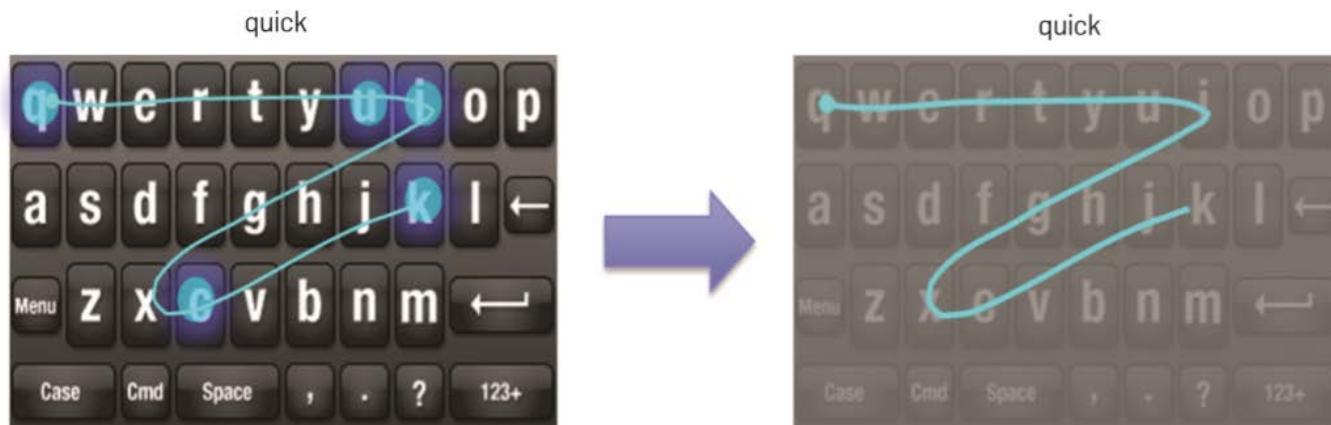


Source: norvig.com

TYPING & KEYBOARDS. MOBILE LAYOUTS

- **Single finger gesture typing** [Kristensson2012, Zhai2012]

- The finger traverses all the letters of a word without lifting off the screen
- More comfortable (subjective evaluation) in tablets [Nguyen2012]
- Not faster than regular typing (objective evaluation) in tablets [Nguyen2012]. Not so negative



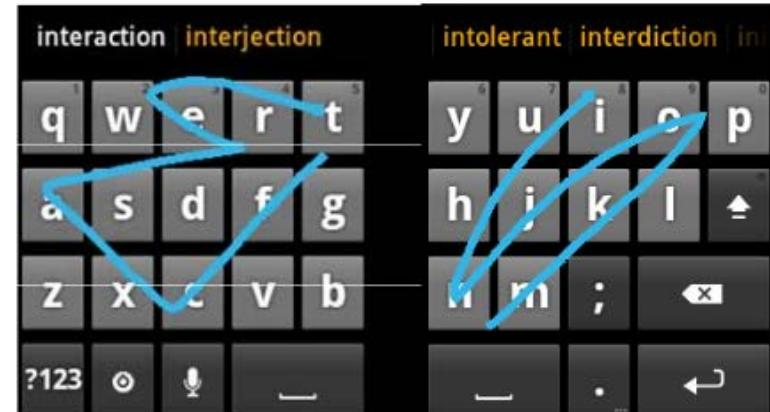
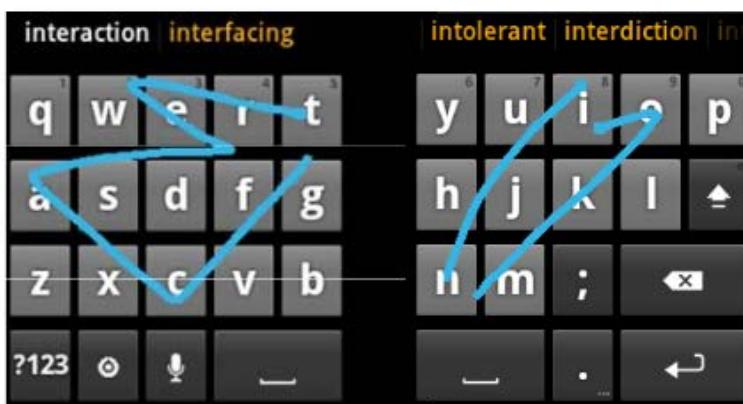
TYPING & KEYBOARDS. MOBILE LAYOUTS

- Proposed mobile layouts. **KALQ**:
 - Optimize layout for better **2 thumb typing**
 - Analyzed hand position, diagram frequency, tablet orientation...



TYPING & KEYBOARDS. MOBILE LAYOUTS

- **Two finger** gesture typing [Bi2012]
 - The two thumbs swipe to compose a word
 - Lifting the finger when a part of the word belongs to the other thumb
 - Or with a continuous trace
 - Finger traveling shortened by 50%
 - Speed does not increase over one finger entry (objective evaluation). Not so negative
 - High demand of attention (subjective evaluation)



TYPING & KEYBOARDS. PRACTICAL ISSUES

Designing virtual keyboards. Elements to consider for usability:

- Auto-correction
- Auto-capitalization
- Input data type & custom keyboards
- (Multiple-)Language support

TYPING & KEYBOARDS. PRACTICAL ISSUES

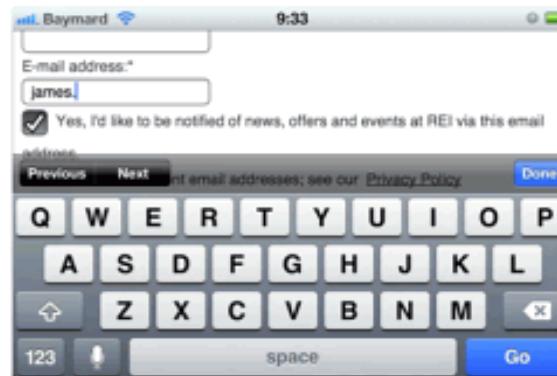
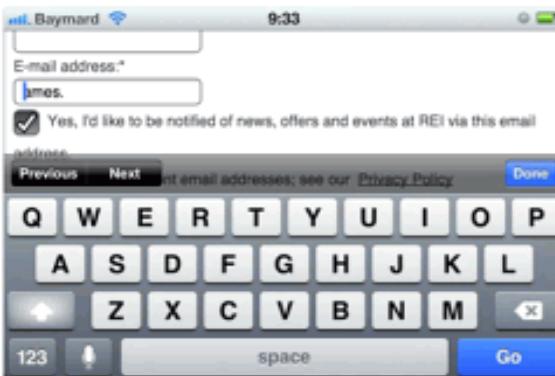
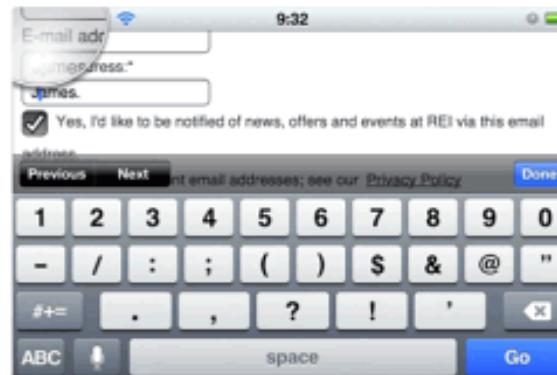
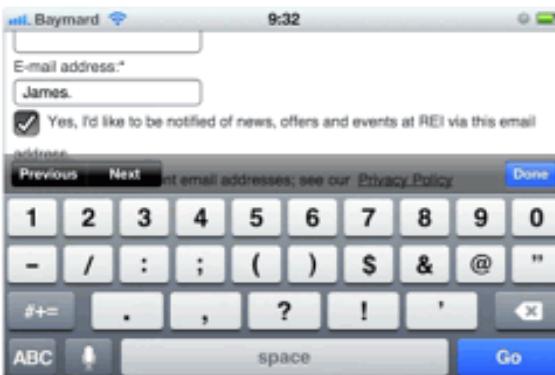
1. Auto-correction:

- Only suitable if proper dictionaries:
 - Commonly, users do not notice the corrections
 - Some data such as address very prone to wrong correction
 - 92% sites do it wrong
- Best practices:
 - Skip auto-correction for certain fields
 - Usually, it is safer to opt for a predictive approach and let the user to choose the best option.

TYPING & KEYBOARDS. PRACTICAL ISSUES

2. Auto-capitalization:

- In e-mail addresses, disable auto-capitalization
 - Even if correct, people tries to fix



TYPING & KEYBOARDS. PRACTICAL ISSUES

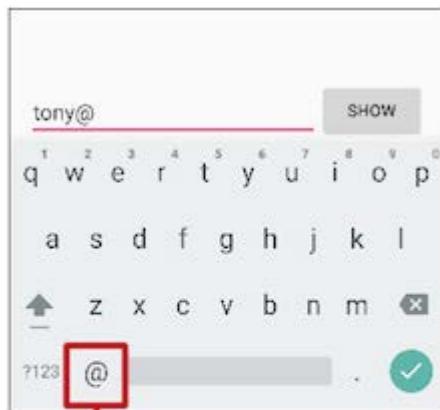
3. Appropriate layouts for the input data type:

- Virtual keyboards are small
 - An iPhone 4 character (portrait) measures 4×5.9 mm
 - Minimum recommended clickable size is 6.85×6.85 mm
 - Increase typos, validation errors...
 - 60% top mobile websites do it wrong
- Dedicated keyboards may increase the size enough
(phone numbers, ZIP codes, currency...)
 - Invoke them, and **do it consistently**

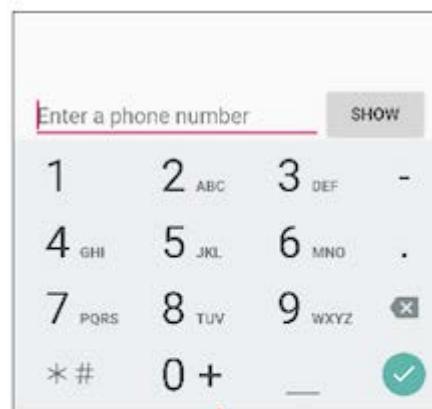
TYPING & KEYBOARDS. PRACTICAL ISSUES

Dedicated keyboards examples (space gain):

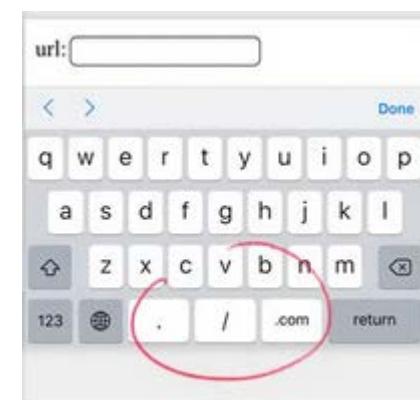
Email address



Phone number



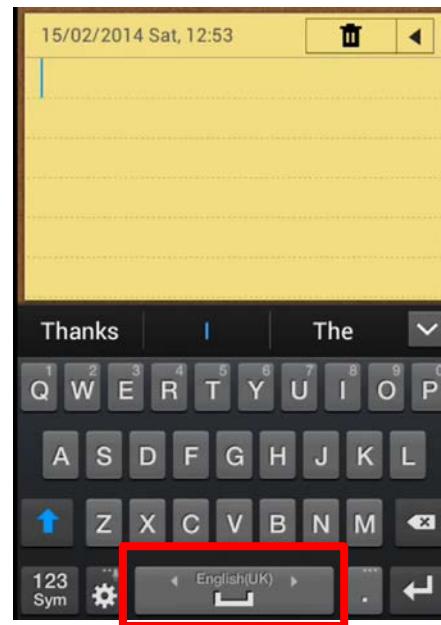
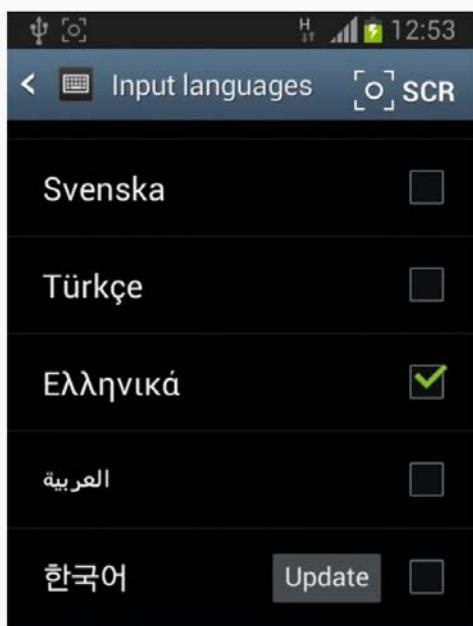
URL



TYPING & KEYBOARDS. PRACTICAL ISSUES

4. (Multiple-)Language support:

- Most custom keyboards provide the possibility of changing the language on demand
 - In many cases correctors or word predictions mix languages



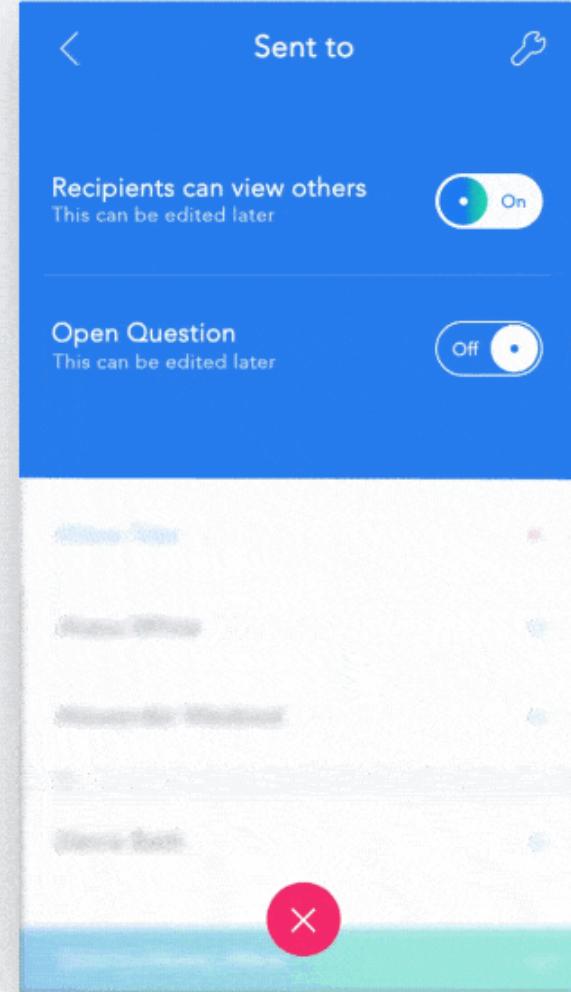
INTERACTION DESIGN AND EVALUATION. SESSION 1

Dept. Computer Science – UPC

IDI – Mobile Interaction Design



Professors IDI
ViRVIG Group – UPC



Motivation

Mobile devices have different requirements for design:

- More personal
- The environment where users use them competes for their attention
- Entering data is difficult
- Small screen sizes

Motivation

Desired features for mobile UIs:

- Quick find what they intend to
- Minimum cognitive load for interaction
- Information presented in small chunks

User Interface and Interaction Design different from desktop

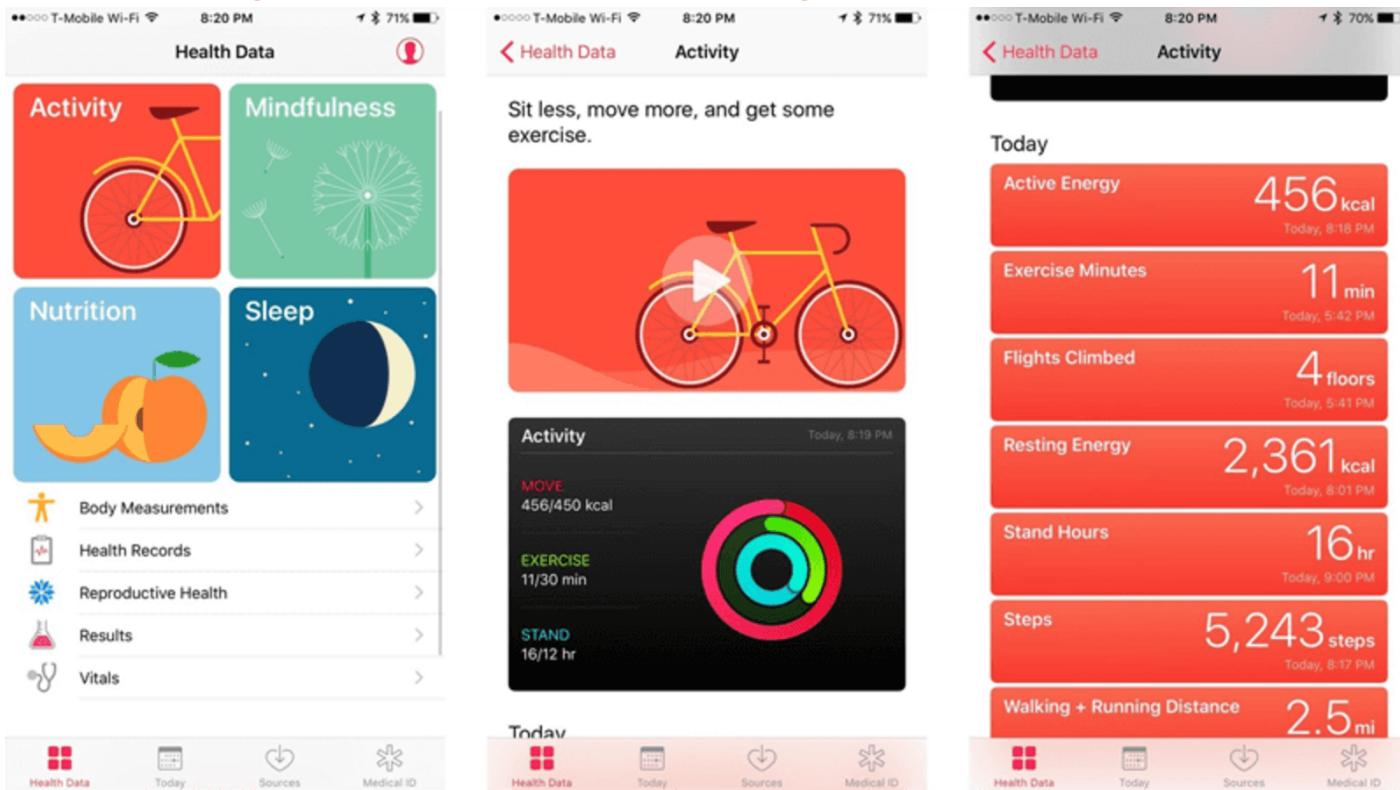
Design Guidelines

Keep navigation simple

- Ensure navigation feels familiar
- Design good information architecture
- Navigation should not grab user attention
- Ensure users know their location
- Strive for consistency
- Clear path to objectives
- Clear visual hierarchy

Design Guidelines

Keep navigation simple: Communicating the current section of the app



Design Guidelines

Finger-friendly tap targets

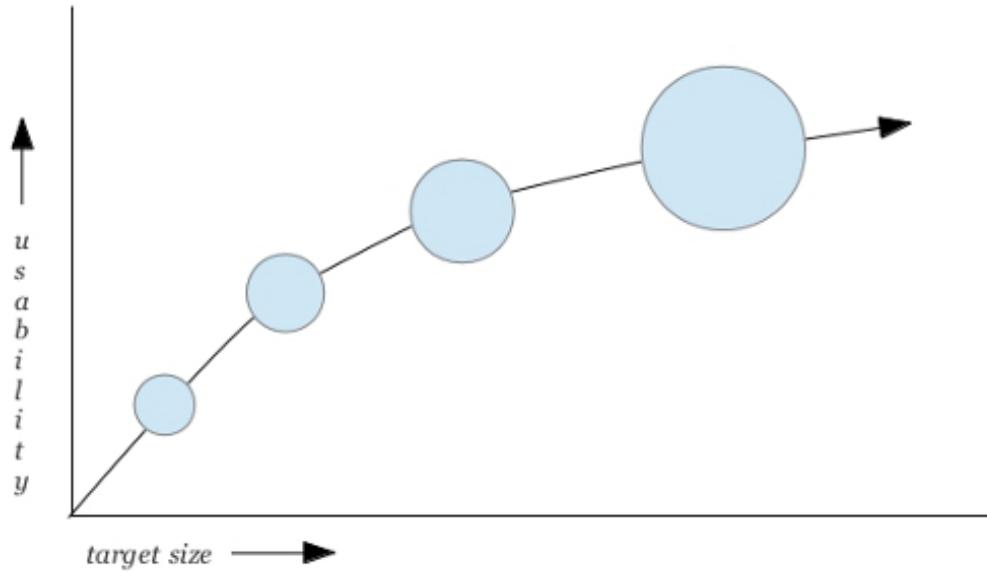
- Around 10x10mm minimum
- Keep good spacing between elements



Design Guidelines

Finger-friendly tap targets

Predicted usability of a button according to its size



Design Guidelines

Finger-friendly tap targets

For mobile take into account the thumb zones

- Consider Fitts only within the operation range of the thumb
- Outside elements require extra effort



Design Guidelines

Progressive disclosure and cognitive load reduction

Cognitive load: amount of brain power required to use the app

- Keep amounts of information (required to remember) low
- Progressively show new features or tasks
- Helps simplifying UI

Design Guidelines



Design Guidelines

Make text legible

- Choose typeface that works well in multiple sizes and weights
- Use legible font sizes: at least 11 points
- Use adequate contrast
- Correct vocabulary

Design Guidelines

Make text legible

Heading

Sub-Headline

Adipiscing elit. Sed neque nisl, blandit vel ipsum eu, imperdiet blandit lectus. Morbi tristique urna ut volutpat ornare. Curabitur semper vitae urna ac tempus. Duis vehicula elit nulla, eleifend egestas nisl vehicula nec. Nullam varius est dui, nec accumsan lectus posuere ut. Nullam viverra purus laoreet euismod tempor.

Adipiscing elit. Sed neque nisl, blandit vel ipsum eu, imperdiet blandit lectus. Morbi tristique urna ut volutpat ornare. Curabitur semper vitae urna ac tempus. Duis vehicula elit nulla, eleifend egestas nisl vehicula nec. Nullam varius est dui, nec accumsan lectus posuere ut. Nullam viverra purus laoreet euismod tempor.

Adipiscing elit. Sed neque nisl, blandit vel ipsum eu, imperdiet blandit lectus. Morbi tristique urna ut volutpat ornare. Curabitur semper vitae urna ac tempus. Duis vehicula elit nulla, eleifend egestas nisl vehicula nec. Nullam varius est dui, nec accumsan lectus posuere ut. Nullam viverra purus laoreet euismod tempor.



Heading

Sub-Headline

Adipiscing elit. Sed neque nisl, blandit vel ipsum eu, imperdiet blandit lectus. Morbi tristique urna ut volutpat ornare. Curabitur semper vitae urna ac tempus. Duis vehicula elit nulla, eleifend egestas nisl vehicula nec. Nullam varius est dui, nec accumsan lectus posuere ut. Nullam viverra purus laoreet euismod tempor.

Adipiscing elit. Sed neque nisl, blandit vel ipsum eu, imperdiet blandit lectus. Morbi tristique urna ut volutpat ornare. Curabitur semper vitae urna ac tempus. Duis vehicula elit nulla eleifend.

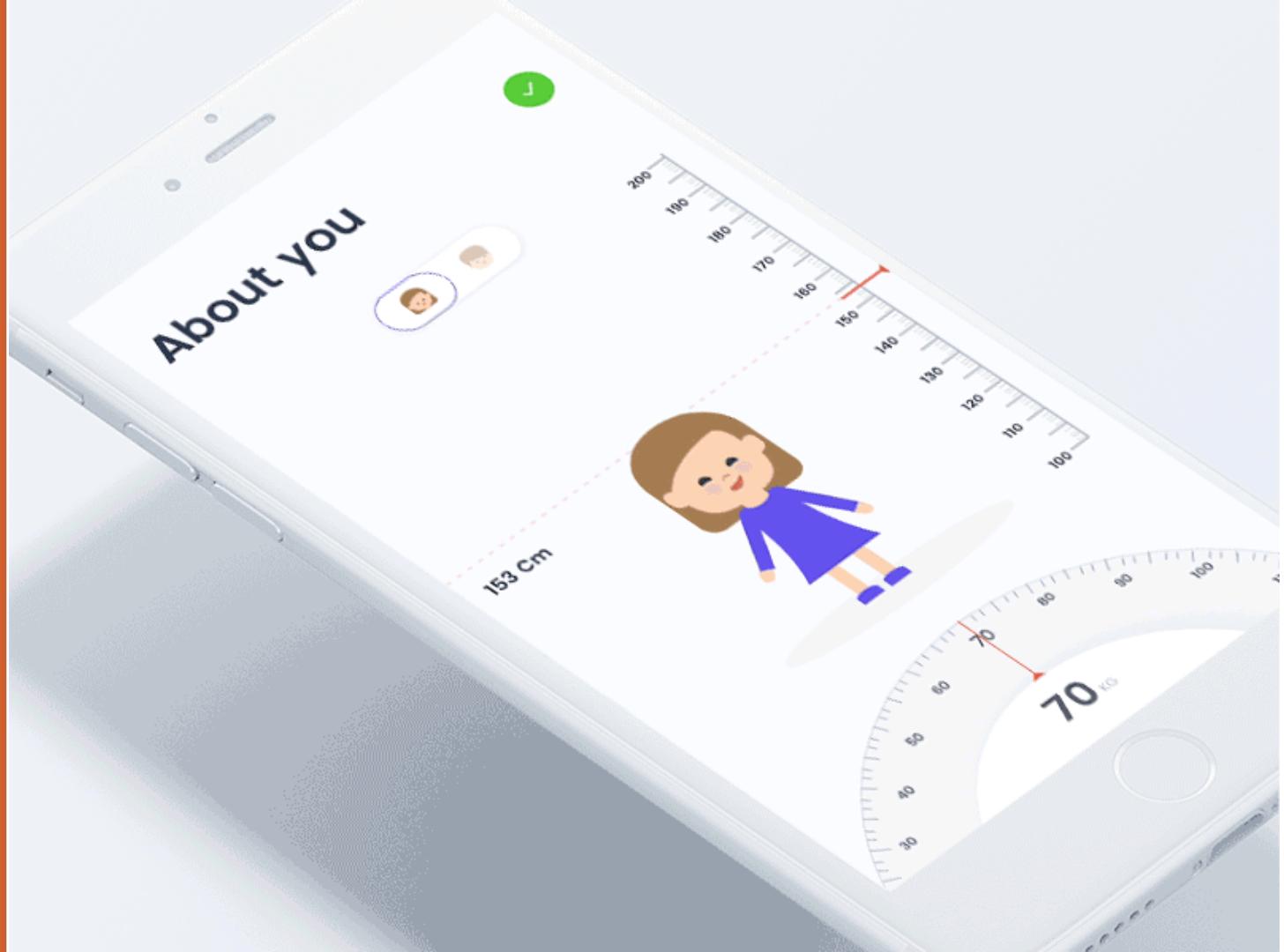


Design Guidelines

Provide feedback on interactions

- Use microinteractions if possible
- Add progress indicators when required

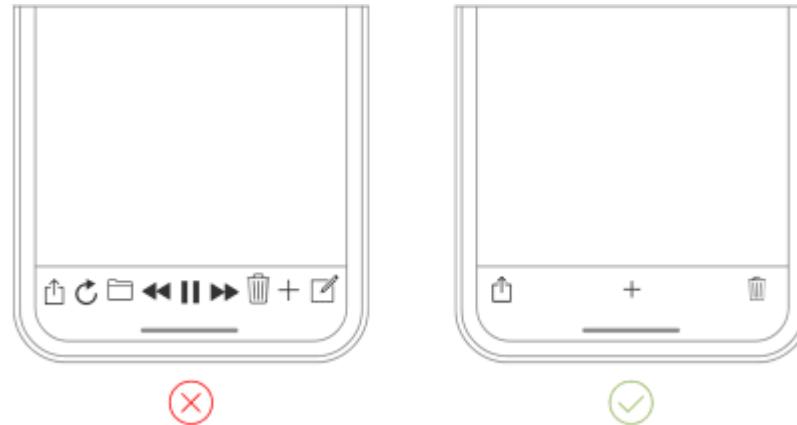
Design Guidelines



Design Guidelines

Reduce clutter

- Keep content to a minimum
- Keep interface elements to a minimum
- Alternatively, use progressive discovery
- Strive for minimalism



Design Guidelines

Reduce user inputs

Simplify procedures: onboarding, logon...

Onboarding:

- Break in multiple steps
- Delay information retrieval
- Inform properly on the needs

Logon:

- Use one-time passwords or QRs when possible

Design Guidelines

Reduce user inputs: recommendations

- Keep forms as short as possible
- Provide input masks
- Use smart features such as autocomplete
- Dynamically validate field values
- Customize the keyboard for the type of entry
- When possible, substitute text entry for options

Design Guidelines



Design Guidelines

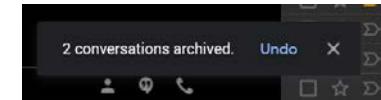
Manage friction

Some alternative to increase the size that improve usability:

- Visual stimulus, undo,..

Some “editing” actions must be dealt with care (send, upload, download, burn, share):

- Possibility of undoing (even temporarily)
 - E. g. Google's mail
- Highlight relevant elements
 - E.g. Call To Action buttons (they guide users towards your goal conversion)

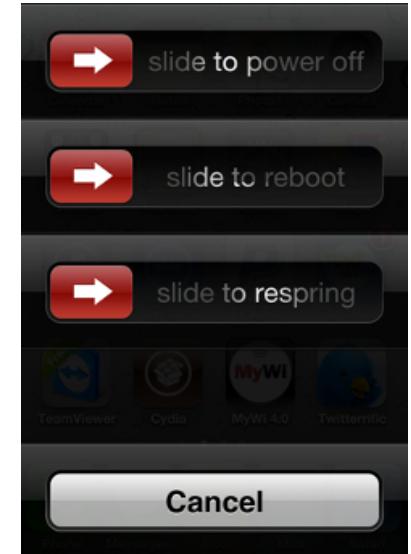


Design Guidelines

Manage friction

Design with friction to avoid mistakes. Rule of the thumb:

- Make destructive/delicate tasks more difficult
- Increasing the effort to prevent accidents
 - Buttons for non-destructive
 - Slides for destructive



Design Guidelines

Don't make users wait for content

- Mobile connections are not stable: don't present blank pages to the user
- Use skeletons, lower resolution images...
- Update as soon as possible



Design Guidelines

Use gestures prudently

Gestures can save space: they do not require visual representations

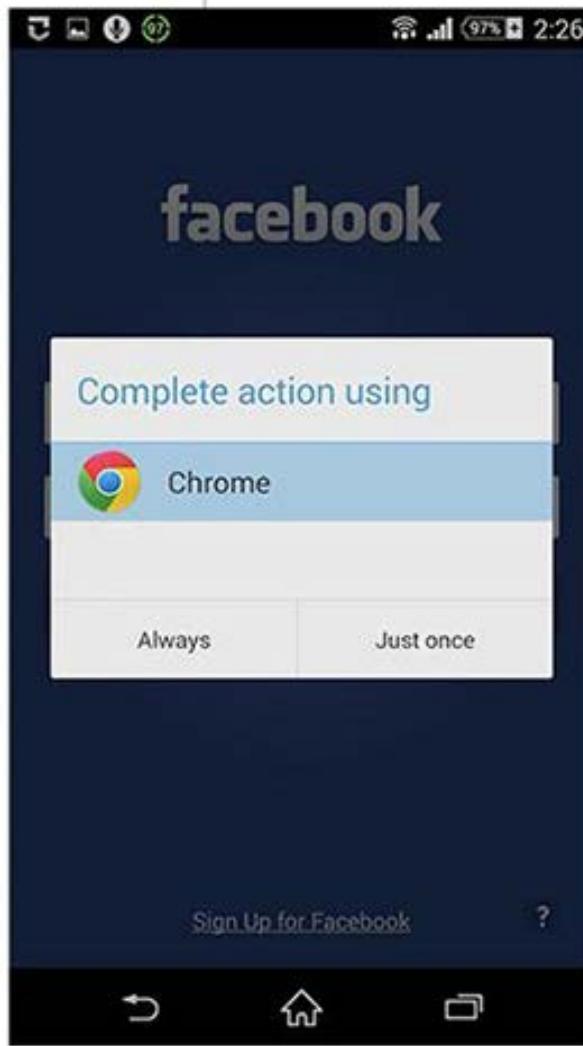
- Hard to remember and use
- Not currently standarized
- Make use of standard gestures
- Don't use standard gestures for non-standard tasks

Design Guidelines

Continuous integrated experience

- When possible, synchronize app with desktop interaction (tasks can be continued on different devices)
- Do not replicate exact (web) experience on mobile
- Be consistent with users' expectations: in terms of visual elements, interactions...
- Don't open external web browsers to complete tasks
- Don't create dead end pages

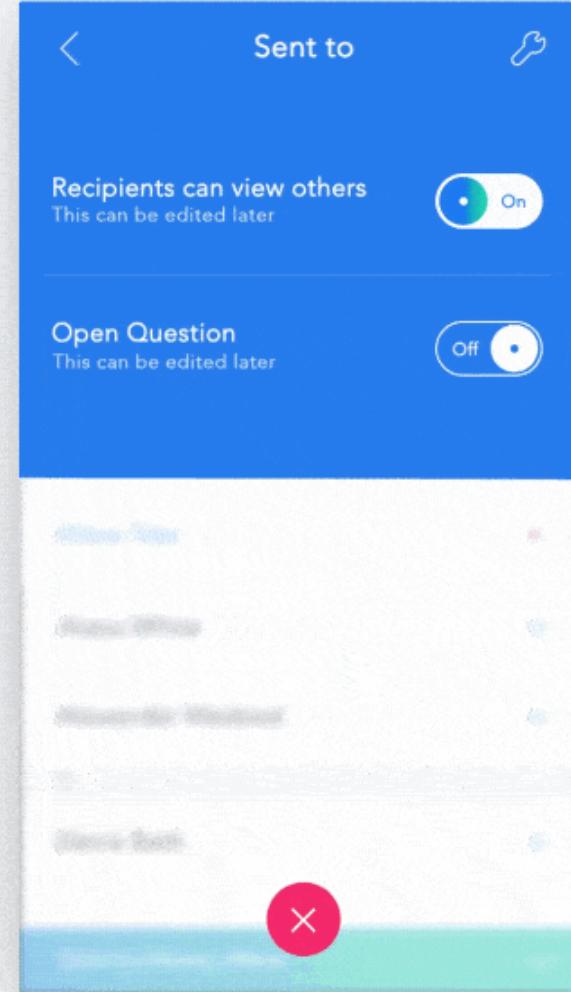
Design Guidelines



IDI – Mobile Interaction Design



Professors IDI
ViRVIG Group – UPC



Interaction Design and Evaluation

Pere-Pau Vázquez – Dep. Computer Science, UPC

Contents

<u>1 INTRODUCTION</u>	1
<u>2 BACKGROUND</u>	1
2.1 INFORMATION THEORY	1
2.2 INFORMATION AND UNCERTAINTY	2
2.3 SHANNON ENTROPY	4
<u>3 CHOICE-REACTION TIME</u>	4
3.1 HICK-HYMAN LAW	4
3.2 EVIDENCES OF HICK-HYMAN LAW	5
<u>4 FITTS' LAW – LAW OF POINTING</u>	6
4.1 FORMULATION	6
4.1.1 INITIAL RESEARCH	6
4.1.2 REFINEMENTS TO FITTS' LAW	7
4.1.3 IMPLICATIONS OF FITTS' LAW IN UI DESIGN	8
4.2 EXAMPLE	10
4.3 FITTS' LAW EXTENSIONS	10
4.3.1 FITTS LAW FOR 2D	10
4.3.2 FITTS FOR FINGER TOUCH	11
4.4 FITTS' LAW IN KEYBOARDS	12
4.4.1 KEYBOARD LAYOUTS	12
4.4.2 WHY EXPERIMENTING WITH KEYBOARD LAYOUTS IS DIFFICULT	13
4.4.3 TOUCH-SCREEN KEYBOARDS	14
4.4.4 DIGRAM-BASED KEYBOARDS FOR SINGLE-FINGER TYPING	15
4.4.5 SWIPE-BASED KEYBOARDS	16
4.5. EVIDENCES OF FITTS LAW	17
4.6 FITTS VARIANTS AND LIMITATIONS	18
<u>5 APPLICATIONS OF FITTS' LAW IN INTERFACE DESIGN</u>	19
5.1 USE OF SCREEN EDGES	19
5.2 PLACING RELATED THINGS CLOSE	21
5.3 COMPLICATE THE ACCESS TO ELEMENTS	22
5.4 PIE MENUS	22
<u>6 LAW OF CROSSING</u>	23
6.1 INTRODUCTION	23
6.2 CROSSING CONFIGURATIONS	25
6.2.1 CONTINUOUS VS DISCRETE	25
6.2.2 COLLINEAR VS ORTHOGONAL	25
<u>7 LAW OF STEERING</u>	26

7.1 INTRODUCTION	26
7.2 STEERING THROUGH STRAIGHT PATHS	27
7.3 EVALUATIONS OF STEERING LAW	27
<u>8 ACCELERATING TARGET ACQUISITION</u>	<u>29</u>
8.1 INTRODUCTION	29
8.2 MODIFICATION OF TARGET WIDTH	29
8.2.1 TARGET AND SCREEN SIZE	30
8.2.2 EXPANDING TARGETS	30
8.2.3 BUBBLE TARGETS	31
8.3. INCREASING CURSOR SIZE	32
8.4 TARGET MOVING AND OTHER TECHNIQUES	34
8.4.1 TARGET MOVING	34
8.4.2 OTHER TECHNIQUES	35
8.5 DISCUSSION	36
8.6 CONTROL DISPLAY RATIO	36
8.6.1 CONCEPT	36
8.6.2 CONTROL-DISPLAY RATIO ADAPTATION TECHNIQUES	36
<u>9 POINTING TASKS</u>	<u>37</u>
9.1 POINTING DEVICES	38
9.1.1 DIRECT-CONTROL DEVICES	38
9.1.2 INDIRECT-CONTROL DEVICES	41
9.2 COMPARISON OF POINTING DEVICES	41
<u>10 3D SELECTION</u>	<u>42</u>
10.1 INTRODUCTION	42
10.2 DEFINITIONS	43
10.3 3D SELECTION	43
7.3.1 3D SELECTION TECHNIQUES	43
10.3.2 RAY-BASED SELECTION AND POINTING	44
<u>11 BIBLIOGRAPHY</u>	<u>47</u>

1 Introduction

Successful interface design usually leads to positive feelings and enthusiasm among users. An enthusiastic user will likely recommend the application to another user. He or she will also be satisfied with his or her competence in achieving the tasks, the knowledge of the interface, and will enjoy its use.

Such satisfying interfaces are also referred to as *direct-manipulation interfaces*. They provide visible support for the actions of interest, that is, objects and actions are carried out with visual feedback. Moreover, they also provide rapid, reversible, and incremental actions. Typed commands are replaced by pointing actions on the object of interest.

In order to provide the best service to our users, we need to design the interaction properly. Some tasks will be easily accomplished by direct manipulation, such as when editing images, but other tasks, such as aligning objects in an image can be easier through a menu option than moving the objects directly. Therefore, we must properly select the adequate interaction for each task.

Pointing and selection are two related fundamental tasks in graphical user interfaces. Object selection techniques involving physical interaction are constrained by the human motor system as the speed and the accuracy of any gesture are limited by the nervous and muscular systems. In order to understand how we process information and interact with elements in the interfaces, it is very useful to know which laws seem to govern our behavior (reading capacity, clicking time and effort...). Getting a knowledge into these laws will make possible to analyze and to design better user interfaces.

The objective of this document is to present both the Hick-Hyman law and the three “Laws of Action”, as defined by some researchers (e. g. Zhai), as well as to point some guidelines to use them in effective user interface design.

2 Background

Pointing and selection tasks have been studied thoroughly from a theoretical point of view. We will present here the theories that try to give an explanation on users' performance on pointing and selection tasks. Since all of them are based on a previous mathematically grounded theory: *Information Theory*, we will give first some insights on it.

2.1 Information Theory

Information Theory was born as theory that dealt with data communication. In 1948 Claude E. Shannon presented his seminal work *A Mathematical Theory of Communication* [Shannon48]. Based on previous works by Nyquist and Hartley [Nyquist24, Hartley28] on the transmission of electrical signals for telegraphic communication, he developed a measure named Shannon Entropy that measures the amount of information to be transmitted by a message. Shannon's approach to communication uses the following elements:

2 | Interaction Design and Evaluation

- **Information source:** The element that produces a message or sequences of message.
- **Transmitter:** Operates on the message to make it transmissible through a medium.
- **Channel:** The medium that transmits the message.
- **Receiver:** The element that reconstruct the message to the destination.

Shannon's system was a telegraph. In this context, the message is encoded into a signal and this signal transmitted through the channel. The receiver transforms back the signal into a message and delivers it.

The general scheme of this communication process appears in Figure 1.

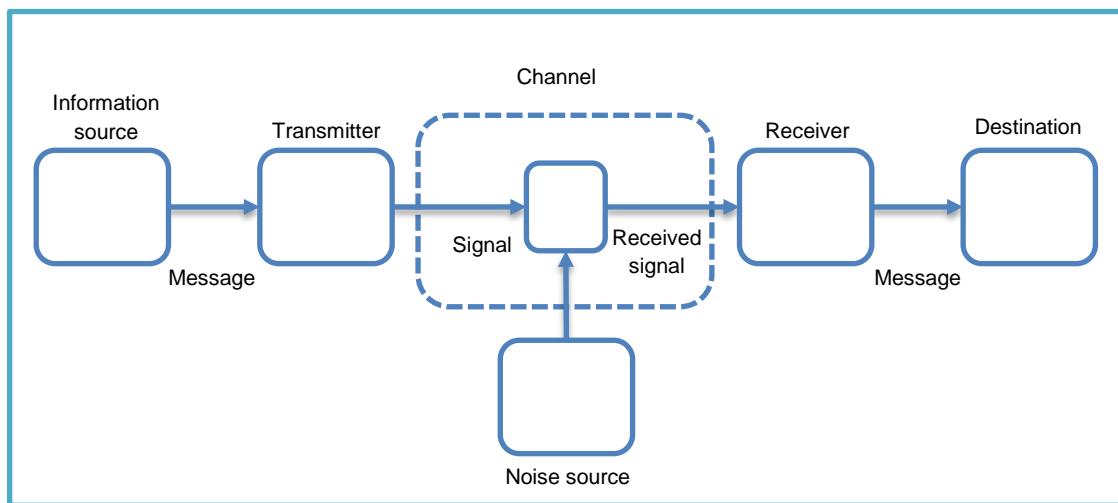


Figure 1: Diagram of a general model of communication system.

The amount of information that a communication channel transmits in a fixed amount of time is referred to as the channel capacity. Channels are bound by physical limitations and thus have different capacities. Other factors, such as noise, affect the communication channel. Thus, the effective capacity is always lower than are the theoretical specifications.

2.2 Information and Uncertainty

Information and Uncertainty are two terms that appear commonly together in Information Theory, so they need a deeper discussion. The first thing to take into account is that those two terms are used to describe the process of selecting one or more elements from a set of elements.

The following development is taken from Schneider's "Information Theory Primer" [Schneider2013].

Suppose we have a device that generates any of these 3 symbols: A, B, or C, every time we ask for one symbol. Before the symbol is produced, we do not know which is the one to come. We can say that we are *uncertain* on the next symbol. Once a symbol has appeared, since we know which of the symbols has appeared, our uncertainty decreases. It decreases because we *have received information*. That is, **information is a decrease in uncertainty**.

Then it comes the problem of measuring this uncertainty. Without many details on how they appear, we may say that a device that produces three different symbols has an uncertainty of $\log(3)$. We will discuss the units later. Then, if we have a second device that produces two different symbols (say 1 and 2), we will say that it has an uncertainty of $\log(2)$. We can make the following experiment: we iteratively produce elements from the first and the second device and combine both. If we do this, the number of different (combined) elements we will have will be six: A1, A2, B1, B2, C1, and C2. Note that this implies an uncertainty of $\log(6)$. Intuitively, when we combine the two devices, we would expect the uncertainty to increase. Actually, if we add the uncertainties of the both original devices we will have $\log(3)+\log(2) = \log(6)$. So if we use logarithms to encode uncertainty, we can add it as we would intuitively expect.

This can be applied with any logarithm, but the typically used here are logarithms with base 2. If we use such logarithms, the value measured by the uncertainty will be *bits* [Shannon54].

Thus if a device produces one single symbol, we are uncertain by $\log_2 1 = 0$ bits. This means that we have **no uncertainty** about what the device will do next, since it will produce the symbol we expect. If, on the contrary, we have a device that produces two symbols, the uncertainty would be $\log_2 2 = 1$ bit.

So far, we have assumed that our devices produce symbols with equal probability, and in that case, the uncertainty is $\log_2(M)$, where M is the number of different symbols a device may produce. If we have M symbols and the probability of producing any symbol is the same, we can say that the probability we have is $1/M$. Then, we can change the logarithm in the following way:

$$\log_2(M) = \log_2\left(\left(\frac{1}{M}\right)^{-1}\right) = \log_2(P)^{-1} = -\log_2(P)$$

That is, the uncertainty of getting a certain value is the $-\log$ of the probability of this symbol. This value, also called the *surprise* (or *surprisal*) indicates whether we may or not expect a certain value. If we are *certain* on a certain symbol, there will be no *surprise* when it emerges. If we recall the previous example, we have total certainty when the set of symbols is composed by a single element. In that case, the surprise ($\log_2(P)$) will be zero, as can be seen when we calculate it: $\log_2(1) = 0$.

For the different symbols that can be produced, we have that their probability sums up to 1:

$$\sum_{i=1}^M P_i = 1$$

This can be used to analyze the common case where the probability of producing different symbols varies. That is, some symbols appear more often than others. In this case, we may have that probability values are not all $1/M$, but this value can change (provided that the total sum adds up to 1). In that case, we will be more *surprised* to see a low probability symbol than seeing a higher probability symbol.

2.3 Shannon Entropy

As said, information is defined in Information Theory as a reduction in uncertainty. And the purpose here is to measure the information.

The classical measure of information, as presented by Shannon is measured over a distribution of probabilities. It is defined as the average *surprisal* from a set of symbols produced by a device. If we measure the surprise for the infinite set of symbols that can be produced by a device, the frequency of each symbol transforms to the probability. Then, if we sum over all the symbols set, we will get:

$$H = \sum_{i=1}^n p_i \log_2 \left(\frac{1}{p_i} \right) = - \sum_{i=1}^n p_i \log_2 p_i$$

where n is the number of alternatives, and p_i is the probability of the i th alternative (here I changed P by p since it is the common notation found in literature). H is the entropy of the message that is to be transmitted. It is actually the amount of information expected to be received at the destination. It is also designated as $H(x)$. Since there is usually an interference during transmission, the information actually received in destination is reduced, and it is designated $H(y)$. The average information that is faithfully transmitted is calculated as:

$$R = H(x) - H_y(x)$$

where $H_y(x)$ is the *equivocation*, or the conditional entropy of x when y is known. As we will see later, in Hick's and Fitts's laws, when a participant commits no errors, he or she is said to be extracting all the expected information of the stimuli.

3 Choice-reaction time

3.1 Hick-Hyman Law

William E. Hick was one of the first to apply Information Theory to psychological problems. His theory had as objective to measure the relationship between choice reaction time and stimulus information content (entropy). It has been previously discovered, as early as 1868 [Donders68] and Merkel [Merkel85], that it takes longer to respond to a stimulus when it belongs to a large set as opposed to a smaller set of stimuli. This regularity caught the attention of psychologists, who saw its analogy to the classic Information Theory.

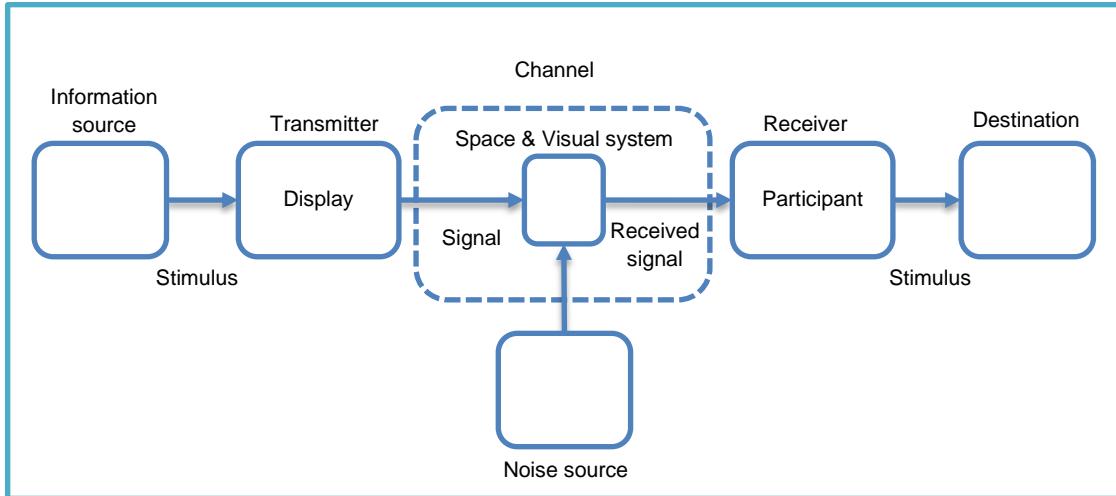


Figure 2: Diagram for the choice-reaction time experiment as a model of communication system.

Hick's Law describes human decision time (or *Reaction Time*) as a function of the information content conveyed by a visual stimulus. After his first experiments [Hick51, Hick52], Ray Hyman [Hyman53] extended his work, and nowadays this law is also called Hick-Hyman Law [Seow2005]. It is defined, in its most general way, as:

$$RT = a + b H_T$$

where a and b are empirically determined constants, and H_T is the transmitted information. The transmitted information is commonly fitted as

$$H_T = \log_2(n + 1)$$

where n is the number of equally probable alternatives and the $+1$ indicates the uncertainty about whether to respond or not, as well as about which response to make [Card83].

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.

3.2 Evidences of Hick-Hyman law

Hick-Hyman's Law has been demonstrated in many experiments, and has been used to justify menu design. For instance, Cockburn *et al.* [Cockburn2007] and Cockburn and Gutwin [Cockburn2008] show that novice users tend to select with a visual search that takes linear time on the number of elements of a menu, while experts decide upon item location, and their decisions times are adequately predicted by Hick-Hyman's logarithmic formula. Allison *et al.* [Allison2004] also showed that Hick-Hyman's Law predicts accurately the decision time in card sorting tasks.

Landauer and Nachbar [Landauer85] observed that expert performance in hierarchical full-screen menu selections is well described by Hick-Hyman and Fitts components applied in series.

Sears and Shneiderman [Sears94] observed that selection times decay logarithmically with menu length for frequently selected items, but linearly with infrequent ones. It seems that participants move from linear visual search with unfamiliar items to Hick-Hyman decision times as the locations are learnt.

4 Fitts' Law – Law of Pointing

4.1 Formulation

4.1.1 Initial research

Published in 1954, Fitts' Law states a linear relationship between task difficulty and movement time (MT). His formulation is also based on Information Theory. In this case, the human motor system is the communication *channel*, the amplitude of movement is the *signal*, and the target width is the *noise*. The task difficulty, in the first writings by Fitts ([Fitts54, Fitts54b, Fitts54c]), is expressed as:

$$ID = \log_2 \left(\frac{2A}{W} \right)$$

where *ID* is the *Index of Difficulty*, *A* is the amplitude of the movement, and *W* is the width of the target. Movement Time is then defined as a function of the Index of Difficulty:

$$MT = a + b ID$$

where *a* approximates the start/stop time in seconds for a given device, and *b* measures the inherent speed of the device. Both must be empirically determined for each device. Note the similarity of this formulation with the Hick-Hyman's Law. Posterior experiments noted that there is a lower adjustment of the regression curve when testing with low IDs. Actually, the relationship originally established by Fitts was using an approximation of Shannon's theorem that is valid only if the signal-to-noise ratio is large ([Fitts54, McKenzie89]).

In order to arrive to this definition, Fitts made a series of four experiments: Two reciprocal or serial tapping tasks (with a 1-oz stylus and 1-lb stylus), a disc transfer task, and a pin transfer task. For the tapping condition, a participant moved a stylus back and forth between two plates as quickly as possible, and tapped the plates at their centers, as shown in Figure 3. Four different target amplitudes (*A*) were tested, and four target widths (*W*).

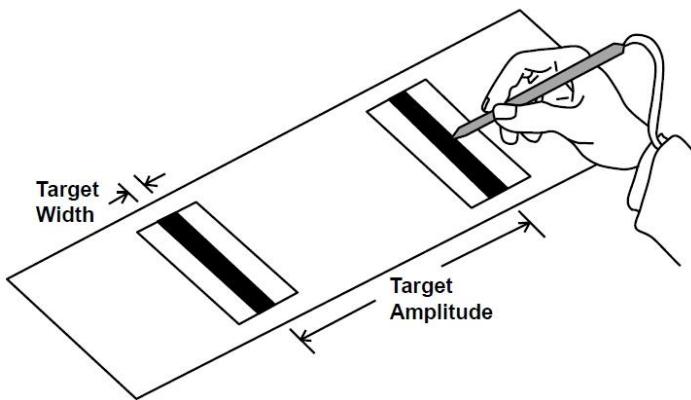


Figure 3: Reciprocal tapping.

As the data obtained by the experiment was published, one can re-examine his results. We can see a plot of the results in Figure 4.

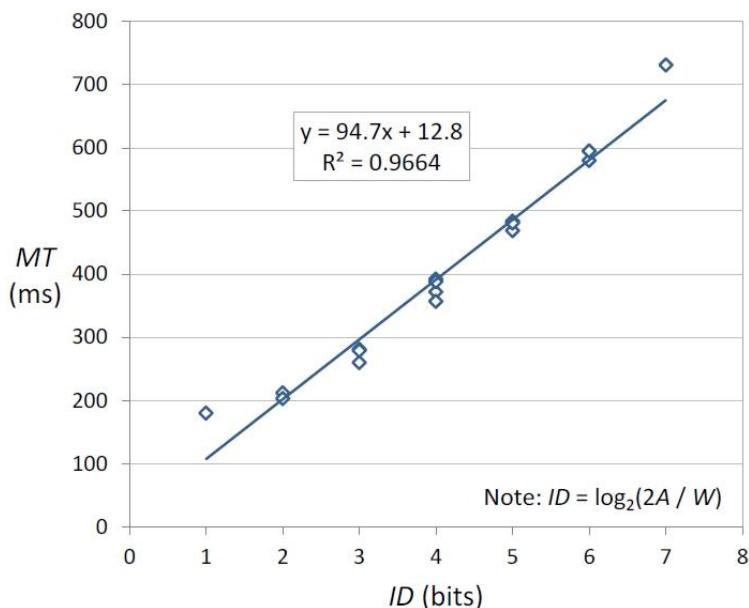


Figure 4: Scatter plot and regression analysis for the data of Fitts.

4.1.2 Refinements to Fitts' Law

After the first publication in 1954, many authors have developed changes or refinements to Fitts' law. The most important rationale behind those is the need for precise mathematical formulations in human computer interaction and other related fields for the purpose of measurements.

An early refinement is due to Welford [Welford68]. He noticed that the MT-ID data points curved away from the regression point, with the most deviate point at ID=1, as can be seen in Figure 4. In order to improve the data-to-model fit, he introduced the following formulation:

$$MT = a + b \log_2 \left(\frac{D + 0.5W}{W} \right)$$

Note that we substituted here the amplitude of movement (A) by the Distance (D). This version has been used frequently over the years, and for instance was the one used in the comparative evaluation of the computer mouse by Card ([Card78]).

Later, it was shown that the relationship deduced by Fitts was based on the approximation of Shannon's theorem. But this only applies if the signal-to-noise ratio is large ([MacKenzie89]). As a result, MacKenzie proposed another refinement to the Fitts' law ([MacKenzie92]):

$$MT = a + b \log_2 \left(\frac{D}{W} + 1 \right)$$

Several versions of Fitts' Law are used, and you can find them consistently in literature, but this formula has been demonstrated to provide good predictions in a wide range of situations.

If we compare the plots of the different models, we can see that they are pretty similar, as shown in Figure 5 ([MacKenzie18]).

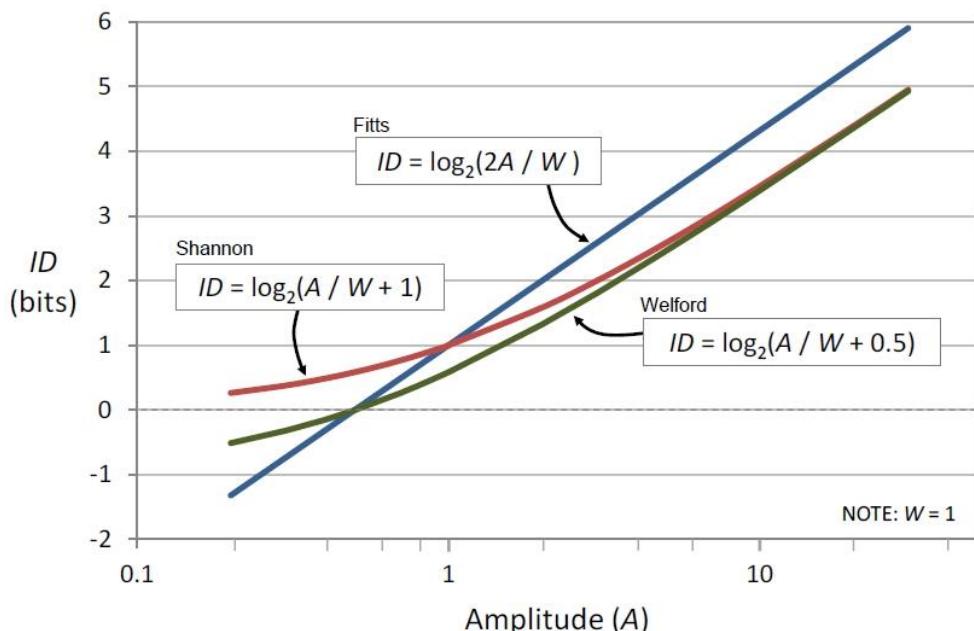


Figure 5: Plots of the main formulations of Fitts' law.

4.1.3 Implications of Fitts' law in UI design

Commonly, the Fitts' Law is applied to the task of pointing or clicking a button. In this case, the parameters involved are the distance D the pointer (e. g. mouse) has to cover, and the width W of the button in the direction of the movement. In Figure 6 all these elements are represented.

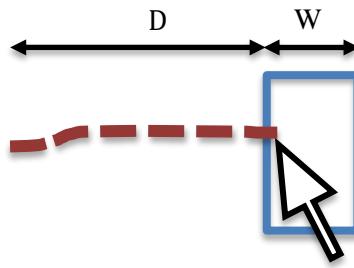


Figure 6: The pointing task according to Fitts' Law.

Several of the other formulae may be better indicated to address some concrete tasks. Note that non trivial variations may be due to differences such as changes in the direction of motion (vertically vs horizontally), device weight (heavier devices are harder to move), shape or size of targets (the original formulation, for instance, does not hold for low ID values), or arm position. So when experimenting with such a priori simple tasks, we must have all the variables under control, because trivially extending the Fitts' law may be invalid. We will see some extensions later.

Fitts' Law describes the Movement Time in terms of one dimensional displacement, as we have already seen. This is valid for either purely horizontal or purely vertical movements. The only thing that has to be taken into account is that we need to use the dimension of the pointing target in question that goes in the direction of the movement. Apart from that, the formula can be applied indistinctively (see Figure 7).

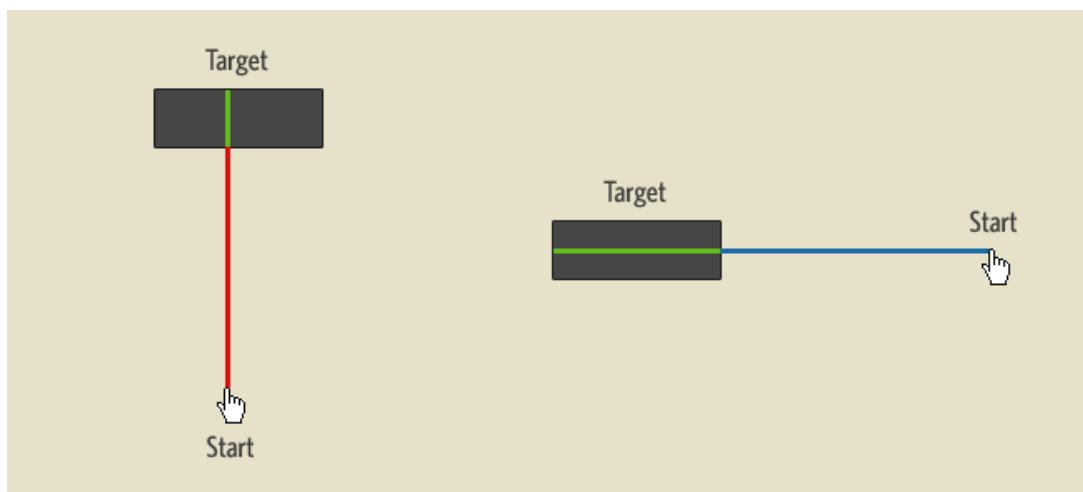


Figure 7: Fitts' Law can be applied to one dimensional movements, either in horizontal or vertical.

Another related measure is the Index of Performance, which is analogous to the Capacity in Shannon's theorem. Some authors call it throughput (TP). IP is measured as bits per unit, and calculated as:

$$TP = ID/MT$$

However, the use of TP , commonly applied to measure performance characteristics of input devices, is quite problematic. The main reason is that TP is only a constant (hence, something that can be representative of the device) when a (also called the intercept) is zero. If a was zero, then the throughput would be $TP = 1/b$ (b is the slope).

4.2 Example

We can show an example here. Let's imagine that we have empirically determined the constants a and b for a certain device. For instance, let $a=300\text{ ms}$ and $b=200\text{ msec/bit}$. If we want to reach a target of 2 cm ($W=2$) at a distance of 14 cm ($D=14$), then, according to Fitts' law, the Movement Time would be

$$MT = 300 + 200 \log_2 \left(\frac{14}{2} + 1 \right)$$

that is, $MT = 900\text{ ms}$.

4.3 Fitts' Law extensions

4.3.1 Fitts law for 2D

The original formulation of Fitts' Law is in one dimension. In all the very initial experiments, the movements demanded to the users were in one direction. With the advent of graphical user interfaces, the navigation region increases, and therefore it is interesting to evaluate the difficulty to reach objectives that require two-dimensional displacements.

Note that, when having non one-dimensional moves, we may point to a target that has different dimensions in height and width (see Figure 8). As a result, we do not have a proper dimension of the target, and some modifications must be applied, since the original formula cannot be used directly.

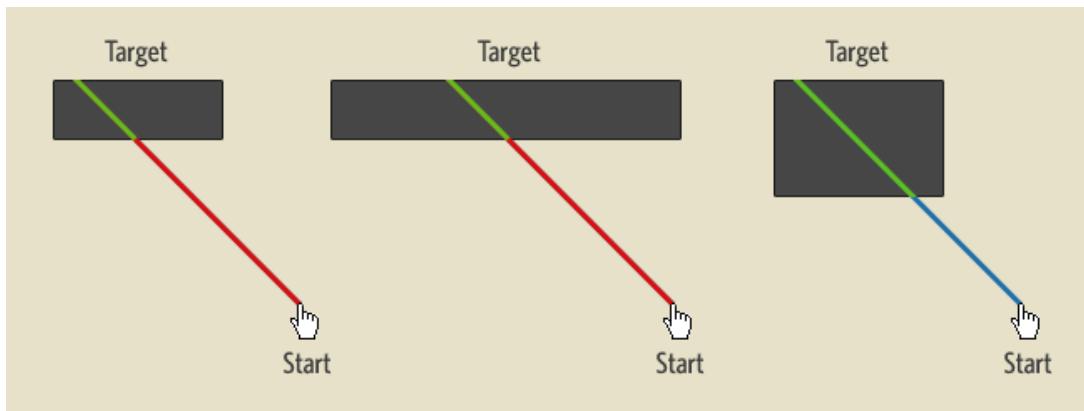


Figure 8: Pointing to a target with a 2D movement.

There have been several developments with this goal in mind. For instance, Crossman and Goodeve [Crossman93] suggested that a second term has to be added to take into account for the height of the target, developing the following formulation:

$$MT = a + b \log_2 \left(\frac{2A}{W} \right) + c \log_2 \left(\frac{2A}{H} \right)$$

Like in the previous cases, the constant c must be determined empirically. W is the amplitude constraint size and H is the height. Other researchers later refined this formulation. A more recent approach is due to Accot and Zhai [Accot97]. They suggest that a better formulation is this:

$$MT = a + b \log_2 \left(\sqrt{\left(\frac{D}{W} \right)^2 + \eta \left(\frac{D}{H} \right)^2} + 1 \right)$$

Where η is a constant commonly in the range 1/3 to 1/7. This indicates somewhat that the directional constraint is less difficult to handle than the amplitude constraint of the same magnitude. Like in the previous case, W is the width and H the height constraints of the target. More formulations have been proposed, but the details are beyond the scope of this course.

4.3.2 Fitts for finger touch

This predictive model is a great help to decide location and size of buttons and other elements in the user interface.

For the case of very small targets, we cannot expect the user to be as efficient as with regularly sized elements. Some studies have found that, for very small elements, there is an extra time devoted to fine adjustment. Moreover, the use of touchscreens also modifies the timing we require to point targets. Sears and Shneiderman ([Sears91]) derived an extension of the Fitts' Law by analyzing the behavior both in tactile screens and small targets. This led to what some authors call the Precision Pointing Movement Time (the original authors call it modified version of Fitt's Law for touchscreens, or FFits):

$$FFits = a + b \log \left(\frac{cD}{W} \right) + d \left(\frac{e}{W} \right)$$

where the first logarithmic factor measures the time to place the finger on the screen initially, while the second factor measures the time to position the cursor once the finger has been placed on the screen. D is the distance, measured in three dimensions, from the original hand location to the location of first contact. W is some measurement of target size. The rest of the constants, a , b , c , d , and e must be determined for each specific case.

If a succession of rapid tasks are performed (e. g. clicking one button and then clicking another), then D will be the distance between both buttons. Since in this formulation we have more than the original two freedom degrees, it might turn that it is easier to fit in a regression curve.

As noted previously, the validity of the formulation is constrained to the datasets on which it was proven. This means, that we cannot simply extrapolate. This is especially important since the experiments in this case were performed in quite early hardware, with much lower precision than current capacitive screens commonly available for most hand held devices. In the case of Sears and Shneiderman, the authors had to implement a stabilization

software that was able to reliably yield a single touch position (though this did not mean that the position resolution detection was even closer to the most accurate mouse interaction, in part due to the resolution of the screen, and in part to the recognition software).

4.4 Fitts' Law in keyboards

The primary method for textual data entry is the keyboard. Keyboards have large history and have even been criticized devices. The average ratio of an office worker is roughly 50 words per minute, although higher rates (up to 150 words per minute) can be achieved. Although regular computer keyboards only allow a key press at a time, some specialized devices, such as the chord keyboards, allow multiple key presses at a time. These may achieve 300 words per minute, but require lengthy training to retain the complex pattern of chord presses.

4.4.1 Keyboard layouts

The typewriter was built in different ways by the middle of the nineteenth century. Several ways to put the paper and several designs for the keys failed. The initial successful proposal was due to Christopher Latham Shole. His design was successful in part because the intelligent placement of the keys facilitated the reduction in key jams. The original layout was called QWERTY due to the keys arrangement in the top left part of the keyboard. This has been the dominant keyboard layout since then. In Europe, some countries use small variations over these layouts, such as the AZERTY layout used in French keyboards or the QWERTZ versions used in Germany (see Figure 9).

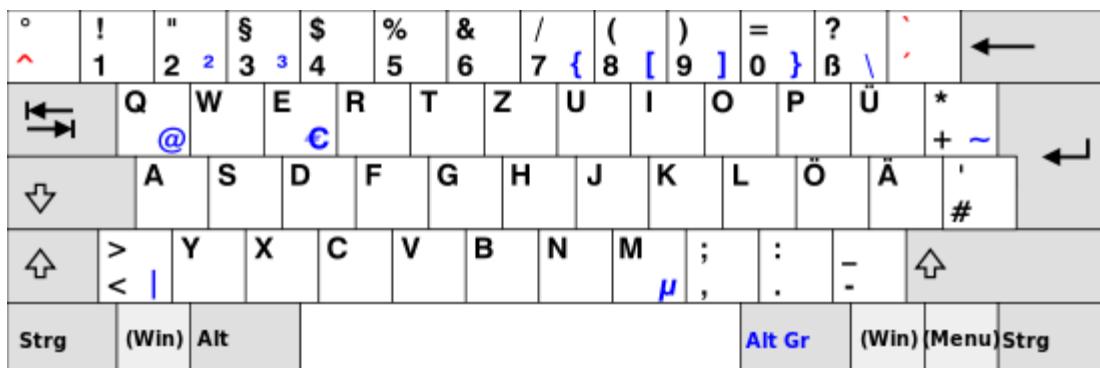


Figure 9: QWERTZ layout used in German keyboards.

By the 20s the Dvorak layout was developed. It was a new arrangement that supposedly reduces finger travel distances by at least one order of magnitude (shown in Figure 10), thereby increasing the number of words per minute approximately a 30% (some other researchers talk of 5-10% of improvement [Norman82]), and reducing errors. Although with a number of devotees, the acceptance of this layout has been low.

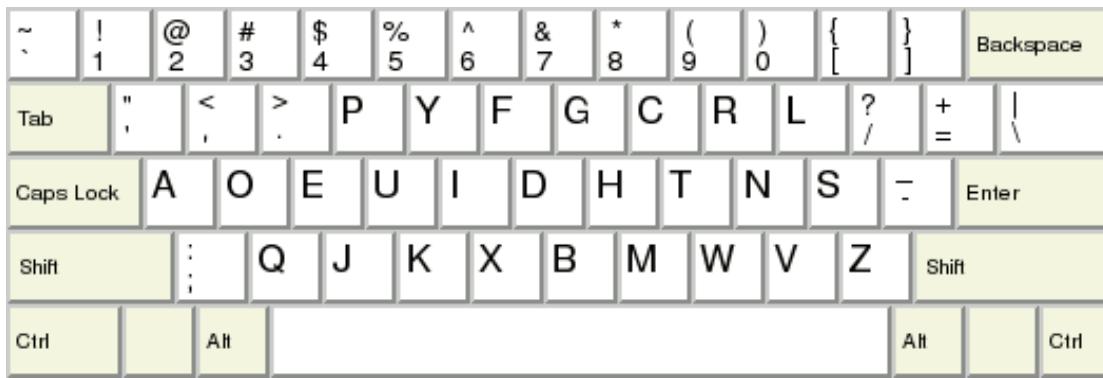


Figure 10: Dvorak layout.

Ideally, as shown by different experiments, for optimum typing speed, keyboard arrangements should be designed so that:

- The loads on the right and left hands should be equalized: both hands do the same amount of work.
- The load on the home (middle) row should be maximized: thus reducing the displacements.
- The frequency of alternating hand sequences should be maximized: this way one may maximize the frequency of typing because the fingers alternate and therefore one does not need to wait for the end of the movement of the first finger before starting the second movement.
- The frequency of same finger typing should be minimized: this may avoid fatigue.

If we analyze the Dvorak layout under these variables, it does a good job, especially on the first two elements: 67% of the typing is done on the home row, and the left-right hand balance is 47-53%. Although the QWERTY keyboard fails at these two conditions, (most of the typing is done on the top row and the balance between the two hands is 57-43%), the policy to put successively typed keys as far apart as possible favors the third condition, thus leading to relatively rapid typing. As a consequence QWERTY layout is superior to other arrangements such as the alphabetical ones. And it is roughly only 5 to 10% (though some reports raise this up to 30%) slower than DVORAK layout.

Note that this is only valid for 10-finger typing. Using this technique, the fingers are placed at fixed positions on the keys (A, O, E, U – for left hand, and H, T, N, S – for the right one), and these makes that very common letters such as vowels, require less effort than other, more uncommon letters. However, different layouts must be evaluated differently if we assume one or two finger typing. As a consequence, the comparison is not quite simple, and the distances the finger must cover may increase in the DVORAK layout in the case of one-finger typing.

4.4.2 Why experimenting with keyboard layouts is difficult

As we have commented many times, the experimentation is a difficult task. You can only assert conclusions on the exact parameters of the experiment. Testing with keyboard layouts adds a second difficulty here: users get their proficiency from practice. Therefore, in order to be able to perform a proper testing, users would require months of training in

any layout we can imagine is the best for typing before one can conduct the user study [Norman82]. Moreover, since the layouts are artificial in some sense, the same people would require to be trained back to the original (e. g. QWERTY) arrangement they use such as to be able to use normal computers.

As a consequence, it is commonly accepted that formal results are yield by using a predictive human performance model rather than user testing for evaluation [Lewis99].

4.4.3 Touch-Screen Keyboards

Modern multi-touch-screens enable text entry that is adequate for mobile interaction, thus bringing touch-screen keyboards to the mainstream. There are many problems when using touchscreens as input method, and several decisions must be taken, such as the keys sizes.

In practice, key size is usually determined by the screen size and its orientation, so the designers try to take advantage of all the possible space, while keeping some room for the display of the entered text. This has a major impact, since the available room is usually not enough: An iPhone 4 character in portrait mode measures 4×5.9 mm, while the Apple Guidelines for user interface design recommend that the minimum clickable size to be 6.85×6.85 mm.

Virtual keyboards also require significant visual attention because the user must look at the screen to press the correct key. Since there is no rest position such as in regular keyboards (which also increases the effort and results in a more fatiguing experience), the user does not know where his or her hands are with respect to the keys. Moreover, there is no physical feedback on the user's actions. Therefore, we can only ensure the key typing by looking at the keyboard and/or the rest of the screen, where the letter is inserted. For larger form-factors, this is especially problematic, due to the relatively large distance from the insertion point to the place where the virtual keys appear.

A second problem is the insertion point: since the screen is touchable, if the user accidentally taps on some part of the screen, the cursor may be changed and the following text ends up being inserted mistakenly at an unintended location.

Some attempts to improve the feedback go from audible “clicks”, to a tactile screen that actually requires pressing on it to effectively enter a key. Its benefits are not clear. For example, the pressing screen, present in mobile devices such as the BlackBerry Storm requires relatively large amount of pressure, and its original layout also lost some precision on the corners (which was improved on the Storm 2 version).

Another big disadvantage of virtual keyboards is that they occlude an important portion of the screen, resulting in less space for the document we are editing or the form we are filling. This is a clear problem when talking of smartphones, and only partially alleviated in tablets.

Some more innovative designs combine the power of touch and stylus-based typing with stroke gestures. They have been shown to produce high rates of text entry, once the user masters them, although it remains unclear if such approaches will achieve widespread adoption.

Finger touch in tactile screens also follows Fitts' Law. Once the users have reached expertise, the time to move the tapping device with a single finger from one key (i) to another (j) depends basically on the distance and key width of the keys, following this equation [Bi2013]:

$$MT_{ij} = a + b \log\left(\frac{D_{ij}}{W_{ij}} + 1\right)$$

Where D_{ij} is the distance between keys i and j , and W_{ij} is the width of each key. They further refine using the effective width of the key, instead of the nominal value, but a deeper analysis goes beyond the needs for our course.

4.4.4 Digram-based keyboards for single-finger typing

As already noted before, the design of an efficient typing keyboard for single-finger typing should minimize the overall displacement of the finger for the most common words. This is what DVORAK actually does for 10-finger typing, but this is not a current typing method in smartphones or tablets. Commonly, one hand is devoted to hold the device and the user types with the other.

Several attempts have shown their proficiency, but usually as a matter of good prediction + excellent error correction together with some tricks to accelerate input (see for example Research In Motion's new keyboard for the Blackberry 10 mobile OS [RIM2012]). However, most of the keyboards still rely on the QWERTY layout, with a notable exception that has attracted a lot of attention: the Minuum keyboard by Whirlscape ([Whirlscape2014]), which uses the QWERTY layout but in a single row (plus a strong word prediction and correction).

For hand-held tablet and portable computers (including pen-based systems), it is important to evaluate the best arrangement of keys for typing layouts when users type with one finger or a stylus. A nonstandard typing-key layout (in a roughly 5 x 5 key matrix) based on digram Predictive human performance models for 10-finger keyboards use the frequency matrix of English-language digrams (pairs of letters that commonly appear together) and a matrix of empirically derived interkey typing times [Lewis99].

To improve the layout for single-fingered typing, it is reasonable to analyze the language digrams to determine the relative associative strengths among the letters so the layout can minimize the distance between strongly associated letters.

This may lead to an arrangement such as the one in Figure 11, where those distances are minimized and thus the predicted improvement raises to approximately 25 words per minute (over the typical 20 words per minute on a complete QWERTY layout and one single finger typing). The authors claim that, after the corresponding training, such a layout should be about 27% better than the QWERTY layout. And even, an alphabetic typing-key arrangement, again in a roughly 5 x 5 key matrix, should be about 13% better than the QWERTY layout for single-finger entry.

Q	R	W	X	Y	
L	U	A	O	F	
Z	T	H	E	N	G
V	D	I	S	P	
B	C	M	J	K	

Figure 11: Diagram-based layout for single-finger typing.

4.4.5 Swipe-based keyboards

Entering text has always been painful in the mobile space. There are many issues that affect the text entry, especially on mobile. To name a few:

- Size of the keyboards: As already noted, the keys' size is smaller than most UX guidelines recommend, especially in smartphones. As a consequence, the size of the keyboard is prone to errors in typing.
- One hand vs two hand entry: When having both thumbs free, we may probably write better, but it is quite common to have to type with one single finger, which, again, is more prone to errors because the hand location is more variable (than two fingers with the rest of the hand in a fixed position). But even in those cases, the number of errors is quite larger than with a regular keyboard.

There is a tendency in mobile space to provide different keyboard interaction techniques for accelerating text entry. Sometimes this acceleration is real, while sometimes it is only felt by the users [Nguyen2012].

Gesture typing basically consists on using a single trace that goes from the first letter of the word until the last one without lifting off the finger, and by moving it on the screen over all the letters of the word. This is illustrated in Figure 12.



Figure 12: Illustration of the gesture keyboarding for entering the word “quick” in an application for gesture-based typing in mobile devices.

One of such tendencies is the use of gestures (swipes) to write a whole word [Kristensson2012, Zhai2012]. Nguyen *et al.* [Nguyen2012] have studied the effect in tablets, where they found that regular typing and gesture typing perform at similar speeds, however, there is a higher user satisfaction ratio when using one of the concrete applications that provide gesture typing (Swype) than regular keying.

Note that all those approaches are commonly executed single handed and in a QWERTY keyboard, which, as we saw before, do not especially reduce the distances the finger has to travel to edit words.

Among the other different improvements, a notable one is a two finger gesture keyboard [Bi2012] which showed that the finger travelling was shortened about the 50% but the speed does not increase over single finger entry, most probably due to the high demand in attention to coordinate both hands. In Figure 13 we can see two proposals for writing “interaction” in such keyboard: one using continuous tracing (right), and the other by interrupting the trace between hands’ change (left).

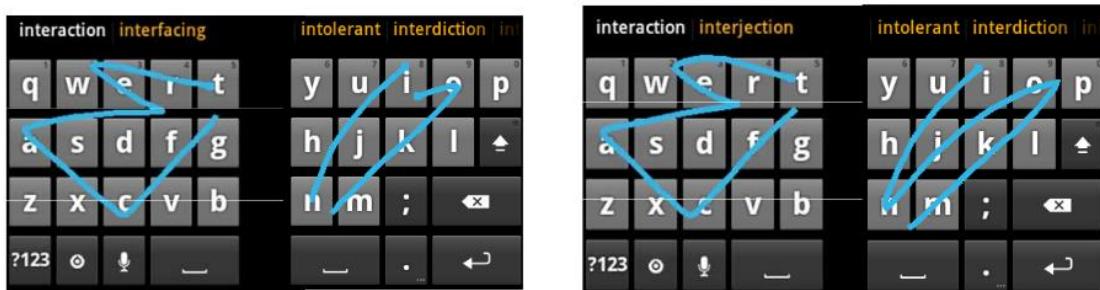


Figure 13: Two different approaches for a bimanual gesture-based input of the word “interaction”.

4.5. Evidences of Fitts law

The Fitts law has been used for successfully modeling many UI access tasks. However, when assessing the validity of such law, we must be very careful on the analysis of the experiment. Experiments cannot cover all the variety of users and tasks, and therefore are constrained to a set of configurations that may be not very large (e. g. 4 different target sizes and 4 different distances). As a consequence, although Fitts’ law has proven its validity in many experiments, we may not freely extrapolate (e. g. assuming it works for elder people or children...) for all sizes and distances. More concretely, as we have already seen, there are limits of the validity of the Fitts’ Law: for very small targets, for instance, the MT function regression curve starts to be less aligned with the resulting values.

Apart from the concrete limitations given by the experimental setup, Fitts’ Law has shown its validity in multiple setups and devices, which go from mouse, joystick, finger, stylus... and different screen types of varying sizes ([Chapuis2011]).

An interesting observation that has been analyzed in some experiments is the previous knowledge by the users of the targets’ sizes and positions. It is an important issue, because users who select objects in point-and-click interfaces quite often know these features of the targets. Studies show that for precued targets the preparations lead to more efficient and precise pointing movements than for non-precued targets [Hertzum2013]. Target precuing also interacts with pointing device, distance to target, and target size, but not with user age. In particular, the benefit of precuing is larger for the mouse than the touchpad, suggesting that the movement preparations users are able to make on the basis of precues depend on how demanding the pointing device is to use.

It is curious to note that, as Balakrishnan and MacKenzie found [Balakrishnan97], the index finger alone does not perform as well as the wrist or forearm in pointing tasks, but the thumb and index fingers in coordination outperform all above cases.

4.6 Fitts variants and limitations

Note that Fitts' law models properly adult users, but children behave differently. Some results show that for point and click tasks, Fitts' law adequately predicts the first time the children enter a target, but not the time of the final selection.

Some researchers suggest measuring the ID using the *effective target width* (or W_e), instead of the *nominal width*. This arises from the observation that users do not adjust their performance as much as might be expected when target width is changed. More concretely, when changing target width, the changes in endpoint variability are typically much smaller would be expected ([Fitts64, Sourkoreff2004]).

Therefore, this formula, commonly called ID_e is written as:

$$ID_e = \log_2 \left(\frac{W}{W_e} + 1 \right)$$

is the one that is suggested also in the ISO 9241-9 ([ISO2000]). However, there is quite a large controversy on its use. There are two main reasons for that:

1. The data does not always fit better than when using ID .
2. The effective width is the result of the actions of the users. As a result, it cannot be measured previously, and therefore is not useful for design without experimentation.

The first problem is still in debate. Wright and Lee [Wright2013] performed a set of studies and found that "it has been shown to compensate for the differences in effective target width that are present across conditions and participants". However, they do not recommend its use without much care since the obtained gain may not compensate for the extra effort to generate such data. Moreover, they believe that when those values are not available, it is safe enough to proceed using ID .

Zhai *et al.* [Zhai2004] believe that both methods (regressions from uncorrected widths and from effective widths) are valid, but the appropriate method depends on the goal [Chapuis2011]:

- Using uncorrected widths seems useful to reliably assess actual user pointing times in user interfaces, accounting for natural biases in target utilization, and to assess errors separately.
- Using effective widths instead usually deteriorates the fit but is strongly recommended when one needs to interpret the constants a and b , for example, in order to compare the performance of different input devices.

5 Applications of Fitts' Law in interface design

Fitts' Law describes the Movement Time in terms of one dimensional displacement, as we have already seen.

5.1 Use of screen edges

There are several interfaces of Operative Systems that put some of the graphical elements of the UI next to the edges of the screen. This is true, for instance in the Mac OS system (see Figure 14), where the top menu is attached to the top edge of the screen. Since there is no distance from the top edge to the menu, the mouse movement required to select any of the options of the menu does not require precision in the Y direction (while it requires in the X axis). As a consequence, the selection process may benefit because the movement can be done fast (using mouse acceleration techniques that will be visited later).



Figure 14: The Mac OS X operative system has a menu bar on top of the screen.

From the point of view of Fitts' Law, this kind of behavior can be interpreted as having targets with virtual infinite length in one dimension. This, at its time, reduces the ID of the target acquisition and therefore intuitively makes the movement simpler/faster to achieve (see Figure 15).

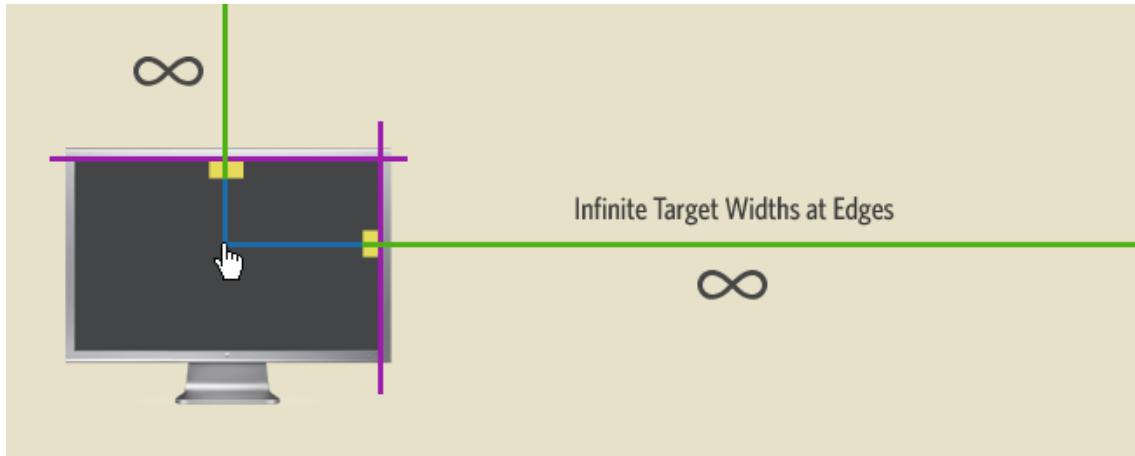


Figure 15: Interpretation of edge elements according to Fitts' Law: They can be considered as having infinite width.

With this in mind, it turns out that corners are the most accessible places in such an UI (see Figure 16), because they have virtual infinite dimensions.

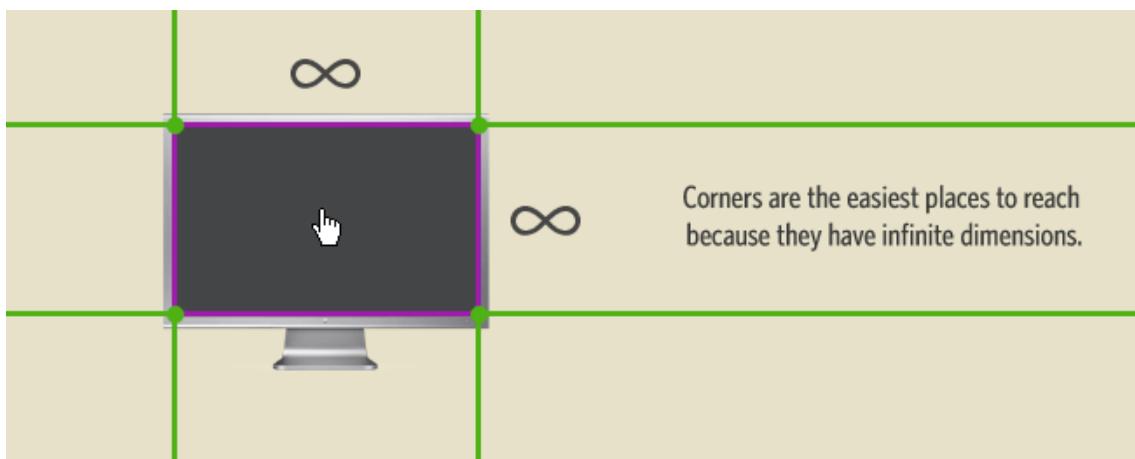


Figure 16 Elements at the corners are the most accessible because they do not require neither horizontal nor vertical precision in one direction.

We may compare this to the typical interface that some programs have presented, such as most Windows applications (in Figure 17 we can see a snapshot of the old Microsoft Word 2003), where the top items were slightly displaced from the top of the screen. We can intuitively see that an extra effort will be necessary for properly placing the pointer in any of the menu titles.

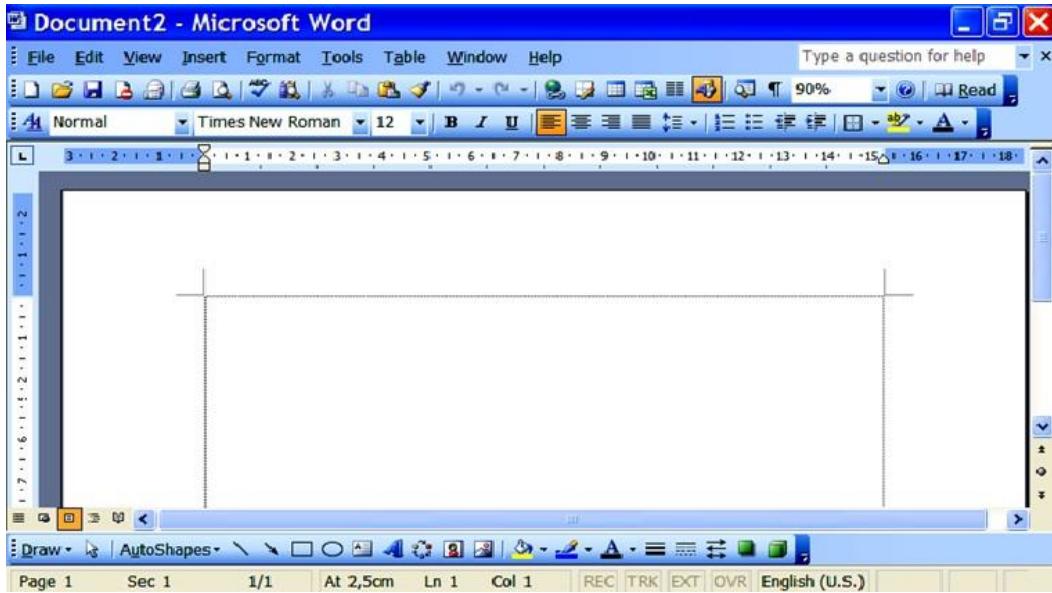


Figure 17: Elements of the menu in Microsoft Word 2003 are not placed at the edge, and therefore require more effort (since their ID is larger) than if they were placed right next to the edge.

5.2 Placing related things close

The use of Fitts' Law can go even further than that. By placing elements that are commonly used together, we are going to facilitate the work of the user. We can see an example in Figure 18, where we compare the Mac OSX scrolling bar with that of Windows.

By analyzing ID we may deduce that the Mac OSX scrolling bar may be faster to use than the Windows one. There are two reasons for this:

- The slider is larger and thus easier to reach.
- The direction buttons are together, so if we need to change the direction, we will need to perform a smaller movement of the mouse.

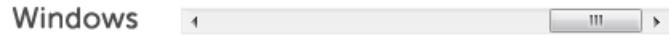


Figure 18: Mac OSX vs Windows scroll bars. The Mac OSX is theoretically faster because it is easier to reach (its slider is larger) and changing the scrolling direction only requires to move the mouse slightly. In the case of the Windows scroll bar, if we want to change the scrolling direction, we need to traverse the whole bar.

5.3 Complicate the access to elements

In the case of elements that are not supposed to be clicked together, or the elements that may be dangerous to confound, it may be useful to do the contrary. That is, separating those elements may avoid mistakes.

Figure 19 shows an image where two buttons that perform very different tasks are placed next to each other, and therefore the user might accidentally chose the wrong one.

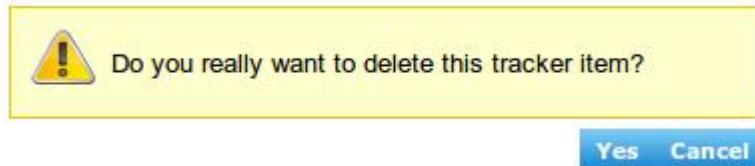


Figure 19: Buttons doing very different tasks and that are placed together may induce errors. In this case, buttons *Yes* and *Cancel* have opposite meanings. Moreover, there is not a clear separation on both, which is a second mistake. They should be placed at separate positions.

5.4 Pie menus

Another not so popular types of menus are circular and semi-circular (or pie) menus (see Figure 20). These menus appear around the contact point of the finger or the clicking point of the mouse. The distance from each menu item to the cursor is constant and very small.



Figure 20: Pie menus rely on the fact that the items can be placed at the same distance from the pointer. When the number of items is high, several layers of menus are required.

The idea is quite interesting. However it requires some elements to make it usable:

- First, the menu has to be created on demand, on a certain position, that leaves room for the whole menu to appear. This turns this kind of menu into a contextual one, with its advantages (it may vary its contents in function of the position) and shortcomings (it will be used only by expert users).
- Second, for the menu to be practical, it may have no occlusions (the original pie menus are circular, which is something that will necessarily be partially occluded in a tactile device), and the elements must be sorted in such a way that the access is simple.

The original first pie menus were circular (see Figure 21) since they were developed for desktop systems. In such systems, the pointer (mouse) does not occlude the menu when it appears (since the menu is all around the pointer representation).



Figure 21: Circular menu with all the elements at the same distance to the pointer.

6 Law of crossing

6.1 Introduction

Apart from simply pointing or clicking, the use of stylus or analogous to stylus devices (e.g. finger) in tactile surfaces naturally leads to other quite intuitive tasks such as crossing elements (see Figure 22). Accot and Zhai ([Accot97, Accot99]) and Zhai *et al.* [Zhai2002] demonstrated that there exists the so-called *Law of crossing*. It models the time to move a cursor or a stylus across two goals. Moreover, the Law of crossing follows the same characterization than the Fitts' Law:

$$T = a + b \log_2 \left(\frac{D}{W} + 1 \right)$$

Where T is the average moving time between passing the two goals. D is the distance between the two goals and W is the width of each goal. Like in the previous case, a and b are constants to be determined.

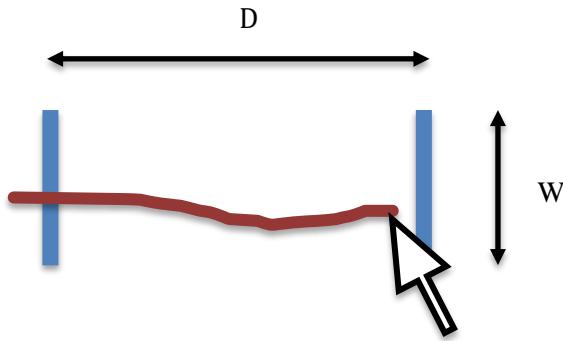


Figure 22: Crossing movement with the pointer.

The crossing action is well related to the pointing action described by Fitts' Law as we commonly interpret it in a graphical user interface with buttons. The different elements are compared in Figure 23.

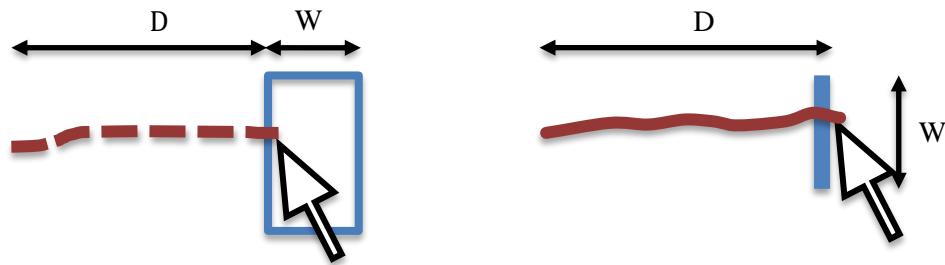


Figure 23: Pointing vs crossing: the interaction techniques differ in where the precision is required: in the direction of the movement (such as in the crossing interface – right – where the width determines how far we may move from the initial direction) or in the termination point (when pointing – left – the limit is bound by the width of the target).

The way to analyze this kind of interaction is not simple, since it can be performed in very different ways. Hence, many variables come here into play:

- **Discreteness vs continuity of the movement:** In some cases it is necessary to land the stylus before crossing the first target and lift off after crossing it, and the same for the second objective, while the interaction can also be done without lifting off the stylus (see Figure 24).
- **Direction of the targets vs direction of the movement:** Targets may be aligned in the same direction of the movement or orthogonal to it. In the first case, a line may cross many targets while in the second case, a different trace will be necessary (shown in Figure 25).

Note that, since the so-called *Law of Crossing* takes the same form as the Fitts' Law, it is important to try to isolate such constants in order to be able to determine whether crossing is better than pointing.

6.2 Crossing configurations

In contrast to pointing, the crossing technique and targets can be configured in different ways (e. g. is the target to cross orthogonal to the direction of movement?). Thus, for the sake of completeness, the performance of crossing must be evaluated in each of the configurations. The different varying variables include the direction of the movement vs the orientation of the target, and whether it is necessary or not to lift the pointer before crossing the target. We start discussing this last element.

6.2.1 Continuous vs Discrete

The implementation of the technique may be so that the pointer (stylus, finger...) can move in contact to the screen or it may be required to lift it up before crossing. These two configurations are called continuous and discrete, respectively, and are depicted in Figure 21.

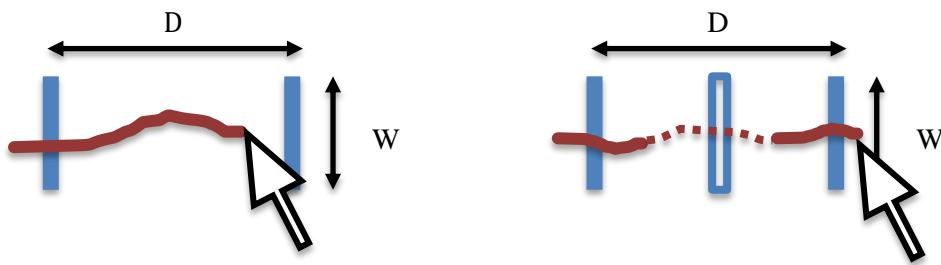


Figure 24: Continuous crossing (left) vs Discrete Crossing (right) interaction.

Intuitively, discrete is more costly than continuous, but the studies show that this is only more time consuming when the distance between the targets (and thus the obstacle) is small.

6.2.2 Collinear vs Orthogonal

The targets, both in pointing and in crossing, can have their width aligned or not with the direction of the movement. When the width W is orthogonal to the direction of movement, we have the configuration on the left in Figure 22, where the movement seems more simple and continuous, while when the objectives are not aligned with the direction of the movement, the pointer must trace a slightly more complicated path. This is also applicable to crossing targets with obstacles (discrete crossing).

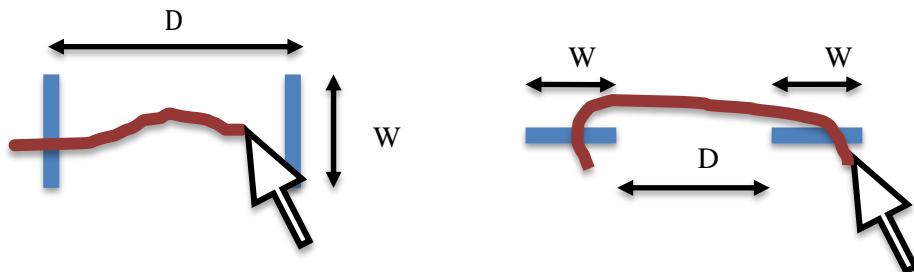


Figure 25: Collinear crossing (left) vs Orthogonal Crossing (right) interaction.

In order to analyze the performance of crossing, it is necessary to perform a factorial combination of the different configurations, that is, one should test continuous traces with orthogonal crossing, continuous tracing with targets in the same direction of the movement plus discrete tracing with these two configurations of the targets.

The studies by Apitz *et al.* [Apitz2010] compare the different configurations of the more classical pointing technique: collinear and orthogonal, with all the different configurations of the crossing method: the collinear discrete, collinear continuous, orthogonal discrete and orthogonal continuous. And the experiments are carried out for several different target sizes and distances. The different configuration scenarios are analyzed in relation to *ID* and error rate. The conclusions the researchers reach are, very briefly:

- Crossing-based interfaces achieve similar (or faster) times than pointing.
- The error rate in crossing is smaller than in pointing.
- Discrete crossing becomes more difficult if the distance between the targets is small.
- Crossing (especially continuous) seems superior than pointing for *ID* values > 5.

7 Law of Steering

7.1 Introduction

Another commonly necessary task for the interaction with modern user interfaces is the creation of trajectories. These tasks are useful for navigating through nested menus, 3D navigation, dragging elements, and other drawing tasks.

Very often, such tasks are directional movements with a lateral constraint (see Figure 26).

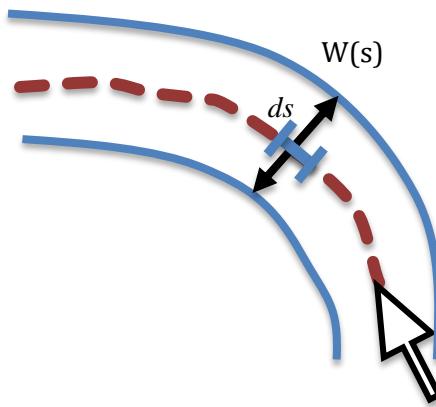


Figure 26: Generalized path steering with amplitude constraint.

Accot and Zhai found that the performance of manual steering tasks (T_s) for a generalized path C (see Figure 26) can be expressed in the following integral form:

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

Where T is the time to successfully steer through the path C , $W(s)$ is the path width at s . With this formulation, the Index of Difficulty of the steering task is

$$ID_s = \int_C \frac{ds}{W(s)}$$

Accot and Zhai arrived to this formulation by transforming the steering task into an accumulation of an infinite number of goal-crossing tasks ([Accot97]). The time to cross a goal of width W at distance D follows the Fitts' Law equations. This way, by transforming the goal crossing task into a succession of goal crossings along a trajectory, they can calculate the summed up index of difficulty of N consecutive goals as:

$$ID_N = N \log_2 \left(\frac{D}{NW} + 1 \right)$$

Taking N to infinity yields the equation of ID_s .

7.2 Steering through straight paths

Using these results, we can calculate the time to steer through a straight path. The straight path can be determined using the D and W variables for the distance to trace and the width of the path, respectively (see Figure 27).

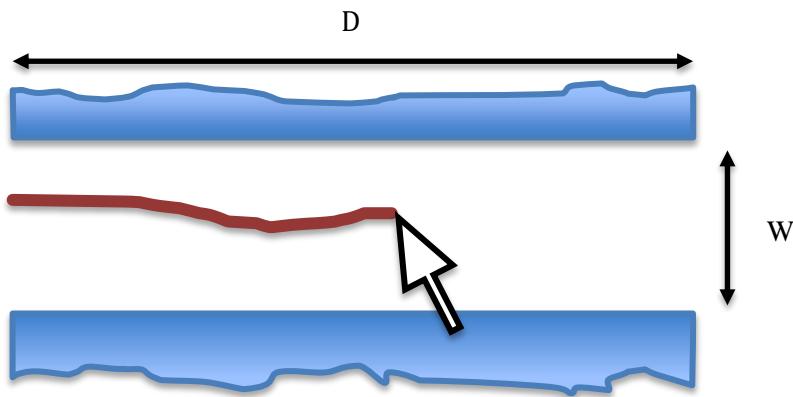


Figure 27: Path movement in a straight path with amplitude constraint.

For a straight path, the time required to successfully steer through it would be:

$$T_p = a + b \frac{D}{W}$$

Where W is the path width, and D the path length. The index of difficulty of the task will be then:

$$ID_p = \frac{D}{W}$$

The previous equation also applies to circular paths with constant width.

7.3 Evaluations of Steering Law

Accot and Zhai [Accot97] showed steering law works for paths of different shapes: a cone, a spiral shape, or a circle, and that it also works for different input control devices. The upper limit on the application of this law is the human body limitations: This means that we can change configuration parameters so that the theoretical speed can be increased, but if this exceeds human body capacities, the results saturate.

One of the direct applications of steering law is the design of nested menus. Since accessing an item of a submenu is addressed by performing a path that follows an ‘L’ shape, this path can be subdivided in two different steering paths, and therefore the access time can be calculated by directly applying the steering law. As a result, different menu configurations (e. g. classical nested menu vs pie or hierarchical pie menu) can be evaluated using this law.

A more complex interaction experiment was also developed by Zhai *et al.* [Zhai2004] to evaluate locomotion in a VR setup of 160 degrees of field of view. The users had to navigate through two types of path, one straight and the other circular. Graphics were rendered in real time in response to users’ actions. The path was 1885 meters long. The users were instructed to drive as quickly as possible without going out of the path boundary.

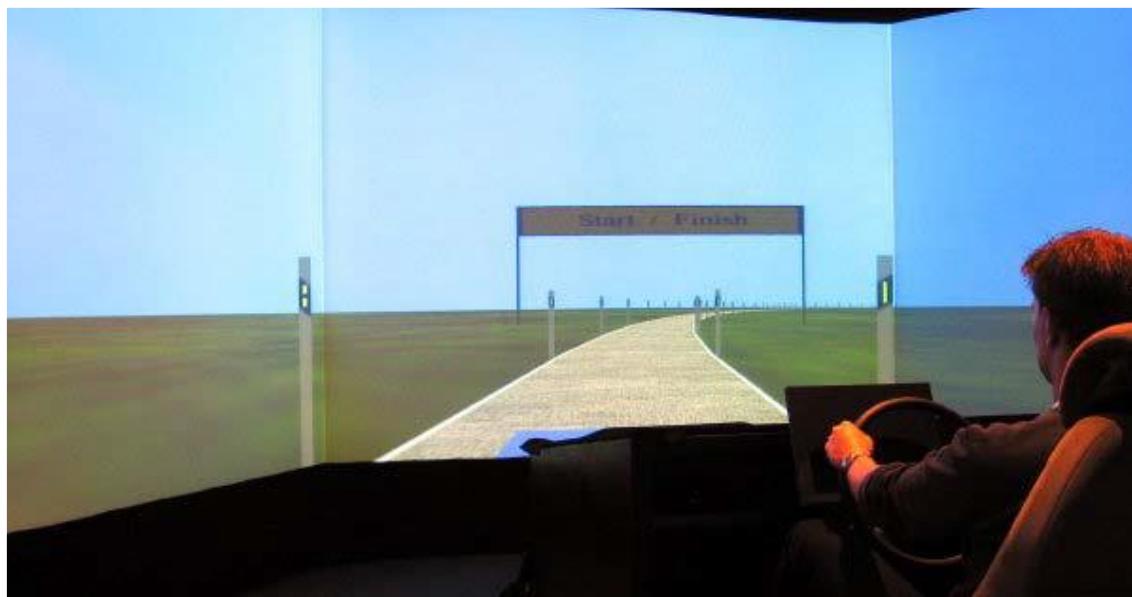


Figure 28: The Virtual Reality simulator setup used in the steering experiment.

The user’s behavior was measured every tenth of a second throughout the experiment. The recorded variables were: vehicle’s X and Y coordinates, heading, gas pedal amplitude, brake pedal amplitude, steering wheel amplitude, speed, deviation from middle line, and collisions.

The users had a representation of the car also projected on the screen, to help them have references of their relative position with respect to the road. Some examples of the images seen by the users throughout the driving experiment are shown in Figures 28 and 29.

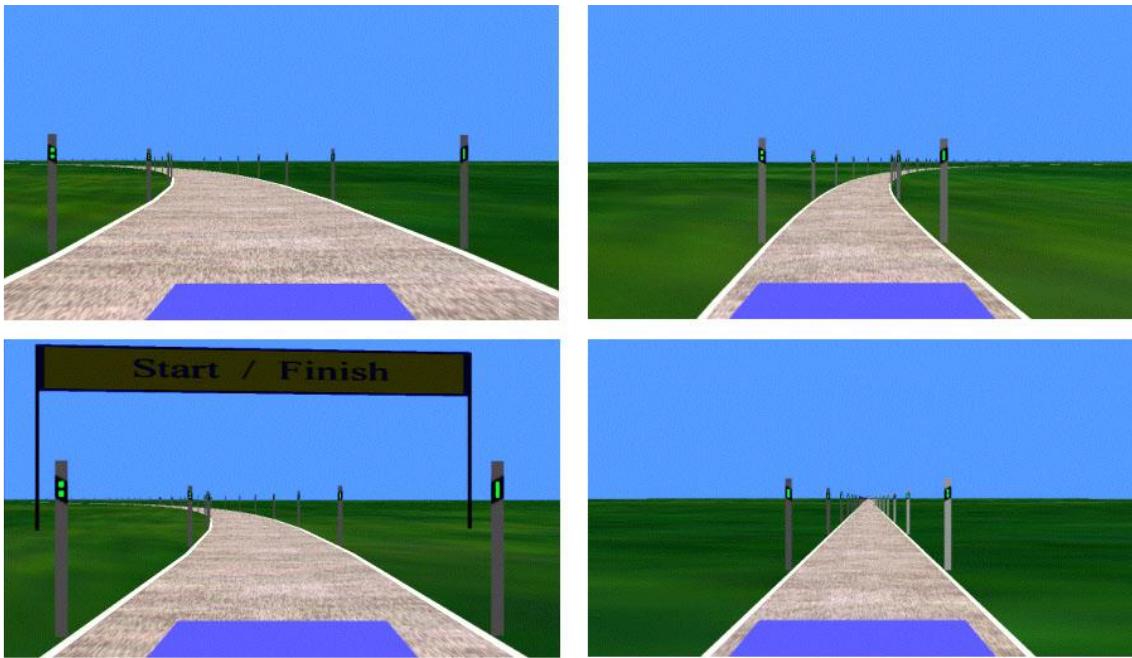


Figure 29: The images projected in the simulator, as seen by the users.

The results show that there is higher variance between users than in other experiments where no so complex tasks are performed. For example, some users used the total power of the gas pedal, unlike in the pilot experiment. The results of those users were eliminated from the analysis. Otherwise the results would not be comparable.

Results show that the steering law behaves well for locomotion, although with a lower dependency on the path width, since driving involves stronger dynamics than simple gestures such as hand drawing. In any case, the authors found that there is a linear relationship between the trial completion times versus the index of difficulty of the path, with a high fitness value.

8 Accelerating target acquisition

8.1 Introduction

Besides the uses of Fitts' Law in the design of User Interfaces, commonly the screen space is restricted enough to avoid decisions that imply the use of larger targets or cluttered screens to accelerate target acquisition. This, together with an improvement in the computation power, has led to the use of techniques that modify such parameters dynamically.

Commonly, the modifications that are performed are the same than we have seen for static UIs: amplitude reduction and increase of target size. However, we may change these parameters a non-constant value. Moreover, we may add to this the modification of the speed of the pointer, the so-called dynamic control-display ratio change.

8.2 Modification of Target Width

In order to reach and manipulate virtual objects, applications provide a pointer together with the virtual representations of the elements to manage.

8.2.1 Target and screen size

As stated by the Fitts' Law, the size of the target we are trying to reach has a great influence on the time required to select it. Actually, the cost or effort to reach an object depends on the amplitude of the movement and the size of the target. The larger the target, the easier it is to select, but the longer the distance to cover, the more difficult it is to reach.

There is a tight relation between screen size and target size, the larger the screen, the larger the targets can be. However, since the amount of information we want to include in our windows is also relatively large, we have to balance between information and controls.

Although nowadays screen resolutions often leave enough room for the design of large target icons, the fact that screens are large also make that the distance the user has to cover can also be large. Therefore, according to Fitts' Law, the cost of reaching the element also increases with the distance.

On the other hand, the resolutions of portable devices, although being rather high (such as 900x600 pixels), the size is small (around 4" on a modern smartphone), and therefore, the number of pixels that has to be devoted to a single icon is relatively high, thus making the screen *effectively smaller*. There is a second issue: The fact that the pointer in most modern screens is a finger. Hence, its size relative to the size of the screen makes the display also *effectively smaller*.

With respect to Fitts' law, there are two simple ways to reduce the difficulty of a pointing task: enlarging the target or moving it closer to the cursor. Both have been explored in several ways. These are also coupled with the velocity of the cursor. We will deal with this issue, named Control-Display ratio later. Now, we introduce some techniques that try to reduce interaction times by manipulating the target sizes dynamically.

8.2.2 Expanding targets

One approach to easing acquisition of small targets consists in enlarging the target size when the cursor approaches the element to be selected. This technique is called expanding targets [McGuffin2002] and has been implemented in different ways. Mac OSX operative system uses this technique on the icon panel that serves as program launcher, also known as Dock, shown in Figure 30.

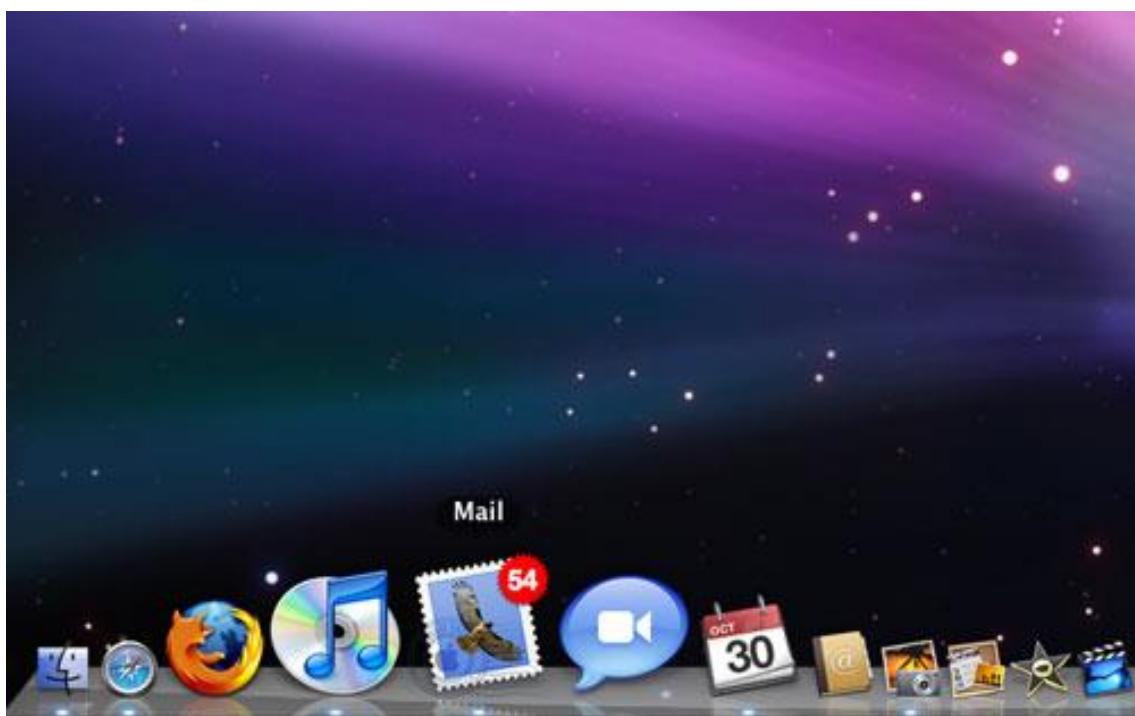


Figure 30: Mac OSX Dock with its size-enlargement and position-changing icons.

Note that this is a mix on two techniques: target size increase and moving target. The system changes the size and moves the position of the elements in the Dock in relation to the position of the cursor. Therefore, the *selectable* area is smaller than it seems to be, because when approaching the center of the icon (in a horizontal move), the icon itself moves towards the opposite direction of the cursor movement. This has been reported to cause frustration on the users because the expansion induces these movements that may be not totally predictable if the cursor approaches the Dock in a perpendicular direction. This can be alleviated if the system allows the enlarged icons to overlap over the neighbors. A second method that reduces frustration is expanding the icons only when the cursor velocity decreases on final target approach.

8.2.3 Bubble targets

Other not so successful methods for target expansion have been tested, such as the *bubble targets* [Cockburn2003]. These add a selection region around the target with a bubble shape (Figure 31).

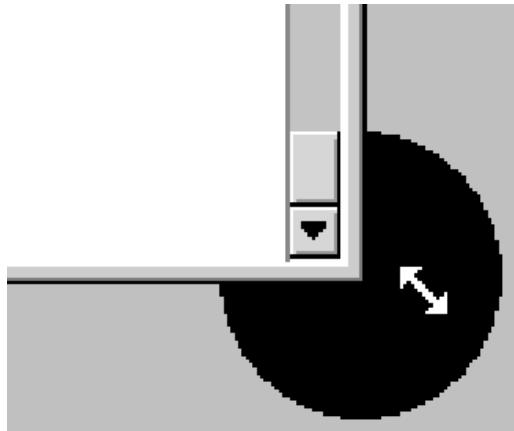


Figure 31: Bubble-based targets for easier selection.

The bubbles only appear when the mouse is pretty close to the selectable point. This technique improves the speed of selection of small targets, but some users have reported distraction due to the appearing of the bubbles.

8.3. Increasing cursor size

Some authors address the same problem by changing the area that is affected by the cursor, instead of modifying the target size. This leads to an effective reduction of amplitude movement. But it can also be seen as a magnification of the target size.

Grossman and Balakrishnan [Grossman2005] increase the size of the cursor in a circular region that embraces the closer target for a more comfortable and easy selection. They call this technique the *Bubble Cursor*. The cursor's size is dynamically adjusted so that only a single target is enclosed by it. Moreover, its shape also is modified to envelop the closer target when it is not completely inside the main cursor bubble. In Figure 32 we can see a sketch of such technique.

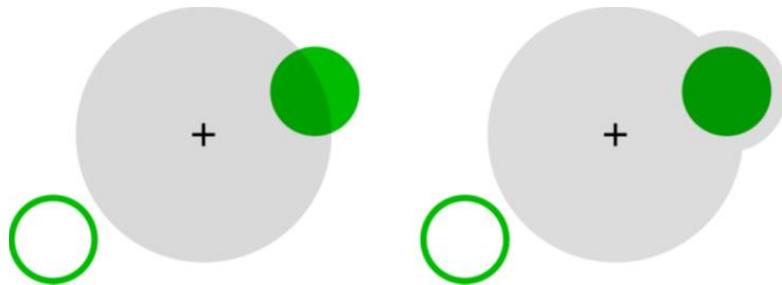


Figure 32: Bubble cursor adapting its size to cover the closer target.

Their implementation is quite straightforward, since they divide the space of targets using a Voronoi diagram where each region contains a single target and the effective width of the cursor is modified to envelope the target contained in that region (see Figure 33).

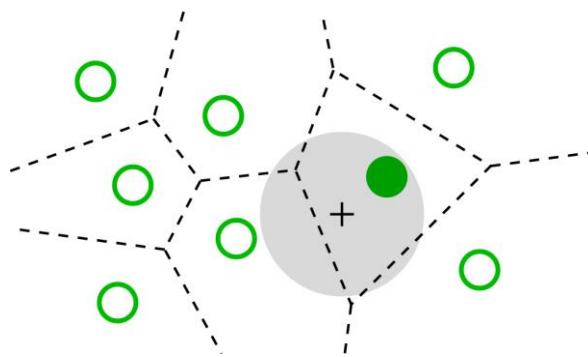


Figure 33: Voronoi subdivision of the target space. When the cursor enters in one of the regions of the target Voronoi map, it may adapt its size to envelop only the target of interest.

This technique has been proven very useful and fast. However, the experiments were performed using a zero mouse acceleration value. Chapuis *et al.* [Chapuis2009] improve on this technique by taking into account also the speed of the mouse. The technique they develop is called *DynaSpot*. In this case, since the area of the cursor grows with the speed, it is likely that the cursor area will overlap several targets at the same time. However, only the closer to the cursor center is the one candidate to selection. We can see in Figure 34 how the area is changed according to the speed of the cursor.

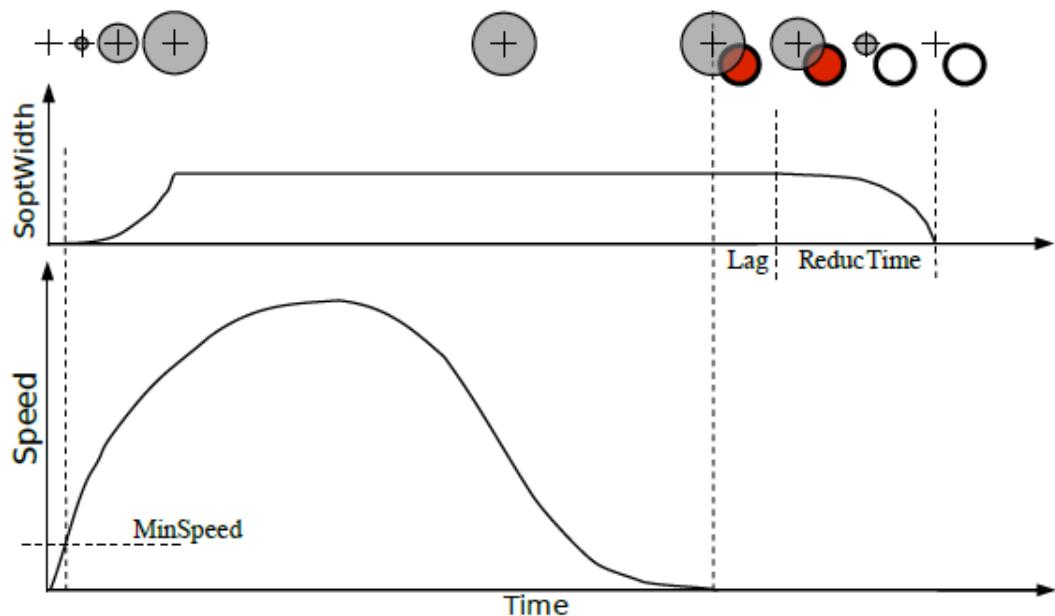


Figure 34: Area change of the cursor as a function of the mouse speed.

Like in the previous case, visual cues are provided to inform the user which are the targets that are going to be selected as well as the current size of the area cursor. This is shown in Figure 35.

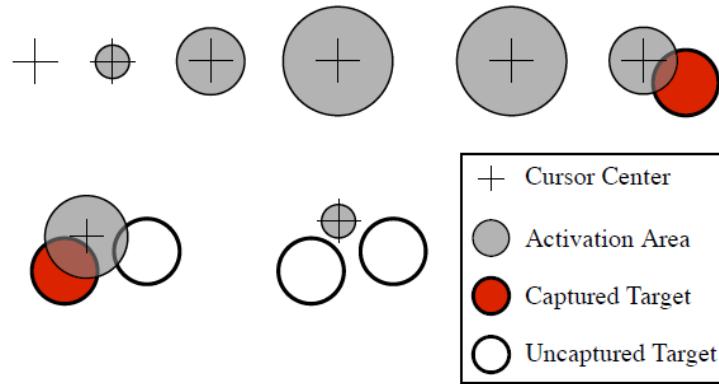


Figure 35: Indication of the target to be selected (the closer to the center of the cursor).

Chapuis *et al.* [Chapuis2009] also compare this method with the Bubble Cursor and find that this new method obtains better results both in acquisition time and reduction of number of errors.

8.4 Target moving and other techniques

8.4.1 Target moving

Besides enlarging targets, there is another popular possibility that enhances selection and consists in moving the targets to the cursor.

The method used by the Mac OSX Dock combines target enlargement and target movement, however the movement is relatively small compared to other techniques.

Probably the most widely used direct application of target movement is contextual pop-up menus. Contextual menus are the ones that appear at the cursor location after some user action (such as selection and/or right click). This makes the distances to the items to be minimal, as compared to visiting the classical top menus. Moreover, as compared to the classical pop-down menus, the contextual menus can adapt their contents to the previous action: For instance, if the user selected a word, several options related to word formatting can appear, which may differ on the possible actions if no region or word is selected.

Another not so popular types of menus are circular and semi-circular (or pie) menus (see Figure 36). These menus appear around the contact point of the finger or the clicking point of the mouse. The distance from each menu item to the cursor is constant and very small.



Figure 36: Semi-circular menu on Android's Honeycomb version of the browser.

8.4.2 Other techniques

Other experiments have shown that the selection can also be helped by the *sticky targets* technique. The main idea is to convert the selectable points as *attractors* of the cursor. Therefore, when the cursor is close to the selectable point, it moves to it. This method improves efficiency and has been implemented both in 2D and 3D user interfaces. Experiments have shown that users adapted easily to this technique and were preferred to other methods such as the bubble targets.

This technique has also been implemented in 3D virtual environments by Andújar and Argelaguet [Andujar2007], by modifying the selection ray direction to point to the target it is getting close (see Figure 37).

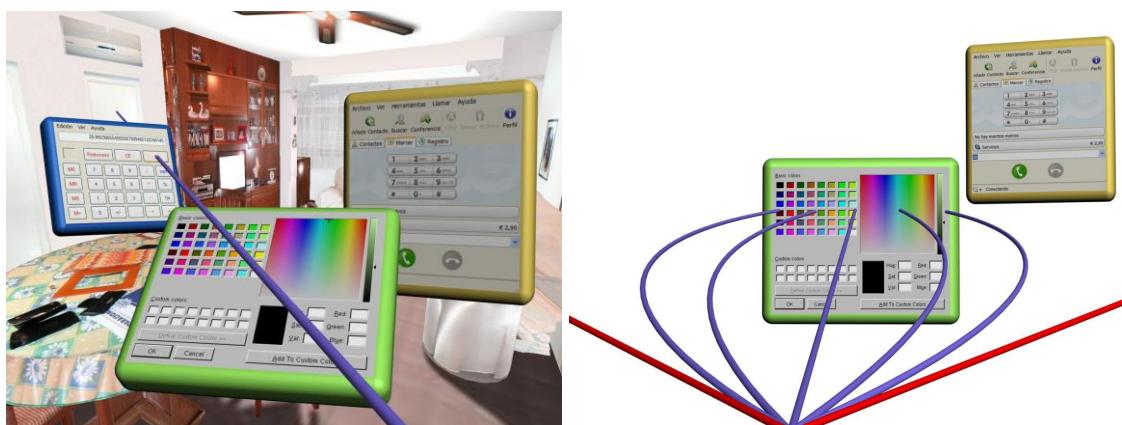


Figure 37: Ray direction modification in a 3D Virtual Reality environment to point to the closer target.

8.5 Discussion

Comparisons with other techniques show that target resizing facilitates pointing even if the expansion is late and unpredictable. The main problem of such techniques is that in order to expand a target if other targets surround it, those must be shrunk. As a consequence, some experiments showed that no performance improvement is obtained for systems like the Mac OSX Dock. More generally, techniques that dynamically change the screen layout cause a spatial disorganization that may limit their expected benefits.

In other environments, where the scene may be cluttered, and a lot of selectable targets may be placed in a small region of space, such as in many 3D virtual reality scenarios, a combination of techniques may be useful. Argelaguet and Andújar [Argelaguet2008] combine expanding targets with clipping away other closer candidate targets with a technique they call *forced discocclusion*.

8.6 Control Display Ratio

8.6.1 Concept

Control Display ratio is the relation between the amplitude of movements of the user's real hand and the amplitude of movements of the virtual cursor. More specifically, the Control Display Ratio usually refers to the distance the mouse has to cover in the physical world to move the pointer on the screen by a given distance. These are usually measured in meters (physical move) and pixels (cursor movement).

Obviously, any control system must translate physical displacements of the interaction devices to virtual movements of the cursor. The way this translation is implemented may affect the performance of the user.

8.6.2 Control-Display Ratio adaptation techniques

There is no clear idea on how the definition of C-D ratios affects our perception of the virtual world and therefore improves productivity by reducing selection times. There are three typical C-D ratio configurations:

- Constant
- Dependent on mouse speed
- Dependent on the cursor position

The idea behind the *mouse acceleration* is to provide easier ways to access further targets. The cursor moves by a larger distance when the mouse covers a given amplitude more quickly, capturing an intention: when users move the physical device quickly, they typically wish to go further, so the cursor can be displaced even faster to cover the distance more quickly.

An opposite effect can be *mouse slowing* when the cursor is near a target: covering the same number of pixels requires moving the mouse by a longer displacement. The idea here is to improve pointing precision.

Although some studies conclude that nonlinear mappings are not intuitive enough to provide positive improvements for pointer movements, other studies have shown increase

in performance for 3D navigation, 3D rotations and other pointing problems. Therefore, there are no definitive results.

Compared to the previous approaches that address target magnification, C-D ratio adaptation can also be interpreted as dynamic magnification of the physical motor space where the mouse movements take place. In this sense, these techniques are related to zoomable interfaces that use a local or global magnification of the visual space.

Pointer acceleration is the default behavior on the Microsoft Windows, Linux, and Apple Mac OS X operating systems. The implementation in those systems dynamically changes the C-D ratio the following way: When the speed of the control device is high, CD gain is high (typically well above 1) and when the control device moves slowly, the CD gain is low (in some cases less than 1). In some Operative Systems, this behavior can be configured, as it is also related to other accessibility options (e. g. large fonts and icons for people with vision problems...).

9 Pointing tasks

Complex information displays generate the necessity of pointing and selecting items. It is a fundamental and frequent task in graphical user interfaces, so even a marginal improvement in pointing performance can have a large effect on a user's productivity. This direct-manipulation approach is attractive because it prevents the users from learning commands. It has other advantages such as reducing the chance of typos and keeps the attention of users on the display. As a result, pointing and selecting often results in faster interaction, fewer errors, easier learning, and higher satisfaction.

Pointing, however, does not come without particularities. Depending on the display we want to use, different devices or techniques may be more convenient. One of the first elements we must consider when analyzing the properly pointing technique and device is the goal. We may find six types of interaction tasks in literature:

- **Selection:** Users must choose one or more than one element from a set of items. This technique is used for the simple task of selecting an item in a menu, to more elaborate tasks such as the selection of a part in a CAD design.
- **Position:** Positioning is another fundamental task in sketching, drawing, or CAD/CAM applications. Users must choose a point in one to three dimensions.
- **Orientation:** Like in the previous case, designing software often requires determining orientations for the new elements to be added or to modify certain aspects of the scene.
- **Path creation:** The path must be built from a set of positions and orientations or in a continuous way.
- **Quantify:** The selection of a quantitative value is often implemented as a one-dimension selection task.
- **Text:** Adding, deleting, or modifying text is also a task accessible through selection and pointing processes. Text must be added simply, but other operations such as text font and size, or page layout also fall into this category.

Although many of these tasks can also be implemented with a keyboard by typing numbers or letters to determine position, orientation, and so on, it is often far less efficient than with a direct manipulation tool. Expert users, however, may use a combination of direct pointing and keyboard shortcuts (such as *Ctrl+C*) to further accelerate the processing.

9.1 Pointing devices

There is a great amount of pointing devices and its number has increased constantly for several years. We may classify the pointing devices in two families:

- Direct-control devices
- Indirect-control devices

Direct-control devices are those that work directly on the surface of the display, such as the stylus or our fingers on a capacitive screen.

Indirect-control devices work away from the surface, such as the mouse, joystick, graphics tablet, and so on.

9.1.1 Direct-control devices

Direct-control devices have existed for a long time. One of the earliest devices is the *lightpen*, which enabled users to point to a spot on a screen and press a button to perform a selection operation.

The *lightpen* dates from back the middle seventies (you can see an example in Figure 38), as it was a device that already worked on UNIVAC 1652 in 1976. Compared to today's touch screens, the *lightpen* was heavy and fragile. Its operation caused fatigue and the hands over the screen obscured part of the information.

Although today's designs still produce higher fatigue than the mouse, a proper design, such as with a surface on which to rest the arm, greatly alleviates the problems of the *lightpen* and older screens.

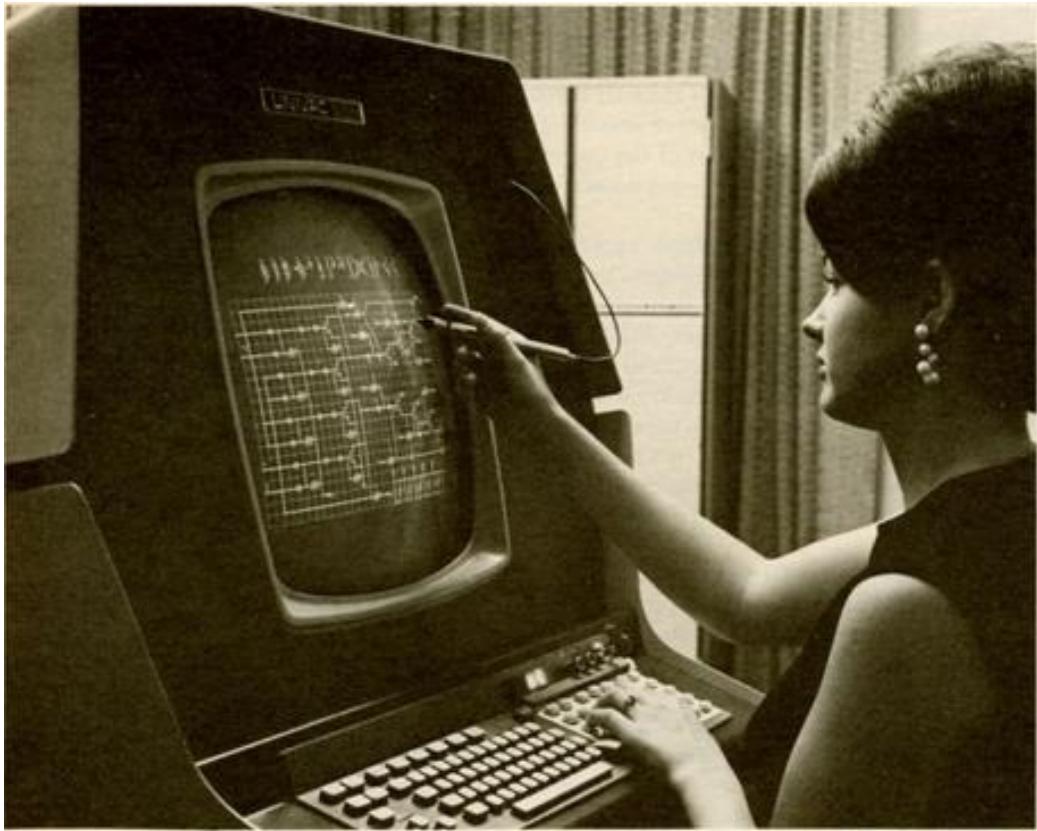


Figure 38: Univac's interaction with a lightpen (around 1976).

Direct-control devices have other problems, still present in mobile phones, such as the imprecise pointer (in this case it is mainly related to the quality of the screen), and the ability of the software to accept the touch immediately (also called *land-on* strategy). This has some advantages, such as the fast response, but also some problems, such as the inability to correct the mistakes from the user. This may be somewhat painful for iPhone users. Since the unlock screen does not require a confirmation of the entry code, as can be seen in Figure 39, the user may be wrong, let's say, when keying the last digit, and then, the iPhone tries to unlock with a wrong code. Since the user may be aware on his or her mistake, it is frustrating not to be able to correct the input, and this may lead to completely lock the device if the same mistake is committed three times.

For some tasks, touch screens use the *lift-off* strategy. It has three steps: When the user taps on the screen, a cursor appears that can be dragged (without removing the finger from the screen) to finely adjust the correct position. When the user is satisfied with the cursor position, they lift the fingers off the display to activate. This is the technique used in the virtual keyboard of iOS system (see Figure 40) for accentuated or other special characters. In this concrete case, the new characters appear right on top of the finger's position, and the user moves horizontally over them to activate the desired character.



Figure 39: Unlock screen of the iOS devices. The lack of confirmation button is error-prone, since the user has no time to correct the input even if he or she detects that the last key was wrongly pressed.



Figure 40: Lift-off strategy for introducing accented letters in the iOS operating system. The user selects the letter, and leaves the finger on, until the submenu appears with the differently accented versions of the selected letter.

Touch screens are often integrated into applications directed at novice users in which the keyboard can be removed completely. They have some advantages in public-access systems. Because they can be implemented with no moving parts, their durability in high-use

environments is good (some people say that the touchscreens are the only input device that has survived at Walt Disney World theme parks).

Multi-touch screens allow a single user to perform multiple finger entry, such as pinch-to-zoom gestures, or multiple users to work on the same application together.

Although some people may prefer finger input on touch screens, some tasks are better addressed with a pen. Pens are familiar and comfortable for users, and facilitate the movement of the cursor while leaving almost the whole context in view. The use of a pencil has also some shortcomings. The most clear is probably the additional work to pick up and put down the stylus.

9.1.2 Indirect-control devices

Indirect pointing devices alleviate hand-fatigue and eliminate the problems of hand obscuring the screen. On the other hand, they require the hand to locate the device and demand more cognitive processing and hand/eye coordination to bring the cursor to the desired target.

One of the most popular devices is the mouse: It is cost-effective, allows for a comfortable position of the hand, its buttons are easy to press, and positioning can be done quickly and with precision. However, for long movements, users must pick up and replace the mouse position.

Another popular device is the trackball, which resembles an upside-down mouse. The ball is used to move the cursor in the screen as it is moved. Joysticks have been long used in games. They are appealing for tracking purposes because they only need relatively small movements to move a cursor.

Touchpads have become very popular lately due to their presence in portable computers (and its decrease in price). They offer the convenience and precision of a tactile screen while keeping the user's hand off the line of sight. Like in the case of touch screens, their lack of moving parts is usually an advantage.

The graphics tablet is a touch-sensitive surface that is separated from the screen. It is often used lying flat on the table and can be operated with a finger, pencil, stylus, and so on, and usually provides input possibilities not only by position, but also by pressure. This last feature, almost unique in touchscreens, is highly appreciated by artists. Like the mouse, when resting in a flat surface, may be used for long periods without producing fatigue.

9.2 Comparison of pointing devices

When analyzing pointing devices, there are two major variables to take into account: speed and accuracy. Some studies have found that direct pointing devices are often faster but more prone to errors than indirect control devices.

For decades, the mouse has shown its superiority to other devices in speed and accuracy. For instance, the mouse is faster than the *trackpoint* due to tremors in finger motion during fine finger movements.

When analyzing two devices we must take into account the task. For scrolling large lists or large documents or webpages, mice equipped with a scroll wheel may be convenient, although some studies showed that they are not superior to regular mice.

The usual belief is that pointing devices are faster than keyboard cursor keys, but depending on the task, these may be superior due to the smaller movement required to use them, and the fact that some shortcuts, if mastered, are very convenient for document edition. Usually, short tasks that mix typing and pointing are faster addressed with cursor keys.

Users with motor disabilities may prefer devices that are fixed, such as joysticks or trackballs.

10 3D Selection

10.1 Introduction

Several designers believe that 3D interfaces can make several tasks easier than classical 2D systems. They believe that this would allow a behavior that is similar to real life, and therefore, tasks can be accomplished more efficiently. In Figure 41 we see an example of 3D environment being used by two people at the same time.



Figure 41: An inspection session in a 3D virtual environment. Both the users can see in 3D but the view is calculated accurately for the user whose googles are tracked (left).

Besides that, some people are also studying whether enhanced interfaces may even be better than reality. Of course, the term *better* may have different meanings, but the idea is that some scenarios can be performed using a virtual reality environment in a more effective way than in the 3D world. There is some controversy, because the tasks a 3D

environment may be more suitable or even superior to *reality* need to be computer tasks, such as computer-assisted design, molecular modelling, and so on. In these tasks, we can provide the user with super-natural powers such as travelling back in time to undo some of the actions.

10.2 Definitions

Over the last decade, the field of 3D user interfaces has grown out of its infancy, forming the basis for many game and industry applications. There are some terms that are important to get familiar with:

- **3D interaction:** An interaction between a person and a computer in which the user's tasks are carried out in a 3D spatial context using directly 3D input devices or 2D input devices with direct mappings to 3D.
- **3D user interface:** A User Interface that involves 3D interaction.
- **3D interaction technique:** The technique designed for solving a certain task. This may involve both the use of hardware (a device) and software (a module that has as input the device signals and that implements the behavior in the virtual space).

Note that none of those definitions implies the use of a 3D environment in the sense of Virtual Reality (i. e. stereo viewing as a minimum). Although many 3D interfaces are commonly linked to 3D environments, we may implement those in a desktop system without stereoscopy.

There are many devices tailored to produce 3D stereo and many devices used to interact with 3D interfaces. We will not visit them, since this is beyond our scope. We will only address one of the problems that arises in 3D environments and whose solution is complex: 3D selection.

10.3 3D selection

We refer to 3D selection as the selection task when it is carried out in a 3D immersive environment. Compared to other selection methods, new problems appear, such as the occlusion of the target.

7.3.1 3D Selection techniques

Commonly, 3D object selection techniques are implemented with a selection tool such as a data glove or a Wanda device. Since we need to select in a 3D environment, we need to define a 3D position, which is often given by a ray, a 3D cursor or a simple 3D shape such as a sphere. The computer must perform a 3D intersection or proximity test with the environment. In contrast to the *selection tool*, the method used for effectively selecting the elements is usually called *selection technique*.

The selection technique defines how the user will control the selection element. In typical VR environments tracking devices will usually perform the monitoring of the users' actions.

Selection in 3D Virtual Reality environments can be accomplished using different techniques. However, the two main paradigms are:

- **Hand extension techniques:** These techniques are also called 3D point cursors, since they represent a 3D point in space as a mapping of the user's hand position (see Figure 42).
- **Ray-based techniques:** Instead of using the hand position, these techniques use the position and some other element to indicate an orientation to generate a ray in space that is used as a pointer. These techniques are also called aperture-based selection techniques or ray cursors. The different configurations can be seen in Figure 43.



Figure 42: The representation of a hand in a VR driving simulator as a mapping of its real position and orientation.

Previous research in VR has demonstrated that ray-based techniques are often superior to hand extension techniques due to the faster selection times.

10.3.2 Ray-based selection and pointing

In order to define a ray, we need an initial position, and an orientation. Usually, the user controls the 3D cursor by moving its dominant hand: the control is performed using the hand's position as the initial point, and the direction of the ray is calculated using the users' wrist orientation. Sometimes, the virtual ray starts from the head of the observer and that goes along the hand position.

For selecting single objects or points in scenes, ray cursor techniques have an inherent problem. Since the ray is a 2D geometric object, the ray often intersects multiple objects, and so the actual target of interest is ambiguous. Moreover, selecting occluded elements, if needed, may become a problem, because the occluding objects must be erased, moved, or visually removed in an easy way. A second possibility is to define a selection point on the pointing ray. Then, some mechanism has to be provided to move the cursor along the ray.

This is also problematic because the hand can move (especially if we also use it to operate a wheel or a joystick that performs the 3D cursor movement on the ray) and the desired direction can change. One possibility is to lock the ray after the direction has been chosen appropriately (a technique is called *lock ray*) and then allow for the movement of the 3D cursor on the ray. The three methods are illustrated in Figure 43.

There is a second problem that arises when using rays defined by the hand as their initial position: the ray direction and the viewing direction is not the same. As a consequence, some object that is visible from the user's viewpoint can be unreachable from the user's hand position. This is not easy to address, since many computations have to be done on real time if we want the application to detect this problem. Some elements that may alleviate this issue consist of using elements such as the sticky targets in 3D [Andujar2007] (Figure 44) and enlarging the pointed elements in a similar way to the bubble controls [Argelaguet2008] (see Figure 45), or to locally flatten the elements to facilitate pointing.

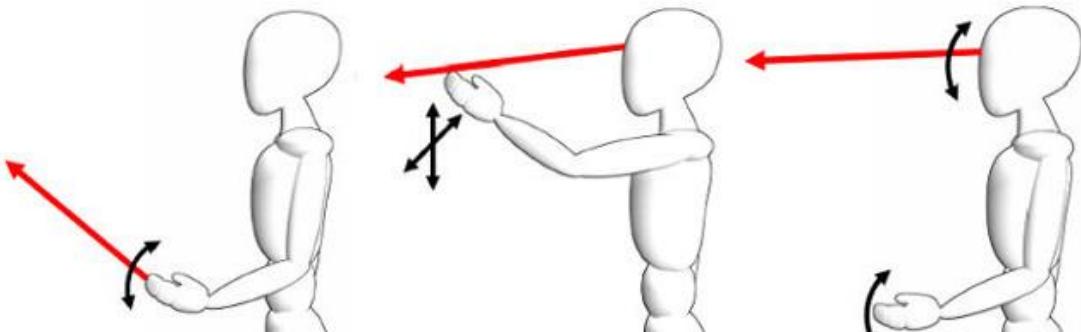


Figure 43: The three typical paradigms of ray pointing in VR: Left shows the interaction being controlled by the hand position and wrist orientation. Center shows the ray being started at the eye, and the intersection with the hand creates the ray direction. Right shows the ray starting in the eye, while the direction is controlled by the wrist orientation.

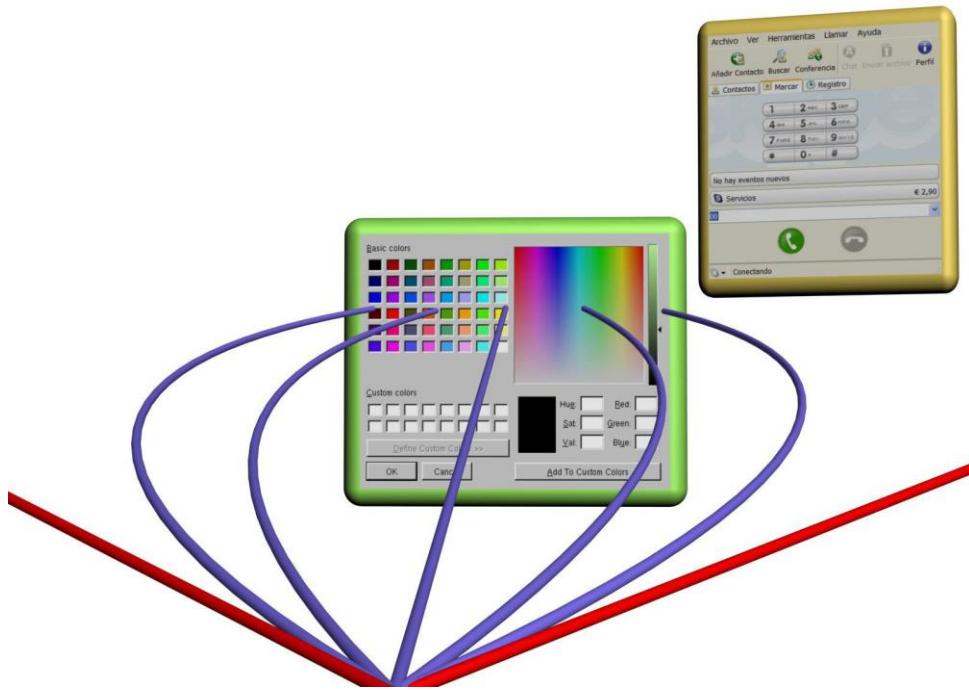


Figure 44: Implementation of sticky targets for ray attraction in a 3D environment for the help in menu selection.

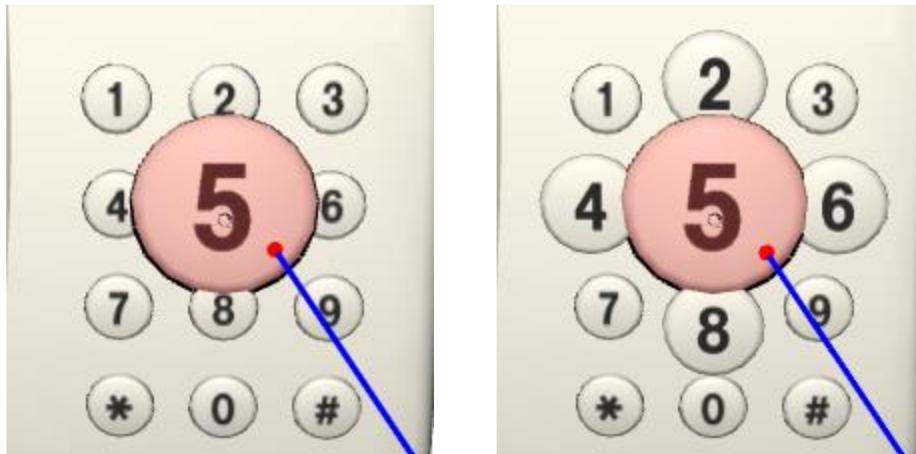


Figure 45: Expanding targets in a 3D environment [Argelaguet2008]. Different configurations are possible, such as only expanding a single target, the one closer to the user (left), or an expansion of all closer selectable targets (right).

11 Bibliography

- [Accot97] Accot, J. & Zhai S. (1997) *Beyond Fitts' law: models for trajectory-based HCI tasks.* In Proceedings of ACM CHI'97 Conference on Human Factors in Computing Systems, pages 295–302.
- [Accot99] Accot, J. & Zhai. (1999), Performance evaluation of input devices in trajectory-based tasks: an application of the steering law. *CHI '99 Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pp. 466-472.
- [Accot2002] Accot, J., & Zhai, S. (2002). *More than dotting the i's - foundations for crossing-based interfaces.* Proc. CHI: ACM Conference on Human Factors in Computing Systems, Minneapolis, Minnesota, 73 - 80.
- [Allison2004] Allison, B., Desai, A., Murphy, and Sarvary, R.M., (2004) Choice reaction time in card sorting tasks: validation of the Hick-Hyman Law, San Jose State University ISE 212, Summer 2004.
- [Argelaguet2008] Argelaguet, F., Andújar, C. (2008). Improving 3D Selection in Immersive Environments through Expanding Targets. In *Proceedings of the 8th International Symposium on Smart Graphics (SG '08)*. pp:45–57.
- [Andujar2007] Andújar, C., Argelaguet, F. (2007). Anisomorphic Ray-Casting Manipulation for Interacting with 2D GUIs. *Computers & Graphics*. 31:15–25.
- [Apitz2010] Apitz, G., Guimbretière, F., Zhai, S. (2010). Foundations for designing and evaluating user interfaces based on the crossing paradigm. *ACM Transactions Computer-Human Interaction*, 17(2).
- [Balakrishnan97] Balakrishnan, R. and MacKenzie, I.S. (1997). Performance differences in the fingers, wrist, and forearm in computer input control. *CHI ' 97: ACM Conference on Human Factors in Computing Systems*. p. 303-310.
- [Bi2013] Bi, X., Li, Y., & Zhai, S. (2013). FFitts law: Modeling finger touch with Fitts' law. In *Proceedings of the 2013 ACM annual conference on Human factors in computing systems*, pp. 1363-1372.
- [Card78] Card, S. K., English, W. K., & Burr, B. J. (1978). Evaluation of mouse, rate-controlled isometric joystick, step keys, and text keys for text selection on a CRT. *Ergonomics*, 21, 601-613.
- [Card83] Card, S.K., Moran, T.P., & Newell, A. (1983). *The Psychology of Human-Computer Interaction*. Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.
- [Chapuis2009] Chapuis, O., Labrune, J.-B., and Pietriga, E. (2009). DynaSpot: Speed-dependent area cursor. *Proc. CHI '09: ACM Conference on Human Factors in Computing Systems*. New York: ACM Press, 1391–1400.

[Chapuis2011] O. Chapuis, and P. Dragicevic, (2011) Effects of motor scale, visual scale, and quantization on small target acquisition difficulty. *ACM Transactions on Computer-Human Interaction (TOCHI)*, Volume 18 Issue 3, July 2011, pp. 13-32, ACM New York.

[Cockburn2003] Cockburn, A. and Firth, A. (2003). Improving the acquisition of small targets. *British Human Computer Interaction Conference*. p. 181-196.

[Cockburn2007] Cockburn, A., Gutwin, C. and Greenberg, S. (2007). A Predictive Model of Menu Performance. Proceedings of CHI'07: ACM Conference on Human Factors in Computing Systems, San Jose, CA, 627-636. ACM Press.

[Cockburn2008] Cockburn, A., Gutwin, C. (2008). A Predictive Model of Human Performance with Scrolling and Hierarchical Lists. In *Human Computer Interaction*, vol. 24 no. 3, 273-314.

[Crossman83] Crossman E., Goodeve P. (1983). Feedback control of hand movement and Fitts' law. *Quarterly Journal Experimental Psychology*, 35A:251-278.

[Donders68] Donders, F. C. (1868). Die Schnelligkeit psychischer Prozesse. (On the speed of mental processes). *Archiv für Anatomie und Physiologie und wissenschaftliche Medizin*, 657-681.

[Fitts54] Fitts, P. M. (1954). The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology*, 47, 381-391.

[Fitts54b] Fitts, P. M., & Deininger, R. L. (1954). S-R compatibility: Correspondence among paired elements within stimulus and response codes. *Journal of Experimental Psychology*, 48, 483-492.

[Fitts54c] Fitts, P. M., & Seeger, C. M. (1954). S-R compatibility: Spatial characteristics of stimulus and response codes. *Journal of Experimental Psychology*, 46, 199-210.

[Fitts64] Fitts, P. M., & Peterson, J. R. (1964). Information capacity of discrete motor responses. *Journal of Experimental Psychology*, 67, 103-113.

[Grossman2005] Grossman, T. and Balakrishnan, R. (2005). The Bubble Cursor: Enhancing target acquisition by dynamic resizing of the cursor's activation area. *Proc. CHI '05: ACM Conference on Human Factors in Computing Systems*. New York: ACM Press, 281-290.

[Hartley28] Hartley, R.V. L. (1928). Transmission of information. *Bell System Technical Journal*, 535, 1928.

[Hertzum2013] Hertzum, M., and Hornbæk, K., (2013). The Effect of Target Precuing on Pointing with Mouse and Touchpad, *International Journal of Human-Computer Interaction*, vol. 29, no. 5 (2013), pp. 338-350.

[Hick51] Hick, W. E. (1951). A simple stimulus generator. *Quarterly Journal of Experimental Psychology*, 3, 94-95.

[Hick52] Hick, W. E. (1952). On the rate of gain of information. *Quarterly Journal of Experimental Psychology*, 4, 11-26.

- [Hick53] Hick, W. E. (1953). Information theory in psychology. *IEEE transactions on Information Theory*, 1, 130–133.
- [Hyman53] Hyman, R. (1953). Stimulus information as a determinant of reaction time. *Journal of Experimental Psychology*, 45, 188–196.
- [ISO2000] ISO. (2000). *ISO9241-9 International standard: Ergonomic requirements for office work with visual display terminals (VDTs)—Part 9: Requirements for non-keyboard input devices*: International Organization for Standardization.
- [Kristensson2014] Kristensson, P.O. (2014). From wax tablets to touchscreens: an introduction to text entry research. *ACM XRDS* 21(1): 28-33. *Invited. Special Issue on Computers and Language.*
- [Landauer85] Landauer, T. K., & Nachbar, D. W. (1985). Selection from alphabetic and numeric menu trees using a touch screen: Breadth, depth, and width. *Proceedings of the CHI'85 Human Factors in Computing Systems*, 73–78. New York: ACM.
- [Lewis99] Lewis, J. R., Kennedy, P. J., & LaLomia, M. J. (1999). *Development of a Diagram-Based Typing Key Layout for Single-Finger/Stylus Input*. Proc. of The Human Factors and Ergonomics Society 43rd Annual Meeting.
- [MacKenzie89] MacKenzie, I. S. (1989) A note on information-theoretic basis for Fitts' law. *Journal of Motor Behavior*, 21, 323-330.
- [MacKenzie92] MacKenzie, I. S. & Buxton, W. (1992). Extending Fitts' law to two-dimensional tasks. *Proceedings of the CHI'92 Conference on Human Factors in Computer Systems*. New York: ACM.
- [MacKenzie18] MacKenzie, I. S. (2018). Fitts' law. In K. L. Norman & J. Kirakowski (Eds.), *Handbook of human-computer interaction*, pp. 349-370. Hoboken, NJ: Wiley. doi:10.1002/9781118976005
- [McGuffin2002] McGuffin, M. and Balakrishnan, R. (2002). Acquisition of expanding targets. *ACM CHI Conference on Human Factors in Computing Systems*. p. 57-64.
- [Merkel85] Merkel, J. (1885). Die zeitlichen Verhältnisse der Willenstätigkeit (The Temporal Relations of the Actions of Will, or The Timing of Voluntary Action). *Philosophische Studien (Philosophical Studies)*, 2, 73–127.
- [Nguyen2012] Nguyen, H., Bartha, M. C. (2012). Shape Writing on Tablets: Better Performance or Better Experience? *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, San Diego, CA, HFES, pp. 1591-1593
- [Norman82] Norman, D. A. and Fisher, D. (1982). *Why alphabetic keyboards are not easy to use: Keyboard layout doesn't much matter*. Human Factors, 24, 509-519.
- [Nyquist24] Nyquist, H. (1924). Certain factors affecting telegraph speed. *Bell System Technical Journal*, 47, p. 617.

[RIM2012] Research in Motion (2012). *The BlackBerry 10 keyboard demoed.* http://www.youtube.com/watch?v=l0wclug_TUU, checked Oct. 2014.

[Schenider2013] Schneider, T. D. (2013). Information Theory Primer, With an Appendix on Logarithms, <http://alum.mit.edu/www/toms/paper/primer/> (checked October 2014).

[Sears91] Sears, A., and Shneiderman, B. (1991). High precision touchscreens: Design strategies and comparison with a mouse. *International Journal of Man-Machine Studies*, 34, 4, pp. 593-613.

[Seow2005] Steven C. Seow (2005). Information Theoretic Models of, HCI: A Comparison of the, Hick-Hyman Law and Fitts' Law, *Human-Computer Interaction*, 2005, Volume 20, pp. 315–352.

[Sears94] Sears, A., and Shneiderman, B. (1994). Split menus: Effectively using selection frequency to organize menus. *ACM Transactions On Computer-Human Interaction* 1, 1, 27 - 51.

[Shannon48] Shannon, C. E. (1948) A mathematical theory of communication. *Bell System Technical Journal*, 27, 379–423, 623–656, 1948.

[Sourkoreff2004] Soukoreff, R.W., & MacKenzie, I.S. (2004). *Towards a standard for pointing device evaluation, perspectives on 27 years of Fitts's law research in HCI*. International Journal of Human- Computer Studies, 61, 751–789.

[Welford68] Welford, A. T. (1968). Fundamentals of skill. London: Methuen.

[Whirlscape2014] Whirlscape (2014). *The Minuum Keyboard*. <http://minuum.com/>

[Wright2013] Wright, C. E., & Lee, F., (2013) Issues Related to HCI Application of Fitts's Law, *Human-Computer Interaction*, Volume 28, Issue 6, pp. 548-578, 2013.

[Zhai2002] Zhai, S., Accot, J., Woltjer, (2002) R. Human Action Laws in Electronic Virtual Worlds – An Empirical Study of Path Steering Performance in VR, *Presence: Teleoperators and Virtual Environments*, Vol.13(2), 113-127.

[Zhai2004] Zhai, S., Kong, J., & Ren, X. (2004). Speed-accuracy tradeoff in Fitts' Law tasks: On the equivalency of actual and nominal pointing precision. *International Journal of Human-Computer Studies*. 61, 6, 823–856.

[Zhai2012] Zhai, S. and Kristensson, P.O. (2012). The word-gesture keyboard: reimagining keyboard interaction. *Communications of the ACM* 55(9): 91-101. Invited. (Research Highlights).

INTERACTION DESIGN.

SESSION 1

Dept. Computer Science – UPC

MOTIVATION

- **Usability & Design Principles**
- **Perception Laws**

DIRECT MANIPULATION INTERFACES (Pointing, choice selection)

- **Interaction Design and Evaluation:**
 - Design User Interfaces
 - Measure/Predict performance
 - Design interaction

OUTLINE

Session 1:

- Understanding the fundamentals of basic interaction in UI
 - Background (Information Theory)
 - Hick-Hyman Law: *Measuring Choice-Reaction Time*
 - Fitts' Law: *Measuring Pointing Time*
 - Crossing and Steering Laws: *Continuous Gestures*
- Fitts' Law in UI Design
 - Applications in UI Design
 - Accelerating Target Acquisition
- Exercises

Session 2:

- Typing & Keyboards
- Pointing Devices
- Mobile Interaction Design
- Exercises

OUTLINE

Session 1:

- Understanding the fundamentals of basic interaction in UI
 - **Background (Information Theory)**
 - Hick-Hyman Law: *Measuring Choice-Reaction Time*
 - Fitts' Law: *Measuring Pointing Time*
 - Crossing and Steering Laws: *Continuous Gestures*
- Fitts' Law in UI Design
 - Applications in UI Design
 - Accelerating Target Acquisition
- Exercises

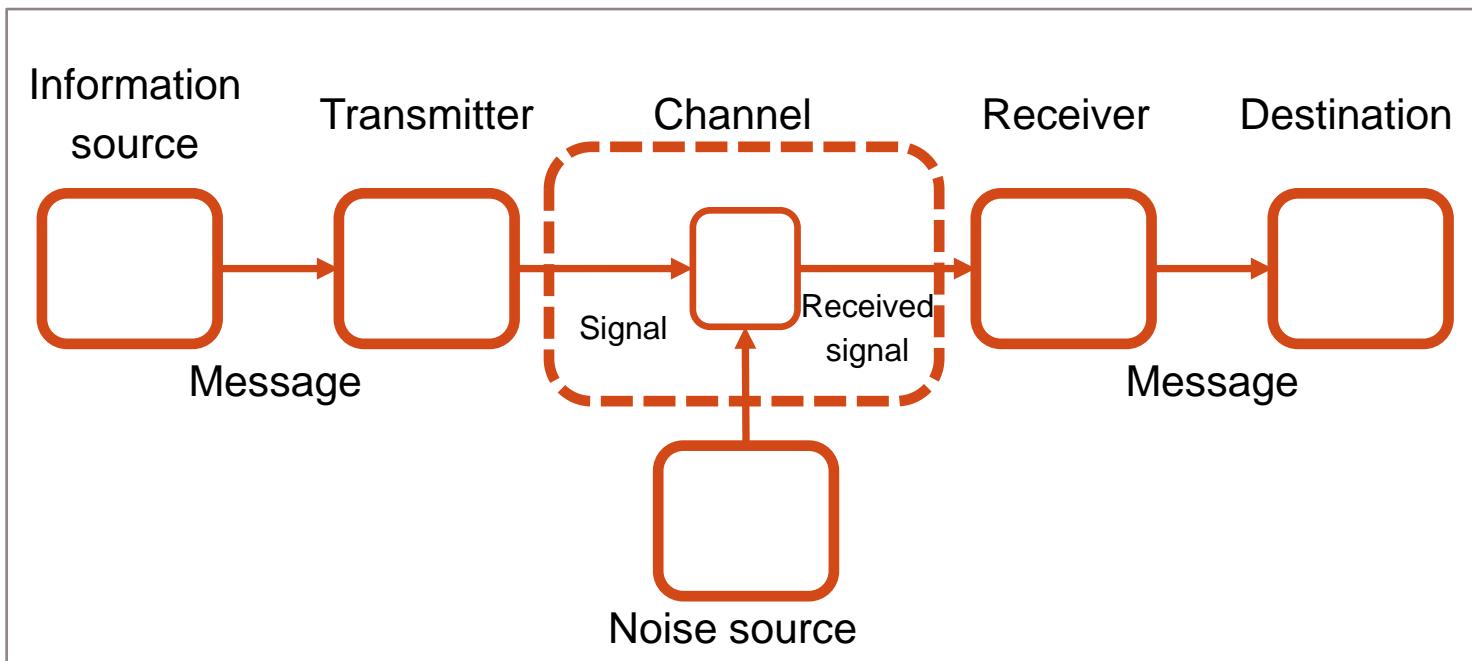
BACKGROUND. BASICS

- **Information Theory:**

- Due to Claude E. Shannon
 - *A Mathematical Theory of Communication* (1948)
 - Based on previous works by Nyquist and Hartley
 - Analysis of transmission of electrical signals for telegraphic communication
 - Shannon Entropy measures:
The amount of information to be transmitted by a message

BACKGROUND. BASICS

- Information Theory. Elements (telegraph):



BACKGROUND. BASICS

- Information Theory. Elements (telegraph):
 - **Information source:** The element that produces a message or sequences of message.
 - **Transmitter:** Operates on the message to make it transmissible through a medium.
 - **Channel:** The medium that transmits the message.
 - **Receiver:** The element that reconstruct the message to the destination.

BACKGROUND. INFORMATION MEASURES

- Let d be a device that produces symbols A, B, C and D with the same probability
 - $M = 4$ is the total number of symbols
 - Each time a symbol is produced we are uncertain on which symbol is going to be generated
 - This uncertainty is not so big, since there are only four possibilities
 - The probability of a symbol to appear is $1/M : 1/4$
- The **uncertainty** is measured by $\log_2(M) \rightarrow$ here $\log_2(4)=2\text{bits}$
- Logarithms are commonly taken in base 2, and the units are bits.

BACKGROUND. INFORMATION MEASURES

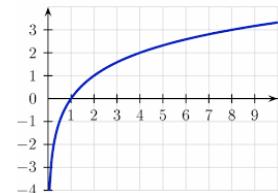
- **Example 1:** Let d be a device that produces one single symbol: C
 - $M = 1$ is the total number of symbols
 - We **have no uncertainty** and $\log_2(1) = 0$
 - The probability of getting the symbol C is 1
 - We previously know which symbol will appear!
- **Example 2:** Let d and e be two devices, one with outputs A, B, C, and the second with outputs 1, 2.
 - We combine *words* by concatenating one symbol of device d and one with device e .
 - We will have 6 different words: A1, A2, B1, B2, C1, C2
 - 6 symbols → uncertainty of $\log_2(6)$ → $\log_2(2) + \log_2(3) = \log_2(6)$.
- The uncertainty of combined the signals of a set of devices is the sum of their uncertainties.

BACKGROUND. INFORMATION MEASURES

- For M symbols with equal probability → each symbol has probability $P=1/M$

- Rewriting the uncertainty

$$\log_2(M) = \log_2\left(\left(\frac{1}{M}\right)^{-1}\right) = \log_2(P^{-1}) = -\log_2(P)$$



- $-\log_2(P)$ is called the **surprise** or *surprisal* of finding a certain symbol
 - We will use p_i from now on for the probability of a symbol i
- For M symbols that have different probabilities, we may have a different p_i for each, provided that

$$\sum_{i=1}^M p_i = 1$$

BACKGROUND. INFORMATION MEASURES

- **Information is the reduction of uncertainty or average surprise of a set of symbols**

- Measuring the surprise for an *infinite* set of N symbols (produced by a device) → the frequency of each symbol transforms to the probability.
- Shannon Entropy measures the amount of information:

$$H = \sum_{i=1}^N p_i \log_2 \left(\frac{1}{p_i} \right) = - \sum_{i=1}^N p_i \log_2 p_i$$

- N is the number of alternatives
- p_i is the probability of the i th alternative.
- **H is the entropy of the message that is to be transmitted,**
→ the amount of information expected to be received (no noise).

BACKGROUND. INFORMATION MEASURES

- **Example 1: Source with two equiprobable symbols: A and B**
- $p(A)=0.5, p(B)=0.5$
- $H= -0.5 \log_2(0.5) - 0.5 \log_2(0.5) = -\log_2(0.5) = -\log_2(2^{-1}) = 1$
- The source requires an average of 1 bit per symbol.

- **Example 2: Source with two symbols: A and B**
- $p(A)=0.1, p(B)=0.9$
- $H= -0.1 \log_2(0.1) - 0.9 \log_2(0.9) = 0,332 + 0,137 = 0,47$
- The source requires an average of 0,47 bit per symbol.

BACKGROUND. INFORMATION MEASURES

$$p(A)=0.1, p(B)=0.9$$

H=0,47 bits → Is it possible? We can achieve it using a smart codification of the information. For instance:

Symbols	Codification	Probability	Bits	Weighted bits
AA = 00	000	0,1*0,1=0,01	3	0,03
AB = 01	001	0,1*0,9=0,09	3	0,27
BA = 10	01	0,1*0,9=0,09	2	0,18
BB = 11	1	0,9*0,9=0,81	1	0,81
		1		1,29bits in average to send 2 symbols
				0,645 bits per symbol

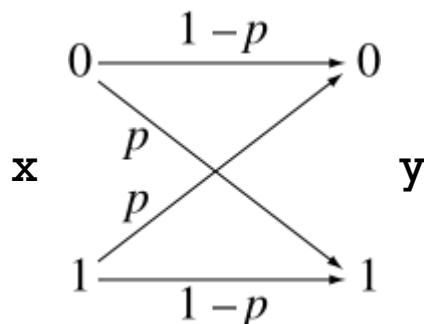
BACKGROUND. INFORMATION MEASURES

- Information Theory. Shannon entropy:

- There is interference: Not all information will reach the receiver
- Average information faithfully transmitted (R):

$$R = H(x) - H_y(x)$$

- $H_y(x)$ is the equivocation or conditional entropy of x when y is known. Measures the information required to quantify the error.



$$H_y(x) = \sum_{i=0}^N \sum_{j=0}^N p(x_i, y_j) \cdot \log_2(p(x_i | y_j))$$

p : error probability

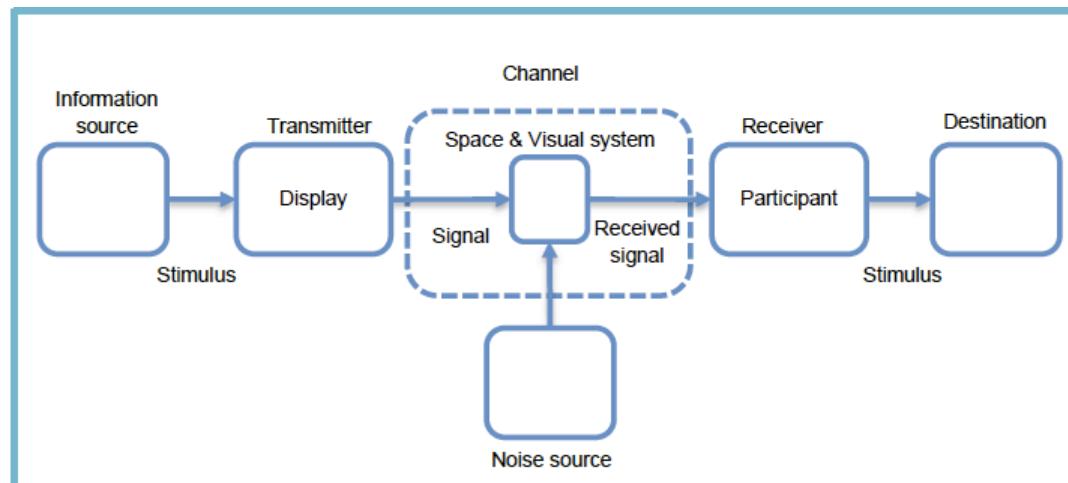
OUTLINE

Session 1:

- Understanding the fundamentals of basic interaction in UI
 - *Background (Information Theory)*
 - **Hick-Hyman Law: Measuring Choice-Reaction Time**
 - Fitts' Law: *Measuring Pointing Time*
 - Crossing and Steering Laws: *Continuous Gestures*
- Fitts' Law in UI Design
 - Applications in UI Design
 - Accelerating Target Acquisition
- Exercises

HICK-HYMAN LAW

- Hick-Hyman Law:
 - Initially stated by William E. Hick (1951)
 - Describes human decision time as a function of the information content conveyed by a visual stimulus
 - It takes longer to respond to a stimulus when it belongs to a large set as opposed to a smaller set of stimuli
 - Extended by Ray Hyman (1952)



HICK-HYMAN LAW

- Time to make a decision (Reaction Time):

$$RT = a + bH_T$$

- a, b constants
- H_T transmitted information

HICK-HYMAN LAW

- Hick-Hyman Law:

- H_T : Transmitted information:

$$H_T = \log_2(n + 1)$$

- n are the equiprobable alternatives
 - original formulation did not have the “+1” attends for the uncertainty whether to respond or not

- Time to answer is the Reaction Time:

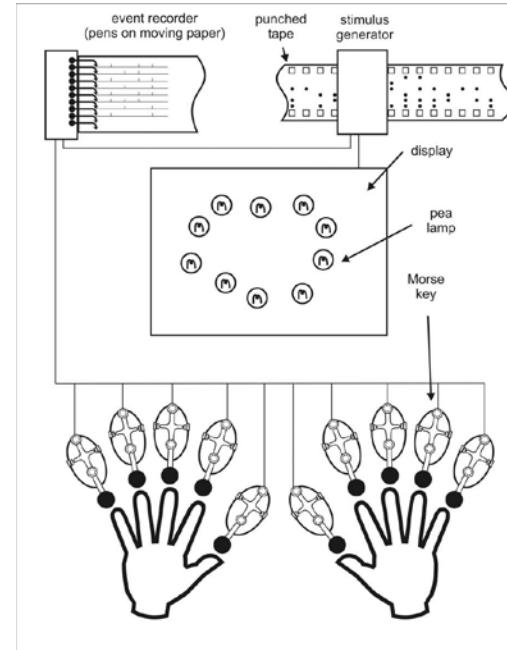
$$RT = a + b \log_2(n + 1)$$

HICK-HYMAN LAW.

EXPERIMENTAL ASSESSMENT

- Hick's initial experiment:

- 10 pea lamps are arranged in an irregular circle
- One random lamp is lit every 5 seconds
- User has to press the correct key corresponding to the lamp that is lit
- Stimulus and response encoded in a moving paper in binary code



HICK-HYMAN LAW. EXPERIMENTAL ASSESSMENT

- Time to answer. Reaction Time is a linear function of stimulus information

$$RT = a + b \log_2(n + 1)$$

- Hyman [Hyman53] found that it
also holds for not equiprobable alternatives

- Experiment:
 - 8 lights (whose names were *Bun*, *Boo*, *Bee*, *Bore*, *By*, *Bix*, *Bev*, and *Bate*)
 - The users had to name the one lit
 - A microphone attached to the throat detected the voice and stopped the timer
 - First with equal probabilities
 - Then, with varying probabilities

HICK-HYMAN LAW. EVIDENCES

- **Evidences of Hick-Hyman Law**
 - Performance *in hierarchical full-screen menu selections* is well described by Hick-Hyman [Landauer85]
 - Selection times decay logarithmically with menu length for frequently selected items, but linearly with infrequent ones [Sears94].
 - Learnt locations (most frequent) fit Hick-Hyman decision times
 - Non-learnt locations fit a linear search
 - Novice users search linearly while experts decide upon item location and fit a Hick-Hyman curve [Cockburn2008]

OUTLINE

Session 1:

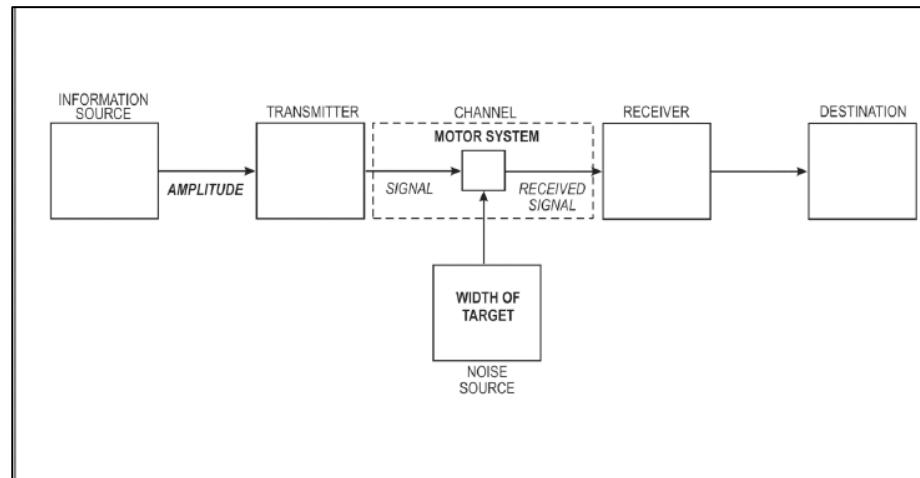
- Understanding the fundamentals of basic interaction in UI
 - *Background (Information Theory)*
 - *Hick-Hyman Law: Measuring Choice-Reaction Time*
 - **Fitts' Law: Measuring Pointing Time**
 - Crossing and Steering Laws: *Continuous Gestures*
- Fitts' Law in UI Design
 - Applications in UI Design
 - Accelerating Target Acquisition
- Exercises

FITTS' LAW. ORIGINAL FORMULATION

- States a **linear relationship between the movement time (MT) and task difficulty**

$$MT = a + bID$$

- Formulation is also based on Information Theory
 - Amplitude of movement is the *signal*
 - Human motor system is the communication *channel*
 - Target width is the *noise*



FITTS' LAW. ORIGINAL FORMULATION

- **Task difficulty:**

$$ID = \log_2 \left(\frac{2A}{W} \right)$$

- ID : Index of difficulty
- A : Amplitude of movement
- W : Target width

- The larger the amplitude the higher the difficulty
- The larger the target the lower the difficulty

FITTS' LAW. ORIGINAL FORMULATION

- **Movement Time:** Time to point a certain objective (target)

$$MT = a + bID$$

- a start/stop times in seconds
- b inherent speed of the device

$$ID = \log_2 \left(\frac{2A}{W} \right)$$

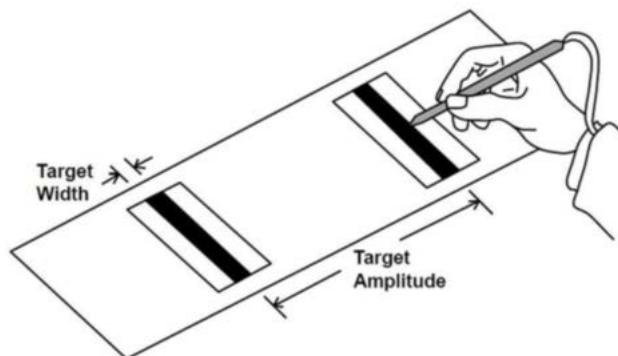
- A: Amplitude of movement
- W: Target width

FITTS' LAW. EXPERIMENTAL EVIDENCES

Fitts' Law. Original experiments:

- Experiment 1: Reciprocal tapping:

- Participants used a metal-tipped stylus:
 - Two experiments with two different stylus: ~ 28.35 and 453.6 gr
 - Tap two strips of metallic targets of width from ~ 0.635 to 5.08 cm
 - At distance 5.08 to 40.64 cm
 - Participants instructed to be accurate!



FITTS' LAW. EXPERIMENTAL EVIDENCES

Fitts' Law. Original experiments:

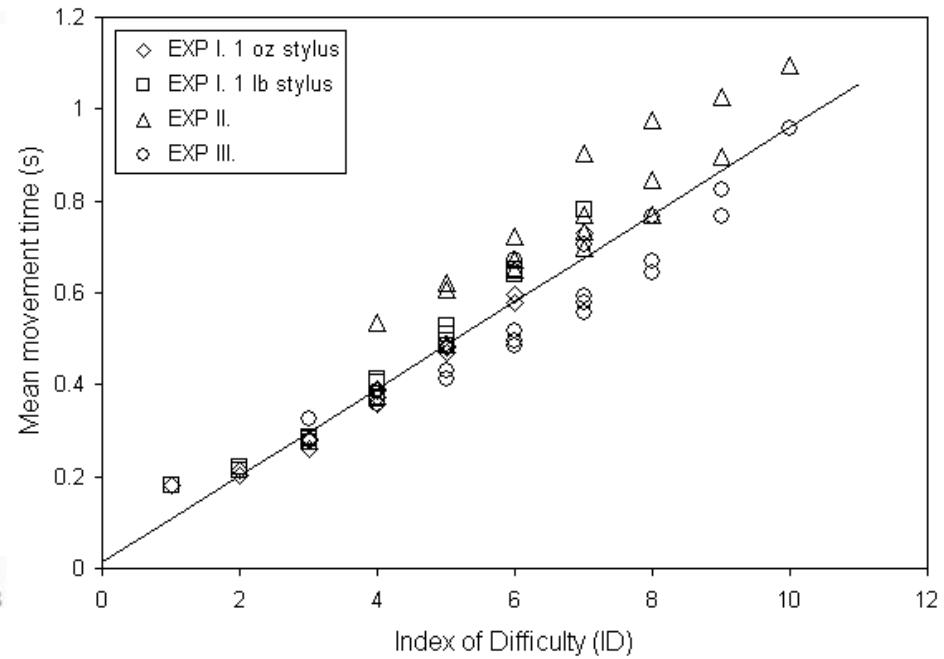
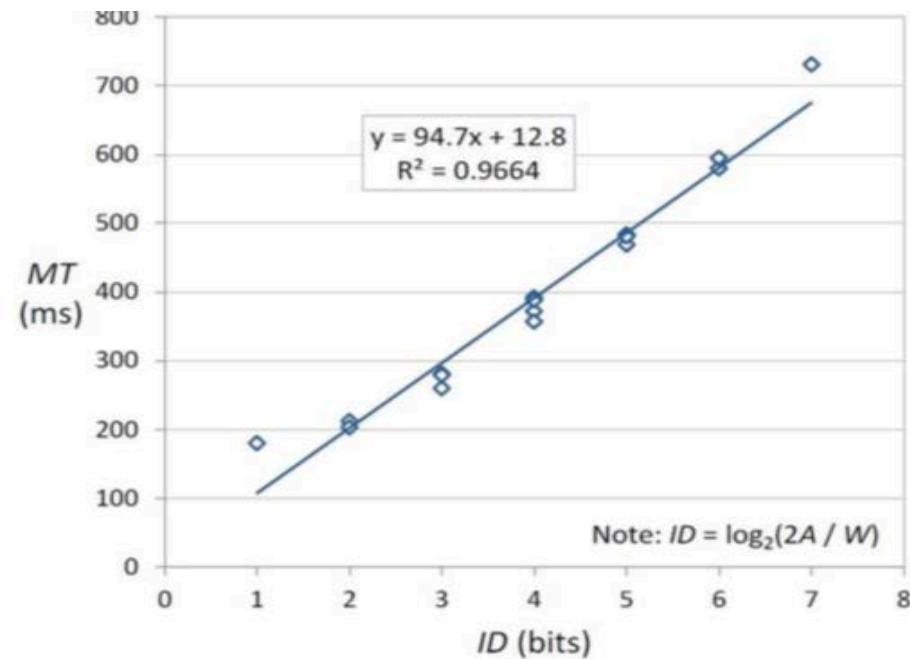
- Experiment 2: Disk transfer
 - Participants had to transfer stack round plastic disks (with holes drilled through the middle) from one pin to another
 - Holes of different sizes and pins of different diameters used
- Experiment 3: Pin transfer
 - Participants had to transfer pins of different diameters from a set of holes to another set of holes

FITTS' LAW. EXPERIMENTAL EVIDENCES

$$ID = \log_2 \left(\frac{2A}{W} \right)$$

■ Fitts' Law Results.

$$MT = 12.8 + 94.7 ID$$



- Results show that there is a linear relationship between MT and ID
- **Most difficult condition: Smaller W and largest A**
- Only valid for the experiments carried out
 - One curve per experiment fits better (different a and b values)

FITTS' LAW. VARIANTS

- Original formulation fits well to the original experiments
 - But it might fit better
- Other researchers have found different formulations that better model the experimental data
 - Including the experimental data by Fitts
- Welford [Welford68]:

$$MT = a + b \log_2 \left(\frac{D + 0.5W}{W} \right)$$

- D is the distance of movement
- W is the width of the target

FITTS' LAW. VARIANTS

- **MacKenzie's** approach [MacKenzie92] is one of the most accepted:

$$MT = a + b \log_2 \left(\frac{D}{W} + 1 \right)$$

- D is the distance of movement
- W is the width of the target

FITTS' LAW. VARIANTS

- Vertical and horizontal movements can be treated equally

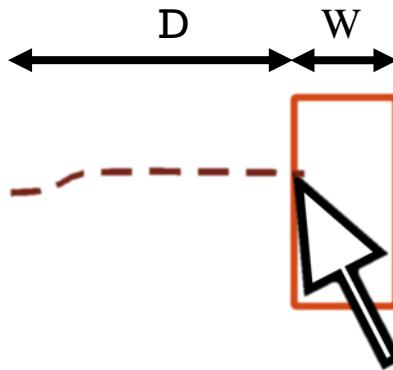


FITTS' LAW EXTENSIONS

- Main application of Fitts in HCI is evaluation/design of UI and interaction
- Today's interfaces are much more complex
 - Variety of sizes
 - 2D movements
 - Use of fingers

FITTS' LAW. EXTENSIONS

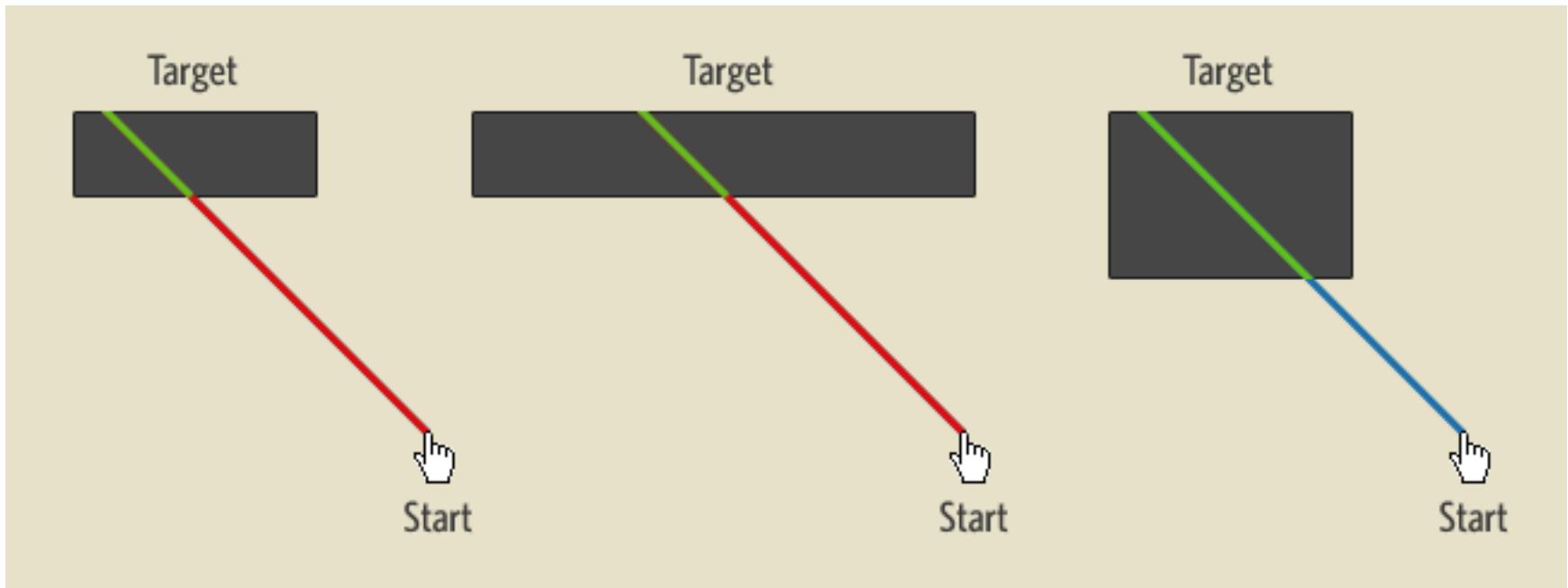
- Use in UI design or evaluation:



- D is the distance the pointer (mouse) covers to reach the target (button)
- W is the width of the target (button)

FITTS' LAW. EXTENSIONS 2D

- Fitts' Law is designed for 1D movements
BUT...most movements in a UI are 2D
- Vertical and horizontal movements can be treated equally... or not?



FITTS' LAW. EXTENSIONS 2D

- Several extensions deal with 2D movements
 - Mimicking Fitts' Law, but changing some of the parameters

- [Crossman83]:

$$MT = a + b \log_2 \left(\frac{2D}{W} \right) + c \log_2 \left(\frac{2D}{H} \right)$$

- [Accot97]:

$$MT = a + b \log_2 \left(\sqrt{\left(\frac{D}{W} \right)^2 + \eta \left(\frac{D}{H} \right)^2} + 1 \right)$$

FITTS' LAW EXTENSIONS: PRECISION POINTING

- Fitts Law does not model properly very small targets:
 - Extra time devoted to fine adjustment
 - Increase of errors
 - ...
- Very small targets yield a lower fit of the regression curve of the MT function
- Touchscreens also modifies the timing we require to point targets.

FITTS' LAW EXTENSIONS: PRECISION POINTING

Extension of Fitts' Law by analyzing the behavior both in tactile screens and small targets ([Sears91]):

- Named FFitts (**Finger Fitts**), also PPMT (Precision Pointing Movement Time) by some other authors :

$$FFits = a + bID + dID_2$$

$$FFitts = a + b \left[\log_2 \left(\frac{cD}{W} \right) \right] + d \left[\log_2 \left(\frac{e}{W} \right) \right]$$

- The higher number of freedom degrees, the easier to fit in a regression curve

FITTS' LAW. EXTENSIONS: PRECISION POINTING

- **FFitts:**

$$FFitts = a + b \left[\log_2 \left(\frac{cD}{W} \right) \right] + d \left[\log_2 \left(\frac{e}{W} \right) \right]$$

- the first logarithmic factor measures *the time to place the finger on the screen initially*
- the second factor measures *the time to position the cursor*
- D is the distance, measured in three dimensions, from the original hand location to the location of first contact
- If the task consists of iteratively clicking targets: D is the distance from one target to the next one
- W is some measurement of target size
- a, b, c, d , and e must be determined for each specific case

FITTS' LAW. ASSESSED RESULTS

- Validation of Fitts' Law may not extrapolate to outside the experiments carried out
 - ***Validity Fitts → Experimentation***
- Fitts' Law have been formulated in a number of ways, however its prediction is consistent:
 - "the ID to acquire a target is functions of the distance to and the size of the target"

FITTS' LAW. ASSESSED RESULTS

- Fitts' Law has shown its validity in multiple setups and devices:
 - Mouse, joystick, finger, stylus...
 - Different screen types of varying sizes...
 - **But the results cannot be extrapolate to data outside the experiment. Validity Fitts → Experimentation**
- Fitts' law is a really good predictive model of human movement.
- Precued targets lead to more efficient and precise pointing movements than for non-precued targets [Hertzum2013].
 - Most common case: we know the buttons' positions in advance.
 - The benefit of precuing is larger for the mouse than the touchpad
 - Maybe movement preparation is more effective if the device is more demanding

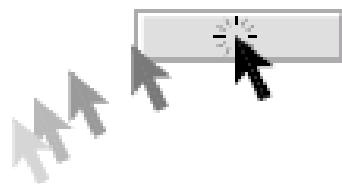
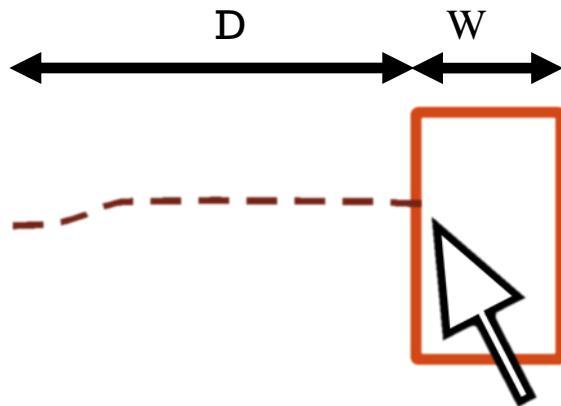
OUTLINE

Session 1:

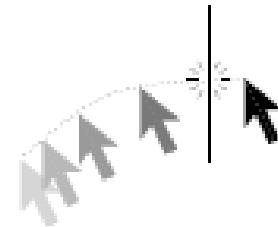
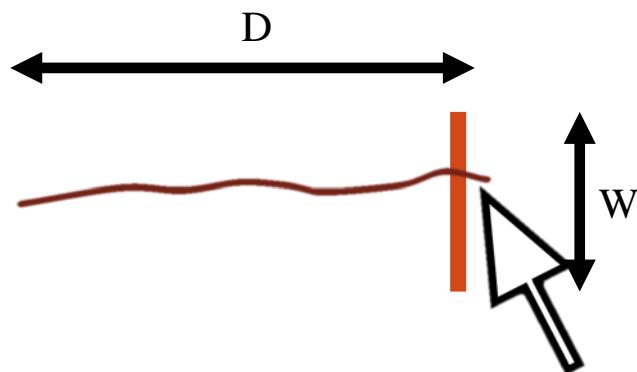
- Understanding the fundamentals of basic interaction in UI
 - *Background (Information Theory)*
 - *Hick-Hyman Law: Measuring Choice-Reaction Time*
 - *Fitts' Law: Measuring Pointing Time*
 - **Crossing and Steering Laws: Continuous Gestures**
- Fitts' Law in UI Design
 - Applications in UI Design
 - Accelerating Target Acquisition
- Exercises

LAW OF CROSSING

- Crossing movement as compared to pointing



(a) Pointing a target



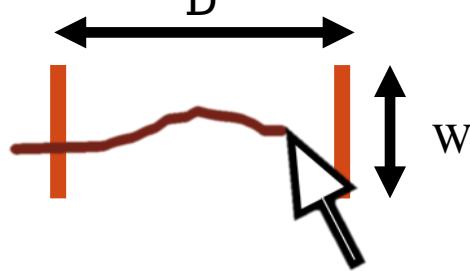
(b) Crossing a goal

LAW OF CROSSING

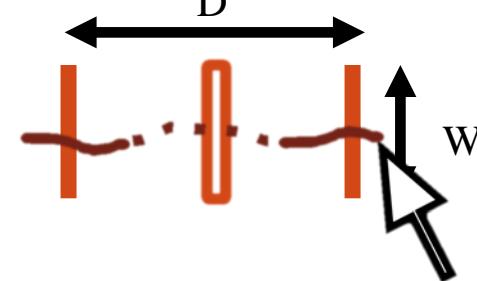
▪ Crossing configurations:

- Discreteness vs continuity of the movement:
 - Landing and lifting off the stylus

Continuous crossing



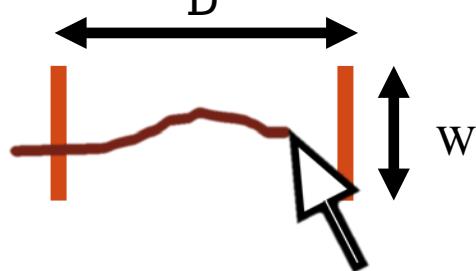
Discrete crossing



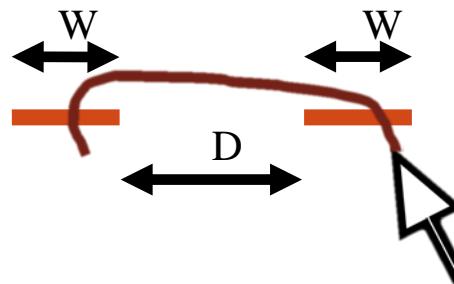
- Direction of the targets vs direction of the movement:

- If parallel, the trace will be larger

Orthogonal crossing

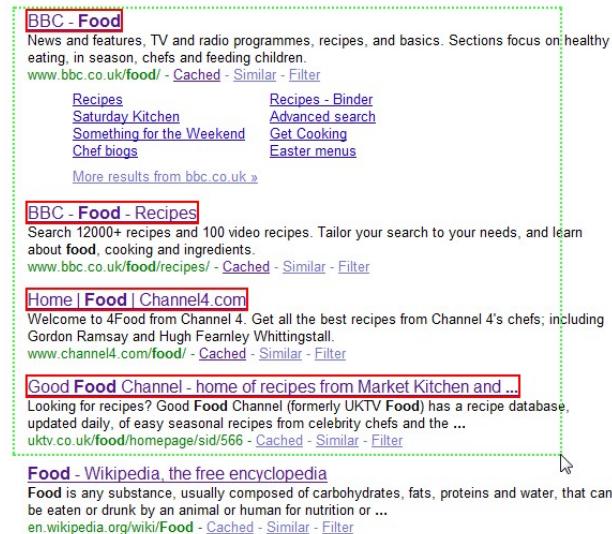


Collinear crossing



LAW OF CROSSING

- Stylus or fingers naturally lead to crossing gestures
 - Especially useful in tactile devices
 - Crossing an object is easier than double-clicking.
 - Drag & drop, multiple selections
 - Crossing can be a good alternative for users who have difficulties with clicking or double-clicking.
- Several objects can be crossed at the same time within the same gesture



Multi-links
extension for
Chrome
(LinkClump)

LAW OF CROSSING

- Crossing performance across two goals [Accot99, Zhai2002]:
 - Follows the same characterization than the Fitts' Law:

$$T = a + b \log_2 \left(\frac{D}{W} + 1 \right)$$

- T is the average moving time between passing the two goals.
- D is the distance between the two goals
- W is the width of each goal
- a and b are constants to be determined

LAW OF CROSSING

- **Results of the experiments:**

- Crossing-based interfaces achieve similar (or faster) times than pointing.
- The error rate in crossing is smaller than in pointing.
- Discrete crossing becomes more difficult if the distance between the targets is small.

LAW OF CROSSING



<https://www.youtube.com/watch?v=C5L4vV3T2mU>

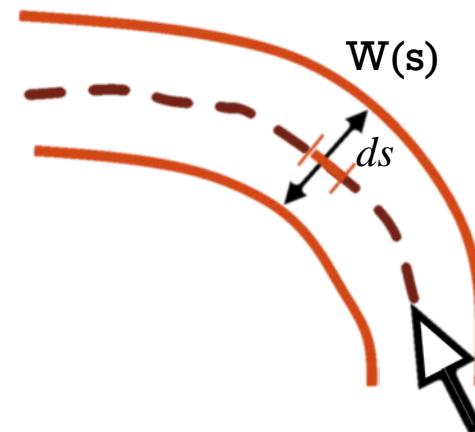
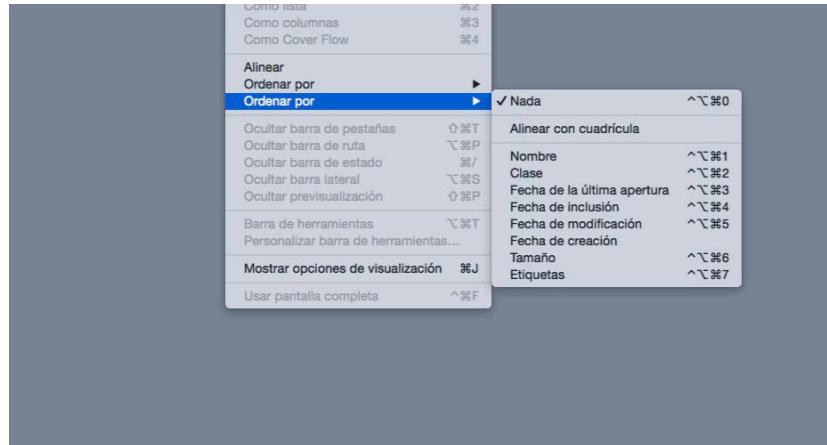
OUTLINE

Session 1:

- Understanding the fundamentals of basic interaction in UI
 - *Background (Information Theory)*
 - *Hick-Hyman Law: Measuring Choice-Reaction Time*
 - *Fitts' Law: Measuring Pointing Time*
 - **Crossing and Steering Laws: Continuous Gestures**
- Fitts' Law in UI Design
 - Applications in UI Design
 - Accelerating Target Acquisition
- Exercises

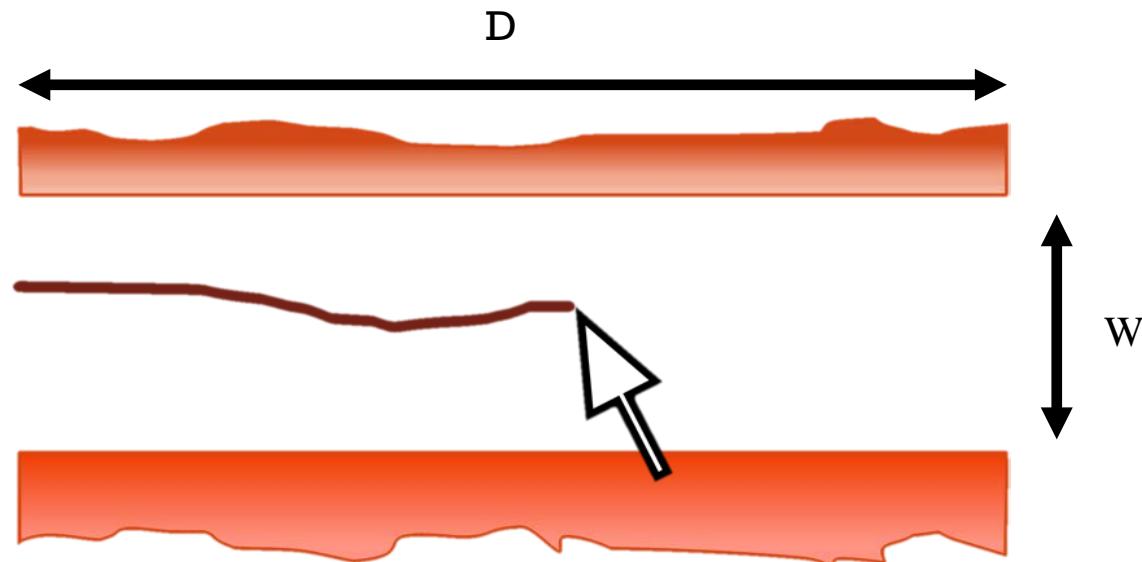
STEERING LAW

- Navigating through a constrained path is an useful operation in modern UIs
 - Navigating through nested menus
 - 3D navigation
 - Dragging elements
 - Free-hand Sketching/Drawing



STEERING LAW

- Steering through a **straight path**:

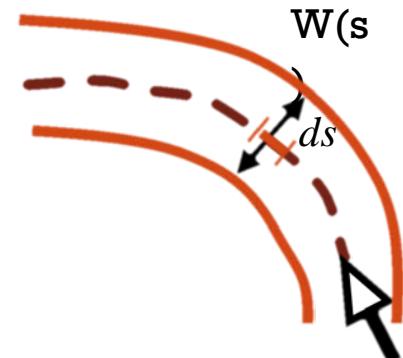


STEERING LAW

- Navigating through a **generalized path** can be expressed as infinite crossings [Accot97]
- Movement time across the path T_s :

$$T_s = a + bID_s \quad T_s = a + b \int_C \frac{ds}{W(s)}$$

- C is the length of the path
- $W(s)$ is the path width at point s



STEERING LAW

- Time to navigate through a **straight path** (tunnel) T_p [Accot97]:

$$T_s = a + b \int_c \frac{ds}{W(s)} \quad T_p = a + b \frac{D}{W}$$

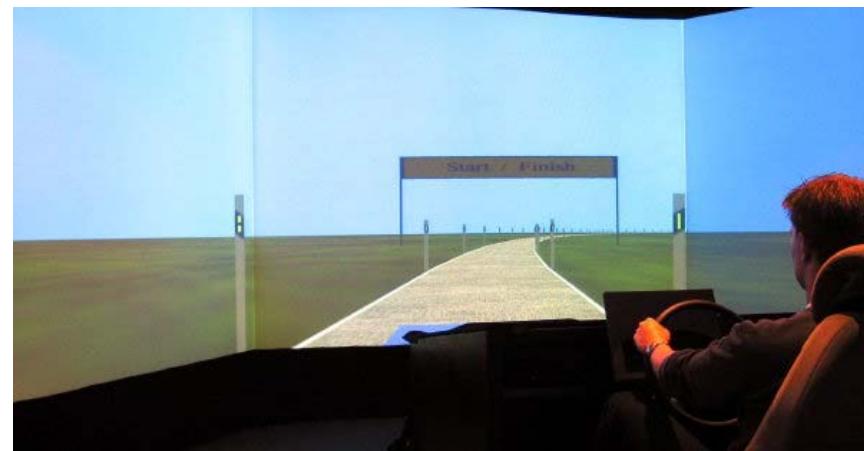
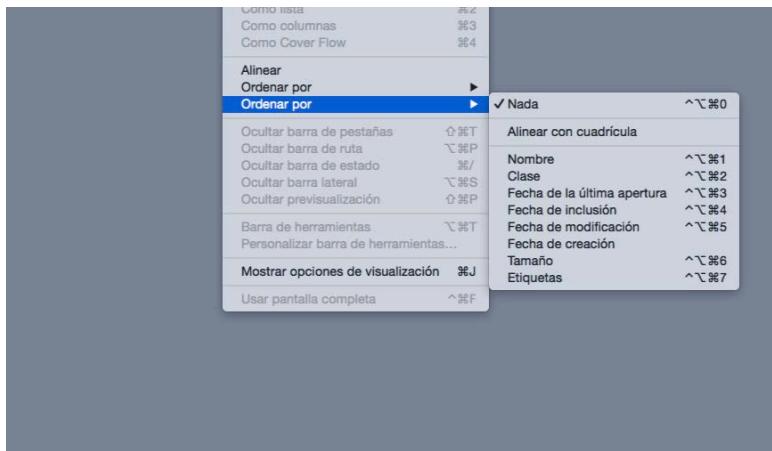
- D is the length of the path/tunnel
- W is the width of the path/tunnel
- Applying Fitts' formatting:

$$T_p = a + bID_p \quad ID_p = \frac{D}{W}$$

- Which also applies to circular paths of constant width

STEERING LAW

- Results [Accot97, Zhai2004] show that the steering law is applicable to different configurations:
 - Different path shapes: cone, spiral, straight
 - Works with different devices, works in VR...
 - Can be used to analyse navigation through nested menus, compare menu designs...



OUTLINE

Session 1:

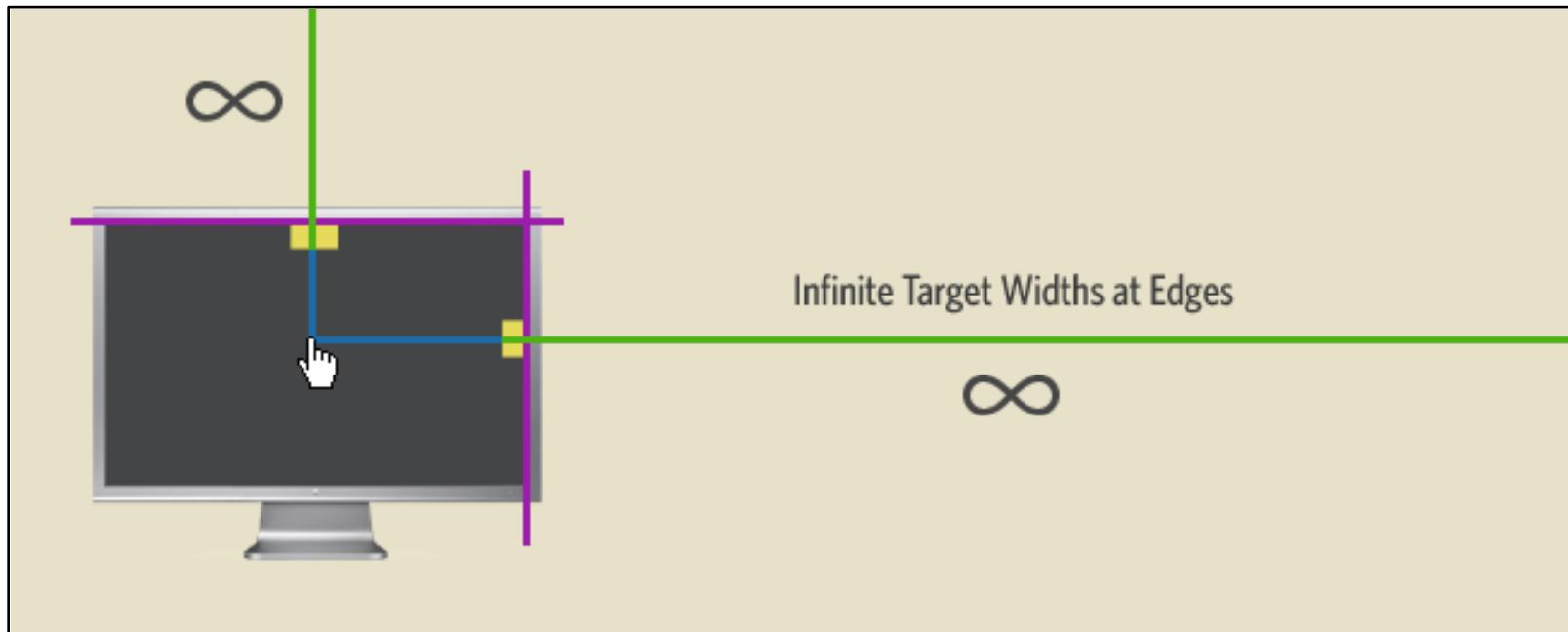
- *Understanding the fundamentals of basic interaction in UI*
 - *Background (Information Theory)*
 - *Hick-Hyman Law: Measuring Choice-Reaction Time*
 - *Fitts' Law: Measuring Pointing Time*
 - *Crossing and Steering Laws: Continuous Gestures*
- **Fitts' Law in UI Design**
 - Applications in UI Design
 - Accelerating Target Acquisition
- Exercises

FITTS' LAW IN UI DESIGN. IMPLICATIONS

- Fitts' Law accurately predicts **pointing** movement
 - Further distance → Harder to select
 - Larger target → Easier to select
- If improvement required, it can help us modify our UI
 - **Change target width:**
 - Increase size for faster reach
 - **Change de “virtual distance” or pointer movement:**
 - Increase speed, pop-up menus,....
- But visual stimuli must also be taking into account...

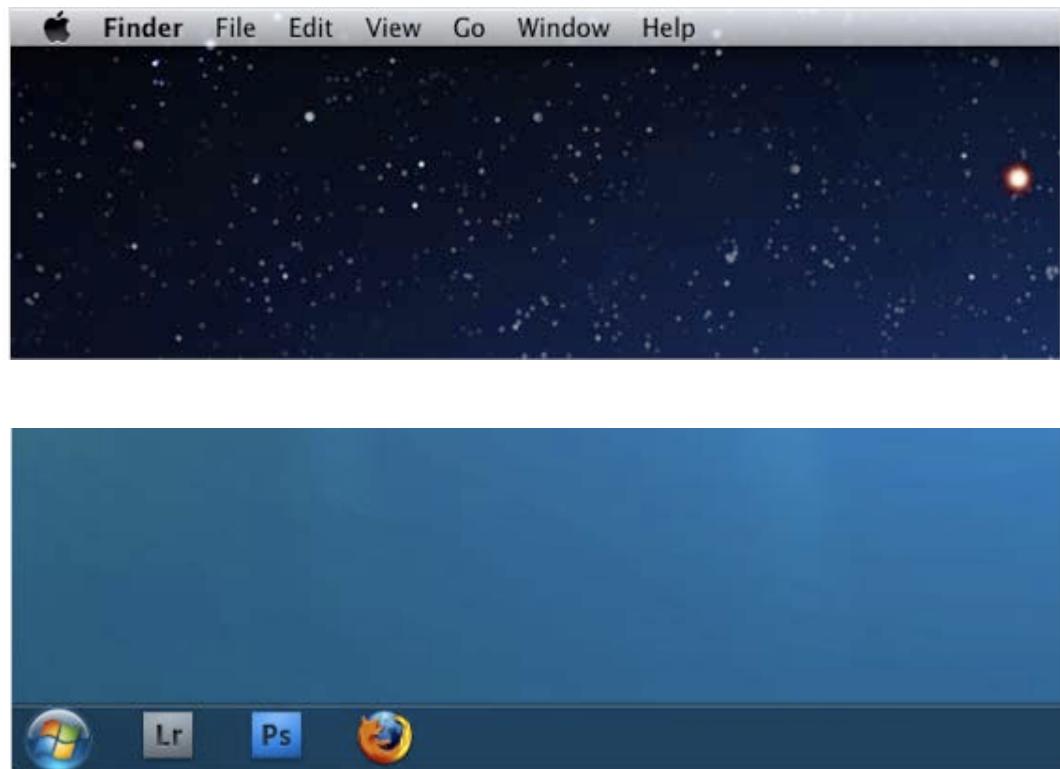
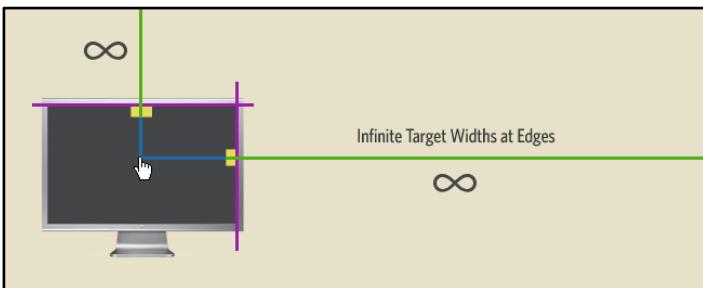
FITTS' LAW IN UI DESIGN. APPLICATIONS

The outer edges and corners

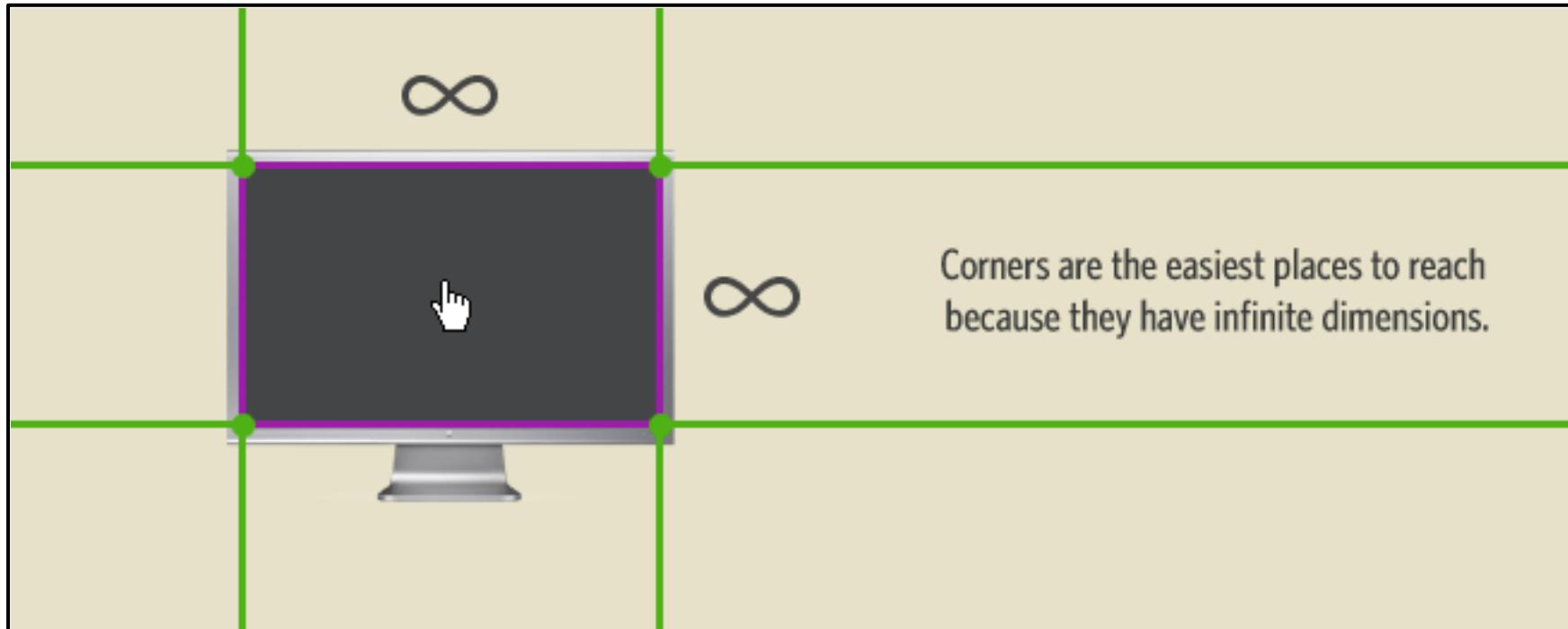


$$MT = a + b \log_2 \left(\frac{D}{W} + 1 \right)$$

FITTS' LAW IN UI DESIGN. APPLICATIONS



FITTS' LAW IN UI DESIGN. APPLICATIONS



FITTS' LAW IN UI DESIGN. APPLICATIONS

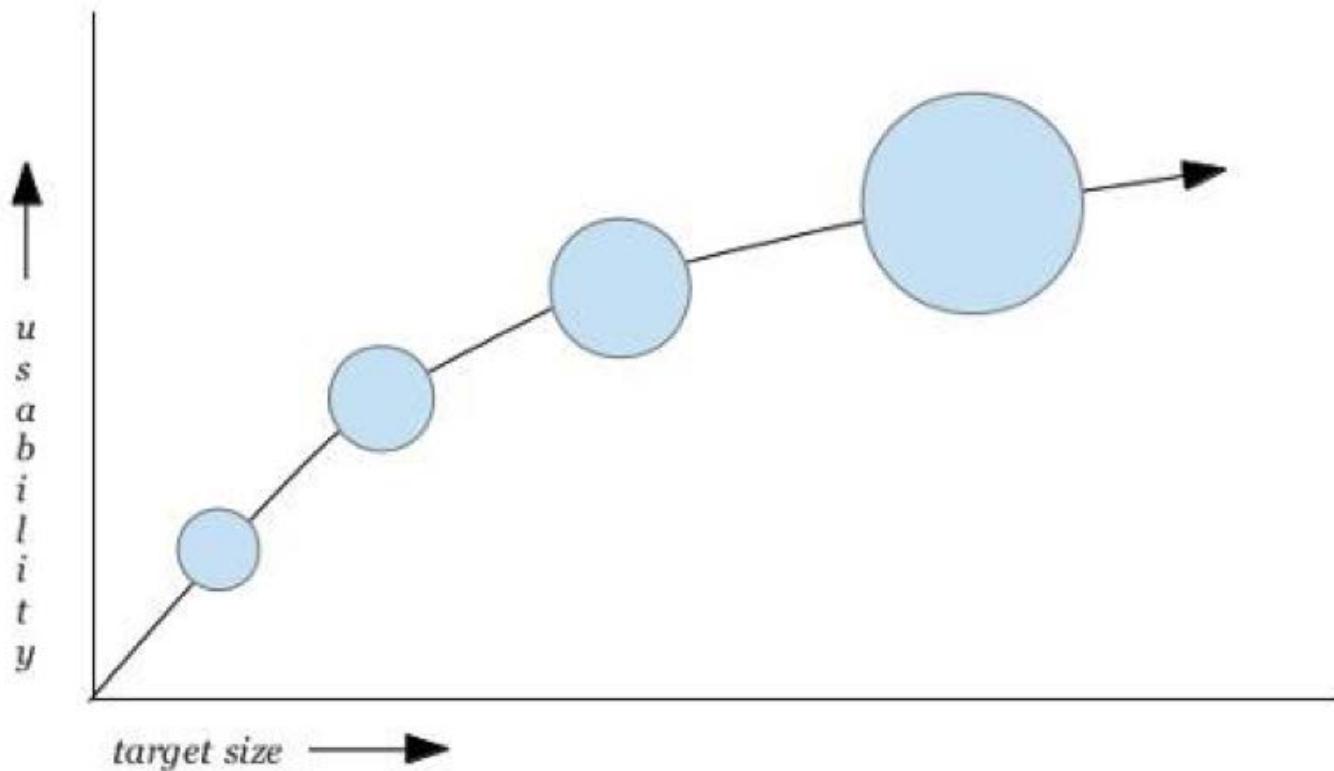


Web sites do not have edges or corners of infinite width.

:(|

FITTS' LAW IN UI DESIGN. APPLICATIONS

Create larger target size



FITTS' LAW IN UI DESIGN. APPLICATIONS

Keep related things close

- Filters should be placed close to the search field



FITTS' LAW IN UI DESIGN. APPLICATIONS

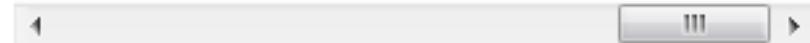
Keep related things close

- Mac OS scrolls are faster to navigate

OSX Snow Leopard



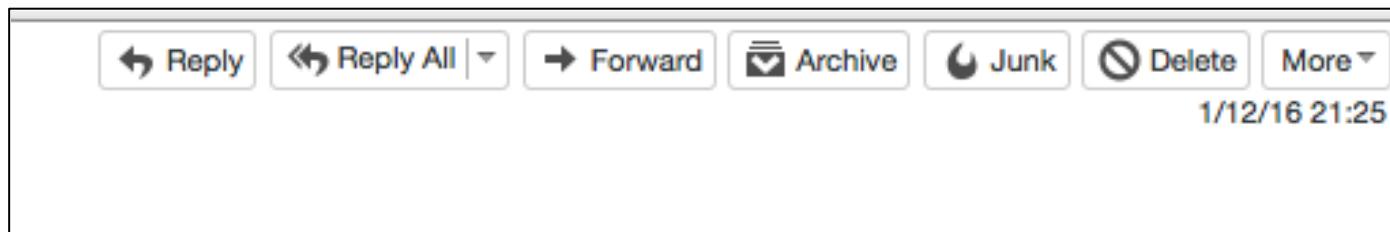
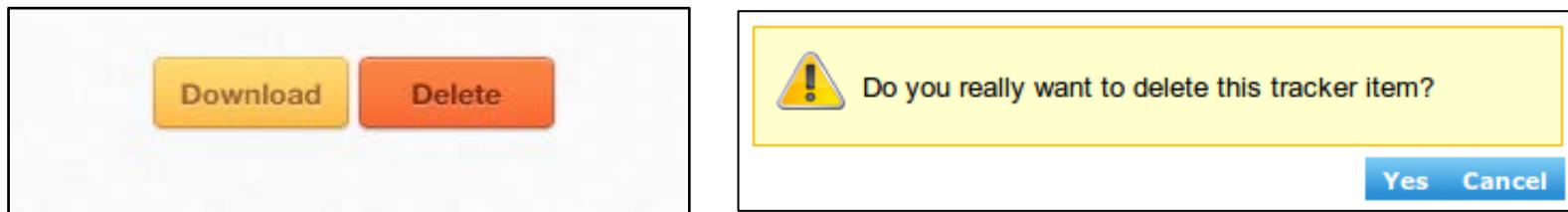
Windows



FITTS' LAW IN UI DESIGN. APPLICATIONS

Keep related things close and **Opposite Elements Far**

- These buttons should be placed far away from each other



But...don't forget the usability principles!!!

FITTS' LAW IN UI DESIGN. APPLICATIONS

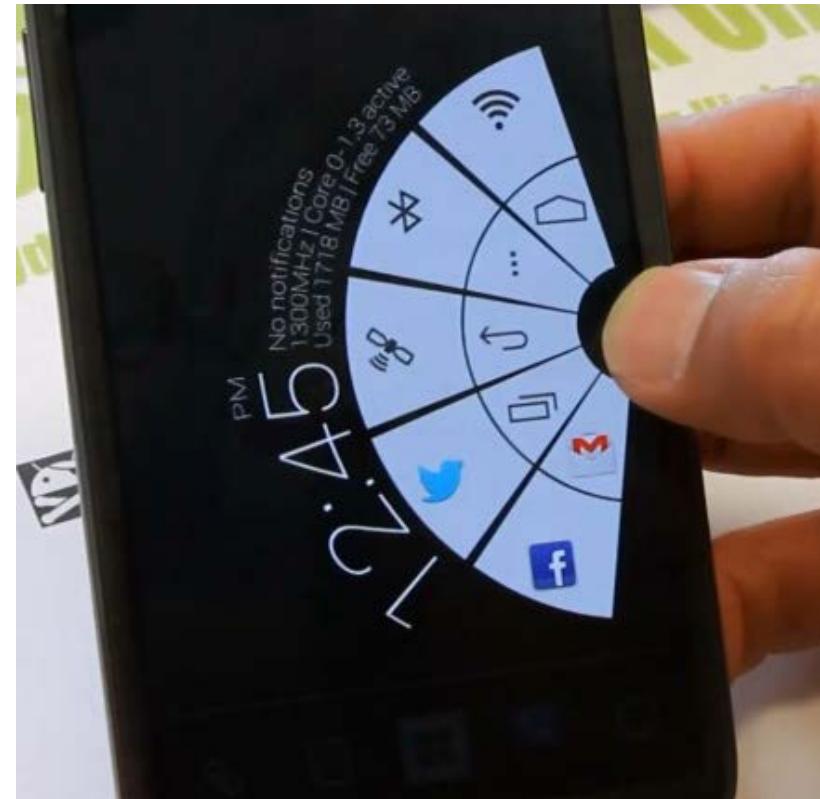
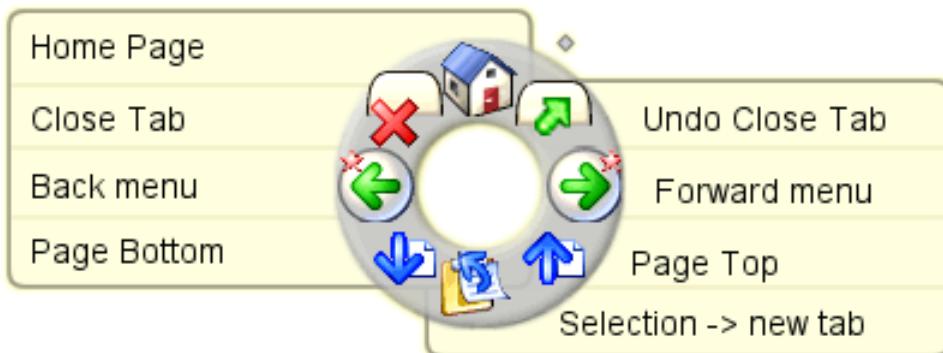
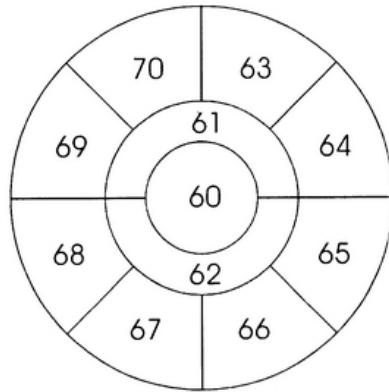
Pop-up menus: Reduce travelling distance

- **Improve two aspects:**
 - Reduction of distance to travel (Fitts)
 - The option is close to the menu emerging place
 - Frequency-enabled may improve the time to pick an option:
 - Based on Hick-Hyman:
Recall that users are able to point faster objects that are known
- Only used by experts!



FITTS' LAW IN UI DESIGN. APPLICATIONS

- *What about pie menus?*



FITTS' LAW IN UI DESIGN. APPLICATIONS

- *What about pie menus?*

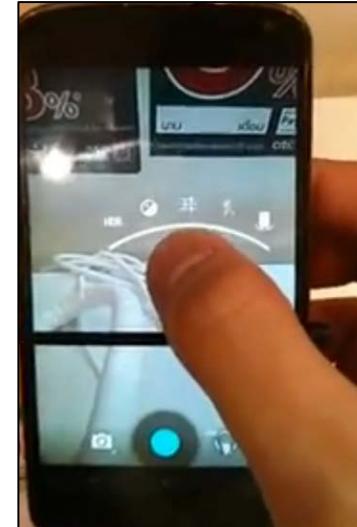
- Sort of contextual menu

- Needs to be created on demand

- **Needs some room!**

- Should not have occlusions

- On mobile half-pie menus better than fully circular



FITTS' LAW IN UI DESIGN. APPLICATIONS

Pie menus difficult to design!

- Second layer changes the size and distance
- Organizing by frequency may be a problem (learning)



FITTS' LAW IN UI DESIGN. APPLICATIONS

+ Perception: Grouping things may improve over distance



OUTLINE

Session 1:

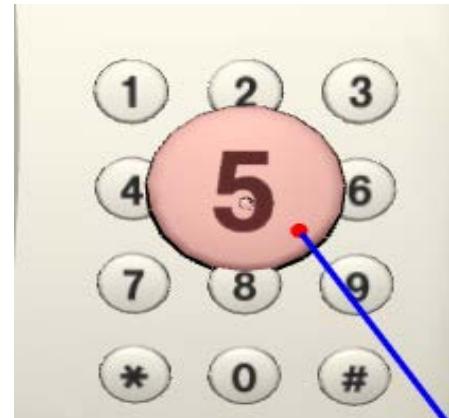
- *Understanding the fundamentals of basic interaction in UI*
 - *Background (Information Theory)*
 - *Hick-Hyman Law: Measuring Choice-Reaction Time*
 - *Fitts' Law: Measuring Pointing Time*
 - *Crossing and Steering Laws: Continuous Gestures*
- **Fitts' Law in UI Design**
 - *Applications in UI Design*
 - **Accelerating Target Acquisition:**
 - Dynamic Expanding Targets
 - Target moving
- Exercises

ACCELERATING TARGET ACQUISITION: EXPANDING TARGETS

➤ Increase the size of targets close to the pointer

Two implementation approaches:

- Size-enlargement and position-changing icons
- Enlarged icons overlap over their neighbours

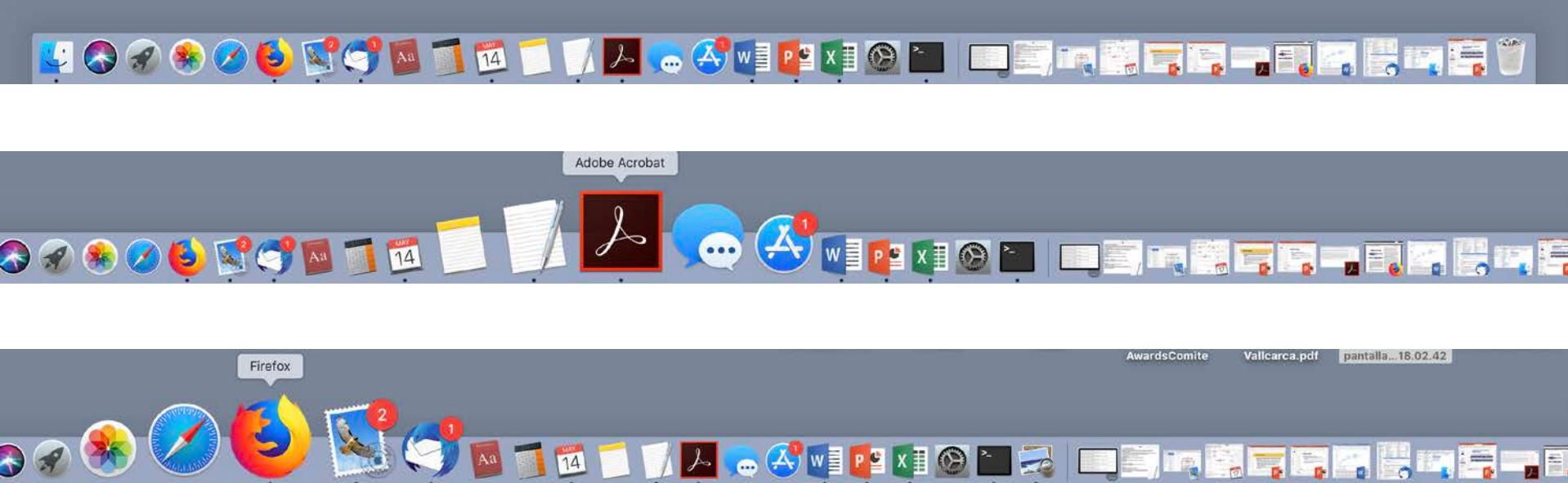


ACCELERATING TARGET ACQUISITION: EXPANDING TARGETS

Increase the size of targets close to the pointer

Exemple 1: Implemented in Mac OSX Dock:

- Mix of target size increase and moving target



ACCELERATING TARGET ACQUISITION: EXPANDING TARGETS

Enlarged icons overlap over their neighbours

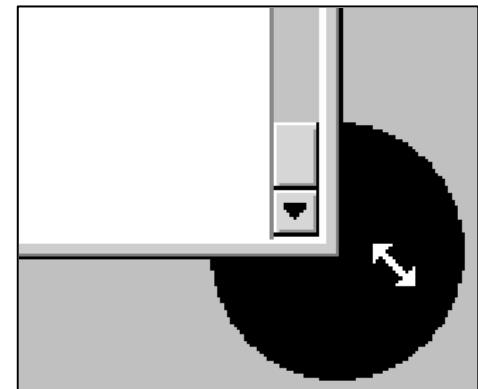
Dynamic Scaling (DS)

Objects near the selection ray are dynamically scaled

ACCELERATING TARGET ACQUISITION: EXPANDING TARGETS

➤ **Bubble targets:**

- Increase selectable region around target
 - Only when the mouse is close
 - Improves selection times
- Issues:
 - Bubble appearing may distract users
 - Overlapping targets:
Close selection points may generate several bubbles

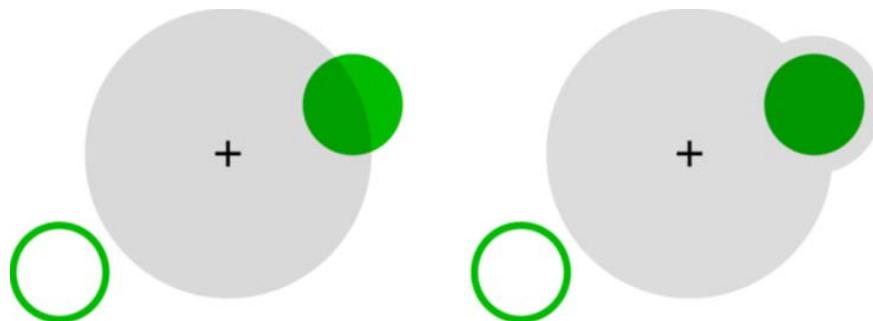


ACCELERATING TARGET ACQUISITION: EXPANDING TARG

➤ **Bubble cursor** [Grossman2005] →

Reduction of amplitude movement

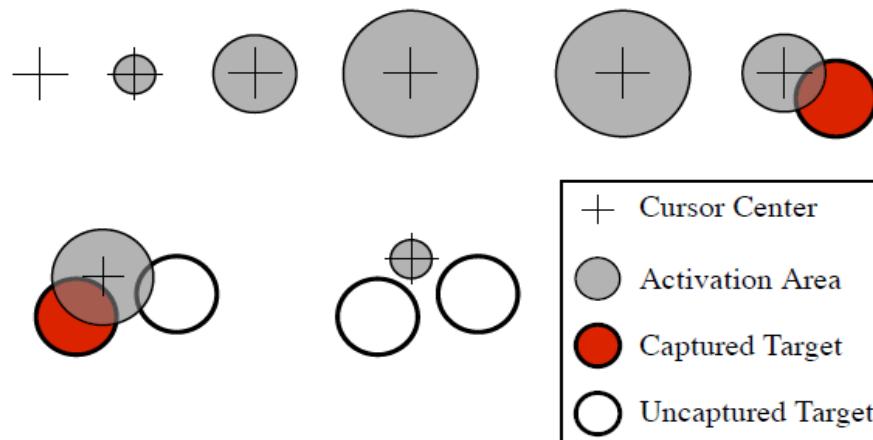
- Cursor size increases when it is close to objectives
- It may even grow to *absorb* the closer target when it is not completely inside the main cursor bubble.
 - Based on position, no speed
 - In experiments Control-Display ratio fixed to 1



ACCELERATING TARGET ACQUISITION: EXPANDING TARG

➤ **Dynamic Bubble cursor** [Chapuis2009]:

- Based on the Bubble cursor idea
- It takes into account the speed of the mouse
 - Area increases according to speed and position
 - Visual cues to indicate the captured target: the target closer to the cursor center.



ACCELERATING TARGET ACQUISITION: TARGET MOVING

- Move targets to the user:



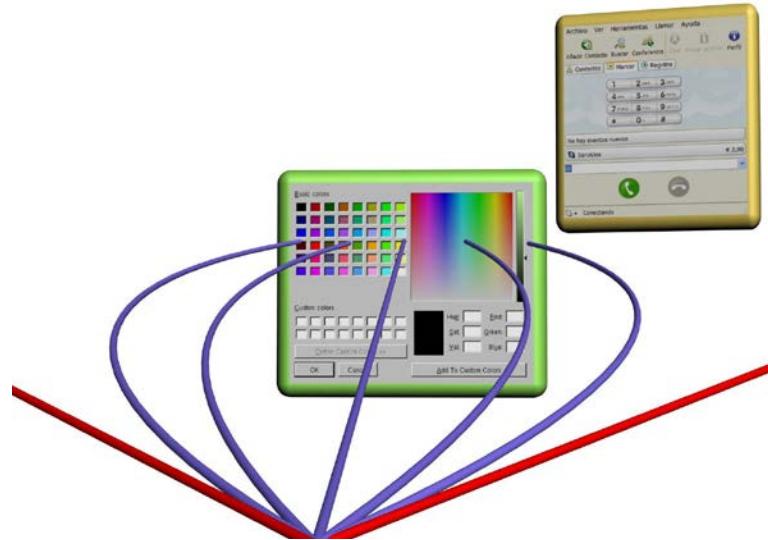
- Generate targets next to the user: pop-up menus



ACCELERATING TARGET ACQUISITION: TARGET MOVING

➤ Sticky targets:

- Attract pointer
 - When the pointer is close to a selectable area
 - May reduce selection time
 - Precision not required
 - Users adapt easily



ACCELERATING TARGET ACQUISITION: CONTROL-DISPLAY RATIO

- Relation between the amplitude of movements of the user's real hand and the amplitude of movements of the virtual cursor
- Moves in real world (physical move) mapped to moves in virtual desktop (cursor move)
- Different strategies:
 - Constant
 - Dependent on mouse speed
 - Dependent on cursor position
- Interpretation according to Fitts Law:

Dynamic C-D ratio adaptation can be interpreted as dynamic change of physical motor space

ACCELERATING TARGET ACQUISITION: CONTROL-DISPLAY RATIO

- Mac OSX and Windows both use mouse acceleration
 - When mouse moves fast, it is accelerated
 - Reducing the amplitude of movement to cover large distances
 - When mouse moves slow, it is decelerated
 - Magnifying amplitude of movement to improve precision
- No clear how the mapping affects perception and productivity
 - Some studies say it is not intuitive
 - Some studies say it improves some pointing tasks

OUTLINE

Session 1:

- *Understanding the fundamentals of basic interaction in UI*
 - *Background (Information Theory)*
 - *Hick-Hyman Law: Measuring Choice-Reaction Time*
 - *Fitts' Law: Measuring Pointing Time*
 - *Crossing and Steering Laws: Continuous Gestures*
- *Fitts' Law in UI Design*
 - *Applications in UI Design*
 - *Accelerating Target Acquisition*
- **Exercises**

Donades les constants $a = 400$ ms, $b = 200$ ms/bit i un objectiu de mida 2.1 cm a una distància de 10.5 cm. Marca la resposta correcta assumint que fem els càlculs amb la versió de McKenzie de la llei de Fitts.

- a. ID ≈ 3.4 .
- b. $2 < \text{ID} < 3$.
- c. ID ≈ 4.3 .
- d. MT està entre 1100 i 1200 ms.

La llei de Hick-Hyman:

- a. Modela el temps de decisió com una funció de la informació transmesa.
- b. Modela el temps de selecció d'un element com a funció de la distància a recórrer i la mida de l'element.
- c. Modela el temps de decisió com una funció de la distància a recórrer i l'entropia dels elements a seleccionar.
- d. Utilitza l'entropia de Shannon per a mesurar la distància del recorregut mínim.

$$MT = a + b \log_2 \left(\frac{D}{W} + 1 \right)$$

- Els *expanding targets*:

- a. Es basen en la llei de Hick-Hyman.
- b. Pretenen reduir el temps d'accés als elements basant-se en el fet que, segons la llei de Fitts, el temps d'accés es redueix si s'augmenta la longitud del desplaçament.
- c. Si es combinen amb el moviment dels objectius poden causar confusió a l'usuari.
- d. Cap de les anteriors.

Ens han encarregat fer un disseny d'una interfície per a un sistema tipus desktop en la qual hi haurà botons i menús drop-down.

- a. Podem predir la dificultat d'accedir als botons utilitzant la llei de Fitts i la dificultat de recórrer els menús amb la llei de crossing.
- b. Podem analitzar el nombre d'elements a posar en un menú utilitzant la llei de steering i en funció dels digrams.
- c. Podem analitzar el nombre d'elements a posar en un menú utilitzant la llei de Fitts.
- d. Podem analitzar la dificultat de recórrer els menús utilitzant la llei de steering.

▪ **La llei de steering:**

- a. No es pot derivar a partir de la llei de *crossing*.
- b. Serveix per a modelar el temps necessari per a recórrer un camí de forma arbitrària.
- c. Diu que hi ha una relació logarítmica entre l'índex de dificultat de creuar un objectiu i el temps que requerit per a fer-ho.
- d. Diu que l'índex de dificultat de creuar un objectiu és D/W.

- Dos elements T1 i T2 a distàncies $D_1 = 10$ cm i $D_2 = 8$ cm en direcció horitzontal i d'amplades 5 cm i 2 cm, respectivament. Per a T1 emprem un dispositiu amb $a_1 = 200$ ms i $b_1 = 200$ ms/bit. Per a T2 utilitzem un dispositiu amb $a_2 = 200$ ms i $b_2 = 100$ ms/bit. Assumint la formulació original de la llei de Fitts:
 - ID₁ > ID₂.
 - ID₁ = ID₂.
 - MT₁ = MT₂.
 - MT₂ < MT₁.

OUTLINE

Session 1:

- *Understanding the fundamentals of basic interaction in UI*
 - *Background (Information Theory)*
 - *Hick-Hyman Law: Measuring Choice-Reaction Time*
 - *Fitts' Law: Measuring Pointing Time*
 - *Crossing and Steering Laws: Continuous Gestures*
- *Fitts' Law in UI Design*
 - *Applications in UI Design*
 - *Accelerating Target Acquisition*
- **Exercises**

INTERACTION DESIGN.

SESSION 1

Dept. Computer Science – UPC

MOTIVATION

- **Usability & Design Principles**
- **Perception Laws**

DIRECT MANIPULATION INTERFACES (Pointing, choice selection)

- **Interaction Design and Evaluation:**
 - Design User Interfaces
 - Measure/Predict performance
 - Design interaction

OUTLINE

Session 1:

- Understanding the fundamentals of basic interaction in UI
 - Background (Information Theory)
 - Hick-Hyman Law: *Measuring Choice-Reaction Time*
 - Fitts' Law: *Measuring Pointing Time*
 - Crossing and Steering Laws: *Continuous Gestures*
- Fitts' Law in UI Design
 - Applications in UI Design
 - Accelerating Target Acquisition
- Exercises

Session 2:

- Typing & Keyboards
- Pointing Devices
- Mobile Interaction Design
- Exercises

OUTLINE

Session 1:

- Understanding the fundamentals of basic interaction in UI
 - **Background (Information Theory)**
 - Hick-Hyman Law: *Measuring Choice-Reaction Time*
 - Fitts' Law: *Measuring Pointing Time*
 - Crossing and Steering Laws: *Continuous Gestures*
- Fitts' Law in UI Design
 - Applications in UI Design
 - Accelerating Target Acquisition
- Exercises

BACKGROUND. BASICS

- **Information Theory:**

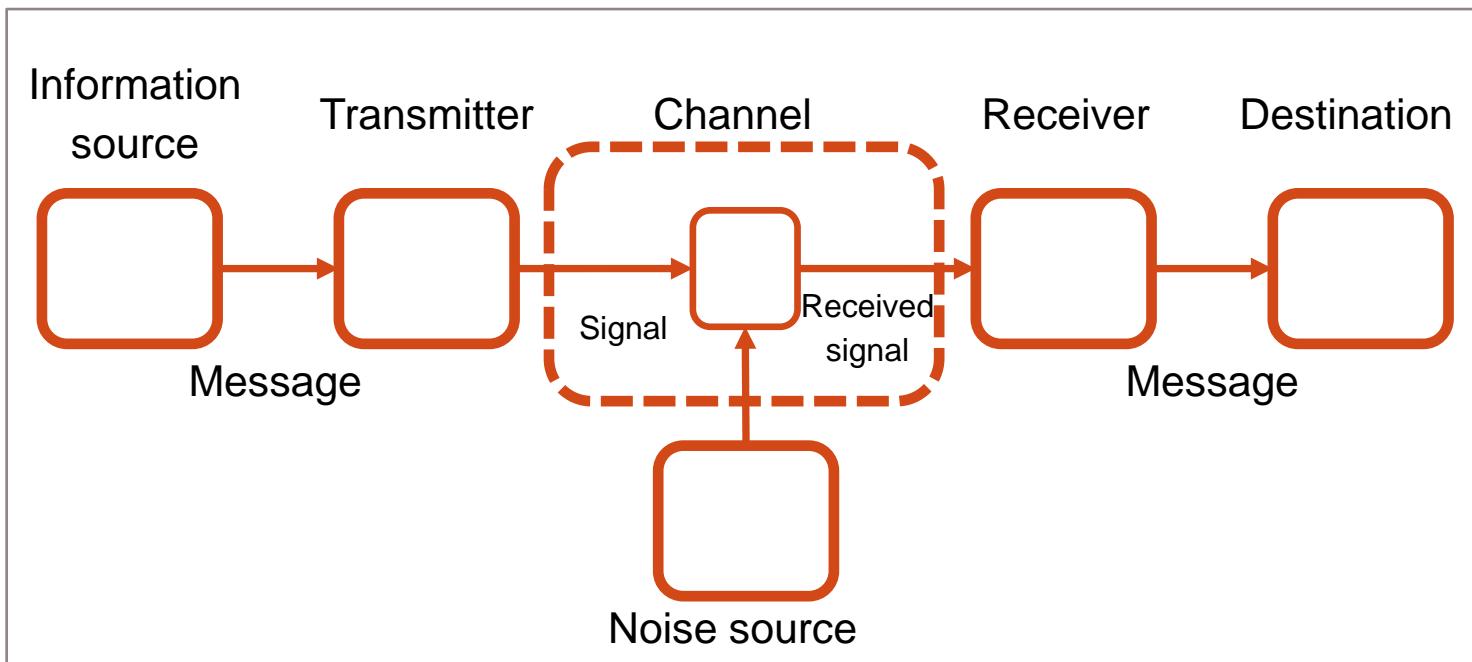
- Due to Claude E. Shannon

- *A Mathematical Theory of Communication* (1948)
 - Based on previous works by Nyquist and Hartley
 - Analysis of transmission of electrical signals for telegraphic communication
 - Shannon Entropy measures:

The amount of information to be transmitted by a message

BACKGROUND. BASICS

- Information Theory. Elements (telegraph):



BACKGROUND. BASICS

- Information Theory. Elements (telegraph):
 - **Information source:** The element that produces a message or sequences of message.
 - **Transmitter:** Operates on the message to make it transmissible through a medium.
 - **Channel:** The medium that transmits the message.
 - **Receiver:** The element that reconstruct the message to the destination.

BACKGROUND. INFORMATION MEASURES

- Let d be a device that produces symbols A, B, C and D with the same probability
 - $M = 4$ is the total number of symbols
 - Each time a symbol is produced we are uncertain on which symbol is going to be generated
 - This uncertainty is not so big, since there are only four possibilities
 - The probability of a symbol to appear is $1/M : 1/4$
- The **uncertainty** is measured by $\log_2(M) \rightarrow$ here $\log_2(4)=2\text{bits}$
- Logarithms are commonly taken in base 2, and the units are bits.

BACKGROUND. INFORMATION MEASURES

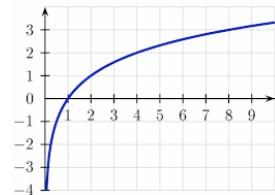
- **Example 1:** Let d be a device that produces one single symbol: C
 - $M = 1$ is the total number of symbols
 - We have **no uncertainty** and $\log_2(1) = 0$
 - The probability of getting the symbol C is 1
 - We previously know which symbol will appear!
- **Example 2:** Let d and e be two devices, one with outputs A, B, C, and the second with outputs 1, 2.
 - We combine *words* by concatenating one symbol of device d and one with device e .
 - We will have 6 different words: A1, A2, B1, B2, C1, C2
 - 6 symbols → uncertainty of $\log_2(6)$ → $\log_2(2) + \log_2(3) = \log_2(6)$.
- The uncertainty of combined the signals of a set of devices is the sum of their uncertainties.

BACKGROUND. INFORMATION MEASURES

- For M symbols with equal probability → each symbol has probability $P=1/M$

- Rewriting the uncertainty

$$\log_2(M) = \log_2\left(\left(\frac{1}{M}\right)^{-1}\right) = \log_2(P^{-1}) = -\log_2(P)$$



- $-\log_2(P)$ is called the **surprise** or *surprisal* of finding a certain symbol
 - We will use p_i from now on for the probability of a symbol i
- For M symbols that have different probabilities, we may have a different p_i for each, provided that

$$\sum_{i=1}^M p_i = 1$$

BACKGROUND. INFORMATION MEASURES

- **Information is the reduction of uncertainty or average surprise of a set of symbols**

- Measuring the surprise for an *infinite* set of N symbols (produced by a device) → the frequency of each symbol transforms to the probability.
- Shannon Entropy measures the amount of information:

$$H = \sum_{i=1}^N p_i \log_2 \left(\frac{1}{p_i} \right) = - \sum_{i=1}^N p_i \log_2 p_i$$

- N is the number of alternatives
- p_i is the probability of the i th alternative.
- **H is the entropy of the message that is to be transmitted,**
→ the amount of information expected to be received (no noise).

BACKGROUND. INFORMATION MEASURES

- **Example 1: Source with two equiprobable symbols: A and B**
- $p(A)=0.5, p(B)=0.5$
- $H= -0.5 \log_2(0.5) - 0.5 \log_2(0.5) = -\log_2(0.5) = -\log_2(2^{-1}) = 1$
- The source requires an average of 1 bit per symbol.

- **Example 2: Source with two symbols: A and B**
- $p(A)=0.1, p(B)=0.9$
- $H= -0.1 \log_2(0.1) - 0.9 \log_2(0.9) = 0,332 + 0,137 = 0,47$
- The source requires an average of 0,47 bit per symbol.

BACKGROUND. INFORMATION MEASURES

$$p(A)=0.1, p(B)=0.9$$

H=0,47 bits → Is it possible? We can achieve it using a smart codification of the information. For instance:

Symbols	Codification	Probability	Bits	Weighted bits
AA = 00	000	0,1*0,1=0,01	3	0,03
AB = 01	001	0,1*0,9=0,09	3	0,27
BA = 10	01	0,1*0,9=0,09	2	0,18
BB = 11	1	0,9*0,9=0,81	1	0,81
		1		1,29bits in average to send 2 symbols
				0,645 bits per symbol

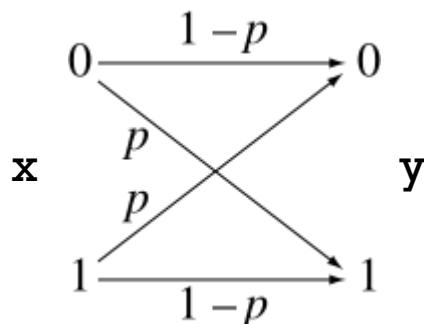
BACKGROUND. INFORMATION MEASURES

- Information Theory. Shannon entropy:

- There is interference: Not all information will reach the receiver
- Average information faithfully transmitted (R):

$$R = H(x) - H_y(x)$$

- $H_y(x)$ is the equivocation or conditional entropy of x when y is known. Measures the information required to quantify the error.



$$H_y(x) = \sum_{i=0}^N \sum_{j=0}^N p(x_i, y_j) \cdot \log_2(p(x_i | y_j))$$

p : error probability

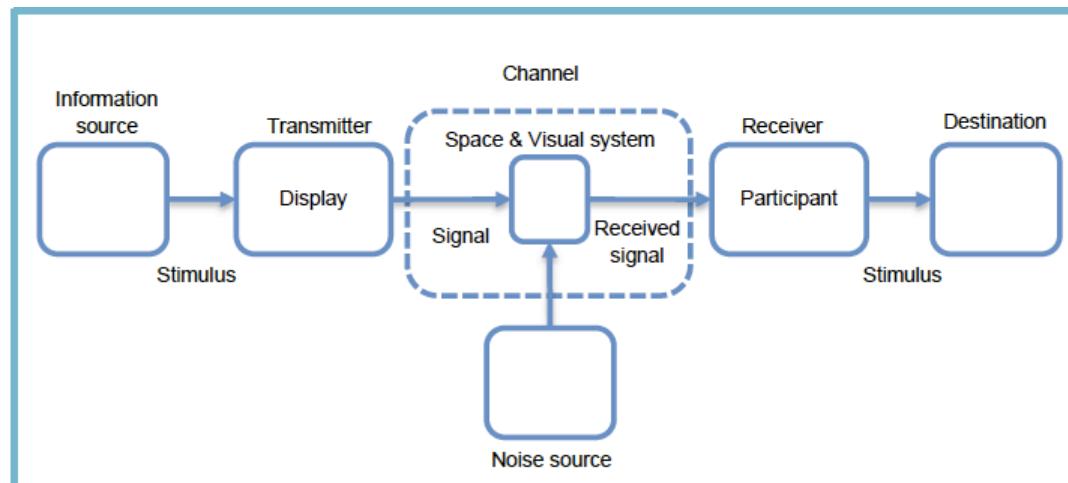
OUTLINE

Session 1:

- Understanding the fundamentals of basic interaction in UI
 - *Background (Information Theory)*
 - **Hick-Hyman Law: Measuring Choice-Reaction Time**
 - Fitts' Law: *Measuring Pointing Time*
 - Crossing and Steering Laws: *Continuous Gestures*
- Fitts' Law in UI Design
 - Applications in UI Design
 - Accelerating Target Acquisition
- Exercises

HICK-HYMAN LAW

- Hick-Hyman Law:
 - Initially stated by William E. Hick (1951)
 - Describes human decision time as a function of the information content conveyed by a visual stimulus
 - It takes longer to respond to a stimulus when it belongs to a large set as opposed to a smaller set of stimuli
 - Extended by Ray Hyman (1952)



HICK-HYMAN LAW

- Time to make a decision (Reaction Time):

$$RT = a + bH_T$$

- a, b constants
- H_T transmitted information

HICK-HYMAN LAW

- Hick-Hyman Law:

- H_T : Transmitted information:

$$H_T = \log_2(n + 1)$$

- n are the equiprobable alternatives
 - original formulation did not have the “+1” attends for the uncertainty whether to respond or not

- Time to answer is the Reaction Time:

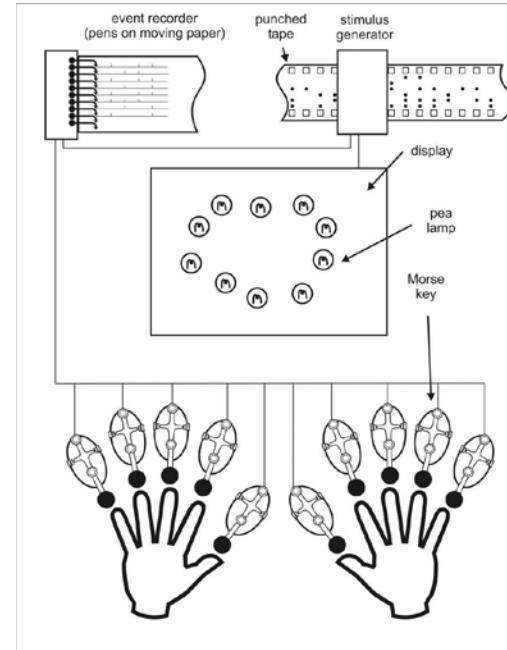
$$RT = a + b \log_2(n + 1)$$

HICK-HYMAN LAW.

EXPERIMENTAL ASSESSMENT

- Hick's initial experiment:

- 10 pea lamps are arranged in an irregular circle
- One random lamp is lit every 5 seconds
- User has to press the correct key corresponding to the lamp that is lit
- Stimulus and response encoded in a moving paper in binary code



HICK-HYMAN LAW. EXPERIMENTAL ASSESSMENT

- Time to answer. Reaction Time is a linear function of stimulus information

$$RT = a + b \log_2(n + 1)$$

- Hyman [Hyman53] found that it
 - also holds for not equiprobable alternatives
- Experiment:
 - 8 lights (whose names were *Bun*, *Boo*, *Bee*, *Bore*, *By*, *Bix*, *Bev*, and *Bate*)
 - The users had to name the one lit
 - A microphone attached to the throat detected the voice and stopped the timer
 - First with equal probabilities
 - Then, with varying probabilities

HICK-HYMAN LAW. EVIDENCES

- **Evidences of Hick-Hyman Law**
 - Performance *in hierarchical full-screen menu selections* is well described by Hick-Hyman [Landauer85]
 - Selection times decay logarithmically with menu length for frequently selected items, but linearly with infrequent ones [Sears94].
 - Learnt locations (most frequent) fit Hick-Hyman decision times
 - Non-learnt locations fit a linear search
 - Novice users search linearly while experts decide upon item location and fit a Hick-Hyman curve [Cockburn2008]

OUTLINE

Session 1:

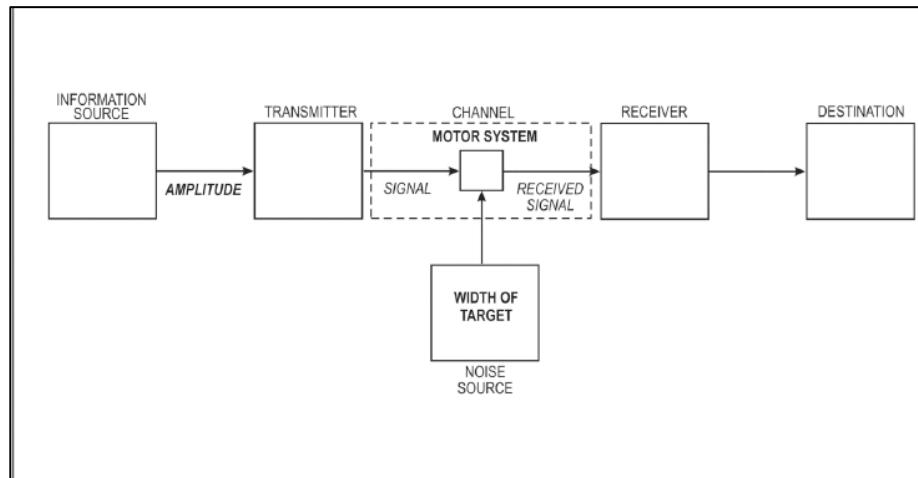
- Understanding the fundamentals of basic interaction in UI
 - *Background (Information Theory)*
 - *Hick-Hyman Law: Measuring Choice-Reaction Time*
 - **Fitts' Law: Measuring Pointing Time**
 - Crossing and Steering Laws: *Continuous Gestures*
- Fitts' Law in UI Design
 - Applications in UI Design
 - Accelerating Target Acquisition
- Exercises

FITTS' LAW. ORIGINAL FORMULATION

- States a **linear relationship between the movement time (MT) and task difficulty**

$$MT = a + bID$$

- Formulation is also based on Information Theory
 - Amplitude of movement is the *signal*
 - Human motor system is the communication *channel*
 - Target width is the *noise*



FITTS' LAW. ORIGINAL FORMULATION

- **Task difficulty:**

$$ID = \log_2 \left(\frac{2A}{W} \right)$$

- ID : Index of difficulty
- A : Amplitude of movement
- W : Target width

- The larger the amplitude the higher the difficulty
- The larger the target the lower the difficulty

FITTS' LAW. ORIGINAL FORMULATION

- **Movement Time:** Time to point a certain objective (target)

$$MT = a + bID$$

- a start/stop times in seconds
- b inherent speed of the device

$$ID = \log_2 \left(\frac{2A}{W} \right)$$

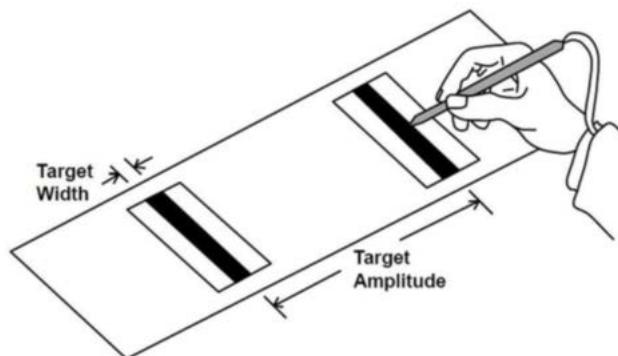
- A: Amplitude of movement
- W: Target width

FITTS' LAW. EXPERIMENTAL EVIDENCES

Fitts' Law. Original experiments:

- Experiment 1: Reciprocal tapping:

- Participants used a metal-tipped stylus:
 - Two experiments with two different stylus: ~ 28.35 and 453.6 gr
 - Tap two strips of metallic targets of width from ~ 0.635 to 5.08 cm
 - At distance 5.08 to 40.64 cm
 - Participants instructed to be accurate!



FITTS' LAW. EXPERIMENTAL EVIDENCES

Fitts' Law. Original experiments:

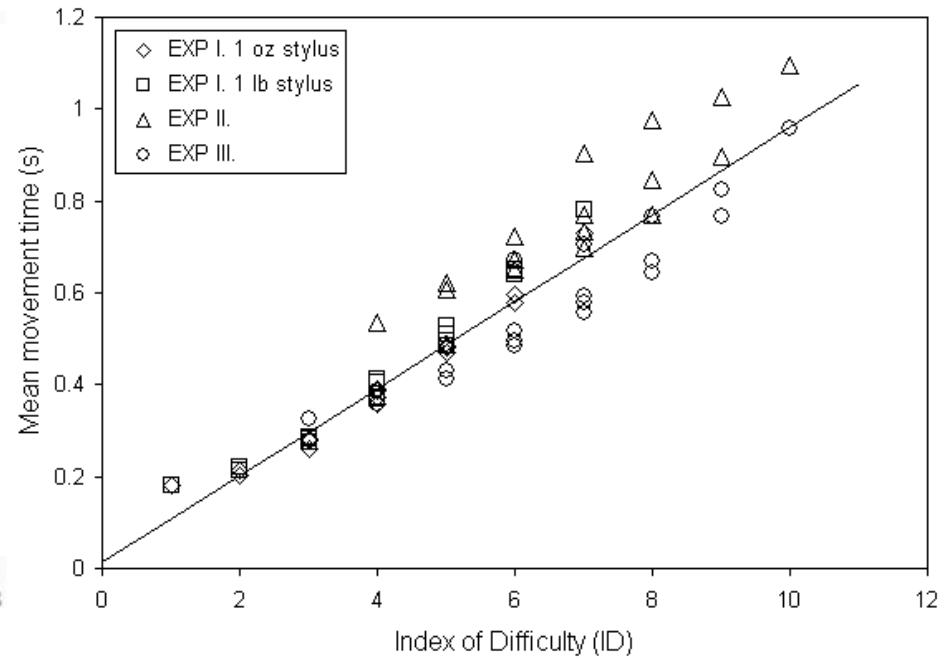
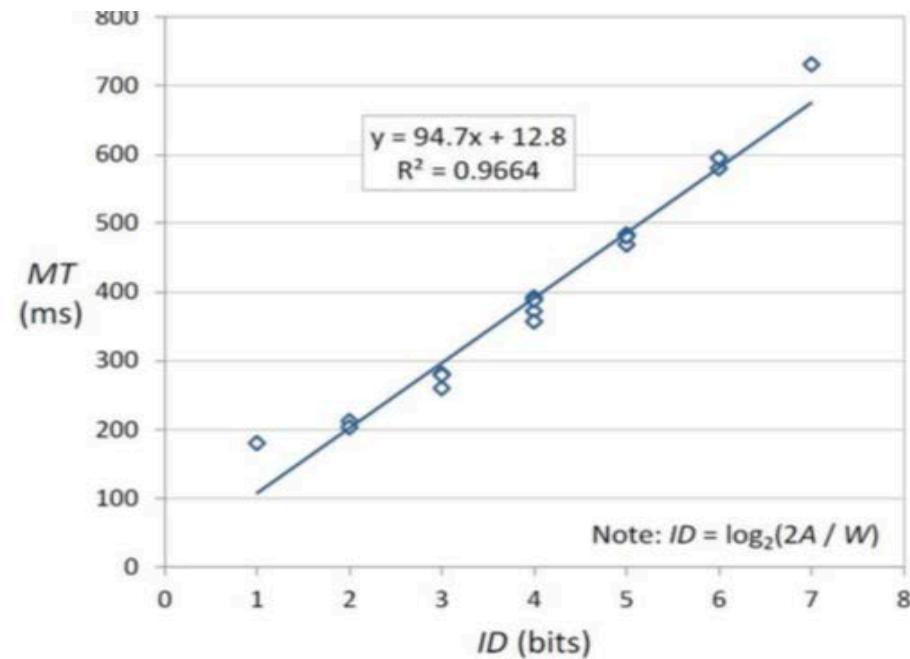
- Experiment 2: Disk transfer
 - Participants had to transfer stack round plastic disks (with holes drilled through the middle) from one pin to another
 - Holes of different sizes and pins of different diameters used
- Experiment 3: Pin transfer
 - Participants had to transfer pins of different diameters from a set of holes to another set of holes

FITTS' LAW. EXPERIMENTAL EVIDENCES

$$ID = \log_2 \left(\frac{2A}{W} \right)$$

■ Fitts' Law Results.

$$MT = 12.8 + 94.7 ID$$



- Results show that there is a linear relationship between MT and ID
- **Most difficult condition: Smaller W and largest A**
- Only valid for the experiments carried out
 - One curve per experiment fits better (different a and b values)

FITTS' LAW. VARIANTS

- Original formulation fits well to the original experiments
 - But it might fit better
- Other researchers have found different formulations that better model the experimental data
 - Including the experimental data by Fitts
- Welford [Welford68]:

$$MT = a + b \log_2 \left(\frac{D + 0.5W}{W} \right)$$

- D is the distance of movement
- W is the width of the target

FITTS' LAW. VARIANTS

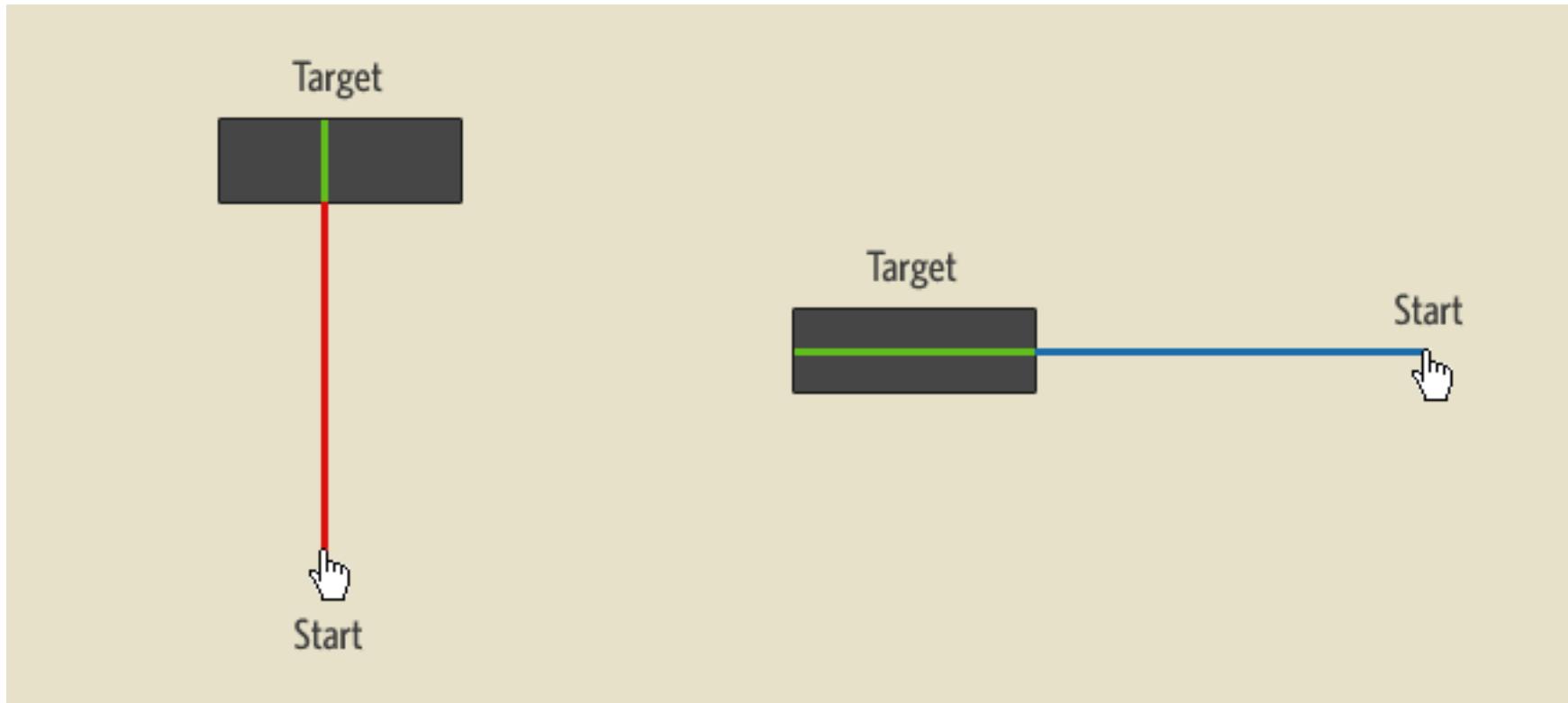
- **MacKenzie's** approach [MacKenzie92] is one of the most accepted:

$$MT = a + b \log_2 \left(\frac{D}{W} + 1 \right)$$

- D is the distance of movement
- W is the width of the target

FITTS' LAW. VARIANTS

- Vertical and horizontal movements can be treated equally

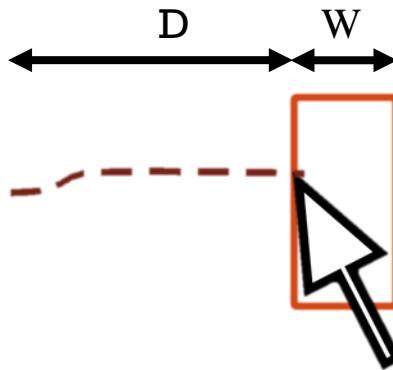


FITTS' LAW EXTENSIONS

- Main application of Fitts in HCI is evaluation/design of UI and interaction
- Today's interfaces are much more complex
 - Variety of sizes
 - 2D movements
 - Use of fingers

FITTS' LAW. EXTENSIONS

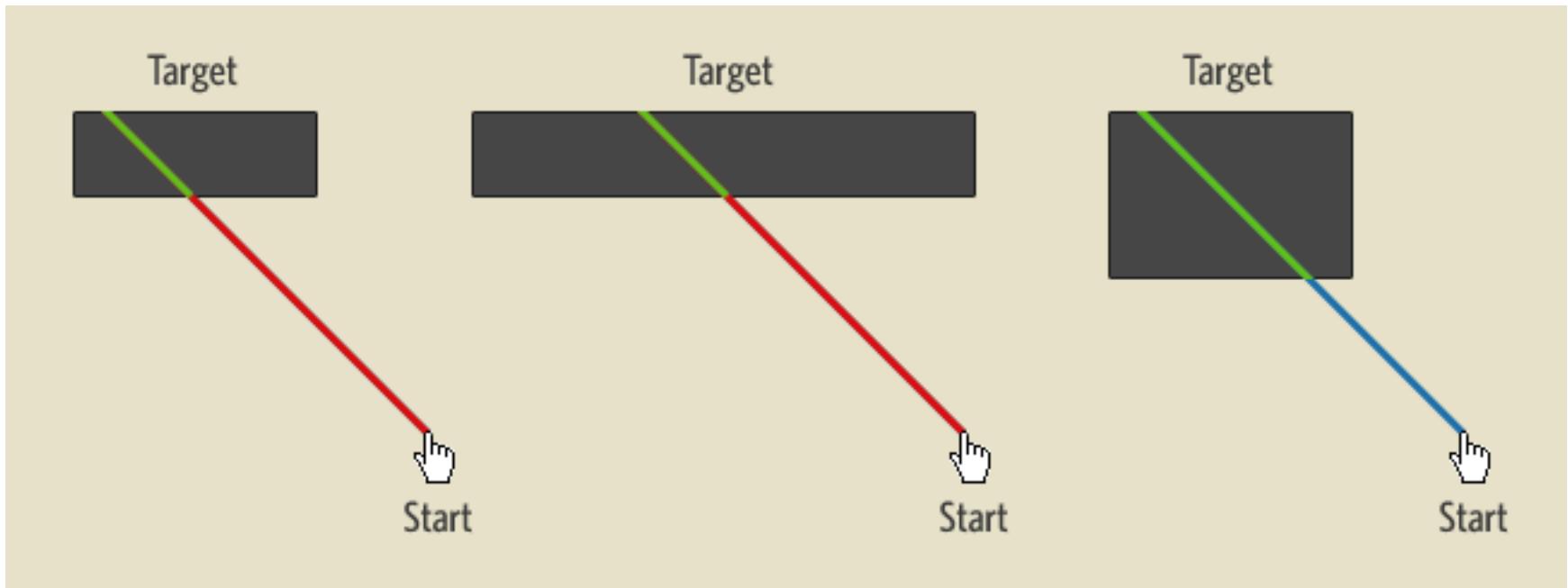
- Use in UI design or evaluation:



- D is the distance the pointer (mouse) covers to reach the target (button)
- W is the width of the target (button)

FITTS' LAW. EXTENSIONS 2D

- Fitts' Law is designed for 1D movements
BUT...most movements in a UI are 2D
- Vertical and horizontal movements can be treated equally... or not?



FITTS' LAW. EXTENSIONS 2D

- Several extensions deal with 2D movements
 - Mimicking Fitts' Law, but changing some of the parameters

- [Crossman83]:

$$MT = a + b \log_2 \left(\frac{2D}{W} \right) + c \log_2 \left(\frac{2D}{H} \right)$$

- [Accot97]:

$$MT = a + b \log_2 \left(\sqrt{\left(\frac{D}{W} \right)^2 + \eta \left(\frac{D}{H} \right)^2} + 1 \right)$$

FITTS' LAW EXTENSIONS: PRECISION POINTING

- Fitts Law does not model properly very small targets:
 - Extra time devoted to fine adjustment
 - Increase of errors
 - ...
- Very small targets yield a lower fit of the regression curve of the MT function
- Touchscreens also modifies the timing we require to point targets.

FITTS' LAW EXTENSIONS: PRECISION POINTING

Extension of Fitts' Law by analyzing the behavior both in tactile screens and small targets ([Sears91]):

- Named FFitts (**Finger Fitts**), also PPMT (Precision Pointing Movement Time) by some other authors :

$$FFits = a + bID + dID_2$$

$$FFitts = a + b \left[\log_2 \left(\frac{cD}{W} \right) \right] + d \left[\log_2 \left(\frac{e}{W} \right) \right]$$

- The higher number of freedom degrees, the easier to fit in a regression curve

FITTS' LAW. EXTENSIONS: PRECISION POINTING

- **FFitts:**

$$FFitts = a + b \left[\log_2 \left(\frac{cD}{W} \right) \right] + d \left[\log_2 \left(\frac{e}{W} \right) \right]$$

- the first logarithmic factor measures *the time to place the finger on the screen initially*
- the second factor measures *the time to position the cursor*
- D is the distance, measured in three dimensions, from the original hand location to the location of first contact
- If the task consists of iteratively clicking targets: D is the distance from one target to the next one
- W is some measurement of target size
- a, b, c, d , and e must be determined for each specific case

FITTS' LAW. ASSESSED RESULTS

- Validation of Fitts' Law may not extrapolate to outside the experiments carried out
 - ***Validity Fitts → Experimentation***
- Fitts' Law have been formulated in a number of ways, however its prediction is consistent:
 - "the ID to acquire a target is functions of the distance to and the size of the target"

FITTS' LAW. ASSESSED RESULTS

- Fitts' Law has shown its validity in multiple setups and devices:
 - Mouse, joystick, finger, stylus...
 - Different screen types of varying sizes...
 - **But the results cannot be extrapolate to data outside the experiment. Validity Fitts → Experimentation**
- Fitts' law is a really good predictive model of human movement.
- Precued targets lead to more efficient and precise pointing movements than for non-precued targets [Hertzum2013].
 - Most common case: we know the buttons' positions in advance.
 - The benefit of precuing is larger for the mouse than the touchpad
 - Maybe movement preparation is more effective if the device is more demanding

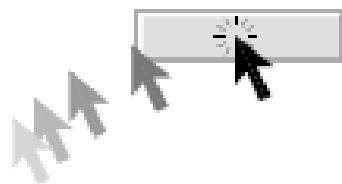
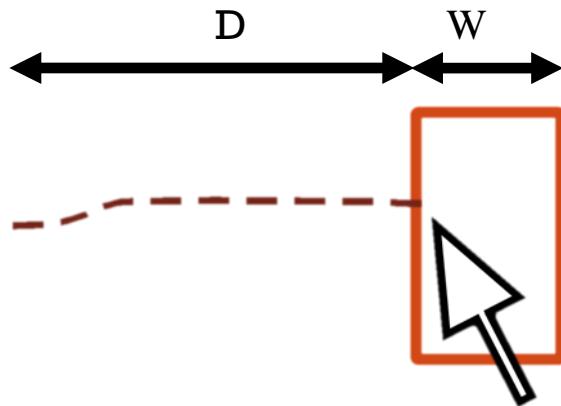
OUTLINE

Session 1:

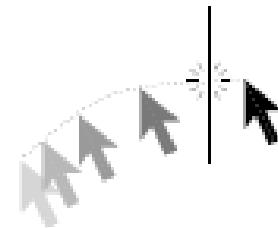
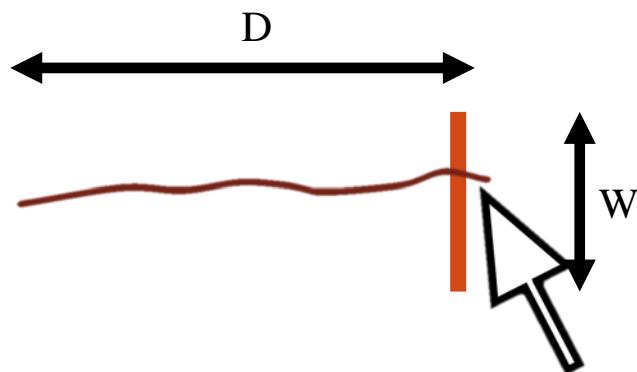
- Understanding the fundamentals of basic interaction in UI
 - *Background (Information Theory)*
 - *Hick-Hyman Law: Measuring Choice-Reaction Time*
 - *Fitts' Law: Measuring Pointing Time*
 - **Crossing and Steering Laws: Continuous Gestures**
- Fitts' Law in UI Design
 - Applications in UI Design
 - Accelerating Target Acquisition
- Exercises

LAW OF CROSSING

- Crossing movement as compared to pointing



(a) Pointing a target



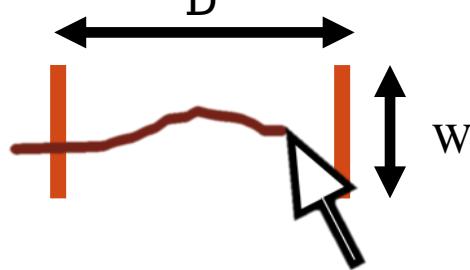
(b) Crossing a goal

LAW OF CROSSING

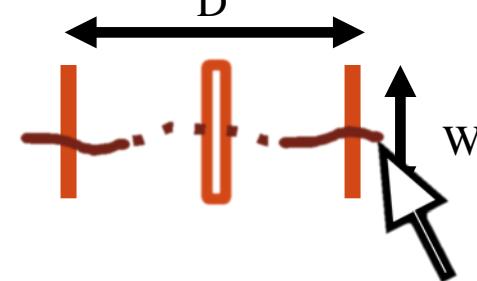
▪ Crossing configurations:

- Discreteness vs continuity of the movement:
 - Landing and lifting off the stylus

Continuous crossing



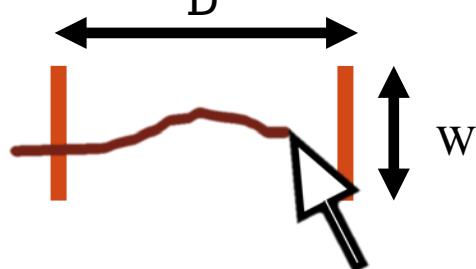
Discrete crossing



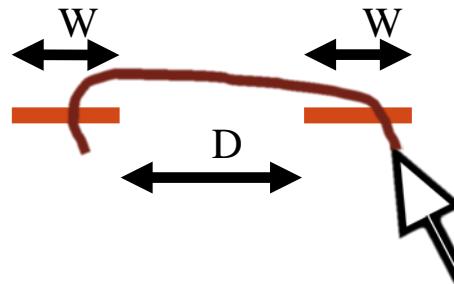
- Direction of the targets vs direction of the movement:

- If parallel, the trace will be larger

Orthogonal crossing

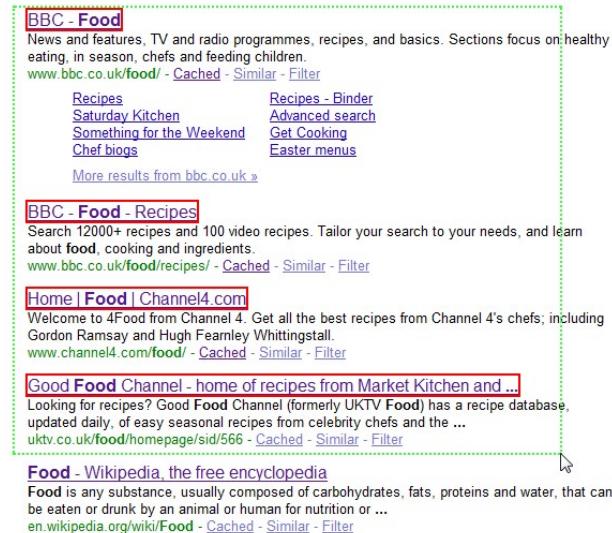


Collinear crossing



LAW OF CROSSING

- Stylus or fingers naturally lead to crossing gestures
 - Especially useful in tactile devices
 - Crossing an object is easier than double-clicking.
 - Drag & drop, multiple selections
 - Crossing can be a good alternative for users who have difficulties with clicking or double-clicking.
- Several objects can be crossed at the same time within the same gesture



Multi-links
extension for
Chrome
(LinkClump)

LAW OF CROSSING

- Crossing performance across two goals [Accot99, Zhai2002]:
 - Follows the same characterization than the Fitts' Law:

$$T = a + b \log_2 \left(\frac{D}{W} + 1 \right)$$

- T is the average moving time between passing the two goals.
- D is the distance between the two goals
- W is the width of each goal
- a and b are constants to be determined

LAW OF CROSSING

- **Results of the experiments:**

- Crossing-based interfaces achieve similar (or faster) times than pointing.
- The error rate in crossing is smaller than in pointing.
- Discrete crossing becomes more difficult if the distance between the targets is small.

LAW OF CROSSING



<https://www.youtube.com/watch?v=C5L4vV3T2mU>

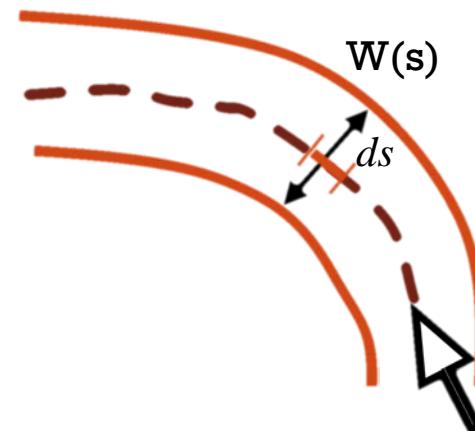
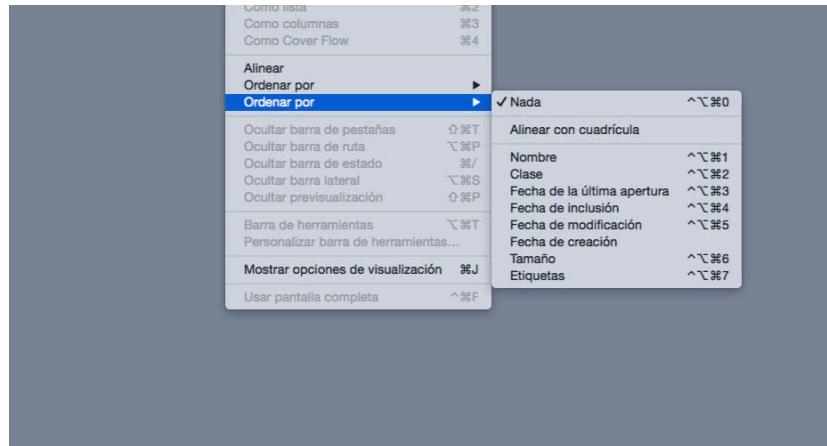
OUTLINE

Session 1:

- Understanding the fundamentals of basic interaction in UI
 - *Background (Information Theory)*
 - *Hick-Hyman Law: Measuring Choice-Reaction Time*
 - *Fitts' Law: Measuring Pointing Time*
 - **Crossing and Steering Laws: Continuous Gestures**
- Fitts' Law in UI Design
 - Applications in UI Design
 - Accelerating Target Acquisition
- Exercises

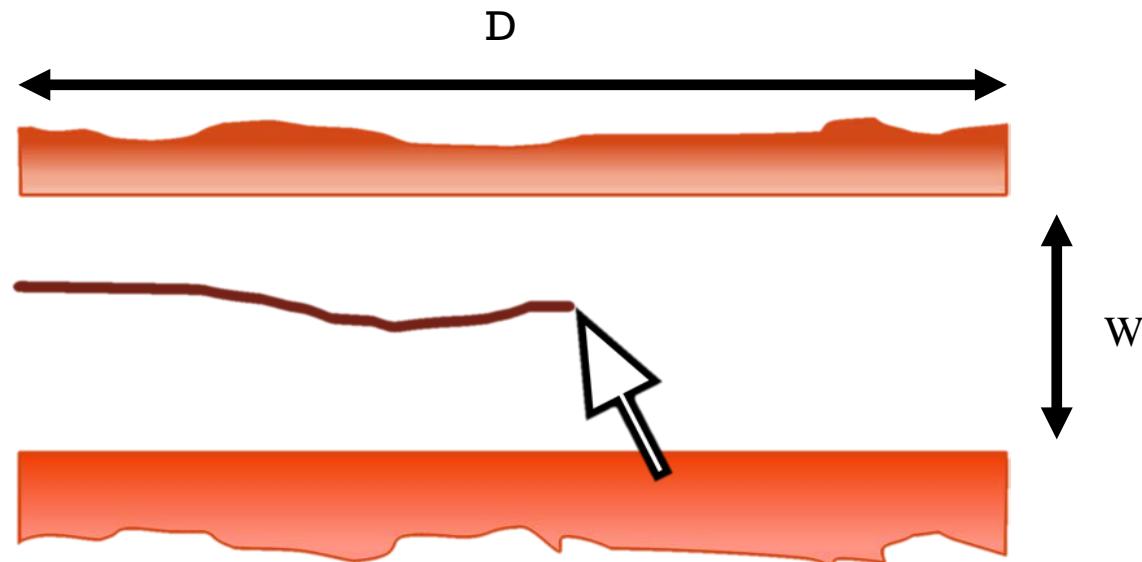
STEERING LAW

- Navigating through a constrained path is an useful operation in modern UIs
 - Navigating through nested menus
 - 3D navigation
 - Dragging elements
 - Free-hand Sketching/Drawing



STEERING LAW

- Steering through a **straight path**:

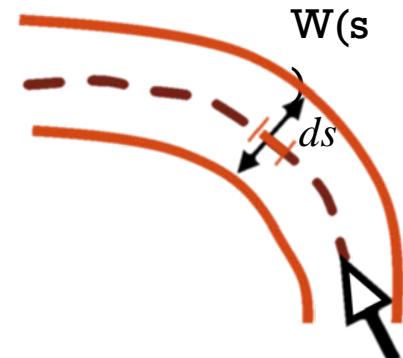


STEERING LAW

- Navigating through a **generalized path** can be expressed as infinite crossings [Accot97]
- Movement time across the path T_s :

$$T_s = a + bID_s \quad T_s = a + b \int_C \frac{ds}{W(s)}$$

- C is the length of the path
- $W(s)$ is the path width at point s



STEERING LAW

- Time to navigate through a **straight path** (tunnel) T_p [Accot97]:

$$T_s = a + b \int_c \frac{ds}{W(s)} \quad T_p = a + b \frac{D}{W}$$

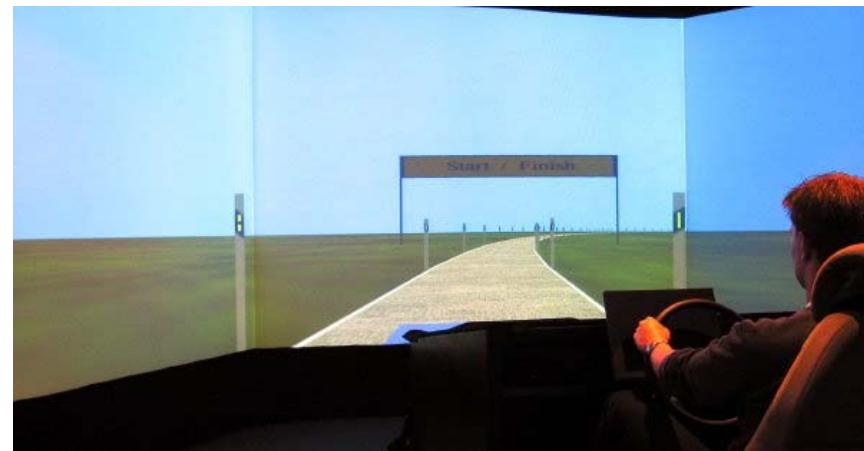
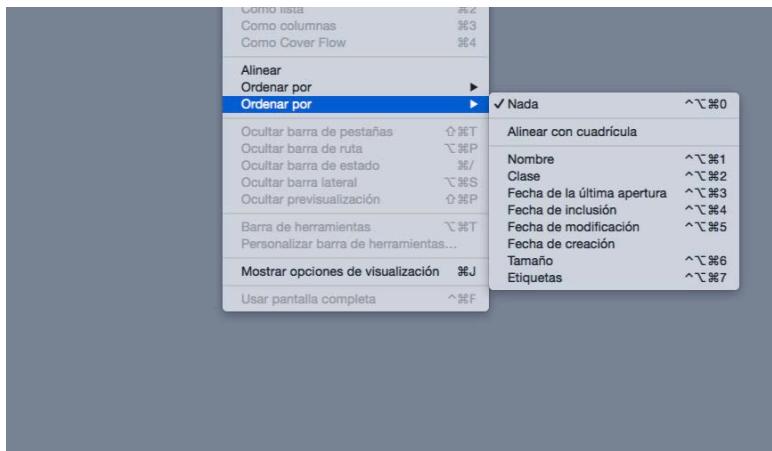
- D is the length of the path/tunnel
- W is the width of the path/tunnel
- Applying Fitts' formatting:

$$T_p = a + bID_p \quad ID_p = \frac{D}{W}$$

- Which also applies to circular paths of constant width

STEERING LAW

- Results [Accot97, Zhai2004] show that the steering law is applicable to different configurations:
 - Different path shapes: cone, spiral, straight
 - Works with different devices, works in VR...
 - Can be used to analyse navigation through nested menus, compare menu designs...



OUTLINE

Session 1:

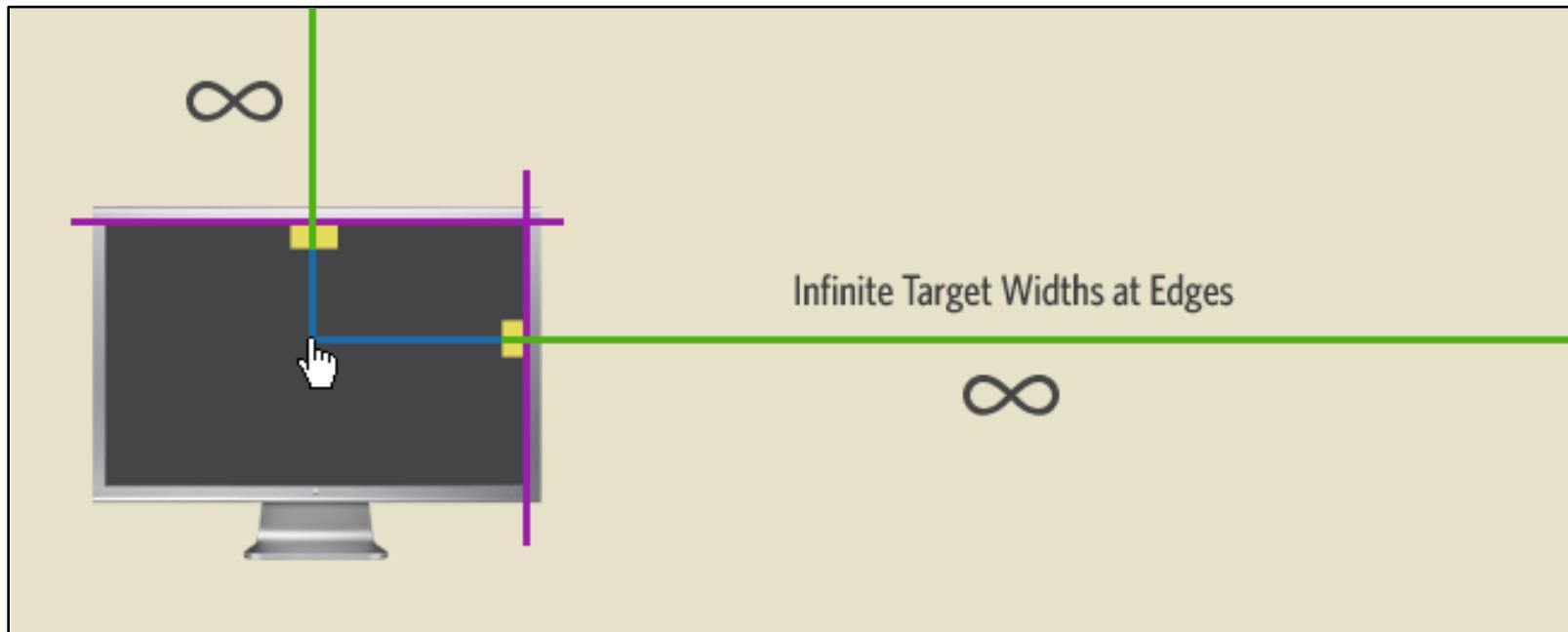
- *Understanding the fundamentals of basic interaction in UI*
 - *Background (Information Theory)*
 - *Hick-Hyman Law: Measuring Choice-Reaction Time*
 - *Fitts' Law: Measuring Pointing Time*
 - *Crossing and Steering Laws: Continuous Gestures*
- **Fitts' Law in UI Design**
 - Applications in UI Design
 - Accelerating Target Acquisition
- Exercises

FITTS' LAW IN UI DESIGN. IMPLICATIONS

- Fitts' Law accurately predicts **pointing** movement
 - Further distance → Harder to select
 - Larger target → Easier to select
- If improvement required, it can help us modify our UI
 - **Change target width:**
 - Increase size for faster reach
 - **Change de “virtual distance” or pointer movement:**
 - Increase speed, pop-up menus,....
- But visual stimuli must also be taking into account...

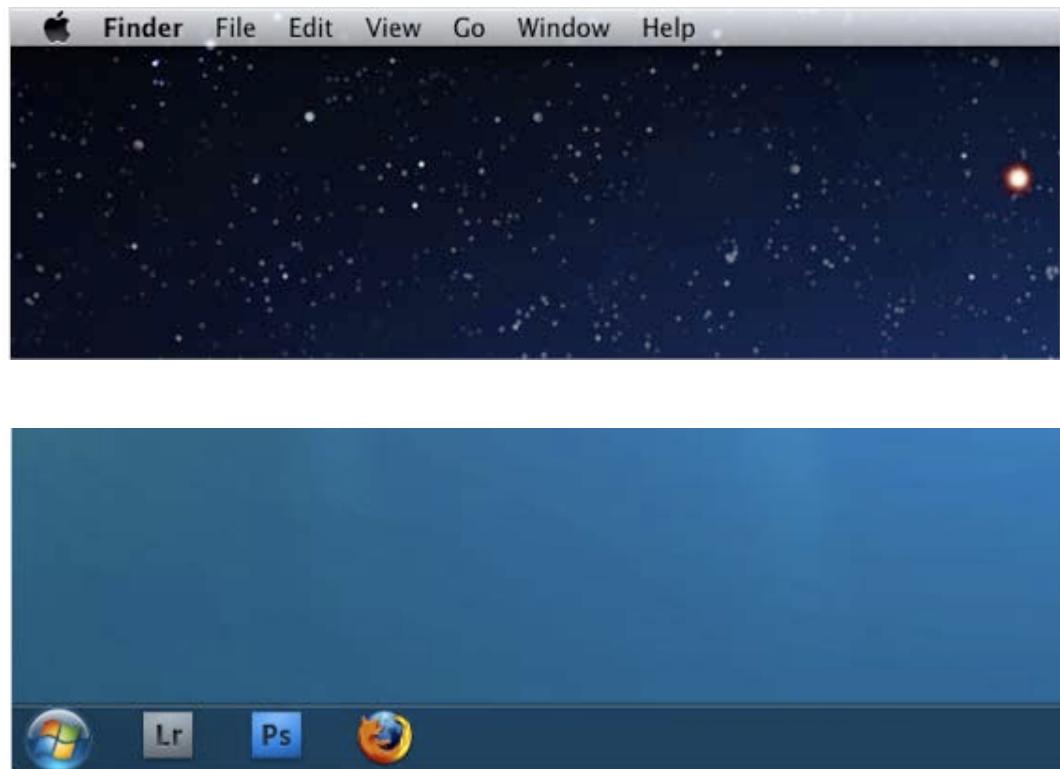
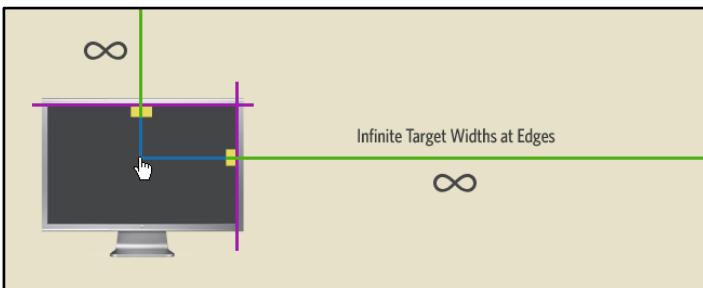
FITTS' LAW IN UI DESIGN. APPLICATIONS

The outer edges and corners

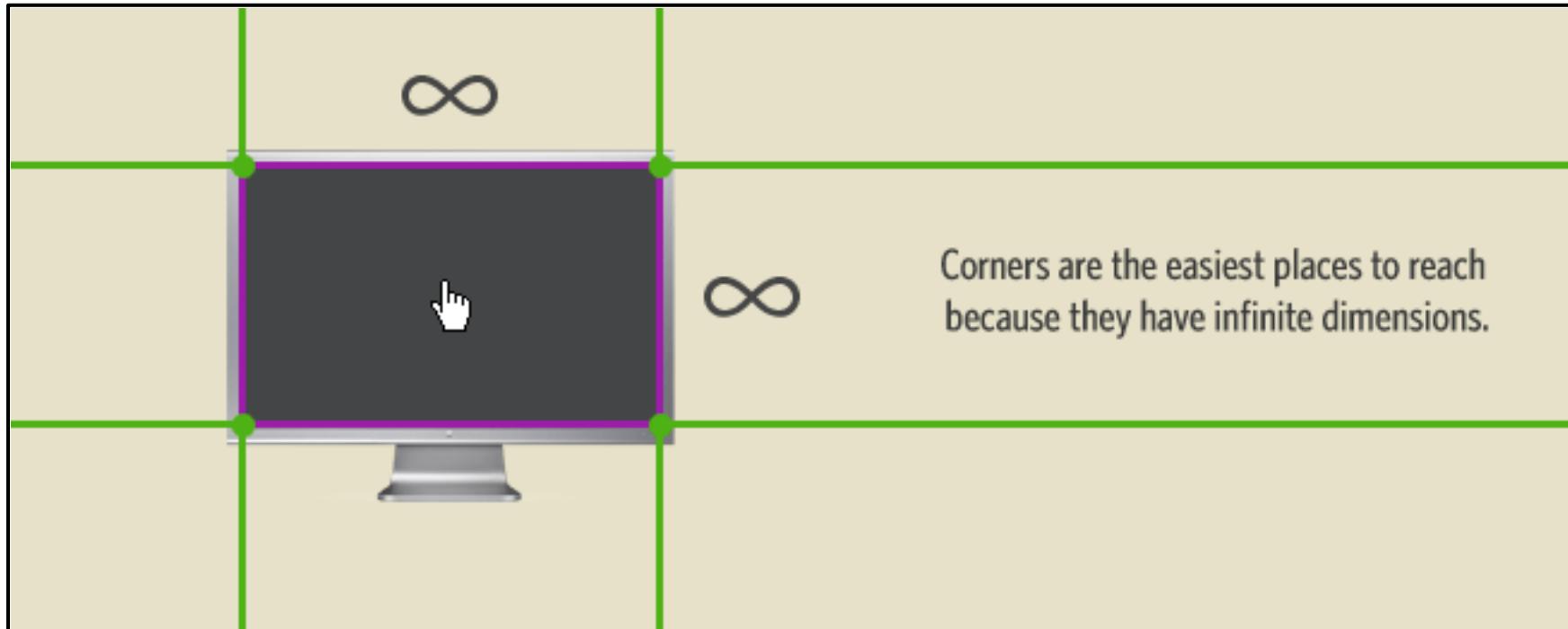


$$MT = a + b \log_2 \left(\frac{D}{W} + 1 \right)$$

FITTS' LAW IN UI DESIGN. APPLICATIONS



FITTS' LAW IN UI DESIGN. APPLICATIONS



FITTS' LAW IN UI DESIGN. APPLICATIONS

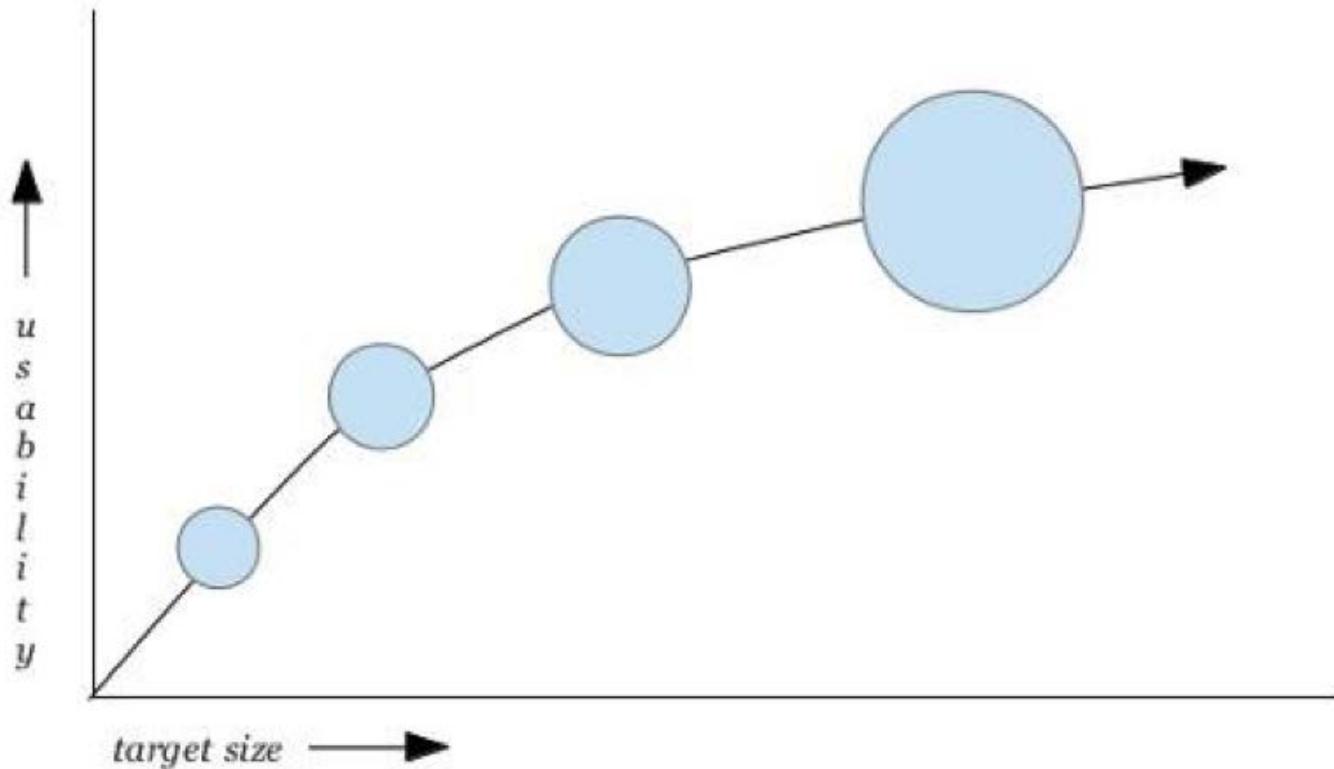


Web sites do not have edges or
corners of infinite width.

:(|

FITTS' LAW IN UI DESIGN. APPLICATIONS

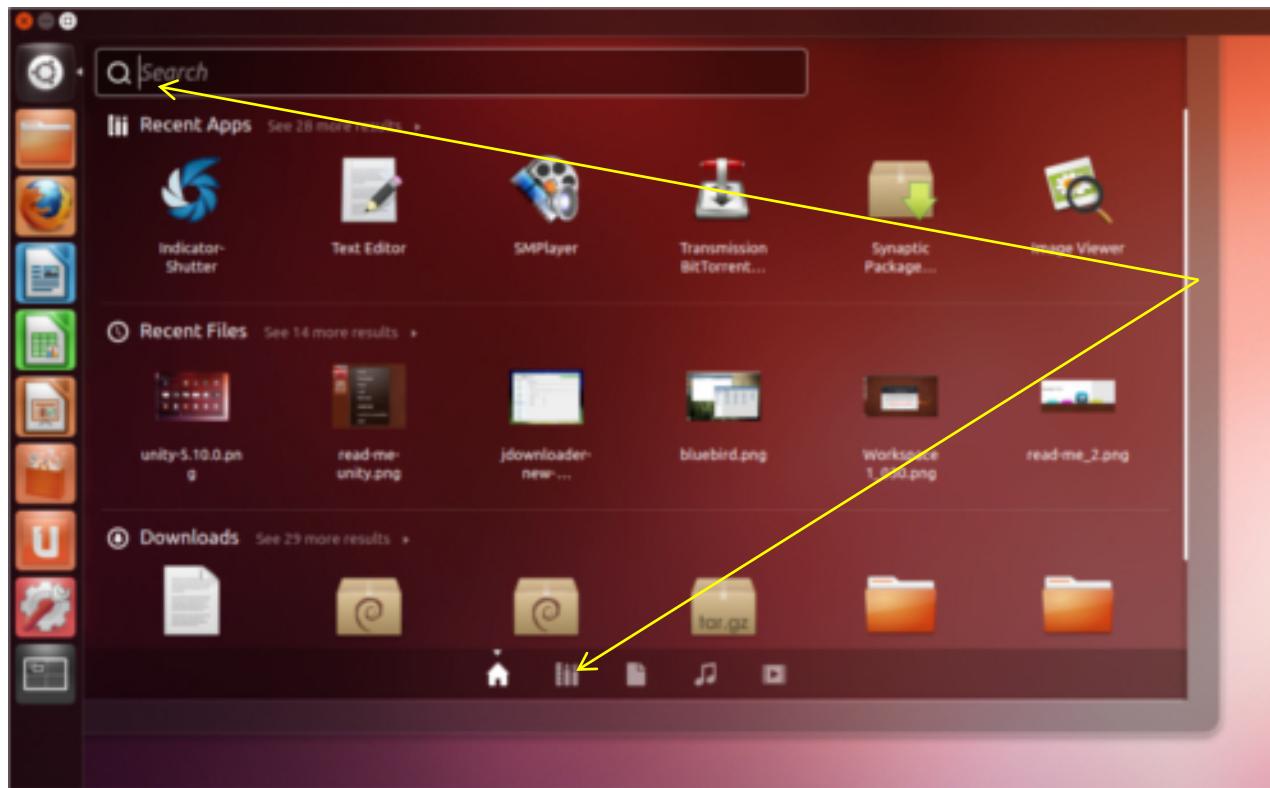
Create larger target size



FITTS' LAW IN UI DESIGN. APPLICATIONS

Keep related things close

- Filters should be placed close to the search field



FITTS' LAW IN UI DESIGN. APPLICATIONS

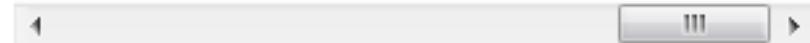
Keep related things close

- Mac OS scrolls are faster to navigate

OSX Snow Leopard



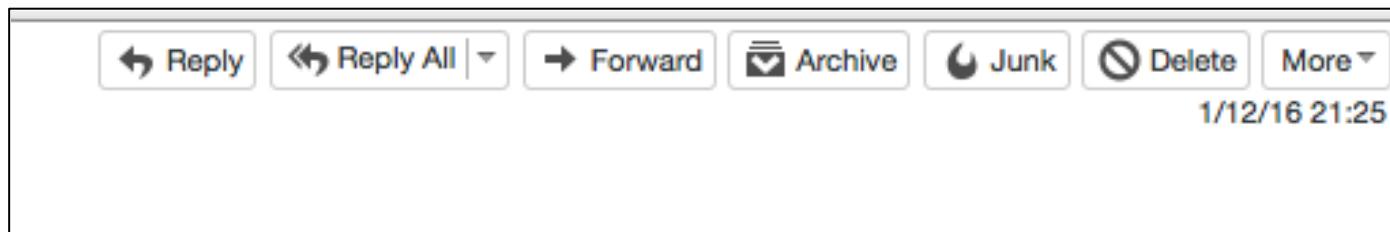
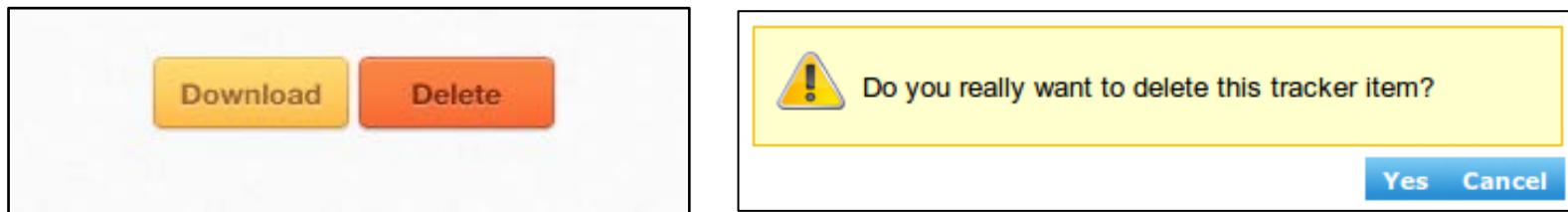
Windows



FITTS' LAW IN UI DESIGN. APPLICATIONS

Keep related things close and **Opposite Elements Far**

- These buttons should be placed far away from each other



But...don't forget the usability principles!!!

FITTS' LAW IN UI DESIGN. APPLICATIONS

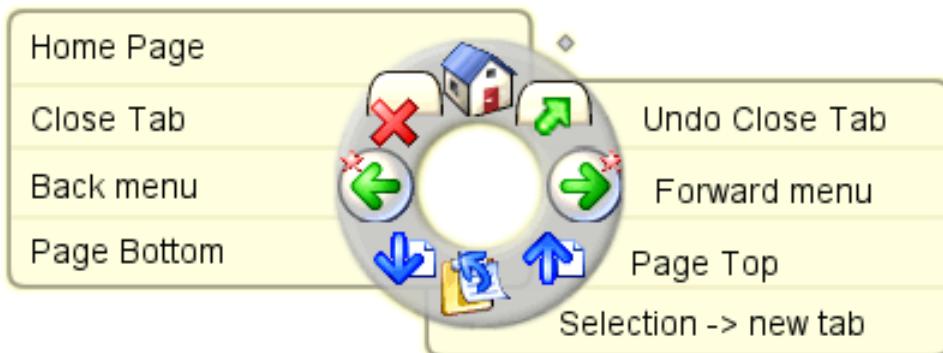
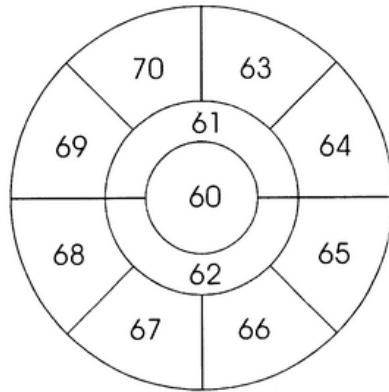
Pop-up menus: Reduce travelling distance

- **Improve two aspects:**
 - Reduction of distance to travel (Fitts)
 - The option is close to the menu emerging place
 - Frequency-enabled may improve the time to pick an option:
 - Based on Hick-Hyman:
Recall that users are able to point faster objects that are known
- Only used by experts!



FITTS' LAW IN UI DESIGN. APPLICATIONS

- *What about pie menus?*



FITTS' LAW IN UI DESIGN. APPLICATIONS

- *What about pie menus?*

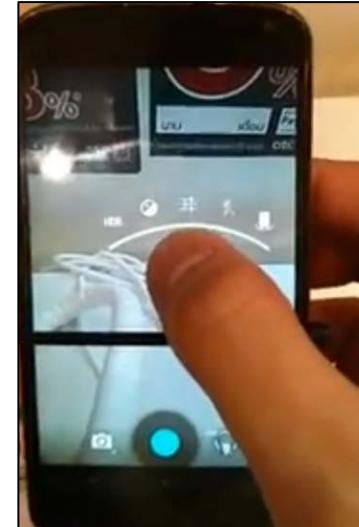
- Sort of contextual menu

- Needs to be created on demand

- **Needs some room!**

- Should not have occlusions

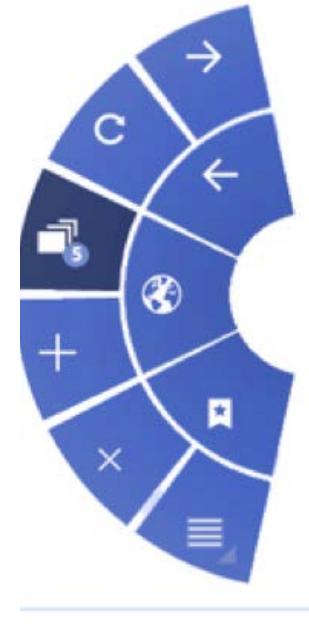
- On mobile half-pie menus better than fully circular



FITTS' LAW IN UI DESIGN. APPLICATIONS

Pie menus difficult to design!

- Second layer changes the size and distance
- Organizing by frequency may be a problem (learning)



FITTS' LAW IN UI DESIGN. APPLICATIONS

+ Perception: Grouping things may improve over distance



OUTLINE

Session 1:

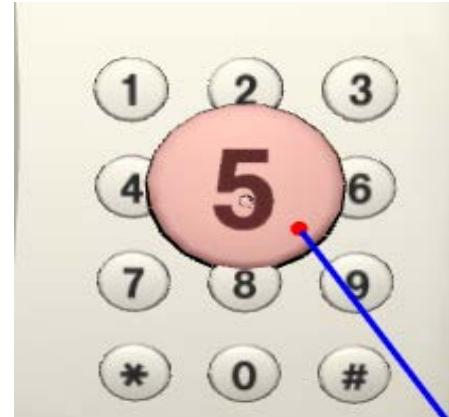
- *Understanding the fundamentals of basic interaction in UI*
 - *Background (Information Theory)*
 - *Hick-Hyman Law: Measuring Choice-Reaction Time*
 - *Fitts' Law: Measuring Pointing Time*
 - *Crossing and Steering Laws: Continuous Gestures*
- **Fitts' Law in UI Design**
 - *Applications in UI Design*
 - **Accelerating Target Acquisition:**
 - Dynamic Expanding Targets
 - Target moving
- Exercises

ACCELERATING TARGET ACQUISITION: EXPANDING TARGETS

➤ Increase the size of targets close to the pointer

Two implementation approaches:

- Size-enlargement and position-changing icons
- Enlarged icons overlap over their neighbours

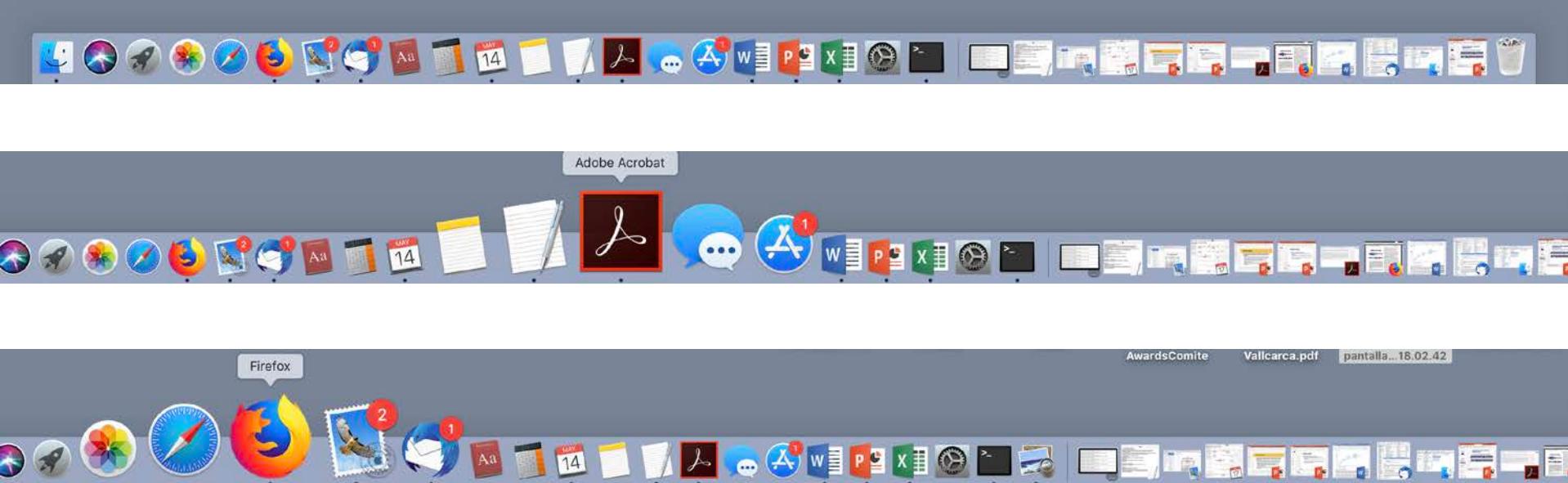


ACCELERATING TARGET ACQUISITION: EXPANDING TARGETS

Increase the size of targets close to the pointer

Exemple 1: Implemented in Mac OSX Dock:

- Mix of target size increase and moving target



ACCELERATING TARGET ACQUISITION: EXPANDING TARGETS

Enlarged icons overlap over their neighbours

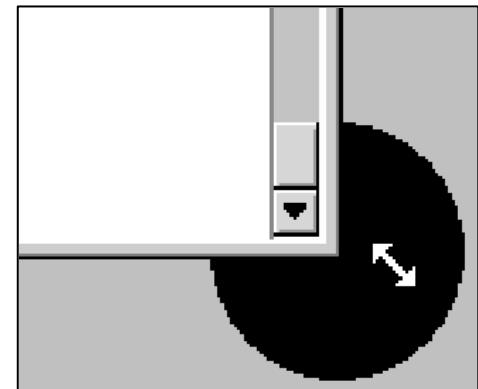
Dynamic Scaling (DS)

Objects near the selection ray are dynamically scaled

ACCELERATING TARGET ACQUISITION: EXPANDING TARGETS

➤ **Bubble targets:**

- Increase selectable region around target
 - Only when the mouse is close
 - Improves selection times
- Issues:
 - Bubble appearing may distract users
 - Overlapping targets:
Close selection points may generate several bubbles

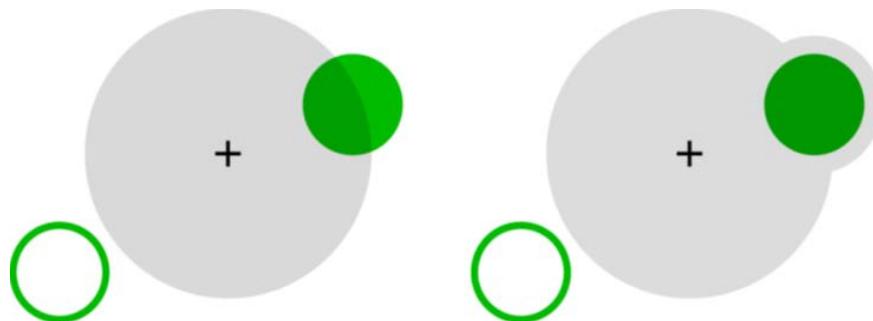


ACCELERATING TARGET ACQUISITION: EXPANDING TARG

➤ **Bubble cursor** [Grossman2005] →

Reduction of amplitude movement

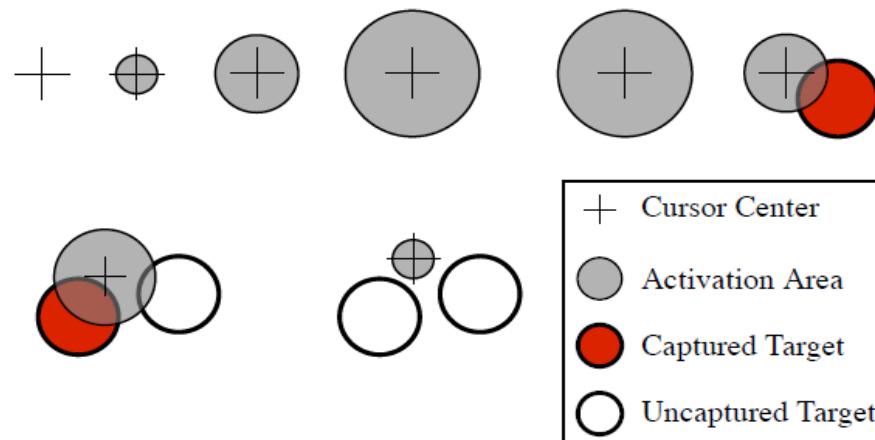
- Cursor size increases when it is close to objectives
- It may even grow to *absorb* the closer target when it is not completely inside the main cursor bubble.
 - Based on position, no speed
 - In experiments Control-Display ratio fixed to 1



ACCELERATING TARGET ACQUISITION: EXPANDING TARG

➤ **Dynamic Bubble cursor** [Chapuis2009]:

- Based on the Bubble cursor idea
- It takes into account the speed of the mouse
 - Area increases according to speed and position
 - Visual cues to indicate the captured target: the target closer to the cursor center.

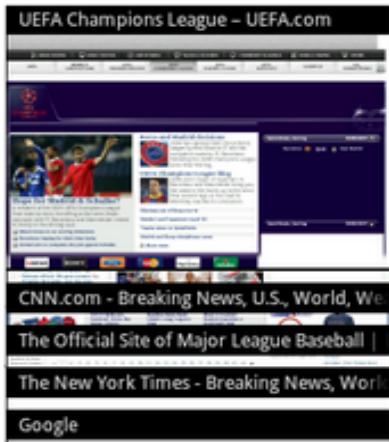


ACCELERATING TARGET ACQUISITION: TARGET MOVING

- Move targets to the user:



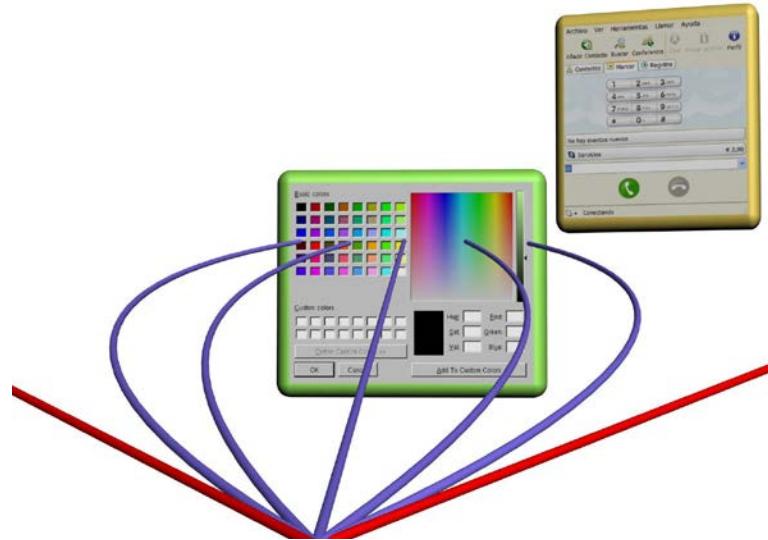
- Generate targets next to the user: pop-up menus



ACCELERATING TARGET ACQUISITION: TARGET MOVING

➤ Sticky targets:

- Attract pointer
 - When the pointer is close to a selectable area
 - May reduce selection time
 - Precision not required
 - Users adapt easily



ACCELERATING TARGET ACQUISITION: CONTROL-DISPLAY RATIO

- Relation between the amplitude of movements of the user's real hand and the amplitude of movements of the virtual cursor
- Moves in real world (physical move) mapped to moves in virtual desktop (cursor move)
- Different strategies:
 - Constant
 - Dependent on mouse speed
 - Dependent on cursor position
- Interpretation according to Fitts Law:
Dynamic C-D ratio adaptation can be interpreted as dynamic change of physical motor space

ACCELERATING TARGET ACQUISITION: CONTROL-DISPLAY RATIO

- Mac OSX and Windows both use mouse acceleration
 - When mouse moves fast, it is accelerated
 - Reducing the amplitude of movement to cover large distances
 - When mouse moves slow, it is decelerated
 - Magnifying amplitude of movement to improve precision
- No clear how the mapping affects perception and productivity
 - Some studies say it is not intuitive
 - Some studies say it improves some pointing tasks

OUTLINE

Session 1:

- *Understanding the fundamentals of basic interaction in UI*
 - *Background (Information Theory)*
 - *Hick-Hyman Law: Measuring Choice-Reaction Time*
 - *Fitts' Law: Measuring Pointing Time*
 - *Crossing and Steering Laws: Continuous Gestures*
- *Fitts' Law in UI Design*
 - *Applications in UI Design*
 - *Accelerating Target Acquisition*
- **Exercises**

Donades les constants $a = 400$ ms, $b = 200$ ms/bit i un objectiu de mida 2.1 cm a una distància de 10.5 cm. Marca la resposta correcta assumint que fem els càlculs amb la versió de McKenzie de la llei de Fitts.

- a. ID ≈ 3.4 .
- b. $2 < \text{ID} < 3$.
- c. ID ≈ 4.3 .
- d. MT està entre 1100 i 1200 ms.

La llei de Hick-Hyman:

- a. Modela el temps de decisió com una funció de la informació transmesa.
- b. Modela el temps de selecció d'un element com a funció de la distància a recórrer i la mida de l'element.
- c. Modela el temps de decisió com una funció de la distància a recórrer i l'entropia dels elements a seleccionar.
- d. Utilitza l'entropia de Shannon per a mesurar la distància del recorregut mínim.

$$MT = a + b \log_2 \left(\frac{D}{W} + 1 \right)$$

- Els *expanding targets*:

- a. Es basen en la llei de Hick-Hyman.
- b. Pretenen reduir el temps d'accés als elements basant-se en el fet que, segons la llei de Fitts, el temps d'accés es redueix si s'augmenta la longitud del desplaçament.
- c. Si es combinen amb el moviment dels objectius poden causar confusió a l'usuari.
- d. Cap de les anteriors.

Ens han encarregat fer un disseny d'una interfície per a un sistema tipus desktop en la qual hi haurà botons i menús drop-down.

- a. Podem predir la dificultat d'accedir als botons utilitzant la llei de Fitts i la dificultat de recórrer els menús amb la llei de crossing.
- b. Podem analitzar el nombre d'elements a posar en un menú utilitzant la llei de steering i en funció dels digrams.
- c. Podem analitzar el nombre d'elements a posar en un menú utilitzant la llei de Fitts.
- d. Podem analitzar la dificultat de recórrer els menús utilitzant la llei de steering.

▪ **La llei de steering:**

- a. No es pot derivar a partir de la llei de *crossing*.
- b. Serveix per a modelar el temps necessari per a recórrer un camí de forma arbitrària.
- c. Diu que hi ha una relació logarítmica entre l'índex de dificultat de creuar un objectiu i el temps que requerit per a fer-ho.
- d. Diu que l'índex de dificultat de creuar un objectiu és D/W.

- Dos elements T1 i T2 a distàncies $D_1 = 10$ cm i $D_2 = 8$ cm en direcció horitzontal i d'amplades 5 cm i 2 cm, respectivament. Per a T1 emprem un dispositiu amb $a_1 = 200$ ms i $b_1 = 200$ ms/bit. Per a T2 utilitzem un dispositiu amb $a_2 = 200$ ms i $b_2 = 100$ ms/bit. Assumint la formulació original de la llei de Fitts:
 - ID₁ > ID₂.
 - ID₁ = ID₂.
 - MT₁ = MT₂.
 - MT₂ < MT₁.

OUTLINE

Session 1:

- *Understanding the fundamentals of basic interaction in UI*
 - *Background (Information Theory)*
 - *Hick-Hyman Law: Measuring Choice-Reaction Time*
 - *Fitts' Law: Measuring Pointing Time*
 - *Crossing and Steering Laws: Continuous Gestures*
- *Fitts' Law in UI Design*
 - *Applications in UI Design*
 - *Accelerating Target Acquisition*
- **Exercises**

Interaction Design and Evaluation

Pere-Pau Vázquez – Dep. Computer Science, UPC

Contents

<u>1 INTRODUCTION</u>	1
<u>2 BACKGROUND</u>	1
2.1 INFORMATION THEORY	1
2.2 INFORMATION AND UNCERTAINTY	2
2.3 SHANNON ENTROPY	4
<u>3 CHOICE-REACTION TIME</u>	4
3.1 HICK-HYMAN LAW	4
3.2 EVIDENCES OF HICK-HYMAN LAW	5
<u>4 FITTS' LAW – LAW OF POINTING</u>	6
4.1 FORMULATION	6
4.1.1 INITIAL RESEARCH	6
4.1.2 REFINEMENTS TO FITTS' LAW	7
4.1.3 IMPLICATIONS OF FITTS' LAW IN UI DESIGN	8
4.2 EXAMPLE	10
4.3 FITTS' LAW EXTENSIONS	10
4.3.1 FITTS LAW FOR 2D	10
4.3.2 FITTS FOR FINGER TOUCH	11
4.4 FITTS' LAW IN KEYBOARDS	12
4.4.1 KEYBOARD LAYOUTS	12
4.4.2 WHY EXPERIMENTING WITH KEYBOARD LAYOUTS IS DIFFICULT	13
4.4.3 TOUCH-SCREEN KEYBOARDS	14
4.4.4 DIGRAM-BASED KEYBOARDS FOR SINGLE-FINGER TYPING	15
4.4.5 SWIPE-BASED KEYBOARDS	16
4.5. EVIDENCES OF FITTS LAW	17
4.6 FITTS VARIANTS AND LIMITATIONS	18
<u>5 APPLICATIONS OF FITTS' LAW IN INTERFACE DESIGN</u>	19
5.1 USE OF SCREEN EDGES	19
5.2 PLACING RELATED THINGS CLOSE	21
5.3 COMPLICATE THE ACCESS TO ELEMENTS	22
5.4 PIE MENUS	22
<u>6 LAW OF CROSSING</u>	23
6.1 INTRODUCTION	23
6.2 CROSSING CONFIGURATIONS	25
6.2.1 CONTINUOUS VS DISCRETE	25
6.2.2 COLLINEAR VS ORTHOGONAL	25
<u>7 LAW OF STEERING</u>	26

7.1 INTRODUCTION	26
7.2 STEERING THROUGH STRAIGHT PATHS	27
7.3 EVALUATIONS OF STEERING LAW	27
<u>8 ACCELERATING TARGET ACQUISITION</u>	<u>29</u>
8.1 INTRODUCTION	29
8.2 MODIFICATION OF TARGET WIDTH	29
8.2.1 TARGET AND SCREEN SIZE	30
8.2.2 EXPANDING TARGETS	30
8.2.3 BUBBLE TARGETS	31
8.3. INCREASING CURSOR SIZE	32
8.4 TARGET MOVING AND OTHER TECHNIQUES	34
8.4.1 TARGET MOVING	34
8.4.2 OTHER TECHNIQUES	35
8.5 DISCUSSION	36
8.6 CONTROL DISPLAY RATIO	36
8.6.1 CONCEPT	36
8.6.2 CONTROL-DISPLAY RATIO ADAPTATION TECHNIQUES	36
<u>9 POINTING TASKS</u>	<u>37</u>
9.1 POINTING DEVICES	38
9.1.1 DIRECT-CONTROL DEVICES	38
9.1.2 INDIRECT-CONTROL DEVICES	41
9.2 COMPARISON OF POINTING DEVICES	41
<u>10 3D SELECTION</u>	<u>42</u>
10.1 INTRODUCTION	42
10.2 DEFINITIONS	43
10.3 3D SELECTION	43
7.3.1 3D SELECTION TECHNIQUES	43
10.3.2 RAY-BASED SELECTION AND POINTING	44
<u>11 BIBLIOGRAPHY</u>	<u>47</u>

1 Introduction

Successful interface design usually leads to positive feelings and enthusiasm among users. An enthusiastic user will likely recommend the application to another user. He or she will also be satisfied with his or her competence in achieving the tasks, the knowledge of the interface, and will enjoy its use.

Such satisfying interfaces are also referred to as *direct-manipulation interfaces*. They provide visible support for the actions of interest, that is, objects and actions are carried out with visual feedback. Moreover, they also provide rapid, reversible, and incremental actions. Typed commands are replaced by pointing actions on the object of interest.

In order to provide the best service to our users, we need to design the interaction properly. Some tasks will be easily accomplished by direct manipulation, such as when editing images, but other tasks, such as aligning objects in an image can be easier through a menu option than moving the objects directly. Therefore, we must properly select the adequate interaction for each task.

Pointing and selection are two related fundamental tasks in graphical user interfaces. Object selection techniques involving physical interaction are constrained by the human motor system as the speed and the accuracy of any gesture are limited by the nervous and muscular systems. In order to understand how we process information and interact with elements in the interfaces, it is very useful to know which laws seem to govern our behavior (reading capacity, clicking time and effort...). Getting a knowledge into these laws will make possible to analyze and to design better user interfaces.

The objective of this document is to present both the Hick-Hyman law and the three “Laws of Action”, as defined by some researchers (e. g. Zhai), as well as to point some guidelines to use them in effective user interface design.

2 Background

Pointing and selection tasks have been studied thoroughly from a theoretical point of view. We will present here the theories that try to give an explanation on users' performance on pointing and selection tasks. Since all of them are based on a previous mathematically grounded theory: *Information Theory*, we will give first some insights on it.

2.1 Information Theory

Information Theory was born as theory that dealt with data communication. In 1948 Claude E. Shannon presented his seminal work *A Mathematical Theory of Communication* [Shannon48]. Based on previous works by Nyquist and Hartley [Nyquist24, Hartley28] on the transmission of electrical signals for telegraphic communication, he developed a measure named Shannon Entropy that measures the amount of information to be transmitted by a message. Shannon's approach to communication uses the following elements:

2 | Interaction Design and Evaluation

- **Information source:** The element that produces a message or sequences of message.
- **Transmitter:** Operates on the message to make it transmissible through a medium.
- **Channel:** The medium that transmits the message.
- **Receiver:** The element that reconstruct the message to the destination.

Shannon's system was a telegraph. In this context, the message is encoded into a signal and this signal transmitted through the channel. The receiver transforms back the signal into a message and delivers it.

The general scheme of this communication process appears in Figure 1.

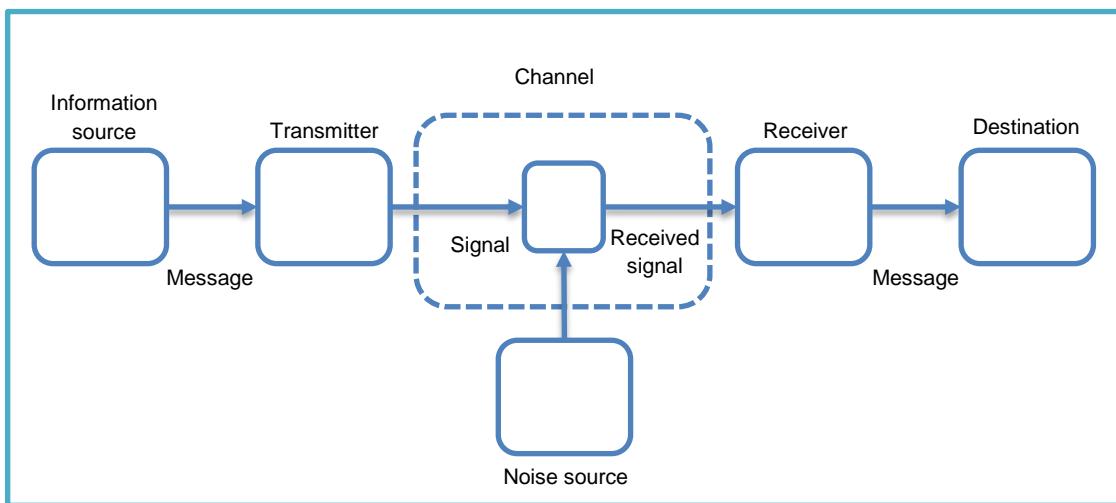


Figure 1: Diagram of a general model of communication system.

The amount of information that a communication channel transmits in a fixed amount of time is referred to as the channel capacity. Channels are bound by physical limitations and thus have different capacities. Other factors, such as noise, affect the communication channel. Thus, the effective capacity is always lower than are the theoretical specifications.

2.2 Information and Uncertainty

Information and Uncertainty are two terms that appear commonly together in Information Theory, so they need a deeper discussion. The first thing to take into account is that those two terms are used to describe the process of selecting one or more elements from a set of elements.

The following development is taken from Schneider's "Information Theory Primer" [Schneider2013].

Suppose we have a device that generates any of these 3 symbols: A, B, or C, every time we ask for one symbol. Before the symbol is produced, we do not know which is the one to come. We can say that we are *uncertain* on the next symbol. Once a symbol has appeared, since we know which of the symbols has appeared, our uncertainty decreases. It decreases because we *have received information*. That is, **information is a decrease in uncertainty**.

Then it comes the problem of measuring this uncertainty. Without many details on how they appear, we may say that a device that produces three different symbols has an uncertainty of $\log(3)$. We will discuss the units later. Then, if we have a second device that produces two different symbols (say 1 and 2), we will say that it has an uncertainty of $\log(2)$. We can make the following experiment: we iteratively produce elements from the first and the second device and combine both. If we do this, the number of different (combined) elements we will have will be six: A1, A2, B1, B2, C1, and C2. Note that this implies an uncertainty of $\log(6)$. Intuitively, when we combine the two devices, we would expect the uncertainty to increase. Actually, if we add the uncertainties of the both original devices we will have $\log(3)+\log(2) = \log(6)$. So if we use logarithms to encode uncertainty, we can add it as we would intuitively expect.

This can be applied with any logarithm, but the typically used here are logarithms with base 2. If we use such logarithms, the value measured by the uncertainty will be *bits* [Shannon54].

Thus if a device produces one single symbol, we are uncertain by $\log_2 1 = 0$ bits. This means that we have **no uncertainty** about what the device will do next, since it will produce the symbol we expect. If, on the contrary, we have a device that produces two symbols, the uncertainty would be $\log_2 2 = 1$ bit.

So far, we have assumed that our devices produce symbols with equal probability, and in that case, the uncertainty is $\log_2(M)$, where M is the number of different symbols a device may produce. If we have M symbols and the probability of producing any symbol is the same, we can say that the probability we have is $1/M$. Then, we can change the logarithm in the following way:

$$\log_2(M) = \log_2\left(\left(\frac{1}{M}\right)^{-1}\right) = \log_2(P)^{-1} = -\log_2(P)$$

That is, the uncertainty of getting a certain value is the $-\log$ of the probability of this symbol. This value, also called the *surprise* (or *surprisal*) indicates whether we may or not expect a certain value. If we are *certain* on a certain symbol, there will be no *surprise* when it emerges. If we recall the previous example, we have total certainty when the set of symbols is composed by a single element. In that case, the surprise ($\log_2(P)$) will be zero, as can be seen when we calculate it: $\log_2(1) = 0$.

For the different symbols that can be produced, we have that their probability sums up to 1:

$$\sum_{i=1}^M P_i = 1$$

This can be used to analyze the common case where the probability of producing different symbols varies. That is, some symbols appear more often than others. In this case, we may have that probability values are not all $1/M$, but this value can change (provided that the total sum adds up to 1). In that case, we will be more *surprised* to see a low probability symbol than seeing a higher probability symbol.

2.3 Shannon Entropy

As said, information is defined in Information Theory as a reduction in uncertainty. And the purpose here is to measure the information.

The classical measure of information, as presented by Shannon is measured over a distribution of probabilities. It is defined as the average *surprisal* from a set of symbols produced by a device. If we measure the surprise for the infinite set of symbols that can be produced by a device, the frequency of each symbol transforms to the probability. Then, if we sum over all the symbols set, we will get:

$$H = \sum_{i=1}^n p_i \log_2 \left(\frac{1}{p_i} \right) = - \sum_{i=1}^n p_i \log_2 p_i$$

where n is the number of alternatives, and p_i is the probability of the i th alternative (here I changed P by p since it is the common notation found in literature). H is the entropy of the message that is to be transmitted. It is actually the amount of information expected to be received at the destination. It is also designated as $H(x)$. Since there is usually an interference during transmission, the information actually received in destination is reduced, and it is designated $H(y)$. The average information that is faithfully transmitted is calculated as:

$$R = H(x) - H_y(x)$$

where $H_y(x)$ is the *equivocation*, or the conditional entropy of x when y is known. As we will see later, in Hick's and Fitts's laws, when a participant commits no errors, he or she is said to be extracting all the expected information of the stimuli.

3 Choice-reaction time

3.1 Hick-Hyman Law

William E. Hick was one of the first to apply Information Theory to psychological problems. His theory had as objective to measure the relationship between choice reaction time and stimulus information content (entropy). It has been previously discovered, as early as 1868 [Donders68] and Merkel [Merkel85], that it takes longer to respond to a stimulus when it belongs to a large set as opposed to a smaller set of stimuli. This regularity caught the attention of psychologists, who saw its analogy to the classic Information Theory.

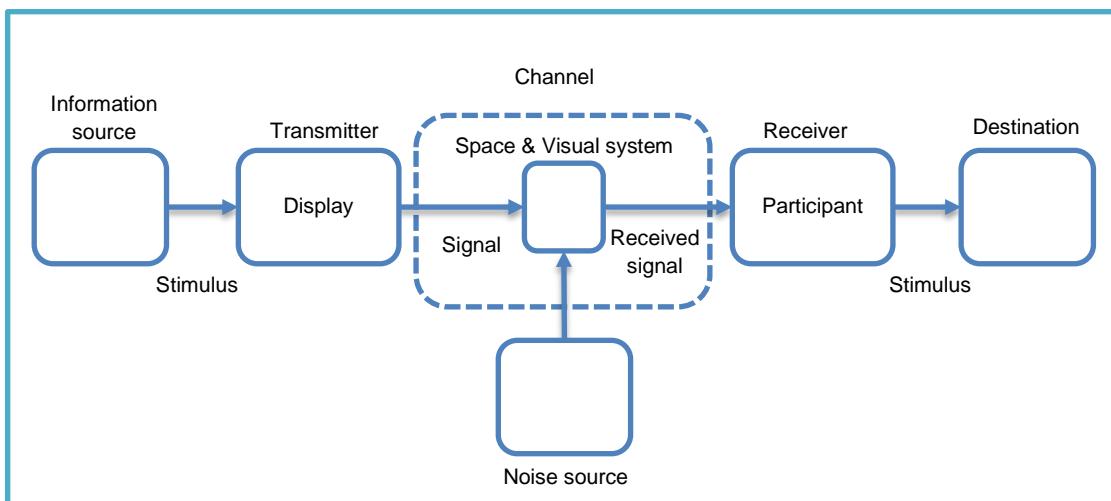


Figure 2: Diagram for the choice-reaction time experiment as a model of communication system.

Hick's Law describes human decision time (or *Reaction Time*) as a function of the information content conveyed by a visual stimulus. After his first experiments [Hick51, Hick52], Ray Hyman [Hyman53] extended his work, and nowadays this law is also called Hick-Hyman Law [Seow2005]. It is defined, in its most general way, as:

$$RT = a + b H_T$$

where a and b are empirically determined constants, and H_T is the transmitted information. The transmitted information is commonly fitted as

$$H_T = \log_2(n + 1)$$

where n is the number of equally probable alternatives and the $+1$ indicates the uncertainty about whether to respond or not, as well as about which response to make [Card83].

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.

3.2 Evidences of Hick-Hyman law

Hick-Hyman's Law has been demonstrated in many experiments, and has been used to justify menu design. For instance, Cockburn *et al.* [Cockburn2007] and Cockburn and Gutwin [Cockburn2008] show that novice users tend to select with a visual search that takes linear time on the number of elements of a menu, while experts decide upon item location, and their decisions times are adequately predicted by Hick-Hyman's logarithmic formula. Allison *et al.* [Allison2004] also showed that Hick-Hyman's Law predicts accurately the decision time in card sorting tasks.

Landauer and Nachbar [Landauer85] observed that expert performance in hierarchical full-screen menu selections is well described by Hick-Hyman and Fitts components applied in series.

Sears and Shneiderman [Sears94] observed that selection times decay logarithmically with menu length for frequently selected items, but linearly with infrequent ones. It seems that participants move from linear visual search with unfamiliar items to Hick-Hyman decision times as the locations are learnt.

4 Fitts' Law – Law of Pointing

4.1 Formulation

4.1.1 Initial research

Published in 1954, Fitts' Law states a linear relationship between task difficulty and movement time (MT). His formulation is also based on Information Theory. In this case, the human motor system is the communication *channel*, the amplitude of movement is the *signal*, and the target width is the *noise*. The task difficulty, in the first writings by Fitts ([Fitts54, Fitts54b, Fitts54c]), is expressed as:

$$ID = \log_2 \left(\frac{2A}{W} \right)$$

where *ID* is the *Index of Difficulty*, *A* is the amplitude of the movement, and *W* is the width of the target. Movement Time is then defined as a function of the Index of Difficulty:

$$MT = a + b ID$$

where *a* approximates the start/stop time in seconds for a given device, and *b* measures the inherent speed of the device. Both must be empirically determined for each device. Note the similarity of this formulation with the Hick-Hyman's Law. Posterior experiments noted that there is a lower adjustment of the regression curve when testing with low IDs. Actually, the relationship originally established by Fitts was using an approximation of Shannon's theorem that is valid only if the signal-to-noise ratio is large ([Fitts54, McKenzie89]).

In order to arrive to this definition, Fitts made a series of four experiments: Two reciprocal or serial tapping tasks (with a 1-oz stylus and 1-lb stylus), a disc transfer task, and a pin transfer task. For the tapping condition, a participant moved a stylus back and forth between two plates as quickly as possible, and tapped the plates at their centers, as shown in Figure 3. Four different target amplitudes (*A*) were tested, and four target widths (*W*).

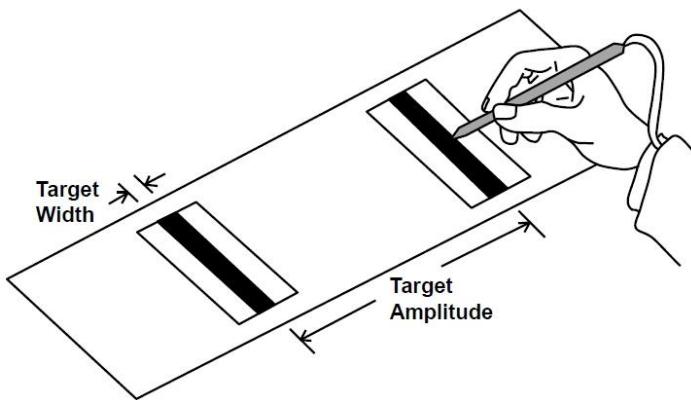


Figure 3: Reciprocal tapping.

As the data obtained by the experiment was published, one can re-examine his results. We can see a plot of the results in Figure 4.

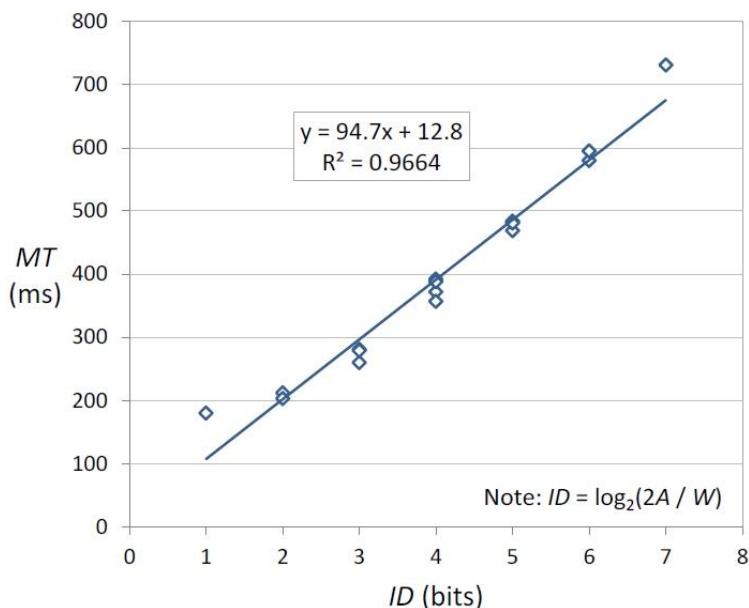


Figure 4: Scatter plot and regression analysis for the data of Fitts.

4.1.2 Refinements to Fitts' Law

After the first publication in 1954, many authors have developed changes or refinements to Fitts' law. The most important rationale behind those is the need for precise mathematical formulations in human computer interaction and other related fields for the purpose of measurements.

An early refinement is due to Welford [Welford68]. He noticed that the MT-ID data points curved away from the regression point, with the most deviate point at ID=1, as can be seen in Figure 4. In order to improve the data-to-model fit, he introduced the following formulation:

$$MT = a + b \log_2 \left(\frac{D + 0.5W}{W} \right)$$

Note that we substituted here the amplitude of movement (A) by the Distance (D). This version has been used frequently over the years, and for instance was the one used in the comparative evaluation of the computer mouse by Card ([Card78]).

Later, it was shown that the relationship deduced by Fitts was based on the approximation of Shannon's theorem. But this only applies if the signal-to-noise ratio is large ([MacKenzie89]). As a result, MacKenzie proposed another refinement to the Fitts' law ([MacKenzie92]):

$$MT = a + b \log_2 \left(\frac{D}{W} + 1 \right)$$

Several versions of Fitts' Law are used, and you can find them consistently in literature, but this formula has been demonstrated to provide good predictions in a wide range of situations.

If we compare the plots of the different models, we can see that they are pretty similar, as shown in Figure 5 ([MacKenzie18]).

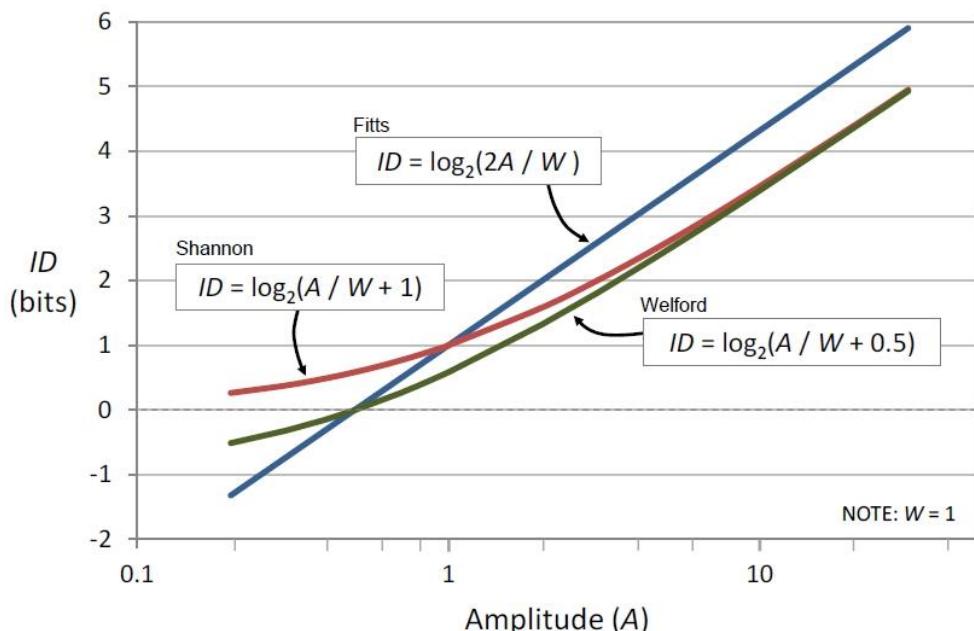


Figure 5: Plots of the main formulations of Fitts' law.

4.1.3 Implications of Fitts' law in UI design

Commonly, the Fitts' Law is applied to the task of pointing or clicking a button. In this case, the parameters involved are the distance D the pointer (e. g. mouse) has to cover, and the width W of the button in the direction of the movement. In Figure 6 all these elements are represented.

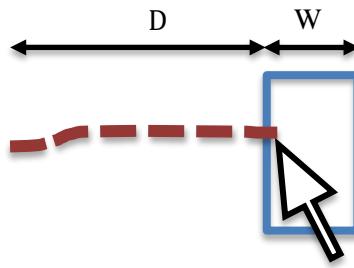


Figure 6: The pointing task according to Fitts' Law.

Several of the other formulae may be better indicated to address some concrete tasks. Note that non trivial variations may be due to differences such as changes in the direction of motion (vertically vs horizontally), device weight (heavier devices are harder to move), shape or size of targets (the original formulation, for instance, does not hold for low ID values), or arm position. So when experimenting with such a priori simple tasks, we must have all the variables under control, because trivially extending the Fitts' law may be invalid. We will see some extensions later.

Fitts' Law describes the Movement Time in terms of one dimensional displacement, as we have already seen. This is valid for either purely horizontal or purely vertical movements. The only thing that has to be taken into account is that we need to use the dimension of the pointing target in question that goes in the direction of the movement. Apart from that, the formula can be applied indistinctively (see Figure 7).

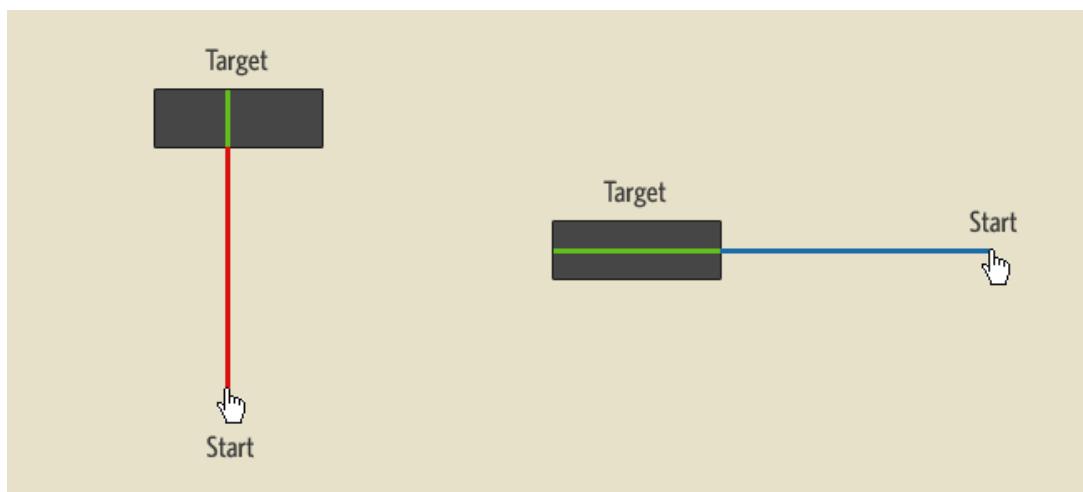


Figure 7: Fitts' Law can be applied to one dimensional movements, either in horizontal or vertical.

Another related measure is the Index of Performance, which is analogous to the Capacity in Shannon's theorem. Some authors call it throughput (TP). IP is measured as bits per unit, and calculated as:

$$TP = ID/MT$$

However, the use of TP , commonly applied to measure performance characteristics of input devices, is quite problematic. The main reason is that TP is only a constant (hence, something that can be representative of the device) when a (also called the intercept) is zero. If a was zero, then the throughput would be $TP = 1/b$ (b is the slope).

4.2 Example

We can show an example here. Let's imagine that we have empirically determined the constants a and b for a certain device. For instance, let $a=300\text{ ms}$ and $b=200\text{ msec/bit}$. If we want to reach a target of 2 cm ($W=2$) at a distance of 14 cm ($D=14$), then, according to Fitts' law, the Movement Time would be

$$MT = 300 + 200 \log_2 \left(\frac{14}{2} + 1 \right)$$

that is, $MT = 900\text{ ms}$.

4.3 Fitts' Law extensions

4.3.1 Fitts law for 2D

The original formulation of Fitts' Law is in one dimension. In all the very initial experiments, the movements demanded to the users were in one direction. With the advent of graphical user interfaces, the navigation region increases, and therefore it is interesting to evaluate the difficulty to reach objectives that require two-dimensional displacements.

Note that, when having non one-dimensional moves, we may point to a target that has different dimensions in height and width (see Figure 8). As a result, we do not have a proper dimension of the target, and some modifications must be applied, since the original formula cannot be used directly.

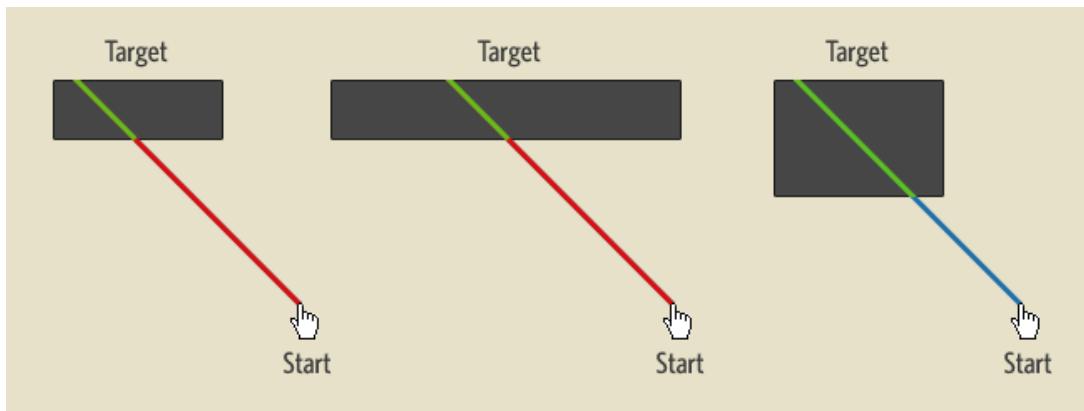


Figure 8: Pointing to a target with a 2D movement.

There have been several developments with this goal in mind. For instance, Crossman and Goodeve [Crossman93] suggested that a second term has to be added to take into account for the height of the target, developing the following formulation:

$$MT = a + b \log_2 \left(\frac{2A}{W} \right) + c \log_2 \left(\frac{2A}{H} \right)$$

Like in the previous cases, the constant c must be determined empirically. W is the amplitude constraint size and H is the height. Other researchers later refined this formulation. A more recent approach is due to Accot and Zhai [Accot97]. They suggest that a better formulation is this:

$$MT = a + b \log_2 \left(\sqrt{\left(\frac{D}{W} \right)^2 + \eta \left(\frac{D}{H} \right)^2} + 1 \right)$$

Where η is a constant commonly in the range 1/3 to 1/7. This indicates somewhat that the directional constraint is less difficult to handle than the amplitude constraint of the same magnitude. Like in the previous case, W is the width and H the height constraints of the target. More formulations have been proposed, but the details are beyond the scope of this course.

4.3.2 Fitts for finger touch

This predictive model is a great help to decide location and size of buttons and other elements in the user interface.

For the case of very small targets, we cannot expect the user to be as efficient as with regularly sized elements. Some studies have found that, for very small elements, there is an extra time devoted to fine adjustment. Moreover, the use of touchscreens also modifies the timing we require to point targets. Sears and Shneiderman ([Sears91]) derived an extension of the Fitts' Law by analyzing the behavior both in tactile screens and small targets. This led to what some authors call the Precision Pointing Movement Time (the original authors call it modified version of Fitt's Law for touchscreens, or FFits):

$$FFits = a + b \log \left(\frac{cD}{W} \right) + d \left(\frac{e}{W} \right)$$

where the first logarithmic factor measures the time to place the finger on the screen initially, while the second factor measures the time to position the cursor once the finger has been placed on the screen. D is the distance, measured in three dimensions, from the original hand location to the location of first contact. W is some measurement of target size. The rest of the constants, a , b , c , d , and e must be determined for each specific case.

If a succession of rapid tasks are performed (e. g. clicking one button and then clicking another), then D will be the distance between both buttons. Since in this formulation we have more than the original two freedom degrees, it might turn that it is easier to fit in a regression curve.

As noted previously, the validity of the formulation is constrained to the datasets on which it was proven. This means, that we cannot simply extrapolate. This is especially important since the experiments in this case were performed in quite early hardware, with much lower precision than current capacitive screens commonly available for most hand held devices. In the case of Sears and Shneiderman, the authors had to implement a stabilization

software that was able to reliably yield a single touch position (though this did not mean that the position resolution detection was even closer to the most accurate mouse interaction, in part due to the resolution of the screen, and in part to the recognition software).

4.4 Fitts' Law in keyboards

The primary method for textual data entry is the keyboard. Keyboards have large history and have even been criticized devices. The average ratio of an office worker is roughly 50 words per minute, although higher rates (up to 150 words per minute) can be achieved. Although regular computer keyboards only allow a key press at a time, some specialized devices, such as the chord keyboards, allow multiple key presses at a time. These may achieve 300 words per minute, but require lengthy training to retain the complex pattern of chord presses.

4.4.1 Keyboard layouts

The typewriter was built in different ways by the middle of the nineteenth century. Several ways to put the paper and several designs for the keys failed. The initial successful proposal was due to Christopher Latham Shole. His design was successful in part because the intelligent placement of the keys facilitated the reduction in key jams. The original layout was called QWERTY due to the keys arrangement in the top left part of the keyboard. This has been the dominant keyboard layout since then. In Europe, some countries use small variations over these layouts, such as the AZERTY layout used in French keyboards or the QWERTZ versions used in Germany (see Figure 9).

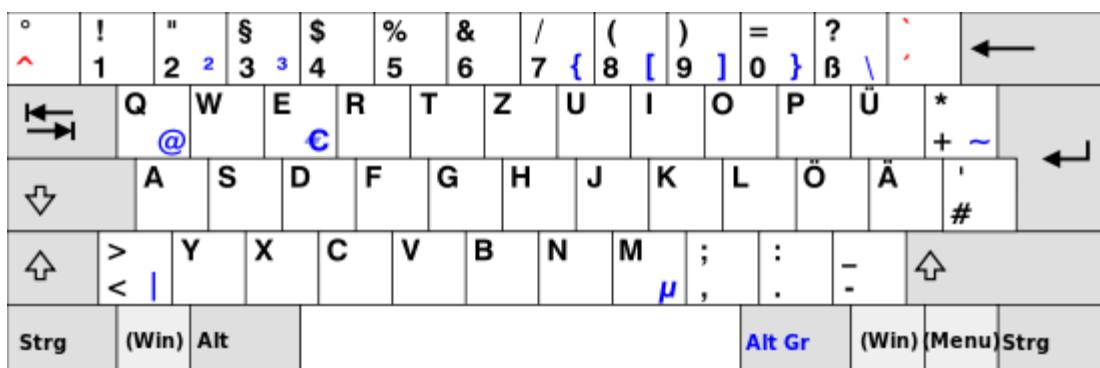


Figure 9: QWERTZ layout used in German keyboards.

By the 20s the Dvorak layout was developed. It was a new arrangement that supposedly reduces finger travel distances by at least one order of magnitude (shown in Figure 10), thereby increasing the number of words per minute approximately a 30% (some other researchers talk of 5-10% of improvement [Norman82]), and reducing errors. Although with a number of devotees, the acceptance of this layout has been low.

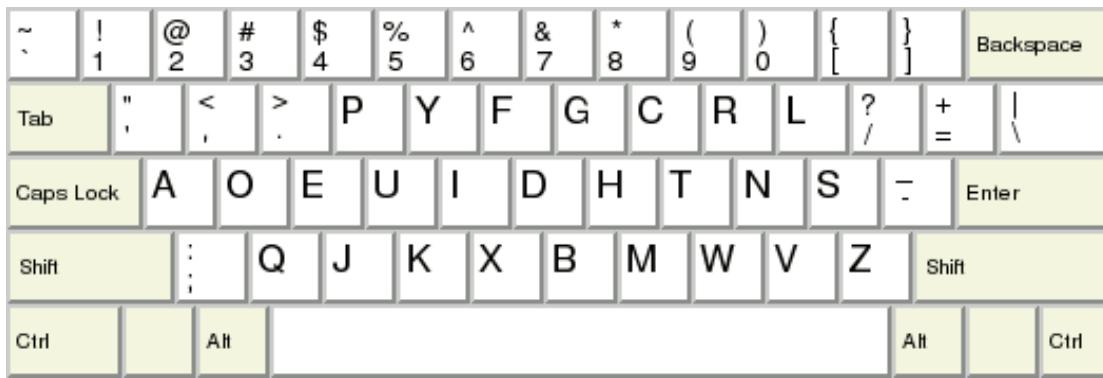


Figure 10: Dvorak layout.

Ideally, as shown by different experiments, for optimum typing speed, keyboard arrangements should be designed so that:

- The loads on the right and left hands should be equalized: both hands do the same amount of work.
- The load on the home (middle) row should be maximized: thus reducing the displacements.
- The frequency of alternating hand sequences should be maximized: this way one may maximize the frequency of typing because the fingers alternate and therefore one does not need to wait for the end of the movement of the first finger before starting the second movement.
- The frequency of same finger typing should be minimized: this may avoid fatigue.

If we analyze the Dvorak layout under these variables, it does a good job, especially on the first two elements: 67% of the typing is done on the home row, and the left-right hand balance is 47-53%. Although the QWERTY keyboard fails at these two conditions, (most of the typing is done on the top row and the balance between the two hands is 57-43%), the policy to put successively typed keys as far apart as possible favors the third condition, thus leading to relatively rapid typing. As a consequence QWERTY layout is superior to other arrangements such as the alphabetical ones. And it is roughly only 5 to 10% (though some reports raise this up to 30%) slower than DVORAK layout.

Note that this is only valid for 10-finger typing. Using this technique, the fingers are placed at fixed positions on the keys (A, O, E, U – for left hand, and H, T, N, S – for the right one), and these makes that very common letters such as vowels, require less effort than other, more uncommon letters. However, different layouts must be evaluated differently if we assume one or two finger typing. As a consequence, the comparison is not quite simple, and the distances the finger must cover may increase in the DVORAK layout in the case of one-finger typing.

4.4.2 Why experimenting with keyboard layouts is difficult

As we have commented many times, the experimentation is a difficult task. You can only assert conclusions on the exact parameters of the experiment. Testing with keyboard layouts adds a second difficulty here: users get their proficiency from practice. Therefore, in order to be able to perform a proper testing, users would require months of training in

any layout we can imagine is the best for typing before one can conduct the user study [Norman82]. Moreover, since the layouts are artificial in some sense, the same people would require to be trained back to the original (e. g. QWERTY) arrangement they use such as to be able to use normal computers.

As a consequence, it is commonly accepted that formal results are yield by using a predictive human performance model rather than user testing for evaluation [Lewis99].

4.4.3 Touch-Screen Keyboards

Modern multi-touch-screens enable text entry that is adequate for mobile interaction, thus bringing touch-screen keyboards to the mainstream. There are many problems when using touchscreens as input method, and several decisions must be taken, such as the keys sizes.

In practice, key size is usually determined by the screen size and its orientation, so the designers try to take advantage of all the possible space, while keeping some room for the display of the entered text. This has a major impact, since the available room is usually not enough: An iPhone 4 character in portrait mode measures 4×5.9 mm, while the Apple Guidelines for user interface design recommend that the minimum clickable size to be 6.85×6.85 mm.

Virtual keyboards also require significant visual attention because the user must look at the screen to press the correct key. Since there is no rest position such as in regular keyboards (which also increases the effort and results in a more fatiguing experience), the user does not know where his or her hands are with respect to the keys. Moreover, there is no physical feedback on the user's actions. Therefore, we can only ensure the key typing by looking at the keyboard and/or the rest of the screen, where the letter is inserted. For larger form-factors, this is especially problematic, due to the relatively large distance from the insertion point to the place where the virtual keys appear.

A second problem is the insertion point: since the screen is touchable, if the user accidentally taps on some part of the screen, the cursor may be changed and the following text ends up being inserted mistakenly at an unintended location.

Some attempts to improve the feedback go from audible “clicks”, to a tactile screen that actually requires pressing on it to effectively enter a key. Its benefits are not clear. For example, the pressing screen, present in mobile devices such as the BlackBerry Storm requires relatively large amount of pressure, and its original layout also lost some precision on the corners (which was improved on the Storm 2 version).

Another big disadvantage of virtual keyboards is that they occlude an important portion of the screen, resulting in less space for the document we are editing or the form we are filling. This is a clear problem when talking of smartphones, and only partially alleviated in tablets.

Some more innovative designs combine the power of touch and stylus-based typing with stroke gestures. They have been shown to produce high rates of text entry, once the user masters them, although it remains unclear if such approaches will achieve widespread adoption.

Finger touch in tactile screens also follows Fitts' Law. Once the users have reached expertise, the time to move the tapping device with a single finger from one key (i) to another (j) depends basically on the distance and key width of the keys, following this equation [Bi2013]:

$$MT_{ij} = a + b \log\left(\frac{D_{ij}}{W_{ij}} + 1\right)$$

Where D_{ij} is the distance between keys i and j , and W_{ij} is the width of each key. They further refine using the effective width of the key, instead of the nominal value, but a deeper analysis goes beyond the needs for our course.

4.4.4 Digram-based keyboards for single-finger typing

As already noted before, the design of an efficient typing keyboard for single-finger typing should minimize the overall displacement of the finger for the most common words. This is what DVORAK actually does for 10-finger typing, but this is not a current typing method in smartphones or tablets. Commonly, one hand is devoted to hold the device and the user types with the other.

Several attempts have shown their proficiency, but usually as a matter of good prediction + excellent error correction together with some tricks to accelerate input (see for example Research In Motion's new keyboard for the Blackberry 10 mobile OS [RIM2012]). However, most of the keyboards still rely on the QWERTY layout, with a notable exception that has attracted a lot of attention: the Minuum keyboard by Whirlscape ([Whirlscape2014]), which uses the QWERTY layout but in a single row (plus a strong word prediction and correction).

For hand-held tablet and portable computers (including pen-based systems), it is important to evaluate the best arrangement of keys for typing layouts when users type with one finger or a stylus. A nonstandard typing-key layout (in a roughly 5 x 5 key matrix) based on digram Predictive human performance models for 10-finger keyboards use the frequency matrix of English-language digrams (pairs of letters that commonly appear together) and a matrix of empirically derived interkey typing times [Lewis99].

To improve the layout for single-fingered typing, it is reasonable to analyze the language digrams to determine the relative associative strengths among the letters so the layout can minimize the distance between strongly associated letters.

This may lead to an arrangement such as the one in Figure 11, where those distances are minimized and thus the predicted improvement raises to approximately 25 words per minute (over the typical 20 words per minute on a complete QWERTY layout and one single finger typing). The authors claim that, after the corresponding training, such a layout should be about 27% better than the QWERTY layout. And even, an alphabetic typing-key arrangement, again in a roughly 5 x 5 key matrix, should be about 13% better than the QWERTY layout for single-finger entry.

Q	R	W	X	Y	
L	U	A	O	F	
Z	T	H	E	N	G
V	D	I	S	P	
B	C	M	J	K	

Figure 11: Diagram-based layout for single-finger typing.

4.4.5 Swipe-based keyboards

Entering text has always been painful in the mobile space. There are many issues that affect the text entry, especially on mobile. To name a few:

- Size of the keyboards: As already noted, the keys' size is smaller than most UX guidelines recommend, especially in smartphones. As a consequence, the size of the keyboard is prone to errors in typing.
- One hand vs two hand entry: When having both thumbs free, we may probably write better, but it is quite common to have to type with one single finger, which, again, is more prone to errors because the hand location is more variable (than two fingers with the rest of the hand in a fixed position). But even in those cases, the number of errors is quite larger than with a regular keyboard.

There is a tendency in mobile space to provide different keyboard interaction techniques for accelerating text entry. Sometimes this acceleration is real, while sometimes it is only felt by the users [Nguyen2012].

Gesture typing basically consists on using a single trace that goes from the first letter of the word until the last one without lifting off the finger, and by moving it on the screen over all the letters of the word. This is illustrated in Figure 12.



Figure 12: Illustration of the gesture keyboarding for entering the word “quick” in an application for gesture-based typing in mobile devices.

One of such tendencies is the use of gestures (swipes) to write a whole word [Kristensson2012, Zhai2012]. Nguyen *et al.* [Nguyen2012] have studied the effect in tablets, where they found that regular typing and gesture typing perform at similar speeds, however, there is a higher user satisfaction ratio when using one of the concrete applications that provide gesture typing (Swype) than regular keying.

Note that all those approaches are commonly executed single handed and in a QWERTY keyboard, which, as we saw before, do not especially reduce the distances the finger has to travel to edit words.

Among the other different improvements, a notable one is a two finger gesture keyboard [Bi2012] which showed that the finger travelling was shortened about the 50% but the speed does not increase over single finger entry, most probably due to the high demand in attention to coordinate both hands. In Figure 13 we can see two proposals for writing “interaction” in such keyboard: one using continuous tracing (right), and the other by interrupting the trace between hands’ change (left).



Figure 13: Two different approaches for a bimanual gesture-based input of the word “interaction”.

4.5. Evidences of Fitts law

The Fitts law has been used for successfully modeling many UI access tasks. However, when assessing the validity of such law, we must be very careful on the analysis of the experiment. Experiments cannot cover all the variety of users and tasks, and therefore are constrained to a set of configurations that may be not very large (e. g. 4 different target sizes and 4 different distances). As a consequence, although Fitts’ law has proven its validity in many experiments, we may not freely extrapolate (e. g. assuming it works for elder people or children...) for all sizes and distances. More concretely, as we have already seen, there are limits of the validity of the Fitts’ Law: for very small targets, for instance, the MT function regression curve starts to be less aligned with the resulting values.

Apart from the concrete limitations given by the experimental setup, Fitts’ Law has shown its validity in multiple setups and devices, which go from mouse, joystick, finger, stylus... and different screen types of varying sizes ([Chapuis2011]).

An interesting observation that has been analyzed in some experiments is the previous knowledge by the users of the targets’ sizes and positions. It is an important issue, because users who select objects in point-and-click interfaces quite often know these features of the targets. Studies show that for precued targets the preparations lead to more efficient and precise pointing movements than for non-precued targets [Hertzum2013]. Target precuing also interacts with pointing device, distance to target, and target size, but not with user age. In particular, the benefit of precuing is larger for the mouse than the touchpad, suggesting that the movement preparations users are able to make on the basis of precues depend on how demanding the pointing device is to use.

It is curious to note that, as Balakrishnan and MacKenzie found [Balakrishnan97], the index finger alone does not perform as well as the wrist or forearm in pointing tasks, but the thumb and index fingers in coordination outperform all above cases.

4.6 Fitts variants and limitations

Note that Fitts' law models properly adult users, but children behave differently. Some results show that for point and click tasks, Fitts' law adequately predicts the first time the children enter a target, but not the time of the final selection.

Some researchers suggest measuring the ID using the *effective target width* (or W_e), instead of the *nominal width*. This arises from the observation that users do not adjust their performance as much as might be expected when target width is changed. More concretely, when changing target width, the changes in endpoint variability are typically much smaller than would be expected ([Fitts64, Sourkoreff2004]).

Therefore, this formula, commonly called ID_e is written as:

$$ID_e = \log_2 \left(\frac{W}{W_e} + 1 \right)$$

is the one that is suggested also in the ISO 9241-9 ([ISO2000]). However, there is quite a large controversy on its use. There are two main reasons for that:

1. The data does not always fit better than when using ID .
2. The effective width is the result of the actions of the users. As a result, it cannot be measured previously, and therefore is not useful for design without experimentation.

The first problem is still in debate. Wright and Lee [Wright2013] performed a set of studies and found that "it has been shown to compensate for the differences in effective target width that are present across conditions and participants". However, they do not recommend its use without much care since the obtained gain may not compensate for the extra effort to generate such data. Moreover, they believe that when those values are not available, it is safe enough to proceed using ID .

Zhai *et al.* [Zhai2004] believe that both methods (regressions from uncorrected widths and from effective widths) are valid, but the appropriate method depends on the goal [Chapuis2011]:

- Using uncorrected widths seems useful to reliably assess actual user pointing times in user interfaces, accounting for natural biases in target utilization, and to assess errors separately.
- Using effective widths instead usually deteriorates the fit but is strongly recommended when one needs to interpret the constants a and b , for example, in order to compare the performance of different input devices.

5 Applications of Fitts' Law in interface design

Fitts' Law describes the Movement Time in terms of one dimensional displacement, as we have already seen.

5.1 Use of screen edges

There are several interfaces of Operative Systems that put some of the graphical elements of the UI next to the edges of the screen. This is true, for instance in the Mac OS system (see Figure 14), where the top menu is attached to the top edge of the screen. Since there is no distance from the top edge to the menu, the mouse movement required to select any of the options of the menu does not require precision in the Y direction (while it requires in the X axis). As a consequence, the selection process may benefit because the movement can be done fast (using mouse acceleration techniques that will be visited later).



Figure 14: The Mac OS X operative system has a menu bar on top of the screen.

From the point of view of Fitts' Law, this kind of behavior can be interpreted as having targets with virtual infinite length in one dimension. This, at its time, reduces the ID of the target acquisition and therefore intuitively makes the movement simpler/faster to achieve (see Figure 15).

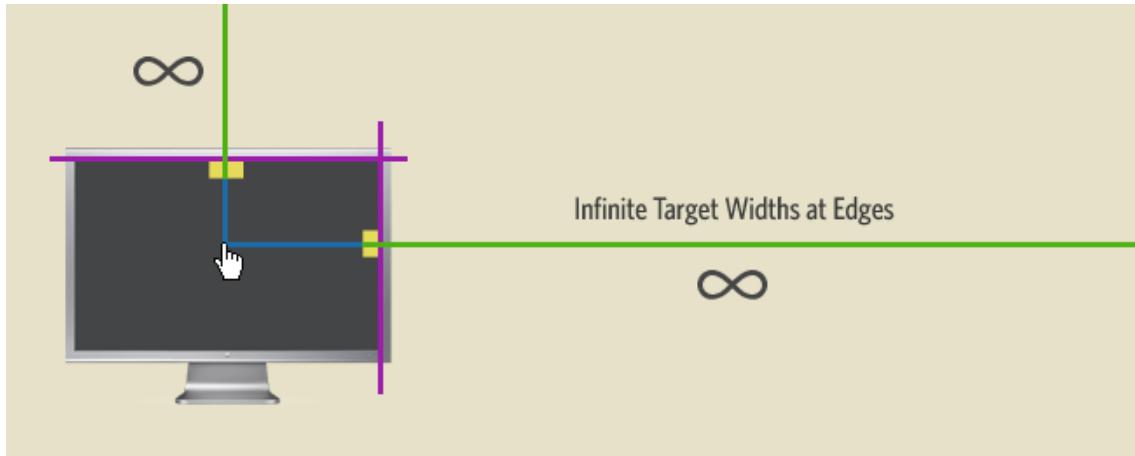


Figure 15: Interpretation of edge elements according to Fitts' Law: They can be considered as having infinite width.

With this in mind, it turns out that corners are the most accessible places in such an UI (see Figure 16), because they have virtual infinite dimensions.

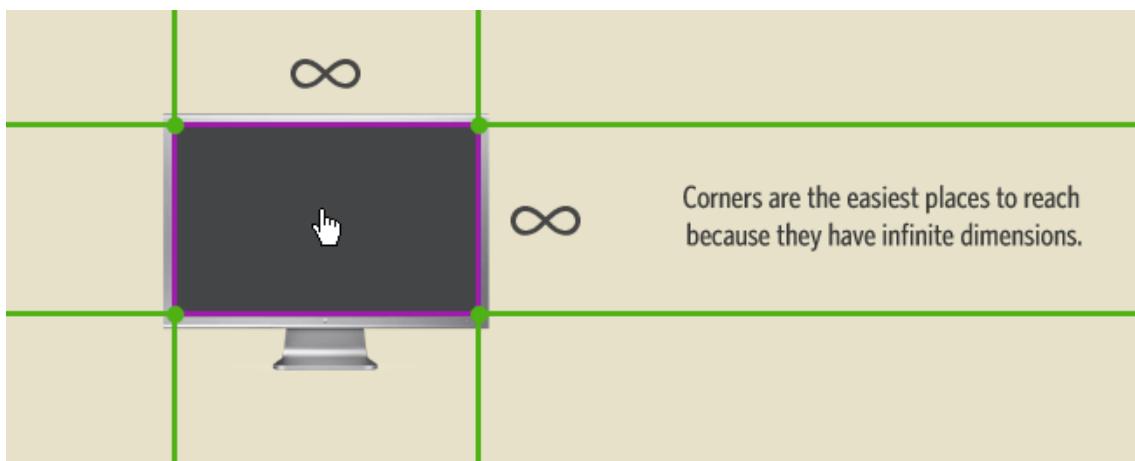


Figure 16 Elements at the corners are the most accessible because they do not require neither horizontal nor vertical precision in one direction.

We may compare this to the typical interface that some programs have presented, such as most Windows applications (in Figure 17 we can see a snapshot of the old Microsoft Word 2003), where the top items were slightly displaced from the top of the screen. We can intuitively see that an extra effort will be necessary for properly placing the pointer in any of the menu titles.

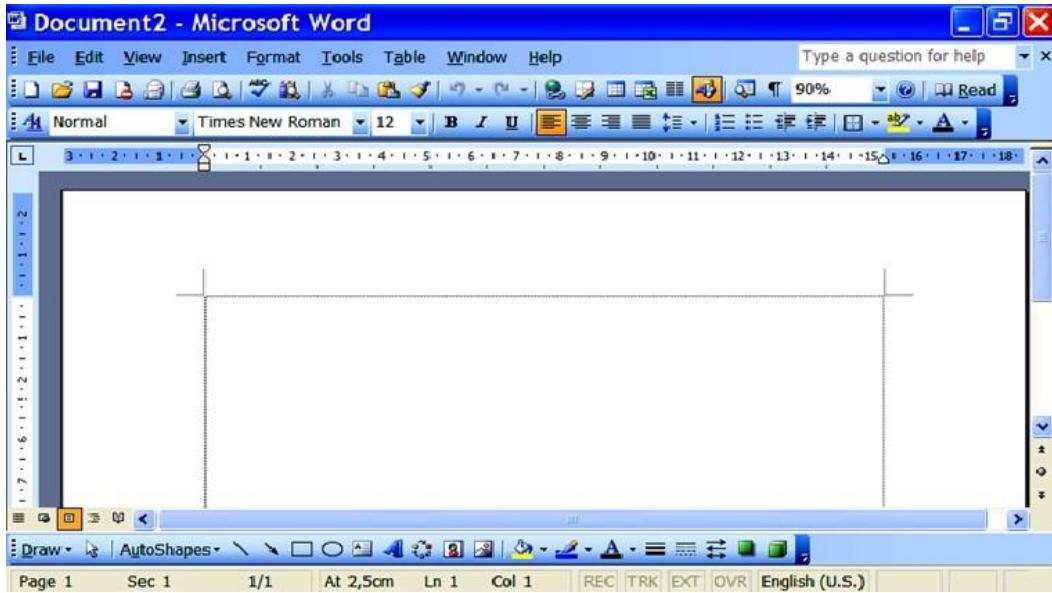


Figure 17: Elements of the menu in Microsoft Word 2003 are not placed at the edge, and therefore require more effort (since their ID is larger) than if they were placed right next to the edge.

5.2 Placing related things close

The use of Fitts' Law can go even further than that. By placing elements that are commonly used together, we are going to facilitate the work of the user. We can see an example in Figure 18, where we compare the Mac OSX scrolling bar with that of Windows.

By analyzing ID we may deduce that the Mac OSX scrolling bar may be faster to use than the Windows one. There are two reasons for this:

- The slider is larger and thus easier to reach.
- The direction buttons are together, so if we need to change the direction, we will need to perform a smaller movement of the mouse.



Figure 18: Mac OSX vs Windows scroll bars. The Mac OSX is theoretically faster because it is easier to reach (its slider is larger) and changing the scrolling direction only requires to move the mouse slightly. In the case of the Windows scroll bar, if we want to change the scrolling direction, we need to traverse the whole bar.

5.3 Complicate the access to elements

In the case of elements that are not supposed to be clicked together, or the elements that may be dangerous to confound, it may be useful to do the contrary. That is, separating those elements may avoid mistakes.

Figure 19 shows an image where two buttons that perform very different tasks are placed next to each other, and therefore the user might accidentally chose the wrong one.

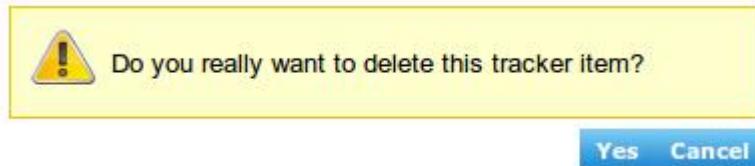


Figure 19: Buttons doing very different tasks and that are placed together may induce errors. In this case, buttons *Yes* and *Cancel* have opposite meanings. Moreover, there is not a clear separation on both, which is a second mistake. They should be placed at separate positions.

5.4 Pie menus

Another not so popular types of menus are circular and semi-circular (or pie) menus (see Figure 20). These menus appear around the contact point of the finger or the clicking point of the mouse. The distance from each menu item to the cursor is constant and very small.

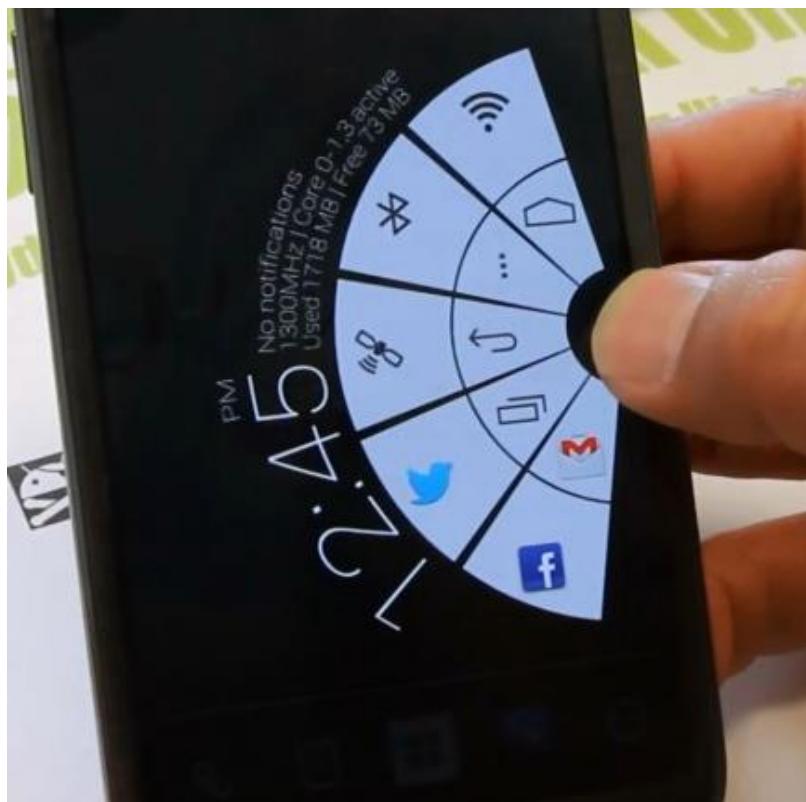


Figure 20: Pie menus rely on the fact that the items can be placed at the same distance from the pointer. When the number of items is high, several layers of menus are required.

The idea is quite interesting. However it requires some elements to make it usable:

- First, the menu has to be created on demand, on a certain position, that leaves room for the whole menu to appear. This turns this kind of menu into a contextual one, with its advantages (it may vary its contents in function of the position) and shortcomings (it will be used only by expert users).
- Second, for the menu to be practical, it may have no occlusions (the original pie menus are circular, which is something that will necessarily be partially occluded in a tactile device), and the elements must be sorted in such a way that the access is simple.

The original first pie menus were circular (see Figure 21) since they were developed for desktop systems. In such systems, the pointer (mouse) does not occlude the menu when it appears (since the menu is all around the pointer representation).



Figure 21: Circular menu with all the elements at the same distance to the pointer.

6 Law of crossing

6.1 Introduction

Apart from simply pointing or clicking, the use of stylus or analogous to stylus devices (e.g. finger) in tactile surfaces naturally leads to other quite intuitive tasks such as crossing elements (see Figure 22). Accot and Zhai ([Accot97, Accot99]) and Zhai *et al.* [Zhai2002] demonstrated that there exists the so-called *Law of crossing*. It models the time to move a cursor or a stylus across two goals. Moreover, the Law of crossing follows the same characterization than the Fitts' Law:

$$T = a + b \log_2 \left(\frac{D}{W} + 1 \right)$$

Where T is the average moving time between passing the two goals. D is the distance between the two goals and W is the width of each goal. Like in the previous case, a and b are constants to be determined.

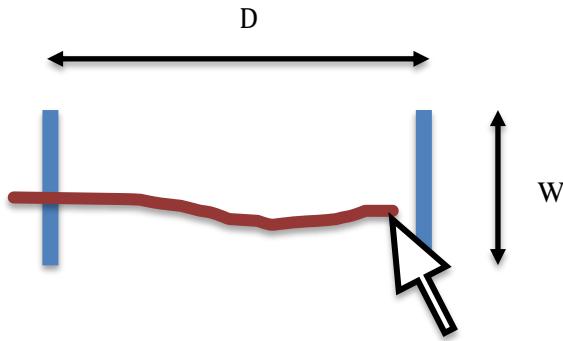


Figure 22: Crossing movement with the pointer.

The crossing action is well related to the pointing action described by Fitts' Law as we commonly interpret it in a graphical user interface with buttons. The different elements are compared in Figure 23.

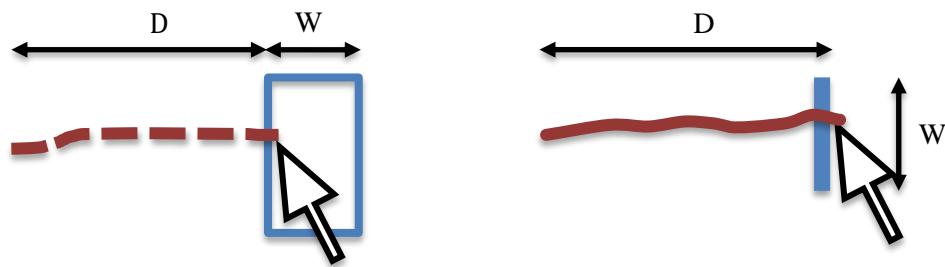


Figure 23: Pointing vs crossing: the interaction techniques differ in where the precision is required: in the direction of the movement (such as in the crossing interface – right – where the width determines how far we may move from the initial direction) or in the termination point (when pointing – left – the limit is bound by the width of the target).

The way to analyze this kind of interaction is not simple, since it can be performed in very different ways. Hence, many variables come here into play:

- **Discreteness vs continuity of the movement:** In some cases it is necessary to land the stylus before crossing the first target and lift off after crossing it, and the same for the second objective, while the interaction can also be done without lifting off the stylus (see Figure 24).
- **Direction of the targets vs direction of the movement:** Targets may be aligned in the same direction of the movement or orthogonal to it. In the first case, a line may cross many targets while in the second case, a different trace will be necessary (shown in Figure 25).

Note that, since the so-called *Law of Crossing* takes the same form as the Fitts' Law, it is important to try to isolate such constants in order to be able to determine whether crossing is better than pointing.

6.2 Crossing configurations

In contrast to pointing, the crossing technique and targets can be configured in different ways (e. g. is the target to cross orthogonal to the direction of movement?). Thus, for the sake of completeness, the performance of crossing must be evaluated in each of the configurations. The different varying variables include the direction of the movement vs the orientation of the target, and whether it is necessary or not to lift the pointer before crossing the target. We start discussing this last element.

6.2.1 Continuous vs Discrete

The implementation of the technique may be so that the pointer (stylus, finger...) can move in contact to the screen or it may be required to lift it up before crossing. These two configurations are called continuous and discrete, respectively, and are depicted in Figure 21.

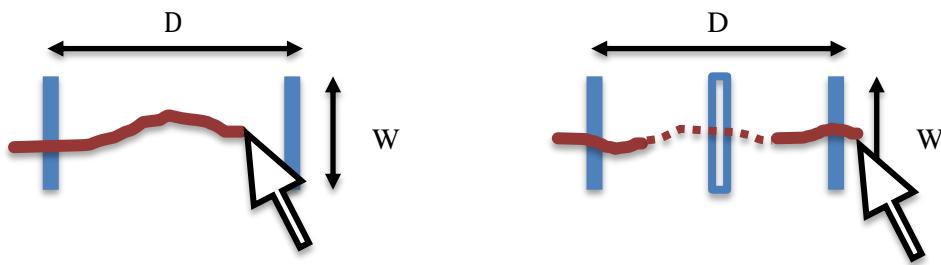


Figure 24: Continuous crossing (left) vs Discrete Crossing (right) interaction.

Intuitively, discrete is more costly than continuous, but the studies show that this is only more time consuming when the distance between the targets (and thus the obstacle) is small.

6.2.2 Collinear vs Orthogonal

The targets, both in pointing and in crossing, can have their width aligned or not with the direction of the movement. When the width W is orthogonal to the direction of movement, we have the configuration on the left in Figure 22, where the movement seems more simple and continuous, while when the objectives are not aligned with the direction of the movement, the pointer must trace a slightly more complicated path. This is also applicable to crossing targets with obstacles (discrete crossing).

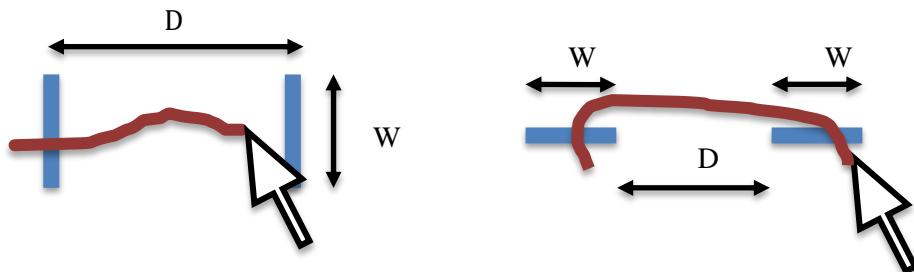


Figure 25: Collinear crossing (left) vs Orthogonal Crossing (right) interaction.

In order to analyze the performance of crossing, it is necessary to perform a factorial combination of the different configurations, that is, one should test continuous traces with orthogonal crossing, continuous tracing with targets in the same direction of the movement plus discrete tracing with these two configurations of the targets.

The studies by Apitz *et al.* [Apitz2010] compare the different configurations of the more classical pointing technique: collinear and orthogonal, with all the different configurations of the crossing method: the collinear discrete, collinear continuous, orthogonal discrete and orthogonal continuous. And the experiments are carried out for several different target sizes and distances. The different configuration scenarios are analyzed in relation to *ID* and error rate. The conclusions the researchers reach are, very briefly:

- Crossing-based interfaces achieve similar (or faster) times than pointing.
- The error rate in crossing is smaller than in pointing.
- Discrete crossing becomes more difficult if the distance between the targets is small.
- Crossing (especially continuous) seems superior than pointing for *ID* values > 5.

7 Law of Steering

7.1 Introduction

Another commonly necessary task for the interaction with modern user interfaces is the creation of trajectories. These tasks are useful for navigating through nested menus, 3D navigation, dragging elements, and other drawing tasks.

Very often, such tasks are directional movements with a lateral constraint (see Figure 26).

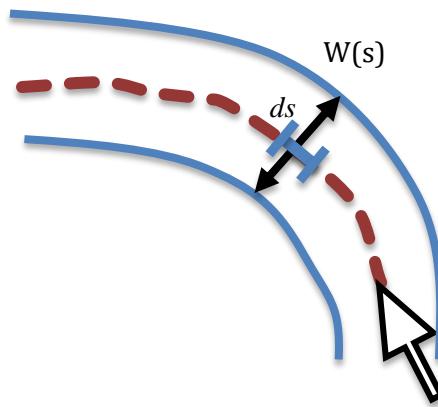


Figure 26: Generalized path steering with amplitude constraint.

Accot and Zhai found that the performance of manual steering tasks (T_s) for a generalized path C (see Figure 26) can be expressed in the following integral form:

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

Where T is the time to successfully steer through the path C , $W(s)$ is the path width at s . With this formulation, the Index of Difficulty of the steering task is

$$ID_s = \int_C \frac{ds}{W(s)}$$

Accot and Zhai arrived to this formulation by transforming the steering task into an accumulation of an infinite number of goal-crossing tasks ([Accot97]). The time to cross a goal of width W at distance D follows the Fitts' Law equations. This way, by transforming the goal crossing task into a succession of goal crossings along a trajectory, they can calculate the summed up index of difficulty of N consecutive goals as:

$$ID_N = N \log_2 \left(\frac{D}{NW} + 1 \right)$$

Taking N to infinity yields the equation of ID_s .

7.2 Steering through straight paths

Using these results, we can calculate the time to steer through a straight path. The straight path can be determined using the D and W variables for the distance to trace and the width of the path, respectively (see Figure 27).

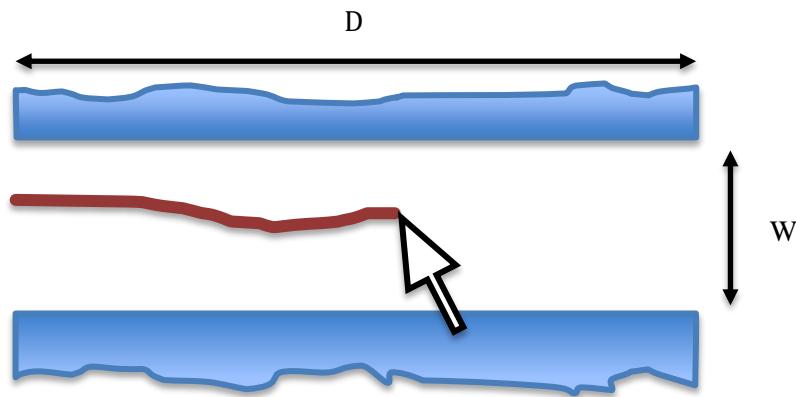


Figure 27: Path movement in a straight path with amplitude constraint.

For a straight path, the time required to successfully steer through it would be:

$$T_p = a + b \frac{D}{W}$$

Where W is the path width, and D the path length. The index of difficulty of the task will be then:

$$ID_p = \frac{D}{W}$$

The previous equation also applies to circular paths with constant width.

7.3 Evaluations of Steering Law

Accot and Zhai [Accot97] showed steering law works for paths of different shapes: a cone, a spiral shape, or a circle, and that it also works for different input control devices. The upper limit on the application of this law is the human body limitations: This means that we can change configuration parameters so that the theoretical speed can be increased, but if this exceeds human body capacities, the results saturate.

One of the direct applications of steering law is the design of nested menus. Since accessing an item of a submenu is addressed by performing a path that follows an ‘L’ shape, this path can be subdivided in two different steering paths, and therefore the access time can be calculated by directly applying the steering law. As a result, different menu configurations (e. g. classical nested menu vs pie or hierarchical pie menu) can be evaluated using this law.

A more complex interaction experiment was also developed by Zhai *et al.* [Zhai2004] to evaluate locomotion in a VR setup of 160 degrees of field of view. The users had to navigate through two types of path, one straight and the other circular. Graphics were rendered in real time in response to users’ actions. The path was 1885 meters long. The users were instructed to drive as quickly as possible without going out of the path boundary.

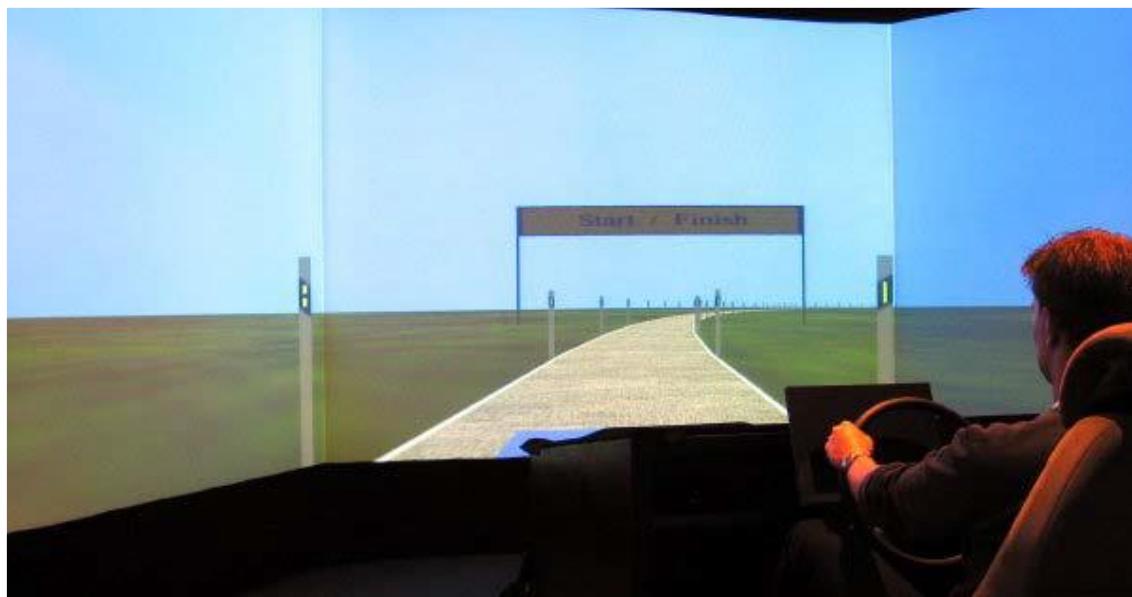


Figure 28: The Virtual Reality simulator setup used in the steering experiment.

The user’s behavior was measured every tenth of a second throughout the experiment. The recorded variables were: vehicle’s X and Y coordinates, heading, gas pedal amplitude, brake pedal amplitude, steering wheel amplitude, speed, deviation from middle line, and collisions.

The users had a representation of the car also projected on the screen, to help them have references of their relative position with respect to the road. Some examples of the images seen by the users throughout the driving experiment are shown in Figures 28 and 29.

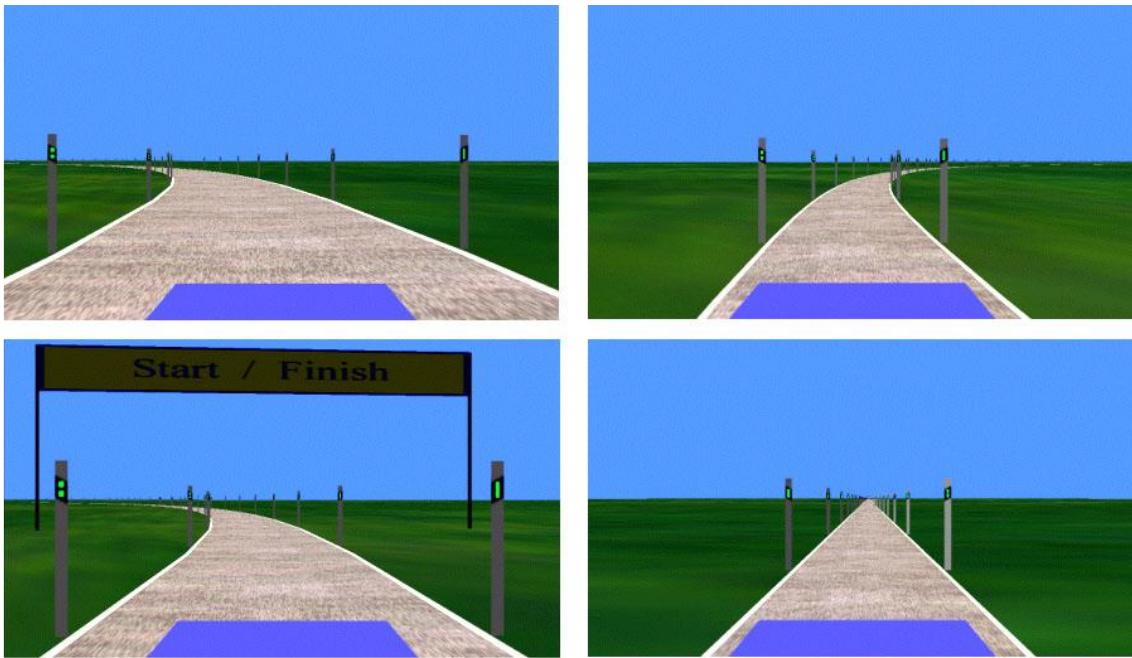


Figure 29: The images projected in the simulator, as seen by the users.

The results show that there is higher variance between users than in other experiments where no so complex tasks are performed. For example, some users used the total power of the gas pedal, unlike in the pilot experiment. The results of those users were eliminated from the analysis. Otherwise the results would not be comparable.

Results show that the steering law behaves well for locomotion, although with a lower dependency on the path width, since driving involves stronger dynamics than simple gestures such as hand drawing. In any case, the authors found that there is a linear relationship between the trial completion times versus the index of difficulty of the path, with a high fitness value.

8 Accelerating target acquisition

8.1 Introduction

Besides the uses of Fitts' Law in the design of User Interfaces, commonly the screen space is restricted enough to avoid decisions that imply the use of larger targets or cluttered screens to accelerate target acquisition. This, together with an improvement in the computation power, has led to the use of techniques that modify such parameters dynamically.

Commonly, the modifications that are performed are the same than we have seen for static UIs: amplitude reduction and increase of target size. However, we may change these parameters a non-constant value. Moreover, we may add to this the modification of the speed of the pointer, the so-called dynamic control-display ratio change.

8.2 Modification of Target Width

In order to reach and manipulate virtual objects, applications provide a pointer together with the virtual representations of the elements to manage.

8.2.1 Target and screen size

As stated by the Fitts' Law, the size of the target we are trying to reach has a great influence on the time required to select it. Actually, the cost or effort to reach an object depends on the amplitude of the movement and the size of the target. The larger the target, the easier it is to select, but the longer the distance to cover, the more difficult it is to reach.

There is a tight relation between screen size and target size, the larger the screen, the larger the targets can be. However, since the amount of information we want to include in our windows is also relatively large, we have to balance between information and controls.

Although nowadays screen resolutions often leave enough room for the design of large target icons, the fact that screens are large also make that the distance the user has to cover can also be large. Therefore, according to Fitts' Law, the cost of reaching the element also increases with the distance.

On the other hand, the resolutions of portable devices, although being rather high (such as 900x600 pixels), the size is small (around 4" on a modern smartphone), and therefore, the number of pixels that has to be devoted to a single icon is relatively high, thus making the screen *effectively smaller*. There is a second issue: The fact that the pointer in most modern screens is a finger. Hence, its size relative to the size of the screen makes the display also *effectively smaller*.

With respect to Fitts' law, there are two simple ways to reduce the difficulty of a pointing task: enlarging the target or moving it closer to the cursor. Both have been explored in several ways. These are also coupled with the velocity of the cursor. We will deal with this issue, named Control-Display ratio later. Now, we introduce some techniques that try to reduce interaction times by manipulating the target sizes dynamically.

8.2.2 Expanding targets

One approach to easing acquisition of small targets consists in enlarging the target size when the cursor approaches the element to be selected. This technique is called expanding targets [McGuffin2002] and has been implemented in different ways. Mac OSX operative system uses this technique on the icon panel that serves as program launcher, also known as Dock, shown in Figure 30.



Figure 30: Mac OSX Dock with its size-enlargement and position-changing icons.

Note that this is a mix on two techniques: target size increase and moving target. The system changes the size and moves the position of the elements in the Dock in relation to the position of the cursor. Therefore, the *selectable* area is smaller than it seems to be, because when approaching the center of the icon (in a horizontal move), the icon itself moves towards the opposite direction of the cursor movement. This has been reported to cause frustration on the users because the expansion induces these movements that may be not totally predictable if the cursor approaches the Dock in a perpendicular direction. This can be alleviated if the system allows the enlarged icons to overlap over the neighbors. A second method that reduces frustration is expanding the icons only when the cursor velocity decreases on final target approach.

8.2.3 Bubble targets

Other not so successful methods for target expansion have been tested, such as the *bubble targets* [Cockburn2003]. These add a selection region around the target with a bubble shape (Figure 31).

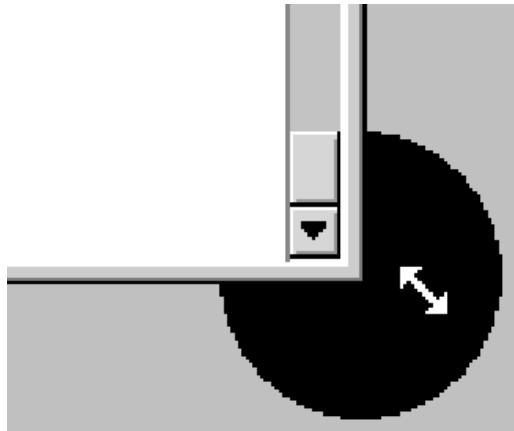


Figure 31: Bubble-based targets for easier selection.

The bubbles only appear when the mouse is pretty close to the selectable point. This technique improves the speed of selection of small targets, but some users have reported distraction due to the appearing of the bubbles.

8.3. Increasing cursor size

Some authors address the same problem by changing the area that is affected by the cursor, instead of modifying the target size. This leads to an effective reduction of amplitude movement. But it can also be seen as a magnification of the target size.

Grossman and Balakrishnan [Grossman2005] increase the size of the cursor in a circular region that embraces the closer target for a more comfortable and easy selection. They call this technique the *Bubble Cursor*. The cursor's size is dynamically adjusted so that only a single target is enclosed by it. Moreover, its shape also is modified to envelop the closer target when it is not completely inside the main cursor bubble. In Figure 32 we can see a sketch of such technique.

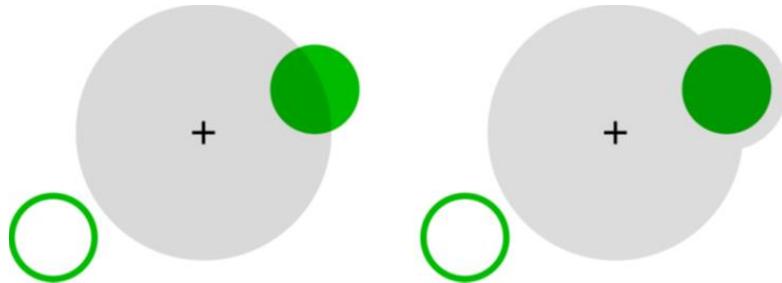


Figure 32: Bubble cursor adapting its size to cover the closer target.

Their implementation is quite straightforward, since they divide the space of targets using a Voronoi diagram where each region contains a single target and the effective width of the cursor is modified to envelope the target contained in that region (see Figure 33).

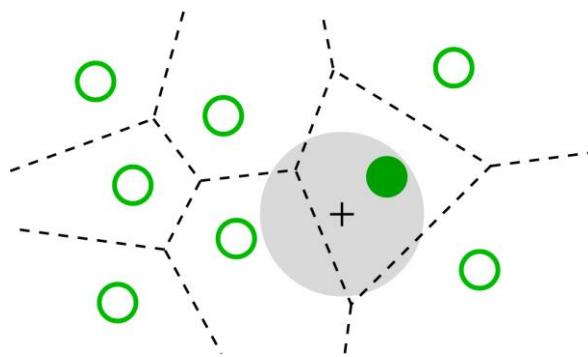


Figure 33: Voronoi subdivision of the target space. When the cursor enters in one of the regions of the target Voronoi map, it may adapt its size to envelop only the target of interest.

This technique has been proven very useful and fast. However, the experiments were performed using a zero mouse acceleration value. Chapuis *et al.* [Chapuis2009] improve on this technique by taking into account also the speed of the mouse. The technique they develop is called *DynaSpot*. In this case, since the area of the cursor grows with the speed, it is likely that the cursor area will overlap several targets at the same time. However, only the closer to the cursor center is the one candidate to selection. We can see in Figure 34 how the area is changed according to the speed of the cursor.

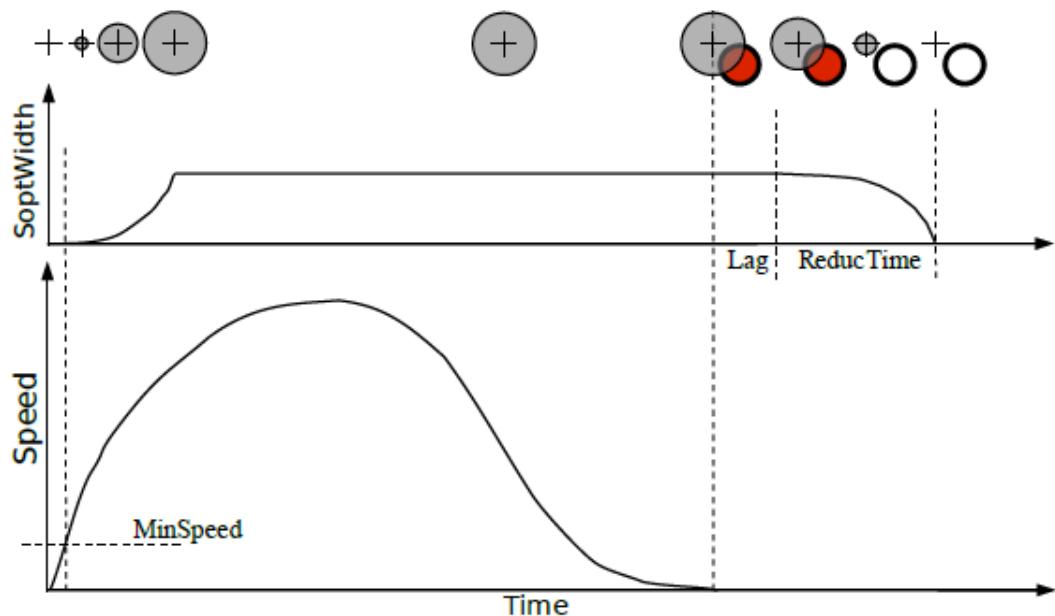


Figure 34: Area change of the cursor as a function of the mouse speed.

Like in the previous case, visual cues are provided to inform the user which are the targets that are going to be selected as well as the current size of the area cursor. This is shown in Figure 35.

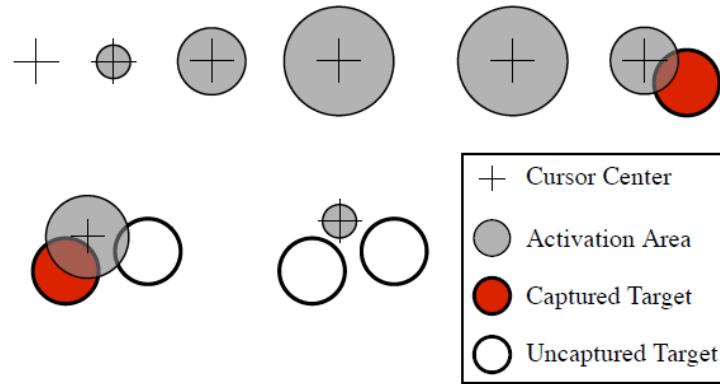


Figure 35: Indication of the target to be selected (the closer to the center of the cursor).

Chapuis *et al.* [Chapuis2009] also compare this method with the Bubble Cursor and find that this new method obtains better results both in acquisition time and reduction of number of errors.

8.4 Target moving and other techniques

8.4.1 Target moving

Besides enlarging targets, there is another popular possibility that enhances selection and consists in moving the targets to the cursor.

The method used by the Mac OSX Dock combines target enlargement and target movement, however the movement is relatively small compared to other techniques.

Probably the most widely used direct application of target movement is contextual pop-up menus. Contextual menus are the ones that appear at the cursor location after some user action (such as selection and/or right click). This makes the distances to the items to be minimal, as compared to visiting the classical top menus. Moreover, as compared to the classical pop-down menus, the contextual menus can adapt their contents to the previous action: For instance, if the user selected a word, several options related to word formatting can appear, which may differ on the possible actions if no region or word is selected.

Another not so popular types of menus are circular and semi-circular (or pie) menus (see Figure 36). These menus appear around the contact point of the finger or the clicking point of the mouse. The distance from each menu item to the cursor is constant and very small.



Figure 36: Semi-circular menu on Android's Honeycomb version of the browser.

8.4.2 Other techniques

Other experiments have shown that the selection can also be helped by the *sticky targets* technique. The main idea is to convert the selectable points as *attractors* of the cursor. Therefore, when the cursor is close to the selectable point, it moves to it. This method improves efficiency and has been implemented both in 2D and 3D user interfaces. Experiments have shown that users adapted easily to this technique and were preferred to other methods such as the bubble targets.

This technique has also been implemented in 3D virtual environments by Andújar and Argelaguet [Andujar2007], by modifying the selection ray direction to point to the target it is getting close (see Figure 37).

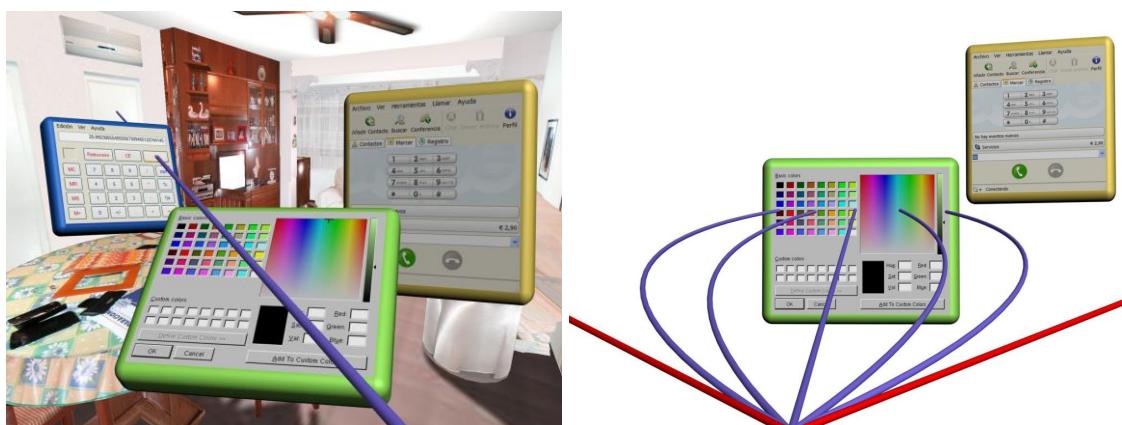


Figure 37: Ray direction modification in a 3D Virtual Reality environment to point to the closer target.

8.5 Discussion

Comparisons with other techniques show that target resizing facilitates pointing even if the expansion is late and unpredictable. The main problem of such techniques is that in order to expand a target if other targets surround it, those must be shrunk. As a consequence, some experiments showed that no performance improvement is obtained for systems like the Mac OSX Dock. More generally, techniques that dynamically change the screen layout cause a spatial disorganization that may limit their expected benefits.

In other environments, where the scene may be cluttered, and a lot of selectable targets may be placed in a small region of space, such as in many 3D virtual reality scenarios, a combination of techniques may be useful. Argelaguet and Andújar [Argelaguet2008] combine expanding targets with clipping away other closer candidate targets with a technique they call *forced discocclusion*.

8.6 Control Display Ratio

8.6.1 Concept

Control Display ratio is the relation between the amplitude of movements of the user's real hand and the amplitude of movements of the virtual cursor. More specifically, the Control Display Ratio usually refers to the distance the mouse has to cover in the physical world to move the pointer on the screen by a given distance. These are usually measured in meters (physical move) and pixels (cursor movement).

Obviously, any control system must translate physical displacements of the interaction devices to virtual movements of the cursor. The way this translation is implemented may affect the performance of the user.

8.6.2 Control-Display Ratio adaptation techniques

There is no clear idea on how the definition of C-D ratios affects our perception of the virtual world and therefore improves productivity by reducing selection times. There are three typical C-D ratio configurations:

- Constant
- Dependent on mouse speed
- Dependent on the cursor position

The idea behind the *mouse acceleration* is to provide easier ways to access further targets. The cursor moves by a larger distance when the mouse covers a given amplitude more quickly, capturing an intention: when users move the physical device quickly, they typically wish to go further, so the cursor can be displaced even faster to cover the distance more quickly.

An opposite effect can be *mouse slowing* when the cursor is near a target: covering the same number of pixels requires moving the mouse by a longer displacement. The idea here is to improve pointing precision.

Although some studies conclude that nonlinear mappings are not intuitive enough to provide positive improvements for pointer movements, other studies have shown increase

in performance for 3D navigation, 3D rotations and other pointing problems. Therefore, there are no definitive results.

Compared to the previous approaches that address target magnification, C-D ratio adaptation can also be interpreted as dynamic magnification of the physical motor space where the mouse movements take place. In this sense, these techniques are related to zoomable interfaces that use a local or global magnification of the visual space.

Pointer acceleration is the default behavior on the Microsoft Windows, Linux, and Apple Mac OS X operating systems. The implementation in those systems dynamically changes the C-D ratio the following way: When the speed of the control device is high, CD gain is high (typically well above 1) and when the control device moves slowly, the CD gain is low (in some cases less than 1). In some Operative Systems, this behavior can be configured, as it is also related to other accessibility options (e. g. large fonts and icons for people with vision problems...).

9 Pointing tasks

Complex information displays generate the necessity of pointing and selecting items. It is a fundamental and frequent task in graphical user interfaces, so even a marginal improvement in pointing performance can have a large effect on a user's productivity. This direct-manipulation approach is attractive because it prevents the users from learning commands. It has other advantages such as reducing the chance of typos and keeps the attention of users on the display. As a result, pointing and selecting often results in faster interaction, fewer errors, easier learning, and higher satisfaction.

Pointing, however, does not come without particularities. Depending on the display we want to use, different devices or techniques may be more convenient. One of the first elements we must consider when analyzing the properly pointing technique and device is the goal. We may find six types of interaction tasks in literature:

- **Selection:** Users must choose one or more than one element from a set of items. This technique is used for the simple task of selecting an item in a menu, to more elaborate tasks such as the selection of a part in a CAD design.
- **Position:** Positioning is another fundamental task in sketching, drawing, or CAD/CAM applications. Users must choose a point in one to three dimensions.
- **Orientation:** Like in the previous case, designing software often requires determining orientations for the new elements to be added or to modify certain aspects of the scene.
- **Path creation:** The path must be built from a set of positions and orientations or in a continuous way.
- **Quantify:** The selection of a quantitative value is often implemented as a one-dimension selection task.
- **Text:** Adding, deleting, or modifying text is also a task accessible through selection and pointing processes. Text must be added simply, but other operations such as text font and size, or page layout also fall into this category.

Although many of these tasks can also be implemented with a keyboard by typing numbers or letters to determine position, orientation, and so on, it is often far less efficient than with a direct manipulation tool. Expert users, however, may use a combination of direct pointing and keyboard shortcuts (such as *Ctrl+C*) to further accelerate the processing.

9.1 Pointing devices

There is a great amount of pointing devices and its number has increased constantly for several years. We may classify the pointing devices in two families:

- Direct-control devices
- Indirect-control devices

Direct-control devices are those that work directly on the surface of the display, such as the stylus or our fingers on a capacitive screen.

Indirect-control devices work away from the surface, such as the mouse, joystick, graphics tablet, and so on.

9.1.1 Direct-control devices

Direct-control devices have existed for a long time. One of the earliest devices is the *lightpen*, which enabled users to point to a spot on a screen and press a button to perform a selection operation.

The *lightpen* dates from back the middle seventies (you can see an example in Figure 38), as it was a device that already worked on UNIVAC 1652 in 1976. Compared to today's touch screens, the *lightpen* was heavy and fragile. Its operation caused fatigue and the hands over the screen obscured part of the information.

Although today's designs still produce higher fatigue than the mouse, a proper design, such as with a surface on which to rest the arm, greatly alleviates the problems of the *lightpen* and older screens.

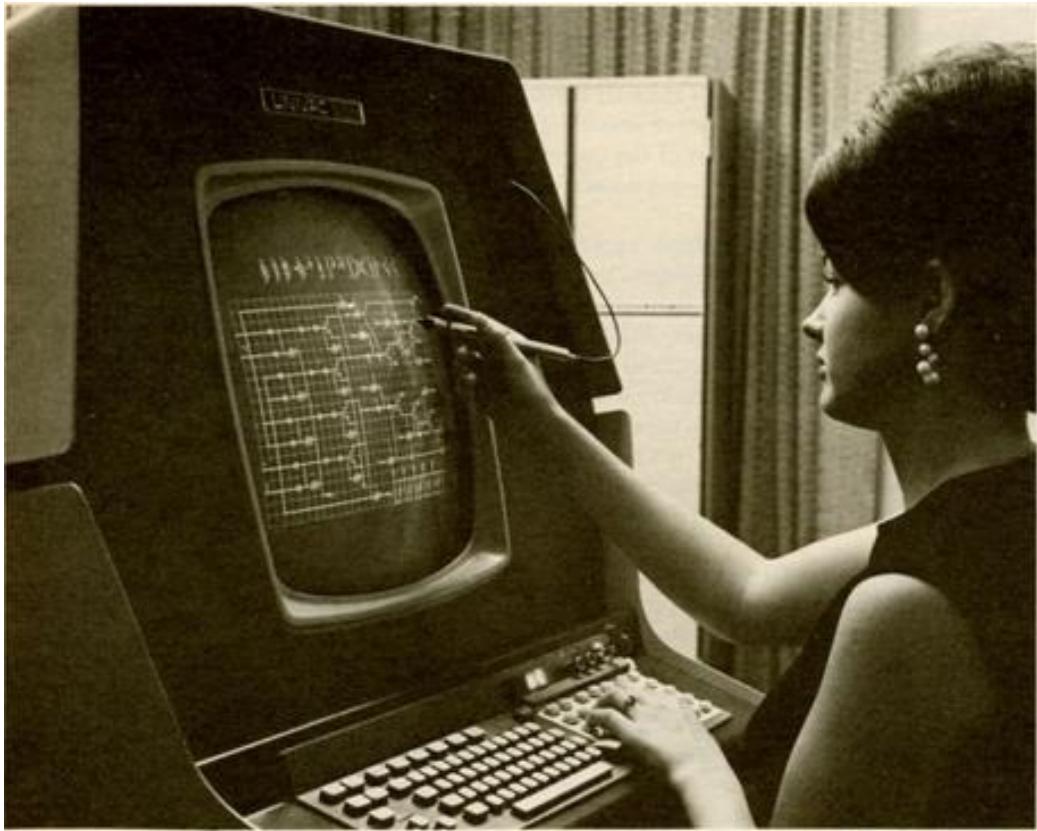


Figure 38: Univac's interaction with a lightpen (around 1976).

Direct-control devices have other problems, still present in mobile phones, such as the imprecise pointer (in this case it is mainly related to the quality of the screen), and the ability of the software to accept the touch immediately (also called *land-on* strategy). This has some advantages, such as the fast response, but also some problems, such as the inability to correct the mistakes from the user. This may be somewhat painful for iPhone users. Since the unlock screen does not require a confirmation of the entry code, as can be seen in Figure 39, the user may be wrong, let's say, when keying the last digit, and then, the iPhone tries to unlock with a wrong code. Since the user may be aware on his or her mistake, it is frustrating not to be able to correct the input, and this may lead to completely lock the device if the same mistake is committed three times.

For some tasks, touch screens use the *lift-off* strategy. It has three steps: When the user taps on the screen, a cursor appears that can be dragged (without removing the finger from the screen) to finely adjust the correct position. When the user is satisfied with the cursor position, they lift the fingers off the display to activate. This is the technique used in the virtual keyboard of iOS system (see Figure 40) for accentuated or other special characters. In this concrete case, the new characters appear right on top of the finger's position, and the user moves horizontally over them to activate the desired character.



Figure 39: Unlock screen of the iOS devices. The lack of confirmation button is error-prone, since the user has no time to correct the input even if he or she detects that the last key was wrongly pressed.



Figure 40: Lift-off strategy for introducing accented letters in the iOS operating system. The user selects the letter, and leaves the finger on, until the submenu appears with the differently accented versions of the selected letter.

Touch screens are often integrated into applications directed at novice users in which the keyboard can be removed completely. They have some advantages in public-access systems. Because they can be implemented with no moving parts, their durability in high-use

environments is good (some people say that the touchscreens are the only input device that has survived at Walt Disney World theme parks).

Multi-touch screens allow a single user to perform multiple finger entry, such as pinch-to-zoom gestures, or multiple users to work on the same application together.

Although some people may prefer finger input on touch screens, some tasks are better addressed with a pen. Pens are familiar and comfortable for users, and facilitate the movement of the cursor while leaving almost the whole context in view. The use of a pencil has also some shortcomings. The most clear is probably the additional work to pick up and put down the stylus.

9.1.2 Indirect-control devices

Indirect pointing devices alleviate hand-fatigue and eliminate the problems of hand obscuring the screen. On the other hand, they require the hand to locate the device and demand more cognitive processing and hand/eye coordination to bring the cursor to the desired target.

One of the most popular devices is the mouse: It is cost-effective, allows for a comfortable position of the hand, its buttons are easy to press, and positioning can be done quickly and with precision. However, for long movements, users must pick up and replace the mouse position.

Another popular device is the trackball, which resembles an upside-down mouse. The ball is used to move the cursor in the screen as it is moved. Joysticks have been long used in games. They are appealing for tracking purposes because they only need relatively small movements to move a cursor.

Touchpads have become very popular lately due to their presence in portable computers (and its decrease in price). They offer the convenience and precision of a tactile screen while keeping the user's hand off the line of sight. Like in the case of touch screens, their lack of moving parts is usually an advantage.

The graphics tablet is a touch-sensitive surface that is separated from the screen. It is often used lying flat on the table and can be operated with a finger, pencil, stylus, and so on, and usually provides input possibilities not only by position, but also by pressure. This last feature, almost unique in touchscreens, is highly appreciated by artists. Like the mouse, when resting in a flat surface, may be used for long periods without producing fatigue.

9.2 Comparison of pointing devices

When analyzing pointing devices, there are two major variables to take into account: speed and accuracy. Some studies have found that direct pointing devices are often faster but more prone to errors than indirect control devices.

For decades, the mouse has shown its superiority to other devices in speed and accuracy. For instance, the mouse is faster than the *trackpoint* due to tremors in finger motion during fine finger movements.

When analyzing two devices we must take into account the task. For scrolling large lists or large documents or webpages, mice equipped with a scroll wheel may be convenient, although some studies showed that they are not superior to regular mice.

The usual belief is that pointing devices are faster than keyboard cursor keys, but depending on the task, these may be superior due to the smaller movement required to use them, and the fact that some shortcuts, if mastered, are very convenient for document edition. Usually, short tasks that mix typing and pointing are faster addressed with cursor keys.

Users with motor disabilities may prefer devices that are fixed, such as joysticks or trackballs.

10 3D Selection

10.1 Introduction

Several designers believe that 3D interfaces can make several tasks easier than classical 2D systems. They believe that this would allow a behavior that is similar to real life, and therefore, tasks can be accomplished more efficiently. In Figure 41 we see an example of 3D environment being used by two people at the same time.



Figure 41: An inspection session in a 3D virtual environment. Both the users can see in 3D but the view is calculated accurately for the user whose googles are tracked (left).

Besides that, some people are also studying whether enhanced interfaces may even be better than reality. Of course, the term *better* may have different meanings, but the idea is that some scenarios can be performed using a virtual reality environment in a more effective way than in the 3D world. There is some controversy, because the tasks a 3D

environment may be more suitable or even superior to *reality* need to be computer tasks, such as computer-assisted design, molecular modelling, and so on. In these tasks, we can provide the user with super-natural powers such as travelling back in time to undo some of the actions.

10.2 Definitions

Over the last decade, the field of 3D user interfaces has grown out of its infancy, forming the basis for many game and industry applications. There are some terms that are important to get familiar with:

- **3D interaction:** An interaction between a person and a computer in which the user's tasks are carried out in a 3D spatial context using directly 3D input devices or 2D input devices with direct mappings to 3D.
- **3D user interface:** A User Interface that involves 3D interaction.
- **3D interaction technique:** The technique designed for solving a certain task. This may involve both the use of hardware (a device) and software (a module that has as input the device signals and that implements the behavior in the virtual space).

Note that none of those definitions implies the use of a 3D environment in the sense of Virtual Reality (i. e. stereo viewing as a minimum). Although many 3D interfaces are commonly linked to 3D environments, we may implement those in a desktop system without stereoscopy.

There are many devices tailored to produce 3D stereo and many devices used to interact with 3D interfaces. We will not visit them, since this is beyond our scope. We will only address one of the problems that arises in 3D environments and whose solution is complex: 3D selection.

10.3 3D selection

We refer to 3D selection as the selection task when it is carried out in a 3D immersive environment. Compared to other selection methods, new problems appear, such as the occlusion of the target.

7.3.1 3D Selection techniques

Commonly, 3D object selection techniques are implemented with a selection tool such as a data glove or a Wanda device. Since we need to select in a 3D environment, we need to define a 3D position, which is often given by a ray, a 3D cursor or a simple 3D shape such as a sphere. The computer must perform a 3D intersection or proximity test with the environment. In contrast to the *selection tool*, the method used for effectively selecting the elements is usually called *selection technique*.

The selection technique defines how the user will control the selection element. In typical VR environments tracking devices will usually perform the monitoring of the users' actions.

Selection in 3D Virtual Reality environments can be accomplished using different techniques. However, the two main paradigms are:

- **Hand extension techniques:** These techniques are also called 3D point cursors, since they represent a 3D point in space as a mapping of the user's hand position (see Figure 42).
- **Ray-based techniques:** Instead of using the hand position, these techniques use the position and some other element to indicate an orientation to generate a ray in space that is used as a pointer. These techniques are also called aperture-based selection techniques or ray cursors. The different configurations can be seen in Figure 43.



Figure 42: The representation of a hand in a VR driving simulator as a mapping of its real position and orientation.

Previous research in VR has demonstrated that ray-based techniques are often superior to hand extension techniques due to the faster selection times.

10.3.2 Ray-based selection and pointing

In order to define a ray, we need an initial position, and an orientation. Usually, the user controls the 3D cursor by moving its dominant hand: the control is performed using the hand's position as the initial point, and the direction of the ray is calculated using the users' wrist orientation. Sometimes, the virtual ray starts from the head of the observer and that goes along the hand position.

For selecting single objects or points in scenes, ray cursor techniques have an inherent problem. Since the ray is a 2D geometric object, the ray often intersects multiple objects, and so the actual target of interest is ambiguous. Moreover, selecting occluded elements, if needed, may become a problem, because the occluding objects must be erased, moved, or visually removed in an easy way. A second possibility is to define a selection point on the pointing ray. Then, some mechanism has to be provided to move the cursor along the ray.

This is also problematic because the hand can move (especially if we also use it to operate a wheel or a joystick that performs the 3D cursor movement on the ray) and the desired direction can change. One possibility is to lock the ray after the direction has been chosen appropriately (a technique is called *lock ray*) and then allow for the movement of the 3D cursor on the ray. The three methods are illustrated in Figure 43.

There is a second problem that arises when using rays defined by the hand as their initial position: the ray direction and the viewing direction is not the same. As a consequence, some object that is visible from the user's viewpoint can be unreachable from the user's hand position. This is not easy to address, since many computations have to be done on real time if we want the application to detect this problem. Some elements that may alleviate this issue consist of using elements such as the sticky targets in 3D [Andujar2007] (Figure 44) and enlarging the pointed elements in a similar way to the bubble controls [Argelaguet2008] (see Figure 45), or to locally flatten the elements to facilitate pointing.

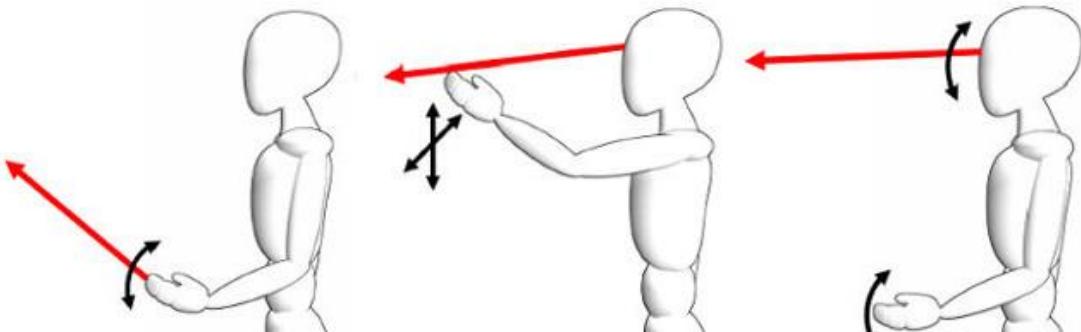


Figure 43: The three typical paradigms of ray pointing in VR: Left shows the interaction being controlled by the hand position and wrist orientation. Center shows the ray being started at the eye, and the intersection with the hand creates the ray direction. Right shows the ray starting in the eye, while the direction is controlled by the wrist orientation.

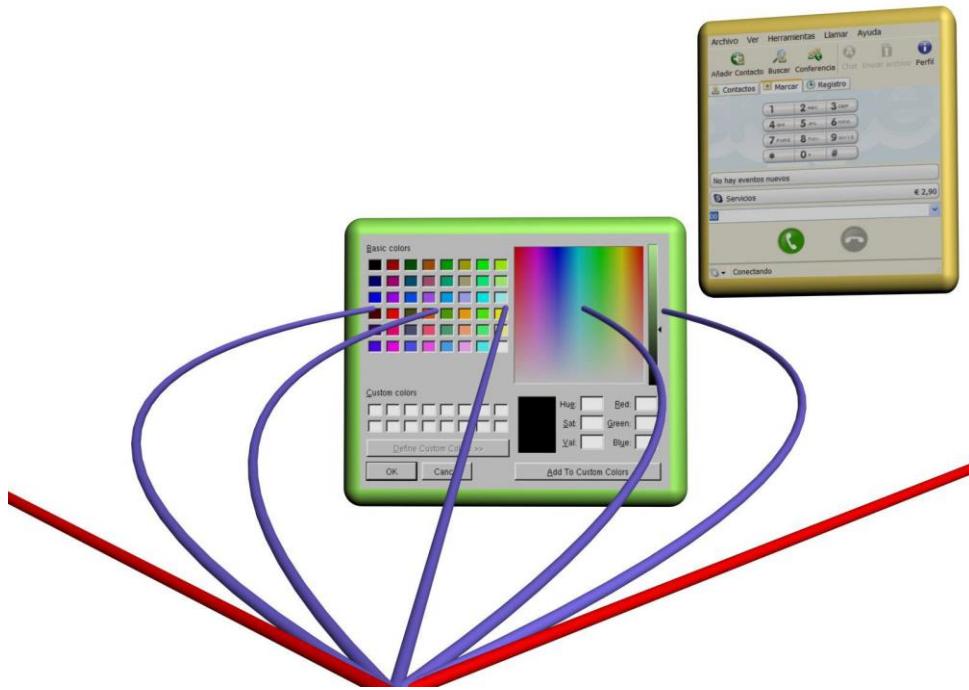


Figure 44: Implementation of sticky targets for ray attraction in a 3D environment for the help in menu selection.

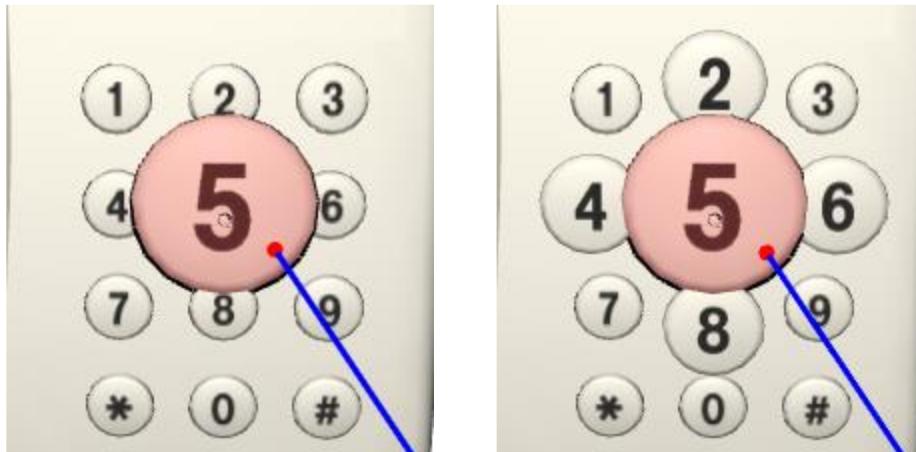


Figure 45: Expanding targets in a 3D environment [Argelaguet2008]. Different configurations are possible, such as only expanding a single target, the one closer to the user (left), or an expansion of all closer selectable targets (right).

11 Bibliography

- [Accot97] Accot, J. & Zhai S. (1997) *Beyond Fitts' law: models for trajectory-based HCI tasks.* In Proceedings of ACM CHI'97 Conference on Human Factors in Computing Systems, pages 295–302.
- [Accot99] Accot, J. & Zhai. (1999), Performance evaluation of input devices in trajectory-based tasks: an application of the steering law. *CHI '99 Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pp. 466-472.
- [Accot2002] Accot, J., & Zhai, S. (2002). *More than dotting the i's - foundations for crossing-based interfaces.* Proc. CHI: ACM Conference on Human Factors in Computing Systems, Minneapolis, Minnesota, 73 - 80.
- [Allison2004] Allison, B., Desai, A., Murphy, and Sarvary, R.M., (2004) Choice reaction time in card sorting tasks: validation of the Hick-Hyman Law, San Jose State University ISE 212, Summer 2004.
- [Argelaguet2008] Argelaguet, F., Andújar, C. (2008). Improving 3D Selection in Immersive Environments through Expanding Targets. In *Proceedings of the 8th International Symposium on Smart Graphics (SG '08)*. pp:45–57.
- [Andujar2007] Andújar, C., Argelaguet, F. (2007). Anisomorphic Ray-Casting Manipulation for Interacting with 2D GUIs. *Computers & Graphics*. 31:15–25.
- [Apitz2010] Apitz, G., Guimbretière, F., Zhai, S. (2010). Foundations for designing and evaluating user interfaces based on the crossing paradigm. *ACM Transactions Computer-Human Interaction*, 17(2).
- [Balakrishnan97] Balakrishnan, R. and MacKenzie, I.S. (1997). Performance differences in the fingers, wrist, and forearm in computer input control. *CHI ' 97: ACM Conference on Human Factors in Computing Systems*. p. 303-310.
- [Bi2013] Bi, X., Li, Y., & Zhai, S. (2013). FFitts law: Modeling finger touch with Fitts' law. In *Proceedings of the 2013 ACM annual conference on Human factors in computing systems*, pp. 1363-1372.
- [Card78] Card, S. K., English, W. K., & Burr, B. J. (1978). Evaluation of mouse, rate-controlled isometric joystick, step keys, and text keys for text selection on a CRT. *Ergonomics*, 21, 601-613.
- [Card83] Card, S.K., Moran, T.P., & Newell, A. (1983). *The Psychology of Human-Computer Interaction*. Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.
- [Chapuis2009] Chapuis, O., Labrune, J.-B., and Pietriga, E. (2009). DynaSpot: Speed-dependent area cursor. *Proc. CHI '09: ACM Conference on Human Factors in Computing Systems*. New York: ACM Press, 1391–1400.

[Chapuis2011] O. Chapuis, and P. Dragicevic, (2011) Effects of motor scale, visual scale, and quantization on small target acquisition difficulty. *ACM Transactions on Computer-Human Interaction (TOCHI)*, Volume 18 Issue 3, July 2011, pp. 13-32, ACM New York.

[Cockburn2003] Cockburn, A. and Firth, A. (2003). Improving the acquisition of small targets. *British Human Computer Interaction Conference*. p. 181-196.

[Cockburn2007] Cockburn, A., Gutwin, C. and Greenberg, S. (2007). A Predictive Model of Menu Performance. Proceedings of CHI'07: ACM Conference on Human Factors in Computing Systems, San Jose, CA, 627-636. ACM Press.

[Cockburn2008] Cockburn, A., Gutwin, C. (2008). A Predictive Model of Human Performance with Scrolling and Hierarchical Lists. In *Human Computer Interaction*, vol. 24 no. 3, 273-314.

[Crossman83] Crossman E., Goodeve P. (1983). Feedback control of hand movement and Fitts' law. *Quarterly Journal Experimental Psychology*, 35A:251-278.

[Donders68] Donders, F. C. (1868). Die Schnelligkeit psychischer Prozesse. (On the speed of mental processes). *Archiv für Anatomie und Physiologie und wissenschaftliche Medizin*, 657-681.

[Fitts54] Fitts, P. M. (1954). The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology*, 47, 381-391.

[Fitts54b] Fitts, P. M., & Deininger, R. L. (1954). S-R compatibility: Correspondence among paired elements within stimulus and response codes. *Journal of Experimental Psychology*, 48, 483-492.

[Fitts54c] Fitts, P. M., & Seeger, C. M. (1954). S-R compatibility: Spatial characteristics of stimulus and response codes. *Journal of Experimental Psychology*, 46, 199-210.

[Fitts64] Fitts, P. M., & Peterson, J. R. (1964). Information capacity of discrete motor responses. *Journal of Experimental Psychology*, 67, 103-113.

[Grossman2005] Grossman, T. and Balakrishnan, R. (2005). The Bubble Cursor: Enhancing target acquisition by dynamic resizing of the cursor's activation area. *Proc. CHI '05: ACM Conference on Human Factors in Computing Systems*. New York: ACM Press, 281-290.

[Hartley28] Hartley, R.V. L. (1928). Transmission of information. *Bell System Technical Journal*, 535, 1928.

[Hertzum2013] Hertzum, M., and Hornbæk, K., (2013). The Effect of Target Precuing on Pointing with Mouse and Touchpad, *International Journal of Human-Computer Interaction*, vol. 29, no. 5 (2013), pp. 338-350.

[Hick51] Hick, W. E. (1951). A simple stimulus generator. *Quarterly Journal of Experimental Psychology*, 3, 94-95.

[Hick52] Hick, W. E. (1952). On the rate of gain of information. *Quarterly Journal of Experimental Psychology*, 4, 11-26.

- [Hick53] Hick, W. E. (1953). Information theory in psychology. *IEEE transactions on Information Theory*, 1, 130–133.
- [Hyman53] Hyman, R. (1953). Stimulus information as a determinant of reaction time. *Journal of Experimental Psychology*, 45, 188–196.
- [ISO2000] ISO. (2000). *ISO9241-9 International standard: Ergonomic requirements for office work with visual display terminals (VDTs)—Part 9: Requirements for non-keyboard input devices*: International Organization for Standardization.
- [Kristensson2014] Kristensson, P.O. (2014). From wax tablets to touchscreens: an introduction to text entry research. *ACM XRDS* 21(1): 28-33. *Invited. Special Issue on Computers and Language.*
- [Landauer85] Landauer, T. K., & Nachbar, D. W. (1985). Selection from alphabetic and numeric menu trees using a touch screen: Breadth, depth, and width. *Proceedings of the CHI'85 Human Factors in Computing Systems*, 73–78. New York: ACM.
- [Lewis99] Lewis, J. R., Kennedy, P. J., & LaLomia, M. J. (1999). *Development of a Diagram-Based Typing Key Layout for Single-Finger/Stylus Input*. Proc. of The Human Factors and Ergonomics Society 43rd Annual Meeting.
- [MacKenzie89] MacKenzie, I. S. (1989) A note on information-theoretic basis for Fitts' law. *Journal of Motor Behavior*, 21, 323-330.
- [MacKenzie92] MacKenzie, I. S. & Buxton, W. (1992). Extending Fitts' law to two-dimensional tasks. *Proceedings of the CHI'92 Conference on Human Factors in Computer Systems*. New York: ACM.
- [MacKenzie18] MacKenzie, I. S. (2018). Fitts' law. In K. L. Norman & J. Kirakowski (Eds.), *Handbook of human-computer interaction*, pp. 349-370. Hoboken, NJ: Wiley. doi:10.1002/9781118976005
- [McGuffin2002] McGuffin, M. and Balakrishnan, R. (2002). Acquisition of expanding targets. *ACM CHI Conference on Human Factors in Computing Systems*. p. 57-64.
- [Merkel85] Merkel, J. (1885). Die zeitlichen Verhältnisse der Willenstätigkeit (The Temporal Relations of the Actions of Will, or The Timing of Voluntary Action). *Philosophische Studien (Philosophical Studies)*, 2, 73–127.
- [Nguyen2012] Nguyen, H., Bartha, M. C. (2012). Shape Writing on Tablets: Better Performance or Better Experience? *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, San Diego, CA, HFES, pp. 1591-1593
- [Norman82] Norman, D. A. and Fisher, D. (1982). *Why alphabetic keyboards are not easy to use: Keyboard layout doesn't much matter*. Human Factors, 24, 509-519.
- [Nyquist24] Nyquist, H. (1924). Certain factors affecting telegraph speed. *Bell System Technical Journal*, 47, p. 617.

[RIM2012] Research in Motion (2012). *The BlackBerry 10 keyboard demoed.* http://www.youtube.com/watch?v=l0wclug_TUU, checked Oct. 2014.

[Schenider2013] Schneider, T. D. (2013). Information Theory Primer, With an Appendix on Logarithms, <http://alum.mit.edu/www/toms/paper/primer/> (checked October 2014).

[Sears91] Sears, A., and Shneiderman, B. (1991). High precision touchscreens: Design strategies and comparison with a mouse. *International Journal of Man-Machine Studies*, 34, 4, pp. 593-613.

[Seow2005] Steven C. Seow (2005). Information Theoretic Models of, HCI: A Comparison of the, Hick-Hyman Law and Fitts' Law, *Human-Computer Interaction*, 2005, Volume 20, pp. 315–352.

[Sears94] Sears, A., and Shneiderman, B. (1994). Split menus: Effectively using selection frequency to organize menus. *ACM Transactions On Computer-Human Interaction* 1, 1, 27 - 51.

[Shannon48] Shannon, C. E. (1948) A mathematical theory of communication. *Bell System Technical Journal*, 27, 379–423, 623–656, 1948.

[Sourkoreff2004] Soukoreff, R.W., & MacKenzie, I.S. (2004). *Towards a standard for pointing device evaluation, perspectives on 27 years of Fitts's law research in HCI*. International Journal of Human- Computer Studies, 61, 751–789.

[Welford68] Welford, A. T. (1968). Fundamentals of skill. London: Methuen.

[Whirlscape2014] Whirlscape (2014). *The Minuum Keyboard*. <http://minuum.com/>

[Wright2013] Wright, C. E., & Lee, F., (2013) Issues Related to HCI Application of Fitts's Law, *Human-Computer Interaction*, Volume 28, Issue 6, pp. 548-578, 2013.

[Zhai2002] Zhai, S., Accot, J., Woltjer, (2002) R. Human Action Laws in Electronic Virtual Worlds – An Empirical Study of Path Steering Performance in VR, *Presence: Teleoperators and Virtual Environments*, Vol.13(2), 113-127.

[Zhai2004] Zhai, S., Kong, J., & Ren, X. (2004). Speed-accuracy tradeoff in Fitts' Law tasks: On the equivalency of actual and nominal pointing precision. *International Journal of Human-Computer Studies*. 61, 6, 823–856.

[Zhai2012] Zhai, S. and Kristensson, P.O. (2012). The word-gesture keyboard: reimagining keyboard interaction. *Communications of the ACM* 55(9): 91-101. Invited. (Research Highlights).