Interlude: User Models

- Idea: if we had a model of how users "work", then we could predict how they will interact with a specific UI and what their user performance will be
- Advantage (theoretically): no user studies and no UI mock-ups necesary any more
- Related fields: psychophysics, user interface design, usability

The Power Law of Practice

- Describes, what time is needed to perform an activity after the *n*-th repetition:

$$T_n = \frac{T_1}{n^a}$$

T_1 = time needed for first performance of the activity,
T_n = time for *n*-th repetition,
$a \approx 0.2 \ldots 0.6$

- Warning:
  - Applies only to mechnical activities, e.g. :
    - Using the mouse, typing on the keyboard
  - Does not apply to cognitive activities, e.g., learning for exams! ;-)
- This effect must be kept in mind when designing experiments!

Hick's Law

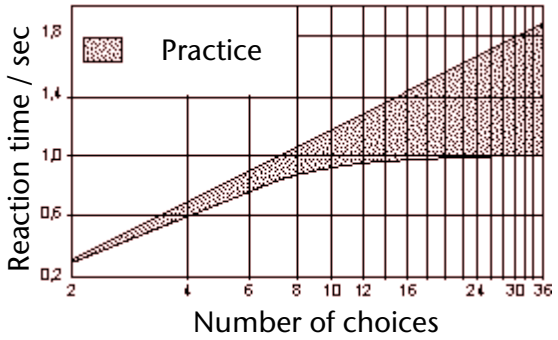
- Describes the time needed to make a 1-out-of-*n* selection, but there cannot be any cognitive workload involved:

$$T = I_c \log_2(n+1) \quad , \quad I_c \approx 150 \text{ msec}$$

where *n* = number of choices

  - Example: *n* buttons + *n* lights, one is lighted up randomly, user has to press corresponding button
  - Assumption: the distribution of the choices is uniform!

- Warning: don't apply this law too blindly!
  - E.g., practice has a big influence on reaction time



- Sometimes, Hick's law is taken as proof that one large menu is more time-efficient than several small submenus ("rule of large menus") ... I argue this is — mathematically — correct only because of the "+1", for which there is no clear experimental evidence! Besides, there are many other factors involved in large menus (clarity, Fitts' law, ...)

Fitts's Law

- Describes the time needed to reach a target
- Task: reach and hit a specific target as quickly and as precisely as possible with your hand / pencil / mouse /etc., from a resting position → "target acquisition"
- The law:

$$T = b \log_2\left(\frac{D}{W} + 1\right) + a$$

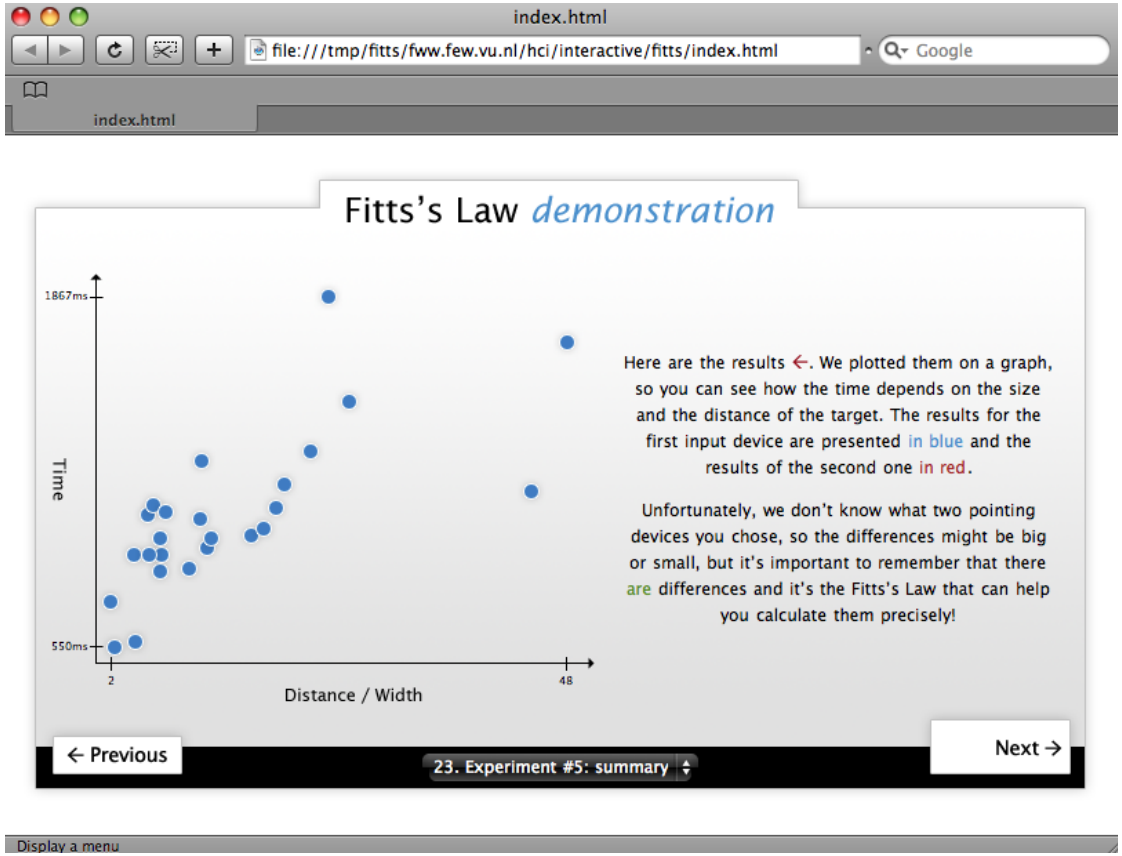
where D = distance between resting position and target,
W = diameter of the target

- The "index of difficulty" (ID) =

$$\log_2\left(\frac{D}{W} + 1\right)$$

Demo / Experiment

- Fitt's Law does apply directly to mouse movements needed to hit icons and buttons



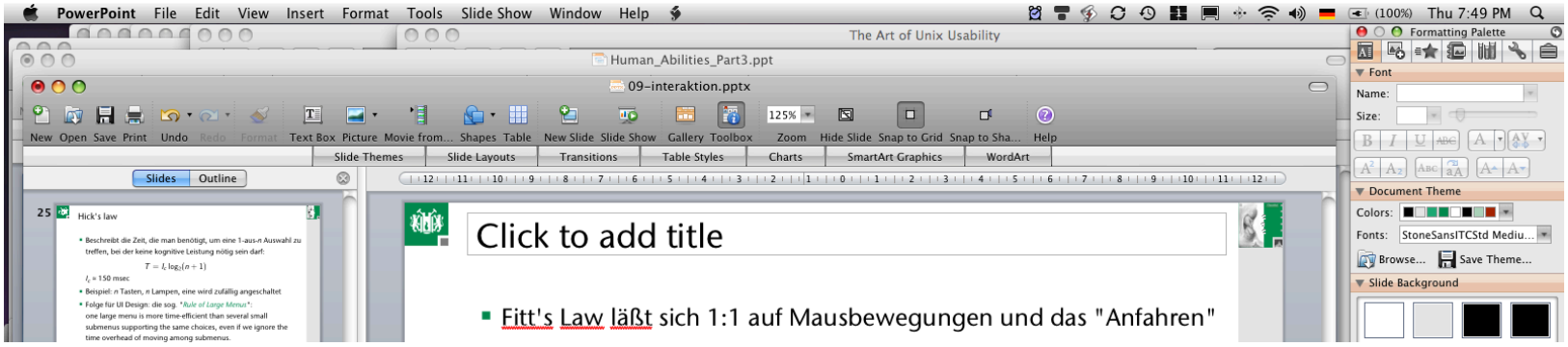
Marcin Wichary , Vrije Universiteit: http://fww.few.vu.nl/hci/interactive/fitts/

Applications of Fitts' Law

- "Rule of Target Size": The size of a button should be proportional to its expected frequency of use
- Other consequences:

  "*Macintosh fans like to point out that Fitts's Law implies a very large advantage for Mac-style edge-of-screen menus with no borders, because they effectively extend the depth of the target area off-screen. This prediction is verified by experiment.*"

  [Raymond & Landley: "The Art of Unix Usability", 2004]



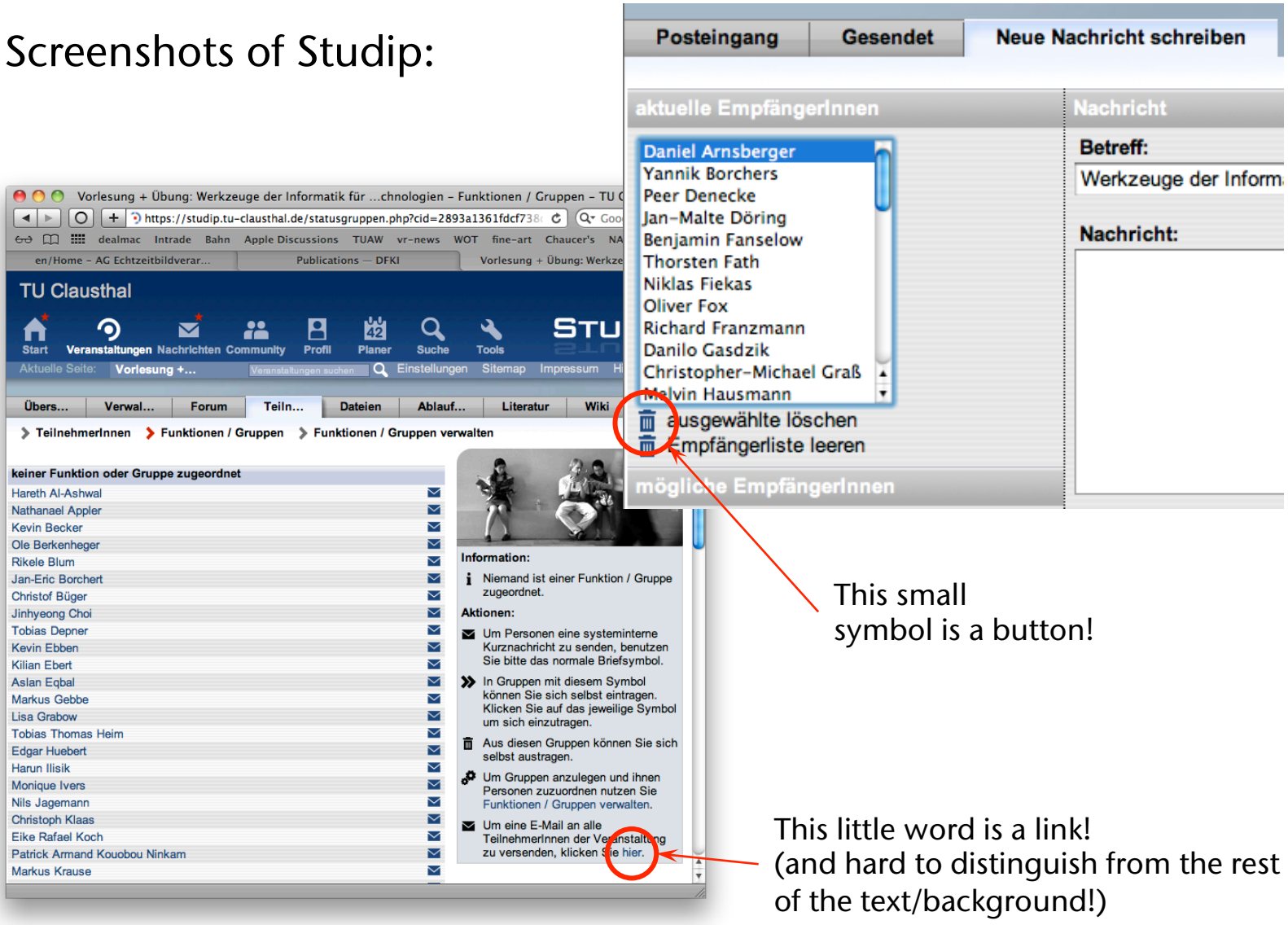
- *Tear-off menus* and *context menus*: they decrease the average travel distance *D*
- Apple's "Dock": the size of the icons gets adjusted dynamically

- Obvious limitations of Fitts's Law:
  - Fitts's Law cannot capture all aspects/widgets of a GUI
    - E.g. moving target (like scrollable lists)
  - There are many other decisions with regards to the design of a UI that are contrary to an application of Fitts's law

Fun and intructive quiz: go to the homepage of this VR course → scroll down to section "Online Literatur und Resources im Internet" → find "A Quiz Designed to Give You Fitts"

Bad Examples

- Screenshots of Studip:



Digression: the 80/20 Rule

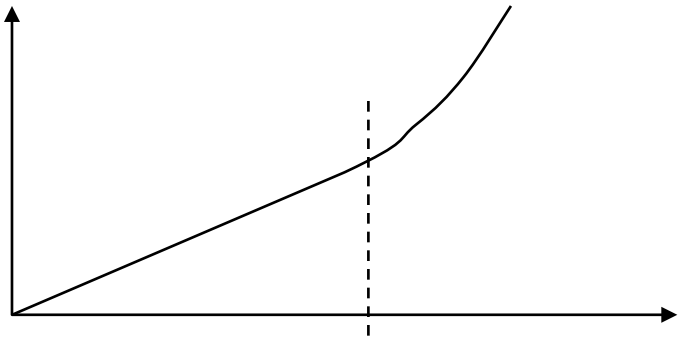
- 80% all the total usage time of a product, we utilize only 20% of its features
  - Applies to menus, software as a whole, "consumer electronics", cars, ...
- 80% of the malfunctions of a product have their cause in only 20% of its components
- 80% all the old box in the software are caused by only 20% of its programmers and designers
- 80% all the revenue of the company is generated by only20% of their products
- ...

Selection

- *Task decomposition*:
  1. Switch selection mode on
  2. Identify object(s) to be included in selection
     - Give some kind of feedback during this step
  3. Confirm/cancel
  4. Feedback: which objects are actually selected
- Definitions:
  - Display / visual space = the space in the VE = the space containing the virtual "pointer" (e.g. virtual hand)
  - Control / motor space (physical space) = space outside the VE = the space containing the tracker
  - Control-Display ratio (C-D ratio) = the ratio of the movement (translation and/or rotation) in control space (physical space) over the resulting movement in display space (interaction space)
    - Simple example: mouse acceleration in 2D desktop GUIs

Direct Selection & Manipulation with Non-Linear Mapping (the "Go-Go Technique")

- Direct selection/manipulation requires direct touching & grasping objects with virtual hand
- Goal: increase working volume
- Idea:
  - Scale tracking values non-linearly outside the "near field"
  - Keep linear scaling in near-field for better precision
- Suitable for head and hand tracking
- Works only with absolute input devices
- Disadvantages:
  - Proprioception gets lost
  - No remote precision handling

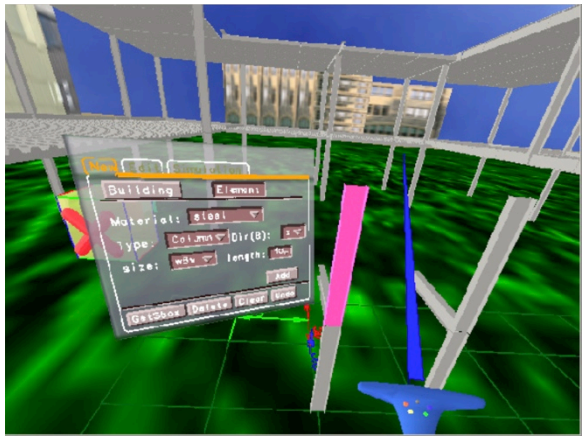


The Go-Go Interaction Technique:

... to reach farther in virtual environments

Some Possibilities for Step 2 (Identifying Objects)

- Ray-based (ray casting)
  - E.g.. shoot "laser pointer" from virtual hand into scene
  - Or: extend conceptual ray from current viewpoint through finger tip (a.k.a. occlusion technique)
- Volume-based, e.g. using a cone around the ray
- Direct = touch with hand
- Speech (name objects)
- Menues
- Mixed techniques:
  - Image plane interaction (later)
  - World-in-Miniature (dito)
  - Etc.



Laser pointer

Menu with ray technique

Overview of Some Ray-Based Techniques

- Variables used in the following:

  H = hand position

  E = viewpoint

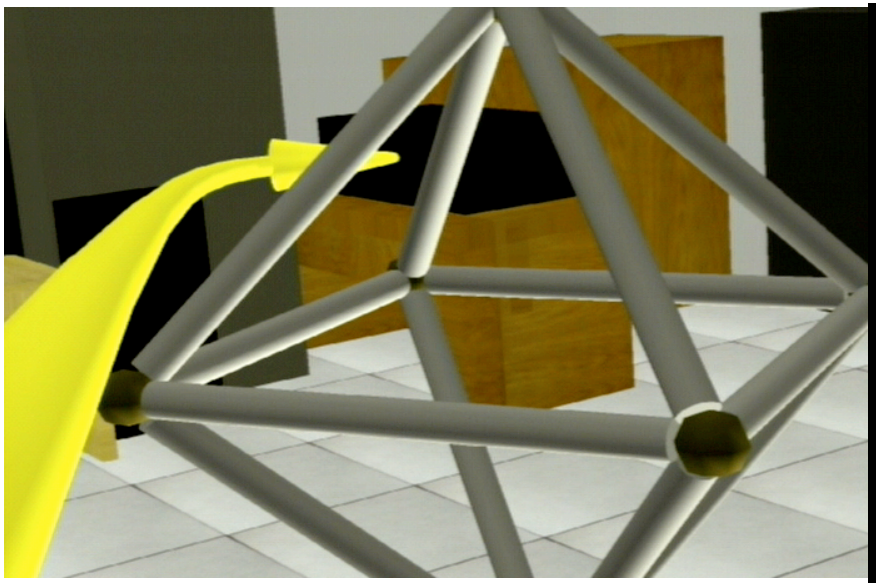
  h = "pointing direction" of the hand

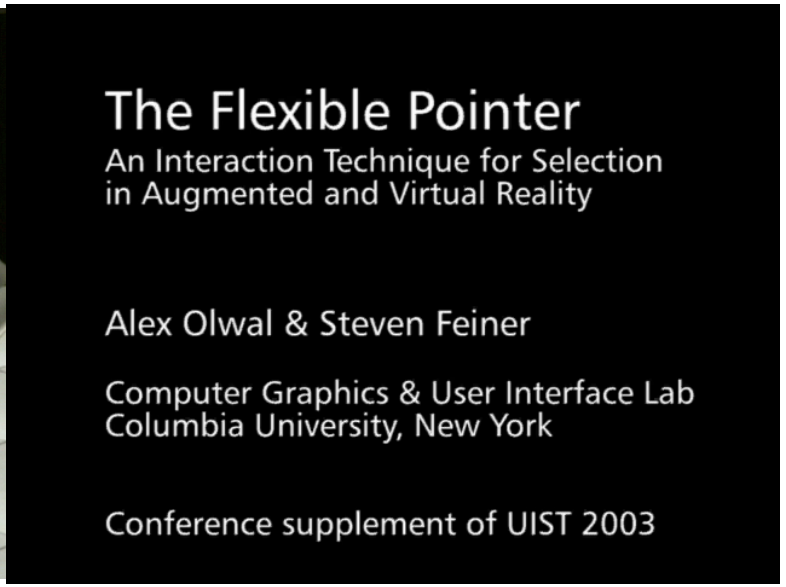
  H_2 = position of the left hand

Technique	Volume	Origin	Direction
Raycasting	ray	H	h
Flashlight	cone	H	h
Two-handed pointing	ray	H_2	$H - H_2$
Occlusion selection	ray	E	H - E
Aperture	cone	E	H - E

Flexible Pointer [2003]

- Beobachtung: Menschen versuchen, mit der Zeigegeste eine "Kurve" zu beschreiben, wenn sie auf etwas zeigen, das nicht in der "*line of sight*" ist.
- Umsetzung in VR: gebogener Zeigestrahl
- Problem: intuitive und einfache Beschreibung der Krümmung mittels Eingabegeräten (Dataglove, Tracker, ...)



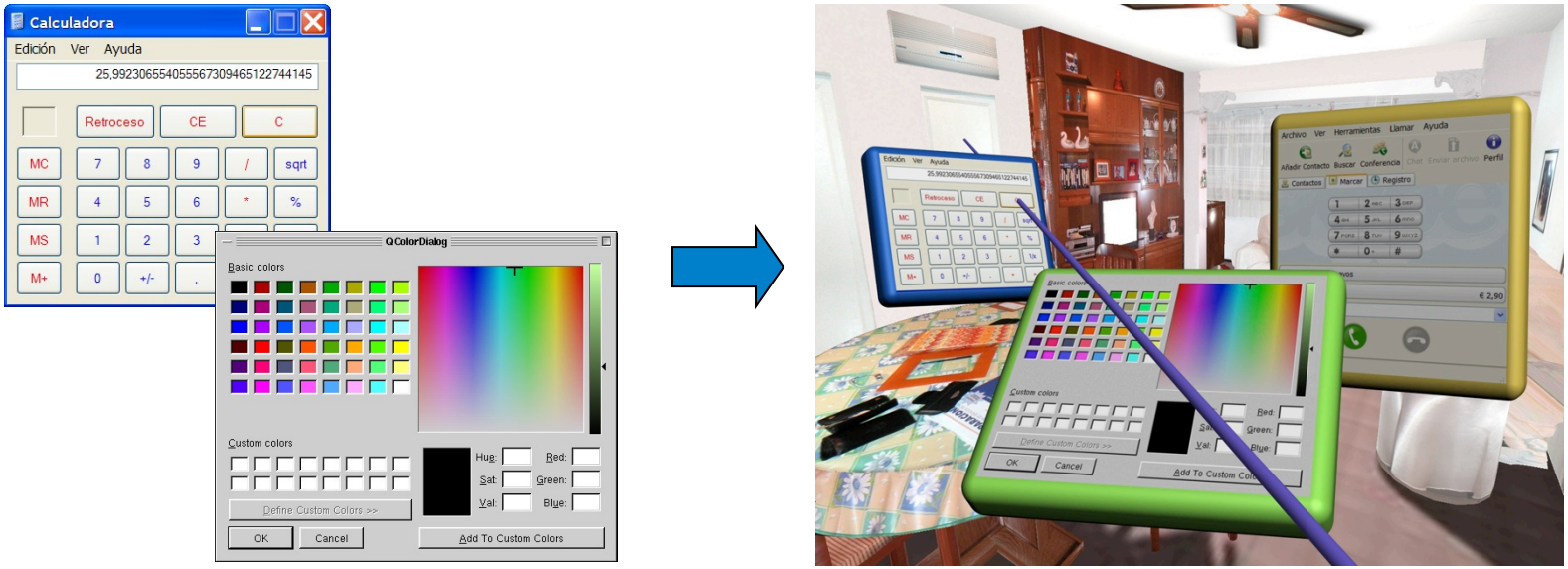


Isomorph vs Non-Isomorph Techniques

- There are 2 different ways to map control space → display space:
- Isomorph:
  - 1:1 correspondence between control (physical) space and display (interaction) space
  - Very natural → very intuitive; imitates interaction in real world
  - Better suited for direct interaction techniques
  - Problem often times: working volume
- Non-isomorph:
  - "Magic" tools (interaction metaphors) expands working volume or precision
  - Most interaction techniques are non-isomorph
  - Problem often times: precision with small / many objects
  - Better suited for remote interaction techniques

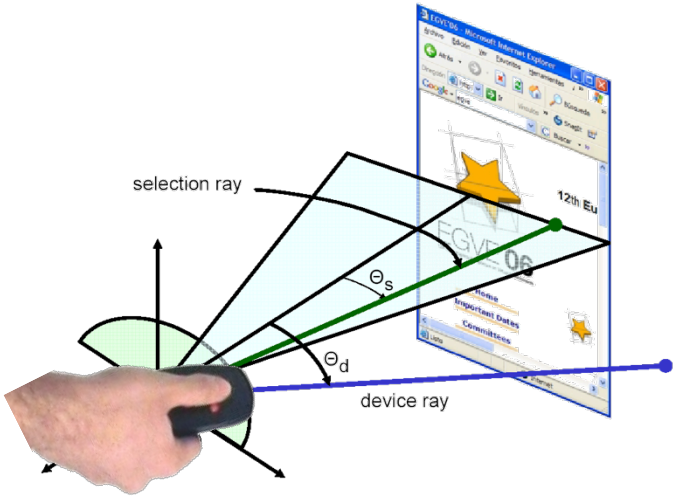
Friction Surfaces — Example for the C-D Ratio [2006]

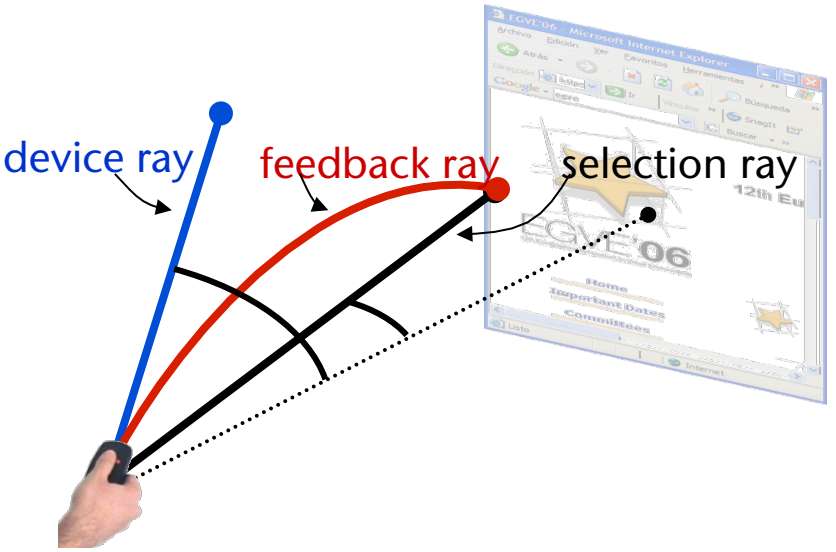
- Task here: control so-called hybrid interfaces
  - Goal: control 2D GUIs of desktop apps directly in VR
  - Implementation: a modified VNC client



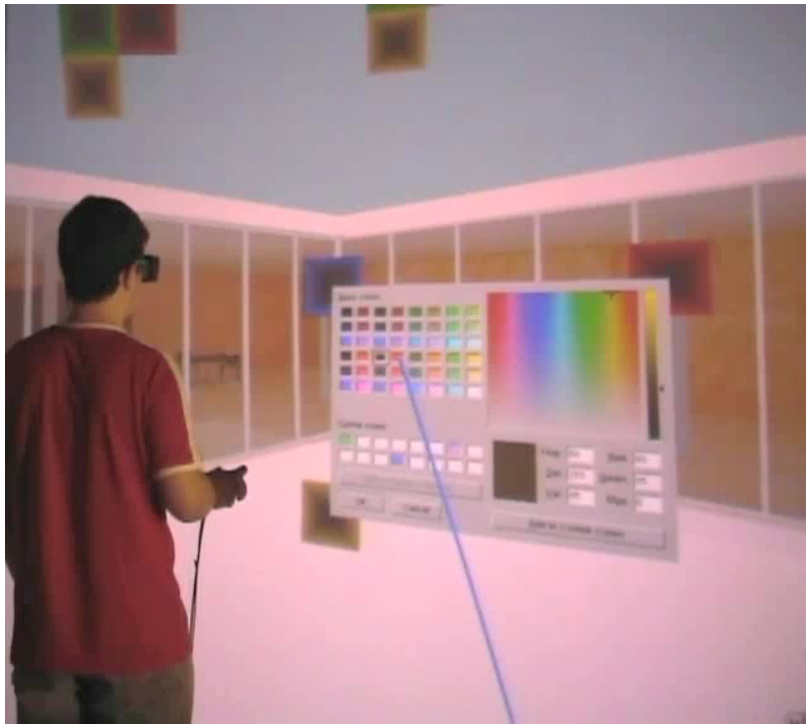
- Problem: the target width (here: solid angle!) is extremely small

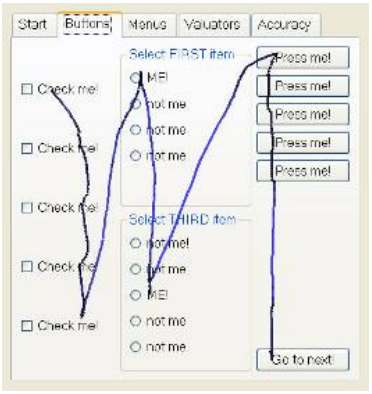
- Idea:
  - Scale the C-D ratio down, as soon as user interacts with a 2D window (in VE)
  - Problem: how to make the non-isomorphism intuitive, how to bridge the noticeable difference between motor & display space?
- Two rays were irritating to users
- Solution: show just one ray, but make it bend



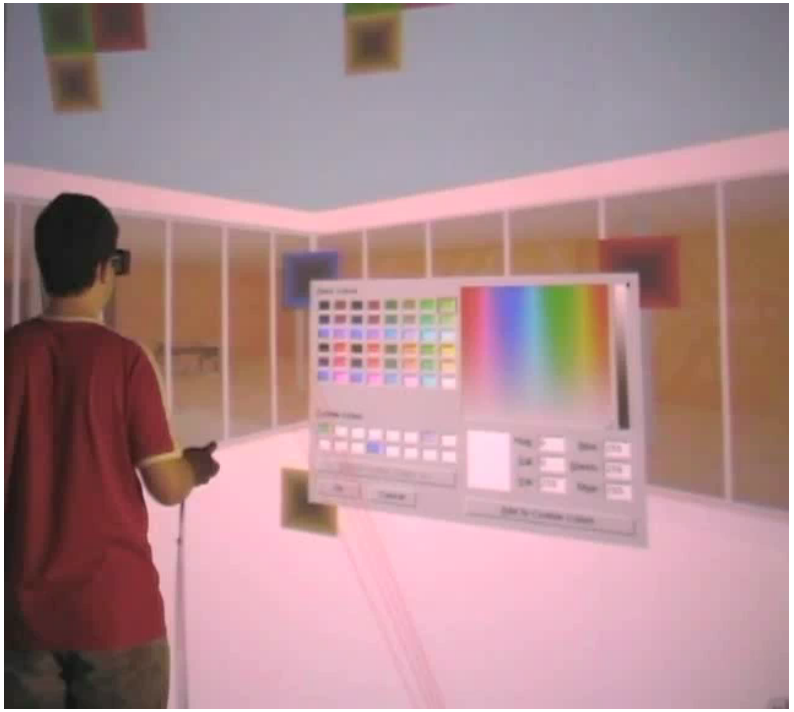


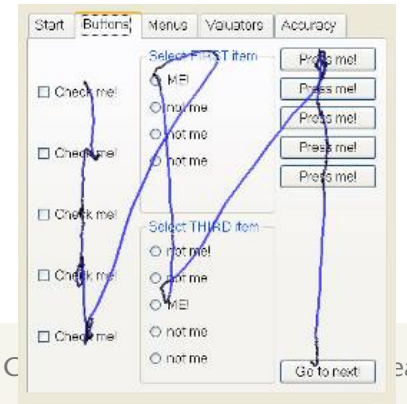
- Result: much increased user efficiency:





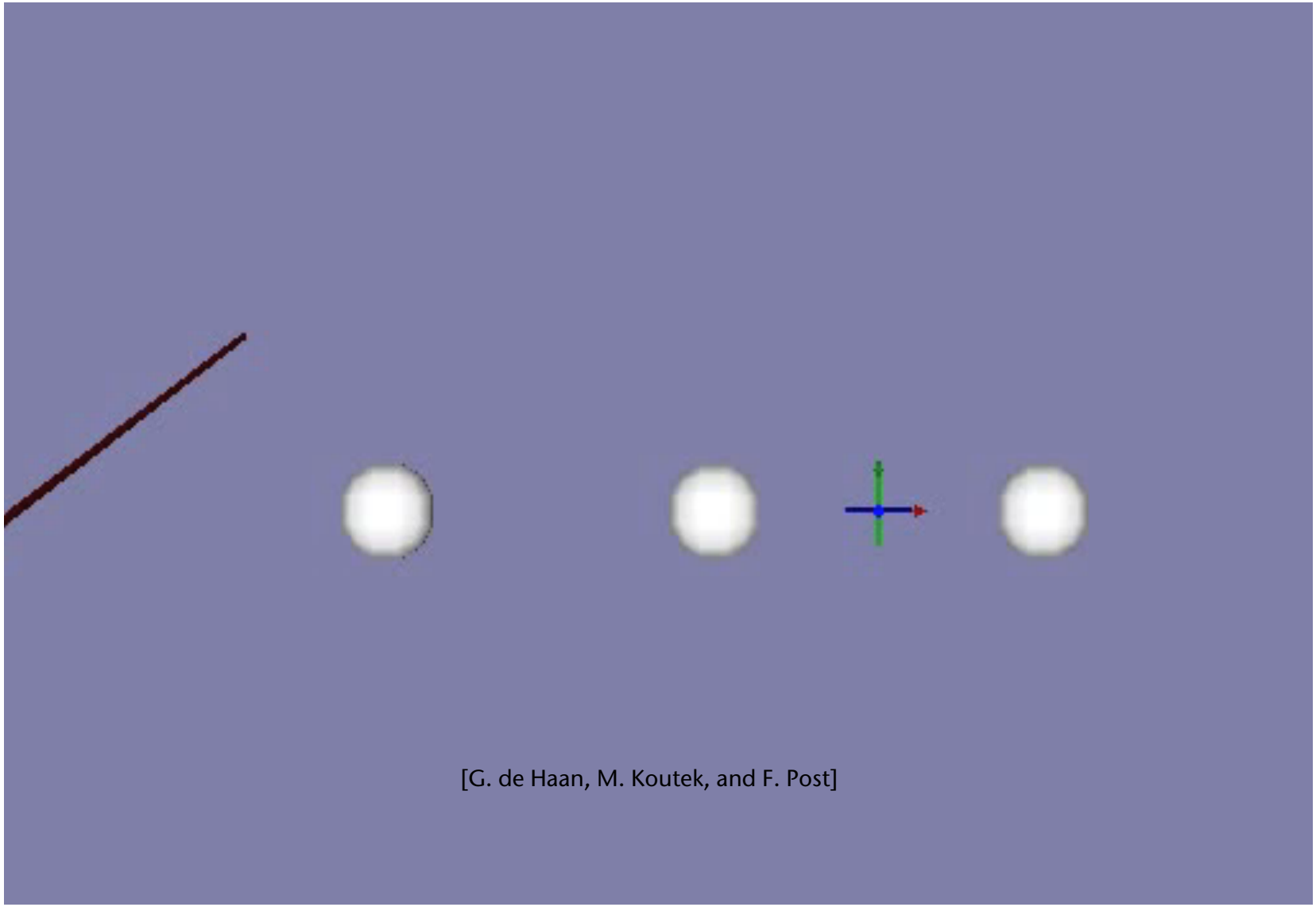
C-D ratio < 1



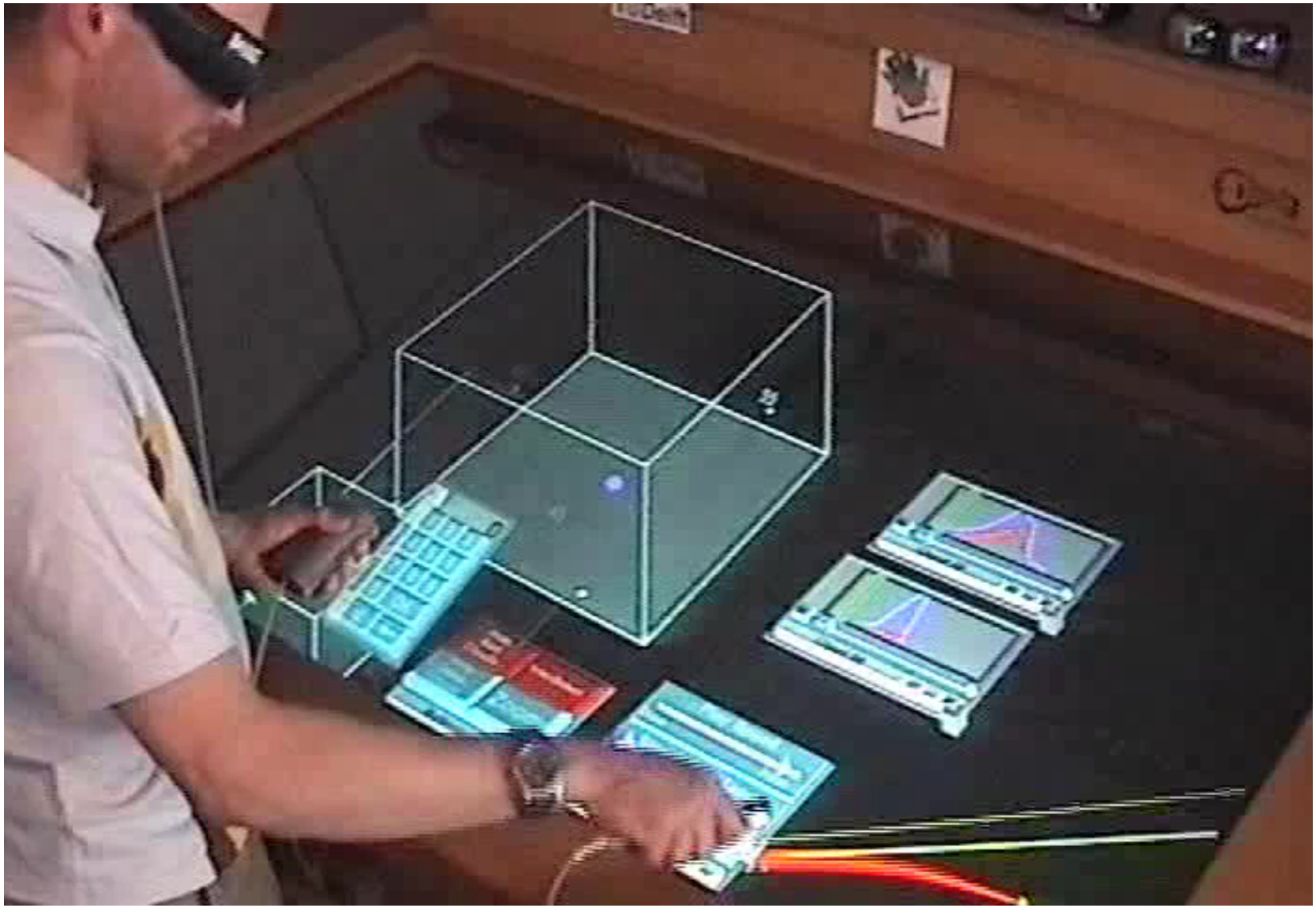


C-D ratio = 1

Video



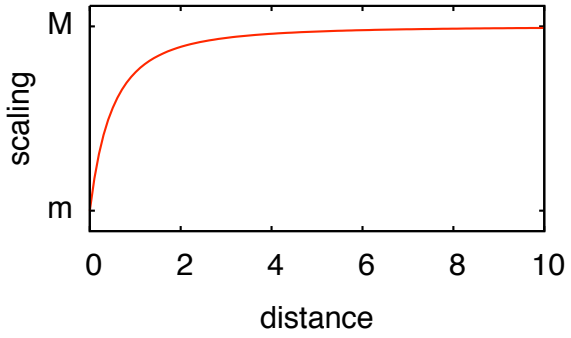
[G. de Haan, M. Koutek, and F. Post]



"Semantic" Pointing (Non-Constant C-D Ratio) [2008]

- Idea:
  - Modify C-D ratio depending on distance between pointer and closest target
  - Large distance $d \rightarrow$ scale motions from motor space up
  - Small distance $d \rightarrow$ scale motion downs = high precision in display space
  - E.g. with a function like this one:

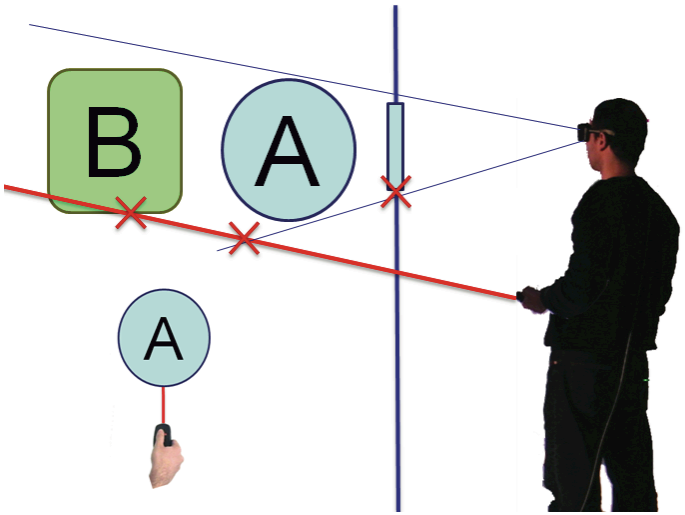
$$s(d) = M + \frac{m - M}{(1 + d)^{\alpha}}$$



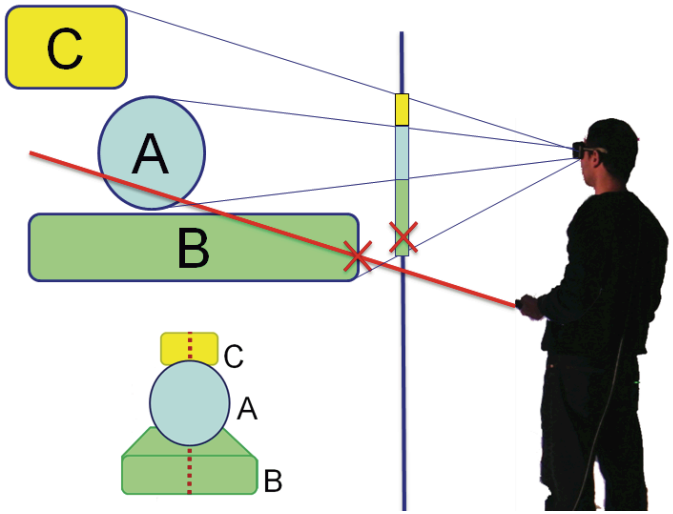
- Visual feedback:
  - Cursor size ~ C-D ratio
  - Color of pointer visualizes distance of target (e.g. "red" = "very close")

The Eye-Hand Visibility Mismatch [2008]

- Two obvious problems of all techniques where ray emanates from the user's (virtual) hand :
  1. The set of objects visible from viewpoint *E* is different from set of objects "visible" from hand position *H*



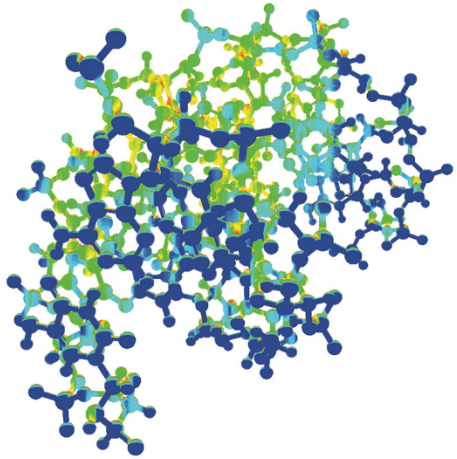
Object B is selectable, but not visible

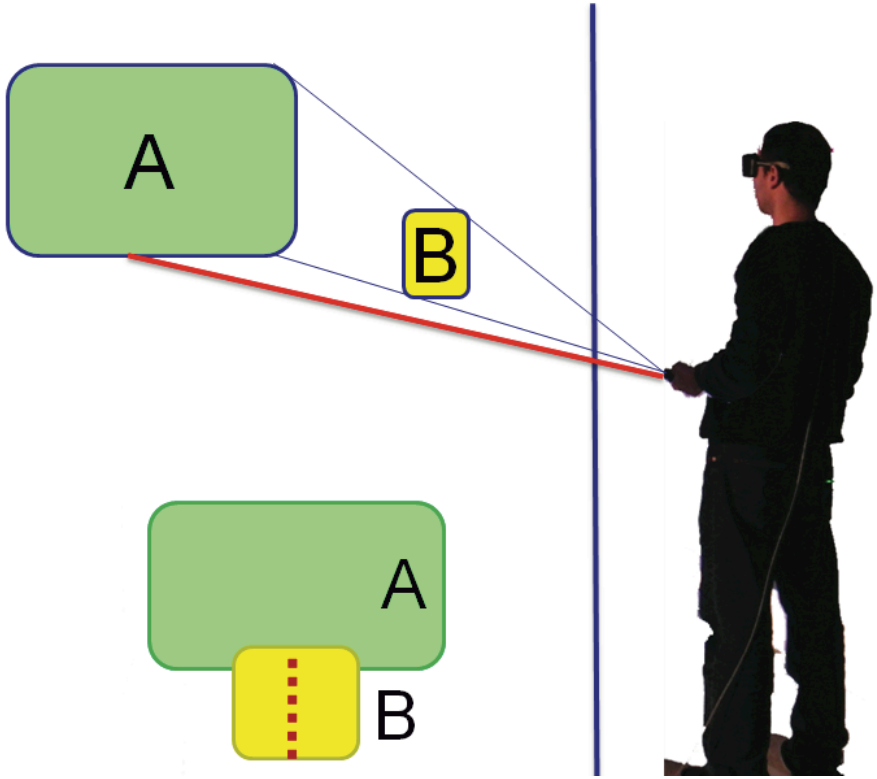


Object C is visisble, but not selectable

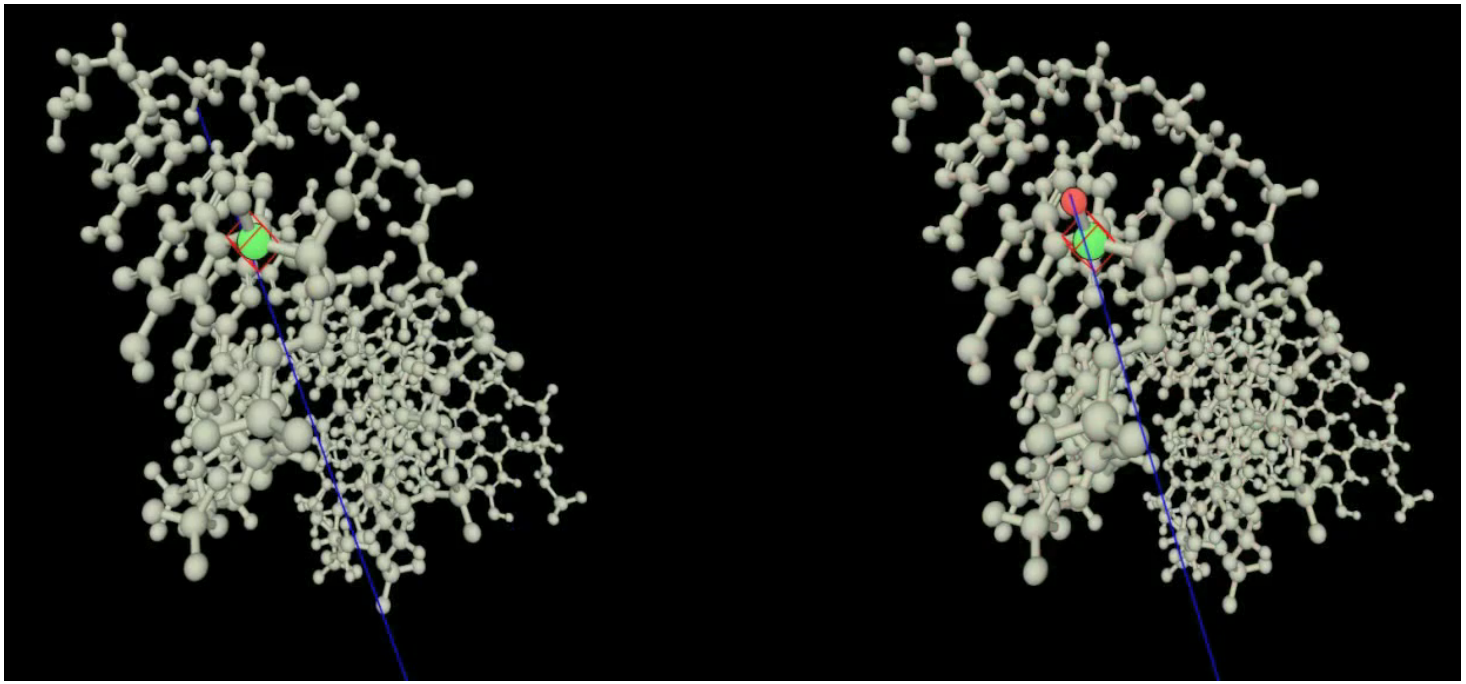
2. The surface of an object "visible" from *H* is different than surface visible from E; consequences:

- Real target width is different than visible target width
- Perhaps no/insufficient feedback during selection process





- Idea:
  - Shoot selection ray emanating from viewpoint *E* into hand direction *h*
  - Visual feedback: ray emanating from *H* to first intersection of selection ray
- Experiment shows: users are ca. 15-20% faster than with normal ray

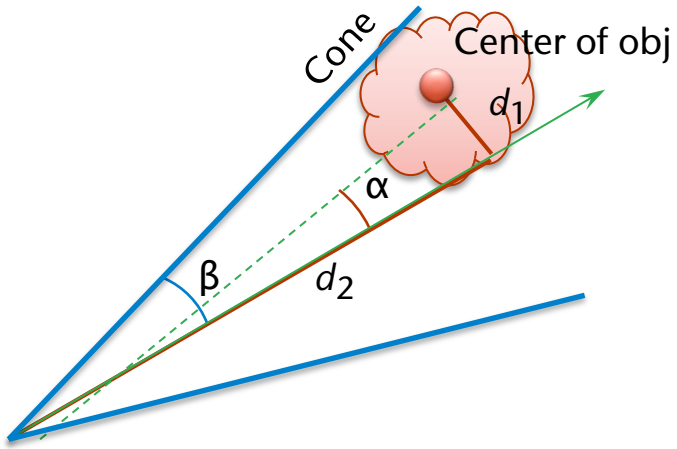


Argelaguet, Andujar, Trueba

IntenSelect: Ranking + Filtering [2005]

- Assumptions:
  - Cone is better than ray
  - In general, a lot of objects are in the cone (i.e., dense environment)
- Idea for disambiguation:
  - Define scalar field inside cone
  - Compute a "score" for each object
  - Create ranking of objects
- Simple scoring function:

$$s = 1 - \frac{\alpha}{\beta}$$

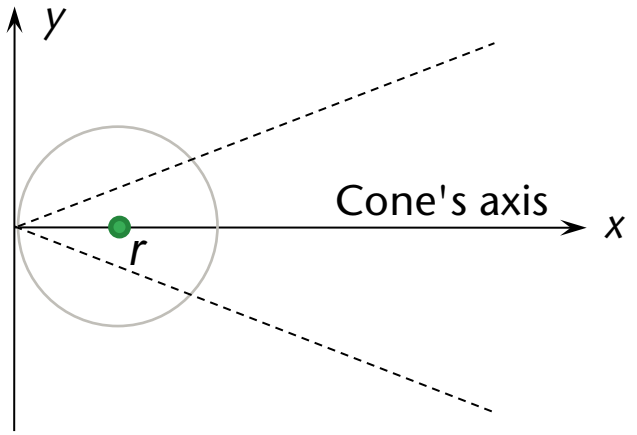


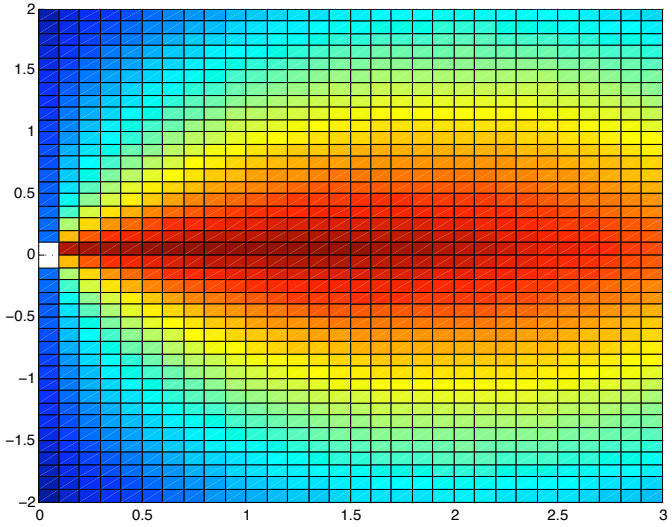
- A scoring function that prefers near objects:

$$s = 1 - \frac{1}{\beta}\tan^{-1}\left(\frac{d_1}{(d_2)^k}\right), \quad k \in [\tfrac{1}{2}, 1]$$

- Even more preference of near objects:

$$s = \left(1 - \frac{1}{\beta}\tan^{-1}\left(\frac{d_1}{(d_2)^k}\right)\right) + 0.1\left(1 - \frac{(x-r)^2 + y^2}{r^2}\right)$$





- Problem: jitter in user's hand leads to frequent changes in ranking
- Solution: filtering

$$s = s(t)$$

$$\hat{s}(t) = \sigma \hat{s}(t-1) + \tau s(t)$$

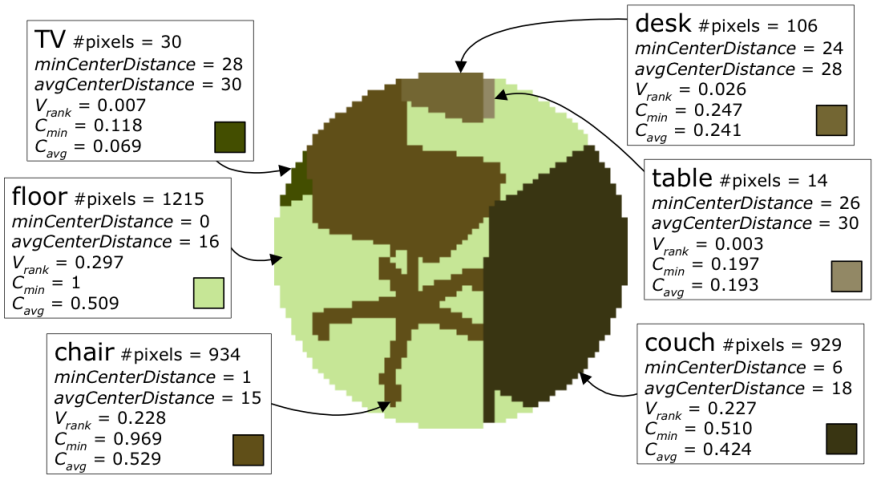
$s(t)$ = score over time, σ = "stickiness", τ = "snappiness"

- Generalization: FIR filter (see Chapter 7)
- Feedback to user:
  - Bend ray towards object with highest ranking
  - Show straight ray for cone's axis

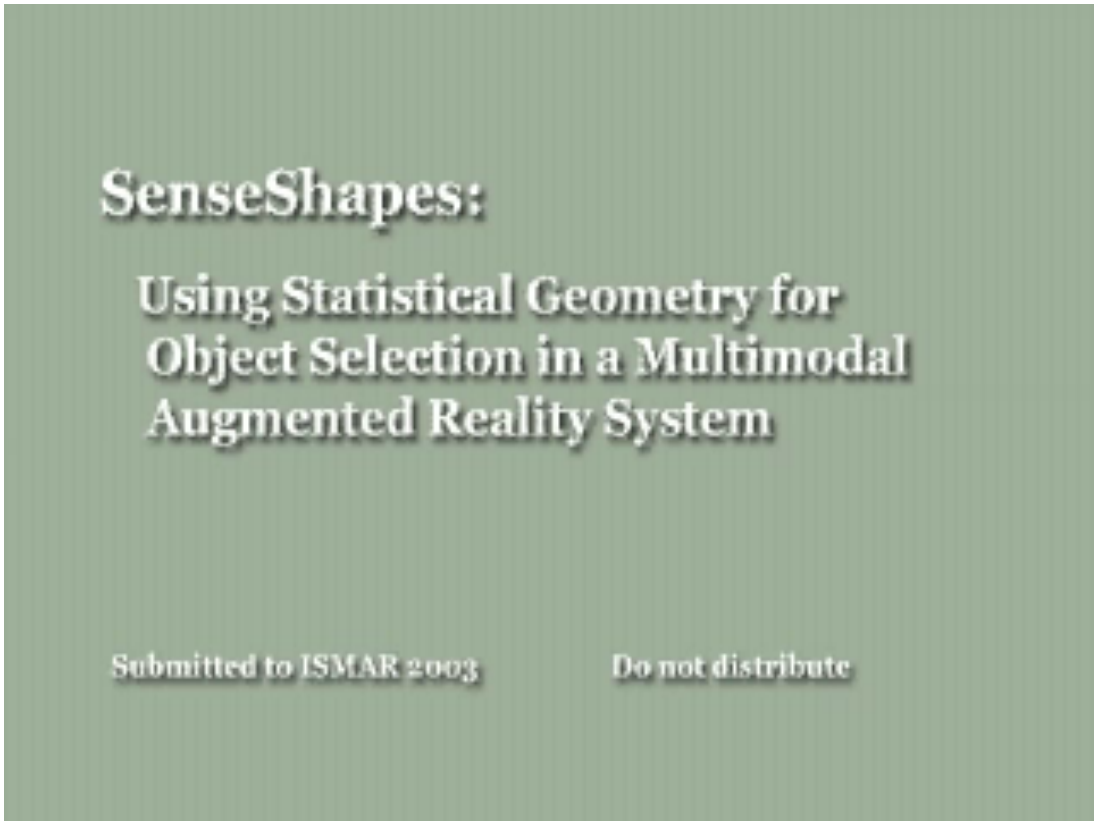
Other Ranking Functions

- Other distance functions, e.g., prefer far-away objects
- Better computation of "distance" from cone's axis:
  - Render object with low resolution into off-screen frame buffer with "viewpoint" = apex of cone, viewing direction = cone's axis
  - Compute average distance of all pixels of object from center (= cone's axis) :

$$s' = 1 - \frac{\frac{1}{n}\sum_{\text{pixel p}} d(p)}{\text{radius}}$$



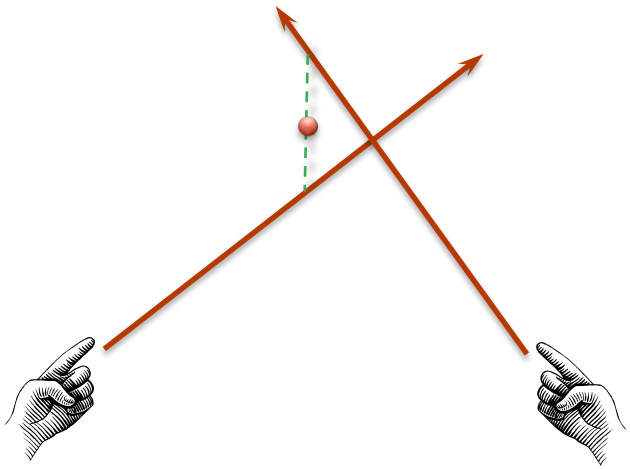
Application in a Multi-Modal AR Interface

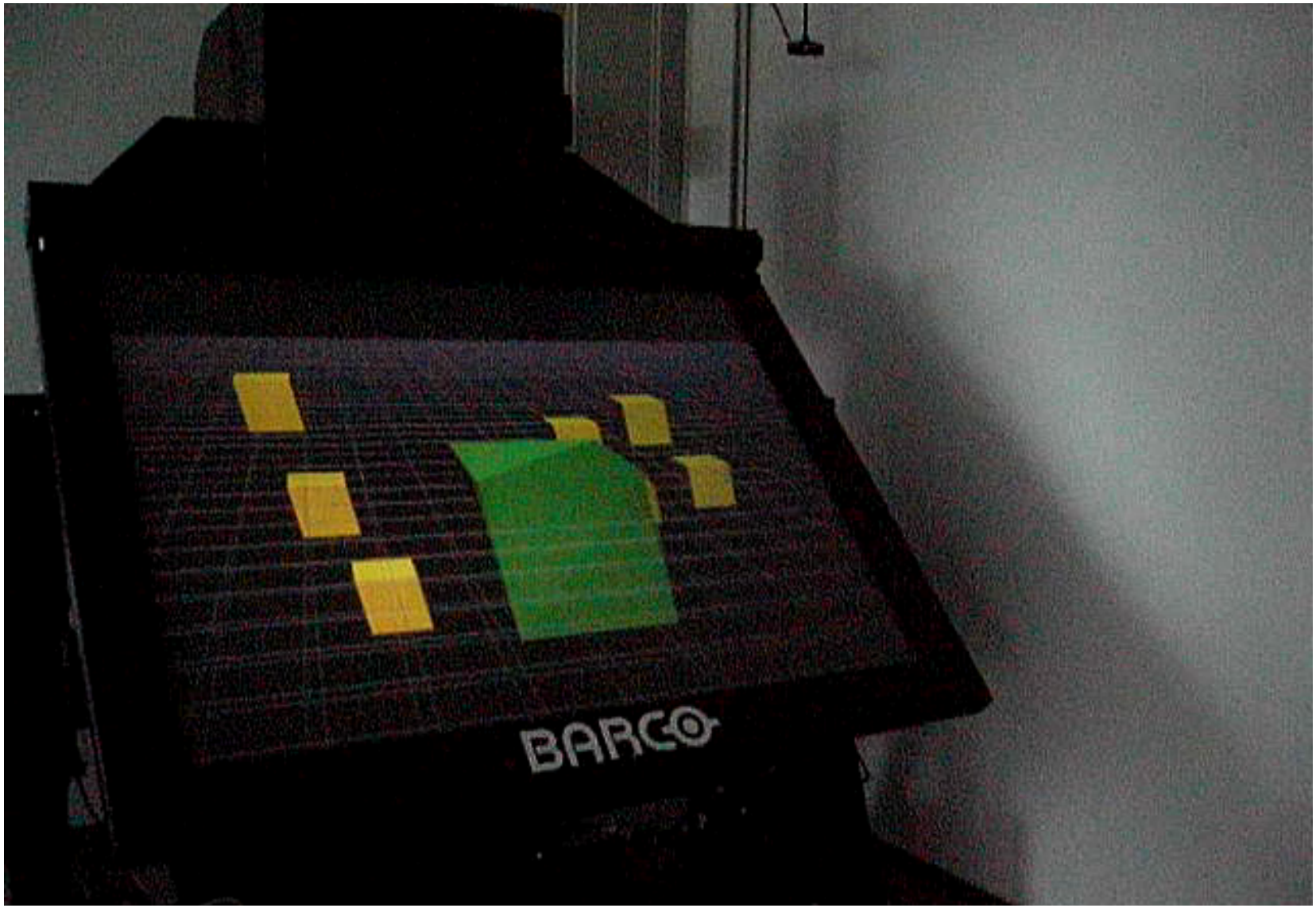


Here, disambiguation is done via voice commands

iSith: Two-Handed Selection [2006]

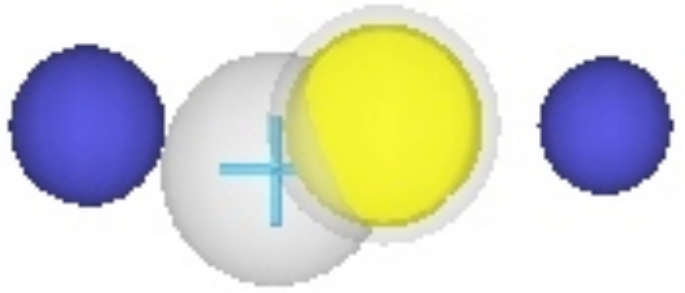
- Idea: intersection of two rays defines "selection center"
- Practical implementation:
  - One ray per hand
  - Trigger selection mode as soon as distance between rays < threshold
  - Midpoint of the shortest line between the rays = "selection center"
  - Select all objects "close enough" to this selection center



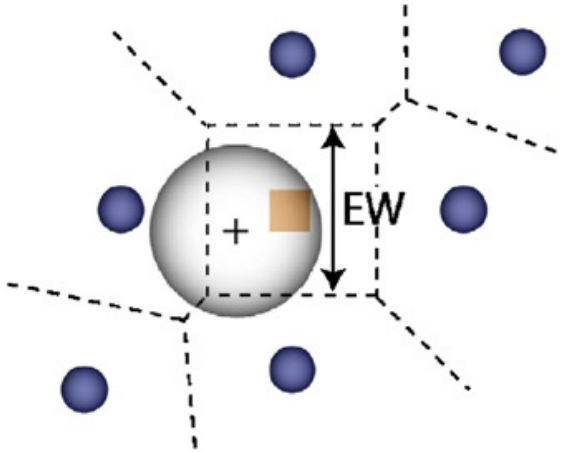

BARCO

The Bubble Cursor and the Depth Ray [2008]

- Another method to increase the effective target size
- Bubble Cursor:
  - 3D cross hairs
  - Select always the closest object
  - Make radius of transparent sphere around 3D crosshairs = distance to closest object (feedback for "density")
  - Feedback to indicate active object: transparent sphere around it

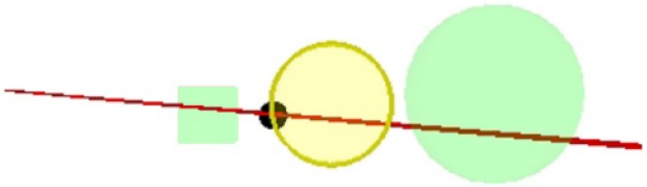


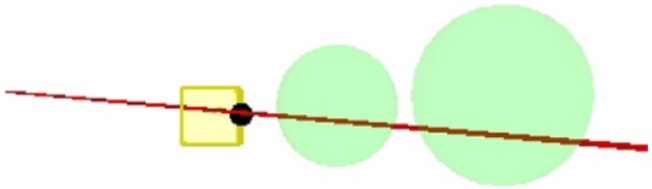
- Effective target size = Voronoi region of object, see Fig→

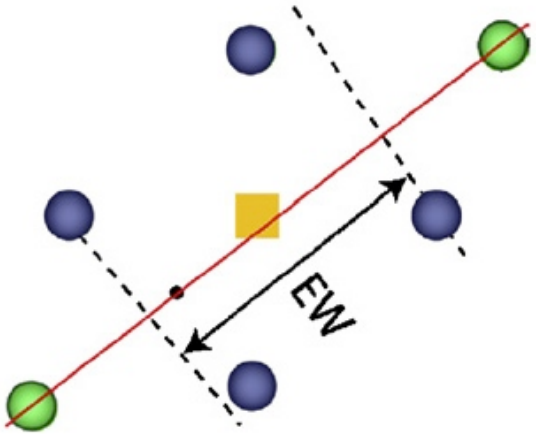


- (For Voronoi regions, see course "Geometrische Datenstrukturen für die CG" ;-))

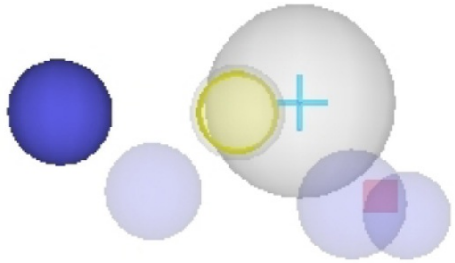
- Depth Ray:
  - Only consider objects being "stabbed" by the ray
  - User moves a "depth marker" along the ray
  - Of all "stabbed" objects, take the one closest to the depth marker
- Effective target size = intersection between ray and Voronoi region of object = segment on the ray

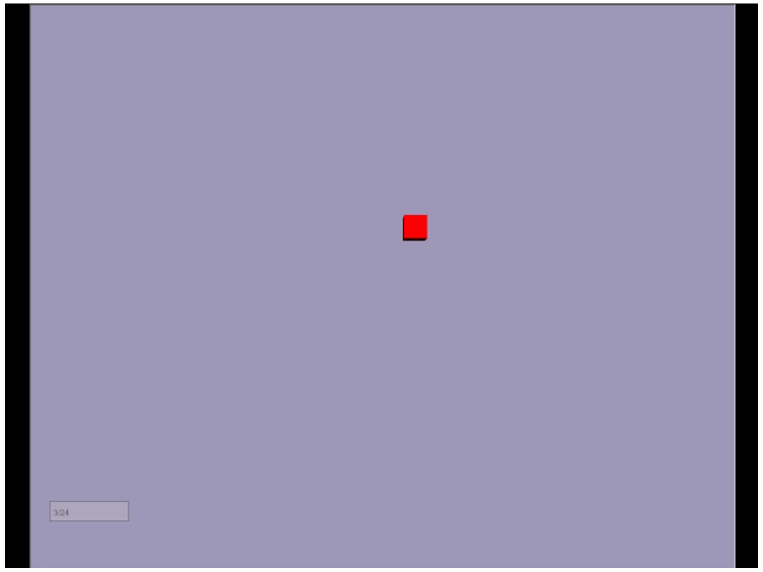




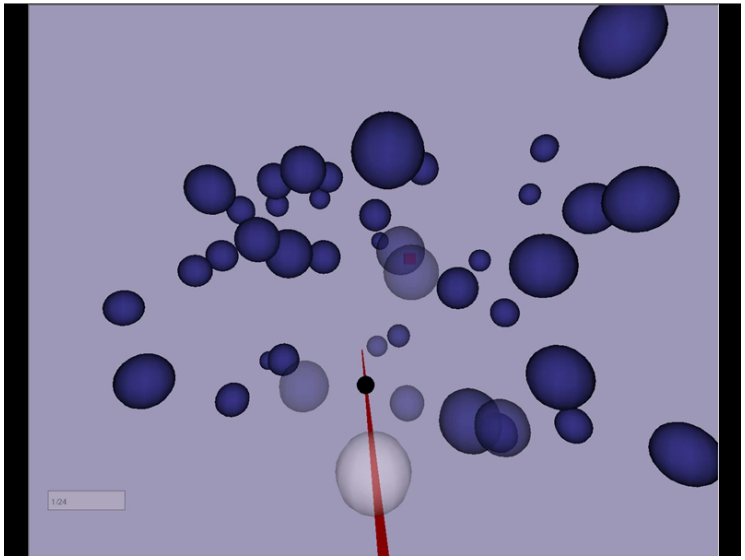


- Handling occlusion: make occluders in front of the depth marker transparent (possibly depending on the distance from the depth marker)





Bubble cursor [Lode van Acken]

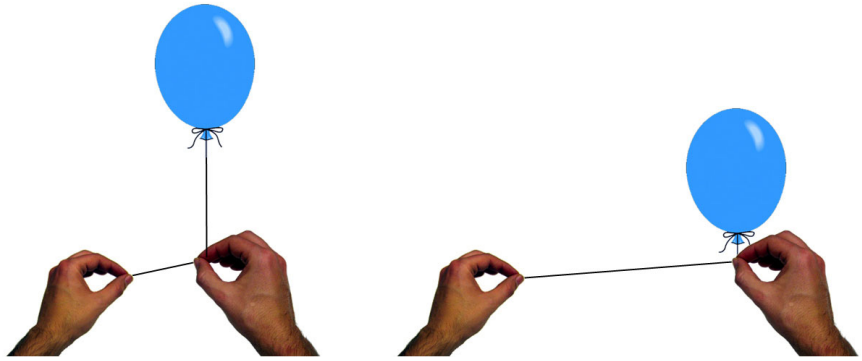


Depth Ray

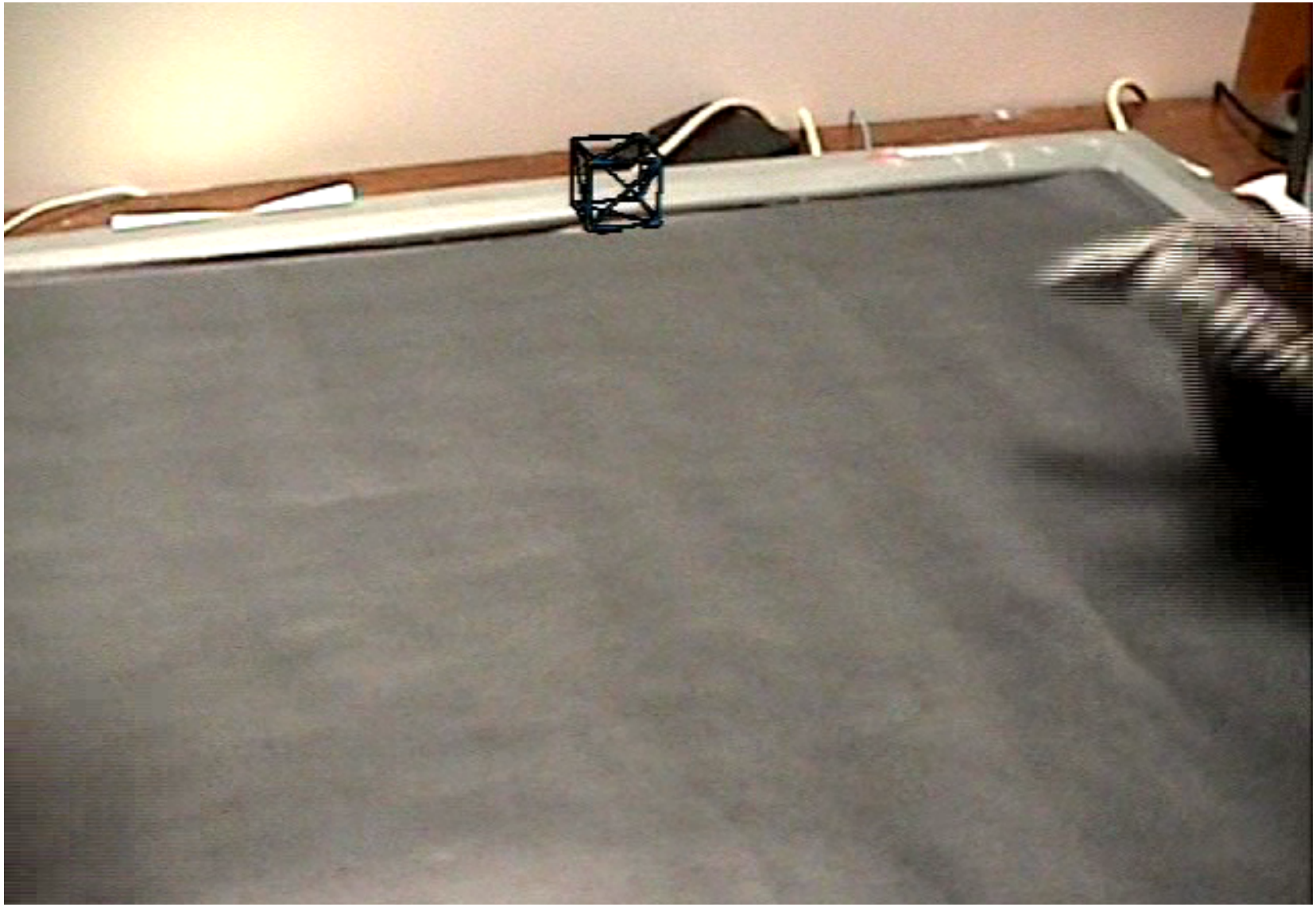
Balloon Selection

[2007]

- Idea: control a Helium balloon
  - Dominant hand controls 2D position
  - Non-dominant hand controls 1D height
  - Meant for usage on work bench

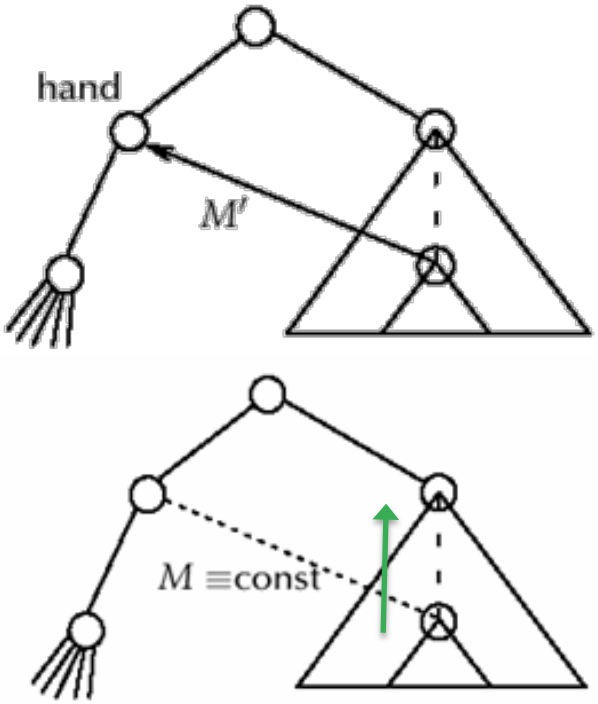


- Implementation:
  - Right/left index finger defines position / height, resp.
  - Both index fingers remain on the table
  - *System control* (e.g., triggers) by contacts in data glove
- Advantage:
  - Decomposition of a 3D tasks in two separate low-dimensional tasks
  - Natural constraint (table, a.k.a. *passive haptics*)



Object Manipulation (Grasping & Moving)

- Another pretty frequent interaction task
- Simple, direct grasping (not a realistic metaphor):
  1. Select object
  2. Trigger grasping (via gesture, speech command, ...)
  3. Wait for collision between hand and any object
  4. Make object "stick" to hand
  5. Trigger release
- How to implement the "sticking"?
  - Either, re-link object to hand node,
  - Or, maintain transformation invariant between hand nd object
  - My experience: with non-trivial applications, re-linking causes trouble!



Taxonomy of Natural Grasping Poses

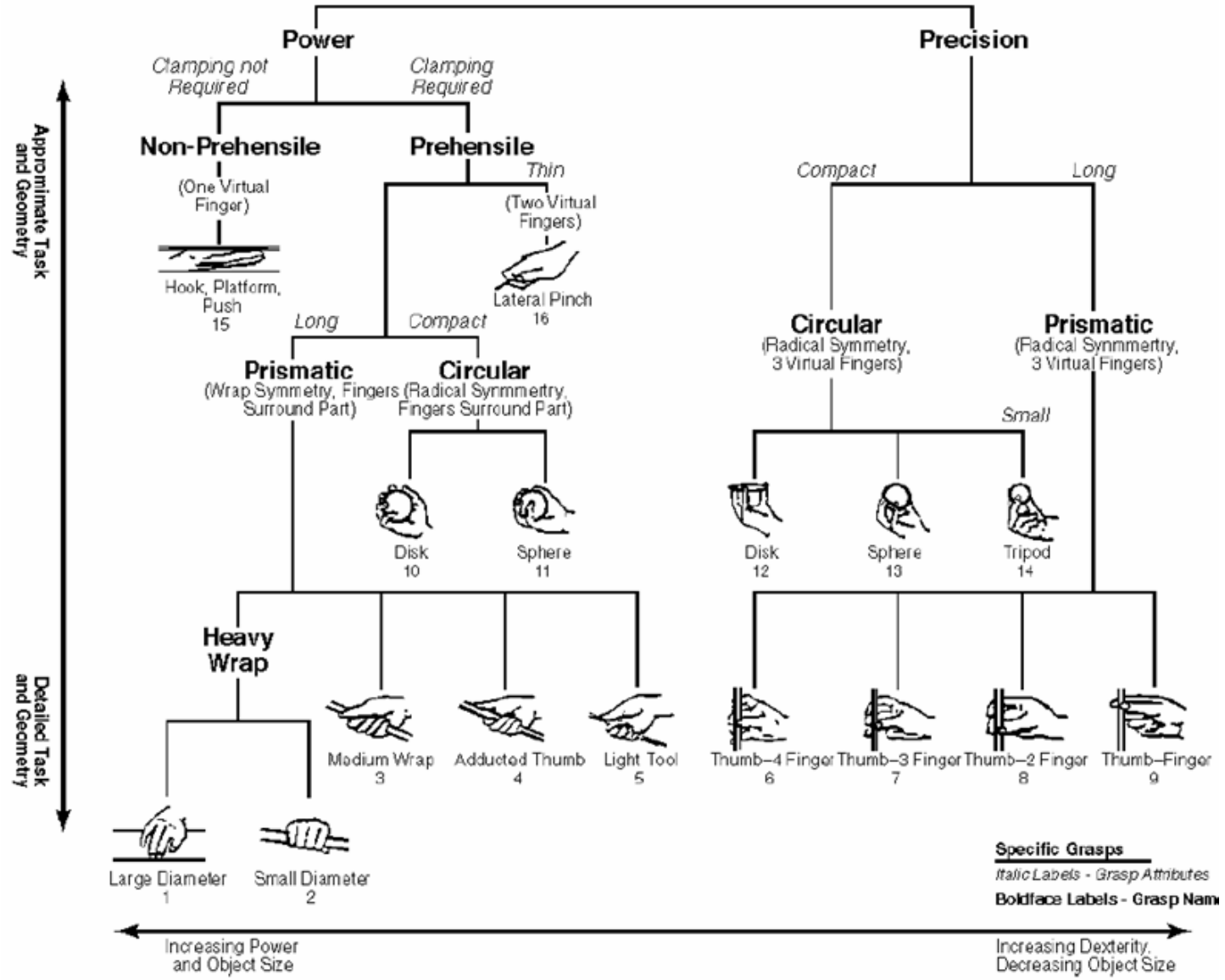


Figure 64. Taxonomy of Manufacturing Grasps

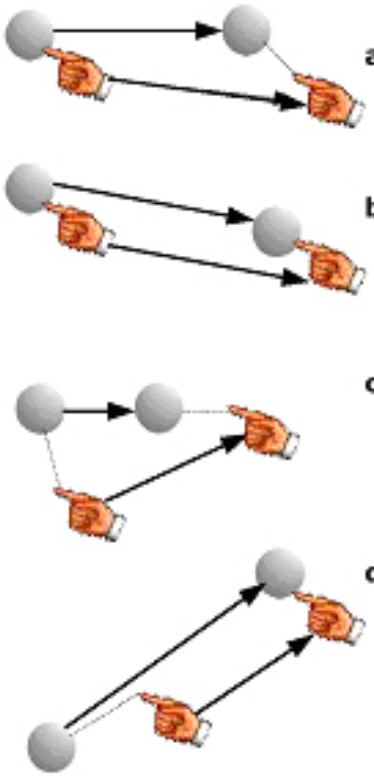
The PRISM Metaphor

- PRISM = "precise and rapid interaction through scaled manipulation"
- Goal: precise manipulation in the near field
- Idea:
  - Detect when the user is trying to be precise and when not
  - Adjust C-D ratio *k* accordingly
- More concretely:
  - Let D_O = translation distance of manipulated object,
    D_H = translation distance of user's hand,
    V_H = average speed of hand during past ½ second,
    S = some threshold;
  - Set distance

$$D_O = k \cdot D_H \qquad k = \begin{cases} 1 & , V_H > S \\ V_H / S & , \min < V_H < S \\ 0 & , V_H \leq \min \end{cases}$$

- Additional idea:
  - Do the scaling independently for each coordinate axis
  - Advantage: helps to move objects exactly along an axis
- Recovery from offsets/drifts:
  - Problem: the positions of hand and object drift apart over time
  - Solution: reduce the offset while the user moves the hand very fast
    - Make the object move even faster
    - During fast movement, the user doesn't notice
- Remark: this technique works almost exactly analogously for rotations
  - Just convert the orientation of the user's hand to axis + angle (see CG1 course), scale the angle, then convert back to rotation matrix

Examples

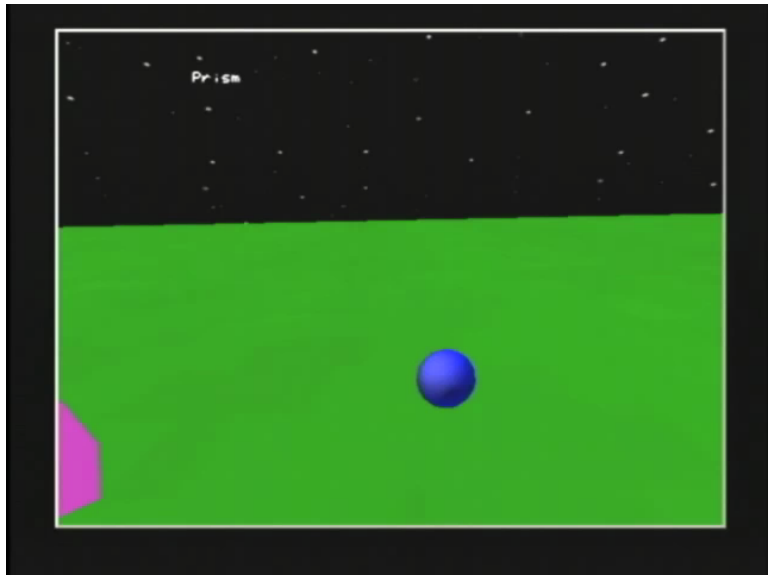


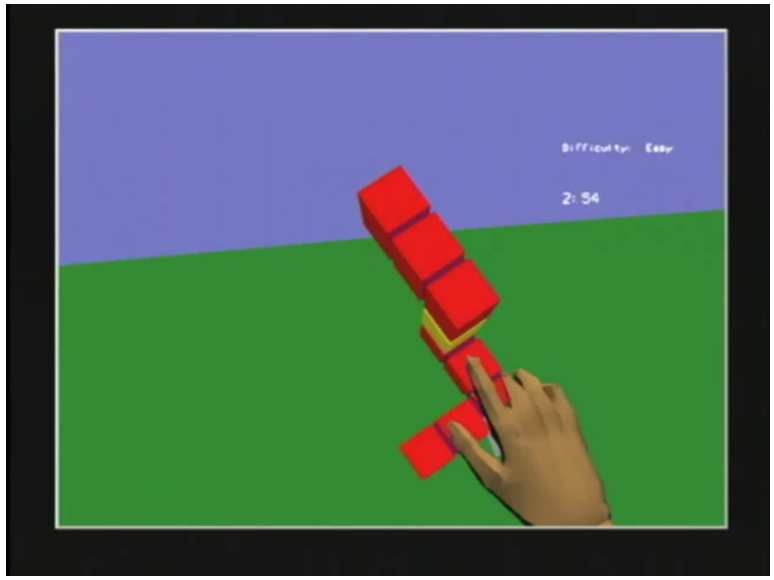
a. User moves hand slowly to the right and down. Some movement is scaled down in the horizontal direction, nearly all is scaled down in the vertical.

b. User moves in the same direction as in (a), this time quickly. Since interaction is in direct mode, object follows hand to its exact location.

c. User slowly moves to the right and up back towards object, vertical offset is recovered. Scaled motion performed in horizontal direction.

d. After accumulating offset, user moves hand quickly up and to the right. Offset is eliminated and mode has switched into direct.





Frees, Kessler, Kay (http://give.ramapo.edu/prism/prism.html)

The Action-at-a-Distance Principle

- Is a general VR interaction design principle
- Example: move remote objects
  - Idea: scale motion of hand such that on the screen (2D) the relative position between the hand and the object does not change
  - Computations:
    1. d_O^t = distance obj — viewpoint at time t
    2. P^t = point on ray S_t with distance d_O^t
    3. d_H^t = distance hand — viewpoint at time t
    4. d_H^{t+1} = distance hand — viewpoint at time t+1
    5. Calculate d_O^{t+1} such that $\frac{d_O^t}{d_H^t} = \frac{d_O^{t+1}}{d_H^{t+1}}$
    6. Calculate P^{t+1} = point on ray S_{t+1} with distance d_O^{t+1}
    7. Translation for the object = $P^{t+1} - P^t$

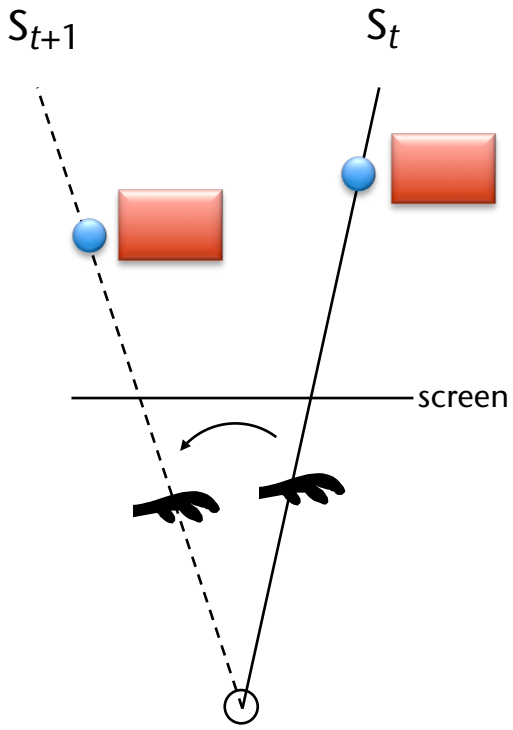
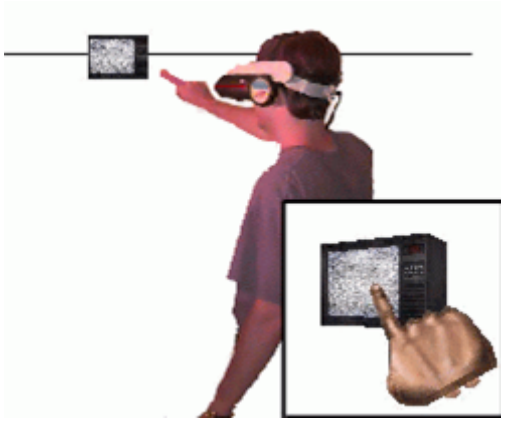
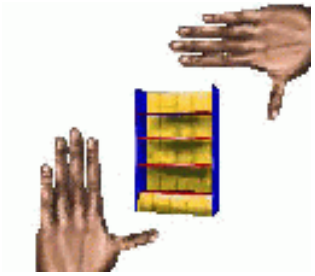


Image Plane Interaction

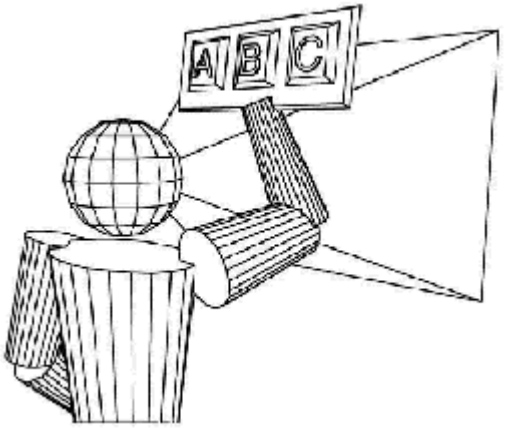
- General idea: user does not interact with 3D objects, but instead with their 2D image
- "Image plane" selection metaphors:
  - Shoot a ray between thumb and forefinger
  - Shoot ray from eye through fingertip
  - "*Lifting palm*"
  - Frame the object with both hands

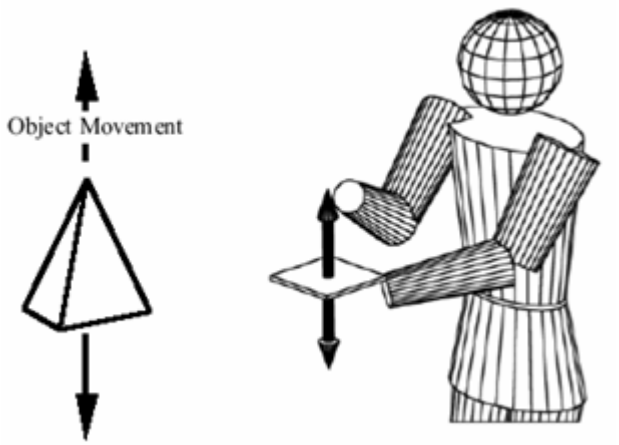




Proprioceptive Interaction

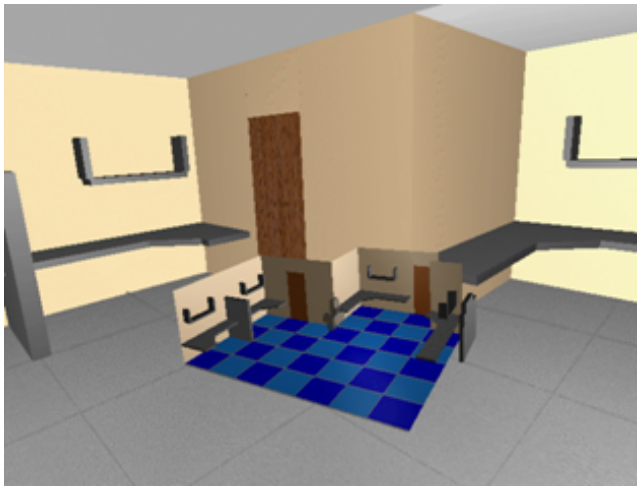
- *proprius* (lat.) = (adj.) *self*
- Idea: utilize the fact that humans know exactly where their limbs are, even with closed eyes
- Metaphors derived from that:
  - "Real pulldown" menus: user reaches up, makes grasping gesture, then pulled his hand down → menu appears
  - Deleting objects: grasp object, throw over the shoulder
- Manipulate remote objects by handheld proxy widgets (= action-at-a-distance)



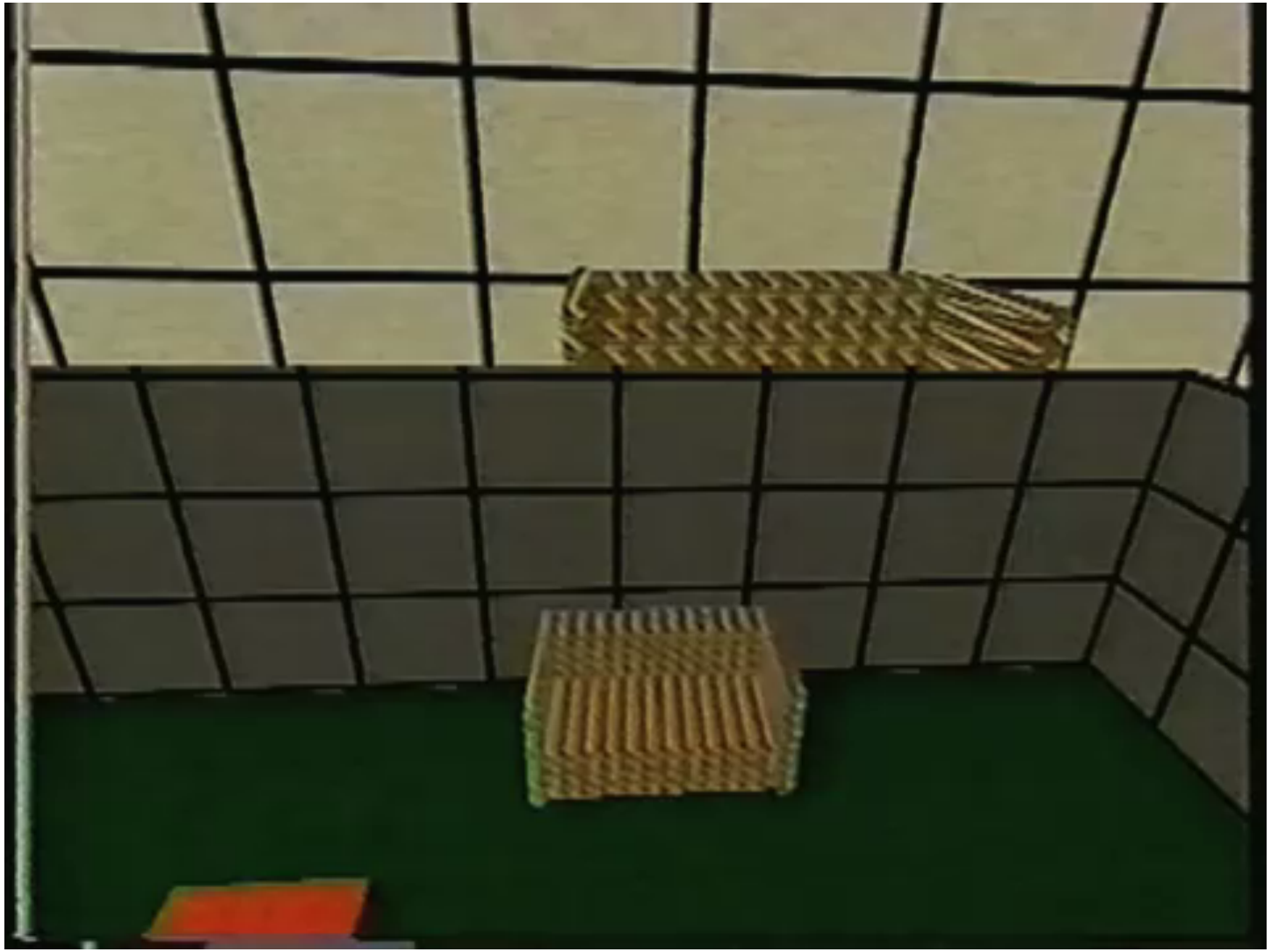


The World-in-Miniature Paradigm

- Idea:
  - Use a 3D miniature "map" (analogously to 2D maps)
    → World-in-Miniature (WIM)
  - All interactions in and with the WIM are mapped to the "real" VE
  - Attach the WIM to the non-dominant hand
- Object manipulation = grasp & move the miniature object in the WIM
- Navigation = move the frustum in the WIM, or select a point in the WIM



Video



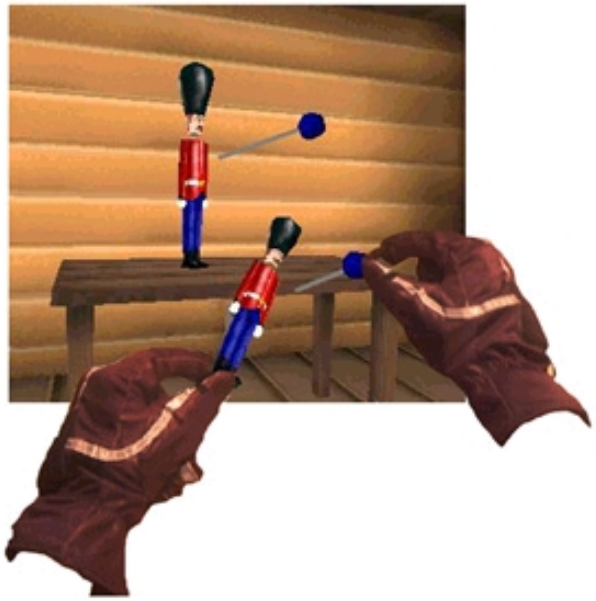
Doug Bowman

Two-Handed Interaction

- Users have 2 hands: a dominant one (usually right) and a non-dominant one (left)
- The roles of each hand:
  - Non-dominant hand = reference coordinate system, positioning of context
  - Dominant hand = fine-skilled motor tasks within that context
- Good metaphors = metaphors that utilize both hands within their respective roles

Example: Voodoo Dolls [1999]

- Technique for remote manipulation / positioning of objects
- Idea: create a temporary copy (= voodoo doll) of the remote object
- Task decomposition:
  - Create copy of the referenced object, attach it to the left hand
    - The original of that object will *not* be moved
  - Create copy of the object to be manipulated, attach it to the right hand
    - The original of that object *will* be moved
  - Movement of the voodoo doll — relative to the *copy* of the reference object — is mapped to the original



- How to create the copy of the object to be manipulated:
  - Use image-plane technique: make pinch gesture "in front" of the object
  - Size of the copy ≈ size of the virtual hand
- How to create a copy of the reference object(s):
  - Use some framing technique (= image-plane technique again)
  - Make copy of all objects within the frame

pinch copy attached to hand

framing by rectangle

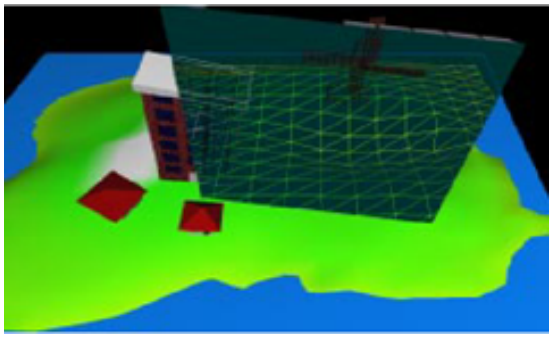
framing by circle

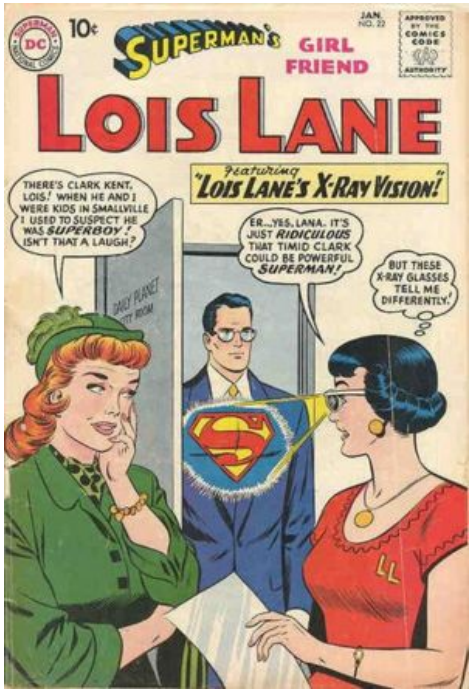
- Example of a manipulation:
  1. User "grasps" table with pinch gesture of the left hand
  2. System creates copy, attaches it to left hand, plus some other "context" objects in the surroundings of that object (e.g., telephone and monitor)
  3. User "grasps" telephone with right hand (pinch)
  4. System creates copy of telephone and attaches it to right hand
  5. User puts *copy* of telephone at some other plce on the *copy* of the table
  6. System maps the translation to the original telephone

- Advantages:
  - Left hand is being used exactly for what it was "designed"
  - User can work on different scales, without having to specify the scaling explicitly
    - The scaling happens implicitly by selection of the reference objects

Magic Lenses

- Idea: user sees a different version of the VE through the magic lens
- Where "different" could mean:
  - Other rendering parameters
  - Other geometry
  - Another viewpoint
  - Different scaling, ...
- Examples:
  - Wireframe rendering
  - Magnification
  - Additional viewpoints (like magic mirror)
  - Geometry beneath the surface
  - Preview window for *eyeball-in-hand* or *scene-in-hand* navigation
  - "X-Ray vision"
- Magic lenses can also be specified by volume





Videos

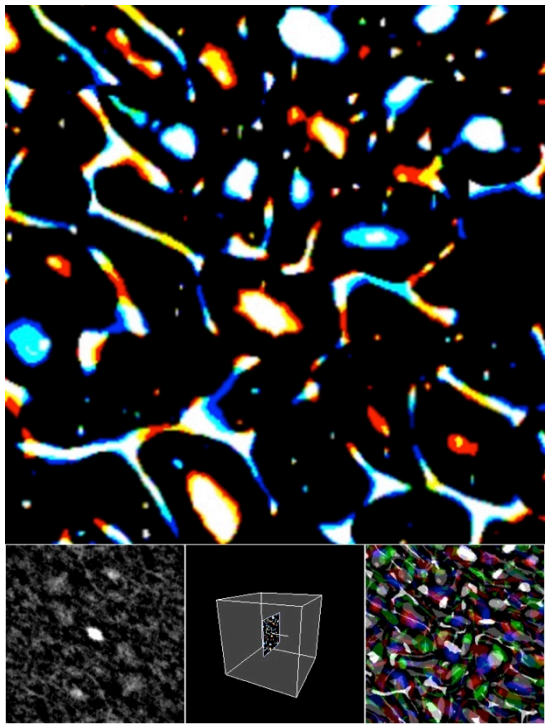

X-Ray Tool
Laying underground cables

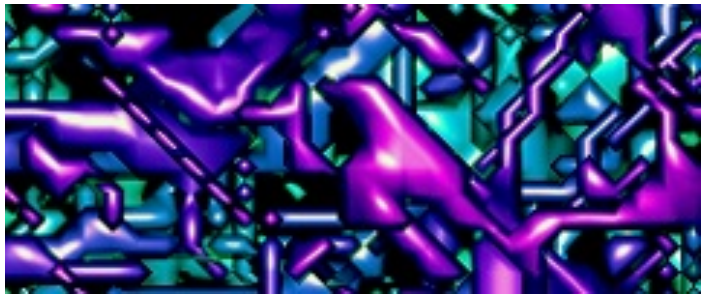

Window Tool
Comparing different versions of a scene


Window Tool
Multiple Views into scene

An Application in Scientific Visualization

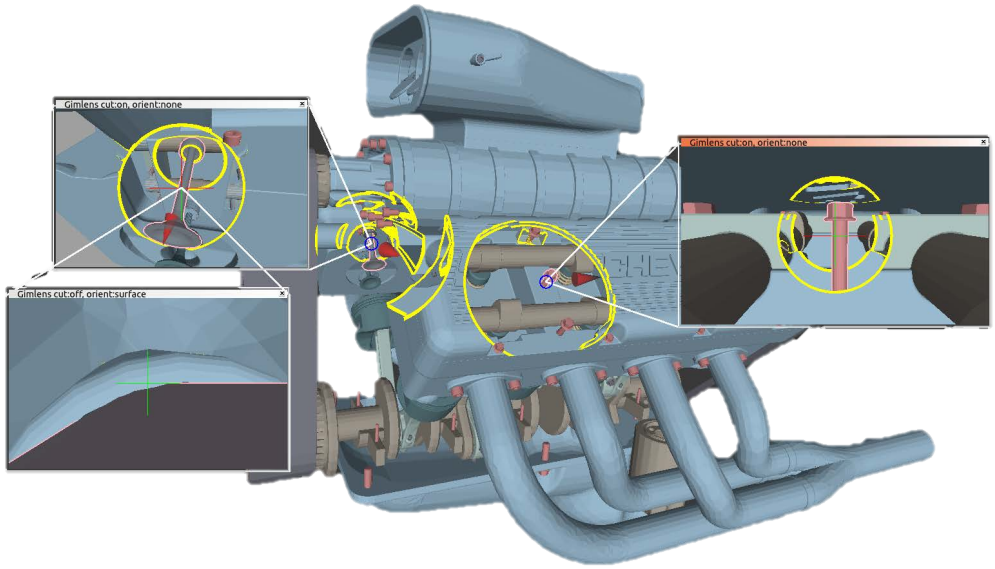
- Task:
  - Visualization of volume data (here, CT) on an iPad
  - Intuitive navigation (= specification of the viewpoint)
- Solution: regard the iPad as a "magic lens" into the VE



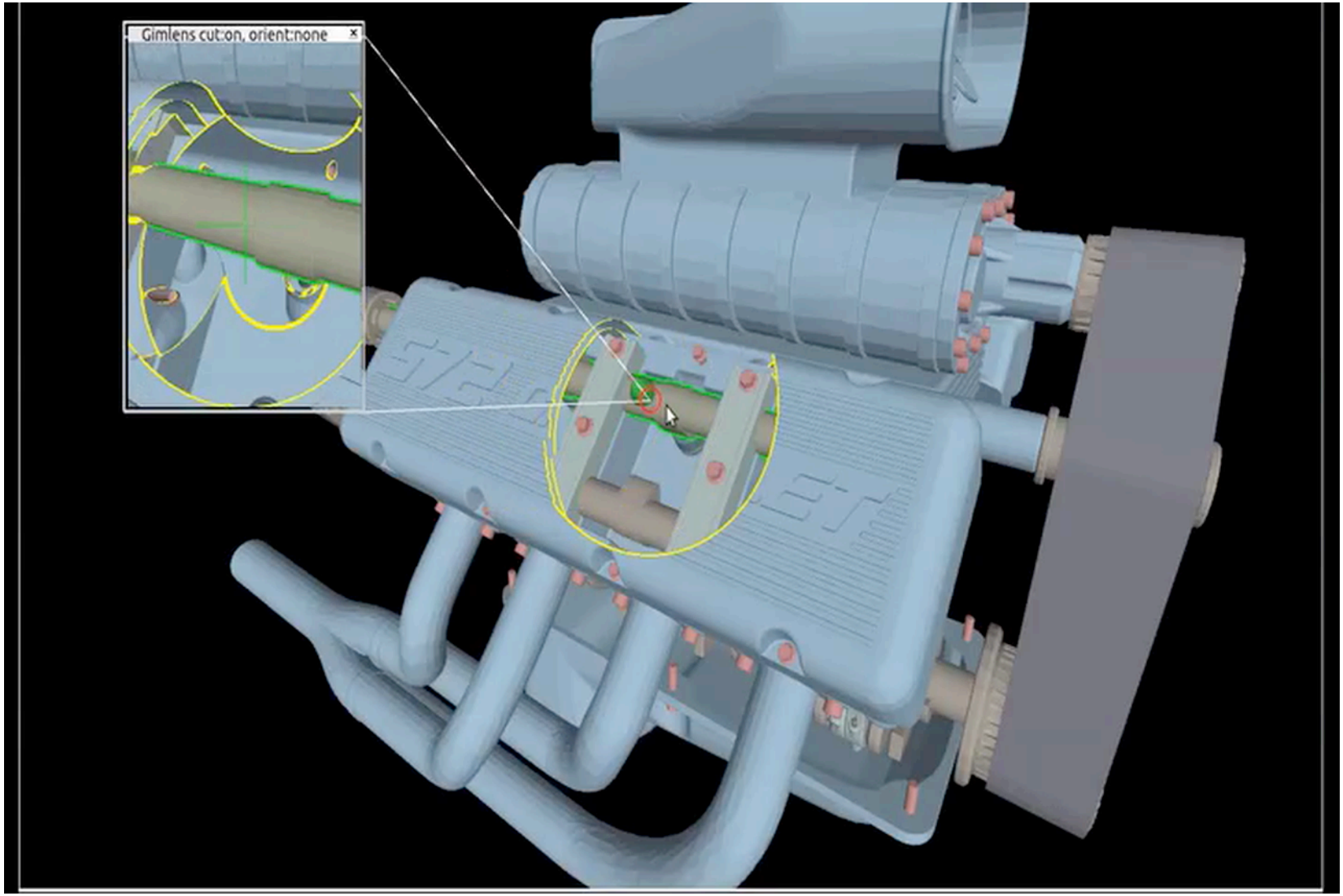


Gimlenses: an Application in CAD Visualization

- *Gimlens* = magic lens to specify cut-aways
- Positioning by cone-shaped proxies
- Cut-away = truncated cone



- Implementation: fragment shader that tests fragments against the cone

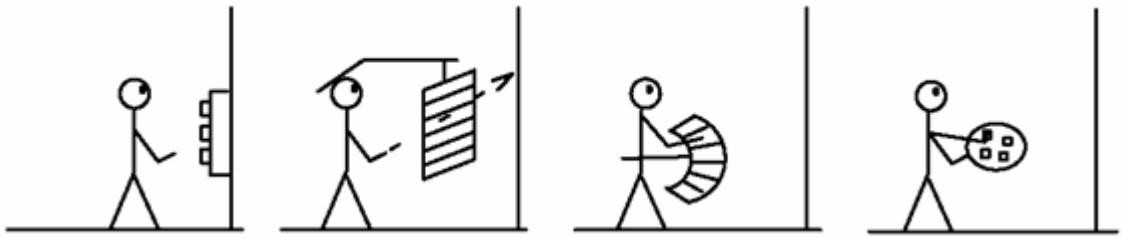

Gimlens cut:on, orient:none

System Control

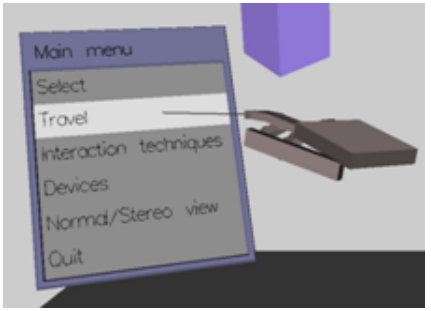
- The 3rd big category of interaction tasks in VR:
  - The somewhat unbeloved child in the VR interaction community
  - The general task of these interactions: change the system's state
    - and everything else that doesn't fit anywhere else
- Consequence: a taxonomy is almost impossible
- Typical techniques:
  - Menus
  - Speech recognition
  - A set of gestures (for 1-out-of-n triggers)
  - Physical devices

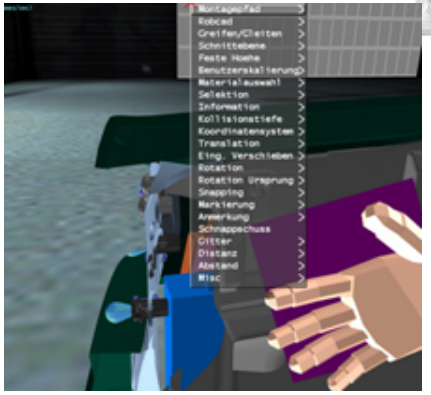
Menus

- Task decomposition:
  1. Bring up menu
  2. Navigate the menu
  3. Select an item
- A possible taxonomy should contain:
  - Input modalities: gestures, speech, buttons, ...
  - Positioning of the menu
  - Selection of items
  - Dimension and shape of the menu
- Examples for positioning the menu:

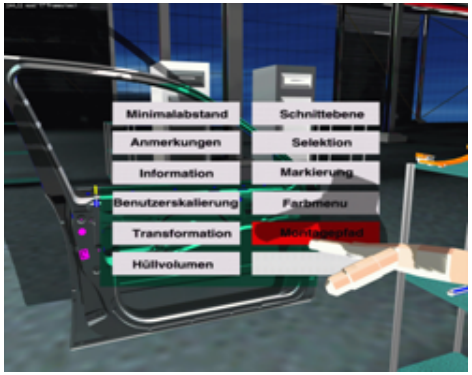


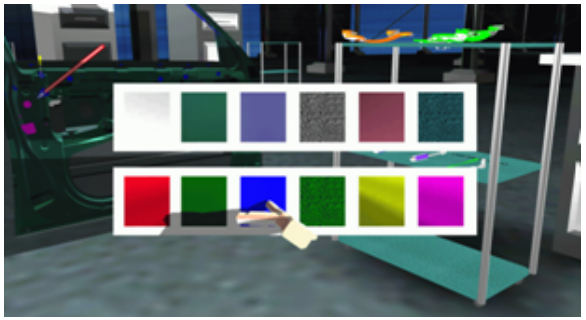
Examples

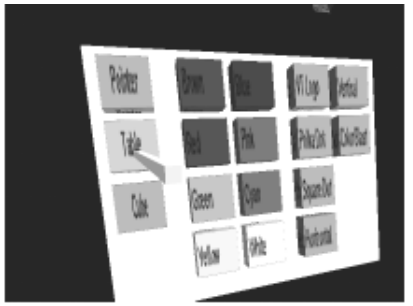




- Embedded in 3D or
  2D overlay (heads-up)
- Item selection: one of the earlier selection techniques, e.g., ray casting or occlusion technique, or map relative hand motion to "active" menu item
- Positioning:
  - Fixed in 3D,
  - Heads-up (moves with head),
  - Attached to left hand, ...

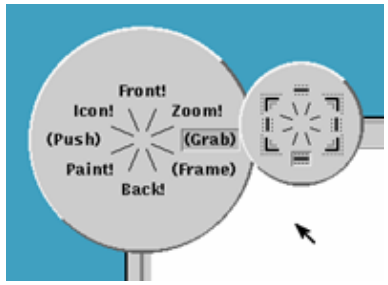


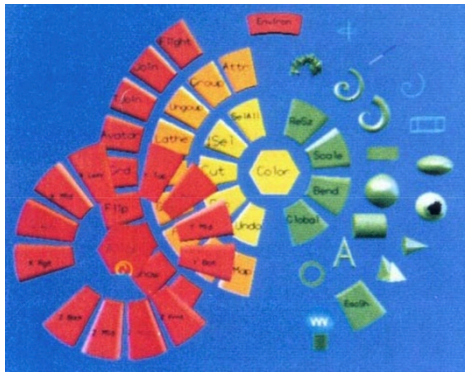


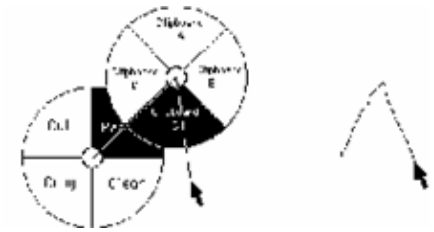


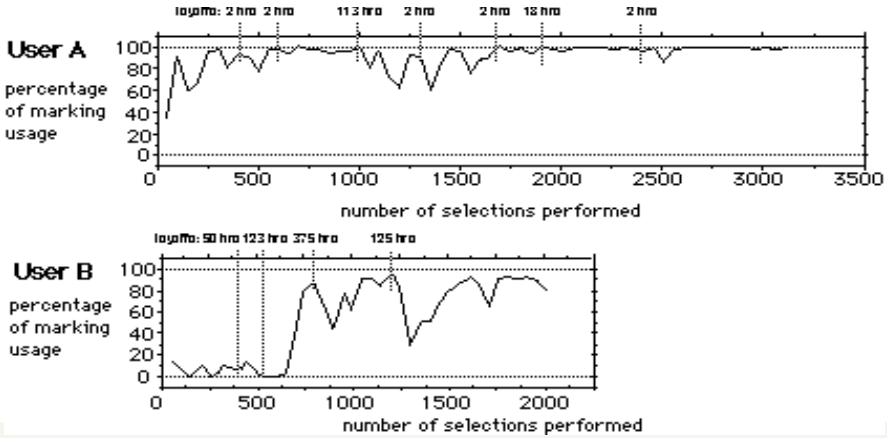
"Marking Menus" (a.k.a Pie Menu)

- Idea: arrange menu items around as center (in a circle, square, ...)
- With menu trigger: position its center at current pointer position
- Advantage:
  - Smooth transition from novice mode to expert mode
  - Experts can navigate the menu "blindfolded"
- Mouse gestures (marks) are much more efficient than a menu:

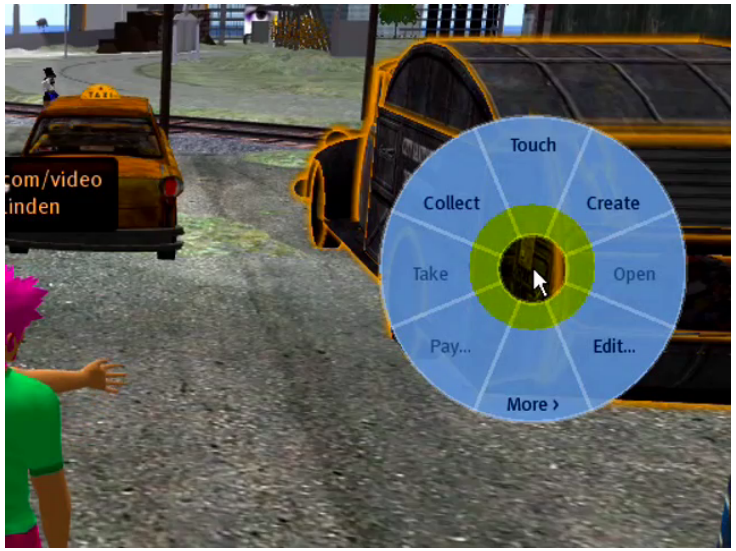






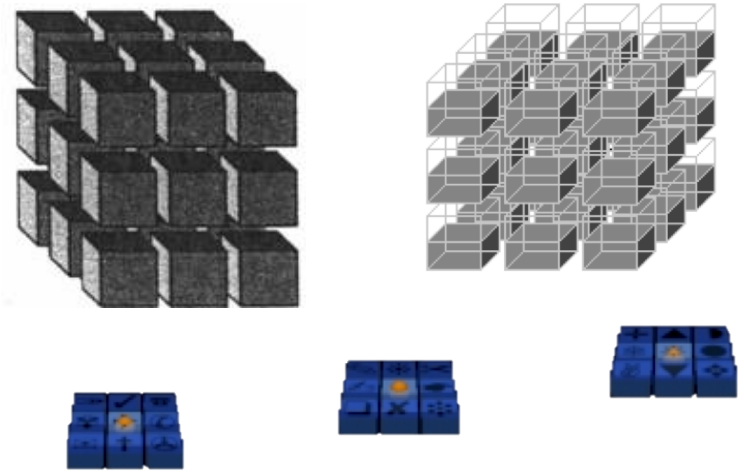


- Video (2D):

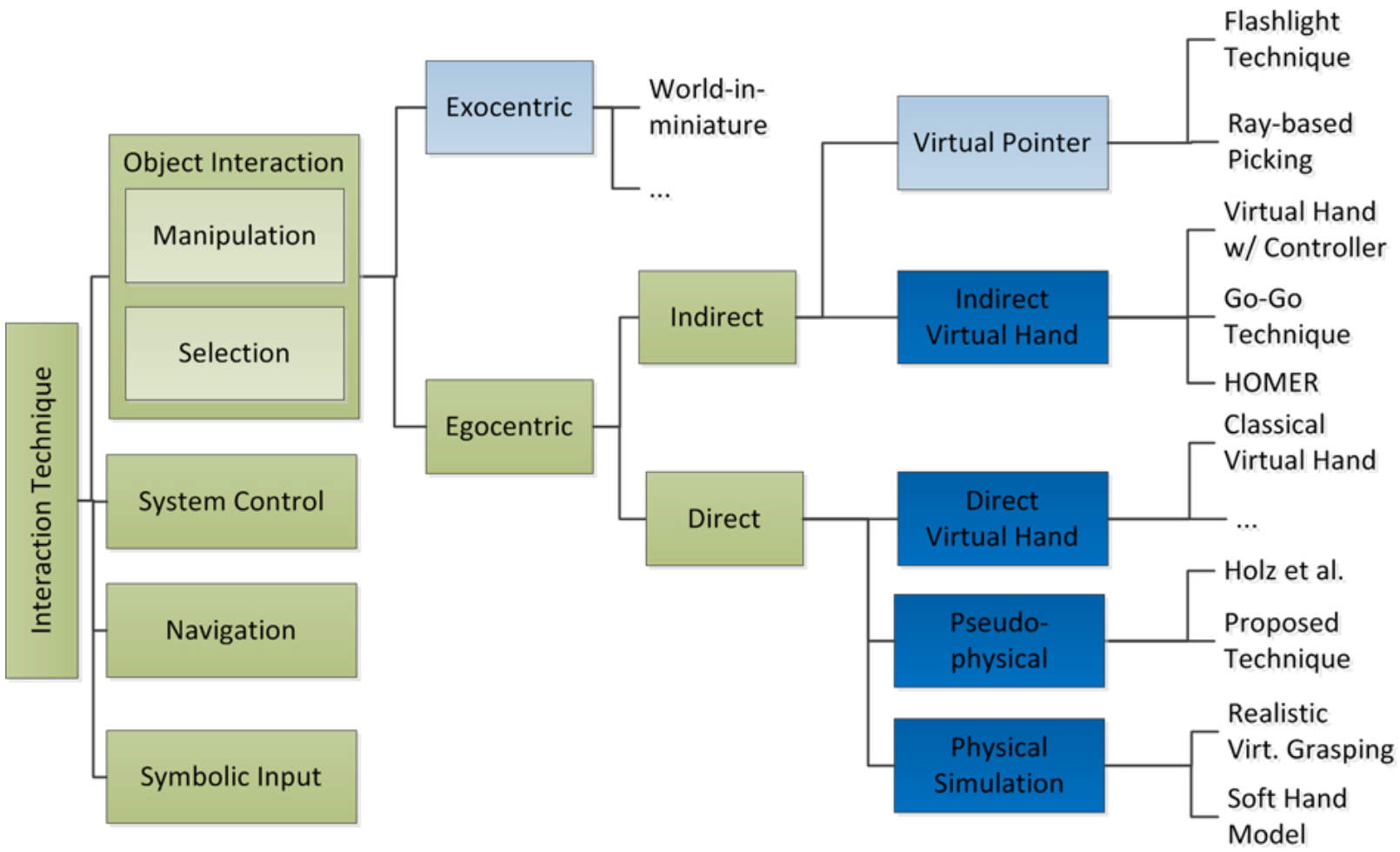


[SecondLife]

- In 3D?
  - Direct "translation"→ "control cube"
  - Not too successful
  - Can you do it better?

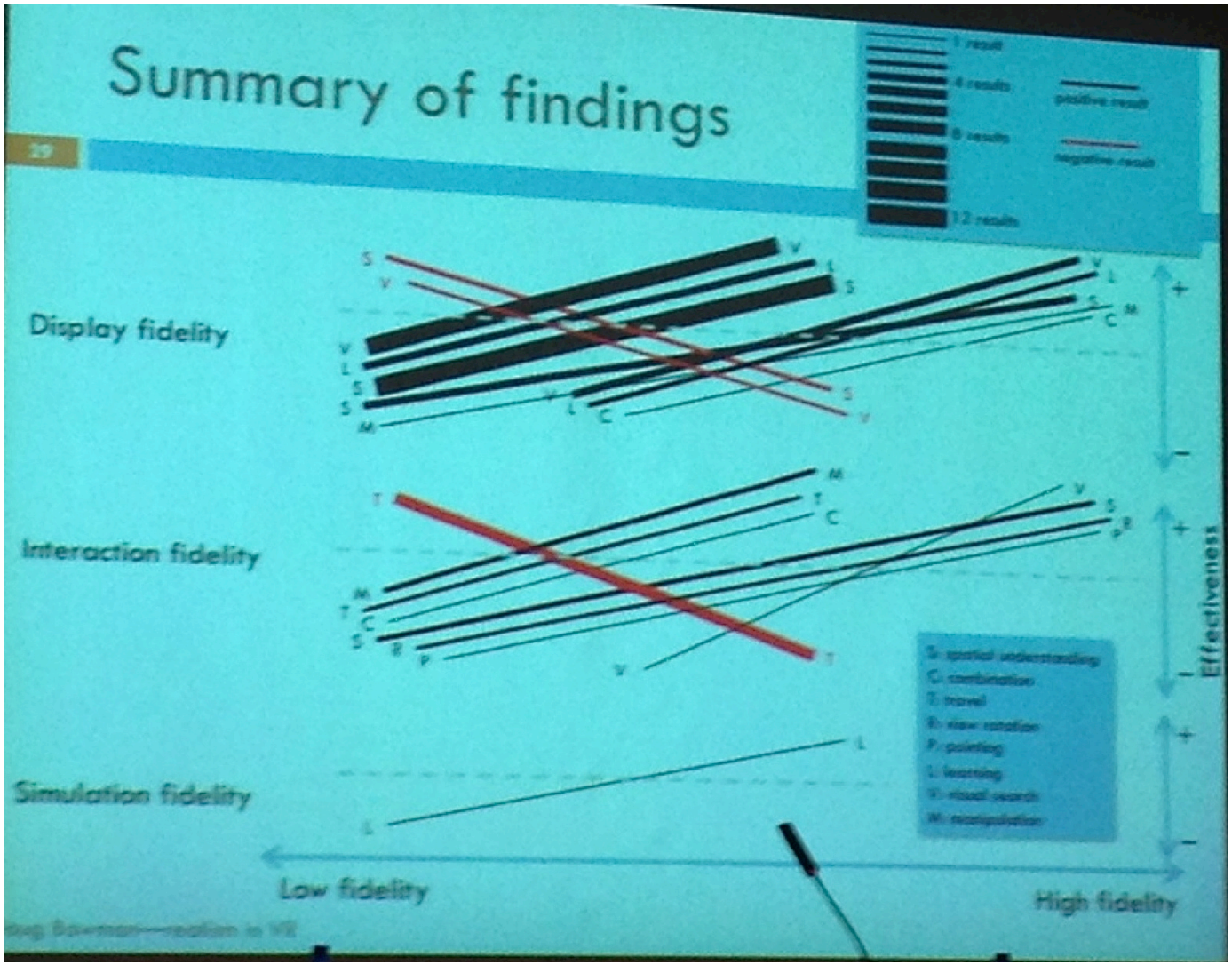


A Taxonomie of "All" Interaction Techniques



Mathias Moehring: Realistic Interaction with Virtual Objects Within Arm's Reach (Diss.; aufbauend auf Bowman's Taxonomie)

Is More Fidelity Always Better?



Doug Bowman, JVRC'12

- Higher (interaction) fidelity often results in higher effectiveness
- Increasing fidelity does not always improve user effectiveness within a virtual environment (it does not decrease it either)
- Very few cases where higher fidelity is detrimental
  - Travel techniques are one strong case for *less* fidelity
- Best cases for high fidelity:
  - Difficult and complex visuo-spatial tasks
  - Learning / training
  - High-DOF interaction tasks

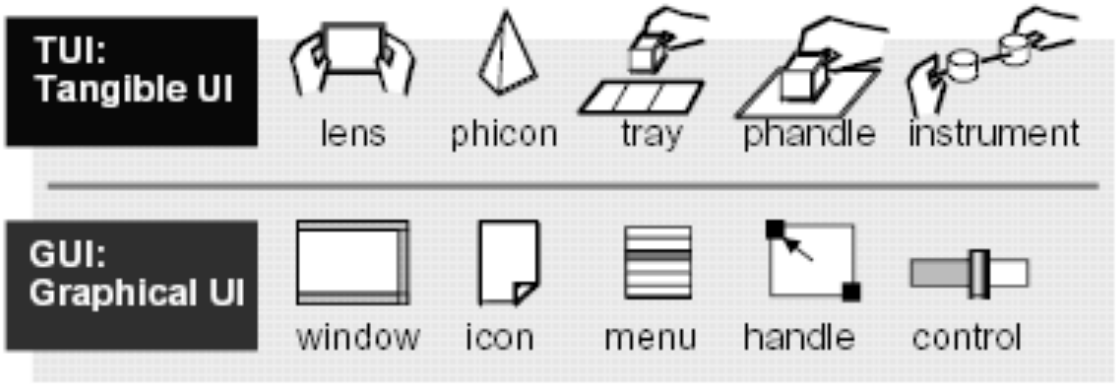
Tangible User-Interfaces

- Idea: instantiate virtual/abstract interaction metaphors (handles, icons, sliders, ...) by physical objects

- **Defintion Tangible User Interface (TUI):**
  An attempt to give physical form to digital information, making "bits" directly manipulable and perceptible by people.

  Tangible Interfaces will make bits accessible through
  - augmented physical surfaces (e.g. walls, desktops, ceilings, windows),
  - graspable objects (e.g. building blocks, models, instruments), and
  - ambient media (e.g. light, sound, airflow, water-flow, kinetic sculpture).

- Analogies between GUIs and TUIs:



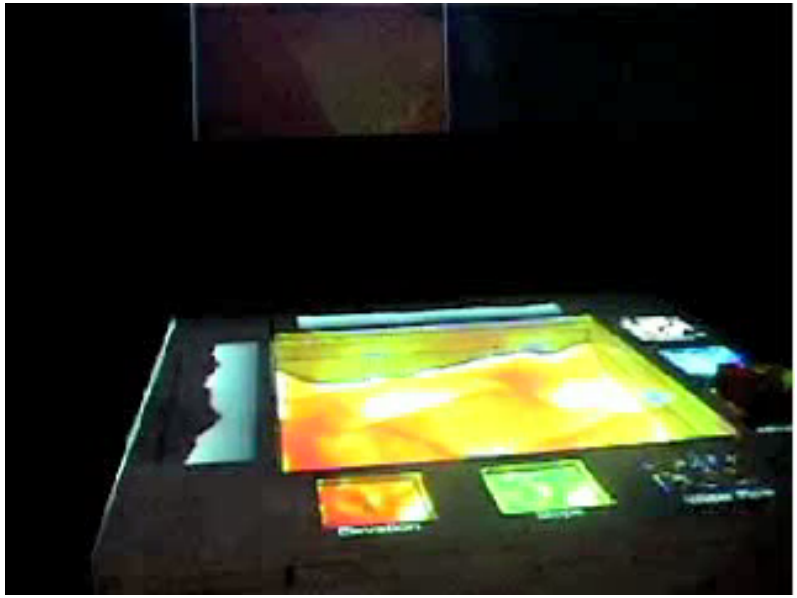
- Examples:

Tangible Magic Lens

Tangible Slider

More (Artistic) Examples

Sandscape
(sand as terrain)



http://tangible.media.mit.edu

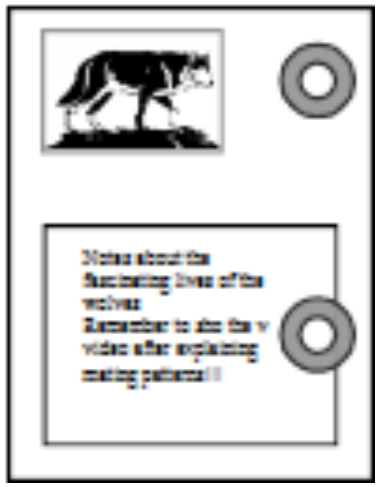
GranulatSynthese
(interactive art installation)

http://imve.informatik.uni-hamburg.de/projects/GranulatSynthese

IP Network Design Workbench
(use pucks for manipulation
of nodes and edges)

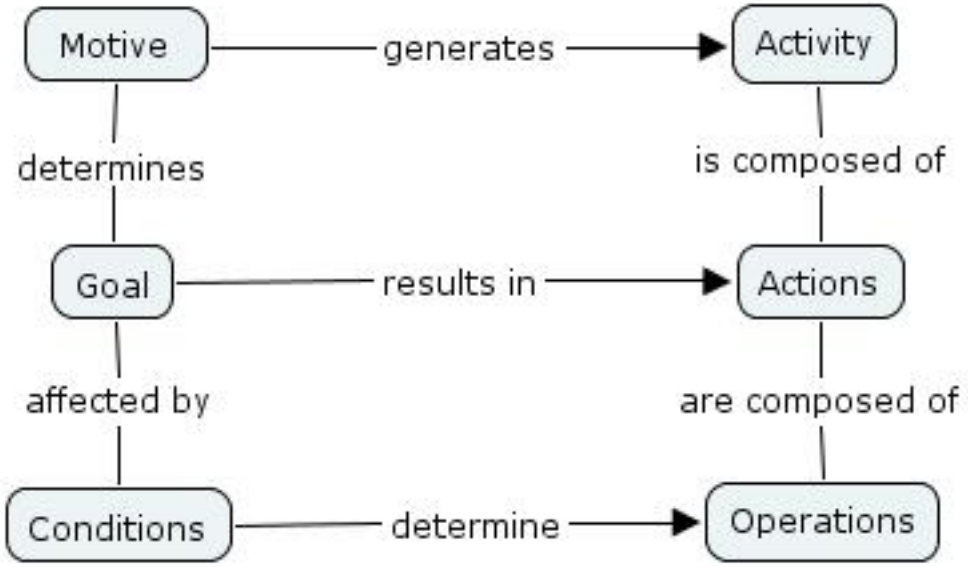
http://tangible.media.mit.edu/projects/ipnet_workbench

Palette & PaperButtons



Activity Theory

- A framework of terminology (not theory in the sense of natural sciences):



- Gains popularity in HCI as a tool