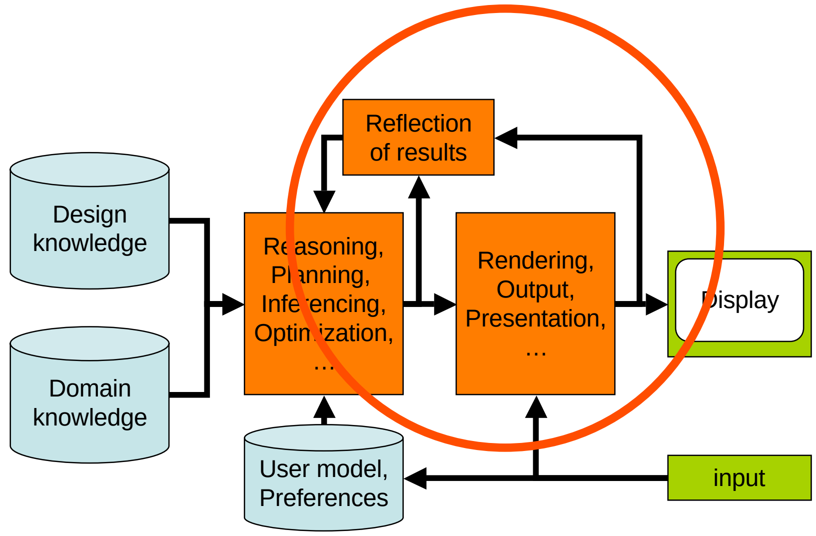
Smart Graphics: Labeling, Feedback Loops

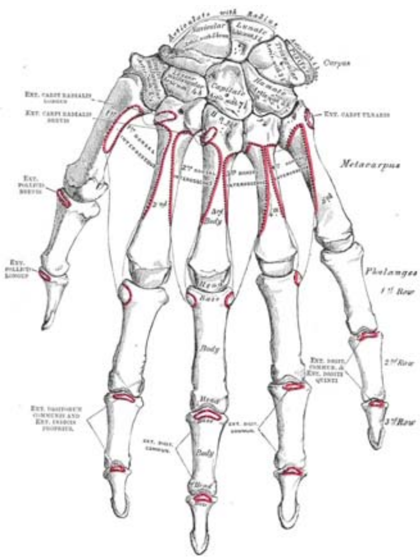
Vorlesung „Smart Graphics”

Themen heute

- Reasoning/Rendering:
  - Floating Labels
- Reflection:
  - Anticipation Feedback loops
  - Putting the human in the loop

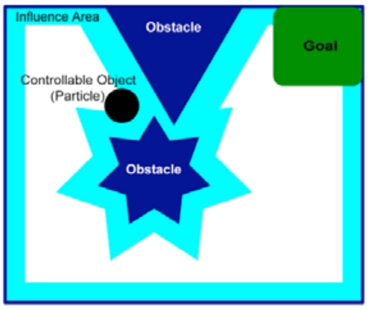


Floating Labels [Hartman, SG 2004]



- Generalization of force-driven graph layout
- Application for labeling medical images
- Goal: fully label a given set of objects
  - Internal labels
  - External labels
  - Connecting lines

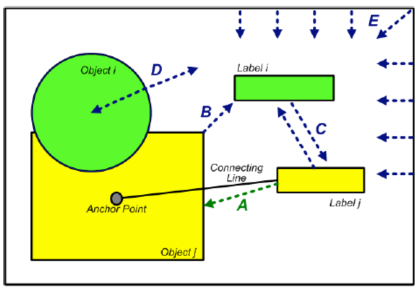
Idea: computing a potential field



goal	**Attractor:** Target
B_i	**Repulsors:** Obstacles
ρ_0	Spatial influence of obstacles
p	Position
$\rho_{goal}(p)$	Minimal goal distance
$\rho_i(p)$	Minimal distance to obstacle B_i

- Reformulate objective function in terms of forces
- Sum of attractor and repulsors: **Potential**
- 1st Derivative: **Field**
- Relaxation: redirect **particles** according to force field

Force Configuration



A : Label and co-reference object
B : Label and all other objects
C : Between labels
D : Object boundary
(internal or external label)
E : Image boundary

attractive

repulsive

Force Configuration

Force A: Label and co-referential object

$$U_{attr}(p) = \begin{cases} 0 & , p \in area(O) \\ c_1 \frac{\rho_O}{\eta} & , p \notin area(O) \end{cases}$$

ρ_O distance to *interia*(O)
η maximal distance

Force B: Label and all other graphical objects

$$U_{rep}(p) = \begin{cases} c_2 & , p \in area(O_i) \wedge O_i \neq O \\ 0 & , else \end{cases}$$

Force D: Object boundary

$$U_{silh}(p) = \begin{cases} c_3 & , \rho_{silh} \leq \rho_S \\ 0 & , \rho_{silh} > \rho_S \end{cases}$$

ρ_{silh} distance to silhouettes
ρ_S silhouette influence factor

Force E: Image boundary

$$U_{wall}(p) = \begin{cases} c_4(1 - \frac{\rho_{wall}}{\rho_W}) & , \rho_{wall} \leq \rho_W \\ 0 & , \rho_{wall} > \rho_W \end{cases}$$

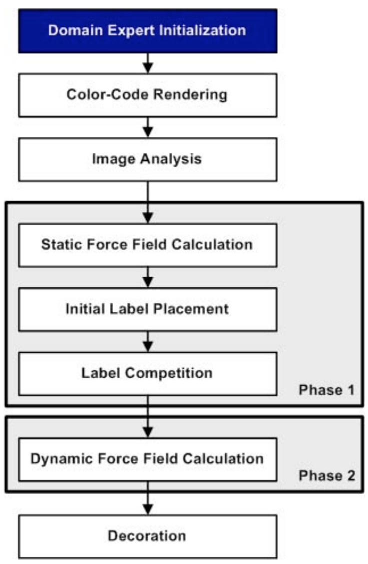
ρ_{wall} distance to boundary
ρ_W boundary influence factor

Static Potential:

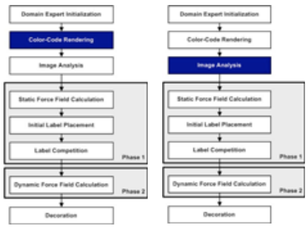
$$U(p) = U_{attr}(p) + max(U_{wall}(p), U_{silh}(p), U_{rep}(p))$$

Initialization

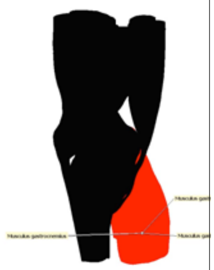
- Assign texts to objects
  - Done by human
- Assign initial placements for labels
  - Simple methods



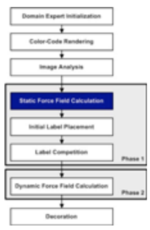
Color Code Rendering



- Render each object in a specific color
- Easy determination of visible areas of an object
- **Internal Point:** Thinning / Erosion
  - Max. distance to silhouette
  - Inside thickest region

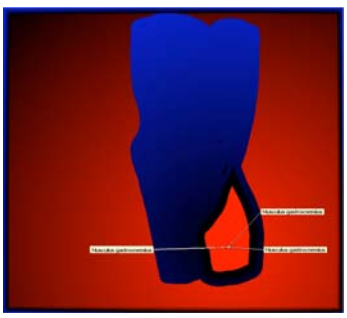


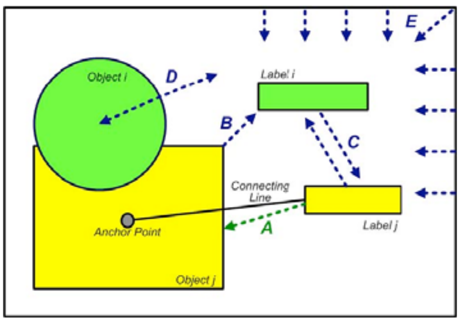
Static Force field calculation



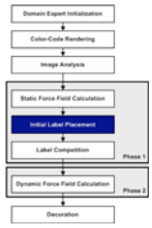
- **Attractive:** reference object
- **Repulsive:** Other graphical objects
- **Repulsive:** Silhouettes and boundary
- **Color-Code:**

repulsive	neutral	attractive
blue	black	red

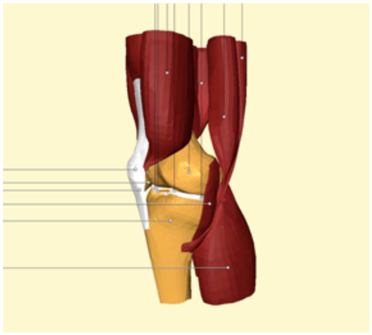




Initial Label Placement

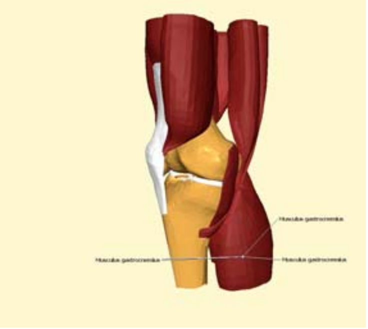


- Determine label candidate positions
  - Initial label positions (corners, preferred direction, interia)
  - Label movement (Point abstraction)
  - Expand point abstraction
    - Area containing point
    - Minimize area potential



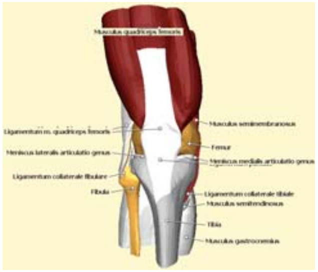
Label Competition

- Evaluate label candidates:
  - Area potential
  - Visibility (label overlap)
  - Length of connecting lines
  - Angle with main axis
- Min. and max. values for labe candidates
- Weighted sum of all components

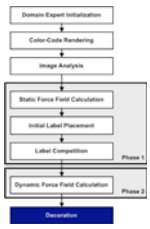


Dynamic Force Field Calculation

- Goal: remove collisions
- Assumption: bigger objects are easier to label
- Alter static force field
- Greedy algorithm:
  – Pin most problematic label (smallest object)
  – Let other labels float
- Relaxation
- Evaluate

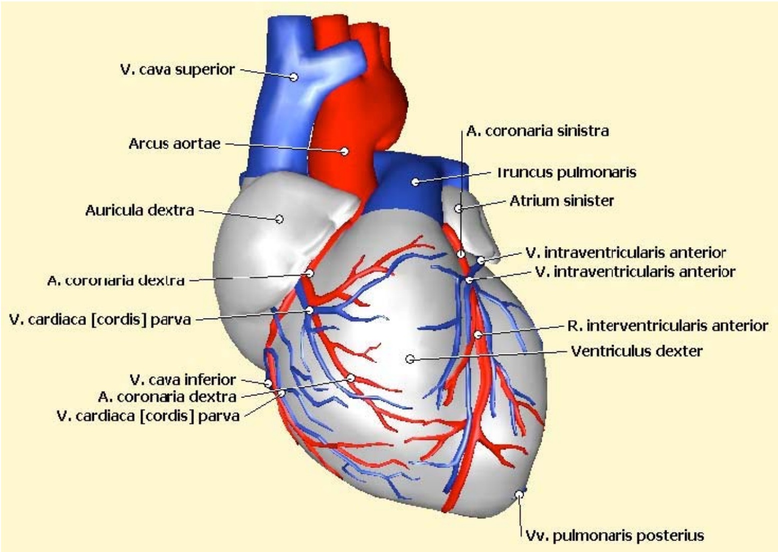


Decoration

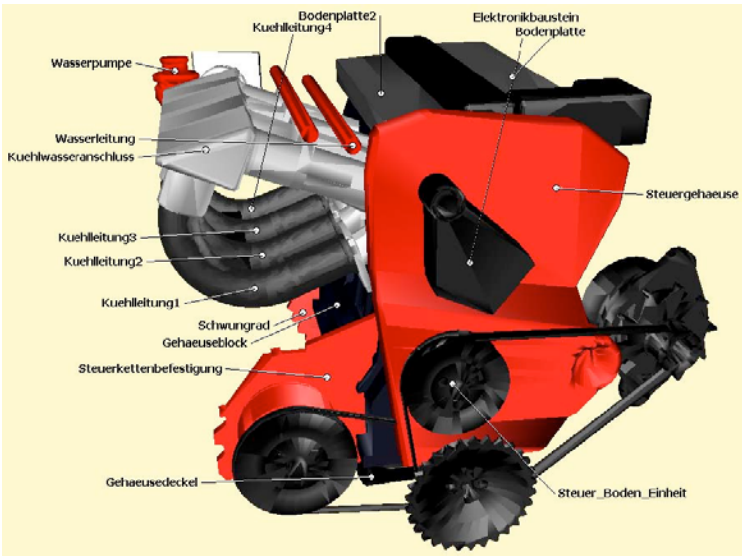


- **Layout of Metagraphical Objects**
- Object-Ground differentiation
  – Object: Metagraphical objects
  – Ground: Graphical objects
  – Styles
- **Anchor Points:** +/-
- **Connecting Lines:**
  – Solid / Dashed
  – Line color vs. Background

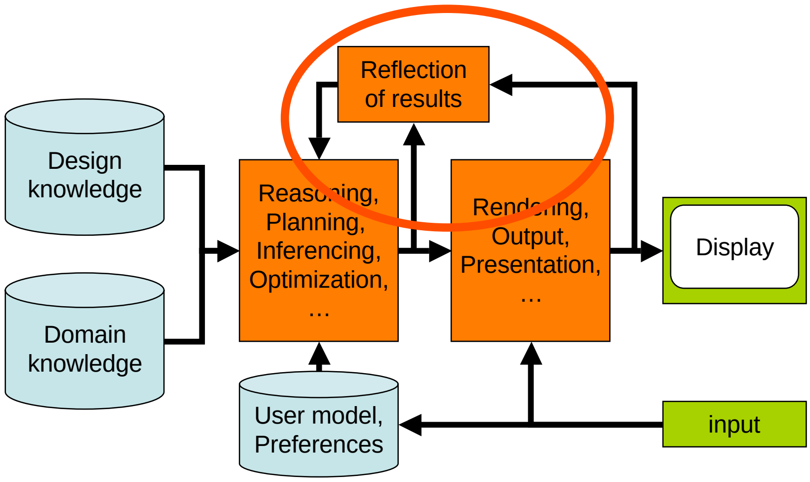
Example (15/23 Objects)



Example (20 Objects)



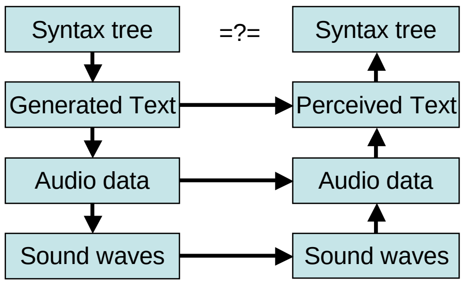
Anticipation-Feedback Loops



Anticipation-Feedback Loops

- Original idea in natural language generation [Wahlster 1982]
  - Generate an (elliptic) expression
  - Try to anticipate what the user will understand
  - If this is correct → OK, Else: generate new expression
- Generalization to other modalities:
  - Generate output (done anyway!)
  - Apply analysis to it (but how? This is the tricky part!)
  - Compare the result with the intended effect
- Can be applied at different levels
  - Structural level (before rendering)
  - Output level (after rendering)

AFLs at different levels



Questions answered:

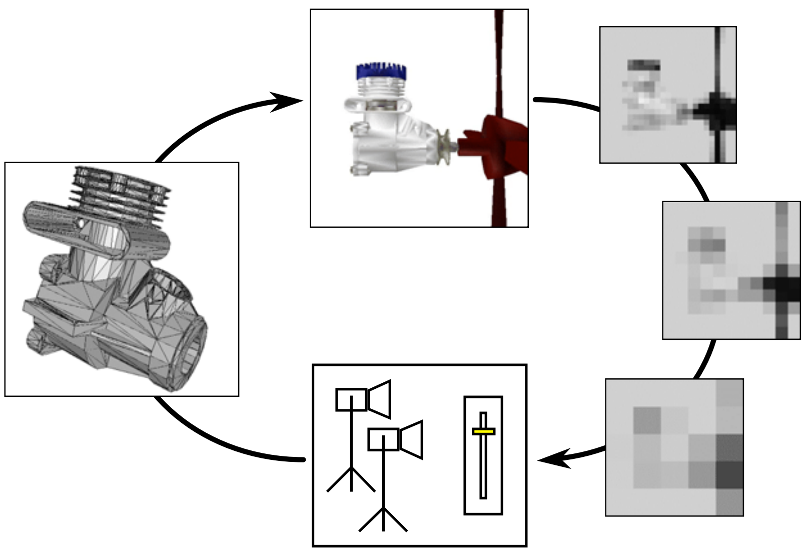
Is the text correct?
No homonyms?

Are the sounds correct?
No homophones?

Is the sound played correctly?
Are the speakers OK?

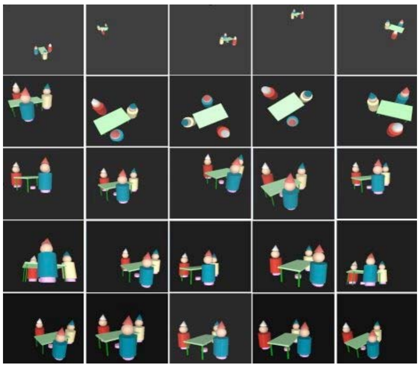
- AFLs can be used at different levels for different purposes
  - Example above is for language generation
- Assumption: the analysis process is perfect
- Problem if generation and analysis have the same bug

Example: AFL in Cathi



Example: CamPlan [Halper 2000]

- 2-part system
  - evaluation tool for specified visual communicative goals
    - specification issues
    - high-level composition
  - camera state generator using Genetic Alg.
    - convergence issues
    - optimization strategies



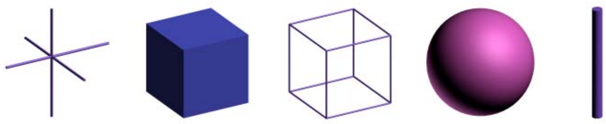
Selection of Presentation Method

- Aimed to establish a set of properties for a variety of applications
  - sufficiently flexible to construct high-level communicative goals
- Created taxonomy of shot objectives based on
  - shot types (position, size, orientation, visibility)
  - reference frames (viewplane, viewport, viewpoint)
  - quantative character (absolute, relative)

```
SizeRelativeViewport ObjectA 50 10 ObjectB
```

Specification Problems

- Which best reflect how big an object appears?



LengthX/Y	0.5	0.5	0.5	0.5	0.02
Area	2%	50%	2%	50%	2%
BoundSphere	0.6	0.6	0.6	0.5	0.5

High-Level Shot Composition

- Templates work over spatially similar scenes

  - Use base properties to define high-level compositional shots
    - classify spatial properties of a scene to instantiate the template
    - understand cognitive and perceptual issues

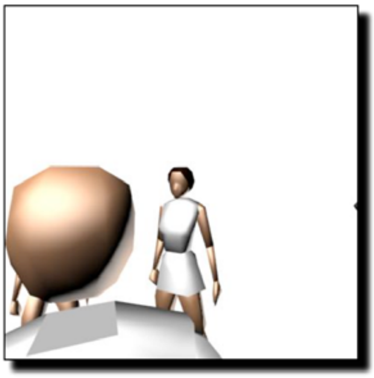
Composition Issues

CenterShot "woman01 head“
(PersonTalking and PersonTalkedTo heads entirely inside screen)
PositionViewplaneBetweenX/Y "woman01 head" -1.0 1.0
PositionViewplaneBetweenX/Y "man01 head" -1.0 1.0

(PersonTalking head and PersonTalkedTo head fully visible)
VisibleInViewport "woman01 head" 100.0 0.0
VisibleInViewport "man01 head" 100.0 0.0

(PersonTalkedTo should be overlapped by PersonTalking but not with his head)
OccludedInViewportBy "woman01" 100.0 99.0 "man01"
OccludedInViewportBy "woman01" 0.0 0.0 "man01 head"

(PersonTalking cut at torso, head quite big and level with PersonTalkedTo head, and in front of PersonTalkedTo)
PositionViewplaneBetweenY "torso" -1000.0 -0.5
PositionRelativeY "woman01 head" x i "man01 head"
SpanXorY "man01 head" 1.0 0.4
PositionRelativeZ "woman01" b b "man01"

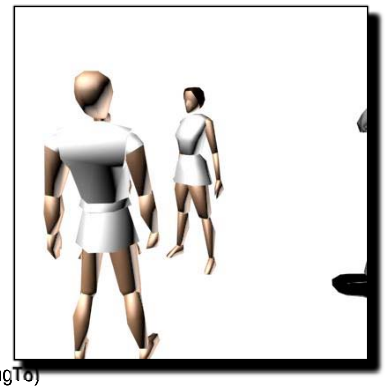


Composition Issues

CenterShot "woman01“

(PersonTalking and PersonTalkedTo entirely inside screen)
PositionViewplaneBetweenX "woman01" -1.0 1.0
PositionViewplaneBetweenY "woman01" -1.0 1.0
PositionViewplaneBetweenX "man01" -1.0 1.0
PositionViewplaneBetweenY "man01" -1.0 1.0

(PersonTalking and PersonTalkedTo mostly visible, with heads totally visible, and PersonTalkedTo not occluded by PersonTalking)
VisibleInViewport "woman01" 90.0 10.0
VisibleInViewport "woman01 head" 100.0 0.0
VisibleInViewport "man01" 90.0 10.0
VisibleInViewport "man01 head" 100.0 0.0
OccludedInViewportBy "woman01" 0.0 0.0 "man01"



(PersonTalking large on screen and in front of PersonTalkingTo)
SizeViewplaneLengthY "man01" 2.0 0.5
PositionRelativeZ "woman01" b b "man01"

Evaluation of Shots

- Map the satisfaction of the shot property to range 0.0 to 1.0
- How to evaluate a partial satisfaction?
  - Perceived satisfaction vs. evaluated satisfaction
  - Unweighted vs. Weighted properties

Adjust camera state

- Make an analysis of different shot specifications according to certain criteria
  - Actual and Perceived
- Evaluation of partial satisfaction of shots
  - how does this affect choice of best partial satisfaction
  - how does this affect convergence towards best camera state solution
  - under what criteria should we modify the scene
- Automatically applying model of aesthetics for a shot
- Use Genetic Algorithm to choose new shots

CamPlan Summary

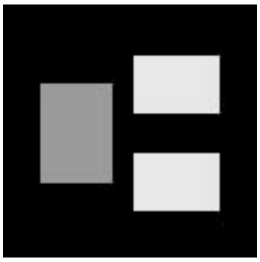
- Feedback loop at the structural level
  - Purely geometric calculations
  - No actual rendering needed
- Optimization by maximizing the satisfaction of the initially given constraints

Example: visual balance in layout [Lok 2004]

- Visual balance:
  - Symmetric, asymmetric
    - Big light objects can balance small dark ones
  - radial
  - Crystallographic

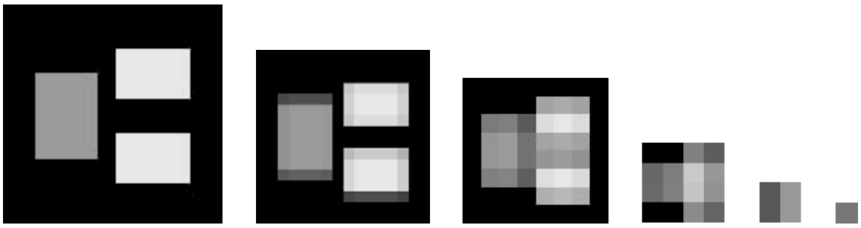
Approach: balancing visual weights

- Analyze a layout for regions of different visual weights
  - Dark colors weigh more
  - Light colors weigh less
- Data structure: weightmap
  - For each pixel, store visual weight
  - Weight of images = mean or median grey value
  - Weight of text depends on font etc.
  - Range 0 (light) – 255 (heavy)
  - Can be shown as greyscale img. →



Weightmap pyramids

- Treat weightmaps like greyscale images
- Apply image pyramid technique to it
  - Simplify image stepwise by interpolating pixels
- Goal: reduce data for computation



Evaluating Balance

- Construct a weightmap pyramid $P_0 \ldots P_n$
- Select an image P_i for $i \approx n$
- Compute the gradients of row+col sums

$$r(q) = \sum_x P_i(x,q) \qquad grad_y(q) = r(q) - r(q+1)$$
$$c(p) = \sum_y P_i(p,y) \qquad grad_x(p) = c(p) - c(p+1)$$

- Move object under pixel (x,y) in dir. of gradients
  - If $grad_x > 0$ → move object at (x,y) right
  - If $grad_x < 0$ → move object at (x,y) left
  - If $grad_y > 0$ → move object at (x,y) down
  - If $grad_y < 0$ → move object at (x,y) up

Evaluating: a simple example

- Choose P_i at size 2x2 Pixels

- Object of weight W_0 in the center:
  - Gradients both zero
  - No movement needed

$$\begin{bmatrix} \frac{W_0}{4} & \frac{W_0}{4} \\ \frac{W_0}{4} & \frac{W_0}{4} \end{bmatrix}$$

- Object of weight W_0 in the upper left corner:
  - Gradients both = $W_0 > 0$
  - Move object right and down

$$\begin{bmatrix} W_0 & 0 \\ 0 & 0 \end{bmatrix}$$

Implementation, adding constraints

- Implementation in Java: BalanceManager
  - Layout manager for canvas with arbitrary content
- Choice of weightmap at 16x16 Pixels
- Iterative movement of objects by c * length of gradient

- Using just the gradients would move all objects to the center
- Constraint 1: objects can‘t overlap
- Constraint 2: objects stay within display area

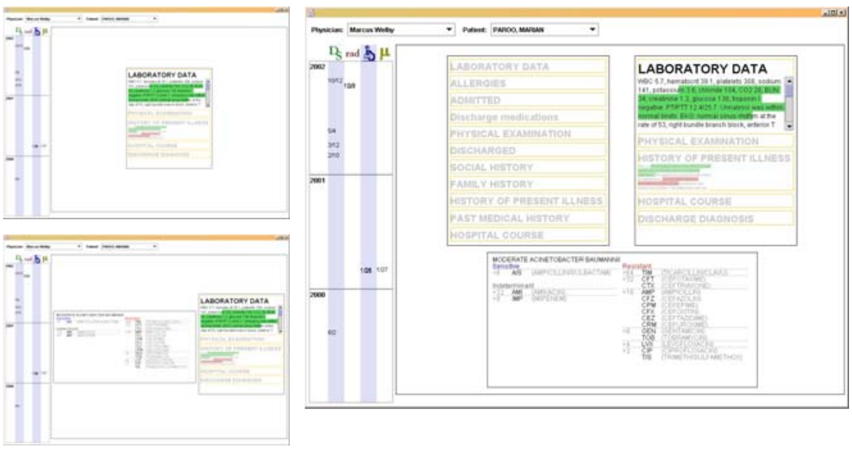
Results for free placement

- Start position of all objects was (0,0)
- Iteration until visual imbalance below threshold
- Collects objects in the center

Results on a layout grid

- Starts by placing images near the center
- Balances light images against dark ones
- Distributes images evenly

Results in an actual application



Simon Lok, Steven Feiner, Gary Ngai. "Evaluation of Visual Balance for Automated Layout". Proceedings of IUI 2004, p. 101-108.

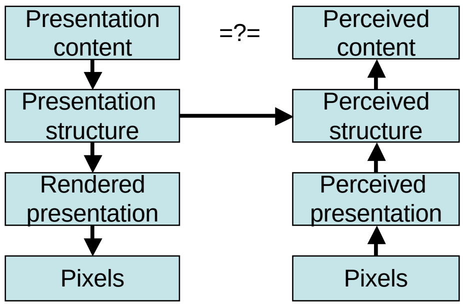
Key Features of this approach

- Fully image-based
  - Analysis is done after complete rendering process
  - Established methods from image processing
- Feedback gives also a direction for change

A more general look at AFLs

..and a helpful extension..

AFLs at different levels

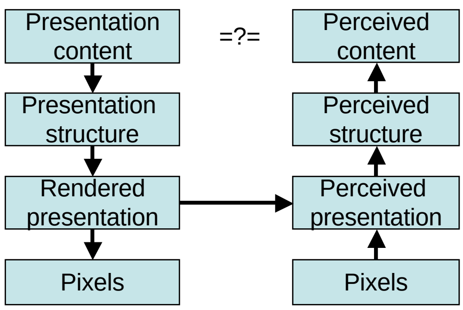


Question answered:

Is the structure of the generated presentation adequate?

- AFLs can be used at different levels
  - Feedback at different levels
  - Comparison at different levels

AFLs at different levels

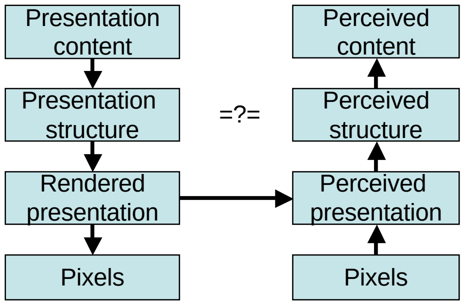


Question answered:

Is the rendering of the generated presentation adequate?

- AFLs can be used at different levels
  - Feedback at different levels
  - Comparison at different levels

AFLs at different levels

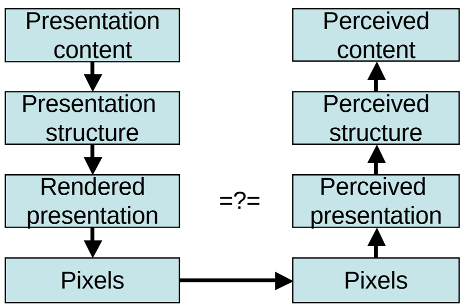


Question answered:

Is the structure correctly rendered?

- AFLs can be used at different levels
  - Feedback at different levels
  - Comparison at different levels

AFLs at different levels

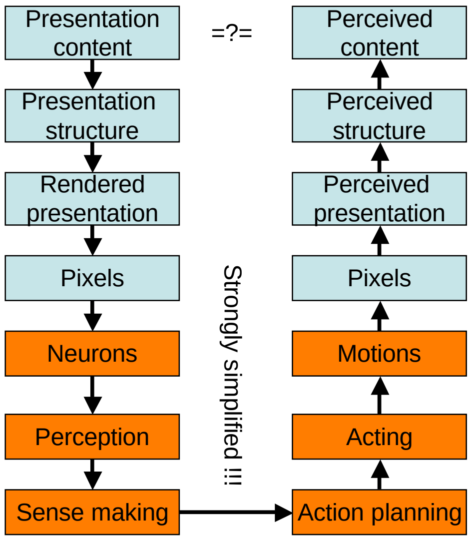


Question answered:

Is the rendering displayed in a readable way?

- AFLs can be used at different levels
  - Feedback at different levels
  - Comparison at different levels

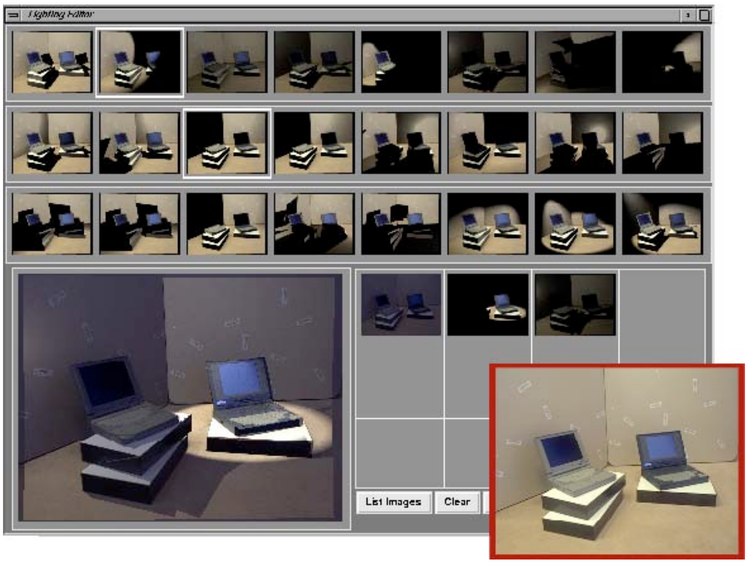
Putting the human in the loop



Putting the human in the loop

- Advantages:
  - Humans are truly smart (most, at least ;-)
  - Let the machine do the „dumb“ tasks
  - Human judgment is better than ability to formulate rules: „I know it when I see it“
- Disadvantages:
  - System can‘t work fully automatically
  - User is not just consumer, but also author (this can also be an advantage!)

Design Galleries [Marks, 1997]



Design Galleries: motivation

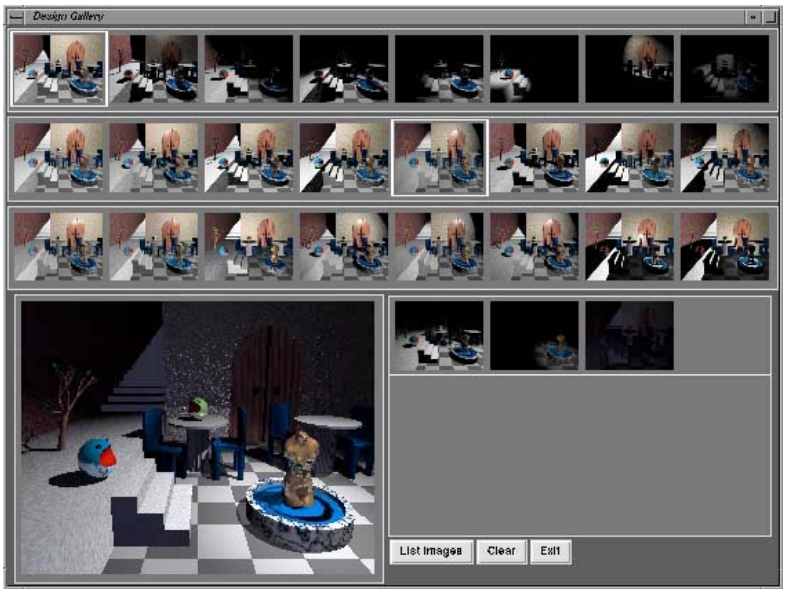
- Let the computer do what it is good at
  - Rendering a scene
  - Generating variations of the light setup
- Let the human do what (s)he is good at
  - Judging the visual quality of a scene
  - Choosing the best variation to proceed
- A form of human-machine symbiosis
  - Generate-and-test approach involving both the computer and the user

DG: general procedure:

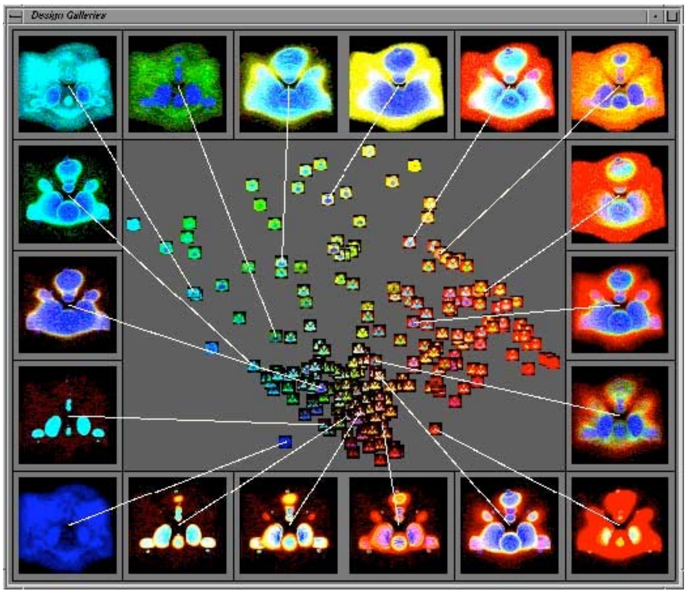
1. Choose initial settings for parameters
2. Disperse settings (i.e. choose variations which visibly change the result)
3. Render an image for each variation
4. Arrange the images in a gallery
5. Let the user choose one variation
6. Take this variation as the new starting point and goto 2.

More Design Galleries

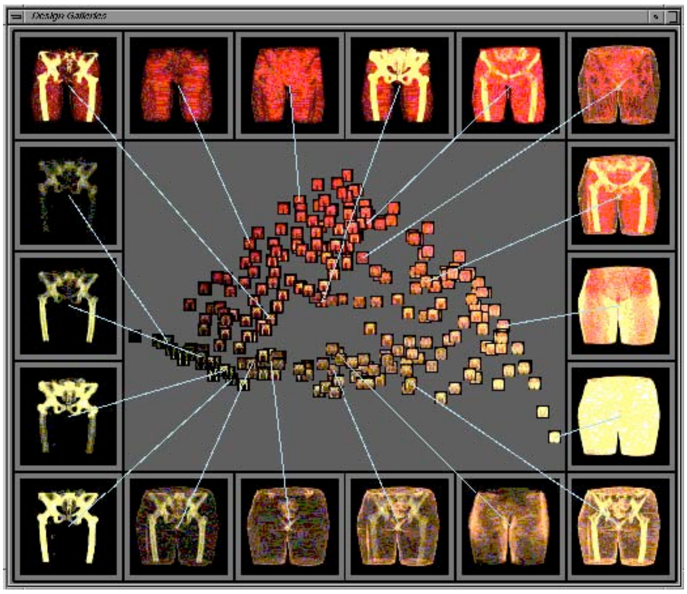
More Design Galleries



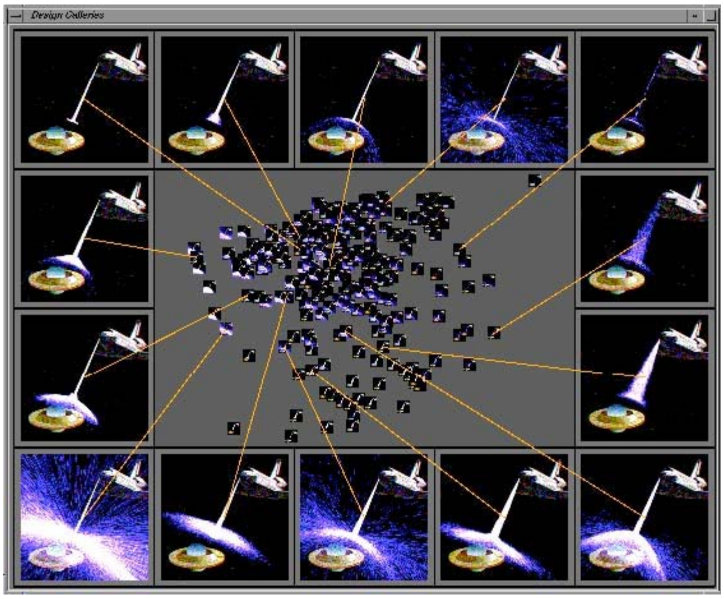
More Design Galleries



More Design Galleries



More Design Galleries



Feedback loops: Summary

- Can be built at different levels
  - Check just syntactic aspects (→ graph layout)
  - Check semantic aspects (→ need good analysis)
- Can also involve the user
  - „Smart graphics“ where „smart“ is inside the human and „graphics“ is inside the machine