

# Informationsvisualisierung

## Worum es geht

*"Data visualization is about comprehension, not graphics. Think of it as a range of techniques that enable you to display abstract numerical data and statistics in graphical form."*

BYTE-Magazin. [2]

# Informationsvisualisierung

---

## Information Murals

*Mural (engl.) = Wandmalerei, Wandgemälde*

Murals sind 2D-Repräsentationen von riesigen Datenräumen, die eine "*Gestalt Overview*" über ebendiesen Raum geben sollen und nebenbei in einen einzigen Viewport passen.

# Übersicht

---

1. Motivation
2. Beispiele
3. Algorithmen
4. Implementierung
5. Limits, Beurteilung
6. Ausblick, Philosophisches

# Motivation

---

## Probleme

- Große Datenmengen
- Keine Übersicht
- Zurechtfinden im Datenraum
- Abstrakte Daten (zB: Zahlenkolonnen)

---

Zu viele Daten ⇒ Benutzer überfordert und verwirrt.

# Motivation

---

## Informationsräume

- zeitorientiert
- Bilder
- Lieder
- Graphen
- Zahlen
- Textdateien

# Motivation

---

## Wunschträume

- Übersicht über Informationsflut
  - akustisch (Tonhöhe, Lautstärke, Klangfarbe)
  - visuell (Farbe, Form, Schattierung, Sättigung)
- Bandbreitenlimit  $\Rightarrow$  möglichst verlustfreie Darstellung
- Browsing, Navigationshilfe
- "Gestalt Overview": Merkmale und Strukturen

# Motivation

---

## Lösungsansätze

- 2D Repräsentation eines höherdimensionalen Raumes
- verschiedene Techniken
  - Overplotting
  - Aggregation
  - Mitteln
  - Abstraktion
- Informationsverlust (gewollt, aber kontrolliert!)

# Motivation

---

## Information Murals

Erfunden von John Stasko und Dean Jerding am GaTech.

- Großen Informationsraum abbilden
- Ganzheit der Information nachahmen
- Datenverlust minimieren
- Soll in einen Viewport passen
- Quintessenzen erkennen

---

Intensitäten und Farben, um Informationsverlust zu minimieren.

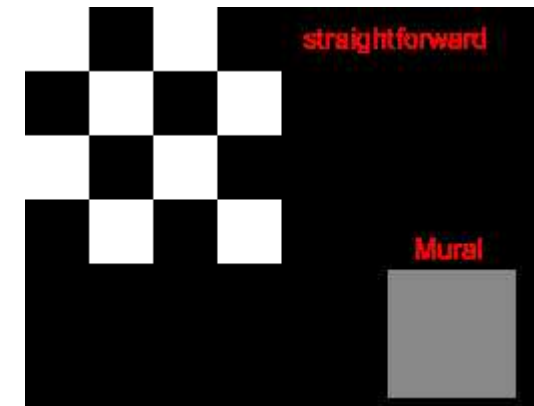
# Beispiele

---

## S/W Bild reduzieren

Einfach: Pixel aus definierten Zeilen und Spalten 1:1 übernehmen; oder Overplotting.

Besser: Miteinbeziehen anderer Originalpunkte  $\Rightarrow$  Intensität oder Farbreigenbogen



© Stefan Huber. Reduktionsvarianten.

# Beispiele

---

## Execution Mural (1)

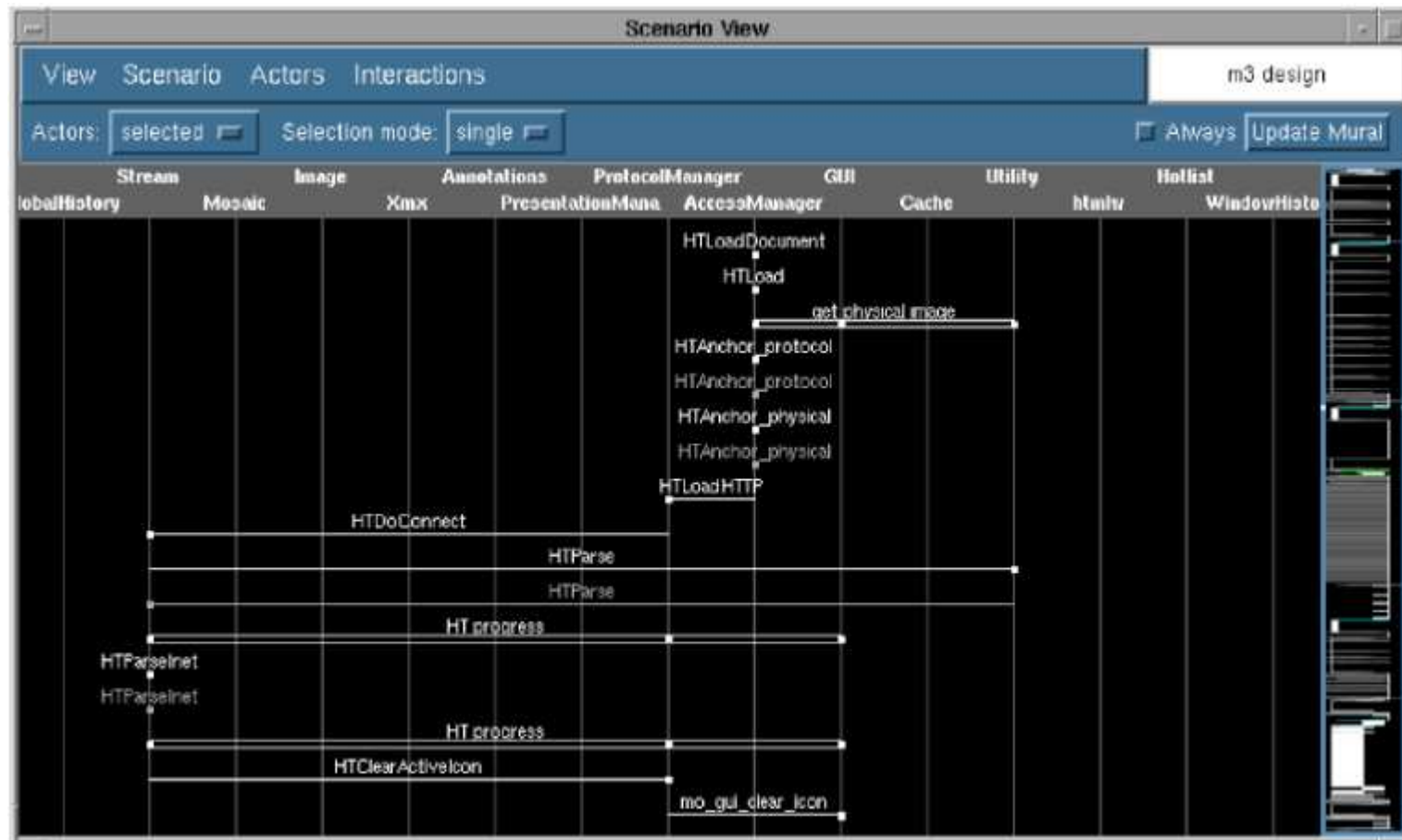
- Message traces zwischen Objekten (OOP)
- Teile verbergen oder anzeigen
- Sehr tiefgehend
- Erfahrung notwendig  $\Rightarrow$  neue Erkenntnisse

---

Theoretischer Einsatzbereich: Debugging, etwa im Bereich UI

# Beispiele

## Execution Mural (2)



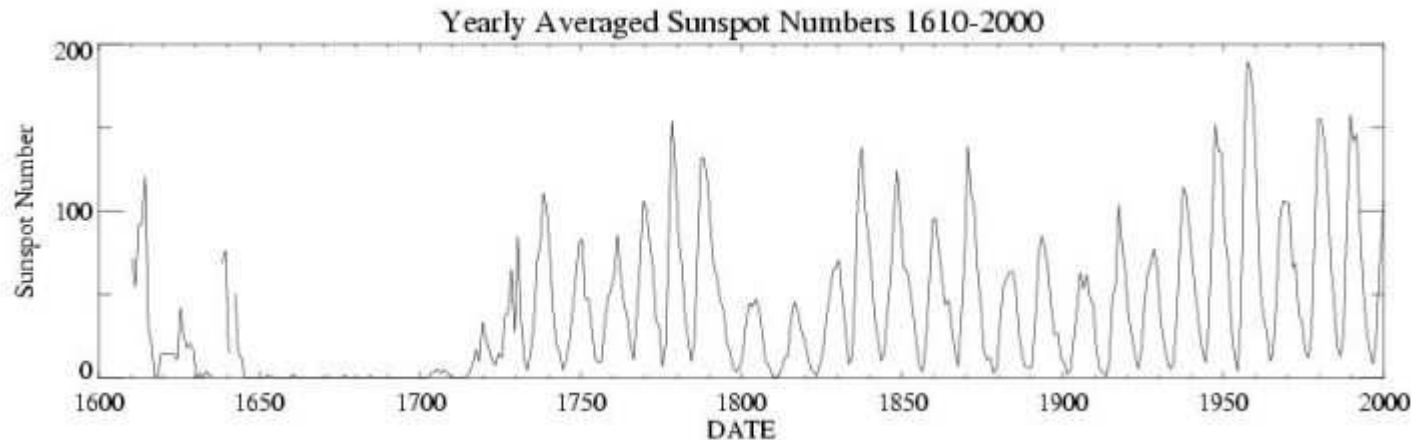
© **John Stasko**. Message trace eines Mosaic browsers. Im Mural ganz unten: PostScript Dokument. [1]

# Beispiele

---

## Sonnenflecken (1)

### Konventionelle Darstellung



© NASA. Sonnenflecken gemittelt. [4]

Nur Mittelwerte pro Zeiteinheit (hier Jahre).

Grafik antialiased  $\Rightarrow$  Graustufen sagen nichts aus!

# Beispiele

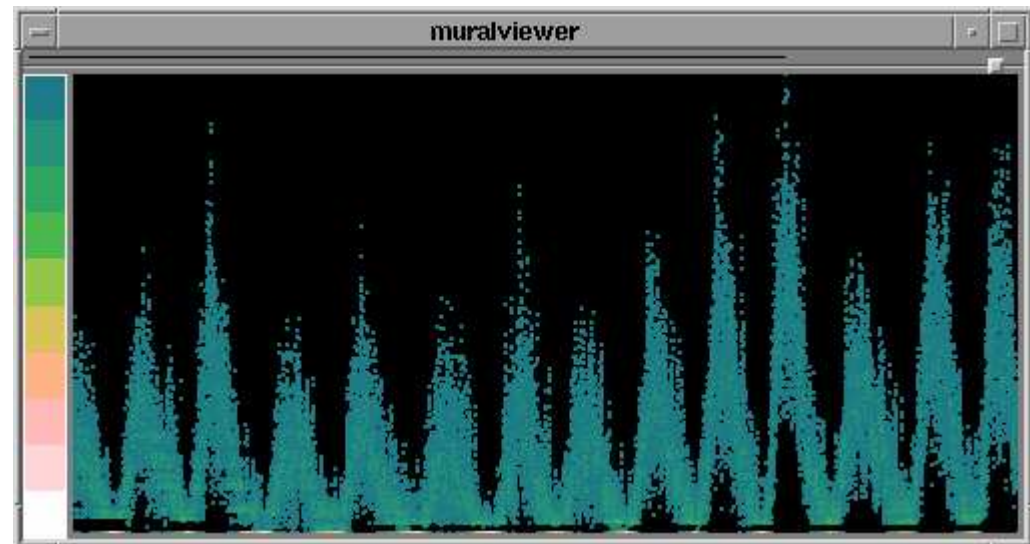
---

## Sonnenflecken (2)

### Mural

Jeder Pixel = ein Messwert  
(täglich)

Farbskala:  
dunkel = wenig Flecken  
weiß = viele Flecken.



© **John Stasko**. Muralviewer für Sonnenflecken. Interessant:  
Weiße Pixel bei 0 Messungen. [1]

# Beispiele

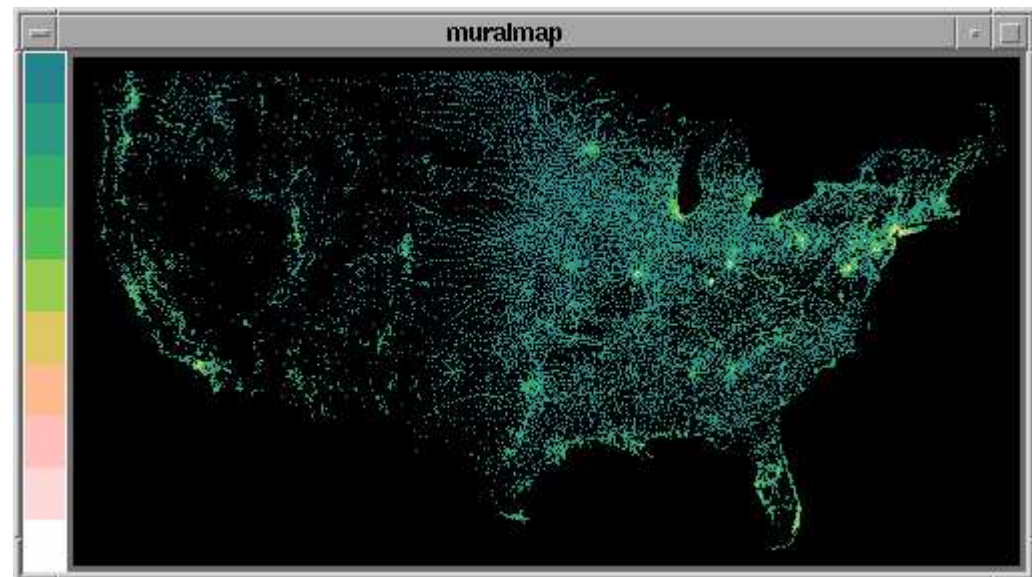
---

## Geographische Daten (1)

### Volkzählungsdaten

USA eingeteilt in  
Zählungsquadrate.

- logarithmische Skala
- Pixel = Zentrum eines  
Zählungsblockes



© John Stasko. Volkszählung 1990. [1]

# Beispiele

---

## Geographische Daten (2)

### "Realmural"?

- Elektrizität "ist überall"
- Wo Elektrizität, da Licht
- Licht summiert sich wunderbar



"Realmural" nach: [www.redrat.net/blackhole/earthlights.htm](http://www.redrat.net/blackhole/earthlights.htm)

# Beispiele

---

## Textstruktur und Textsuche (1)

Bell Laboratories: `seeSoft`.

Visualisieren von Textdokumenten.

Farben mit definierten Bedeutungen:

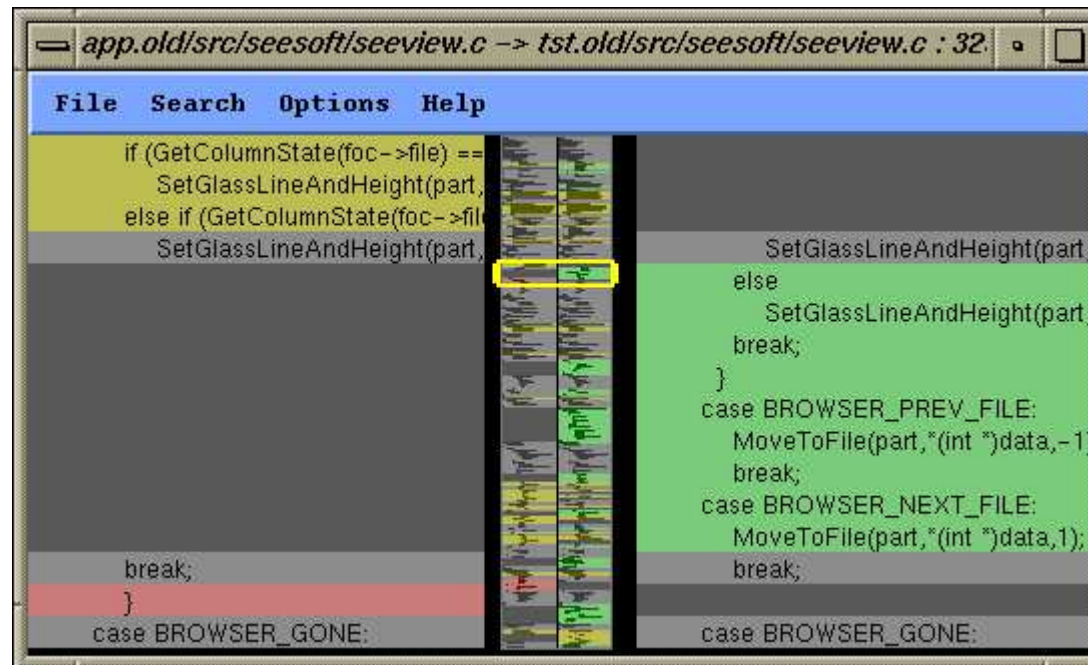
- rot: letzte Änderung
- blau: am weitesten zurückliegende Änderung

Problem: Jede Pixelzeile entsprach einer Textzeile  $\Rightarrow$  kein Gesamtüberblick.

# Beispiele

## Textstruktur und Textsuche (2)

### SeeSoft - Grafik [3]



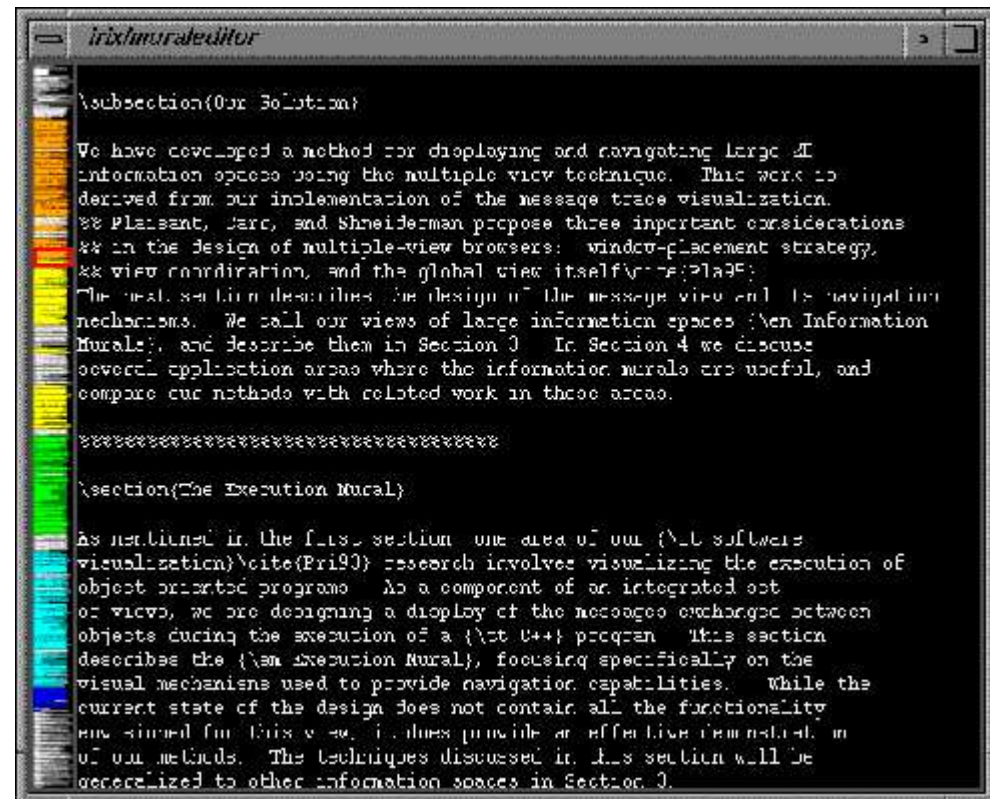
```
app.old/src/seesoft/seeview.c -> tst.old/src/seesoft/seeview.c : 32
File Search Options Help
if (GetColumnState(foc->file) ==
    SetGlassLineAndHeight(part,
else if (GetColumnState(foc->fil
    SetGlassLineAndHeight(part,
SetGlassLineAndHeight(part,
else
    SetGlassLineAndHeight(part,
break;
}
case BROWSER_PREV_FILE:
    MoveToFile(part,*(int *)data,-1)
break;
case BROWSER_NEXT_FILE:
    MoveToFile(part,*(int *)data,1);
break;
break;
}
case BROWSER_GONE:
case BROWSER_GONE:
```

**SeeSoft-Screenshot.** Man beachte das "app.old".

# Beispiele

## Textstruktur und Textsuche (3)

- Mural komprimiert gesamtes Dokument auf den Viewport
- Farben stehen für thematische Unterschiede
- Navigationsrechteck



```
irixmuraleditor
\subsection{Our Solution}
We have developed a method for displaying and navigating large AT
Information spaces using the multiple view technique. This work is
derived from our implementation of the message trace visualization.
** Pleasant, Jarc, and Shneiderman propose three important considerations
** in the design of multiple-view browsers: window-placement strategy,
** view coordination, and the global view itself\cite{Ple95}.
The next section describes the design of the message view and its navigation
mechanisms. We call our views of large information spaces (Open Information
Murals), and describe them in Section 3. In Section 4 we discuss
several application areas where the information murals are useful, and
compare our methods with related work in these areas.
=====
\section{The Execution Mural}
As mentioned in the first section one area of our (X11 software
visualization)\cite{Pri90} research involves visualizing the execution of
object oriented programs. As a component of an integrated set
of views, we are designing a display of the messages exchanged between
objects during the execution of a (not C++) program. This section
describes the (X11 Execution Mural), focusing specifically on the
visual mechanisms used to provide navigation capabilities. While the
current state of the design does not contain all the functionality
envisioned for this view, it does provide an effective demonstration
of our methods. The techniques discussed in this section will be
generalized to other information spaces in Section 3.
```

© John Stasko. Muraleditor für ein LaTeX Dokument. [1]

# Beispiele

---

## Textstruktur und Textsuche (4)

### Suche nach Substrings.

Im konventionellen Fall:

- Springen von einer Fundstelle zur nächsten
- Kaum Übersicht (Wo bin ich gerade?)
- Nur eine Suche zur selben Zeit

# Beispiele

---

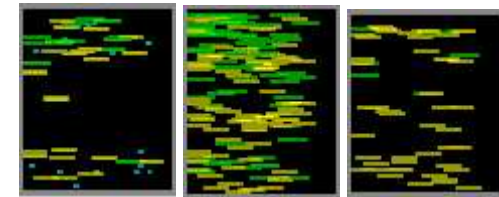
## Textstruktur und Textsuche (5)

### Suche nach Substrings.

Murals bieten sofort Überblick über Verteilung der Fundstellen.

Suche nach verschiedenen  
Stichwörtern.

- Farben repräsentieren  
Schlüsselwörter
- Mehrere Suchen übereinander
- Sofortige Übersicht



© John Stasko. Suche nach  
Stichwörtern in drei Dokumenten. [1]

# Algorithmen

---

## Allgemeines

1. Erste Algorithmen für zeitorientierte Daten
2. Optimierungen (Performance)
3. Unterstützung für Graustufen und Farben hinzugefügt

### Funktionsweise

$N \times M$  Bild in  $X \times Y$  einpassen, wobei  $X \ll N$  und  $Y \ll M$

### Datenstruktur

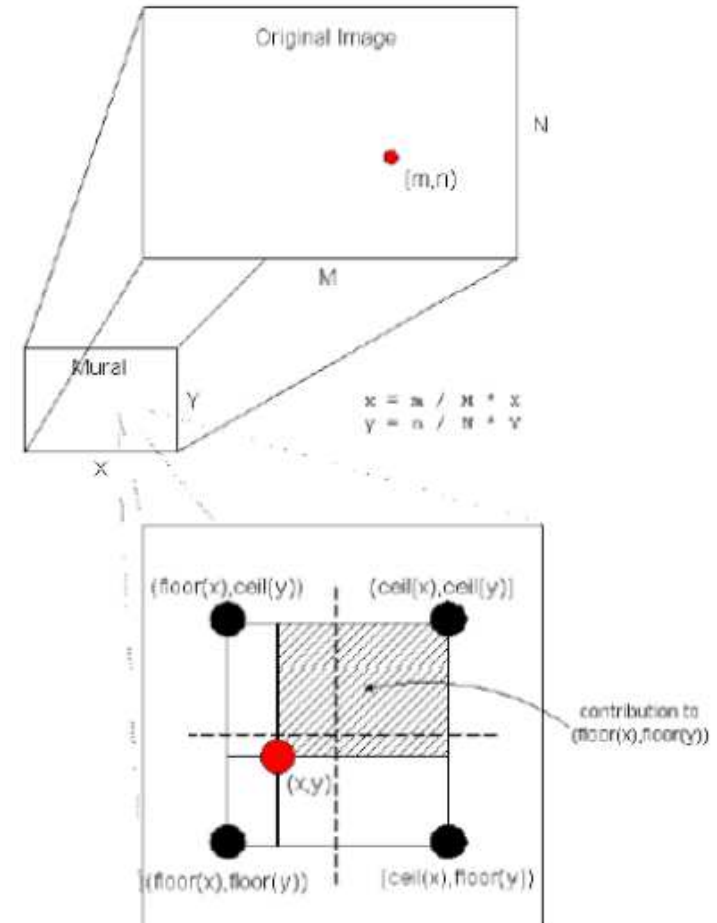
Datenstruktur: Mural + Originaldaten

# Algorithmen

## Ursprünglicher Algorithmus (1)

Gewichtetes Area Sampling mit überlappenden Gewichtsfunktionen.

Originalpixel trägt proportional zu umgebenden Muralpixeln bei.



© John Stasko. Schema des Algorithmus. [1]

# Algorithmen

---

## Ursprünglicher Algorithmus (2)

### Prinzip

1. Punkt transformieren
2. Einheitsquadrat zu benachbarten 4 Pixeln
3. Intensität jedes Pixels gemäß der gegenüberliegenden Fläche erhöhen
4. Intensität relativieren

# Algorithmen

---

## Ursprünglicher Algorithmus (3)

1. `mural_array[x][y]` auf 0 setzen
2. für jeden Punkt  $(m, n)$  des Originalbildes:
  - a. Berechne:  $x = m/M * X$  und  $y = n/N * Y$
  - b. Fläche der Quadranten berechnen  
(definiert durch  $(x, y)$  und dem Einheitsquadrat:  $(f(x), f(y)), (f(x), c(y)), (c(x), c(y))$   
und  $(c(x), f(y))$ )
  - c. Fläche zum diagonal gegenüberliegenden Eckpunkt  
(=Bildpunkt) addieren
  - d. ggf. `max_mural_array_value` erhöhen
3. Für alle  $(x, y)$  in `mural_array`
  - a. Wert aus `mural_array[x,y]` → Grauwert
  - b. Pixel zeichnen

# Algorithmen

---

## Ursprünglicher Algorithmus (4)

### Bewertung

Implizites Antialiasing durch die Gewichtsfunktion.

Bei photorealistischen Bildern ist Antialiasing angenehm, Murals brauchen das aber nicht  $\Rightarrow$  Gewichtsfunktion eliminieren.

Das Resultat ist ein effizienterer Algorithmus!

# Algorithmen

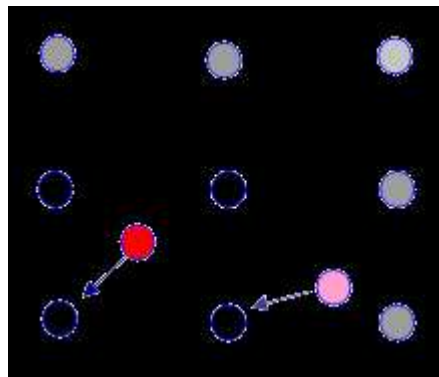
---

## Effizienter Algorithmus (1)

### Prinzip

Gewichtsfunktion eliminiert  $\Rightarrow$  "schärferes" Bild.

Diesmal: Ein Boxfilter mit Ausmaßen  $M/X \times N/Y$ .



© Reinhard Partmann. Schema eines Boxfilters.

Intensität wiederum ein relativierter Wert.

# Algorithmen

---

## Effizienter Algorithmus (2)

1. `mural_array[x][y]` auf 0 setzen
2. für jeden Punkt  $(m, n)$  des Originalbildes:
  - a. Berechne:  $x = m/M * X$  und  $y = n/N * Y$
  - b. `1.0` zu `mural_array[f(x)][f(y)]` addieren
  - c. ggf. `max_mural_array_value` erhöhen
3. Für alle  $(x, y)$  in `mural_array`
  - a. Wert aus `mural_array[x,y]` → Grauwert
  - b. Pixel zeichnen

# Algorithmen

---

## Farbbehandlung (1)

Bisher nur Graustufenbilder.

Farben machen Probleme.

Problem: Verschiedene Farben werden auf einen Muralpixel abgebildet.

---

Lösungsvorschläge?

# Algorithmen

---

## Farbbehandlung (2)

Eine Möglichkeit: RGB mixen.

Nicht schlau, weil der Originalzustand dennoch verschleiert oder verzerrt wird. (Gelb = Rot & Grün)

Kompromiss: Die am häufigsten auf einen Pixel abgebildete Farbe gewinnt.

---

Aber wie organisiert man das am besten?

# Algorithmen

---

## Farbbehandlung (3)

Zwei Ansätze:

1. Intensität jeder Farbe merken:  
Für jede Farbe ein Wert (ansonsten wieder RGB mischen).  
Welchen maximalen Intensitätswert nehmen?
2. Farblistenverwaltung:  
Wie viele Punkte einer Farbe bilden auf denselben Muralpixel ab?



# Algorithmen

---

## Farbbehandlung (5)

Ohne Gewichtsfunktion: jeder Punkt mit Intensität  $1.0$ .

Mit Gewichtsfunktion müsste man Intensitätswert mitführen  
Dennoch verliert man die Information, welche Farbe welchen  
Teil der Intensität beigetragen hat.

5 blaue mit  $0.1$  und 1 roter mit  $0.8 \Rightarrow$  1 roter mit  $1.3$ .

# Algorithmen

---

## Farbbehandlung (6)

### Schlussfolgerungen

- Nur die häufigste Farbe anzeigen
- Verlust der anderen Farben in Kauf nehmen. Abhilfen:

Interaktion

Rotierende Farbanzeige

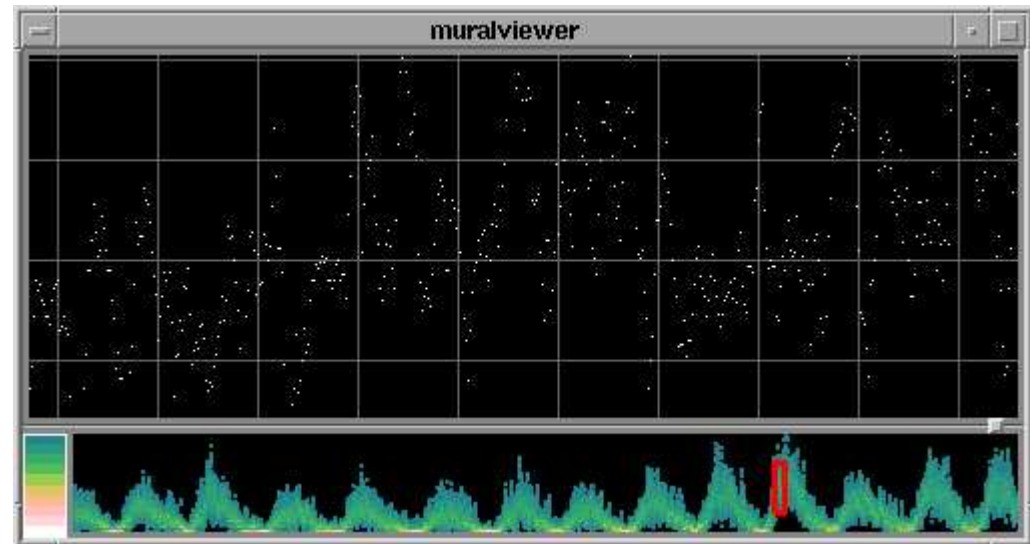
# Implementierung

---

## Einsatz

Stasko und Jerding haben Murals als Navigations- und Zoomwidgets konzipiert.

Zusätzlich eine Klasse als Zeichenfläche für die Originaldaten.



© John Stasko. Muralviewer: eigenständige Anwendung. [1]

# Implementierung

---

## Umgebung

- Implementierung als abstrakte C++ Klasse (leider nicht verfügbar)
- Motif/X Window System (unter Irix)
- Verwendung von `vz Library` der Bell Labs

---

Vz ist tot. Bell Labs führt sie nicht mehr auf ihrer Homepage.

# Implementierung

---

## Funktion

- War wie jedes heutige Widget verwendbar
- Stelle bereit:
  - Einstellbare Farbskala
  - Graustufeneinstellungen
  - Navigationsrechteck
  - Zoom/Scroll Events

# Limits

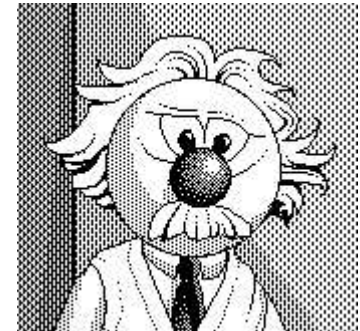
---

## Intensitäten

Differenzierung von Intensitäten fällt dem Menschen schwer. [5]

*Dithering* ist besser geeignet.

Dithern ist nicht möglich, da jeder Pixel ausgenutzt werden soll.



# Limits

---

## Farben

Farbe wird mit Klasseneinteilung assoziiert  
(Parteien bei der Wahl, ...)

Kontinuierliche Werte unangenehm, wie zB beim Muralviewer zu sehen ist.

Murals: Farben für "Klassen"  
(LaTeX `\section` udgl.: Muraleditor)

Benachbarte Farben stören sich gegenseitig.

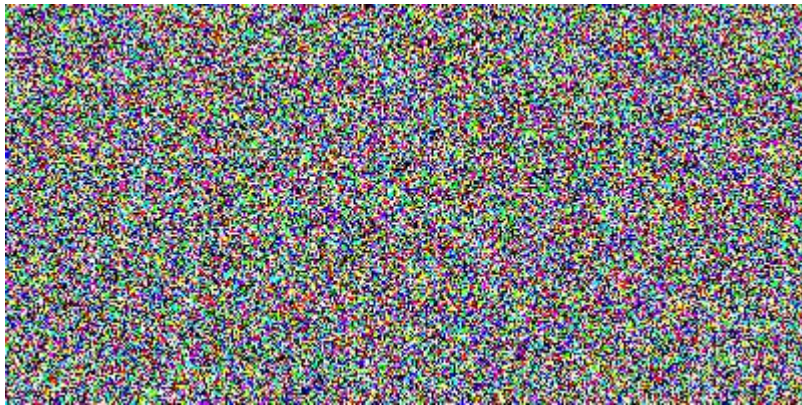
# Limits

---

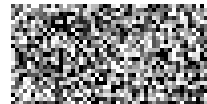
## Datenstrukturierung

Murals für abstrakte, aber strukturierte Daten.

Zufällige Daten → graue Nebelwolke:



...würde in etwa so aussehen:



Logisch denn: Mural soll ja Struktur darstellen.

# Limits

---

## Performance

Zur Zeit der Entstehung (1995!) Probleme mit der Performance.

Beim Zoomen oder Scrollen muss das gesamte Quellobject untersucht und angezeigt werden.

Selbst optimierte Algorithmen waren manchmal zu langsam.

Heute hätte ein Mural wohl keine Probleme mehr damit.

# Zusammenfassung und Ausblick

---

## Erfassungsmethoden

- Sunspots
- Verkehrsmessung
- Geographische Daten
- Textdateien
- Grafiken
- Abstrakte Datenmengen

# Zusammenfassung und Ausblick

---

## Aufgaben

- Browsing
  - Navigieren
  - Strukturen erkennen
  - **Überblick**
- 

Plaisant: *"Dense global views provide experts with direct access to details that would otherwise require several zooming operations (even if these global views appear unreadable to others!)"* [6]

# Zusammenfassung und Ausblick

---

## Mural (1)

- "Gestalt Overview" bekommen
- Überblick über riesige Datenkonvolute
- Navigationshilfe
- Leider langsam

Schlussfolgerung der Autoren: Information Murals hauptsächlich als Browsing- und Navigierhilfe sinnvoll.

Voraussetzung: Gute Datenbasis

# Zusammenfassung und Ausblick

---

## Mural (2)

### HCI

Gesichtspunkte nach **H**uman-**C**omputer **I**nteraction

- Anwenderzufriedenheit: gut
- Einarbeitungsdauer: kurz
- Benutzerperformance, Fehlerrate, Merkvermögen: hängt von Visualisierung ab

# Zusammenfassung und Ausblick

---

## Ähnliche Techniken

- Fisheye lens: Kombiniert mit einem Mural tolle Möglichkeiten
- Cone Trees, Perspective Wall
- SeeSoft (Bell Labs)
- Visual Insights™ [3] (Lucent Technologies)
- POLKA: Animationen, Fraktale. Von Stasko propagiert (scheint Murals abgelöst zu haben)

# Zusammenfassung und Ausblick

---

## Diskussionspunkte

- Geeignete Methode um Informationsraum in Farbbild zu transformieren notwendig
- Warum werden Murals nicht verwendet?  
Wo sind die Applikationen?  
Würden Sie sowas verwenden?
- Ist RGB mischen (für den geübten Betrachter) wirklich so schlecht?

# Zusammenfassung und Ausblick

---

Prof. John T. Stasko in einem Mail an uns:

*"Dean just created some simple applications for us to use in the work, and we never pushed it any further. I often think about the one with the source code in the scrollbar and wished I had something like it!"*

---

Danke fürs Zuhören!

# Literaturhinweise

---

1. The Information Mural, *Dean F. Jerding and John Stasko*. März 1996.  
[http://www.cc.gatech.edu/gvu/softviz/infoviz/information\\_mural.html](http://www.cc.gatech.edu/gvu/softviz/infoviz/information_mural.html)
2. State of the Art, *BYTE*. April 1993. p120-147
3. Visual Insights™, Data Visualization Solutions. *Bell Labs*..  
<http://www.bell-labs.com/project/visualinsights/brochure/Oindex.html>
4. The Sunspot Cycle. *NASA Science Directorate*.  
<http://science.nasa.gov/ssl/pad/solar/sunspots.htm>
5. Psychische und Physiologische Aspekte der Visualisierung. *Christian Ziegler u. Michael Marty*.  
<http://www.mysunrise.ch/users/martymichael/visu/content.htm>
6. Image-Browser Taxonomy and Guidelines for Designers. *C. Plaisant, D. Carr, and B. Shneiderman*. IEEE Software, vol. 12, no. 2, pp. 21-32, Mar. 1995.