Smart Views in Smart Environments

Axel Radloff, Martin Luboschik, Heidrun Schumann
Computer Graphics Group
University of Rostock, Germany
Smart Environments

Heterogeneous displays
Heterogeneous devices
Heterogeneous software
Heterogeneous sensors
Increased user assistance
Heterogeneous displays
Heterogeneous devices
Heterogeneous software
Heterogeneous sensors
**Increased user assistance**

Replace complex configuration
Heterogeneous displays
Heterogeneous devices
Heterogeneous software
Heterogeneous sensors
**Increased user assistance**

Replace complex configuration
Instantaneous hardware changes
Heterogeneous displays
Heterogeneous devices
Heterogeneous software
Heterogeneous sensors
Smart Environments: Information Displays

Adaptation (Layout, Size, ...)!

Heterogeneous displays
Heterogeneous devices
Heterogeneous software
Heterogeneous sensors
Heterogeneous displays
Heterogeneous devices
Heterogeneous software
Heterogeneous sensors

Who?
Intertwining of visual outputs instead of intertwining of systems.

User:
What (Content)?
[What does the user want to show / see?
How does the content belong together?
How important is the content?...]

Our Smart Environment:
How, Where....
Generate Views by:
Generate Views by:
- **Grabbing (proprietary software support)**
Generate Views by:
- Grabbing (proprietary software support)
Generate Views by:
- **Grabbing (proprietary software support)**
Generate Views by:
- **Grabbing (proprietary software support)**
Views: API

Generate Views by:
- Grabbing (proprietary software support)
- API-Integration
Generate Views by:
- Grabbing (proprietary software support)
- API-Integration
Generate Views by:
- Grabbing (proprietary software support)
- API-Integration
Generate Views by:
- Grabbing (proprietary software support)
- API- Integration
Views: everything displayable
View-Packages: Views + Semantic

I need...
I want to show...
Person X should see...

AView 1.2.1
AView 1.2.2
AView 1.2.3

AView 1.2.1
AView 1.2.2
AVViewx.2

GView 1.1
GView 1.2

GView x.1
GView x.2
GView x.3

AVView 1.1
AVView 1.2
AVView 1.3.x

GView 1.1
GView 1.2
Views A and C belong together...
View Y should not be distorted...
View-Packages: Views + Semantic

System 1

View Grabber 1.1
AView 1.1.1
AView 1.1.x
AView 1.2.1
AView 1.2.x
AView 1.3.1
AView 1.3.x
GView 1.1
GView 1.2

Application 1.1

System x

View Grabber x.1
AView x.1.1
AView x.1.2
AView x.1.3
GView x.1
GView x.2
GView x.3

Application x.1

AView x.1.x
AView x.2.1
AView x.2.x
AView x.3.1
AView x.3.x

GView x.1
GView x.2
GView x.3

View Y should not be distorted...
View-Packages: Views + Semantic

System 1
- View Grabber 1.1
- Application 1.1
- GView 1.1
- AView 1.1.1
- AView 1.2.1
- AView 1.1.x
- AView 1.2.x
- AView 1.3.1
- AView 1.3.x
- AView 1.3.x

System x
- View Grabber x.1
- Application x.1
- GView x.1
- AView x.1.1
- AView x.2.1
- AView x.1.x
- AView x.2.x
- GView x.1
- GView x.2
- GView x.3

Interactive View Package Definition

- AView 1.1.1
- GView x.1
- GView 1.1
- AView 1.2.1
- AView 1.2.2
- AView 1.2.3
- GView x.3
- GView x.2
View-Packages: Views + Semantic
View-Packages: Views + Semantic

Interactive View Package Definition
= Assigning views to displays
Display mapping by Heider:

Maximizing (NP-hard):

\[ q(m_{t-1}, m_t) = a \cdot q_s(m_t) + b \cdot q_t(m_{t-1}, m_t) + c \cdot q_p(m_t) \]

\( m_t \) – current mapping
\( m_{t-1} \) – previous mapping
\( q_s \) – spatial quality
\( q_t \) – temporal continuity quality
\( q_p \) – semantic proximity quality
\( a, b, c \) – weighting factors
Display mapping by Heider:

Maximizing (NP-hard):

\[ q(m_{t-1}, m_t) = a \cdot q_s(m_t) + b \cdot q_t(m_{t-1}, m_t) + c \cdot q_p(m_t) \]

\[ \text{spatial quality} \]

- \( m_t \) – current mapping
- \( m_{t-1} \) – previous mapping
- \( q_s \) – spatial quality
- \( q_t \) – temporal continuity quality
- \( q_p \) – semantic proximity quality
- \( a, b, c \) – weighting factors
Display mapping by Heider:

Maximizing (NP-hard):

$$q(m_{t-1}, m_t) = a \cdot q_s(m_t) + b \cdot q_t(m_{t-1}, m_t) + c \cdot q_p(m_t)$$

- $m_t$ – current mapping
- $m_{t-1}$ – previous mapping
- $q_s$ – spatial quality
- $q_t$ – temporal continuity quality
- $q_p$ – semantic proximity quality
- $a, b, c$ – weighting factors
Smart Display Mapping

Display mapping by Heider:

Maximizing (NP-hard):

\[ q(m_{t-1}, m_t) = a \cdot q_s(m_t) + b \cdot q_t(m_{t-1}, m_t) + c \cdot q_p(m_t) \]

- \( m_t \) – current mapping
- \( m_{t-1} \) – previous mapping
- \( q_s \) – spatial quality
- \( q_t \) – temporal continuity quality
- \( q_p \) – semantic proximity quality
- \( a, b, c \) – weighting factors
Enhancements:

- Enhanced quality
- Reduction of calculation costs
- Adapted semantic proximity
Smart Display Mapping – enhanced quality

\[ \cos \alpha \cdot a \]

\[ \sin \alpha \cdot a \]

\[ \text{FOV} \]

\[ \beta \]
Smart Display Mapping – reduced costs

Dismiss temporal continuity

\[ q(m_{t-1}, m_t) = a \cdot q_s(m_t) + b \cdot q_e(m_{t-1}, m_t) + c \cdot q_p(m_t) \]
Dismiss temporal continuity
but materialize mappings only if ...
Dismiss temporal continuity but materialize mappings only if ...
Dismiss temporal continuity
but materialize mappings only if ...
Smart Display Mapping – semantic proximity

weight = 1
Smart Display Mapping – semantic proximity

weight = 0
Smart Display Mapping

Interactive View
Package Definition

System 1
- View Grabber 1.1
- Application 1.1
  - AView 1.1.1
  - GView 1.1

System x
- View Grabber x.1
- Application x.1
  - AView x.1.1
  - GView x.1

Environment Knowledge
(User Positions, Viewing Directions...)

Smart Display Mapping

AV 1.1.1
AV 1.2.1
AV 1.3.1
AV 1.3.x
GV 1.1
GV 1.2
GV x.1
GV x.2
GV x.3

Realization of n : 1 mappings: Smart Layout

System 1
- View Grabber 1.1
- Application 1.1
- AView 1.1.1
- AView 1.2.1
- GView 1.1
- GView x.1
- GView 1.1

System x
- View Grabber x.1
- Application x.1
- AView x.1.1
- AView x.1.x
- AView x.2.1
- GView x.1
- GView x.3
- GView x.2

Interactive View Package Definition

Environment Knowledge
(User Positions, Viewing Directions...)

Smart Display Mapping

Smart Layout
- Meta Renderer 1
- Meta Renderer 2
- Meta Renderer 3
- Meta Renderer 4
Smart Layout

[Image of a technical diagram or interface]
Smart Layout

Spring Force directed Layout
Smart Layout
Pressure based resizing
Pressure based resizing
Pressure based resizing
Pressure based resizing (resizing restricted)
Dynamic Occluder handling
Dynamic Occluder handling
Smart Views in our SmartLab

a. radolff, m. luboschik, h. schumann: smart views in smart environments
university of rostock
sg 2011
Conclusion and Future Work

• Smart View Management - to intertwine visual outputs
  – User defines:
    • “what to see and what is seen” – View Definition (grabbing, API), View Package Generation
  – Smart View Management handles:
    • “how to display” – View Package Generation
    • “where to display” – Smart Display Mapping
    • “how to layout” – Smart View Layout

• Future
  – Task-based enhancement
    • View selection based on task in workflow
    • Temporal and causal dependencies of views
  – View adaptation
    • Enhance effectivity of displayed information
THANKS FOR YOUR ATTENTION!
QUESTIONS WELCOME!