Smart Views in Smart Environments

Axel Radloff, Martin Luboschik, and Heidrun Schumann

Institute for Computer Science,
University of Rostock, Germany
{axel.radloff,luboschik,schumann}@informatik.uni-rostock.de

Abstract. Smart environments integrate a multitude of different device ensembles and aim to facilitate proactive assistance in multi-display scenarios. However, the integration of existing software, especially visualization systems, to take advantage of these novel capabilities is still a challenging task. In this paper we present a smart view management concept for an integration that combines and displays views of different systems in smart meeting rooms. Considering these varying requirements arising in such environments we provide a smart viewing management taking e.g. the dynamic user positions, view directions and even the semantics of views to be shown into account.

Keywords: view management, smart environment, layout, display mapping, smart configuration

1 Introduction

Smart environments are physical spaces collecting and processing information about the users and their environment to estimate the intended situation and adapting the environment and its behavior in order to reach an intended state ([6, 19, 15]). Smart environments focus on heterogeneous ad-hoc device assemblies consisting of multiple sensors, displays, software infrastructure, etc. ([11, 5]) In particular, smart meeting rooms are used to support groups working together to reach a common goal [17]. Here, three typical scenarios can be distinguished: (1) the presenter scenario, including one dedicated presenter imparting information to the audience, (2) the exploration scenario, a group working together to explore data to gain insights and (3) the work group scenario, composed of small sub-groups working together discussing information. However, these scenarios are not static and can alter dynamically. To give an example, a group of climate researchers may join up for a meeting. First, one group member presents an agenda for the meeting, the topics to be discussed and further information - the presenter scenario. Then, the group splits into sub-groups to discuss the information and gain insights - work group scenario. After that, the leaders of the sub groups present the insights they acquired - presenter scenario, followed by an exploration session of the whole team - exploration scenario.

In these scenarios the users often use their own personal devices and software. However, presenting and sharing information with other users requires combining information on public displays (projectors, large monitors). In most cases this is solved by binding private devices directly to one projector. In this case the location of presented information is fixed or requires configuration effort to change. Furthermore, such scenarios do not take advantage of the capabilities of multi-display environments, e.g. the displays do not act as a whole entity but are used as multiple single-display environments. Hence, information may be spread over different displays and thus, users have to turn their head, move around or move their eyes repeatedly resulting in an uncomfortable working environment.

To face these problems we introduce a smart view management in section 3. The main idea is to show views from different software systems by combining *views* to be streamed via network. We provide an *ad-hoc option* that allows the integration of proprietary software and a *prepared option*. Here, the systems provide their views themselves and thus allow for an adaptation of the displayed information taking further characteristics of the environment into account. Our approach requires grouping several views based on semantic relationships between provided views (sec. 4). These groups of views are then automatically assigned to displays by a smart display mapper (sec. 5) also taking position and view-direction of users into account. Finally, a smart layout (sec. 6) allows for an appropriate scaling and layouting of different views on one display. We summarize our results and give an outlook on future work in section 7.

2 Background and Related Work

The development of smart environments raises many research questions like how to determine the users intention, how to interact with those environment naturally or how to make use of available devices in service of the user to ease life and work [5]. An associated open problem in such scenarios is how to use the available display devices for a maximum support of communication with a minimum configuration effort at the same time. This problem comprises the integration of displays, the adaptation of content to different display devices and so on.

The integration of heterogenous displays is an open problem, as they may change over time (joining and leaving devices), the content to be shown may change but meanwhile shall facilitate collaborative work (e.g., in [7, 14]). The specifically developed, rare approaches for those ensembles typically combine the individual displays into a large single one or replicate content at multiple displays. Thus, current research mainly addresses the problems of synchronously sharing content from multiple devices on multiple displays and sharing the corresponding multiple interactions on the devices (e.g., [9, 14]). But all those systems are only restrictively applicable in such dynamically changing heterogenous ensembles, we would like to address. The approach in [9] requires the proprietary software who's content shall be distributed running at each system. This implicates hard constraints concerning the operating system and hardware requirements to run the proprietary software and excludes heterogenous ensembles. The Diskotheque environment [14] needs an adapted X-Server running on each machine with the same constraints. Although WeSpace [18] is less constrained, it accounts only for one large public display on which different content shall be shared upon. Moreover those works generally do not consider the current state of the smart environment and assume static displays like mounted projectors or static servers.

Although there are publications in the field of multi display environments that study the effectiveness of such environments (e.g., [18]) or of single display types (e.g., [16]), they typically do not provide a strategy for reacting to dynamic changes. To the best of our knowledge, the work of Heider [10] is currently the only automatic approach, which covers spontaneous changes that concern displays and the corresponding usability within a smart environment. Display surfaces, user positions, projectors and the content to be shown are used, to optimize the assignment of views to displays. The approach automatically integrates displays and assigns content to them, but it does not comprise important factors like perceivability of information or semantics of content yet.

A further important issue is, how to adapt content to heterogenous displays (size, resolution, color depth...). The adaptation of graphical representations according to given output devices, in particular considering the reduced display size of mobile devices, has been addressed extensively (e.g., for images [3], for maps [8], for 3D-models [12]). Beside adapting the content itself, other works focus on the adaptive layout of different contents (e.g., from markup languages). This kind of adaptation is strongly required in the described scenario, if contents from different sources are to be shown on one display. Beach [4] presented that the general layout problem of arranging non overlapping rectangles into a minimum rectangular space is NP-complete. Thus, automatic layout approaches are commonly based upon templates defined by an author and arrange content by abstract (e.g. author-of) or spatial (e.g. left-of) constraints [1]. To solve these constraints optimization approaches like force based models are used. Unfortunately, dynamic ensembles generally can not provide any templates at all.

The approaches described above primarily relate to specific problems. We present a general smart view management combining state-of-the-art methods with newly developed techniques to allow for a smart information presentation in smart meeting rooms.

3 Principal Approach

In this paper we present the basic concept of a smart view management. For this purpose, we have to consider the characteristics that come along with a smart meeting room: ad-hoc device ensembles, heterogeneous personal devices, heterogeneous (perhaps proprietary) software and operating systems, dynamically varying scenarios (presenter, work group or exploration), as well as positions, varying view directions and interests of the users. Hence, our concept is designed to be independent from concrete information displaying software systems. Instead we focus on the output of those systems – the views. These views have to be provided to the smart room first. For this, we define two options: (a) ad-hoc and (b) prepared view generation, that both come along with specific advantages and disadvantages. Note that both options can be used in parallel.

The **ad-hoc option** uses view grabbers that run on the physical machine where the information displaying software is running. Each view grabber provides an image stream of a rectangular area of the screen. This way, even proprietary software without the opportunity of customization can be integrated as well. However, this option provides the views as they have been generated and offers no possibility for adapting the content. There is also no meta-information about the content of the view available (e.g. data source).

The **prepared option**, in contrast, requires a slight adaptation of the information displaying software, in such a way that the software system provides an image stream of the content by itself. In doing so, each generated image stream defines a view. Customizing the information displaying software allows for providing meta-information about the displayed view, e.g. which views belong together. Furthermore, views to be presented on a public display in the smart room do not necessarily have to be similarly shown on the personal device. However, this option needs a slight implementation effort to customize the software. This effort can be reduced by providing a prepared library.

Gathering views is only the first step. The major issue is to adequately display these views with regard to the characteristics of a smart meeting room. This requires:

– Grouping views (defining so-called *view packages*), in a way that semantically related views can be shown on the same or close-by displays. We address this point with an interactive view package definition (sec. 4).
– Automatically assigning views to display surfaces. To accomplish this we use a smart display mapper (sec. 5).
– Automatically generating a layout for the display of combined views. Our smart layout automatically arranges these views (sec. 6).

Additionally, our smart view management comprises meta renderers that encapsulate the characteristics of displays and thus provide information for the display mapping and layout. Figure 1 shows the framework of our smart view manager who's functionality is described in the following sections.

4 Interactive generation of view packages

Usually the number of views to be shared among users should not be limited and thus, there are often more views to be displayed than available display surfaces. This requires an appropriate combination of different views to be presented upon one display with regards e.g. to the user's goals, the current scenario or the content of the views. We use the term *view packages* to describe a group of semantically belonging views. Although an automatic generation of view packages is desirable, the number and complexity of influencing factors that should be considered in real time is too high. Moreover, using the *ad-hoc option* does not provide the necessary meta-information to appropriately group views. Thus, we use an interactive approach to define view packages. However, we support the user by providing default setups that comprise different aspects. We do not claim these aspects to be complete, but they can be easily extended and adapted.

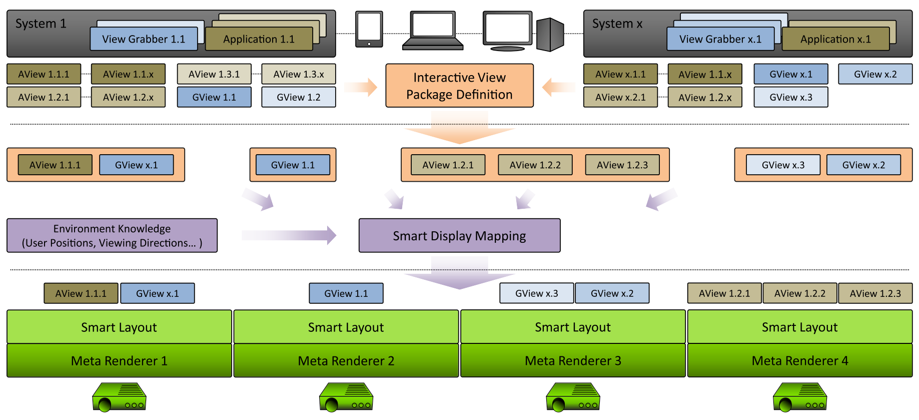

Fig. 1. Smart view management. Views from applications (A) and view grabbers (G) are grouped interactively into view packages to encode semantic. That semantic and the current environment state is regarded to assign views to display surfaces. If several views are to be presented at one display (managed by a meta renderer), the views are layouted according to semantics and display characteristics.

Task (present, compare): We support two typical tasks: *present* and *compare*. The first task typically means that a view shall be displayed to a maximum of users in a way that a good visibility is provided. The second task addresses the issue that users often have to compare different views. Hence, to support an appropriate comparison the views and associated display surfaces should be close to each other.

View content (fragile, robust): We distinguish two modes related to the resizing of a view. A *fragile* view means that the content cannot be distorted, since important information may be lost in this way. In contrast, *robust* means that the view can be resized.

Scenario (presentation, discussion): This aspect describes the general setting of a smart environment that (in our case) reflects the view directions of the auditory. For example, in the presentation scenario the users generally visually follow the presenter. In contrast, within the discussion scenario, any single user has their own view direction e.g. for acquiring necessary information from the closest display.

Based upon these aspects, we define customizable presets. For example, if several views origin from the same application on one device, we assume a compare task and build one *compare* view package. Typically view content can be resized *robustly* in case of the prepared option since alternative views could be generated. In contrast, the ad-hoc option leads to *fragile* view packages to be on the safe side. Through this approach of interactive view package generation, we switch from complex hardware configuration and hardware presets to an expedient configuration of content and interests. We realized the view package generation by a lightweight and easy-to-use drag-and-drop GUI.

5 Smart Display Mapping

The next step is the automatic assignment of view packages to display surfaces. For this purpose, we introduce a smart display mapper that reduces configuration effort, thereby improves efficiency. Since the idea of view packages is basically to encode the semantic between different views, it is the task of the display mapper to materialize that semantic within the smart meeting room. Before explaining our approach, we briefly describe the basic idea of display mapping as it has been proposed in [10].

5.1 Basic Approach

In [10] the problem of assigning views of interest to display surfaces is defined as an optimization problem, optimizing spatial quality q_s, temporal continuity (quality q_t) and semantic proximity (quality q_p):

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

The different qualities are weighted (a, b, c) and m_{t-1} and m_t denote consecutive mappings. The spatial quality rates the visible spatial arrangement of views and comprises influencing factors like the visibility of display surfaces towards users, rendering quality of projectors upon those surfaces and importance of views. Temporal quality q_t judges assignment changes that may be distracting and q_p rates the correspondence of spatial and semantic proximity of views.

The quality function 1 has be to maximized, which is a NP-hard problem. Therefore, Heider [10] uses a distributed greedy algorithm to randomly generate several mappings to associate devices, views and display surfaces. These mappings have to be judged with the above function to find the optimum. Heider uses only elementary approximations (e.g., to calculate visibility) that do not account for important aspects like readability. Hence, we introduce some purposive enhancements that improve the display mappings.

5.2 Our Approach

Improved Spatial Quality The term spatial quality q_s significantly influences the spatial arrangement of views, but usability is not represented adequately yet: In [10] only the angle between a surface's normal and the user/projector is used to judge visibility and rendering quality. Thus, current viewing directions, the users distance towards the surface and available projection directions are not regarded. As a result, users may have to turn around repeatedly, may not be able to perceive the displayed content and steerable projectors may not hit the surface due to a limited deflexion range.

To improve the situation, we consider additional geometrical measurements. First we regard the position of display surfaces according to the users current body orientation and to the projection directions. Concerning users we use a cosine function that is scaled to fall off towards the borders of maximum head deflexion and results in 0, if the maximum is exceeded. If a steerable projector is not able to hit a surface (with tolerance) due to a limited deflexion range, the rendering quality is assumed to be 0. These calculations are combined with the former angle measures by multiplication.

Secondly, we judge distance since angles between users and surfaces do not reflect the perceivable information or readability. But Euclidian distance is ineligible as for heterogenous displays: A distance of 2 m to a smart phone has a different effect than to a power wall. Due to that, we introduce a field of view (FOV) calculation as the FOV correlates with distance and is more meaningful in such scenarios. A horizontal FOV of 20° and a vertical FOV of 60° are known to be the focus area of human perception [13] and thus any difference is considered to have a negative effect. Since perspective correct calculations are computationally complex, we use a simplifying approximation considering horizontal and vertical distortions of a display surface (regarding the users point of view) independently. Thus, the approximated FOV γ_h and γ_v for a surface is

$$\gamma_{\{h,v\}} = \arctan\left(\frac{\cos\alpha_{\{v,h\}} \cdot hd_{\{h,v\}}}{d + \sin\alpha_{\{v,h\}} \cdot hd_{\{h,v\}}}\right) + \arctan\left(\frac{\cos\alpha_{\{v,h\}} \cdot hd_{\{h,v\}}}{d - \sin\alpha_{\{v,h\}} \cdot hd_{\{h,v\}}}\right).$$

With that, α denotes the rotation angle of the surface, d is the distance of the user towards the surfaces center and hd is the half dimension of the surface. 180° have to be added, if $(d - \sin\alpha \cdot hd)$ is negative. We define the following rating functions for the FOV in [0..1]:

$$fov_h(\gamma_h) = \begin{cases} \frac{1}{20^n}\gamma_h^n, \ \ n > 1 & , \gamma_h < 20^\circ \\ cos\left(\frac{9}{16}\gamma_h - 11\right) & , \gamma_h \geq 20^\circ \end{cases};$$
$$fov_v(\gamma_v) = \begin{cases} \frac{1}{60^n}\gamma_v^n, \ \ n > 1 & , \gamma_v < 60^\circ \\ cos\left(\frac{9}{7}\gamma_v - 77\right) & , \gamma_v \geq 60^\circ. \end{cases}$$

A reduced FOV results in a loss of perceivable information and thus needs an increased penalty. An enlarged FOV may lead to less comfort (additional movements) and yields a lower penalty. Since both, $fov = \min(fov_h, fov_v)$ and the above angle measures are important, we combine them again by a multiplication.

Our improvements ensure that practical needs like readability of content, comfort (less movements) and realizability of mappings (e.g., steerable projectors) are presented adequately within the spatial view arrangement – and thus the perceived spatial quality is increased.

Reduced Temporal Quality Tests The above improvements in q_s ensure readability and less eye or head movements, but increase the computational costs. Thus, we examined equation 1 to somehow reduce complexity again.

We examined the term of temporal quality q_t, which ensures that views do not unnecessarily change the display surface they are assigned to, i.e. small changes shall have no effect. As realized in [10] this means considering all users, views and the corresponding visibilities again. We found that a reduction of assignment changes can also be reached by regarding only those circumstances, that definitely need an update of the current mapping. This way, a new mapping will replace the former only (1) if a device or user joins or leaves the smart meeting room or (2) if views change the display surfaces they are assigned to and the distance between the corresponding surfaces is bigger than a threshold and they remain there for a minimum time period (to prevent flickering). In this way, we do not consider temporal quality repeatedly within the optimization process and consequently dismiss q_t in equation 1. Instead we take an already optimized display mapping, analyze it, and materialize it if necessary. Thus, temporal continuity is retained and the computational costs are reduced, as the temporal analysis has to be carried out only in well defined situations.

Since meta renderers comprise more than simply a projector, those renderers are able to fade between former and currently assigned views and moreover may give visual clues e.g. where a view moves to.

Realization of Multiple Views on One Display A meta renderer is able to present multiple views via one projector by adapting and arranging the views (see sec. 6). To represent this ability within the display mapping optimization, we logically split the according projector into n identical logical sub-projectors, where n is the number of overall views. Running the optimization is done as usual, but every mapping that comprises the same view assigned to more than one of the belonging sub-projectors is rejected. Since all belonging sub-projectors are managed by one meta renderer, it is easy to obtain the necessary values, e.g. scaling quality, rendering quality, visibility.

5.3 Smart Realization of View Package Semantics

Although equation 1 already includes the term q_s, an implementation of semantic proximity is still missing. However, the view packages approach (see sec. 4) encodes a basic semantic, which can be used for an automatic assignment of views to display surfaces:

Task (present, compare): The *present* task can be seen as the standard functionality of the display mapper. But to realize a spatial separation of view packages, defined by different users, we define a slight semantic proximity for views of one package. In contrast, the *compare* task defines a high semantic proximity and thus needs a strong consideration. Thus, we use a basic realization of semantic proximity $p(v, v')$ as it can be used in q_p:

$$p(v,v') = \begin{cases} 0.5, & (v \in V) \wedge (v' \in V) \wedge (\text{task is } \textit{present}); \\ 1, & (v \in V) \wedge (v' \in V) \wedge (\text{task is } \textit{compare}); \\ 0, & \textit{else}; \end{cases}$$

where v, v' are views of the view package V. This way, there is no penalty if views do not belong to each other, a small penalty if they are from one package in a *present* task and a big penalty, if views shall be compared – on condition that views are not displayed close to each other (see fig. 2(b)).

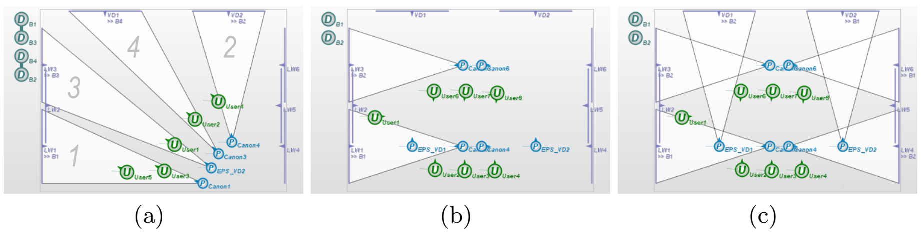

Fig. 2. Our smart meeting room as a floor plan screenshot of our room observer (U: users, P: projectors, D_{B1-B4}: views and LW_{1-6}, VD_{1-2} the canvases). (a) Belonging views are displayed close to each other due to *compare* packages. (b) Views are projected close to the speaker in the *presentation* scenario ignoring individual viewing directions. (c) During *discussion* an optimal visibility of views is aspired.

View content (fragile, robust): Meta renderers allow a scaling adaptation of views according to the screen of the displaying devices. In a *fragile* case, any scaling is forbidden and thus results in a zero scaling factor. If views are *robust*, the scaling factor is in [0..1) if a view is shrunk and 1 if it is enlarged. Within the spatial quality q_s we multiply the scaling factor with the measure expressing the rendering quality of a projector.

Scenario (presentation, discussion): Within the *presentation* scenario, all users visually follow the presenter. Thus, the presented views have to be arranged near the presenter. For that, we presume the connecting vectors between users and the presenter as the users' viewing directions despite the real directions (fig. 2(b)). In the *discussion* scenario, the visibility of the important content for a single user is decisive and thus views are arranged optimally according to the extended visibility issues (see sec. 5.2, fig. 2(c)).

The design of our smart display mapper allows for a dynamic reaction to changing situations within a smart meeting room. It regards our simplified semantic definition (sec. 4) by using the above semantic dependent automatic modifications within the optimization process. Finally, to realize the possible assignment of multiple views to one display surface and to complete our smart view management we make use of smart view layout presented next.

6 Smart View Layout

Automatically arranging multiple views on displays of different size is an open research problem. For instance, Ali suggests a force-based approach for the layout of dynamic documents requiring preprocessed content, templates and constraints [2]. Because of the ad-hoc characteristic of smart meeting rooms a preprocessing is generally not feasible. We chose a simplified approach. The major idea is to give a quick response by displaying a first layout. This layout is calculated by a spring force directed approach in combination with a pressure-directed resizing and is

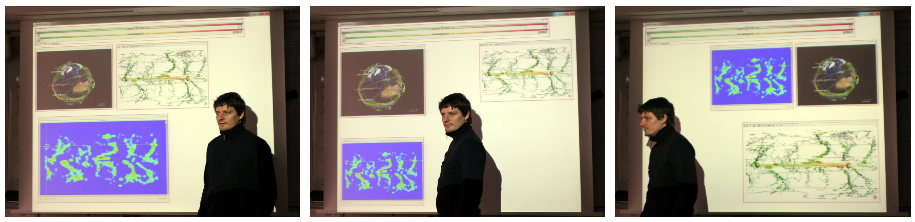

Fig. 3. Users in front of a display surface act as occluders and can be integrated into the dynamic layout of views to increase visibility.

improved subsequently. According to Ali et. al., missing constraints may result in artifacts. For this reason, we recalculate the layout repeatedly and use a quality function to rate the new solution and finally replace the currently displayed layout if a threshold is exceeded. This background process dynamically calculates and rates new layouts and thus dynamically considers condition changes, like e.g. changed user positions, views or display mappings.

Moreover, we are able to guarantee the visibility of views: Real objects within the smart meeting room that occlude parts of a display (e.g., a user in front of a screen) are represented as empty (non-visible) non-resizable elements within the layout. Those elements influence the forces affecting the views to be arranged, but remain fixed themselves (see fig. 3).

7 Results and Concluding remarks

In this paper we present a concept for a smart view management. We gather views from attendant systems – using a view grabber or software streaming – and make them available to the smart environment. Our simplified interactive definition of view packages encodes a basic semantic that reflects different aspects of working scenarios within a smart meeting room. The smart display mapper assigns views from view packages to display surfaces according to the encoded semantic and the current room state provided by sensors (positions, directions,. . .). The smart layout arranges multiple views on one display surface and reacts to changing conditions, too.

We implemented the smart view management in Java for a maximum independence of operating systems. It has been integrated into our smart meeting room in Rostock. Here, the positions of the users are tracked via Ubisense tracking system combined with probabilistic models for enhancing the accuracy and to determine viewing directions (e.g., by traces). In this way we provide a working solution to show views of different systems on different displays with regards to varying viewing conditions. Figure 4 shows a snapshot of our smart view management in use, demonstrating a presentation scenario. Here, the talk slides are defined as one view package with the semantic aspects *present, fragile, presentation* and hence, are shown in full resolution next to the presenter. Furthermore,

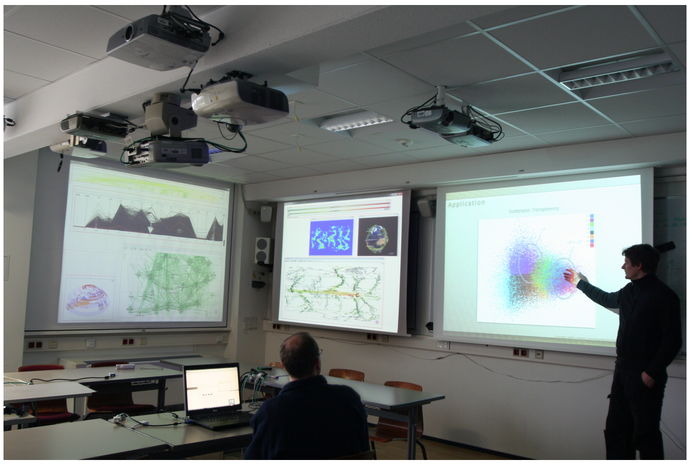

Fig. 4. Smart view management in use. Only a few semantic aspects have to be defined interactively, avoiding any further configuration effort.

eight views are displayed to be compared. These views are grabbed and combined to one view package with semantic aspects *compare, robust, presentation*.

Since view package definition is easily done via a lightweight user interface, but all layouts and optimizations according to semantics and the current room state are carried out automatically, we reduce the configuration effort significantly and hence increase verifiably (see [10]) the usability of a smart meeting room. Although our management uses optimization, the available display space is still limited. Thus, if too much views have to be displayed on to less display space, the work efficiency or the number of finally displayed views decreases. Furthermore, the processing time of display mapping and force based layout increases along with the number of views and the bandwidth and latencies of network connections are a limiting factor, too.

Nevertheless, we intend to enhance our smart view management even further. For example, since meta renderers have access to all views, the views could be easily enriched e.g. by view connecting arrows. Additionally we focus on a task based improvement of our system by generating and adapting view packages based on the user's current tasks and workflows. Future work will also concentrate on interaction to support users interacting with the multiple views across the boundaries of devices at hand.

Acknowledgments Axel Radloff is supported by a grant of the German National Research Foundation (DFG), Graduate School 1424 (MuSAMA).

References

1. Ali, K.: Adaptive Layout for Interactive Documents. PhD-Thesis, University of Rostock, Germany (2009)

2. Ali, K.; Hartmann, K.; Fuchs, G. and Schumann, H.: Adaptive Layout for Interactive Documents. SG 2008. LNCS, vol. 5166, pp. 247–254. Springer (2008)
3. Avidan, S. and Shamir, A.: Seam Carving for Content-Aware Image Resizing. ACM Transactions on Graphics (26):3, pp. 10.1–10.9, ACM Press (2007)
4. Beach, R.J.: Setting Tables and Illustrations with Style. PhD-Thesis, University of Waterloo, Ontario, Canada (1985)
5. Brumitt, B.; Meyers, B.; Krumm, J.; Kern, A. and Shafer, S.: EasyLiving: Technologies for Intelligent Environments. LNCS vol. 1927, pp. 12–29, Springer (2000)
6. Cook, D.J. and Das, S.K.: How Smart are our Environments? An Updated Look at the State of the Art. Pervasive and Mobile Computing, (3), pp. 53–73, 2007.
7. Encarnao, J. L. and Kirste, Th.: Ambient Intelligence: Towards Smart Appliance Ensembles. LNCS vol. 3379, pp. 261–270, Springer (2005)
8. Follin, J.-M.; Bouju, A.; Bertrand, F. and Boursier, P.: Management of Multi-Resolution Data in a Mobile Spatial Information Visualization System. In Proceedings of Web Information Systems Engineering Workshops (WISEW'03), pp. 92–99, IEEE Computer Society (2003)
9. Forlines, C. and Lilien, R.: Adapting a Single-user, Single-display Molecular Visualization Application for Use in a Multi-user, Multi-display Environment. In Proceedings of Advanced Visual Interfaces (AVI'08), pp. 367–371, ACM Press (2008)
10. Heider, Th.: A Unified Distributed System Architecture for Goal-based Interaction with Smart Environments. PhD-Thesis, University of Rostock, Germany (2009)
11. Heider Th. and Kirste Th.: Smart Environments and Self-Organizing Appliance Ensembles. In Mobile Computing and Ambient Intelligence: The Challenge of Multimedia, Schloss Dagstuhl, Germany, (2005)
12. Huang, J.; Bue, B.; Pattath, A.; Ebert, D. S. and Thomas, K. M.: Interactive Illustrative Rendering on Mobile Devices. IEEE Computer Graphics and Applications, (27):3, pp. 48–56, IEEE Computer Society (2007)
13. Kaufmann, H.: Strabismus. Georg Thieme Verlag Stuttgart (2003)
14. Pirchheim, C.; Waldner, M. and Schmalstieg, D.: Deskotheque: Improved Spatial Awareness in Multi-Display Environments. In Proceedings of IEEE Virtual Reality (VR'09), pp. 123–126, IEEE Computer Society, (2009)
15. Ramos, C.; Marreiros, G.; Santos, R. and Freitas, C.F.: Smart Offices and Intelligent Decision Rooms. Handbook of Ambient Intelligence and Smart Environments, pp. 851–880, Springer (2009)
16. Tan, D. S.; Gergle, D.; Scupelli, P. and Pausch, R.: Physically Large Displays Improve Performance on Spatial Tasks. ACM Transactions on Computer-Human Interaction, (13):1, pp. 71–99 (2006)
17. van der Vet, P.; Kulyk, O.; Wassink, I.; Fikkert, W.; Rauwerda, H.; Van Dijk, B.; van der Veer, G; Breit, T. and Nijholt A.: Smart Environments for Collaborative Design, Implementation, and Interpretation of Scientific Experiments. In Proceedings of AI for Human Computing (AI4HC07), pp. 79–86, (2007)
18. Wigdor, D.; Jiang, H.; Forlines, C.; Borkin, M. and Shen, C.: WeSpace: The Design, Development and Deployment of a Walk-Up and Share Multi-Surface Collaboration System. In Proceedings of Human Factors in Computing Systems (CHI'09), pp. 1237–1246, ACM Press (2009)
19. Youngblood, G.M.; Heierman, E.O.; Holder, L.B. and Cook, D.J.: Automation Intelligence for the Smart Environment. In Proceedings of International Joint Conference on Artificial Intelligence (IJCAI'05), pp. 1513–1514, Morgan Kaufmann (2005)