Display-Adaptive Window Management for Irregular Surfaces

Manuela Waldner, Raphael Grasset, Markus Steinberger, Dieter Schmalstieg
Institute for Computer Graphics and Vision
Graz University of Technology, Austria
{waldner | grasset | steinberger | schmalstieg}@icg.tugraz.at

ABSTRACT
Current projectors can easily be combined to create an everywhere display, using all suitable surfaces in offices or meeting rooms for the presentation of information. However, the resulting irregular display is not well supported by traditional desktop window managers, which are optimized for rectangular screens. In this paper, we present novel display-adaptive window management techniques, which provide semi-automatic placement for desktop elements (such as windows or icons) for users of large, irregularly shaped displays. We report results from an exploratory study, which reveals interesting emerging strategies of users in the manipulation of windows on large irregular displays and shows that the new techniques increase subjective satisfaction with the window management interface.

ACM Classification: H5.2 [Information interfaces and presentation]: User Interfaces. - Graphical user interfaces.

General terms: Design, Human Factors

Keywords: window management, irregular displays

INTRODUCTION
Large displays are becoming more commonplace in everyday environments, such as offices or meeting rooms, through the widespread adoption of inexpensive projectors. Besides, multiple projectors can be combined to seamless imagery, providing a large amount of display pixels while overcoming the limited screen-estate of standard monitors. A specific advantage of projection is that any empty surface can be turned into a display area, resulting in an *everywhere display* [22].

In such an everyday environment, physical discontinuities, like room corners or table-wall combinations, segment the resulting projected imagery. In addition, oblique projection angles and multiple overlapping projections result in possibly concave polygonal display outlines with unconventional display aspect ratios. While distorted images caused by oblique projection angles could be compensated by the projector hardware itself, the creation of seamless imagery on irregular, non-planar surfaces requires software-based compensation of the images. The resulting non-rectangular, non-planar projected imagery will be referred to as *irregular displays* in this paper.

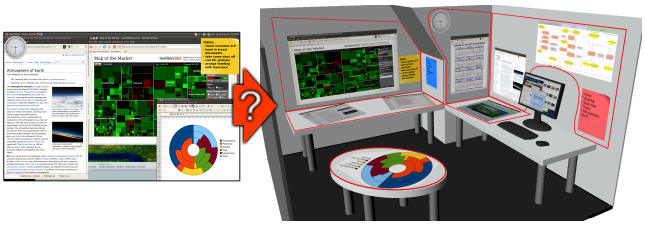


Figure 1: Window managers are designed for simple rectangular displays, such as monitors, single projector setups or tabletop displays. Bringing their functionality to arbitrary shaped irregular displays (red outline in the concept figure), is an open research question.

Unsurprisingly, irregular displays can be rarely found in actual working environments, as conventional window managers do not support such displays. Usually, window managers rely on a very simple spatial model of the available screen space: multiple adjacent output devices are treated as rectangles, with their respective size is only inferred from their pixel dimensions. Spatial properties such as real projection size, visible display outline, gaps between the displays, or orientation towards the user are not considered. Thus, window managers are well adapted for rectangular display devices, while conforming them to irregular displays is still an unresolved issue (Figure 1).

In contrast to previous work that tries to circumvent display irregularities, we hypothesize that irregularities need to be hidden neither from the windowing system nor the user. On the contrary, we believe that if the window manager is aware of the physical display form factors, it can actively support the user in creating and maintaining more useful spatial window layouts. Thus, our contributions are twofold.

First, we present three display-adaptive window management techniques, which actively exploit the knowledge of the physical display form factors to semi-automatically optimize the spatial window and desktop layout. Our techniques support the user in maintaining a "carefully coordinated" window layout [13] on a large, irregular display. Windows and desktop widgets are automatically repositioned to suitable

ITS'11, November 13-16, 2011, Kobe, Japan.

places inside the irregular display boundaries, avoiding other windows' important content and making use of spare display regions. We present a real-time implementation of our technique that has practical value as it is fully embedded in a widely used window manager (Compiz on Linux).

Second, we report the results of an exploratory study investigating emerging window management strategies to cope with display irregularities and comparing selective display-adaptivity techniques to conventional window management.

RELATED WORK

Previous research has demonstrated that window management on very large displays is a mentally demanding task for the user. Considering that the number of application windows increases on large displays [12], Bi and Balakrishnan [6] observed that users spend significantly more time on window management on large displays than on conventional monitors. In particular, users spend much more time moving and resizing windows. On monitors, minimizing and maximizing windows is a more frequent activity.

There are several reasons for this window management overhead. Andrews *et al.* [2] observed an increased amount of spatial window organization, such as piling and clustering of windows, which users applied to support their sense-making activities. Piling is also a well-known strategy for management of paper documents on the physical desk [17, 18]. Thus, interaction techniques to manage piled documents on virtual desktops have been researched and implemented in the past [1, 18]. However, the focus of these implementations has been solely on relatively small displays.

For large, seamless displays without any physical separation, Bi and Balakrishnan [6] found that users employ a focus and context separation when laying out their windows. In the peripheral context region, windows provide awareness of particular background information. Task management systems like *Scalable Fabric* [25] or *Kimura* [16] provide dedicated context display regions or separated background displays, where down-scaled representations of windows for suspended tasks are displayed for easy task reconstitution. However, these systems use a pre-defined separation of the available display space and do not dynamically adapt window management behavior to arbitrary display irregularities.

Hutchings and Stasko [13] observed that users "carefully coordinate" their windows on large displays to keep a small portion of occluded windows visible for direct access with the mouse. In this way, explicit window switching, like Alt+Tab, which has been described as "tedious" [11], can be avoided. However, manually maintaining such a carefully coordinated window layout is time-consuming. Researchers therefore suggested automatic window layout techniques to reduce window overlaps and thereby increase the visibility and accessibility of content on the display. Tiled window managers (*e.g.*, [15]) resize windows to display all windows side-by-side without overlap. However, strict window tiling has not been well received in the past [8]. Semi-automatic techniques like overlap-avoiding dragging [5] or constraint-enabled window management [3] aim to avoid window overlaps by solely adjusting the window locations, while keeping the window size unmodified. Another approach deals with situations involving occlusion by temporarily altering the layout of windows [9, 27]. All of these techniques assume the display to be rectangular, without any physical discontinuities. Our approach differs, as we do not only resolve window overlaps to better exploit the large display space, but also adjust the window layout with respect to the physical display form factors.

Research on window management for multi-monitor settings has shown that users explicitly facilitated the partitioning introduced by the physical monitor bezels for task separation [11]. Grudin showed that users rarely span application windows across the monitor bezels. Thus, current window managers usually provide interaction techniques to snap application windows to monitor edges or to maximize windows to individual monitors. However, when using multi-projector displays, output device boundaries visually vanish, while physical irregularities may be introduced instead.

In a multi-display environment, Nacenta *et al.* [21] perspectively correct individual application windows to align with the user's field of view. Windows are rendered as if floating on a virtual plane in front of the user and may also span multiple discontinuous displays. The aim of this technique is to create the illusion of a seamless display spanning multiple surfaces, which is also a popular approach for rendering 3D content on continuous irregular displays (*e.g.*, [7]). However, visually compensating for display irregularities leads to a loss of display partitioning, which users appreciate for their task separation when working with application windows [11]. We therefore propose the opposite approach: instead of adapting window management to compensate for display irregularities, we demonstrate display-adaptive window management, which explicitly makes use of irregular display form factors.

DISPLAY-ADAPTIVE WINDOW MANAGEMENT

In contrast to previous approaches, which aim to fit an optimal rectangular screen area into the projection area (*e.g.*, [30, 24, 23]), our goal is to use *all* available projector pixels for showing desktop content. We achieve this by dynamically adjusting the desktop content to the physical display form factors. As a prerequisite for display-adaptive window management, we need a representation of the irregular display that can be interpreted by the window manager.

Our approach can be summarized as follows. In a first step, we capture the irregular projection surfaces – the physical setup which serves as projection area. From this three dimensional representation, we deduce the irregular display shape, which we store in a two dimensional map – the *display map*. This map forms the basis for our display-adaptive window management techniques, which we explain in detail after the map creation process.

Display Map Creation

The first phase of the mapping process captures the irregular projection surfaces (Figure 2(a)). For this purpose, we consider all surfaces that are lit by the multi-projector system and we construct a three dimensional display model of the unified projection surfaces (Figure 2(b)).

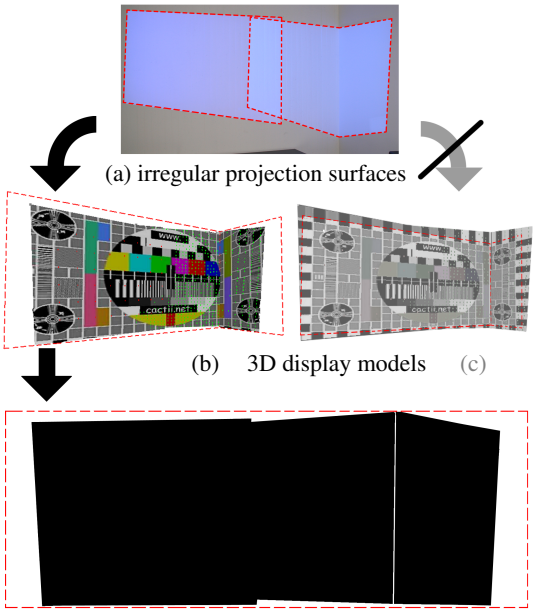


Figure 2: (a) The irregular projection areas (outlined in red) (b) are captured to create a 3D model of the display. (d) Unfolding the planar patches creates the display map, where black/white defines usable/unusable areas. The resulting virtual desktop outline is indicated as red ouline in (b), the traditional approach in (c).

In the second phase, the *display map* is generated. Therefore, we unfold the planar patches of the three dimensional irregular display model and create a two dimensional representation. By circumscribing a rectangle around this representation, we define the display map (*cf.* Figure 2(d)), which holds the entire irregular display and thus the *usable* (*i.e.*, visible) display pixels. This display map not only contains the outline of the irregular display, but also the physical corners (note the one-pixel line along the 90° corner in Figure 2(d)). Mind that circumscribing a rectangle of the desktop's aspect ratio around the 2D display representation yields a larger virtual desktop and a larger number of actively used pixels as compared to the more traditional approach to inscribe the desktop rectangle within this representation (compare Figures 2(b) and (c)). Using software-based warping and blending similar to [24], pixels from the display map can be assigned back to the individual projectors and displayed on the irregular projection surfaces.

In our current implementation, we are constrained to irregular displays composed of continuous multi-planar surfaces (*e.g.*, adjacent walls, tables next to walls, or other planar architectural elements). However, the concept could also be employed for any – potentially discontinuous – display configuration, as long as a meaningful planar representation can be established. With physical gaps between adjacent projection areas, additional challenges for navigation with indirect pointing devices and visualization of off-screen content need to be addressed, which goes beyond the scope of this paper.

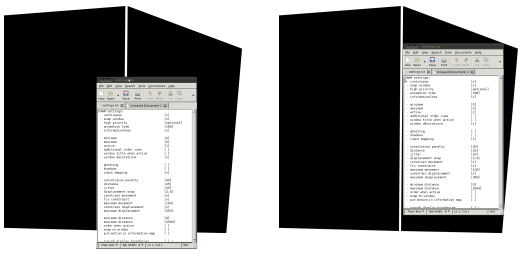


Figure 3: (a) Releasing a dragged window (text editor) outside the irregularly shaped display outline (display map of Figure 2) triggers display-geometry snapping and (b) the window is placed in the most suitable location such that only a small number of pixels will be mapped outside the irregular display.

Display-Window Manager Mapping

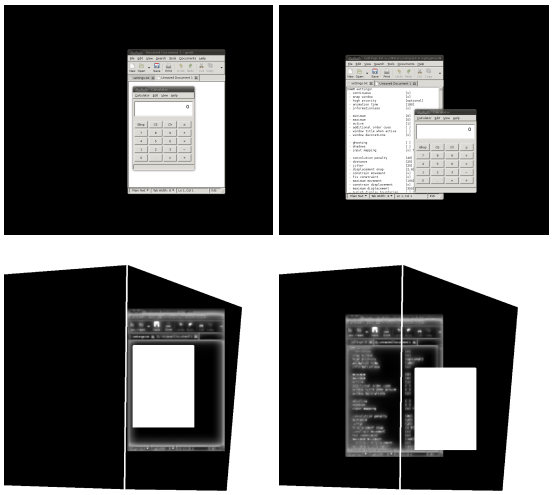
To incorporate the knowledge of the irregular display into a window management system, we build on our former work on window managers for regular displays [29]. Our previously developed system, Importance-Driven Compositing Window Management (IDWM), optimizes the desktop layout by minimizing the occlusion of "important" desktop elements on the display. It thereby relies on *importance maps*, which show the pixel-wise distribution of important content in the individual windows (*window importance maps*) and on the final desktop composition (*desktop importance map* accumulated of the display map and all window importance maps). To determine the window importance maps, we constantly calculate the visual saliency [20] of the window textures, describing the amount of visual attraction of the individual window regions as a measure of importance. An example of a window importance map is shown for the text editor in Figure 4.

To enable display-adaptive window management, we use the display map as input for the desktop importance map: Unusable pixels outside the irregular display and along physical edges are set to a maximal importance value, which prevents IDWM from placing content in these areas. Additionally to this binary mapping, we can include information about preferred display regions. The user can manually modify the suitability of certain display regions in the display map image. For instance, she could add gradients to the display map describing the decreasing suitability for placing desktop elements far away from the user.

Based on this infrastructure, we implemented three display-adaptive window management functions for irregular displays: display snapping, semi-automatic window coordination, and desktop widget layout.

Display-Geometry Snapping

In conventional window management systems, monitor bezels often act as *sticky edges*, perfectly aligning the window along the edge if it was released within a certain distance threshold. However, on irregular displays, display outlines are rarely rectangular and potential physical discontinuities are not recognized by the window manager.



(a) (b)

Figure 4: Moving a high-priority dragged window (calculator) to a small, irregularly shaped display region (display map of Figure 2): (a) as the underlying window (text editor) does not have any important content at the drop region, it is not necessary to re-arrange it, (b) if the underlying window shows salient content, it is re-positioned to keep important areas visible. The top row shows the screen shots, while the bottom row shows the desktop importance used for calculating the information overlap.

We propose a *display-geometry snapping* technique, which can be activated when the user releases a dragged window. For this matter, we adjust the location of a dragged window by analyzing the display map. If the window is moved across the irregular boundary of the display, display-geometry snapping will find a suitable location for the window, so that a minimal number of unusable pixels is covered by the window, while the required displacement is also kept to a minimum, as illustrated in Figure 3.

Semi-Automatic Window Coordination

Projected displays composed by multiple projectors usually offer a large screen estate and thus sufficient space to manually maintain a "carefully coordinated" window layout [13]. To decrease the maintenance effort for the user, we propose a semi-automatic window coordination technique. Conceptually, it works similar to *overlap-avoiding dragging* [5]: As the user releases a dragged window, either the underlying windows are re-positioned or the position of the dragged window is adjusted to avoid information overlap. In contrast to overlap-avoiding dragging [5] based on a purely rectangular screen layout, windows will only be re-positioned if they occlude important information according to the desktop importance map.

As overlap-avoiding dragging [5], we support two different dragging modes that either treat the dragged window with high or low priority. In the high-priority mode, the location where a dragged window has been released by the user is not modified by the window manager. However, other windows slightly adjust their position if the released window or any other higher prioritized window covers important content, as illustrated in Figure 4. In the low-priority mode, the existing layout of the other windows is stable, while the location of the dragged window may be adjusted when important content in other windows is about to be covered.

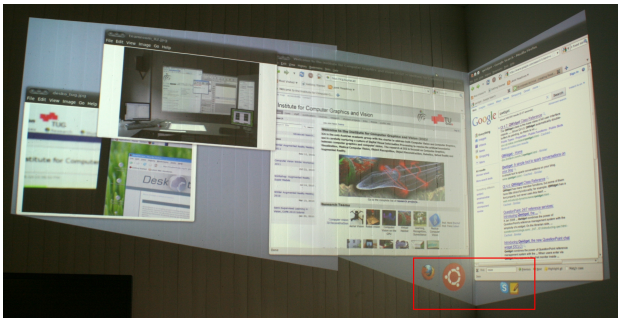


Figure 5: Desktop widgets on an irregular display are subject to semi-automatic window coordination and will therefore be gradually moved to peripheral areas where no larger items can be reasonably placed.

Since our scenarios are targeted towards projected displays and therefore comparably large display surfaces, semi-automatic window layout decreases the manual window layout optimization effort for the user. In combination with the previously described display-geometry snapping, windows do not only avoid important content of other windows, but also positions outside the irregular display boundaries. As a consequence, important window content is made visible and windows can efficiently be accessed.

Desktop Widget Layout

The non-rectangular outlines of irregular displays make it difficult to place common rectangular desktop elements, like window lists, application launchers, or notifiers. In current windowing systems, these elements are usually arranged in menus along the display boundaries and therefore will be partially lost when cropping the screen along its irregular outline. In contrast, the concave regions of an irregular display's boundaries remain largely unused by conventional desktop elements, like windows or menus.

As an alternative to conventional desktop menus and panels along the screen bezels, we are using free-floating desktop widgets, which are nowadays increasingly popular in conventional windowing systems, such as clocks, calendars, or application launchers. In our window manager, desktop widgets are subject to low-priority semi-automatic window coordination. As a result, the small widgets are gradually moved towards small display areas where no other desktop content is likely to be positioned (Figure 5).

IMPLEMENTATION

Display-adaptive window management is split into two components: an offline calibration process to gather the physical display form factors and the runtime component integrated into a fully functional window manager (Compiz).

Display Geometry Acquisition

Considering that our application area in irregular displays is for conventional indoor geometries, like offices or meeting

Figure 6: Unmodified output image for the irregular display in Figure 5.

rooms, we rely on a piecewise planar approximation of the individual projection surfaces, as illustrated in Figure 2(b). Such a model can be automatically created using a camera-assisted offline calibration procedure, as described in [28].

From this polygonal three dimensional model, the approximated display surface is mapped onto a planar representation using a texture mapping technique (as described by Raskar *et al.* [23]), from where the display map is derived (Figure 2(d)). In addition, homographies describing the individual polygons' distortions towards the circumscribed desktop rectangle and alpha masks describing linear ramps to ensure uniform brightness across projection overlap regions [24] are derived from this representation.

Window Manager Integration

Display-adaptive window management is implemented as an extension for the compositing window manager *Compiz* for Linux. At start-up, the window manager is supplied with the display map, created during the offline calibration process. Window importance maps are constantly updated by determining the saliency maps [20] of the individual window textures. Window importance maps are additionally augmented if changes in background windows have been registered by the window manager.

Window layout optimization is based on the IDWM algorithm [29], which has originally been introduced for see-through windows. For display-adaptive window management, the algorithm aims to minimize the amount of important information covered by other desktop elements, while keeping the displacement of the elements low. The layout algorithm is formulated as an iterative optimization problem over all possible desktop element locations and is executed on the GPU to ensure interactive frame-rates.

The layout optimization procedure is initiated when a dragged window is released and takes up to a user-defined time period to ensure a smooth animation. In each rendering step, the possible displacement is constraint to a small surrounding region, so the algorithm can perform sufficiently fast and the user is supported in keeping track of the layout changes. For our experiments, we used an animation time of 500 ms and constrained the maximum window displacement to 60 pixels per frame.

Finally, the resulting desktop image with optimized spatial window layout is subject to warping and blending to compensate for distortions (caused by oblique projection angles) and non-uniform brightness (caused by overlapping projections). Homography-based warping and blending of projector overlap regions, as proposed in [24], are conducted in an additional rendering step [28]. Figure 6 shows the original projected imagery for the adjusted display in Figure 5.

EVALUATION

To initially evaluate selective features of display-adaptive window management and to assess the future directions for window management on irregular displays, we conducted an exploratory user study.

On an irregularly shaped display, users were asked to solve an information analysis task involving multiple application windows with two window management techniques: conventional window management without spatial awareness and window management with selected display-adaptivity features.

Research Questions

The aim of the experiment was to assess two research questions:

Q1 *How do users manage their windows on an irregularly shaped, non-planar display?*
Although window management strategies have been explored for very large displays [6, 2] and multi-monitor settings [11, 12, 13], there has so far been no attempt to observe window management behaviors on irregularly shaped displays. We informally observed the participants and established a thinking aloud protocol to initially assess the basic suitability of irregular displays for conventional window management, and to furthermore discover emerging interaction patterns to cope with the situation.

Q2 *Does semi-automatic window coordination support users in managing windows?*
For large-scale displays, it has been demonstrated that window management overhead for the user increases – especially for basic operations, like moving and resizing windows [6]. This management overhead is partially caused by an increased number of open application windows [12]. The attempt to make most of the information in application windows visible and directly selectable [13] lead to an increased amount of spatial organization [6, 2], as well as the assignment of windows to different monitors [11]. Thus, we hypothesize that automating some of these activities, by providing semi-automatic window coordination and display-geometry snapping, will decrease the amount of manual window management operations and thereby increase the user's performance in an information analysis task.

Participants

We recruited 8 experienced computer users from a local institute (1 female, 7 male, aged 27 to 33). Five users were primarily using Microsoft Windows as their main operating system, two Linux, and one Mac OS X. All users stated that they worked with a dual-monitor setup on a regular basis, where the size of the larger monitor was given as 22" by one user and 24" for the others.

Conditions

To accomplish the information analysis task, we limited the available window interaction techniques to a single activity which is known to be used frequently on large displays [6]: window moving. Resizing, maximizing, minimizing, or closing of windows was not supported, so we could evaluate this single aspect in full detail. We also deactivated any window switching techniques like Alt+tab, solely allowing users to move windows by dragging them by the title bar. The limited set of possible activities was chosen not to leave the user with a big choice of options in such an unknown environment.

Users had to accomplish the task with two window management techniques:

Manual window management (M) was limited to the ability to drag a window and to change the stacking order by directly clicking within the window's boundary.

Display-adaptive window management (DA) also required the users to drag the windows manually. However, windows were snapped to be entirely contained within a planar, fully visible display area, if dragged outside or if the dragged window was released on top of a physical corner. In addition, it supported semi-automatic window coordination in the high-priority mode on demand. In a pilot study, we discovered that constant semi-automatic coordination was perceived as too "patronizing". Therefore, we provided it as an optional feature: if pressing the Start key while releasing a dragged window, underlying windows were subject to semi-automatic coordination. Conventional dragging of windows did not influence the spatial window layout of underlying windows.

For both techniques, we enabled image warping and blending to compensate for projection discontinuities.

Apparatus

The experiment was conducted on an irregular display driven by two XGA projectors, connected to a PC running Ubuntu 10.04. Accordingly, the display spanned over two planar display regions, where the left region was larger and also contained the overlap region between the two projectors. Due to the strongly oblique projection angles, some parts of the display suffered from noticeable interpolation artifacts when applying warping and blending – in particular the right half of the left display area. Our irregular display configuration was simulating standard office environments where space restrictions often require non-optimal projector setups, resulting in similar irregularities. The participant was sitting on a table facing the left display area. The windows were controlled by a conventional pairing of mouse and keyboard. Figure 7 shows our apparatus.

Task

In each trial, users were presented eight to ten small windows containing a picture of a car, its name, and other attributes, such as price, power, and mileage. Additionally, a sightly larger main window contained instructions for the task. For each trial, a new set of windows was loaded – initially in a cascaded arrangement, with the main window placed on top of the cascaded window stack. Using these windows, the users had to solve three task types (following typical window management tasks presented in [15]):

Figure 7: Irregular display setup for the evaluation of display-adaptive window management techniques.

Sort: Users were asked to sort the content windows according to some attribute (*e.g.*, the price of the cars) in a linear sequence. We did not explicitly instruct the users what the resulting arrangement should look like. They were only told to make the ordering clearly visible and understandable. This task was chosen to represent task environment setup activities.

Count: To simulate a sequential scanning task, we asked our users to count all cars of a given brand. Thus, they had to sequentially visit all windows and count the number of occurrences. The number then had to be entered in the main window.

Compare: As a complex comparison task, we asked the users to find the most suitable car by evaluating three given parameters (*e.g.*, the cheapest car with at least 10 km/l mileage and at least 5 seats). Thus, they had to sequentially scan all content windows to filter those cars violating the given constraints. Subsequently, they had to scan the remaining windows for the most appropriate parameters. The most suitable candidate then had to be selected by directly clicking a button in the content window.

Procedure

For each window management technique, users had first a practice period with one repetition for each type of task. During this practice, users were encouraged to "think aloud", to describe their window management strategies, their emotional response to the physical environment or the interaction technique. Subsequently, the users had to accomplish two repetitions of each type of task in an actual trial, where we logged task completion times and correctness. The sequence of window management techniques was counter-balanced across the participants. Additionally, the task sequence and the initial window stacking order was randomized.

After each run, users had to fill out a questionnaire. After both runs had been accomplished, users were asked to assess the two window management techniques, indicate how much they used the display-adaptivity features, and how they liked the display setup overall. A semi-structured interview was conducted at the end of the experiment. Additionally, the experimenter took notes during the practice and actual trials.

As we collected a considerable amount of observations and user feedback, we issued the users a complementary follow-

up questionnaire after the experiment. The questionnaire contained informal observations and statements by users about the display arrangement, their employed window management strategies on the irregular display, and suggestions how to improve window management in such an environment. Users were asked to indicate how much they agree with the statements: these *agreement* values were used for evaluating research question Q1 (general window management strategies on irregular displays).

RESULTS

In this section, we discuss the results of our experiment by presenting observational evidence and informal user feedback. In addition, we report on task completion times (measured in ms), questionnaire results (7-point Likert scale), and results from the follow-up questionnaire, expressing the agreement of the users with certain observations, feedback, and suggestions for future research directions.

Emerging Window Management Strategies

In the manual window management condition – in particular during the practice period – we observed emerging window management strategies of our users to deal with the irregularities of our display setup. Users mentioned several reasons why they would *not* want to have such a display in their offices (*e.g.*, the non-rectangular outline, the low resolution, or the high position of the display), where the most agreed point of criticism was the blur in some areas, which lead to readability problems. Interestingly, the physical corner was judged as rather useful by most participants, and some users mentioned they *"explicitly used"* the physical separation to create meaningful spatial window arrangements. Although we observed that users occasionally placed their windows across the physical corner in the manual condition (for six participants), users largely denied that they did not care about the window placement with respect to the physical corner. Users mentioned the reduced readability and *"awkward interaction"* as reasons not to span windows across the corner. Also, the irregular display outline was not particularly disliked, although participants largely agreed that having the outline visible would have been an advantage. However, users did not invest much effort to keep windows within the visible display area. One user explained it as: *"I kept the important windows in the inner part of the display, and for the others, I did not care if they were cropped"*.

Users agreed that they established an explicit strategy how to make use of the right display area. Some users reported that they used the right area for placing persistent background information which did not require any further intervention, such as the main window. This window was carefully positioned and was kept uncovered during the entire task, so they could quickly access the information by just turning their head. Four participants established a "dump pile" on the right display area, as illustrated in Figure 8(a). Windows identified as irrelevant were quickly dragged to the right display half – usually without even looking.

Interestingly, most users agreed that the right area would have been useful even if the option to minimize or close windows had been available. One user shared his opinion by saying *"It would have cost more time to click the close button. Moving the window to the right was faster."* Another user mentioned the ability to quickly re-acquire the information as more appropriate compared to closing the window. This corresponds to the observed tendency to keep more windows open if sufficient display space is available [12].

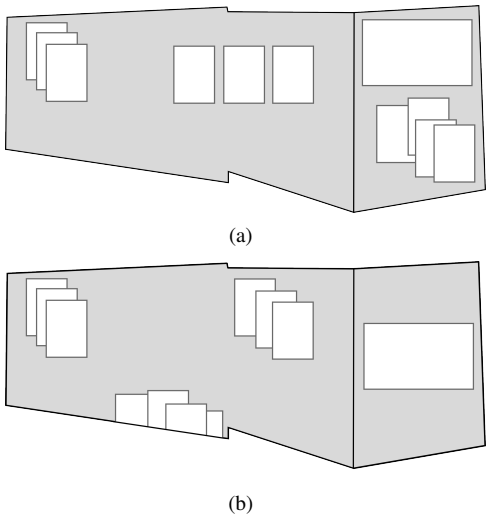


Figure 8: Exemplary window piling strategies observed during the experiment, illustrated on the 2D display map of the employed display setup: (a) from the original, cascaded stack on the left, useful candidates were linearly aligned in the focus region. The main window was carefully placed in the context region, next to casually piled up irrelevant windows. (b) Others arranged relevant candidates in a pile and placed irrelevant windows partially outside the visible display outline on purpose.

Two users created dump piles by dragging windows partially outside the visible display outline (Figure 8(b)). Two participants employed both strategies for placing dump piles (Figure 8(a) and (b)), depending on the task type.

It has to be noted that our windows were rather small (in term of pixels) and that the overall number of display pixels (2560x1024) was lower than our participants' everyday working environment. Thus, assuming a similar control/display gain of the mouse, the display could be traversed with comparably little mouse movement. Still, some users mentioned that dragging the windows to the right display area caused a lot of effort *"because the display is so large"*. Therefore, they suggested simple mouse gestures or buttons in the window title bar to quickly relocate windows to distant areas. Especially the user suggestion of having a "throw" gesture (similar as proposed by Geißler [10]) was well received. Also, having the possibility to move multiple windows concurrently (either manually grouped or existing piles) was suggested to decrease window management operations (similar to *snapping windows* [4] or *storage bins* [26]).

Display-Adaptive Window Management

Contrary to our expectations, our display-adaptivity features could not enhance the users' performance. Task completion times for the three task types were almost equal, with

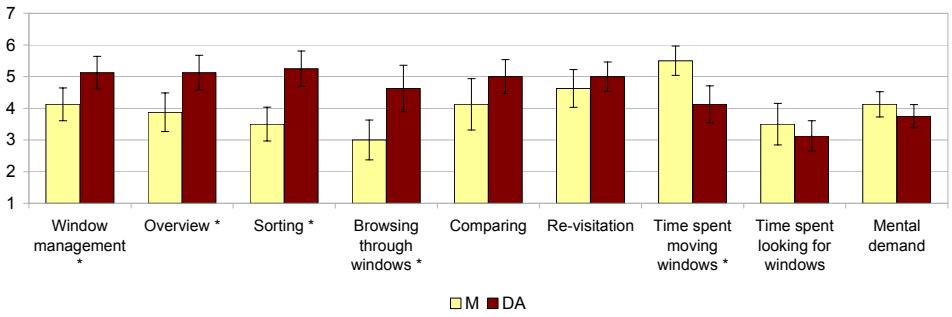


Figure 9: Questionnaire results for comparison of manual window management (M) and display-adaptive window management (DA) on an irregular display (mean and standard error on a 7-point Likert scale). Significance found using Wilcoxon Signed-Rank tests ($p < .05$) is indicated by *.

display-adaptive window management being even slightly slower than manual window management (sort: $t_7 = -.460$, $p = .660$; count: $t_7 = -1.611$, $p = .151$; compare: $t_7 = -.172$, $p = .868$; using paired t-tests). Correctness values were not evaluated separately, as only three questions were answered incorrectly in sum (all in the compare tasks). However, subjective assessment indicated that users appreciated display-adaptive window management for sorting, browsing, and gaining an overview (Figure 9). Overall, users ranked display-adaptive window management slightly higher than manual window management ($r_{da} = 5.0$ and $r_m = 3.375$), which is borderline significant ($Z = -1.930, p = .054$; using Wilcoxon Signed-Rank test).

In the post-study questionnaire, users had to indicate whether they explicitly employed display-geometry snapping or semi-automatic window layout (from "not at all" to "very often" on a 7-point Likert scale). Display snapping usage was rated low with 2.63 on average, while subjective usage of semi-automatic window layout was higher with 5.38. Most users never placed windows close to the display boundary and if so, only irrelevant windows where the placement was not considered as important. Two users intentionally wanted to create dump piles by placing windows partially outside the visible display region (*cf.*, Figure 8(b)), so display-geometry snapping actually interfered with their intentions.

In contrast, the semi-automatic window layout feature was facilitated by most users. We observed two usage types: Users employed it as an "explode" tool to resolve piles of windows – either the initial window cascade at the beginning of the task or manually created piles in the middle of the task. Users commented that this was a fast way to get a quick overview. However, one user demanded more control about the "force" of the tool. Some users expressed the wish for a more "tidy" arrangement, such as a regular grid, for easier visual comparison. The second usage type for semi-automatic window layout was to "squeeze in" windows into an existing spatial layout – usually towards the end of the task. This was a common approach to solve the sort task.

Users also complained that the semi-automatic layout feature sometimes resulted in an non-intuitive window layout or destroyed a previously established spatial layout. Indeed, we observed that our initial approach to prioritize windows for the layout algorithm according to their stacking order (*i.e.*, their recency of use) may lead to unpredictable results on a large display. A more appropriate priority queue would lay out the windows according to their proximity to the dragged window, so windows gradually move away from the focus window.

DISCUSSION

Results of our experiment show that users established unconventional window management strategies to cope with the size and irregularities of the display. As diverse patterns emerged among the participants, our display-adaptivity features did not support all the users as much as initially expected. However, from our observations and user feedback, we could establish future research directions to support users in their emerging window management strategies.

Q1 *How do users manage their windows on an irregularly shaped, non-planar display?*

Irregular displays are not the most obvious choice for information work with conventional window management interfaces. A common goal in related research therefore has been to visually compensate for irregularities in projected displays (*e.g.*, [7, 21]). However, our exploration indicates that users do not necessarily dislike certain irregularities, such as physical corners or non-rectangular projection outlines. They explicitly facilitated physical discontinuities to separate their workspace into *focus areas*, where their primary windows were located and most interaction took place, and *context areas*, where irrelevant windows were casually piled up and persistent background information was placed.

According to user feedback, we identified two types of windows in context areas of irregular displays: First, *background windows* hold valuable context information, which requires frequent visual access but little interaction. Background windows are conventionally dragged into a context region and manually positioned by the user. Important information should remain uncovered by adjusting the existing windows' positions, if possible, so quick visual scanning is supported.

Second, *dump windows* are (temporarily) irrelevant windows, which are moved to a distant location to keep valuable focus areas unoccupied. As large irregular displays obviously increase the subjectively perceived mouse navigation effort (independent of their resolution), simple mouse gestures (like

throwing [10]) should support the user in easily relocating the window to the context area. After the throw-gesture, windows should snap to a suitable region – neither occluding any background information window, nor spanning across a physical corner or cropped display outline. Dump windows should be scaled down so recognizability is supported, while keeping the amount of occupied space to a minimum. This concept is similar to *montages* in the *Kimuara* system [16], where windows of suspended tasks are scaled down, grouped into piles, and presented on a peripheral display.

Some users indicated that dump windows in context areas should be smaller (*e.g.*, showing only a thumbnail or cropped version of the window). The most important properties of these windows are that they can be easily moved back to the focus area and they are easily recognizable, according to our users. Content readability was judged as less important. The concept of scaled-down [25] or cropped [14, 19] context windows has already been suggested in previous work for planar large displays and could be similarly employed for context areas of irregular displays.

Q2 *Does semi-automatic window coordination support users in managing windows?*
Contrary to our expectations, users rarely made use of the display-geometry snapping facility. We observed that most windows were never located close to any display boundary. If windows were dragged close to the border they were usually not relevant. In contrast, users commented positively on snapping to the physical edge. Making snapping less aggressive at the boundaries, compared to physical edges, may increase user acceptance.

In contrast, semi-automatic window layout, was employed frequently by most participants – mainly to "explode" piled up windows or to "squeeze in" windows into an existing spatial layout without manually adjusting the remaining windows. While the main purpose of the "explosion" of existing piles was to increase visibility for visual comparison and *"to get a better overview"*, users disagreed about their anticipated exploded window layout: about half of the user tended towards a unordered layout, while the other half clearly preferred a regular grid layout.

In the future, we require a better prioritization criterion than the currently used reversed window stacking order to improve the quality of the resulting window layout. Sorting the windows according to their proximity to the dragged window for the layout algorithm seems to be a promising measure. Furthermore, users suggested to show a preview of the resulting window layout and the opportunity to interactively control the "force" of the semi-automatic coordination.

CONCLUSIONS AND FUTURE WORK

We have presented a system assisting users of large irregular displays in creating and maintaining a meaningful spatial window layout. An exploratory evaluation revealed that users do not categorically dislike irregular projected displays for knowledge work with conventional application windows. They rather established window management strategies to cope with the emerging display irregularities. We observed a clear tendency of the participants to divide the display into a close focus area and a distant context area. In contrast to planar displays, where the transition from focus to context is rather blurred [6], users facilitated the physical corner as hard boundary. Compared to conventional window management, display-adaptive window management does not increase task performance for knowledge work tasks, but user satisfaction is slightly higher.

In order for our display-adaptive window management concept to better support the user in dealing with irregular displays, we need to incorporate an increased amount of high-level information. Physical discontinuities – either introduced by irregularities in projected displays or by employing discontinuous displays – need to be analyzed with respect to the users' seating arrangements, so they can be automatically segmented into physically close focus and more distant or less conveniently oriented context areas. Interaction techniques, like throwing [10], and multi-window operations (*e.g.*, [4, 26]), can support the user in quickly moving windows between these focus and context regions. Display-geometry snapping and semi-automatic window layout are then responsible to locally adjust the window layout according to some given high-level information. For instance, they ensure that windows do not span physical discontinuities, do not cover background windows, and act as foundation to explode window piles on demand, similar to previous desktop piling techniques (*e.g.*, [1, 18]).

In the future, the concept of display-adaptive window management should also be extended beyond optimizing the window layout. Knowledge of display form factors may also influence the size of the windows or their rotation, when incorporating horizontal display areas. In addition, we can facilitate more information about the display geometry, such as projector overlap regions or strongly oblique projections and the user's viewing angle, to avoid information placement in display regions with low image quality or visibility for the user.

Acknowledgements

The authors would like to thank Erick Mendez for providing parts of the implementation.
This work was funded by the Austrian Research Promotion Agency *FFG* under contract *BRIDGE 822716* and the Austrian Science Fund *FWF*: *P22902*.

REFERENCES

1. Anand Agarawala and Ravin Balakrishnan. Keepin' it real: pushing the desktop metaphor with physics, piles and the pen. In *Proc. CHI 2006*, pages 1283–1292. ACM, 2006.

2. Christopher Andrews, Alex Endert, and Chris North. Space to think: large high-resolution displays for sensemaking. In *Proc. CHI 2010*, pages 55–64. ACM, 2010.

3. Greg J. Badros, Jeffrey Nichols, and Alan Borning. Scwm: An extensible constraint-enabled window manager. In *Proc. USENIX 2001*, pages 225–234. USENIX Association, 2001.

4. Michel Beaudouin-Lafon. Novel interaction techniques for overlapping windows. In *Proc. UIST 2001*, pages 153–154. ACM, 2001.

5. Blaine A. Bell and Steven K. Feiner. Dynamic space management for user interfaces. In *Proc. UIST 2000*, pages 239–248. ACM, 2000.

6. Xiaojun Bi and Ravin Balakrishnan. Comparing usage of a large high-resolution display to single or dual desktop displays for daily work. In *Proc. CHI 2009*, pages 1005–1014. ACM, 2009.

7. Oliver Bimber, Gordon Wetzstein, Andreas Emmerling, and Christian Nitschke. Enabling view-dependent stereoscopic projection in real environments. In *Proc. ISMAR 2005*, pages 14–23. IEEE Computer Society, 2005.

8. Sara A. Bly and Jarrett K. Rosenberg. A comparison of tiled and overlapping windows. *SIGCHI Bull.*, 17(4):101–106, 1986.

9. Olivier Chapuis and Nicolas Roussel. Copy-and-paste between overlapping windows. In *Proc. CHI 2007*, pages 201–210. ACM, 2007.

10. Jörg Geißler. Shuffle, throw or take it! working efficiently with an interactive wall. In *Proc. CHI 1998*, pages 265–266. ACM, 1998.

11. Jonathan Grudin. Partitioning digital worlds: focal and peripheral awareness in multiple monitor use. In *Proc. CHI 2001*, pages 458–465. ACM, 2001.

12. Dugald Ralph Hutchings, Greg Smith, Brian Meyers, Mary Czerwinski, and George Robertson. Display space usage and window management operation comparisons between single monitor and multiple monitor users. In *Proc. AVI 2004*, pages 32–39. ACM, 2004.

13. Dugald Ralph Hutchings and John Stasko. Revisiting display space management: understanding current practice to inform next-generation design. In *Proc. GI 2004*, pages 127–134. Canadian Human-Computer Communications Society, 2004.

14. Dugald Ralph Hutchings and John Stasko. Shrinking window operations for expanding display space. In *Proc. AVI 2004*, pages 350–353. ACM, 2004.

15. Eser Kandogan and Ben Shneiderman. Elastic windows: evaluation of multi-window operations. In *Proc. CHI 1997*, pages 250–257. ACM, 1997.

16. Blair MacIntyre, Elizabeth D. Mynatt, Stephen Voida, Klaus M. Hansen, Joe Tullio, and Gregory M. Corso. Support For Multitasking and Background Awareness Using Interactive Peripheral Displays. In *Proc. UIST 2001*, pages 41–50, 2001.

17. Thomas W. Malone. How do people organize their desks?: Implications for the design of office information systems. *ACM Trans. Inf. Syst.*, 1:99–112, January 1983.

18. Richard Mander, Gitta Salomon, and Yin Yin Wong. A "pile" metaphor for supporting casual organization of information. In *Proc. CHI 1992*, pages 627–634. ACM, 1992.

19. Tara Matthews, Mary Czerwinski, George Robertson, and Desney Tan. Clipping lists and change borders: improving multitasking efficiency with peripheral information design. In *Proc. CHI 2006*, pages 989–998. ACM, 2006.

20. Erick Mendez, Steven K. Feiner, and Dieter Schmalstieg. Focus and context in mixed reality by modulating first order salient features. In *Proc. SG 2010*, pages 232–243. Springer-Verlag, 2010.

21. Miguel A. Nacenta, Satoshi Sakurai, Tokuo Yamaguchi, Yohei Miki, Yuichi Itoh, Yoshifumi Kitamura, Sriram Subramanian, and Carl Gutwin. E-conic: a Perspective-Aware Interface for Multi-Display Environments. In *Proc. UIST 2007*, pages 279–288. ACM, 2007.

22. Claudio Pinhanez. The everywhere displays projector: A device to create ubiquitous graphical interfaces. In *Proc. of Ubiquitous Computing*, September 2001.

23. Ramesh Raskar, Jeroen van Baar, Paul Beardsley, Thomas Willwacher, Srinivas Rao, and Clifton Forlines. iLamps: geometrically aware and self-configuring projectors. *ACM Transactions on Graphics*, 22(3):809–818, 2003.

24. Ramesh Raskar, Jeroen van Baar, and Jin Xiang Chai. A Low-Cost Projector Mosaic with Fast Registration. In *Proc. ACCV 2002*, pages 114–119, 2002.

25. George Robertson, Eric Horvitz, Mary Czerwinski, Patrick Baudisch, Dugald Ralph Hutchings, Brian Meyers, Daniel Robbins, and Greg Smith. Scalable fabric: flexible task management. In *Proc. AVI 2004*, pages 85–89. ACM, 2004.

26. Stacey D. Scott, M. Sheelagh T. Carpendale, and Stefan Habelski. Storage bins: Mobile storage for collaborative tabletop displays. *IEEE Comput. Graph. Appl.*, 25:58–65, July 2005.

27. Daniel Vogel and Ravin Balakrishnan. Occlusion-aware interfaces. In *Proc. CHI 2010*, pages 263–272. ACM, 2010.

28. Manuela Waldner, Christian Pirchheim, and Dieter Schmalstieg. Multi projector displays using a 3d compositing window manager. In *Proc. IPT/EDT 2008*, pages 1–4. ACM, 2008.

29. Manuela Waldner, Markus Steinberger, Raphael Grasset, and Dieter Schmalstieg. Importance-driven compositing window management. In *Proc. CHI 2011*, pages 959–968. ACM, 2011.

30. Ruigang Yang, David Gotz, Justin Hensley, Herman Towles, and Michael S. Brown. PixelFlex: A Reconfigurable Multi-Projector Display System. In *Proc. VIS 2001*, pages 167–174. IEEE Computer Society, 2001.