SpiderEyes: Designing Attention- and Proximity-Aware Collaborative Interfaces for Wall-Sized Displays

Jakub Dostal, Uta Hinrichs, Per Ola Kristensson and Aaron Quigley
University of St Andrews
St Andrews, Fife, United Kingdom
{jd67, uh3, pok, aquigley}@st-andrews.ac.uk

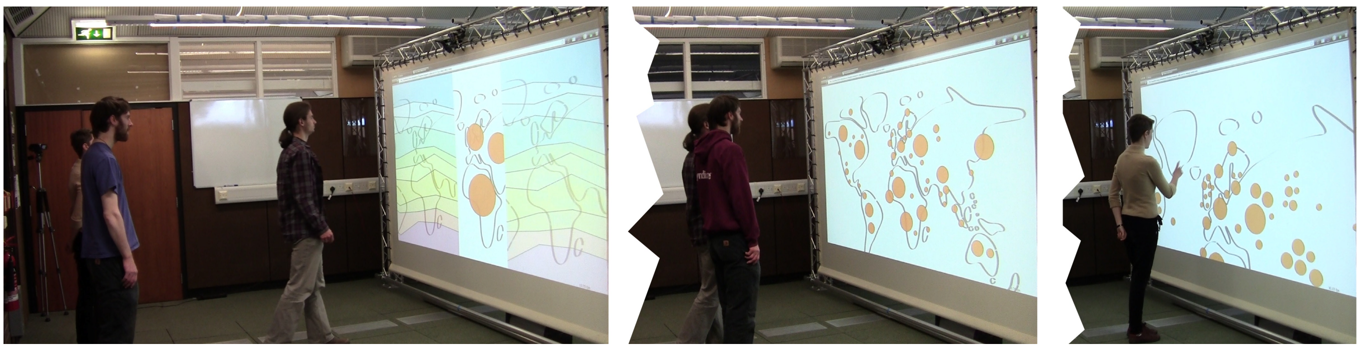


Figure 1. An illustration of our attention- and proximity-aware collaborative visualisation interface in-use. The first image shows three people using the system in parallel. The second image shows two users forming a group. The third image shows an exploration by a single user.

ABSTRACT
With the proliferation of large multi-faceted datasets, a critical question is how to design collaborative environments, in which this data can be analysed in an efficient and insightful manner. Exploiting people's movements and distance to the data display and to collaborators, proxemic interactions can potentially support such scenarios in a fluid and seamless way, supporting both tightly coupled collaboration as well as parallel explorations. In this paper we introduce the concept of collaborative proxemics: enabling groups of people to collaboratively use attention- and proximity-aware applications. To help designers create such applications we have developed SpiderEyes: a system and toolkit for designing attention- and proximity-aware collaborative interfaces for wall-sized displays. SpiderEyes is based on low-cost technology and allows accurate markerless attention-aware tracking of multiple people interacting in front of a display in real-time. We discuss how this toolkit can be applied to design attention- and proximity-aware collaborative scenarios around large wall-sized displays, and how the information visualisation pipeline can be extended to incorporate proxemic interactions.

IUI'14, February 24–27, 2014, Haifa, Israel.

ACM 978-1-4503-2184-6/14/02..$15.00.
http://dx.doi.org/10.1145/2557500.2557541

Author Keywords
Collaborative proxemics; attention-aware user interfaces

ACM Classification Keywords
H.5.3. Information Interfaces and Presentation: Group and Organization Interfaces – Collaborative computing

INTRODUCTION
With the advent of affordable, accurate, and efficient depth sensors and computer vision systems, proximity-aware interfaces that can track a user's position relative to a display become feasible (e.g. [3, 11]). In the same vein, proxemic toolkits and techniques leveraging the proxemic relations between people and objects (digital and physical) have emerged in the literature (e.g. [6, 7, 17]). However, using proxemics to support collaborative scenarios around large vertical displays is still an unexplored area. Further, it is non-trivial to implement attention- and proximity-aware interfaces for collaboration.

In this paper we introduce the concept of *collaborative proxemics*: enabling groups of people to collaboratively use attention- and proximity-aware applications. We present a toolkit that enables markerless attention-aware tracking of multiple users by combining data from a Kinect depth sensor and an off-the-shelf RGB camera. Our toolkit tracks up to four people in real time. People's body positions and eye-pair locations are estimated with an error less than 10 cm and within a range between 0.5–5 meters from the display. Our toolkit allows developers to easily create both native and web-based applications leveraging multi-user proxemic interactions. As part of the the toolkit, we also present a web-based tool for implementing proximity- and attention-aware visualisation applications. We describe how these tools can be used to create novel visualisation applications.

We then present an analysis of how our toolkit can be used to support collaborative scenarios. We introduce visualisation-based scenarios and explore how proxemic interactions can be leveraged to support parallel and collaborative exploration of large multi-faceted datasets on a wall-sized display. We describe how proxemic interactions can be used to navigate and combine different visualisation layers with varying levels of detail and context in co-located collaborative scenarios.

RELATED WORK

Previous work has discussed different dimensions that proxemics introduce to the interaction between people and interactive objects. This work has included both absolute and relative *positions* of people and objects, which implies *distance*, *orientation*, *movement*, and *identity* [3]. In this paper, we focus on how the relative position to the display and to other people in front of the display, as well as knowledge about people's orientation to the display, can be used to support collaborative scenarios (see Figure 2).

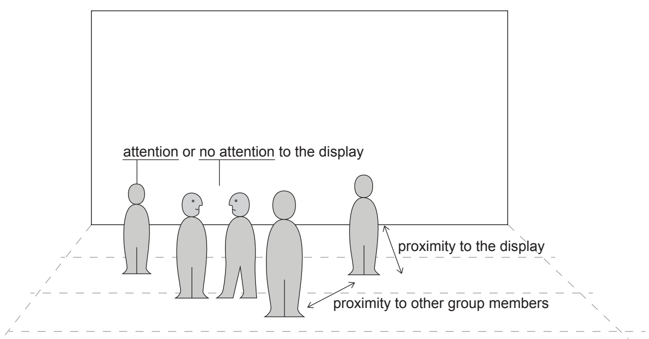


Figure 2. Using proxemics to support collaborative scenarios.

A direct parameter that can be used to drive proxemic interactions with a wall-sized display is the distance of people to the display. Previous work on proxemic interactions has described a variety of techniques and example scenarios of how the absolute and relative position of people to a stationary display can drive interactions [2, 3, 12, 14, 16, 18, 22].

Prior systems have adjusted the size or magnification factor of information as people move closer to or away from the display [5, 12, 14, 15, 19]. The distance to the display has also been used as a parameter to adjust the amount of information displayed, or to help users navigate visual information [2, 3, 22]. Finally, the type of information on the display can be adjusted based on people's distance to the display, an approach that some previous research has explored [14, 18, 22].

In the context of multi-user interactive whiteboards, the technique "Field of View" developed by Seifried et al. [20] takes people's distance to the display into account to determine the horizontal impact area of undo/redo actions: being closer to the display results in a smaller field of view and, therefore, a narrower impact area.

In our work, we expand our understanding of proxemic interactions by demonstrating how the distance to the display can be used to support *collaborative* information analysis. We call this *collaborative proxemics*.

THE SPIDEREYES TOOLKIT

We contribute a novel, attention- and proximity-aware multi-user tracking toolkit. The features of the toolkit include:

Multi-User Tracking: The toolkit tracks up to four users in real-time.

Separates Foreground and Background Activity: The toolkit uses computer vision to track users' eyes and uses this information to separate users actively engaging with the system from users in the background attending to other activities or just passing by.

Markerless Tracking: The system does not use any markers to track users and does not require any calibration for users to be tracked.

Easy Setup: The toolkit only relies on a single depth camera (e.g. a Kinect) and a high-resolution RGB camera, making it easy to set up and deploy in a variety of environments.

Programming Language Independent: The tracking system communicates its results in a programming language independent format, which allows designers to use a programming language of their choice.

Distributed Deployment: The tracking system can be deployed on a different computer than the application that uses it. This allows developers more flexibility and independence from specific deployment environments.

Visualisation Design Tool: The toolkit contains a web-based tool for designing visualisation sets. It allows designers to make their existing visualisations attention- and proximity-aware.

The toolkit is realised via two components. The first component is a multi-user tracking system. Developers can use this component to obtain real-time information about multiple users' positions and attention-aware statuses in relation to a large wall-sized display. The second component is a design tool that enables toolkit users to design attention-aware visualisation applications.

Toolkit Component 1: Tracking System

It is difficult to develop attention- and proximity-aware markerless multi-user tracking systems. To enable designers to easily create new attention- and proximity-aware interfaces we have therefore created a flexible toolkit that provides easy-to-use programming abstractions. While the bulk of our system is written in C++, our toolkit runs either a TCP/IP or a WebSockets server, which enables both native applications and browser-based applications to use the data. The scenarios we have described in this paper have all been implemented using this toolkit.

Currently, the tracking system is configured by defining several simple parameters about the environment in a text file.

```
{
"timestamp": 0003030300,
"users": [
  {"userid": 1,
   "position": {"x": 1234, "y": 1750, "z": 1063},
   "orientation": {"x": 0, "y": 0, "z": -1},
   "confidence": 1},
  {"userid": 1,
   "position": {"x": -570, "y": 1640, "z": 1534},
   "orientation": {"x": 0, "y": 0, "z": 0},
   "confidence": 0}
]
}
```


Figure 3. Example JSON object sent by the tracking system. The timestamp is in milliseconds since the system has been initialised; position is in millimeters; orientation is reported as a unit vector. Both position and orientation are in sensor coordinates. Confidence is in the interval 0-1 (values reported by the system: 1 = full detection: eye-pair and individual eyes are detected; 0.8 = partial detection: only the eye-pair is detected; 0 = no detection).

Designers can choose from several tracking algorithms (Computer Vision (CV) only, multi-user Kinect-CV fusion, or single-user Kinect-CV fusion) as well as configuring the data output (TCP/IP server, WebSockets server, or local logging). Once the system is running, the server sends the tracking results to all connected clients as a valid JSON object. See Figure 3 for an example of a data point.

Toolkit Component 2: Visualisation Design Tool

The visualisation design tool is web-based and written in JavaScript. The use of the design tool requires setting the values of several parameters and the implementation of a single function. The following parameters can be set (Figure 4 provides an example implementation for the *Vis-Active* Display with *Constant Zoom*):

Viewport Sizing (*entity_size_type*): *fraction* This parameter gives each active user an equal fraction of the display space. *angle* will dynamically resize the viewports so that they occupy equal visual angles for all active users.

Layer Magnification (*layer_zoom_type*): This parameter defines which of the magnification methods described in the scenarios section should be used for the visualisation layers. Possible values are: *physical, constant, amplified*. For *constant*, an additional parameter that defines the desired visual angle must be set (*layer_angle*). For *amplified*, the amplification ratio (*zoom_amplification*) and neutral point (*amplified_midpoint_distance*) need to be set.

Default Visualisations (*generateVisualisationSet(uid)*): This function allows the designer to define the visualisations and their distance boundaries, for each user. Each detail layer is defined by the *url* to its content (which can be an image or a URL to a webpage) and the *start* and *end* distance boundaries for its visibility.

Grouping Distance (*group_distance*): This parameter defines the maximum distance between a pair of users for them to be considered a group by the system.

Our visualisation design tool automatically manages the creation of the application itself. In addition, it also provides the

```
var entity_size_type = "angle";
var layer_zoom_type = "constant";
var layer_angle = 20.0; //in degrees
var amplified_midpoint_distance = 1500; //mm
var zoom_amplification = 0.5; //1.0 = neutral
var group_distance = 400; //mm

function generateVisualisationSet(uid) {
  var words = new detailLayer("w.svg", 0, 5000);
  var heat = new detailLayer("h.svg", 0, 5000);
  var clusters = new detailLayer("c.svg", 0,
      5000);
  var default_visLayer = new visualisationLayer();
  switch(uid) {
  case 1: {
    default_visLayer.addLayer(words);
    break; }
  case 2: {
    default_visLayer.addLayer(heat);
    break; }
  case 3: {
    default_visLayer.addLayer(clusters);
    break; }
  case 4: {
    default_visLayer.addLayer(heat);
    break; }
  }
  var result = new visualisationSet();
  result.addLayer(default_visLayer);
  return result;
}
```

Figure 4. Code listing for the JavaScript API used for implementing the scenarios. In this code sample, the *generateVisualisationSet()* function implements the passive scenario with constant zoom and a different visualisation for each user.

following functions. The tool automatically distinguishes active users and people passing by in the background and foreground based on whether their visual attention is on the display or not and only displays visualisations for active users.

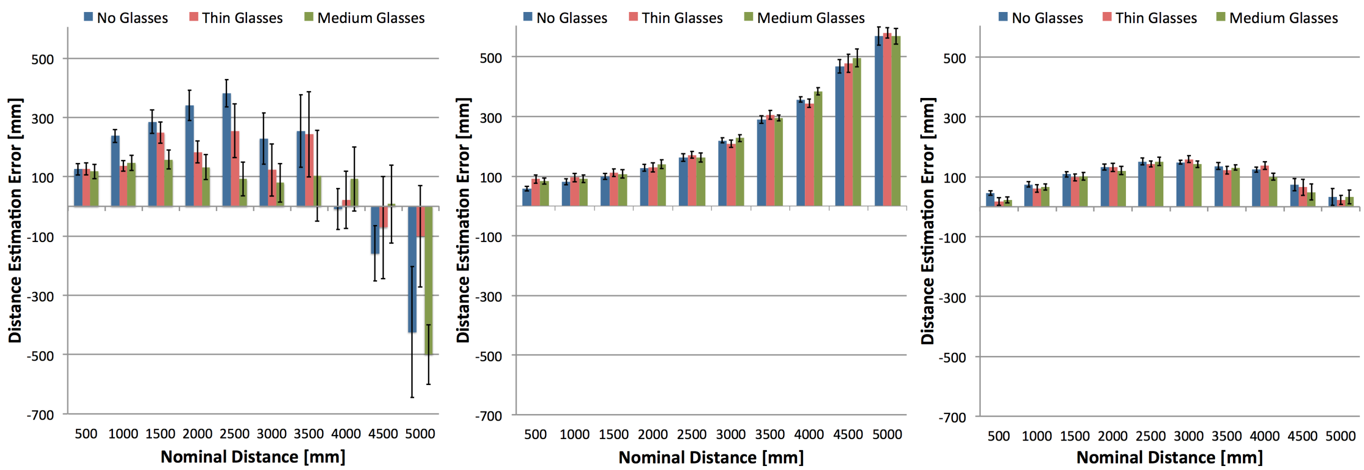
TRACKING ALGORITHMS

In this section we describe our system algorithms that underpins our toolkit. We use depth sensing (in our case a Kinect) *in combination* with computer vision algorithms to detect the users' eyes in regular RGB camera streams. The tracking system consists of four separate parts: user identification, head position tracking, attention detection, distance estimation and distance estimate correction. Some of parts of the tracking system are based on preliminary earlier work by Dostal el al. [8], which describes an approach to fusing data from a depth camera and an RGB camera for distance estimation.

In our implementation, we use the OpenCV and OpenNI frameworks coupled with our custom code. OpenCV offers implementations of standard computer vision algorithms as well as access to camera hardware. OpenNI allows us to work with the Kinect depth sensor and its data.

User Identification

To track multiple users, it is essential to have a robust user identification system. We use the Kinect's depth-based blob segmentation accessible from OpenNI as it has proved more reliable and less resource intensive compared to a computer



(a) Computer Vision Only (b) Kinect-CV Fusion (c) Kinect-CV Fusion–Corrected

Figure 5. Mean distance estimation error using computer vision-only, using computer vision-guided fusion with a Kinect depth sensor, and using computer vision-guided fusion with a Kinect depth sensor and applying a pre-computed linear regression correction model. Error bars show standard error. The three conditions in the experiment were participants wearing no glasses, glasses with a thin rim, and glasses with a rim of medium thickness.

vision approach. Due to the use of depth data, this approach is relatively robust to body occlusion and fast movement. However, this approach may lead to misidentification of users when they leave the field of view and later rejoin at a different distance. We have tested user identification with up to four users.

Head Position Tracking

After a user has been identified, we establish the position of the head within the depth/RGB images. This is a necessary step that allows our system to perform multi-user tracking in real time. This is because it enables us to significantly reduce the search area within the images. The tracking is accomplished with a cascade of three head position predictors. The primary predictor uses past data from our computer vision-based attention detector (described in the next section). If the position of the user's eye-pair is known, we use it as the centre of the search area. If the eye-pair data is not available, the secondary predictor is based on the skeleton data from the depth camera. We use the head joint as the centre of the search area. If the skeleton data is not available, then the tertiary predictor attempts to predict where the head is from the depth blob used to identify the user.

Attention Detection

Once the search area for the likely position of the user's head has been established, we translate the rectangular search area into the coordinate space of the RGB camera and perform a search for the user's eyes. We use the OpenCV implementation of the Viola-Jones feature tracking algorithm [21] to identify the users' eyes. Our algorithm is an extension of previous work [6, 8, 9], which uses eye-pair and single-eye classifiers and a custom tracking algorithm to provide coarse-grained gaze- and user-tracking.

In the first stage of the search a classifier attempts to locate the user's eye-pair. If successful, the second stage classifiers attempt to confirm the result by locating the left and right eye separately in the left and right halves of the eye-pair area. The confidence of the attention detector depends on which of the search stages were successful. The system will report either a full detection (both first and second stage detections were successful), a partial detection (only the eye-pair was detected), or no detection. The detector will also report detailed information about the detected eyes. This is used for distance estimation. It can also be used for head position tracking in the future.

Distance Estimation

Distance estimation is performed by a cascade of estimators that use available sensor data. All of the estimators estimate the distance from points within the depth data, the only difference is the method of choosing the sampling points. The primary estimator uses the points between the eyes translated into the coordinate space of the depth camera, if the eye-pair data from the attention detector is available. The secondary estimator uses points between the head and neck joint of the skeleton data if it is available. The tertiary estimator uses the mean distance of the top 25% of the user's blob if only the depth-segmented user blob is available.

Distance Estimation Correction Model

In the evaluation of our system (described later in this section), we found that the Kinect depth camera systematically over-estimates actual distances exponentially as a function of nominal distance (see Figure 5(b)). We therefore use a pre-computed correction model that adjusts the overestimation error using a linear regression model. The linear regression correction model is[1]: $y = 0.9005x$. Using experimental data we found that this correction model explains 99% of the variance of the overestimation error ($R^2 = 0.99$). The final result is that when users are between 0.5 and 5 metres away from the

[1]The model for figure 5(c) also includes an offset of 48.411 mm to account for the distance between user's feet and their eyes.

display, our system can reliably estimate their distance from the display with an error of less than 10 cm (see Figure 5(c)).

Tracking Latency

Using a 2.8GHz Quad-Core Intel Core i5 processor we can track four users with a latency of approximately 30–40 ms at 20–30 fps, which results in a system that is both scalable and fast. To speed up the tracking of multiple users to this level we have parallelised the tracking procedure. As we mentioned before, we use the OpenCV implementation of the Viola-Jones feature tracker and OpenNI to access the Kinect data. Unfortunately, OpenCV and OpenNI are difficult to multithread due to critical data structures being exposed in shared memory without appropriate locking mechanisms.

We work around this by using a series of locks around OpenCV and OpenNI's core data structures and by spinning off a separate worker thread for each user we are tracking. This enables multiple users to be tracked at approximately the same speed as a single user if there are enough available cores on the machine performing the tracking.

Advantages

It is possible to perform fast tracking of multiple users using just the Kinect data or by estimating a distance directly from a blob obtained from the depth data in the usermap (although this is non-trivial and body occlusion is a serious problem). However, our Computer Vision-Kinect fusion procedure provides three distinct advantages to designers of attention- and proximity-aware interfaces.

First, we can obtain a more specific and accurate distance estimation compared to what is possible using just the Kinect skeleton interface. The range obtainable using our system is between 50 cm and 5 metres compared to the Kinect skeleton's range between 80 cm and 4 metres. For our system, 5 metres is the maximum range we tested; the actual maximum range is likely even greater. The limitation is the availability of user blobs from the OpenNI user tracker (which starts to degrade at around 4.5 m) rather than the distance estimation procedure. The spatial resolution of the Kinect depth data at 8 metres is still < 20 cm [1]. The other limitation is the image resolution of the RGB camera. The maximum distance at which an eye-pair can be detected depends on the amount of pixels occupied by the eye-pair of the tracked person in the image. For a person with 60 mm pupil distance, using a 5 megapixel (2592×1944 pixels) image taken with a camera with a $62°$ horizontal field of view, the maximum theoretical distance at which the person can be detected is approximately 684 cm.

Second, our system is attention-aware, which means the system can tell whether a user is looking at the screen or not. This information is not possible to obtain from the Kinect skeleton data as it only provides a single point for the head joint. Our approach makes it possible to design a wide range of attention-aware interfaces. For example, it is possible to enable the interface to visualise display changes when the user is reengaging with the display (e.g. [8]).

Third, as the system is attention-aware it can distinguish between people actively viewing the system and people that are casually passing by or are standing in the background, engaged in other activities. This makes the system more practical in open office and large laboratory environments. In general, we believe systems that are able to separate "attentive signals" from background signals are crucial for real-world adoption of markerless proximity-aware interfaces.

Evaluation

To evaluate the potential of fusing computer vision and depth sensing we conducted an experiment. We recruited eight participants (three females and five males; their ages ranged from 21 to 39) from our university campus. The experiment followed a within-subjects design with two factors: Glasses (participants wearing no glasses, participants wearing glasses with a thin frame, and participants wearing glasses with a thick frame) and Sensor (Computer Vision Only, Kinect-CV Fusion, and Kinect-CV Fusion Corrected). The Computer Vision Only condition used the distance of the participant's pupils that was available from the attention detector to estimate distance using a 5 megapixel (2592×1944 pixels) image taken with a Logitech C910 RGB camera. The Kinect-CV Fusion condition used the fusion algorithm we have previously described, without the pre-computed correction model. The Kinect-CV Fusion Corrected condition used the correction model.

We positioned the RGB camera on top of the Kinect sensor. We also marked the floor at 50 cm intervals at a range from 50 cm to 5 metres. Each participant was asked to stand with their feet aligned to each of the distance markers, while the study administrator manually read the distance value from each of the sensors. We repeated the process for each participant three times. Each time the participant either wore glasses with thin or thick frames, or no glasses at all.

Figures 5(a), 5(b) and 5(c) show the distance estimation error for Computer Vision Only, Kinect-CV Fusion, and Kinect-CV Fusion Corrected respectively. In each case, the perfect performance would be represented by a constant error of approximately 5 cm (due to the difference in the position between the tips of the feet of the participants and their eyes). As is evident in the figure, the final system that uses the linear regression correction model resulted in an estimation error less than 10 cm for a range between 0.5 and 5 metres. The evaluation also showed that the system can accurately detect the user even if the user wears glasses.

USING THE TOOLKIT

We now illustrate example scenarios of how proxemic interactions can be leveraged for individual and collaborative activities around large wall-sized displays. Previously, Jakobsen et al. [14, 15] introduced proxemic interactions with information visualisations but concentrated on single-user scenarios. While we believe that proxemic interactions can potentially be applied to a number of different collaborative scenarios, we focus on a collaborative setting where a small group of information workers explore a multi-faceted dataset

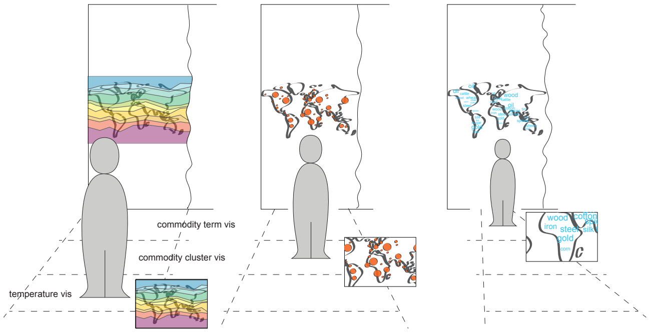


Figure 6. ***Vis-Active*** **Display: proximity to the display determines the type of visualisation.**

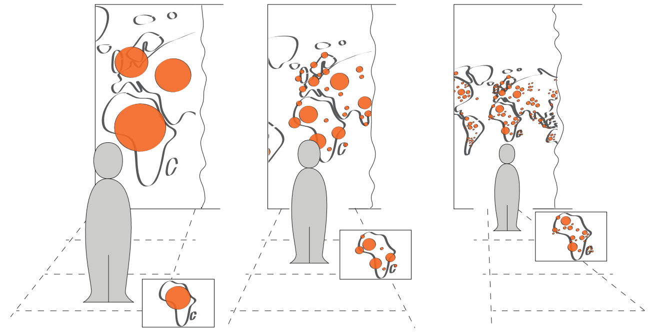


Figure 7. ***Detail-Active*** **Display with** ***Constant Zoom*****: proximity to the display determines the level of data granularity.**

from different perspectives, using a variety of visual representations.

General Considerations

We considered mapping the distance between people and the display to three different parameters: visualisation type, detail level and zoom level.

Mapping Proxemics to Visualisation Type (Vis-Active)

Depending on the distance to the display the visualisation type can be adjusted. For example, people far away from the display can see the temperature layer. However, as they get closer to the display, the temperature layer can be replaced by a commodity cluster visualisation. Directly in front of the display people can see a commodity word cloud (see Figure 6).

Mapping Proxemics to Detail Level (Detail-Active)

Similarly to the adjustment of visualisation depending on the distance to the display, the amount of detail shown within the same visualisation can be adjusted. See Figure 7 for an example.

Mapping Proxemics to Zoom Level

Depending on the distance to the display the zoom-level of the visualisation can be adjusted. We distinguish three different variations:

Physical Zoom. The visualisation layer does not actively react to people's movements in front of the display but retains a constant width and height at all times. However, people's proximity to the display naturally increases or decreases the (perceived) size of information represented in the visualisation layer (see Figure 8).

Constant Zoom. The viewing angle of the visualisation layer is kept constant no matter how close people are to the display. That is, the size of the visualisation layer (width and height) is actively changed as people move back and forth in front of the display in such a way that the perceived size of the represented information remains constant at all times (see Figure 7 and Figure 9).

Amplified Zoom. The visualisation layer is scaled up or down depending on people's proximity to the display. As people move closer, the visualisation layer enlarges, providing a magnified view on the information represented (see Figure 10).

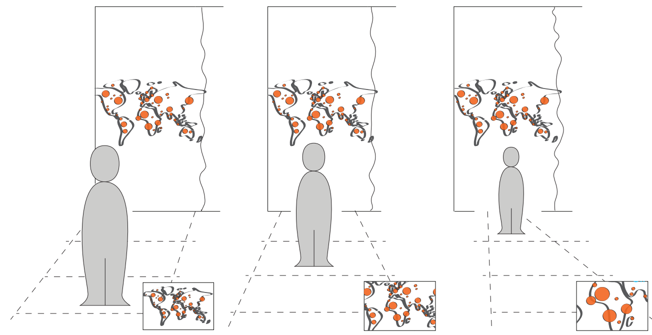


Figure 8. ***Physical Zoom*****: The visualisation layer remains the same size.**

These three different proxemic-based parameter mappings can be combined in different ways, resulting in nine different scenarios: *Vis-Active* with *Physical*, *Constant* or *Amplified Zoom*, *Detail-Active* with *Physical*, *Constant* or *Amplified Zoom* and complex *Vis-and-Detail-Active* scenarios with *Physical*, *Constant* or *Amplified Zoom*. We describe four of the nine scenarios in detail to point out the potential advantages and disadvantages of the parameter combinations.

Scenario 1: Vis-Active with Physical Zoom

On the *Vis-Active* Display (see Figure 6), the visualisation layer changes based on an individual's distance to the display while the size of the visualisation remains constant. The mapping between proximity and visualisation type can be continuous with the visual layers blending into each other as people move towards and away from the display.

In a multi-user scenario (see Figure 11) the visualisations change as group members move toward and away from the display. One of the possible advantages of this setup is that it supports independent explorations by users: each user can easily shift between visualisation views, and the continuous blending of visualisations even allows each user to explore correlations within the different data sets and perspectives. A possible disadvantage is that in collaborative situations, users cannot easily blend different types of visualisations while standing next to each other because they would need to be at different vertical distances from the display. According to

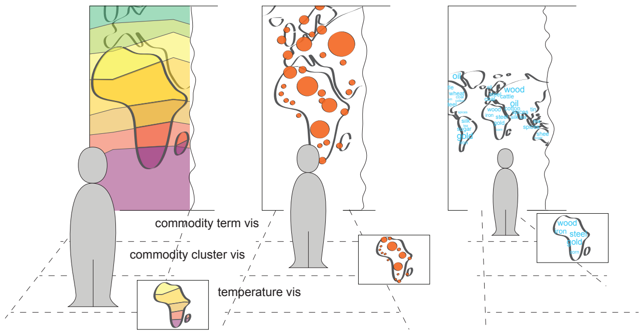


Figure 9. ***Vis-Active*** **Display with** ***Constant Zoom*****: the proximity to the display determines the type of visualisation while people's viewing angle remains stable.**

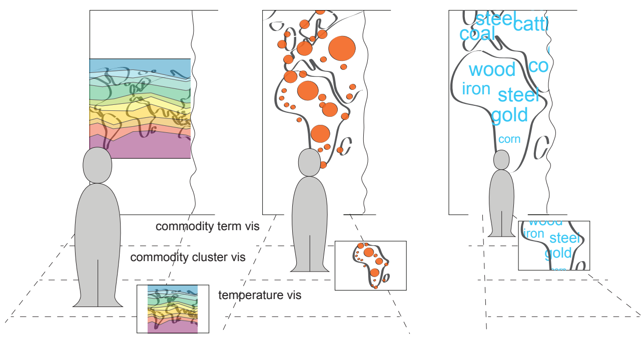


Figure 10. ***Vis-Active*** **Display with** ***Amplified Zoom*****: the proximity to the display determines (1) the type of visualization and (2) the amount of context/magnification level of information.**

findings by Hawkey et al. [13], forcing group members to position themselves at different vertical positions in front of the display may hamper communication and coordination, which are important factors in more closely coupled collaborative work phases.

Scenario 2: Vis-Active with Constant Zoom

In Scenario 2 we adjust the type of visualisation layer as people move back and forth in front of the display while adjusting its size to keep people's viewing angle constant.

As shown in Figure 9, more context information can be added to the display as a person moves closer—because the viewing angle remains constant as a person moves toward the display, the (physical) size of information in focus does not change. This can facilitate direct interaction with information when close to the display because information is visible in a constrained space; people do not have to reach far or crouch to manipulate particular data of interest, something that has been reported as problematic on large wall-sized displays that show map representations [13].

In a multi-user setting of the *Vis-Active* with *Constant Zoom* scenario (see Figure 12), users standing at different distances from the display will see different types of visualisation layers. At the same time, their viewing angle remains stable as they move towards or away from the display. Similarly to Scenario 1, in a collaborative scenario, individual users can

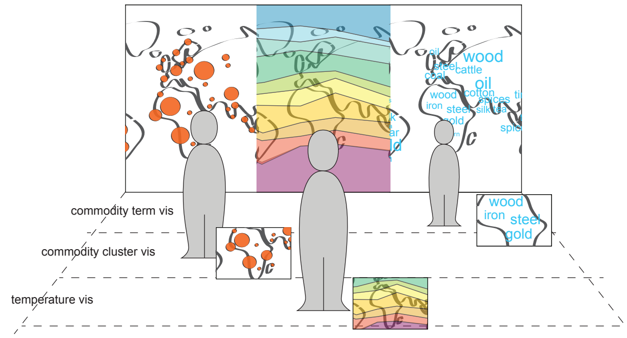


Figure 11. ***Vis-Active*** **Display with** ***Physical Zoom*****: multi-user scenario.**

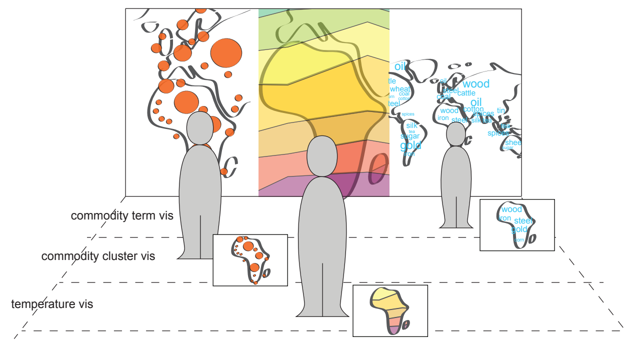


Figure 12. ***Vis-Active*** **Display with** ***Constant Zoom*****: multi-user scenario.**

work on different data perspectives and explore different visualisations at the same time, while each group member can observe the visualisations that their collaborators are working on in their periphery. This may inspire further explorations of their own visualisation. However, the constant viewing angle has the limitation that comparisons of size between data items in the different visualisations are difficult if group members stand in different zones since the viewing angle remains constant with changing distances to the display. However, as group members start to collaborate in a more closely coupled way on two different visualisations, for instance to actively compare trends within different data, it is likely that they will choose to stand in horizontal proximity, according to previous studies [13].

Scenario 3: Vis-Active with Amplified Zoom

Figure 10 shows a version of the *Vis-Active* scenario that is effectively the inverse of Scenario 2: more context information is shown from afar, while content becomes magnified as the person moves closer to the display. Note that in both variations, the type of visualisation is also changed according to the distance to the display, as described in Scenario 2. Also, while the magnification behaviour is inverted, the advantages and disadvantages when used by multiple users are likely to be similar as with *Constant Zoom* in Scenario 2.

Scenario 4: Detail-Active Display

In Scenario 4, the *Detail-Active* Display (see Figure 7), the level of data detail within the same visualisation is changed

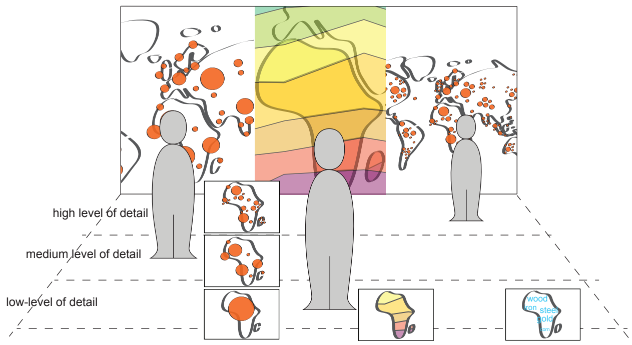


Figure 13. ***Detail-Active*** **Display with** ***Constant Zoom*****: multi-user scenario.**

as people move toward and away from the display. We assume that showing more detail of the data can be helpful when people are close to the display for perceptual and interaction reasons. Standing close to the display makes it possible to *perceive* more subtle nuances and distinguishable features within the data that may not even be visible from further away, even if they would be represented. Furthermore, people may want to engage in more elaborate active explorations (e.g. via direct-touch), in which case it makes sense to show more data and, therefore, provide a more fine-grained visualisation of the dataset. We only depict one variation of the *Detail-Active* Display, which uses *Constant Zoom* and directly corresponds to the concept shown in Figure 13. Further variations using *Physical Zoom* and *Amplified Zoom* are also possible and are analogous to Scenarios 2 and 3, respectively.

In the multi-user *Detail-Active* Display with *Constant Zoom* scenario, group members could, again, focus on different (previously chosen) visualisations, which will remain the same as they move back and forth in front of the display (see Figure 13). The level of detail for each individual group member and the viewport on the visualisation change as a group member's distance to the display is altered. In a collaborative scenario, providing different levels of detail along with different types of visualisations can be beneficial: similar to the other scenarios we described, group members can work in parallel to explore different perspectives of the data. In more closely-coupled collaborative phases which may be about discussing particular patterns or discoveries, it can be beneficial to have different levels of detail on the data available and blend visualisations as we described in Scenario 1.

Uniting the Design of the Scenarios

The *Vis-Active* and *Detail-Active* scenarios can be seen as two special cases of a single hierarchical structure. Each detail layer is essentially the same visualisation with a different amount of detail visible. Therefore, each visualisation layer in the *Vis-Active* scenario can be defined as a visualisation layer containing only a single detail layer. This means that we can unite the scenarios by defining the visualisation set as a set of one or more visualisation layers, each of which contains one or more detail layers. Figure 14 is an example design of a complex *Vis-Detail-Active* visualisation set.

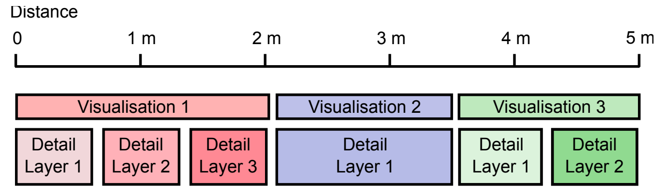


Figure 14. An illustration of a complex visualisation set consisting of three visualisations with each visualisation using a different amount of detail layers.

CASE STUDY

Using the D3 JavaScript library we modified a well known information visualisation to realise a proximity-aware visualisation using the SpiderEyes toolkit. The purpose was to evaluate the difficulty of interfacing our toolkit with an established information visualisation framework. For this, we selected the "Wealth and Health of Nations" example[2], which visualises a complex, high-dimensional dataset (country, per-capita income, life expectancy, population size, and time). In our example, we mapped the lateral movement along the horizontal axis to the temporal dimension of the dataset. Stepping from one side to another in front of the wall-sized display changed the displayed data to a specific year. This case study visualisation can be seen in Figure 15, Figure 16, and in the supplemental video of this paper.

Many alternative designs are also possible. Here we list a few examples. The forward and backward physical movements can be mapped to a scale/zoom mechanism, such as *Constant Zoom*. While *Constant Zoom* is used, the lateral movements along the horizontal axis may also be mapped to a translation function, moving the position of the visualisation on the display so that it is always centred in front of the user.

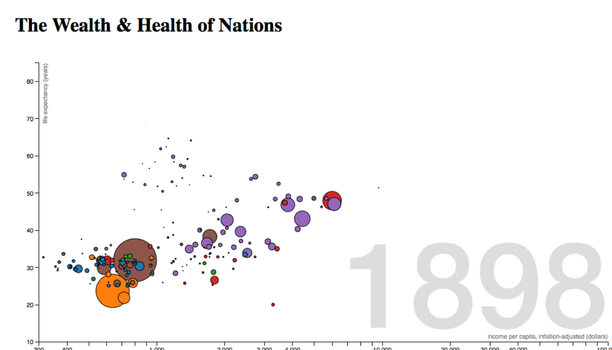


Figure 15. The example D3 visualisation used as a case study.

EXTENDING THE INFOVIS PIPELINE

Numerous authors have built on, and extended, the original Information Visualisation Pipeline as a means the decompose and better conceptualise the tasks and sub-tasks involved in representing data in a graphical form [4]. Each model emphasises different aspects of the problem, analysis, scale of data, domain, or indeed the current understanding of data use. Fry [10] extended the pipeline to emphasise seven stages, or elements of design, to be considered when moving from data to display.

[2]`http://bost.ocks.org/mike/nations/`

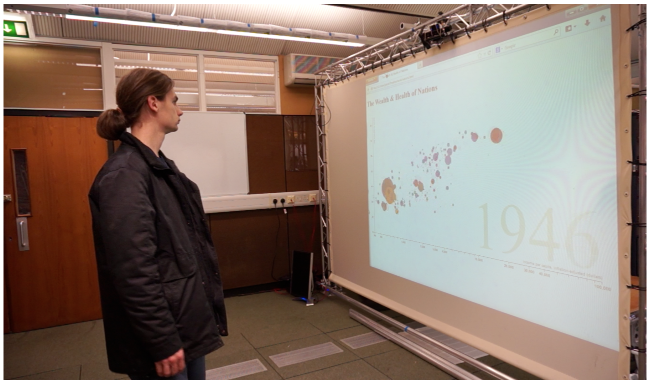


Figure 16. The example D3 visualisation explored by a user.

We have conducted an initial evaluation of this pipeline with respect to the influence of proxemics, proximity-aware interactions, and related forms of physical interaction with displayed information. The results of this preliminary exploration are shown in Figure 17. The Fry [10] information visualisation pipeline consists of seven stages. The acquisition stage is concerned with the collection or input of data from a source. In the parsing stage, data is converted into a desired internal format. The filtering stage removes data that will not be relevant for a particular visualisation. The mining stage generates higher level abstractions or compound measures from the filtered data. The representation stage is concerned with the design decisions around the section of the appropriate visual structures for presentation of data. The refinement stage optimises the visual form of the data using a set of design practices or contextual information to maximise impact of the visualisation. Finally, the interaction stage deals with any interaction mechanisms or techniques related to the visual form of the visualisation itself [10].

Our analysis of this pipeline suggests that, with the exception of the parsing stage, all other stages can be directly or indirectly affected when physical interaction systems, such as proxemic-based interactions, are introduced. Figure 17 provides a summary of the relationships between such interactions in this pipeline. Here the interaction stage is directly affected in a wide variety of ways. Direct manipulation of the viewpoint for the data and associated movement, as well as scaling and zoom, belong to this stage. The scenarios in this paper describe some of the possible proxemic mappings to these parameters. The changes to these parameters also indirectly influence the acquisition stage, as additional data may have to be provided when the change occurs. Working backwards through the pipeline, the refinement stage can be indirectly influenced when changes in position between the users and the display may render parts of the visualisation unintelligible, thereby forcing a change in refinement or representation. The influence of proxemic mappings on the representation stage is best exemplified in the *Vis-Active* scenarios, which show different types of representations as the user moves through the space. The mining stage is influenced directly through the higher number of dimensions that can be displayed compared to other non-proxemic views, as shown in our D3 visualisation example. This allows for navigation along all the standard axes, as well as time. Additionally, the *Detail-Active* scenarios show an example that directly influences both the filtering and mining stages via the need for clustering and other grouping and filtering mechanisms. The application of these mechanisms may also trigger a further data acquisition when transitioning from a low-detail to a high-detail view, thus indirectly influencing the acquisition stage.

Our points of interconnection between individual and group physical movements with respect to information visualisation are exploratory in nature and requires further study and evaluation. However, we believe the results of this exploration are valuable as a starting point for a more detailed inquiry.

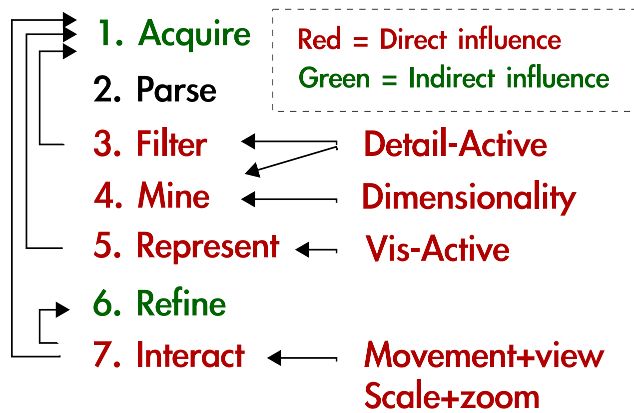


Figure 17. The information visualisation pipeline adapted to proxemic interactions.

LIMITATIONS AND FUTURE WORK

A series of user studies are necessary to fully understand the limitations and capabilities of our system and application scenarios. A particularly fruitful avenue of future research might be to investigate how groups of people negotiate the sharing of the space in collaborative attention- and proximity-aware user interfaces.

A limitation of the current work is that the application scenarios presented in the paper might underutilise the potential of attention-driven interactions. In our currently implemented application scenarios, users' visual attention is only used to filter out passers-by. This is an important function because distinguishing between users engaging with our system and users that are present in the background is a critical practical issue in a deployed system. Nevertheless, one direction of future work is to embed the visual attention detection feature in our toolkit more deeply into the application scenarios.

All application scenarios we have presented in this paper are grounded on existing evidence from the research literature. We believe these application scenarios form a sensible starting point for better understanding the design space of collaborative attention- and proximity-aware user interfaces. A more extensive classification and validated design space exploration can in the future serve as a foundation for collaborative proxemics.

CONCLUSIONS

In this paper we have introduced the concept of collaborative proxemics: enabling groups of people to collaboratively use attention- and proximity-aware applications. To help designers we have developed SpiderEyes: a toolkit that enables people to design proximity- and attention-aware co-located collaborative interfaces. We have presented an analysis of how proxemics can be used to support collaborative scenarios and we have introduced visualisation-based scenarios that explore how proxemic interactions can be used to support parallel and collaborative exploration of large multi-faceted datasets on a wall-sized display.

The SpiderEyes toolkit uses our novel tracking system that provides fast and accurate markerless attention- and proximity-aware tracking of up to four users at 30 frames per second. It also includes a visualisation design tool that allows designers to easily augment their visualisations with attention- and proximity-awareness with minimal programming efforts. We exemplified this by interfacing SpiderEyes with an example from the D3 JavaScript library. Finally, we extended the infovis pipeline so that it supports proxemics.

The SpiderEyes toolkit can be downloaded here: `http://sachi.cs.st-andrews.ac.uk/software/spidereyes`

ACKNOWLEDGEMENTS

This work was supported by the Scottish Informatics and Computer Science Alliance (SICSA).

REFERENCES

1. Andersen, M., Jensen, T., Lisouski, P., Mortensen, A., Hansen, M., Gregersen, T., and Ahrendt, P. Kinect Depth Sensor Evaluation for Computer Vision Applications. Tech. rep., Department of Engineering, Aarhus University, Denmark, 2012.
2. Ball, R., North, C., and Bowman, D. A. Move to Improve: Promoting Physical Navigation to Increase User Performance with Large Displays. In *Proc. CHI'07*, ACM (2007), 191–200.
3. Ballendat, T., Marquardt, N., and Greenberg, S. Proxemic Interaction: Designing for a Proximity and Orientation-Aware Environment. In *Proc. ITS'10*, ACM (2010), 121–130.
4. Card, S. K., Mackinlay, J. D., and Shneiderman, B., Eds. *Readings in information visualization: using vision to think*. Morgan Kaufmann Publishers Inc., 1999.
5. Clark, A., Dünser, A., Billinghurst, M., Piumsomboon, T., and Altimira, D. Seamless Interaction in Space. In *Proc. OzCHI'11*, ACM (2011), 88–97.
6. Dostal, J., Kristensson, P. O., and Quigley, A. Estimating and using absolute and relative viewing distance in interactive systems. *Pervasive and Mobile Computing* (July 2012).
7. Dostal, J., Kristensson, P. O., and Quigley, A. Multi-view proxemics: distance and position sensitive interaction. In *Proc. PerDis'13*, ACM (2013), 1–6.
8. Dostal, J., Kristensson, P. O., and Quigley, A. Subtle Gaze-Dependent Techniques for Visualising Display Changes in Multi-Display Environments. In *Proc. IUI'13*, ACM (2013), 137–147.
9. Dostal, J., Kristensson, P. O., and Quigley, A. The Potential of Fusing Computer Vision and Depth Sensing for Accurate Distance Estimation. In *CHI'13 Extended Abstracts*, ACM (2013), 1257–1262.
10. Fry, B. *Visualizing Data*, 1st ed. O'Reilly Media, 2007.
11. Greenberg, S., Marquardt, N., Ballendat, T., Diaz-Marino, R., and Wang, M. Proxemic Interactions: The New Ubicomp? *ACM Interactions 18*, 1 (2010), 42–50.
12. Harrison, C., and Dey, A. K. Lean and Zoom: Proximity-Aware User Interface and Content Magnification. In *Proc. CHI'08*, ACM (2008), 8–11.
13. Hawkey, K., Kellar, M., Reilly, D., Whalen, T., and Inkpen, K. M. The Proximity Factor: Impact of Distance on Co-Located Collaboration. In *Proc. Group'05*, ACM (2005), 31–40.
14. Jakobsen, M. R., and Hornbæk, K. Proxemics for Information Visualization on Wall-Sized Displays. In *Proxemics'12: Workshop in Conjunction with NordiCHI'12* (2012).
15. Jakobsen, M. R., Sahlemariam, H. Y., Knudsen, S., and Hornbæk, K. Information Visualization and Proxemics: Design Opportunities and Empirical Findings. *IEEE Transactions on Visualization and Computer Graphics 19* (2013), 2386–95.
16. Ju, W., Lee, B. A., and Klemmer, S. R. Range: Exploring Implicit Interaction through Electronic Whiteboard Design. In *Proc. CSCW'08*, ACM (2008), 17–26.
17. Marquardt, N., Ballendat, T., Boring, S., Greenberg, S., and Hinckley, K. Gradual engagement: facilitating information exchange between digital devices as a function of proximity. In *Proc. ITS'12*, ACM (2012), 31–40.
18. Prante, T., Röcker, C., Streitz, N. A., Stenzel, R., Magerkurth, C., van Alphen, D., and Plewe, D. A. Hello.Wall—Beyond Ambient Displays. In *Adj. Proc. UbiComp'03* (2003), 277–278.
19. Rädle, R., Butscher, S., Huber, S., and Reiterer, H. Navigation Concepts for ZUIs Using Proxemic Interactions. In *Proxemics'12: Workshop in Conjunction with NordiCHI'12* (2012).
20. Seifried, T., Rendl, C., Haller, M., and Scott, S. D. Regional Undo/Redo Techniques for Large Interactive Surfaces. In *Proc. CHI'12*, ACM (2012), 2855–2864.
21. Viola, P., and Jones, M. J. Robust Real-Time Face Detection. *International Journal of Computer Vision 57*, 2 (2004), 137–154.
22. Vogel, D., and Balakrishnan, R. Interactive Public Ambient Displays: Transitioning from Implicit to Explicit, Public to Personal, Interaction with Multiple Users. In *Proc. UIST'07*, ACM (2007), 137–146.