Proximity-Based Adaptation of Web Content on Public Displays

Amir E. Sarabadani Tafreshi(✉), Kim Marbach, and Moira C. Norrie

Department of Computer Science, ETH Zurich, 8092 Zurich, Switzerland
{tafreshi,norrie}@inf.ethz.ch, marbachk@student.ethz.ch

Abstract. Viewers of public displays perceive the content of a display at different sizes according to their distance from the display. While responsive design adapts web content to different viewing contexts, so far only the characteristics of the device and browser are taken into account. We show how these techniques could be extended to consider viewer proximity as part of the viewing context in the case of public displays. We propose a general model for proximity-based adaptation and present a JavaScript framework that we developed to support experimentation with variants of the model in both single and multi-viewer contexts. We also report on an initial user study based on single viewers which yielded promising results in terms of improved user perception and engagement.

Keywords: Responsive design · Proximity-based adaptation · Web development framework

1 Introduction

Responsive design is an approach to web design aimed at crafting sites to provide an optimal viewing and interaction experience. Currently, only the characteristics of devices and browsers are taken into account in adapting a website to a particular viewing context. However, it is interesting to note that earlier drafts proposing extensions to media queries in CSS4[1] did also consider the environmental factor of ambient light level. Since the distance of the viewer from the screen is highly variable in public display contexts with the effect that users perceive content in radically different sizes, we propose that responsive design techniques should be extended to take user proximity into account in such settings.

An open issue in the design of pervasive display systems for public and semi-public spaces is how to attract and retain user interest [1]. We wanted to investigate how web content could be adapted according to the proximity of the viewers and whether this would improve the user experience. In this paper, we present a model that uses the distance of the viewers to the screen in combination with the display characteristics to adapt the display content. While the basic model is based on a single viewer, public displays also differ from traditional web viewing

[1] https://www.w3.org/TR/2016/WD-mediaqueries-4-20160126/.

J. Cabot et al. (Eds.): ICWE 2017, LNCS 10360, pp. 282–301, 2017.
DOI: 10.1007/978-3-319-60131-1_16

contexts in the fact that there are typically multiple viewers. We therefore propose a number of variants of the model for multi-viewer contexts.

We also introduce a development framework that implements the model and acts as an experimental platform for proximity-based interaction. To detect the number and distance of viewers, we use the camera-based Kinect sensing technology which is readily available as a commercial product. The framework, called ResponDis, is based on JavaScript and builds on the Kinect SDK. Similar to responsive design breakpoints specified using CSS3 media queries, the framework supports a zone concept which allows designers to specify more radical changes to the content and layout as viewers move between zones.

The ResponDis framework is characterised by four main aspects: (1) useful programming abstractions for the use of our proposed approach (2) lightweight support for cross-platform, multi-device development based on native web technologies with which many developers are already familiar, (3) a flexible client-server architecture enabling a variety of multi-device ecosystems around Kinect, (4) an extensible architecture which allows multiple Kinects to be used and connected to different distributed servers in order to track more people in different locations and at different viewing angles.

We begin with a discussion of related work in Sect. 2, before introducing our proximity-based adaptation model in Sect. 3. The features of the ResponDis framework are presented in Sect. 4 followed by a description of the architecture and implementation in Sect. 5. We then report on an initial user study carried out to evaluate our adaptation model in Sect. 6. The paper concludes with a discussion of our results in Sect. 7 followed by final remarks in Sect. 8.

2 Related Work

Responsive design is a recent trend in web development that caters for the wide diversity of devices now used to access websites [2]. There are three main parts to responsive design: a fluid rather than fixed layout that adapts to the viewport size, media queries that optimise the design for different viewing contexts, and means of selecting and sizing images according to viewing context [3]. *Fluid layout* relies on the use of relative units such as percentages and `em` instead of absolute units such as pixels and points, along with flexible grid layouts such as that supported by Flexbox[2]. *Media queries* are used to specify design breakpoints in terms of alternative CSS style rules to be applied in specific viewing contexts defined by values such as the viewport size, pixel density and device orientation [4].

In the research domain, the focus has been on support for desktop-to-mobile adaptation [2], while an increasing number of practitioners advocate a mobile-first strategy in conjunction with responsive design that could be categorised as mobile-to-desktop adaptation. However, relatively little attention has been paid in either research or practice to adaptation to very large screens, even though

[2] https://css-tricks.com/snippets/css/a-guide-to-flexbox/.

these are now common in public and semi-public spaces. An exception was work investigating adaptation to large screens, particularly in the case of text-centric websites such as online newspapers [5]. In this work, they proposed a set of design metrics, developed an adaptive layout template and carried out a user study to show the benefits of the adaptation.

While CSS3 introduced media queries to allow designs to be adapted to particular viewing contexts, they only take into account characteristics of the device and browser such as viewport size and pixel density. Earlier draft proposals for extensions to media queries in CSS4 included the property *luminosity* for the ambient light level and therefore considered contextual factors beyond the device and browser. We note however that this has been removed from the latest draft.

A factor that has not yet been taken into account in responsive design is the proximity of the user to the screen which is an issue in the case of public displays where the distance of the user to the screen can be hugely variable. In contrast, the variation in viewing distances for desktop and mobile displays is relatively small. For example, one study showed that the average smartphone viewing distance is about 12 in. [6], and it is easy for a user to adjust the distance to improve legibility if required. In the case of a public screen, it might require that they significantly adjust their position or motion path, which they would only do if they were already engaged with the display. Therefore, we advocate that, ideally, several other factors need to be taken into account when adapting web content to public displays such as the distance of the user, their visual acuity, and the resolution of content.

Screen manufacturers typically recommend either a fixed sitting distance or a range within which a viewer should be seated to have an optimal view. However, the recommendations differ depending on which manufacturer you ask and on the use of the display itself. We summarise some of the available recommendations below.

The most common fixed viewing distance recommendations are proposed by SMPTE[3] (Society of Motion Picture and Television Engineers) and the company THX[4], as well as manufacturers and retailers. A widescreen high-definition television (HDTV) normally has horizontal lines of vertical resolution (pr) and an aspect ratio of 16:9. One of the popular fixed viewing distance recommendations for a 1080*pr* resolution is 2.5 times the display's diagonal size (DS) which corresponds to a 20-degree viewing angle. However, SMTPE standards recommend 1.6264 times the DS for a 16:9 TV which is a very popular viewing distance recommendation in the home theatre enthusiast community [7]. This recommendation corresponds to a position where the display occupies a 30-degree field of view. In contrast, THX, which develops high fidelity audio/visual reproduction standards, recommends that the "best seat-to-screen distance" is one where the view angle approximates 40 degrees. To achieve this, they recommend multiplying the DS by 1.2.

[3] http://www.smpte.org.

[4] http://www.thx.xom.

In addition to the fixed viewing distance, different optimum viewing range recommendations based on the screen size also exist. For the minimum and maximum viewing distance, some manufacturers recommend a view angle of approximately 31 and 10 degrees, respectively. On the other hand, some retailers recommend the minimum viewing distance that allows a view angle of just a little over 32 degrees on average and the maximum viewing distance that provides a viewing angle of approximately 16 degrees. However, THX certified cinema screen placements offer a different range. THX still contends that minimum viewing distance is set to approximate a 40-degree view angle, and the maximum viewing distance is set to approximate 28 degrees. Their maximum horizontal view angle recommendation is based on the average human vision, and, in their opinion, a 40-degree view angle provides the most "immersive cinematic experience". Therefore, they consider their minimum viewing distance recommendation as the optimum viewing distance which provides the maximum viewing angle based on average human vision.

Regardless of inconsistencies in the recommendations, almost all of them are based on variable screen size but on a fixed display resolution with the same content resolution and what is considered to be normal vision. However, these factors certainly affect the calculation of optimum viewing distance taking into account the limitations of the human visual system [8].

Previous research [9] in information interaction at multiple distances and angles has shown that adaptive interfaces are useful for addressing the user's various attention states. Dostal et al. [10] implemented a multi-user interface in a wall-sized display that exploits people's movements and distance to the content display to enable collaborative navigation. This work also offers a toolkit called SpiderEyes for designing attention- and proximity-aware collaborative interfaces for wall-sized displays. However, the interface design in such works are still based on experiments for their specific setup. Consequently, migration of such designs to a new setup requires new design decisions. Therefore, a general model that enables responsive web design based on the limitations of the human visual system and influencing factors is a necessity.

3 The Model

As discussed in the previous section, the ideal viewing distance is based on visual acuity. The human eye with *normal* vision (20/20) can detect or resolve details down to as small as $1/60th$ of a degree of arc [11]. This distance represents the point beyond which some details in the image can no longer be resolved. Being closer to the screen than this results in the need for higher image resolution per degree of arc (or angular resolution) as well as increased pixel count of the display. This value should be lowered if visual acuity is worse than normal human vision, and raised if it is better.

An equation that considers these factors to calculate the minimum (optimum) viewing distance has been proposed [12]:

$$\mathrm{VD} = \frac{\mathrm{DS}}{\sqrt{((\frac{\mathrm{NHR}}{\mathrm{NVR}})^2 + 1)} \times \mathrm{CVR} \times \tan(\frac{\frac{1}{60}}{\mathrm{eyeAccuracy}} \times \frac{\pi}{180})} \tag{1}$$

where the meaning of the parameters is shown in Table 1.

Table 1. Explanation of the parameters of the equation.

VD	Viewing distance (in inches)
NHR	Display's native horizontal resolution (in pixels)
NVR	Display's native vertical resolution (in pixels)
DS	Display's diagonal size (in inches)
CVR	Vertical resolution of the content being displayed (in pixels)
eyeAccuracy	Visual acuity, 20/20 is considered as "normal" eye sight

Given these parameters, one can see that many factors play a role in computing the appropriate viewing distance. Not only is the distance of the user to the display an important factor, but also factors such as the size and resolution of the display, the resolution of the content being displayed and human visual acuity. Using this formula, one can see, for example, that the optimum viewing distance becomes closer to the screen, as the screen size decreases.

Since the goal of responsive web design is to provide an optimum web interface, we need to calculate the optimal content size corresponding to the viewing distance of users, rather than the optimal distance for a viewer given fixed content. Therefore, we transformed the formula given above to instead have the viewing distance (VD) as an input and the resolution of the content being displayed as an output. The resulting formula calculates the optimal content vertical resolution (CVR) as follows.

$$\mathrm{CVR} = \frac{\mathrm{DS}}{\sqrt{((\frac{\mathrm{NHR}}{\mathrm{NVR}})^2 + 1)} \times \mathrm{VD} \times \tan(\frac{\frac{1}{60}}{\mathrm{eyeAccuracy}} \times \frac{\pi}{180})} \tag{2}$$

Similar to the use of screen size in current responsive design methods, the calculated CVR could be used to define design breakpoints in the form of media queries as well as for fluid web layout.

This can also be done using the optimal content horizontal resolution (CHR) which is equivalent to CVR multiplied by the screen aspect ratio ($\frac{\mathrm{NHR}}{\mathrm{NVR}}$). Defining media queries using optimal content resolution rather than fixed user distances automatically results in the specification of distance ranges which define *zones* in front of a display.

For example, consider an application that displays a world information map on a public display. We could define three zones as shown in Fig. 1.

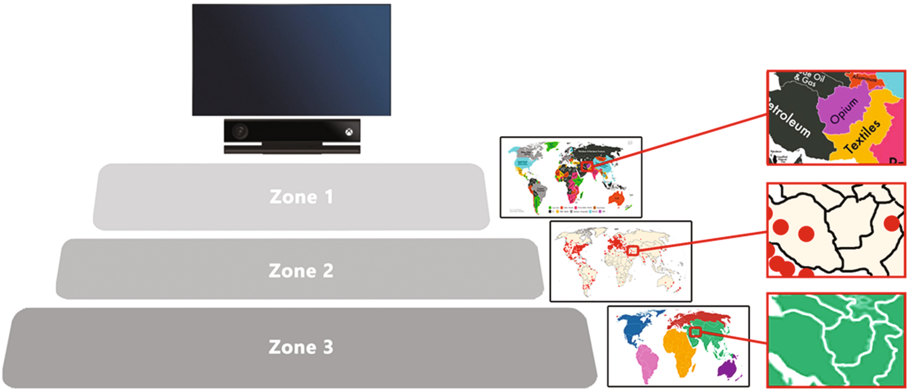


Fig. 1. Zone based UI adaptation of world information map

The furthest zone, Zone 3, could be designed to attract the attention of users by simply displaying a world map showing the continents. This could be defined as the range $0pr < \text{CHR} < 360pr$. Curious users might move closer into Zone 2 defined by the range $360pr < \text{CHR} < 720pr$. At this point, the display content would be adapted to show more detailed information such as a world population cluster visualisation. Moving even closer to the screen, they enter Zone 1 defined by the range $720pr < \text{CHR}$. Here, even more details could be provided such as a commodity word cloud which would be easily read by viewers at this distance.

Using the optimal content resolution to define the zones makes the layout adaptation independent of fixed distance ranges which also means that it is easy to cater for different setups just as responsive design caters for multiple devices. Our model can therefore support pervasive display systems (PDS) which manage content for heterogeneous display networks [1].

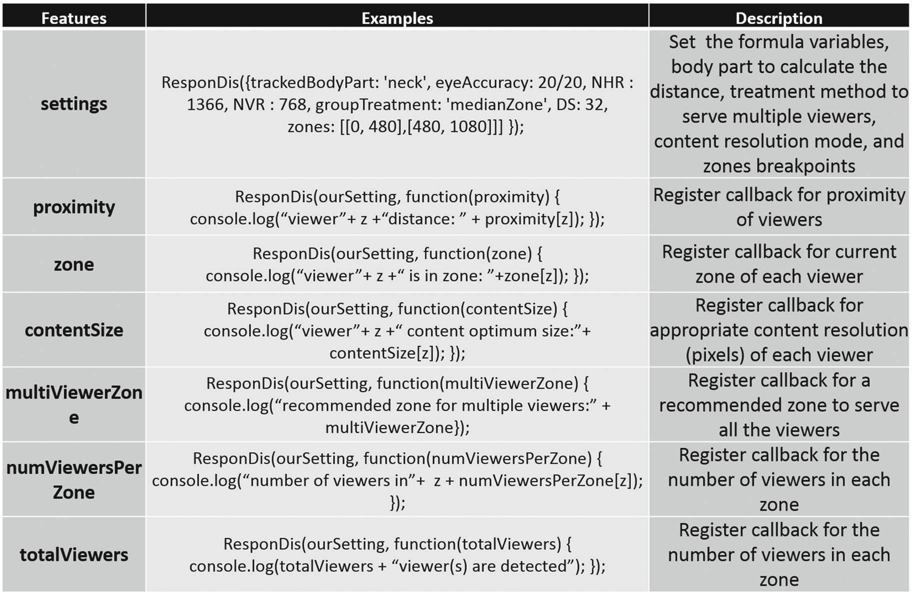
4 The ResponDis Framework

As a first solution to test the model and enable rapid prototyping, we designed and developed ResponDis–a JavaScript framework for proximity-based adaptive display user interfaces. The framework provides developers with crucial information such as the proximity of viewers, the current zones of the viewers, individually optimal content resolution (in pixels) of all tracked viewers, and the recommended activated zone for the case of multiple viewers based on the number of viewers in each zone as well as the total number of viewers.

As already mentioned in the previous section, our intention was to use the optimal content resolution as design breakpoints, similar to how viewport size is often used in CSS3 media queries to define layout breakpoints. In the case of our adaptation model, these breakpoints correspond to a zone-scheme so that content is adapted depending on the zone in which a user is currently located. As will be discussed later, in the case of multiple users located in different zones, there needs to be a strategy that determines which zone to activate and we have experimented with different strategies. We start by describing the features and operation of the framework in terms of a single user.

The framework uses our model to compute the optimum content resolution "*contentSize*" at each point in time, which, according to the developer's customised setting, can be partitioned into multiple ranges defining the set of zones. In addition to the *contentSize*, the framework provides the distance of the viewers which can be used for fluid design.

Table 2. ResponDis features with code examples. Callback functions are based on the *settings* object.



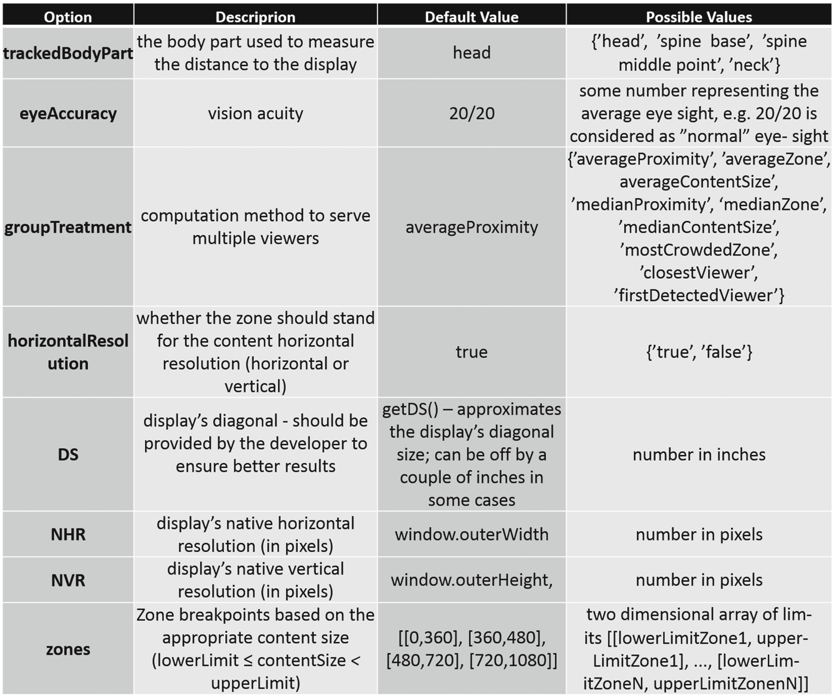
Features	Examples	Description
settings	ResponDis({trackedBodyPart: 'neck', eyeAccuracy: 20/20, NHR : 1366, NVR : 768, groupTreatment: 'medianZone', DS: 32, zones: [[0, 480],[480, 1080]]] });	Set the formula variables, body part to calculate the distance, treatment method to serve multiple viewers, content resolution mode, and zones breakpoints
proximity	ResponDis(ourSetting, function(proximity) { console.log("viewer"+ z +"distance: " + proximity[z]); });	Register callback for proximity of viewers
zone	ResponDis(ourSetting, function(zone) { console.log("viewer"+ z +" is in zone: "+zone[z]); });	Register callback for current zone of each viewer
contentSize	ResponDis(ourSetting, function(contentSize) { console.log("viewer"+ z +" content optimum size:"+ contentSize[z]); });	Register callback for appropriate content resolution (pixels) of each viewer
multiViewerZone	ResponDis(ourSetting, function(multiViewerZone) { console.log("recommended zone for multiple viewers:" + multiViewerZone});	Register callback for a recommended zone to serve all the viewers
numViewersPerZone	ResponDis(ourSetting, function(numViewersPerZone) { console.log("number of viewers in"+ z + numViewersPerZone[z]); });	Register callback for the number of viewers in each zone
totalViewers	ResponDis(ourSetting, function(totalViewers) { console.log(totalViewers + "viewer(s) are detected"); });	Register callback for the number of viewers in each zone

Table 2 presents the key features that are encapsulated in the ResponDis framework. *Proximity*(s) provides an array containing the proximity of all current viewers of the display. *Zone*(s) gives an array containing the zone of all current viewers. *contentSize*(s) makes available an array containing the optimal content size (in pixels) of all current viewers. *multiViewerZone*(s) gives a number representing the recommended zone considering all current viewers. *totalViewers*(s) provides a number that shows the number of simultaneous viewers.

Settings is one of the key features of ResponDis and allows the framework parameters and computation methods to be configured. Table 3 shows the parameters together with their possible and default values along with the corresponding description.

trackedBodyPart defines the proximity of a viewer by measuring the distance between the screen and the viewer's body part. The proximity of the head

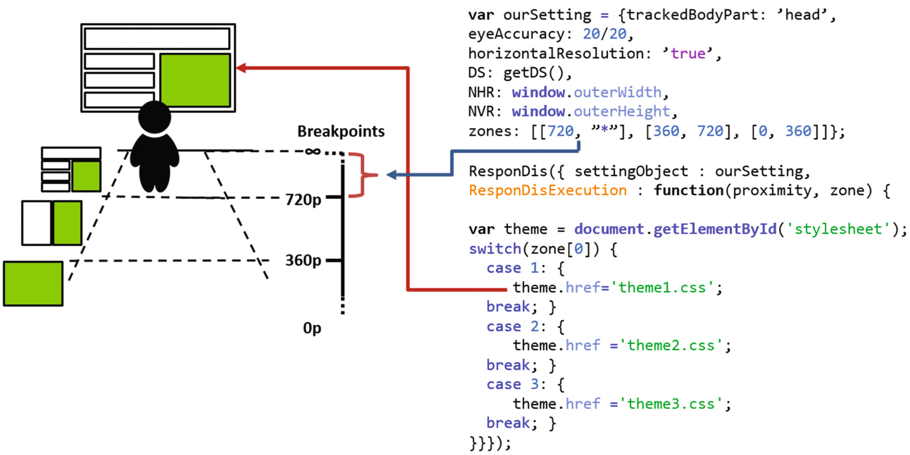
Table 3. ResponDis' configurable setting options



Option	Descriprion	Default Value	Possible Values
trackedBodyPart	the body part used to measure the distance to the display	head	{'head', 'spine base', 'spine middle point', 'neck'}
eyeAccuracy	vision acuity	20/20	some number representing the average eye sight, e.g. 20/20 is considered as "normal" eye- sight
groupTreatment	computation method to serve multiple viewers	averageProximity	{'averageProximity', 'averageZone', averageContentSize', 'medianProximity', 'medianZone', 'medianContentSize', 'mostCrowdedZone', 'closestViewer', 'firstDetectedViewer'}
horizontalResol ution	whether the zone should stand for the content horizontal resolution (horizontal or vertical)	true	{'true', 'false'}
DS	display's diagonal - should be provided by the developer to ensure better results	getDS() – approximates the display's diagonal size; can be off by a couple of inches in some cases	number in inches
NHR	display's native horizontal resolution (in pixels)	window.outerWidth	number in pixels
NVR	display's native vertical resolution (in pixels)	window.outerHeight,	number in pixels
zones	Zone breakpoints based on the appropriate content size (lowerLimit ≤ contentSize < upperLimit)	[[0,360], [360,480], [480,720], [720,1080]]	two dimensional array of limits [[lowerLimitZone1, upperLimitZone1], ..., [lowerLimitZoneN, upperLimitZonenN]]

indicates at what distance the passer-by perceives the content of the display. *eyeAccuracy* corresponds to the user's vision acuity. As a default, we assume the common viewer has normal or corrected-to-normal eyesight (20/20). *groupTreatment* offers some possible group treatment methods to be able to serve groups of users an optimal view. The integration of this feature into the framework is because public displays are considered to have multiple simultaneous viewers. We set the *averageProximity* as a group Treatment because it is a simple model and should provide all of them with a relatively good view. *horizontalResolution* specifies which resolution mode should be used to get the optimal content resolution. This resolution mode is important when deciding which zone is currently active. We set the default mode to the horizontal resolution to make it similar to the common use of screen width in responsive design. *DS* stands for the diagonal size of the display in inches. However, using JavaScript and HTML, it is impossible to calculate the exact diagonal size since it is not feasible to determine how many pixels correspond to exactly one inch. Therefore, as a default value, we approximate the *DS* using a "div" element of size 1 cm. By relating it to its pixel width and height, we compute the size of the display and its diagonal. But since a "div" of width 1 cm always gets assigned a fixed number of pixels independent of the display size, the result might be off by a couple of inches. Due to this, we strongly depend on the *DS* setting of the developer to ensure precise zone computations.

NHR is the native horizontal resolution of the screen and we set the default value to the width of the entire visible section of the screen. *NVR* is the native vertical resolution of the screen and the default value is set to the height of the full screen. *zones* defines media queries for optimal content resolution ranges. We assigned the default zones in the breakpoints at the standard resolutions used in practice i.e. 360pr, 480pr, 720pr, and 1080pr.



```
var ourSetting = {trackedBodyPart: ’head’,
eyeAccuracy: 20/20,
horizontalResolution: ’true’,
DS: getDS(),
NHR: window.outerWidth,
NVR: window.outerHeight,
zones: [[720, ”*”], [360, 720], [0, 360]]};

ResponDis({ settingObject : ourSetting,
ResponDisExecution : function(proximity, zone) {

var theme = document.getElementById('stylesheet');
switch(zone[0]) {
  case 1: {
     theme.href='theme1.css';
  break; }
  case 2: {
     theme.href ='theme2.css';
  break; }
  case 3: {
     theme.href ='theme3.css';
  break; }
}}});
```

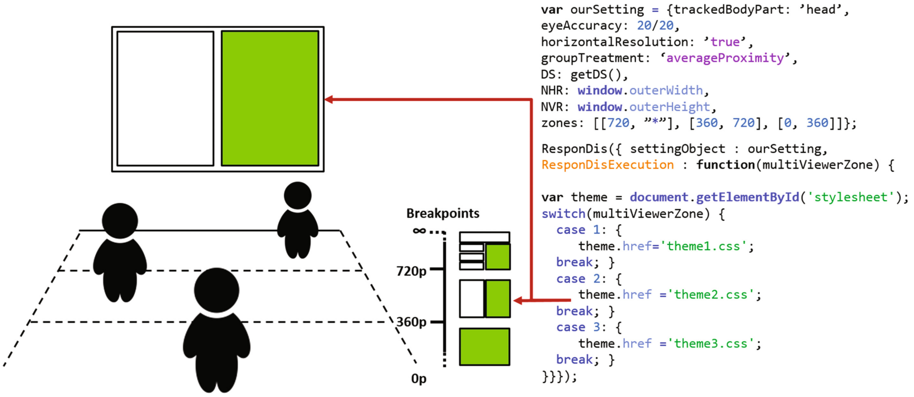
Fig. 2. A sample use of ResponDis framework for one viewer

As shown in Fig. 2, to use the framework, the developer has to construct a *settingObject*, in which they can redefine all factors of their choosing. They then have to define the *RespondDis* object, which takes as arguments the constructed *settingObject* and the function *ResponDisExecution*. The *ResponDisExecution* function has to be defined by the developer. Depending on the given parameters (*proximity*, *zone*, *contentSize*, *multiViewerZone*, *numViewersPerZone*, *totalViewers*), the content, layout or design of the display can be responsive. The *ResponDisExecution* function will be re-executed automatically by the ResponDis framework every time new data arrives from the Kinect. To simplify the definition of media queries based on the viewers' optimal content resolution ranges, these breakpoints can be defined in the *zones* parameter of the *settingObject*. Then, as represented in Fig. 2, the framework based on the configuration in the *settingObject* provides which zone each user is standing in. Likewise, the designer can choose what CSS style should be loaded for each optimal content resolution range (zone). When considering a single viewer, the developer can either extract the zone information of that one viewer from the parameter *zone* or use the zone value of *multiViewerZone*, as it returns the same value for a single viewer. Using the calculated parameters, the designer would also be able to make the adaptation fluid. To do so, they can use the *contentSize* as well as *proximity* of viewers.

As mentioned before, the framework also offers different methods for dealing with the case of multiple simultaneous users. These methods include:

(1) average proximities of all current viewers "*averageProximity*", (2) the average zone number of individual viewers "*averageZone*", (3) the average appropriate content size of all viewers "*averageContentSize*", (4) the median proximity of all viewers "*medianProximity*", (5) the median of the zones of individual viewers "*medianZone*", (6) the median of the appropriate content size for all viewers "*medianContentSize*", (7) the zone with the most viewers "*mostCrowdedZone*", (8) the zone of the closest viewer "*closestViewer*", (9) the zone of the first detected viewer "*firstDetectedViewer*"

These are used to calculate which zone should be activated in order to determine what and how content should be displayed. The developer can specify the preferred method as part of the framework setting. If different elements of the UI should be adapted based on different *groupTreatments*, several setting objects can be constructed, and multiple functions can be defined. Since the framework, in addition to the appropriate content size, provides other functionalities such as the total number and the proximity of the viewers, the zone of each viewer, and the number of viewers in each zone, developers can easily define their own group treatment method. As shown in Fig. 3, the designer using the *multiViewerZone* feature of the framework can decide what CSS style should be be loaded for each optimal content resolution range (zone).



```
var ourSetting = {trackedBodyPart: ’head’,
eyeAccuracy: 20/20,
horizontalResolution: ’true’,
groupTreatment: ‘averageProximity’,
DS: getDS(),
NHR: window.outerWidth,
NVR: window.outerHeight,
zones: [[720, ”*”], [360, 720], [0, 360]]};

ResponDis({ settingObject : ourSetting,
ResponDisExecution : function(multiViewerZone) {

var theme = document.getElementById('stylesheet');
switch(multiViewerZone) {
  case 1: {
     theme.href='theme1.css';
  break; }
  case 2: {
     theme.href ='theme2.css';
  break; }
  case 3: {
     theme.href ='theme3.css';
  break; }
}}});
```

Fig. 3. A sample use of ResponDis framework for multiple viewers

5 Architecture and Implementation

ResponDis is based on a client-server architecture (see Fig. 4). The architecture consists of a *Kinect 2* device connected to a Node.js server through the kinect2

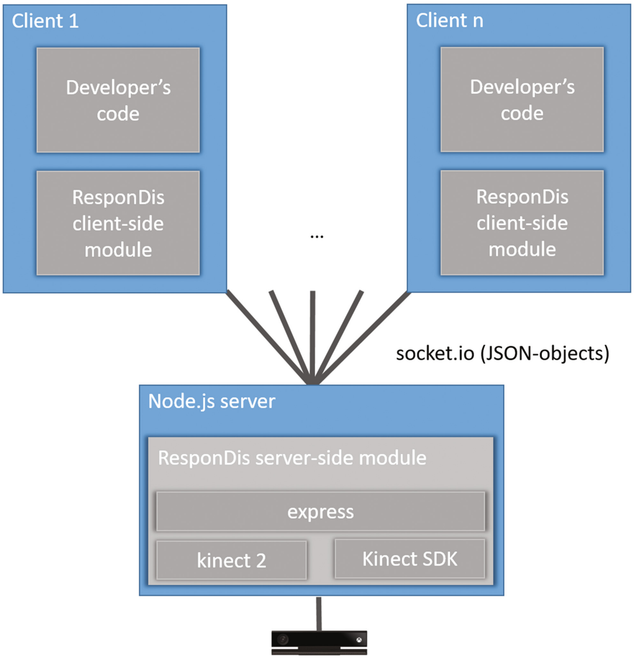


Fig. 4. ResponDis architecture

Nodejs library[5] which provides access to the Kinect 2 data from the official Microsoft Kinect SDK[6]. Node.js is a JavaScript runtime built on Chrome's V8 JavaScript engine and uses an event-driven, non-blocking I/O model. Express library[7] is also used to set up an HTTP server. The server collects all data of possible interest and sends the data in real-time as a JSON-object to the connected clients using the Nodejs socket.io[8] library. Every time new data arrives from the Kinect device, the client-side module receives the data, and, using our model with the customised settings of the developer, it computes the arguments for the framework functions.

One of the advantages of our architecture is that it enables scenarios in which the Kinect is not directly connected to the client computer. This includes cases where there is cross-device interaction involving multiple distributed clients, which is possible based on a single Kinect server. In addition, the architecture could be extended to multiple Kinects which would allow more than six people to be tracked given that Kinect can track a maximum of six people. It could also be used to handle proximities in different locations and/or at different angles. To do this, we could run multiple servers, each of which could be connected to

[5] https://www.npmjs.com/package/kinect2.

[6] https://developer.microsoft.com/en-us/windows/kinect.

[7] https://www.npmjs.com/package/express.

[8] https://www.npmjs.com/package/socket.io.

multiple Kinects and clients. Additionally, some slave servers (S) could not only send their data to their clients, but also to a master server (M). In this case, all the received data from the n slaves is combined into a single bulk object: $D_M = \{D_{S1}, D_{S2}, ..., D_{Sn}\}$ which is then be sent to all clients of (M).

6 User Study

To evaluate the proposed adaptation model, we ran a brief user study in a controlled lab setting. Our experiment had two primary goals. First, we aimed to examine whether our model improves the perception of viewers as well as usability and engagement. Second, we wanted to evaluate how our model compares to current characteristics-based static UIs. Our investigation of user engagement was due to the fact that the means of attracting and engaging viewers are considered major factors in PDS [13]. It has been shown that current user interfaces for large displays often cause difficulties in information perception with the result that user engagement is relatively low [14].

6.1 Participants

The study had 13 participants (6 females; age range (median): 19–40 (25) years). Participants were recruited at our university and were mostly ($n = 10$) from the Department of Computer Science. All the participants had normal or corrected-to-normal eyesight.

6.2 Methodology and Procedure

Before participants started performing the tasks, we introduced the system, the purpose of the study, and asked for their consent to record the experiment using a video camera.

The main task in the user study was to find a specific character in a wimmelbook [15] picture. The characters and pictures were chosen so that the character was relatively well known to most people and fitted well into the pictures. The size of the character was carefully adjusted so that it integrated well with the other characters in the picture. The presentation size of the picture for the static UI was adjusted to cover 50% of the entire display so that the view of the content for both the closet and furthest viewers was fairly good. For the adaptive UI, we defined four zones and resized the presentation of the picture for each so that a user standing in a zone had a close to optimal view for that distance.

We kept the type of adaptation simple on purpose, as our goal was to compare the adaptive and static approaches, and we thus wanted to avoid as far as possible that the user focused on the adaptation method itself.

Each participant performed the tasks using these two different interfaces. The design of the studies was cross-over, i.e. the starting order of both approaches was randomised in such a way that any user was equally likely to start with one or other UI and then use the other in the second half of the study. Furthermore, the content orders in each case were randomised.

In the study, the participants had to perform two tasks: (1) Each participant was first shown an image of a character and then was asked to enter the room from the farthest distance to the display and walk around freely to locate the character on the display. The participants were instructed to stop moving and inform the experimenter as soon as they found the character. In this way, we were able to record the time that each participant took to find the character. (2) In the second task, we wanted to measure the effect of each method for different distances. Therefore, each participant was asked to stay within a particular zone. For each zone, a different wimmelbook picture was displayed and the corresponding character shown. Similar to the first task, the participants had to look for a specific character and the time to find it was noted. This experiment was repeated for each of the zones in a random order. After finding the character in each particular zone, the participants were asked which system they preferred for that zone.

After performing the tasks, the participants were free to move around to experience and examine both the static and adaptive approach. Afterwards, we asked participants to fill out a questionnaire and answer several semi-structured questions about their experience. The questionnaire first asked participants to provide demographic information about themselves and their visual acuity before prompting questions from the Software Usability Scale (SUS) questionnaire [16] as well as the questions focusing on different aspects of user engagement. In addition, the participants were asked to give an overall rating on a 10-point Likert-scale to each of the methods, separately.

The SUS consisted of 10 questions each with a 5 point Likert scale, resulting in a single measure of usability that is between 0 and 100. We used the SUS score as the main measure of the usability of both the static and adaptive UI. Above 68%, 74%, and 80.3% usability scores are considered as average (grade C), good (grade B), and excellent (grade A) usability performances, respectively [16,17].

To measure user engagement, multiple scales are required. Previous research has proposed several user engagement scales, such as exploratory information search [18], mobile user engagement scales [19] and video game-based [20] each focusing on different aspects of user engagement. Since our study is concerned with both physical and virtual content navigation, we used O'Brien and Tom's [18] user engagement scales (UES), which combine a wide range of user engagement attributes from previous studies and consists of six dimensions (see Table 4). This wide range of dimensions enabled us to evaluate our scheme from many different points of view. While one design might perform better in some dimensions, it is possible that no difference might be found in other dimensions. By differentiating between these dimensions, we are capable of learning from where the differences come [18].

Using the O'Brien and Toms's guideline [18], we designed eleven questions relevant to the UES six-dimensions on a 5-point Likert-scale. The corresponding questions for each dimension are shown in Table 5. To evaluate each dimension,

Table 4. Factors of User Engagement (six dimensions) and their definitions

Aesthetic appeal	The visual presentation of the interface
Endurability	Overall success, whether they would recommend it to others
Felt involvement	User's feeling of being involved, having fun while interacting
Focused attention	Users's concentration during the interaction, whether they are absorbed in their task
Novelty	Whether the system is new to the user
Perceived usability	Users's cognitive and emotional feedback to the system

Table 5. User Engagement dimensions and their corresponding questions

Aesthetic appeal:	I like the kind of interaction of the adaptive/static display.
Endurability:	I would recommend the adaptive/static display to my friends and family.
	Searching using the adaptive/static display was worthwhile.
Felt involvement:	It was fun interacting with the adaptive/static display.
	I felt involved in the searching task.
Focused attention:	I was absorbed in my searching task.
	The time I spent searching just slipped away.
Novelty:	This kind of interaction with the adaptive/static display is new to me.
	I have not seen anything alike before.
Perceived usability:	Using the adaptive/static display confused me.
	I felt annoyed with the adaptive/static display.

we combined the results of the questions related to each dimension by adding up the received scores.

Related-samples Wilcoxon signed rank tests were used to compare the characteristics of the static and adaptive approaches. Furthermore, when appropriate (i.e. no violation of the normality assumption $p > 0.05$ by the Shapiro-Wilk test etc.), repeated measure ANOVA was deployed. To perform the statistical analysis, we used IBM SPSS Statistics (version 22.0, Armonk, NY: IBM Corp.) and set the minimum significance level at $p = 0.05$.

For our experiments, we used a 27" LED display, operating in landscape mode. We implemented the tasks for the user studies using our ResponDis framework. We configured the framework with the corresponding information about the display, namely DS = 27, NHR = 1920, and NVR = 1080. The other configurable options of the framework remained untouched to use the default settings. As shown in Fig. 5, we indicated all four zones on the ground using tape. A camera was positioned next to the display to film the participants while they performed the tasks.

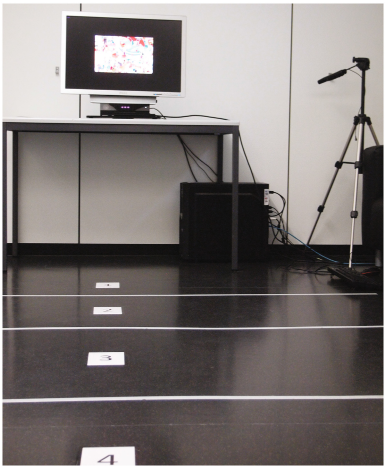


Fig. 5. Setup

6.3 Results

Viewer Perception. For the first task, the difference in walk-in time measurements for the adaptive (Median = 5.2 s) and static approaches (Median = 6.9 s) were not statistically significant ($Z = -1.503, p = 0.133$). However, the difference in the zone where the participants ended up finding the character was statistically significant for the two approaches ($Z = -2.226, p = 0.026$). Using the adaptive approach (Median = Zone 4), participants moved less distance towards the display in comparison to the static method (Median = Zone 3). Only the difference in time measurement for "Zone 4" was statistically significant (see Table 6).

To have an overall evaluation, we averaged the measured times from all four zones. Then we ran a one-way repeated measure ANOVA ($F(1, 12) = 8.191, p = 0.014$) which revealed a statistically significant difference between the overall measured times of the adaptive and static approaches. The mean difference between the two approaches was 1.867 s, in favour of the adaptive approach, showing that it had required less time. These results show that the adaptive approach (Mean ± standard deviation = 3.5 ± 1.35 s) in comparison to the static approach (Mean ± standard deviation = 5.37 ± 2.39 s) improves the content perception of viewers by 24.35%.

User Engagement. Table 7 summarises the results of the UES for both approaches where related-samples Wilcoxon signed rank tests were used. The comparison column expresses which approach received a statistically significant higher value, where a mark(**?**) indicates that there was no statistically significant difference between them. The conclusion column contains a check mark(✓) if the

Table 6. The outcome of the statistical analyses comparing the time measurements of the approaches for each individual zone

	Z	p	Adaptive time (median) [s]	Static time (median) [s]
zone 1	$Z = -1.363$	$p = 0.173$	2.7	2.2
zone 2	$Z = -0.629$	$p = 0.529$	2.2	2.5
zone 3	$Z = -1.155$	$p = 0.248$	2.5	3.6
zone 4	$Z = -3.180$	$p = 0.001$	2.7	7.2

Table 7. The results and comparison of the single-viewer user study on user engagement dimensions for adaptive and static approaches. ✓: the adaptive approach performs better, ?: no difference between the two approaches was found.

User engagement dimension	Median	Z-value	p-Value	Comparison	Conclusion
Aesthetic appeal	Adaptive: 4 Static: 3	$Z = -1.997$	$p = 0.046$	Adaptive > Static	✓
Endurability	Adaptive: 8 Static: 7	$Z = -1.930$	$p = 0.054$	Adaptive ? Static	?
Felt involvement	Adaptive: 8 Static: 6	$Z = -2.381$	$p = 0.017$	Adaptive > Static	✓
Focused attention	Adaptive: 6 Static: 7	$Z = -0.187$	$p = 0.852$	Adaptive ? Static	?
Novelty	Adaptive: 9 Static: 3	$Z = -3.099$	$p = 0.002$	Adaptive > Static	✓
Perceived usability	Adaptive: 4 Static: 4	$Z = -0.486$	$p = 0.627$	Adaptive ? Static	?

adaptive approach performed better than the static model. There was no case where the static approach was considered better than the adaptive one.

Usability and Overall Rating. The overall rating on a 10-point Likert scale for the adaptive approach (Median = 8) was statistically significantly higher than for the static approach (Median = 5), $Z = -2.641, p = 0.008$. The adaptive approach (Median = 4) also achieved a statistically significant higher score compared to the static approach (Median = 2), $Z = -3.088, p = 0.002$, when participants scored the statement "I did not feel the urge to step out of the assigned zone, I felt comfortable in my zone" statement. However, there was no statistically significant difference between the usability score of adaptive (Median = 77.5) and static (Median = 82.5) approaches ($Z = 1.652, p = 0.099$).

Qualitative Feedback. The feedback provided as comments gave us a better insight into the opinions of the participants.

The static approach was considered as the current state of the art: *"The static display is everywhere, so I've already got used to it and it was not that difficult to find something in a picture. ..."*(P11). *"This is the normal situation. So I go closer to see picture better. ..."*(P4).

The behaviour of the adaptive UI confused some viewers: *"I first was puzzled when the picture got smaller. I was approaching the screen to see it better and* *the picture got smaller. So I was thinking about going back to get the bigger picture."*(P4). (P2) highlighted one of the advantages of providing the optimal view when resizing the content *"... to display additional content or hints."* on the display's empty spaces. Another participant suggested that the content should have a fluid design: *"It would be nice if the image scaling would be smooth between the distance states."*(P5).

The adaptive approach was generally well-received: *"Adaptive approach was superior to the static approach, as I could always have a larger view when necessary. ..."* (P11) and *"I understand the approach and find it very good. ..."* (P4). At the same time, participants suggested using the approach for other purposes: *"... I was just thinking what about enlarging the picture if you step closer? This would be useful in cases I am interested in some details. Comfortable would be that I don't need to go very close but the system detects my goal and enlarges the picture for me."* (P4). Another participant added: *"I think the zones are interesting, but I'm not sure that making the content smaller is the way to go. Maybe showing different content or more content could be more interesting."* (P2). These are good inputs and could be considered in future work. A few participants commented on the design decision for Zone 1: *"At least in this setting I would not go to zone 1. It was too close for me since I had to look at the whole image to solve the task. Maybe for searching some particular spot on a map I would go as close as zone 1."* (P3). Another participant added: *"... in zone one (1) there was no significant difference between the two approaches, which was logical as I was so close to the display that I didn't need any enhancement/picture enlargement by the display."* (P11).

7 Discussion

As reported in the previous section, we found no statistically significant difference in the time it took participants to find the character when walking into the room. Nevertheless, the medians differ by 1.7 s in favour of the adaptive design. We did, however, find statistical evidence that users did not have to walk as close to the screen with the adaptive approach compared to the static one.

We noticed a vast difference in the manner in which participants walked into the room. Some walked in quickly, eager to find the character, while others expressed uncertainty and walked slowly. This could be one of the reasons why there is no significant difference in the time measurements.

When taking the average of the zone-by-zone time measurements, one can see a statistically significant difference between the two approaches, showing that the seek-time on the adaptive display was less. This could be because participants did not have to walk and adapt their view to the content. Since participants were asked to stay within their assigned zone, the difference in the seek-time measurements could not differ due to their slow or quick movement and, thus, the observed significant difference can be associated with the corresponding approaches.

Although the adaptive approach generally performed better as participants were faster in finding the character in the pictures, the static approach performed slightly better for "Zone 1" (see Table 6). However, the difference was not statistically significant. We believe that the better performance of the static approach for "Zone 1" could be a result of the framework's default setting for "Zone 1" which forces too small a distance to the screen. This closeness sometimes caused Kinect user detection failure and, thus, inappropriate adaptation, which might be the reason why the participants could not quickly find the characters. In addition, we also received feedback that *"At least in this setting I would not go to zone 1"* (P3). This suggests that careful design decisions for each zone is required.

Studying Table 7, it can be seen that the adaptive approach was always rated better or the same as the static approach. The adaptive approach was systematically better in the *aesthetic appeal, felt involvement and novelty* of the user engagement dimensions. The participants' rating of the interaction of the adaptive design was significantly better, which could be due to the fact that it adapts itself to the viewer.

The participants also had fun interacting using the adaptive approach and rated it as more novel. This is not surprising since the static approach is the one used in most existing systems, while adaptive UIs are still a topic of research.

While no systematic difference between the approaches was observed in the perceived usability dimension, there was clear feedback that the adaptive UI confused some viewers. Reviewing the comments, we learned that some participants walked closer to try to perceive more detail, and were confused that the content got smaller instead of larger. Others wished that the intentional blank space around the content used for the study had been used to zoom in or provide additional hints when the content size decreased as they approached the screen. While this highlights a potential advantage of our approach over the static approach, we had decided not to use the free space to display more content during the study to avoid potential conflicts in content. This is something that we now feel would be an important addition in future studies.

Previous research has shown that low user engagement is a result of poor system usability [21]. The statistical analysis of the usability scores for both approaches showed that the scores are close, with both above the usability average, and that there is not a systematic difference between the approaches. This means that the low user engagement rating of the static approach is not due to the low usability.

The adaptive approach was considered to be significantly better when looking at the overall rating compared to the static approach. Participants also did not feel the urge to step out of their assigned zone when using the adaptive model, whereas they did while using the static design. We noticed however that when participants were asked to stay within a particular zone while using the static approach, some leant forward as far as their balance permitted in order to get a better view.

8 Conclusion and Future Work

We have proposed a model which could be used to integrate the proximity of viewers to a public display as an additional dimension of the viewing context considered in responsive design. In order to experiment with the model, we developed a JavaScript framework that supported the rapid prototyping of applications. This was used to carry out a basic user study which compared a conventional static UI typical of current public displays with an adaptive UI based on our model. The results of the study with single viewers showed that the adaptive approach not only provides a better view, but also improves the user engagement in terms of aesthetic appeal, felt involvement and novelty. In the future, we aim to investigate the effect of the model on multiple viewers and in more realistic settings such as the deployment of an information service within our department.

References

1. Sarabadani Tafreshi, A.E., Norrie, M.C.: ScreenPress: a powerful and flexible platform for networked pervasive display systems. In: Proceedings of the 6th ACM International Symposium on Pervasive Displays (PerDis) (2017)
2. Nebeling, M., Norrie, M.C.: Responsive design and development: methods, technologies and current issues. In: Daniel, F., Dolog, P., Li, Q. (eds.) ICWE 2013. LNCS, vol. 7977, pp. 510–513. Springer, Heidelberg (2013). doi:10.1007/978-3-642-39200-9_47
3. Gardner, B.S.: The spark of innovation begins with collaboration. Inside the Digital Ecosyst. **11**(1) (2011)
4. Frain, B.: Responsive Web Design with HTML5 and CSS3. Packt Publishing Ltd. (2015)
5. Nebeling, M., Matulic, F., Streit, L., Norrie, M.C.: Adaptive layout template for effective web content presentation in large-screen contexts. In: Proceedings of the 11th ACM Symposium on Document Engineering (DocEng) (2011)
6. Bababekova, Y., Rosenfield, M., Hue, J.E., Huang, R.R.: Font size and viewing distance of handheld smart phones. Optom. Vis. Sci. **88**(7), 795–797 (2011)
7. Rushing, K.: Home Theater Design. Rockport Publishers (2004)
8. Sugawara, M., Masaoka, K., Emoto, M., Matsuo, Y., Nojiri, Y.: Research on human factors in Ultrahigh-Definition Television (UHDTV) to determine its specifications. SMPTE Motion Imaging J. **117**(3), 23–29 (2008)
9. Dostal, J., Kristensson, P.O., Quigley, A.: Multi-view proxemics: distance and position sensitive interaction. In: Proceedings of the 2nd ACM International Symposium on Pervasive Displays (PerDis) (2013)
10. Dostal, J., Hinrichs, U., Kristensson, P.O., Quigley, A.: SpiderEyes: designing attention-and proximity-aware collaborative interfaces for wall-sized displays. In: Proceedings of the 19th International Conference on Intelligent User Interfaces (IUI) (2014)
11. Snellen, H.: Probebuchstaben zur bestimmung der sehschärfe. H. Peters (1873)
12. Jiang, F., Zhou, L., Liu, K., He, X., Liu, X., Li, Z.: P-35: the prospect assessment of 65+ Inch TVs based on the size of mainstream living rooms in china. In: SID Symposium Digest of Technical Papers, vol. 46. Wiley Online Library (2015)

13. Müller, J., Wilmsmann, D., Exeler, J., Buzeck, M., Schmidt, A., Jay, T., Krüger, A.: Display blindness: the effect of expectations on attention towards digital signage. In: Tokuda, H., Beigl, M., Friday, A., Brush, A.J.B., Tobe, Y. (eds.) Pervasive 2009. LNCS, vol. 5538, pp. 1–8. Springer, Heidelberg (2009). doi:10.1007/978-3-642-01516-8_1
14. Li, A.X., Lou, X., Hansen, P., Peng, R.: Improving the user engagement in large display using distance-driven adaptive interface. Interact. Comput. **28**, 462–478 (2015)
15. Rémi, C.: Reading as Playing. Emergent Literacy: Children's Books from 0 to 3 13 (2011)
16. Sauro, J.: A Practical Guide to the System Usability Scale: Background, Benchmarks & Best Practices. Measuring Usability LLC (2011)
17. Sauro, J.: Measuring Usability with the System Usability Scale (SUS) (2011)
18. O'Brien, H.L., Toms, E.G.: Examining the generalizability of the user engagement scale (UES) in exploratory search. Inf. Process. Manage. **49**(5), 1092–1107 (2013)
19. Kim, Y.H., Kim, D.J., Wachter, K.: A study of mobile user engagement (MoEN): engagement motivations, perceived value, satisfaction, and continued engagement intention. Decis. Support Syst. **56**, 361–370 (2013)
20. Wiebe, E.N., Lamb, A., Hardy, M., Sharek, D.: Measuring engagement in video game-based environments: investigation of the user engagement scale. Comput. Hum. Behav. **32**, 123–132 (2014)
21. Robertson, G., Czerwinski, M., Baudisch, P., Meyers, B., Robbins, D., Smith, G., Tan, D.: The large-display user experience. IEEE Comput. Graph. Appl. **25**(4), 44–51 (2005)