ISSN 2464-4617 (print)
ISSN 2464-4625 (CD)
Computer Science Research Notes
CSRN 2802
Short Papers Proceedings
http://www.WSCG.eu

RDSpeed: Development Framework for Speed-Based Adaptation of Web Content on Public Displays

Amir E. Sarabadani Tafreshi, Adrian Wicki, and Gerhard Tröster
Wearable Computing Lab., ETH Zürich,
CH-8092 Zürich, Switzerland,
{tafreshi | troester} @ife.ee.ethz.ch

Abstract

Viewers of public displays perceive the content of a display at different walking speeds. While responsive design (RD) adapts web content to different viewing contexts, so far only the characteristics of the device and recently the proximity of the viewers are taken into account. Yet, little attention has been paid to speed-based adaption of content and its potential in case of public displays. We therefore decided to develop a framework that would support speed-based adaptation of public display applications. In this paper, we present a framework called RDSpeed that allows developers and designers alike to easily utilize our speed-based adaptation technique and integrate them into their own applications. RDSpeed extends the standard RD definition by adding new media queries for each adaptation technique. Media queries have long been established as the go-to technique for developing responsive web applications when dealing with a variety of different devices. A user study was conducted to investigate the potential of our content adaptation technique, and possible use and extensions in the future. We show several example adaptive applications of RDSpeed, as well as discussing advantages and limitations of our framework as revealed by our user study.

Keywords

Development framework, Responsive Design, Pervasive displays.

1 INTRODUCTION

The issue of engaging viewers of pervasive display systems (PDS) is a well-known problem [MWE+09]. The vast majority of PDS effectively disappear as people have become so accustomed to their low utility that they became highly skilled at ignoring them. Therefore, researchers are continuing to explore new forms of PDSs and studying their impact on users and user communities [STN17]. However, regardless of what content or services PDSs offer, their usability is limited on whether and how viewers perceive the content. Viewers' viewing experience depends on viewers' viewing context which can be utilized for adapting the content. Content adaptation not only can improve viewer's perception and viewing experience, but also enhance user engagement [TMN17]. The well-known issue that public display viewers ignore PDSs is promoting researchers to experiment with responsive design (RD) techniques to increase the utility and user engagement of PDS.

While responsive design adapts web content to various viewing contexts, viewer walking speed has not been taken into account as part of the viewing context in the case of public displays. It is important to note that viewers of public displays perceive the content at different walking speeds (see Fig. 1) which limit viewers' perception and content consumption. Humans have limited visual bandwidth and adapting content based on viewer walking speed can boost viewers' perception [Wic17].

In this paper, to enable researchers and developers to explore new forms of adaptation based on viewers' walking speed, we present a framework that supports the rapid development of web-based PDS applications featuring speed-based adaptation. The resulting Speed-Based-Responsive-Design (RDSpeed) framework extends media queries in the standard style sheet language used for describing the presentation of content (i.e., CSS[1]), to also consider the viewers' walking speed as part of viewing context in the case of public displays. To detect the number and speed of viewers, we use the camera-based Kinect sensing technology which is readily available as a commercial product. The framework is based on the standard CSS definition and builds on

[1] https://www.w3.org/standards/webdesign/htmlcss

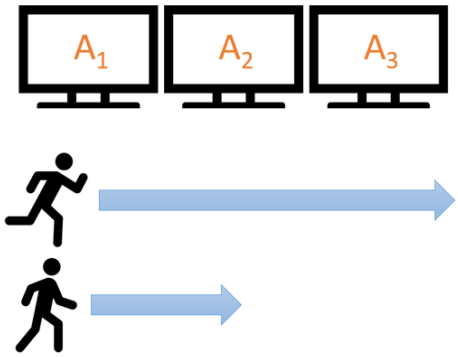


Figure 1: Demonstration of viewers passing in front of multiple displays at slow and fast walking speeds.

the Kinect SDK. Similar to responsive design breakpoints specified using CSS3[2] media queries, the framework supports a speed range concept which allows designers to specify more radical changes to the content and layout as viewers walk at different speed ranges.

The RDSpeed framework is characterised by four main aspects: (1) useful programming abstractions for the use of our proposed approach using extension of standard RD techniques (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 adaptation of a variety of multi-display ecosystems, (4) an extensible architecture which allows multiple Kinect sensors to be used and connected to different distributed servers in order to cover wider viewing-field, and track more people in different locations and at different viewing angles.

We begin with a discussion of related work in Sect. 2, before presenting our speed-based adaptation model and the features of the RDSpeed framework in Sect. 3. The architecture and implementation are discussed in Sect. 4, followed by a description of the two adaptive sample applications developed using RDSpeed in Sec. 5. We then, in Sect. 6, report on an initial user study carried out to assess how our content adaptation techniques based on walking speed perform and how they are perceived when users experience them. We then present the results of the study in Sect. 7. The paper concludes with a discussion of our results in Sect. 8 followed by final remarks in Sect. 9.

2 BACKGROUND

Responsive web design (RWD) is an approach to web design to adapt sites to deliver an optimal viewing and interaction experience [Mar13]. With current RWD techniques, content can be adapted by considering device characteristics; but contextual factors beyond the device have so far received only little attention. By emergence of global networks of PDSs, the contextual factors are now playing an essential role in the adaptation [STN17]. Developers have used different techniques for RWD, and understanding these techniques would better clarify the appropriate techniques that could be extended to support additional contextual factors. Peterson [Pet14] gives a brief history of responsive web design where she states that in the year 2000 most websites were designed for a screen width of 800px. As screens grew wider over the years, developers would usually resort to fixed-width designs, so websites would look exactly the same on all screen widths. These fixed-width designs then led to the first mobile websites in the mid-2000s. Consequently, developers usually had to maintain two separate versions of a website — one for full-width desktop computer screens and one for much smaller mobile screens. As a consequence, most mobile websites were heavily stripped down versions of their full-width counterparts.

To deal with increasing variety of devices, web developers started to move to fluid instead of fixed-width layouts. *Flexible layouts* utilize flexible units such as percentages instead of absolute units such as points and pixels. Later, with the introduction of *media queries* and means of selecting and sizing images according to viewing context [Gar11], it was also possible to dynamically adjust a web layout to a given device screen. But it was not until 2010 that [Mar13] first introduced the concept of responsive web design which combined the concept of *media queries* with *flexible layouts*. *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 device orientation, viewport size and pixel density [Fra15]. Both of these concepts are needed in order to create adaptive layouts which display content in a comfortable and visually appealing way on all available devices.

While an increasing number of practitioners advocate a mobile-to-desktop adaptation which is a mobile-first strategy in conjunction with responsive design, the focus in the research domain has mainly been on support for desktop-to-mobile adaptation [BN13]. However, only little attention has been paid in either practice or research to adaptation to very large displays, even though these are now common in (semi-)public spaces. However, one study investigated adaptation to large displays, particularly in the case of text-centric websites (e.g., online newspapers) [BMSN11]. In this work, the authors proposed a set of design metrics, developed an adaptive layout template and carried out a user study to show the benefits of the adaptation.

As the benefits of web technologies have been recognized in the PDS research community [TJS15], web architectures are mostly now the go-to sys-

[2] https://www.w3schools.com/cssref/css3_pr_mediaquery.asp

ISBN 978-80-86943-41-1

tem design used for public display applications (e.g., [NEL16, NEL15, ESNL15]). Responsive design support is one advantage of web technologies for PDS applications and adapting the content can improve the user engagement [TMN17, STMT18, STBT18]. Dostal et al. [DKQ13, DHKQ14] also showed that adaptive interfaces are useful for addressing the user's various attention states. It's worth noting that content adaptation is passive, and unlike the interaction methods that require active interaction of users [RLC+06, LLCB08, BBKP07], happens automatically.

While it might be sufficient to consider the device size and the resolution for adapting the content to personal devices, it is important to also take other factors into account when adapting web content to public displays [TMN17]. Further, unlike personal devices that are easily accessible by users [STSTST17], some larger displays are out of reach. For instance, a study revealed that the average mobile smartphones viewing distance is about 12 inches [BRHH11], and therefore a user can easily adjust the distance to improve legibility if required. However, in the case of a public screen, users might require to significantly adjust their position or motion path, which they would only do if they were already engaged with the display. To address this gap, Tafreshi et al. [TMN17] proposed a model that integrates the proximity of viewers to a public display as an additional dimension of the viewing context considered in responsive design. This work also included a framework to support rapid prototyping for proximity-based adaptive display user interfaces. However, viewer proximity is just one factor when dealing with public displays; other factors such as viewer walking speed might be potential extension points for responsive PDS which gives researchers and developers the opportunity to study and apply new forms of adaptation in their PDS applications. As recent research [Wic17] showed, content adaptation based on walking speed improves the speed performance in searching tasks. Despite the advantages of walking speed-based content adaptation, little attention has been paid to use and study them for public display applications. One reason may be the lack of support offered to researchers and web developers.

3 THE RDSPEED FRAMEWORK

The main goal of the RDSpeed framework was to enable PDS designers and researchers to explore different styles and uses of speed-based adaptation on web-based PDS applications with focus on establishing design guidelines that could improve the public displays viewing experience in certain settings. RDSpeed provides an extension to the standard responsive design techniques (i.e., CSS) which provides a simple and familiar way for developers and designers alike to integrate our content adaptation technique into their own applications.

An initial starting point to build the framework was to identify the walking speed of viewers. To do so, it is needed to measure how quickly a viewer moves from one place to another. In principle, the speed (S) is a mathematical determination of the distance (D) a viewer moved divided by the time (ΔT). Likewise, we build our adaptation model given the $S = \frac{D}{\Delta T}$ formula. The distance that a viewer moved is based on the viewer position history – the previous (h) and current ($h+1$) position. Given the user position history in two dimensions of x-axis and y-axis, the viewers' walking distance (D) can be calculated using *Euclidean metric*. In two-dimensional Euclidean space, if $h = (x_h, y_h)$ and $h+1 = (x_{h+1}, y_{h+1})$ then the distance is given by

$$D = \sqrt{(x_h - x_{h+1})^2 + (y_h - y_{h+1})^2} \quad (1)$$

Accordingly, in our adaptation model, the resulting formula that calculates the walking speed as follows:

$$WalkingSpeed = \frac{\sqrt{(x_h - x_{h+1})^2 + (y_h - y_{h+1})^2}}{\Delta T} \quad (2)$$

Similar to how viewport size is often used in CSS3 media queries to define layout breakpoints, the calculated *WalkingSpeed* could be used to define design breakpoints in the form of media queries as well as for fluid web layouts. With respect to our adaptation model, these breakpoints correspond to a speed range so that content is adapted depending on the speed range in which a user is currently walking. As will be discussed later, in the case of multiple users walking at different speeds, there needs to be a strategy that determines which speed range to consider. We begin by describing the features and operation of the RDSpeed framework in terms of a single user.

The framework uses our model to compute the walking speed "walkingSpeed" at each time point. Accordingly, the framework provides the walking speed of the viewers which can be used for fluid design. In addition, a developer, in a customized setting, can partition the walking speed into multiple ranges defining the set of speed ranges.

Table 1 lists the key features that are encapsulated in the RDSpeed framework. *walkingSpeed* provides a number that shows the joint walking speed of current viewers of the display. *walkingSpeedRange* makes available a number containing the joint speed range of the viewers. In the case of multiple viewers, the speed and the range are computed based on the computation method

 ISBN 978-80-86943-41-1

Features	Examples	Description
Settings	RDSpeed ({bodyJoint: 'head', speedRange: [{'slow': 0, 'normal': 2, 'fast': 4}], multiUserAction: 'average'});	Set the body part to compute the walking speed, and the method to serve multiple viewers
walkingSpeed	body {font-size: calc(walkingSpeed * 10px);}	Register callback for the walking speed
walkingSpeedRange	@media (walkingSpeedRange: slow) {body{font-size: 12px;}}	Media query-based register callback for current walking speed range
numPeopleTracked	@media (numPeopleTracked >= 2) {#single { display: none; }}	Media query-based register callback for number of tracked viewers

Table 1: RDSpeed features with code examples. Callback media queries are based on the *settings* object.

Option	Description	Default Value	Possible Values
bodyJoint	which joint is used to position the body	'head'	'head', 'spine base', 'spine middle point', 'neck'
speedRange	speed breakpoints based on viewer's walking speed speed thr. N <= speedClassN < speed thr. N+1	['slow': 0, 'normal': 2, 'fast': 4]	array of names and thresholds [['SpeedName1', speed threshold1], ... , ['SpeedNameN', speed thresholdN]]
multiUserAction	computational method to serve multiple viewers	'average'	'average'; 'firstViewer'; 'majority';

Table 2: RDSpeed configurable setting options

configured in the framework. *numPeopleTracked* gives a number representing the number of current simultaneous viewers.

Settings is one of the key features of RDSpeed and allows the framework parameters and computation methods to be configured. The parameters together with their description along with the corresponding defaults and possible values are shown in Table 2.

bodyJoint defines what part of a viewer's body should be tracked to calculate the walking speed of a viewer. *speedRange* defines media queries for walking speed ranges. We adjusted the default value of this feature based on people's walking speed experiences[3] on treadmill i.e., slow (walking very slowly or standing still): 0-2mph, normal (walking at a normal everyday pace): 2-4mph, and fast (walking very quickly or running): above 4mph. *multiUserAction* offers some possible group handling methods to be able to serve groups of viewers an optimal view. The integration of this feature into the framework is because public displays are considered to often have multiple simultaneous viewers. The offered methods are devised from the feedback and discussions with the participants from the study which will be discussed in Sec. 6. These methods include: (1) the walking speed of the first detected viewer "*firstViewer*", (2) the average walking speed of individual viewers "*average*", and (3) the speed range with the most viewers "*majority*".

As shown in Fig. 2, to use the framework, the developer can change the constructed configurations in the RDSpeed setting object. They then have to write the style codes within the <*RDSpeedStyle*> tag, which allows to make use of special media-queries. The reason we did not use the normal <*style*> tag is because most modern browsers automatically attempt to correct invalid CSS code. In that case, browsers would remove the information not corresponding to the official CSS specification, such as the *walkingSpeed* keyword. Depending on the given parameters ("*bodyJoint, speedRange, MultiUserAction*") the content, layout or design of the display can be responsive. The corresponding media-queries and functions will be re-executed automatically by the RDSpeed framework every time new data arrives from the Kinect sensor. To simplify the definition of media queries based on the viewers' walking speed ranges, these breakpoints can be defined in the *speedRange* parameter of the *RDspeedSettings*. Then, as represented in Fig. 2, the framework provides the *speedRange* a viewer is moving at, based on the config-

[3] https://www.runnersworld.com/for-beginners-only/what-are-the-right-walking-and-running-speeds

 ISBN 978-80-86943-41-1

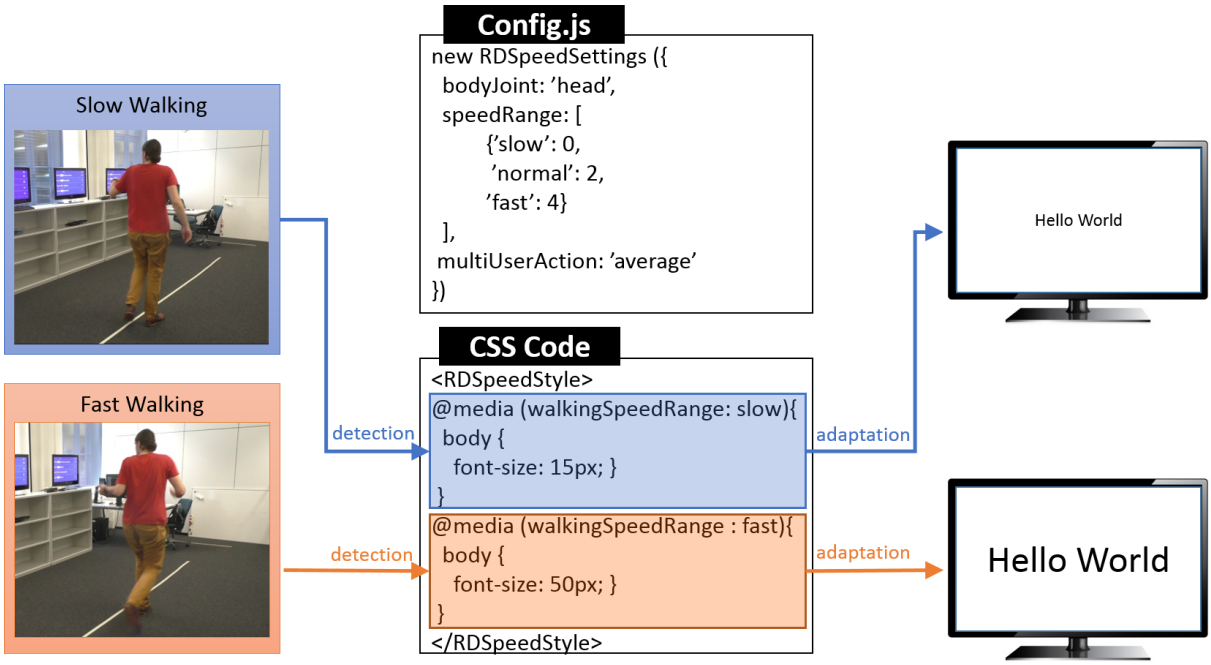


Figure 2: A sample use of the RDSpeed framework

uration in the *RDspeedSettings*. Likewise, the designer can choose what CSS style should be loaded for each walking speed range (*speedRange*). When considering a single viewer, the developer can extract the *walkingSpeedRange* information of that one viewer from the parameter *walkingSpeedRange*.

Using the computed parameters, the designer would also be able to make the adaptation fluid. To do so, they can use the *walkingSpeed* of a viewer. The *walkingSpeedRange* , *walkingSpeed*) parameters in the case of multiple simultaneous users, will be calculated using the preferred method that the developer can specify as part of the framework setting. Since the framework, in addition to the viewer walking speed and speed Range, provides more functionalities (i.e., the total number of the viewers), developers would be able to define different adaptations to handle single and multiple viewer(s).

4 ARCHITECTURE AND IMPLEMENTATION

Figure 3 illustrates the RDSpeed architecture. RDSpeed is based on a client-server architecture and consists of three main components – sensor, client and server-side components.

The server component forms the central backbone of the architecture and is written in Node.js. Node.js is a JavaScript runtime and uses an event-driven, non-blocking I/O model. The server component receives the pre-processed Kinect sensor data from the sensor components and aggregates them together. The server sends the aggregated data in real-time as a JSON-object to the connected clients using the Nodejs socket.io[4] library.

The client-side modules are responsible to parse the RDSpeed CSS code, extract the RDSpeed media queries and functions, receive the event notifications, and apply the developer's RDSpeed media queries and functions. The client-side module receive the new data from the server component, and then, using our model fed with the customized settings of the developer, they compute the arguments for the framework functions.

Sensor component is responsible for gathering the data from the *Kinect 2* device and relay the data to the server component. The data from the sensor component is acquired through the kinect2 Nodejs library[5] which provides access to the Kinect 2 data from the official Microsoft Kinect SDK[6]. The sensor component enables extension of the architecture with multiple Kinects which would allow more than six people to be tracked given the fact that a single Kinect can track a maximum of six people concurrently. It could also be used to handle speeds at different angles and/or in different locations. To do this, multiple sensor components can be connected to the server. One of other advantages of our architecture is that it enables scenarios in which the Kinect is not directly connected to the client computer. This includes the scenarios where there is cross-device interaction involving multiple dis-

[4] https://www.npmjs.com/package/socket.io
[5] https://www.npmjs.com/package/kinect2
[6] https://developer.microsoft.com/en-us/windows/kinect

ISBN 978-80-86943-41-1

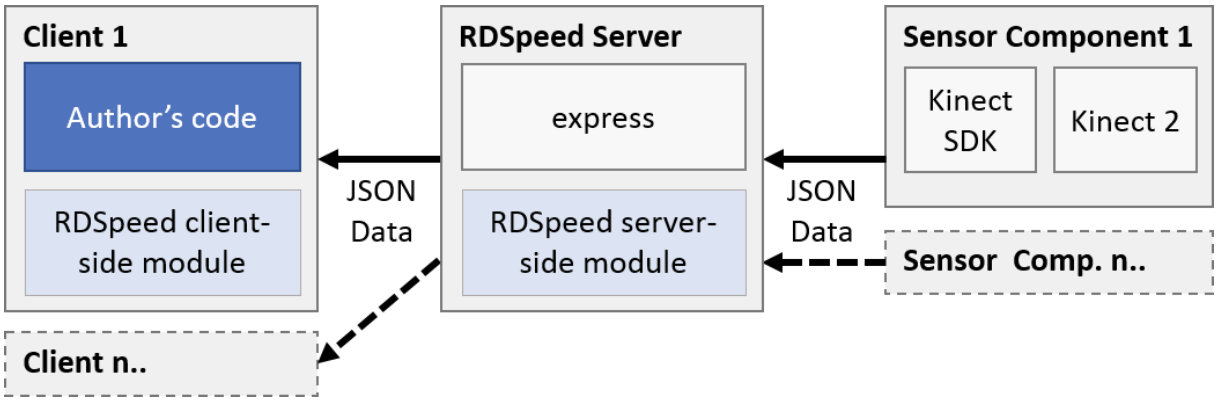


Figure 3: RDSpeed architecture.

tributed clients, which is possible with a single Kinect server [STST18]. Further, thanks to our cross-device web framework, our adaptation method can run across any kind of device.

5 APPLICATIONS

To define, demonstrate and test the capabilities of the RDSpeed framework, we now present two speed-based adaptive PDS applications created using our framework.

A first example application is a Train schedule application shown in Figure 4a. As the name suggests, the application shows departure times and destinations of trains. We chose this application because many people frequently use trains in their daily life and use public displays to look-up their train connections while some of them are in rush. We implemented two layouts with the intent of making one suitable for slow walking or still-standing viewers (see Figure 5a) and the other one suitable for fast walking viewers (see Fig. 5b). Here, our application design is so that a display only shows the necessary minimum information that is expected from a train schedule (i.e., train destination, platform, and departure time) in larger sizes to be better readable by fast-walking viewers. When a viewer stands or walks slowly, the display shows additional information such as occupancy and acceptability which requires reducing the font-sizes to show all information.

Our second example application, news application, was designed to increase awareness (see Fig. 4b). Our decision to create the adaptive news application was motivated by the fact that many people are curious about the current news while they are walking/running in front (semi-) public displays at different speeds. We believe that adaptation of current news content, particularly for emergency cases such as natural disasters or terrorist incidents, could help more people to be informed about the current situation. Here, in a context where a viewer is walking quickly or running, a display only shows the title of the news. In the case where the viewer walks normally or slowly, the display also shows the news abstract. If the viewer stands, the display shows the full news.

6 USER STUDY

We ran a lab study to get better insight on how our content adaptation technique based on walking speed perform and how they are perceived when users experience them. We opted for a lab study, which gives us control over the equipment and the environment, and the ability to directly interact with the participants and ask for clarification or feedback.

6.1 Participants

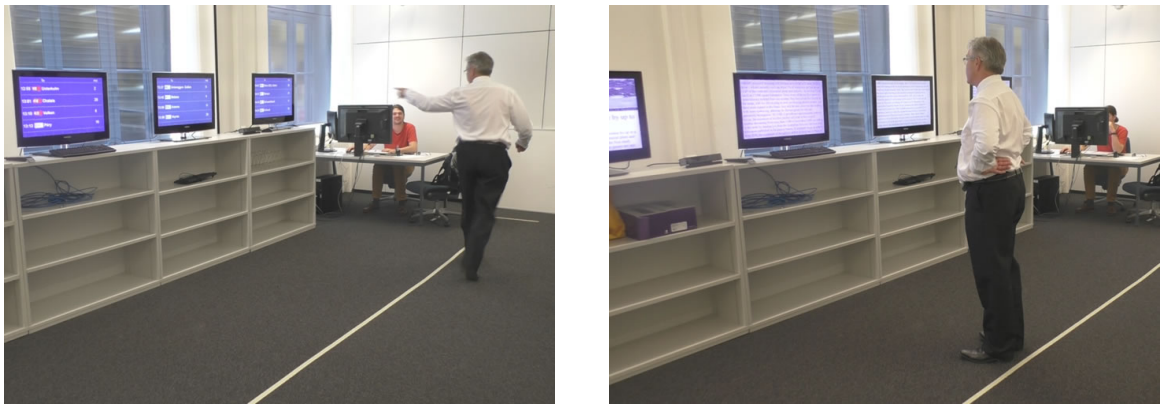
We recruited a total of 16 participants, 15 male, and one female. Most participants (14) had no computer science background. All of the participants either had normal or corrected to normal eye vision. Our participants were recruited using the *snow ball* sampling method at our University and also our social circle. 11 participants had age range of 25-34, two (18-24) , one 35-44, and two 55-64.

6.2 Setting

We used our lab's meeting room to install four 32-inch displays. The displays are placed on four shelves, each with a height of 1.2m. This ensures that the displays are located at eye-height for the majority of our participants. We introduced a small gap of approximately 60cm between the screens because this gives our participants more space when they have to walk in front of the setup at various speeds. We used two Kinect sensors to cover the space in front of the displays.

Figure 6 illustrates our study setup. To conduct the study, we opted for the train schedule application. This application is an example of an existing real-world application on public displays. In addition, most participants are already familiar with the layouts as they should have encountered similar applications at a train

ISBN 978-80-86943-41-1



(a) Train schedule (b) News application

Figure 4: The two adaptive sample applications: **(a)** Train schedule and **(b)** News application.

station or bus stop. The exemplary content generated by the application is implemented using the ScreenPress platform which is a platform developed for the rapid prototyping of PDS [STN17]. Each screen shows a total of 4 trains departing to randomly generated destinations. The destinations are chosen from a list of all Swiss municipalities[7].

6.3 Procedure and Methodology

We conducted the study with individual participants. Before they started the experiment, the participants were given a small introduction on what the system does and how they can expect it to adapt the content based on to their movement. We also asked for their consent to record the experiment using a video camera. We then enabled the walking speed-based content adaptation for the train schedule application and allowed the participants to freely experiment and explore the system. 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 before answering the questions about usability (easy and efficient to use), utility (functionality useful), stimulation (inspiration, wow experiences), value (importance), novelty, enjoyment, ease of learning, and responsiveness of the system. In addition, we asked several semi-structured and optional open ended questions about whether they encountered any technical issues or limitations of the system, the problems such a system might have when deployed to public settings, and the features they recommend to improve in the future system design as well as comments or suggestions.

Figure 5: The two different layouts of the train schedule application – **(a)** all additional information such as occupancy, train composition and icons are shown. To fit all information, the font-sizes have been decreased; **(b)** only the destinations, departure times and platform numbers are shown. As more space is available the font-sizes have been slightly enlarged compared to layout (a).

[7] Obtained from `https://www.bfs.admin.ch/bfs/de/home/grundlagen/agvch.html` accessed on the 06-05-2017.

7 RESULTS

The participants were able to freely explore the speed-based adaptive train schedule application.

Figure 7 shows how the participants rated the system according to novelty, utility, usability, value and importance, enjoyment and stimulation.

As can be seen in figure 7, the system was quite well accepted by our participants. Especially the novelty, utility (usefulness), usability, the value and importance are rated highly. However, the enjoyment and stimulation received a bit worse rating compare to other factors.

Participants gave the system an average overall rating of 6.375 (see Figure 8). The majority of participants (11/16) agreed or strongly agreed on the system ease of use. Three were neutral about the system's ease of learn. Most participants stated that, at first, the system was confusing, but after some time they were able to understand it and fully use its functionality.

Almost half of the participants (n=7) felt that the system was fairly or very responsive. Five participants stated that the system was somewhat responsive while four felt it was a little responsive.

ISBN 978-80-86943-41-1

Figure 6: Panorama view of the study setup.

7.1 Qualitative feedback

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

P2 and P4 mentioned that they would prefer a system where only one screen is adapted (the one they stand in front of) instead of all the displays at once. P15 however, brought up a counter argument to the above statement: "*I liked, that all the screens changed, because I can see multiple screens from a single point of view. In a setting where only a single screen would change, I would have to actively move in order to see the detail content on another screen.*" P5 mentioned concerns on how the system would adapt to multiple viewers. P6 mentioned that instead of adapting while somebody is in front of the screens, it would be better to adapt before somebody enters the viewing field (to avoid confusion caused by the layout switch). P6 also stated that without prior instructions, the system might be confusing. P7 mentioned that he would like the system to support actual gestures (for zooming or moving content between screens). P12 showed interest "*to have an adaptation system on a cellphone. A system that recognizes walking speed and then adapts the displayed content.*" Such adaptation could be possible by making use of the accelerometer. P2 and P4 noticed a small uncovered angle between the two Kinect sensors where the adaptation system did not detect any movement. This issue points out one of the main limitations of the framework and the need for careful positioning of the sensors in the case of using multiple Kinects.

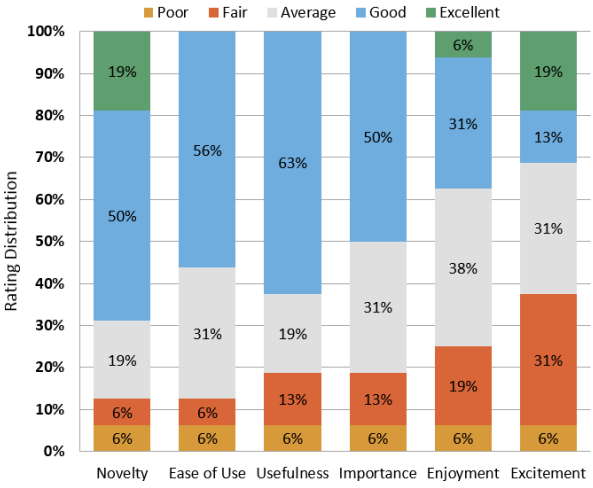


Figure 7: System adaptation rating of different factors.

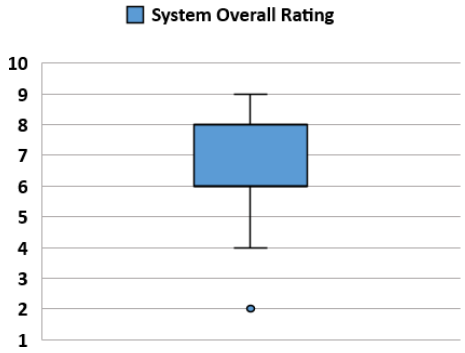


Figure 8: System overall rating

8 DISCUSSION

The participants highly rated the novelty, utility, usability, and value and importance of the system and rated the other factors such as enjoyment and stimulation a bit lower. A majority of the participants also stated that the system was easy to learn. In addition to the questionnaire feedback we were able to gather a plethora of qualitative feedback through conversations and observations. An important argument that was brought up by many participants was, that without prior instructions the system would be very confusing. Because it is not directly clear what the system does after users move in front of a display. A proposed solution to this specific problem was, to position the Kinect sensors before and after the displays, resulting in any adaptation happening before viewers are in front of the screens. This would additionally solve the problem of viewers getting confused or irritated by layout switches, as changes would occur before they are able to see the screens.

During the exploration phase many participants raised concerns regarding the suitability of the automatic

ISBN 978-80-86943-41-1

adaptation system for multiple users. Through discussions and conversations with the participants, we were able to devise three different solutions to dealing with multiple users. **First come, first adapted** The system only adapts to the first user who enters the viewing field of the sensors. The adaptation system then waits until no bodies are tracked anymore until it starts to adapt again (as soon as another user enters the viewing field of the sensors). **Majority adaptation** The system adapts to the majority of the users, i.e., adaptation is done according to whatever walking speed the majority of the current viewers exhibit. **Average adaptation** All calculated values are averaged over all tracked users and then the adaptation is done based on these values. We integrated these strategies into the RDSpeed framework as discussed in Sec. 3

9 CONCLUSION

We have presented a model which could be used to integrate the walking speed of viewers to a public display as an additional dimension of the viewing context considered in responsive design. In order to enable researchers and developers to explore new forms of adaptation based on viewers walking speed, we developed a framework that supports the rapid prototyping of speed-based adaptive applications. The framework was used to carry out a basic user study which allowed to get better insight on how our speed-based content adaptation technique perform and how users experience them. The feedback and the discussion with the study participants helped us to devise methods to handle multiple simultaneous viewers and extend our framework to support them. Now that we have the framework, we plan to experiment further with multi-viewer, multi-device settings in-the-wild, and investigate the potential benefits of speed-based adaptation of PDS.

10 REFERENCES

[BBKP07] Nadia Bianchi-Berthouze, Whan Kim, and Darshak Patel. Does Body Movement Engage You More in Digital Game Play? and Why? *Affective computing and intelligent interaction*, pages 102–113, 2007. DOI: 10.1007/978-3-540-74889-$2_1$0.

[BMSN11] M. Beneling, F. Matulic, L. Streit, and M. C. Norrie. Adaptive Layout Template for Effective Web Content Presentation in Large-Screen Contexts. *Proc. 11th ACM Symposium on Document Engineering (DocEng)*, 2011. DOI: 10.1145/2034691.2034737.

[BN13] M. Beneling and Moira C. Norrie. Responsive design and development: Methods, technologies and current issues. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7977 LNCS:510–513, 2013. DOI: 10.1007/978-3-642-39200-$9_4$7.

[BRHH11] Yuliya Bababekova, Mark Rosenfield, Jennifer E Hue, and Rae R Huang. Font Size and Viewing Distance of Handheld Smart Phones. *Optometry & Vision Science*, 88(7), 2011. DOI: 10.1097/OPX.0b013e3182198792.

[DHKQ14] Jakub Dostal, Uta Hinrichs, Per Ola Kristensson, and Aaron Quigley. SpiderEyes: Designing Attention-And Proximity-Aware Collaborative Interfaces for Wall-Sized Displays. In *Proc. 19th International Conference on Intelligent User Interfaces (IUI)*, 2014. DOI: 10.1145/2557500.2557541.

[DKQ13] Jakub Dostal, Per Ola Kristensson, and Aaron Quigley. Multi-View Proxemics: Distance and Position Sensitive Interaction. In *Proc. 2nd ACM International Symposium on Pervasive Displays (PerDis)*, 2013. DOI: 10.1145/2491568.2491570.

[ESNL15] Ivan Elhart, Federico Scacchi, Evangelos Niforatos, and Marc Langheinrich. ShadowTouch: A Multi-user Application Selection Interface for Interactive Public Displays. In *Proc. 4th International Symposium on Pervasive Displays (PerDis)*. ACM, 2015. DOI: 10.1145/2757710.2757735.

[Fra15] Ben Frain. *Responsive Web Design with HTML5 and CSS3*. Packt Publishing Ltd, 2015.

[Gar11] Brett S Gardner. The Spark of Innovation Begins with Collaboration. *Inside the Digital Ecosystem*, 11(1), 2011.

[LLCB08] Siân E Lindley, James Le Couteur, and Nadia L Berthouze. Stirring up Experience through Movement in Game Play: Effects on Engagement and Social Behaviour. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 511–514. ACM, 2008. DOI: 10.1145/1357054.1357136.

[Mar13] Ethan Marcotte. *Responsive Web Design*. Number 4 in . Editions Eyrolles, 2013.

[MWE+09] Jörg Müller, Dennis Wilmsmann, Juliane Exeler, Markus Buzeck, Albrecht Schmidt, Tim Jay, and Antonio Krüger.

Display Blindness: The Effect of Expectations on Attention towards Digital Signage. In *International Conference on Pervasive Computing*, pages 1–8. Springer, 2009. DOI: 10.1007/978-3-642-01516-8$_1$.

[NEL15] Evangelos Niforatos, Ivan Elhart, and Marc Langheinrich. Public Displays for Monitoring and Improving Community Wellbeing. In *Proc. ACM Intl Joint Conference on Pervasive and Ubiquitous Computing and Proc. ACM International Symposium on Wearable Computers (UbiComp/ISWC)*. ACM, 2015. DOI: 10.1145/2800835.2807954.

[NEL16] Evangelos Niforatos, Ivan Elhart, and Marc Langheinrich. Weatherusi: User-based weather crowdsourcing on public displays. In *International Conference on Web Engineering*, pages 567–570. Springer, 2016. DOI: 10.1007/978-3-319-38791-8$_5$0.

[Pet14] Clarissa Peterson. *Learning Responsive Web Design: A Beginner's Guide*. " O'Reilly Media, Inc.", 2014.

[RLC+06] Enrico Rukzio, Karin Leichtenstern, Vic Callaghan, Paul Holleis, Albrecht Schmidt, and Jeannette Chin. An Experimental Comparison of Physical Mobile Interaction Techniques: Touching, Pointing and Scanning. In *International Conference on Ubiquitous Computing (UbiComp)*, pages 87–104. Springer, 2006.

[STBT18] Amir E. Sarabadani Tafreshi, Milan Bombsch, and Gerhard Tröster. Chained Displays: Configuration of Multiple Co-Located Public Display. *International Journal of Computer Networks & Communications (IJCNC)*, 10(3):27–44, 2018. DOI: 10.5121/ijcnc.2018.10303.

[STMT18] Amir E. Sarabadani Tafreshi, Kim Marbach, and Gerhard Tröster. Proximity-Based Adaptation of Content to Groups of Viewers of Public Displays. In *International Journal of Ubiquitous Computing (IJU)*, 2018. DOI: 10.5121/iju.2018.9101.

[STN17] Amir E. Sarabadani Tafreshi and Moira C Norrie. ScreenPress: A Powerful and Flexible Platform for Networked Pervasive Display Systems. In *Proceedings of the 6th ACM International Symposium on Pervasive Displays*, page 13. ACM, 2017. DOI: 10.1145/3078810.3078813.

[STST18] Amir E. Sarabadani Tafreshi, Andrea Soro, and Gerhard Tröster. Automatic, Gestural, Voice, Positional, or Cross-Device Interaction? Comparing Interaction Methods to Indicate Topics of Interest to Public Displays. In *Frontiers in ICT*. Frontiers, 2018.

[STSTST17] Amir E. Sarabadani Tafreshi, Sara C. Sarabadani Tafreshi, and Amirehsan Sarabadani Tafreshi. TiltPass: Using Device Tilts As an Authentication Method. In *Proceedings of the 2017 ACM International Conference on Interactive Surfaces and Spaces (ISS)*, pages 378–383. ACM, 2017. DOI: 10.1145/3132272.3134112.

[TJS15] Constantin Taivan, Rui José, and Bruno Silva. Web-Based Applications for Open Display Networks: Developers' Perspective. *International journal of computer systems science and engineering*, 30(1):21–30, 2015.

[TMN17] Amir E Sarabadani Tafreshi, Kim Marbach, and Moira C Norrie. Proximity-Based Adaptation of Web Content on Public Displays. In *International Conference on Web Engineering (ICWE)*, pages 282–301. Springer, 2017. DOI: 10.1007/978-3-319-60131-1$_1$6.

[Wic17] Adrian Wicki. Investigating Content Adaptation for Sequential Content Configuration of Chained Displays, 2017.

 ISBN 978-80-86943-41-1