Mobile Implementation of a Web 3D Carousel with Touch Input

Christoffer Björkskog, Giulio Jacucci, Bruno Lorentin
Helsinki Institute for Information Technology HIIT, Helsinki University for Technology TKK
P.O. Box 9800, FIN-02015 TKK, Finland
+358 9 4511
firstname.lastname@hiit.fi

Luciano Gamberini
HTLab – Dep. Of General Psychology, University of Padova
Via Venezia, 8
35131 Padova ITALY
+ (39) 049 8277425
luciano.gamberini@unipd.it

ABSTRACT

Mobile devices such as the iPhone provide state of the art interaction capabilities also for web browser applications. Our mobile development is targeted to a Energy Awareness application that provides playful access to detailed and realtime information on energy consumption of appliances of a household. Using the available Safari Browser that adopts W3C web standards we demonstrate the implementation of a 3D carousel giving access to cards on a web page where each card gives access to information on one appliance. The carousel can be browsed using the multitouch capability of the iPhone. We describe the programming approach and discuss the lesson learned in developing the touch interaction with the carousel.

Categories and Subject Descriptors

H.5.2 [Information Interfaces and Presentation]: User Interfaces – Input devices and strategies, Interaction styles.

General Terms

Algorithms, Documentation, Design, Experimentation, Standardization, Languages.

Keywords

HCI, Mobile device, iPhone, MultiTouch interaction, 3D interface, carousel, W3C, Javascript, CSS3, XHTML, WebKit

1. INTRODUCTION

Mobile devices are increasing powerful in being able to provide state of the art interaction modalities and media processing. For example mobile phones have evolved to the extent that manufacturers do not call them phones anymore. Features include cameras with ever increasing resolution for picture and video capture, internet connectivity (based on operators network or WLAN), near field communication as RFID (Radio Frequency Identification), a variety of sensors like accelerators, GPS (Global Positioning System), and through the bluetooth interface virtually any sensor can be used in applications furthermore the bluetooth itself being a sensor that is able to scan and sense the presence of other devices.

Recently devices like the iPhone also provide intuitive user interaction like multitouch making the use of applications like maps and web browsing much more compelling.

Until recently mobile applications have been developed mostly through with "native" languages (for example C++ for Symbian), or other higher level languages like Java as these were the only way to be able to do graphic and image processing or access resources on the phone.

Now mobile devices are delivering browsers that not only give access to the interactive features of the phone (see Nokia Widget platform and iPhone) but that also provide state of the art support as browser as in the case of the Safari in the iPhone that includes CSS3 with 3D support not yet included as a standard.

Web application development has a variety of advantages and in this paper we show how to implement an advanced interface using the standard web browser Safari on the iPhone.

Our development aimed at an Energy Awareness application that delivers information on the current consumptions of a variety of appliances in a household. We were interested in a platform that provided:

- Playfulness: the platform must allow an interactive and playful experience. It is really important to offer to user an engaging way to use the application.
- Portability: in order to be accessible to a large panel of user, the application has o be as portable as possible. The diversity of the existing mobile platforms makes this point difficult to reach.
- Was easy to code: we tried to find solution to avoid spending too much time in learning complex programming code in order to concentrate our efforts in building very high quality functionalities.
- Possibility to handle graphics and rich multimodal interaction.

We present a mobile web 3D carousel interface build on the iPhone using Web technologies such as XHTML, CSS and JavaScript. It also utilizes WebKit's implementations to CSS3 proposals that enable hardware accelerated 3D positioning, transformation and animation of elements. The interface takes advantage of the multitouch input method provided by the iPhone.

2. RELATED WORK

The idea to adopt a carousel as a closed-loop menu to select item is not new at all. The ones of us that were playing video games in

MobileHCI'09, September 15 - 18, 2009, Bonn, Germany.
ACM 978-1-60558-281-8.

the 80s on the Neo Geo arcade machine might remember that a lot of these games were using such a carousel to ask the best players to enter their initials and appear in the high scores screen. An example of more recent carousel can be seen in Microsoft's Encarta 2004. As Wang et al. [1] explained, the carousel design provides straightforward and great looking layout, its mechanism is easy to understand, the 3D visualization enables users to easily spot the selected item and the rotation effect is engaging.

Such models are now widely used in webpages in order to display fancy image galleries or menus. With recent browsers it is possible to use different tools and languages to implement them such as JavaScript[2], Adobe Flash[3] or Microsoft Silverlight[4]. But even if these solutions are great options to display our 3D carousel on desktop browser, it is not suitable when we use browsers on mobile devices due to technical restrictions. On the iPhone, using 100% "classic" JavaScript would be way too slow and Flash and Silverlight are not even supported by Safari for iPhone. In addition, these examples do not include any touch and movement recognition.

3D carousels have also been used on multi-touch installations. The Citywall[5] project used two 3D rings (one vertical and the other horizontal) to allow users to represent a "time travel" and to display pictures taken in the specified period of time. Fingertapps[6] provides a software platform for delivering commercial multi-touch solutions. They have implemented for Lexus a 3D carousel menu, very close to the one we did, to navigate between the different options to customize a virtual car. Once again the technologies used by these installations cannot be used on the iPhone.

Even if some applications on the iPhone also have components that can be described as carousel like the vertical rotating menu that is used to set up the alarm clock or the timer, they are part of native applications. As we did not want users to have to install anything on their mobile phone, we could not use this solution either.

3. IMPLEMENTATION

The Carousel is part of an Energy Awareness application that displays detailed power consumption for each appliance. Therefore each card in the carousel is representing an appliance or electrical device in the house. In addition the card can be clicked and turns to offer additional information and functionality for the given appliance.

The user interface and the 3D carousel component runs in the client browser powered by JavaScript.

Each card has a front side and a backside. The amount of cards created for the menu varies in this application based on data that is fetched from a server.

In the developed application, the cards represent electrical appliances in the household. When a card is tapped, it flips around and shows a menu for that device.

When the circle is initialized, each card is created and positioned standing in an elliptical circle level to the plane. They are distributed along the circumference of the circle with their front facing the user. Each card is a div element consisting of two child div elements. They are positioned in the same position, but one of them is rotated 180 degrees around the y-axis. This way each card has a backside and a frontside. Their container has the impression of being a two faced card.

3.1 Evolution of the carousel implementation

The carousel is elliptical in order to allow all cards to fit inside the screen. The first prototype was, however, a round circle with one sided cards facing outwards from the center (see figure 1). The rotation was smooth since there was only one transformation needed to rotate the circle. That was to rotate the element that contained all the cards. We realized that this approach had drawbacks. It did not match the design and the content of the cards on the edges were not visible.

We then implemented a prototype that had cards that faced the user all the time (see Figure 2). This approach, however, needed several transformations when users interacted with it. Each card needed to be moved into a new position in the x,z plane. A function was created to calculate a card position based on the current rotation of the circle. A major efficiency problem was discovered; On the iPhone simulator the flow and rotation of the circle was smooth. On the iPhone, however, it was slow. The problem was that for each pixel the finger moved over the interface, the cards positions were updated. When we set that the finger must move ten pixels before an update was triggered, the interface was responsive on the iPhone too, while still being smooth in its rotation.

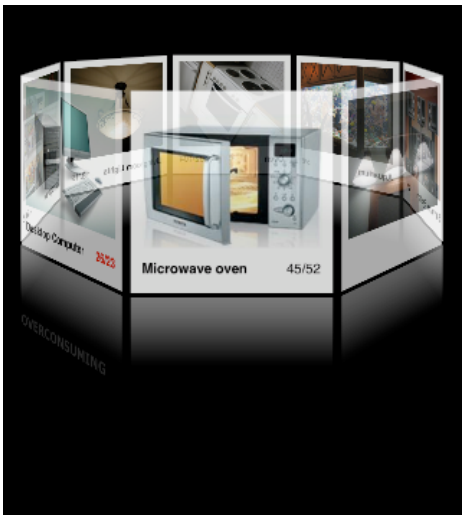

Figure 1. The first prototype with the cards facing outward inside a container that could be rotated.

3.2 Touch interaction with the Carousel

The first prototypes registered only the fingers movements along the x-axis. If you moved your finger to the left, the circle rotated clockwise, if the user moved the finger to the right, it rotated counter-clockwise. The cards appear larger the closer to the viewport they are, so most of the area on the screen consists of cards that are near. When the finger is above a card that is close, this interaction seemed intuitive. But then we noticed that users tried to keep their finger on a card and "follow it around". They tried to rotate the circle by performing circular movements with the finger on one card. The problem was, that when the card they had their finger on came to the backside of the circle, the finger would move in the opposite direction in order to continue in a

clockwise movement. When the user then changed the direction, the ring started to rotate the other way.

The first solution to this problem was to define a point along the y-axis (the top edge of the card closest to the user), that if the finger is under that and moves towards the left the circle rotates clockwise, and counter-clockwise if the finger is above that point.

It was now possible to move the finger in a circular motion and the carousel rotated in a similar fashion.

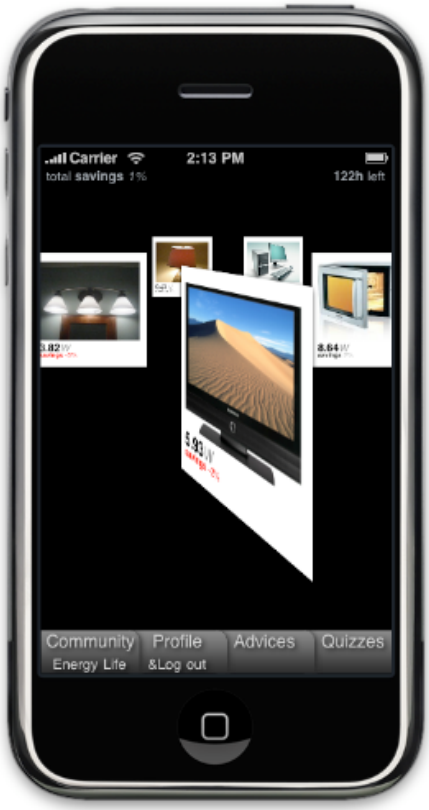

Figure 2. Later prototypes had cards facing the user where the positions of the cards change as the user rotates the carousel. When a card is tapped, it flips around and you can access more content on the backside.

The problem was now that when the finger was near the left and right edges, the rotation was not accurate in relation to the finger's movements. If the finger moved in a circular motion, it would travel along the y-axis near the left and right edges and not so much along the x-axis that was used to monitor the input. The result of this was that the circle rotated in a "jumpy" fashion, where the speed of the rotation would alternate between slow and fast.

In order to give the interaction a natural feel, we calculated what the fingers angle in relation to the center of the circle was and set the new angle of the circle in relation to that. We now took the position of the finger both along the x- and y-axis into consideration. This required us to have the movement threshold to be 10 px in either x or y direction.

The unsolved problem at the time of writing is to map the fingers position and rotate the circle so that the same card sticks to the finger at the same position during rotation. Now when the finger moves, the card you initially had your finger on is not fixed to your finger during rotation.

3.3 Web Standards

The application uses vendor implementations for certain W3C CSS3 suggestions [9]. These are transition-property, transition-duration, transform, transform-style, animation-name, animation-duration, animation-iteration-count, animation-timing-function, backface-visibility, perspective, perspective-origin and @-keyframes [x2, x4, x5]. In order for a suggestion to become a recommendation, at lest two browsers need to implement the features [12]. When browsers implement these rules, they prepend the names of the rules with a prefix. The rule transform, becomes -webkit-transform as a WebKit implementation.

In order to enable 3D positioning of HTML elements, their container needs to have the CSS3 attribute transform-style to preserve-3d instead of the default value flat [7]. If the transform style is flat, all children of the element are rendered on its surface. If the transform style is preserve-3d, it is possible to adjust its children's position in 3d space and rotate them in all three dimensions.

An elements backface is by default visible. If you rotate that element so that its backside is towards the viewer, the content of the element is visible but mirrored. If you set the backface-visibility CSS3 attribute to hidden, the element becomes invisible when looked from behind. A two faced card can be created with two elements that have their back face visibility hidden, and one is rotated 180 degrees around the y-axis but positioned at the same place. If their container is rotated, it gives the impression of rotating a card with two sides.

4. Conclusions

In our development for a energy awareness application we looked for a platform that provides playfulness, portability and rich interaction (modalities and graphics).

Playfulness. The iPhone represents a very innovative interface. The multi-touch screen, the accelerometer and all the others features offered by this platform reflect a lot of playfully interaction, new interface concepts and engaging way to handle it. In addition, Safari for iPhone supports CSS3, which allows using very smooth transition effect on webpage elements in order to have a pleasant and attractive interface. Thus, we could avoid having a too "common" webpage-like look for our application.

Portability. As our application will be web pages, it could be displayed on any browser. But we will have to implement slightly different version for the iPhone and for the others browsers (like desktop browser) as we will have to handle different viewport and as desktop browser don't support CSS3 yet. It will be possible by using conditional CSS. In addition all the iPhone-specific application implementation will be suitable for he latest model of iPod touch, which represents a large number of appliances.

Rich interaction through modalities and graphics. Using HTML Canvas, it becomes possible to handle graphics with only HTML and Javascript, in 2D or 3D. The Safari for iPhone's webkit also allows using the multi-touch screen events to interact with webpages.

The carousel is implemented using emerging web standards without any browser plugins. The interface serves as an example

of what will be possible to implement in future web browsers. It will be interesting to see what kind of web based plugin-free user interface components we will see implemented as the standards for CSS evolve and browsers follow the recommendations.

We aim at contributing to anticipate the deployment of rich multimodal applications on mobile devices that are built on web technologies. These provide a variety of advantages as they are fast to implement, are portable and are increasingly rich in interaction functionality.

On the other hand web applications do not give the same performance and quality of a native application and provide restricted functionality.

5. REFERENCES

[1] Shen, Y., Ong, S. K., and Nee, A. Y. 2008. Collaborative design in 3D space. In Proceedings of the 7th ACM SIGGRAPH international Conference on Virtual-Reality Continuum and Its Applications in industry (Singapore, December 08 - 09, 2008). VRCAI '08. ACM, New York, NY, 1-6. DOI= http://doi.acm.org/10.1145/1477862.1477900

[2] Carousel.us : a JavaScript 3D Carousel : http://www.piksite.com/carousel.us/carousel.us.php

[3] Carousel 3D Slideshow – demo 2, http://www.paulvanroekel.nl/picasa/carouseldemo2/

[4] VectorForm's Silverlight 2.0 Carousel, http://www.vectorform.com/silverlight/silverlight2/ImageCarousel/

[5] Peltonen, P., Kurvinen, E., Salovaara, A., Jacucci, G., Ilmonen, T., Evans, J., Oulasvirta, A., and Saarikko, P. 2008. It's Mine, Don't Touch!: interactions at a large multi-touch display in a city centre. In Proceeding of the Twenty-Sixth Annual SIGCHI Conference on Human Factors in Computing Systems (Florence, Italy, April 05 - 10, 2008). CHI '08. ACM, New York, NY, 1285-1294. DOI= http://doi.acm.org/10.1145/1357054.1357255

[6] FingerTapps, Interactive Show Room Display – Lexus. (http://www.fingertapps.com/default.aspx?pagename=Showcase), 2009

[7] Apple Inc., 2009. Safari CSS Reference.

[8] W3C, CSS Transition Module Level 3 (http://dev.w3.org/csswg/css3-transitions/), 2004

[9] W3C, CSS3 mudle: Synax. (http://www.w3.org/TR/css3-syntax/#vendor-specific), 2009

[10] W3C, CSS 3D Transforms Module Level 3. (http://dev.w3.org/csswg/css3-3d-transforms/), 2009

[11] W3C, CSS Animations Module Level 3. (http://dev.w3.org/csswg/css3-animations/), 2009

[12] CSS3.info, Vendor specific extensions to CSS3. (http://www.css3.info/vendor-specific-extensions-to-css3/), 2007