Lance Castillo

AIP 197PR: PRIME Program

August 28, 2011

Final Report

For PRIME 2011 my designated project was to create a multitouch program that allowed users to create their own galleries in a fun and interactive way. The project was a collaboration between PRIME from UCSD, The Museum of Photographic Arts (MOPA) from San Diego, and The National Institution of Information and Communication Technology (NICT) from Japan. NICT showcased the project in an exhibition called Knowledge Capital 2011 held in Osaka, Japan and MOPA plans to use the project in an upcoming arts festival called Tokyo Photo 2011. In order to complete the project, I had to research the available touch software, create a demo application, and finally write the actual code for the program. There were a few challenges when developing the application, but they were not impossible to overcome.

I started the project by researching the different options for a platform to develop the multitouch application on. Upon searching different options on the NUI Group's wiki page (Natural User Interface Group), I decided to use the Open Exhibits multitouch framework (Ideum) which was recommended to me by my UCSD mentor, Dr. Jason Haga. The framework was developed using a language called ActionScript and used Adobe Flash, which allowed for more creativity with the user interface. It was also very easy to learn and use, so I decided that it would be the best option for the project. Another option would have been to use another programming language called C++, but the multitouch framework seemed overly complicated and had a very steep learning curve. Also, it wasn’t entirely clear whether or not the multitouch table the software would be running on would accept the C++ framework. With Open Exhibits,

my partner Wesley Hsu and I were guaranteed that the software would work on the table since we knew that the table would output FLOSC touch data which is what the software takes as input.

Once the software was chosen, I created a demo application to show that it was a viable option that would reach the project's goals. The demo application was fairly simple. It was a virtual pinwheel that rotated when you dragged its edge. The great thing about Open Exhibits was that it came with a very nice touch simulation system that allowed you to use your computer to simulate touches and multitouch gestures. Once my UCSD mentor saw the demo, he also agreed that it was the best choice for the project. I also showed the demo to my Japanese mentor, Shinji Shimojo and he really enjoyed it. After that, the preliminary steps to creating the application were finished and we were free to start designing the application.

MOPA supplied us with a list of what they wanted the application to do, and we used that list to create it. The application's purpose was to allow the user to create their own mini galleries of ten images or less from a collection of fifty images that MOPA supplied. The application had to save those mini galleries into memory so that users would be able to view other saved galleries. The mini galleries, once created, had to be given a title along with the name of the creator before being submitted and saved. In order to help the user create a gallery, the images had to be displayed to the user along with the information about them. The images also had to be resizable so that the user could view them more closely. Those were the required abilities that the program had to have. If there was time, the museum also asked us to try and use NICT's Tile Wall Display to show saved mini galleries. After we were clear on the design, we got to work on the application.

The development of the application started with the main screen. Wesley designed the user interface, and I started working on the core features of the program. The collection of fifty

images had to shown to the user in an efficient manner to save screen space, so I implemented a scroll area for them. The scroll area was a bit challenging at first since I had to build it from scratch. After a few attempts, I came up with an algorithm for moving the images in two columns with a drag gesture. I also had to implement a way for the user to select an image for viewing purposes, so I made it possible for the user to touch and hold the image to select it. Once the image was selected it would inflate its size so that the user would know that they selected an image. After that I made it possible for the user to drag the image over into the viewing area. Later in the internship, some user tests were done by MOPA and we were able to determine that the touch and hold procedure was not user friendly, so I changed it to touch and drag. User tests confirmed that touching and dragging the images was more natural and easy to do. The user tests also showed us that the scroll area needed an indication of where its position was, so I added a scroll indicator much like the one you see to the right of an internet page you view in your browser. Once an image was dragged into the viewing area, it was converted into a Photo Display object that the user could interact with.

Creating the Photo Display object was another challenging part of the project. I had some trouble with the user interface. For some reason, ActionScript did not allow interface objects to have negative coordinates. Once I figured that out, it wasn't too hard implementing the Photo Display object. I added the ability to resize the photo and I displayed the metadata about the photo to the user. There were three buttons around the frame of the photo. The first allowed the user to close the photo and send it back to the scroll area, the second allowed the user to add the photo to their mini gallery, and the third allowed the user to hide the information about the photo so that they could view it better. MOPA requested that the photo not overlap the scroll area or the mini gallery area, so I set up some boundaries for both of those areas.

While I was working on the scroll area and the photo viewing area, Wesley was working on the mini gallery area. The mini gallery area allowed users to choose ten of their favorite photos. We made it possible for images from the scroll area or the viewing area to be dragged into the mini gallery area. Wesley added the ability to delete or drag photos out of the mini gallery area if the user decided that they didn't want a certain image anymore. Once the user had chosen two to ten images, then they could move to the edit mini gallery screen by pressing the edit button.

The edit mini gallery screen was where the user could organize and save their collection of photos. In order to title a mini gallery and leave a name, Wesley designed a keyboard that had Japanese and English keys. I implemented the keyboard functionality along with the functionality behind saving a mini gallery. I tried using ActionScript to save the information from the mini galleries to an XML file, but the language did not support it. The challenge was to find a suitable alternative for saving mini galleries. I came up with a solution by using a local Apache server with PHP on it. I sent the information from the mini galleries to the local server via an HTTP GET request. The PHP server took that data and saved it to an XML file. The PHP server also sent the data to the Tile Wall Display so that users could view their mini galleries on it.

The last screen to be implemented was the other galleries screen. On this screen users were able to view all of the saved galleries that past users had created. I made it possible for users to sort the saved galleries by title, name, and time created. If a user were to click on a saved gallery, then they would be taken to another screen where they could view each image in it. I also implemented a way for the user to view that gallery on the Tile Wall Display if they wanted to see the images more closely. One challenge that I experienced with the other galleries screen

was sending Japanese characters to the Tile Wall Display. For some reason the Japanese characters were showing up garbled on the display. I found the source of the problem on the local server. I understood that the information had to be encoded in UTF-8 format in order for the table to be able to read the characters. For that reason, I encoded the title and name of the mini gallery in UTF-8 before sending the data. What I didn't know was that the data was already being automatically encoded to UTF-8 by the ActionScript side of the application. The double encoding of the information was garbling the Japanese characters, so I removed the extra encoding and the problem was solved.

In conclusion, the application turned out to be a success. It was taken well by the participants of the exhibition in Osaka. This gives me great hope for its use in Tokyo Photo 2011. Overall, the development process went smoothly and we were able to add the important features that MOPA requested. Open Exhibits also turned out to be a great choice. The ease of use and simplicity of the framework made the entire process of developing a multitouch application much faster. There were a few confusing things about the framework, but after playing with it a little I was able to figure it out. The application was a fun challenge, especially considering that I was in an exotic location surrounded by new and exciting things.

Bibliography

Ideum. Open Exhibits [Computer Software]. Available from http://openexhibits.org/

Natural User Interface Group. (2011, July 17). *Applications and libraries*. Retrieved from http://wiki.nuigroup.com/Applications_and_libraries