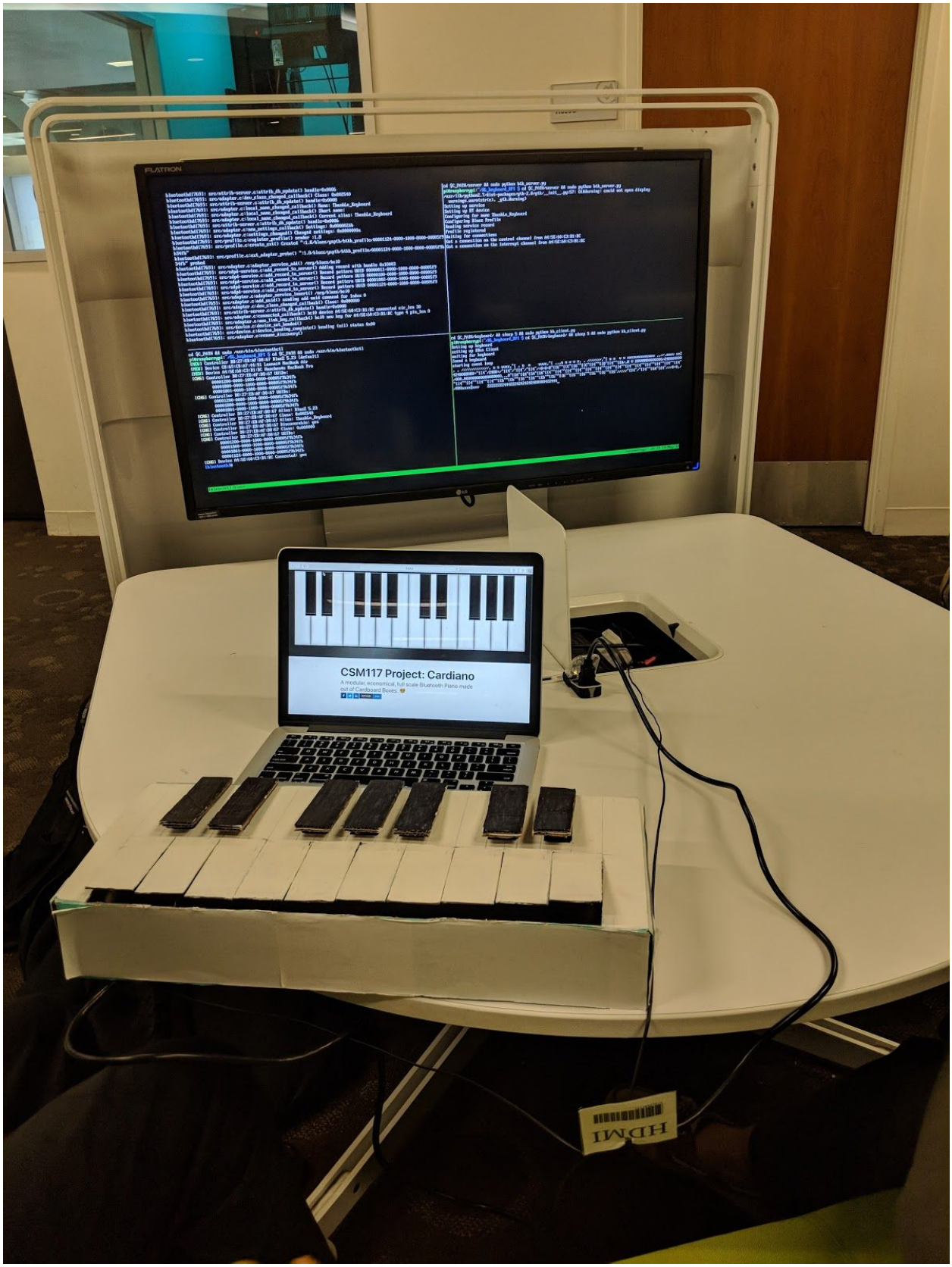
Spring 2018 CS M117

Team Name	Virtuoso
Project Name	Cardiano
Member Name:	**UID:**
Duzhi Chen	004773782
Haochen Li	204739914
Yansong Huang	404956794
Ziheng Xu	704756821
Hongjie Zhang	104778185
Jiashu Zhou	804663317


CSM117 Project: Cardiano
A modular, economical, full scale Bluetooth Piano made out of Cardboard Boxes.
HDMI

Motivation

Piano is an expensive and luxurious instrument for most people to play. Therefore, piano apps from phone or computer could satisfy people's curiosity. However, it is suffering for people who try to play with piano app with their limited screen. Therefore, we propose to design a self-made piano keyboard cooperated with the bluetooth devices that allows people to have a real piano experience. On the other hand, we would like to create a prototype of cardboard made piano with the most cost-effective design and budget.

Technical Design & Implementation

● Code Implementation - GitHub Link

https://github.com/LawrenceXu13467/Cardiano

● Hardware Components

At hardware level, our product consists of a Raspberry Pi 3 with bluetooth Module, an HDMI cable, and a cardboard made piano case. The handmade piano is made up of 10 white keys and 7 black keys. At system level, we utilize a standard computer keyboard, an HDMI-input monitor for Raspberry Pi interface and an internet device to run a web application.

● Bluetooth Connection via Raspberry Pi

After we updated the system to the latest version, we downloaded Bluez5 tools and corresponding firmwares. And then we created 4 separate windows. One of them is used to enable and run Bluetooth in the foreground. Next step is to create a server, which serves as the bluetooth piano emulator. Then we created a btk server client application that sends local input to the server. After these two steps, we also configured a DBUS system bus for later use. In the third window, we used ‘hciconfig’ command to find available connections for all bluetooth devices within a certain range, and we pair our target device. Then we open a fourth window to run the python script which creates an event loop listening for input. All of these steps were later transformed and integrated into a single shell script, which is run after we boot the Raspberry Pi.

● Cardboard

Our original design was to make similar size and aspect as the real piano. However, we encountered the problem to find some large material to make the cover. Since our original idea is to build affordable piano, we accept the cardboard instead which is more cheaper and

recycle-able than other materials. We also get rid of the original size because it requires too many handwork and is not portable to demonstration in the class. After researching and analyzing on the proper size, our team realize that we can take example of popular Nintendo switch LABO kit which use the cardboard box to implement on game console to achieve the same piano goal. Nevertheless, it is way too expensive and doesn't obey our design goal. But if we use keyboard as input instead, the price will decrease significantly. Even though we decide to use cardboard as material, we still compare and investigate several brand to make sure it can support the pressure from user to get the balance between price and product service time.

Since we use keyboard as the input, we are restricted to the keyboard size and dimension of cardboard keystroke and only design restricted number of music scales. Even it doesn't has full music scale, our design still is good enough of the normal use. We have the 10 white and 5 black keys. We map each key to different key position on the keyboard such as left shift, x ,v , n for white, and 2 ,4 8 for black. Thus, we find the corresponding keystroke event for each individual one for the coding the and bluetooth transmission part.

● Software Components

○ Website Design

At software level, our project used the CSS and Javascript to implement our web page for the piano emulator. We used some CSS built in API for our frontend: such as Font Magician to get web front, Rucksack for animation, Autoprefixer to make easy use of browser prefix. We based on a piano emulator implementation from the internet and we learned from it for our own design. The design is included mapping the piano keys onto the corresponding appropriate keys on the keyboard. We also Bind with W3C Touch Events API to complete the I/O request.

○ Keyboard Event Loop Implementation

For the keyboard connection part, the script of how to implement the actual connection is written in Python. The basic procedure for our code is included Define a bluez 5 profile object for our piano at first. Then Initialize the bluz profile to advertise device capabilities from a loaded server record. After that Define a D-Bus service that emulates a bluetooth keyboard. In order to make transportation of data between the device through bluetooth faster, we decided to implement the event driven algorithm. This is accomplished through Create a event loop that keep for looking for input information. When received, forward those input information to Dbus device. The signal is converting through a keymap from the bluetooth keyboard. Finally the input info is then forwarded from Dbus to the client.

Demonstration Video URL

https://youtu.be/PWZmeqmuyn4

Challenges

Our cardiano project is challenging in many ways. First of all, the microcontroller is used to both receive wireless data from an input device, process the data, and send to the corresponding output device. In this case, the bluetooth module operates as both a master and a slave, and the procedure is more complicated than the ordinary operation of a microcontroller. On the other hand, the underlying difficulty can be found in setting up bluetoothctl pairing and connectivity by emulating an input device, such as unrecognized hardware errors. Also, we did not notice that the default BlueZ setting is incompatible with the version of our Raspberry Pi. We spent quite a lot of time debugging these related problems caused by default API. Moreover, team members have no previous experience on JavaScript, and thus, we have spent time on understanding tutorials and resources.

Future Work

As our objective presents, our group is trying to use inexpensive materials and the Bluetooth wireless communication to provide the users an emulated full-sized and hands-on piano experience. However, after the design, we as a group definitely find some areas that we can improve to make it a better self-contained device and offer better user experience.

- ## Integration

In terms of integration, currently we are using Raspberry Pi as our microcontroller to serve as a bluetooth wireless connection point to transfer signal and data to an endpoint personal computer that can output the audio. Even though Raspberry Pi is a good starting point to make a prototype and prove the functionalities of the concept in our mind, it is still a full-sized computer that has numerous unnecessary modules or peripherals. Therefore, instead of having a Raspberry Pi, we can replace it with a more compact microcontroller such as a stm32. Furthermore, as everyone can see during the demonstration, the system still needs a power cord to provide power to the Raspberry Pi, which is somewhat cumbersome. If the original Raspberry Pi is replaced with other compact size microcontroller as mentioned, we can develop a battery powered system instead of using power cord, which will make the whole system more portable. Additionally, we can emulate the full-fledged wired computer keyboard using switches and breadboard. Overall, we can make many modifications to make it more self-contained and efficient.

- ## Accessibility

In terms of accessibility, we can break it down into two categories. Firstly, accessibility in terms of cost. As we mentioned before, one of our motivation for this project is to provide a similar full-sized piano experience without paying a high cost for an actual full sized piano. With our current design, we are just using ordinary computer keyboard (either wired or wireless), a Raspberry Pi (that costs $20-$40), some cardboards, and paints (if necessary). All these materials mentioned are relatively accessible, and with these, the users would be able to have a relatively practical piano experience. Secondly, accessibility in terms of the situation that the users can actually build it and modify the details by themselves. With all the materials we mentioned above, the users can in fact follow the implementation tutorial from our GitHub link to build up their own piano step by step. Moreover, by understanding the tutorial we provide, the users can also make necessary changes to cater their personal needs such as more white and black keys, location of the keys and so forth.

- ## Ease to Use

Currently, the process to use our piano is to plug in the system, run the script to pair the Bluetooth device which audio is played on, and play the piano. The improvement we can make is to embed the code/script to the microcontroller to make it solely running this particular piano program, so that the users do not need to manually run the script themselves.

Conclusion

In conclusions, our group has completed the tasks stated in our proposal. In a high level concept, we are trying to achieve two goals. One is to make any electronic device to connected to a personal computer via Bluetooth. It does not limit to a keyboard in this particular case, and it can be majority of the electronic devices in the market. If we understand the Bluetooth wireless communication, by utilizing a microcontroller such as Raspberry Pi, Arduino or other processors as a Bluetooth communication node, we can turn speakers, computer mouse, headphones and other possible devices wireless. The other goal is to provide the users a full-sized piano experience with inexpensive materials. In the project, the keyboard is served as a functional foundation with an outer case that emulates the appearance of a piano keyboard with white and black keys. Then the Raspberry Pi is used as a Bluetooth communication node to the personal computer, sending keystroke signals to the personal computer, and the web application outputs the audio accordingly. As shown in the demonstration video, our group has successfully completed these functionalities.

References

"Emulating a Bluetooth Keyboard with a Raspberry Pi and Python (Raspbian Jessie/Bluez 5 version)", *Yetanotherpointlesstechblog.blogspot.com*, 2018. [Online]. Available: http://yetanotherpointlesstechblog.blogspot.com/2016/04/emulating-bluetooth-keyboard-with.html?m=1&from=groupmessage&isappinstalled=0. [Accessed: 08- Jun- 2018].

"Minor Update To Bluetooth Keyboard Client Code", *Yetanotherpointlesstechblog.blogspot.com.au*, 2018. [Online]. Available: http://yetanotherpointlesstechblog.blogspot.com.au/2017/08/updated-bluetooth-keyboard-client-code.html. [Accessed: 08- Jun- 2018].

"LFeh/piano", *GitHub*, 2018. [Online]. Available: https://github.com/LFeh/piano. [Accessed: 08- Jun- 2018].

G. Thomas and G. Thomas, "Emulate a Bluetooth keyboard with the Raspberry Pi", *SciFiNow*, 2018. [Online]. Available: https://www.gadgetdaily.xyz/emulate-a-bluetooth-keyboard-with-the-raspberry-pi/. [Accessed: 08- Jun- 2018].

"This DIY cardboard piano syncs with your favorite MIDI music apps", *Engadget*, 2018. [Online]. Available: https://www.engadget.com/2018/02/07/diy-cardboard-piano-midi-kami-oto/. [Accessed: 08- Jun- 2018].

Contribution

Duzhi Chen: Designed physical structure and manufactured prototype of cardboard piano.

Yansong Huang: Designed physical structure and manufactured prototype of cardboard piano.

Haochen Li: Studied and modified portable piano software using CSS and HTML.

Ziheng Xu: Implement the Bluetooth Emulator on the Raspberry Pi side and Wrote shell scripts to automate the process of transferring the input to raspberry pi emulator.

Hongjie Zhang: Studied references to establish Bluetooth connection from Raspberry Pi to conventional personal computer (PC), and transferring keyboard keystroke signals to PC display, realizing a wireless keyboard. Assisted Haochen on web application development.

Jiashu Zhou: Tested BlueZ implementations and debugged bluetoothctl pairing and connection under linux environment.