Halo mirror

Mirror with embedded display to provide futuristic convenience to users.

Pitom Saha
Computer Science
SUNY Binghamton
Binghamton, NY USA
psaha5@binghamton.edu

Ozden Ertan
Computer Science
SUNY Binghamton
Binghamton, NY USA
oertan1@binghamton.edu

ABSTRACT

IoT devices have gone from commercial use to being placed in people's bedrooms. Smart speakers, TV's, thermostats, and a suite of other internet connected devices allow control of devices and group device policy like never before. The halo mirror embeds modern features like commute time prediction, facial recognition, Google calendar, voice assistant, and music streaming apis to create a subtle but helpful mirror experience.

KEYWORDS

Internet of Things, Smart Mirror, Face Recognition, Voice Recognition, Touch Control

1 Introduction

The halo mirror aims to provide that "futuristic" feel of internet connectivity. While disabled the device is just a mirror, but when a recognized individual approaches, a whole suite of helpful data appears. The user will see their tasks for today, along with news headlines, weather, commute time, and a music player. A whole new realm of features also become available by an integrated voice assistant, Amazon Alexa, that has been customized to respond to "halo" instead.

2 Design

There are many pieces in creating a subtle interactive experience with a mirror. On the software side, UI/UX is a large piece of the puzzle due to the nature of a mirror having limited display space for information. Additionally, the backend must work properly with failsafe's in place. If the software is to break down, it must do so in a way that the display shows minimal errors. On the hardware side, we must ensure 1) structural integrity, 2) it looks like a mirror (clear reflection), sounds like a mirror (minimal sound from working components), and finally 3) balance cost and availability of parts with design. Our design has changed several times through this semester. With COVID-19, some parts could not be sourced, which required us to do some extensive labor with the wrong parts.

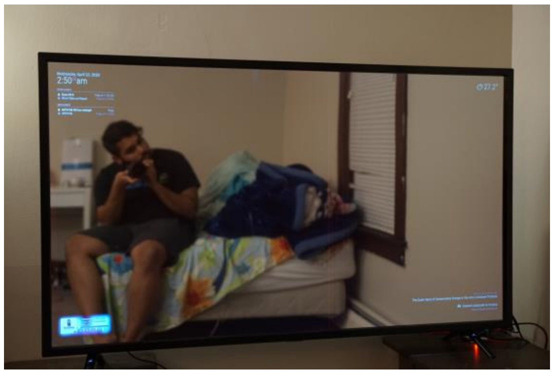

Figure 1: The complete halo mirror and UI pictured above.

2.1 Hardware

A complete display behind the mirror was decided to perfect the mirror display quality as opposed to a partial display on one side of the mirror. Any imbalance of light behind the mirror will clearly **look like a screen behind glass rather than a conventional mirror**. The major components are as follows:

1. Raspberry Pi 4 Overclocked
2. 55 inch Display
3. Two Way Mirror Foil
4. Power/ Display Cables
5. Infrared Touch Frame
6. PSEye Camera

2.1.1 Raspberry Pi 4 Given that there are several features that require offline computation, and the display supports a higher resolution, a Raspberry Pi 4 is a good choice for the processing unit. We found that some features still have visual lag attributed to the computational limitation of the Pi and decided to overclock it as well for an approximate 26% performance boost.

2.1.2 55 inch Display The display is large enough for a mirror. In typical large mirror constructions, builders use a small display to cover costs, but end up with a product that just looks like a screen behind reflective glass. It is explicitly a VA panel as opposed to IPS or TN because of their ability to reproduce dark blacks. The darker the screen behind the foil, the more unadulterated the reflection will be.

2.1.3 Two Way Mirror Foil The A partially translucent and partially reflective foil is cheaper, but typically takes many applications to get a tint applied correctly. The halo mirror cuts cost with this part choice but in turn the creation process become much more exhaustive.

2.1.4 Power/Display Cables The Pi and all other electronic are wired to a surge protector so only one cable comes out of the mirror. A OnePlus DashCharger is used as the power supply.

2.1.5 IR Touch Frame The black frame around the mirror is black stainless steel. There are hidden IR lights that run across the length and width of the frame that enable touch detection. When light is interrupted from reaching the other side a touch is detected. The original display TV frame was dismantled and the touch frame has been hot glued into place.

2.1.6 PSEye Camera This camera was created for motion control video games on Playstations. Sony ensured that it would work in low-light scenarios where only a TV provides illumination. They did not catch on and now sell for $6 USD on Amazon. For facial recognition, this is an optimal, budget-friendly solution.

2.2 Assembly

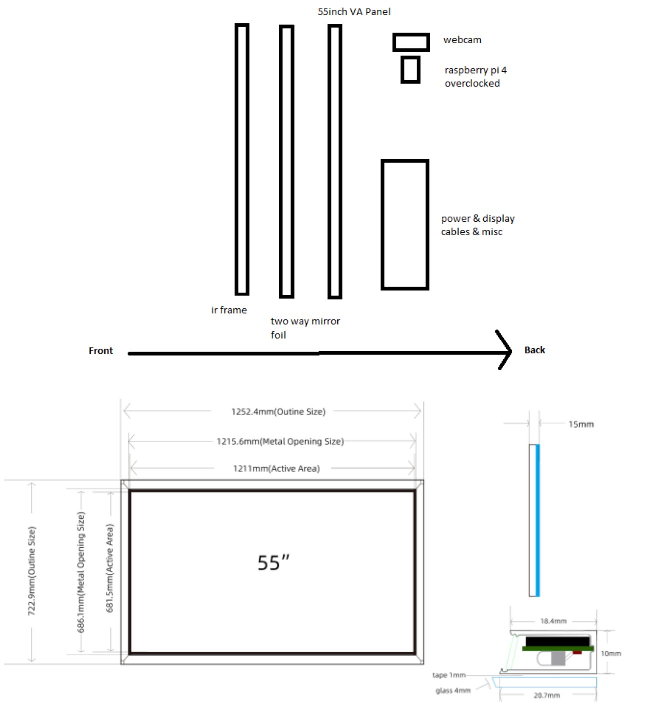

Figure 2: The assembled hardware design from side view (upper) and front view (lower).

Because of COVID-19, glass could not be sourced to place in front of the display. Instead the tint was applied directly to the display screen. Getting a proper tint took 6 tries and 12 hours of labor. From front to back, the IR Frame is attached to the display, with electronics hidden behind it. The display panel is hot glued between the IR frame and TV housing.

2.3 Software

There are many ways to create a interactive UI to display behind mirror glass. Halo mirror is built using magic mirror 2, one of the mostly widely used open-source programs. It essentially runs a black webpage in electron with modules that users can customize in. Halo mirror uses several third party modules to integrate features like Voice control, Commute time, Calendar and music but has significant modifications to stylistically fit and function as one cohesive package. The major software utilized is listed below:

1. Raspbian Buster Linux OS
2. MagicMiror^2
3. OpenCV
4. Amazon Voice Services
5. Spotify Developer API
6. Google Calendar API
7. Google Directions API
8. OpenWeatherMap API
9. RokuTV External Control Protocol API
10. Alsa Audio System
11. Raspotify

This list is not exhaustive. As the project has taken months of work on the software side, there are a myriad of utilities installed to get low level details working. The high level data flow is as shown:

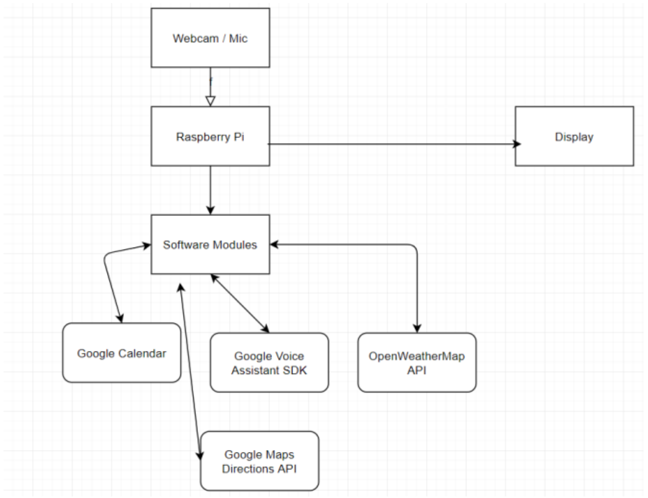

Figure 3: Data flow of the halo mirror.

3 Implementation

3.1 Software

MagicMirror and node dependencies are first installed. All sensing hardware is connected and several configuration steps are necessary. Using a webcam as a mic but not as a playback device takes some effort. The Touch IR Frame is a plug and play device but requires calibration. Since a foreign vendor was used, the software was in a foreign language and had little support. Then installation of specific modules was necessary. Each module is a unique service. Each one requires an API key to access data. The following open-source modules have been used. Their links can be accessed for specific dependency and installation instructions.

1. MMM-Spotify by Eouia [1]
2. MMM-Traffic by MichMich [2].
3. MMM-Face-Reco-DNN by Nischi [3]
4. CurrentWeather by MichMich [4]
5. Newsfeed by MichMich [5]

Typically, most modules would only need dependencies, an API key and simple Javascript modifications. However, the facial recognition component took a significant amount of modification. The halo mirror supports multiple user profiles and the existing login system did not correctly function. On top of adding support for multiple users, we integrated wireless control of the display using Roku's External Control Policy API. Now when a face is recognized, a specific user's layout is displayed. When they leave the display shuts off instead of just hiding all the modules. This (1) saves a significant amount of power and (2) prevents the display from illuminating when not in use. Typically smart mirrors try to resolve this issue with additional hardware sensors, or they just display an all black screen. Either solution requires more budget or continuous power usage with decreased mirror visibility.

3.2 Hardware

A complete disassembly of the TV allowed direct access to the display panel. Applying mirror foil on a thin flexible 55 inch panel is a long tedious process, that must be repeated with new foil if a single mistake is made. Because we could not source glass, we tinted directly onto the display, spraying soapy water liberally over the display and then squeegeeing out bubbles from underneath. It is an extremely difficult, and risky task. Large broad strokes are required while ensuring precision that the water does not seep into electronic underneath. Afterwards the IR frame is hot glued to the TV housing, sandwiching the display panel. Then on top the PS eye camera is hot glued to the frame. Finally the pi, power strip and wiring are strapped down and glued in the back, so they are invisible when the mirror is mounted to a wall.

.

4 Evaluation

We successfully delivered all of our planned features, even while parts sourcing became difficult. In exchange for parts, we put in labor and creativity. We have achieved 1) Accurate facial detection for multiple users, along with custom display profiles, 2) interactive display with touch, 3) Host of convenience features, 4) Robust services that handle failure without visual elements. The overall experience of the mirror is genuinely fascinating and we are extremely satisfied with the product we have created.

ACKNOWLEDGMENTS

The internet has a big part in this project. We learned about computer vision, web browsers, tinting car windows, functional programming languages, open-source software collaboration, risk-reward-budgeting principles and so much more that we do not realize. This project has made us realize our potential by applying theoretical CS concepts into a real working mirror. The MagicMirror community has been extremely helpful and the developers have been responsive when questioned about their work.

REFERENCES

[1] https://github.com/eouia/MMM-Spotify
[2] https://devhub.io/repos/MichMich-MMM-Traffic
[3] https://github.com/dvbit/MMM-Face-Reco-DNN
[4] https://github.com/MichMich/MagicMirror/tree/master/modules/default/currentweather
[5] https://github.com/MichMich/MagicMirror/tree/master/modules/default/newsfeed