

Project Dawggles

David Gelody, Aidan Lincke, and Patrick Lucas

Department of Electrical and Computer Engineering, Carnegie Mellon University

Abstract—Dawggles are goggles that feature a forward-facing camera and a transparent, user-facing display. By using inexpensive, widely available components, a 3D-printable design, and open-source software, Dawggles lower the barrier to development. Developers can experiment with head-worn applications without investing in costly, closed ecosystem platforms. Beyond serving as an extensible platform, Dawggles include built-in translation and navigation applications that showcase the benefits of glasses-based software. At the press of a button, Dawggles display an English translation of text within view. By entering a walking destination on a paired smartphone, users can view step-by-step directions at a glance through the display.

Index Terms—Wearable computing, smart glasses, translation, navigation

I. INTRODUCTION

The ever increasing reliance on phones in modern day is somewhat unavoidable. With that said, several efforts have been made so that these digitally reliant tasks can be fluidly integrated into daily life rather than separate, distracting entities. One such major advancement related to this is the concept of ‘Smart Glasses’, and most recently Meta’s Ray-Ban Smart Glasses. These glasses essentially provide users the major capabilities of their phone as a handsfree, partially transparent projection, allowing them to comfortably see both the digital media and their surrounding environment at once. The concept of glasses with heads-up displays addresses several use cases. For example, the transparency effect allows users to follow GPS directions without having to look back at their phone every few seconds, helping the user stay aware of their surroundings. The glasses also provide privacy and safety benefits, where a user can now freely read personal messages, and they don’t have to worry about their phone being stolen if doing so in a crowded area.

Again, the concept is great, but unfortunately the number of affordable options are limited. For example, the cheapest pair of Meta’s smart glasses are the Wayfarer (Gen 1) [1] glasses, listed at \$299. Moreover, these glasses only have camera, voice, and audio functionalities and lack a display. Meta’s glasses are closed source, and their glasses with a display lack any way for third-party developers to interact with it. There are some fully open-source companies, such as Mentra [2], but again they do not have an affordable option with a built in display. The possibilities of smart glasses are endless, and quite frankly, they are the future. The Dawggles open source AR goggles aim to provide users with a customizable AR experience at an affordable price.

Because of a limited one semester project length, we will illustrate the advantage of an open source option by optimizing our design to fit one major use case: traveling.

There are two major functionalities that the Dawggles aim to provide for travelers. The first function is near real time optical translation. Users will be able to translate, with multiple input/output languages, any text they see, without the need to pull out their phone every time. This allows users to not only translate, but also helps them to learn the language they’re translating, as the translation will be displayed in the same relative location as where the text was. The second function is navigation. Users can set their destination and put their phone away, reducing distraction and vulnerability. This allows them to stay fully immersed in their environment while receiving minimalist, yet informative, GPS instructions.

We believe that these two functionalities are within the one semester scope of our project and provide significant improvement to the traveling experience. Finally, as the traveling use case is relatively undiscovered in the field of smart glasses, we believe that improving this use case sufficiently motivates the utility of having an affordable open source option.

II. USE-CASE REQUIREMENTS

We have split up our use-case requirements into 5 major categories in which we aim to optimize the experience for the user: mechanical design, display, translation, navigation, and lifetime.

Mechanical: 1) The mechanical design must not pose any risk of injury to the user. 2) The Dawggles must be comfortably wearable for a full battery lifetime—1.5 hours. 3) The Dawggles must be able to be worn comfortably by any user.

Display: 1) The user must be able to comfortably view active display for 30 minutes at a time to mimic a lengthy phone interaction. 2) The display must appear to the user at least 2 meters away to minimize eye strain. Unused pixels of the display must be virtually transparent to reduce obstructions to the user’s view. 3) The display must be able to show at least 2-3 sentences (approximately 40-50 words) worth of readable text at a time. 4) The user must be able to adjust the text size to accommodate near/farsighted users.

Translation: 1) The translation must be displayed to the user in under 2 seconds. 2) The translation must be able to handle large, arbitrary quantities of text that are longer than the display can fit. 3) For images with multiple groupings of foreign language in the user’s view, the relative position of each translation must be indicated to the user. 4) The user must be able to toggle between translations of most major languages.

Navigation: 1) The display must indicate and update distance until the next action. 2) The display must indicate what the next action is with a glanceable symbol and text. 3) The display must be able to display complex instructions such

as boarding a specific subway or bus line. 4) The display must indicate when the user has arrived at the destination. 5) The distance must be updated every 5 meters so the user doesn't miss an instruction.

Lifetime: 1) The Dawggles must be able to last for 1.5 hours with continuous normal use to simulate an average outing without access to a power source. 2) Dawggles battery and low battery warnings (10%) must be indicated to the user through the display.

III. ARCHITECTURE AND/OR PRINCIPLE OF OPERATION

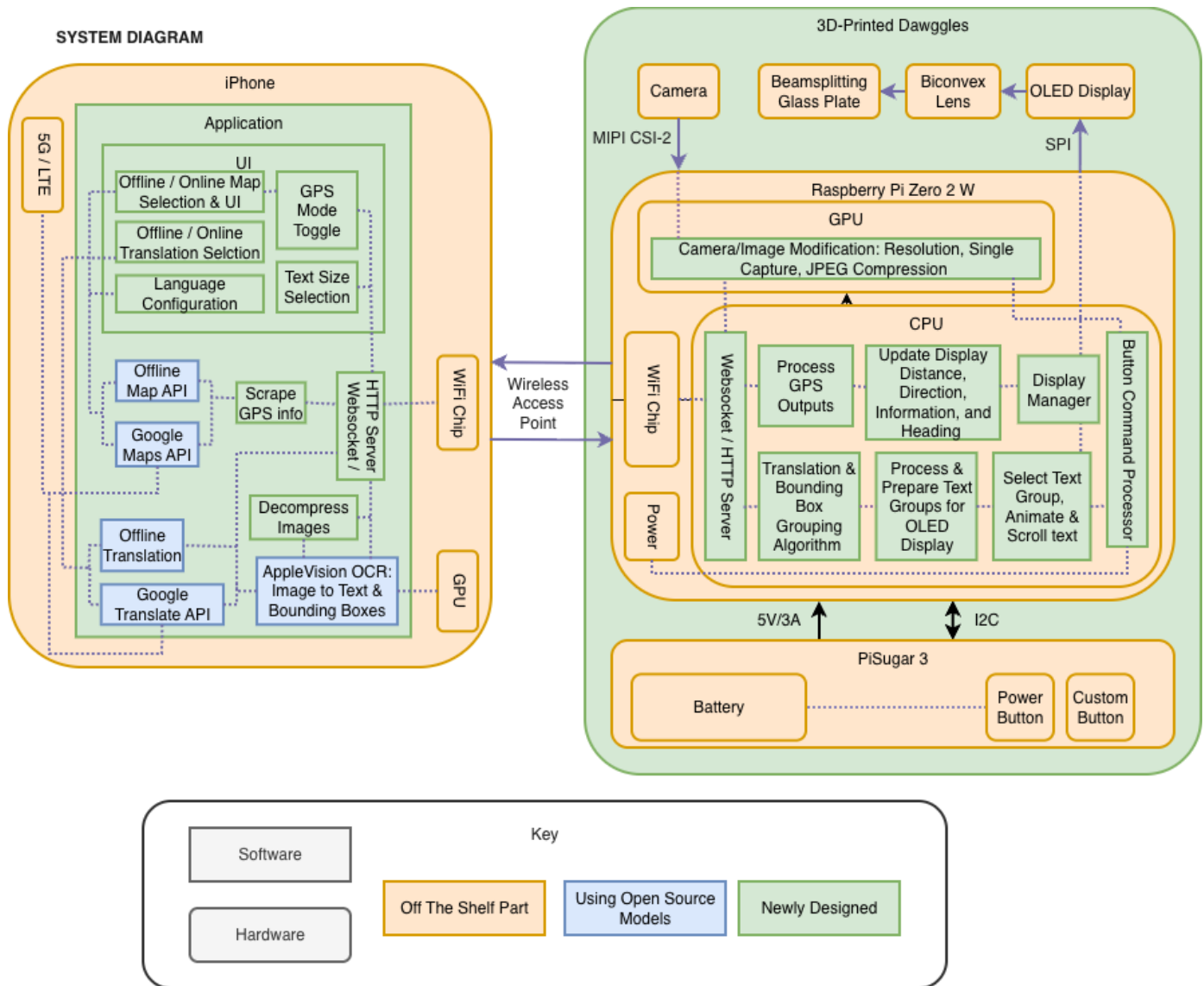


Fig 1. System Block Diagram

The Dawggles system architecture is centered entirely around the use-case requirements. The major architectural choice of the Dawggles was to have the majority of computation done on an iPhone so as to optimize computation time, battery efficiency, and weight of the goggles. The computer on the Dawggles is primarily used as a messenger,

hosting a Wi-Fi access point so that the iPhone can join it to receive images, send translations, and send navigation directions. The system starts on a very simple application, as seen by Fig 1. This application provides users with four configuration options, as well as a GPS toggle.

The first configuration option is Text Size, which allows the user to modify the text size on their Dawggles display. The second configuration is a GPS Mode Toggle button. This button serves two purposes. For one, the GPS Mode Toggle opens permissions for the user to access GPS navigation tools from the application. Secondly, the button provides an

indication is sent to the SBC on the Dawggles. Both the GPS Mode Toggle and Text Size Selection buttons are directly collected in a request to be sent to a server hosted on the Dawggles SBC. This provides the user a basic and streamlined experience with instant text modification and mode switching.

The Offline / Online Map Selection allows the user to use pre-downloaded maps or Google Maps. The user can then put

in their destination and click 'GO'. As the navigation information comes in, the application will use the respective (offline / online) API to scrape the necessary information and forward it to the SBC server. This gives the user the options to either conserve data usage for a bare bones experience or include the cellular-necessary gps information, such as bus schedule times.

The last two configurations are with respect to translation functionality. The Offline / Online Translation Selection serves a similar purpose, allowing the user to select between offline translation models, or the more accurate Google Translation. Lastly, the user can change the from / to languages to translate with the Language Configuration. This configuration is forwarded directly to the translation models.

The other part of the system is the Dawggles, which consist of a microcontroller, battery, and several external components. The primary component of user control on the Dawggles is the side button. From the side button the user can turn on / off the Dawggles, snap a picture for translation, and click forward and backward through groups of translation text. For navigation, the display is just simply updated.

If the user begins a GPS navigation, the SBC server receives a packet with Distance, Direction, Information, and Heading data. Once this data is processed, the SBC assigns it the correct size and pixel locations and sends it to a microOLED display. A biconvex lens is aligned to be parallel to and exactly its focal length away from the display to make the image appear at optical infinity. Then, this image is reflected by a beam-splitting plate which focuses 50% of the image and 50% of the world light so that the user can comfortably see both the display and their surroundings.

The last external component of the SBC is the Camera, which is a primary facilitator of the translation functionality. When the user clicks the button to take an image, the microcontroller requests a single image, resizes and compresses it, and forwards it to the server—the compression significantly decreasing bandwidth. The iPhone receives this image, decompresses it, and inputs it into AppleVision OCR, which is optimized on the iPhone's GPU. This model looks at the image, creates bounding boxes around the characters, and outputs the actual character literals of each box, as well as the bounding box coordinates. The literals are input into the selected translation model, and the output text, as well as the bounding box coordinates from the OCR, are forwarded to a request to be sent back to the SBC.

The SBC receives the translated characters and bounding box coordinates. Then, a basic grouping algorithm determines which, if any, boxes should be merged together into one translation text group. These text groups are processed and prepared for display, and a preliminary screen is displayed, showing just the final group bounding boxes so that the user knows how and where the goggles grouped the text. Then each individual translation group is zoomed into and cycled through on the display. While the translations are being displayed, the user may click the external button once to skip forward translation groups and click twice to go backward a translation group, until the translation reaches the end where the data is deleted and snapshot functionality returned to the user. It should be noted also that both GPS navigation and translations

go through a Display Manager which prioritizes translation displays so that a user may interrupt a navigation session with an image translation.

IV.

DESIGN REQUIREMENTS

This section translates the travel use-case requirements into qualitative and quantitative design requirements. Dawggles must enable a traveler to translate encountered text such as signage, menus, and instructions, and follow navigation cues without repeatedly looking down at a phone. The dominant constraints are end-to-end translation latency, outdoor legibility while preserving real-world visibility, and wearable power and thermal behavior.

End-to-end translation latency must be low enough that a user walking past signage can still read and act on the translated content. The translation pipeline latency t_{E2E} must satisfy $t_{E2E} \leq 1.0$ s typical and $t_{E2E} \leq 2.0$ s worst-case.

Navigation guidance must update frequently enough to remain actionable while walking. The update rate must be at least 2 Hz, meaning new cues at least every 0.5 s. For general UI responsiveness, especially navigation cues, one-way phone-to-goggles message latency should be ≤ 100 ms typical and ≤ 200 ms worst-case. This latency is measured from the phone send timestamp to the first rendered display frame showing the updated cue. Because connectivity is variable during travel, the system must automatically reconnect after a link drop within 5 s, preserve the last-known navigation state, and avoid displaying stale guidance by clearing the overlay after a timeout if fresh navigation data is not received.

The optical system must preserve real-world visibility while still providing readable overlays. Ambient transmission through the primary line-of-sight must be greater than 40% to maintain heads-up safety and situational awareness. To remain readable across lighting conditions, the display and optics must support automatic brightness control from indoor to outdoor environments, along with a night mode to reduce glare in low light. To reduce refocusing strain and improve comfort, the virtual image distance must be greater than 1.5 mm.

Overlay content must be readable but concise. The UI should support at least 10 characters per line and at least 20 characters per message, such as a short translation fragment. character-height target. Translation usability also depends on stable overlay placement. The translation overlay should appear near the original text region, with placement error ≤ 1.5 degrees of visual angle, approximately one character height, during stationary viewing at typical sign distances. Navigation overlays should not exhibit noticeable jitter during normal walking head motion.

Camera and sensing performance directly constrain translation quality. OCR requires sharp, sufficiently detailed imagery, so the minimum effective camera resolution is 640 x 48. The capture pipeline should operate at ≥ 15 fps (preferred 30 fps) and include exposure handling to reduce motion blur under normal head motion. While translation may operate on selected frames, higher frame rates improve the probability of capturing a crisp OCR frame.

Compute and communications constraints emphasize

responsiveness without continuous video streaming. The system must transmit compact overlay primitives for rendering, including text and simple geometry, with low jitter, rather than streaming full-resolution video continuously. After an overlay payload is received, rendering should occur within 50 ms of receipt to preserve perceived responsiveness. The wireless link should remain stable at ≥ 2 m line-of-sight in typical RF environments, while also meeting the reconnection and stale-overlay clearing behavior described above.

Wearable constraints bound practical travel usefulness. The system should provide at least 1.5 hours of continuous mixed use, including periodic translation and continuous navigation, with a stretch goal of 2 to 3 hours. Full recharge time should be under 2 hours in case the user plans on using the Dawggles twice in a day. For safety and comfort, user-contact surface temperature must remain below 45 C during operation. Total goggle assembly mass should be under 350 g, with balanced mass distribution to minimize forward torque on the face.

Finally, the project targets affordability and openness. A reasonable bill of materials target is under 150 to 200 USD, excluding the user's phone, while still meeting the critical latency, readability, and thermal constraints. The system should use modular interfaces and a documented phone-to-goggles protocol, alongside reproducible CAD assembly and an open-source software stack.

V. DESIGN TRADE STUDIES

This section looks at the main trade-offs that actually decide whether Dawggles can meet the use-case requirements across mechanical design, display, translation, navigation, and lifetime. The point is not to optimize one part in isolation, but to understand what constraints “push back” when we try to hit the user-facing goals, and what design choices help multiple requirements at once.

On the mechanical side, the biggest trade-off is stability versus comfort. If the goggles grip harder, weigh more, or concentrate pressure in small contact areas, they will feel stable but become uncomfortable over a two-hour wear period and can create pressure points. If the goggles are too light and loose, they may shift during walking, which is bad for both comfort and display alignment. That pushes the design toward keeping mass low, keeping heavy parts close to the head so they do not create forward torque, and spreading contact forces over larger surfaces. Another important trade-off is adjustability versus rigidity. The goggles need to fit many different head sizes comfortably, but too many moving parts can reduce stiffness and make optical alignment inconsistent. Since alignment affects perceived image quality and overlay stability, the mechanical design needs enough adjustability to fit most users while still “locking” into a repeatable position on the face.

For the display, the requirements couple together in a way that forces compromise between comfort, transparency, and how much text can fit. Making the virtual image appear far away reduces eye strain, especially if the user is viewing the overlay for around 30 minutes continuously, but longer virtual distances and see-through optics can make the overlay look dimmer and lower contrast in bright environments. That means the display has to have enough brightness headroom

and automatic brightness control, and it also means we need rendering modes that preserve readability without always blocking the user's view. The transparency requirement is a good example. In practice, text is easiest to read when it has a background plate behind it, but that background plate also obstructs the real world. A reasonable compromise is to keep the default overlay minimal and transparent, and only add subtle background help when the scene is bright or visually cluttered. The “2–3 sentences readable at a time” requirement interacts directly with adjustable text size. Bigger text helps near- and farsighted users, but it reduces how many words fit. So the design needs layout rules that keep the most important content visible first, and then provide a clean way to show more text without turning the display into a wall of words. This is especially important for translation, where the incoming text can be arbitrarily long.

For translation, the under-2-second requirement is the main “hard constraint” that shapes everything else. Consider the following equation:

$$t_{E2E} = t_{capture} + t_{tx} + t_{OCR+NMT} + t_{rx} + t_{render} \quad (1)$$

where t_{E2E} is the total latency from capture timestamp to the first visible translated overlay; $t_{capture}$ is frame acquisition time (exposure + readout + buffering); t_{tx} is transmit time for the request or payload; $t_{OCR+NMT}$ is OCR and translation compute time (including required pre-processing and post-processing); t_{rx} is receive, parse, and queue time on the goggles; and t_{render} is overlay rendering and display presentation latency to the first visible frame.

If we try to run very heavy OCR and translation models, quality may improve but latency increases, and the user experience suffers because the overlay arrives too late to be useful. The more practical strategy is to prioritize responsiveness, then handle longer or more complex translation requests through progressive display. In other words, show something helpful quickly, then allow the user to step through additional translated text in chunks rather than trying to fit everything at once. The “large arbitrary quantities of text” requirement is basically a UI and interaction problem, not just a model problem. We need a way to paginate or scroll translated content without forcing repeated captures of the same scene. Multi-region translation introduces a different trade-off: spatial accuracy versus overlay stability. In scenes with multiple blocks of text, the user needs to know which translation corresponds to which block. Trying to perfectly track every block in real time can create jitter and visual distraction during head motion. A simpler and often more robust approach is to label detected regions with small markers or numbers and show the translations in a list that references those markers. This preserves correspondence even if the user moves slightly. Language switching adds a trade-off between flexibility and complexity. Supporting most major languages pushes us toward an interface that makes target language changes quick and persistent, and ideally supports auto-detection as a convenience, but not as the only mechanism because mis-detection is frustrating when traveling.

For navigation, the main trade-off is fast updates versus a stable, glanceable overlay. Updating distance every 5 meters helps the user stay oriented, but constant changes can feel noisy and distracting if the display is always flickering numbers. A good compromise is to update the internal estimate continuously, but quantize what is shown, so the displayed distance changes in 5-meter steps rather than jittering constantly. The requirement to show both a symbol and text also forces hierarchy. The user should be able to understand “what to do next” instantly from a glanceable icon and short phrase, while secondary details can carry more complex information like subway line names, directions, and boarding instructions. Complex instructions are almost never glanceable in full, so the design needs to break them into a short primary cue and a secondary detail line. Arrival indication has a similar trade-off: calling “arrived” too early is confusing, but calling it too late defeats the purpose. This suggests using a clear threshold and possibly a two-stage state like “Arriving” then “Arrived” once the system is confident.

Finally, lifetime is shaped by what stays on continuously versus what happens in bursts. Navigation tends to be a steady drain because it needs persistent connectivity and periodic updates, while translation is bursty and costs power mainly during capture and compute. Since the requirement is five hours of normal use with around 20 translation images and one hour of navigation, the design should minimize always-on power draw in the goggles and offload heavy compute to the phone whenever possible. Battery indication is also a user-trust issue. If the user cannot see battery status, they will not rely on the system during travel. At the same time, battery UI cannot be distracting. The practical approach is a small persistent battery indicator plus a clear low-battery warning at 10 percent, which matches the requirement without spamming the user.

Putting these together, the design direction is fairly consistent. Mechanically, keep the device light, balanced, and adjustable while maintaining stable alignment. For the display, prioritize comfort with a sufficiently long virtual image distance while maintaining readability through brightness control and careful overlay rendering that stays transparent by default. For translation, hit the latency target first and handle long and multi-region text through chunking and simple correspondence cues rather than fragile real-time tracking. For navigation, keep cues glanceable and stable while meeting the 5-meter update requirement through quantized display updates and a clear information hierarchy. For lifetime, reduce always-on power, offload compute to the phone, and provide clear battery feedback so the system remains reliable during real travel use.

VI. SYSTEM IMPLEMENTATION

As suggested by the system block diagram (Fig. 1) the two major aspects of our system are the iPhone and the Goggles. The iPhone must have at least iOS 15 in order to run AppleVision OCR. The microcontroller we chose for the goggles is a Raspberry Pi Zero W: a low power SBC with a wifi module and proper I/O connection ports for our devices. We will only be implementing new software on the iPhone

through a swiftUI application. On the other hand, the Goggles involve external hardware connections, new software, and a 3d printed mechanical design of the goggles themselves. In this section, we will outline the specific hardware, software, and mechanical design that we plan to implement.



Fig 2. Goggles Front

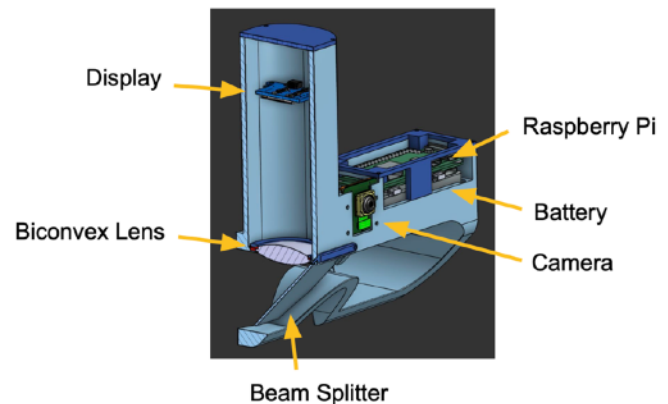


Fig 3. Goggles Labeled Cad

On the Goggles, there are 3 major subsystems: Button Management, Image Capture, and Display. A mechanical CAD of the Goggles system can be seen for reference (Fig 3).

A. Button Management

Hardware

We will be using two standard pushbutton switches for button control. The pushbuttons will be connected to GPIO pins and GND pins on the Pi Zero 2W. These buttons will be used to trigger image captures for translation, as well as scrolling forward and backward through applications on the display. The wiring between the Pi zero 2W and external button can be seen in Fig 2. There will also be a button on the battery which the user can use to shut off just the display or power off the app entirely.

Software

The main GPIO button will be tied to a callback function in

software. This function handles 2 cases. If the goggles are in the DEF (default) state, then the function sets them to the TRANSLATION state and gets a picture from the camera for translation. The goggle state is not set back to default until the end of the display, as seen by the main() function in the Display Thread (Fig 6). In the other case, where the goggles are already in the TRANSLATION state, the display index (managed by the SharedClass) is iterated by 1 forward or backwards depending on a single or double press. This allows the user to skip forward or backward through groups of translations.

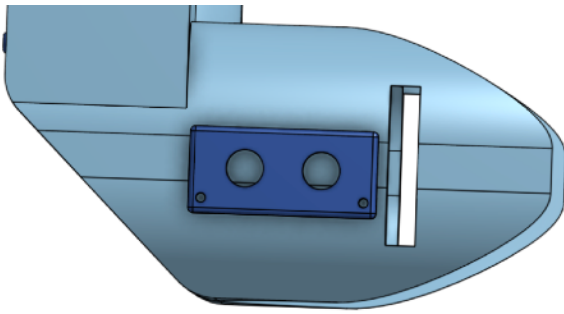


Fig 4. Goggles Button Mount Cad

Mechanical

The button will GPIO button will simply be mounted on the right side of the goggles.

B. Image Capture

Hardware

As seen by Figure 5, the Pi Camera 3 is connected to the Pi Zero 2W through a ribbon cable using MIPI CSI-2. Image capture and processing is performed on the Pi's GPU so as not to crowd CPU cores and to optimize resizing and JPEG compression.

Software

The software for image capture is relatively simple. A subthread sleeps until it is signaled by the button to take a picture. Then, the subthread sets the SharedClass picture so that the global picture can be quickly retrieved by the button thread. This thread then calls sendPicture to send a POST request with the picture to the iPhone's IP address.

Mechanical

The camera is centered horizontally between the eyes and mounted on top of the goggles to get a clear picture extremely close to what the user is actually viewing.

C. Display

Hardware

The display is a monochrome 128x64 monochrome microOLED display. It is using SPI communication with the Pi Zero 2W's header pins in order to optimize data communication speed (Fig. 5). With 8px font, a 128x64 display provides us with over 100 readable characters, which easily meets our translation and display requirements

Software

If the iPhone send the a request to the server with translation info, then the main thread groups the Translations

together. It does so using an algorithm to check how close the adjacent bounding boxes are and determine if this text should be grouped into one combined box. After these boxes are grouped, the display thread thread main() function displays a barebones animation of the bounding boxes using their relative locations and sizes. Then, each translation is animated and displayed. The user can again click forward or backward though these displays through the side button which changes the display index. If instead of receiving translation data, the server receives gps_info, then the display simply renders the data (distance, direction, information, & heading) using predefined parameters.

Mechanical

The display is placed above the goggles, 45mm—a focal length—away from the biconvex lens. The modified image is then reflected using a 50/50 beamsplitter at a 45 degree angle. This allows the user to see the display overlaid on their own surroundings.

VII. TEST, VERIFICATION AND VALIDATION

This section describes the testing methods used to evaluate the Dawggles prototype, summarizes the measured results, and compares the implemented system performance against the original design specifications and use-case requirements. The goal of verification was to determine whether each subsystem met its quantitative design requirements, while the goal of validation was to determine whether the full system was usable for the intended translation and navigation use cases.

The primary use-case requirement for Dawggles is that a user should be able to view short translated text or navigation instructions through a wearable heads-up display without needing to look down at a phone. To support this use case, the system needed to satisfy several design specifications: readable projected text, sufficient camera resolution for text recognition, low end-to-end translation latency, acceptable battery life, and a wearable physical form factor. Testing was performed on both individual subsystems, such as the camera and optical display, and on the full integrated system.

The display subsystem was tested to verify that the projected text was readable and contained enough information to support short translations. The design specification required the display to show at least 10 characters per line. After implementation, the projected display was able to show approximately 15 characters per line. The display also supported roughly 8 lines of text, corresponding to approximately 120 total characters.

These measurements exceed the original display-capacity requirement and help validate the translation use case. A typical translated phrase, menu item, sign, or short instruction can fit within this amount of text. This means the user can receive useful translated information without needing to constantly scroll or look at a separate phone screen.

The virtual image distance was also measured as part of the optical verification. The design specification required a virtual image distance greater than 15 mm. The implemented prototype achieved a virtual image distance of approximately 25 mm. This result exceeded the design target and improved the usability of the projected display by making the text easier to focus on.

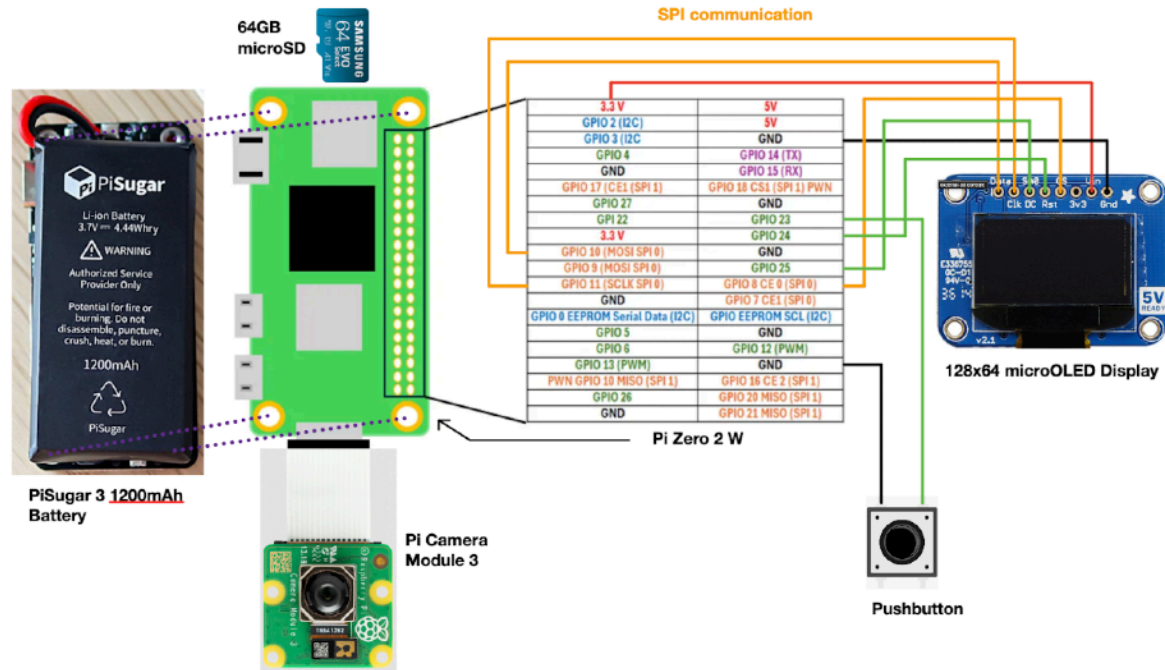
GOGGLES HARDWARE DIAGRAM

Fig 5. Goggles Hardware Diagram

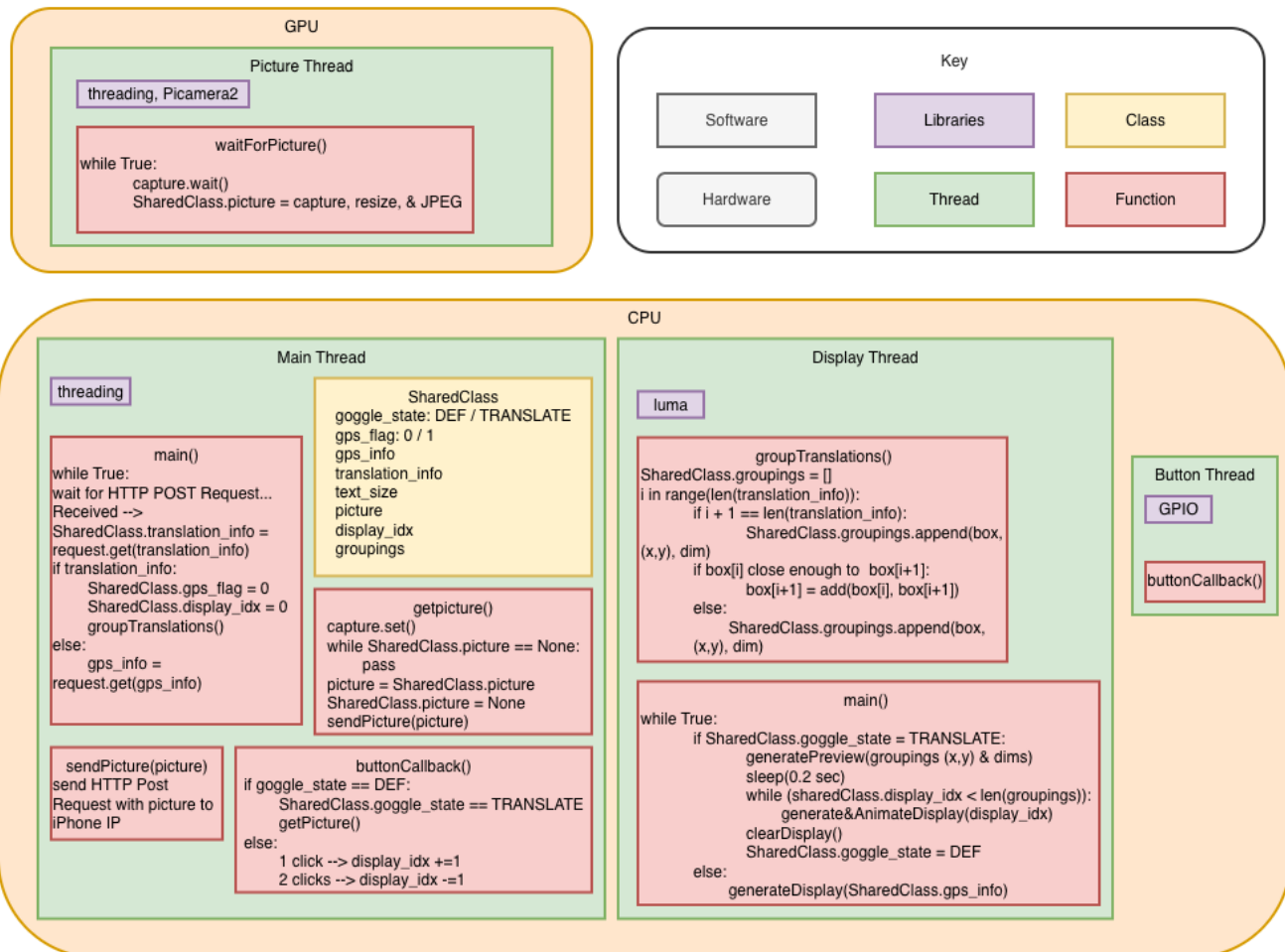
PI ZERO 2W SOFTWARE DIAGRAM

Fig 6. Pi Zero 2W Software Diagram

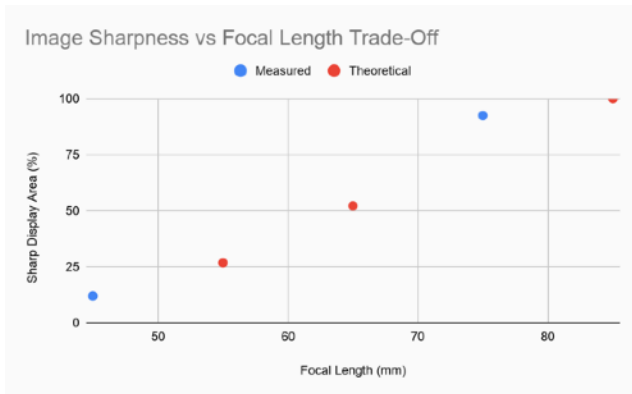


Fig. 7 Image Sharpness vs Focal Length Trade-off Graph

The main factor limiting this specification was the trade-off between display sharpness and the physical height of the goggles. Increasing the focal length improved image sharpness, but also required a larger physical optical path. This made the goggles taller and less comfortable to wear. Testing multiple 3D-printed prototype bodies showed that shorter optical paths produced a more compact form factor but reduced image sharpness, while longer optical paths improved readability but increased size.

This matched the theoretical prediction from the optical design analysis. The Pareto relationship showed that better display quality required a longer focal length, which increased the size of the physical prototype. The final implemented design was therefore a compromise between readability and wearability. Although the display was not perfectly sharp, it was readable enough to satisfy the design requirement and support the intended use case.

The camera subsystem was tested to verify that it could capture images with enough resolution for text recognition. The design specification required a minimum camera resolution of 640 by 480 pixels. The implemented camera provided a resolution of approximately 4608 by 2592 pixels, which significantly exceeded the minimum requirement.

This result was important because text recognition performance depends heavily on image quality. A higher-resolution camera gives the OCR system more detail to identify characters, especially when text is small, partially blurred, or viewed from a distance. In the integrated prototype, the camera was able to capture text for translation and send it through the recognition pipeline.

However, camera resolution alone did not guarantee perfect recognition. The translation reliability survey results showed that users rated translation reliability as only moderate. This suggests that OCR performance was also limited by lighting, camera stability, text contrast, image blur, and the quality of the input frame. Therefore, while the camera resolution specification was verified, the broader use-case performance still depended on environmental conditions and software processing.

The system also required a design choice between the fast and accurate modes of Apple's VNTxtRecognize framework. The fast mode provided approximately 30 frames per second but achieved less than 75% recognition accuracy.

The accurate mode achieved greater than 90% accuracy but operated closer to 5 frames per second and required more processing. Since the Dawggles use case depends more on correct translation than maximum frame rate, the accurate mode was preferred.

This result matched the theoretical software trade-off between speed and accuracy. A faster recognition system improves responsiveness, but if the recognized text is incorrect, the translation is less useful. A more accurate recognition system reduces errors but limits live-frame performance. For this prototype, the accurate mode gave better use-case performance because the user needs reliable translated text more than continuous high-frame-rate recognition.

The full translation pipeline was tested to verify that the system could provide translated text quickly enough for interactive use. The design specification required a round-trip translation time of less than 2 seconds. The implemented system achieved a typical round-trip translation time of less than 1 second.

This test measured the full path from capturing an image, sending it to the app, performing text recognition and translation, and rendering the translated output back to the display. Since the measured latency was below the 2-second requirement, the system successfully met the latency specification.

This result directly validates the main translation use case. If the system took several seconds to return a translation, the user would have to pause and wait, making the device feel awkward or impractical. Because the measured translation time was under 1 second, the system felt nearly instantaneous during typical use. This allowed the user to point the camera at text and receive translated output without a disruptive delay.

The limiting factors for translation latency were OCR processing time, translation time, communication delay between the hardware and app, and rendering time. In live mode, the system averaged roughly 5 seconds across 25 frames, while static mode processed a single frame and allowed more complex text grouping and UI behavior. This showed that the static pipeline was better for accurate text grouping, while live mode was better for interactive feedback.

The measured latency was consistent with the expected design trade-off. Faster live recognition provides a more interactive user experience but reduces the complexity and accuracy of text grouping. Static recognition is slower in terms of interaction flow but can produce cleaner and more reliable translation output. The final implementation favored a more accurate and usable translation result rather than maximizing frame rate.

The power subsystem was tested to verify that the prototype could operate long enough for realistic short-term use. The design specification required a battery life greater than 1.5 hours. The implemented system operated for approximately 2 hours, exceeding the requirement.

The charging requirement was that the device should charge in less than 2 hours. The measured charge time was approximately 30 minutes, which also exceeded the design specification.

These results validate the intended use case of short-

duration wearable translation. A 2-hour battery life is sufficient for demonstrations, short trips, reading signs, navigating unfamiliar spaces, or translating menus and instructions. The 30-minute charge time also makes the system easier to reuse between testing sessions.

Battery life was primarily limited by the display, camera, wireless communication, and processing load. Improving the display brightness or running more continuous recognition would likely increase power consumption. This creates a trade-off between performance and operating time. The current prototype achieved an acceptable balance, but future iterations could improve battery life by using lower-power hardware, optimizing the translation pipeline, or reducing display power when text is not actively being shown.

To validate the full system use case, a survey was conducted with 32 respondents. Participants rated the prototype on a 1 to 5 scale across several categories: readability of text, comfort of the Dawggles, ease of the iOS app, ease of camera translation, connection reliability, and translation reliability.

The highest-rated category was connection reliability, with an average rating close to 5. This suggests that users found the connection between the Dawggles hardware and the app stable. This is important for the use case because connection failures would interrupt translation and reduce user trust in the device.

Readability of text received a positive rating, which supports the measured display results. Users were generally able to read the projected text, confirming that the display subsystem was usable in practice. However, the rating was not perfect, which suggests that improvements to sharpness, contrast, and alignment would further improve the experience.

Comfort received a moderate rating. This result reflects the mechanical trade-off observed during design and testing. The optical system needed enough physical space to produce a readable virtual image, but this increased the size of the goggles. As a result, the prototype was functional but not yet optimized for long-term comfort. This shows that the use case was validated for short-term operation, but extended wearability remains an area for improvement.

The iOS app and camera translation workflow received decent ratings, indicating that users could operate the system successfully. However, the results suggest that the interface could be improved with clearer connection status, more intuitive controls, and better feedback during translation.

Translation reliability also received a moderate rating. This indicates that the prototype successfully demonstrated the translation workflow, but performance was not fully robust across all conditions. Translation reliability was limited by OCR accuracy, image quality, lighting, and the complexity of the text being captured. This result shows that the system validates the basic translation use case, but future work is needed to make it dependable in more real-world environments.

Overall, the Dawggles prototype met or exceeded the main quantitative design specifications. The display showed approximately 15 characters per line compared to the required 10, supported about 120 total characters, achieved a virtual image distance of approximately 25 mm compared to the

required 15 mm, used a camera resolution much higher than the minimum 640 by 480 requirement, achieved translation latency under 1 second compared to the 2-second requirement, lasted approximately 2 hours compared to the 1.5-hour requirement, and charged in approximately 30 minutes compared to the 2-hour requirement.

The user validation results showed that the prototype was readable, reliable in connection, and usable through the app, but also revealed areas for improvement in comfort and translation reliability. These results align with the theoretical design trade-offs. Improving display sharpness required a larger optical path and less compact goggles, while improving OCR accuracy required lower frame rate and more processing. The final prototype balanced these trade-offs well enough to demonstrate the intended wearable translation use case, while leaving clear opportunities for future design improvements.

VIII. PROJECT MANAGEMENT

This section outlines the the project management strategy of the Dawggles project.

A. Schedule

A view of our schedule in Jira is attached to the end of the document as Figure 7.

B. Team Member Responsibilities

David will serve as the primary project manager, ensuring that tasks stay on-time. In addition, David will handle the electrical aspects of Dawggles, such as making sure that components are properly powered and that protocols work correctly together. For example, he will write code for the Dawggles display to correctly show certain UI on the screen over SPI.

Aidan will primarily develop the iOS app that serves as the Dawggles's bridge to the internet and the translation layer. He'll also develop the UI and interactions that help users navigate and use Dawggles.

Pat will primarily design the CAD model of the 3D-printable Dawggles. David and Aidan will also assist with the CAD and hardware portion of the Dawggles. As a secondary responsibility, Pat will develop the ML character recognition and translation layer that produces translated English from photos.

C. Bill of Materials and Budget

See Table I attached to the end of the document for the bill of materials and budget.

D. TechSpark Use Plan

We did not use TechSpark. We used RoboClub to 3D print our designs.

E. Risk Mitigation Plans

There are a few critical risk factors that are present in the Dawggles project, and we have plans to address each.

Firstly, the mechanical design of the Dawggles may be too bulky or heavy to be comfortably worn by a user. To mitigate this, we've taken several actions since the beginning of the Dawggles project. For example, we decided to design and

print our own model of goggles instead of modifying store-bought snow goggles. We believe that modifying store-bought goggles would be too difficult, complex, and would also oppose the open-source use case because it would not be reproducible. We've also added significant time to our schedule for mechanical design and prototypes due to the time required to design and iterate on 3D-printed goggles.

Secondly, we've considered the risk of the optical system not working properly. Specifically, none of our team has experience designing systems that use mirrors or lenses to show a display. We're managing the risk by using the simplest optical system that can display a transparent screen in a field of view, and we have avoided more complex systems to achieve higher-end display styles. We have also added more time to the schedule to iterate on mirror angles and placement.

IX. ETHICAL ISSUES

The Dawggles have a few ethical issues that arose during development. Firstly, they contain a forward-facing camera that would be worn by the user and could capture potentially privacy-sensitive areas. This fact makes security important because a vulnerability would compromise the privacy of anyone in sight of any pair of Dawggles. Secondly, the goggles don't contain any indicator light to indicate that the camera is in use like the Meta Ray-Bans do. We made this choice because the Dawggles's camera is particularly obvious; however, if a streamlined sunglasses form factor is achieved, the Dawggles would need to adapt an indicator light to make it apparent to others that the goggles contain a camera and the camera is recording.

Besides the ethical issues relating to the Dawggles's camera, the Dawggles platform also raises concerns related to user distraction and a reality where everyone wears displays. In the present day, it can already be difficult to hold conversations with people or meet new people when they're often distracted or using their smartphones, and frequent smartphone usage has already significantly altered society. If people constantly wear smart glasses or goggles with displays, these distractions could increase drastically. To alleviate this, we created a button on the Dawggles that instantly toggles the display on or off so that anyone having a conversation or needing to pay attention to the outside world can quickly do so.

X. RELATED WORK

Dawggles are most comparable to Meta Ray-Ban Displays. Using advanced glass manufacturing techniques, Ray-Ban Displays achieve ultra-high-resolution transparent visuals, allowing users to see the outside world with minimal obstruction when the display is off. Their visible light transmission (VLT) far exceeds what we can achieve in the Dawggles project. However, these glasses are expensive, restricted to approved developers, and raise significant privacy concerns.

XI. SUMMARY

Overall, our system met its design specifications. We made a pair of smart glasses with a display that exceeded the weight,

battery, display, camera, latency, and user experience requirements that we established at the beginning of the semester. With more time, we'd improve the form factor of the Dawggles, find ways to increase compute on the goggles, and also create additional apps for it.

With a glasses form factor rather than goggles, Dawggles users could begin to use the goggles outside without it being so obvious or strange to wear. To accomplish that, we'd make changes to the optical system to be less physically obtrusive.

With more compute available on the goggles, we'd be less reliant on the user's phone, and the goggles could be more of a standalone, functional device. However, this is difficult at present without increasing the physical size of the goggles or making them warmer.

GLOSSARY OF ACRONYMS

RPi – Raspberry Pi

WAP – Wireless Access Point

REFERENCES

1. Meta, "Ray-Ban Meta Wayfarer AI Glasses – Shiny Black with Clear Lenses," Meta, [Online]. Available: <https://www.meta.com/ai-glasses/wayfarer-shiny-black-clear/>. Accessed: Feb. 28, 2026.
2. Mentra, "Mentra | Build smart glasses apps" Mentra, Mentra Labs, Inc., 2026. Available: <https://mentraglass.com>. Accessed 2 May 2026.

Table I: Bill of Materials

Name	Model Number	Manufacturer	Quantity	Link	Price + Shipping	Notes
OTG Snow Goggles (Black Frame VLT 99%)	UPC 198387004558	OutdoorMaster	1	Link	19.49	VLT 99%
Adafruit Industries Monochrome 0.96" 128x64 OLED Graphic Display	UPC P326	Adafruit Industries	1	Link	31.49	
Beam Splitting Glass Plate Splitter 50*50*1.1mm Split Ratio 50/50	AliExpress 325680116230 0946	Sophisticated Optical Tools	2	Link	21.64	
45mm Focal Length Biconvex Lens Set	Amazon B012HYF4A2	PopTech	1	Link	7.99	
Pi Zero 2 W with Pre-Soldered Header (Board Only)	SKU PI-Z2WH	Raspberry Pi	2	Link	36.55	
Mini HDMI to HDMI Cable	Amazon B0C3LR54C3	KELink	1	Link	5.99	
64GB microSD Memory Card + Adapter	UPC 887276821061	Samsung	1	Link	19.99	
Raspberry Pi Camera Module 3	SKU SEN-21331	Raspberry Pi	1	Link	41.44	
PiSugar 3 1200mAh Raspberry Pi Zero Battery	SKU 5842-PISUGA R31001-ND	PiSugar	1	Link	39.99	
75mm Focal Length Biconvex Lens Set	Amazon B0FDW8VXP6	PATIKIL	1	Link	8.24	
M2 Screws	Amazon B0DDXZH9L2	QOOSIKICC	1	Link	7.98	
Head Band	Signature Series Google Straps	Dang Shades	2	Link	19.99	
Neoprene Head Foam	Amazon B07WTRK3LX	Dualplex	1	Link	14.95	
Head Foam	Amazon B001EQON6Q	Frost King	1	Link	6.48	

Grand Total: 323.65

Figure 8: Schedule

