

GNG 1103
FALL 2024
October 13, 2024

DELIVERABLE G

PROTOTYPE 2 AND CUSTOMER FEEDBACK

Team DISMISS

GABRIELLE CHÉNIER
RUSAFI KAMAL
DARRIEN CHEN
HANNAH KNIPE
QUAN LU

Table of contents

.....	1
Table of contents	2
Introduction.....	3
Prototype	4
Prototyping analysis.....	5
Prototyping test plan	6
Results.....	7
Face Detection.....	7
Voice Assistance	8
Object Recognition.....	9
Updated target specifications, detailed design & BOM.....	11
Updated BOM	11
Updated Target Specifications.....	15
Prototyping test plan for prototype 3	17
Conclusion	18

Introduction

As part of the previous deliverable, we continued the prototyping stage by establishing and creating our first prototype, a Figma document. We had a better sense of how our final application and prototype would look like and what we would want to include in it because of that prototype. To resume this stage, we will construct our second prototype, which focuses on developing and testing individual subsystems. We will then analyze all the components, then we will define a test plan to get feedback and comments from several possible users. Each test will guide future improvements. Finally, we will establish our prototyping test plan for the final prototype, which we will combine all the APIs into one final application.

Prototype

For the second prototype, we planned to focus on the different subsystems we are planning to have in our final product and analyze and test them independently to make sure that they are meeting the standards to be added to the final product. This prototype consists of a few branches: voice assistant, face detection and face tracking. Below are some pictures attached of some snippets of the ode to visualize its fidelity since this prototype is fully software based. This prototype is experimental since we are testing all the different subsystems separately. Additionally, the program depends on many different features, like different lighting conditions.

```
from ultralytics import YOLO
import cv2

# Load YOLOv8 model
model = YOLO('yolov8n.pt')

# Open the camera (0 is the default camera, try 1 or 2 if it doesn't work)
cap = cv2.VideoCapture(0)

# Check if camera opened successfully
if not cap.isOpened():
    print("Error: Could not open camera.")
    exit()

# Read frames
while True:
    ret, frame = cap.read()

    if ret:
        # Detect objects and track them
        results = model.track(frame, persist=True)

        # Plot results
        frame_ = results[0].plot()

        # Display the frame
        cv2.imshow('frame', frame_)

        # Exit loop when 'q' is pressed
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break
    else:
        print("Error: Failed to capture frame.")
        break

# Release the capture and close the window
cap.release()
cv2.destroyAllWindows()
```

Figure 1: Object recognition and face detection

```
# Function to run omega based on the recognized command
def run_omega():
    command = take_command()
    if command: # Only proceed if there's a command after "omega" is detected
        if 'time' in command:
            time = datetime.datetime.now().strftime('%I:%M %p')
            print(time)
            talk('Current time is ' + time)
        elif 'play' in command:
            song = command.replace('play', '').strip()
            talk('playing ' + song)
            pywhatkit.playonyt(song)
        elif 'tell me about' in command:
            look_for = command.replace('tell me about', '').strip()
            info = wikipedia.summary(look_for, sentences=1)
            print(info)
            talk(info)
        elif 'joke' in command:
            talk(pyjokes.get_joke())
        elif 'open camera' in command:
            open_camera()
        elif 'stop' in command:
            talk("Goodbye!")
            return False # Exit the loop in main code
        else:
            talk('Searching ' + command)
            pywhatkit.search(command)
    return True
```

Figure 2: Voice Assistant (1)

```
# Function to listen for a voice command
def take_command():
    try:
        with sr.Microphone() as source:
            listener.adjust_for_ambient_noise(source, duration=1) # Adjust for ambient noise
            print('listening...')
            voice = listener.listen(source, timeout=5, phrase_time_limit=5) # Adjust time limits
            command = listener.recognize_google(voice).lower()
            if 'omega' in command:
                command = command.replace('omega', '').strip()
                return command
            else:
                return ""
    except sr.WaitTimeoutError:
        print("Listening timed out while waiting for phrase to start")
        return ""
    except sr.UnknownValueError:
        print("Could not understand the audio")
        return ""
    except sr.RequestError:
        print("Could not request results; check your network connection")
        return ""
```

Figure 3: Voice Assistant (2)

Prototyping analysis

For our second prototype, we have created multiple focused subsystems which are all inputs for our software. We are creating and testing viable software for each of these subsystems that work on their own. Such as Omega Voice assistant, our prototype consists of the input of user asking question, and the working output of the software answering commands. We have decided to make many focused input prototypes for input as we need our input data to create the output software. Without creating an object detecting software, our output software on what the user should do would not exist. Our previous prototype of creating the Figma document on the app user navigation helped find out what features we need to incorporate into our different subsystems. For our second prototype we are creating the development of these features.

Prototyping test plan

Test ID	Objective	Description of test methods and materials needed	Description of results to be recorded	Estimated test duration
1 Face Detection	Testing the reliability of face detection	Having different faces and seeing whether they are detected or not. Seeing the cutoff distance until the face is no longer detected.	We will have two columns; if the face was detected or not, and the cutoff distance. Realistically, we want a cutoff distance large enough to potentially warn the user wearing the glasses about anything appearing in front of them.	40 minutes
2 Voice Assistant	Testing the reliability of the voice assistant	Using different people call the voice assistant and noting if it activates and follows the instructions as provided.	We will record the number of times the voice assistant was called and successfully activated by means of a Boolean.	1 hour
3 Object Recognition	Testing the reliability of object recognition	Using different objects and checking whether they are being recognized and checking the cutoff distance for the recognition.	We will again record if an object has been detected or not using a Boolean of for object detection, we also want it to be around so there is a safe distance between the user and the object.	50 minutes

Results

For this prototype, we decided that for the prototype test plan, we would question people of a variety of backgrounds to obtain a range of answers. For instance, we asked family members for their opinions and input at the beginning of the testing stage and ended by asking fellow classmates. We may have also asked strangers to get an even wider range of responses. To try and test every attribute of every feature we tried to have each user test a different attribute.

Increments	Test ID 1		Test ID 2		Test ID 3	
	Face detected	Cutoff distance (>1.0m) (m)	Voice Assistant Activated	Understood Instructions	Object detected	Distance (m)
User 1	Yes	1.3	Pass	“Omega What time is it “	Yes	0.9
User 2	Yes	0.5	Fail	“Omega what is the date”	Yes	1.0
User 3	Yes	0.9	Fail	“Omega what’s the weather”	Yes	1.1
User 4	Yes	1.2	Pass	“Omega play Never going to give you up YouTube”	Yes	0.8

Face Detection

Code 1 – Face detection			
Users	Pros	Cons	Additional comments and feedback from the user
User 1	The code identifying faces very well. The system recognized the user’s face in less than a second.	Although the user finds this code interesting, they are unsure of how this could be utilized without a steady and constant internet connection. The applications objective cannot be supported by an irregular internet connection	N/A

User 2	Under ideal circumstances, the system had no trouble quickly identifying and following the user's face.	The system had trouble identifying the user's face due to some poor lighting conditions. The system may need external lighting to fix this issue.	N/A
User 3	The face detections worked even when the user wore glasses or a hat. It detected their face instantly.	The user was disappointed that the feature could not remember faces.	The user mentions some possible room for improvements: face REGOGNITION and a better colour for the frame that surrounds the faces.
User 2 and user 3	Both users tried to test if the face detection devices would work for multiple faces. The program was able to easily detect and track both faces.	N/A	N/A

To briefly sum up the first test, the user-style pros and cons provided a personal perspective onto the strengths and weaknesses of the face detection feature. The feature is strong in its response time and tracking ability. It can quickly detect faces. However, the feature experienced trouble when exposed to poor lighting. Some useful attributes to add to it would be not needing an internet connection and having the ability to remember faces.

Voice Assistance

Code 2 – Voice Assistance			
Users	Pros	Cons	Additional comments and feedback
User 1	The user found it interesting that OMEGA could search anything and give quick answers without having to reach for their phone.	The feature sometimes had trouble understanding what the user was saying or searched something unrelated.	Making it have some more custom commands such as “what is the weather” or “set a timer for _ minutes”.

User 2	The device quickly gave the correct time and date when asked.	It took a couple of tries to get the time	The user thinks that a useful attribute to add would be a voice recognition. It could be useful to have it remember and differentiate voices.
User 3	This feature was able to answer a lot of knowledge questions, such as “What is the University of Ottawa?”. It gives some detailed and accurate answers.	Again, in some cases the voice assistant would ‘search’ instead of answering the questions asked.	N/A
User 4	The user found it interesting that this feature could open any page on the internet. For instance, the user was able to play a song with the command “play”.	The program sometimes had trouble understanding the user’s commands.	N/A

Ultimately, the second test was a success for multiple reasons. The code was able to give quick and correct answers to any questions. Additionally, it could open any page on the internet. However, the voice assistance feature could also improve in certain areas. For instance, the program occasionally had trouble understanding what the user was saying. For better input, we are thinking of using a professional microphone for better inputs and better voice recognition for the voice assistant.

Object Recognition

Code 3 – Object recognition			
Users	Pros	Cons	Additional comments and feedback
User 1	The user thinks that it is highly impressive how the program can accurately recognize and identify different objects. In this case, the user showed the device a teddy bear and a television, which the device was	Even though it showed the name of the item, there were errors sometimes when an object resembled something different, and the recognizer would	N/A

	able to both identify correctly.	suggest a different object.	
User 2	The devices were able to quickly identify almost all the objects the user showed.	The device sometimes had trouble identifying more complex items.	N/A
User 3	The user agrees that this feature is great for accessibility. It can help visually impaired individuals identify objects and navigate their surroundings	Does not make audio cues when objects are detected.	There could have been a feature which dictates the objects being recognized.
User 4	It was very responsive in terms of identifying as well as tracking it.	The program could not correctly identify every object presented by the user.	The user thinks that the program should be able to identify more objects.

Overall, our object recognition device offers impressive benefits like fast responses times, high accuracy and some valuable applications for accessibility. Although, it also faces challenges like occasional errors in recognizing objects. Besides that, one of the users mentioned the need of audio cues, which in fact is something a visually impaired individual might need to determine what is in front of them or how far something is in front of them. We have taken that into consideration and are trying to create a solution for that issue. Other than that, this subsystem looks promising as it is one of the most important steps to creating a wearable technology for visually impaired individuals.

Updated target specifications, detailed design & BOM

Updated BOM

PROTOTYPE 1 - Bill of Materials					
Item #	Item Name	Quantity	Purpose and Description	Price	Amount
1	Paper	1	1. To list all features and functions for the smart glasses 2. To testing the User Experience, by creating a paper user interface. Each button will navigate to a different paper copy of a concept for example audio. This will help us brain map the navigation tree of our app, then we can apply it to software. a. (Aesthetics = low fidelity) b. (Functions = high fidelity)	\$0.00	\$0.00
2	Colored Markers	Max 5	To add color, to show the color palette concepts of our app.	\$0.00	\$0.00
3	Figma	1	To create the pages our OMEGA platform has to offer and accurate navigation. a. (Aesthetics and Navigation = High Fidelity)	\$0.00	\$0.00
Total product cost				\$0.00	\$0.00

PROTOTYPE 2 - Bill of Materials

Item #	Item Name	Quantity	Purpose and Description	Price	Amount
1	Reserved API Calls	TBD	Using APIs outside of the SHABODI sandbox. High quality APIs have a cost per call. (1 call < 1\$)	\$5.00	\$5.00
2	API - Ultralytics yolo v8	1	An API for object detection	\$0.00	\$0.00
3	SpeechRecognition	1	Machine's ability to listen and identify words. For voice control.	\$0.00	\$0.00
4	Pytsx3	1	Text to speech conversion. To read text.	\$0.00	\$0.00
5	PyAudio	1	To play and record audio.	\$0.00	\$0.00
7	OpenCV-python	1	Image processing, machine learning. To identify objects and faces.	\$0.00	\$0.00
8	PyWhatKit	1	Answering user-specific questions from trusted sources	\$0.00	\$0.00
9	Phone Camera	1	Connecting our software to a phone camera. Camera is an input source for the software.	\$0.00	\$0.00
Total product cost				\$5.00	\$5.00

PROTOTYPE 3 - Bill of Materials					
Item #	Item Name	Quantity	Purpose and Description	Price	Amount
1	Figma	1	This software will help us create prototype images of outputs our software will give to the user based on the object recognized. We will create scenario-based images for example if	\$0.00	\$0.00

			this object is recognized, our user will see this		
2	Visual Studio Code	1	To code all the APIs, libraries, and Figma. Into one final product.	\$0.00	\$0.00
	TBD				\$5.00
Total product cost					\$0.00

DESIGN DAY- Bill of Materials					
Item #	Item Name	Quantity	Purpose and Description	Price	Amount
1	VR headset	1	To run our code and visualize the output	\$0.00	\$0.00
2	Glasses	1	A prop to display during design day to give an idea of what the HUD would look and feel like.	\$0.00	\$0.00
3	Trifold	2	To present our project	\$3.00	\$6.00
4	Karaoke machine	1	Has a speaker and microphone: To enhance volume in the crowd	\$0.00	\$0.00
5	TV	1	To always display an advertisement video and attract attention	\$0.00	\$0.00
6	Laptop 1	1		\$0.00	\$0.00
7	Extension cord	1	To connect to the TV	\$0.00	\$0.00
8	Candy (caramel) Bowl	1	To attract attention and reward audience members who ask about us	\$2.00	\$2.00
	TBD				
Total product cost					\$10.00

BUDGET	
PROT 1	\$0.00
PROT 2	\$5.00
PROT 3	\$5.00
Design Day	\$15.00
Total	\$25.00

Updated Target Specifications

ID	Design specifications	Relation	Value	Units	Verification method
1	Face detection reliability	=, >	Pass, 1m	Boolean, Meters	Tracking different faces from different distances away from the camera and keeping track of the distance, it cuts off.
2	Speech Recognition	=	Pass	Boolean	Testing different voices and keeping a note on how many times the voice assistant responds when called.
3	Object recognition reliability	>	80%	Percentage	Testing different objects and seeing whether the program recognizes them correctly.
4	Software and device compatibility	>	Many	Devices and software	Test and analysis
5	Bandwidth	>	50-100	Mbps	Analysis
6	Low Latency	<	20	ms	Analysis
7	Processing power	>	1.6-2.5	GHz	Analysis
8	Network connection	=	5	Gs	Test
9	Display resolution	=	1080	pixels	Analysis
10	Obstacle Detection	=	Yes	Boolean	Test
11	Object recognition	=	Yes	Boolean	Test
12	Facial Detection	=	Yes	Boolean	Test

13	Plans Routes	=	Yes	Boolean	Test
14	Knows current location	=	Yes	Boolean	Test
15	Voice control	=	Yes	Boolean	Test
16	Functional microphone	<	100	dB	Test
17	High camera quality	>	12	MPX	Analysis
18	Audio warnings	=	Yes	Boolean	Test
20	Informs user of system errors	=	Yes	Boolean	Test
21	Audio instructions	=	Yes	Boolean	Test

Prototyping test plan for prototype 3

Test ID	Objective	Description of test methods and materials needed	Description of results to be recorded	Estimated test duration	Stopping Criterion
1	For the final (to be decided) prototype, we would like to test how all the subsystems would work together if they were combined to one code. This will allow us to debug any issues with the software as well as make any changes which are necessary to make the final product better.	For this test, we need access to a good high-end computer since there is a chance the code might end up being soft-ware heavy, requiring more processing power in general. We will be using our team members' faces as well as multiple objects to test the object recognition and the face detection.	For the main part, we will record how well each subsystem has adapted itself to the final code. This could be done by for example, asking the voice assistant to start object recognition and take control.	Depends on if we are happy enough with our results and data but realistically, we would aim for at least 2 hours of raw testing and gathering the data.	We will continue our tests until its passes 80 percent of the time. For the compatibility of our codes

Conclusion

One of the most crucial steps in the design thinking process is prototyping. Prototyping is essential since it brings ideas to life and allows more thorough testing. In the last deliverable, prototype 1, we constructed our first prototype and specified all its crucial components. In this deliverable, we created our second prototype, analyzed all its critical components, mostly made of codes for each subsystem, established our prototype test plan, and gathered input and comments to improve this prototype for the final one. Finally, we developed the prototyping test plan for the final prototype. We will conclude prototyping test plan in the next deliverable, prototype 3. In that prototype, we are planning to join every system of this code to one joint prototype. In that prototype, we will create the final version of our application.