Deliverable F

Group Members: Cadence Greer, Adeife Olomola, Westley Martin-Root, Charla McEachran Morin

Prototype #1: Software Logic Testing

Goal of prototype

Determine if our system's logic is sound. I.e., would it do what we want when we want?

Type of prototype

Low fidelity comprehensive

Method

We want to simulate the input the code would receive and then compare the given output against the desired output. To do this, we simulated potential users by blindfolding our participants. Then, we asked the user to navigate to the local milkshake spot (For God Shakes). We then analyzed their actions based on the logic paths we had drawn (attached below). Based on these actions, we would simulate the software's response real-time.

In this test, we made a few assumptions: first, we assumed that during a motion event, the glasses would be helping the user navigate. To simulate this, when a motion event started in our logic, we had someone offer their arm to the user to help them navigate. This is a fair assumption to make because even though our software does not have any actual video analysis functionality at this point, the purpose of our software is to stream the video feedback to the network where it can then be analyzed as any client may desire. If these glasses are made to help the visually impaired navigate, we can then assume that any network applications that Shabodi may develop will be for this purpose also.

We simulated the location API by having a person walk beside the participant holding a measuring tape stuck out 0.5m (motion event starting radius). When the participant stopped, the location API would stop as well. Then, when the participant moved again, the location API would not move unless the participant moved outside of the 0.5m radius, signalling the start of a motion event.

If the participant successfully made it to the milkshake store without encountering any errors in the logic, then the logic would pass. If the participant encountered any errors, regardless of whether they made it to the destination, then the logic would fail this test.

Test Participant	**Results of Test (Pass/Fail)**	**If fail, why?**
Noa	Fail	This participant often started and stopped. This exposed an important gap in the code's logic: a motion event starts if a person moves 0.5m. However, we did not clearly define the

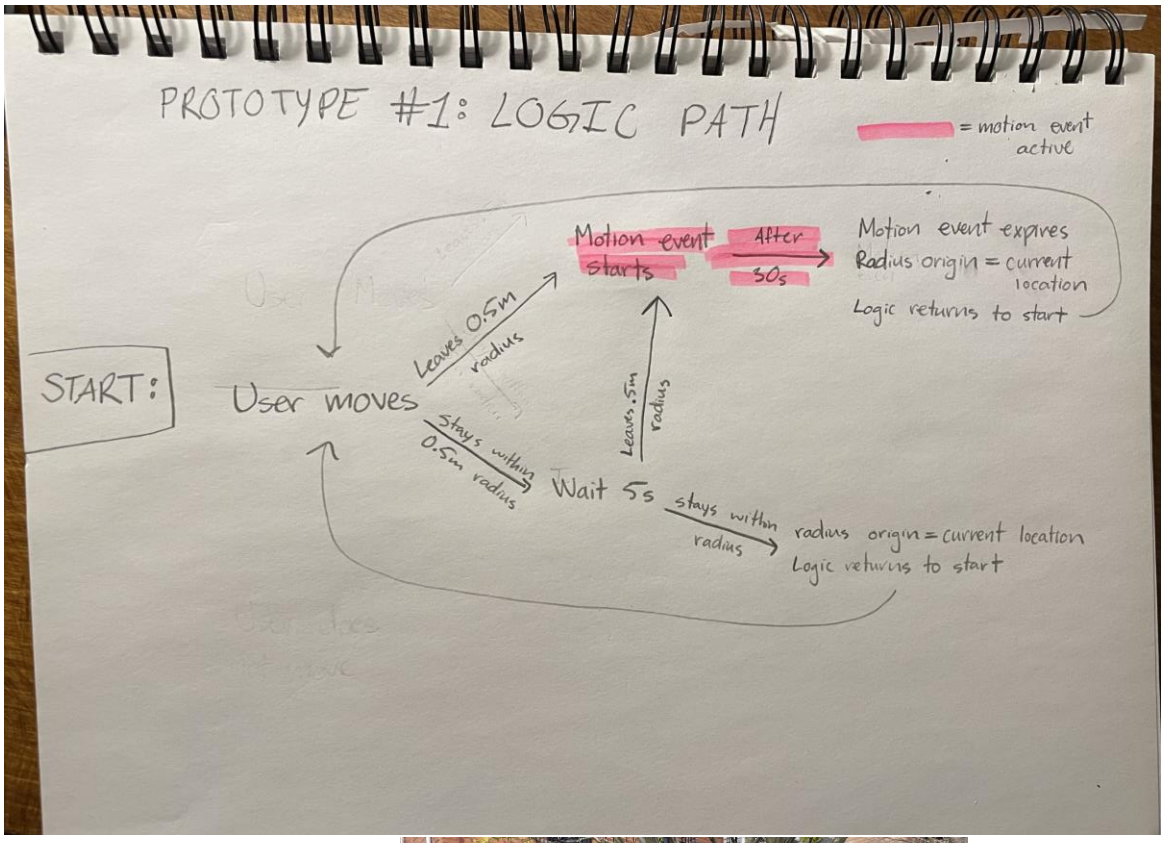
		starting point to measure this displacement from. This meant as the participant meandered around, the starting point would move with the participant and the motion events would not start.
Amy	Pass	
Maguire	Fail	This participant would stop every time a motion event expired and had trouble moving enough to start a new one. However, this is likely an artifact of the prototype. This is because the reason she could not move is because she found it scary/difficult to walk around without being able to see. However, the users for this product are not completely blind and are also more accustomed to moving around with their level of visual impairment.

Results of test prototype:

Noa: During the test, Noa exhibited a behavior of frequent starting and stopping, which uncovered a significant gap in the code's logic. Specifically, the prototype was programmed to initiate a motion event when a person moved 0.5 meters, but it lacked a clear definition of where to measure this displacement from. As a result, when Noa shifted position, the starting point for measuring distance moved with them. This caused the system to fail to trigger motion events, highlighting the need for a fixed initial position in the code to accurately measure displacement.

Amy: The test was successful. No obvious issues.

Maguire: Maguire's experience with the prototype revealed a unique challenge: she had difficulty moving sufficiently to trigger new motion events after stopping. This issue stemmed from her hesitation, as she found it daunting to move around without visual input. However, this limitation is likely related to the testing conditions rather than the prototype's fundamental design. The intended users of this product are not completely blind and generally have some level of visual ability, making them more familiar with navigating their environment despite their impairment.


PROTOTYPE #1: LOGIC PATH
= motion event active
START:
User moves
Leaves 0.5m radius
Motion event starts
After 30s
Motion event expires
Radius origin = current location
Logic returns to start
Leaves 5m radius
Stays within 0.5m radius
Wait 5s
stays within radius
radius origin = current location
Logic returns to start

Critical Components

Component	Reasoning	Contingency
Network Connection	If the glasses are disconnected from the network, they will no longer be able to process the data from the glasses and the user. This will render the glasses useless.	Set up a warning in the code to warn the user if they are leaving a network zone or changing networks.
Object detection	If the object detection fails, then the failsafe that comes from the object detection will not be able to help the user. This acts as a higher resolution navigation as it can tell if there are obstacles that can't be detected with location.	There could be a notification that the object detection is no longer working with the glasses. The application could allot more bandwidth to the object detection than needed.
Location API	If the location API fails, then there is a safety aspect. Since part of the application functionality comes from the ability to navigate safely, if the API fails, then the user could be in danger of cars or other hazards.	Set up a warning if the API pulls fail. If this happens, the user can know if there is a risk to their safety. Otherwise, there could be a saved database for the application to safe the navigation path. This would allow the user to navigate to a safe position while their glasses are recalibrating.

Updates Target Specifications

Target Specifications:

- **Object Detection and Recognition**:
  - Real-time detection of objects in the user's surroundings (e.g., obstacles, walls).
  - Recognize common objects like doors, chairs, vehicles, etc.
  - Have a radius of detection radius of minimum 1 m. This will provide the buffer for the navigation API. The API is accurate to within 30 cm so this will help the application provide proper information to the user.
- **Energy Efficiency:**
  - Reduce the amount of energy so user can use the glasses for a long period of time.
- **Device Compatibility:**
  - Make it able to connect to different platforms such as; iPhone, android, Nokia for device integration (if we pursue with this idea).
- **Should have a relatively high API rate usage:**

    - Try to use the minimum number of APIs possible (effectively) in order to make the application function. At this point that will include 2 APIs
- **Navigation assistance:**
    - In case user is walking through a flight of stairs, glasses give user step by step instructions.
    - Allow the user to use the glasses as a navigating agent through town etc.

Test #1 – Design Concept: Location API Accuracy and Constraints

Reason for Prototype	Communication
Evaluation Criteria/Determine Measurables	We are testing the APIs ability to work with our code/our codes ability to work with the API. Communication between the two to transfer information (specifically coordinates.)
Level of Prototype	High fidelity
Kind of Prototype	Analytical
Metrics	Correct information relayed: yes/no.
Test Description	Specifically, we're looking to test that the API will function in the way that we intend with the information we are providing to it. With the coordinates being passed into the API we want it to recognize the users location and be able to detect changes in that location.
Analysis Method	We will test this by inputting various values into our code which get passed to the API and recording the results. We want to simulate a users movement in the coordinates we pass to the API so we ensure the code and API are communicating accurately and we understand what the motion looks like analytically.
Notes	We will require access to the sandbox and shabodi's resources to perform this test. It may need to be pushed
Why is this the best model choice for your stated test objective?	An analytical model is the best choice for this test because "running the numbers" or running our code really is the best way to determine what is going to happen when the glasses are being used and information is being taken in and passed to the code. Checking the functionality of these pieces together is crucial to the functionality of the glasses.

Test #2 – Design Concept: Effect of Bandwidth on the Code

Reason for Prototype	Performance Measurement
Evaluation Criteria/Determine Measurables	With this concept we're testing how well the code runs and can display information using different bandwidths. We want to

	increase bandwidth during motion events and decrease when not in motion.
Level of Prototype	High Fidelity Focused
Kind of Prototype	Analytical
Metrics	Speed: bits per second
Test Description	Specifically we will test the functionality of our code using varying specified bandwidth values to simulate the experience the user would have. In specifying the bandwidth we will be able to control this variable during different events in the code.
Analysis Method	We will test by running the code with the specified bandwidth values or there are speed tests available online that could be utilized for the code. A number of speeds will be chosen and multiple tests will be run at each of these speeds.
Notes	We are making sure that changing the bandwidth variable still allows the code to function as we intended and provides us with satisfactory visuals.
Why is this the best model choice for your stated test objective?	An analytical model is the best choice for our test objective because simply changing variables in code is cost effective and the best way to determine what happens to the results when this unique variable is changed. In the code all variables are controlled, so the analytical model provides a controlled environment where testing can be specific and consistent.

Test #3 – Design Concept: Testing Motion Events Detection

Reason for Prototype	Communication (between device and network)
Evaluation Criteria/Determine Measurables	Testing whether motion events are detected by the device and can be handled by our code. If code is triggered by a motion event the test was successful.
Level of Prototype	High Fidelity Focused
Kind of Prototype	Analytical/Physical
Metrics	Metrics measured by a simple fail or pass. We are not measuring distance as the test, but will be using specific distances in our process.
Test Description	Specifically we are testing our code to see that it works with a device and accurately detects motion events using the location information from the location API. Using the provided APIs motion detection should be a fairly simple process and should pass the test.
Analysis Method	We will test by holding a device and walking certain distances from a machine running our code to see that the machine still recognizes motion events at varying distances.

Notes	This test overall checks the functionality of our code and the feasibility of using this code in a pair of smart glasses.
Why is this the best model choice for your stated test objective?	An analytical model is the best choice for our test objective because it will provide useful feedback on whether or not the code written will be effective in completing our goal of guiding the user. Testing values in the code has little repercussion. The physical portion of this model includes the device that we will use to communicate with the code, in tandem with the analytical bit of the model this physical piece can feed the code information from a real-world demonstration rather than an entirely simulated one.