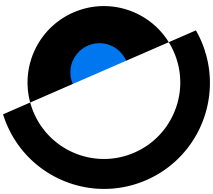
NASA SUITS 2023 RETROSPECTIVE

Team AEGIS



University of Southern California

3650 McClintock Ave Los Angeles, CA 90089

University of California, Berkeley

Berkeley, CA 94720

Team Contact

Spencer Lin

linspenc@usc.edu

+1(626) 492-2828

Faculty Advisors

USC	UC Berkeley
David Barnhart	Allen Y. Yang
barnhart@isi.edu	yang@eecs.berkeley.edu
+1 (310) 448-8644	+1 (510) 643-5798

Team Members

Rianne Aguas — Project Management — raguas@usc.edu

Junior/Communication, Web Development and Design

Sophia Aguilar — Unity Lead, Human-In-The-Loop — sophiaa@usc.edu

Sophomore/Electrical and Computer Engineering

Andy Artze — Systems Design — artze@usc.edu

M.Sc/Astronautical Engineering

Subham Banga — Artificial Intelligence — sbanga@usc.edu

M.Sc/Computer Sc.

Michele Chung — Project Management — mjchung@usc.edu

Senior/Industrial and Systems Engineering

Athang Gupte — Artificial Intelligence — aagupte@usc.edu

M.Sc/Computer Sc.

Miru Jun — Unity Developer — mdjun@usc.edu

Junior/Computer Sc. major, 3D Animation minor

Jennifer Lee — Unity Developer — jlee3900@usc.edu

Senior/Computer Sc. and Applied Math

Spencer Lin — Team Lead, Navigation, Artificial Intelligence, UI — linspenc@usc.edu

Junior/Computer Sc. major, Immersive Media minor

Stanley Lin — Marketing — linstanl@usc.edu

Senior/Business Administration (Marketing) & Cinema & Media Studies

Rohit Penumarti — Artificial Intelligence — penumart@usc.edu

M.Sc/Electrical Eng. Machine Learning & Data Sc.

Basem Rizk — Artificial Intelligence Lead — brizk@usc.edu

M.Sc/Computer Sc. specializing in AI

Karen Shieh — Unity Developer — karen93@berkeley.edu

Freshman/Computer Sc.

Lawrence Shieh — Artificial Intelligence, Unity Developer — lawrence92@berkeley.edu

Sophomore/Computer Sc. & Data Sc.

Prashant Singh — Artificial Intelligence — singhpra@usc.edu

M.Sc/Computer Sc.

Rohan Shukla — Unity Developer — rsshukla@usc.edu

M.Sc/Computer Sc.

Caitlín Sullivan — User Interface/User Experience — ccsulliv@usc.edu

Freshman/Electrical and Computer Engineering

Akshita Swaminathan — Human-In-The-Loop, Outreach — as76496@usc.edu

M.Sc/Astronautical Engineering

Darlene Villicana — Human-In-The-Loop, Human Factors Lead — dvillica@usc.edu

M.A/Visual Anthropology

Team members are ordered alphabetically

ABSTRACT

The second generation of the Surface Exploration and Navigation Visual Assistant system, SENVA 2, was developed through the collaborative efforts of the University of Southern California (USC) and the University of California Berkeley for the 2023 NASA Spacesuit User Interface Technologies for Students (SUITS) Challenge.

SENVA 2 boasts several features that are specifically designed to aid astronauts during exploration EVA operations:

1. The hand-based UI is designed to be unimposing to an astronaut by maintaining a clear field of view when not interacting with the menus, while separate floating panels anchored to world allows the astronaut to naturally interact with the information displays which contain suit, biometric, and spectrometry data.

2. The Navigation subsystem consists of two main modules: Short-Range and Long-Range. Short-Range analyses the astronaut's immediate surroundings, and Long-Range facilitates a Pathfinder system to dynamically generate a 'breadcrumb' trail to waypoints the astronaut places on a deployable 3D map. Based on custom A* heuristics, the Pathfinder system optimally minimizes distance, effort, and hazards to safely guide the astronaut to their destination.

3. Rover controls are also built into the deployable 3D map for intuitive navigation and recall commands. Flags on the deployable map denote the sites of interest relative to the localized astronaut location on the map. Using the flags as guides, the astronaut can place a Rover waypoint on the map or through voice commands to navigate the Rover to the location.

4. The AI companion, which responds to the name 'Traveller', is an off-cloud digital assistant that can run on a laptop server. Once connected through a websocket, the astronaut can ask Traveller to open or close panels, read out current vitals information, add waypoints with eye gaze, and more. For the UIA egress procedure, a computer vision module detects and overlays instructions onto the UIA panel, while Traveller reads out the next steps.

Many features from last year's SUITS Challenge were polished, and many more were developed. Through a post-mortem analysis, future avenues of improvement have been outlined for the 2024 NASA SUITS Challenge.

ACKNOWLEDGMENTS

We would like to express our deepest appreciation for our advisors Professor David A. Barnhart for his continuous support throughout our team's journey for a second year, Dr. Allen Yang for his mentorship, encouragement, and expertise, former astronaut Dr. Garrett Reisman for his support and expert guidance, and to our NASA advisor Dr. Timothy M. McGrath for his expert guidance and unwavering support. We would also like to extend our gratitude to Adam Peterson for his assistance throughout our participation in the NASA SUITS program. We gratefully acknowledge the assistance of our NASA advisor Dr. Timothy M. McGrath, and our testers for testing S.E.N.V.A. 2 on-site in Houston and for their valuable feedback. Additionally, we want to thank Paromita Mitra for testing our digital assistant and computer vision at the final hour and for her encouragement and advice.
We would also like to acknowledge the support of the California Space Grant Consortium, USC Viterbi School of Engineering, and UC Berkeley College of Engineering for making it possible for our team to travel to the Johnson Space Center for test week. Lastly, we thank our friends and families for their support throughout this project.

OBJECTIVES

Create an innovative spacesuit user interface (UI) leveraging AR (and artificial intelligence) that demonstrates how AR can be used in a crewmember's display during EVA operations in a non-obtrusive way.

Requirements:

1. Design should work in harsh lighting and outdoor conditions.
2. The device must be able to access and display the telemetry stream data.
3. A caution and warning system must be implemented to inform the astronaut about a spacesuit anomaly.
4. In case of an interruption, the astronaut must be able to continue to task on hand seamlessly.
5. The UI shall assist the astronaut during navigation.
6. The UI shall display necessary EVA task instructions.
7. The UI design should not distract the user or crow their field of view with non-critical information.

TABLE OF CONTENTS

Chapter 1

Software and Hardware Design Description

1.1 User Interface (UI)

SENVA 2's user interface focuses on minimizing distractions and streamlining information display methods to facilitate a cleaner, more comprehensive experience. In order to achieve this goal, the user interface took on a dynamic, hand-based design with only one element permanently visible: an AI assistant microphone indicator in the bottom corner of the field of view.

Figure 1.1: Display with no menus opened. Short range navigation markers are visible on the ground.

Each palm has its own menu attached to it with different information available. The main menu is shown on the non-dominant hand in order to allow the dominant hand to interact with the busier of the two menus. The main menu contains toggles to open and close panels containing information on telemetry, AI integration, and configuration settings. Navigation, being the most extensive of SENVA's features, has its own menu on the opposite hand. All panels are accessible through these two menus, and open up freely in the field of view rather than on the hands. Thus, the user can use either hand to interact with a panel and its content. The dynamic panel display enables multiple displays at once and prevents overlapping by configuring the location of the panels into a circle that surrounds the user. If more than one panel is opened at once, the second panel to open will slide to the side of the one already open so all information is always on display and the user need not manually organize them.

Figure 1.2: Main menu is tethered to the non-dominant left hand.

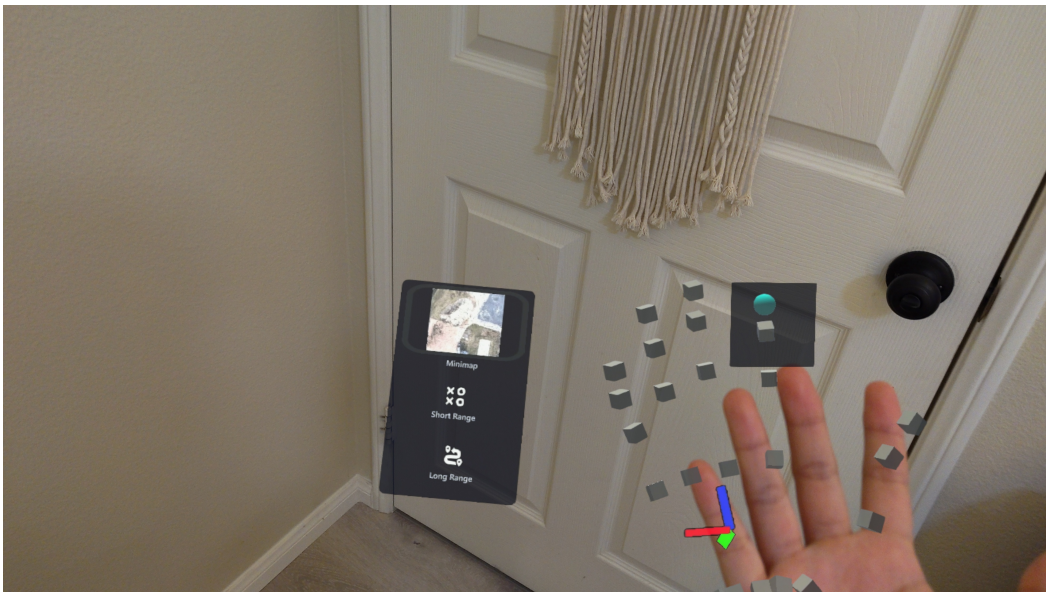


Figure 1.3: Navigation menu is tethered to the dominant right hand.

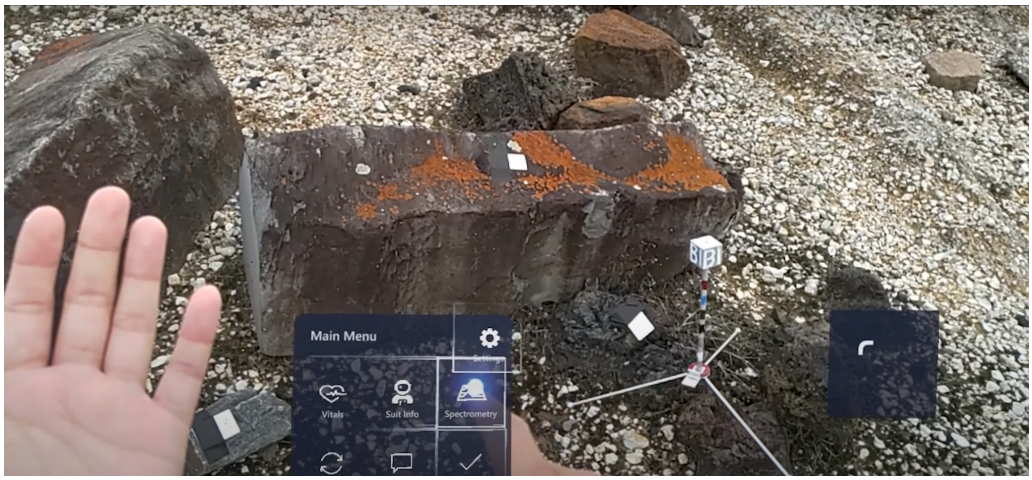


Figure 1.4: Main menu shown on the left hand being interacted with by the right hand.

Figure 1.5: Multiple panels opened in space at the same time are spaced out and rotated to face the user.

The navigation menu contains a quick-reference minimap and two menus that enable Short-Range and Long-Range Navigation. The terrain markers of the Short-Range Navigation feature are adjustable in the application's settings on the main menu, as to not crowd the enable button with information not often accessed. Additionally, the Long-Range navigation menu button opens up both the high-quality 3D map with pins reflecting real-world points of interest and a panel anchored to the map but detached from the hands which includes prompts to begin pathfinding for the user or the rover and place new pins on the map which will show up in the user's field of view in its associated real-world location.

Figure 1.6: The 3D map is displayed near the floor when opened. A palette of settings is tethered to it for the user to place their own pins or begin navigation to a location on the map.

1.2 User Experience (UX)

Safety is a priority for our configurability and accessibility-focused UX concept. To combine easy access to warnings from the telemetry stream with the UI's simplified main menu design, telemetry data that needs the user's immediate attention is presented as an alert with a distinct sound effect. These alerts appear within the user's FOV in a static position with different levels of priority distinguished by color and alert sound. The warnings are hidden after 5 seconds and may be reviewable later on either wrist. These warnings also have the capability to be read out by text-to-speech and to be dismissed via voice command.

Unfortunately, due to time constraints, configuration and accessibility options such as handedness and variability were not able to be implemented as originally planned in the proposal. Thus, in future

iterations, more human variability options would be employed, such as handedness switches. Although options such as font size and contrast were considered in the proposal, through development it was decided that adding settings for these were redundant due to the nature of AR panels - size is defined relative to the panel, and panel size is best changed by moving it in space rather than scaling. Thus, one can move closer to the panel improve readability if need be.

Figure 1.7: Two panels float in real-world space and have varying distances to the user.

Based on feedback from human interaction mentors during SENVA 2's test week at the Johnson Space Center in May 2023, the palm-based menu design may be changed in favor of a minimally-invasive docked UI placed above the user's head in future iterations. Due to the heftiness of a spacesuit and the physical strain of turning and moving, needing to bring one's hand up towards one's face in order to see the menu and hide it again would become quite taxing over time. Additionally, when tethering a menu to one hand, the user does not just lose that hand for other activities while interacting with the menu, but loses both hands as the other hand interacts with buttons. In space, where efficiency is paramount and time is limited, this could pose greater risk to mission success. Given this, a docked UI which better utilizes gaze instead of touch may be more efficient; a single panel containing each palm menu option displayed at the top of the field of view to be interacted with via gaze and voice commands only, for example, could facilitate this. All main components of the hand menus would be displayed there with no need to use either hand to select options or open panels. Panels would still be the primary mode of displaying information, but these need only one hand at a time and can be put to the side while both hands are needed during the mission.

1.3 Navigation

The Navigation subsystem is responsible for collecting, compiling, processing, and displaying navigational data in an accessible format for the astronaut during an EVA. This subsystem is further compartmentalized into two distinct modules that can either work independently or in tandem: Short-Range and Long-Range.

1.3.1 Short-Range

The Short-Range module displays terrain markers in a five meter deep, 52 degree cone (maximum field of view of the HoloLens 2) in front of the user. A grid of green circles and red crosses is generated by raycasting and calculating the normal of the ground mesh generated by the HoloLens 2. Hazardous areas which normals exceed a threshold of 5 degrees are demarcated with red crosses where as green circles are placed over flat areas. Accessibility and user preference were taken into account by adding

configurations to the Short-Range terrain module. Building upon last year's feedback, we added configurations for marker count, size, and transparency. We also implemented "near clip" and "far clip" configurations as a means to modify the range that the terrain markers start and end. However, after careful consideration, we decided the clip configurations to prevent overloading the astronaut with options. Human-in-the-loop testing was done and determined simple addition and subtraction buttons were more intuitive as opposed to pinch sliders due to an astronaut's limited mobility in astronaut gloves.

For the future, we're looking towards methods of leveraging a lightweight computer vision package for edge detection or moving away from the current grid-based system to explicitly identify hazards. Solely highlighting individual obstacles and terrain gradients allows us to declutter the user's display while the hazard detection overlay is in use. Highlighting purely pertinent hazards would maintain the practicality of the feature while not distracting the astronaut with terrain markers on known safe areas. Furthermore, limitations of raycasting include the fact that raycasting cannot detect vertical and near vertical surfaces such as walls or the sides of ledges due to the rays being casted from directly above. These limitations can be circumvented by implementing variant methods of hazard detection.

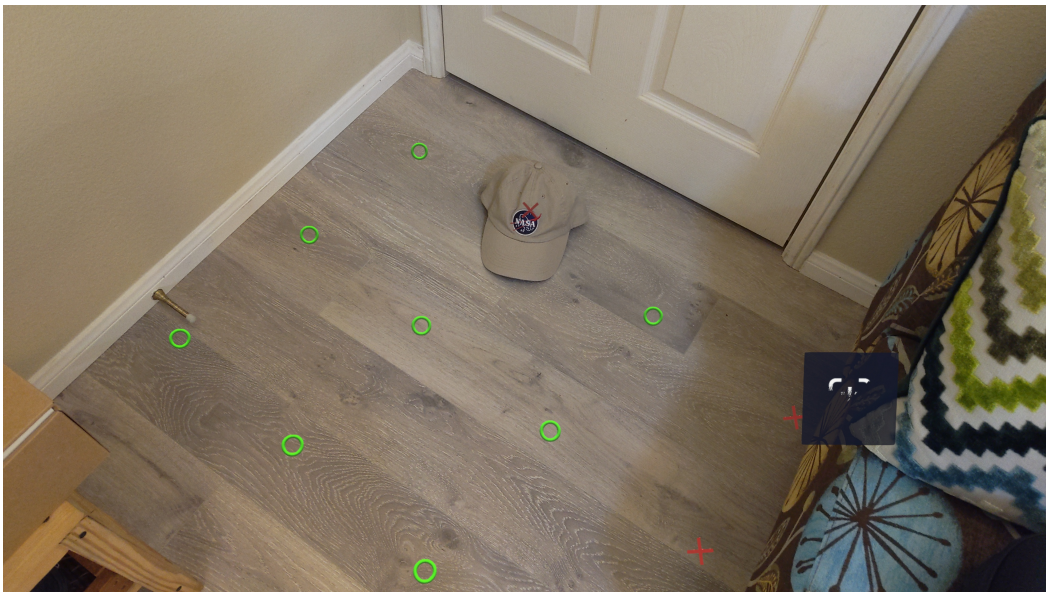


Figure 1.8: On the ground, the short-range terrain analysis feature displays circles and crosses to mark safe areas and potentially hazardous areas.

Figure 1.9: The settings panel contains configurations for modifying the terrain markers for the short-range terrain analysis feature.

1.3.2 Long-Range

The Long-Range Navigation module is responsible for navigating users across large distances, potentially beyond line of sight. Long-Range Navigation interactions occur on a locally-saved holographic digital terrain model (DTM) of the test site extracted from Google Earth.

1.3.3 DTM

The DTM is a deployable 3D model that displays the locations of the user, rover, breadcrumb trail, challenge checkpoints and user-defined waypoints. Users may place waypoints and pathfinding destinations on the DTM using voice commands, a point and pinch gesture, or by using their eye gaze. The DTM is also grabbable and scalable meaning users can configure the location size of the DTM.

Additional improvements could include easier manipulation of waypoints and obstacles for creation, modification, and removal. Currently, the DTM can only be translated by the user, but not rotated. Adding support for rotations is a potential feature that may be useful according to feedback from a test subject during test week.

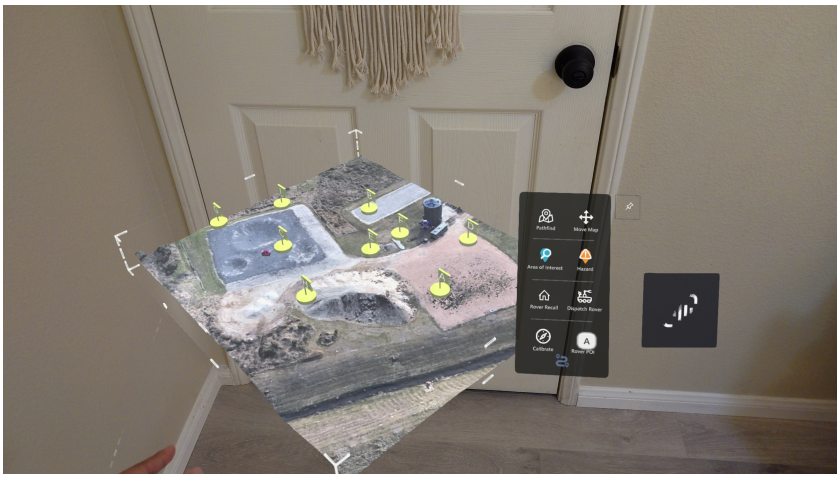


Figure 1.10: The DTM can be scaled and translated using the white handles on its bounding box after clicking the ”move” button on the Long-Range floating menu.

1.3.3.1 User Localization

A total of three coordinate systems are localized: GPS, Unity Coordinate System (UCS), and Model Coordinate System (MCS). GPS is converted to UCS and vice versa using a converter library[1]. Conversions from UCS to MCS and back is done through scaling within Unity. This year's user localization featured multiple improvements over last year's work [1] being much more robust and operable in regions outside of just the Houston area. This is because we are no longer relying on arbitrary hard-coded values nor are we assuming GPS to be a perfectly Cartesian system. The upgraded capability now allows for scaling and moving of the DTM as well as supporting a wide variety of additional features such as rover controls based on user-specified points rather than only a static lookup table.

Further work can be done to correct floating point errors when converting MCS to UCS to GPS coordinates. In addition, our current system relies on users setting true north by facing north due to the HoloLens 2 lacking an internal compass. We recognize this may not be feasible in circumstances where the user has no access to a compass or they are operating in an environment with no magnetic field. Thus, we will explore methodologies that allow for calibration using only reference points and linear decomposition.

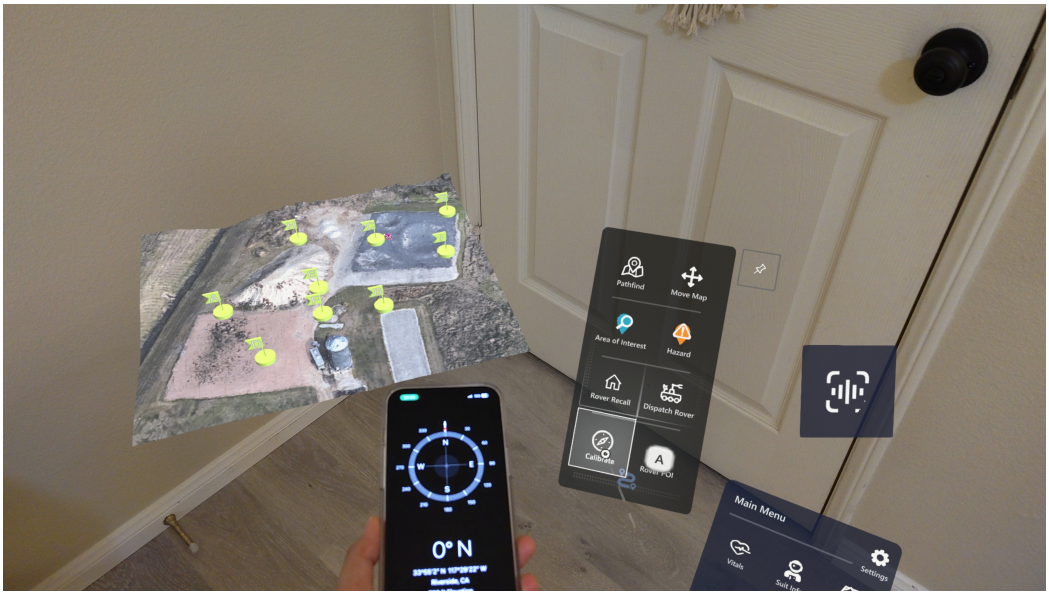


Figure 1.11: Calibration for user localization is performed by pressing the calibration button while facing true north.

1.3.4 Minimap

An orthographic camera within the Unity scene looks at the DTM to generate a 2D image for use by the Minimap. The Minimap is accessible from the right hand menu and may be expanded into its own panel.

[1] `https://github.com/MichaelTaylor3D/UnityGPSConverter`

Figure 1.12: The Minimap display when expanded into its full-sized panel

Due to time constraints, the Minimap was unable to be completely integrated with the DTM. Future work may include integrating the Minimap so that all changes on the DTM are reflected in the Minimap and with higher resolution than currently allowed in the current implementation. Undesired pixellation occurs in the current implementation which may be caused by the orthographic camera method.

1.3.4.1 Pathfinder

Our advanced Long-Range Navigation system uses a unique combination of several technologies: a Digital Terrain Mesh (DTM) of the rock yard, the A* Pathfinding Project for NavMesh generation, the Recast graph for shortest path computation, and real-time user calibration to ensure navigation accuracy.

The DTM, a representation of the bare ground surface without any objects like plants and buildings, served as the basis for our navigation system. It provided detailed terrain information which was then transformed into a Navigation Mesh (NavMesh) using the A* Pathfinding Project. This NavMesh is a data structure that enables efficient path computation.

For the pathfinding process, we employed the Recast graph system, which dynamically rasterizes the NavMesh and applies the A* algorithm to find the optimal path. The A* algorithm is a powerful tool that combines the benefits of Dijkstra's Algorithm and Greedy Best-First-Search, offering fast and memory-efficient results.

A critical part of our approach was the management and conversion of three distinct coordinate systems: the DTM's map coordinates, Unity's coordinate system, and the real-world coordinates. First, the DTM map coordinates were scaled, translated, and rotated to align with Unity's left-handed coordinate system.

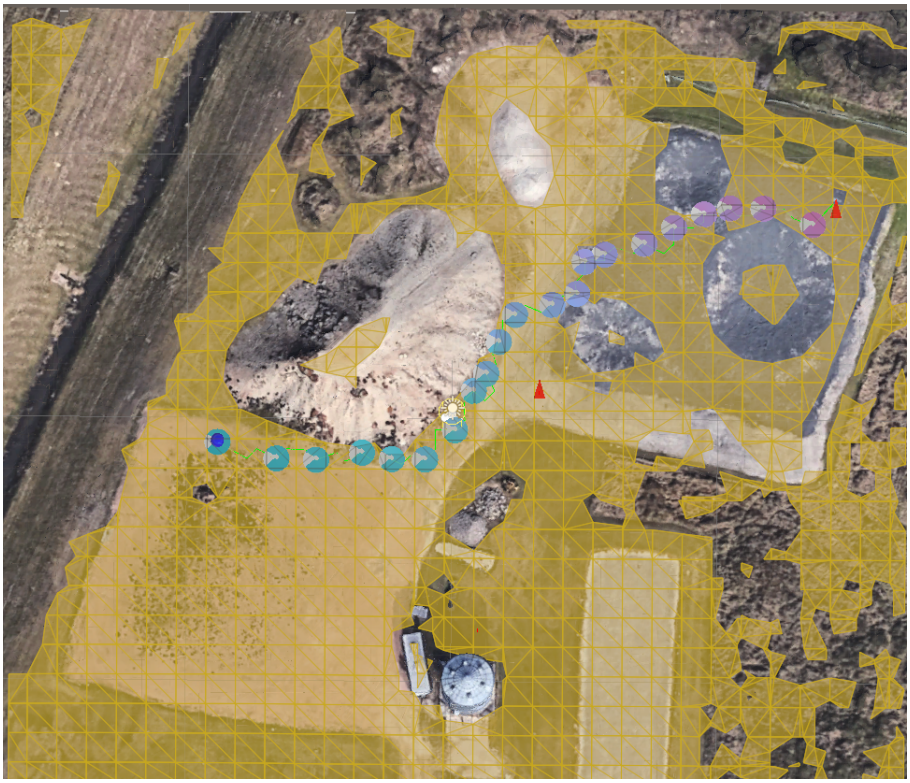


Figure 1.13: The pathfinder system generates 'breadcrumbs' on the DTM.

At this stage, to ensure accurate orientation within our virtual environment, we included a user calibration step. Here, users are asked to calibrate True North at the onset of their navigation. This process aligns the virtual environment with real-world magnetic fields and allows for precise placement of breadcrumbs into the UCS, a feature that significantly enhances the utility of our long-range navigation system.

Once the user completes the True North calibration and selects a point on the DTM to pathfind to, the pathfinder runs the A* algorithm in MCS to generate the 'breadcrumbs' on top of the DTM. The trail is then converted into UCS for viewing in the real world.

Figure 1.14: The pathfinder system generates 'breadcrumbs' translated to the real world (UCS) relative to the astronaut.

In conclusion, the synergy of these elements: DTM, NavMesh, A* pathfinding, Recast graph, and user calibration, delivers a highly capable long-range navigation system. It presents a robust and

efficient solution for navigating complex environments like rockyards, while the integration of user calibration and breadcrumb trails ensures a user-friendly and intuitive interface.

1.4 Telemetry

1.4.1 Server Connection

Connection to the TSS server was streamlined through a dropdown menu that is populated with a list of URI addresses from all previously successful connections. This list is locally stored on the HoloLens 2 in a JSON file. In the future, we can implement an auto-reconnect feature in case the websocket connection drops.

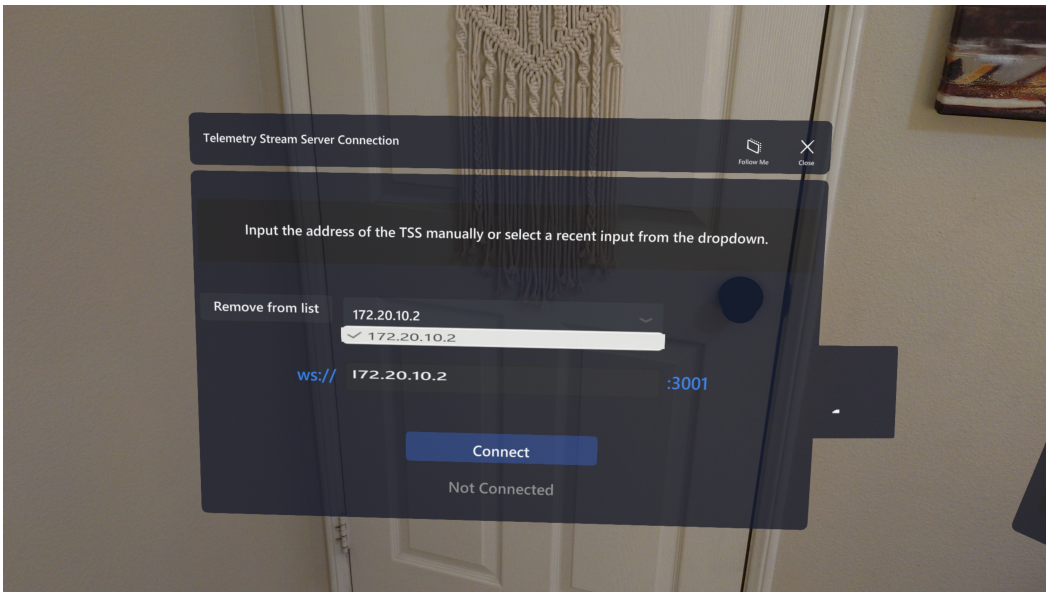


Figure 1.15: The TSS connection panel contains a dropdown menu with URI addresses.

1.4.2 Suit and Biometric data

1.4.2.1 Data display

Suit and Biometric telemetry is displayed in the Suit and Vitals panel respectively. Both panels are deployable and closeable through voice commands or the left hand menu and closeable from the panel itself. Visual elements add context to the data via elements such as radial percentage gauges and status colors in order to make the significance of the values apparent. In this version of SENVA, only the most commonly referenced values were given graphics, but this could be expanded in future iterations. Additionally, other display methods could be explored, for example, linear bars instead of radial gauges or animations to bring more attention to values above or below a safe range.

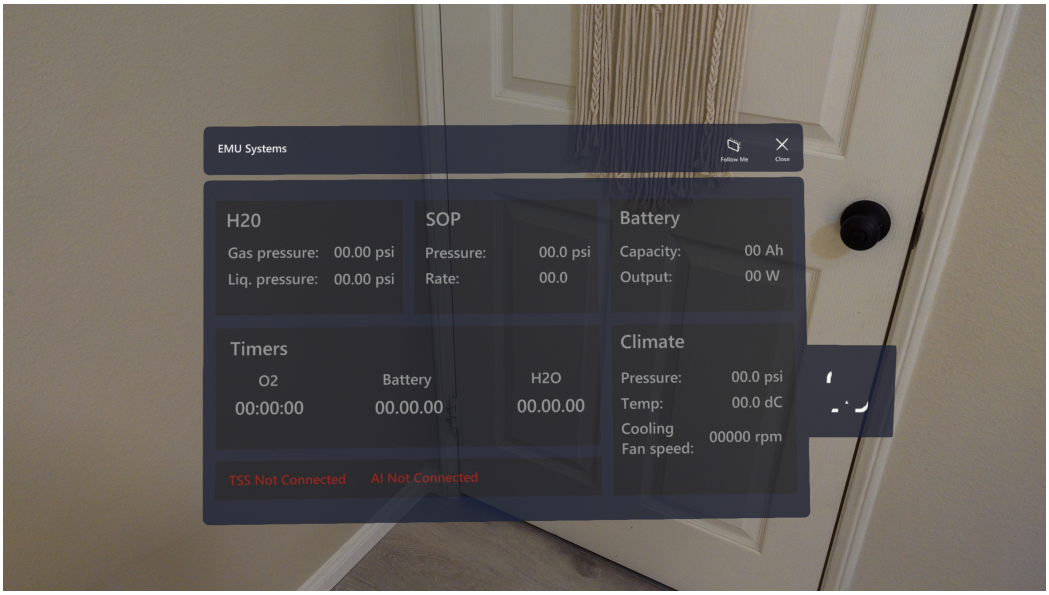


Figure 1.16: The Suit panel displays data relevant to EVA operations.

Figure 1.17: The Vitals panel displays Biometric data with radial percentage gauges.

1.4.2.2 Anomalies and warnings

Anomaly checks are done on every new message sent through the TSS using given ranges. Depending on the type of anomaly found, the anomaly checker signals the UI to simultaneously display and sound a unique repeating alarm sound to alert the astronaut. A more in-depth explanation of the design of the warnings can be found in the UX section 1.2.

1.4.3 Telemetry Logging

A local "blackbox" logging system was implemented, as well as logging over the AI server for information parsed from the VISION kit such as EVA, GPS, IMU, Rover, spectrometry, and AI companion data. The new information is appended each update frame to a locally saved JSON file specific to that session on the HoloLens 2. The session logs can be found using the naming convention that this example follows: "EVA_Log_2023-06-18_17-46-47.json".

For data from the AI companion, the user's speech and the recognized commands are sent through the websocket connection with the AI server, as managed by the Storage Manager defined in 1.5.1

Structure. All other VISION kit data was also transmitted to and stored on the AI server for redundancy. This ensures no single point of failure for logs that would be used for post-analysis.

1.4.4 Rover

1.4.4.1 Rover Data

The SensorManager polls the rover GPS coordinates via the telemetry server periodically to log the rover positions for future analysis. The Localization script uses this data to plot the current position of the rover at any given point in time.

1.4.4.2 Rover Controls

The astronaut must be able to summon the rover back to their location or ask the rover to travel to a particular location. The following three methods may be used to achieve these goals:

1. **Static Lookup Table**: The user may select the rover's sites of interest from a predetermined list of GPS coordinates provided by NASA. The locations of these stations may be visualized on the DTM. Users may use the Send Rover button to send the rover to a destination on the lookup table. The user may also summon the rover to their location using the Return Home function.

2. **Voice Inputs**: Traveller would identify voice instructions and the specified site of interest. The sites of interest are mapped to the phonetic alphabet to enable better voice recognition. Another voice command will be able to recall the rover to the user's location.

3. **Waypoint on DTM**: The astronaut may simply access the deployable map to place a waypoint. Internally, the system converts the waypoint position to GPS coordinates and sends them to the Rover. Currently, however, there are accumulated calculation errors when converting the position of the Rover waypoint to GPS coordinates. Thus, there is future work to do for eliminating floating point errors to send accurate GPS coordinates to the Rover.

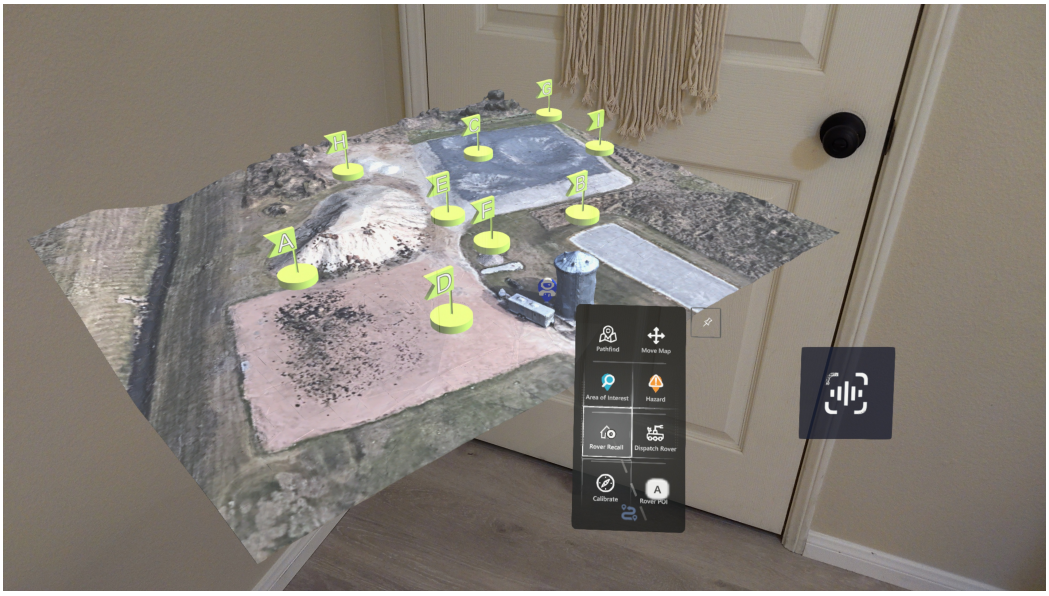


Figure 1.18: The DTM has sites of interest depicted as flags, and the Long-Range menu has dedicated buttons for navigating Rover to location and for recalling the rover to the user's location.

1.4.5 Spectrometer

A 2D spectrometry bar graph is updated upon receiving spectrometry data from the TSS. This graph panel is accessible from the left hand menu and can be deployed or closed either by voice commands, the left hand menu, or the panel itself. Currently, only the last received data is displayed by our graph. Further work can be done to allow for multiple datasets to be saved and displayed.

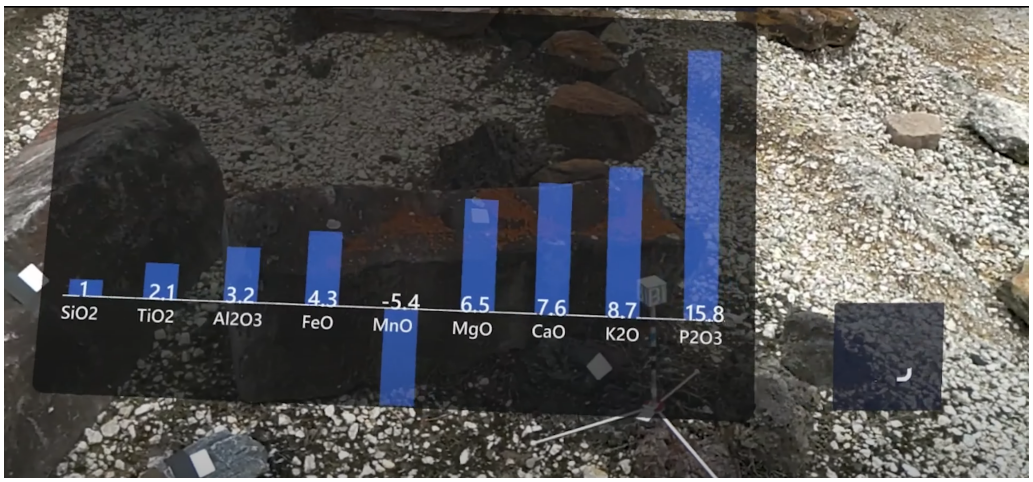


Figure 1.19: The spectrometry data is dynamically graphed on the panel when new data is received.

1.5 AI Companion (Traveller)

Motivated to distribute the computational load, and experiment with extended capabilities, we have developed a scalable standalone framework for what we call an AI-Server or a Digital-Assistant. Our assistant utilizes an array of open-source libraries and pre-trained models. A client, such as in our case the Microsoft HoloLens 2, can communicate in real-time with the digital assistant via speech, text, and/or vision (camera stream).

1.5.1 Structure

Our Digital-Assistant runs on top of an off-cloud server implementation hosted on a laptop with a GTX 1050 Ti GPU. We build on our existing base code [1] from last year's NASA SUITS challenge. The implementation of the Assistant/Server can be broken down into components as follows:

- **Speech-to-Text (STT) Controller** is responsible for transcribing the speech over the incoming audio stream. It is broken down further into a low-powered speech-to-text model and a relatively significantly heavier and more accurate state-of-the-art STT model. The former instance is employed to understand a predefined wake-up (hot) word, while the latter is meant to transcribe actual commands in natural language that carry 50+ featured user intents.
- **Bot Controller** is wrapping a task-oriented dialogue management unit trained to understand the intents carried over the text of natural language. Moreover, it is capable of managing dialogues that are tied to pre-defined custom flows. These custom flows can be added through a simple framework we built. UIA Egress Procedure and Lunar Sample Tagging scenarios are examples of these custom flows/procedures we have set up; however, the system is extendable.
- **Text-to-Speech (TTS) Controller** is responsible for converting the text of natural language (i.e., Bot Controller's responses) into audio bytes representing speech that can be transmitted to a client and/or played over a speaker.
- **Visual Processor** acts as a wrapper for whichever model(s), or algorithm(s) we wish to employ to analyze a live stream of a visual channel (i.e., HoloLens 2 world-facing camera).

- **Assistant Manager** orchestrates the flow across all the previously mentioned components. Moreover, its implementation is meant to interface between these components efficiently and parallelize whenever feasible.
- **Storage Manager** acts up as a file manager dedicated to logging the session's data such as speech command requested, failed audio, the assistant-generated responses as well as a complete file of the history of the world state (aka. blackbox).
- **Memory** is a module dedicated to carrying the current as well as history of the state of the world. It is frequently called upon by the Bot Controller, particularly in cases where the state of the world might change the order or type of operations (i.e., Suits battery level in UIA Egress Procedure). Moreover, it is the source and can be viewed as a volatile version of the blackbox.

1.5.2 In Practice

Here we give a high-level view of the implementation details and flow of our assistant and how it is integrated with our client on the HoloLens 2 end:

STT. We have used Porcupine Wake-Up word engine [2] as our low-power efficient wake-up word detector, and Mozilla DeepSpeech [3] pre-trained recent acoustic model for our main Automatic Speech Recognition. Despite having access to a pre-trained and optimized scorer based on a language model that has been pre-trained by Mozilla DeepSpeech. Upon our tests on samples of our featured commands, we have observed low accuracy, particularly on the commands that carry uncommon terminologies in everyday tasks; hence we have decided to re-train a language model based on our curated list of 50+ featured intents through a defined procedure (Explained in 1.5.3.

TTS. TTS unit uses Pyttsx3 [4] for text to speech, and inflect [5] used to convert numbers in English words as necessary before conversion.

Dialogue and Featured Tasks. We have migrated this year our dialogue management and trained Rasa v3 dialogue management to understand a set of 50+ intents; we have defined responses in a dynamic scalable format:

```
{
    "text": "response if any in natural language"
    "commands": {
        "target": "object_of_action"
        "action": "type_of_action_to_be_applied"
        "additionalInfo": [
            #any # of arguments that might be relevant to the action
        ]
    }
}
```

Client-Server Protocol. The server/assistant starts a connection using *SocketIO* for real-time performance over *Flask*; accordingly, it features a set of listening and emitting events that shall be implemented by a client (i.e., the HoloLens 2) to employ the assistant.

- *stream_audio*: Listening event for stream feeds of audio bytes.

[2] https://pypi.org/project/pvporcupine/2.2.0/
[3] https://github.com/mozilla/DeepSpeech/tree/v0.9.3
[4] https://pypi.org/project/pyttsx3/2.90/
[5] https://pypi.org/project/inflect/6.0.4/

- *stream_video*: Listening event for live stream feed of video frames bytes.
- *read_tts*: Listening event for explicitly defined telemetry elements read or natural language plain text out loud.
- *stream_text*: Listening event for plain natural language text to communicate with the bot in text medium rather than the audible format in *stream_audio*.
- *update_world_state*: Listening event for changes in the world state (i.e., telemetry data).
- *wake_up*: Emitting event to acknowledge detection of the wake/hot word (i.e., *Traveller*).
- *stt_response*: Emitting event that contains a transcription (i.e., command) of speech incoming from *stream_audio* given that the wake-up word was earlier detected.
- *bot_voice*: Emitting event that contains audible response, if any, from the digital assistant in natural language.
- *bot_response*: Emitting event that contains the complete response produced out of the Bot Controller in *JSON* per our definition.

Moreover, we have implemented a client in Unity for HoloLens 2 as a prefab that employs MRTK APIs for the microphone and the camera streams, making it sufficient to be dropped in any MRTK scene. The client can be composed into a few components: a Microphone Controller that makes use of the Universal Windows Platform microphone wrapper, a *BotVoice* unit that manages incoming audio bytes containing assistant voiced responses, a Response Handler, and finally, a Client Controller hooking the former pieces together and serving as a *SocketIO v*4 client.

Response Handling. When the client issues a command to the Assistant, the client is expected to receive one or more commands in the defined *JSON* format mapped directly to a *BotResponse* object. Naturally speaking, a command will usually contain a subject, a verb, and potentially some adjectives. Therefore, a command returned from a bot response contains 3 main fields: target, action, and additional information. Each command represents a task that has to be completed, and the structure we create allows a simple translation between user intent and actions. For example, a user command "*closingthevitalspanel*" will expect RASA to generate a command with target = "*panel*", action = "*close*", and additional information = ["*vitals*"].

We allow multiple additional information associated with a single command which makes the project scalable. We foresee in future projects; there will be more complicated tasks that include multiple adjectives and descriptions that could be included in the additional information field. For example, "dropping a hazard waypoint at where I am looking" may translate to a command with target = "*waypoint*", action = "*add*", additional information = ["*gaze*","*hazard*"]. Having additional information as a dynamic list allows for adding different information as tasks become more complex.

The format of a bot response is structured so that each bot response can contain many commands. When the client processes the *BotResponse*, each command will be parsed and queued in order to be processed one at a time. We create locks to ensure no race conditions occur between each command. By setting up the *BotResponse* with multiple commands, we can break complicated tasks into multiple simpler commands. As a result, each command can be represented by a simpler subject, verb, or adjective format.

1.5.3 Task-oriented Language Model

We aimed to train our machine learning models (STT and RASA bot) to excel in recognizing speech in EVA missions (e.g., recognizing domain-specific jargon). To achieve this, we did a few quick experiments upon training some small task-oriented language models to serve as scorers for our STT. Some of these use historical EVA mission transcripts, which are available from NASA's archive.

To generate a task-oriented language model to have a custom scorer for our STT model, we employed a three-step process. Initially, we augmented a corpus of manually created voice commands using ChatGPT. Subsequently, we utilized DeepSpeech scorer generation pipeline [6] employing KenLM [7] to generate a 3-gram language model based on the augmented text command corpora. Finally, we optimized the model by running a Deespeech optimizer script [8] that found the optimal scorer hyperparameters (alpha-beta for beam searching) running in USC's High-Performance Computing Cluster [9], which greatly expedited this process, and experimentation of few optimization processes to a matter of hours.

Our curated set of commands based on +50 intents was augmented using ChatGPT. For each command, we utilized the auto-regressive power of ChatGPT to generate an average of five variations that preserved semantic meaning. These variations were then generated and screened to ensure the absence of grammatical errors or incorrect assumptions regarding acronyms. Additionally, we created other models using a selection of commands from the NASA Apollo 11 mission, extracted from official transcripts [2], to further specialize the pre-trained language models for improved accuracy in performing the designated task.

1.5.4 Features

UIA Egress Procedure Guidance using Computer Vision Tackling the problem of understanding the world view upon a request from the digital assistant to enter the UIA Egress Procedure, initially, we aimed to leverage YOLO v5 [10] based computer vision (CV) models to detect individual components of the Umbilical Interface Assembly (UIA) panel, including $waste - 1$, $waste - 2$, $supply - 2$, among others. These models, while sophisticated, faced challenges in correctly classifying similar panels for waste and supply located on opposite sides. Moreover, we tried to integrate Optical Character Recognition (OCR) systems. Despite our promising result using OCR, we decided to switch to a model to be hosted on HoloLens 2 end and not using our established visual channel pipeline for this year's channel to due to complexity of the task of project back the 2D image from the HoloLens 2 world-facing camera into the 3D world of the HoloLens 2 augmented reality space.

Consulting our advisors and carefully reviewing NASA's UIA guidelines, we saw that the placement of each panel remains consistent. Consequently, we shifted our focus to detecting the entire panel using object detection rather than detecting individual components separately. Moreover, we employed standard measurements to locate individual components. For instance, we knew pwr-1 is consistently situated on the top left side of the panel.

Our refined CV module employs Image Targets provided by Vuforia [11] to detect the position and orientation of the UIA panel. Employing Vuforia, we perform computations directly on the HoloLens 2 and accordingly map markers and arrows back to the 3D world AR space. We highlight all components and indicate arrow on the component with specified action. Firstly, it provides a fail-safe mechanism for astronauts, enabling them to spatially locate each component, even in the event of improper scaling of arrows or markers. Secondly, it instills confidence in the astronauts, ensuring their interactions with the individual components are precise and reliable.

We indicate switches and buttons with arrows to guide the astronaut through the next steps. These guided steps are based on the instruction manual provided by the SUITS 2023 challenge. To avoid obstructing the astronaut's interaction with the UIA panel, text instructions are displayed on a floating panel below their regular line of sight. The assistant simultaneously vocalizes these instructions for added clarity.

[6] https://deepspeech.readthedocs.io/en/master/Scorer.html
[7] https://github.com/kpu/kenlm
[8] https://github.com/mozilla/DeepSpeech/blob/v0.10.0-alpha.3/lm_optimizer.py
[9] https://www.carc.usc.edu/
[10] https://docs.ultralytics.com/yolov5/
[11] https://library.vuforia.com/objects/image-targets

Figure 1.20: The mockup UIA panel has holographic overlays when using the CV module with Vuforia for UIA/Egress guidance.

EyeGaze Commands There are many scenarios where we want the astronauts to operate the interface with both of their hands being occupied. Therefore, we established a few commands that would allow the astronauts to operate simple tasks. For example, an astronauts can stare at a currently open panel and says "*closepanel*" to hide it. An astronaut can also stare at the mini-map and say "*DropthehazardwaypointwhereI'mlooking*" to execute the command. By extensively using this technology, there are potentials for a whole interface that don't require hand signal inputs.

1.5.5 Challenges

MicStream instability on HoloLens 2. Ensuring a stable microphone stream using the MRTK microphone stream wrapper [12] was challenging since the stream would terminate, flicker and go down abruptly for reasons we could not identify. It might be due to giving the responsible process a low priority under high CPU usage [13].

To mitigate the issue of an in-stable microphone as much as possible, building on our work from last year, we introduced a negligible delay upon starting the microphone. We automatically forced a restart of the microphone upon the persistence of empty stream buffers for a defined period. Moreover, we forced other background applications running on HoloLens 2 are not accessing the microphone, along with the migration to the newer *SocketIO* protocol and a more stable library [14] with an auto-connection feature to lower the probability of the potential points of failure and be able to recover quickly. Furthermore, we pushed all the computation load of pre-processing the audio stream to the server/assistant end, also allowing the integration with new clients a lot easier.

Additionally, we added a microphone stream signal panel to inform the tester whether is running or down for that current instance of time. Despite being not very useful for production software, it helped us to convey the concept and debug the system for the time being.

[12] `https://github.com/reneschulte/HoloToolkit-Unity/blob/master/Assets/HoloToolkit/Input/Scripts/Microphone/MicStream.cs`

[13] `https://forum.unity.com/threads/audio-popping-at-high-cpu-usage-hololens-micstream.509265/`

[14] `https://github.com/itisnajim/SocketIOUnity`

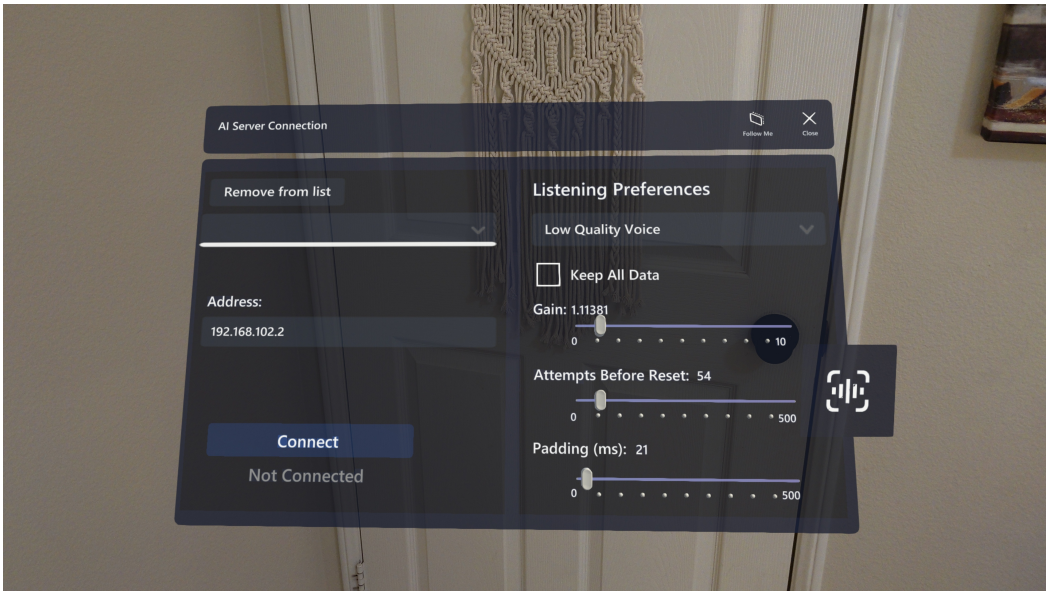


Figure 1.21: The AI panel contains configurations for HoloLens microphone settings, and the mic indicator icon is white in the bottom right corner.

1.5.6 Future work

Over UIA Egress Procedure Guidance. Looking forward, we plan to refine the integration of the notification system and consider incorporating additional information into the instruction panel. Based on user testing, we noted some confusion arising in the absence of visual or auditory feedback while astronauts awaited threshold values, such as oxygen levels, to proceed with the next step. One way is to show a menu with threshold values along with the text instructions in floating panel below their regular line of sight. By addressing these concerns, we aim to enhance our system's user interface and experience.

Affective Computing. Exploiting our designed framework we open the space of experiments incorporating tenets from Affective Computing (AC) into the AI systems. AC aims to develop AI models that are *affectively*-aware, or more plainly, that try to interpret and convey emotion (e.g. pro-activity, colloquialism, urgency, etc.) [3]. Given that Open-source libraries exist to analyze body movements [4], as well as facial, and speech signals, one may inject affective information into the Assistant context.

An AC module may be able to analyze the user's movements and changes of orientation based on the client's sensors (e.g., HoloLens' Accelerometer, Gyro, and Magnetometer from HoloLens Research Mode [15]). A relevant down-stream user case may be in situations where a slope or ridge might be too difficult to traverse, or if a user is exhausted or distracted while moving.

Life-like Communication. Further work that shall be investigated includes fine details in user experience, such as the assistant voice sound spatial perception on the different types of signals. Moreover, a more human-like synthetic voice would enhance the agent's expressions to communicate in a more intent-aware fashion (i.e., depending on the category, the agent would sound different based on whether it is a warning, regular info relay, response to UI manipulation commands, etc.).

[15] `https://learn.microsoft.com/en-us/windows/mixed-reality/develop/advanced-concepts/research-mode`

Chapter 2

Human In The Loop

2.1 Objectives

The main objectives for our Human-in-the-loop study were to meet NASA SUITS mission and technical requirements through software reliability metrics and test the user interface based on metrics of responsiveness, intuitiveness, and effectiveness through usability and workload measures.

In order to establish benchmarks for each subsystem to meet the first objective, the software was tested in low-light conditions on USC's campus. We inspected and documented whether virtual objects would appear for the various modules and if they displayed as expected within the real-world environment. Until all UI elements were stable, each subsystem was tested directly via Unity's holographic remoting on the HoloLens 2. Due to the slowness of builds, the team utilized four HoloLens units spread amongst the subteams in order to parallelize testing for different deliverables.

2.2 Human Factors Testing

Although we haven't been able to progress with the studies yet, our future approach plan includes the design of two pilot studies and one robust human factors study. For our first study, we plan to enroll 10 healthy adult volunteers, then randomize them into 1 of 3 tasks. The tasks are designed to assess the various NASA SUITS technical objectives and overall mission objectives. Task 1 would test navigation and UI visibility by asking the user to navigate to a designated location under low-light conditions. Task 2 would test our egress AI companion by asking the user to complete egress procedures using computer vision assistance and our AI companion. Task 3 would test the telemetry stream, the anomaly notification system, and the UI/UX by instructing the user to navigate and open the Life Support and EMU systems panels containing telemetry stream data.

Our second study would randomize a set of volunteers into two groups with two different abbreviated missions involving a combination of a shortened egress procedure, navigation, rover command, anomalies deployment, and spectrometry activities. For both study one and study two, we would have used the following human metrics: NASA TLX, Systems Usability Scale, Voice Usability Scale, and performance and error rate.

Lastly, our third study would be an experiment involving a robust computer vision for UIA egress. In our experiment, we would assess the system's usability via performance (completion time, accuracy, and margin of error), situational awareness (utilizing SAGAT), and workload (using NASA TLX and a secondary task) by comparing two groups. Group 1 would be asked to complete egress procedures with IVA ground simulated support while Group 2 would be asked to complete egress procedures with AR computer vision and AI support. We hypothesized that we would see a statistically significant difference in workload and performance in favor of the AI support group.

Many lessons were learned over the progress of planning and carrying out human in the loop experiments over the development cycle. Primarily, there can be improvements in communication to allow human in the loop experimentation to run in parallel with the workflow so that there can be consistent input and incorporation of human factors principles to improve overall design and usability.

Chapter 3

Project Management

Project Management will be tracked using 3 methods: 1) Gantt Charts, 2) Compliance Matrix, and 3) Peer Evaluation.

3.1 Successes

During the academic year, the project managers and team leads utilized Gantt charts and Asana to streamline our project management processes. The Gantt charts helped us plan and visualize our overarching goals, while Asana proved necessary to handle more granular objectives. Asana helped with efficiently tracking tickets, dividing work between different members, and recording the individual hours invested.

A key aspect of our approach involved regular weekly meetings between project managers and team leads. These sessions were crucial for discussing new tasks, addressing updates, and ensuring projects remained on track. We incorporated compliance matrices into our project management framework to maintain quality assurance. These matrices were used to verify that the solutions implemented by our team met the desired objectives outlined in the project scope.

3.2 Improvements for Next Year

Managing and monitoring each task alongside the team leads and ensuring consistent and timely updates on Asana posed challenges. Therefore, next year, we would like to have proactive measures to enhance our project management process by introducing a comprehensive Asana training day in which every member participates.

This year's approach did not encompass a fully implemented Agile system, but we believe that establishing this system next year will help improve efficiency and productivity. An Agile framework can help foster communication and streamline task-tracking processes.

In addition to other efforts for next year, we would like to re-introduce the idea of a Peer Feedback system. By establishing a structured feedback mechanism, team members can provide constructive criticism, share insights, and recognize others' strengths.

Chapter 4

Outreach

4.1 Outreach

With the aim to focus Team Aegis' outreach to encourage more women and children in STEM and corresponding careers, a number of events were organized in and around Los Angeles. The outreach team lead got in touch with USC student organisations and schools in Los Angeles.

1. WIE Connect 2022: Women In Engineering is an organization run by students at USC that gives academic, professional, and social opportunities to the women in Viterbi School of Engineering and organizes events to encourage young girls to pursue a career in STEM. WIE Connect is a great opportunity for mentorship, empowers women to explore engineering, and exposes students to the various paths in engineering and what being an engineer may look like. On 6 November 2022, approximately 80 female high school students came to USC to participate in STEM-related activities. Female engineers from Team Aegis volunteered at the event and conducted an interactive session with the students. The team introduced the girls to NASA SUITS Challenge and the project, explained the team's work. The activity included a brief presentation on the NASA SUITS Challenge 2022-23 and the work from last year, an interactive quiz on space-related facts, and a demonstration of the ISS using a VR Headset, Meta's Oculus Quest 2.

Figure 4.1: A student interacts with the ISS in virtual reality.

2. City of Stem: This is one of the largest STEM program held in Southern California where over 100 organizations, scientists, companies, and museum experts gather in person from all over the country to exhibit, explore and learn about the exciting innovations in science. The event, held on 1 April 2023, was open to the public and over 200 people visited Team Aegis' booth. The team was able to showcase a demo of the User Interface and also interacted with the audience and created awareness about careers in STEM and informed young minds and adults about NASA's mission, the SUITS Challenge, and what Team Aegis is working on.

Figure 4.2: A City of STEM attendee tries on a HoloLens 2.

3. Hollenbeck Middle School Visit: Members of Team Aegis visited Hollenbeck Middle School on 2 June 2023, and met with their robotics team of Grade 8 students. The target audience for this event were about 40 middle school kids graduating in a week to go to high school and it seemed like a good time to create an awareness about STEM and encourage them in learning to code and develop techology that could aid the future. Hollenbeck Middle is a public middle school in Los Angeles, CA. This outreach event was a great opportunity to conduct a seminar on augmented reality and space exploration and demonstrated the software to the students on HoloLens 2 unit along with all other hardware the team had available in an effort to engage the students in the highest, most hands-on degree possible. Students have the ability to operate the HUD and they came out of this session with a working knowledge on how the system works, the technical/social benefits of augmented reality, and space travel in the near future.

We plan to continue our outreach mission and expand to more communities and schools in the future. We were unable to obtain our additional HoloLens 2 units in a timely manner this year, but we are optimistic that our increased inventory of HoloLens 2 headsets coupled with a more experienced outreach team will greatly help facilitate more and better outreach events in the coming years.

References

[1] AEGIS, "Nasa suits challenge 2022."

[2] *Apollo 11 transcripts*, https://www.hq.nasa.gov/alsj/a11/a11trans.html, (Accessed on 10/22/2022).

[3] J. Lee and W. Lee, *Compm: Context modeling with speaker's pre-trained memory tracking for emotion recognition in conversation*, 2021. DOI: 10.48550/ARXIV.2108.11626. [Online]. Available: https://arxiv.org/abs/2108.11626.

[4] Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh, "Openpose: Realtime multi-person 2d pose estimation using part affinity fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.