BURGS Weekly Presentation

Broadening Undergraduate Research Groups

10/24/2025

Allie, Casie, Gayatri, Kim

GazeXR Challenges

Cyber attack- with lib/APIs

01

Difficult for undergraduates to complete within time frame

02

No alternative APIs/libraries that do eye tracking

03

Not possible to follow android suggestions for minimizing permissions

App Auditing

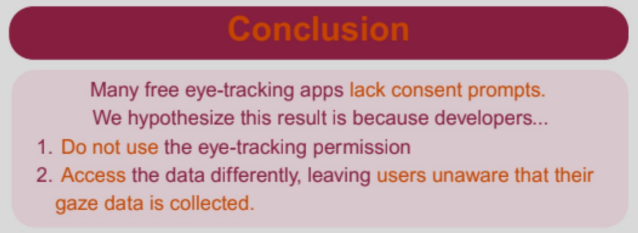
04

Different features are listed on each store, making comparisons hard

05

Not many apps under training/fitness and wellness use eye tracking

GazeXR: Proposal Part 1



Permissions

For using eye tracking functionality in your app, you must declare the "com.oculus.permission.EYE_TRACKING" permission in the Android manifest.

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android">
    <uses-feature android:name="oculus.software.eye_tracking" android:required="true" />
    <uses-permission android:name="com.oculus.permission.EYE_TRACKING" />

    ....
</manifest>
```

For details, read the OpenXR Support for Meta Quest Headsets guide.

The com.oculus.permission.EYE_TRACKING permission is a "runtime" permission, so the app must explicitly ask the user to grant permission. For details about runtime permissions, read Runtime permissions. Here is a sample Main Activity to handle permissions:

```
private static final String PERMISSION_EYE_TRACKING = "com.oculus.permission.EYE_TRACKING";
private static final int REQUEST_CODE_PERMISSION_EYE_TRACKING = 1;

@Override
protected void onCreate(Bundle savedInstanceState) {
  super.onCreate(savedInstanceState);
  requestEyeTrackingPermissionIfNeeded();
}

private void requestEyeTrackingPermissionIfNeeded() {
  if (checkSelfPermission(PERMISSION_EYE_TRACKING) != PackageManager.PERMISSION_GRANTED) {
    requestPermissions(
        new String[] {PERMISSION_EYE_TRACKING}, REQUEST_CODE_PERMISSION_EYE_TRACKING);
  }
}
```

App Auditing Results Explained

01

Developer Permission Ignorance

- https://developers.meta.com/horizon/documentation/native/android/move-eye-tracking/

02

Work with Paul for Cyber Attack

- See if we can collaborate if so…
  - Learn what he's been up to
  - Don't need to finish/just focus on learning

GazeXR: Proposal Part 2

Plan for GenAI data collection and evaluating:

1. Design

- Prompt engineering (how to design your prompts)
- Start with LLMs (for text extraction, summarization, etc.)
  - Colab in Python (e.g., Google Gemini)
    - https://colab.research.google.com/github/google/generative-ai-docs/blob/main/site/en/tutorials/quickstart_colab.ipynb#scrollTo=j51mcrLD4Y2W
  - If needed, add some other image/LLM generative AI
    - GPT-4o (LLM)
    - ClaudeAI(LLM)

2. Evaluation

- If having ground-truth
  - Generate your own ground-truth
    - Divide original script (original privacy policy)
      - Sensor related documentation versus non-related sensor data
    - Measure: ROUGE, BLEU, GPT evaluation
      - Check https://fabianofalcao.medium.com/metrics-for-evaluating-summarization-of-texts-performed-by-transformers-how-to-evaluate-the-b3ce68a309c3
- If not having
  - Human evaluation: Google form for visualizations
    - Users to answer questions
  - GPT-4 (other models) evaluation
    - https://arxiv.org/pdf/2310.13800

[Proposed Timeline]
(i) Start with Gemini + ROUGE
(ii) Try some different prompts (prompt engineering) – see what happened in ROUGE score => better prompts
(iii) Try other measures => Is BLEU score showing the similar trends w/ ROUGE score
(iv) Try other LLM models
——— I know which prompt and which LLM model are good ————
(+) Let's add some image generative AI (we should think about the measure again)

LLM Interpretations Continue

01

Evaluate

- Rouge Evaluation
- Sensor accuracy evaluation

02

Increase Scope

- More policies to interpret
- More prompt engineering for other results
  - Run through evaluations

GazeXR Work Allocated

App Auditing Results Explained

01 Show off the resource if asked (don't need to make a project)

02 *Cyber attack project*↠ Work with Paul to understand project/learn

LLM Interpretations

01

Rouge score with Google Gemini

02

Sensor information evaluation figure out

03

Gather more policies and do more prompt engineering, repeat evaluation

Spatial Seer - Current Progress

Objectives

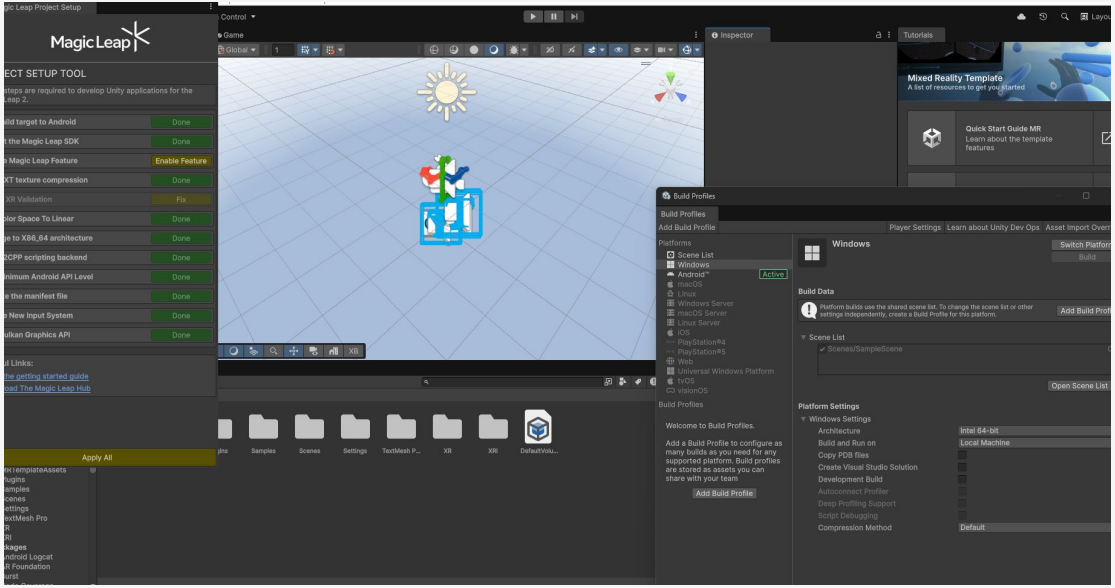
- Prove room predictability across headsets
  - We were able to do this with Perfetto Profiler
- Prove room predictability across AR & MR
  - Almost done with AR, need to develop some MR applications for Meta Quest
  - Developing for ML2 has been difficult

Challenges

- Impossible to find the ML2 Power Profiler performance label names and meanings
- ML2 is even more deprecated, so hard to gather more trials
- We believe we need to gather data through Unity Profiler as well, since attacker would more likely have access to that information than Perfetto's API (though they do have significant overlap)

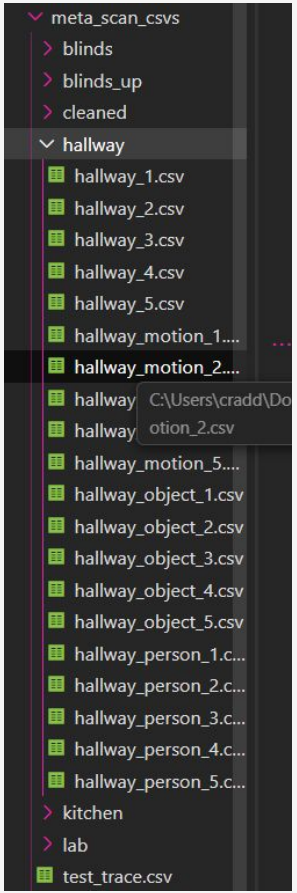
Spatial Seer - ML2 MR Application Development

- Errors with the Unity Plug-Ins for ML2 with Android
- Documentation outdated again
- Challenge: how necessary is Magic Leap 2, especially since the company has continued to be decommissioned

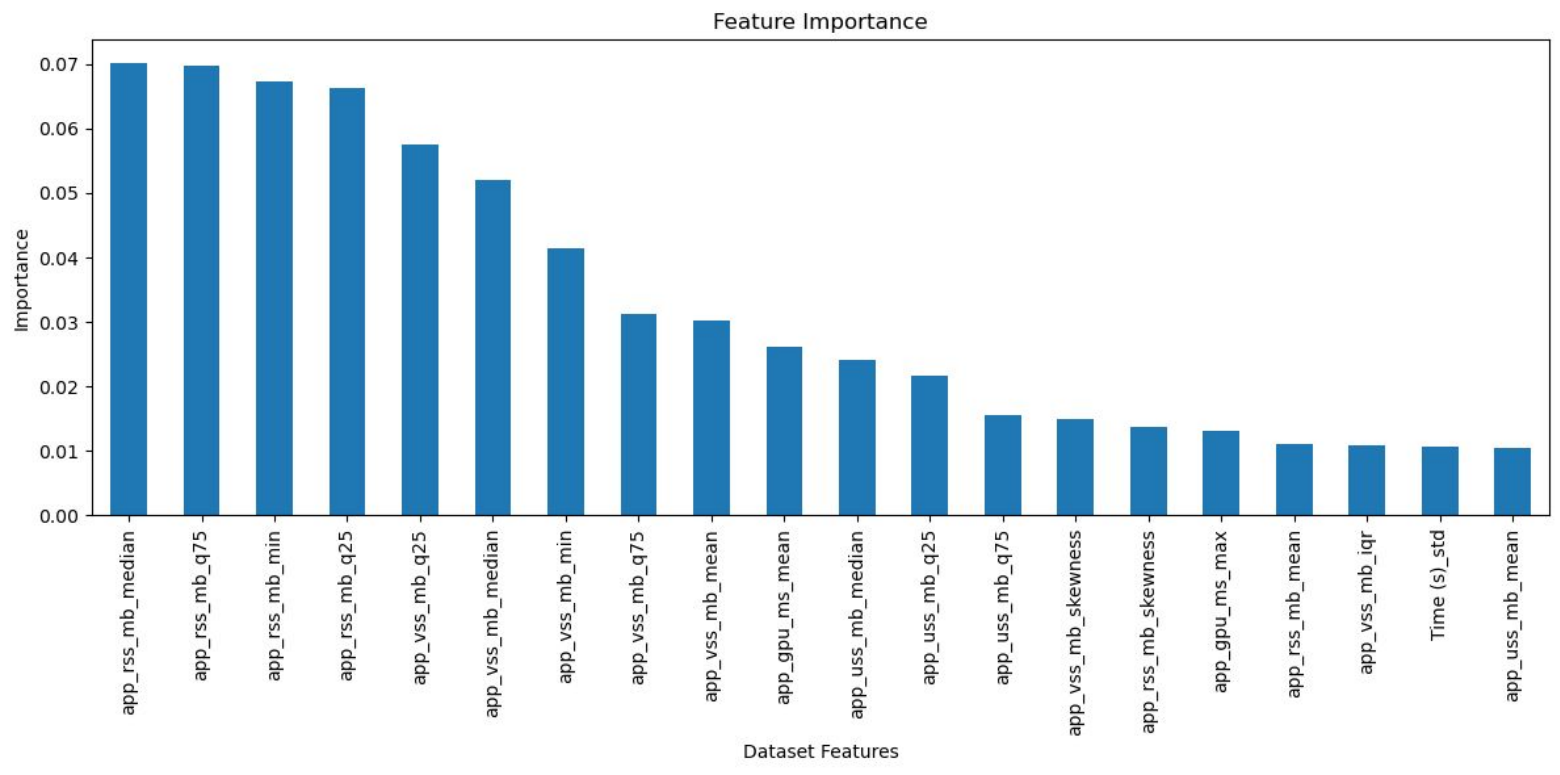


Spatial Seer - Data Collection

- 5+ Room Types
- 50+ scans uploaded as CSVs, working towards 100
- 4 trial conditions, 5 trials per scan type
  - Base
  - Altered Environment
  - Moving Objects (Noise)
  - Mobile XR User

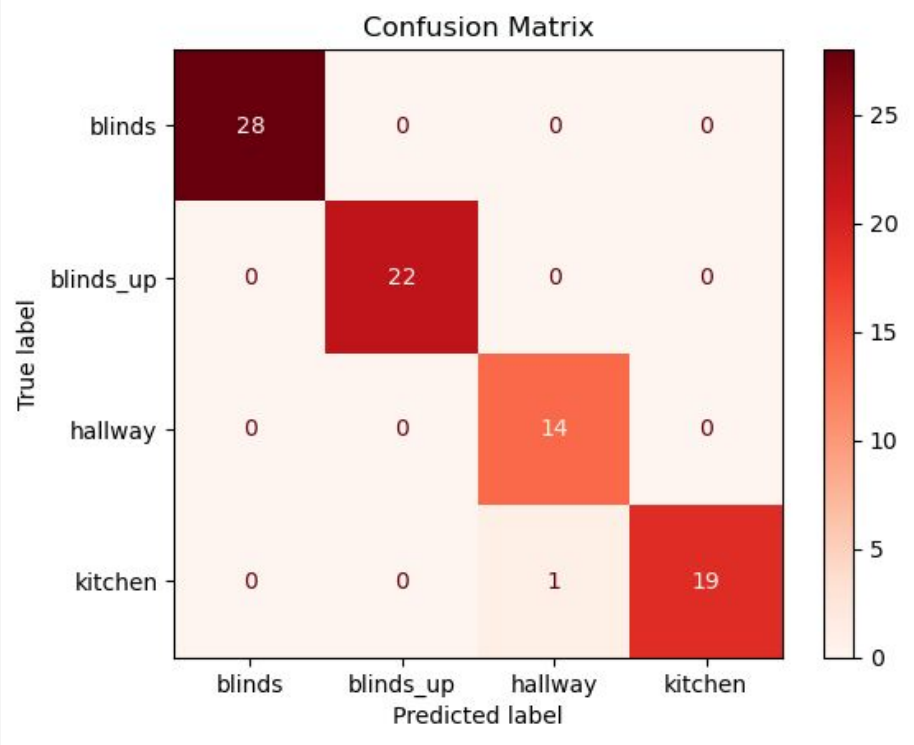


Spatial Seer - Data Analysis



Spatial Seer - Data Analysis

- Random Forest model run on 20 trials (5 trials per 4 room types)
- Tumbling windows to segment time-series data
- Planning to switch to ROCKET/MINIROCKET, as RF takes incredibly long to run/test accuracy, and will be hard to optimize when we eventually have 100 CSVs



Spatial Seer - Next Steps

1. Finish collecting data for remaining rooms across the 4 different scan types
2. Upload all .pftrace files, transfer to CSV, and run through the data cleaning Jupyter Notebook
3. Implement a data segmentation/augmentation technique to create more data points from the 100 CSVs (1 trace should not just be 1 data point, it can be split into many) without data leakage
4. Revisit the Random Forest model or switch to a ROCKET time-series classifier
5. Once satisfied with the accuracy of our model for AR, develop a MR app for Meta Quest (simple ball shooting game)
6. Repeat a more condensed data collection + model development process for MR
7. Unity applications + profiler
8. Figure out the names of the perfetto

Questions

1.