Final Project Report

Kaiwen Guo
kg3354
kg3354@nyu.edu

Abstract

This project focuses on using Unity to create an augmented reality (AR) visualization of network connections surrounding users. Augmented reality (AR) enhances real-world environments through interactive digital visual elements, sounds, and other sensory inputs, often facilitated by holographic technology [1]. Wireless networking, particularly network protocols based on IEEE 802.11 (Wi-Fi), surrounds each AR user. With this AR application, users can easily visualize network performance, including Wi-Fi upload and download speeds, in their immediate vicinity.

1. Introduction

AR combines the "real world" and the virtual world. Essentially, AR takes the real world and enhances it through the use of computer-generated perceptual information or virtual overlays. [2] An interpretation of the virtual world can offer visible demonstrations of the invisible elements surrounding individuals, such as network connections. This project aims to extend the real-world experience to an AR environment by enabling users to visualize wireless network connections. This project enables the users to self-diagnose wireless network connectivity issues and move the device to locate optimal spots for the best wireless performance. This application has broad potential for AR applications, especially AR games that requires a stable network connection.

2. Results and Demonstration

There are three core functionalities of the project. First of all, it shows the Uplink and Downlink Speed in real-time. The textmeshpro object containing the real-time speed information will relocate itself to the center of the camera when the end user tabs the screen. Secondly, when the end user holds the screen for over 3 seconds, the textmeshpro object with the real-time network logging information will also relocate itself to the center of the camera when the end user tabs the screen. Lastly, the project contains 2 cone objects whose texture and height changes also in real time. Their texture color (red, yellow, green) represents the range of byte per second, and the height represents the relative speed.

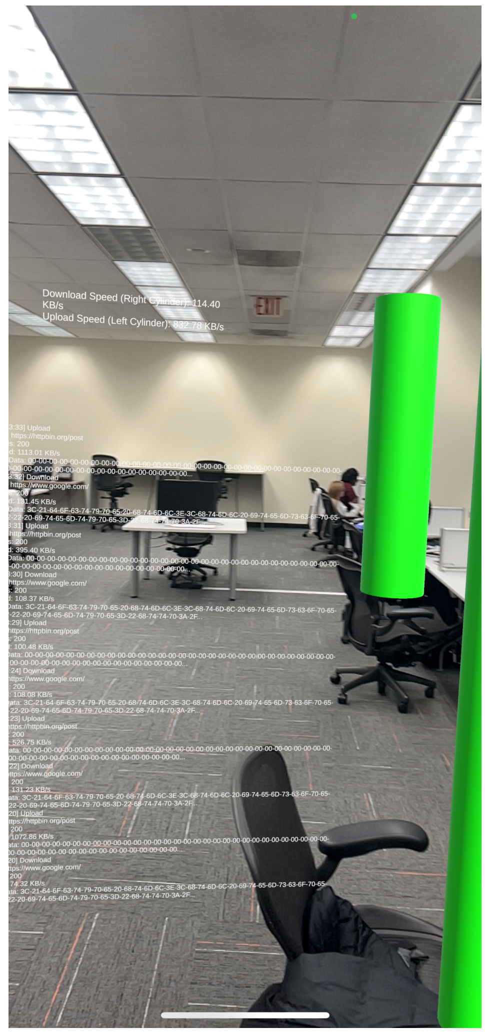

Figure 1. Sample screen shot of the project.

Thus, end users will have two ways of iterating with the application: holding down the screen over 3 seconds, or tapping the screen. Each will centralize different objects to the center of the camera.

3. Implementation

The implementation of this Unity AR project focuses on visualizing real-time network performance metrics in a user-friendly and interactive manner. The project incorporates four main Unity assets: two cones and two TextMeshPro objects. The cones dynamically adjust their height and color to represent uplink and downlink network speeds visually. Meanwhile, the TextMeshPro objects display detailed network speed statistics and raw packet data logs directly within the scene.

1. Real-Time Network Speed Visualization

- The `MeasureDownloadSpeedCoroutine` and `MeasureUploadSpeedCoroutine` functions compute real-time network speeds.
  - Download speed is measured by fetching content from `https://www.google.com`.
  - Upload speed is gauged using a POST request to `https://httpbin.org/post`.
- These functions track the time taken to transfer data and calculate speeds in KB/s.
- The results are reflected in the cones' heights using the `UpdateConeHeights` function, which normalizes the speed values to a defined scale. Cones also change color based on the speed, with red, yellow, and green representing slow, moderate, and fast speeds, respectively.

2. Raw Packet Data Logging

- The `LogPacketData` function captures network packet details, including:
  - Timestamps
  - URLs
  - Response codes
  - Transfer speeds
  - Raw data previews
- These logs are stored and displayed in a TextMeshPro object, offering real-time insight into network activity.
- A list structure stores up to 10 recent log entries, which are updated every second while holding the screen.
- This ensures clarity and relevance in the displayed network activity.

3. Interactive AR Interface

- The `HandleInput` function detects screen taps and holds, enabling users to interact intuitively with the AR scene.
  - A tap centers the network speed display in front of the camera.
  - Holding the screen for over three seconds reveals detailed network packet logs.
  - The `CenterTextInFrontOfCamera` and `PositionLogInFrontOfCamera` functions position the TextMeshPro objects dynamically in front of the main camera. This ensures that data remains visible and accessible regardless of the user's perspective.

The project is entirely open-source, with the C# code accessible on GitHub at: `https://github.com/kg3354/UnityARNetworkVisualizer`

4. Discussion

The current system has a limitation: the target build is iOS, which restricts many built-in Unity network functionalities compared to Android. As a result, all network speeds in this project are calculated manually using various functions. This approach is inefficient and can be problematic, especially if not all network data is transparent to developers or users by default.

An extension of this project could involve improving the interaction model. Instead of requiring the user to tap the screen at various locations to move the TextMeshPro object, the objects should remain at their initial spawn points and retain the speed data from when they were first created. When the user moves a certain distance or requests another text object, a new TextMeshPro object containing the current data could be spawned at the user's new location. This enhancement would provide a more intuitive way for users to visualize how network performance changes as they move between different locations.

References

[1] Microsoft. What is augmented reality or AR?, 2024. `https://www.microsoft.com/en-us/dynamics-365/topics/augmented-reality/what-is-augmented-reality`. 1

[2] Pluralsight. Augmented Reality (AR): What is it, and how is it being used? www.pluralsight.com, Accessed: Oct. 28, 2024. `https://www.pluralsight.com/blog/creative/what-is-augmented-reality-ar`. 1