ISDN6830B Homework 3 Report

Siyan HU (12217679)

May 5, 2024

1 Brief

This assignment is designed to facilitate a deeper understanding and practical engagement with the server/client architecture within the context of object detection, specifically implemented in an augmented reality (AR) framework. By navigating through this exercise, student will be exposed to the intricacies of deploying and managing AR applications that utilize a distributed computing model to enhance real-time interactions with digital environments.

2 Design and Programming

In preparation for this project, I started by revisiting the foundational principles of server/client architecture with going through previous projects. This review includes a thorough examination of each component involved in a network system, ensuring a solid theory foundation upon which to build the practical aspects of the project.

I created a system architecture diagram, which is presented in Figure 1. The diagram is divided into two primary sections to delineate the roles and functionalities of the different components within the system. On the left side of the diagram, the mobile application framework is depicted. This part of the system is designed to interact directly with the user, capturing real-time data and user inputs, which are essential for the augmented reality interface.

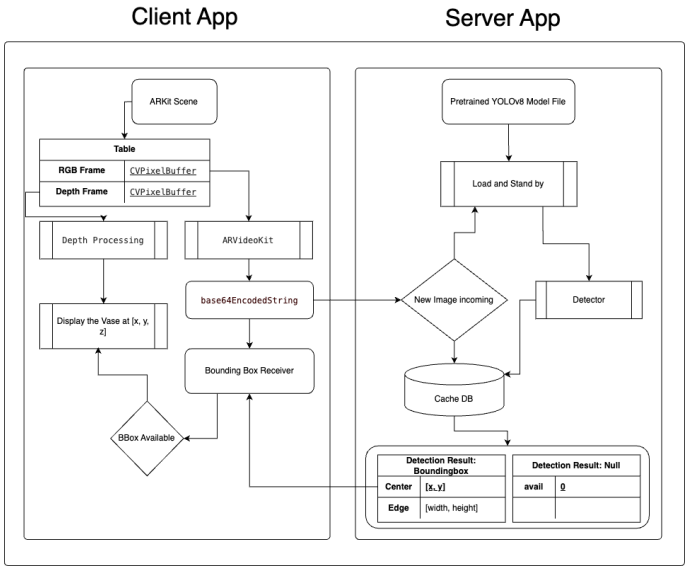

Figure 1: Full Diagram

The right side of the diagram details the server design, which is crucial for image data processing and object detection tasks. This server architecture is configured to receive data transmitted

from the mobile application (with Flask [3] Library), process this information using a pretrained YOLOv8 [5] detector for chessboard detection, and then send the processed data back to the mobile application. The server is thus a critical component in managing the computational complexities and ensuring that the mobile application can function efficiently without being burdened by processing overhead.

3 Usage

At first I initiate the server by using flask_server.py. Then the client application is installed on an iPhone, as depicted in the left image of Figure 2. Upon installation, the application prompts a camera usage request, which is an essential step for the subsequent functionalities of the client app. Users must select "Allow" to grant the application access to the camera.

Once permission is granted, the application activates the back side camera of the iPhone. The display then shows the live feed from the back camera. This setup is to enable real-time data capture and interaction within the augmented reality environment facilitated by the client application.

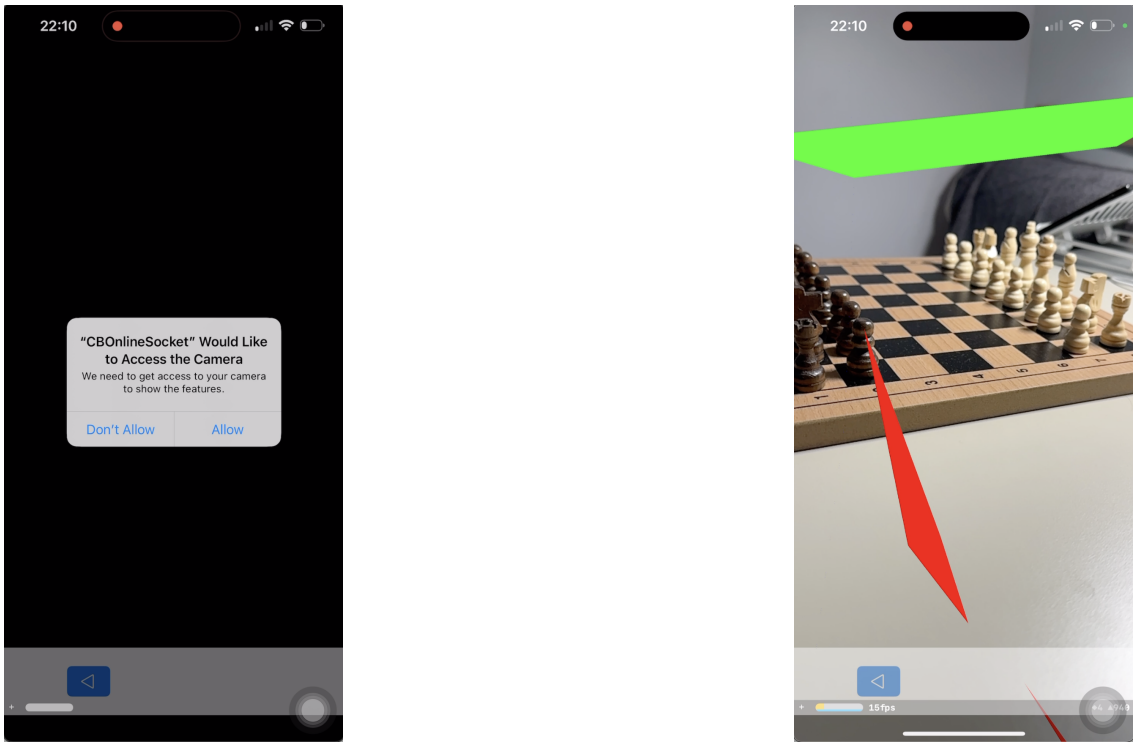

Figure 2: Left: camera privacy request; Right: Initial View.

Initially, for testing it integrates a feature that allows a 3D vase to be positioned at a specific location within the screen area where the user taps. Taps will relocate the vase to the new position, effectively updating its location to reflect the most recent user interaction.

Otherwise, the user is instructed to activate the system by clicking the blue start button located in the lower left corner of the interface. Then RGB data captured by the device's camera is encoded and transmitted to the server for analysis [2, 4]. The server, which processes this incoming images and do the detection.

Upon processing the received data, the server generates and sends back to the client app the bounding box information, which includes the width, height, and center position of the detected

object. In scenarios where no chessboard is detected, the server will instead return a result indicating zero detection.

In the final step, as you hold your phone, the vase will be automatically positioned at the center of the bounding box, as indicated by the received data and illustrated in Figure 3. To generate the world coordinate of this placement, the 2D center coordinate of the bounding box is treated as if it were a point on the screen where a user has tapped. This 'pseudo tapping point' is then translated into a world coordinate within the ARKit scene, effectively placing the vase in the virtual environment.

Additionally, depth information is gathered at the same time with the previously captured RGB data from the ARKit scene. This depth data provides a three-dimensional context that enhances the placement accuracy of the vase within the scene with a more realistic integration of the digital object into the physical environment, rather than 'floating onto the surface'.

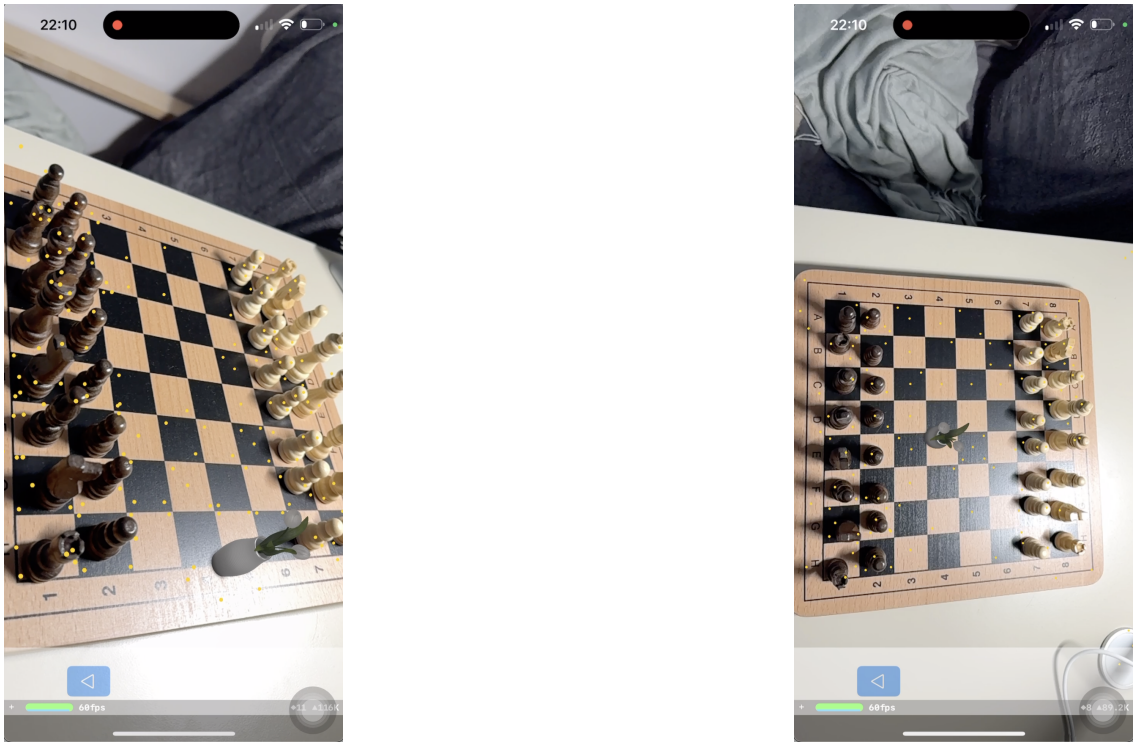

Figure 3: Vases are put onto the chessboard.

4 Known Issues

- As the object detection process is handled by the server, a significant delay is anticipated between the moment the Capture button is clicked and when the vase appears on the chessboard. This latency arises from the time required to transmit the data to the server, process it, and return the results.
- The placement of the vase is determined based on the center coordinates of the bounding box. Consequently, there may be a discrepancy between the actual center of the chessboard in world coordinate and the location where the vase is placed. This displacement is due to the bounding box potentially not aligning perfectly with the true center of the chessboard.
- The depth data utilized in this application is derived from ARSession [1] Frame, which relies on hardware capabilities such as a depth camera or LiDAR sensor. Therefore, this

functionality is not supported on iPhone or iPad models that lack these specific hardware features. This limitation restricts the depth data generation to devices equipped with the necessary technology.

References

[1] Augmented Reality - Apple Developer — developer.apple.com. https://developer.apple.com/augmented-reality/. [Accessed 05-05-2024].

[2] GitHub - AFathi/ARVideoKit: Capture & record ARKit videos, photos, Live Photos, and GIFs. — github.com. https://github.com/AFathi/ARVideoKit. [Accessed 05-05-2024].

[3] Quickstart &x2014; Flask Documentation (3.0.x) — flask.palletsprojects.com. https://flask.palletsprojects.com/en/3.0.x/quickstart/. [Accessed 05-05-2024].

[4] 262588213843476. How to upload jpeg image using URLSession. — gist.github.com. https://gist.github.com/nnsnodnb/efd4635a6be2be41fdb67135d2dd9257. [Accessed 05-05-2024].

[5] Jocher, G., Chaurasia, A., and Qiu, J. Ultralytics YOLO, Jan. 2023.