Project synopsis:

Short Title: CloudRenderVR

Title: Cloud-based VR game rendering

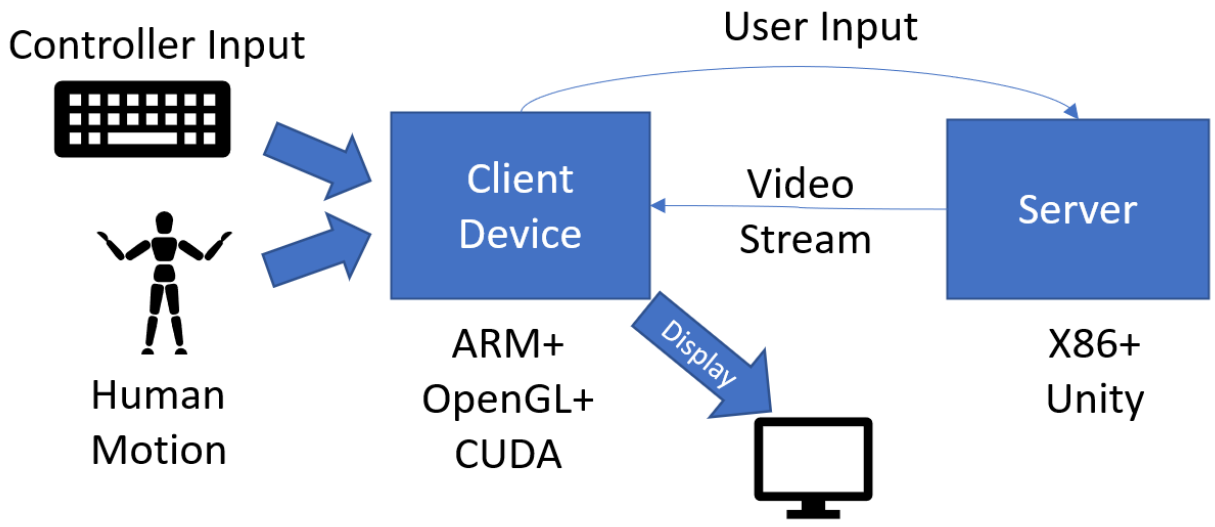
Project lead and contact details: Mehmet Belviranli, belviranli@mines.edu

Suggested team size: 4-5

Logistics: Can work from anywhere, the client component may require physical presence in the lab where the device is located.

Project description:

Untethered Virtual Reality (VR) platforms struggle to keep up with the rendering requirements necessary for VR gaming due to the limited capability of mobile GPUs. Many strategies for untethered VR utilize the cloud for rendering frames, but network limitations lead to high latency with this remote-only rendering. We need to implement a framework that pre-renders frames for untethered VR on the cloud to lower this latency. Controller inputs would be used as input for the game being rendered on the cloud. Human motion as an additional input is a stretch goal for this project (see below).



Project components:

The project requires multiple components that uses existing open-source technologies to be stitched together and work coherently. The components are detailed below along with their expected core and stretch functionality. Only core functionalities are required for the duration of the field project.

Client side (mobile VR platform): The client platform will utilize a low powered CPU+GPU device such as Qualcomm Snapdragon or NVIDIA Jetson Xavier. Due to programmability limitations of existing VR platforms, instead of using VR glasses, we will use development boards that host the same system-on-chips that such VR platforms embed. The core responsibility of client will be to gather inputs from the user, transmit them to the cloud server and encode/display the incoming video stream. Target platform will be either Android (with NDK for logic integration) or Linux with CUDA/OpenGL.

- Core functionality will include:
    - C++ for the main program logic
    - Use keyboard/mouse as input
    - Use H.264 video decoding and display
- Stretch goals for this component include:
    - Human motion as input (via OpenPose)
    - OpenGL for frame error recovery and frame interpolation (IBR, ATW)

Server side (cloud platform): The cloud platform is a powerful multi-GPU/multi-core server that is settled within Mines network. It is capable of rendering frames significantly faster with a better resolution and quality than the mobile platform, at the expense of network and rendering latency. The server will receive the inputs from the client over the network and pass them into the Unity gaming engine. Once the engine renders the next set of frames, the server will encode those frames into H.264 video and transmit back them to the client.

- Core functionality will include:
    - C++ for the main program logic.
    - Setting up a working instance of Unity gaming engine on the server:
    - H.264 video encoding via NVIDIA Video Codec SDK
- Stretch goals for this component include the integration of the following additional Unity plugins including: Furion Unity and “Cutting the Cord” Unity plugin using OpenVR API.

Desired skills:

Following skills are preferred but not necessarily required:

- AR/VR frameworks
- Computer graphics
- Unity
- Development experience on heterogeneous SoCs
- Computer networks

Devices available:

- Mobile/Client platform
  - NVIDIA Xavier AGX
  - Qualcomm Snapdragon 865 Dev kit
  - (We will not be using actual VR glasses for this project)
- Istanbul server as edge-cloud device
  - 48-core AMD EPYC 7002 server with 512 GB Memory
  - 2x NVIDIA Titan RTX graphics cards with NVLink
  - 10 GBps Network.

Expected Outcome:

At the end of the project, the team is expected to create a working prototype of a playable game (e.g. Quake). There is no expectation to create a mobile UI, or a mechanism to remotely start games on the server. The teams may assume that the game already runs on the server, and clients just connect to the running instance, transmit inputs and streams the output.