Khen Cruzat

# Progress Report

**Progress**

For the project I have conducted research and literature review on the different methods implementing a cloud gaming system as well as the challenges that can arise as well as possible solutions to solve them. The project is essentially split up in two parts where the first part will be creating the cloud gaming system and the second part where I have to tackle the problem of network latency.

For the first part I have discovered that it is better to offload most of the computing power which is the game pipeline to the cloud server. Another element of this is to create a simple game/simulation that will be used to test the cloud gaming system with. The progress I have made with this is that I have created a simple game which renders a procedurally generated tree on a plane. This tree also has shadows which are generated through projective rendering which is where the tree is rendered twice but the second render is transformed and projected on to the plane. To make the program interactive, I have implemented simple flight simulator controls so the user can move around. I have made the game so it will try to meet a standard monitor's refresh rate of 60 Hz. The performance of the game can be seen by the frames-per-second counter I have implemented.

As for the cloud gaming system, I have implemented a simple client and server. The client connects to the server using a hostname and a port number. Once a connection is made, the user can then send keyboard inputs to the server and control the camera of the game. This is done by using the Qt framework and its QKeyEvent functionality to detect which keyboard button is pressed. Once the server has received an input event, it simulates the button press so the game responds accordingly. Also, I have started the process of capturing the rendered frames. It currently captures the rendered frames as QImage objects which are then simply saved as .jpg picture files.

For the networking part, I have researched into reducing network latency in cloud gaming through the use of SDN (Software-Defined Networking).  In traditional networks, the routing protocols usually use predefined static routes and don't take into account dynamic parameters like bandwidth utilization and packet loss. SDN provides a means for controlling the network programmatically.

**Work left to do**

The work that is left to do with the cloud gaming system is to implement offscreen rendering. This is done so the game is not displayed on the server side but instead rendered offscreen to improve performance. The game renders in to frame buffer object with Qt's QOpenGLFramebufferObject class which includes a function to return a frame as a QImage. I will then use an encoding library such as FFMPEG/libav to convert to H264 MP4 format. These will then be streamed with the help of the live555 library which uses RTSP (Real Time Streaming Protocol). On the client side, I need to implement a video player to display the video stream of the game on a Qt window.

When the cloud gaming system is done, I will need to work on the SDN implementation to improve the latency that will arise on the cloud gaming system. This will be done creating virtual networks that will use an SDN controller to manage the routing of the packets in the network. To improve the latency, I will use Djikstra's algorithm to find the shortest path between two hosts. I will simulate network traffic and see how the SDN controller affects the latency. The network traffic produced from the connection to the client and server messages and video stream of the cloud gaming system will be measured for example, with the Wireshark software.

I will then produce a performance evaluation of SDN for cloud gaming systems seeing how different amount network traffic will affect the SDN implementation in terms of latency produced. Other solutions/routing algorithms may be explored if I have more time.

**Issues experienced**

One of the first issues that I came across is with narrowing down my research topic. I knew that it will involve creating a game or simulation that will benefit by moving it to the cloud, but I could not find a hypothesis that I will need to answer. Latency in cloud gaming is what I decided to tackle and Microsoft's Outatime project that speculates the frames needed ahead of time is an example solution. After thorough reading, this was deemed not feasible due to limited time I had and that there was not a lot of information available on the implementation since it is not open source.

I have encountered a number off issues whilst trying to experiment with SDN and its feasibility. One of the problems was that whilst researching I have found that the School of Computing's cloud computing platform OpenNebula has the ability to integrate the OpenvSwitch virtual switch feature, but these were not used when creating the cluster. The only way for them to be enabled is upon creation which meant that I could not use the School of Computing testbed for SDN development. I spent many hours on setting up virtual machine environment which lead to pushing my scheduling back. As a result of this, I decided to move on to using virtual networks and simulated network traffic instead.