ONLIVE CLOUD GAMING SERVICE

SE 172/272 Enterprise Software

Project Topic:
Web Service

Professor:
Rakesh Ranjan

By:
Kelvin Leung
Talhah Asharaf
Xiang Liu

May 2011

TABLE OF CONTENTS

ABSTRACT

OnLive cloud gaming services provide on-demand gaming access to many popular game titles. User can play graphic intensive games without having a high-end computer. All data and graphic are rendered at OnLive remote servers. Servers just take users' inputs and stream real-time video of the game back to the user. However, one of the biggest problems of this approach is that it requires decent internet bandwidth. Users will experience high latency if they do not have a fast internet. Our objective is to identify a solution that can reduce the bandwidth usage of this problem without significantly impacting performance and quality.

I. INTRODUCTION

1.1 ONLIVE CLOUD GAMING SERVICE

Figure 1: OnLive Service Main Screen

(http://pc.ign.com/articles/965/965542p1.html)

“OnLive delivers high-end video games from the cloud to your PC, Mac®, TV and even mobile devices at blindingly fast speeds, so you can play the games you want, the second you want them. No discs. No downloads. No fancy hardware. Just you, the Internet and the games you love.” www.onlive.com

1.1.1 WHAT IS IT?

OnLive is a cloud gaming service provider that allows its users to play graphic intensive games without having the hardware that meet the minimum requirements of these games. All games are stored and rendered in OnLive’s data centers, so the user only needs a device that can play video and receive inputs, and access to high speed internet. This feature opens many new doors for many existing computers. Many of these computers can now play higher quality games without having to meet the minimum specifications. As new games are being developed, the minimum system requirements continue to rise. However with OnLive, users will no longer need to upgrade their existing hardware in order to play newer games.

OnLive service also offers several other appealing features. One feature that OnLive provides is on-demand access. OnLive’s users can play the games that OnLive is partnered with anytime they want. However, they will need to first purchase the game on OnLive. After purchasing the game, users can then play that game anytime they want with any computers. Since all the games are stored and executed at OnLive’s data centers, no installation or patching is required for the users. OnLive service not only provides on-demand access to games, but it also provides on-demand access to demos. So users can play demos without downloading or installing before purchasing the games.

Another interesting feature that OnLive provides is the ability to view other users' gameplay at any time. Traditionally, users can only view replays after the game is over, and users must download the replay. However with OnLive, users not only can view others' replays, but they can watch the gameplays at real-time. Since game data are stored in OnLive's data centers, this can save users' physical disk spaces. Not only that, this also allows users to pause and resume anytime, and idling game will not use up users' system resources.

1.1.2 HOW DOES IT WORK?

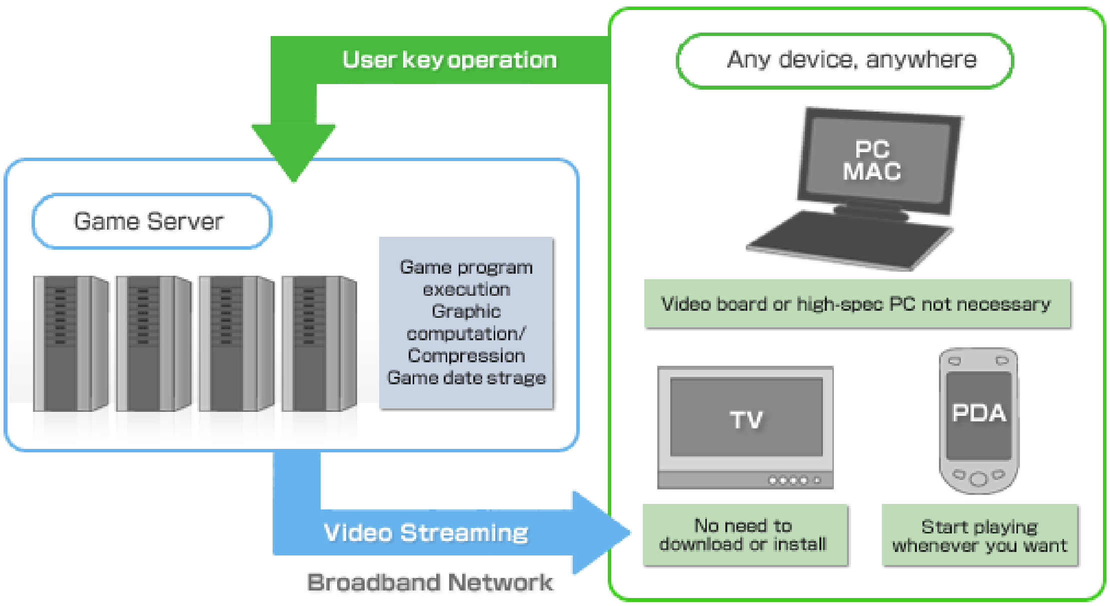

Figure 2: Cloud Gaming Overview

(http://www.gcluster.com/eng/eng_outline/eng_outline.html)

The detailed architecture behind cloud gaming can be very complicated. However, the logic behind this can be very simple. In the simplest form, cloud gaming is just executing the games in the cloud instead of users' devices.

The basic workflow can be shown in **Figure 2**, all game program execution, graphic computation and compression are done in remote game servers. All game data are stored in these remote game servers as well. The cloud gaming process begins when user requests a game. Then the game server will start executing the requested game and streams the video feed of that game to the user. On the client side, devices such as a PC, TV, or Smartphones will be used to display this video feed. User will also use these devices to send key operations back to the remote servers. After receiving user inputs, the game server will send these key operations to the running game program to process. The game server will continue to take inputs from user and stream the video feed back to user until user terminates the connection.

1.1.3 THE PROBLEM

OnLive is still a relatively new technology; therefore, there are still a lot of limitations. One of the biggest limitations of OnLive is that it requires high speed internet. Also, users must be within 1000 mile radius of OnLive's data centers in order to receive services. Because OnLive streams HD video to users and receive user's inputs at the time, the biggest problem that they are currently facing is high latency. In order words, users might send key operations, but there is a noticeable delay in the change of the video display.

There are two factors that caused this latency problem. The first factor is limited bandwidth, so it is not possible to stream uncompressed digital images without significant delays. Unlike other streaming services like YouTube and Hulu, OnLive streams at real-time, which means the video source is being created as it streams. Therefore, it cannot apply traditional video compression techniques to compress the entire video beforehand. In order for OnLive to reduce the bandwidth requirement, it must compress video feed in real-time. However, this approach creates another problem which is the second factor that caused this latency problem (Shi, Shu). Because it takes time to compress and decompress digital images, thus increasing latency. Not only that, compression will also lower the video quality.

1.2 OBJECTIVE

The objective of this research is to find a solution that can reduce the bandwidth requirement and latency of OnLive. In order to achieve that, this research must identify a real-time video compression technique that can reduce the amount of data required to send back to the user without significantly impacting performance and quality.

II. BACKGROUND RESEARCH

2.1 WHAT ARE VIDEO COMPRESSION AND VIDEO CODEC?

Video is basically a sequence of images. Each image represents a frame in the video. The algorithms and logics behind video compression are very complex. However what video compression does is simply reduce the amount of data it requires to represent digital video images. For instance, H.264/MPEG-4 AVG is a standard for video compression. It is also the most commonly used format for HD videos.

There are two types of video compression. One is lossy and another one is lossless. Lossy compression means there are some information that has been discarded during compression, and such information cannot be recovered in decompression. On the other hand, lossless compression means all the data is the same after being compressed and decompressed. So no information is lost forever in this case.

Video data contains spatial and temporal redundancies. Spatial redundancy is the similarities within one frame while temporal redundancy refers to the similarities between frames. These redundancies are used to encode digital data to create the compressed video file (Symes, Peter). Then by using a decoder, one can understand the compressed video file and display accordingly. These encoder and decoder can be refereed as video codec. They are hardware or software that is used to compress or decompress an input video.

2.1.1 VIDEO COMPRESSION ALGORITHMS

Data compression is the technique by which we try to reduce the size of bits of data for a multiple of reasons most significantly to save space and also to make the data flow through a mechanism faster because of its smaller size than the original bits it was using.

Compression can be achieved by a multiple of ways and algorithms. One single algorithm cannot work for every case or data type because there are different formats and types of data bits and every algorithm has its own way of compressing data.
An algorithm that uses compression technique for video will have different requirements than that algorithm that uses it for data compression like word files.
There are many types of compression algorithms but on a broader level they can be divided into two basic categories namely loss less data compression and lossy data compression.

Some of the examples for lossless data compression are Run length encoding, context mixing, entropy encoding. Some of the examples of lossy data compression are fractal compression, cosine transform, vector quantization. We will examine both of these basic technologies more thoroughly (Symes, Peter).

Loss less data compression:

This particular type of data compression is used where we need exact replica of the data that we had compressed. This is the exact opposite of lossy type of data compression. In this type of data compression the most important thing is that the exact state and type of data bits are retained after compression and decompression takes place.

Loss less data compression is very important for some applications and a lot of algorithms are based on this technique and use loss less data compression.
The basic phenomenon for loss less compression is two sequences. The first step generates a model while the second inputs the bits in such a way that those bits that are not required for the final outcome are put aside. The encoding algorithms that are used are basically Huffman coding and also other types of code sequence. There are two ways to construct a model. The first is the dynamic way while the second is the statistical way. In statistical way a single data model is created and then the data is compressed while in dynamic model the model is made as the data is being compressed (Symes, Peter).

Lossless data compression algorithms cannot guarantee compression for all input data sets. In other words, for any lossless data compression algorithm, there will be an input data set that does not get smaller when processed by the algorithm.

Lossy data compression:

This is the exact opposite of lossless data compression. In this type of compression the data is compressed by discarding some of it.

Lossy compression is most commonly used to compress multimedia data especially in applications such as media and internet. By contrast, Lossless is required for text and data files, such as bank records, text articles, etc. In many cases it is advantageous to make a master lossless file which can then be used to produce compressed files for different purposes; for example a multi-megabyte file can be used at full size to produce a full-page advertisement in a glossy magazine, and a 10 kilobyte lossy copy made for a small image on a web page.

2.2 HOW DOES VIDEO STREAMING WORK?

Video streaming is very similar to playing a DVD. The only difference is that the location of the DVD is at a remote location. Video streaming is process of transferring digital video images onto a local device to display. Video streaming is usually used along with video compression, because raw (uncompressed) video data can be very huge. Therefore, video compression is done before streaming to user to reduce the bandwidth usage.

III. RESEARCH

3.1 CURRENT APPOACH

To address the latency issue, OnLive currently uses h.264 video compression format to encode their digital video images before transmitting to users. According to OnLive's competitor OTOY, "OnLive uses an h.264 encoder chip on a separate piece of hardware to grab the video output of each GPU running on their servers" (Waite, Stephen). This approach requires an intensive use of the CPU. In this case, OnLive uses additional hardware that are equipped with h.264 specific encoding chips just to compress the digital images. Therefore, it can be very costly and sometimes unnecessary.

3.1.1 WHAT IS H.264?

H.264 is video compression standard. It is also known as MPEG-4 Part 10/AVG. H.264 consists of several video compression techniques such as motion compensation and deblocking filters. The most important feature of H.264 is that it is customizable which allow user to further decrease the bitrate. However, encoding and decoding H.264 requires extensive use of the CPU (Salient Systems Corp).

3.2 POSSIBLE ALTERNATIVES

3.2.1 VC-1 (MICROSOFT WINDOWS MEDIA VIDEO 9)

VC-1 is the informal name of the SMPTE video codec standard. VC-1 is one of the popular alternatives to H.264. However it is not as flexible as H.264. For testing, we will be conducting tests with Microsoft Windows Media Video 9 (WMV 9) since it is compliance with this standard.

3.2.2 MJPEG

MJPEG is a lossy intraframe compression technique. Unlike H.264 and VC-1, MJPEG uses spatial compression instead of temporal compression. As a result, this technique requires much less CPU power than H.264. According to Jason Spielfogel, "Temporal compression offers little or no advantage over frame-by-frame compression when there is a lot of motion in the scene or when the background is changing." Therefore, if the background or the scene of the video changes rapidly, then temporal compression will not produce a better result than spatial compression. It would just waste computation time (Spielfogel, Jason).

3.2.3 OTHER CODECS AND COMPARISON

There are many other codecs that follow other standards. For instance, MPEG-1, MPEG-2, Ogg Theora, etc. Even for MPEG-4, there are few variants. **Figure 3** below shows the comparisons of different video coding standards.

Features	Standards (Profiles)							Non Std
	MPEG-1	MPEG-2	MPEG-4 ASP	MPEG-4 AVC Main	MPEG-4 AVC High	VC-1	Ogg Theora	SNOW
B-Frames	✓	✓	✓	✓	✓	✓		
Slices Error Resilience	✓	✓	✓	✓	✓	✓		
Interlace (PAFF/MBAFF)		✓	✓	✓	✓	✓		
Entropy Encoding	Huffman	Huffman	Huffman	Exp-Golomb or Adaptive Arithmetic	Exp-Golomb or Adaptive Arithmetic	Huffman and Bitplane	Adaptive Huffman	Adaptive Range
Motion Block Size	16x16	16x16	16x16/8x8	16x16/16x8 8x8/8x4/4x4	16x16/16x8 8x8/8x4/4x4	16x16/8x8 8x4 or 4x4	16x16/8x8	32x32/16x16 or 8x8
Motion Search Precision	One Pixel or Half Pixel	Half Pixel	Half Pixel or Quarter Pixel	Quarter Pixel	Quarter Pixel	Half Pixel or Quarter Pixel	Half Pixel	Half Pixel or Quarter Pixel
OBMC								✓
GMC (Global Motion Compensation)			✓					
Intra Prediction	DC 8x8	DC 8x8	AC 8x8	Spatial 16x16/4x4	Spatial 16x16/8x8 or 4x4	AC 8x8	DC 8x8	DC 32x32/16x16 or 8x8
Spatial Transform	DCT 8x8	DCT 8x8	DCT 8x8	HCT 4x4	HCT 8x8/4x4	8x8/8x4/4x4	DCT 8x8	Wavelet 13/7, 9/7 or 5/3
Bit Exact Decoding				✓	✓	✓	✓	✓
Lossless Mode					✓			✓
Custom Quantizer Matrix	✓	✓	✓		✓		✓	
In-loop Deblocking Filter				✓	✓	✓	✓	N/A
Multiple Reference Frames				✓	✓		2 Max	
Weighted Prediction				✓	✓	PP Only		
YUV Colour Formats	4:2:0	4:2:0 4:2:2, 4:4:4	4:2:0	4:2:0	4:0:0, 4:2:0 4:2:2, 4:4:4	4:2:0	4:2:0 4:2:2, 4:4:4	4:0:0, 4:2:0

Figure 3: Video Coding Format Comparison (http://forum.doom9.org/showthread.php?p=674819#post674819)

3.3 TEST METHODS

In order to select the best video codec, one should look at some scenarios. There are mainly four test cases. First test case is a still image video, which has very little change in the video. Second test case is a moving object in a stationary background. In this case, only the object in the video is moving and the background does not change. The third test case is the opposite of the second test case, which has a moving background and a stationary object. The last test case is the worst case scenario, which has a moving background and a moving object. So everything in this video changes rapidly, thus very little temporal redundancy.

By comparing the compression ratio, quality, and the performance, one can determine which codec is the best in which scenario. However, for this research, we are ignoring the performance factor. There are too many other factors can contribute to performance, so we are assuming that H.264 requires the most CPU/GPU processing power while MJPEG requires the least. H.264 requires the most because it uses interframe compression, and MJPEG uses intraframe compression. Interframe compression uses earlier or later frames in sequence to process the current frame while Intraframe uses only the current frame (CalArts).

The programs that we will be using to compress/decompress are Directshow Graph Tool by Microsoft and H.264 Encoder by GPL.

3.4 TEST RESULTS

3.4.1 STILL IMAGE

Description:	In this video, the person and the background remains relatively stationary. There are only slight movements and eye blinking.
Video Length:	30 seconds
Original File Size:	97,246 KB

Codec	Compressed File Size	Compression Ratio	Image Quality
H.264	9,569 KB	0.098	Good
WMV 9	9,583 KB	0.099	Good
MJPEG	16,256 KB	0.167	Good

3.4.2 MOVING OBJECT IN STATIONARY BACKGROUND

Description:	In this video, the person is moving but the background remains stationary.
Video Length:	32 seconds
Original File Size:	93,665 KB

Codec	Compressed File Size	Compression Ratio	Image Quality
H.264	12,426 KB	0.132	Acceptable
WMV 9	11,846 KB	0.126	Acceptable
MJPEG	17,957 KB	0.192	Good

3.4.3 STATIONARY OBJECT IN MOVING BACKGROUND

Description:	In this video, the person remains relatively stationary, but the background is moving rapidly.
Video Length:	31 seconds
Original File Size:	95,388 KB

Codec	Compressed File Size	Compression Ratio	Image Quality
H.264	18,317 KB	0.192	(Acceptable) Blurry Background
WMV 9	15,797 KB	0.166	(Acceptable) Blurry Background
MJPEG	21,086 KB	0.221	Good

3.4.4 MOVING OBJECT AND MOVING BACKGROUND

Description:	In this video, both the person and the background changes radically.
Video Length:	32 seconds
Original File Size:	105,250 KB

Codec	Compressed File Size	Compression Ratio	Image Quality
H.264	25,028 KB	0.238	Poor (Blurry Motions)
WMV 9	27,160 KB	0.258	Acceptable
MJPEG	27,179 KB	0.258	Good

IV. DISCUSSION

4.1 CONCLUSION

In the first test case, which has no major movement from the background and the object, the compression rate for H.246 and WMV 9 is extremely high comparing to MJPEG. This is as expected, because both H.246 and WMV 9 uses temporal compression techniques while MJPEG uses spatial. The qualities of each of the compressed videos are not much different than the original source video.

In the second test case, the person in this video keeps moving his head while the background remains stationary. WMV 9 has a best compression ratio. H.246 has a slightly low compression rate. Both WMV 9 and H.236 have a much better compression ratio than MJPEG. As for video quality, MJPEG has the best quality.

In the third test case, the person in this video remains stationary while the background is changing rapidly. WMV 9 has the best compression ratio for this case.

H.246 finished second but this time it is not as near as the last case. MJPEG has the worst compression rate as previous two cases. However, in both WMV 9 and H.246, the output videos have a blurry background due to compression.

In the last test case, both the person and the background move rapidly. This is considered the worst case scenario. Therefore, they don't have a very good compression ratio. However, for the H.246, the video quality is very blurry, and WMV 9 is blurry but much less blurry than H.246. Like every other cases, MJPEG has the best video quality, but unlike other cases, MJPEG's compression ratio is not much higher than H.246 and WMV 9.

Based on these four test cases, the average compression rate for both H.246 and WMV 9 are about the same. However, WMV 9 performs better when there is a stationary background or a stationary object. For most games other than first-person-shooting game, these two cases are the norms. Based on that result, it is possible for OnLive Cloud Gaming Service to be benefited by encoding WMV 9 format other than H.246 for non-first-person shooting games.

As for first-person-shooting games, both the background and the objects change rapidly. It would be wise to use MJPEG encoder instead because the compression ratio of H.246 and WMV 9 are not much better, and the video quality of MJPEG is better than the other two formats. Not just video quality, it will take less time and resource to compress the digital video images before streaming to users.

4.2 FUTURE WORK

OnLive is the not the only company that is impacted by the problem of bandwidth limitation. Almost all internet-related technologies are impacted by the limitation of bandwidth. For instance, cloud gaming on mobile phones as mentioned in Shu Shi's presentation is greatly impacted by this limitation. However, until the internet infrastructure is fully mature, it is crucial to reduce the amount of bandwidth required to transfer data.

A breakthrough in video encoding format can greatly compress the amount of information required to be transmitted across in different channels. Technologies such as video conference, video streaming services, and online 3D TV can be benefited.

REFERENCES

Waite, Stephen. Thought Leader Interview with Jules Urbach, Founder & CEO of OTOY. September 20, 2010. <http://blog.research2zero.com/wp-content/uploads/2011/01/R2-Thought-Leader-Interview-OTOY-Sept-2010.pdf>

Shi, Shu. Cloud Gaming on Mobile Devices. Spring 2011. <https://wiki.engr.illinois.edu/download/attachments/45449331/110420Cloud_Gamingfor_ECE498.pdf?version=1&modificationDate=1303395657040>

Symes, Peter. Digital Video Compression. McGraw-Hill Professional, 2004. <http://books.google.com/books?hl=en&lr=&id=GaKnpP39hv8C&oi=fnd&pg=PR15&dq=window+media+video+9+compression&ots=bo3C_yAmOc&sig=dHTndL7XRTPZamuhXPU4rpWDcg4#v=onepage&q&f=false>

CalArts. Intraframe vs Interframe. <http://ea.calarts.edu/post/Interframe_VS_Intraframe.pdf>

Salient Systems Corp. How Does H.264 Work? <http://www.salientsys.com/files/whitepaper/Understanding%20H%20264.pdf>

Spielfogel, Jason. Why we like MJPEG Compression. 2-6-2009. <http://www.securityinfowatch.com/The+Latest/why-we-mjpeg-compression>

H.264 Encoder. Freeware. GPL. <http://www.h264encoder.com>

GraphicEdit – Direct Show Graph Tool. Microsoft Corporation. <http://blog.monogram.sk/janos/tools/monogram-graphstudio/>