**Improving Remote Access of a Data-Intensive Computing Application: Effects of Encoding and GPU Virtualization**

Jerry Adams1, Bradley Hittle2, Eliot Prokop3, Ronny Antequera4, Dr. Prasad Calyam5

University of Hawaii-West Oahu1, The Ohio State University2, University of Missouri-Columbia3,4,5

*jerrya6@hawaii.edu*1*, bhittle@osc.edu*2 *, ebp25c@mail.missouri.edu*3*, rcb553@mail.missouri.edu*4*, calyamp@missouri.edu*5

Abstract

*Large and complex computing systems often require significant amounts of human intervention and visual inspection to produce useful results. An example is MRI scans which requires a human to not only be able to view an image but also be able to rotate it, view slices of it, and manipulate the image in other ways. However, the data set involved in such a program may be gigabyte or petabyte sized resulting in it being impractical to download for analysis. To resolve this issue remote access is used which raises new issues regarding how to optimize Quality of Experience (QoE) for the user. These remote access systems must have exceptional QoE to provide end users with the best performance possible. Upon examination there are many different options for enhancing the QoE by adjusting aspects of the Quality of Application (QoA) and Quality of Service (QoS). A decision tree model has been developed to provide context awareness through feedback loops to adjust the QoA and QoS. In these adjustments it is preferred to alter the client first, network second, and client third to cause minimal disruption.*

*This paper used the Remote Interactive Volume Visualization Infrastructure for Researchers (RIVVIR) system as a case study. The case study adjusted aspects of the client by altering the video encoding and color density selection to attempt to improve the QoE for the user. It also examined the scalability of GPU virtualization on the system by connecting multiple clients to test how many clients the server can support before degradation of QoE for the users. Experimental results showed that, contrary to default recommendations for the client, the Tight encoding scheme should always be used. For exceptionally poor connections where the Tight encoding scheme has lower QoE, lowering the color density is not an option due to significantly degrading the image. In addition, the server is able to support five users simultaneously without noticeable loss of QoE and appears capable of supporting several more.*

*In conclusion, encoding selection and GPU virtualization are valid ways to improve the QoE of a remote access system. Future work includes plans to conduct more subjective tests with human subjects to explore the conclusions of the encoding selection scheme. Also, implementing distributed load balancing for GPU virtualization could greatly improve server side resource allocation.*

# 1. Introduction

Remote Desktop Access (RDA) applications in the class of Virtual Network Computing (VNC) [1] allow remote viewing and control of computer resources through the internet. The vision of such applications is to provide an “at-the-server” experience, giving users equivalent performance and quality to being physically present at the computing resources which they are utilizing. Due to this, providing optimal levels of Quality of Experience (QoE) is thus vital to fulfilling the objective of RDA computing.

The QoE is partially determined by the Quality of Service (QoS), which refers to the performance and quality of the network connection between the client and server, such as available bandwidth, packet loss levels, and service delay. The other aspect that determines QoE is Quality of Application (QoA), which refers to the measurements such as image color density, image compression density, presence of image impairment events, application frame rate, and time require completing tasks. The general goal of this research is to improve the overall QoE by making adjustments based on the QoS and QoA.

To help optimize the QoE a “3Q” decision tree model was created to analyze the relationship between the many variables of the QoE, QoS, and QoA. The model also views the many variables of a RDA application and improves it in three sections; client, network and server. First improvements would be made are at the client side, where improvements are easiest to implement. Such improvements at the client are image compression, pixel encoding and pre- fetching images. Another area for improvements in this model is the server. Server side improvements are the most difficult to improve upon as the resources are “static” in the sense that they least able to be changed. Some improvements here are resource scheduling or sharing, hybridization of the client-server resources and even physical hardware upgrades. The decision tree model also provides feedback in that the user or human perceived feedback determines the configuration of the application in order to improve QoE. Likewise the application feedback is used to modify and control the QoS variables to meet the QoA requirements. If QoA adequately meets the needs of the user, then their QoE is positively impacted. If the QoA does not meet their needs, then the QoE is negatively impacted. In summary it takes any context awareness and prescribes actions to maximize user QoE.

# 2. Related Work

Previous work has been done in regards to improving the QoA and QoS of applications. One such paper [2] uses a human-and-network-aware encoding selection scheme that considers both the condition of the network and the perceived performance of the application by the end user. The best encoding type is automatically selected based on this feedback. That paper used a similar remote desktop access application as a case study to test their selection scheme and it shows that human-and-network-aware encoding selection scheme outperforms network-aware only schemes.

The “3Q” decision tree used is somewhat comparable to the QoE prediction model in [3]. However, that model attempts to predict QoE given QoS while this research is aimed at how to change QoA and QoS to achieve a desired QoE. Rather than simply predict poor QoE due to QoA and QoS factors, the goal is to determine methods to alter settings to compensate.

Altering the encoding schemes to compensate for QoS is one of the prime focuses of this paper and thus a variety of encoding schemes were needed. The following is a brief summary of several of the encoding types.

Tight was an attempt to increase both the compression and encoding speed of zlib and was developed specifically to try to improve performance on low band-width networks. It was also designed to have variable compression depending on the speed of the networks.

Zlib is a simple encoding that uses zlib library to compress raw pixel data. It achieves good compression but consumes a lot of CPU time. Support for this encoding is provided for compatibility with VNC servers that might not understand Tight encoding.

Hextile uses rectangles split up into 16 by 16 tiles that are sent in a pre-determined order. Data within the tiles is sent as Raw or a variant of RRE. This is typically the best choice for high speed (such as ethernet) networks (and in this study was the default choice for bandwidths above a certain amount).

Rise-and-Run-length-Encoding (RRE) uses sequences of identical pixels compressed to a single value and repeat count. This is very efficient for large blocks of constant color since less data needs to be transmitted. CoRRE is a variation that uses 255 by 255 pixel rectangles, thus allowing single-byte values to be used and reduces the packet size.

Raw is the simplest scheme which literally involves simply sending width by height pixels. It is the fastest when server and viewer are on the same machine as there is no connection speed to worry about and this scheme minimizes processing time.

A previous paper, [7] presents some bandwidth consumption statistics and spatial redundancy algorithms for some of those encoding schemes.

# 3. Problem

The primary focus of this paper is the bandwidth aspect of the QoS and the QoA delivered by experimenting with different encoding schemes in an attempt to improve QoE. QoS and QoA effectively act as inverses to deliver a desired level of QoE. For example, a server on a fast and stable network could use an encoding scheme that allows the server to send the image data to the thin client in a less efficient but easier to display format. If the network was slower, though, and the goal was to maintain the same responsiveness, tradeoffs would have to be made. One tradeoff would be to use an encoding scheme that has the server compress the data to make the transmission more efficient – but this results in the thin client needing to be able to process and unpack the data. If the thin client does not have the computing power to do so, however, another alternative would be to use an encoding scheme that sacrifices image quality in exchange for less data needing to be sent. Either of those examples would result in the same overall responsiveness on the user’s end. Similarly, if the goal is to maintain a given image quality on a slower network, either a more data efficient but computationally intensive scheme or less user responsiveness might be required. In addition, different encoding schemes have their own strengths and weaknesses – it is not always a linear tradeoff. One encoding scheme may sacrifice 10% image quality for 50% less data needed to be transmitted, which would be valuable on networks where speed is the a major factor.

The secondary focus of the paper is the scalability of the server to multiple clients to determine how many users the system can support without degradation of QoE. In most cases servers do not use all of their available resources to service one user and thus these leftover resources can be utilized to service other users. At the same time, it is important not to service too many users or inefficiently allocate resources and thus lower the QoE for the users. Performance testing using multiple clients was therefore undertaken in order to gauge the capacity of the server.

# 4. Methods

## 4.1. Remote Interactive Volume Visualization Infrastructure for Researchers (RIVVIR)

The University of Missouri (MU) in cooperation with Ohio State University (OSU) has developed a ‘Remote Interactive Volume Visualization Infrastructure for Researchers’ (RIVVIR) application that allows “health care” professionals to access virtual desktops (VDs) that host computationally intensive applications in a cloud platform. RIVVIR is a RDA application that allows interactive visualization of massive data sets such as MRI’s that can be accessed over the internet with low powered thin-clients or mobile devices. This application is used as a case study to verify the 3Q decision tree model.

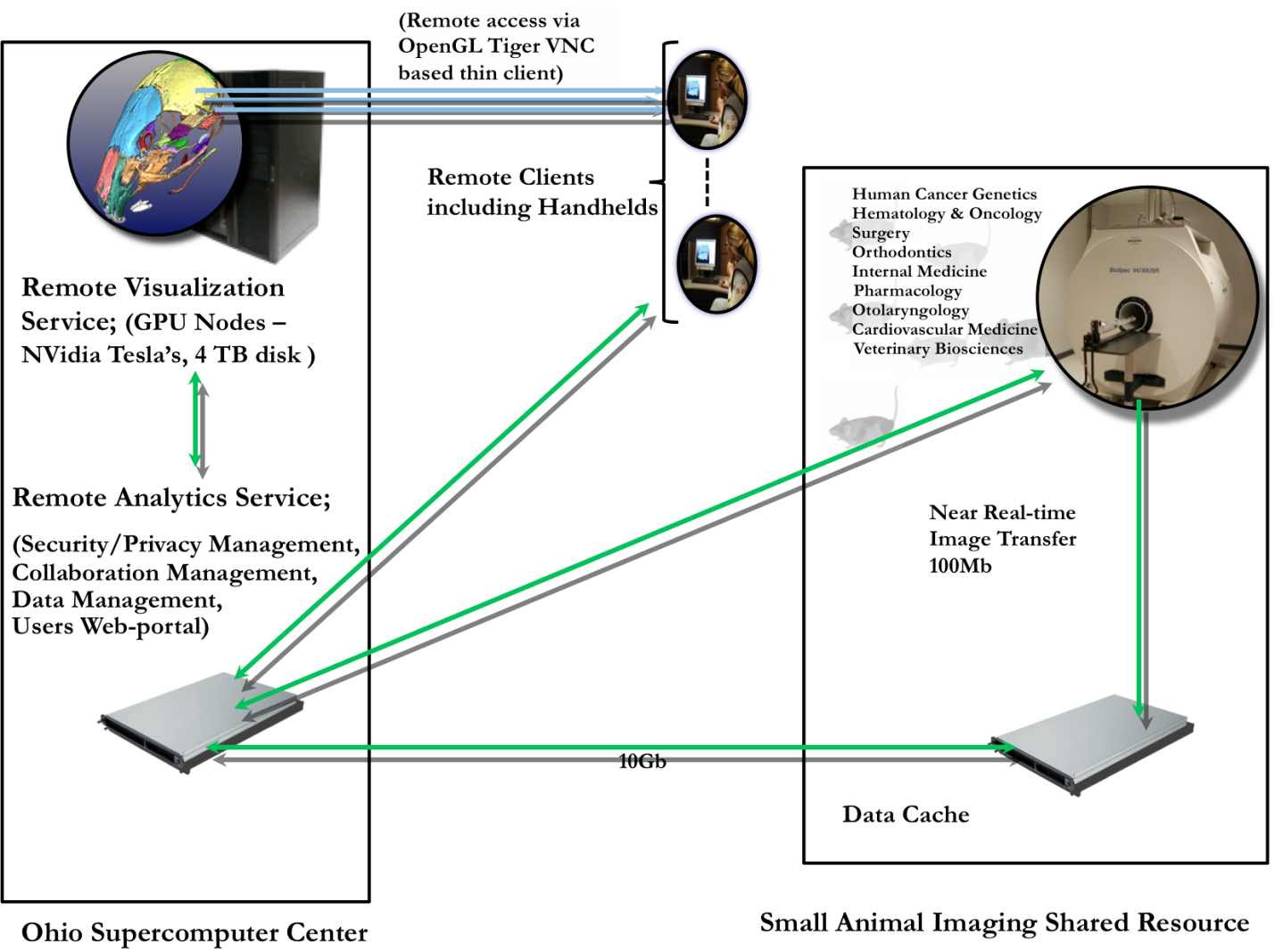


Figure 1: Overview of the RIVVIR system

## 4.2. Encoding selection scheme

In order to help RIVVIR users determine the best encoding scheme for a given network situation, this study tested various encoding schemes at several levels of speed. For each of the following trials, the website Speedtest.net was used to determine relative network speed and then the performance of the VNC client itself was measured. First, baseline wired readings were taken on a solid network. Second, wireless readings were taken for a network using an individual router. Finally, poor wireless readings were taken by simulating a poor network via simply moving the thin client further from said router to the point where the network connection was severely degraded.

For obtaining results both subjective and objective data was gathered. The encoding scheme trials featured objective measurements of the bandwidth usage for each scheme and subjective measurements of user QoE utilizing a tournament method to determine the best QoE overall. The scalability trials used subjective measurements to determine if QoE noticeably degraded for the users and objective measurements of GPU utilization.

The RIVVIR application uses the VNC protocol to connect an end user to the VNC server running at OSU. VNC uses the RFB protocol which allows for pixel encoding. Pixel encoding this case refers to the encoding of pixels that can be used to transport the image data in which VNC generates.

This paper tests ten encoding types: Tight, ZRLE, Zlib, ZlibHex, Ultra, Hextile, ZYWRLE, CoRRE, RRE and Raw. Each encoding type has the option to be encoded in four different color densities: Full Colors, 256 Colors, 64 Colors and 8 Colors. Every encoding type along with each of the four color densities are tested against different arbitrary network conditions such as available bandwidth, packet loss or delay. End users have many VNC clients to choose from, this paper uses one of the more popular clients, Ultra VNC [8]. Ultra VNC allows for easy selection of encoding schemes along with color density through a GUI or command line.

## 4.2. GPU Virtualization

GPU Virtualization allows for the sharing of a single physical GPU with multiple users. GPU Virtualization on the RIVVIR system is accomplished by virtualizing or brokering the physical GPUs using 3D X Server. The 3D X Server acts as the hypervisor, providing a layer of abstraction between the physical GPU and virtual desktops, or the X Proxy Displays as it is known in this environment. The X Proxy Displays and their applications behave as if they have their own dedicated GPU and the server’s physical GPU and driver think they are responding to one master host. 3D X Server intercepts the applications API calls and translates commands through the X protocol to the X Proxy display. VNC Server then transfers the display to user and receives remote client keyboard and mouse events.

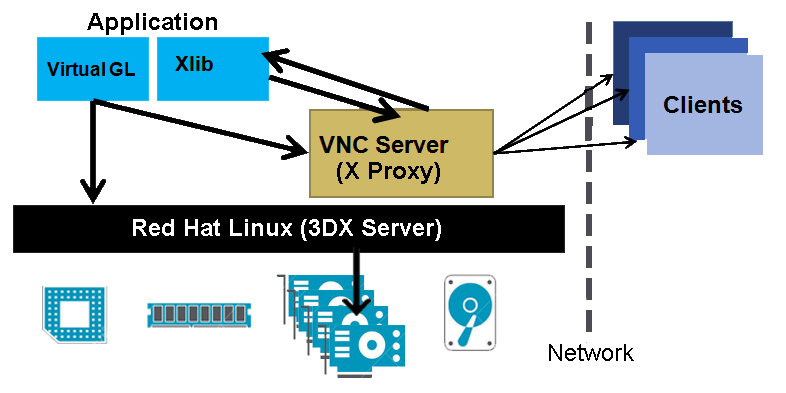


Figure 2: GPU Virtualization on the RIVVIR system

# 5. Implementation

## 5.1. Encoding selection scheme experimental setup

UltraVNC was chosen as the software because it featured a wide variety of encoding schemes as opposed to other VNCs such as TurboVNC and TightVNC. As indicated earlier, the fact various schemes being tested use radically different methods implies specific encoding schemes may be preferable in certain situations. Thus, having a wide variety of schemes to test is important.

During this testing all selections were held constant except for the encoding scheme itself unless specified otherwise. The default selections (which were maintained) were Full Colors, CopyRect Encoding enabled, Cache Encoding enabled, Zip/Tight Compression of 6, and Jpeg (Tight) Quality of 6.

Ideally, for subjective measurements, every encoding scheme would be compared against every other encoding scheme. However, this results in N! comparisons which was impractical in this case. To circumvent this problem a tournament scheme was utilized which reduces the comparison from N! to log(N). In addition, every encoding scheme is implicitly compared against every other encoding scheme using this method.

Since it was possible for no encoding scheme to offer sufficient QoE at full colors, experimentation was done with the 256 colors, 64 colors, 8 colors, 8 dark colors, 4 grey colors, and black and white settings to determine if any maintained sufficient image quality while offering improved responsiveness and thus increasing the overall QoE.

Subjectively speaking, responsiveness and image quality were used to determine the quality of an encoding scheme. Responsiveness refers to how quickly an image responds to commands by the user and image quality refers to how clearly the user can see the smallest details of the subject.

Objectively speaking, the amount of data transmitted across the connection while loading and displaying a rotating MRI for a 60 second period was measured. The data transmitted to load the desktop view when opening the VNC connection was noted and then the MRI was loaded and a timer started. At the end of the 60 seconds the final amount of data transmitted across the connection was noted and the difference of the final amount and initial amount was the data needed to load and maintain the rotating MRI scan.

These two measurements interact in several ways. First, the objective measurement serves as validation and explanation for the subjective performance of certain encoding schemes. One encoding scheme may perform exceptionally well subjectively on a fast network and poorly on a slow network – which is then objectively explained if the encoding scheme is shown to consume a large amount of bandwidth which is simply not available on the slower connection. Second, if two encoding schemes perform subjectively the same then the scheme which consumes less bandwidth should be considered the superior option.

Three networks were used in this research. Network 1 was a wired connection, Network 2 was a wireless connection, and Network 3 was the same wireless network as Network 2 except the thin client was moved about 10 meters away from the router to the point where it could barely connect to the network. The following information was determined using the website SpeedTest.net and the Elite Systems, LLC server.

|  |  |  |  |
| --- | --- | --- | --- |
| Network | Latency | Average Download | Average Upload |
| **Network 1** | 40.00 ms | 38.63 Mbps | 5.07 Mbps |
| **Network 2** | 40.67 ms | 20.69 Mbps | 5.06 Mbps |
| **Network 3** | 45.67 ms | 0.73 Mbps | 2.12 Mbps |

Figure 3: Networks used in encoding experiments.

## 5.2. GPU virtualization experimental setup

The GPU scalability of the system was measured in two ways. First, a subjective QoE baseline for each client machine was established individually using the winner of the encoding scheme trials (which happened to be the Tight encoding scheme). Then, as more clients were added to the server, each client was tested to check whether there was a noticeable QoE degradation. Second, the GPU utilization for various combinations of thin clients was objectively measured to determine how many resources were still available. To gather this data the GPU utilization was measured every second for 15 seconds and the resulting average was used.

There were five client machines used for the research described in this paper. Laptop 1 was utilized for all objective encoding scheme testing, half of the subjective encoding scheme testing, and the scalability trials. Laptop 2 was used for half of the subjective encoding scheme testing and the scalability trials. Laptop 3, Tablet, and Smartphone were used solely for the scalability trials. See Figure 4.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Device | Model | CPU | RAM | GPU | O.S. |
| **Laptop 1** | Asus G50V | Intel Core T5750 @ 2.00 GHz | 4 GB RAM | NVIDIA GeForce 9700M GT | Window 7 |
| **Laptop 2** | Asus U36SD | Intel Core i5-2410M @ 2.3 GHz | 8 GB RAM | NVIDIA GeForce GT 520M | Window 7 |
| **Laptop 3** | Apple MacBook Pro | Intel Core i5 @ 2.5 GHz | 4 GB RAM | Intel HD Graphics 4000 | OS X 10.9.1 Mavericks |
| **Tablet** | Apple iPad (1st Gen) | ARM A4 SoC @ 1 GHz | 256 MB RAM | PowerVR SGX535 | iOS 5.1.1 |
| **Smartphone** | Samsung Galaxy S4 | ARM Krait 400 @ 1.9 GHz | 2 GB RAM | Adreno 330 | Android 4.1 Jelly Bean |

Figure 4: Specifications of client machines

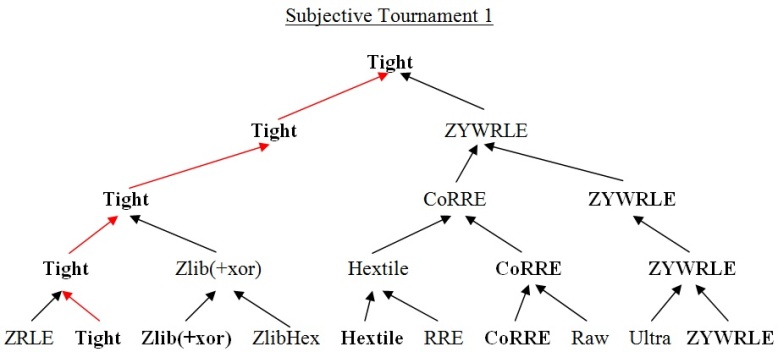
# 6. Experimental Results

## 6.1. Encoding selection scheme subjective results

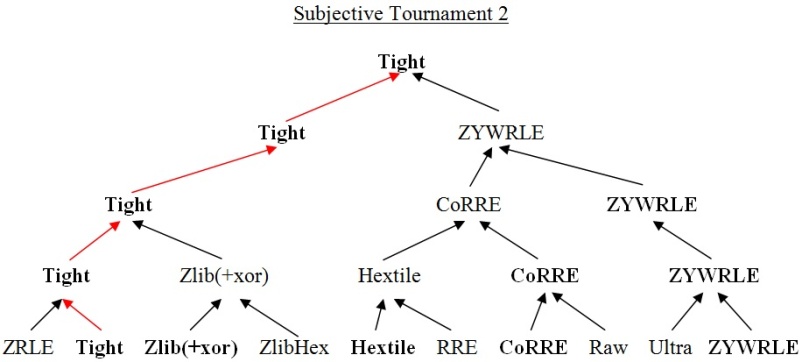
Unless specified otherwise, no particular encoding scheme offered superior or inferior image quality compared to others for the data set used for this research as far as the researchers could tell. It is possible that professional end users of MRIs may be able to tell a difference but none was visible to the layman’s eye in this experiment. Thus, the subjective QoE measurement relied primarily on load time and responsiveness.

Two different machines (Laptop 1 and Laptop 2) were used for the subjective measures. Though the QoE was worse on the second laptop compared to the first, the relative ranking of each encoding scheme and the winner of each comparison was the same.

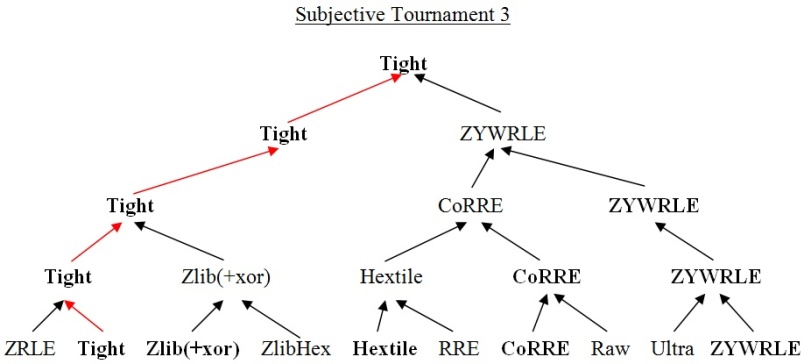
The first subjective tournament was done on Network 1. The winner was a type of encoding known as Tight which offered extremely high responsiveness and low load time along with equal image quality as noted above.



The second subjective tournament was done on Network 2. The winner was also the Tight encoding scheme which maintained exceptional responsiveness and load times.



The third subjective tournament was done on Network 3. The winner was yet again the Tight encoding scheme but the particularly (and artificially) poor network conditions resulted in noticeably lower user QoE.



## 6.2. Encoding selection scheme objective results

The first objective trial was done on Network 1. The lowest bandwidth consumed was the Tight encoding scheme.

Objective Trial 1

The second objective trial was done on Network 2. The lowest bandwidth consumed was again the Tight encoding scheme.

Objective Trial 2

The third objective trial was done on Network 3. The lowest bandwidth consumed was once more the Tight encoding scheme.

Objective Trial 3

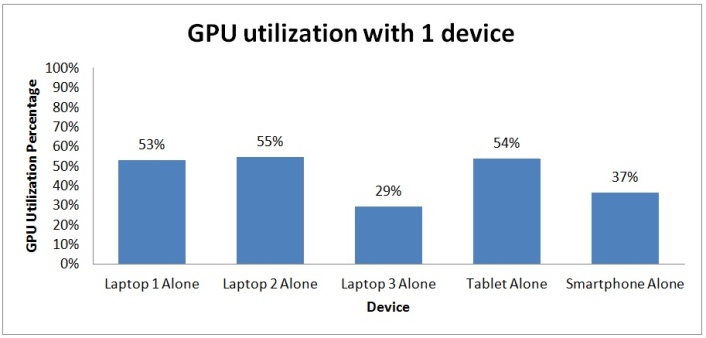
## 6.3. Encoding selection scheme color density trials

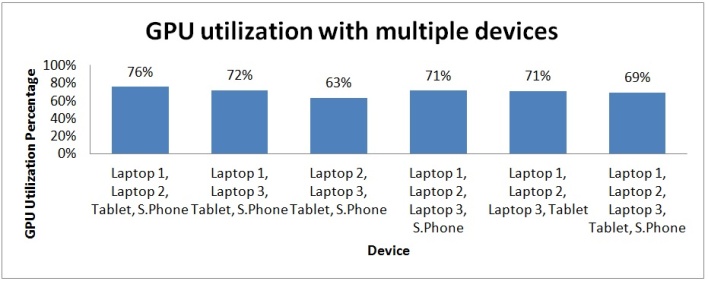
Since even Tight, which rated the highest both objectively and subjectively on all networks, offered less than ideal QoE on the poor network, experimentation was done with altering the color density for Tight and several other encoding schemes which rated well. Theoretically using a less complex color scheme could improve performance and thus QoE on these exceptionally poor networks.

Unfortunately, none of the color scheme options below Full Colors offered satisfactory QoE in terms of image quality. In the best case scenarios the MRI lost incredible amounts of detail to the point where significant features of the image were not recognizable. In the worst case scenarios entire portions of the MRI essentially vanished and appeared blank due to the color scheme and thus were not visible to the user.

## 6.4. GPU Virtualization scalability results

Trials to determine the scalability of the system for servicing multiple clients simultaneously were also done. Each machine had a baseline QoE established while being the only connected client. Next, different machines were added in different combinations in different trials up to all five machines acting as clients simultaneously. None of the clients suffered any noticeable loss of QoE.





# 7. Discussion

First, for every network, Tight consistently performed the best on both the objective and subjective measurements with equivalent image quality, extremely high responsiveness, and low bandwidth usage. This would seem to indicate that with RIVVIR the limiting factor is essentially always the network connection and thus hybridization is always useful so less data needs to be sent from server to client. Tight does a far better job of compressing the data than a scheme such as Hextile. Hextile, incidentally, is the default recommendation for higher speed networks, particularly wired ones, yet is outperformed by Tight even on those networks. ZRLE or a similar encoding scheme is often recommended for medium speed networks, yet those encoding schemes were again outperformed by Tight. There does not seem to be a reason to ever use an encoding scheme other than Tight for RIVVIR.

Second, in addition to determining that Tight is the best choice for VNCs where the user must manually select a scheme, the user can also outperform the automatic optimization algorithm for UltraVNC. If told to automatically select settings for the wired network it will choose Hextile, yet there is a pronounced difference in QoE between Hextile and Tight with Tight being vastly superior in responsiveness. If told to automatically select settings for the wireless network it will choose ZRLE, which again demonstrably performs subjectively and objectively worse than Tight. Perhaps in other settings and other applications those encoding schemes may prove superior, but for accessing RIVVIR the Tight encoding scheme appears to be the best choice in all situations in spite of the automatic selection.

Third, on particularly poor networks even Tight encoding shows a degraded QoE. However, attempting to decrease the color density is not a viable option as even 256 or 64 colors removes large amounts of detail or flat out makes part of the MRIs invisible in some cases. Ideally these very low performance networks are not encountered but otherwise a different solution will need to be found.

Fourth, it appears that the current RIVVIR setup is adept at handling multiple clients simultaneously without noticeable degradation of QoE. Laptop 3 (the Mac) appears to cause far less GPU utilization for some unknown reason. The Smartphone (the Android) also seems to require less GPU utilization from the server and, oddly, enough, having all five client machines connected to the server caused less overall GPU utilization than most combinations of four machines. No combination of clients, however, ever caused more than 80% GPU utilization of the server.

# 8. Conclusion

This research demonstrates that superior user QoE for RIVVIR can be achieved through smart manual selection of encoding schemes. Despite the general rule of thumb that Hextile tends to be the best choice for fast wired connections, Tight performed better subjectively while using a fraction of the bandwidth. However, it is possible that an even faster connection and/or a wired LAN connection may result in superior QoE for Hextile.

The results also show that manual selection can outdo the automatic selection algorithm used by UltraVNC and quite possibly other VNC applications. There was a significant improvement in user QoE compared to the automatic setting when choosing Tight or even several other encoding schemes (which themselves were slightly inferior to Tight).

In regards to scalability, the server appears capable of handling five clients simultaneously without loss of QoE and would seem to be capable of servicing several more by extrapolation, though further research is needed on this front. Some clients also appear to use less server resources than others and certain combinations of more clients require fewer resources than combinations of fewer clients. Again, more research is needed.

Further testing with human subjects would help explore the conclusions about encoding scheme selection. An ideal set-up would include thin clients with both wired and wireless access and ideally emulators to artificially produce a poor network for some thin clients for testing purposes. Subjects would use both the tournament model and Mean Opinion Scores (MOS) to determine which encoding schemes perform the best – preferably without having any prior knowledge of which thin clients are wired/wireless/poor network and without knowing which encoding scheme is being used on each thin client. The users could also be given access to thin clients which use the automatic selection algorithm in order to demonstrate the benefit of using manual selection to improve performance and QoE while using the RIVVIR system.

# 9. References

[1] T. Richardson, Q. Stafford-Fraser, K. R. Wood, and A. Hopper. (1998). “Virtual Network Computing.” IEEE Internet Computing, 2, 33-38.

[2] P. Calyam, A. Kalash, A. Krishnamurthy, G. Renkes. “A Human-and-Network Aware Encoding Adaptation Scheme for Remote Desktop Access.” IEEE Workshop on Multimedia Signal Processing (MMSP), 2009..

[3] Vlado Menkovski, Adetola Oredope, Antonio Liotta, and Antonio Cuadra Sanchez, 2009. “Predicting Quality of Experience in Multimedia Streaming.” In proceedings of the 7th International Conference on Advances in Mobile Computing and Multimedia (MoMM ’09). ACM, New York, NY, USA, 52-59.

[4] T. Richardson. “The Remove Frame Buffer (RFB) Protocol.” <http://www.realvnc.com/docs/rfbproto.pdf>. 2010.

[5] K. Kaplinsky. “VNC Tight Encoder-data Compression for VNC.” Proc. of Scientific and Practical Conference of Students, Post-graduates and Young Scientists, 2001.

[6] P. Deutsch and J-L. Gailly. “ZLIB Compressed Data Format Specification.” IETF RFC 1950. <http://www.zlib.net>. 1996.

[7] W. Jiang, H. Jin, and et. al. “A Novel Remote Screen Synchronization Mechanism for Ubiquitous Environments.” Symposium of Pervasive Computing and Applications, 2006.

[8] Ultra VNC - http://www.uvnc.com/