USE IMAGE STREAMING TECHNOLOGIES TO PRESENT HIGH RESOLUTION IMAGES ON THE INTERNET

Steven Y. Hu, Vincent Tao
Geospatial Information and Communication Technology (GeoICT) Lab
Department of Earth and Space Science and Engineering
York University
Toronto, Ontario, Canada M3J 1P3
Email: stevenhu@yorku.ca, Email: tao@yorku.ca

Huayi Wu
State Key Lab of Information Engineering for Surveying, Mapping and Remote Sensing (LIESMARS)
Wuhan University, 129 Luoyu Road, Wuhan, 430079, P.R.China,
Email: wuhuayi@mail.liesmars.wtusm.edu.cn

KEY WORDS: Internet/Web, GIS, Mapping, Development, Image, Resolution, Software, Web based

ABSTRACT:

Geospatial information services over the Internet become a technological trend driven by the innovations of geospatial information and communication technology. One key issue is how to present high resolution images with high performance. The biggest challenge of high-resolution images is their huge sizes of data even after the finest image compression. It is impossible that we download the whole image, which is of hundreds Megabytes after compression, then display it. This paper presents the implementation of a special browser (GeoServNet, www.geoict.net) for processing large images over the Internet. In order to gain the best performance, a lot of optimizations should be done. It implements a level of detail (LOD) algorithm to display images in 2D scene which means it will always process images near the viewers' point. It also implements an image pool and streaming algorithm to cache and deliver the images which are viewed again, the data can be directly retrieved from the cache instead of request it again from the server. The system has been tested in different bandwidth networks from low speed to high speed.

1. Introduction

Geospatial information services over the Internet become a technological trend driven by the innovations of geospatial information and communication technology. There are few commercial systems emerged in late two years, such as two dimensional systems, MapInfo, MapGuide, etc. and three dimensional systems, TerraExplorer, EarthViewer3D and G-Vista etc. Until now, GeoServNet (GSN), developed by the GeoICT Lab, York University, is the only advanced Internet-based geospatial information system (GIS) having the capability of 2D and 3D analysis in the world. GeoServNet serves as a web-based GIService engine, offering both 3D and 2D geospatial exploration capabilities to access heterogeneous spatial data sets over a distributed environment. It integrates vectors, images, DEMs, 3D models, multimedia dada, documents and databases seamlessly in an open, secure, scalable and fast fashion. The system will handle several key data: vectors, images and DEMs. It is apparent that the challenge work is the large data set and the limited computer and network resources.

Currently, there are two development directions for Internet-based web-mapping systems. One is traditional client/server based web-mapping system and the other is web service based web-mapping system. The latter uses a common protocol to provide different services to the users on the Internet. The most famous web services system is standardized by OGC (Open GIS Consortium). Even though it is a trend for developing web-mapping system by web services technology, the low performance remains a big problem for this type of applications. Due to the web services are negative systems and their protocols are based on XML language, it will be very difficult to make an application with the streaming technologies which are the most useful technologies to speed up the data exchanges within Internet applications. That is why most of web-mapping systems still keep the traditional client/server architecture. By definition of itself data structure and transform protocol, this type of application can be designed being much more fancy and can gain higher performance than web services systems. Recently, most commercial web-mapping systems mentioned in abstract are not web services based and they have their own system architectures and internal data formats and they are not compatible to each other. Additionally, high-quality images have become a mission critical component of professional techniques and operating procedures in most fields of academic and corporate endeavor. Affected arenas include but are by no means limited to: education, medicine, criminal justice, defense, commerce, entertainment and more. The ability to capture high-quality images has increased dramatically in the last decade and has, in fact, surpassed the ability to share and make use of the level of resolution now commonly achievable. Digital cameras now surpass film in capture quality and are approaching film in annual exposures with over 9.1 billion exposures in 2000 to 29.5 billion expected in 2005 (IDC, March 2000). Multi-mega-pixel cameras now capture images so large that storage costs have become a major implementation consideration and Web-based presentation a daunting technical challenge. Evolving at an equally unprecedented pace, scientific and security-related imaging systems previously capturing megabytes of data through complicated manual processes over many hours now capture gigabytes of data automatically in minutes.

This paper shows a new approach which uses a traditional image visualization technology to make large ultra-high-resolution images fast and interactive on the Web and expands it to be used as an infrastructure of 2D applications. The technology is accomplished using image *streaming*: incremental, on-demand access to image data by level of detail and area of view. It also explains how GSN's image streaming technology works and how it is implemented to support 2D GSN applications: 2D viewer. It also discusses the technologies which are used to cache huge image data set on client side. Chapter 2 will talk about some difficulties exist in Internet-based Image application. Two techniques pyramid and streaming will be discussed in chapter 3. Chapter 4 presents the implementation of the Image Streaming System – GSN image. And a test and conclusion will be introduced in chapter 5 and chapter 6.

2. Difficulties for Internet-GIS Image Applications

Imagery forms an essential and well-understood part of GIS applications. There are unique challenges associated with making effective use of imagery, especially when providing Internet or Intranet GIS web mapping solutions. But image size is extremely larger. Even city wide coverage with high resolution color air photos can reach 300GB to 1.5TB for a typical GIS web mapping application covering a city. As can be seen, a GIS web server needs to be able to easily serve 300GB to 40TB of imagery. Yet the benefits from using imagery are compelling (for example, vector GIS and CAD data is often created from imagery as a starting point), and it provides an intuitive, accurate and up to date view of the real world for users.

Basically said there is only one problem which needs to be solved exists in Internet-GIS image applications: the conflict between extremely large data set and limited Internet bandwidth. Besides image size, network is another issue for Internet based image applications. Internet is the most complex network around the world. It includes a range of networks. The increasing availability of high-bandwidth network access greatly eases access to images of moderate quality. Images of high quality, however, easily exceed the capacity of even the highest volume transmission technologies. While hundreds of kilobytes of data are commonly downloaded, megabytes, tens of megabytes, and certainly gigabytes of data are impractical to transfer over current infrastructure. Advanced compression techniques achieve 4 to 1 or even greater *lossless* compression and 10 to 1 or even 100 to 1 *lossy* compression. Even in combination with a high-bandwidth connection, however, compression is inadequate for publishing high-quality images. A 32 megabyte image licensed from a stock photo house – even if compressed 100-fold with wavelet technology to approximately 320k – is inconvenient for Web viewing. The time required to download larger files makes them impractical to publish without image streaming. Only the streaming technology can support real-time viewing of images many gigabytes in size.

Studies have shown that around 60% of the Internet users' access speeds are 56Kbit/s and about 30% of the Internet users are using high speed connection. The average speed for high speed users are 2Mbit/s and therefore the mean connection speed of Internet users is 1Mbit/s. What will happen if we use this bandwidth to display an image at 2048X2048. The image size with raw format will be 1024*1024*3*8 = 96Mbit. That means it will take 96 second to transfer it to the user who want to see it. One easiest way to reduce the transportation time is to compress the image to reduce its size. For instance, we can store it as JPEG image with ratio of 10 and the response time will be 1/10th of the original time (9.6 second). But it is still a problem. For normal high resolution images in JPEG format, their size is 800Mbit, we have to figure out other solutions to solve this problem.

3. Principle of image pyramid and image streaming

It is well known that the decompression of images will take a lot of time and memory. Here is an example running on a PIV2.0GHZ/500M machine. Using Adobe Photoshop6 to open a JPEG image file will take 5 minutes and 40 seconds. It will use 2G hard disk to store the temporal data. The 80MB file, which covers Canada East area, is a 27845 x 23177 in width and height and 100m resolution image. Therefore, all the available Internet-GIS image systems will pre-process the images to a new format image data set first before put them to the server. The new format is normally called image pyramid. In order to handle the image pyramid via the Internet, a technique which is called image streaming has been applied to it.

As we all know, every image has a resolution. If you want to buy a digital camera, the first question you ask yourself is what the maximum resolution is. Different resolutions of an image mean different quality and different size of it. In web-mapping system, the multi-resolution approach is very useful.

- Images usually contain features of physically significant structure at different scales of resolution
- For some problems, this allows us to select a desired level of detail
- For some problems, the coarse-to-fine approach can reduce the computational complexity
- There is strong evidence that the human visual system processes information in multi-resolution fashion
- Image compression

Pyramid representation is another important technical factor. An image pyramid is formed by repeated smoothing and subsampling of an image ("Subsample" = discard pixels). There are two very common pyramid: Gaussian pyramid and Laplacian pyramid (Burt and Adelson, 1983). Each new level of a Gaussian pyramid is formed from the previous level by smoothing and subsampling.

Image Streaming, on the other hand, allows the Slideshow to start running before all images are loaded. With Image Streaming, an image is shown immediately once it is loaded; while the rest of the images are being loaded in the background. The Slideshow can even loop with this partial list of loaded images. Eventually, all images will have completed loading and the Slideshow will run in its full form. Image Streaming dramatically cuts down waiting and loading time thus improving the user's experience.

In order to implement image streaming, large images need to be reconstructed with image pyramid and image cutting. Pyramid is a technology where a high resolution image is created and saved in a multi-resolution format. If you think of it as a pyramid, with the full resolution at the base, say 3000x3000 pixels (Base layer in Figure 3.1), then 1500x1500, all the way down to the thumbnail (Top layer in Figure 3.1), which is 10-20K in size. If you click on the thumbnail, you are taken to the next level of resolution. At that level, the image is broken up into different tiles and we only serve up the tiles that are necessary to make that part of the image appear.

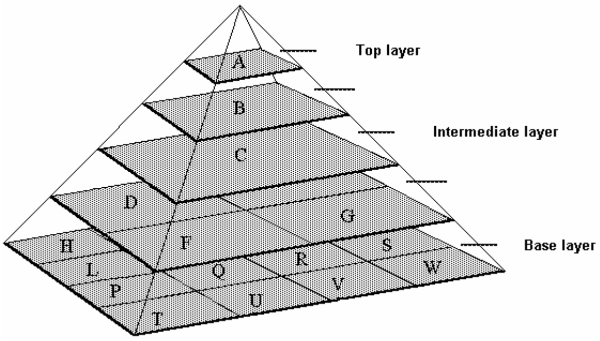


Figure 3.1. Image Pyramid

4. Implementation of the System

Working with the pyramid on the server side, GSN client is easily to be designed. The easiest and highest performance solution is to keep an image pyramid which is the same as the pyramid on the server side.

Firstly, GSN builds an image pyramid on the server side for a large high-resolution image. Some information is needed to notice GSN client: the original image resolution, the bounding box of the image, the maximum pyramid level and the minimal pyramid level. On the same time, every pyramid image in different level needs to be cut to small image partitions (128X128, 256X256 or more). In GSN system, they are represented by Image Objects.

Secondly, an image pool is needed to store images which have been downloaded from server. This pool acts as a cache of your web browsers or windows file systems. In fact, it is not only a cache which serves GSN client, but also an image server on the client side. Instead communicating with GSN server, GSN client send image requests to this image pool. The image pool will transfer and store image partitions to the client side. While the client request images, it will search the image pool first, if image is in pool, it will send the image partitions requested to display module. Otherwise, it will forward requests from display module to the server if the requested images are unavailable from the pool.

Finally, an overview image needs to be stored in GSN client. It is important to generate image overviews so that users can familiarize themselves with large-scale imagery features. Getting lost in the trees really can happen when viewing forested images. Having an overview image makes viewing a large image much the same as viewing a map. Users can quickly identify roads, landmarks and water bodies in overviews, then zoom in on regions of interest. Without an overview, or at least geographically coordinated displays, most users quickly become frustrated with orientation. Additionally, an overview image can help image pool response users' request as soon as possible. Users' operation is usually unpredictable. Therefore, it will easily happen that the images requested by users are not in image pool. In order to present map smoothly without "black" screen, GSN image pool will return images re-sampled from overview image immediately to display module. The users' operation will be seamless.

It looks like two modules are needed for expand GSN client. Actually, the GSN client should be divided to four modules and each module itself must be a thread. They run concurrently but without interrupting and communicating with each other.

- Display Module
  This module has four features: response to all users' operations from both mouse and keyboard, drawing map or images to users' screen, all APIs related with map presentation, and all APIs about map analysis.
- Image Pool Module
  This module has three features: process users' requests, store images loaded from server and manage pool.
- Draw Image Object Module
  It will draw Image Object on users' screen. In this paper, Image Object refers to basic elements which consist of image pyramid.
- Image Streaming Transmission Module
  This module communicates with GSN server and gets GSN client required Image Objects. It can transmit Image Objects either serially or concurrently.

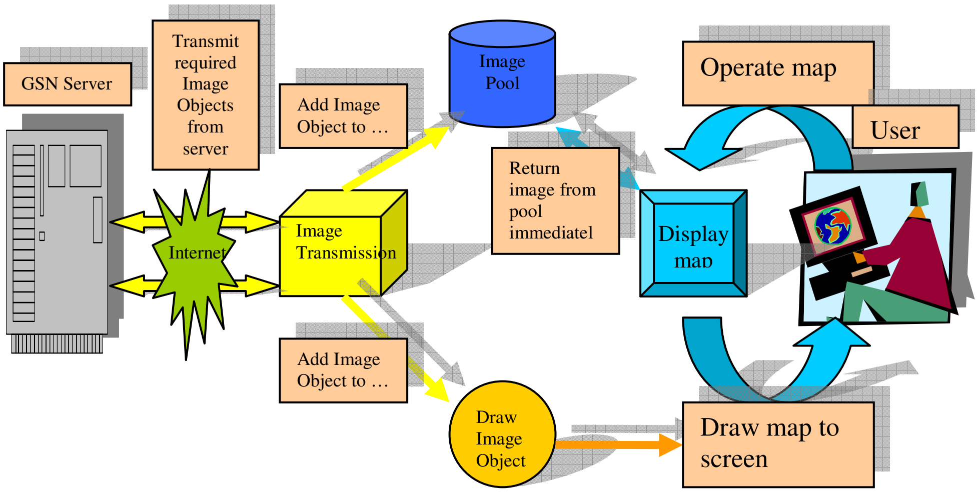


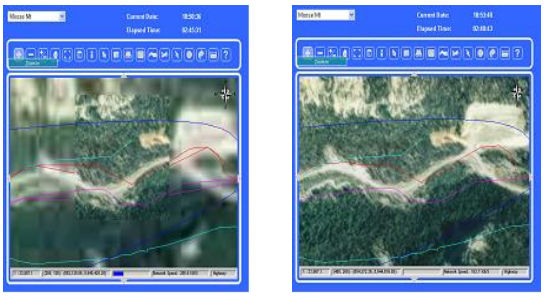
Figure 4.1. Improved GSN client structure

Figure 4.1 is the improved GSN client structure which includes all the modules and their processing flow. “Add Image Object to…” mean that Image Transmission Module will add Image Objects that it gets from server to both Image Pool Module and Draw Image Object Module. “Return image from pool immediately” means that Image Pool Module will send Image Objects back to Display Module from pool if they are available in the pool or from overview map if they are not available. And then, if users’ requested Image Objects are not in the pool, it will forward users’ requests to Image Transmission Module. “Transmit required Image Objects from server” are used for transferring Image Objects between GSN client and GSN server. Its implementation can be either single thread or multi-threads.

4.1. Display Module

The display module is the interfaces between GSN system and users. It processes all operations of both mouse and keyboard from users and then responses or presents with related results to them.

- Zoom In
- Zoom Out
- Smooth Zoom
- Pan
- Zoom to Full Extent
- Toggle Legend
- Identify
- Show Attribute Table
- Find Feature
- Toggle Overview Map
- 3D View
- 2D Fly
- Stop
- Refresh
- Measure Distance
- Hyperlink



(A) (B)

Figure 4.2. Zoom In Operation: (A), (B)

Figure 4.2 shows the algorithm of GSN display module. From it, it will be very easy to conclude that the response time of display module to users’ operation is only related with users’ computers hardware and software.

Figure 4.3 is the flow chart of its algorithm.

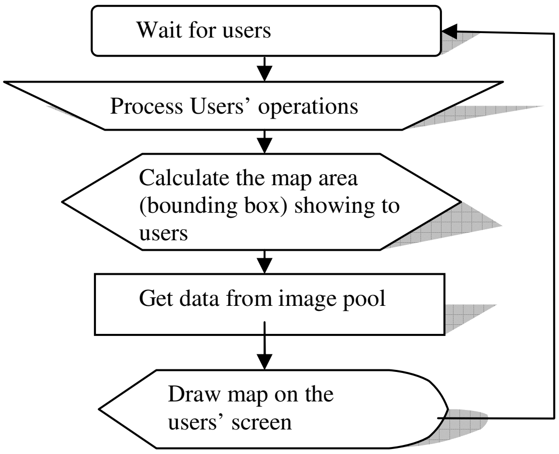

Wait for users
Process Users’ operations
Calculate the map area (bounding box) showing to users
Get data from image pool
Draw map on the users’ screen

Figure 4.3. Flow Chart – Display Map

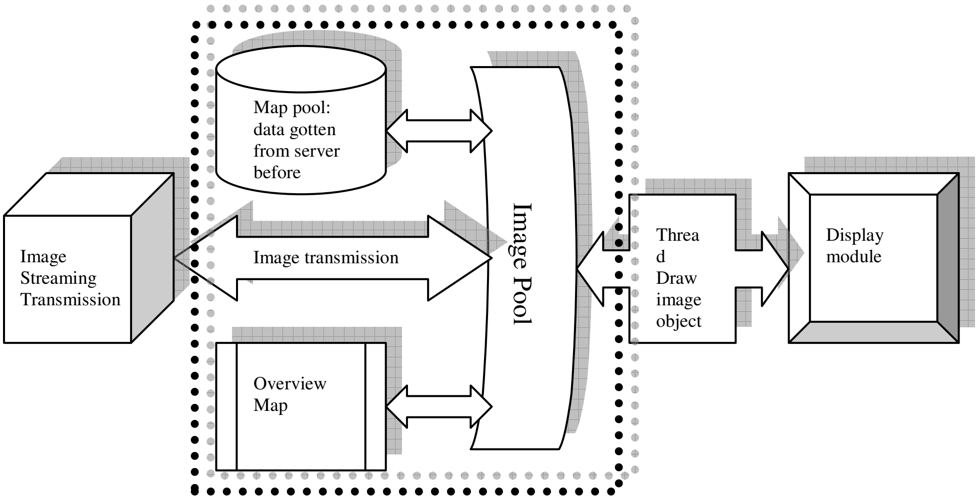


Image Streaming Transmission
Map pool: data gotten from server before
Image transmission
Overview Map
Image Pool
Threa d Draw image object
Display module

Figure 4.4. Image Pool

4.2. Image Pool Module

The pool will process users’ request, store images loaded from server and manage pool. Image pool is showed within dashed lines in Figure 4.4. Figure 4.5 is the flow chart of image pool. It is a thread. There is a queue in image pool. It is used to communicate with other modules. There is also a interface which will be used to accept image objects coming from image streaming transmission module.

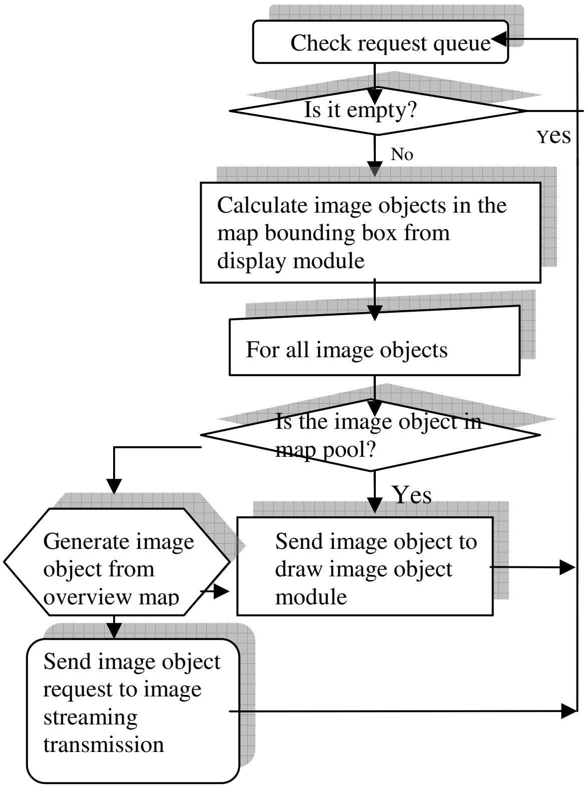

Check request queue
Is it empty?
Yes
No
Calculate image objects in the map bounding box from display module
For all image objects
Is the image object in map pool?
Yes
Generate image object from overview map
Send image object to draw image object module
Send image object request to image streaming transmission

Figure 4.5. Flow Chart – Image Pool

4.3. Draw image object module

The essential issue is to make sure the final image is seamless. For this reason, every image object (tile) is assigned a bounding box. Where to put the tile is decided by its bounding box and the screens bounding box.

4.4. Image Streaming Transmission Module

This module as shown in Figure 4.7 links the GSN client and GSN server.

There is one important thing needed to do before save image objects to image pool which is to call image pool memory management method to maintain the footprint of the pool. Otherwise, system will crash after the pool burns all the memory of users' machine. Figure 4.6 is its flow chart of algorithm.

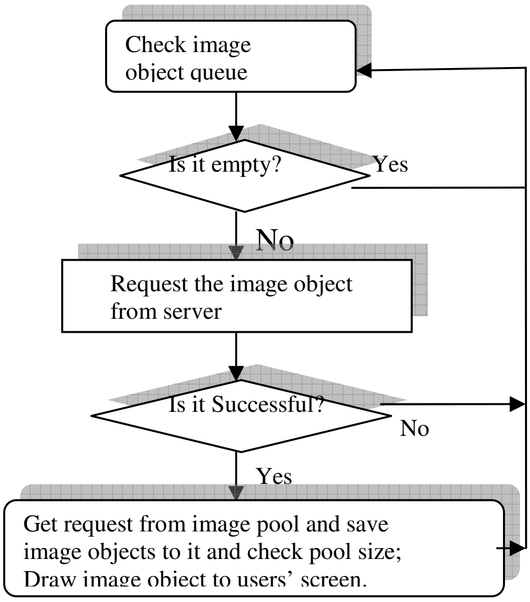

Check image object queue
Is it empty?
Yes
No
Request the image object from server
Is it Successful?
No
Yes
Get request from image pool and save image objects to it and check pool size; Draw image object to users' screen.

Figure 4.6. Image Streaming Transmission Module

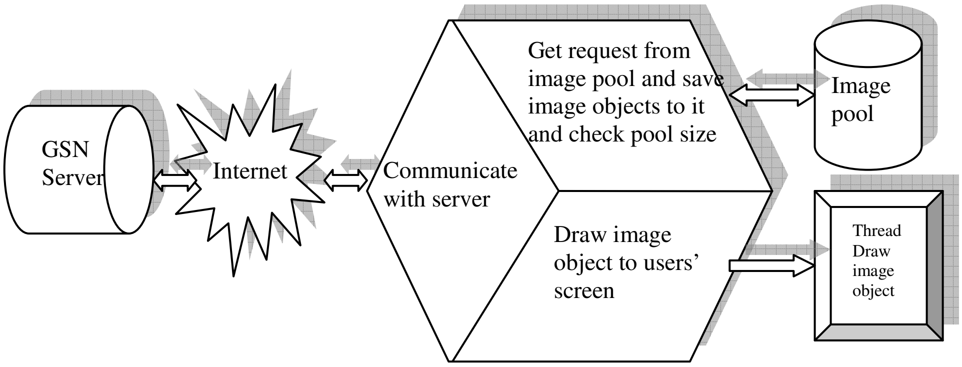

GSN Server
Internet
Communicate with server
Get request from image pool and save image objects to it and check pool size
Draw image object to users' screen
Image pool
Thread Draw image object

Figure 4.7. Image Streaming Transmission Module

5. A Test Result

Constrained by the available hardware resources (network bandwidth, machines' performance), the GSN can not be tested on a real various simulating users' working environment. The alternative approach to do this test is to count the transferred data size while doing different operations on GSN Image Viewer. The two criteria items used to evaluate are image amount and image quality of data transmitted from server to client. Experiments showed that the response time is 0 for any operation due to a high quality overview image and image pool was always being able to show user some thing ignoring the show's quality. The image data transferred were very depending on the viewer's windows size. Table 5.1 shows the relationship between windows size and transferred data amount based on the three different operations.

Table 5.1. Windows Size .vs. Data volume

Windows size	**Zoom to a neighboring pyramid level (KB)**	**Zoom to a pyramid level beyond neighboring level (KB)**	**pans across screen (KB)**
640X480	337.84	422.30	370.50
512X380	226.79	283.49	257.97
400X300	150.99	188.73	179.31
320X240	105.46	131.83	130.81
250X190	73.19	91.48	95.38
200X150	52.88	66.11	72.35

The data volume in Table5.1 is the data which will be transferred to client side to show the highest quality of images according to users' requests. The data volume transferred to client was reduced while the windows size became small. It shows that the windows size should be smaller on low network bandwidth than that on high network bandwidth. The data volume arranges from 52.88KB to 422KB on different windows size and different operations. If the mean connection speed of Internet users is 1Mbit/s, 125KB/s, the sharpening (get all highest resolution images in that request) time will be from 0.4 to 3.4 seconds. Due to the data is been streamed to the users, even for the modem users, they will feel they are getting serves any time after clicking on the viewer.

6. Conclusions

Based on this improved structure, GSN client becomes much more efficient and smart than the old one. Firstly, the response time is equal to a desk top system. User will not feel any delay while operating map. Secondly, the network will not be the decisive factor to the end users. Thirdly, many fancy functions are able to be integrated to GSN client. They are Fly In/Out, 2D Image Fly (any direction) of 2D Images. The last feature will be the most important core component in this project. The module can be easily migrated to a true 3D model if the digital terrain data is available. Finally, it becomes more reliable and extensible. Further works should be to use this technique on 3D Presentation and the storage technologies to handle huge imagery.

By applying streaming and pyramid technologies, the system's performance has been improved efficiently and looks like a desktop GIS application. Additionally, enable to support GSN Globe has promoted it to be a huge image data set server. GSN Image System technology offers these key features and benefits:

- **Easy Implementation**
  GSN Image requires no special server-side software or complicated authoring tools, allowing for quick and easy content creation and implementation.
- **Bandwidth Friendly**
  GSN Image only downloads the files being used and streams required files on demand allowing for high-resolution images over all bandwidth environments.
- **High Resolution Images**
  GSN Image allows web designers the ability to post high-resolution JPEG and vector graphics files that users can view, regardless of connection speed.
- **'unlimited' amount of images support**
  If hardware is available, unlimited images can be put to its storage.

7. References

[1] Harder, C. (1998). *Serving maps on the Internet*. Redlands, CA: ESRI Press.

[2] A primer on Internet-based mapping technologies, *for novices*. Thomas Baker, Ph.D. PathFinder Science

[3] Federal Communications Commission, 2001. "FEDERAL COMMUNICATIONS COMMISSION RELEASES DATA ON HIGH-SPEED SERVICES FOR INTERNET ACCESS", US http://ftp.fcc.gov/Bureaus/Common_Carrier/News_Releases/2001/nrcc0133.html

[4] ERMapper White Paper, 2003. "Using ECW Connector™ and Image Web Server™ with ArcIMS®", US. http://www.ermapper.com/document/doc.aspx?doc_id=34

[5] William B. Pennebaker and Joan L. Mitchell, 1992. *JPEG: Still Image Data Compression Standard*, Van Nostrand Reinhold, New York.

[6] J.M. Ogden, E.H. Adelson, J R. Bergen, P.J. Burt. 1985. Pyramid-based computer graphics, RCA Engineer 30-5

[7] Computer Graphics Group, *Remote 3D Visualization using Image-Streaming Techniques*, Universit¨at Erlangen-N¨urnberg,Germany http://www.vis.uni-stuttgart.de/ger/research/pub/pub1999/ISIMADE99.pdf

8. Acknowledgements

We would like to thank the Geospatial Information and Communication Technology (GeoICT) Lab, Department of Earth and Space Science and Engineering, York University for their support and for providing the GSN source code and some high resolution images to support the image-streaming algorithm.