The Effect of Slicing JPEG File on the Web Page Download Time

Fakhrul Hazman Yusoff
Anita Mohd Yasin
Rozianawaty Osman

Image compression plays a vital part in displaying image on the Internet. Image compression format such as JPEG makes the file smaller hence speed up the download time. This paper will evaluate the usage of slicing technique on JPEG-based image file. Experiments are conducted in establishing whether slicing technique do contribute to reduction in total file size and speed up the download of an image file. Based on the finding, it is concluded that image slicing can reduce image size in certain situation. On the other hand image slicing will not significantly reduce the download time. However, as pointed out by the research image slicing do play some important role in overall image display mechanism in a website to compensate its slightly slower download time.

Introduction

Compression can be divided into two broad category namely lossless and lossy compression (Chapman, 2000). In lossless compression, the original data will be preserved throughout the process. In other word, the data will be compressed and decompressed to its original state without losing any information. Among known techniques that can be used under this type of compression are Delta Compression, Run Length Encoding, Statistical Encoding and Huffman Encoding. Lossless compression is suitable for highly sensitive information which cannot afford to lose even a single data. Lossless compression is appropriate for text-based data (Sayood, 2000). The reason is losing a character for example is enough to change the meaning such as from 'crowd' to 'crow'. Lossless compression however is not confined only to text-based data, any data type which is highly sensitive to changes may opt for lossless compression. The data types include medical imaging, military surveillance and astronomical imaging where dropping an information might causes problems to the related parties.

Lossy compression on the other hand will drop some information from the original data to ensure that the compressed version is small (Sayood, 2000). It operates on the basis that dropping some information may not bring harm to the overall presentation of the data. Ordinary images for example will be using lossy compression. This is because losing data for some pixels will go unnoticed to human eye. JPEG file format can be considered to be under this category since it will discard some information to ensure that the compressed data will be small (Hughes, 1998). JPEG can actually be configured to produce lossless compressed data. However it is mostly used as a lossy compression tool (Hankerson, 1998).

JPEG file format has been extensively used on the Internet. One of the reasons that may attribute to the popularity of JPEG in the Internet is its ability to compress big image file and decompress it without much apparent distortion to the users' perception. Although lots of information is needed to capture an image, JPEG can shrink the file significantly without introducing much distortion.

However, with limited bandwidth, transferring large image over the Internet will require a lot of waiting time in user part. Many users still use dial-up as their medium for Internet connection. While the modem state that it has the speed of 56kbps, the actual throughput usually is less than that ranging from 22 kbps to 48kbps to as little as 2 kbps to 3 kbps during high volume traffic.

One of the approaches being suggested in increasing the download time of an image file is by slicing the image file ((Haynes, 2000). It has been argued whether slicing the image into small components can actually contributes to increase in the transfer rate. There are many proponent and opponent of slicing the image. This research will try to slice a given set of images and investigate whether it has an impact on the file size and contributes towards increment in downloading time.

Literature Review

According to Nielson (2000), Design Web Usability : The Practice of Simplicity, web page download times depends on page design, on web server and client hardware and software configurations, and on the performance characteristics of the Internet route connecting a client to the site. Of these, only page design is truly under the site designer's control.

Jing (2002), Web Page Design and download Time. This paper analyzes measurement data based on test pages to explore various relationships between web page design and page download time. Two different linear models were built to understand the bulk of page download for simple test pages. Another model was built to investigate how multi-threaded improves performance for more complex pages. This model was built and discussed for pages with one to 64 embedded images. Implication that is useful for site designers include the goal of keeping all available threads "busy" for the same amount of the same amount of time. For example, the site might be improved by separating a page's single largest image into parallelizable pieces. Jing also found that large number of images causes undesirable overhead, and image sizes. A page with four images has the smallest mean downloading time which was the optimum performance.

According to Georgia Tech University's 10th GVU World Wide Web User Survey, which is conducted from October 10, 1998 through December 15, 1998 and includes data from over 5,000 Web surfers, 66% of users are surfing the Web using a 56 kbps modem or slower. A T1 line takes fully 55 seconds to load over a 28.8 k modem (ignoring the impact of server connections), and over a third of visitors are browsing at about that speed.

Wu (1999), Speed Up Your Web Pages. This paper is trying to give some practical solutions and tips to decrease page size to speed up web page thus to provide users better surfing experience within the sites. This is done by optimizing the graphics and tables, and aware of special effects on multimedia. Wu conclude that the main factors affecting page size significantly such like graphics, tables, frames, and multimedia files should be chosen carefully and be put into the right place to ensure a user-friendly Web site.

Hyun (2003), Optimizing Web Pages. Optimizing web pages using markup coding and graphic design techniques to make Web page download faster and increase acceptability. Optimize markup coding by using HTML optimization, head minimization and DHTML optimization while optimizing graphical design techniques via file size and quality (GIFs, JPEGs, and PNGs), use CSS and reuse graphics with same URLs, use RGB color format. Optimizing Web pages using several techniques such as HTML coding and image file saving methods may speed up the web page load time and pursue the maximum effectiveness of graphical images at the same time.

According to Smart Computing (April 2001, p. 85), image slicing is pretty simple in concept and in practice. Instead of placing one giant image on your Web page, an image-editing program, such as Adobe Photoshop, Paint Shop Pro, or even Microsoft Paint, can be used to cut the image into several smaller pieces. Then use simple HTML table coding to rebuild the image on Web page. Slicing the image won't necessarily make it load any faster, but when bits and pieces of the image load and display quickly, it appears to the visitor as though the page is loading quickly. In fact, it may actually increase the load time slightly, but the perception by the user is that it is faster.

According to Philip (2003), Image Slicing: Building Complex Imagery for Web Sites, said that many web designers develop page layouts in Photoshop (or Illustrator) before use an application such as Dreamweaver to assemble the page in HTML. This approach allows the designer to use all of the powerful layout and design capabilities of Photoshop to work on the creative aspects of laying out the page. Once a design is solidified, the designer translate the Photoshop layout into the realities of HTML and limited download time. This involves compressing different parts of the layout in different ways (e.g. some parts will be GIF while others will be JPEG), and leaving some parts of the layout for live HTML. To do this, it's usually necessary to cut the original Photoshop image into smaller pieces – image slicing. ImageReady is a program specifically designed for image slicing, and contains many tools for this process. Macromedia Fireworks is another program designed for image slicing.

Methodology

In order to investigate the impact of slicing an image on the download time, we have devised the following steps.

A set of 100 images of size bigger than 500 kb each will be used for this experiment. The images will be duplicated and resized into three categories namely 100 kb, 350 kb, and 500 kb. The purpose of selecting 100 kb and 350 kb is because these sizes are the range of size that is commonly used to display image in a website. Since 500 kb is considered to be a big size, it is included to see the impact of big file size on slicing process in relation to the download time.

The images on these different categories will be sliced into different slices using Adobe Photoshop. The image will be sliced using the following formation:

1. 4 slices (2 x 2)
2. 16 slices (4 x 4)
3. 64 slices (8 x 8)
4. 100 slices (10 x 10)

The slicing process slices the image into grid matrix-like (n x n) division. This makes the slices to take a near-square form. The division will be referred to as slicing dimension throughout this paper. Adobe Photoshop is being used because it has slicing features incorporated inside one of its features. In addition Adobe Photoshop offers an automated creation of HTML page that will display the sliced image. A Javascript-based timer is incorporated in the HTML file to be used as time measurement during the experiment later on.

The purpose of having the categories (100 kb, 200 kb, 350 kb, 400 kb and 500 kb) and different slice formation is to establish a pattern between slicing image within the same file size and among different file size.

The sliced images will be uploaded into a web server running Internet Information Server version 5.0. The server is installed on an Intel-based personal computer with the following configuration:

1. Microprocessor - Pentium IV (Single rocessor)
2. Memory - 400 MB SDRAM
3. Hard Disk - 30 GB

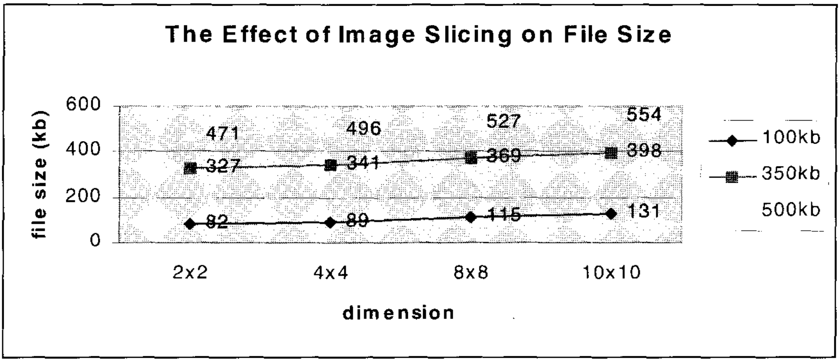
Experiment will be conducted by investigating the file size pre and post slicing process. This investigation will enable us to find out if the total file size will grow bigger or smaller relative to the preset size category as we slice our images in different formation.

Another experiment will be conducted investigating the download time of each of the HTML page that displays the sliced image by using <table> tag. The data obtained will be in unit of milliseconds. This experiment will be conducted locally on the web server to eliminate other contributing factor that may affect the download time. The experiment will be repeated for 10 times and the average download time will be taken. The purpose of this activity is to reduce any other contributing factor that may affect the downloading time.

Result and Discussion

Slicing Impact on the File Size

Slicing initially contribute to some reduction in aggregate file size of an image. This can be best sum up by the following Graphs:

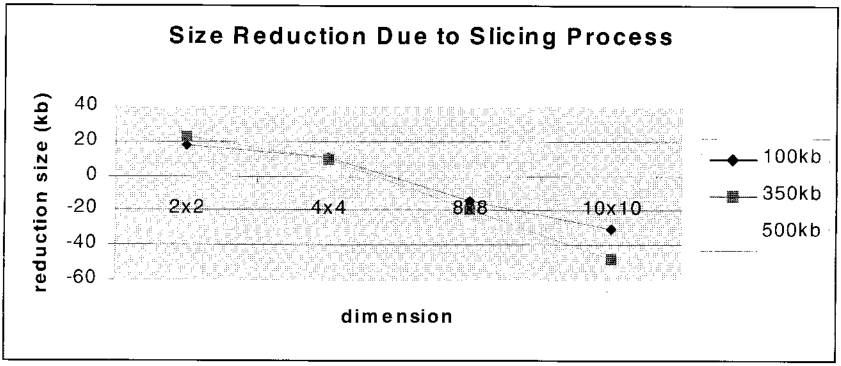


Graph 1

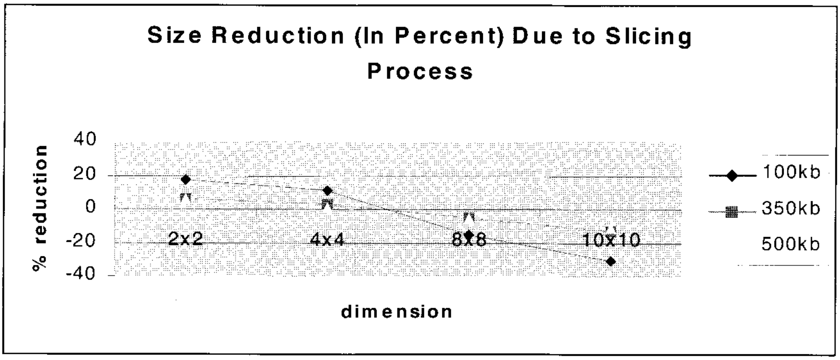
Based on **Graph 1**, the file size seems to be smaller than non-sliced version in 2 x 2 and 4 x 4 slicing process. The result appears to be constant in all categories (100 kb, 350 kb and 500 kb). This can be attributed to the different size reduction for each individual slice. As some of the slices will have color information that is highly correlated, the compression rate will be significantly higher for these particular slices hence contributed to aggregate size reduction.

Graph 2 depict amount of size reduction gain through image slicing. As illustrated, slicing an image will lead to gain in term of size reduction for any slicing dimension of 4 x 4 and less. As the slicing dimension gets bigger than 4 x 4, the aggregate file size increases defeating the purpose of reducing the file size.

Graph 3 depicts the percentage of reduction for each slicing category (2 x 2, 4 x 4, 8 x 8 and 10 x 10). File with size of 100 kb stand to gain almost 20% of sizing



Graph 2



Graph 3

reduction in 2 x 2 category and lesser in 4 x 4 category. Likewise it size will bloat the most when sliced at dimension of 10 x 10. The images with file size of 350 kb and 500 kb saw a modest gain in size reduction (Less than 10%). Interestingly the percentage of increment of file size at 8 x 8 and 10 x 10 seems not to be so much if compared to images with file size of 100 kb.

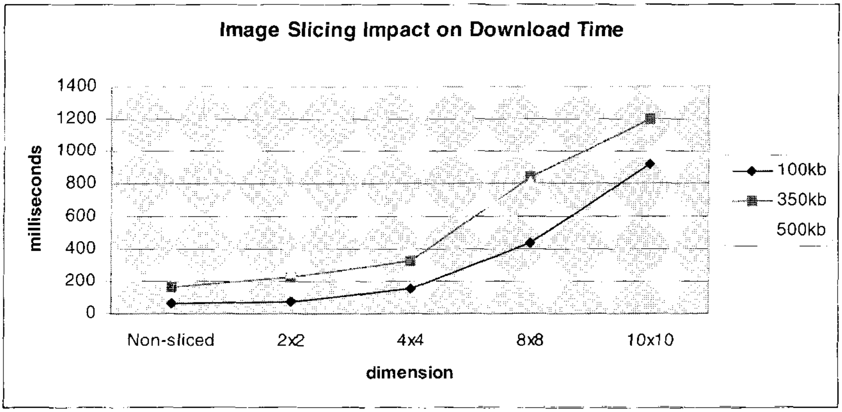
Based on the given facts, Image slicing can contribute to smaller file size if the slicing is smaller or equal to 4 x 4. Smaller file size may contribute to faster download time. However the file size increases dramatically from that point (4 x 4). The notion that each individual slice compresses differently and some may lead to reduction in size does not apply as the amount of slices grows. The file size will grow as the slicing dimension increases. Interestingly however the growth in percentage wise seems to be biggest with the smallest file category of the experiment (100 kb). The other categories (350 kb and 500 kb) saw a modest size increase in term of percentage when the image slicing dimension increases.

One of the explanations for the size growth is the operating system will include icon information for each of the file produced (Haynes, 2002). Since each slice results in a new file, icon information will be incorporated in this file hence increases the file image. The addition of icon information seems to be a significant to a small file size that lead to higher percentage of increment in image size of 100 kb. On the other hand since the icon information data size will be constant and considerably small it does not give big impact to larger file such as 350 kb and 500 kb size categories hence smaller percentage of file growth.

Another explanation for increase in file size is due to the increase of markup tag information inside the HTML page. As slices are bigger, more markup tags are needed to build a complete table that displays the image. As such the file size of the html file will be bigger.

Slicing Impact on the Download Time

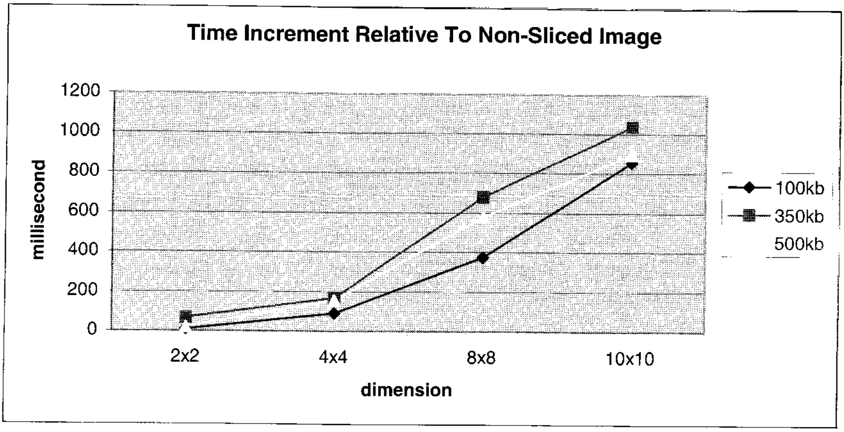
Although in certain cases after-sliced file sizes are technically smaller than non-sliced version, which more dramatic increases in download time starts to appear in dimension of 4 x 4.



Graph 4

The download time data indicates an upward trends as depicted in Graph 4. **Graph 4** indicates that time taken to download a sliced image of 2 x 2 is about the same as downloading the non-sliced version at least for the 100kb and 500 kb file categories and take slightly more time for 350 kb category.

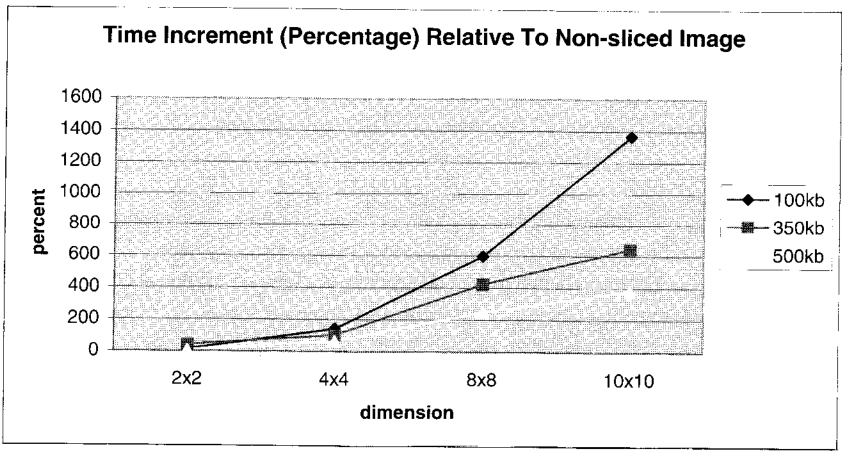
Slicing an image into dimension of 8 x 8 and 10 x 10 will definitely lead to a very long waiting time as the graph suggest. The 350 kb and 500 kb files are downloaded at slower rate than 100 kb. However it seems that download time for both 350 kb and 500 kb does not show much different. Surprisingly files actually download faster for 500 kb image file compared to 350 kb at slice dimension of 10 x 10.



Graph 5

Graph 5 illustrates the additional time taken to download for sliced image relative to non-sliced version. Like Graph 4, the graph has an upward trend showing the slice dimension of 2 x 2 register the smallest increment in download time.

Interestingly file size in the 350 kb seems to register bigger increase in download time compared to 500 kb. The image with file size of 500 kb requires additional time that is comparable with image with file size of 100 kb. All and all, the 100 kb image file category performs poorly in 10 x 10 sector. Although it clocks increment in download time that is lower than 500 kb and 350 kb, the different is very small if compared with 500 kb category.



Graph 6

In analyzing further, **Graph 6** shows to us the percentage of increment relative to the non-sliced version of the image. It is very obvious in this graph that image file with size of 100 kb performs very badly if we slice it in large amount. While the percentage increment in download time seems to be small in slice dimension of 2 x 2 and 4 x 4, the same thing does not hold for slice dimension of 8 x 8 and 10 x 10 especially for 100 kb file category which clocked additional 1400% than the time taken to download the original non-sliced version of the same image.

Based on the finding it is safe to say that image slicing actually increases the download time of an image. However the increment seems to be very minimal as long as the slice dimension is smaller or equal to 4 x 4. Slicing an image contribute to a very long download time when the slice dimension is bigger than 4 x 4. This is especially true if the file size is 100 kb. While slicing a large image will lead to longer download time, the variation is small at the upper level. As indicated in the case of 350 kb and 500 kb file, if we want to slice a large image, it does not make much different (download time-wise) if we chose 500kb image file or 350 kb image file. As such, given a choice of 500 kb and 350 kb, it may be better to slice the 500 kb version when we want to slice it in large dimension.

Conclusion

Slicing the image actually can lead to smaller total file size. In our experiment, it has been proven that slicing can contribute to smaller file size. However image slicing work best at most by slicing it into slice dimension of 4 x 4 (16 slices). Any slicing process that is bigger than 4 x 4 contributes to bigger file size. As such it is advisable for developers to slice an image not more than slice dimension of 4 x 4. Any slice bigger than that will increase the file size.

Reduction of file size during the slicing process does not guarantee reduction in downloading time. Generally all slicing dimension demonstrate an increase in download time beginning from slice dimension of 2 x 2, 4 x 4, 8 x 8 to 10 x 10. The increment in downloading time takes a logarithmic pattern in which tremendous increase in download time is obvious given high slice dimension such as 8 x 8 and 10 x 10. On the other hand increment in download time is very negligible for slice dimension of 2 x 2 and 4 x 4. As such, if the designer want some flexibility in its image presentation and require fast download time, slicing the image at slice dimension of 2 x 2 and 4 x 4 may do the trick. However if the developer are dealing with larger file image such as 350 kb and 500 kb, the impact of slicing a 500 kb image will not be very different from 350 kb. As such, if the developer insists on quality of image and want to slice the image, he may want to slice 500 kb image compare to 350 kb version because the download time for the sliced 500 kb file will not be far off from the 350 kb version.

References

Chapman, Nigel (2000). *Digital Multimedia*, Wiley.pp 35-66.

Georgia Tech University (1998), Georgia Tech University's 10th GVU World Wide Web User Survey, Georgia Tech Research Corporation. Retrieved October

Forouzan, Behrouz A. (2003). *Data Communication and Networking*, McGraw Hill

Haynes, Barry (2002). *Photoshop 7 Artistry – Mastering the Digital Image*, New Riders Publishing. Pp 42-43.

Hankerson, Darrel (1998). *Introduction to Information Theory and Data Compression*, CRC Publication.

Hughes, Kevin. (2004). *What is the difference between JPEG, GIF and PNG?*, Retrieved October 08, 2004.

Jing Zhi (2001). *Web Page Design and Download Time, Journal of Computer Resource Management*, Issue 102, Computer Management Group Incorporated.

Lee, Hyun Joo (2003). Optimizing Web Pages.

Nielson, Jakob (2000). *Design Web Usability: The Practice of Simplicity*, New Riders Publishing. Pp 5 –10.

Sayood, Khalid (2000). *Introduction to Data Compression*, Morgan Kaufmann.p4

Smart Computing (2002). *Precision Cutting: How to Make a Sliced Image*, Volume Issue **1**, Pp 72 – 74.

Van Allen, Philip (2003), Image Slicing: Building Complex Imagery for Web Sites, Speed Image Slicing, Retrieved October 01, 2004.

Wu, Jie (1999). Speed Up Your Web Pages, Retrieved October 01 2004.

FAKHRUL HAZMAN YUSOFF, Department of Computer Science, UiTM Perlis. E-mail: fakhrul@perlis.uitm.edu.my

ANITA MOHD YASIN, Faculty of Information Technology and Quantitative Sciene, UiTM. E-mail: my_aniss@yahoo.com

ROZIANAWATY OSMAN, Department of Computer Science, UiTM Perlis. E-mail: rozianawaty@perlis.uitm.edu.my