Image Thumbnailing

Introduction

As the digital imagery market continues to grow both in size and diversity of industries the software for browsing and searching this imagery has also grown and matured. Virtually all of this software uses image “thumbnails”, small or very small reproductions of the original image, to show users what the larger, more cumbersome originals look like. However, while the software for managing image libraries has matured, the basic thumbnailing techniques used have not. The processes developed decades ago, and tailored to the limitations of that era’s hardware, are still the dominant techniques used today. Oddly enough, this stagnating of technology is not for lack of more sophisticated solutions – academia has provided us with numerous thesis and papers on the subject – rather, the relatively small incremental benefit they offer fails to justify the associated complexity and performance losses incurred.

This paper introduces the two common techniques and presents a minor variation of the averaging algorithm that enhances [perceived] thumbnail quality in a simple and efficient manner.

Subsampling & Averaging

The evolution of thumbnailing algorithms can be tied directly to performance and simplicity. As mentioned above, two algorithms are in common use today and have been around for at least 30 years. To introduce these algorithms, the following example images will be used. Figure 1 is representative of digitized images typically produced by digital cameras and scanners. Figure 2 is a test image that features the elements commonly found in diagrams, figures, or other work created using a graphics application.

Figure 1

Figure 2

The first technique is subsampling. It is the simplest and fastest method and, hence, the most primitive. Subsampling is simply the reduction of an image by taking only a subset of the existing information. In our examples, we reduce the images to 1/4th their original size. As a result, only one pixel in sixteen is represented from the original images. Figures 3 and 4 show the effects of this (note that these figures are shown at 4x their actual size to allow for examination of image detail).

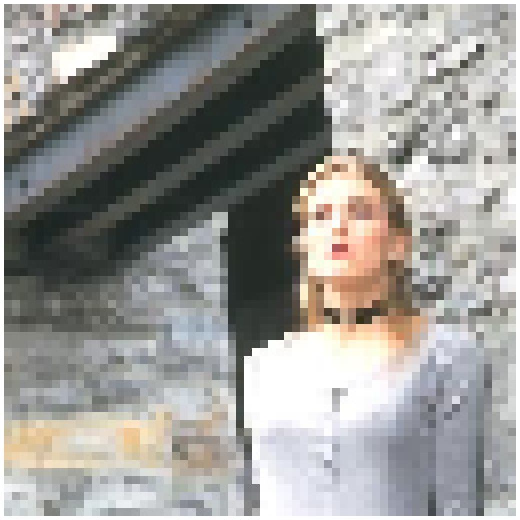


Figure 3

Figure 4

As can be readily seen this process results in jagged edges on slanted lines and a butchering of detailed information that may be useful in recognizing the content. (E.g. the woman's expression changes dramatically because the sides of the mouth disappear). Because these artifacts are so noticeable, this technique is typically found in only the most basic applications.

The second thumbnailing technique is averaging. This is more refined in that it guarantees that each pixel in the original image is represented. Where subsampling uses only a single pixel to represent a block of pixels, averaging, as the name implies, uses the average of all pixels in a block, shown here in Figures 5 and 6.

While this technique eliminates the problems associated with subsampling, it suffers from a softening effect that makes the result appear out of focus. This is particularly noticeable in regions of the image where a sharp, well-defined edge is expected regardless of the resolution.

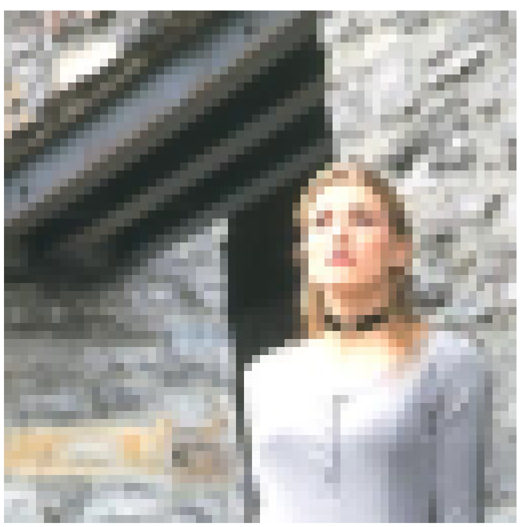


Figure 5

Figure 6

For the most part, averaging produces satisfactory results. The blurring effect is constrained to within one to two pixels of the region in question and is usually not significant enough to warrant comment.

Averaging + Sharpening

Even though the averaging technique is sufficient for most purposes, it can be improved by adding a single, simple step – a sharpening filter. Furthermore, with a little judicious programming, the performance impact of applying a sharpening filter can be made negligible (See appendix A).

As shown in figures 7 and 8, sharpening emphasizes areas of high contrast.

The sharpen filter used above is set at 100% - it doubles the difference between a pixel and the average of it's neighbors. While this solves the blurring problem, upon closer inspection it can be seen that this, too, has undesired artifacts. The emphasis in contrast actually becomes too pronounced causing noticeable shadows and highlights that don't exist in the original. If this is tempered slightly and only 50% sharpening applied, an esthetic balance is achieved, as shown in figures 9 and 10.

Here we see that the shadows and highlights introduced by the sharpening serve to emphasize distinguishing features rather than becoming features in and of themselves. In the test pattern image, the text is emphasized, the lines are slightly darker, and the hexagons reveal themselves as hexagons. In the image of the woman the texture of the stone wall emerges, her head and arm emerge from the wall behind her, and the facial expression is heightened.

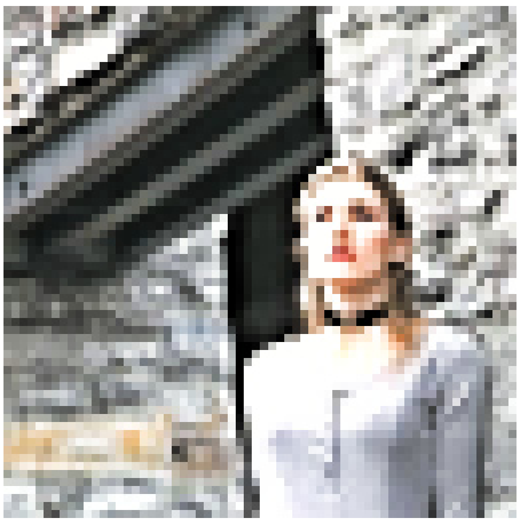


Figure 7

Figure 8

Author's Note: The choice of 50% as the amount of sharpening to apply is purely empirical. It is based on a substantial amount of experience with this process. A more rigorous derivation for the theoretically optimal value could be done, but is out of the scope of this document

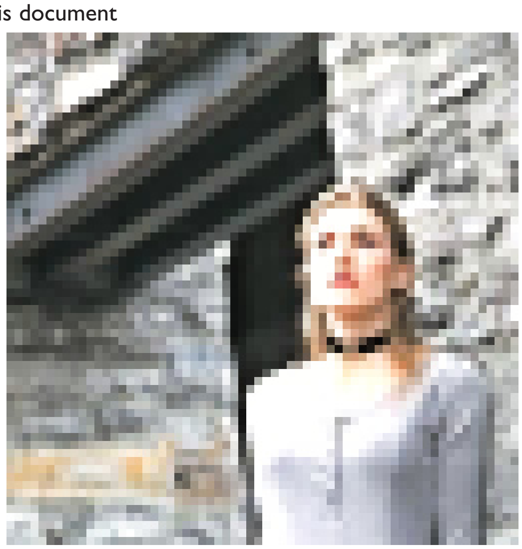


Figure 9

Figure 10

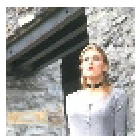
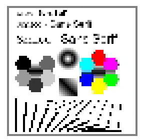
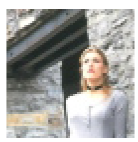
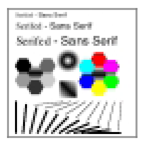
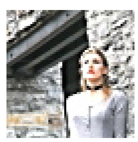
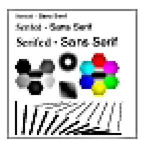
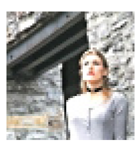
Summary

The following table shows each of the four variations discussed above at actual size. It is left to the reader to draw his or her own conclusions.

Appendix A: Implementation Notes

The implementation of a sharpening filter offers several, perhaps not so obvious, optimizations.

When doing the initial resizing, it is possible to use a hybrid approach that combines the speed of subsampling with the quality of averaging. If the averaging is done on a subsample of the original image that is some reasonable factor larger than the desired thumbnail (e.g. 2x in both dimensions) a reasonably high quality result is obtained. The advantage of this it establishes the amount of data that needs to be processed to a known quantity. If the size of images being thumbnailed is unknown, or large scale factors are expected, this can result in substantial performance gains. (Note however that this can still result in some of the more esoteric artifacts associated with subsampling, such as those that occur with dithered colors)

Subsampling		
Averaging		
Averaging & 100% sharpening		
Averaging & 50% sharpening		

As part of performing a sharpen operation, a neighborhood average is computed, something that is already being done during the resizing process. If the sharpening and resizing processes are tied together, this neighborhood average need only be computed once.

If need be, the entire process can be implemented using integer arithmetic. Both the sharpen filter and the average filter involve very straightforward arithmetic operations (addition, multiplication and division).