

# High-Performance Image Content-Based Search

Rahman Tashakkori  
Department of Computer Science  
Appalachian State University  
Boone, NC 28608

Steven H. Heffner  
Wake Forest School of Medicine  
Winston-Salem, NC 27157

Darren W. Greene  
Department of Computer Science  
Appalachian State University  
Boone, NC 28608

Barry L. Kurtz  
Department of Computer Science  
Appalachian State University  
Boone, NC 28608

**Abstract** – *The existing content-based techniques used to search for a small sample image within a larger one, become progressively slower as the size and resolution of the image increases. This paper focuses on increasing the efficiency of the traditional content-based search techniques employed within the spatial domain. Four different methods were explored for improving the speed and accuracy of traditional Standard Convolution. We used an enhanced implementation of the Standard Convolution which we refer to as the Iterative Convolution along with the Mean-Square Measures of Variance, Pearson’s Correlation, and Haar Wavelet Lifting techniques. In addition to the enhanced algorithm implementations, we have used the MATLAB Distributed Computing Toolbox to improve the performance of the Standard Convolution method that was used as a baseline. The completion-time and accuracy measurements of applying each of these techniques on a set of mammograms were studied and will be presented in this paper. It was found that the Pearson’s Correlation method yielded the best results in the content-based search experiments we have conducted. Also, we were able to show that the MATLAB Distributed Toolbox running on an eight-node Linux cluster was able to reduce the completion time to as low as 1/8 of the original time.*

**Keywords:** *Distributed image processing, high-performance convolution, and Distributed MATLAB Toolbox*

## 1.0 Introduction

In the past few decades both the CT and MRI medical image modalities have been significantly improved and have become vital tools in diagnosis. The highly detailed images produced by these modalities allow physicians to detect abnormalities that previously would have gone unnoticed. The computer-based analysis of these images has been used with some success as a diagnostic aid. Though there are many existing techniques that enable computers to search for a small sample image within a larger one, they all lack sufficient efficiency as they become unacceptably slower as the size and resolution of images increases. In this research we have focused on increasing the efficiency of content-based searches employed within the spatial domain. Four methods were explored to enhance the performance and accuracy of Standard Convolution. These enhanced methods include the Iterative Convolution, Mean-Square Measures of Variance, Pearson’s Correlation, and Haar Wavelet Lifting. The results, consisting of completion-time and accuracy, of applying each of these techniques on a set of large digital medical images (mammograms) are reported. In addition we report the results of using Distributed MATLAB to speed up the computations.

## 2.0 Spatial Domain Filtering

The use of spatial masks for image processing is often called *spatial domain filtering* [Gonzalez 93]. Spatial filters are procedures that operate directly on the pixels of the image. These operations can be expressed as:

$$g(x,y) = T[f(x,y)]$$

where  $f(x,y)$  is the input image,  $g(x,y)$  is the processed image, and  $T$  is an operator that operates on  $f$ . This operator is defined over some neighborhood of  $(x, y)$ . The main task in spatial domain filtering is to define a square or

rectangular neighborhood about  $(x, y)$  as a sub-image that will be the target area and is of the same size as the operating mask. The pixel at the center of the target area,  $(x,y)$ , is replaced with a new value,  $r$ , after the process is completed. Then the mask is moved to the next pixel and the process will be repeated using the original data. As shown in Figure 1, the center of the mask is moved pixel-to-pixel starting at a corner and applying the operator at each location  $(x,y)$  row-by-row or column-by-column to create a  $g(x,y)$  for each location. This will continue until all the pixels in the image are processed. The size of the mask may vary depending on what we plan to accomplish; however the number of rows and columns in a mask are almost always odd numbers.

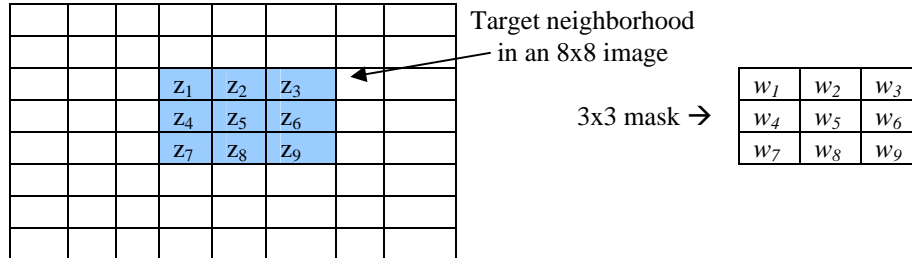


Figure 1 – Operation of a 3x3 mask in the neighborhood of pixel  $(x,y)$

Depending on the gray level of pixels under the mask at any location  $(z_1, z_2, \dots, z_9)$  the operation of a linear mask will produce a single value that replaces the center pixel of the target neighborhood. A linear mask operates on the target neighborhood in a manner shown in (1).

$$r = w_1 * z_1 + w_2 * z_2 + w_3 * z_3 + \dots + w_9 * z_9 = \sum_{i=1}^9 w_i z_i \quad (1)$$

Non-linear spatial filters operate in a similar manner on the neighborhood pixels but their operations are based directly on the values of the pixels in the neighborhood under consideration. An example of such a non-linear filter is a median gray-level value in which the median of the pixel values in the target neighborhood is selected to represent the center value of the region of interest.

## 2.1 Standard Convolution

Standard Convolution is a well-known digital image processing technique that is used for a variety of purposes; its main application is in content-based search. For the purpose of searching for a sample piece within an image, the filter is essentially a reverse reading (sample image rotated by 180 degrees around the x-axis then 180 around the y-axis) of the original sample. As in all filtering, the image must be padded so that when the center of the filter is placed over top of any element at the edge of the image, there is no element in the filter that does not have a corresponding element underneath it. After all the formatting of the filter and the image has been completed, the actual process of convolution can proceed as follows:

- Step 0: Create a matrix of the same size as the original image.
- Step 1: Pad the image with as many number of rows or columns needed to suit the filter.
- Step 2: Place the center of the filter over the first element of the original image.
- Step 3: Multiply corresponding elements on the filter with the neighborhood pixels on the image.
- Step 4: Sum the products together to get  $r$  as shown in (1).
- Step 5: Store the result in its proper location in the new matrix.
- Step 6: Iterate through the entire image repeating Steps 2 – 5 until all pixels are covered.

The result will be a new matrix, the same size as the original image, holding the results from the steps above. The process explained above is referred to as the “Same Convolution” since the size of the resulting matrix is the same size as the original image [Gonzalez 04].

When using Standard Convolution to search for a sample within a larger image, the filter is a reversed reading of the sample. The desired outcome of performing Standard Convolution is that the result of the calculations will be greatest at the location where the pixels on the filter and those on the neighborhood sample are very similar. Then by thresholding the top values we obtain the location of the sample piece in the image.

## 2.2 Iterative Convolution

Iterative Convolution is an enhanced version of the Standard Convolution introduced in this research to improve content-based search performance. The concept behind Iterative Convolution is to first make a coarse pass through the image to recognize likely regions of the target image where the sample piece might be found. In the next pass, the Standard Convolution is performed only at those smaller regions. To perform Iterative Convolution, both the target image and the sample should go through the same pre-processing as they would go through with the Standard Convolution. In the first pass, the Standard Convolution might be performed horizontally and vertically on every  $n$  number of pixels of the target image. This reduces the number of calculations by  $n^2$ . When the result is calculated, it is compared to a predetermined threshold value. This threshold value can be determined empirically to obtain the best result. If the calculated value is greater than the threshold value, then the coordinates of the location of the filter over the target image are stored in a matrix. During the second pass, Standard Convolution is performed only on the pieces that are stored in the first pass.

## 2.3 Mean-Square Measures of Variance

Both Standard Convolution and Iterative Convolution use the procedure shown in (1) which computes the sum of the products of corresponding elements. It is expected that the largest value be obtained when the values in the sample piece matches the corresponding values on the neighborhood pixels on the image. While this may frequently be the case, it is not always true. For example, when a sample is taken from a dark portion of a bright image, the sum of products will be greater at the brighter locations. When the results are thresholded, the location of the sample will not be clearly identified, if it is identified at all. This shortcoming of the convolution-based methods required that a new search algorithm be developed. To solve this problem, we took an approach that was based on the variance. In this method, the variance between the sample piece and the neighborhood piece on the target image is computed. Unlike the convolution-based methods, however, the sum of products between the elements in the filter and the matrix is not calculated. Instead, the product of the corresponding elements in the filter and the matrix are calculated, and then from them, the mean of the elements in the sample is subtracted. These values are then squared and summed. The results from these calculations are then stored in the results matrix. The following equation defines the process:

$$\sum (x_i - \mu_{sample})^2$$

The results from this filtering can then be thresholded against some acceptable level of variance. This should improve the effectiveness of convolution-based methods. In order to increase the efficiency (speed) of this method, iteration is employed as before.

## 2.4 Pearson's Correlation

Often the sample image is scaled, thus it may no longer be in the same intensity as it had originally possessed. The result of scaling is a sample that is lighter or darker than it was supposed to be. This scaling of the sample has an adverse effect on the results of the convolution and Mean-Square-Measures-of-Variance-based methods. To overcome these shortcomings, a search method based on Pearson's Correlation was developed. In this method, the data with linear-positive-slope and linear-negative-slope produce +1 and -1 for the Pearson's Correlation respectively. Any relationship between the data points which deviate from linearity will have a Pearson's Correlation with an absolute value less than 1; the less linear the data, the closer the Pearson's Correlation is to zero. The Pearson's Correlation for a set of data is given by the following equation:

$$r = \frac{\sum (x_i - \bar{x}) \cdot (y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

Since a scaled sample and its corresponding portion in an image should differ by only a constant factor (i.e. the scaling factor), the relationship between the two sets of numbers can be defined as linear:  $sample = scale * imageSegment$ . This means that performing a search based upon Pearson's Correlation should be able to locate the sample in the image, regardless of the scaling.

## 2.5 Haar Wavelet Lifting

The final method used in an attempt to improve both the efficiency and effectiveness of Standard Convolution was an approach based on Haar Wavelet Lifting. Wavelets are a large class of functions that can extract spatial and frequency characteristics from an image by breaking the image into a trend and three details. One of the artifacts

produced by the Haar Wavelet function is the trend, which is an image one-fourth the size of the original that contains the overall trends from the original. When performed on small images, the results can be blurry; however, for large images, the results are often visually indistinguishable from the original. This ability to reduce the size of an image by one-fourth after each level of wavelet decomposition and still keep its features makes the Haar Wavelet function an ideal method of reducing the time required to search a digital image. Haar wavelet is done in two steps known as *predict* and *update* as shown below.

$$\text{Predict: } d = s_{\text{odd}} - s_{\text{even}} \qquad \text{Update: } s = s_{\text{odd}} + \left\lfloor \frac{s_{\text{even}}}{2} \right\rfloor$$

Through a permutation, the odd columns are then moved to the beginning of the matrix and the even columns to the end. These three steps are then repeated on the rows of the matrix [Daubechies 96].

### 3.0 Computational Procedure

We have used a set of mammograms as our target images. We have cut some pieces of different sizes from the images in the set to create our sample pieces and scaled all pieces. We scaled these sample pieces and padded the target image to the appropriate size, in our case one row and one column was added to all sides. We used the symmetric boundary [Tashakkori 01] to extend the boundary of the target image by one row. Thus, we have repeated the values on the first row, first column, last row, and last column once to create the pad at the top, bottom, left-, and right-hand-side respectively. A summary of the computation process is shown in Figure (2).

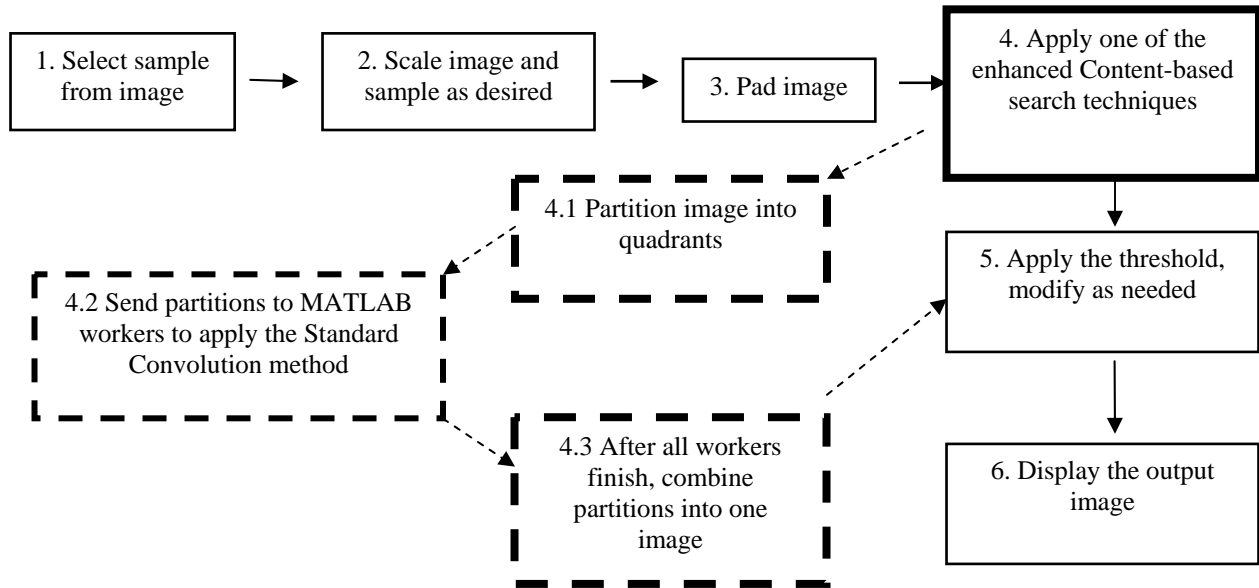


Figure 2 – Computational Process, the Distributed Computational Process is shown in dotted line

The computation steps shown in solid-line in Figure (2) are for a sequential process. To conduct the computation under the Distributed MATLAB Toolbox, we modified the Step (4) in Figure (2) where we applied the content-based techniques in parallel by dividing independent tasks among different processors. The Distributed Computation Process is shown in dotted-line in Figure (2).

### 3.1 Distributed MATLAB Computation Procedure

To conduct our computation in Distributed MATLAB Toolbox, we provided basic required information, such as the name of our function, the number of tasks to divide the job into, and the variable into which the results are returned. We used *Synchronous* evaluation which means that MATLAB is blocked until the evaluation is complete and the results are assigned to the designated variable. We used four different types of partitioning methods to create 2, 4, 8, and 16 partitions. MATLAB distributed tasks were created for each of the partitions and then submitted to the MATLAB workers to perform the computations. Once all the workers completed their computations, their results

were collected to form a single image. The partitioning process of an image for different number of processors is shown in Figure (3).

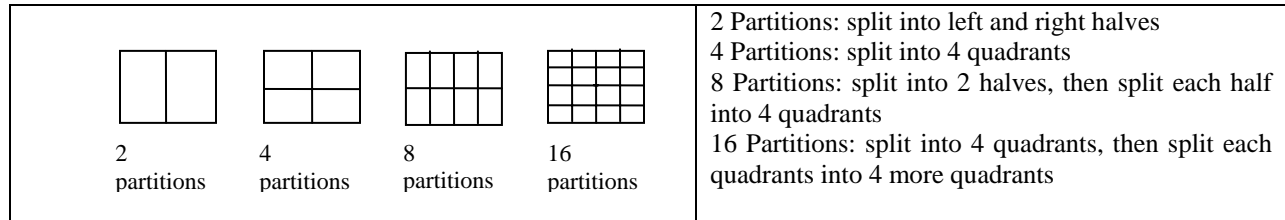


Figure (3) – Partitioning process for different task distributions

When we execute our program, the toolbox performs all the steps required for running a job: Finds a job manager, creates a job, creates tasks in that job, submits the job to the Job Manager queue in the job manager, and finally retrieves the results from the job. We encourage the reader to refer to [Gonzalez 04] for more information on how Distributed MATLAB jobs are created.

When you create and run a job, it progresses through a number of stages. Each stage of a job is reflected in the value of the job object’s State property, which can be pending, queued, running, or finished. Each of these stages is briefly described in this section. Figure (4) illustrates the stages in the life cycle of a job. In the job manager, the jobs are shown categorized by their state. Some of the functions used for managing a job are `createJob`, `submit`, and `getAllOutputArguments`. Our main server played both the role of the Manager and one of the eight Clients.

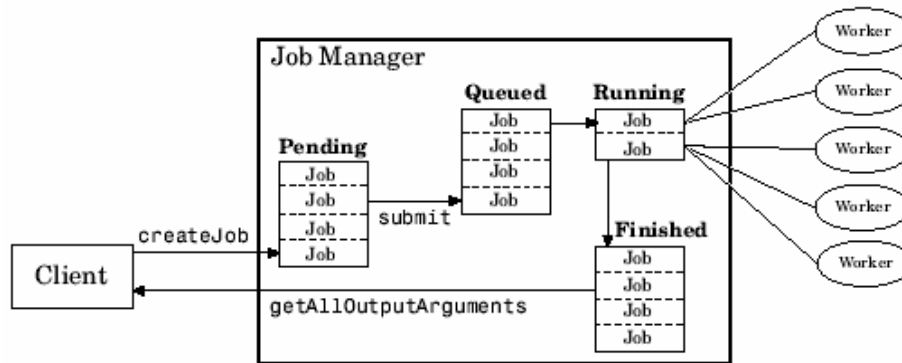


Figure (4) – The stages of a job in Distributed MATLAB [MATLAB 04]

## 4.0 Results

Four different approaches were taken to enhance the effectiveness and efficiency of the Standard Convolution. Our primary goal was to search for sample images of different sizes inside target digital mammograms. The template segments were of different sizes and were cut from the template images. Some of the sample pieces contained microcalcification clusters, which are usually seen as the early indicators of cancer.

Table (1) – Performance of the Sequential Approach

Method*	Mammogram 1		Mammogram 2		Mammogram 1	
	time (sec)	Performance	time (sec)	Performance	time (sec)	Performance
Std. Con	701	Found area	1916	Found area	9118	Failed
Iterative	144	Found area	1561	Found area	2065	Failed
Mean Sq	28	Found area	45	Failed	134	Failed
Pearson	302	Found exact	332	Found exact	639	Found exact
Haar	454	Found area	1351	Failed	6282	Failed

\* Std. Con – Standard Convolution, Iterative – the Iterative version of the Standard Convolution, Mean Sq – Means Square Distance, Pearson – Pearson Correlation, and Haar – Haar Wavelet Lifting Scheme.

Two different results are presented here. The first results are on the performance of the four enhanced techniques introduced on this research without parallelization and the second results for the Distributed MATLAB on the example that had the longest completion time. Initial results were obtained on a Pentium4 PC and the second results were obtained on an 8-node Linux cluster of Pentium PCs.

## 5.0 Conclusions

As the results in Table (1) show, the four different sequential methods introduced in this research were more efficient than their traditional counterparts. Among the techniques introduced in this research, the method that utilized Pearson's Correlation produced the best content-based search results. There were no clear winner among the other techniques as some of the techniques that performed better on one or two images produced worse completion time.

Regardless of the performance in the search process, the Distributed MATLAB was used for the technique that resulted in the longest completion time to demonstrate its application. Thus, we have used the Standard Convolution for this purpose. Figure (5), shows the result of the MATLAB Distributed Toolbox for different number of nodes.

As this results shows we have obtained up to 8 times reduction in the completion time when 8 nodes are used to conduct the search on Mammogram3. We also observed that although we only had 8 single-processor nodes on our cluster, we obtained better completion-time with 16 partitions than that with 8 partitions.

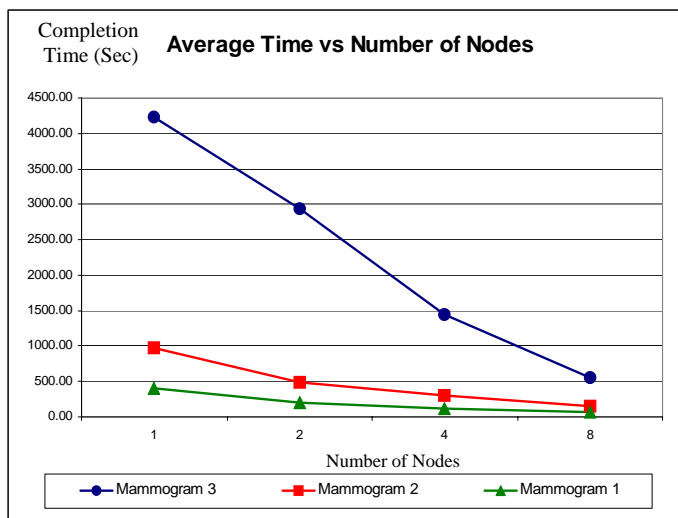


Figure (5) – Performance of Distributed MATALAB for the Standard Convolution

## 6.0 References

- [Daubechies 96] Daubechies, Ingrid, and Sweldens, Wim., Factoring Wavelet Transforms into Lifting Steps, Technical report, Bell Laboratories, Lucent Technologies, 1996.
- [Gonzalez 04] Gonzalez, Rafael C., Woods, Richard E, and Eddins, Steven L. *Digital Image Processing Using MATLAB*. Upper Saddle River, New Jersey: Pearson/Prentice Hall, 2004.
- [Gonzalez 95] Gonzales, Rafael, Woods, Richard. *Digital Image Processing*. Addison Wesley, 2<sup>nd</sup> Edition, 1995.
- [Heffner 05] Heffner, Steven, Efficient Medical Image Content-Based search Approach, an Honor's Thesis, Appalachian State University, May 2005.
- [MATLAB 04] Distributed MATLAB Toolbox's User's Guide, The MathWorks, 2004.
- [Tashakkori 01] Tashakkori, Rahman, Medical Image Set Compression Using Wavelet and Lifting Combined with Scanning Techniques, a Dissertation, Louisiana State University, May 2001.

## Acknowledgments

This research was supported by the "A Consortium to Promote Computational Science and High Performance Computing" grant that was funded by the University of North Carolina – Office of the President in 2004.