Parallel Implementation of Shape based Image Retrieval Approach on CUDA in Compressed Domain

Kuldeep Yadav
Department of CSE,
College of Engineering Roorkee,
Roorkee-247667,Uttarakhand, India

Avi Srivastava and Ankush Mittal
Department of CSE,
College of Engineering Roorkee, Roorkee-247667,
Uttarakhand, India

M.A Ansari
Department of Electrical,
GBU, Greater Noida,
Uttar Pradesh, India

ABSTRACT
Fast and accurate algorithms are necessary for Content based image retrieval (CBIR) systems to perform operations on compressed images databases such as jpeg or through compressive sensing. Feature extraction and feature matching are two important steps in any CBIR system. Wrong matching may affect the accuracy rate of CBIR systems. The matching of query image which is in uncompressed form to image in database which is in compressed form is very challenging. However, existing algorithms suffer from a flawed tradeoff between accuracy and speed. In this research work, shape based image retrieval is carried out using modified standard DCT approach and parallelized it on Graphics Processing Unit (GPU). The main goal of this research work is to make CBIR faster for processing a large number of images database using parallel implementation of algorithms on GPU. GPUs are emerging as powerful parallel systems at a cheaper cost. Our work employs extensive usage of highly multithreaded architecture and shared memory of multi-cored GPU. An efficient use of shared memory is required to optimize parallel reduction in Compute Unified Device Architecture (CUDA). Experimental results show that our method can achieve a speedup of about 15x over the serial implementation when running on a GPU named GeForce 9500 GT having 32 cores. Shape based retrieval method of CBIR is also evaluated using Recall, Precision, F-measure, True Negative rate, and Accuracy evaluation measures

General Terms
Content Based Image Retrieval; DCT; Shape; Parallel Computing.

Keywords
Shape based Image Retrieval; Parallelization; GPU; CUDA

1. INTRODUCTION
Research on CBIR systems is very difficult due to its challenging properties. CBIR research came into existence because of visual feature like shape of the objects in image can be analyzed and the traditional methods [1-5] of image indexing have proven to be insufficient, laborious, and extremely time consuming. These old methods of image indexing, ranging from storing an image in the database and associating it with a keyword or number, to associating it with a categorized description, have become obsolete. In CBIR, each image that is stored in the database has its features extracted and compared to the features of the query image. It involves two steps: first is feature extraction to extract features for comparing entities. Second is feature matching to match extracted features to get similar content image.

The whole process of CBIR can be done with two type of image database first, uncompressed image database and second, compressed image database. The former one is traditional method [1-5] and consumes a lot of time because of limitation in space or memory available to store thousands of images. So the research migrated towards compressed image [6-7] database but it also has some shortcoming too like first decompress the image in database and then feature extraction and matching part of CBIR is applied which is expensive in terms of time consumption. This problem gives birth to new approach of comparing the compressed images [8-15] in database directly without decompressing it with the query uncompressed/compressed image. In this work, we modified the traditional shape based retrieval method to extract feature from compressed database images of JPEG format [16-17] and parallelized it to make faster using CUDA.

Graphical Processing Units (GPUs) have been proved its importance in terms of performance as hardware for computer graphics [8]. Many researchers have already been applied GPUs to implement many algorithms in various areas such as image processing, computational geometry, and scientific computation, as well as computer graphics [19-24]. GPUs play important role to speedup processing of database images matching algorithms because it has more inbuilt execution cores. The parallel implementation of image analysis algorithms using GPU encounters two problems. First, the programmer should master of the fundamentals of GPU and CUDA [25]. CUDA platform is used to implement the parallel implementation of algorithms. Second, in a job it needs much process cooperation between CPU and GPU. Parallel implementations on GPUs have been applied to various numerical problems [26-29] to reduce the computation time without sacrificing the degree of accuracy. Fast CBIR is one of the important problems in the field of computer vision. The decompression of images and their high computation cost are the main drawbacks of slow implementations of uncompressed CBIR systems. Computational cost reduction approaches of CBIR were proposed in [30] by Emmanuel at al. recently.

In the following sections, we present a detailed description of the proposed methodology as well as experimental results that demonstrate the efficiency of the proposed methodology.

2. INTRODCTION TO nVIDIA CUDA ARCHITECTURE

nVIDIA® CUDA™ is a general purpose parallel computing architecture introduced by NVIDIA. It includes the CUDA Instruction Set Architecture (ISA) and the parallel compute engine in the GPU. C language is used to program to the CUDA architecture. One of the most widely used high-level programming languages, which can then be run with great performance on a CUDA enabled processor [31]. CUDA-enabled GPUs have hundreds of cores that can collectively run thousands of computing threads. Each core has shared resources, including registers and memory. The on-chip shared memory allows parallel tasks running on these cores to share data without sending it over the system memory bus [32].

A fundamental building block of CUDA programs is the CUDA kernel function. When launching a CUDA kernel function, a developer specifies how many copies of it to run. We call each of these copies a task. Because of the hardware support of the GPU, each of these tasks can be small, and the developer can queue hundreds of thousands of them for execution at once. These tasks are organized in a two-level hierarchy, block and grid. Small sets of tightly coupled tasks are grouped into blocks. In a given execution of a CUDA kernel function, all blocks contain the same number of tasks. The tasks in a block run concurrently and can easily communicate with each other, which enables useful optimizations such as those of the section "Shared Memory". GPU's hardware keeps multiple blocks in flight at once, with no guarantees about their relative execution order. As a result, synchronization between blocks is difficult. The set of all blocks run during the execution of a CUDA kernel function is called a grid.

The three key abstractions of CUDA are the thread hierarchy, shared memories and barrier synchronization, which render it as only an extension of C. All the GPU threads run the same code and, are very light weight and have a low creation overhead. A kernel can be executed by a one dimensional or two dimensional grids of multiple equally-shaped thread blocks. A thread block is a 3, 2 or 1-dimensional group of threads as shown in Figure 1. Threads within a block can cooperate among themselves by sharing data through some shared memory and synchronizing their execution to coordinate memory accesses. Threads in different blocks cannot cooperate and each block can execute in any order relative to other blocks. The number of threads per block is therefore restricted by the limited memory resources of a processor core. In current GPUs, a thread block may contain up to 512 threads. The multiprocessor SIMT (Single Instruction Multiple Threads) unit creates, manages, schedules, and executes threads in groups of 32 parallel threads called warps.

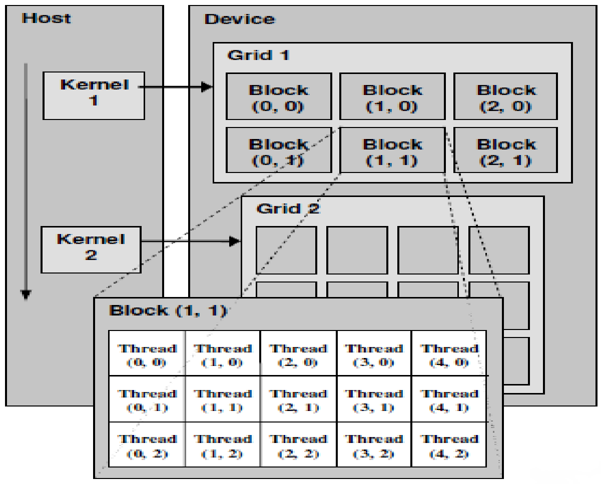


Fig. 1: Thread Hierarchy in CUDA

3. PROPOSED METHODOLOGY

CBIR process consists of two distinct stages. In the first stage, a preliminary feature extraction part is performed that executes JPEG [16-17] method of finding DC/AC coefficients using DCT. This process leads to the isolation of DC and AC coefficients of the images. This sub-group contains the feature of the image. In the second stage, we perform feature matching process which results into the correct difference or distance of database image from the query image and displays top eight results according to their decreasing order of similarity.

3.1 Feature Extraction

Feature extraction is very important part of the shape based image retrieval process which extracts lines and overlap out of the images. The image as a whole is of no use due to the limitation of feature extraction or classification phases. So we need to extract each line out of the images for the use of feature extraction phase. In our approach we are using DCT method which gives DC and AC coefficients.

According to model [33] 5 AC coefficients that are AC_{01}, AC_{10}, AC_{02}, AC_{20}, and AC_{11} can be used as the feature extractor and can be used as feature space for matching it with that of query image.

3.2 Feature Matching

In the feature matching part of the CBIR feature space are taken as an input from feature extraction part and been matched with the query image's feature using Euclidean distance method (eq1.)

$$E = 1/MN\sum_{i=1}^{M}\sum_{j=1}^{n}|x(m,n)| \qquad (1)$$

4. EVALUATION MEASURES

The method of shape based image retrieval is evaluated using the six evaluation measures: Precision, Recall, F-measure, True negative rate, (Negative Rate Metric) NRM and accuracy.

- Precision:

$$\text{Precision} = \frac{TP}{TP+FP} \tag{2}$$

- Recall:

$$\text{Recall} = \frac{TP}{TP+FN} \tag{3}$$

- F-Measure:

$$F - Measure = \frac{2 \times \text{Re}\,call \times \text{Pr}\,ecision}{\text{Re}\,call + \text{Pr}\,ecision} \tag{4}$$

- True Negative Rate:

$$\text{True Negative Rate} = \frac{TN}{TN+FP} \tag{5}$$

- Accuracy:

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \tag{6}$$

- NRM:

$$NRM = \frac{NR_{FN} + NR_{FP}}{2} \tag{7}$$

Where $NR_{FN} = \frac{FN}{FN+TP}$ and $NR_{FP} = \frac{FP}{FP+TN}$

5. IMPLEMENTATION

In this research work, the implementation of proposed approach is based on the two set of experiments. In the first set of experiment, proposed algorithm is implemented in C language and in second set; proposed algorithm is parallelized using CUDA. The following sections 5.1 and 5.2 dictate the detailed description of the sequential and parallel implementation of algorithm.

5.1 Sequential Implementation

The following pseudo codes (algorithm 1) outline the structure of algorithm for different parts.

Algorithm 1: Approach for Feature extraction

```
1) Retrieve the 5 AC coefficients of each block and save it in a matrix.
2) While each block is converted
3) Analyze the AC coefficients of a block and convert it to corresponding binary form.
4) End while
5) Combine all the binary blocks.
6) Traverse 8-connected way to get boundary and
   holes.
7) Arrange the retrieved blocks and holes in decreasing order of their dimension.
8) Discard blocks with dimension less than threshold (in our case it is 500x1)
9) Give the array as the feature space.
```

The following pseudo codes (algorithm 2) outline the structure of algorithm for Feature matching of the query and database image.

Algorithm 2: Approach for Feature Matching

```
1) Select energy set of query image.
2) Select one energy set of image in database.
3) Euclidean distance (eq. 1) and save it in Euclidean vector.
4) If more image in database
           Goto step 2
   Else
           Goto step 4
5) Arrange Euclidean vector in decreasing order of its magnitude.
```

5.2 Parallel Implementation

In CUDA, it is assumed that both host and device maintain their own DRAM. Host memory is allocated using malloc and device memory is allocated using cudaMalloc. CUDA threads are assigned a unique thread ID that identifies its location within the thread, block and grid. This provides a natural way to invoke computation across the image, by using the thread IDs for addressing. The parallel implementation of algorithm of Feature Extraction is shown in the pseudo code (algorithm 3) shown below.

Algorithm 3: Parallel Implementation of Feature Extraction

```
1) Using 5 threads parallel retrieve the 5 AC coefficients
   of each block and save it in a matrix.
2) In matrix using 5 threads analyze the AC coefficients
   of a block and convert to binary form.
3) Combine all binary blocks.
4) Call g_boundary function to compute boundary and
   holes parallel and pass 8 as argument.
5) Discard blocks with dimension less than 500 and
   arrange remaining in decreasing order.
6) Give the result array as feature space.
```

The following pseudo codes (algorithm 4) outline the structure of Feature Matching's parallel algorithm for CBIR.

Algorithm 4: Parallel Implementation of Feature Matching

```
1) Select energy set of query image.
2) Select one energy set of image in database.
3) Parallely calculate Euclidean distance (eq. 1) and save
   it in Euclidean vector.
4) If more image in database
           Goto step 2
   Else
           Goto step 4
5) Arrange Euclidean vector in decreasing order of its
   magnitude.
```

6. HARDWARE SPECIFICATIONS

All the experiments are carried out using the hardware specifications of GPU: GeForce 9500 GT, 1 MB DDR2, No of Processors = 4, No of core =32, RAM 1 GB, Frequency 1.35 GHz, DDR2 and CPU: Intel Core 2 Duo, 2.66 GHZ, No of cores available =2, No of thread=1, No of thread/core=1, Physical Memory =2 GB, DDR2

7. RESULTS AND DISCUSSION

For the testing of shape based retrieval approach of CBIR, we collected a data set of MRI, CT-scan and X-ray to form database of images in compressed format of JPEG. The results of shape based retrieval approach are shown in fig. 5 that demonstrates the efficiency of this approach. On the basis of visual observation, shape based retrieval method of CBIR manages to find images similar to query image in database but with a drawback of a lot of time consumption. To make faster the method, we parallelized it on CUDA and achieved an average speed up of 30x (approx) over the serial implementation when running on a GPU. The comparison of serial implementation over parallel is shown in table 1. Table 1 also shows that execution time depends on the image resolution.

Further, the performance of method is evaluated using Precision, Recall, F-measure, True Negative Rate, NRM and Accuracy measures, which show the effectiveness of method shown in table 2. Fig.4 shows the graph of execution time of GPU in seconds, fig3 shows that of CPU. Fig2 shows the speedup graph. Hence presented method proved that it works better than standard single thread based method and run faster on GPU.

Table 1: Comparison of execution time of CBIR on CPU over GPU

Resolution (a X a)	**Serial**	**Parallel**	**Speed-Up**	**Speed-Up Average**
512	5.342	0.348162	15.343424	14.903994
256	4.566	0.315668	14.464564	
512	4.998	0.3097738	16.134353	14.849887
256	4.112	0.303123	13.565422	
512	5.123	0.338791	15.121415	14.939468
256	3.877	0.262713	14.757522	
512	5.121	0.327125	15.654545	15.395535
256	4.435	0.292999	15.136562	
512	4.787	0.303889	15.752427	15.214049
256	4.122	0.280872	14.675672	
	Average Speed-Up			15.060586

Table 2: Evaluation Measures

Image	**Precision**	**Recall**	**F-Measure**	**TNR**	**NMR**	**Accuracy**
1	50	33.33	39.9976	50	58.33	40
2	66.66	33.33	44.44	66.66	50	44.44
3	100	50	66.67	100	25	75
4	75	75	75	75	75	75
5	100	100	100	50	00	100

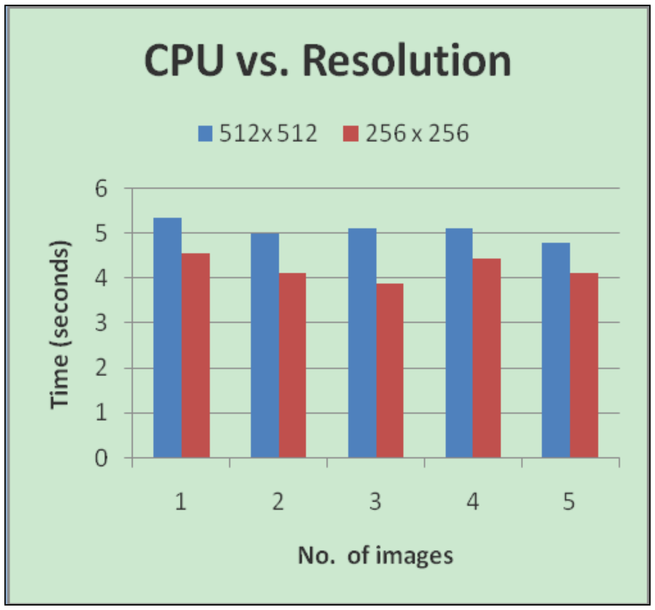


Fig3. CPU vs. Resolution graph

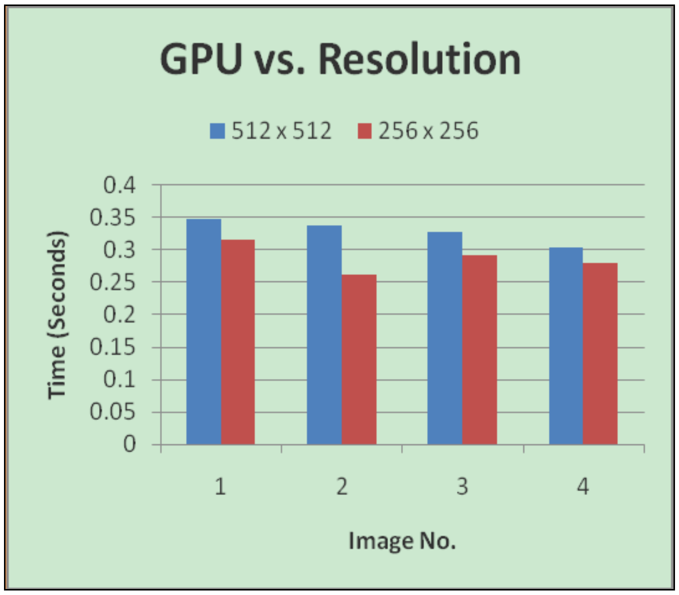


Fig4. GPU vs. Resolution graph

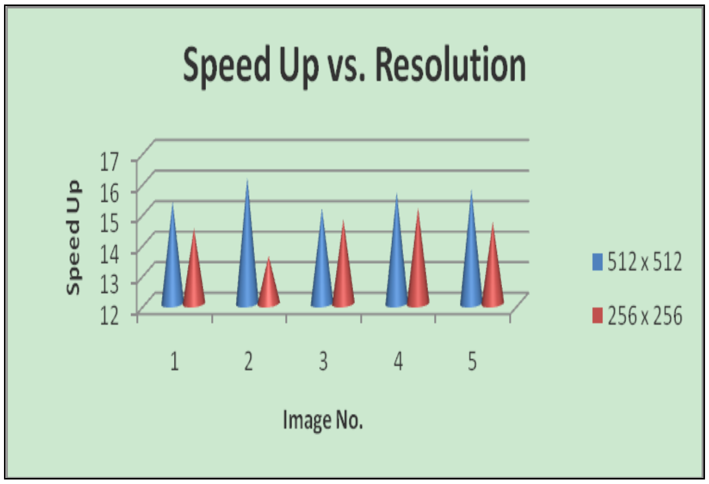


Fig2. Speed up vs. resolution graph

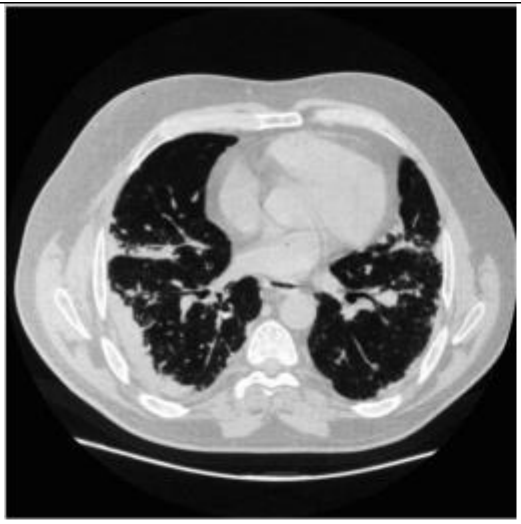
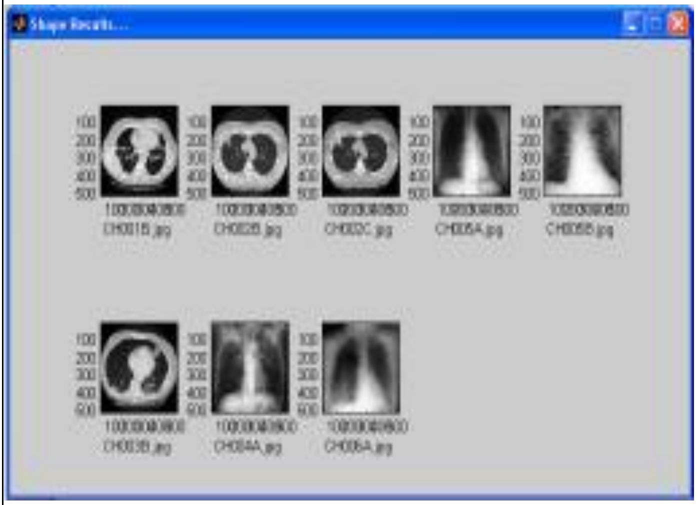
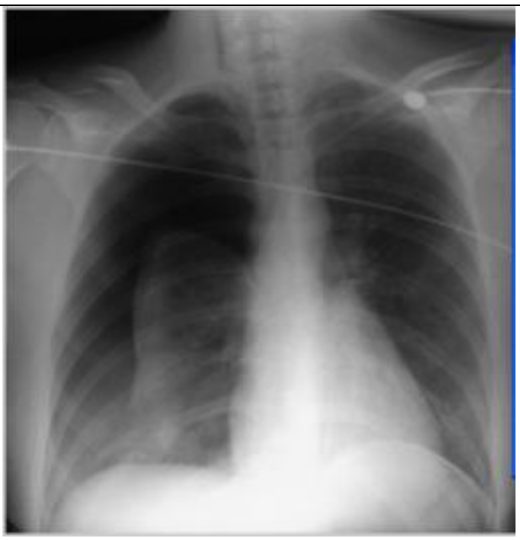
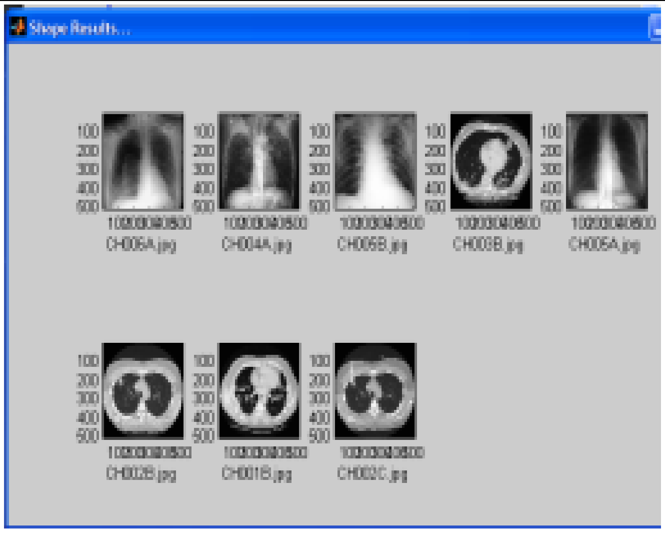
S.N.	Query Image	CBIR output
1.		 Shape Results... CH001B.jpg CH002B.jpg CH002C.jpg CH005A.jpg CH005B.jpg CH003B.jpg CH004A.jpg CH006A.jpg
2.		 Shape Results... CH006A.jpg CH004A.jpg CH005B.jpg CH003B.jpg CH005A.jpg CH002B.jpg CH001B.jpg CH002C.jpg

Fig 5: Shape based Image retrieval results.

6. CONCLUSION

Feature extraction and matching are two of the important steps of any CBIR system. In this research work, a modified parallel algorithm has been presented and analyzed with traditional sequential based approach. The implementation of proposed algorithm on the graphics device is promising, with large two dimensional images (on a relatively low performance GPU) than sequential algorithms. The method finding image in database and how to accelerate this process using GPUs has been discussed in great detail. This algorithm serves as an excellent framework to solve a diverse number of compressed domain CBIR problems. The experiments show that proposed CBIR method even works better than normal single thread execution and also the six evaluation measures shows its' accuracy.

CUDA itself has been shown to be an excellent framework to accelerate computational problems in engineering, and is gaining more features and fewer limitations every few months. The principal disadvantages of CUDA are that it is only effective for very data parallel problems, and that it is not an industry standard. Recently, to counter the latter, it is very likely that it will in fact be replaced by OpenCL (Open Computing Language). The syntax and architecture between CUDA and OpenCL will be very similar, allowing this code to be easily ported to OpenCL. Nonetheless the impressive speedups attained using such low end hardware demonstrate the power of this parallel CBIR algorithm.

7. REFERENCES

[1] Antonio da Luz Jr, Daniel D. Abdala, Aldo v.wangenheim, Eros Comunello," Improving Performance and Quality in Content-Based Medical Image Retrieval", Twentieth IEEE International Symposium on Computer-Based Medical Systems (CBMS'07) .

[2] Mohamad Obeid', Bruno Jedynak, Mohamed Daoudi, "Improving Performance and Quality in Content-Based Medical Image Retrieval", Twentieth IEEE International Symposium on Computer-Based Medical Systems (CBMS'07) .

[3] Aliaa.A.A.Yousiff, A.A. Darwish, R.A. Mohmed," Content based medical image retrieval based on pyramid structure wavelet", IJCSNS International Journal of Computer Science and Network Security, VOL.10 No.3, March 2010

[4] T. M. Lehmann, M. O. Güld, C. Thies, B. Fischer, K. Spitzer, D. Keysers, H. Ney,M. Kohnen, H. Schubert, B. B. Wein, "Content-based Image Retrieval in Medical Applications", 2004.

[5] Yong Rui and Thomas S. Huang, Shih-Fu Chang, "Image Retrieval: Current Techniques, Promising Directions, and Open Issues", Journal of Visual Communication and Image Representation 10, pp 39–62 1999 .http://www.idealibrary.com on ideal.

[6] Ch. Theoharatos, V.K. Pothos, G. Economou and S. Fotopoulos, "compressed domain image indexing and retrieval based on the minimal spanning tree", 2005.

[7] Matthew J. Zukoski, Terrance Boult, " A novel approach to medical image compression", Int. J. Bioinformatics Research and Applications, Vol. 2, No. 1, 2006

[8] Daidi. Zhong, Irek. Defeel, "Study of Image Retrieval Based on Feature Vectors in Compressed Domain".

[9] Jinshan Tang, Eli Peli, and Scott Acton, "Image Enhancement Using a Contrast Measure in the Compressed Domain", IEEE signal processing letters, vol. 10, no. 10, october 2003

[10] Salih Burak Gokturk, Carlo Tomasi, Bernd Girod, Chris Beaulieu, " medical image compression based on region of interest, with application to colon ct images", Electrical Engineering, Computer Science, Radiology Departments, Stanford University

[11] Vidya R. Khapli, Anjali S Bhalchandra, "Compressed Domain Image Retrieval Using Thumbnails of Images", First International Conference on Computational Intelligence, Communication Systems and Networks, 978-0-7695-3743-6/09 2009.

[12] Ruey-Feng Chang, Wen-Jia Kuo and Hung-Chi Tsai, "image retrieval on uncompressed and compressed domains", Department of Computer Science and Information Engineering , National Chung Cheng University, Chiayi, Taiwan 621, R.O.C., 0-7803-6297-7/00/ 2000 .

[13] Rami Al-Tayeche and Ahmed Khalil, "CBIR: Content Based Image Retrieval"Department of Systems and Computer Engineering Faculty of Engineering Carleton University" Tech. Rep. April 4, 2003.

[14] M. Hatzigiorgaki and A. N. Skodras, "Compressed Domain Image Retrieval: A Comparative Study of Similarity Metrics", Visual Communications and Image Processing Touradj Ebrahimi, Thomas Sikora, Editors, Proceedings of SPIE Vol. 5150 (2003)

[15] Gerald Schaefer and Roman Starosolski, "A comparison of two methods for retrieval of medical images in the compressed domain", 30th Annual International IEEE EMBS Conference Vancouver, British Columbia, Canada, August 20-24, 2008.

[16] Farzad Zargari, Ali Mosleh and Mohammad Ghanbari, " A Fast and Efficient Compressed Domain JPEG2000 Image Retrieval Method", IEEE Transactions on Consumer Electronics, Vol. 54, No. 4, november 2008

[17] Lin NI, "A Novel Image Retrieval Scheme in JPEG2000 Compressed Domain Based on Tree Distance",ICICS-PCM 15 Dec ,2003.

[18] A.Ashbrook and N.A.Thacker, "Tutorial: Algorithms for 2-Dimensional Object Recognition", 1 / 12 / 1998.

[19] Fernando, R and Kilgard, M. J. The Cg tutorial the definitive guide to programmable real-time graphics. Addison-Wesley 2003.

[20] Moravanszky, Linear algebra on the GPU, in: W.F. Engel (Ed.), Shader X 2, Wordware Publishing, Texas,2003.

[21] Manocha, D. "Interactive geometric & scientific computations using graphics hardware", SIGGRAPH 2003

[22] Moreland, K. and Angel E. The FFT on a GPU. In Proceedings of SIGGRAPH Conference on Graphics Hardware, 112-119, 2003.

[23] Mairal, J., Keriven, R. and Chariot, A. "Fast and efficient dense variational Stereo on GPU". In Proceedings of International Symposium on 3D Data Processing, Visualization, and Transmission, 97-704, 2006.

[24] Yang, R. and Welch, G. "Fast image CBIR and smoothing using commodity graphics hardware". Journal of Graphics Tools, Vol. 17, (4), 91-100, 2002.

[25] Fung, J. and Man, S. OpenVIDIA: Parallel GPU computer vision. In Proceedings of ACM International Conference on Multimedia, pp. 849-852, 2005.

[26] Owens, J. D. Luebke, D., Govindaraju, N., Harris, M., Kruger, J., Lefohn, A. E. and Purcell, T. J. "A survey of general-purpose computation on graphics hardware". In

proceeding of Eurographics, State of the Art Reports, 21–51, 2005.

[27] Larsen, E. S., McAllister, D. "Fast Matrix Multiplies using Graphics Hardware". In Proceeding of International Conference for High Performance Computing and Communications, pp.159-168, 2001.

[28] Trendall C. and Stewart, A. J. "General calculations using graphics hardware with applications to interactive caustics. Rendering Techniques": 11th Eurographics Workshop on Rendering, 287-298, 2000.

[29] Li, Wei, Wei, Xiaoming, A. and Kaufman, "Implementing lattice boltzmann computation on graphics hardware". In proceeding of the International Conference for High Performance Computing and Communications.

[30] M. Emmanuel, D.R. Ramesh Babu, Jayashree Jagdale, Pravin Game and G.P. Potdar," Parallel Approach for Content Based Medical Image Retrieval System", Journal of Computer Science 6 (11):pp. 1258-1262, 2010.

[31] NVIDIA CUDA Programming Guide Version 2.0, available at www.nvidia.com/object/cuda_develop.html.

[32] NVIDIA Corporation: NVIDIA CUDA programming guide. Jan 2007, available at http://developer.download.nvidia.com/compute/cuda/2_0/docs/NVIDIA_CUDA_Programming_Guide_2.0.pdf

[33] Zhang Xihuang, Bian Guochun,Xu Wenbo, "A Shape Feature Based Image Retrieval in DCT Compressed-Domain",The Fifth International Conference on Computer and Information Technology (CIT'05), Proceedings of the 2005