GPU-based normalized cuts for road extraction using satellite imagery

J. Senthilnath[a], S. Sindhu[a], S.N. Omkar[a*]

[a]*Department of Aerospace Engineering, Indian Institute of Science, Bangalore,INDIA*

Abstract

This paper presents a GPU implementation of normalized cuts for road extraction problem using panchromatic satellite imagery. The roads have been extracted in three stages namely pre-processing, image segmentation and post-processing. Initially, the image is pre-processed to improve the tolerance by reducing the clutter (that mostly represents the buildings, vegetation and fallow regions). The road regions are then extracted using the normalized cuts algorithm. Normalized cuts algorithm is a graph based partitioning approach whose focus lies in extracting the global impression (perceptual grouping) of an image rather than local features. For the segmented image, post-processing is carried out using morphological operations - erosion and dilation. Finally, the road extracted image is overlaid on the original image. Here, a GPGPU (General Purpose Graphical Processing Unit) approach has been adopted to implement the same algorithm on the GPU for fast processing. A performance comparison of this proposed GPU implementation of normalized cuts algorithm with the earlier algorithm (CPU implementation) is presented. From the results, we conclude that the computational improvement in terms of time as the size of image increases for the proposed GPU implementation of normalized cuts. Also, a qualitative and quantitative assessment of the segmentation results has been projected.

Keywords: Road extraction; Normalized cuts; Graphical processing unit.

[*] Corresponding author. Tel.: +91-080-229-32873; fax: +91-080-236-00134
Email address: omkar@aero.iisc.ernet.in (S.N.Omkar).

1. Introduction

Roads are among the most important topographical objects for extraction from aerial and satellite imagery for the use in data acquisition and update of the Geographic Information Systems (GIS). Automatic extraction of roads has been an active area of research from the past two decades now (Baumgartner et al. 1996, Valero et al. 2010). The road network information is of utmost importance for transportation, urban planning, automated road navigation and emergency response applications. The automatic extraction of the roads is posed with many challenges. The most important ones include huge objects that might cause occlusions like shadows on the roads, the presence of vehicles and road markings (Senthilnath et al. 2009). Therefore, we employ an algorithm that considers low level coherence of brightness, colour and texture of the image for the automation of this semantic task (Shi and Malik, 2000, Senthilnath et al. 2013). This study focuses on the extraction of roads using satellite imagery.

Several methods have been developed for road extraction and are summarized in (Mayer et al. 2006). To deal with satellite images of urban and sub-urban areas, a very few algorithms retain their reliability owing to the highly complex structure found in urban scenes (Grote et al. 2007). Semi-automated approaches require human intervention and user provided clues for the identification of features on the image. Gruen and Li (1997) formulated the problem using an active contour model in a least square context (LSB-Snake) where, by initializing seed points, road segments is generated by optimizing an energy function based on geometric characteristics. Geman and Jedynak (1996) proposed a semi-automatic method, where, given a start point and a start direction, a road is extracted from panchromatic SPOT satellite image by playing “tests” about the “true hypothesis”. Park and Kim (2001) has successfully applied road extraction based on template matching.

Barzohar and Cooper (1996) successfully automated road extraction model to extract roads through dynamic programming. In their study, roads are found by Maximum Aposteriori Probability (MAP) estimation, which is handled by partitioning an image into multiple regions. In their study, road segments are extracted using geometric-stochastic model. The geometric-stochastic model of a road is built on certain assumptions in which, the width of the road is assumed to be less varying whereas abrupt change in its direction is not considered. Also, roads are assumed to have high contrast against their background and are unlikely to be short. However, these assumptions will seldom be universally applicable as road images may vary with ground resolution, road type and density of surrounding objects (Bong et al. 2009). Youn and Bethel 2004, extracted road networks, by assuming roads are semi-regular grid pattern. This constraint may not be suitable for many European urban areas (Grote et al. 2007). Roads in different parts of the world exhibit different characteristics. The notion of straightness is more pronounced in a road in Australia than in central Europe. Similarly in India, the spectral information, textures, width of the roads, road markings and clutter vary depending on the location (Senthilnath et al. 2009). Hence, it is helpful to include both local and global features in an extraction strategy (Grote and Heipke, 2008).

From the above works, we deduce that image segmentation formulated on specific types of image scenes and road models must not only consider spatial and spectral properties but must also include global features of the image for reliable segmentation. In the present study, we have adopted the normalized cuts algorithm. It includes several spectral features to describe pixel similarity. The similarity factor takes into account several features of the image including the context information, which is necessary for the correct segmentation of images with complex entities such as built-up areas (Grote et al. 2007). In literature, normalized cut method has been successfully applied for road extraction using high resolution satellite image (Grote et al. 2007, Grote and Heipke, 2008, Senthilnath et al. 2009). In (Senthilnath et al. 2009), it has been observed that normalized cut method performs better than texture progressive analysis for high resolution satellite. The fact that for high resolution data, with high variability leads to predominant edges where normalized cut algorithm identifies these edges to form and find the optimal “cut” for iterative segmentation of the image. Hence, the use of high resolution image stimulates a better percentage of correctness using normalized cut algorithm in comparison with conventional methods.

Normalized cut is a grouping algorithm based on the view that perceptual grouping should be a process that aims to extract local and global impressions of a scene (Shi and Malik 2000). The drawback of this method is that the calculation is computationally expensive and as the image size grows, the images need to be split into subsets for processing (Grote et al. 2007). With the increase in image size, the image segmentation time also increases. An insight into fast processing based on the power of GPU (Graphical Processing Unit) plays an important role at this juncture. Hence, in our study we use GPU implementation of normalized cuts for road extraction problem using high resolution satellite image to increase the computation time.

At the inception of NVIDIA’s CUDA (Compute Unified Device Architecture) - a hardware and software co-processing architecture for parallel computing, several image processing algorithms have been implemented using GPGPU (Rubin 2011; Wendykier and Nagy, 2011; Moreland and Angel, 2003). Recently, to increase the computational speed, GPU’s is widely used. GPU has evolved from special-purpose graphics rendering processor to a programmable processor, while ease of programmability has been the most important feature (Yang et al. 2008). Since the real-time aspect is getting more and more important in image processing and especially in image segmentation, parallel hardware architectures and programming models for multi-core computing have been developed to achieve acceleration (Abramov et al. 2011). In (Pan et al. 2008), medical image segmentation has been performed using the watershed method and region growing algorithm on CUDA enabled GPUs. Abramov et al. (2011), have achieved image segmentation by super paramagnetic clustering using metropolis algorithm. This work has been executed on the framework NVIDIA CUDA. Apart from the above works (Moreland and Angel, 2003) and (Yang et al. 2008) reflect many other image processing algorithms - the FFT and Histogram equalization respectively, that have been adapted to run on the GPU.

To implement the normalized cuts algorithm on the GPU, we have made use of a middleware called Jacket (Acclereyes®). Jacket enables us to execute the computationally expensive parts of the algorithm on the GPU with a negligible change in its counterpart implementation (Rubin et al. 2011). It provides a transparent mapping by allowing the standard MATLAB code to be run on the GPU. In (Wendykier and Nagy, 2011), 2D and 3D image de-blurring techniques have been implemented on the GPU using the Jacket software - the results show a significant acceleration in time. In addition, works in image processing, implemented on Jacket, that have proven to produce convincing results can be found in (Rubin et al. 2011) and (Kong et al. 2010).

In this paper, we consider a CPU implementation of normalized cuts method that was used in earlier studies for the image segmentation (Shi and Malik, 2000). Previous works on road extraction using normalized cuts are available in (Senthilnath et al. 2009) and (Grote et al. 2007). These works have successfully implemented normalized cuts algorithm to extract road segments in urban and suburban scenes but have faced the problem of increasing time taken for the segmentation of larger images. In this paper, we present a GPU-based implementation of normalized cuts using Jacket software for road extraction problem. Our paper addresses the issue of road extraction considering multi-sensor panchromatic satellite image with different scenarios such as the Indian and

Australian. The roads have been extracted in three stages namely pre-processing, image segmentation and post processing. Initially, the image is pre-processed to improve the tolerance by reducing the clutter (that mostly represents the buildings, vegetation and fallow regions). Further, the road segments are extracted using the normalized cuts algorithm. For the road segmented image, erosion and dilation have been applied as post-processing and then the road extracted image is overlaid on the original image. The image segmentation mainly depends on the spectral sensitivity and spatial resolution of the sensor. We have made use of the high-resolution panchromatic satellite images such as IKONOS and QuickBird.

The important aspects of the present work are: a) implementation of the normalized cut method on the GPU to achieve acceleration; b) The acceleration in time is compared between CPU and GPU implementation of normalized cuts for road extraction using satellite imagery (IKONOS and QuickBird); c) A quantitative and qualitative analysis of the road segmented results is done considering the results of both the CPU and GPU-based normalized cuts. From the results, we conclude that the computational improvement in terms of time as the size of image increases for the proposed GPU implementation of normalized cuts.

The remainder of this paper is organized as follows: Section 2 describes the methodology for road extraction, Section 3 discusses system architecture and implementation, Section 4 provides an overview of the study area and the results, and conclusions are presented in Section 5.

2. Methodology

The proposed road extraction has been broadly divided into three steps: pre-processing, application of normalized cuts algorithm on the pre-processed image and post-processing.

2.1 Pre-processing

Pre-processing is required to improve the image quality and to generate the elongated road network for further processing. This involves three steps viz. classification, grouping and filtering.

2.1.1 Classification.

Initially all the images were subjected to classification using the ISODATA –unsupervised classification method through ERDAS tool®. For each image a level one classification was carried out based on two classes: roads and non-roads. Totally 15 clusters are generated among which five clusters correspond to road networks. Here, a lot of misclassifications were noted among roads and barren land. A few cases of misclassification of roads to vegetation and built-up region were also seen.

2.1.2 Grouping

The purpose of grouping comes into picture post classification due to the occurrence of misclassification. A nearest neighbourhood grouping (NNG) operation is applied to the above classified image. This smoothes the spectral response within the pixel's local neighbourhood. In this process the vehicular occlusions and small patches of road contrast abnormalities are eliminated. In this method, a pixel is chosen and its surrounding eight neighbouring pixels' classes are considered. Now, the class of the considered pixel is assigned by majority vote of similar class. The pixel retains its class otherwise. Road pixels in the neighbourhood grow over such noise elements and homogeneous regions/segments are generated.

2.1.3 Filtering

The method of road extraction in urban areas poses challenge because the spectral reflectance of some of the old buildings (or buildings with a type of construction that renders a dark road like effect) which resembles the road surface. Such buildings form the clutter; hence these non-road structures need to be removed. Median filtering technique can be applied as it is very useful in removing random noise (Huang et al. 1979). The median filtering approach is similar to averaging filter, where output pixel is set to an average of the pixel values considering the neighborhood of the corresponding input pixel. However, in median filtering, the value of an output pixel is determined by the median of the neighborhood pixels. The median is much less sensitive than the mean to extreme values (called outliers) (Huang et al. 1979). Median filtering is therefore better suited to remove these outliers without reducing the sharpness of the image.

2.2 Image Segmentation using Normalized Cuts

In this section, we pursue image segmentation in the framework of normalized cuts introduced by Shi and Malik (2000). For the pre-processed image, road segments are extracted automatically using normalized cuts.

Inspired by spectral graph theory (Chung 1997), Shi and Malik (2000) formulated visual grouping as a graph partitioning problem. Normalized cut is a graph based method which is used to divide an undirected graph with weighted edges into segments with similar features. The method in detail is described in (Shi and Malik 2000). In image segmentation, pixels are the entities that we want to partition and hence become the nodes of the graph. The edges between two nodes indicate the strength with which these two nodes belong to one group. The image is considered to be a weighted graph where the nodes i and j are pixels and edge weights, W_{ij}, denote a local measure of similarity between the two pixels. The criterion for partitioning the graph is to minimize the sum of weights of connections across the groups and maximize the sum of weights of connections within the groups (Shi and Malik 2000).

Theoretically, every pixel can be connected to every other pixel in the graph. But, in practice only pixels in the vicinity (by specifying the region of neighbourhood parameter) of the reference pixel are connected with weights different from zero (Grote et al. 2007).

Consider an undirected graph *G = (V, E),* where *V* is the set of nodes and *E* is the set of edges. A pair of nodes *p* and *q* is connected by an edge and is weighed by *W(p, q)*. Let *X* and *Y* be a partition of the graph where $X \bigcup Y = V$ and $X \bigcap Y = \varphi$. Therefore cost of cut is defined as the similarity between the groups *X* and *Y* and it is formulated as:

$$Cut(X,Y) = \sum_{p \in X, q \in Y} w(p,q) \qquad (1)$$

Minimum cut is the cut of minimum weight, where weight of cut *(X,Y)* is given in equation (1). The limitation of using minimum cut is described in (Shi and Malik, 2000). However, the minimum cut criterion favours grouping small sets of isolated nodes in the graph because it cuts with lesser weight than the ideal cut (Shi and Malik, 2000). In other words, the minimum cut usually yields over clustered results when it is recursively applied. This calls for a different cut criterion. Shi and Malik (2000) proposed to use a normalized cut criterion

$$Ncut(X,Y) = \frac{cut(X,Y)}{volume(X)} + \frac{cut(X,Y)}{volume(Y)} \qquad (2)$$

where, *volume(X)* is the sum of cost of all edges that touch *X* and *volume(Y)* is the sum of cost of all edges that touch *Y* i.e.

$$volume(X) = assoc(X,V) = \sum_{p\in X, q\in V} w(p,q) \quad (3)$$

$$volume(Y) = assoc(Y,V) = \sum_{p\in Y, q\in V} w(p,q) \quad (4)$$

where, *assoc(X,V)* is the total connection from nodes in *X* to all the nodes in the graph and *assoc(Y,V)* is the total connection from nodes in *Y* to all the nodes in the graph.

One prominent advantage of this algorithm is that it takes into account both the local and global characteristics of the image. Local characteristics are based on image cues like intensity and texture that are incorporated into the similarity matrix by the calculation of weights between neighbouring pixels. The global characteristics are involved in the calculation of global optimal partition criterion (Grote et al. 2007). The global optimal partition criterion is shown in equation (5). Let *W* be the cost matrix, i.e. $W(i,j) = c_{i,j}$; is the weight between the nodes *i* and *j* in the graph and *D* be the diagonal matrix such that *D(i, i)* is the sum of costs from node *i* to all the other nodes in the graph and *D(i , j) = 0*. Based on this input, Shi and Malik (2000) showed that the optimal partition can be found by computing:

$$\min N_{cut}(X,Y) = \min_{\mathbf{y}} \frac{\mathbf{y^T}(\mathbf{D}-\mathbf{W})\mathbf{y}}{\mathbf{y^T D y}} \quad (5)$$

Such that $y(i) \in \{1, -b\}$, $0 < b \le 1$, and $\mathbf{y}^T\mathbf{D1} = 0$

A slight relaxation induced in terms of *y(i)* being able to take up real values would lead to the minimization of equation (5). Equation (5) can now be solved as a generalized eigen value problem.

$$(\boldsymbol{D}-\boldsymbol{W})\boldsymbol{y} = \lambda \boldsymbol{D}\boldsymbol{y} \quad (6)$$

Hence, the global minimum criterion that is to be met is computed efficiently. This has a significant advantage where a small disturbances like short or weak edges are ignored by the algorithm (Grote et al. 2007).

In normalized cut method for extracting road segments, each individual edge weight is a measure of the similarity between two connected pixels. The similarity measure is basically application dependent. For example, if there is a significant variation in the intensities between road and non-road regions, then calculation of the similarity matrix is carried out based on intensity. However, it is also possible to combine various similarity criteria based on the image cues (Senthilnath et al. 2009). The weight, which is the combination of all similarity measures, has a value between 0 and 1. The graph representing the image is then cut into segments aiming at a large dissimilarity between different segments and at the same time a large similarity within the segments. The weight calculated for each pixel pair is then inserted into a symmetrical matrix the size of whose rows and columns is each equal to the number of pixels in the image.

$$W(i,j) = \begin{cases} e^{-\|F_i - F_j\|/\sigma_I^2} \times e^{-\|X_i - X_j\|/\sigma_X^2} & \text{if } \|X_i - X_j\|_2 < r \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

where,

$\|F_i - F_j\| =$ Image feature similarity

$\|X_i - X_j\| =$ Spatial Proximity

r = region of neighbourhood definition

With the image cases that we have worked on, the basis for calculation of similarity matrix is the intensity factor. Once the similarity matrix is formed, the minimization of the *Ncut* parameter (indicated in equation (5)) is done, the number of partitions possible or in essence the number of eigen vectors obtained after solving the generalized eigen problem is equivalent to the number of pixels (Shi and Malik, 2000).

2.3. Post-processing

As a part of post-processing, the extracted road region obtained using normalized cuts method is converted into binary image format. The morphological operations such as erosion and dilation are used to remove the noise obtained after segmentation (Haralick et al. 1987). Erosion removes unwanted pixels on object boundaries. Dilation adds pixels to the boundaries of objects in an image. It generally increases the sizes of objects, by filling in holes, broken areas, and also connecting areas that are separated by spaces. In order to illustrate the accuracy, this binary image is overlaid on the original panchromatic image. This leads to display the road topology by avoiding the complex noise elements. In the overlaid image, the thin lines indicate the road topology**.**

2.4. Panchromatic Satellite Image

One of the study areas chosen is the Hobart city in Tasmania, Australia. We have used IKONOS panchromatic image with a resolution of 1*m*. The size of the IKONOS image used for this study is *624 X 488* pixels.

The second study area is the urban part of the Bangalore city in Karnataka, India. We use an image obtained from the QuickBird high-resolution satellite owned and operated by Digital Globe. Quick Bird’s panchromatic image has a resolution of 0.61 m. The size of the image is *900 X 900* pixels.

2.5 Performance Evaluation

We have adopted three measures from the work of Wiedemann et al. (1998). In (Wiedemann et al. 1998), the matched extracted data are denoted as true positive with length *TP* emphasizing the fact that the extraction algorithm has indeed found a road, the unmatched extracted data is denoted as false positive with length *FP*, because the extracted road hypotheses are in-correct. The unmatched reference data are denoted as false negative with length *FN*. The quality measures used to evaluate the reference data and automatic road extraction data are:

i. Completeness: This is the percentage of the reference data which is explained by the extracted data, i.e., the part of the reference network which lies within the buffer around the extracted data. The optimum value for completeness is 1 (or 100% efficiency).

$$Completeness = \frac{\text{length of matched reference}}{\text{length of reference}} = \frac{TP}{TP + FN} \qquad (8)$$

ii. Correctness: It represents the percentage of correctly extracted road data, i.e., the part of the extracted data which lie within the buffer around the reference network. The optimum value for correctness is 1 (or 100% efficiency).

$$Correctness = \frac{\text{length of matched extraction}}{\text{length of extraction}} = \frac{TP}{TP + FP} \qquad (9)$$

iii. Quality: It is a measure of the goodness of the final result by combining both completeness and correctness into a single measure. The optimum value for quality is 1 (or 100% efficiency).

$$Quality = \frac{\text{length of matched extraction}}{\text{length of extraction + length of unmatched reference}} = \frac{TP}{TP + FP + FN} \qquad (10)$$

3. System architecture and Implementation

For the algorithm implementation, we have made use of a quad core CPU-Q9400 running at 2.66 GHz and with 3.99GB of RAM. The configuration of the GPU is NVIDIA GeForce GTX 460 SE with 1024 MB of memory. In this section, the system architecture for satellite image processing, and CPU and GPU implementation of normalized cuts algorithm has been discussed.

3.1 System Architecture

Acclereyes® Jacket is a transparent middleware basically built on the NVIDIA CUDA technology. Jacket basically relies on a set of dubbed '*g*' prefixed classes which mimic the native Matlab classes by performing the same function on the GPU. A pictorial representation of Jacket architecture is shown in figure 1. These classes overload the built-in set of base Matlab functions which provide full GPU computation functionality in the Matlab environment through polymorphism (Acclereyes, 2011). In essence, the standard functions present on Matlab are made available to the '*g*' objects via GPU-enabled mex routines. These functions perform the same counterpart Matlab functions faster. Jacket uses some special features to execute the dubbed classes on GPU. Some of them are lazy execution, Just in time compilation (JIT), garbage collection, thread extraction and instruction caching (Acclereyes, 2011).

(PLEASE INSERT FIGURE 1 HERE)

3.2 CPU implementation of Normalized cuts

We first explain the CPU implementation of the algorithm and next the changes incorporated to convert it to a GPU implementation.

(PLEASE INSERT FIGURE 2 HERE)

The flowchart of the algorithm is presented in figure 2. The image matrix- *I* of size *X*Y* (where, *X*- number of rows and *Y*- number of columns) is given as the input to the algorithm. The parameters σ_x and σ_I have been tuned based on the considered image. '*R*' indicates the region of neighbourhood definition and '*ncut*' indicates a threshold for repetitive partitioning - All these parameters are user specified.

3.3 GPU implementation of normalized cuts

GPU implementation requires two important classes of modifications - the code vectorization and then the GPU execution.

(PLEASE INSERT FIGURE 3 HERE)

In Matlab, "vectorization" refers to taking advantage of *polymorphism*, a compiler feature that allows the same line of code to apply to scalars, vectors, or matrices. Matlab can perform these vectorized calculations much more efficiently than loops and automatically multithread some operations (Rubin et al. 2011). We have used the CPU-based normalized cuts implementation from the earlier study of Seo (2006). The same implementation has been modified for GPU-based normalized cuts. The flowchart shown in figure 3 details the steps involved in the GPU implementation. The steps that have undergone modifications have been explained below:

Step 1: Given an image *I* of size *X*Y*, in order to carry out further calculations on the GPU (using Jacket software), we cast the main image variable onto GPU, by the following syntax

I = imread('image.jpg');   //on CPU: where *I* is of 'single ' data type.//

I = gsingle(*I*); // on GPU through specific data type of single precision by the prefix '*g*'

Step 2: One among the more time consuming parts of the algorithm is the similarity matrix computation. The acceleration in terms of time is achieved basically by performing every iteration of the matrix-vector multiplication on the GPU.

The similarity matrix (W) computation involves $N*N$ iterations where $N= X*Y$. Thus by performing these multiplications on the GPU we have achieved a "per-iteration-speedup". The W calculated is stored as a full matrix since sparse matrices are not fully supported by Jacket. Another computationally expensive part of this algorithm happens to be the evaluation of selective eigen vectors and eigen values. Jacket does not support the associated package viz. LAPACK routine (Wendykier and Nagy, 2011).

$$\text{Let the} \qquad \textit{Full matrix} = W_1 \tag{11}$$

The variable is transferred back to the CPU by merely changing its data type

$$W = \textit{single}\ (W_1); \tag{12}$$

As a result, after the similarity matrix has been calculated, the full matrix is converted to a sparse matrix and the further calculations are as per the CPU implementation.

4. Results and Discussion

In this section, we present the results obtained for the problem of road extraction using panchromatic satellite image. In our study, we consider two panchromatic satellite images. The first satellite image - IKONOS image is the Hobart city in Tasmania, Australia. The second satellite image - QuickBird image is the Bangalore city in Karnataka, India.

The aim of this study is to present the results obtained for IKONOS and QuickBird panchromatic images for road extraction using normalized cuts - CPU and GPU implementation. Finally, the performance of the road extraction and processing time as the increase in the size of satellite image is evaluated and analyzed.

The first two images shown in Table 1 are smaller parts of the *624 X 488* image. The increasing sizes of the images are considered to project the computational improvement in time. For illustrating the result, we consider the extracted road image (*624 X 488 pixels*) using CPU and GPU implementation of normalized cuts are shown in figure 4 and figure 5 respectively. The segmented QuickBird image of size *900 X 900 pixels* using CPU and GPU are shown in Figure 6 and Figure 7 respectively.

In this study, normalized cut algorithm uses intensity factor as similarity measures. The panchromatic images are segmented into two segments: roads and non-roads. This is done by using normalized cut algorithm with spectral reflectance of each pixel. For this segmented image morphological operation is applied to remove other regions falsely segmented as roads. In this process other urban areas such as paved parking areas and non-road regions are also segmented as roads. This is because of the reason that the paved parking areas and non-road regions have spectral reflectance similar to roads. The other urban areas such as paved parking areas and non-road regions are called false roads. In Figures (4)-(7), we can observe false roads are extracted as road segment.

(PLEASE INSERT TABLE 1 HERE)

(PLEASE INSERT FIGURE 4 HERE)

(PLEASE INSERT FIGURE 5 HERE)

(PLEASE INSERT FIGURE 6 HERE)

(PLEASE INSERT FIGURE 7 HERE)

Evaluation of road extraction system is done by comparing the results with the reference data. The quality measures used for road extraction using normalized cut method are completeness, correctness and quality factor. Table 1 shows the numeric results of the qualitative analysis. The average completeness in % is 88.35 for CPU based results and 87.5 for GPU based results. The average correctness is around 91.6% and 87.3% for CPU and GPU based implementations respectively. Average quality factor in % is 80.4 on CPU results and 76.1 on GPU based results.

(PLEASE INSERT TABLE 2 HERE)

Road segmentation involves fine tuning of the four parameters involved in normalized cut. After several runs, these four parameter values of normalized cut algorithm on CPU and GPU were empirically selected for our experiments. For all our images, the parameters are set to values as shown in Table 2. The effect of increasing the '*R*' parameter is that the algorithm yields larger segments. The σ_x and σ_I have to be tuned carefully to obtain meaningful road segmentation. The values of σ_x and σ_I are carefully tuned for each image considered for our experiment. Typical values are set in the range of 10-20% of the range of edge intensity captured in each image under study. A qualitative comparison of Table 1 and Table 2 - an important observation is that the value of *R* is set optimally and as it increases in delta steps we can obtain larger segments of roads from the image. However, to look at an ideal case from our result, consider the 900*900 image where the value of *R* is 1.4 in CPU implementation and 1.69 in GPU. Also, it was observed that *ncut* parameter determines the total number of segments formed in the output. The false roads extracted in figures (4)-(7), are as a result of a lower *ncut* threshold value. To minimize the presence of false roads, an increase in the threshold *ncut* value was attempted, but the observation was that we would sometimes not obtain even the accurate segmentation of the feature i.e roads and the performance of completeness decreases. The completeness and the quality is promising in the case of GPU while the correctness is slightly affected and goes down by 3.1%

(PLEASE INSERT TABLE 3 HERE)

The speed-up of the algorithm can be defined as the CPU time taken divided by the time taken on the GPU for the execution of the algorithm. In the image cases that are considered, the maximum speed up has been obtained for a size of *900 X 900* pixels, where the speed up is 3.16 X. From the results in Table 3, we can infer that the speed-up increases with the size of the image. A plot of the image size (on the X-axis) v/s time in seconds (on the y-axis) is shown in figure 8. Another distinct observation is the steady improvement in the quality factor with image size of GPU based segmentation, as shown in figure 9.

(PLEASE INSERT FIGURE 8 HERE)

(PLEASE INSERT FIGURE 9 HERE)

The results have been recorded as an average of 10 runs of the algorithm for each image (Wendykier and Nagy, 2011). The MATLAB + Jacket based results include the time taken for data transfer to and from the GPU. The timings are shown in Table 3.

5. Conclusions

In this work, we have demonstrated the application of normalized cuts algorithm for road extraction and implemented the normalized cuts algorithm on the GPU using Acclereyes® Jacket software. The best result on the GPU is 3X times faster than the CPU version of the implementation using image size of *900 X 900* pixels. We have tried to reduce the computational complexity in time by implementing a part of the normalized cut algorithm on the GPU. This is the important contribution of this work.

The main advantage of using normalized cut method is that both inter-group and intra-group characteristics of the image are taken into account. Intra-group characteristics are incorporated into the similarity matrix which considers the similarity of pixels in the close neighbourhood. By considering both the local and global aspects of the image scene, normalized cuts works well in extracting roads. The algorithm is able to an extent ignore noise, small surface changes and weak edges. However, in our observation, the parameters in the normalized cut segmentation algorithm have to be tuned carefully and they are image scene dependent. The empirically chosen parameters could pose hindrances while tuning the algorithm for images with different geographical backgrounds.

Although the results are very promising, there is still room for improvements. Hence, one workaround, as a future inclusion would be - to apply appropriate post processing techniques to remove the false road segments that do not belong to the feature of interest.

Acknowledgement

The IKONOS image covering Hobart, Australia was provided by the courtesy of GeoEye. The authors also would like to thank Dr. H. Honne Gowda from KSTA, Bangalore, India, for providing the QuickBird image. Also, the authors would like to thank the anonymous reviewers for their comments that helped in improving this paper.

References

Abramov, A., Kulvicius, T., Worgotter, F. and Dellen, B., 2011. Real-Time Image Segmentation on a GPU. *Facing the Multicore-Challenge,* 131-142.

Acclereyes T., 2011. Jacket: GPU computing without CUDA programming. *www.accelereyes.com/content/collateral/HighLevelGPUComputing*.

Barzohar, M. and Cooper, D. B., 1996. Automatic finding of main roads in aerial images by using geometric-stochastic models and estimation. *Pattern Analysis and Machine Intelligence, IEEE Transactions*, 18(7), 707-721.

Baumgartner, A., Steger, C., Mayer, H., Eckstein, W. and Ebner, H., 1999a. Automatic road extraction based on multi-scale, grouping, and context. *Internat. Arch. Photogrammet. Remote Sensing*, 107–112.

Bong, D.B.L., Lai, K.C. and Joseph A., 2009. Automatic Road Network Recognition and Extraction for Urban Planning. *International Journal of Applied Science, Engineering and Technology*, 5(1), 54-59.

Chung, F. R. K., 1997. Spectral Graph Theory. *ser. Regional Conference Series in Mathematics. Providence, RI: Amer. Math. Soc*.

Geman, D. and Jedynak, B., 1996. An active testing model for tracking roads in satellite images. *Pattern Analysis and Machine Intelligence, IEEE Transactions*, 18(1), 1-14.

Grote, A., Butenuth, M. and Heipke, C., 2007. Road extraction in suburban areas based on normalized cuts. *International Archives of Photogrammetry and Remote Sensing*, 36(3), 51-56.

Grote, A. and Heipke, C., 2008. Road extraction for the update of road databases in suburban areas. *International Archives of Photogrammetry and Remote Sensing*, 37, 563-568.

Gruen, A. and Li, H., 1997. Semi-automatic linear feature extraction by dynamic programming and LSB-snakes. *Photogrammetric Engineering and Remote Sensing*, 63(8), 985-994.

Haralick, R.M., Sternberg, S.R. and Zhuang, X., 1987. Image analysis using mathematical morphology. *IEEE Trans.Pattern Anal. Machine Intell.*, 9, 532-550.

Huang, T., Yang, G. and Tang, G.Y., 1979. A fast two-dimensional median filtering algorithm. *Acoustics, Speech and Signal Processing, IEEE Transactions*, 27(1), 13-18.

Kong, J., Dimitrov, M., Yang, Y., Liyanage, J., Cao, L., Staples, J., Mantor, M. and Zhou, H. 2010. Accelerating MATLAB image processing toolbox functions on GPUs. *GPGPU'10, Pittsburg, PA, USA*, 1-11.

Mayer, H., Hinz, S., Bacher, U. and Baltsavias, E., 2006. A test of automatic road extraction approaches. *International Archives of Photogrammetry, Remote Sensing, and Spatial Information Sciences*, 36(3), 209-214.

Moreland, K. and Angel E., 2003. The FFT on a GPU. In Graphics Hardware, 112–119. http://www.cs.unm.edu/~kmorel/documents/fftgpu/.

Pan, L., Gu, L. and Xu, J., 2008. Implementation of medical image segmentation in cuda, *Proc. Int. Conf. on Technology and Applications in Biomedicine*, 82–85.

Park, S. R. and Kim, T., 2001. Semi-automatic road extraction algorithm from IKONOS images using template matching. *Proc. 22nd Asian Conference on Remote Sensing*, 1209-1213.

Rubin, G., Sager, E.V. and Berger, D.H., 2011, GPU Acceleration of SAR/ISAR Imaging Algorithms, http://www.accelereyes.com/examples/customer_papers_and_talks.

Senthilnath, J., Rajeshwari, M. and Omkar, S.N., 2009. Automatic road extraction using high resolution satellite image based on texture progressive analysis and normalized cut method. *Journal of the Indian Society of Remote Sensing*, 37(3), 351-361.

Senthilnath, J., Vikram Shenoy, H., Ritwik Rajendra, Omkar, S.N., Mani, V., and Diwakar, P.G., 2013. Integration of speckle de-noising and image segmentation using Synthetic Aperture Radar image for flood extent extraction. *Journal of Earth System Science*, 122(3), 559 - 572.

Seo, N. 2006. Normalized Cuts and Image Segmentation. *Technical Report TR-ENEE731 Project.*

Shi, J. and Malik, J., 2000. Normalized cuts and image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions*, 22(8), 888-905.

Valero, S., Chanussot, J., Benediktsson, J.A., Talbot, H. and Waske, B., 2010. Advanced directional mathematical morphology for the detection of the road network in very high resolution remote sensing images. *Pattern Recognition Letters, 31(10), 1120-1127.*

Wendykier, P. and Nagy, J.G., 2011. Image Processing on Modern CPUs and GPUs. *in Technical Report TR-2008-023*, http://www.mathcs.emory.edu/technical-reports/techrep-00148.pdf.

Wiedemman, C., Heipke, C., Mayer, H. and Jamet, O., 1998. Empirical evaluation of automatically extracted road axes. In: Bowyer, K.J., *Jonathon Phillips, P. (Eds.), Empirical Evaluation Methods in Computer Vision. IEEE Computer Society Press, Silver Spring*, MD, pp. 172–

187. Also in: *Proc. 9th Australasian Remote Sensing Photogrammet. Conf., The University of New South Wales, Sydney, Paper No. 239 (CD)*.

Yang, Z., Zhu, Y. and Pu, Y., 2008. Parallel image processing based on CUDA. *In International Conference on Computer Science and Software Engineering*, 3, 198–201.

Youn, J. and Bethel, J.S., 2004. Adaptive snakes for urban road extraction. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Science*, 35(3), 465-470.

Tables

Table 1: Performance evaluation results

Images	Completeness		Correctness		Quality	
	CPU	GPU	CPU	GPU	CPU	GPU
159*642	82.5%	77.5%	81%	75.3%	66.5%	57.5%
327*467	87.2%	84.4%	100%	92.4%	85.2%	76.3%
624*488	88.4%	88.6%	90.3%	89.8%	79.5%	79.3%
900*900	95.3%	99.5%	94.9%	91.8%	90.5%	91.4%

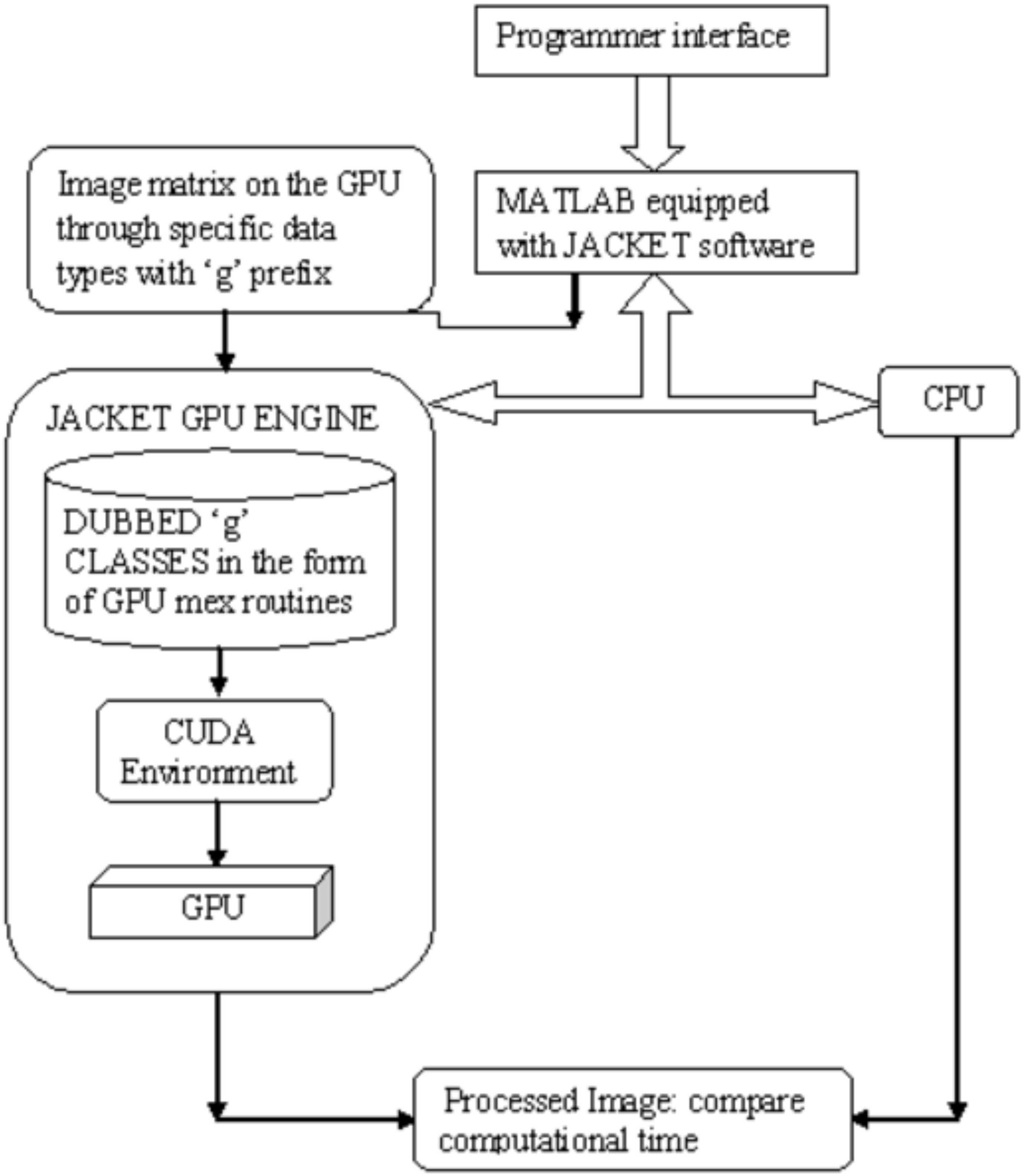
Table 2: Parameters' setting for Normalized cut

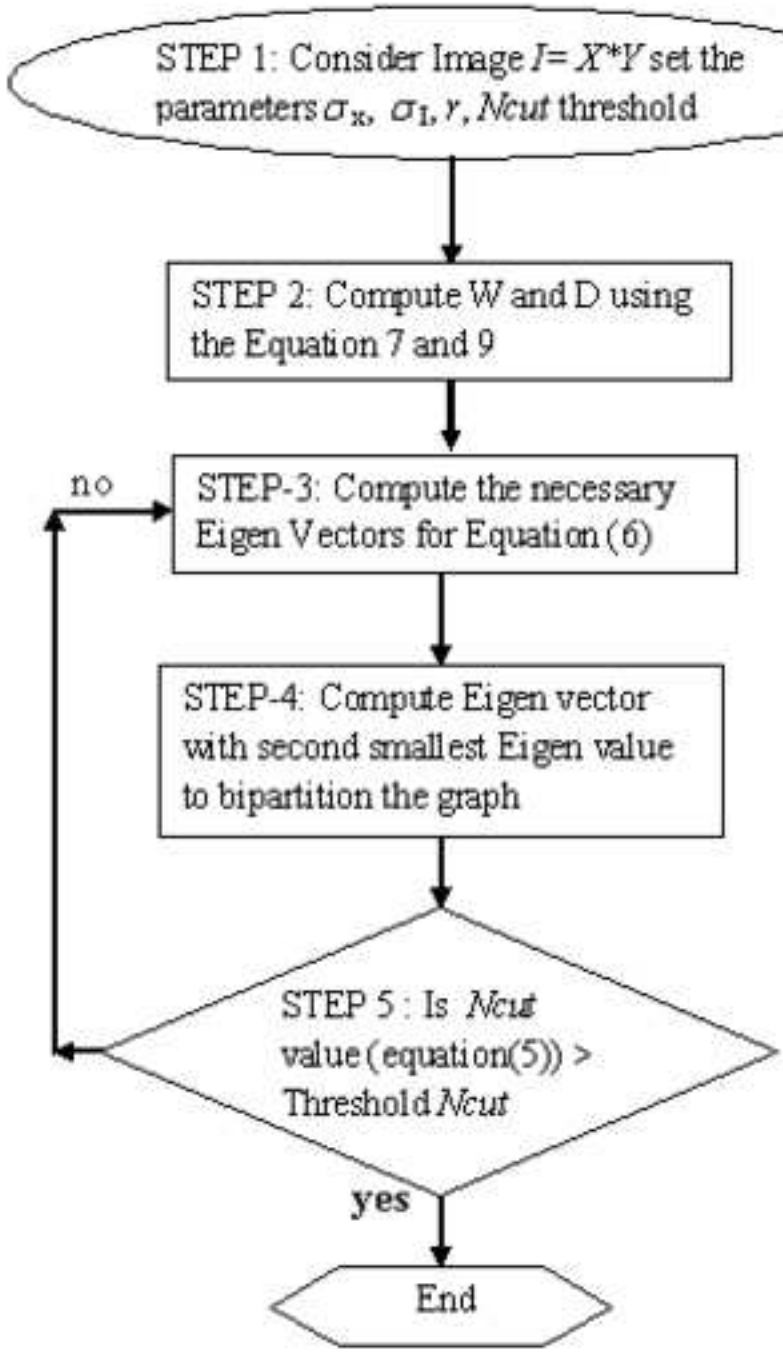
Parameter setting on CPU					Parameter setting on GPU			
Images	σx	σy	R	*Ncut (threshold)*	σx	σy	R	*Ncut (threshold)*
159*642	8	2	1.6	0.01	9.6	4	1.5	0.01
327*467	6	3	1.07	0.89	11	4.8	1.3	0.89
624*488	9	6	1.5	0.04	12	6	1.4	0.04
900*900	9.1	4	1.4	0.04	14	2	1.69	0.04

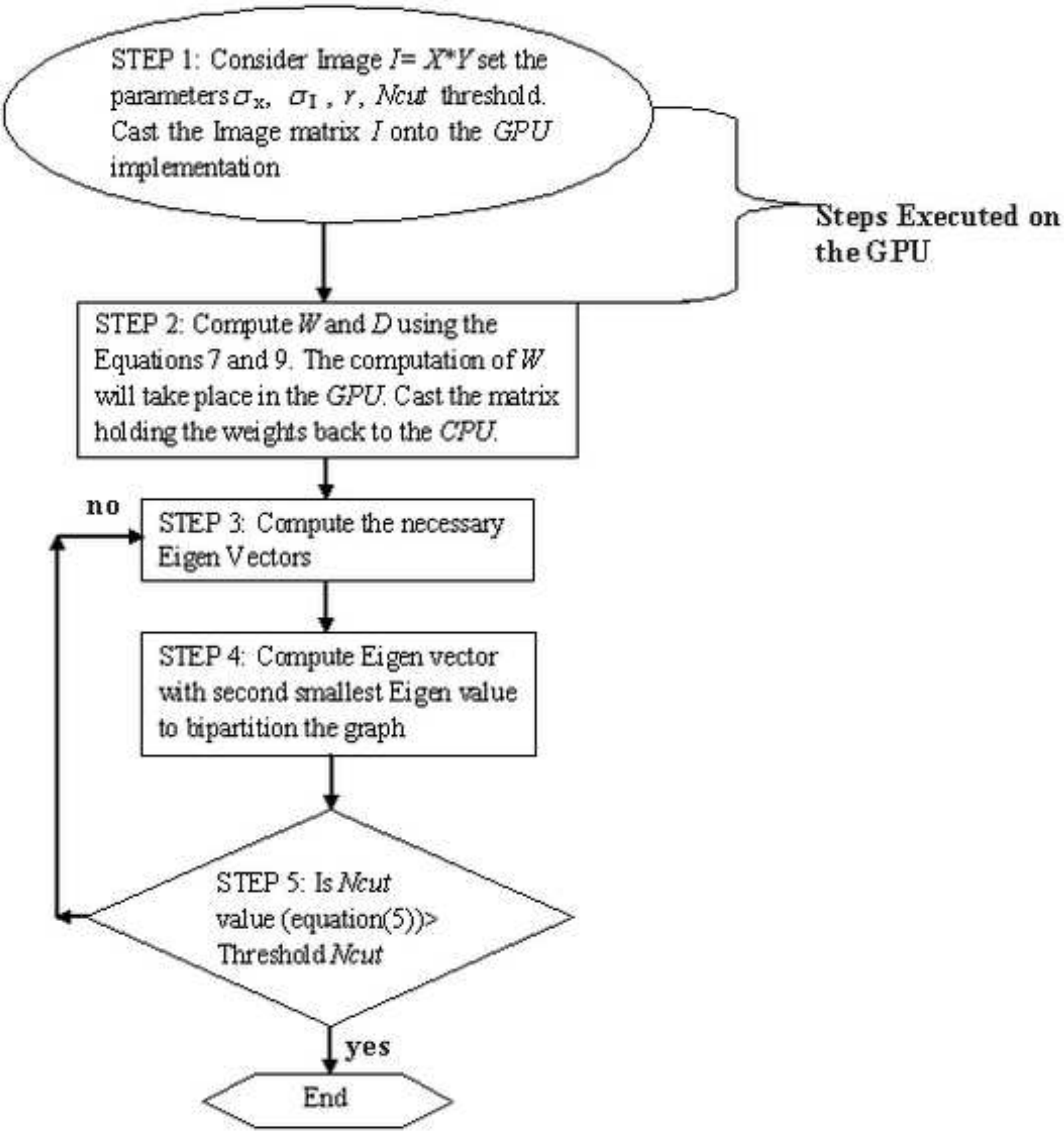
Table 3: Average execution times for Normalized cut-Implemented on MATLAB (CPU version) and on MATLAB + Jacket (GPU version)

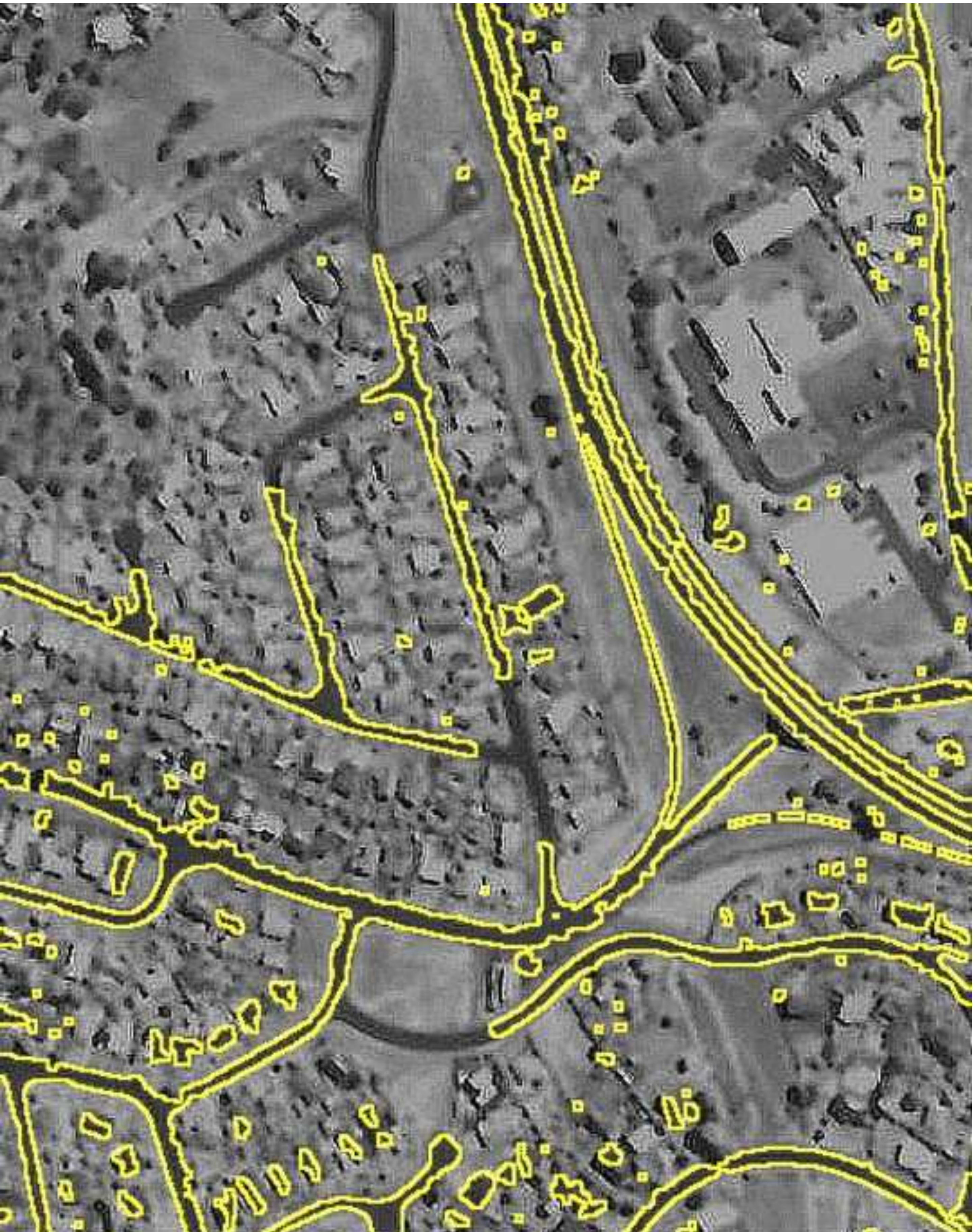
Serial number	Image Size	Cpu time (seconds)	Gpu time (seconds)	Speed-up
1.	159*642	93.83	81.76	1.15 X
2.	327*467	115.84	76.79	1.51 X
3.	624*488	521.72	255.72	2.04X
4.	900*900	3548.1	1122.87	3.16 X

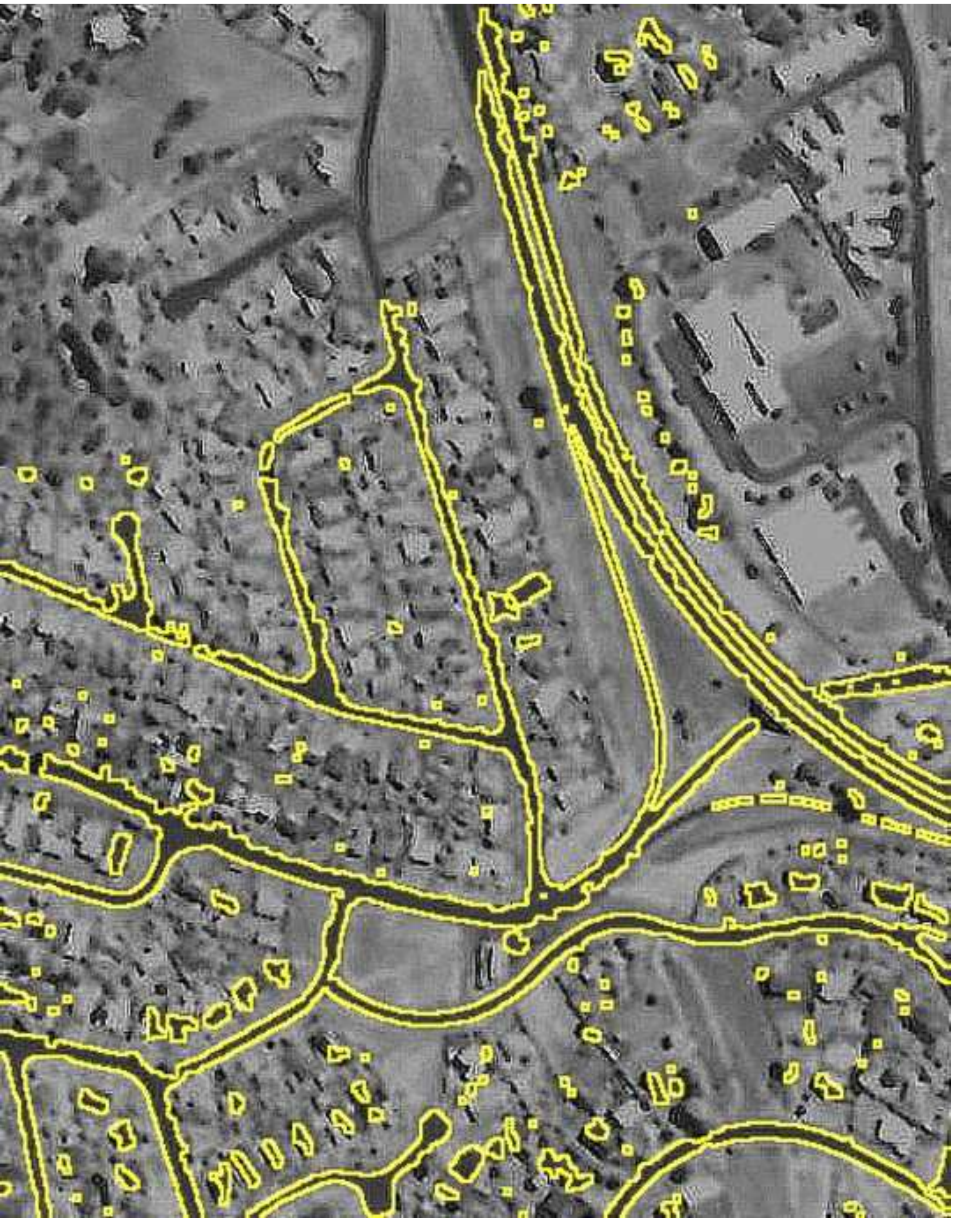
List of figures:

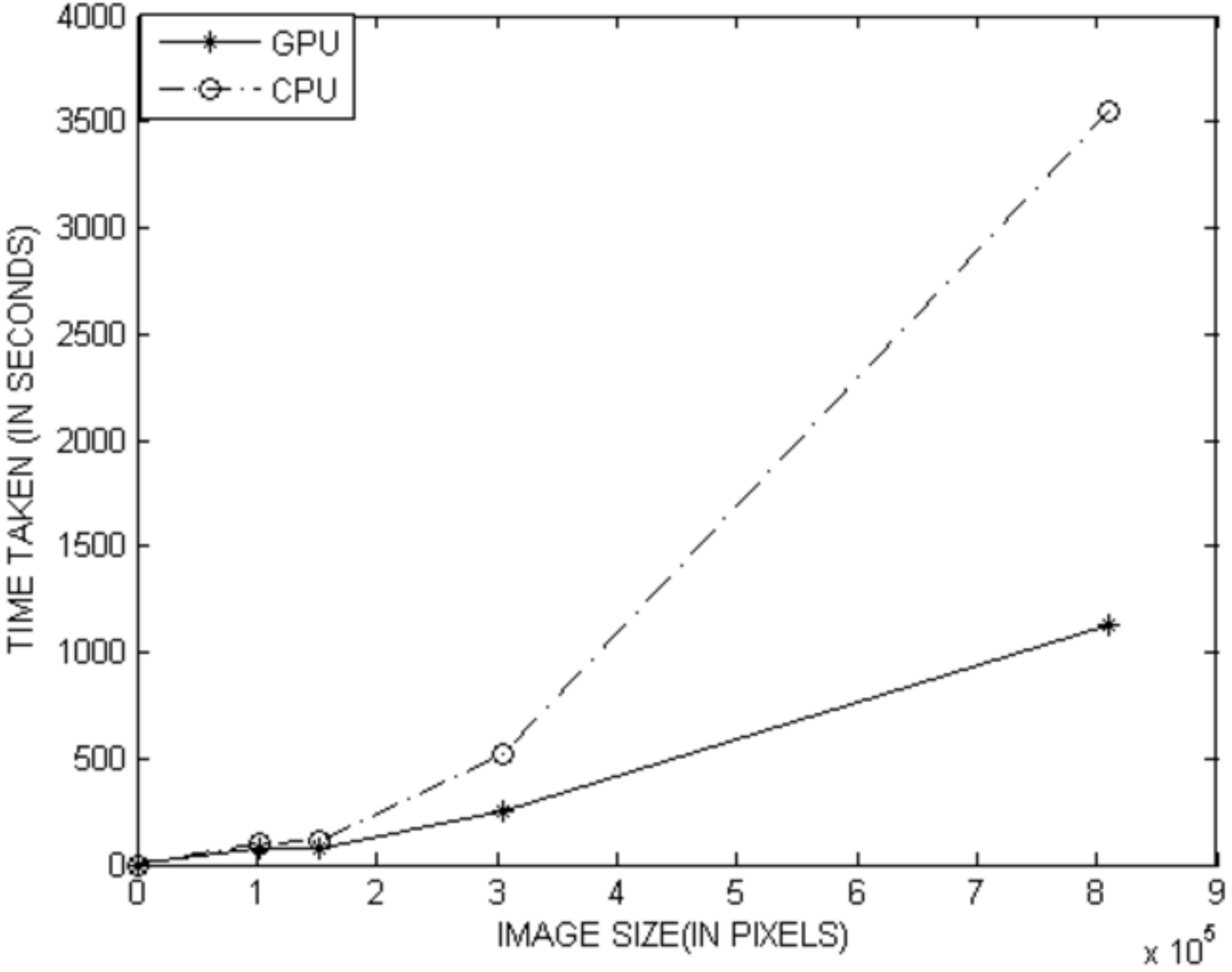

Programmer interface
MATLAB equipped with JACKET software
Image matrix on the GPU through specific data types with 'g' prefix
JACKET GPU ENGINE
DUBBED 'g' CLASSES in the form of GPU mex routines
CUDA Environment
GPU
CPU
Processed Image: compare computational time


STEP 1: Consider Image $I = X*Y$ set the parameters σ_X, σ_I, r, *Ncut* threshold
STEP 2: Compute W and D using the Equation 7 and 9
STEP-3: Compute the necessary Eigen Vectors for Equation (6)
STEP-4: Compute Eigen vector with second smallest Eigen value to bipartition the graph
STEP 5 : Is *Ncut* value (equation(5)) > Threshold *Ncut*
no
yes
End

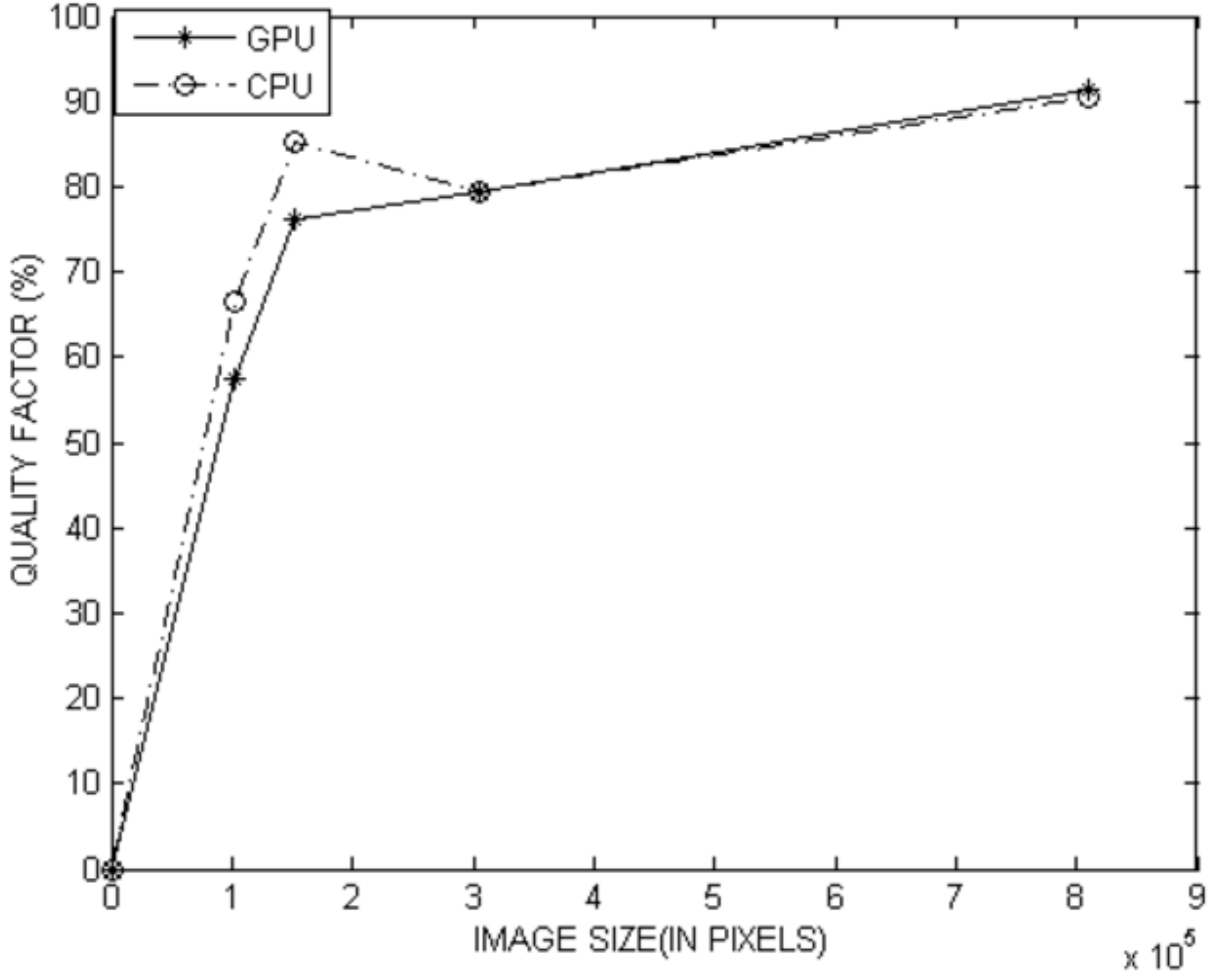

STEP 1: Consider Image $I = X*Y$ set the parameters σ_X, σ_I, r, $Ncut$ threshold. Cast the Image matrix I onto the GPU implementation
Steps Executed on the GPU
STEP 2: Compute W and D using the Equations 7 and 9. The computation of W will take place in the GPU. Cast the matrix holding the weights back to the CPU.
no
STEP 3: Compute the necessary Eigen Vectors
STEP 4: Compute Eigen vector with second smallest Eigen value to bipartition the graph
STEP 5: Is $Ncut$ value (equation(5))> Threshold $Ncut$
yes
End







GPU
CPU
TIME TAKEN (IN SECONDS)
IMAGE SIZE(IN PIXELS)
x 10^5
0
500
1000
1500
2000
2500
3000
3500
4000
0
1
2
3
4
5
6
7
8
9


GPU
CPU
QUALITY FACTOR (%)
IMAGE SIZE(IN PIXELS)
x 10^5
0
10
20
30
40
50
60
70
80
90
100
0
1
2
3
4
5
6
7
8
9