University of Kashan

The Fourth Fuel & Combustion Conference of IRAN

Kashan - IRAN Feb. 2012

Iranian Combustion Institue

FCCI2012-1028

THERMAL ANALYSIS OF COMBUSTION CHAMBER WITH NEW GRAPHIC CARD

Alireza Hajji[*,§], Mahmoud Ashrafizade[**] and Mehdi rahmani[***]

[*] Masters student on energy exchange, Department of Mechanical En. , Isfahan University of technology

** Faculty of department of Mechanical En. , Isfahan University of technology

*** PHD student on energy exchange, Department of Mechanical En., Isfahan University of technology

([§]Correspondent author's E-mail: A.Hajji@me.iut.ac.ir)

ABSTRACT: Simulation of heat transfer in combustion chamber like lots of computational fluid dynamics problems has long run time. To overcome this drawback, parallel processing is one solution. In 2006, new generation of graphic card introduced by NVIDIA company. NVIDIA corporation has manufactured GPU devices that can also be used for the processing of non-graphical data. This game card can be implemented for parallel processing. In order to employ a GPU for general-purpose computations, NVIDIA has introduced a computing architecture called CUDA, which is an extension to C language. In the present work, simulation of heat transfer in combustion chamber accelerated by these GPUs. For solving equations system, General Minimum Residual Method (GMRES) as an iterative linear solver was implemented. It should be noted, this case study had a lot of application like rocket engine. Results show that GPU can simulate 6.7 times faster than CPU. Also our results show excellent agreement with those of available reports in the literature, while demonstrating exciting performance of the GPU simulations.

Keywords: Heat transfer, combustion chamber, GMRES, CUDA, parallel processing, GPU

INTRODUCTION

An accurate estimation of heat transfer for various locations of combustion engine for thermal design is of vital importance. Heat transfer affects the efficiency, performance and emissions, as well as life of the engine components, such as piston, rings and valves. In addition, it is necessary to analyse variation of local heat transfer in order to study thermal stress problems, cycle simulation and to develop a sub model for combustion simulation. Numerous heat transfer measurement and studies have been conducted on combustion chambers during past decade [1-9].

The drawback of these analyses like lots of other CFD cases is running time. This barrier even can influence on simulation. For instance coarse grid must be used for overcoming that directly effect of results. Implementation of parallel processing methods in CFD cases as a solution of run time problem had been proofed previously.

Parallel programming with game cards started from 2006 with introducing new generation of graphic cards that had programmable architectures by NVIDIA Company,.

As a result of increasing demand for real-time and high-definition 3D graphics, the programmable graphic processor unit or GPU has evolved into a parallel, multithreaded, many-core processor with tremendous computational horsepower and a very high memory bandwidth [10]. Recently, the

computational power of GPUs has exceeded that of PC-based CPUs by more than one order of magnitude while being available for a comparable price [11]. While GPUs have originally been developed for fast processing of graphical data, very recently, NVIDIA corporation has manufactured GPU devices that can also be used for the processing of non-graphical data. In order to employ a GPU for general purpose computations, NVIDIA has introduced a computing architecture called CUDA that mentioned in below.
In this paper, conduction heat transfer of combustion chamber simulated with GPU. The present simulation used in lots of engines like rocket engine. In next section, at first problem will be defined clearly, then GPU programming introduced briefly, also governing equation and solution methods will be discussed. At the end of this issue, results and speed up of parallel simulation compared with serial simulation.

PROBLEM DEFINITION

Many combustion chamber designs cool the walls of the combustion chamber by circulating a cooling fluid through the walls. The coolant typically flows down the longitudinal axis of the chamber through channels. The design of the channels can be circular or rectangular, if the combustion chamber is made by brazing or welding tubes. While often a complex combination of layers of different materials, the basic operating principal is that, the walls are constructed from a material with high heat conductivity to allow the heat to be transferred from the hot gas at the wall to the cooling fluid at a rate that keeps the material at a reasonable temperature. As an example, consider a rocket engine design, shown in Figure 1.

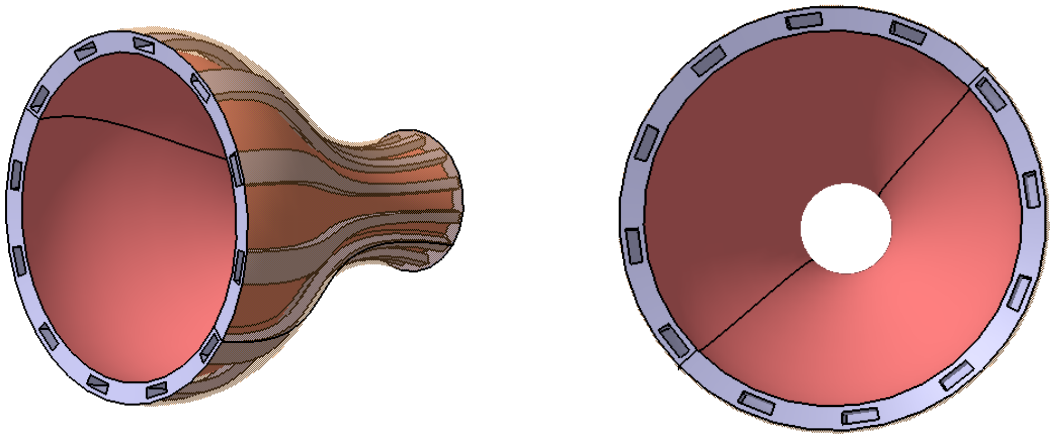


Figure 1. Example Combustion Chamber Application – Rocket Engine

This heat transfer problem will be used to examine the ability of different engineering simulation tools to analyse the temperature profile in the combustion chamber wall, and to try to obtain additional design information such as the rate of heat transferred from the hot gas to the combustion chamber wall, and the rate of heat removed by the coolant. To simplify the problem, we use the symmetry of a cylindrical combustor to consider the 2-D geometry shown in Figure 1. Figure 2 examines the geometry closer.

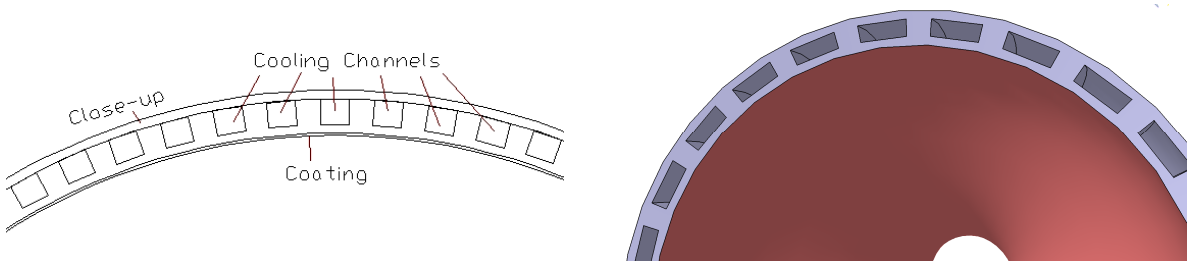


Figure 2. Cooling Channel Geometry

Since the channel geometry repeats itself, we can further reduce the problem to that shown in Figure 3, where "Hot Gas" refers to the inside wall of the combustion chamber.

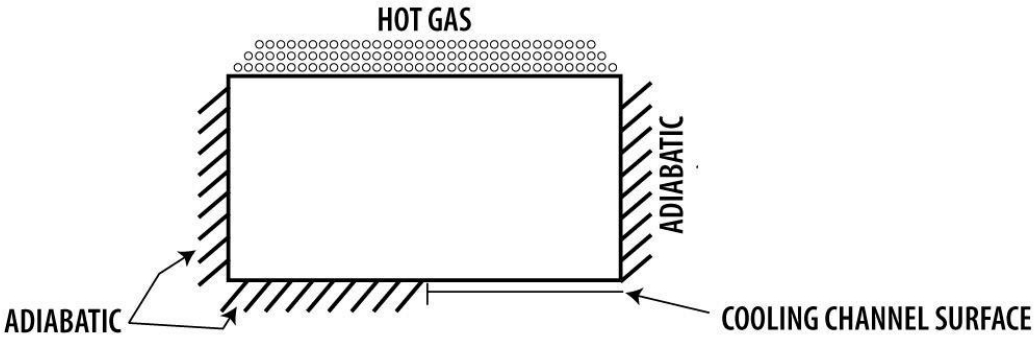


Figure 3. Simplified 2-D Heat Conduction Problem

The problem has now been simplified to 2-D steady state heat transfer in a rectangular block. Boundary conditions can be specified at different levels of complexity. Also following assumptions was implemented:

1-The combustion chamber is made from a uniform material, copper

2-The heat transfer characteristics of a moving hot gas and a moving cold fluid can be neglected, and fixed temperatures at the hot gas and cooling channel boundaries can be utilized as boundary conditions

3-An adiabatic boundary condition on all other boundaries can be assumed to represent the cyclic boundary conditions of the repeated geometry

For this problem, we will consider a slightly larger application than the rocket engine example. We will consider the geometry shown in Figure 4. For the fixed temperature boundary conditions, we will assume a coolant temperature of 60 °C, and a hot gas temperature of 2000 °C.

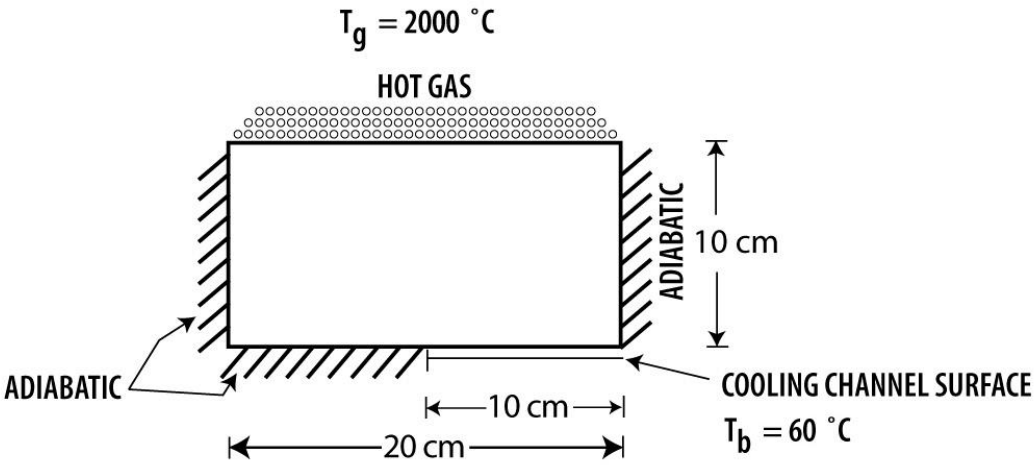


Figure 4. Heat Conduction Problem with Boundary Conditions

PROGRAMMING WITH NVIDIA CUDA

CUDA, an abbreviation for Compute Unified Device Architecture, is a new technology introduced by NVIDIA Corporation. It has been designed for utilizing graphical processing units (GPUs) for non-graphical and general purpose computations. NVIDIA has also provided a "C for CUDA" programming language, which is an extension to the conventional C language and allows the programmer to define new class of functions, called kernels which will be launched on GPU. Whenever a kernel is called, it will be executed N times in parallel by N different CUDA threads, as opposed to only once like regular C functions in serial algorithms [10]. In the following the graphical processing unit is referred to as the "Device" while the CPU is referred to as the "Host". As it can be seen in Figure 5, the GPU is equipped with many cores (a number of multiprocessors) as processing units, and different types of memories.

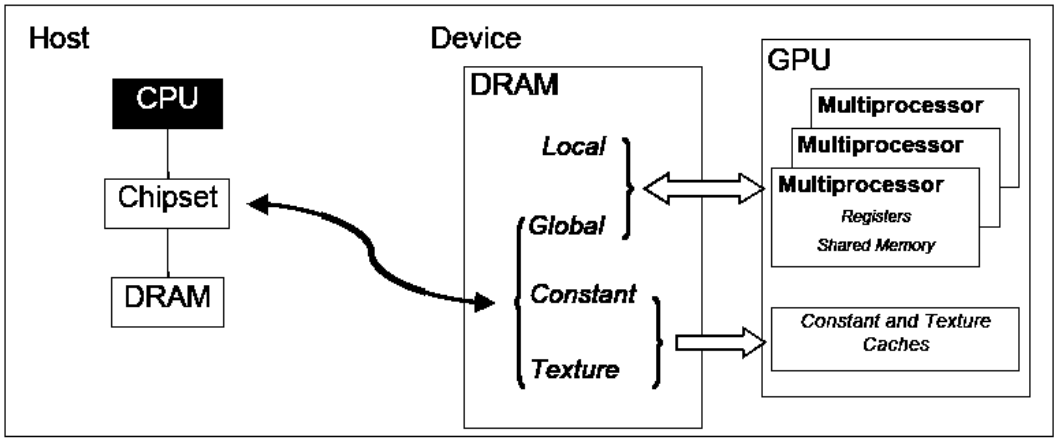


Figure 5. A simple graphical representation of host and device

In a typical CUDA program, data is first transferred from the host to the device. As shown in Figure 6, the host then launches special GPU functions (kernels), which will run the program on the many cores of the device, in parallel, and finally the results are transferred back to the

host. It is important to note that the maximum bandwidth and latency of various memory types of the GPU are quite different. The global device memory (DRAM of the GPU) is large; however, it is much slower than the shared memory which is a limited source of on-chip memory for each multiprocessor. Therefore, in order to achieve the best performance, there should be a careful balance between the use of variables on the global and shared memories which will be discussed later.

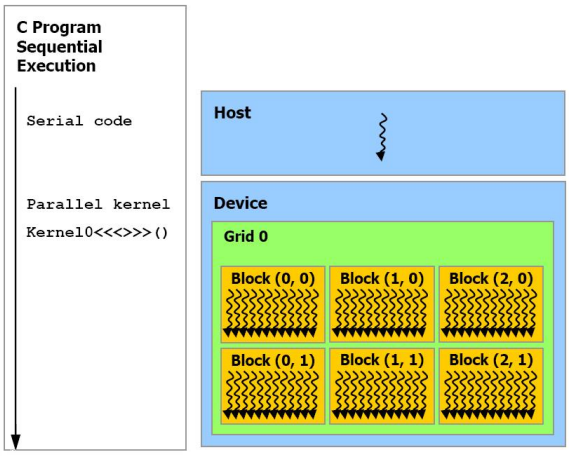


Figure 6. Thread blocks of the GPU

As mentioned before, CUDA creates several threads which will run the kernel commands in parallel. Since the number of threads could be different from the number of available cores, the execution of threads on these available cores is managed by CUDA. Threads are packed in groups which are called “blocks”. A “grid” in CUDA terminology is a batch of thread blocks, and its dimension should be defined in accordance with the size of the problem. All the threads in a block will be passed to one multiprocessor to be processed and these threads can simultaneously access the shared memory while this is not possible for threads running on different blocks. Once a GPU core completes the execution of a thread, it can be utilized by CUDA for the execution of the next thread in line.
For the present work, graphic card witch implemented is GeForce 9800 GTX that its information illustrated in Table 1.

Table 1: identification of GeForce 9800 GTX

parameter	quantity
Double precision cores	-
Single precision cores	112
Memory (GB)	1
Shared Memory (Kbyte)	16
Processor Clock (GHz)	1.5
Mem. Bandwidth (GB/sec)	57.6
Multiprocessors	14

GOVERNING EQUATION AND SOLUTION METHOD

Governing equation in 2D conduction heat transfer at steady state without heat source is:

$$\frac{\partial^2 T}{\partial x^2}+\frac{\partial^2 T}{\partial y^2}=0 \qquad (1)$$

That T represents temperature and x, y show directions. Equation (1) can be solved numerically with various shames like finite difference, finite volume etc. in present work finite difference method will be chosen that based with Taylor series.

After discretisation, group of equation will be made that can be shown in matrix form. In this situation, we have:

$$[A][X]=[B]$$

That $[A]$ represents sparse coefficient matrix, $[X]$ unknown temperature at any node and $[B]$ shows boundary and known temperature:

$$[A]=\begin{bmatrix} a_{11} & a_{12} & & & a_{1n} & & \\ a_{21} & a_{22} & \ddots & & & \ddots & \\ & \ddots & \ddots & \ddots & & & \ddots \\ & & \ddots & \ddots & \ddots & & \\ a_{n1} & & & \ddots & \ddots & \ddots & \\ & \ddots & & & \ddots & \ddots & \ddots \\ & & \ddots & & & \ddots & a_{nn} \end{bmatrix} \quad [X]=\begin{bmatrix} T_1 \\ T_2 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ T_n \end{bmatrix} \quad [B]=\begin{bmatrix} BC_1 \\ BC_2 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ BC_n \end{bmatrix}$$

For solution of this liner system of equation, many methods exist like direct method and iterative shames. One of the most popular of iterative method is general minimum residual than introduced by saad [12]. This method can utilize in both symmetric and asymmetric coefficient matrix. Like lots of other iterative method in this shame, approximated solution is input parameter and correct solution is output. Figure 7 explained this shames how to work:

Input: choose x_0, compute $r_0 = b - Ax_0$ and $v_1 = r_0/\|r_0\|$;
Output: solution of linear system $Ax = b$.
Iterate $j = 1, 2, \cdots, m$,
compute $h_{ij} = (Av_j, v_i)$ for $i = 1, 2, \cdots, j$,
$\tilde{v}_{j+1} = Av_j - \sum_{i=1}^{j} h_{ij} v_i$,
$h_{j+1,j} = \|\tilde{v}_{j+1}\|_2$,
$v_{j+1} = \tilde{v}_{j+1}/h_{j+1,j}$.
End;
Form the solution:
$x_m = x_0 + V_m y_m$, where y_m minimizes $\| \beta e_1 - \tilde{H}_m y \|$ for $y \in \mathbb{R}^m$.
Restart: Compute $r_m = b - Ax_m$, if $\|r_m\|$ is small , then stop,
else , Compute $x_0 = x_m$ and $v_1 = r_m / \| r_m \|$, GoTo Iterate step.

Figure 7. General Minimum Residual method algorithm

By implementing this method in serial and parallel mode, liner system of equation have been solved. In follow results and run time will be discussed.

RESULTS AND DISCOUTION

As mention previously, GPU parallel processing implemented for acceleration of simulation. In this section at first, results compared with each other. Estimate of temperature contour in CPU and GPU illustrated in Figures 8 and 9. As shown in these Figures, near of insulted boundary, temperature contour is normal to boundary and in neighbourhood of isotherm boundary, temperature contour is parallel with boundary.

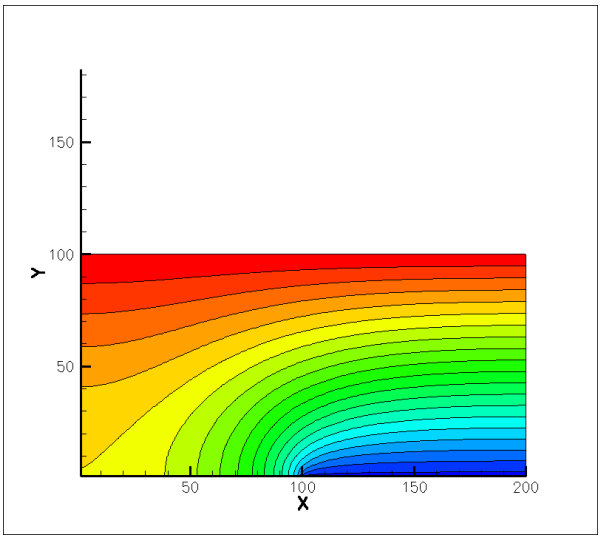


Figure 8. CPU simulation

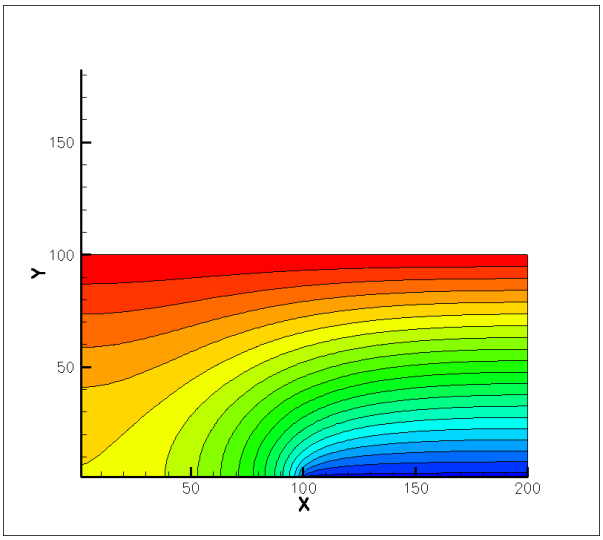
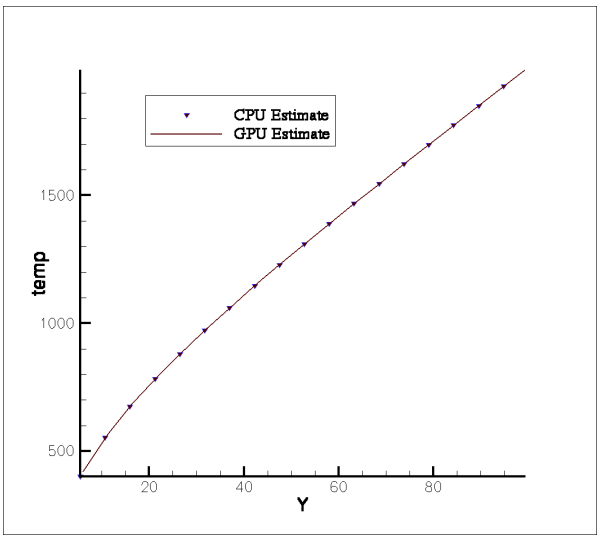


Figure 9. GPU simulation

Other thing that can show similarity between CPU and GPU simulation is temperature distribution that determined in Figure 10.



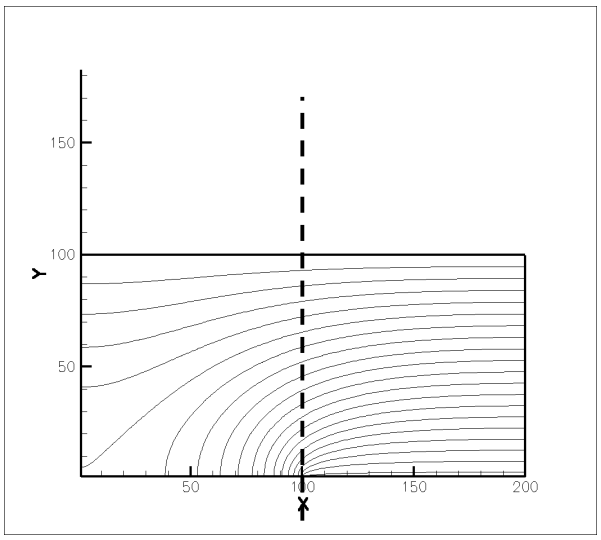


Figure 10. Temperature distribution with CPU and GPU

As results demonstrate CPU and GPU simulation have good approximation for temperature distribution, also their estimation is perfectly same with each other. Now influence of parallel processing will be discussed.
As mention previously, software that implemented for this work is GeForce 9800 GTX. Various simulations have been done for determination of speed up that mention in Table 2.

Table 2: run time of various grid size

Grid Size	CPU Time	GPU Time	Speed Up
100×50	0.37394	0.56491	0.661946
120×60	0.69189	0.6339	1.091481
140×70	1.35079	0.86386	1.563668
160×80	2.47262	1.15982	2.1319
180×90	4.15536	1.77473	2.341404
200×100	5.43817	1.62475	3.347081
220×110	7.27589	2.04369	3.560173
240×120	10.11246	1.81572	5.569394
260×130	13.95587	3.38648	4.121055
280×140	19.76699	2.95255	6.694887
300×150	24.3043	4.96724	4.892918

As shown in this table, in small grid size CPU run time is lower than GPU. This fact shows that in small grid, parallel processing is not efficient. With increasing grid size, effect of GPU parallel processing is visualized. At size of 280×140 best performance is recorded. In this case GPU is about 6.7 time faster than CPU. Also this data shows that due to GPU occupancy and block size, in some grid size GPU runtime is better than others.

CONCLUTION

In this paper, heat transfer of combustion chamber simulated on GPU that used in lot of cases like rocket engine. For solving system of equation, general minimum residual method as an iterative liner solver implemented. Results show that for small grid, effect of parallel processing is not noticeable but with increasing grid size, GPU run time is much lower that CPU run time. Best performance is in 280×140 that GPU is about 6.7X faster than CPU. It must be considered that with improving graphic cards and increasing number of multiprocessor in these cards, this speed up can be greater and better. It is noticeable although CUDA programming is a bit hard, good performance and great run time of GPUs caused this parallel method utilised wider than before.

REFERENCES

1- Gilaber P., Pinchon P. [1988], Measurements and Multidimensional Modeling of Gas-Wall Heat Transfer in a SI Engine, *SAE paper*, No. 880516.
2- Yoo S.J., Kim E.S. [1993], A Study on In-Cylinder Local Heat Transfer Characteristics of a Spark Ignition Engine, , pp. 1–11, *SAE paper*, No. 931981.
3- Angelberger C., Poinsot T., Delhaye B. [1997], Improving Near-Wall Combustion andWall Heat Transfer Modeling in SI Engines Computations. pp. 113–130, *SAE Paper*, No. 972881.

4- Guezennec Y.G., Hamada W. [1999], Tow-Zone Heat Release Analysis of Combustion Data and Calibration of Heat Transfer Correlation in An IC Engine, *SAE Paper*, No. 01-0218.
5- Abd Alla G.H. [2001], Computer Simulation of a Four-Stroke Spark Ignition Engine, pp. 1–11, *SAE paper*, No. 01-0578.
6- Catania A.E., Misul D., Mittica A. [2001], Spessa E., A refined two-zone heat release model for combustion analysis in SI engines, *Proceeding of the Fifth International Symposium on Diagnostic and Modeling of Combustion in Internal Combustion Engines*, Comodia. Nagoya, pp. 290–299.
7- Fergusen C.R. [2003], *Internal Combustion Engines-Applied Thermoscience*, Second ed., John Wiley & Sons, New York, pp. 230–244.
8- Urip E., Liew K.H., Yang S.L., Arici O. [2004], Numerical investigation of heat conduction with unsteady thermal boundary condition for internal combustion engine application, *Proceeding of the ASME International Mechanical Engineering Congress* , November.
9- Jafari A., Hannani S.K. [2006], Effect of fuel and engine operational characteristics on the heat loss from combustion chamber surfaces of SI engines, *Int. Commun. Heat Mass Transf.* 33, 122–134.
10- CUDA Programming Guide V3.2 [2010], Available from: http://www.nvidia.com/object/cuda_3_2_downloads.html. [Accessed 13 November 2011].
11- Tölke J. [2008], Implementation of a lattice Boltzmann kernel using the compute unified device architecture developed by nVIDIA, *Comput. Vis. Sci*.
12- Saad Y. [1996], *Iterative methods for sparse linear systems*. PWS Publishing, New York.