GPU parallel acceleration of transient simulations of open channel and pipe combined flows

GPU parallel acceleration of transient simulations of open channel and pipe combined flows

W W Meng[1], Y G Cheng[1,5], J Y Wu[1,2], Z Y Yang[1],S Shang[3] and F Yang[4]
[1]State Key Laboratory of Water Resources and Hydropower Engineering Science, Wuhan University, Wuhan 430072, P. R. China.
[2]Ministerial Key Lab of Hydraulic Machinery Transients, Ministry of Education, Wuhan University, Wuhan 430072, P. R. China.
[3]Zhangfeng Water Conservancy Management Company LtD, Qinshui 048215, P. R. China.
[4]Construction Management Company for Chushandian Reservoir Project of Henan Province, Zhengzhou 450003, P. R. China.
[5]E-mail: ygcheng@whu.edu.cn

Abstract. Simulating the transient processes in complex water transmission system is time-consuming, and improving computational efficiency by means of parallelization on CPU clusters or even faster GPU platform is demanded. This paper proposes an approach to accelerate the transient simulations of open channel and pipe combined flows on single GPU chip. The Saint-Venant equations for open channel flows is solved by using the method of characteristics (MOC), whose inherent parallelism can be well exploited by GPU implementations in the thread-level parallelism structure of Compute Unified Device Architecture (CUDA). The sub-processes, including open channel computation, pipe flow computation and connecting boundary treatment, are implemented by different kernels. The procedures are first verified by analyzing the parallel computation efficiency of hydraulic transient processes in an open channel. Then the transient processes of a practical engineering project, which involves both open channel flow and pressurized pipe flow, are simulated. The GPU kernels are found to be memory bandwidth bounded, and the proposed single chip GPU parallel can achieve up to hundreds of speedup ratios compared to the sequential counterpart on single CPU chip.
Key words. GPU; Pipe and open channel combined system; Hydraulic transients; Parallel algorithm; Speedup ratio

1. Introduction

The combination of open channel flow and pressurized pipe flow is always adopted in the long-distance water transmission systems due to some unfavorable terrain and hydraulic conditions. Simulating the transient processes in such water transmission systems is really time-consuming if the systems are long or complex. In some situations, such as system optimization that needs a lot of analysis of numerical simulation for many schemes and working conditions and online real-time modelling that requires instant results, computational speed may be the prominent problem. Therefore, efforts on efficient simulations of hydraulic transients are necessary.

Different time steps for open channel flow and pipe flow were adopted to effectively reduce computational amount in references [1-3], but the precision was degraded due to linear interpolations in disposing internal boundaries. Besides, the wave characteristic method(WCM) was proposed by Wood in [4-5] and applied it to transient simulations of water distribution systems. WCM was later

verified in [6-8] to be more efficient than the method of characteristics (MOC) when obtaining the same computational accuracy. The above studies are mainly aimed at reducing computational amount to improve computational efficiency. But improving the computational speed is also important. Meng et al. had proposed an implementation of MOC on GPU architecture to accelerate hydraulic transient simulations of pressurized pipe flows in [9], which achieved speedup ratios up to hundreds compared to a normal CPU core.

In this paper, we propose the implementation of MOC for open channel flows, and apply it to the transient simulations of open channel and pipe combined systems. At first, the GPU implementation of MOC is described in details. Then, the accuracy and efficiency of GPU-MOC for open channel flows are validated. Finally, the transient processes in an open channel and pipe combined system caused by a pump trip are simulated, and the parallel computational efficiencies are analyzed.

2. Implementation of the method of characteristics on GPU architecture

2.1. Characteristic of GPU and CUDA architecture

GPU graphics card is made up of a series of Streaming Multiprocessors (SM), and it allows thousands of threads to execute at the same time [10]. Thread switching is a zero-cost fine-grained execution on GPU, namely, a warp in one thread block can immediately switch to a warp in another thread block that is in ready state, when the warp is waiting for accessing to chip-off memory or synchronizing instructions. In this way, computational latency is hided and high throughput is achieved.

The program is divided into host part and device part in the CUDA architecture. Host part denotes the codes executed on CPU, which is mainly responsible for logical operations, including preparing initial data, allocating memory, transferring data between host and device, etc. Device part denotes codes conducted on GPU, which is also calls kernel function and mainly responsible for floating-point arithmetic operations, updating the parameters of sections in here. Single precision is adopted in this paper, because GPU's single precision arithmetic capability is much higher than double precision.

2.2. Parallel characteristics of MOC for open channel flow simulations

Saint-Venant equations for unsteady open channel flow are transformed into four total differential equations by the method of characteristics. The following formulas are obtained by using arithmetic mean value [11]:

Along positive characteristic line C^+

$$\begin{cases} x_P - x_L = \overline{c^+}_{P,L}\Delta t \\ \overline{\left(Bc^-\right)}_{P,L}\left(H_P - H_L\right) - Q_P + Q_L = \overline{f}_{P,L}\Delta t \end{cases} \tag{1}$$

Along negative characteristic line C^-

$$\begin{cases} x_P - x_R = \overline{c^-}_{P,R}\Delta t \\ \overline{\left(Bc^+\right)}_{P,R}\left(H_P - H_R\right) - Q_P + Q_R = \overline{f}_{P,R}\Delta t \end{cases} \tag{2}$$

where $f = -gA\left(\sin\alpha - \dfrac{n^2|Q|Q}{A^2R^{4/3}}\right)$, α is the angle of channel base and horizontal direction, h is the depth of water, Q is the discharge, A is the cross sectional area, R is the hydraulic radius, B is the top width of prismatic section, n is the Manning roughness factor, Δt is the time step, the "—" above B, c, and f denote average value.

As shown in figure 1, positive and negative characteristic lines C^+ and C^- intersect n time step layer at points L and R, and H_L, Q_L, H_R, Q_R can be attained by quadratic interpolating with the known points D, M and E. Thus, Q_P, H_P, x_R, x_L these four unknowns can be solved by the above four equations.

The solution processes described above reflect the explicit and local features of MOC. The explicit feature refers to that the values at point P at the unknown time level is only related to the values of Point D and E at the previous time level, and having nothing to do with the values of other points at the same time level as P. The local feature refers to that the values of each point are just connected with the values of its adjacent points, and have no connection with the values of other points. These features match the multi-thread parallel characteristics of GPU quite well. Therefore, MOC built on GPU based CUDA architecture platform may fully take advantage of GPU's great ability of floating-point computing to efficiently accelerate the computation.

Note that the GPU implementation MOC for the pressurized pipe is based on our previous work[9].

2.3. Pump boundary

The equation of pump head

$$H_{P_{i,1}} = H_{suc} + H_P - \Delta H_{P_\upsilon} \tag{3}$$

in which H_{suc} is the head of channel at downstream, $\Delta H_{P_\upsilon} = C_\upsilon Q_{P_{i,1}} \left| Q_{P_{i,1}} \right|$ is the head loss in the discharge valve, C_υ is the coefficient of valve head loss. Note that the velocity head in the discharge pipe, which is usually small, is neglected.

The equation of pump torque

$$T_R = 60\gamma H_R / (2\pi N_R \eta_R) \tag{4}$$

in which γ is the specific weight of the liquid, η_R , H_R , Q_R , N_R are the pump efficiency, head, discharge, rotational speed at rated conditions.

By using an average value of β during the time step, obtaining the equation

$$\alpha_P - C_6 \beta_P = \alpha + C_6 \beta \tag{5}$$

in which $C_6 = \dfrac{-15 T_R \Delta t}{\pi I N_R}$, variable I can be replaced by WR^2/g.

Given the characteristic equation for discharge line, for section (i,1)

$$Q_{P_{i,1}} = C_n + C_a H_{P_{i,1}} \tag{6}$$

Since there is no change in the volume stored in the channel and section (i,1), $Q_{P_{i,1}} = Q_P$, Q_P is the flow through the pump at the end of the time step.

With the combination of equations (3)-(6) mentioned above, by eliminating $Q_{P,i}$, $H_{P,i}$ and $\Delta H_{P,\upsilon}$, the resulting equation may be written as

$$Q_P \upsilon_P = C_n + C_a H_{suc} + C_a H_R h_P - C_a C_\upsilon Q_R{}^2 \upsilon_P \left| \upsilon_P \right| \tag{7}$$

According to the head and torque characteristics of pump

$$\frac{h_P}{\alpha_P{}^2 + \upsilon_P{}^2} = a_1 + a_2 \tan^{-1} \frac{\alpha_P}{\upsilon_P} \tag{8}$$

$$\frac{\beta_P}{\alpha_P{}^2 + \upsilon_P{}^2} = a_3 + a_4 \tan^{-1} \frac{\alpha_P}{\upsilon_P} \tag{9}$$

in which α_P, β_P, υ_P, h_P are dimensionless rotational speed, torque, discharge, and head, respectively; a_1, a_2, a_3, a_4 are constants for the straight lines representing the head and torque characteristics.

By eliminating h_P and β_P , obtaining the following equations.

$$F_1 = C_a H_R a_1(\alpha_P{}^2 + \upsilon_P{}^2) + C_a H_R a_2(\alpha_P{}^2 + \upsilon_P{}^2)\tan^{-1}\frac{\alpha_P}{\upsilon_P} - Q_P\upsilon_P - C_a C_\upsilon Q_R{}^2\upsilon_P|\upsilon_P| + C_n + C_a H_{suc} = 0$$

(10)

$$F_2 = \alpha_P - C_6 a_3(\alpha_P{}^2 + \upsilon_P{}^2) - C_6 a_4(\alpha_P{}^2 + \upsilon_P{}^2)\tan^{-1}\frac{\alpha_P}{\upsilon_P} - \alpha - C_6\beta = 0$$

(11)

in which the two unknown variables α_P and υ_P can be solved by the Newton-Raphson method.

As shown in figure 1, the pump boundary that is corresponding to the first thread of Pipeline kernel is disposed by a subprogram. Because the calculation of the pump boundary cannot be conducted in parallel, it can be dealt on host or device. Here, we dispose it on device to avoid too many data traffic between host and device.

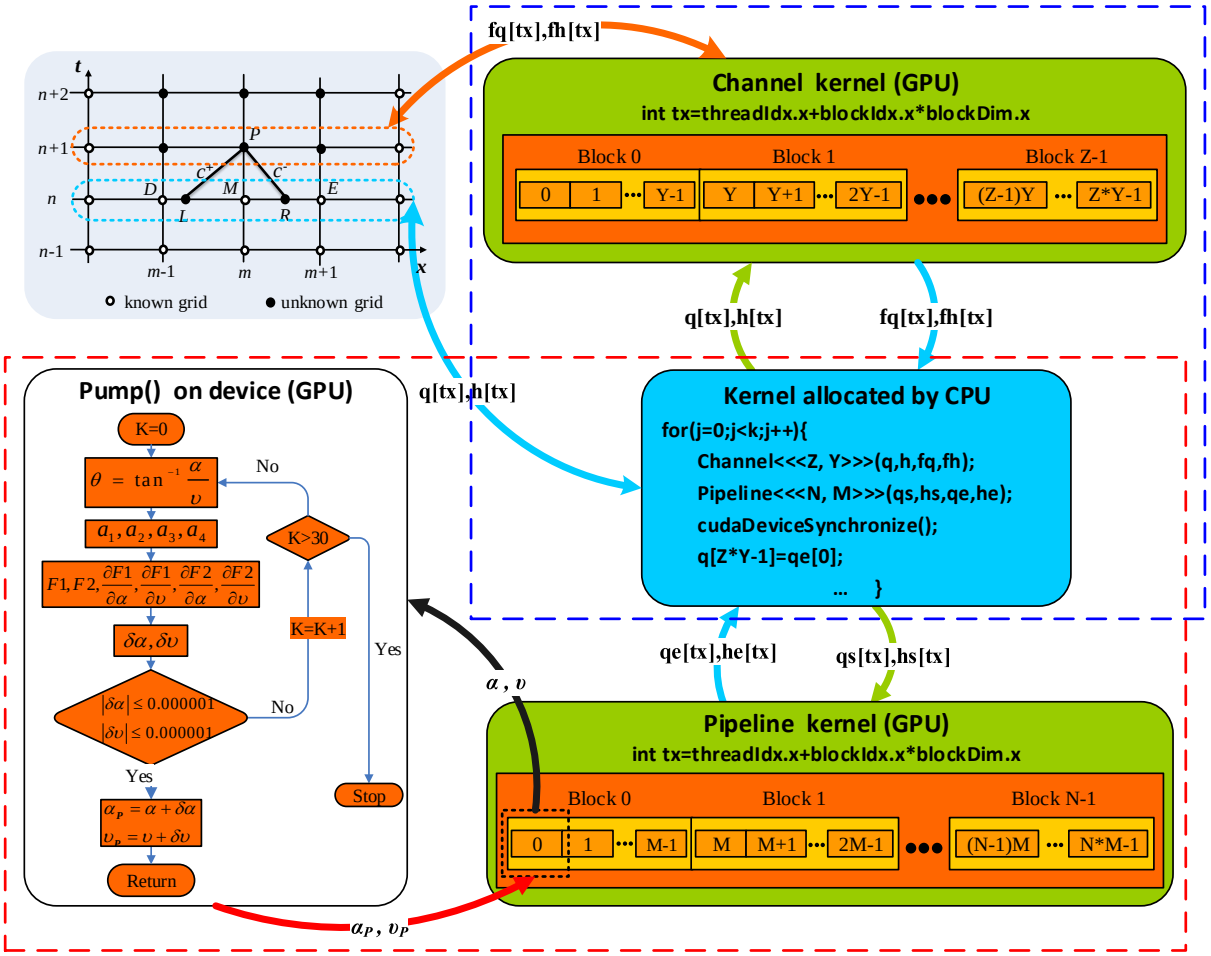

Figure 1. Threading strategy for kernels in GPU-MOC program.

2.4. Implementation scheme for GPU parallel

The GPU-MOC program consists of two parts: device part and host part. Device part mainly conducts floating-point arithmetic operations, including the Channel kernel, the Pipeline kernel, and the Pump function which is a device subprogram. The Channel kernel deals with the transient simulations of open channel flow, and the Pipeline kernel disposes the transient simulations of pressurized pipe flow. Both kernels are conducted on GPU and invoked by CPU. But their parallel computing is independent of each other, and their execution configuration and parameters can be set separately. Pump function is used to calculate υ_P and α_P of pumps, which is invoked by the Pipeline kernel. Host part mainly

disposes logical operations, including declaring variables, allocating memory for one-dimensional discharge and head arrays, transferring data between host memory and device memory, and invoking kernels. As shown in figure 1, procedures within the red dotted frame are responsible for calculating the transient processes of pressurized pipe flow, and those in the blue dotted frame are responsible for calculating the transient processes of open channel flow. A synchronous function is used to make sure that all calculations of one time step are completed. Then, all output data are transmitted to the host, and the discharge of pump is transferred to the end section of the open channel.

As shown in figure 1, the grid for the Pipeline kernel and the Channel kernel are grid(N, 1, 1) and the numbers of the first dimension is N, the numbers of the second and third dimension are both 1. The threads in the block are in the form of block(M,1,1) and block(Y,1,1), respectively, in which M and Y represent the numbers of thread for the first dimension, the numbers of the second and third dimension are both 1.Threads are organized as 1D linear form, which is quite fit into the line structure of the system. Each thread corresponds to a calculation section, and it has a unique thread ID that is accessible within the kernel through the built-in threadIdx variable. The thread ID is determined by its location in the thread grid and can be used to locate the corresponding calculation section. The Pipeline kernel and the Channel kernel can be programmed according to their own MOC equations. The CPU-MOC program is compiled by standard C and the GPU-MOC program is compiled by the combination of standard C and CUDA C (an extended language of standard C).

3. Validation

3.1. Simulation for simple open channel

We simulate the transient processes of a simple reservoir-open channel-sluice gate system [12], as shown in figure 2. The trapezoidal channel is 1500m long, 10m wide, side slopes 0.5 horizontal to 1 vertical, 0.016 Manning roughness, and 0.0002 bottom slope, and the system deliveries a flow of $40\text{m}^3/\text{s}$ in the initial steady state. The transients are caused by linearly shutting down the sluice gate in 2 minutes. Time step sizes are set to $\Delta t = 1\text{s}$ and $\Delta t = 0.01\text{s}$, with the corresponding spatial step sizes of $\Delta x = 10\text{m}$ and $\Delta x = 0.1\text{m}$, respectively.

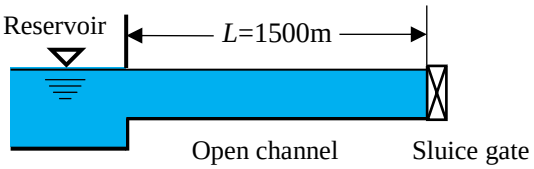

Figure 2. Notation for reservoir-open channel-sluice gate system.

As shown in figure 3, the history curves of water depth at downstream are obtained by GPU program, and water depth at downstream for different time step sizes show similar trends, which have good agreement with the law of hydraulic phenomena. The differences between different time step sizes are caused by iteration errors, which are inevitable in the calculations. It is can be seen that the head h changes between 2.5m and 3.8m, then water velocity $V = 40/(10h + h^2/2) < 2\text{m/s}$, and wave speed $c = \sqrt{gh}$ <6.5m/s, Courant number $C_n = \left[\max(|V| + \sqrt{gh})\Delta t\right]/\Delta x < 1$ .Therefore, Courant condition is satisfied in both calculation conditions.

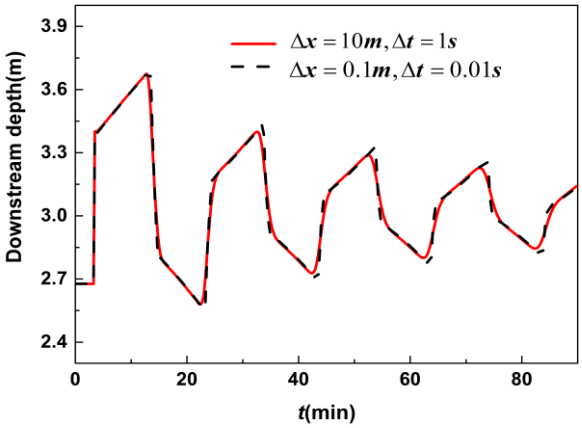

Figure 3. History curves of water depth at downstream.

3.2. Parallel performance analysis

The project mentioned in Section 3.1 is scaled successively, the section numbers are 1.5E3, 1.5E4, 1.5E5, 1.5E6, and the corresponding spatial step sizes are 10m, 1m, 0.1m, 0.01m, the corresponding time-step intervals are 1s, 0.1s, 0.01s, 0.0001s. The calculation steps are 1000. In the GPU program, three scenarios of block size 64, 128, 256 are included, and the computing time of the kernels are obtained by NVIDIA visual profiler analysis software. All the calculations are conducted on the computer that is configured with one CPU processor (Inter Xeon e5-2620 v3 2.4G 12 cores) and one graphics card (NVIDIA Geforce GTX TITAN X), and the testing platform is Windows 10 64-bit operating system. In addition, all the simulations are run for at least several times to avoid the effect of temperature variation and possible frequency down-stepping of GPU. Each simulation starts when the GPU is cooled down to room temperature.

Table 1. Numbers of MOC update per second (MMUPS) and speedup ratios of GPU over CPU.

Section numbers	GPU-MOC			CPU-MOC
	64	128	256	(MMUPS)
N=1.5E3 (Δx =1m)	8.57(13.86)	8.57(13.86)	8.51(13.76)	0.62
N=1.5E4 (Δx =01m)	87.17(127.43)	86.04(125.78)	85.81(125.45)	0.68
N=1.5E5 (Δx =0.01m)	206.4(318.72)	206.62 (319.06)	177.69(274.39)	0.65
N=1.5E6(Δx =0.001m)	241.49(363.64)	243.12(366.09)	215.48(324.47)	0.66

Note: the data before and inside the brackets are the numbers of MOC update per second and speedup ratios, respectively.

Table 1 summarizes the numbers of MOC update per second (MMUPS) of single GPU and CPU, along with the speedup ratios of GPU over CPU in different section numbers. The data in the first column denote the section numbers, data in the row under the header of GPU-MOC represent the thread block size, data before the brackets are the numbers of MOC update per second, and data in brackets are the speedup ratios of GPU-MOC updating section numbers to CPU-MOC updating section numbers. It is shown that the overall performance is improved as the section number and block size become finer, and the speedup ratios increase from 13.76 to 366.09 when the section numbers increase from 1.5E3 to 1.5E6. The reasons of this trend may be ascribed to the characteristics of TITAN X graphics card and the instruction loading way of CUDA. Threads are executed in the form of warps, once warps in one thread block have to wait for accessing to chip-off memory or synchronizing instructions, warps in another thread block can be disposed instead by the SM. Thus, high computational efficiency can be achieved once there are enough active warps in the chip.

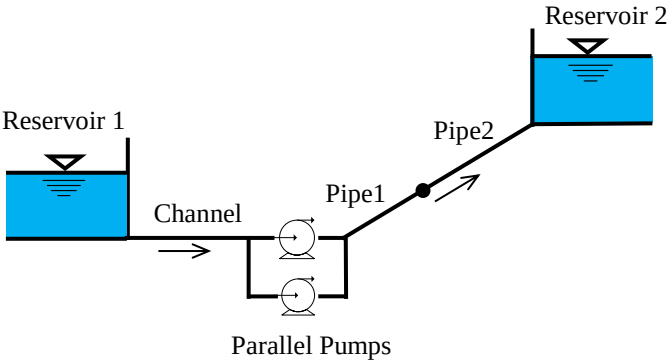

Figure 4. Notation for open channel and pipe combined system.

4. Hydraulic transient simulation of open channel and pipe combined flow

Programs of GPU-MOC and CPU-MOC are developed to simulate the transient of a water supply project, which pumping water from an upstream reservoir to the downstream one. The pumping system is an example from a book written by Chaudhry [13].We modified it by adding a long trapezoidal channel between the upstream reservoir and the pump station, just as shown in figure 4, and the parameters of the project are shown in table 2. The total discharge of pumps is equal to that of the channel at downstream. The characteristic curves of the pumps are given by Chaudhry [13].The discharge of the two parallel pumps is lumped together, and the suction lines being short are not include in the analysis.

Table 2. Computation conditions.

Pipeline	Pipe1	Pipe2	Pump	Trapezoidal channel
Length(m)	450	550	Q_R=0.25m^3/s	Water level of reservoir=1.5m
Diameter(m)	0.75	0.75	H_R=60m	Length=10km, wide=1m
Wave speed(m/s)	1000	1000	N_R=1100rpm	Bottom slope=0.0002
Discharge (m^3/s)	0.5	0.5	WR^2 =16.85kg.m^2	Side slopes 0.5 horizontal to 1 vertical
Friction factor	0.01	0.012	η_R =0.84	Manning roughness=0.012

Here, pump trip is considered to be the cause of the transients, and there is no valve action. Time step sizes of $\Delta t = 0.01\text{s}$ and $\Delta t = 0.0001\text{s}$, with the corresponding spatial step sizes of $\Delta x = 10\text{m}$ and $\Delta x = 0.1m$, respectively, are simulated. In both conditions, the Courant numbers $C_n = 1000\Delta t/\Delta x = 1$. Figure 5 shows the history curves of pump head, discharge, speed and head of open channel at downstream, which are the results of GPU program. The results of different time step sizes are nearly the same, which means the proposed method has a good computing stability.

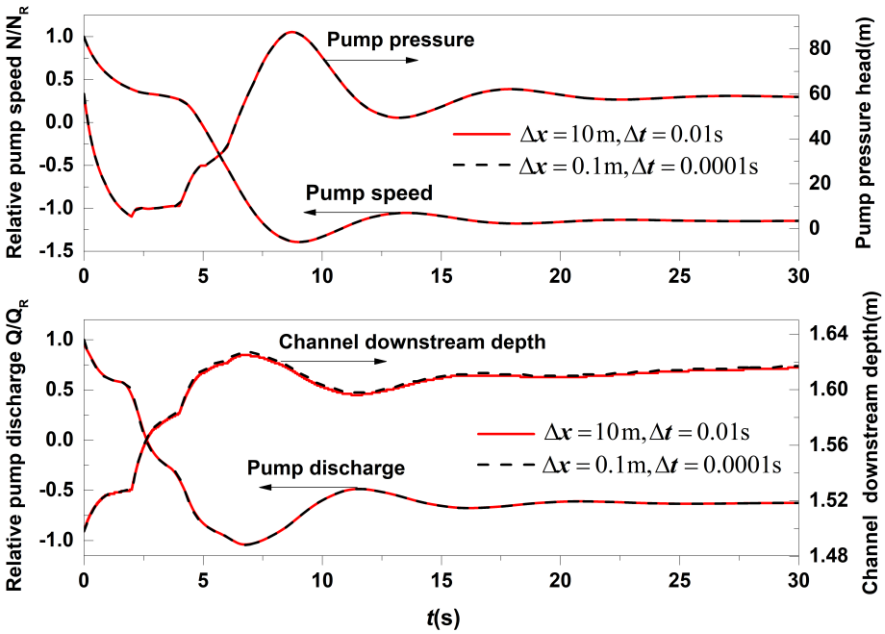

Figure 5. History curves for the pipelines in the transients of pump failure.

The instant that power is lost, the reactive torque of the liquid on the impeller causes it to reduce its rotational speed, which in turn reduces pressure head and discharge. The discharge of pump reverses at time of 2.6s, and then a short time later the rotational speed reverses and it starts to run away backward at time of 4.9s. As it spins up to a high reverse speed at time of 20.5s, the flow is stable at 0.313m^3/s, and it causes a high resistance to the flow, which develops a high head at the pump. The maximum pressure head is 87.5m reached at time of 8.7s. The minimum pressure head is 5.5m reached at time of 2s. The discharge of channel at downstream that delivers to the pump is reduced rapidly at the start, leading to water depth rises quickly. Once pump reach its reverse steady-state conditions, water depth of the channel at downstream increases uniformly and slowly.

5. Parallel performance analysis

The project mentioned in Section 4 is scaled successively, the section numbers are 1.0E3, 1.0E4, 1.0E5, 1.0E6, and the corresponding spatial step sizes are 10m, 1m, 0.1m, 0.01m, respectively. The calculation steps are 1000. In the GPU-MOC program, four scenarios of block size 64, 128, 256, 512, are included, and the consumed time of the kernels are obtained by NVIDIA visual profiler analysis software.

All computations are carried out using single precision on the NVIDIA GeForce GTX Titan X graphics card, which can deliver 7×10^{12} single precision floating-point operations per second (7TFLOP/s) and has a peak memory bandwidth of about 336.5 GB/s. The FLOPS/bandwidth ratio is 20.8, which means the device needs to execute at least 21 floating-point operations for per memory access [14]. If the ratio of floating-point operations to memory accesses exceeds 20.8, the program is computing bounded, and if the ratio of floating-point operations to memory accesses is lower than 20.8, the program is bandwidth bounded. As shown in table 3, the ratio of floating-point operations to memory accesses for Pipeline kernel is 1.78 and for Channel kernel is 4.37, which are much lower than 20.8. Hence, the program is bandwidth bounded.

Table 3. Floating point operations (FP Ops), memory operations(Memory Ops) for GPU kernels.

GPU Kernel	FP Ops	Memory Ops	Ratio
Channel	158	35	4.37
Pipeline	16	9	1.78

Table 4 summarizes the numbers of MOC update per second (MMUPS) of single GPU and CPU, along with the speedup ratios of GPU-MOC over CPU-MOC in different section numbers. It is shown that the speedup ratios increase from 5.44 to 498.98 as the section numbers increase from 1.0E3 to 1.0E6. Additionally, GPU-MOC program can achieve its best performance when the block size is 128 threads. This is because, for each SM of TITAN X, the full block capacity is 8 and the full thread capacity is 2048, the largest register number is 65536. Each thread requires 70 registers in this program, thus a block size of 64 takes up registers 64*8*70=35840<65536, and the achieved efficiency(occupancy) for the device is just 512/2048=25%. When the block size is 128, the block numbers on the SM are 65536/70/128=7.3<8, which means register resources restrict the number of blocks, the achieved occupancy is 7*128/2048=43.75%. Similarly, the occupancy for block size of 256 and 512 are 37.5% and 25%, respectively. This means too many data spill from the registers weaken the floating point arithmetic ability of the card. Therefore, it is essential to allocate computing resources reasonably to achieve a better performance.

Table 4. Numbers of MOC update per second (MMUPS) and speedup ratios of GPU over CPU.

Section numbers	GPU-MOC				CPU-MOC (MMUPS)
	64	128	256	512	
N=1.0E3 (Δx =10m)	3.65 (5.46)	3.71 (5.54)	3.67 (5.49)	3.64 (5.44)	0.46
N=1.0E4 (Δx =1.0m)	37.08 (56.98)	37.25 (57.24)	36.87 (56.65)	36.94 (56.76)	0.45
N=1.0E5 (Δx =0.1m)	164.4 (268.11)	164.68 (268.58)	163.62 (266.85)	117.68 (191.93)	0.47
N=1.0E6(Δx =0.01m)	257.96 (496.92)	259.03 (498.98)	230.64 (444.29)	157.61 (303.61)	0.46

Note: the data before and inside the brackets are the numbers of MOC update per second (MMUPS) and speedup ratios, respectively.

6. Conclusions

A GPU implementation of the method of characteristics for the open channel and pressurized pipe combined flows based on CUDA architecture is presented in this paper. The MOC method for open channel flow and for pressurized pipe flow are implemented by different kernels, and the proposed kernels are all bandwidth bounded. Transient processes of pump trip for open channel and pipe combined system are simulated with single precision by single GPU and CPU. The results show that the capability of GPU-MOC program increases with the scale of the problem, and the obtained GPU speedup ratios are up to hundreds compared to a normal CPU core, which means the GPU-MOC program can be used to efficiently compute large-scale practical problems. The proposed method has great values for practical applications because the algorithm is simple, high-efficiency, easy to program, and convenient to popularize. Our future work will focus on further optimization of the present algorithm in more complex transient simulations and on multiple GPUs.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (NSFC) (grant Nos. 11172219 and 51579187) and the Natural Science Foundation of Hubei Province (Grant No. 2018CFA010).

Nomenclature

H = head (m);
Q = discharge (m^3/s);

α = angle of channel base and horizontal direction;
A = cross-section area (m^2);
c^+, c^- = wave speeds in positive and negative characteristic line, respectively (m/s);
B = top width of prismatic section (m);
R = hydraulic radius (m);
n= Manning roughness factor;
f = friction coefficient;
H_L, Q_L = head, discharge of point L;
H_P = head of point P;
Q_P = discharge of point P, flow through the pump at the end of the time step;
x_L , x_R , x_P = position of points L,R and P, respectively;
H_R = head of point R, head of pump at rated conditions;
Q_R = discharge of point R, discharge of pump at rated conditions;
g = gravitational acceleration (N/Kg);
Δt = time-step interval (s);
H_{suc} = head of channel at downstream (m);
ΔH_{P_υ} = head loss in the discharge valve (m);
C_υ = coefficient of valve head loss;
$Q_{P_{i,1}}$, $H_{P_{i,1}}$ = discharge and head at section i;
γ = specific weight of the liquid;
I, WR^2 =combined polar moment of inertia of the pump;
η_R , N_R , T_R = pump efficiency, rotational speed(rpm) and torque(N.m) at rated conditions;
C_6 , C_n , C_a = coefficients;
α_P, β_P, υ_P, h_P = dimensionless rotational speed, torque, discharge, and head, respectively;
a_1, a_2, a_3, a_4 = constants for the straight lines representing the head and torque characteristic;
α , β = dimensionless rotational speed and torque, respectively;

References

[1] Wang W, Lian J and Wang J 2003 Numerical simulations of hydraulic in open channel-surge pool-pressure conduit system *Chinese J. Hydraulic Eng.* **8** 16-20
[2] Trikha A K 1977 Variable Time Steps for Simulating Transient Liquid Flow by Method of Characteristics *J. Fluids Eng.* **99** 259-61
[3] Lian J, Wang J, Wan W and Wang Y 2003 Calculations of hydraulic transient for complex water supply systems by using characteristic method with variable time steps *Chinese Water Resources and Hydropower Eng.* **34** 12-3
[4] Wood D J, Dorsch R G and Lightner C 1966 Wave-plan analysis of unsteady flow in closed conduits *J. Hydraulics Div.* **92** 83-110
[5] Wood D J, Lingireddy S, Boulos P F, Karney B W and McPherson D L 2005 Numerical methods for modeling transient flow in distribution systems *J. Am. Water Works Ass.* **97** 104-15
[6] Boulos P F, Karney B W, Wood D J and Lingireddy S 2005 Hydraulic transient guidelines for protecting water distribution systems *J. Am. Water Works Ass.* **97** 111-24
[7] Ramalingam D, Lingireddy S and Wood D J 2009 Using the WCM for transient modeling of water distribution networks *J. Am. Water Works Ass.* **101** 75-89
[8] Jung B S, Boulos P F and Wood D J 2007 Pitfalls of water distribution model skeletonization for surge analysis *J. Am. Water Works Ass.* **99** 87-98

[9] Meng W, Cheng Y, Wu J, Yang Z, Zhu Y and Shang S GPU acceleration of the method of characteristics for simulations of hydraulic transients (Submitted)
[10] Kirk D B and Wen-Mei W H 2010 *Programming massively parallel processors: a hands-on approach*(Morgan kaufmann)
[11] Abbott M B 1966 *An introduction to the method of characteristics* (American Elsevier New York)
[12] Wylie E B, Streeter V L and Suo L 1993 *Fluid transients in systems* (Englewood Cliffs, NJ: Prentice Hall)
[13] Chaudhry M H 1979 *Applied hydraulic transients* (New York: Van Nostrand Reinhold)
[14] Wu J, Cheng Y, Zhou W, Zhang C and Diao W 2016 GPU acceleration of FSI simulations by the immersed boundary-lattice Boltzmann coupling scheme *Comput. & Math. Appl.*