ME 759 FINAL PROJECT: SIMULATION OF LASER POWDER BED FUSION (L-PBF) USING CUDA

Yaqi Zhang
Department of Mechanical Engineering
UW-Madison
Madison, WI 53705
zhang623@wisc.edu

Xin Liu
Department of Mechanical Engineering
UW-Madison
Madison, WI 53705
xliu632@wisc.edu

Abstract

As one type of laser powder bed fusion processes (L-PBF), selective laser sintering (SLS) is an 3D printing technology that uses laser to sinter the powdered materials, aiming at the powders located at the laser scanning path can bind together to create the designed geometry. In this project, we implemented sequential version of simulating one layer in selective laser sintering (SLS) process with discrete element method, based on paper [1]. Each particle is modeled as discrete element mechanically and thermally interacting with each others.

Then we implemented the parallel version of simulation using CUDA. The simulation is divided into two independent steps. The first step gives a deposition process, in which particles fall onto the build platform and form a static layer. In this step, firstly each block has one thread to tackle one particle's interaction with environment, secondly each thread in the block calculate the interaction with every other particles, and finally the reduction operation gives the sum of all the interaction. The second step gives the thermal process, in which a laser beam scans over the static deposited layer and temperature variation is observed. In this step, every thread can update the heat exchange and temperature change of every particle simultaneously. Time results show that when number of particles is small, sequential version runs faster than parallel version due to low occupancy of GPU. However when computation scale is large, parallel version can achieve very good speedup ratio compared with sequential version.

Contents

1 Introduction

Selective laser sintering (SLS) is an 3D printing technology that uses laser to sinter the powdered materials, aiming at the powders located at the laser scanning path can bind together to create the designed geometry.Figure 1 gives a schematic of a typical SLS machine. The process works in a layer-by-layer style, in each layer, firstly powder particles are deposited on the substrate by roller or wiper blade. The size of powder particle usually ranges from 10-100 μm, and the layer usually is of 20-200 μm thickness. The particle powders might come from ball milling, therefore its size usually could be obtained by some random distributions. After one layer is deposited, a laser scans over the layer according to the desired shape, high energy of the laser can sinter or melt the particles to fuse and bond them. After finish sintering of the layer, the substrate moves and new particles are deposited on the previously sintered layer, then the SLS machine repeat this process, eventually a part is fabricated in this layer-by-layer style.

It is worthy to note that in 3D printing it is possible to print defected part with inappropriate process parameters, and many process parameters need to be decided, in SLS there are parameters like laser intensity, powder size, laser scanning speed and so on. Optimization of these parameters with simulation before printing can be very useful, so it is necessary to investigate how to simulate the process effectively and accurately.

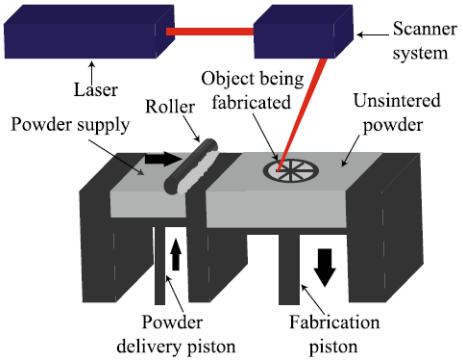

Fig. 1 Schematic of a typical SLS/SLM set-up

Figure 1: Schematic of SLS machine

2 Modeling approach

This project conduct simulation of the behaviour of one-layer particle in SLS. It is assumed that the particles are dropped from a small height to the substrate, the particles might collide with substrate or with each others. The particles eventually become static due to the environmental drag. Then the laser beam, a moving heat source scans the static particles and there is the thermal simulation. In other words, there is two steps in this project, the first step is to simulate the deposition process, essentially it gives a random particle layer, and the second step is to have the thermal simulation.

2.1 Particle dynamics

Essentially, this process is treated as a multibody dynamics problem, where each particle is modeled as a rigid sphere and they would interact with each others and the boundary walls. All the particles are assumed to be dropped to the substrate from a small height of 0.3 mm, then they will fall down to the substrate due to gravity, after interact with each others/walls/environmental dragging, these particles eventually settle them to a layer.

From Newton's law, assume there are N particles in total, the i-th particle has

$$m_i \ddot{\boldsymbol{x}}_i = \boldsymbol{F}_i^{\text{total}} = \boldsymbol{F}_i^{\text{con}} + \boldsymbol{F}_i^{\text{fric}} + \boldsymbol{F}_i^{\text{env}} + \boldsymbol{F}_i^{\text{grav}} \tag{1}$$

Here m_i is the mass for i-th particle, the total force $\boldsymbol{F}_i^{\text{total}}$ is composed of contact force $\boldsymbol{F}_i^{\text{con}}$, friction force $\boldsymbol{F}_i^{\text{fric}}$ environmental dragging $\boldsymbol{F}_i^{\text{env}}$ and gravity $\boldsymbol{F}_i^{\text{grav}}$.

The contact force can be further categorized into contacting with other particles and contacting with walls. For contacting with other particles, suppose i-th (with radius R_i) particles contact with j-th particles (with radius R_j),

according to Hertz contact theory, we have

$$\boldsymbol{F}_{ij}^{\text{con}} = -\frac{4}{3}\sqrt{R^*}E^*\delta_{ij}^{3/2}\boldsymbol{n}_{ij} + d\dot{\delta_{ij}}^{3/2}\boldsymbol{n}_{ij}, \;\; ||\boldsymbol{x}_i - \boldsymbol{x}_j|| < R_i + R_j \tag{2}$$

$$\boldsymbol{F}_i^{\text{con}} = \sum_{j=1, j\neq i}^{N} \boldsymbol{F}_{ij}^{\text{con}} \tag{3}$$

$$d = 2\zeta\sqrt{2E^* m^* \sqrt{R^*}|\delta_{iw}|^{1/4}} \tag{4}$$

Here ζ is the damping coefficient. m^*, R^* and E^* are effective mass, radius and Young's modulus, which are given below

$$m^* = \frac{m_i m_j}{m_i + m_j} \tag{5}$$

$$R^* = \frac{R_i R_j}{R_i + R_j} \tag{6}$$

$$E^* = \frac{E_i E_j}{E_i(1-\nu_i^2) + E_j(1-\nu_j^2)} \tag{7}$$

Here nu is the Possion ratio, and in equation (2) the δ_{ij} and $\dot{\delta_{ij}}$ are the overlap between particles i and j respectively,

$$\delta_{ij} = |||\boldsymbol{x}_i - \boldsymbol{x}_j|| - (R_i + R_j)| \tag{8}$$

$$\dot{\boldsymbol{\delta}}_{ij} = (\boldsymbol{v}_j - \boldsymbol{v}_i)\cdot \boldsymbol{n}_{ij} \tag{9}$$

Here $\boldsymbol{v}$ is the particle velocity and $\boldsymbol{n}_{ij}$ is the unit vector between i and j,

$$\boldsymbol{n}_{ij} = \frac{\boldsymbol{x}_j - \boldsymbol{x}_i}{||\boldsymbol{x}_j - \boldsymbol{x}_i||} \tag{10}$$

For contacting with walls, similar formulas are used, just replace $R_j \to \infty$, also $\boldsymbol{v}_j = 0$ and $\boldsymbol{n}_{ij}$ is the unit vector which orthogonal to walls.

Then it comes to friction. Each contact force associates with a friction force, which is along with the tangent direction and propotional to the magnitude of contact force, so

$$\boldsymbol{F}_{ij}^{\text{fric}} = \mu_d ||\boldsymbol{F}_{ij}^{\text{con}}|| \boldsymbol{t}_{ij} \tag{11}$$

$$\boldsymbol{F}_i^{\text{fric}} = \sum_{j=1}^{N} \boldsymbol{F}_{ij}^{\text{fric}} \tag{12}$$

Here μ_d is friction coefficient and the tangent direction $\boldsymbol{t}_{ij}$ is given by,

$$\boldsymbol{t}_{ij} = \frac{\boldsymbol{v}_{tj} - \boldsymbol{v}_{ti}}{||\boldsymbol{v}_{tj} - \boldsymbol{v}_{ti}||} \tag{13}$$

Here the tangent velocity $\boldsymbol{v}_{ti}$ is given by $\boldsymbol{v}_{ti} = \boldsymbol{v}_i - (\boldsymbol{v}_i \cdot \boldsymbol{n}_{ij})\boldsymbol{n}_{ij}$. The environmental dragging and gravity is given by

$$\boldsymbol{F}_{ij}^{\text{env}} = -6\pi\mu_f R_i \boldsymbol{v}_i \tag{14}$$

$$\boldsymbol{F}_{ij}^{\text{env}} = -m_i g \hat{\boldsymbol{z}} \tag{15}$$

Here μ_f is the viscosity coefficient.

Put these force terms into equation (1), we have the total force is a function of $\boldsymbol{x}_i$ and $\dot{\boldsymbol{x}}_i, i = 1, ..., N$ as well as other parameters $\boldsymbol{\alpha}$, denote $\boldsymbol{r} = [\boldsymbol{x}, \dot{\boldsymbol{x}}]$ and $f(\boldsymbol{r}, \alpha)$, so we can transform the equation into first-order system

$$\dot{\boldsymbol{r}} = \boldsymbol{f}(\boldsymbol{r}, \alpha) \tag{16}$$

Note the right side is independent with time so it is a ordinary differential system. Also, since the particles are dropped from the given initial conditions, $\boldsymbol{r}(0)$ is given so it is a initial value problem.

Numerically, both explicit and implicit finite difference method are used in simulation. The adopted explicit method is 4-th order Runge-Kutta method,

$$
\begin{aligned}
\boldsymbol{Y}_1 &= \boldsymbol{r}(t), \\
\boldsymbol{Y}_2 &= \boldsymbol{Y}_1 + \frac{1}{2}\Delta t \boldsymbol{f}(\boldsymbol{Y}_1), \\
\boldsymbol{Y}_3 &= \boldsymbol{Y}_1 + \frac{1}{2}\Delta t \boldsymbol{f}(\boldsymbol{Y}_2), \\
\boldsymbol{Y}_4 &= \boldsymbol{Y}_1 + \frac{1}{2}\Delta t \boldsymbol{f}(\boldsymbol{Y}_3), \\
\boldsymbol{r}(t+\Delta t) &= \boldsymbol{Y}_1 + \frac{\Delta t}{6}\left[\boldsymbol{f}(\boldsymbol{Y}_1) + 2\boldsymbol{f}(\boldsymbol{Y}_2) + 2\boldsymbol{f}(\boldsymbol{Y}_3) + \boldsymbol{f}(\boldsymbol{Y}_4)\right]
\end{aligned} \tag{17}
$$

2.2 Thermal simulation

Assuming temperature field is uniform throughout each particle (consistent with the low Biot numbers resulting from the small particle sizes), a lumped capacitance model could be built. The thermal governing equation in integral form is

$$
\int_{\partial\omega} \mathbf{Q}\cdot\mathbf{n}dA + \int_{\omega} H dV = \int_{\omega} \rho c \dot{T} dV, \tag{18}
$$

where Q represents heat flux, H is a source term to account for laser heat input, ρ is density, c is the constant pressure specific heat capacity of the material, T is temperature, and ω represents the domain of interest.

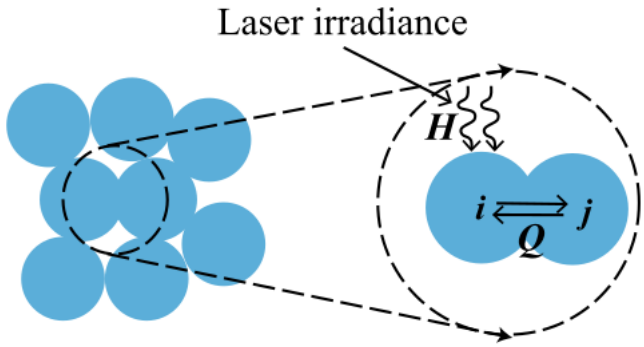

Figure 2: Heat transfer to an individual particle

Discretizing Eq. 18 over each particle (see Fig. 2), we obtain

$$
Q_i + H_i = m_i c_i \dot{T}_i \tag{19}
$$

Heat transfer term in Eq. 19 could be broke down into conductive, convective and radiant parts as follows

$$
\begin{aligned}
Q_i &= Q_i^{cond} + Q_i^{conv} + Q_i^{rad} \\
&= \sum_{j=1}^{N_c} K \frac{T_j - T_i}{||x_j - x_i||} A_{ij} + h_{conv}(T_i - T_{env})A_i + \epsilon\sigma_{SB}(T_i^4 - T_{env}^4)A_i.
\end{aligned} \tag{20}
$$

In Eq. 20, K is thermal conductivity of the material (assuming it is isotropic), A_{ij} is the contact area between particles i and j, h_{conv} is the convective heat transfer coefficient, ϵ is the material emissivity, σ_{SB} is the Stefan-Boltzmann constant ($\sigma_{SB} = 5.67\times10^{-8} W/m^2\cdot K^4$), and $A_i = \pi R_i^2$ is the projection of the particle's surface area facing the surrounding environment. Particles will exchange heat with the environment, which is at a temperature of T_{env}. For two contacting particles the contact area (see Fig. 3) is given by

$$
A_{ij} = \pi h^2 = \pi\left(R_i sin\left(arccos\left(\frac{R_j^2 - R_i^2 - d^2}{-2dR_i}\right)\right)\right)^2, \tag{21}
$$

where $d = ||x_j - x_i||$.

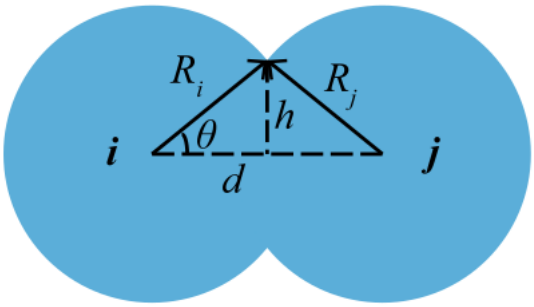

Figure 3: Contact area of two intersecting particles

Assuming a Gaussian laser is used, the laser heating term, H_i in Eq. 19 is given as

$$H_i = \alpha I(r,z) A_i, \tag{22}$$

where α is the absorptivity of the material at the wavelength of the laser ($0 \leq \alpha \leq 1$), $I(r,z)$ is the laser intensity (W/m^2) as a function of radial distance (r) and depth (z) from the beam center, and $A_i = \pi R_i^2$ is the projected area of the particle receiving direct, normal radiation from the laser.

The laser beam is assumed to be Gaussian (as is often the case in the SLS process) so that the intensity decreases exponentially as radial distance from the beam's center increases. A Beer-Lambert type model for the lase penetration is used, where the intensity is assumed to decrease exponentially as a function of penetration depth. Thus the intensity can be expressed as

$$I(r,z) = I_0 e^{-\beta z} e^{-2r^2/\omega^2}, \tag{23}$$

where $I_0 = \frac{2 \times Power}{\pi \omega^2}$ is the peak intensity, ω is the beam spot size measured to where the intensity falls to $1/e^2$ of the peak intensity, and β is the optical extinction coefficient [2]. Note that the laser energy will penetrate much deeper in a powder bed than a solid material due to the multiple reflections that take place off the powder particles. The optical extinction coefficient for a metal powder bed is modeled according to the theory of Gusarov et al [3, 4]. In this theory the extinction coefficient for a homogeneous absorbing powder is given as a function of the powder bed porosity, γ, and the particle diameter, D. Penetration of the laser to the underlying substrate is also taken into account during this derivation. This model has been shown to be agree with ray tracing simulations in the limit of a deep powder bed [3] and is given as

$$\beta = \frac{3(1-\gamma)}{2\gamma D} \tag{24}$$

To account for phase change as the material heats up and subsequently melts, the apparent heat capacity method is used, similar to the method outlined by Bonocina et al [5]. and more recently by Muhieddine et al. [6]. In this technique, the energy needed for phase change to happen is taken into account by computationally raising the specific heat of the material in a small range (ΔT) around the melting or vaporization temperature. The algorithm is outlined as follows and can be graphically seen in Fig. 4.

$$\alpha(x) = \begin{cases} c_{solid}, & T < T_m - \frac{\Delta T}{2} \\ \frac{c_{solid}+c_{liquid}}{2} + \frac{L_{m/s}}{\Delta T}, & T_m - \frac{\Delta T}{2} \leq T \leq T_m + \frac{\Delta T}{2} \\ c_{liquid}, & T_m + \frac{\Delta T}{2} < T < T_v - \frac{\Delta T}{2} \\ \frac{c_{liquid}+c_{gas}}{2} + \frac{L_{v/c}}{\Delta T}, & T_v - \frac{\Delta T}{2} \leq T \leq T_v + \frac{\Delta T}{2} \\ c_{gas}, & T > T_v + \frac{\Delta T}{2} \end{cases} \tag{25}$$

where T_m is the melting temperature, T_v is the vaporization (or boiling) temperature, $L_{m/s}$ is latent heat of melting or solidification, and $L_{v/c}$ is the latent heat of vaporization or condensation.

A standard **Forward Euler** time-marching scheme was used for thermal simulation. And a time step size of $\Delta t = 5 \times 10^{-9}\ s$ is used when solving the thermal problem as the laser is scanned over the powder bed.

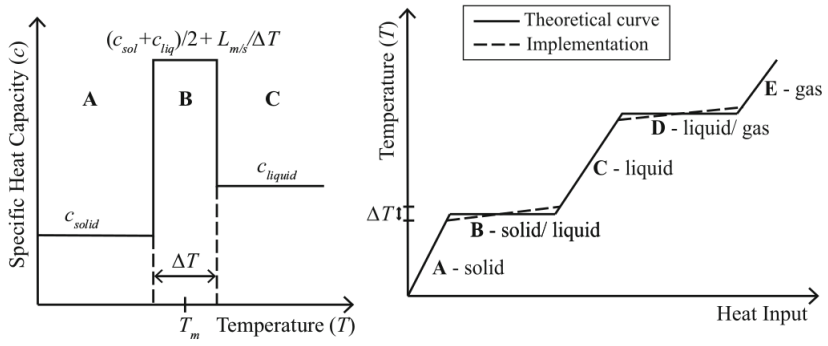

Figure 4: Illustration of how heat capacity is updated *left* and ensuing phase change diagram *right*

3 Numerical experiments

The simulation is composed of two parts, dynamic simulation and thermal simulation. The dimension of build platform is $L_x = 1mm \times L_y = 0.5mm$. Also the simulation was carried out using stainless steel (316L SS) powder particles. The material properties and manufacturing parameters are listed in the below table (see Table 1). In the original paper, some values (e.g. density, thermal conductivity, specific heat) varies with temperatures. In order to keep it simple, we use fixed value in our current implementation. It is not hard to extend it to support this feature.

All the codes are uploaded to below public Github repository.

`https://github.com/zhangyaqi1989/Simulation-of-LPBF-using-CUDA.git`

Table 1: Material properties and simulation parameters used [7]

Parameter	Symbol	value
Young's modulus	E	$193\,GPa$
Poisson's ratio	ν	0.26
Damping parameter	ζ	0.1
Density	ρ	$7952\,kg/m^3$
Dynamic friction coefficient	μ_d	0.1
Viscosity	μ_f	$2.1 \times 10^{-3} Pas$
Heat transfer coefficient	h	$40\,W/m^2K$
Thermal conductivity	K	$17.0\,W/mK$
Specific heat (solid)	c_{solid}	$452\,J/kgK$
Specific heat (liquid)	c_{liquid}	$815\,J/kgK$
Melting temperature	T_m	$1700\,K$
Boiling temperature	T_v	$3130\,K$
Latent heat of melting	L_m	$2.99 \times 10^5\,J/kg$
Latent heat of vaporization	L_v	$6.09 \times 10^6\,J/kg$
Material emissivity	ϵ	0.33
Material absorptivity	α	0.33
Powder bed porosity	γ	0.55
Laser power	$Power$	$100\,W$
Laser scan speed	v_{laser}	$2.0\,m/s$
Laser spot size	ω	$54\,\mu m$
Preheat temperature	T_0	$363\,K$
Environment temperature	T_{env}	$363\,K$

3.1 Dynamic simulation

There is a natural parallelism in the dynamic simulation. At one specific time, with given velocity and positions of every particles, the acceleration of each particle can be calculated independently. Also there is just several hundreds particles in this project, so it is possible to have one thread to tackle with one particle. In collision

detection, for each particle one need to look every other particle if they collide with this particle. Again, since there is just several hundreds particles, in this step it is possible to have one block to deal with one particle, in the block, every thread is responsible for detection of the potential collision with every other particle. Also since the contact force is the sum of all contact interactions with other particles, in each block there is a reduction operation to obtain the sum. The schematic of CUDA parallelism is shown in figure (5) and (6).

It is noteworthy that there is no parallelism from the perspective of temporal sequence. In each time step, accelerations and velocities of particles are updated based on previous step, so it is impossible to parallelize this temporal sequence. Also, to achieve numeric stability, the time step should be set a very small number, in the dynamic part it is 5×10^{-8} sec. To guarantee numeric stability, the maximal interaction force between particles are also bounded. Considering the total simulation time in dynamic part is order of 0.1 sec, the loop number would be order of 10^7, this is one of the critical reason that the code need very long time to run.

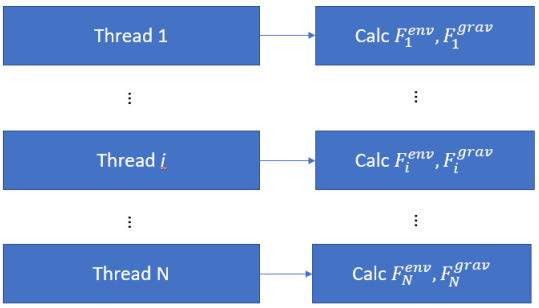

Figure 5: Schematic of CUDA parallelism for dynamic part in step 1: each thread calculate its environmental drag and gravity

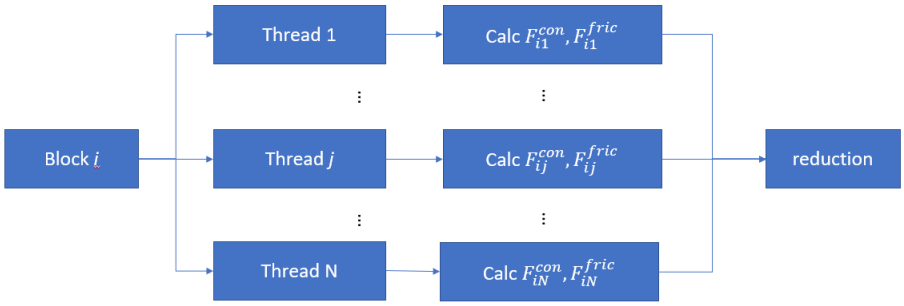

Figure 6: Schematic of CUDA parallelism for dynamic part in step 2: each block correlate to a particle, each of its thread calculate a potential collision pair with other particles

The simulation shows that particles achieve equilibrium at time $t = 0.3$s (initially dropped from the height of 0.3 mm), figure (7), (8), and (9) shows the dynamic process.

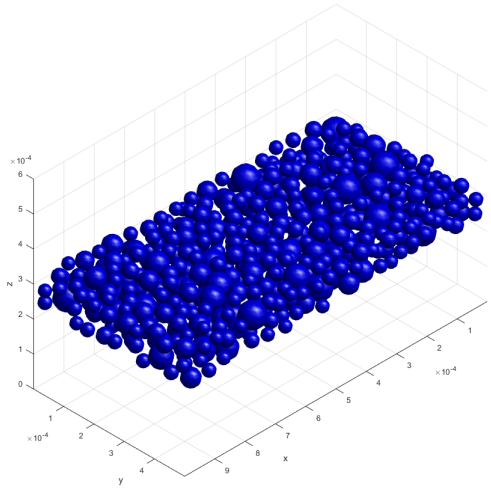

Figure 7: 512 Particles at $t = 0$

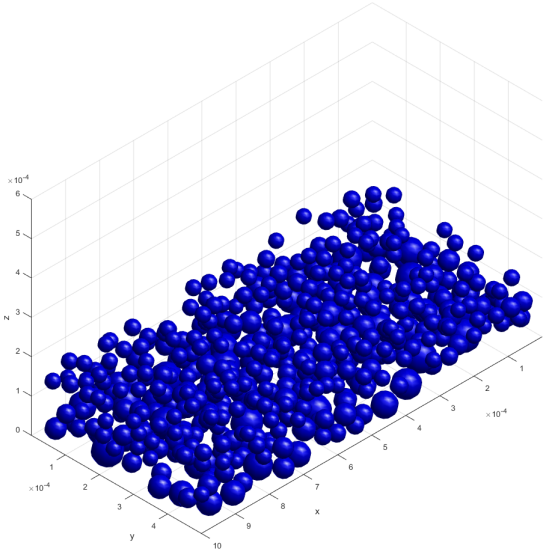

Figure 8: 512 Particles at $t = 0.07$

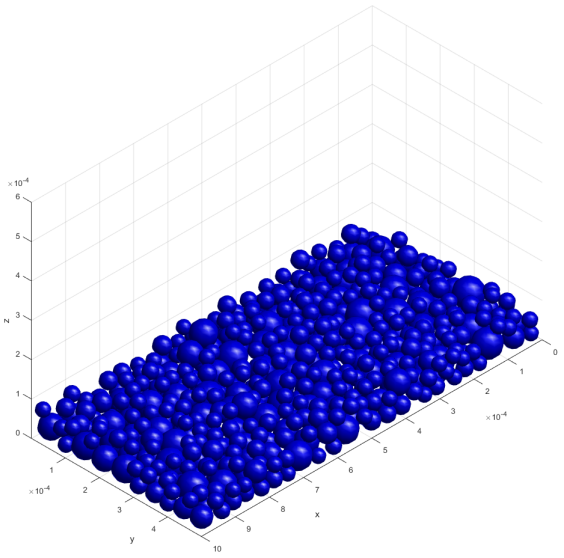

Figure 9: 512 Particles at $t = 0.3$

Also the inclusive wall time of CPU and GPU is compared in figure (10), which shows gpu code can be much faster when the particle numbers are large.

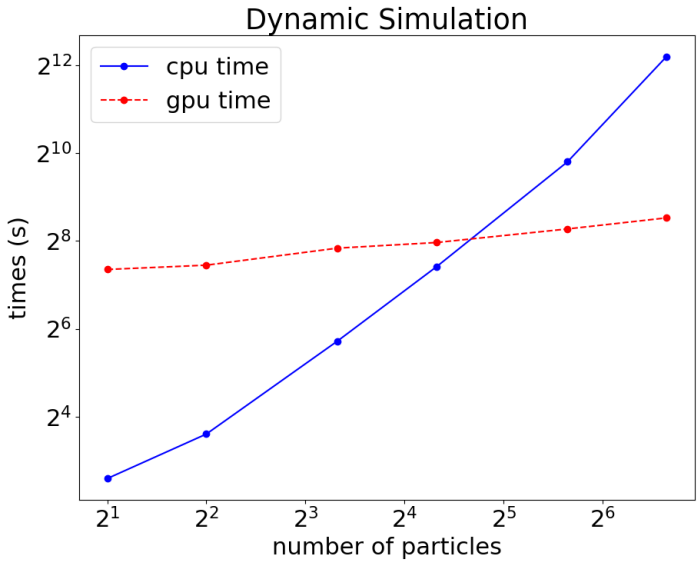

Figure 10: Comparison of CPU and inclusive GPU time. (CPU: Intel Core i7-6500@ 2.5GHz 2.59GHz GPU: NVIDIA GeForce GTX 980 Ti)

3.2 Thermal simulation

Both CPU and GPU version of thermal simulation are implemented. A natural parallelism exists in thermal simulation. In each time step, the temperature updating of each particle is independent with each other. The simulation procedure of CUDA version is listed below.

1. Initialize particles (using the output from dynamic simulation) and laser on the host;
2. Copy particles and laser variables to device;

3. In each time step ($\Delta t = 5 \times 10^{-9}\ s$), launch a CUDA kernel:
   - update the position and power of the laser according to manufacturing plan;
   - update temperature of all the particles;
4. Copy the results back to host;
5. Write temperature information to file.

Figure 11 shows the allocation of threads in the temperature updating kernel. In it, H_i accounts for the laser heat input; Q_i includes heat conduction to other particles, convection and radiation. Each thread is responsible for updating temperature of one particle.

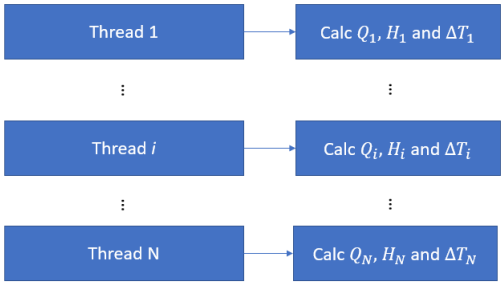

Figure 11: Schematic of CUDA parallelism for thermal simulation

Applying thermal simulation ($nSteps = 20,000$) on different number of particles, we obtained below scaling analysis result (see Fig. 12). From it, we can see when the number of particles (N) is small, CPU version runs faster than GPU version. However when N is large, GPU outperforms CPU. Also we notice that, CPU time is basically $O(N^2)$. And for $N < 2^{14}$, GPU time shows $O(N)$ trend which is very impressive. The detailed discussion is in *Section* 4.2.

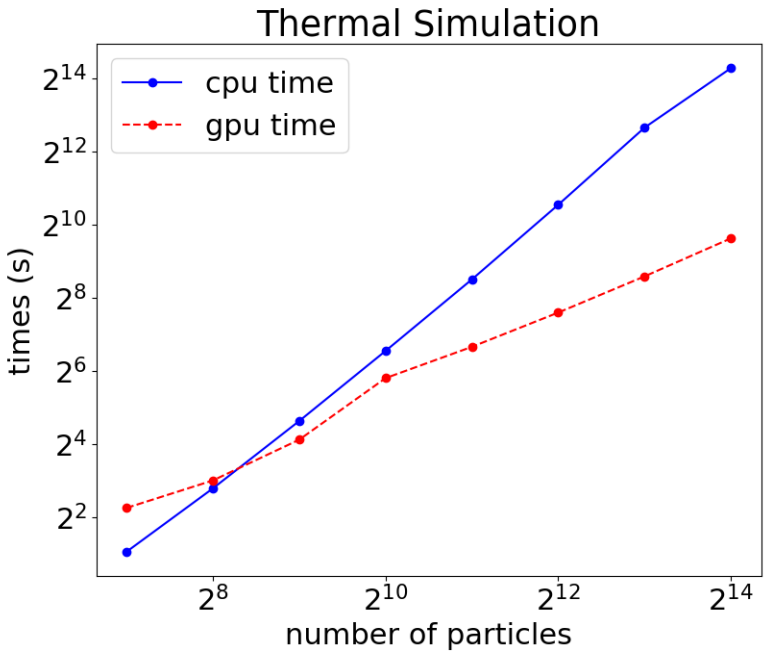

Figure 12: Comparison of CPU and inclusive GPU time. (CPU: Intel(R) Xeon(R) CPU E5-2650 v3 @ 2.30GHz; GPU: NVIDIA GeForce GTX 1080)

Some simulation results are shown below (see Fig. 13, 14, 15). In the simulation, the laser scans through the center line of the table. From them, we can see the particles under laser beam center have the highest temperature

and there is heat transferred from heated particles to their neighboring particles. Also smaller particles tend to have higher temperatures which agrees well with paper.

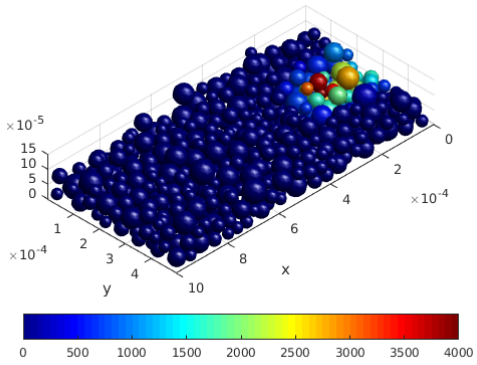

Figure 13: Screenshot at (# of steps = 20,000) showing the temperature evolution of a layer of 316L SS particles as a laser is passed over (temperature in K)

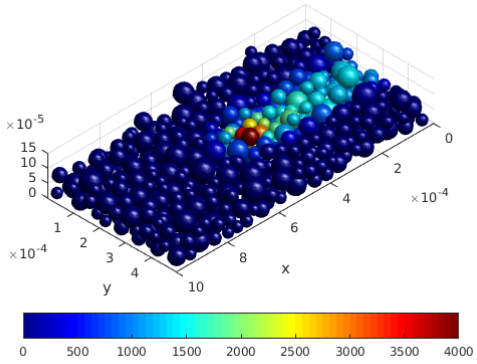

Figure 14: Screenshot at (# of steps = 50,000) showing the temperature evolution of a layer of 316L SS particles as a laser is passed over (temperature in K)

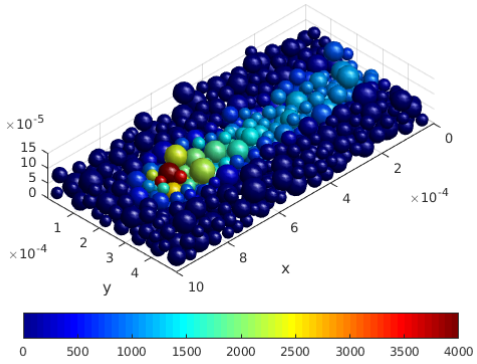

Figure 15: Screenshot at (# of steps = 80,000) showing the temperature evolution of a layer of 316L SS particles as a laser is passed over (temperature in K)

4 Discussion

4.1 Dynamic simulation

It can be seen that in dynamic simulation part, the GPU code achieve a great speed up ratio compared with the CPU code. However, when the particle amount is very small, the CPU code performs better. It is necessary to have a detailed look of the GPU code to explain such performance discrepancy.

The most important kernel is the one which calculate the acceleration of particles. In this kernel, firstly each block tackle with one particle, in the block one thread calculate the contact and friction force with walls and others calculate the potential collision pairs with other particles, then there is a block synchronization. Secondly, after the synchronization there is a reduction operation in the block to have the sum of contact forces and friction forces, in this reduction there are also several inevitable synchronization. These synchronization might not hurt too much when particle quantity is large, but when the quantity is small, the synchronization time might be comparable to the computing time.

Then it is important to have a look at the hardware specification. Though GeForce GTX 980Ti is a very powerful GPU, its max clock rate is 1.08 GHz, which is just less than half of the CPU's frequency (Core i7 6500). GPU only show its advantage when its occupancy is large. When occupancy is small (i.e. number of particles is small), only part of the 980Ti hardware (e.g. $N = 256$, and $1024\,threads/block$, only one SM is used) is used.

4.2 Thermal simulation

In CUDA version of thermal simulation, data is copied from host to device in the beginning and stays in device during the thermal simulation. And basically GPU part is a loop of repeating launching kernel functions which update particle temperatures. In the kernel function, each thread is responsible for one particle (see Fig. 11). We choose $nThreads/block = 1024$. When number of particles is small, for example $N = 256$. There is only one block which means only one SM is used (also very low occupancy). This is like shooting a bird with a cannon. Therefore CPU outperforms GPU when number of particles is small. When particle number increases, GPU starts to show its advantageous over CPU. Since high end GPUs have much more ALUs than CPUs. From Fig. 12, we can see, as number of particles increases, CPU time grows as $O(N^2)$ which agrees with the algorithm time complexity. However GPU time shows $O(N)$ trend (as $N < 2^{14}$). This is because when N is relatively small, GPU can increase its computing power through increasing occupancy.

In conclusion, GPU is suitable for discrete element method.

References

[1] Rishi Ganeriwala and Tarek I Zohdi. A coupled discrete element-finite difference model of selective laser sintering. *Granular Matter*, 18(2):21, 2016.

[2] Jeff Hecht. *Understanding lasers: an entry-level guide*, volume 21. John Wiley & Sons, 2011.

[3] AV Gusarov and J-P Kruth. Modelling of radiation transfer in metallic powders at laser treatment. *International Journal of Heat and Mass Transfer*, 48(16):3423–3434, 2005.

[4] AV Gusarov, I Yadroitsev, Ph Bertrand, and I Smurov. Model of radiation and heat transfer in laser-powder interaction zone at selective laser melting. *Journal of heat transfer*, 131(7):072101, 2009.

[5] Comini Bonacina, G Comini, A Fasano, and M 1973 Primicerio. Numerical solution of phase-change problems. *International Journal of Heat and Mass Transfer*, 16(10):1825–1832, 1973.

[6] Mohamad Muhieddine, Edouard Canot, and Ramiro March. Various approaches for solving problems in heat conduction with phase change. *International Journal on Finite Volumes*, page 19, 2009.

[7] *Mechanical and physical properties of the austenitic chromium-nickel stainless steels at elevated temperatures*. The International Nickel Company Inc., New York (1968).