AAS 16-242

A GPU-ACCELERATED MULTIPHASE COMPUTATIONAL TOOL FOR ASTEROID FRAGMENTATION/PULVERIZATION SIMULATION

Ben J. Zimmerman* **and Bong Wie†**
Iowa State University, Ames, IA 50011

The simulation of asteroid target fragmentation or pulverization is a challenging task which demands efficient and accurate numerical methods with large computational power. To this end, the high-order Spectral Difference (SD) method is implemented with Graphics Processing Unit (GPU) computing. Hypervelocity kinetic-energy impactors are of practical interest, which generate high-pressure, deformational shock waves in the target bodies upon impact. Due to the extremely short deformation time associated with hypervelocity impact, the material behaves in a similar manner to a compressible fluid, and the compressible Euler equations can be applied. To model the multiple material interactions, an n-phase equation model is adopted into the SD method. All simulations presented were solved with GPUs, producing solutions at orders of magnitudes faster than the Central Processing Unit (CPU) counterpart. Several impact cases are compared, including a heavy impactor and multiple impactor system, against an asteroid target. Orbital dispersion effectiveness is evaluated and results indicate that the multiple impactor system outperforms the single heavy impactor.

INTRODUCTION

The impact threat of near-Earth objects (NEOs) is a concern to the global community, as evidenced by the Chelyabinsk event (caused by a 17-m meteorite) in Russia on February 15, 2013 and a near miss by asteroid 2012 DA_{14} ($\sim$30 m diameter), on the same day. The expected energy, from either a low-altitude air burst or direct impact, would have severe consequences, especially in populated regions. To mitigate this threat, several methods have been proposed by changing the trajectory of the incoming asteroid body. While these methods are appealing, a substantial amount of mission lead time is required to alter the asteroid targets trajectory to avoid Earth impact. Hence, they are ineffective when considering short warning cases. In this regime, significant damage must be applied to the asteroid such that it is completely destroyed or the resulting fragments are small enough to disintegrate upon entering the Earth's atmosphere. To accomplish this feat, nuclear explosive devices (NED) [1] or large kinetic-energy impactors (KEIs) must be employed.

A thorough investigation of the aforementioned NED or KEI scenarios should be completed, as non-ideal fragmentation of the target can occur [2]. Several papers have already investigated the NED approaches [3–5], where massive target damage is observed. The current work is focused on non-nuclear methods, specifically KEIs. If the energy imparted by the impact is large enough, disruption of smaller targets (< 150 m) may become feasible. Two different impactor configurations

*Ph.D. Research Assistant, Dept. of Aerospace Engineering, 1200 Howe Hall, bzimmer@iastate.edu.
†Vance Coffman Endowed Chair Professor, Asteroid Deflection Research Center, Dept. of Aerospace Engineering, bongwie@iastate.edu.

are considered in this work, a single heavy impactor and a multi-bodied system. The Multiple Kinetic-Energy Impactor Vehicle (MKIV), first proposed in [6], which stems from work in [7], is composed of mutliple impactors designed to hit the target concurrently, distributing damage over the surface. In previous work [5], the MKIV system was simulated against a target and compared with a Single Kinetic-Energy Impactor Vehicle (SKIV) system. This work seeks to build upon previous results, employing a new numerical approach coupled with a simple damage model.

The KEIs considered in this study strike the target at hypervelocity (the velocity of the impactor is greater than the speed of sound in the target). In this work, the impactor travels at 11.5 km/s, roughly twice that of the speed of sound in a granite target. Due to this high velocity impact, the physical process can be modeled as a fluid instead of a solid. One such reason given in literature is the resulting impact generates pressures within the target much greater than the target materials ultimate strength [8–10]. Perhaps a better argument is the impact will generate longitudinal waves (P-waves) and shear waves (S-waves). P-waves travel faster than S-waves [11] (nearly twice as fast), hence a point in the target beyond the impact location will experience a P-wave disturbance before the S-wave arrives. For general hypervelocity impact, the P-wave particle displacement is greater than the maximum compressive strain of the target, thus causing compressive failure in the target. The S-wave propagation is hindered by the compressive failure from the P-wave, and has difficulty traveling through the damaged regions. This allows a fluid-type model to be applied (S-waves do not travel in fluids), since all damage arises from the P-wave. With this assumption, the compressible Euler equations can be applied.

To discretize the equations, the high-order spectral difference (SD) method (a method found in computational fluid dynamics) is applied [12–14]. The SD method follows an Eulerian finite difference-like formulation, where two sets of points are defined within computational elements to store solution data (solution points) and compute the required derivatives (flux points). The solution points are chosen to support a polynomial reconstruction of a given order of accuracy, while the flux points support a reconstruction of one order higher, since flux derivatives are required to update the governing equations [15]. If the points are distributed in a manner geometrically similar for all elements, then the reconstruction and derivative coefficients become the same for all elements. The coefficients for all elements can be computed from only one element, reducing memory storage cost. In addition, no matrix inversion is required for the method, as the elements are assumed decoupled. Another major benefit of SD is the compactness of the scheme, increasing the level of parallelism. This makes the method appealing to NVIDIA's graphics processing unit (GPU) Compute Unified Device Architecture (CUDA) computing [16].

The numerical simulation of impact phenomenon requires a model for distinguishing multiple materials. In the presented problems, three different phases must be tracked: Aluminum, granite, and empty space. The impactor is a solid body consisting of aluminum, the target asteroid is assumed to be granite composition, and the outside space is modeled as a low density air (setting zero density in a region for Eulerian methods produces infinity). It is known that improper treatment of material interfaces can lead to non-physical pressure oscillations, even in first order models [17, 18]. The introduction of high-order methods to these simulations further destabilizes the interface pressure due to interpolation polynomials, leading to numerical instability. Mutliple approaches have been developed to efficiently handle these multicomponent problems for different equations of states, including the stiffened gas [19] and Mie-Grüneisen [20] equation of states. In order to prohibit non-physical results, the Diffused-Interface Method (DIM) [21, 22] (sometimes referred to as the 4-equation model) is adapted into the SD framework. To the authors knowledge,

this is the first use of the high-order SD method coupled with DIM.

This paper is organized in the following manner. The overall numerical approach is first covered, including the governing equations and the numerical method, followed by the application of CUDA GPU computing. Several validation cases are then presented with the asteroid impact simulations.

NUMERICAL APPROACH

Governing Equations

The two-dimensional Euler equations are chosen for numerical modeling, and are governed by conservation of mass, momentum, and energy

$$\frac{\partial \rho}{\partial t} + \frac{\partial (\rho u)}{\partial x} + \frac{\partial (\rho v)}{\partial y} = 0 \tag{1}$$

$$\frac{\partial \rho u}{\partial t} + \frac{\partial \left(\rho u^2 + p\right)}{\partial x} + \frac{\partial (\rho u v)}{\partial y} = 0 \tag{2}$$

$$\frac{\partial \rho v}{\partial t} + \frac{\partial (\rho u v)}{\partial x} + \frac{\partial \left(\rho v^2 + p\right)}{\partial y} = 0 \tag{3}$$

$$\frac{\partial e}{\partial t} + \frac{\partial [u(e+p)]}{\partial x} + \frac{\partial [v(e+p)]}{\partial y} = 0 \tag{4}$$

In Equations (1) - (4), ρ is the density, u is the x-direction velocity, v is the y-direction velocity, p is the pressure, and e is the total energy per unit volume, a combination of internal energy (e_i) and kinetic energy (e_k). To close the system, an appropriate equation of state must be defined as

$$p = f(\rho, e_i) \tag{5}$$

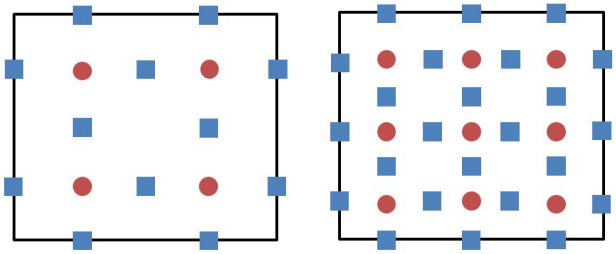
The pressure is a function of density, internal energy, and several constants which depend on the equation of state used and material properties. In order to model multiple materials, the interface of materials must be included in the formulation. Let a specific phase (say phase 1) occupy locations in the domain, where the phase is associated with a volume fraction of the fluid (α_1)

$$\frac{\partial \alpha_1}{\partial t} + u\frac{\partial \alpha_1}{\partial x} + v\frac{\partial \alpha_1}{\partial y} = 0 \tag{6}$$

If only two phases exist, then it follows that $\alpha_2 = 1 - \alpha_1$. For more than two phases, additional equations are required to track the materials. The volume fraction, Equation (6), is in non-conservative form, and can be cast into quasi-conservative form with an added source term as

$$\frac{\partial \alpha_1}{\partial t} + \frac{\partial (u\alpha_1)}{\partial x} + \frac{\partial (v\alpha_1)}{\partial y} = \alpha_1 \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right) \tag{7}$$

The system to be solved is thus composed of five equations for mass, momentum, energy, and fluid phase for a two-phase simulation. If additional phases are required for modeling, the system defined by Equations (1) - (4) and (7) must be modified. Additional mass conservation and phase advection equations are integrated into the system to form a system of $(2n+3)$ equations, where n is the total number of phases assuming $n > 2$. The new system needs two equations for momentum and one



(a) P^1 reconstruction (b) P^2 reconstruction

Figure 1. Solution points (circles) and flux points (squares) for P^1 and P^2 reconstruction.

for energy (like before) with an additional n equations for mass conservation and n equations for volume fractions,

$$\frac{\partial \rho\alpha_k}{\partial t} + \frac{\partial \left(\rho\alpha_k u\right)}{\partial x} + \frac{\partial \left(\rho\alpha_k v\right)}{\partial y} = 0, \tag{8}$$

$$\frac{\partial \rho u}{\partial t} + \frac{\partial \left(\rho u^2 + p\right)}{\partial x} + \frac{\partial \left(\rho uv\right)}{\partial y} = 0, \tag{9}$$

$$\frac{\partial \rho v}{\partial t} + \frac{\partial \left(\rho uv\right)}{\partial x} + \frac{\partial \left(\rho v^2 + p\right)}{\partial y} = 0, \tag{10}$$

$$\frac{\partial e}{\partial t} + \frac{\partial \left[u(e+p)\right]}{\partial x} + \frac{\partial \left[v(e+p)\right]}{\partial y} = 0, \tag{11}$$

$$\frac{\partial \alpha_k}{\partial t} + \frac{\partial \left(u\alpha_k\right)}{\partial x} + \frac{\partial \left(v\alpha_k\right)}{\partial y} - \alpha_k\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right) = 0. \tag{12}$$

In the new system, k is the individual material phase ($k = 1, 2, ..., n$). The system can be written in the hyperbolic conservation form as

$$\frac{\partial \boldsymbol{q}}{\partial t} + \frac{\partial \boldsymbol{f}}{\partial x} + \frac{\partial \boldsymbol{g}}{\partial y} = \boldsymbol{S} \tag{13}$$

where $\boldsymbol{q}$ is the state variable vector, $\boldsymbol{f}$ and $\boldsymbol{g}$ are flux vectors, and $\boldsymbol{S}$ is the source vector defined as

$$\boldsymbol{q} = \begin{bmatrix} \rho\alpha_k \\ \rho u \\ \rho v \\ e \\ \alpha_k \end{bmatrix} \qquad \boldsymbol{f} = \begin{bmatrix} \rho u\alpha_k \\ p + \rho u^2 \\ \rho uv \\ u(e+p) \\ u\alpha_k \end{bmatrix} \qquad \boldsymbol{g} = \begin{bmatrix} \rho v\alpha_k \\ \rho uv \\ p + \rho v^2 \\ v(e+p) \\ v\alpha_k \end{bmatrix} \qquad \boldsymbol{S} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \alpha_k\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right) \end{bmatrix} \tag{14}$$

Numerical Method

Equation (13) is to be solved on a non-overlapping quadrilateral grid. Within each element, two sets of points are defined, solution points and flux points. The solution points are the locations where the conserved variables in $\boldsymbol{q}$ are stored, while the flux points are used to compute the values of the flux ($\boldsymbol{f}$ and $\boldsymbol{g}$). In the current implementation, the solution points and flux points are defined as Gauss-Legendre and Gauss-Lobatto points respectively. To maintain constant coefficients and

point locations throughout the domain, each element is transformed into a standard quadrilateral element $(\xi, \eta) \in [-1, 1] \times [-1, 1]$ [23].

For a given order of accuracy, a specific amount of points are required within each element. The solution is assumed to belong to the space of polynomials degree k ($\boldsymbol{q} \in P^k$). Figure 1 (a) and (b) show the point distribution for P^1 (linear) and P^2 (quadratic) reconstructions. Note that there is one extra point per direction for the flux points. This is due to the spatial derivative, which will decrease the flux polynomial order by one, hence the flux polynomials must belong to degrees $k+1$ ($\boldsymbol{f}, \boldsymbol{g} \in P^{k+1}$). Then the flux derivative will belong to the same space as the solution polynomial.

The $(k + 1)$ solutions at the solution points build a degree k polynomial following a Lagrange polynomial basis as

$$l_j(x) = \prod_{s=1, s\neq j}^{k+1} \left(\frac{x - x_s}{x_j - x_s} \right) \tag{15}$$

where x is the point locations. In a similar manner, the flux polynomial can be built across the $k+2$ flux points as

$$h_{j+1/2}(x) = \prod_{s=0, s\neq j}^{k+1} \left(\frac{x - x_{s+1/2}}{x_{j+1/2} - x_{s+1/2}} \right) \tag{16}$$

In the SD method, the reconstructed solution polynomial is a tensor product of two one-dimensional polynomials of the form

$$\boldsymbol{q}(\xi, \eta) = \sum_{j=1}^{k+1} \sum_{i=1}^{k+1} \boldsymbol{q}_{i,j} l_i(\xi) l_j(\eta) \tag{17}$$

In a similar manner, the flux polynomials are expressed as

$$\boldsymbol{f}(\xi, \eta) = \sum_{j=1}^{k+1} \sum_{i=0}^{k+1} \boldsymbol{f}_{i+1/2,j} h_{i+1/2}(\xi) h_j(\eta) \tag{18}$$

$$\boldsymbol{g}(\xi, \eta) = \sum_{j=0}^{k+1} \sum_{i=1}^{k+1} \boldsymbol{g}_{i,j+1/2} h_i(\xi) h_{j+1/2}(\eta) \tag{19}$$

There has been no continuity requirement across the element interfaces, and the reconstructed flux polynomials are only element-wise continuous. In order to resolve the interface flux, a Riemann solver is needed. In this study, the HLLC Riemann solver [24] is used, which replaces the values of the flux at the interfaces.

Once the flux terms are evaluated, the flux derivatives can be computed at each point within each element as follows:

$$\left(\frac{\partial \boldsymbol{f}}{\partial \xi} \right)_{i,j} = \sum_{r=0}^{k+1} \boldsymbol{f}_{r+1/2,j} l'_{r+1/2}(\xi_i) \tag{20}$$

$$\left(\frac{\partial \boldsymbol{g}}{\partial \eta} \right)_{i,j} = \sum_{r=0}^{k+1} \boldsymbol{g}_{i,r+1/2} l'_{r+1/2}(\eta_j) \tag{21}$$

where i and j are the indices of the solution points in x and y directions. After the flux derivatives are updated, the source term is added to complete the system. To compute the source term, the

velocities, u and v, are stored at the flux points, and their derivatives are computed in a similar manner to Equations (20) and (21).

The flux derivatives and source term are gathered together into the right hand side ($\boldsymbol{r}$) of the system, such that

$$\boldsymbol{r} = -\frac{\partial \boldsymbol{f}}{\partial x} - \frac{\partial \boldsymbol{g}}{\partial y} + \boldsymbol{S} \tag{22}$$

Explicit time integration is completed using a three-stage Runge-Kutta [25] scheme to march the solution forward in time from t to $t+1$ as

$$\boldsymbol{q}^{(1)} = \boldsymbol{q}^t + \Delta t \boldsymbol{r}(\boldsymbol{q}^t) \tag{23}$$

$$\boldsymbol{q}^{(2)} = \frac{3}{4}\boldsymbol{q}^t + \frac{1}{4}\boldsymbol{q}^{(1)} + \frac{1}{4}\Delta t \boldsymbol{r}(\boldsymbol{q}^{(1)}) \tag{24}$$

$$\boldsymbol{q}^{(t+1)} = \frac{1}{3}\boldsymbol{q}^t + \frac{2}{3}\boldsymbol{q}^{(2)} + \frac{2}{3}\Delta t \boldsymbol{r}(\boldsymbol{q}^{(2)}) \tag{25}$$

where Δt is the chosen time step for integration.

Equations of State

In the current study, three different equations of state will be used throughout: Ideal gas, stiffened gas, and Mie-Grüneisen.

Ideal and stiffened gas The internal energy is computed as

$$e_i = \frac{p + \gamma p_o}{\gamma - 1} \tag{26}$$

where γ is the ratio of specific heats and p_o is a pressure coefficient dependent on material properties. From Equation (26), setting the p_o term to zero yields the ideal gas equation of state. It follows that the pressure is computed as

$$p = (e_i - \gamma p_o)(\gamma - 1) \tag{27}$$

Each material is allowed to have differing γ and p_o terms, hence the volume fractions are needed in computing the pressure. To stay consistent with the model, the terms must be computed in the following manner:

$$\frac{1}{\gamma - 1} = \sum_k \frac{\alpha_k}{\gamma_k - 1} \tag{28}$$

$$\gamma p_o = \frac{\sum_k \frac{\alpha_k \gamma_k p_{o,k}}{\gamma_k - 1}}{\sum_k \frac{\alpha_k}{\gamma_k - 1}} \tag{29}$$

This will guarantee no pressure oscillations at the material interface. Finally, the speed of sound is computed as

$$c = \sqrt{\frac{\gamma(p + p_o)}{\rho}} \tag{30}$$

Mie-Grüneisen The Mie-Grüneisen equation of state is more complicated than the stiffened gas model, and it depends on material densities. The pressure is described by

$$p = \left(e_i + \frac{p_{ref}}{\Gamma} - \rho e_{ref}\right) / \left(\frac{1}{\Gamma}\right) \tag{31}$$

where Γ, p_{ref}, and e_{ref} are all functions of density. The Grüneisen coefficient, Γ, takes the form

$$\Gamma = \Gamma_0 \left(\frac{V}{V_0}\right)^{\alpha} \tag{32}$$

where $\Gamma_0 = 1 - \gamma_0$ represents the Grüneisen coefficient at the initial density, $V = \frac{1}{\rho}$, and $\alpha = 1$ in the current work. The reference values for pressure and energy are

$$p_{ref} = p_0 + \frac{c_0^2(V_0 - V)}{[V_0 - s(V_0 - V)]^2} \tag{33}$$

$$e_{ref} = e_0 + \frac{1}{2}(p_{ref} + p_0)(V_0 - V) \tag{34}$$

where p_0, e_0, c_0, and s all depend on the material to be modeled. In a similar manner to the stiffened equation of state, the volume fractions can be applied to compute material mixtures throughout the domain. The speed of sound for the Mie-Grüneisen equation of state is

$$c^2 = \left[\Gamma + 1 + \frac{\rho}{\Gamma}\frac{d\Gamma}{d\rho}\right]\left(\frac{p - p_{ref}}{\rho}\right) + \frac{\Gamma p_{ref}}{\rho} + \frac{dp_{ref}}{d\rho} - \Gamma\rho\frac{de_{ref}}{d\rho} \tag{35}$$

The derivatives can be computed analytically and manually entered into the computational tool.

Damage Model

To enable material fracturing within the fluid model, damage characterization is necessary. It is important to note that our model is a two-dimensional one, which physically represents an infinitely long rod impacting an infinitely long cylinder. Inside an element, a change in the density is related to a change in volume (area for a two-dimensional model) as

$$\frac{\rho_0}{\rho} = \frac{V}{V_0} \tag{36}$$

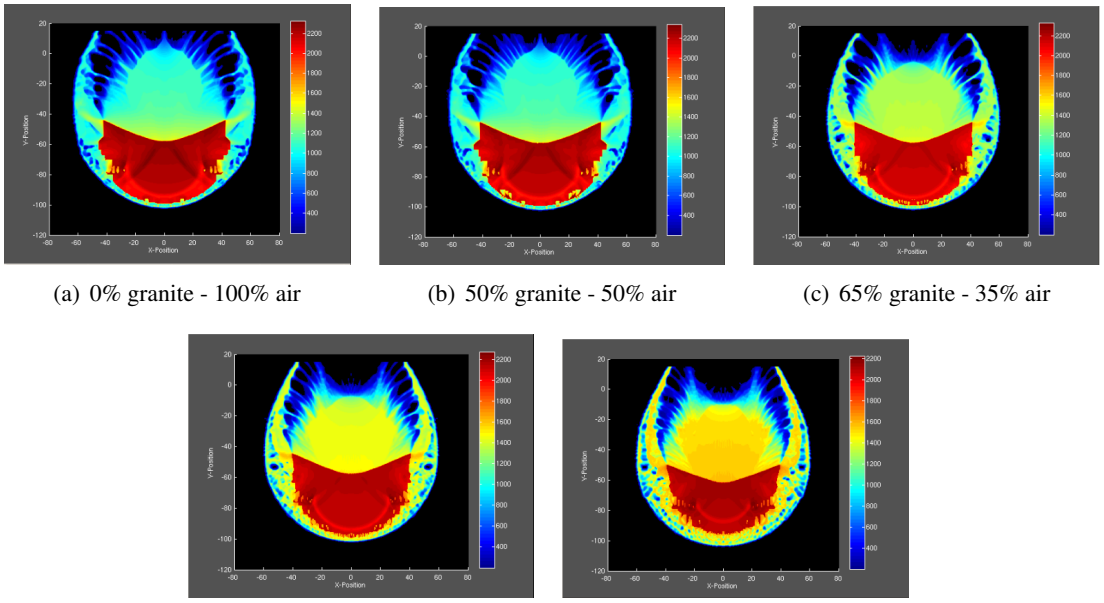
If each element's edge lengths are assumed unity, and the edge length is changed by a small amount Δ, then

$$\frac{V}{V_0} = \left(\frac{1 + \Delta}{1}\right)^2 = (1 + \epsilon)^2 \tag{37}$$

where ϵ is the strain. The strain can be written in terms of the density ratio as

$$\frac{\rho_0}{\rho} = (1 + \epsilon)^2 \tag{38}$$

Properties for the granite asteroid model were taken from experimental average results where the maximum compression strain is $\epsilon_c \approx 2000.0\mu$ strain and the maximum tensile strain is $\epsilon_t \approx 500.0\mu$ strain [26]. The material is in compression if the density ratio $\rho_0/\rho > 1.0$ and in tension if $\rho_0/\rho < 1.0$.



(a) 0% granite - 100% air (b) 50% granite - 50% air (c) 65% granite - 35% air

(d) 70% granite - 30% air (e) 75% granite - 25% air

Figure 2. Density (kg/m^3) contours of different phase mixtures for granite and air

Once the element is detected as damaged, a phase change is completed to inform the element of its new state. This change of phase is a mixture of air and granite, where the mixture proportion investigations were completed in a parameter study. A 100 m granite circular target is hit with a 5,000 kg aluminum impactor traveling at 11.5 km/s. Figure 2 illustrates five different mixtures considered. Figure 2 (a) represents a mixture where the damaged target looses all of its material properties and behaves similar to air. In this simulation, the crater is completely lost, and the damaged material travels in the positive y-direction. Even changing the mixture to 50% granite and 50% air, as shown in Figure 2 (b), shows very little change in crater or damage amount. Once the mixture is above 60% granite, the crater becomes more apparent. Figure 2 (c) shows the formulated crater at the impact location. Further increasing the phase difference, Figures 2 (d) and (e), yields a better crater profile with comparable damage between the two figures. For the remainder of this paper, a 70% granite and 30% air phase change is assumed for the damaged material as shown in Figure 2 (d). Physically, this indicates that the damaged material has lost 30% of its original stiffness and behaves more like a rubble pile. A compressed region in the target is apparent throughout all phase mixtures. In this region, there will most likely be damage due to shear waves, which are not modeled currently. Thus, more damage can be expected within the core of the target.

Slope Limiting

The problems to be simulated contain solution discontinuities, which can cause oscillations, non-physical results, and numerical instability. If a first-order accurate model is adopted, the results become smeared near the discontinuities due to the numerical viscosity associated with a first order method [27]. Once higher-order methods are employed, numerical issues mentioned before arise. In this work, a *minmod* limiter [28] is adopted to located troubled elements and apply slope limiting. At element edges, the *minmod* limiter is applied to reconstruct a element-average based solution. This reconstructed average is compared with the solution at the edge, and if the difference is large

(numerical experiments indicate $> 1.0 \times 10^{-3}$ gives good solutions) then the element is marked for limiting. The new solution then takes the form

$$\boldsymbol{q}_{m,j} = \bar{\boldsymbol{q}}_m + (x_{m,j} - x0)\text{minmod}\left(\frac{\bar{\boldsymbol{q}}_{m+1} - \bar{\boldsymbol{q}}_m}{h}, \frac{\bar{\boldsymbol{q}}_m - \bar{\boldsymbol{q}}_{m-1}}{h}\right) \tag{39}$$

where $\boldsymbol{q}_{m,j}$ is the solution in element m at solution point j, $x_{m,j}$ is the location of the solution point, $x0$ is the element midpoint location, h is the element size, and $\bar{\boldsymbol{q}}_m$ is the averaged solution in the element. This scheme results in a second-order solution reconstruction which eliminates the numerical issues discussed previously.

GPU CUDA COMPUTING

The application to GPU CUDA computing requires programming the SD method in a manner optimal for the GPU architecture. When a GPU function is launched (called a kernel), the GPU forms a grid, which is composed of blocks, which in turn is composed of threads. The blocks are scheduled to run in parallel across the GPUs streaming multiprocessors. Additional consideration is needed to manage specific memory locations to maximize the throughput of the GPU. The memory types implemented in the current work are global, texture, local, and shared. The GPU global memory can be seen by all threads in all blocks, and should be coalesced and minimized. The global memory is bound to the GPUs texture memory, which is read-only and beneficial when non-coalesced access is required. Each thread inherits local memory, which can only be seen by the thread. Most computations are completed within the threads local memory, which acts in a similar manner to registers on a CPU. Finally, shared memory is used when threads within the same block need to use information local to other threads. Typically, this is done during for-loop computations (e.g. derivatives).

SD CUDA

The SD algorithm decomposes naturally into three separate parts: Interpolate the solution to the flux points, couple the elements using a Riemann solver, and update of right-hand side of the system.

The interpolation algorithm runs $(k+1)(k+2)$ threads within each block (corresponding to the number of flux points in one dimension). The solution states, $\boldsymbol{q}$ are read into the GPUs shared memory and are interpolated using

$$\tilde{\boldsymbol{q}}_i^l = \sum_{j=1}^{k+1} \boldsymbol{q}_j^s h_{i,j}^t \tag{40}$$

where $\tilde{\boldsymbol{q}}$ are the solution states evaluated at flux points, i is the flux point, j is the solution point, h is the interpolation polynomial, the superscript s is shared memory, the superscript l is local memory, and the superscript t is texture memory. The fluxes are evaluated according to Equation (14) in each direction and written to the GPUs global memory along with the reconstructed states in Equation (40).

The element-coupling algorithm uses the state information interpolated to element edges to evaluate a Riemann flux at the points. The threads run across the $(k+1)$ points on each elements face, reading the current and neighboring elements states from texture memory. This kernel also handles boundary conditions if there is no neighboring element to the current element. The Riemann fluxes are written into the fluxes global memory, which overwrites the information from the interpolation algorithm.

Table 1. Speed Comparison

Hardware	P^1 (time / iteration)	Speed-up	P^2 (time / iteration)	Speed-up
Intel Xeon	2.266	-	3.156	-
NVIDIA K20c	0.019	119.3	0.032	98.8

The final kernel updates the right hand side, $\boldsymbol{r}$, where $(k+1)(k+1)$ threads operate within each block. The flux derivative is computed as

$$\frac{d\boldsymbol{f}^{\,l}}{dx}_{j} = \sum_{i=1}^{k+2} \boldsymbol{f}_i^t dl_{j,i}^t \tag{41}$$

where dl is the derivative coefficients. The source term derivatives are computed in a similar manner if necessary. The flux derivatives are written to the global memory $\boldsymbol{r}_j^g$, at each solution point j. The right hand side can then be used to step forward in time using Equations (23) - (25).

Speed Comparisons

To demonstrate the computational power of the GPU as compared to a CPU run, consider the following test case. A blast wave is initiated in a domain with 160,000 elements. Both P^1 and P^2 reconstructions are simulated. The CPU is a Intel Xeon E5-2640 at 2.50 GHz and the GPU is a NVIDIA Tesla K20c. Proper compiler optimization flags are employed for both GPU and GPU codes, and all computations are completed in double precision.

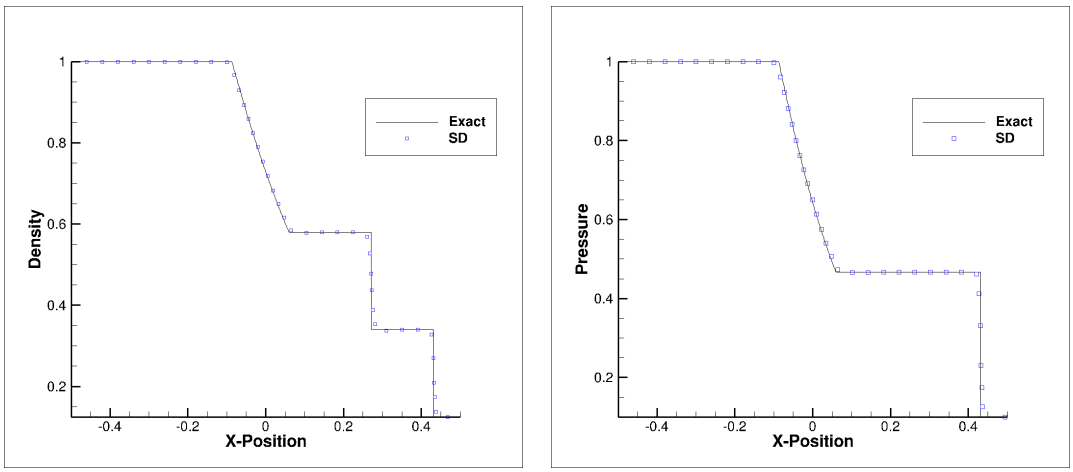
Table 1 shows the time per iteration for both CPU and GPU simulations, with GPU speed-ups. From only utilizing one GPU card, a huge performance increase is observed for both P^1 and P^2 simulations. While P^1 observes a nearly 120 times speed increase, P^2 is close to 100 times speed-up. The computing machines have a total of four GPU K20c cards, and by utilizing all cards an addition 2.5 to 3 times speed-up factor is observed. It is not a linear increase due to hardware limitations, where GPU communication requires CPU communication first.

RESULTS

In this section, several validation cases are discussed to verify the developed model for a variety of numerical problems. Additionally, asteroid impact simulation results are presented and compared.

Modified Sod's Shock Problem

Consider a one-dimensional domain, $x \in [-0.5, 0.5]$, where two fluids are separated at $x = 0$. The left state is $(\rho, u, p)_l = (1.0, 0.75, 1.0)$ and the right state is $(\rho, u, p)_r = (0.125, 0.0, 0.1)$. Note that all variables in this case are dimensionless. The specific heat ratio is $\gamma = 1.4$ and the ideal gas equation of state is used for modeling. This case can produce entropy violations by generating an expansion shock in the rarefaction region. The domain contains 200 elements with a P^2 reconstruction within each element. The solution is shown in Figure 3 at time of $t = 0.6$. From the density profile, Figure 3 (a), no expansion shock is created within the rarefaction region, and the numerical solution compares well with the exact solution. This case also illustrates correct implementation of the slope limiting scheme, as no numerical oscillations are observed and smearing near the discontinuities is minimal.



(a) Density profile (b) Pressure profile

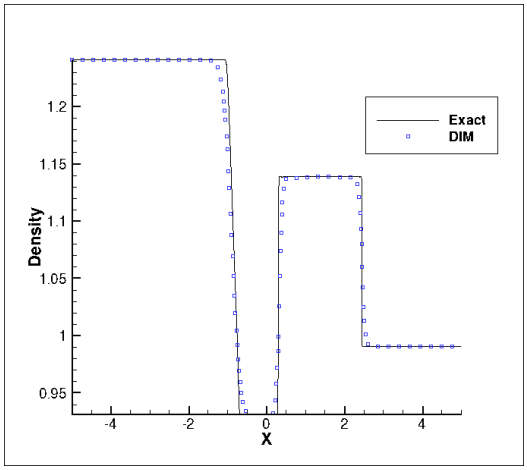
Figure 3. Solution of 2-phase shock problem at time $t = 0.6$

Two-Phase Gas-Liquid Problem

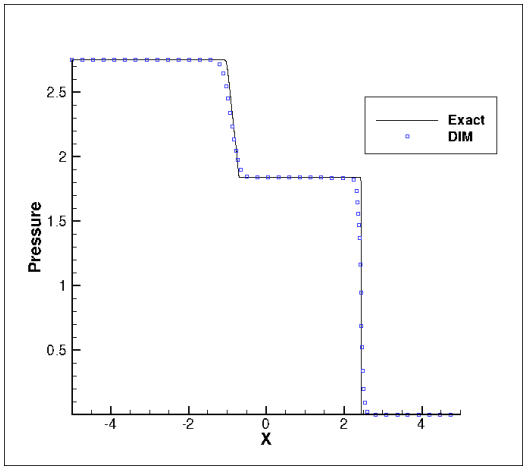
The following case demonstrates the method's ability to handle high pressure ratios and different material properties [29]. The domain, $x \in [-5, 5]$, is discretized with 400 elements for P^2 reconstruction, with a discontinuity defined at $x = 0$. The initial conditions to the left of the separation are $(\rho, u, p, \alpha_1)_l = (1.241, 0, 2.753, 1.0)$, while to the right, the initial conditions are $(\rho, u, p, \alpha_1)_r = (0.991, 0, 3.059 \times 10^{-4}, 0.0)$. These conditions yield an interface pressure ratio of roughly 9000, a high ratio which tests the robustness of the method. The left material is modeled as air under high pressure, with a specific heat ratio of $\gamma_l = 1.4$ and pressure coefficient $P_{\infty,l} = 0.0$. The right material is modeled as a liquid with a high specific heat ratio of $\gamma_r = 5.5$ and pressure coefficient $P_{\infty,r} = 1.505$. Both materials are modeled using two different equations of states, an ideal gas model for the left state and a stiffened gas for the right state. Due to the high pressure associated with the left state, the interface between the materials will advect to the right, causing material mixing. Figure 4 (a) - (c) shows the solution profiles of density, pressure, and γ compared with the exact solution at a time of $t = 0.1$. Like the previous case, all variables are dimensionless. The computed solution matches well with the exact solution in all plots. It is important to note that no pressure oscillations at the interface are apparent, as discussed in the introduction of this paper. This case demonstrates the proper integration of the DIM with SD.

Aluminum Impact

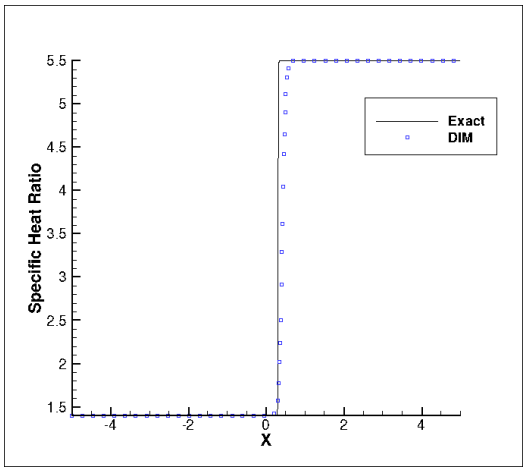
In this case, both one-dimensional and two-dimensional simulations are shown. A pre-shocked and heated semi-infinite aluminum slab is hit by another aluminum slab at ambient pressure traveling at 2 km/s [30]. All variables are again non-dimensionalized in the problem. The domain, $x \in [0, 1]$ for 1D and $[x, y] \in [0, 1] \times [0, 1]$ for 2D, is discretized with 200 and 400 elements respectively. Both simulations use a P^2 reconstruction and are simulated to a final time of $t = 0.04$. The heated aluminum slab is located at $x < 0.5$ with conditions $(\rho, u, p)_l = (4.0, 0.0, 79.3)$, while the ambient slab to the right has conditions $(\rho, u, p)_r = (2.785, -2.0, 0.0)$. Unlike the previous examples, this cases follows the Mie-Grüneisen equation of state, whose parameters for aluminum are given in Table 2. While this case does not require the use of DIM, since both states are modeled



(a) Density profile



(b) Pressure profile

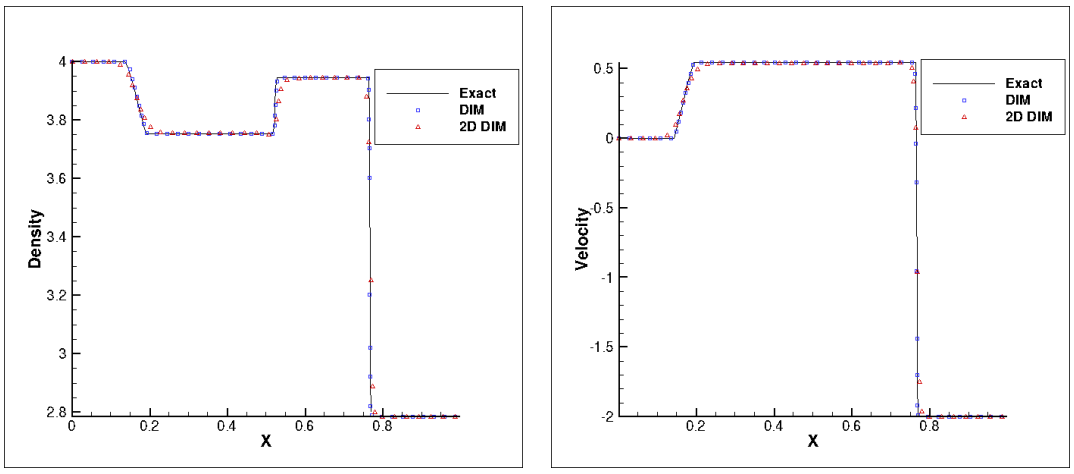


(c) Specific heat ratio profile

Figure 4. Two-phase gas-liquid problem at time $t = 0.1$

Table 2. Mie-Grüneisen aluminum coefficients

Coefficient	Value
c_0	5.238
s_0	1.338
ρ_0	2.785
e_0	0.0
p_0	0.0
Γ_0	2.0



(a) Density profile (b) Velocity profile

Figure 5. Solution of aluminum impact problem at $t = 0.04$

by the same equation of state, it does show the application of the solvers approach to handle more complicated equations of states. The density and velocity profiles are shown in Figure 5 (a) and (b) for both 1D and 2D simulations. For the 2D simulation, the data was gathered along the centerline, $y = 0.5$, and plotted with the 1D results and exact solution. Both the 1D and 2D results demonstrate the correct implementation of the equation of state and the accuracy of the method.

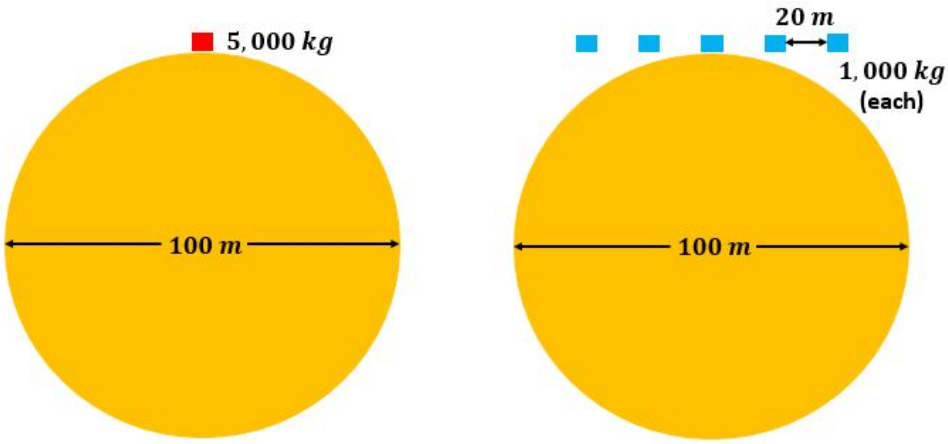
Asteroid Impact Problem

The following asteroid impact simulations are the focal point of the paper. The computational domain is $[x, y] \in [-65, -115] \times [65, 15]$ and partitioned with 250,000 elements. The reconstructed polynomial order is P^1, or four solution points within each element. Since this is a three phase problem (aluminum impactor, granite target, and outside space) the system to be solved consists of nine equations. This yields a computational domain with 9 million degrees of freedom.

Figure 6 shows the initial configuration of the intercept mission. The 2D asteroid target model is assumed to be 100 m in diameter with a constant density of 2,000 kg/m^3. All impactors are modeled as aluminum traveling at 11.5 km/s. The stiffened gas equation of state is chosen for modeling purposes as proof of concept. The parameters for the aluminum impactors are $\gamma = 3.8$ and $p_\infty = 14 \times 10^8$ [31] with a density of $\rho = 2700$ kg/m^3. The granite target parameters are $\gamma = 2.6$ and $p_\infty = 142 \times 10^8$ [32]. The third phase, outside space, is modeled with the ideal gas equation of state with a small density relative to the target and impactor. The numerical scheme can become unstable if the outside density is too small (e.g. around 0.1 kg/m^3), hence a density of 10 kg/m^3 was set at the outside space for numerical stability purposes. The total initial kinetic energy per unit volume between the two cases is held constant. For a 5,000 kg system

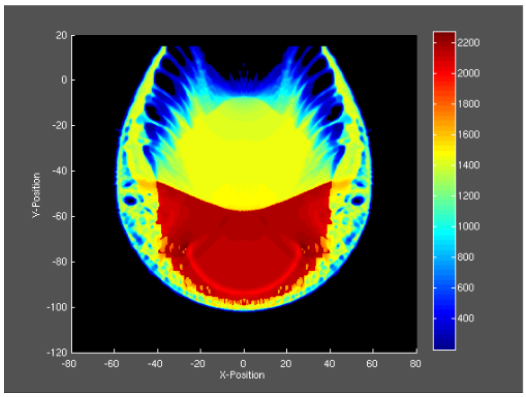
$$e_k = \frac{1}{2}\rho(u^2 + v^2) = 2500\ kg/m^3 \times (11.5\ km/s)^2 = 330.625\ GJ/m^3 \tag{42}$$

The SKIV and MKIV systems are compared in Figure 7. In Figures 7 (a) and (b), the density contours for the SKIV and MKIV impactor systems are shown. The SKIV system still maintains an elliptic-like crater at the impact location. A substantial amount of damage is observed along

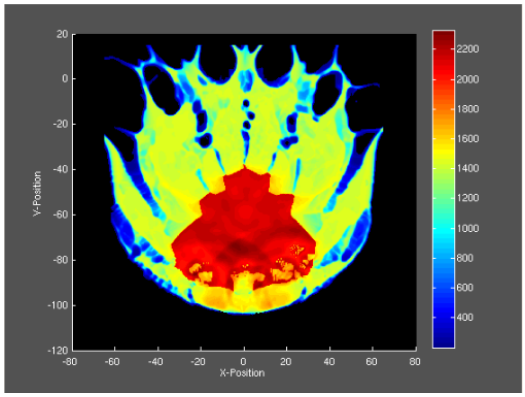


(a) SKIV setup (b) MKIV setup

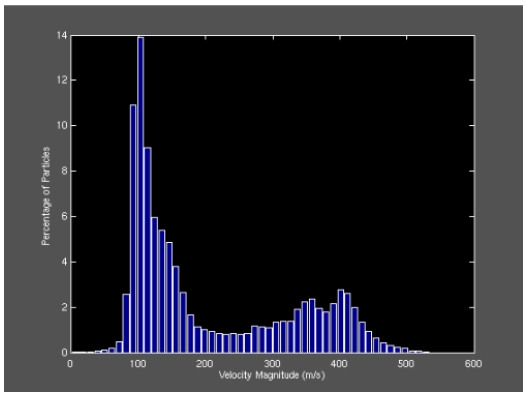
Figure 6. Hypervelocity impact simulation setup: SKIV versus MKIV



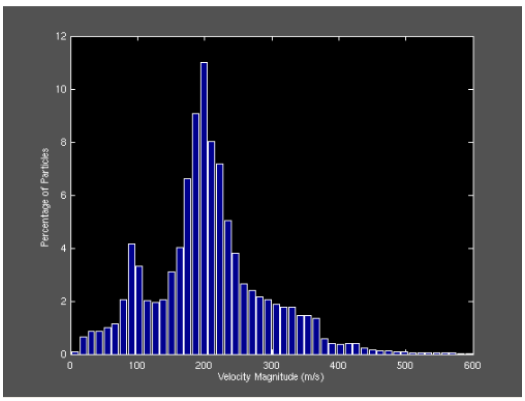
(a) Density contours (kg/m^3) SKIV



(b) Density contours (kg/m^3) MKIV



(c) Velocity histograms SKIV

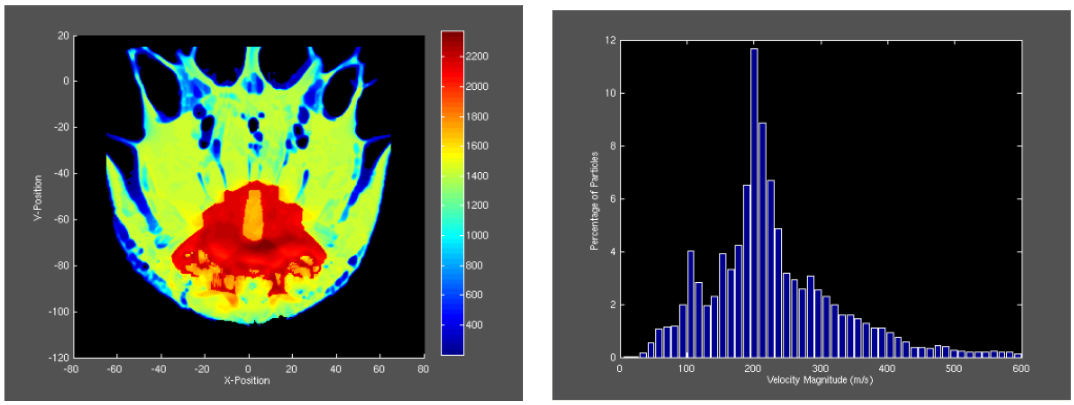


(d) Velocity histograms MKIV

Figure 7. Simulation results at time $t = 0.06$ seconds

Table 3. Averaged Dispersal Speeds at Time 0.03 Seconds

-	SKIV (5,000 kg)	MKIV (5,000 kg)	MKIV (10,000 kg)
$\|\boldsymbol{V}\|(m/s)$	123.3	132.7	164.5



(a) Density contours (kg/m^3) multi impact

(b) Velocity histograms multi impact

Figure 8. MKIV simulation results at time $t = 0.06$ seconds for 10,000 kg MKIV

the sides of the target, specifically near the impact location. A majority of the target does not maintain its initial density, and a compressed region is observed below the -40 m in the y-direction. Small amounts of damage are even seen near the back of the target, mostly due to the shock front striking the interface between granite and outside space and reflecting back towards the targets center. Additionally, the symmetric nature of the results give confidence in implementation.

The MKIV results in Figure 7 (b) show very different contours. All five impactors are assumed to impact the target at the same time, distributing the kinetic energy over the targets surface rather than in a single location. The center of the target contains voids, which may result in smaller fragments when compared to the results in Figure 7 (a). The compressed region also appears to be smaller than the single impact compression region. Visually, the MKIV system spreads more damage throughout the target body. Note there is no symmetry in this case due to the initial conditions, where the impactors could not be spaced out 20 m equally.

Figures 7 (c) and (d) show the velocity histograms taken at the final time or 0.06 s. The SKIV has a high percentage of particles traveling within the 100 - 150 m/s region, while the MKIV system shows a high percentage traveling much faster in the 150 - 250 m/s region. However, the SKIV case does illustrate a higher particle percentage traveling around the 400 m/s mark which is not seen in the MKIV case. In order to further quantify the results, the averaged dispersal speeds are computed and shown in Table 3. The MKIV system shows higher particle dispersal speeds than the SKIV system. An additional simulation is completed where the same MKIV system is initiated with a total mass of 10,000 kg (2,000 kg per impactor). The density contours and velocity histograms are shown in Figure 8 (a) and (b) for comparison. Note the shift in the velocity histogram data when compared to Figure 7 (d), where the 10,000 kg system yields more particles traveling faster.

CONCLUSIONS AND FUTURE WORK

The spectral difference method has been integrated with the diffused interface model to simulate multiple material interfaces accurately. The proposed approach has been applied to the complicated asteroid fragmentation problem, where the MKIV system has roughly 8% faster dispersal speeds than the SKIV system. Density contour plots confirm that the MKIV system produces more damage within the interior of the target body, making the approach more appealing to this problem. The mass of the MKIV system was doubled and simulated against the same target body, where an additional 24% increase in dispersal speeds was observed.

Future work is needed on equations of state and damage modeling. Additional equations of state can be implemented to better capture the physics of the interested problem. The developed damage model is a step-change within the target, which is not necessarily physical accurate. A higher order polynomial, linear or quadratic, should be integrated into the damage model to simulate a more gradual change in the phase, which will give a better prediction of the damage in the model.

REFERENCES

[1] National Research Council. Defending planet Earth: Near-Earth object surveys and hazard mitigation strategies. Report, USNRC, 2010.

[2] Sanchez J., Vasile M., and Radice G. On the consequences of a fragmentation due to a NEO mitigation strategy. *59th International Astronautical Congress*, (IAC-08-C1.3.10), 2008.

[3] Kaplinger B., Premaratne P. D., Setzer C., and Wie B. GPU-accelerated 3D modeling and simulation of a blended kinetic impact and nuclear subsurface explosion. *AIAA*, (2013-4548), 2013.

[4] Zimmerman B. J. and Wie B. Computational validation of nuclear explosion energy coupling models for asteroid fragmentation. *AIAA*, (2014-4146), 2014.

[5] Zimmerman B. J. and Wie B. A GPU-accelerated computational tool for asteroid disruption modeling and simulation. *AAS*, (15-568), 2015.

[6] Wie B., Zimmerman B. J., and Premaratne P. A non-nuclear MKIV (multiple kinetic-energy impactor vehicle) system for dispersive pulverization of small asteroids with short warning times. *AAS*, (15-567), 2015.

[7] Wood L., Hyde R., Ishikawa M., and Teller E. Cosmic bombardment v: Threat object-dispersing approaches to active planetary defense. Report, Lawrence Livermore National Laboratory, Livermore, CA, 1995. Proceedings of the Planetary Defense Workshop.

[8] Prater R. F. *Hypervelocity impact - material strength effects of crater formation and shock propagation in three aluminum alloys*. PhD thesis, AFML, 1970.

[9] MacCormackR. W. The effect of viscosity in hypervelocity impact cratering. *Journal of Spacecraft and Rockets*, 40, 1969.

[10] Cloutman L. D. SPH simulations of hypervelocity impact. Report, Lawrence Livermore National Laboratory, 1991.

[11] Leveque R. J. *Finite-Volume Method for Hyperbolic Problems*. 2002.

[12] Liu Y., Vinokur M., and Wang Z.J. Discontinuous spectral difference method for conservation laws on unstructured grids. *J Comput Phys*, 216:780–801, 2006.

[13] May G. and Jameson A. A spectral difference method for the Euler and Navier-Stokes equations. *AIAA*, (2006-304), 2006.

[14] Sun Y. and Wang Z.J. High-order multidomain spectral difference method for the Navier-Stokes equations on unstructured hexahedral grids. *J Comput Phys*, 2:301–333, 2007.

[15] Wang Z. J. and Liu Y. The spectral different method for the 2d Euler equations on unstructered grids. *AIAA*, (2005-5112), 2005.

[16] Zimmerman B.J., Wang Z.J., and Visbal M. High-order spectral difference: verification and acceleration using GPU computing. *AIAA*, (2013-2941), 2013.

[17] Karni S. Multicomponent flow calcuations by a consistent primitive algorithm. *J Comput Phys*, 112:31–43, 1994.

[18] Abgrall R. How to prevent pressure oscillations in multicomponent flow calculations: A quasi conservative approach. *J Comput Phys*, 125:150–160, 1996.

[19] Shyue K-M. An efficient shock-capturing algorithm for compressible multicomponent problems. *J Comput Phys*, 142:208–242, 1998.

[20] Shyue K-M. A fluid-mixture type algorithm for compressible multicomponent flow with Mie-Grüneisen equation of state. *J Comput Phys*, 171:678–707, 2001.

[21] Allaire G., Clerc S., and Kokh S. A five-equation model for the simulation of interfaces between compressible fluids. *J Comput Phys*, 181:577–616, 2002.

[22] Gryngarten L. D. and Menon S. Shock-bubble interaction simulations using a new two-phace discontinuous Galerkin method. *AIAA*, (2011-294), 2011.

[23] Zienkiewicz O.C. and Tayler R.C. *The Finite Element Method the Basics*, volume 1. 2000.

[24] Toro E.F. *Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction*. Springer, London/New York, 2009.

[25] Shu C.W. Total-variation-diminishing time discretizations. *SIAM J Sci Stat Comput*, 9:1073–1084, 1988.

[26] Stowe R. L. Strength and deformation properties of granite, basalt, limestone, and tuff at various loading rates. Report, Corps of Engineers, 1969.

[27] Leveque R. J. *Numerical Methods for Conservation Laws*. 1992.

[28] Roe P.L. Characteristic-based schemes for the Euler equations. *Rev in Fluid Mech*, 18:337–365, 1986.

[29] Johnsen E. and Colonius T. Implementation of weno schemes in compressible multicomponent flow problems. *J Comput Phys*, 219:715–732, 2006.

[30] Rider W. J. An adaptive Riemann solver using a two-shock approximation. *Comput Fluids*, 28:741–777, 1999.

[31] Saurel R., Le Metayer O., Massoni J., and Gavrilyuk S. Shock jump relations for multiphase mixtures with stiff mechanical relaxation. *Shock Waves*, 16:209–232, 2007.

[32] Saurel R. and Abgrall R. A simple method for compressible multifluid flows. *SIAM J Sci Stat Comput*, 32:1115–1145, 1999.