# SIMULATION OF DETONATION PROBLEMS WITH THE MLS GRID-FREE METHODOLOGY

Jin Yao
Lawrence Livermore National Laboratory
Livermore, CA 94551

M. E. Gunger, D. A. Matuska*
General Dynamics Ordinate and Tactical Systems
Niceville, FL 32578

The Moving Least Squares (MLS) grid-free rezone method, a simple, flexible finite difference method for solution of general continuum mechanics problems, especially detonation problems, is proposed in this paper. The spatial points that carry time dependent data are distributed in space in such a way that provides nearly uniform spacing of points, accurate presentation of boundaries, easy variation of resolution and arbitrary reorganization of the computational domain. Local finite difference operators are obtained with simple MLS differentiation. There is no specific topological or geometrical restriction of the distribution of data points. Therefore this method avoids many drawbacks of the traditional methods. Because of its flexibility, it can be used to simulate a wide range of mechanics problems. Because of its simplicity, it has the potential to become a preferred method.

Most traditional computational continuum mechanics (CCM) methods, from a Smooth Particle Hydrodynamics (SPH) view, can be considered as special cases of grid-free methods of specific kernel functions. Such a generalization allows the development of a unified grid-free method that can represent most finite difference methods by switching the kernel functions. The flexible management and ease of coding make such a unified code attractive for many applications.

## INTRODUCTION

In simulation of problems in mechanics, it is necessary to describe the system with a finite set of data points. Determination of positions of the data points is essential. Traditional Lagrange methods allow the data points to move with the material. Usually these data points must obey certain topological constraints. The material can be traced easily. However it is difficult for a Lagrange method to deal with large, arbitrary deformation.

An Eulerian method, in which the data points are fixed in space, can be fast and accurate. Eulerian methods usually require a grid on which to define simple finite difference approximations of spatial derivatives. Because the regularity of the grid has to be maintained, an Eulerian method is difficult to manage when local high resolution is mandatory. Deletion of unnecessary portions of the computation domain is not an easy task to manage with a grid method. Much effort has been spent on finding reasonable ways of tracking the grid topology.

In this paper, we propose a methodology that we call the Moving-Least-Squared (MLS) grid-free rezone method. It is an extension of the MLSPH method by Gary Dilts[1]. This method is appropriate for the calculation of general, complex physical processes such as detonation and shock propagation. We abandon the concept of a finite element and grid. Instead, the spatial points that carry computational data are arbitrarily distributed in space. We derive a set of finite difference schemes with simple MLS differentiation. The accuracy of these difference equations is comparable to traditional methods. The boundaries can be represented with data points and the treatment of interfaces between different objects is easily implemented.

There are two essential phases associated with the MLS grid-free method. A data point distributor, and a set of MLS algorithms. The data point distributor arranges the data point set in such a way that the data points are regularly spaced; boundaries are simply represented with data points. MLS differentiation is used to derive the local spatial finite difference operators. MLS interpolation is used for rearrangement of points when needed. The data points can be fixed in space (the Eulerian mode) or moving with the material (the Lagrange mode) and it is easy to switch from one mode to the other. Large deformations can be easily traced with the Lagrange mode without regard to topological constraints. If variable resolution or arbitrary deletion of domains is desired, the MLS grid free methodology requires no additional effort.

In a global view, the MLS grid free methodology is a simple, unified frame that consists of a class of computational continuum mechanics (CCM) methods. Many existing CCM methods can be considered as subsets with restrictions of this general, flexible method. The MLS grid-free methodology provides simple derivation of the equations of motion, explicit local error estimation, precise boundary representation with data points, and easy implementation of boundary conditions. Despite some moderate numerical complexity that is associated with the point distributor, it is a very simple method to implement and to manage.

## MLS DIFFERENTIATION

The Moving-Least-Squared method (MLS) has a close relationship with the local finite difference of spatial derivatives. To be specific, the finite difference analog of spatial derivatives based on the linear combination of point data, especially grid data, is equivalent to special cases of direct differentiation of MLS functions. By varying the weight function (the kernel), one may obtain various expressions of local spatial derivatives.

This point is very easy to show, since the MLS function interpolates any specified set of functions exactly, one may choose the function set to be $\{1, x, x^2 \ldots x^n\}$. An analytical function at $\boldsymbol{x}_j$ can be expressed as its Taylor series

$$f(x_j) = \sum_{i=0}^{n} \frac{f^{[i]}(x)}{i!}(x_j - x)^i + O(x - x_j)^{n+1}.$$

The interpolation of $f(x)$ using the MLS functions is defined as

$$f^*(\vec{x}) = \sum_j f(\vec{x}_j) f_j(\vec{x}_j - \vec{x}).$$

With the substitution of $f(x_j)$ into the MLS interpolation, it is clear that except for the very first term, all power terms will vanish till the order of $n$. Therefore $f^*(\boldsymbol{x})$ is the value of $f(\boldsymbol{x})$, plus a remainder of $O[(D\boldsymbol{x})^{n+1}/(n+1)!]$, where $Dx$ is the size of the neighborhood. By taking the $k^{th}$ derivatives of $f^*(\boldsymbol{x})$, one immediately realizes that $f^{[k]}(\vec{x} - \vec{x}_j)$ is the coefficient in front of the function value $f(\boldsymbol{x}_j)$ in a finite difference form of the $k^{th}$ derivative of $f$ at x. Similar analysis for multi-dimensional cases can also be easily shown. Mathematically speaking, any specified order of local derivatives can be obtained by *MLS* differentiation to any specified accuracy if enough neighbor points are involved.

## DATA POINT DISTRIBUTOR

The flexibility of the MLS grid-free method avoids certain constraints associated with the traditional CCM methods. We are allowed to arrange the data points in such a way that facilitates the calculation. The particle spacing is controlled in a way to:

(1). reduce the local truncation error,
(2). accurately represent the boundaries,
(3). allow easy variation of resolutions, and
(4). freely exclude unnecessary domains.

We implemented an intrinsic Huygens construction technique to identify the level sets of the signed minimum distance functions. The data points are placed evenly on the level curves. The distance between neighbor level curves is nearly uniform and equal to the particle spacing. The boundary of a region that requires special resolution can be treated as a level set in which the particle spacing is equal to the required resolution. After all of the level curves are filled with data points, one can delete unusable points arbitrarily.

## PARTICLE AND VORONOI CELL

We assume that each interior data point is associated with a definite volume - the volume of its Voronoi cell. The sum of these volumes represents the physical volume occupied by the material. For a single data point, the mass, momentum and energy it carries can be defined as well as its volume. Thus it is sensible to call these data points ‘particles’. This definition is geometrical compared to the numerical definition of particles in SPH. Because the boundary normal vector of a Voronoi cell is trivially identified, it is easy to compute physical fluxes that enter or leave a particle when required.

## PHANTOM PARTICLES

The real material objects are bounded by phantom particles. These particles help to accurately represent the boundaries. They make each real data point an interior one. In addition, they carry environmental properties (or other specified properties) so the treatment of the boundary condition can be naturally integrated into the solution of the governing equations. Furthermore they improve the stability of the method. Last but not the least, phantom particles help to avoid extrapolation and to maintain accuracy when the solution is evaluated at a boundary.

## NEIGHBOR SEARCH

The particle system is contained within a search matrix (a regular mesh). Its mesh size is comparable to the search-length. For a particular particle $\boldsymbol{A}$, the neighbor search only involves the particles that occupy cells directly connected to the cell in which particle $\boldsymbol{A}$ resides. The effort of such a search method is evidently linear.

## BOUNDARY PARTICLES

A very simple method to detect particles on the boundary is implemented. The basic idea is that a boundary particle is an 'open' one. Let the particle of concern be defined as the origin. Then all the neighbors can be projected onto a unit circle. The maximum of the minimum span of angles formed by two neighbors must be greater than some critical value ($\pi/3$ seems to work fine) for boundary particles. To further ensure the method is well defined, the neighbor set used to determine angular spans also includes the neighbors of neighbors of the particle of concern. Our experience indicates that this simple boundary detection method is quite reliable.

## REPRESENTATION OF OBJECTS

A boundary is represented with boundary particles. We first detect the particles at the boundary, and then carefully link them to form curves. Consistency between the numbering of boundary particles and the arc-length is essential. The boundary curves are considered to be level-set curves of signed minimum distance of function value zero. Interior particles are positioned in a similar fashion on level curves of negative function values.

## ORDER BOUNDARY PARTICLES

Starting from an arbitrarily selected boundary particle, one needs to find an effective algorithm to determine the next boundary particle, till the starting particle is again found. The basic approach we used is to find the interior particle nearest to a known boundary particle, then apply the right hand rule to determine the next boundary particle to be linked. Of course we first consider all the neighbors. When the boundaries have sharp turns, multiple candidates may be found. We take the right most one to fulfill the right hand rule.

A check to eliminate misidentifications is to examine the original particle number on boundaries. If the natural numbering of boundary particles is violated, we use the original order of boundary particles.

## TREATMENT OF BOUNDARIES

Currently we treat an interface particle as an interior particle, if this particle is under compression or is moving toward the interface. An interface particle is treated as a free boundary particle if it is in tension and is moving away from the interface. This simple treatment provides acceptable results.

In the treatment of free boundary particles, we use phantom particles to carry environmental pressure. We also interpolate the environmental pressure with the distance from the center of interior particles to the boundary as if the pressure is applied right at the boundary. This helps to keep the boundary smooth.

## A MONOTONIC ESTIMATOR OF THE SECOND ORDER

The natural neighbors of a given point ***A*** form a convex polygon ***P***, with ***A*** as an interior point of ***P***. From the theory of linear programming, ***A***'s coordinates can be expressed as a linear combination of the coordinators of the corners of ***P***, or ***A***'s natural neighbors in the format

$$\vec{r}_A = \sum_j l_j \vec{r}_j, \text{ also} \sum_j l_j = 1, \text{ and } l_j > 0.$$

Here $j$ is the index of the neighbors. For an analytical function, its value at ***A*** can be interpolated with exactly the same coefficients using the neighbor values with an error that is at most second order. This is easy to confirm by performing a Taylor's expansion at ***A***. *Monotonicity* is achieved with *positiveness* of the coefficients.

In general, the neighbors found are not necessarily the set of natural neighbors. Thus there is no convexity available. However, one may project the neighbors onto the surface of a unit sphere centered at ***A*** and apply geometric similarity to obtain a similar *second order monotonic estimator*. This estimator is useful in the convection phase of the Eulerian mode.

## MLS GRID FREE METHOD

The MLS grid free method can be outlined as follows. The set of points that carry data does not have to obey any local topological constraint. A set of phantom points can be added to this point set. One first searches for the neighbors of each point. Then one rewrites the equations by evaluating the derivatives with MLS differentiation. The time derivative at a point can be estimated from its neighbors with a second order monotonic estimator for monotonicity. The time integration may be done with a numerical integrator or a Runge-Kutta like method.

The boundaries can be represented with boundary particles. Phantom particles ensure every data point is an interior point, and provide boundary conditions. Rezoning may be done to maintain particle spacing as needed. For rezone to be effective, it is essential to correctly identify boundary particles and to connect them in the proper order.

## TRADITIONAL METHODS

Traditional finite difference methods with a grid are degenerate cases of the MLS *grid free* methodology. For example, with a uniform grid in two-dimension, let point $\boldsymbol{A}_1$ to be located at *(0,0)*, its natural neighbors will be located at {$\boldsymbol{A}_2$, $\boldsymbol{A}_3$, $\boldsymbol{A}_4$, $\boldsymbol{A}_5$} = *{(0, 1), (1, 0), (0, -1), (-1, 0)}*. By taking a constant weight function, one finds the MLS functions

$$f_1 = 1 - x^2 - y^2,$$
$$f_2 = \frac{x}{2}(x+1), f_3 = \frac{y}{2}(y+1),$$
$$f_4 = \frac{x}{2}(x-1), f_5 = \frac{y}{2}(y-1).$$

One sees they interpolate $\{1, x, y, x^2, y^2\}$ exactly. To interpolate the function *f(x,y)* using MLS functions, one has

$$f^*(x,y) = \sum_{j=1}^{5} f(A_j) f_j(x,y)$$
.

By taking the Laplacian of the function $f^*(x, y)$, one finds

$$\nabla^2 f^*(x,y) = -4f(A_1) + f(A_2) + f(A_3) + f(A_4) + f(A_5).$$

It is exactly the difference *Laplacian* operator on a uniform grid. One may also obtain the first derivatives by MLS differentiation only once. At (*0,0*), we obtain the central difference format.

## A UNIFIED METHOD

We indicate that most CCM methods can be expressed as SPH methods with specific kernels. Thus, desired CCM methods may be implemented with a unified grid-free code by simply changing kernels and data point distributions. The resulting unified code can be easily managed and the programming work is eased. Furthermore the transformation between different CCM systems can be easily done with the MLS rezoning method.

## REZONING

Rezoning is performed mainly to maintain particle spacing, so as to improve stability and to maintain the accuracy of the spatial derivatives. Larger time steps can be used after rezoning because the particle spacing is regular. Nonphysical material intrusion can also be prevented if the rezoning is done before the intrusion occurs. When large particles and small particles meet, it is difficult to keep symmetry in the neighbor search (i.e., if *a* is *b*'s neighbor, then *b* is *a*'s neighbor too) with particle diameters as the search length. To prevent lose of accuracy by lose of symmetry, a rezone is appropriate. MLS interpolation during rezoning conserves mass, volume, momentum and energy of the system with at least second order local accuracy [2].

## LINEAR VORONOI ALGORITHM

We calculate the Voronoi cells of a particle from its neighbors. By definition, the effort of this method is of $O(n)$ for an $n$-particle system. Compared to other optimized Voronoi solvers, this method is the fastest when n is large, provided that the neighbor search has been done in advance. However, the cost of the neighbor search method we use is also of the order $O(n)$.

We describe this method with the construction of a three-dimensional Voronoi cell. We assume a given interior particle ***A*** has *M* neighbors and all ***A***'s natural neighbors are included. The first step is to find the closest neighbor of ***A***, to define the first facet. Next, one finds the smallest distance from the origin to all possible edges on this facet; the first edge is then determined. The third step is to find the smallest distance from ***A*** to all of the possible vertexes on the first edge, so to determine the first vertex on this facet. The neighbor responsible for this vertex also defines the next edge. To determine the next vertex, one finds the smallest distance from all of the possible vertexes on this edge to the last vertex. Convexity is checked by excluding all the vertexes that are located on the opposite side of the facet defined by the last neighbor point that corresponds to the last edge. One repeats this procedure until the first vertex is found again. By now the first facet is completely determined. We have also identified the particles corresponding to the next level of facets (and their first edge, and the first and last vertices) at this point. A single quick-sort routine is used to perform the sorting, and the cost is of the order *O(lnM)*. A do-loop of the algorithm described above is applied to all the newly found facets until no new facet is found. The connectivity ensures that no facet is left. The entire connecting list is thus obtained naturally and the method is optimized.

A three-dimensional Voronoi cell that is obtained with this approach is shown in figure 1. The two-dimensional Voronoi algorithm has a similar but much simpler implementation. Mathematically speaking, this approach is effective in the *m*-dimensional case.

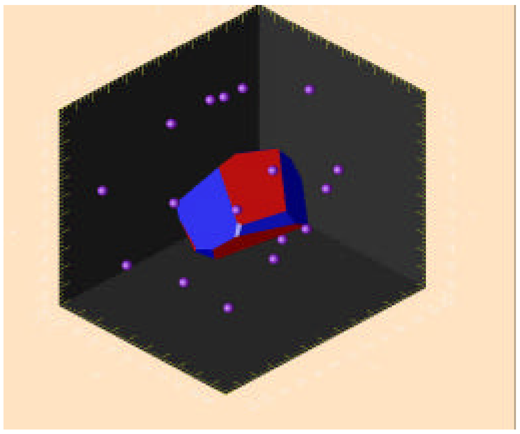

**FIGURE 1. A THREE-DIMENSIONAL VORONOI CELL.**

## INTRINSIC LEVEL-SET METHOD

The particles are positioned on level curves of signed minimum distance to the boundary of an object. The distance between adjacent level curves is equal to the desired local particle size. A Huygens construction is used to define these level curves. The basic idea is to calculate the distance from mesh points to the boundary, then to determine the curves of specific level-set values. To minimize the cost of calculation, the values of the signed minimum distance function are calculated only for those mesh points close to a known level curve, to determine the next level curve. The loop starts from the zero$^{th}$ level-set, the boundary. Exact positioning of curves is not important except for the boundary. The distance between level curves can be gradually increased to reduce the number of points. After all particles have been positioned, by calculation of Voronoi cells, the volume and mass center of each interior particle is then exactly determined. The change of connectivity is automatically done with the level-set method.

## LEVEL OF NEIGHBORS AND THE THREE-DIMENSIONAL SURFACE

Although we have only implemented the two-dimensional algorithm, we are quite confident that a similar algorithm in three dimensions can be implemented. Three-dimensional surfaces can also be represented with boundary points (or equivalently, facets). We use a simple looping algorithm to arrange the boundary points. The loop is done for the 'level of neighbors'. A given particle is its level zero neighbor. Its immediate neighbors are its first level of neighbors. The neighbors of the *n*th level are the immediate neighbors of the (*n* - *1*)th level neighbors. Particles do not change their level once it has been determined. A particle only connects to its immediate neighbors on the surface. The connection list then is naturally obtained, as the neighbors of any specific point are determined in the loop. No points will be left uncounted. Such a method can also be used to propagate three-dimensional surfaces according to intrinsic geometric laws, because the local geometric features of the surface are well defined with neighbors. When a method to arrange data points nearly uniformly on three-dimensional surfaces is given, a rezone algorithm similar to the two dimensional algorithm described in this paper can also be implemented. We suggest a Huygens-like construction algorithm to determine level-set curves on the surface and put points evenly on the level curves, similar to what we have done for the two-dimensional method.

## TREATMENT OF FRACTURES IN TWO-DIMENSIONS

In the two-dimensional case, with the specific method we used to position particles, it is a simple task to detect and deal with fractures. The idea is based on the consistency between the numbering of boundary particles and continuity of the arc-length. When fracture occurs, the continuity of arc-length is broken. Interior particles become boundary particles and the original numbering of boundary particles with the order of natural numbers is lost. It is very easy to determine where the fracture occurred. In the case of a material fracturing into two pieces, the natural numbering of the particles on the original boundary becomes naturally numbered on two separate closed curves. The fractured region is where the disconnection of natural numbering of boundary particles occurs. This helps to accurately determine the newly formed pieces. The reliability of this method is evident by nature.

## BURN MODELS

Various burn models can be easily implemented with the MLS grid-free method. Currently our code includes the following burn models

1). *Programmed Burn*: With given locations and ignition time of the ignition points and the normal detonation velocity (CJ velocity, say), it is easy to calculate the detonation arrival time at each explosive particle confined in a convex region by calculating the distance from a particle to the closest initiating point. When the region of the explosive is not convex, the burn time can be determined with level-set curves.

2). *Neighbor Burn*: It is assumed that only burning neighbors can ignite an explosive particle. When a particle has not been ignited but one or more neighbor particles has started to burn, the burn time of this particle then can be calculated by the distance of this particle to its burning neighbors, divided by the local detonation velocity. Of course the smallest burn time is selected. To be exact, the burn time of the particles closest to the spark is determined with their distance to the spark. After that, in most cases, at every time step there shall be more than two burning neighbors of an unburned particle, the burn time of this particle is determined with the burn times of two burning particles with a geometric method. If there is only one burning neighbor, the burn time is determined directly by the distance between the two particles. The burning time is dynamically determined. For programmed burn, this burn velocity is constant and is equal to the detonation velocity. However if the local properties are not uniform, a variable detonation progression rate can be calculated. When there are inert particles embedded in the material, the distribution of inert particles affects the burn times. The reactive particles that are enclosed by inert particles will not burn. The neighbor burn model is appropriate in this case and the programmed burn will not work.

3). *Greek-Fire Model*: This model is used to deal with the ignition of crushable explosive materials. It is assumed that the explosive material gets crushed in dynamical processes when the elastic deformation exceeds certain limits. The release of the stored elastic energy after crushing gasifies a small amount of explosive and the gasified material starts to react according to an Arrhenius law. This reaction, coupled with other physical processes, may cause local pressure to increase and accelerate the reaction. When the gasified explosive is burnt, a pressure dependent law consistent with experimental data determines the local reaction rate of the solid. Ignition may occur if the local pressure is high enough, otherwise it may die and the detonation process ends. We have simulated ignition of detonations using this model in problems that involve friction and sudden compression.

Other burn models can also be easily implemented. For example, since the differential operator can be easily represented with the MLS algorithm, it is a relatively easy task to solve the level-set equation for the propagation of detonation surface given the law of surface propagation (from DSD theory, say) to better determine the burn time.

## AN EXAMPLE OF DETONATION SIMULATION

*The Viper-Jet*: Figure 2 shows a snap shot of a shaped charge calculation at 30.5 µsec. About 40,000 data points were used with a programmed burn model. The velocity at the tip of the viper is only 0.3% off the experimental value. Figure 3 shows its agreement with the experimental mass-velocity distribution (the red line). The calculation time is about 7 hours on a 0.7 GHZ Linux station. Much time is taken by the calculation of Voronoi cells in the rezone process. The Voronoi algorithm can be omitted by rewriting the governing equations to avoid explicit appearance of volume (and mass) of particles. A better rezone criterion may also be considered to reduce the frequency of rezoning.

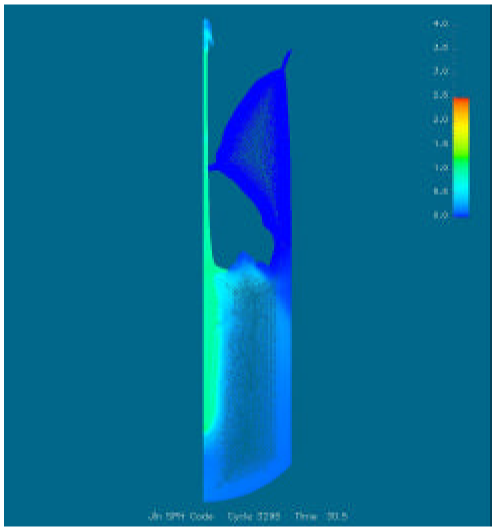

**FIGURE 2. A SNAP SHOT OF THE SHAPED-CHARGE CALCULATION.**

## CONCLUSIONS

The MLS grid-free method, as described, can be viewed as a general, arbitrary, free Lagrange/Euler method by nature. Its variations correspond to various CCM methods. Traditional CCM methods such as Eulerian methods on a regular grid, free Lagrange methods, FEM, SPH, and ALE can all be considered as specific cases of one unified grid-free method. Thus it is possible to develop a unified grid-free code that contains many CCM methods by including various kernel functions. Most importantly, the coding effort is minimal.

The data point distributor is designed under the geometrical requirements that data points are uniformly spaced and material boundaries are represented by data points. Local variable resolution can be accommodated with ease.

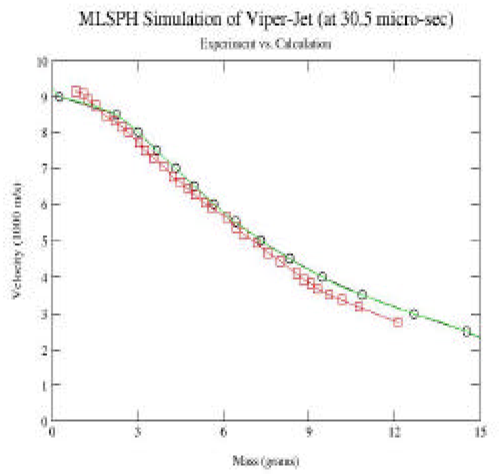

**FIGURE 3.THE MASS DISTRIBUTION.**

Particle redistribution reduces the truncation error, minimizes the number of data points, and clearly identifies material boundaries. The boundary treatment in the MLS grid- free method is natural with the employment of phantom particles that serve as environmental particles or as image data points. They also improve the accuracy and stability of the method.

We list some of the benefits of the MLS grid-free method in the following:

a). trivial deletion of domains,
b). trivial implementation of variable resolution,
c). clear definition of boundaries.
d). no diffusion or intrusion between material boundaries.
e). generalization of traditional CCM methods.

The MLS grid free implementation in three-dimensions should be only a straightforward extension of its two-dimensional case with the presence of a three-dimensional data point distributor. We have described in this paper some practical methods to construct the three-dimensional particle distributor/rezoner. However such an implementation has not been reported in the literature.

The MLS grid-free methodology has been successful so far in dealing with various problems in mechanics, especially for detonation problems. It is expected most problems can be solved with this approach because of its flexibility and simplicity. It has the potential to become a general, very powerful method in computational continuum dynamics.

## ACKNOWLEDGMENT*

This work was supported by General Dynamics Ordinance and Tactical Systems, Niceville, Florida internal research and development, and performed under the auspices of the U.S. Department of Energy by University of California Lawrence Livermore National Laboratory under contract No. W-7405-Eng-48.

## REFERENCES

[1]. Dilts, G. A. "Moving Least Squared Particle Hydrodynamics -- I. Consistency and Stability", Int. J. Numer. Mech. Eng. Vol. 44, 1999, pp.1115-1155.

[2]. J. Yao, M. E. Gunger, and D. A. Matuska, "Simulation of Shaped-Charge with MLS rezone method ", The Twelfth APS SCCM Topical Conference, Atlanta, Georgia, 2001.