17th AIAA Computational Fluid Dynamics Conference, June 7-10, Toronto, Canada

COMPARISON OF SEARCH ALGORITHMS FOR ASSESSING MAXIMUM POSSIBLE BLAST DAMAGE

Fumiya Togashi*, Rainald Löhner†, Joseph D. Baum‡, Hong Luo§, Darren Rice¶, and Shinkyu Jeong‖
School of Computational Sciences, George Mason University, MS 4C7, Fairfax, VA 22030-4444, USA

Advanced Technology Group, Science Applications International Corp., McLean, VA 22102, USA

Institute of Fluid Science, Tohoku University Sendai, Miyagi, 980-8577, Japan

Three spatial search algorithms to assess the maximum possible blast damage as a function of blast origin are compared. Damage assessment is cast as a classic optimization problem. The cost function considered is the number of windows destroyed due to a blast. The 3-D flowfields are computed using an Euler solver on optimized unstructured grids with 1-2 million elements. This allows the use of PCs for a single run, in turn enabling parallel execution of many scenarios on a network of PCs. The three methods compared comprise: a) a standard genetic optimization technique, b) a Kriging model, and c) a Finite Element approximation approach. The results indicate that the two latter models outperform the standard genetic optimization technique by a considerable margin, enabling an accurate search for locations of maximum possible blast damage with far fewer cost function evaluations. In some cases, the number of cost function evaluations (i.e. 3-D Euler runs) can be reduced by an order of magnitude.

I. INTRODUCTION

Explosions remain the most frequently used form of terror attack. They represent a low-tech, cheap, abundantly available resource that produces the desired destructive, psychological (mainly fear and rage), publicity (monopolization of news), economical (disruption of travel, commerce, investment and consumption) and political (destabilization) effects.
The amount of explosive used can vary considerably, from $10^{-1}kg$ for airplanes to $10^{4}kg$ for the more spectacular building attacks. Given the increasingly unstable international situation, as well as the abundant

*Research Scientist, ftogashi@gmu.edu
†Professor, rlohner@gmu.edu
‡Group Leader, joseph.d.baum@saic.com
§Research Scientist, hong.luo@saic.com
¶Research Scientist, darren.rice@saic.com
‖Professor

availability of explosive materials, the number of worldwide attacks has increased in recent years. A major attack ($> 10^3 kg$ of explosive) has occurred every 6-12 months.
For buildings, the physical damage includes window breakage, disruption of amenities and services (water, gas, telephone, air conditioning, etc.). and loss of structural integrity (cracks, progressive collapse).
From a civil engineering point of view, the design of structures is certified by building norms that consider the following static and dynamics loads: gravity (weight, water, contents), wind, vibration due to machinery, as well as earthquakes. Currently buildings do not require any form of certification for blast loads. This implies that most buildings have no inherent 'design for blast survivability' (e.g. in the form of a delayed or gradual collapse), and will remain highly vulnerable for the foreseeable future.
At present, the most effective means of protecting structures are given by standard low-tech options: restriction of access (safe perimeter zone with the ensuing disruption of traffic, comfort, etc.), screening of any material entering the safe zone close to the building, indoor walls and shatter-proof windows.
In order to assess the vulnerability of a given building or place, design for blast survivability, place cameras and sensors, as well as legislate building standards, it is imperative to know the maximum possible damage an explosion can cause. This problem may be cast as the optimization problem: maximize the damage, given a region where explosions can originate from. Damage in this context is defined as the number of people and property affected by the explosion.
Assuming that the amount of explosive is finite, the location of the explosion becomes the main input variable. The shocks caused by an explosion can refract, reflect and focus in many ways, particularly for architecturally interesting buildings in inner cities. Shielding effects due to walls or buildings can lead to a large variability in damage as a function of location. As a result, damage as a function of space can have many local extrema. This implies that traditional gradient-based approximation techniques will fail. For this reason, recourse is taken to search algorithms that can obtain the possibly multiple maxima of such arbitrary functions. For the present study, three such algorithms were tested:

- A traditional genetic optimization algorithm (GA);
- A genetic optimization algorithm with Kriging (GAK); and
- An approximation algorithm based on Finite Element approximation theory.

Each one of these algorithms requires a considerable number of objective function (in this case damage) evaluations. Traditionally, these were done using line of sight, table look-up techniques. While such techniques provide good estimates for direct line of sight impact, more interesting (and in some cases damaging) phenomena such as reflections, Mach stems, shock focusing and backwall pressures are not computed properly. This implies that 3-D Euler solvers should be used for realistic analysis. Recent advances in CFD codes and hardware have made it possible to perform medium-fidelity 3-D blast runs with grids up to $2 \cdot 10^6$ elements in less than 8 hours on PC platforms. While each of these runs is time-consuming, this nevertheless implies that on networks of PCs, or PC clusters, hundreds of explosion scenarios can be simulated per day.
The remainder of the paper is organized as follows: Section 2 defines the damage criterion used; Sections 3-7 describe the genetic optimization algorithm, the Kriging model, the approximation via Finite Elements, the flow solver and the evaluation of the objective function; in Section 8 examples are presented; finally, in Section 9 some conclusions are drawn and an outlook for future work is given.

II. DAMAGE CRITERION

Assessing the damage to people and property is difficult for a number of reasons. The occupancy level of a building can vary considerably depending on the time of day (daily cycle) and the particular day (yearly cycle). Even under the most rigorous construction standards, the mechanical properties of the materials comprising a building can vary considerably, and will change in time due to aging, microcrack formations, etc. For the present study, which is considered a first step towards a more comprehensive

damage assessment capability, the damage to property was not considered. This implies that a complete coupling to 3-D Computational Structural Dynamics (CSD) codes is not necessary. Rather, the injury due to window shatter was taken as the damage criterion. Window breakage was assumed to occur if either the pressure or the impulse exceeded given thresholds. As the occupancy density of buildings was not known, the total number of broken windows was taken as a measure of the damage inflicted by the blast:

$$I = \sum_{i=1}^{n_w} b_i \quad , \tag{1}$$

where n_w denotes the number of windows, and $b_i = 0$ if the window survives the blast, otherwise $b_i = 1$.

III. 3. GENETIC OPTIMIZATION ALGORITHM

Suppose we are given the optimization problem:

$$I(\boldsymbol{\beta}) \rightarrow max \quad . \tag{2}$$

In order to norm the input variables (i.e. the blast origins), a range $\beta^i_{min} \leq \beta^i \leq \beta^i_{max}$ is set for each variable defining a blast origin. In the simplest case, this can be the spatial coordinates of the blast point. An **instantiation** is then given by:

$$\beta^i = (1 - \alpha^i)\beta^i_{min} + \alpha^i \beta^i_{max} \quad , \tag{3}$$

implying $I(\boldsymbol{\beta}) = I(\boldsymbol{\beta}(\boldsymbol{\alpha}))$. By working only with the α^i, an abstract, non-dimensional, bounded ($[0, 1]$) setting is achieved, that allows for a large degree of commonality among various optimization algorithms. Given the optimization problem Eqn.(1), a simple and very general way to proceed is by copying evolution in nature: try variations of $\boldsymbol{\beta}$, and keep the ones that maximize (i.e. improve) the cost function $I(\boldsymbol{\beta})$. This class of optimization techniques is denoted as **genetic algorithms**[9,10]. Although costly, because many function evaluations are required, genetic algorithms (GAs) offer important advantages: they represent a completely general technique, able to go beyond local minima and hence suitable for 'rough' cost functions I with multiple local minima, and only require function evaluations (i.e. no gradient). Genetic algorithms have been used on many occasions in many fields[9,22,19,10]. The key elements of genetic algorithms are:

- A **fitness measure**, given by $I(\boldsymbol{\beta})$, to measure different release scenarios against each other;
- **Chromosome coding**, to parametrize the release locations given by $\boldsymbol{\beta}$;
- **Population size** required to achieve robust optimization;
- **Selection**, to decide which members of the present/next generation are to be kept/used for reproductive purposes; and
- **Mutation**, to obtain 'offspring' not present in the current population.

The most straightforward way to code the release locations into chromosomes is by defining them to be the parameters $0 \leq \alpha^i \leq 1$. An instantiation is then given by Eqn.(2). The population required for a robust selection needs to be sufficiently large. A typical choice for the number of individuals in the population M as compared to the number of chromosomes (release locations) N is: $M > O(2N)$ for large N. Given a population and a fitness measure associated with each individual, the next generation has to be determined. In order to achieve a monotonic improvement in release scenarios, a percentage of 'best individuals' c_k of each generation is kept (value used here: $c_k = O(10\%)$). Furthermore, a percentage of 'worst individuals' c_c are not admitted for reproductive purposes (value used here: $c_c = O(75\%)$). Each new individual is generated by

selecting randomly a pair i, j from the allowed list of individuals, and combining the chromosomes randomly. Many possible ways have been proposed to combine chromosomes, such as chromosome splicing, arithmetic and random pairing, etc.[9,10]. In the present case, arithmetic pairing was chosen. A random pairing factor $-\xi < \gamma < 1+\xi$ is selected and applied to all variables of the chromosomes in a uniform way. The chromosomes for the new individual are given by:

$$\boldsymbol{\alpha} = (1-\gamma)\boldsymbol{\alpha}^i + \gamma\boldsymbol{\alpha}^j \quad . \tag{4}$$

We remark that γ lies outside $[0,1]$ (a typical value is $\xi = 0.5$). This is required, as otherwise the only way to reach locally outside the chromosome interval given by the pair i, j (or the present population) is via mutation, a slow and therefore expensive process. On the other hand, a population that is not modified continuously by mutations tends to become uniform, implying that the optimization may end in a local minimum. Therefore, a mutation frequency: $c_m = O(0.25/N)$ has to be applied to the new generation, modifying chromosomes randomly.
Possible locations for explosions may be restricted by buildings, walls, barriers, etc. The region of possible locations is defined via a triangulation. A check is carried out for any new location selected by the genetic algorithm to see if it falls inside this triangulation. If this is not the case, the individual is rejected.
Summarizing, the basic steps required per generation for genetic algorithms are the following:

Ga1 Evaluate the fitness function $I(\boldsymbol{\beta}(\boldsymbol{\alpha}))$ for all

individuals;

Ga2 Sort the population in ascending (descending)

order of I;

Ga3 Retain the c_k best individuals for the next

generation;

Ga4 `while`: Population incomplete

- Select randomly a pair i, j from c_c list

- Obtain random pairing factors $-\xi < \gamma_k < 1 + \xi$

- Obtain the chromosomes for the new individual:

$\boldsymbol{\alpha} = (1-\gamma)\boldsymbol{\alpha}^i + \gamma\boldsymbol{\alpha}^j$

- Limit the values of $\boldsymbol{\alpha}$ to be in admissible range:

$\boldsymbol{\alpha} = max(0, min(1, \boldsymbol{\alpha}))$

- If the location is outside the allowed region:

reject the new individual

`end while`

IV. KRIGING

Suppose a deterministic function of m variables at n points has been evaluated. A sampled point i is denoted by the m-dimensional vector $\mathbf{x}^i = (x_1, ..., x_m)$.
The present Kriging model is written as follows:

$$y(\mathbf{x}) = \mu(\mathbf{x}) + Z(\mathbf{x}) \quad , \tag{5}$$

where $y(\mathbf{x})$ is the associated function value, $\mu(\mathbf{x})$ denotes a constant global model and $Z(\mathbf{x})$ is a local deviation from the global model. The local deviation at a sample point i is expressed using a stochastic process. The trend of the stochastic process is estimated by interpolating the sample points. The Gaussian random function is used for the interpolation. In the present approach, we assume the correlation between $Z(\mathbf{x}^i)$ and $Z(\mathbf{x}^j)$ is related to the distance between the two corresponding points. However, we do not use the Euclidean distance because the distance weights all the variables equally. Rather, the following special weighted distance is used:

$$d(\mathbf{x}^i, \mathbf{x}^j) = \sum_{k=1}^{m} \theta_k \left| \mathbf{x}_k^i - \mathbf{x}_k^j \right|^2 \quad ; \quad \theta_k \geq 0 \quad , \tag{6}$$

where θ_k denotes a correlation parameter that indicates how local the predictor is. The unknown parameter that is required to construct the Kriging model is θ. This unknown parameter can be estimated by maximizing the following function:

$$Ln(\hat{\mu}, \hat{\sigma}^2, \theta) = -\frac{n}{2}\ln(2\pi) - \frac{n}{2}\ln(\hat{\sigma}^2) - \frac{1}{2}\ln(|R|)$$
$$-\frac{1}{2\hat{\sigma}^2}(y - \mathbf{1}\hat{\mu})' R^{-1} (y - \mathbf{1}\hat{\mu}) \quad . \tag{7}$$

This function is called likelihood function. In this equation $\hat{\mu}$ is the estimated value of μ, $\hat{\sigma}^2$ is the estimated variance, R denotes the $n \times n$ matrix whose (i, j) entry is $Corr\left[Z(\mathbf{x}^i), Z(\mathbf{x}^j)\right]$, $\mathbf{1}$ denotes a unit vector and r denotes the n-vector of correlations between $Z(\mathbf{x})$ and $Z(\mathbf{x}^i)$ (i is the previous sampled point). Namely, i-th element of r is $r_i(\mathbf{x}) \equiv Corr\left[Z(\mathbf{x}), Z(\mathbf{x}^i)\right]$. Given θ, the values of $\hat{\mu}$ and $\hat{\sigma}^2$ can be obtained by the following equations:

$$\hat{\mu} = \frac{\mathbf{1}' R^{-1} y}{\mathbf{1}' R^{-1} \mathbf{1}} \quad , \tag{8}$$

$$\hat{\sigma}^2 = \frac{(y - \mathbf{1}\mu)' R^{-1} (y - \mathbf{1}\mu)}{n} \quad . \tag{9}$$

In this study, Genetic algorithms (GAs) are used for maximizing function.
The Kriging predictor is then written as follows:

$$\hat{y}(x) = \hat{\mu} + r' R^{-1} (y - \mathbf{1}\hat{\mu}) \quad . \tag{10}$$

The derivation of this equation in detail can be found in Sacks et al.[20]. The distance from the sample points largely affects the accuracy of the prediction value. Namely, if an estimated point is closer to sample points, the prediction should be more accurate. This intuition can be written as follows:

$$s^2(x) = \hat{\sigma}^2 \left[1 - r' R^{-1} r + \frac{(1 - \mathbf{1}R^{-1}r)^2}{\mathbf{1}R^{-1}\mathbf{1}} \right] \quad . \tag{11}$$

$s^2(x)$ is the mean squared error of the predictor. $\sqrt{s^2(x)}$ is called a root mean squared error (RMSE) that indicates the uncertainty at the estimate point.

Once the approximation model is generated as mentioned above, the optimum point can be explored with an arbitrary optimizer on the model. However, there is a possibility of missing the global optimum due to the uncertainty at the predicted point. The concept is expressed in the EI criterion[11]. The EI for minimization is written as follows:

$$E\left[I(x)\right] = (f_{min} - \hat{y})\Phi\left(\frac{f_{min} - \hat{y}}{s}\right) + s\phi\left(\frac{f_{min} - \hat{y}}{s}\right) \quad . \tag{12}$$

In this equation, f_{min} is the minimum value among sampled values, Φ and ϕ are the standard distribution and normal density. By choosing the maximum EI point, the robust exploration of the global optimum and the improvement of model can be achieved.

V. APPROXIMATION VIA FINITE ELEMENTS

The task of finding the maximum (or a series of maxima) of a function defined in space can be considered a corollary to the task of approximating the function. One simple way to approximate the damage function in space is via an unstructured mesh of triangular finite elements. The idea is to start from a relatively coarse mesh that reflects the spatial scales of the damage function. These spatial scales are linked to building geometries, and can therefore be specified in a straightforward way. In a first step, the damage function for all points of this so-called base mesh is evaluated. Thereafter, the edge with the likeliest highest error of interest is obtained. A new sampling point is introduced at the mid-point of this edge, the mesh is rearranged, and the actual damage function at this new location is evaluated. The procedure is recursive, and the aim is to achieve the highest likelihood of maxima for damage with the least number of sampling points. The algorithm hinges on the concept of 'highest error of interest'. The actual approximation error ϵ_u on an edge can be computed by evaluating the gradients of the damage function at the points of the mesh, and then evaluating the resulting Hermite polynomial at the midpoints of edges (see Figure 1).

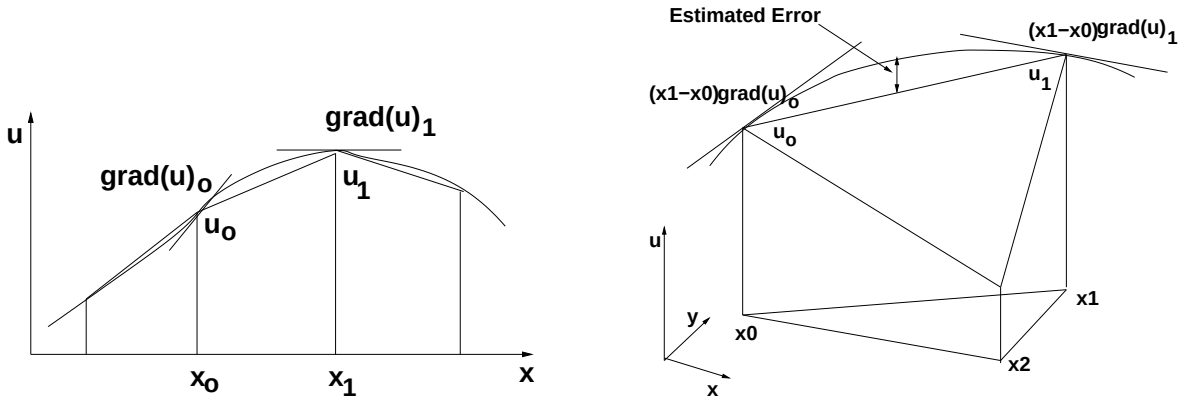


Figure 1 Estimation of Approximation Error

On the other hand, there is no need to sample more points in regions where the damage function is low. Large elements/edges may miss local maxima, and therefore should be emphasized in the 'highest error of interest'. The following measure was chosen to select the edge with the highest error of interest:

$$\epsilon = h^p |u|^q |\epsilon_u| \quad , \tag{13}$$

where h denotes the edge size. Typical values for p, q are $p = 1$, $q = 1$.

VI. FLOW SOLVER

The flow solver used to predict the pressure loads on buildings due to an explosion is FEFLO. FEFLO was conceived as a general-purpose CFD code based on the following general principles:

- Use of unstructured grids (automatic grid generation and mesh refinement);
- Finite element discretization of space;
- Edge-based data structures for speed;
- Optimal data structures for different architectures;
- Bottom-up coding from the subroutine level to assure an open-ended, expandable architecture.

The code has had a long history of relevant blast applications[12, 1, 13, 2, 18, 3, 4, 5, 14, 6, 7, 15, 16, 8]. For the prediction of blast loads, the FEM-FCT solvers[12] are used, as they offer the best compromise of accuracy and speed for this class of problems.

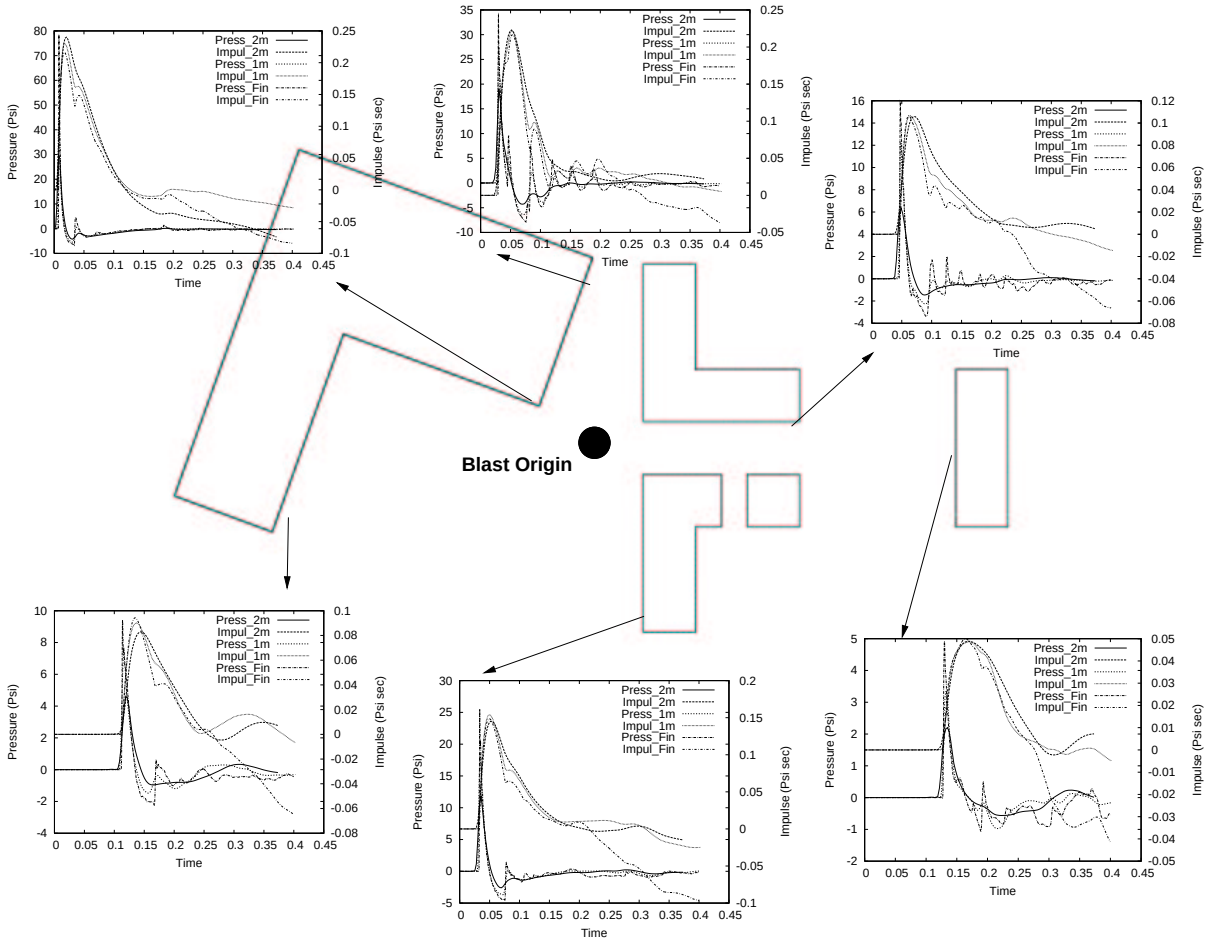


Figure 2 Overview of Overpressure and Impulse at Selected Locations

Given that any of the spatial search algorithms described above requires a considerable number of 3-D runs, and that typical supercomputers are hard to access and in many cases unreliable, a study was conducted

to assess the accuracy that is achievable on mass-market PCs for this class of problems. A series of blast calculations for typical, generic building configurations were conducted. The results obtained for coarse grids (resolution approximately $h = 1\ m$, number of elements $O(10^6)$), medium (number of elements $O(10^7)$) and fine grids (number of elements $O(10^8)$) were compared for many relevant stations. The coarse grids were all run on PCs, whereas the medium and fine grids had to be run on typical supercomputers (here: SGI-O3900). Figure 2 shows one of the calculations carried out. The rather surprising result is that for the medium grids, peak pressures and impulses coincided within 20% for almost all stations, and in most cases were even closer. For further details, see Löhner et al.[17].
By carefully selecting the appropriate element sizes, the number of elements required for a typical run can be held to less than $2 \cdot 10^6$, allowing for PC runs that take less than 2 hours. The element sizes used to achieve such grids are as follows:

- The farfield grid size is of the order of $h = 4 - 5\ m$;
- The grid size between the blast origin and any building surface facing the blast is of the order of $h = 1 - 2\ m$;
- The grid size in the region close to the blast origin is of the order of $h = 0.2 - 0.5\ m$.

The runs are initialized from detailed 1-D and 2-D axisymmetric runs. An attempt is made to carry out these runs as far as possible in time and space. Interpolating from a 'large' 1-D or 2-D axisymmetric region allows for significantly larger element sizes close to the blast origin, with the associated increase in element size, increase in allowable timestep, reduction in the total number of elements required and overall reduction of CPU requirements.
The run is stopped when the maximum pressure difference in the region comprised by the bounding box of all building surfaces and the blast origin falls below a preset tolerance. This tolerance depends on the particular setting of the problem (materials, window types, etc.), but is typically of the order of 1/10th of the atmospheric pressure.
At all windows, so-called station time history points are defined. The flow variables at these points are recorded as the flowfield is advanced in time. This data then serves as the input to other codes that assess if the window has been shattered.

VII. EVALUATION OF THE OBJECTIVE FUNCTION

Given an arbitrary blast origin $\mathbf{x}$, the evaluation of the objective function requires the following steps:

- Modify the grid generation file to obtain proper resolution in the regions close to the blast origin;
- Generate a new 3-D volume grid;
- Modify the input files defining the location of the blast origin to interpolate the output from 1-D or 2-D axisymmetic initialization runs to 3-D;
- Calculate the flowfield based on the new mesh and input;
- Filter out overpressure and impulse data from the time history points;
- Evaluate the number of windows destroyed.

As the evaluation of the objective function can be carried out on an individual processor, the evaluation of many objective functions becomes an embarrassingly parallel process. In fact, the communication overhead is so extremely low (two numbers are passed in, one number is passed out), that even a very rudimentary internet connection will suffice.

The parallelization was carried out for the network of PCs at our CFD laboratory. As is the case in many office settings, these PCs are underutilized most of the time (particularly at night), offering an ample resource of CPU time. The procedure for parallelization is as follows:

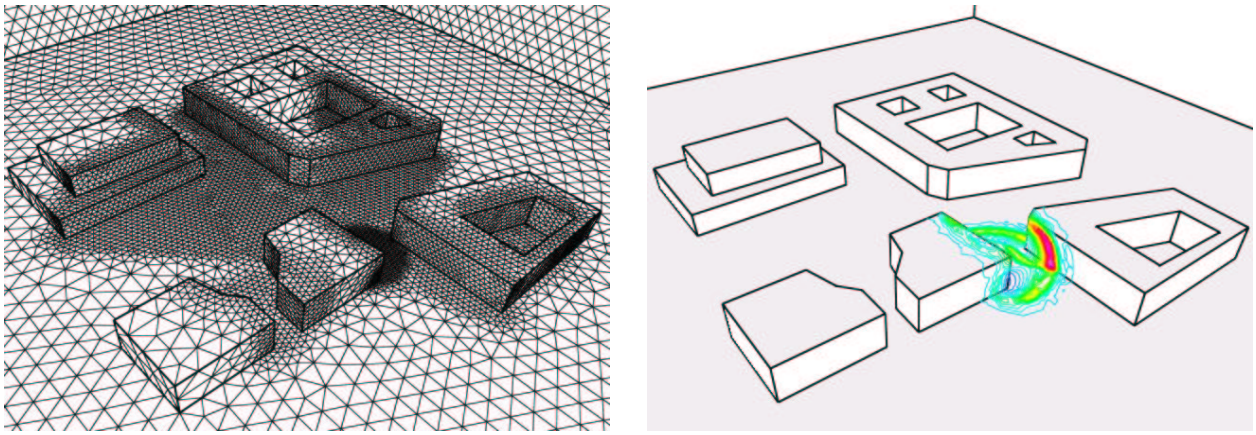
- For each objective function to be evaluated the local host ships the computational data required to the remote hosts;
- The objective functions of each individual are evaluated at the local and remote hosts;
- The objective functions obtained are then collected from the remote hosts by the local host.

The shipment of the computation command and data to multiple PCs is executed using the Perl programming language. One of the Perl modules, "net::ssh::perl", "ncftp", "tar" and various other unix commands are executed on the local and remote hosts. The program development is relatively easy using these methods.

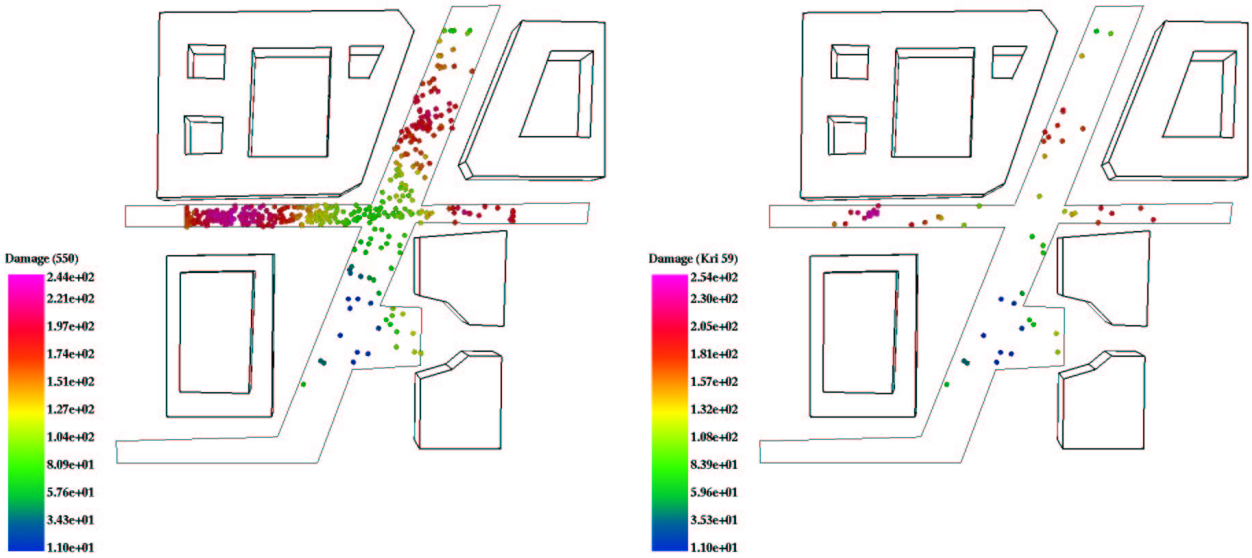
VIII. EXAMPLES

We include two examples to demonstrate the proposed methodology.

8.1 City Configuration: The example considered is that of several blocks of a generic city. The surface mesh and surface pressure obtained for a typical run are shown in Figures 3a,b. Note the increased mesh resolution close to the blast origin and between the blast origin and surfaces facing it. The grids generated for this case had approximately $0.4 \cdot 10^6$ elements and each run took approximately 2 hours of CPU time on PCs (2.0-3.0GHz, 1.0-2.0Gbytes, LinuxOS, Intel Compiler) networked via ethernet.



Figures 3a,b Typical Surface Mesh and Pressure



Figures 3c,d Results from Genetic Algorithm and Kriging Model

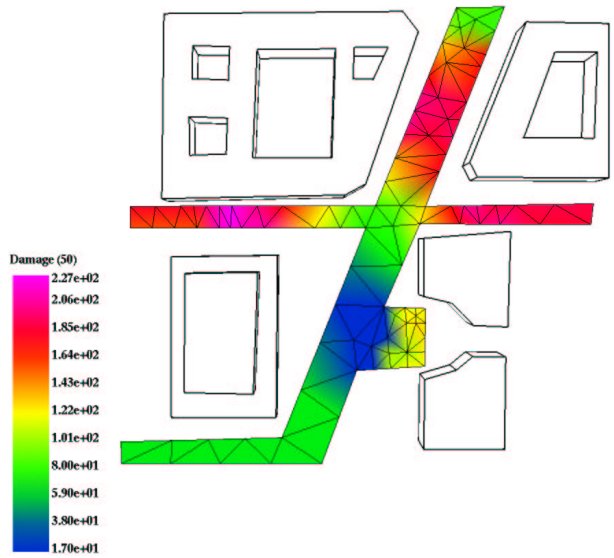


Figure 3e Results from FE Approximation

Figures 3c-e show the results obtained from the GA after 11 generations with 50 individuals (550 runs), the Kriging model (59 runs), as well as the FE approximation (103 runs). Note that all of the algorithms eventually obtain the right maximum, located at the left of the horizontal street. The Kriging model yields a remarkable reduction in the number of required runs. In comparison to the GA, the FE approximation requires a factor of 1:5 less, and obtains fairly accurate bounds for the uncertainty of the results.

8.2 Large Office Building: The example considered is that of a generic large office building. The configuration is shown in Figure 4a. The surface pressures for a typical run are shown in Figure 4b. For each run of this case, an initial grid, with increased mesh resolution close to the blast origin was generated. This solution was initialized from 2-D axisymmetric data, and computed for $t_1 = 0.004$ *sec.*

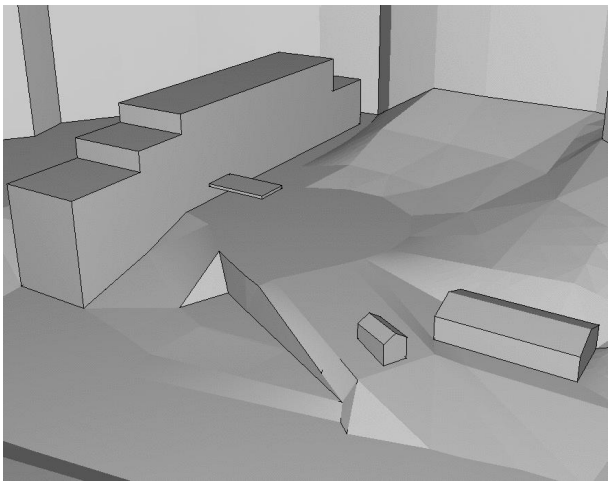


Figure 4a Computational Domain

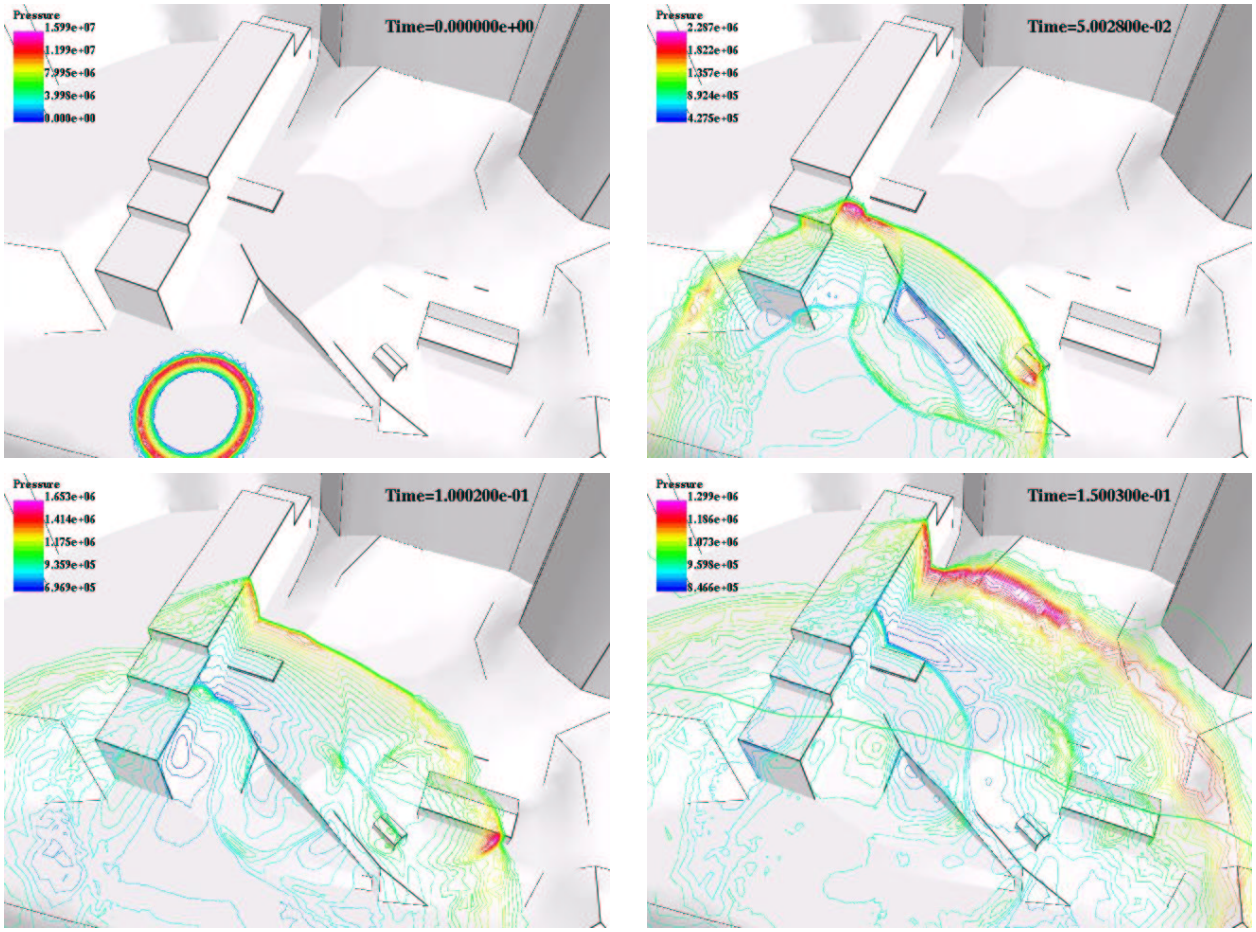


Figure 4b Typical Surface Pressure

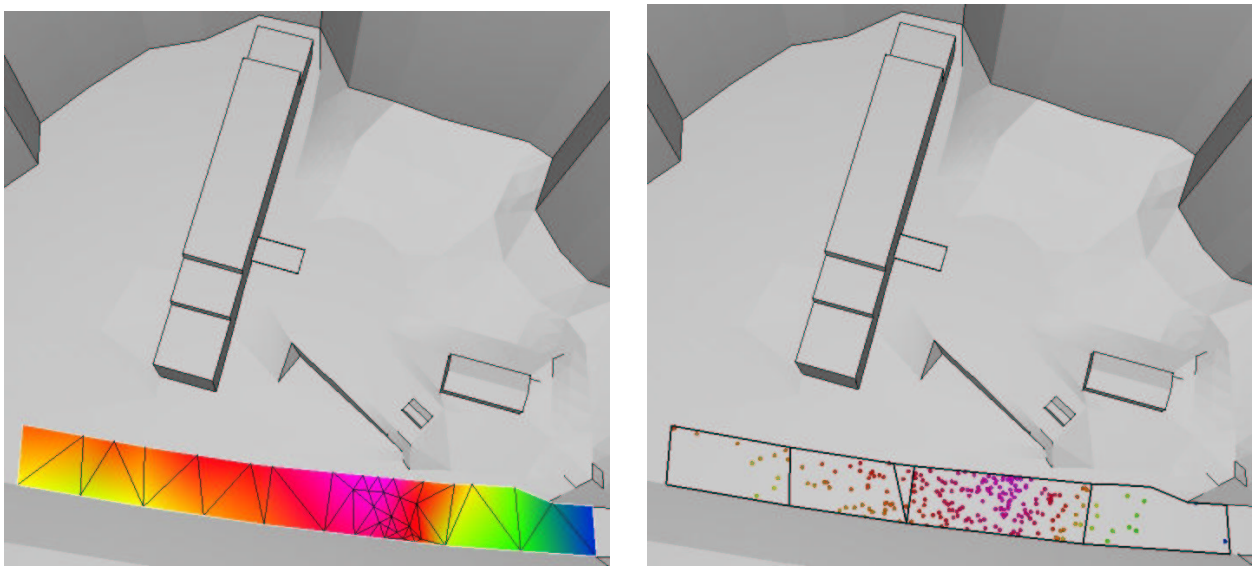
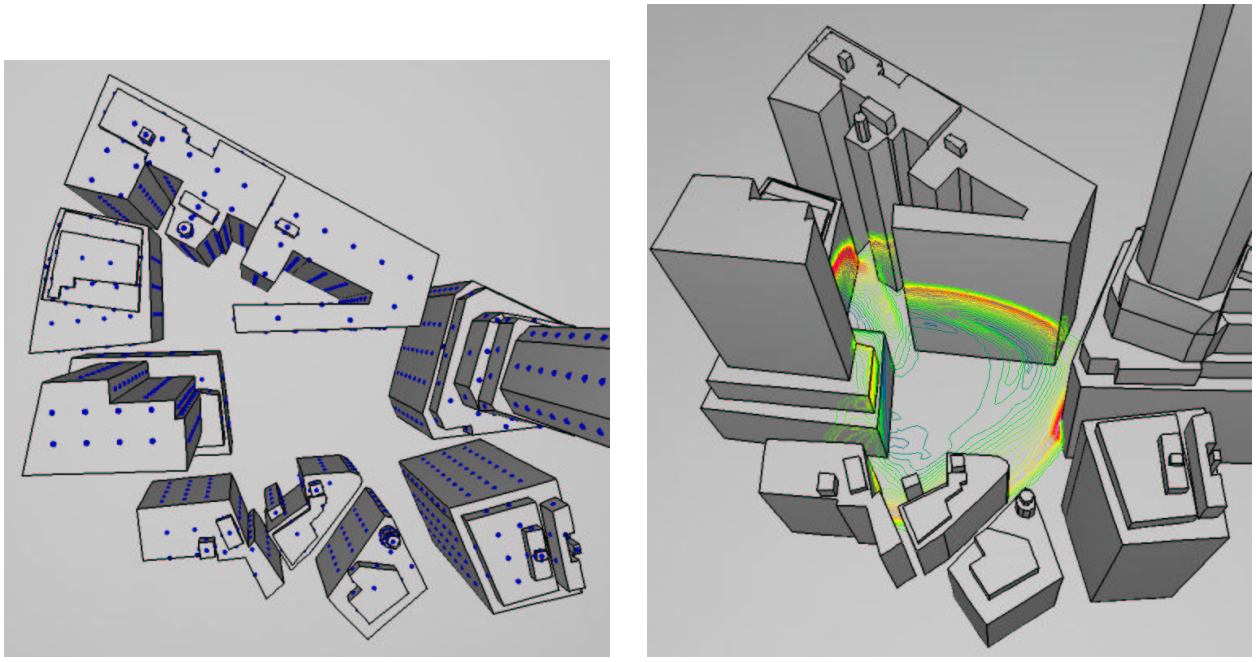


Figure 4c,d Results from FE Approximation and GA

A second grid was then generated, with proper mesh resolution via sources between the blast origin and surfaces facing it. The solution obtained from the first run was then interpolated to this second grid, and the solution was advanced until all shocks of interest exit the domain. The grids generated for this case had approximately $3.2 \cdot 10^6$ elements and each run took approximately 3 hours of CPU time on an SGI Altix (1.3GHz, 32Gbytes, LinuxOS, Intel Compiler) running on 4 processor via shared memory. Based on the experience of the first example, as well as others carried out so far, only the FE approximation was performed. Figure 4c shows the result obtained after 44 runs. A well defined maximum appears, although not at the closest location to the building. This mesh, as well as the distribution of the damage function obtained, were then used to drive the GA. Figure 4d shows the results obtained after 10 generations with 25 individuals (250 runs). The GA is also able to find the maximum, but, as before, requires substantially more runs then the FE approximation.

8.3 Street Canyon Plaza: The example considered is that of a generic plaza surrounded by high-rise buildings. The configuration, together with the window locations chosen, is shown in Figure 5a. The surface pressures for a typical run are shown in Figure 5b. For each run of this case, an initial grid, with increased mesh resolution close to the blast origin was generated. This solution was initialized from 2-D axisymmetric data, and computed for $t_1 = 0.004$ *sec*. A second grid was then generated, with proper mesh resolution via sources close to the origin and the close surfaces. The solution obtained from the first run was then interpolated to this second grid, and the solution was advanced to $t_1 = 0.01$ *sec*. A third grid was then generated, with proper mesh resolution via sources between the blast origin and all surfaces within $l_c = 100\ m$ facing it. The solution obtained from the second run was then interpolated to this third grid, and the solution was advanced until all shocks of interest exit the domain. The grids generated for this case had approximately $1.2 \cdot 10^6$ elements and each run took approximately 3 hours of CPU time on a PC (3.2GHz, 2Gbytes, LinuxOS, Intel Compiler). Based on the experience of the first example, as well as others carried out so far, only the FE approximation was performed. Figure 5c shows the result obtained after 44 runs. A well defined maximum appears, although not at the closest location to the building. This mesh, as well as the distribution of the damage function obtained, were then used to drive the GA. Figure 5d shows the results obtained after 10 generations with 25 individuals (250 runs). The GA is also able to find the maximum, but, as before, requires substantially more runs then the FE approximation.



Figures 5a,b Computational Domain, and Typical Surface Pressure

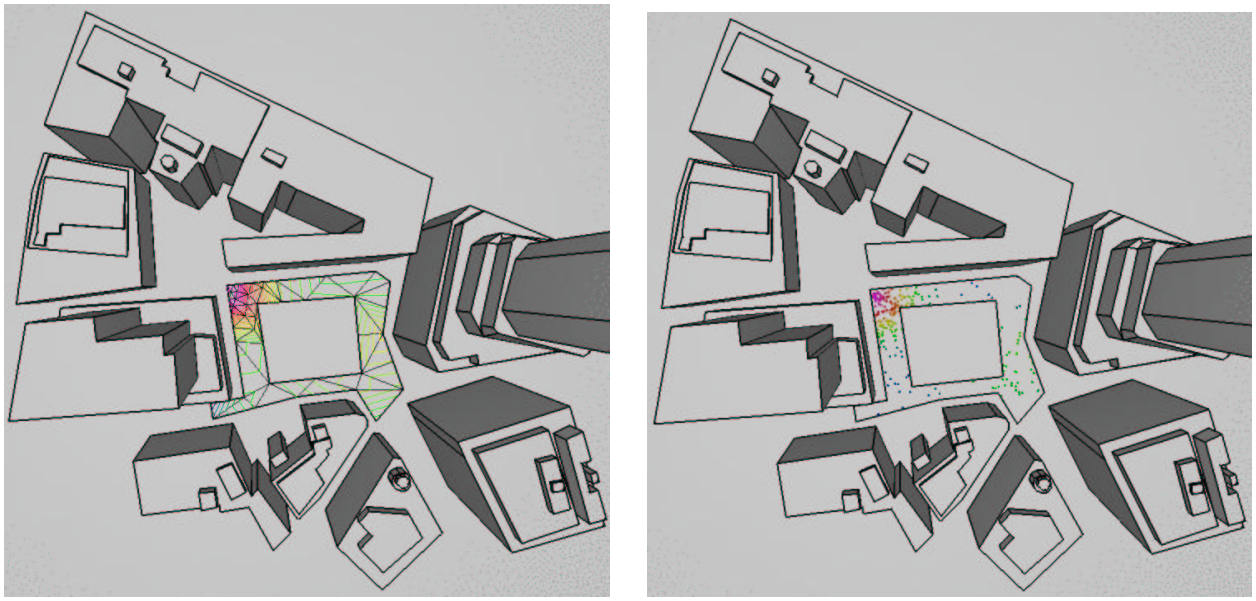


Figure 5c,d Results from FE Approximation and GA

IX. CONCLUSIONS AND OUTLOOK

The use of damage criteria, genetic algorithms and advanced CFD solvers has been proven to be effective to identify blast origins that produce maximum damage. The implementation is simple and does not require

any changes to flow solvers. The main difficulty is the design of proper damage (fitness/ cost) functions to account for injuries, mission impairment and collateral damage. The damage associated with blast origins in the example (as well as others run to date) shows a function with multiple local maxima and minima, making this a difficult optimization problem if treated with more conventional gradient methods. Recent advances in automatic grid generation, flow solvers and hardware have resulted in a considerable reduction of runtime, making realistic damage assessment possible on networks of PCs. Three spatial search algorithms to assess the maximum possible blast damage as a function of blast origin were compared. These are a) a standard genetic optimization technique, b) a Kriging model, and c) a Finite Element approximation approach. The results indicate that the two latter models outperform the standard genetic optimization technique by a considerable margin, enabling an accurate search for locations of maximum possible blast damage with far fewer cost function evaluations.
Future work will focus on:

- Further refinement of the damage (fitness/cost) functions;
- Added realism for geometries as well as boundary conditions;
- Improved fidelity of flow physics, in particular the entry of shocks into buildings;
- Faster flowfield solutions through the use of adaptive h-refinement;
- A link to other lethality assessment codes that can also quantify collateral damage;
- A link to codes that can predict the progressive collapse of structures;
- The use of grid computing to harness the maximum number of resources possible at any given time; and
- An improved treatment of multiple maxima within genetic optimization techniques.

Presently, larger cases are being run, and the methodology is being ported to massively parallel machines using the MPI message passing interface. The final paper will contain results from these runs, as well more details on the algorithms used to steer the optimization.

X. ACKNOWLEDGEMENTS

This work was partially supported by DTRA. The technical monitors were Drs. Michael Giltrud and Young Sohn.

References

[1] J.D. Baum and R. Löhner - Numerical Simulation of Shock Interaction with a Modern Main Battlefield Tank; *AIAA*-91-1666 (1991).

[2] J.D. Baum. H. Luo and R. Löhner - Numerical Simulation of a Blast Inside a Boeing 747; *AIAA*-93-3091 (1993).

[3] J.D. Baum, H. Luo and R. Löhner - Numerical Simulation of Blast in the World Trade Center; *AIAA*-95-0085 (1995).

[4] J.D. Baum, H. Luo, R. Löhner, C. Yang, D. Pelessone and C. Charman - A Coupled Fluid/Structure Modeling of Shock Interaction with a Truck; *AIAA*-96-0795 (1996).

[5] J.D. Baum, H. Luo, E. Mestreau, R. Löhner, D. Pelessone and C. Charman - A Coupled CFD/CSD Methodology for Modeling Weapon Detonation and Fragmentation; *AIAA*-99-0794 (1999).

[6] J.D. Baum, H.Luo, E.L. Mestreau, D. Sharov, R. Löhner, D. Pelessone and Ch. Charman - Recent Developments of a Coupled CFD/CSD Methodology; *AIAA*-01-2618 (2001).

[7] J.D. Baum, H.Luo, E.L. Mestreau, D. Sharov, R. Löhner, D. Pelessone and Ch. Charman - Recent Developments of a Coupled CFD/CSD Methodology for Simulating Structural Response to Airblast and Fragment Loading; Paper presented at ICCS 2001, San Francisco, CA, May (2001).

[8]J.D. Baum, E. Mestreau, H. Luo, R. Löhner, D. Pelessone and Ch. Charman - Modeling Structural Response to Blast Loading Using a Coupled CFD/CSD Methodology; *Proc. Des. An. Prot. Struct. Impact/ Impulsive/ Shock Loads (DAPSIL)*, Tokyo, Japan, December (2003).

[9]D.E. Goldberg - *Genetic Algorithms in Search, Optimization and Machine Learning*; Addison-Wesley (1989).

[10]K. Deb - *Multi-Objective Optimization Using Evolutionary Algorithms*; J. Wiley & Sons (2001).

[11]R.J. Donald, S. Matthias and J.W. William - Efficient Global Optimization of Models for Residual Covariance in Spatial Regression; *Journal of Global Optimization*, Vol. 13, 455-492 (1998).

[12]R. Löhner, K. Morgan, J. Peraire and M. Vahdati - Finite Element Flux-Corrected Transport (FEM-FCT) for the Euler and Navier-Stokes Equations; ICASE Rep. 87-4, *Int. J. Num. Meth. Fluids* 7, 1093-1109 (1987).

[13]R. Löhner and J.D. Baum - Adaptive H-Refinement on 3-D Unstructured Grids for Transient Problems; *Int. J. Num. Meth. Fluids* 14, 1407-1419 (1992).

[14]R. Löhner, Chi Yang, J.D. Baum, H. Luo, D. Pelessone and C. Charman - The Numerical Simulation of Strongly Unsteady Flows With Hundreds of Moving Bodies; *Int. J. Num. Meth. Fluids* 31, 113-120 (1999).

[15]R. Löhner - *Applied CFD Techniques*; J. Wiley & Sons (2001).

[16]R. Löhner, J.D. Baum, E.L. Mestreau, D. Sharov, Ch. Charman and D. Pelessone - Adaptive Embedded Unstructured Grid Methods; *AIAA*-03-1116 (2003).

[17]R. Löhner, J.D. Baum and D. Rice - Comparison of Coarse and Fine Mesh 3-D Euler Predictions for Blast Loads on Generic Building Configurations; *Proc. MABS-18 Conf.*, Bad Reichenhall, Germany, September (2004).

[18]H. Luo, J.D. Baum and R. Löhner - Edge-Based Finite Element Scheme for the Euler Equations; *AIAA J.* 32, 6, 1183-1190 (1994).

[19]D. Quagliarella et al. eds. - *Genetic Algorithms in Engineering and Computer Science*; J. Wiley & Sons (1998).

[20]J. Sacks, W.J. Welch, T.J. Mitchell and H.P. Wynn - Design and Analysis of Computer Experiments (With Discussion); *Statistical Science*, 4, 409-435 (1989).

[21]J. Shinkyu, M. Mitsuhiro and Y. Kazuomi - Efficient Optimization Design Method Using Kriging Model; *AIAA*-04-0118 (2004).

[22]G. Winter, J. Periaux, M. Galan and P. Cuesta eds. - *Genetic Algorithms in Engineering and Computer Science*; J. Wiley & Sons (1995).