Store Separation Simulation using Arbitrary Lagrangian Eulerian based Solver on Oct-tree Grid

Saurabh Pandey 1, Bharat R Agrawal 2
1. CAE Engineer, Zeus Numerix Pvt. Ltd., Powai, Mumbai, India. (saurabh.pandey@zeusnumerix.com)
2. CAE Engineer, Zeus Numerix Pvt. Ltd., Powai, Mumbai, India. (bharat.agrawal@zeusnumerix.com)

Abstract

Work here considers three dimensional simulation of flow involving moving impermeable boundaries with respect to a static oct-tree based Cartesian mesh. Arbitrary Lagrangian Eulerian (ALE) based compressible solver is developed to take into account the relative motion of solid body in the computational domain. Finite Volume Method (FVM) based compressible solver is first validated against some standard test cases of flow over a wedge, forward facing step etc. Later the very well known 1D piston problem is selected for validation of the ALE framework. All the validations done are in good agreement with theoretical, numerical and experimental results. A further application is done to investigate store separation problem.

Keyword: ALE, Finite Volume Method, Cartesian Mesh, Compressible Solver, Moving Boundary, Unsteady Simulation

1. Introduction

Problems involving relatively moving solid bodies in a fluid domain have attracted a lot of attention in recent years. Standard examples include 1D motion of piston, oscillating airfoil, rotating missile, store separation to name a few. Such problems are unsteady in nature and thus the governing equations have to be solved accurately in time. This work discusses on the development of a numerical simulation technique target towards such problems.

Unsteady problems typically of the types mentioned above pose a variety of challenges as far as its numerical simulation is concerned. Firstly to take into account the relative motion demands the grid to be modified according to the moving boundary. Good simulation of such problems can be expected if grid evolves with the changing configuration. Grid modification can be done either only around the body or entire domain of computation has to be touched to take care of the relative motion. Thus grid demands a change, which can be small or large, at each physical time step. Now to solve a problem for a practically feasible time span this re-meshing should be quick and robust.

Cartesian grid is used to capture relative motion and an unsteady compressible solver to capture the physics accurately in time. Grid of this type has a lot of advantage for simulating problems of involving relative boundary motion. Such grid can be created in a jiffy and being a non-boundary fitted type of grid it is comparatively insensitive to geometry complications. Solid body is provided as an input geometry which is then immersed in the grid with cut information updated for cells intersected by geometry boundary. This procedure also makes it easier to include multiple input geometries with or without intersecting surface.

Work on simulating moving boundary problems have been attempted using body-fitted structured grid as well as unstructured meshing method [1-4]. Volume of Fluid (VOF) method [5] and Level-Set method [6] has also been tested for such problems. It was later that boundary immersed methods like Cartesian grid attracted attention for such problems but they were initially restricted to two dimensional problems involving simple geometries. Recently cut-cell based Cartesian mesh approach has been demonstrated with success [7-10].
The work elaborated here is an increment on the previous development of an inviscid compressible solver on oct-tree based Cartesian mesh [18]. The approach taken for simulation is generation of an oct-tree based Cartesian grid in the computational domain. Solver is then executed in the domain taking into account solid boundary motion. Physical time marching is based on both CFL number and velocity of the solid body. Between the two controlling parameters the one which is more restrictive is selected for marching the solver in time. Solid body motion is incorporated by detaching the geometry from the existing grid then moving it for some small time and later re-immersing to in the grid at the new location. Numerical simulation of governing equations is then restarted at this modified, both in space and time, coordinate.

Material presented here starts with a brief introduction of ALE formulation used and then after throwing some light on the governing equations touches upon some aspects of grid generation relevant to unsteady simulation. Next agenda taken up is elaborating the working of compressible solver. Finally some numerical validation results and experimental simulations performed are presented.

2. Arbitrary Lagrangian Eulerian

Fluid dynamics in general takes two ways to describe a fluid flow namely the Lagrangian way of description and Eulerian way. ALE method was developed to use the advantages of both the methods at the same time minimizing the drawbacks of each as much as possible to solve problems requiring large distortions in grid. Moving boundary problems also need large distortions in computational grid for simulation justifying the use of ALE.

Lagrangian method looks at the fluid motion in a way to follow a volume of fluid as it drifts with the flow. It is good way to track free surface and interface between two materials. Main drawback of this approach is its inability to capture very large distortions in computational domain. Eulerian method on the other hand is the one in use in conventional fluid mechanics. This method focuses its analysis at a fixed point in space and the fluid is assumed to flow through this finite fixed volume. Large distortions in computational domain are better handled in these methods which is its major advantage over Lagrangian approach. Accurate interface definition is required a priori for simulation to this method which is its main drawback. In ALE frame work points in computational domain can behave in Lagrangian way by following the moving fluid particles or can be held fixed in space such that the moving fluid passes through it as in Eulerian formulation or can behave in an arbitrary way depending on simulation demands.

Application of ALE in moving boundary problem is achieved by decomposing the motion into two parts one being the uniform motion of computational domain and the other being relative motion of the boundary of moving body in a subset of the domain present in vicinity of the boundary. As an example rotating airframe with oscillating canard motion can be divided into rotational motion of the computational domain to simulate rotation of airframe. Oscillating canards motion is confined to the region of domain closely surrounding the canards thus this relative boundary motion can be treated separately.

ALE methodology by Hirt et. al. [11] takes into account the motion of computational domain. According to this modification the flux calculated at an interface in the computational domain has to be modified to take into account the motion of this interface. Flux in that case is written as

\[ F \cdot n = \left[ \begin{array}{c} \rho v_x \\ \rho u v_x + p n \\ \rho e v_x + p u \cdot n \end{array} \right] \tag{1} \]

where

\[ v_x = (u - u_\Omega) \cdot n \tag{2} \]

is the relative velocity with respect to the moving interface and \( n \) used is unit outward pointing normal. \( u_\Omega \) is the velocity of computational domain to simulate rigid domain motion and \( u \) is the fluid velocity. Hence flux at the interface is modified to take care of the motion of the computation domain by using relative velocity in the convective part component of the flux vector.
3 Governing Equations

Equations governing the flow physics for a relatively moving solid body in a fluid flow are the same conservation laws of mass, momentum and energy with ALE implemented to simulate relatively moving impermeable boundary [12]. This point onwards in the discussion computation domain motion relative to the solid body is assumed to be absent for ease of implementation of relatively moving interface with respect to a static Cartesian Mesh. The conservation law for such problems is.

$$\frac{\partial}{\partial t} \int_{V(t)} U dV + \int_{S(t)} F \cdot n dS = 0$$  \hspace{1cm} (3)

where

$$U = \begin{bmatrix} \rho \\ \rho u \\ \rho \varepsilon_o \end{bmatrix}$$  \hspace{1cm} (4)

$$F \cdot n = \begin{bmatrix} \rho v_n \\ \rho u v_n + p n \\ \rho e v_n + p u.n \end{bmatrix}$$  \hspace{1cm} (5)

$$e_o = \varepsilon + \frac{1}{2} |u|^2$$  \hspace{1cm} (6)

$$v_n = (u - w).n$$  \hspace{1cm} (7)

The velocity $w$ used in eq. (7) is the velocity of the moving impermeable boundary with respect to the static Cartesian mesh. An important thing to note in eq. (3) is that control volume and control surface are no longer constant as is the case with conventional CFD rather they are a function of time.

Simulation of such a flow involves cells whose volume is constant in time and at the same instance variable volume cells also exist in the domain. Above equations change their form to well known Eulerian formulation for static volume cells since $w$ is zero for such cells. Whereas cells which are intersected by the solid body have a finite value of $w$ hence the cell shape is not static in time. Even for such cells, except for the solid body interface other faces are static hence $w$ is zero for these static interfaces as well. It is only for relatively moving boundary of the solid body in the cell $w$ turns out to be a non-zero number. If the assumption of an impermeable boundary is taken then $w$ can be defined as $w.n = u.n$ which in words says that an impermeable boundary doesn’t allow convection. Current work uses equations (3-7) to numerically simulate the flow.

4 Meshing of Domain

A previously developed grid generator module which was tested with an Euler incompressible solver is used in the current work [13]. Grid generation using oct-tree method over a complex geometry is constrained with three requirements from the solver. Firstly, the ratio of volume between two neighbouring cells should be at most eight or at least one-eighth. This constraint is implemented to ease the gradient calculations. The second important constraint in the oct-tree grid is that the uncut volume of all neighbouring cells around a cut-cell should be same as uncut volume of the cut-cell itself. This is an essential requirement for quick merging of small cut-cells. Finally, volume and shape of cut-cell needs to be extracted with maximum possible accuracy so that the volume conservation is held near the geometry and the computations done at the boundary are accurate [14].

4.1 Merging of cut-cells

Oct-tree grid is infected with very well known “Small Cell” problem wherein as the geometry is inserted inside the grid it can intersect some cells in such a way that a very small cell is left in the
computational domain. Such cells being very small in volume adversely affect the time stepping of the solver hence greatly increasing the simulation time [15]. Merging of such small cells is one of the well known remedies. It is done by choosing cells around the selected small cell and merging with it. The basis of selection is the area of the face between the cell selected for merging and the small cell. Neighbouring cell sharing the largest area with small cell is selected and merged. This process is continued as long as the merged cell’s volume is above a threshold limit.

4.1 Maintenance of volume conservation
Another important concern for simulating a moving solid boundary problem in a static Cartesian mesh is the conservation of volume. In order to get a judgment of change in volume of cell on its division volume of cell pre as well as post division is calculated and an error is evaluated. It is in general found that these operations don’t produce significant change in the volume thus maintaining volume conservation.

5 Numerical Simulation
Compressible solver based on FVM is used for the numerical simulation of the governing equations (Equations 3-7). Discretization of the conservation law stated in eq. (3) is done to take into account the changing volume of cell. The discretized equation used is a backward-Euler scheme

\[(IU)^{n+1} = (IU)^n - \Delta t \sum_{faces} (F \cdot \hat{n} \ dS)^n\]  

(8)

where \(U\) and \(F \cdot \hat{n}\) are defined in equations 4 and 5 respectively. Left hand side of the above equation denotes the product of updated conserved variables and updated volume. Right hand side is composed of two terms of which the first term is the product of original conserved variables in the cell and original cell volume. Second term is the residue calculated from the fluxes at the previous time stamp. This is an approximation done in the above scheme wherein fluxes for time marching are calculated from the previous time step.

Flux schemes due to van Leer [16] and HLLC [17] are written and validated for steady state problems. However for unsteady simulation flux scheme due to van Leer is utilized. The solver is made capable to account for AMR by maintaining conservation while such adaptive refinements are performed on the grid. Grid refinement is based on the difference in one of the primitive variables in neighboring cells. If this difference in primitive is above a preset threshold value refinement is performed while maintain conservation.

De-refinement on the other hand is done by calculating the variance of a primitive variable in all child cells. If the variance is not large, again depending on a preset limit, then all the child cells are merged together to results in its parent cell. Needless to say that all the conserved distributed among children are summed up and given to the parent on de-refinement.

6 Implementation of Unsteady Simulation Cycle
The whole process of simulation is started by first solving the flow at the original configuration. With the results in hand the process of movement of solid body is started by detaching the movable geometry from the grid. This is achieved by removing cut information of the movable geometry from existing grid. Result obtained initially is used to calculate the aerodynamic forces on the movable body which further is used to move it in 3D space. Grid is regenerated at this new moved location of the body by re-populating cut-cell information in the grid. Solver is now executed over this new configuration to get an updated time solution. This in turn gives new forces on the body and it is again transported to a new location based on the dynamics. This cycle is repeated for a specified amount of time. Thus solving flow around a moving solid body.

The main restrictive criterion here is the time for which the movable body is allowed to displace is such that no part of that geometry crosses more than one face in the computational domain. This is
implemented to make the process of maintenance of conservation simple. It can also be viewed as a 
CFL type criterion for simulating moving boundary problems.

6.1 Maintenance of mass, momentum and energy conservation

Solid body movement relative to the grid brings new cells in the computational domain at the same 
time erases few cells from the fluid domain. Also adaptive refinement of cells as well as a cell’s de-
refinement creates and deletes cells from the domain. Mass, momentum and energy conservation has 
to be maintained in cells like these to correctly capture the physics of the problem.

Cells which go inside the solid body and hence out of the fluid domain have to distribute the 
conserved quantities in the neighboring cells which will remain in the domain even after the motion of 
the solid body. Hence conserved quantities from such cells are distributed among neighboring cells in 
the ratio of interface area. Similarly for newly created cells, in order to maintain global conservation, 
are required to have ideally no conserved quantities. But zero initialization of conserved quantities will 
cause singularity in the solver as a result of which such newly born cells are initialized with 
infinitiesimally small amount values.

Refinement and de-refinement operations on cells are made conservative by interpolating the 
conserved data from old cell(s) to the new ones and then converting them to their primitive form.

7 Numerical Results

Current section provides a brief overview on the results obtained for various test cases attempted while 
developing the numerical simulation method. Starting with the validation of supersonic flow over a 
 wedge and a forward step the discussion would move to the results obtained over the realistic 
geometry of an aircraft. Next to follow is the validation over 1D piston problem with its comparison 
with analytical result. The discussion concludes with the dynamic simulation of store separating from 
the same aircraft.

7.1 Supersonic flow over a wedge of 10° angle

One of the basic validation cases for any compressible solver is how close the angle of a capture shock 
wave is. For this purpose, a flow of Mach number 2 is considered over a wedge of 10° angle with the 
flow direction. This problem is solved using flux splitting scheme due to van Leer [16]. The results are 
shown in “Figs. 1”. “Fig. 1 (a)” shows the grid for solution. This case was initialized with a uniform 
Cartesian base grid and the clustering along the shocks appears completely due to the adaptive 
framework of the solver. “Fig. 1 (b)” shows the contour plot of the Mach number in the flow field. 
Comparison of obtained data with theoretical is as follows:

*Theoretical solution*:
shock angle = 39.31°, pressure ratio across the shock wave = 1.7067.

*Computed solution*:
shock angle = ~40°, pressure ratio across the shock wave = 1.7063.

![Fig. 1](image)

7.2 Supersonic flow over forward facing step

In the next validation case, a flow of Mach number 3 inside a 2D duct with a forward facing step was 
considered. This test case was solved using HLLC scheme. “Fig. 2” shows the pressure contour plot 
from literature [16] and “Fig. 3” shows the pressure contour plot obtained using the present solver. In 
the obtained solution the maximum pressure is reached inside the Mach disc at the top boundary and
the ratio of this pressure with upstream pressure is obtained as 12.23 which match closely with the literature results.

Fig. 2 Contour plot of Pressure from literature for forward step problem

Fig. 3 Contour plot of Pressure for forward step problem

7.3 Supersonic flow over an aircraft

To demonstrate the speed of the solver, supersonic flow at Mach number 2.0 is considered over aircraft geometry at angle of attack of 5°. “Figs. 4 (a) and 4 (b)” shows the pressure profile over the upper and lower surfaces of the wing respectively. It can be noted that red and blue represents highest and lowest values respectively. Thus, the differences in colours on these surfaces suggest a positive lift over this geometry. This is a converged solution and was obtained in less than 15 minutes on a standard modern day desktop machine.

Fig. 4 Pressure distribution (a) Upper Surface (b) Lower Surface
7.4 One Dimensional Piston
Validation of maintenance of conservation and ALE implementation is in 1D is the goal of this test case. A 1D piston is assumed at the origin and moved with a constant velocity of Mach No. 2 along positive x axis. Whole of the computational domain is assumed at zero Mach No. As the piston rushes along the x direction it pushes the fluid ahead of it and at the same time fluid behind the piston is relived by the forward motion. This problem is dealt in detail by Liepmann and Roshko in their text on gas dynamics [19].

Shock is seen to form ahead of the piston and this proves without any doubt that conservation is well maintained in the system. If it were not the conserved values of mass, momentum and energy would leak from the system and hence formation of shock would be an impossible phenomenon. “Fig. 5” shows a plot of the pressure ahead of the piston face with time. In the same plot it is compared with the analytical result. Looking at the graph it is clear that after an initial transient phase the pressure ahead of the piston exactly matches with the analytical result. Snapshots of pressure ahead and behind the piston at three different iterations (time stamps) is shown in “Fig. 6” which clearly shows a shock wave ahead and an expansion fan behind the piston.

7.5 Sphere and fluid moving with same velocity
Another test case taken for validation of ALE methodology in 3D is a sphere moving in fluid at the same velocity as that of the fluid. Ideally all state variables should be rendered unchanged as relative velocity to the moving sphere is zero or in other words if we climb on sphere and look at the fluid flow around it will look static.
Fluid domain as well as the sphere is given a velocity component of 100 m/s in x, y and z directions. Simulation is carried out for 0.28 seconds with results at initial and final time steps are shown in “Fig. 7”. “Fig. 7 (a) and (b)” shows the initial and final positions of sphere with its grid respectively. “Figs. 7 (c) and (d)” shows the velocity magnitude and pressure profile respectively. Initial velocity magnitude of 173.2 m/s is perturbed by a small amount and final velocity magnitude variation lies within 172.3 m/s and 174.2 m/s. The relative error in this data is around 0.57%. Pressure variation is also between 101116 N/m² and 102868 N/m² as against an initial pressure of 101325 N/m². The average relative error in pressure is also less than 1% which can be assumed to be within limit.

7.6 Sphere motion in still domain

Next experiment done with the current work is simulating flow profile around a sphere again when it is moving in a still fluid domain. Sphere is translated at a constant Mach No. of 2 in still air. Results are shown in “Fig. 8”. Force in X direction is shown in “Fig. 8 (a)” shows a rise in the transient phase and soon settles to a constant force. In the same plot force on sphere is shown with an inflow of 2 Mach and a static sphere and the two results are in fairly good agreement. The drag coefficient on sphere using ALE technique is calculated to be 1.107 and using conventional CFD its value comes around 1.185. “Fig 8 (b)” shows pressure profile around sphere with a bow shock clearly visible ahead of it.

7.7 Store separation

Aircraft and generic store geometry was simulated at supersonic flow of Mach number 2.0. “Fig. 9 (a)” shows the pressure variation over both the geometries after the steady state has been reached by the solver. In this figure a blue patch is clearly seen on the lower surface of the aircraft with. It can be noted that most of the surface of aircraft is blue which is due to the fact that this region lies outside the actual computational domain and does not have any pressure values associated with it.

“Figs. 9 (b-d)” shows the pressure variations and the location of the store at 10 iterations of the dynamics solver. It can be seen that from “Fig. 9 (a)” to “Fig. 9 (b)”, the blue patch which is a low pressure area grows in size and then in subsequent figures it moves towards the trailing edge and fades.
8 Conclusions and Future Scope

Numerical simulation of flow involving relatively moving impermeable boundary is attempted in the current work. Initially a compressible flow solver developed for solving unsteady problems is validated for some very well known test cases viz. flow past a wedge and flow over a forward facing step. The results obtained are in perfect agreement with theoretical results or literature. After accomplishing the validation of the solver over steady state problems attention is diverted towards unsteady simulation. Very well known 1D piston problem is selected for the validation of ALE methodology implemented to simulate moving boundary. Results over 1D piston were also in good agreement with the analytical results provided in the literature. This validation exercise also proved that conservation is well maintained throughout the simulation.

Various next steps could be taken based on the current work to enhance this method’s ability to capture flow properties. First to mention is incorporating rigid domain motion or in other words motion of the computational domain as a whole to allow simulation of rotating bodies coupled with other local relative motions. This work will also include modification of flux implementation according to the ALE formulation. Another enhancement can be brought in the existing method by incorporating higher order time accurate schemes to better estimate the residue for next time step. Finally to relax the time constraint imposed by the motion of solid body implicit schemes could be implemented. This would further remove the requirement of merging of small cell since time would no longer be guided by the size of cell.

Acknowledgements

We would like to thank Prof. G. R. Shevare of Aerospace Engineering Department, IIT Bombay for his valuable guidance and Zeus Numerix Private Limited for providing support for the development.

References


