ILASS-Americas 22nd Annual Conference on Liquid Atomization and Spray Systems, Cincinnati, OH, May 2010

Quadrature-free Discontinuous Galerkin Level Set Scheme

Mark Czajkowski*, Olivier Desjardins
Department of Mechanical Engineering
University of Colorado at Boulder, Boulder, CO 80309

Abstract

This preliminary work describes the implementation of a quadrature-free discontinuous Galerkin (DG) scheme to capture an interface location using a level set. The approach allows for arbitrarily high order implicit representation of the level set function by using a basis of high order polynomials. Additionally, the high order accuracy does not require a large stencil since DG only uses information from the edge of upwind cells. A quadrature-free implementation allows for all the integrals that appear in the derivation of the scheme to be evaluated beforehand to reduce computational cost. Reinitialization of the level set function is important to maintain high accuracy and robustness. We present a quadrature-free DG scheme for the level set reinitialization. The implementation is based on the solution of a Hamilton-Jacobi equation using the DG scheme of Cheng and Shu [Cheng and Shu, Jcp 223, 2007]. Preliminary transport tests illustrate the excellent potential of the method, while a one-dimensional reinitialization test proves the feasibility of a fully quadrature-free DG level set scheme.

*Corresponding Author: mark.czajkowski@colorado.edu

Introduction

In simulations of complex, turbulent, multiphase flows, discontinuities at the interface are difficult to deal with numerically. When the two phases have significantly different properties, including the presence of high density ratios or surface tension, discretization of the Navier-Stokes equations becomes challenging. A variety of methods have been used to handle discontinuities such as the continuum surface force (CSF) approach [1] which smears the discontinuity over a few grid cells and the ghost fluid method (GFM) [2] that uses a sharp representation of the discontinuity through a generalized Taylor series.

Both of the aforementioned methods are based on knowledge of the interface location. Commonly, two approaches are used to locate the interface: 1) interface tracking methods which are Lagrangian based and use re-meshing so that the interface location can be explicitly defined. 2) interface capturing methods which are Eulerian based implicitly represent the interface location on a fixed grid. Details of both methodologies are outlined below.

Interface tracking schemes typically use arbitrary Lagrangian-Eulerian (ALE) methods based on a mesh that deforms with the interface [3, 4, 5, 6], or MAC methods that advect Lagrangian particles which define a given fluid by their location [7]. Interface tracking schemes face difficulties when large interface deformations exist or when the interface disconnects and reconnects. When this happens, significant re-meshing is needed to account for the large interface changes.

Interface capturing methods such as volume of fluid (VOF) and level set methods are very robust and have been used for a variety of applications. VOF methods capture the interface using the volume fraction of fluid within each grid cell [8, 9, 10]. Level set methods advect a function defined such that the zero iso-surface is the interface [11, 12]. VOF methods suffer from limitations including calculating interface curvature and normal and VOF is typically first or second order. Level set methods alleviate many of the problem found with VOF methods but limitations do exist such as explicit mass conservation. Another issue arises when the level set function is stretched and the gradient becomes excessively large or small near the interface. As a result, the location of the interface is not accurately known. To remedy this problem, reinitialization is performed. Reinitialization transforms the deformed level set into a new function with a reasonable gradient without moving the location of the interface.

Discontinuous Galerkin (DG) methods are an attractive technique for solving the advection equation used to evolve the level set function [13, 14]. DG offers many advantages including accuracy, robustness, and compactness that leads to highly parallelizable schemes [15, 16]. The work presented here is based on the DG level set scheme presented by Marchandise et al. [13]. Their work is extended by developing a quadrature-free DG reinitialization scheme.

Numerical Scheme

A DG based level set scheme provides many advantages to accurately and robustly transport and capture the interface in multiphase simulations. The level set function $G(\underline{x}, t)$ is defined to be positive in one fluid, negative in the other, and zero at the interface. The function is initiated as a smooth function (e.g. the signed distance function). Next, an material transport equation is solved to update the interface position. When the flow field develops large or small gradients of the level set function near the interface, a second equation is solved to reinitialize the level set. Derivations of the level set transport and level set reinitialization schemes using a quadrature-free DG approach are given below.

Level Set Transport

An equation that represents the material transport of the interface by the base flow is used to evolve G. The equation is,

$$\frac{\partial G}{\partial t} + \underline{u} \cdot \nabla G = 0, \tag{1}$$

where $\underline{u}$ is the base flow velocity that advects the level set. Solving the transport equation using a quadrature-free DG scheme is the objective. The main idea of DG is to represent a function (e.g. G) as a finite linear combination of basis functions (ϕ). Therefore,

$$G(x) = \sum_{n=0}^{N} g_n \phi_n = g_i \phi_i, \tag{2}$$

where $N + 1$ is the number of degrees of freedom (DOF), ϕ_i is the basis, and g_i is the weight associated with the basis function. While almost any set of basis functions will work, some sets have beneficial properties such as orthogonality. In this project, the Lagrange polynomials are used.

Quadrature-free implementations are a subclass of DG methods in which all integrals that appear in the derivation are written to be only a function of the basis and not a function of time. As a result, the integrals can be precomputed to reduce the computational cost at each update step. To derive our

quadrature-free DG implementation of the level set transport equation, Equation 1 is written using the basis functions, multiplied by a test function, and integrated over the cell.

Rewriting Equation 1 with the basis and using an incompressible flow assumption results in,

$$\frac{\partial g_i \phi_i}{\partial t} + \frac{\partial u_k g_i \phi_i}{\partial x_k} = 0, \tag{3}$$

using Einstein's summation notation. The next step involves multiplying by a test function ϕ_j which in our scheme is a member of the set of basis functions. Applying the properties $g(t)$ and $\phi(x)$ and multiplying by ϕ_j, we obtain,

$$\phi_j \phi_i \frac{\partial g_i}{\partial t} + \phi_j g_i \frac{\partial u_k \phi_i}{\partial x_j} = 0. \tag{4}$$

Integration of Equation 4 over each grid cell using the divergence theorem and integration by parts completes the process. The result is,

$$\left[\int_V \phi_i \phi_j \, dV\right] \frac{\partial g_i}{\partial t} + \left[\oint_S \phi_i \phi_j \, dS\right] g_i^+ u_k n_k - \left[\int_V \frac{\partial \phi_j}{\partial x_k} \phi_i \, dV\right] g_j u_k = 0, \tag{5}$$

where n is the normal to the surface and g^+ is g defined from the upwind side of the face in regards to the base flow velocity u_k.

The final step in this derivation involves a key assumption that allows for the quadrature-free implementation to be successful. In order to remove u_k from the surface and volume integrals, u_k must be assumed to be a constant at the cell interface and within the cell volume, respectively. This is an appropriate assumption because the second order flow solver only provides one value of base flow velocity for each grid cell. The final result is,

$$\frac{\partial \underline{g}}{\partial t} = \underline{\underline{M}}^{-1} \left[\underline{\underline{K}}_k \, \underline{g} \cdot \underline{u} - \sum_{\text{faces}} \underline{\underline{B}} \, \underline{g}^+ \left(\underline{u} \cdot \underline{n}\right)\right], \tag{6}$$

with,

$$\underline{\underline{M}} = \int_V \underline{\phi} \, \underline{\phi}^T dV, \tag{7}$$

$$\underline{\underline{K}} = \int_V \nabla(\underline{\phi}) \, \underline{\phi}^T dV, \tag{8}$$

$$\underline{\underline{B}} = \int_S \underline{\phi} \, \underline{\phi}^T dS. \tag{9}$$

In Equation 6, all of the integrals (Equations 7-9) are only of a function of the basis functions and not a function of time. As a result, they can be precomputed for a chosen set of basis functions and then Equation 6 can easily be updated without evaluating any integrals.

Reinitialization

A variety of approaches to reinitialize G to a signed distance function have been developed. The first class is known as fast marching methods that solve the Eikonal equation $||\nabla G|| = 1$ [17, 12]. Closest point algorithms have also been explored that have a tree-based structure to determine the closest interface point to define the distance function [18, 19]. This method requires a set of points on the interface to be created. The final reinitialization method is based on solving the Hamilton-Jacobi equation,

$$\frac{dG}{d\tau} + S(G)\left(\|\nabla G\| - 1\right) = 0, \tag{10}$$

where τ is pseudo-time and S is a smoothed sign function such as a scaled hyperbolic tangent or $S(G) = G/\sqrt{G^2 + \alpha^2}$ and α is a function of the grid size [14].

Because DG is used to transport the level set, it is advantageous to use the same numerical method for the reinitialization so that the higher order weights can be maintained. Therefore, a DG implementation of the Hamilton-Jacobi equation is highly desired.

Two methods have been developed to solve the Hamilton-Jacobi equation. The first by Hu and Shu [20] applies the DG framework to solve a system of conservation laws satisfied by the derivative of the solution. Cheng and Shu [21] proposed a method to solve the Hamilton-Jacobi equation directly for the solution by adding additional stability terms. Because the associated computational cost is significantly lower with the latter, Cheng and Shu's [21] approach is used to solve Equation 10. Their approach applied to the Solution of Equation 10 is given below.

To simplify notation the following variables are introduced:

$$F \equiv S\left(G\right)\left(\|\nabla G\| - 1\right), \tag{11}$$

$$F_1 \equiv \frac{\partial F}{\partial G_x}, \tag{12}$$

where, $G_x = \partial G/\partial x$. Now the grid notation is defined for a cell $I_j = (x_{j-\frac{1}{2}}, x_{j+\frac{1}{2}})$. And the interval between adjacent cell I_j and I_{j+1} is written as $I_{j+\frac{1}{2}} = (x_j, x_{j+1})$. The jump of G is defined as

$$[G]_{j+1/2} = G(x^+_{j+\frac{1}{2}}, t) - G(x^-_{j+\frac{1}{2}}, t). \tag{13}$$

$G^+_{j+1/2}$ and $G^+_{j-1/2}$ are G evaluated on the right and left sides of the discontinuity at $x = j + 1/2$, respectively.

With this notation, Cheng and Shu's [21] formulation can be written for the case of one-dimensional level set reinitialization as:

$$\begin{aligned}
&\int_{I_j} \left(\partial_t G(x,t) + F\left(\partial_x G(x,t), x\right)\right) \phi_k dx \qquad (14)\\
&+ \frac{1}{2}\Bigg(F_1(\partial_x G, x_{j+\frac{1}{2}})\\
&- \left| F_1(\partial_x G, x_{j+\frac{1}{2}}) \right| \Bigg) \left[G\right]_{j+1/2} \left(\phi\right)^-_{j+1/2}\\
&+ \frac{1}{2}\Bigg(F_1(\partial_x G, x_{j-\frac{1}{2}})\\
&+ \left| F_1(\partial_x G, x_{j-\frac{1}{2}}) \right| \Bigg) \left[G\right]_{j-1/2} \left(\phi\right)^+_{j-1/2} = 0,
\end{aligned}$$

for any $\phi_k \in P^N$. Where, P^N is the set of polynomial of order N. As noted by Cheng and Shu, [21] the terms on the last four lines of Equation 14 were added for stability without affecting the accuracy. In Equation 14, $\partial_x G$ is needed on the interface between cells where a discontinuity may exist. As a result, Cheng and Shu [21] developed a method for determining the derivative at the discontinuity. The main idea is to define a function w of order $2N + 1$ using the value of the level set in the two cells boarding the cell interface, and then setting the derivative of w equal $\partial_x G$. Details of the approach can be found in [Cheng and Shu, Jcp 223, 2007]. It was determined that using w of order $2N + 1$ caused oscillations and the order was reduced in our formulation to order N to be consistent with the rest of our scheme.

The other difficultly with directly solving Equation 14, is in the evaluation of the integral using the desired quadrature-free approach. F is a function of x and t and therefore the integral can not be evaluated beforehand. To circumvent this, F is defined as a linear combination of the basis functions,

$$F = \sum_{i=0}^{N} f_i(t)\phi_i(x), \qquad (15)$$

so that the time dependent forcing weights f_i can be removed from the spacial integral in Equation 14. The remaining integral only contains the basis functions and can be precomputed. In order to calculate the forcing weights, the following procedure is used to define f_i in terms of g_i, the weights corresponding to the level set function. The idea is to write a Taylor series expansion of $F(G, d_x G)$ and remove terms that cannot be represented within the chosen basis, so that F can be written as a linear combination of the forcing weights f and the basis functions ϕ. The forcing weights f will be a algebraic combination of the level set weights g.

The procedure is:

1. Define the level set and the derivative of the level set at $x = 0$,

$$\begin{aligned}
G_0 &= G(x = 0) \qquad (16)\\
&= \sum_{i=0}^{N} g_i(t)\phi_i(x = 0),\\
\partial_x G_0 &= \frac{\partial G}{\partial x}(x = 0) \qquad (17)\\
&= \sum_{i=0}^{N} g_i(t)\frac{\partial \phi_i}{\partial x}(x = 0),\\
F_0 &= F(G = G_0, \partial_x G = \partial_x G_0). \qquad (18)
\end{aligned}$$

2. Write a Taylor series expansion of F around G_o and $\partial_x G_0$,

$$\begin{aligned}
F \approx \sum_{n=0}^{N}\sum_{m=0}^{N} \frac{1}{(n+m)!}\frac{\partial^{(n+m)} F_o}{\partial G^n\ \partial(\partial_x G_0)^m}\Bigg[\qquad (19)\\
\left(G - G_0\right)^n \left(\partial_x G - \partial_x G_0\right)^m \Bigg].
\end{aligned}$$

3. Rewrite Equation 19 using the definition of level set defined in Equation 2, and Equations 16 and 17 to get,

$$\begin{aligned}
F \approx \sum_{n=0}^{N}\sum_{m=0}^{N} \frac{1}{(n+m)!}\frac{\partial^{(n+m)} F_o}{\partial G^n\ \partial G_x^m}\Bigg[\qquad (20)\\
\left(\sum_{p=0}^{N}\left(g_p\phi_p\right) - G_0\right)^n\\
\left(\sum_{q=0}^{N}\left(g_q\frac{\partial\phi_q}{\partial x}\right) - \partial_x G_0\right)^m \Bigg].
\end{aligned}$$

4. Expand Equation 20. The new equation will be of the form,

$$F \approx \sum_r C_r (\phi_p)^s \left(\frac{\partial \phi_q}{\partial x}\right)^t, \qquad (21)$$

where $C_r = f(F, g_i, g_j, G_0, \partial_x G_0)$ are constants in front of each combination of basis functions, and s and t range from 0 to N.

5. Remove high order terms from Equation 21. If $(\phi_p)^s \left(\partial\phi_q/\partial x\right)^t$ can be represented by a linear combination of the chosen basis then it should

be retained. Otherwise, the term can be neglected and eliminated from subsequent equations. Mathematically, if

$$C_r \left(\sum_{i=1}^{N} D_i \phi_i \right)_r = C_r(\phi_p)^s \left(\frac{\partial \phi_q}{\partial x} \right)^t, \quad (22)$$

with constants D_i, can be written with the set of basis functions, then the term should be kept. Otherwise, the order is too high and the term can be neglected.

For example, consider a one-dimensional basis of Lagrange polynomials, $\phi_0 = 1$, $\phi_1 = x$,and $\phi_2 = x^2 - 1/3$, is chosen. Then, $(\phi_0)^1 (d\phi_2/dx)^1 = 2x = 2\phi_1$ and should be kept. In contrast, $(\phi_2)^2 (d\phi_2/dx)^1 = 2x^3$ can not be written as a linear combination of this basis and should be neglected.

6. Define, the sum of the terms in from of each basis function as the desired variable,

$$f_i = \sum_r C_r (D_i)_r. \quad (23)$$

Now, Equation 14 can be written such that it is in a quadrature-free form,

$$\begin{aligned} \frac{\partial \underline{g}(t)}{\partial t} = & - \underline{f}(t) \cdot \underline{\phi}(x) - \left(\int_{I_j} \underline{\phi}\, \underline{\phi}^T dx \right)^{-1} \\ & \left[\frac{1}{2} \Big(F_1(\partial_x G, x_{j+\frac{1}{2}}) \right. \\ & \left. - \left| F_1(\partial_x G, x_{j+\frac{1}{2}}) \right| \Big) [G]_{j+1/2} \left(\underline{\phi} \right)^-_{j+1/2} \right. \\ & + \frac{1}{2} \Big(F_1(\partial_x G, x_{j-\frac{1}{2}}) \\ & \left. + \left| F_1(\partial_x G, x_{j-\frac{1}{2}}) \right| \Big) [G]_{j-1/2} \left(\underline{\phi} \right)^+_{j-1/2} \right] \\ & = 0. \end{aligned} \quad (24)$$

In Equation 24, every term is an algebraic combinations of the level set weights except for the integral. However, the integral is only a function of space and not a function of time, and consequently can be precomputed. This process is lengthy but it is only needed to setup the scheme. Once the algebraic equations have been formulated they will not change at each time-step, only the basis multipliers will.

Numerical Results

DG Level set Transport

The quadrature-free DG level set transport equation has been implemented in a three-dimensional, arbitrarily high order flow solver known as NGA [22]. Using NGA and the new scheme for level set transport the following example problems were studied including Zalesak's disk, two and three-dimensional deformation cases, and a droplet impacting quiescent water.

Zalesak's disk

The first case is Zalesak's disk subjected to 50 rotations. Simulations consisted of a disk with radius 0.15 and notch width of 0.05, initiated with center at $(x, y) = (0, 0.25)$, subjected to 50 full rotations using the two-dimensional velocity field,

$$\begin{aligned} U &= -2\pi y, \\ V &= +2\pi x. \end{aligned} \quad (25)$$

Two meshes were used, a coarse mesh of 50×50 and a fine mesh of 100×100. Number of DOF used in the DG scheme were varied from two through four. Fig. 1 and 2 show the results of varying DOF on the 50×50 mesh and 100×100 mesh, respectively. From the plots it is evident that as the number of DOF is increased (color changes from red to green to blue) the disk shape after 50 rotations converges to the initial condition (black). Mesh convergence is also evident when comparing the results obtained on the 50×50 mesh and 100×100 mesh. With a mesh of 100×100, the result after 50 rotations is almost indistinguishable from the initial condition. And the 50×50 mesh, which has only 2 points across the notch, gives excellent results after 50 rotations.

2D deformation

The next test case is the deformation of a cylinder due to a vortex. Fig. 7 in Section 5 shows how the cylinder with diameter 0.3 and initial center at $(x, y) = (0, 0.25)$ is deformed and then returned to the initial state using the velocity field

$$\begin{aligned} U &= -2 \sin(\pi x)^2 \sin(\pi y) \cos(\pi y) \cos(\pi t/8), \\ V &= +2 \sin(\pi y)^2 \sin(\pi x) \cos(\pi x) \cos(\pi t/8). \end{aligned} \quad (26)$$

The cross-sectional area of the liquid cylinder was calculated at the end of the deformation cycle and compared with the initial area for various number of DOF. Results of the shape are are shown in Fig. 8 in Section 5. The error defined by the final area divided by the initial area is given in Table 1. As expected, when the number of DOF is increased the final area converges to the initial area. Mesh convergence was

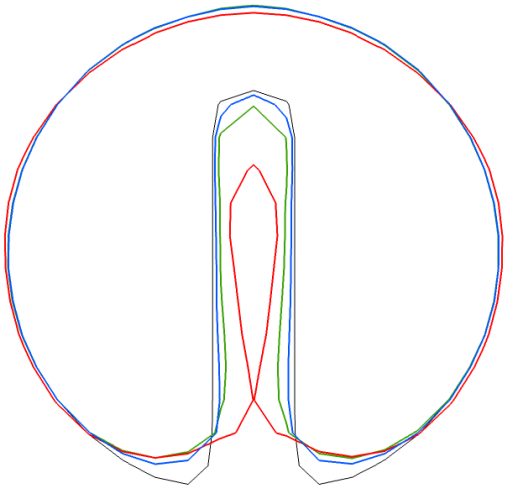


Figure 1. Zalesak disk after 50 rotations using a 50×50 mesh with varying number of DOF (N=2: Red, N=3: Green, N=4: Blue, Initial condition: Black)

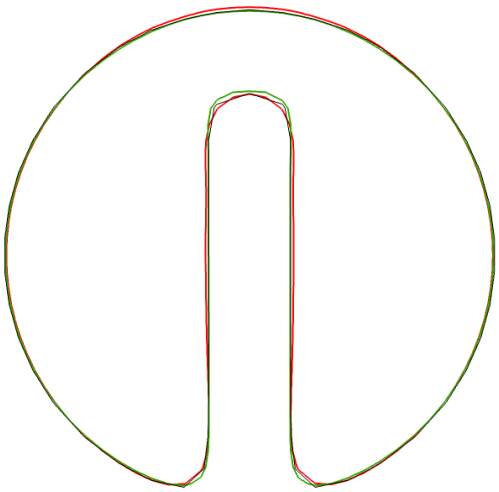


Figure 2. Zalesak disk after 50 rotations using a 100×100 mesh with varying number of DOF (N=2: Red, N=3: Green, N=4: Blue, Initial condition: Black)

also investigated by changing the mesh from 64×64 points to 128×128 points and 32×32 points while keeping the number of DOF at three. Fig. 9 in Section 5 shows that as the number of grid points is increased, the final shape converges to the initial shape as expected. At maximum deformation, the 64×64 and 32×32 cases show significant mass loss, but at the end of the simulation the mass errors are noticeably lower indicating that mass is preserved at the sub-cell level due to the DG scheme.

Degrees of Freedom	2	3	4	6
Error of Final Area [%]	2.8	1.4	0.3	0.1

Table 1. Percent error of final area for two-dimensional deformation simulations with varying number of DOF

3D deformation

A three-dimensional version of the second test case was also performed. For this case a sphere with diameter 0.3 and initial center at $(x, y, z) = (0, 0.25, 0.25)$ is subjected to the three-dimensional velocity field,

$$\begin{aligned} U &= 2\sin(\pi x)^2 \sin(2\pi y) \sin(2\pi z) \cos(\pi t/3), \quad (27) \\ V &= -\sin(2\pi x) \sin(\pi y)^2 \sin(2\pi z) \cos(\pi t/3), \\ V &= -\sin(2\pi x) \sin(2\pi y) \sin(\pi z)^2 \cos(\pi t/3). \end{aligned}$$

A mesh of $64 \times 64 \times 64$ was used with varying number of DOF. Fig. 10 in Section 5 shows the evolution of the initial sphere with time for the case of four DOF. The figures show that when the droplet is stretched there appears to be significant mass loss. However, comparing Fig. 10.b and 10.h there is very little actual mass loss. This is expected and is a result of the sub-cell resolution of the DG level scheme. The volume of the liquid at the end of the simulation was compared with the initial volume for varying number of DOF and the result is shown in Table 2. The results again confirms that as the number of DOF are increased the amount of mass loss is reduced.

Degrees of Freedom	2	3	4
Error of Final Volume [%]	5.8	2.4	1.7

Table 2. Percent error of final volume for three-dimensional deformation simulations with varying number of DOF

Water droplet impacting quiescent surface

The final case shows the interaction of the velocity solver and the level set transport scheme. Fig. 11 in Section 5 shows a water droplet falling through

air and colliding with a quiescent body of water. The droplet had an initial radius of 0.01, center at $(x, y, z) = (0, 0.06, 0)$, and velocity of $0\hat{i}, -1\hat{j}, 0\hat{k}$. The pool had no initial velocity and a surface height of 0.02 from the bottom of the domain. The domain size was $[-0.0375, 0.0375]$ in all directions. The simulation was performed using a mesh of $64 \times 64 \times 64$. Results are as expected however the surface in Fig. 11.c shows slight oscillations that are not physical. It is believed that reinitialization and a limiter will improve numerical robustness and reduce the oscillations.

DG Level set Reinitialization

Successful reinitialization of the level set will result in a signed distance function with a zero isosurface of the level set in the initial location. To test our quadrature-free level set reinitialization a couple one-dimensional test cases were performed. Fig. 3 shows how an initial function, $G_o(x) = 0.05 \sinh(18x/5)$, with very low slope near $x = 0$, and high slope near $x = \pm 1$, evolves using five grid cells and four DOF. As expected the function becomes a straight line with a slope of one and the interface location stays in the same location. A simulation with the same parameters but 49 grid points was also performed and the result is shown in Fig. 4. This plot shows that the method works on large number of grid cells. Additionally, the desired slope of unity is obtained near the origin initially and then spreads throughout the domain. This feature allows for reinitialization to be performed for fewer number of steps and still achieve the sought-after gradient near the interface. To quantify the accuracy of our scheme, L_2 error was calculated using,

$$L_2 = \sum_{i}^{N} \left((G(x_i) - G_{\text{exact}}(x_i)) \right)^2, \qquad (28)$$

for N equal to the number of grid cells. Results are shown in Table 3 for simulations using 5, 20, and 49 grid points and various DOF. Machine zero error was achieved in all cases, which is expected because the solution is entirely included in the polynomial basis.

A second test was conducted using a function with two interfaces. The function used to initialize the simulation was $G_o(x) = (0.6x)^2 - 0.15$. Two meshes were used, the first had five points and the result are shown in Fig. 5, the second is shown in Fig. 6 and was done with 49 points. Again we find that the scheme accurately reinitializes the level set and is able to deal with the multiple interfaces. It should be noted that one modification was made to the scheme. During the calculation of $\underline{f}$ in Equation 24 the terms include division by some of the $\underline{g}$ terms defined in Equation 2. When multiple interfaces exist a division by zero can occur. When this happens the f terms have been set to zero and only the zeroth order stability terms, which appear as the last four lines in Equation 14, are kept.

		Degrees of freedom			
		2	**3**	**4**	**5**
Grid Cells	**5**	6.46E-14	6.49E-15	6.47E-15	1.16E-14
	20	3.68E-14	4.34E-14	3.53E-14	8.30E-14
	49	9.73E-14	1.46E-13	7.94E-14	2.26E-13

Table 3. L_2 error calculated by the difference between the converged solution and the exact solution of the level set reinitialization equation for varying number of DOF and mesh sizes

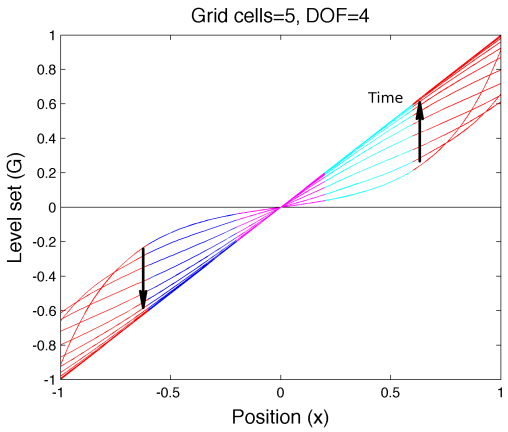


Figure 3. Reinitialization using 5 grid cells (colors correspond to different grid cells)

Conclusions and Future Work

Derivations and results from our implementation of a quadrature-free level set transport and reinitialization scheme were shown in this paper. The three-dimensional implementation of the level set transport and the one-dimensional level set reinitialization are working as desired and show excellent potential to be implemented in a fully three-dimensional multiphase flow solver.

Current work is focused on extending the reinitialization scheme in three-dimensions. Once the three-dimensional reinitialization method is developed and implemented, we expect the scheme to demonstrate excellent mass conservation properties by resolving the interface location at the sub-cell level. The scheme will be integrated with NGA and allow for the simulation of turbulent multiphase

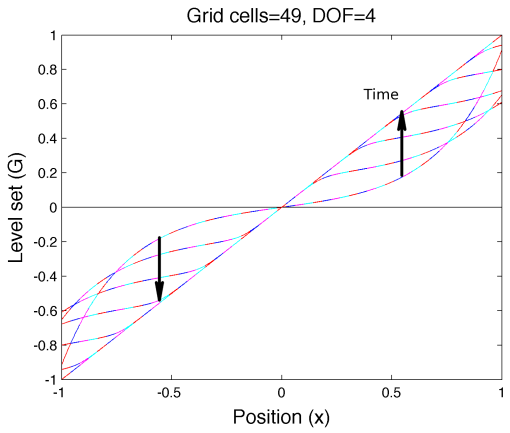


Figure 4. Reinitialization using 49 grid cells (colors correspond to different grid cells)

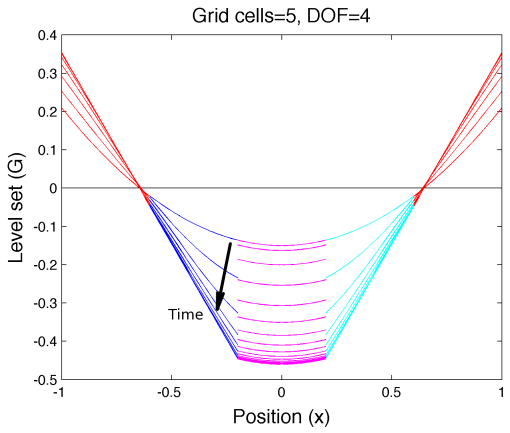


Figure 5. Reinitialization of function with two interfaces using 5 grid cells (colors correspond to different grid cells)

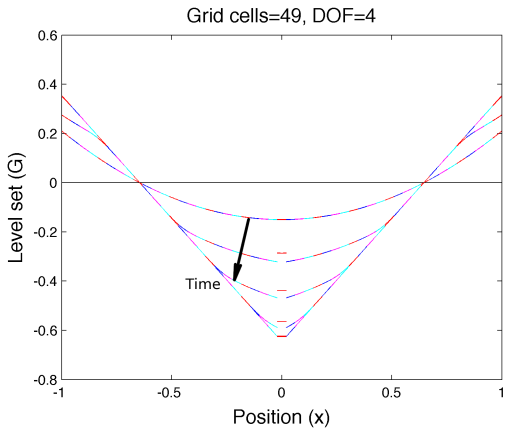


Figure 6. Reinitialization of function with two interfaces using 49 grid cells (colors correspond to different grid cells)

flows using physical parameters and will be capable of resolving small scale structures characteristic of these flows.

References

[1] J. U. Brackbill, D. B. Kothe, and C. Zemach. *J. Comput. Phys.*, 100(2):335–354, June 1992.

[2] Ronald P. Fedkiw, Tariq Aslam, Barry Merriman, and Stanley Osher. *J. Comput. Phys.*, 152(2):457–492, 1999.

[3] W. Dettmer, P. H. Saksono, and D. Perić. *Commun. Numer. Meth. En.*, 19(9):659–668, 2003.

[4] C. W. Hirt, A. A. Amsden, and J. L. Cook. *J. Comput. Phys.*, 135(2):203–216, 1997.

[5] Thomas J. R. Hughes, Wing Kam Liu, and Thomas K. Zimmermann. *Comput. Method. Appl. M.*, 29(3):329–349, December 1981.

[6] Josep Sarrate, Antonio Huerta, and Jean Donea. *Comput. Method. Appl. M.*, 190(24-25):3171–3188, March 2001.

[7] Murray Rudman. *Int. J. Numer. Meth. Fl.*, 24(7):671–691, 1997.

[8] C Hirt and B Nichols. *J. Comput. Phys.*, 39(1):201–225, 1981.

[9] James Edward Pilliod and Elbridge Gerry Puckett. *J. Comput. Phys.*, 199(2):465–502, September 2004.

[10] Ruben Scardovelli and Stephane Zaleski. *Annu. Rev. Fluid Mech.*, 31(1):567–603, 1999.

[11] Stanley Osher and James A Sethian. *J. Comput. Phys.*, 79(1):12–49, November 1988.

[12] J Sethian. Level set methods and fast marching methods: Evolving interfaces in computational geometry, 1998.

[13] Emilie Marchandise, Philippe Geuzaine, Nicolas Chevaugeon, and Jean-Franois Remacle. *J. Comput. Phys.*, 225(1):949 – 974, July 2007.

[14] Emilie Marchandise, Jean-Franois Remacle, and Nicolas Chevaugeon. *J. Comput. Phys.*, 212(1):338–357, February 2006.

[15] Patrick Rasetarinera and M. Y. Hussaini. *J. Comput. Phys.*, 172(2):718–738, September 2001.

[16] Jean-Franois Remacle, Joseph E. Flaherty, and Mark S. Shephard. *SIAM Review*, 45(1):53–72, March 2003.

[17] David L. Chopp. *SIAM J. Sci. Comput.*, 23(1):230–244, 2001.

[18] Sunil Arya, David M. Mount, Nathan S. Netanyahu, Ruth Silverman, and Angela Y. Wu. *J. ACM*, 45(6):891–923, 1998.

[19] John Strain. *J. Comput. Phys.*, 161(2):512–536, 2000.

[20] C. Hu and C. W Shu. *SIAM J. Sci. Comput.*, 21(2):666 – 690, 2000.

[21] Y. Cheng and C. W Shu. *J. Comput. Phys.*, 223(1):398 – 415, 2007.

[22] Olivier Desjardins, Guillaume Blanquart, Guillaume Balarac, and Heinz Pitsch. *J. Comput. Phys.*, 227(15):7125–7159, July 2008.

Supplemental Figures

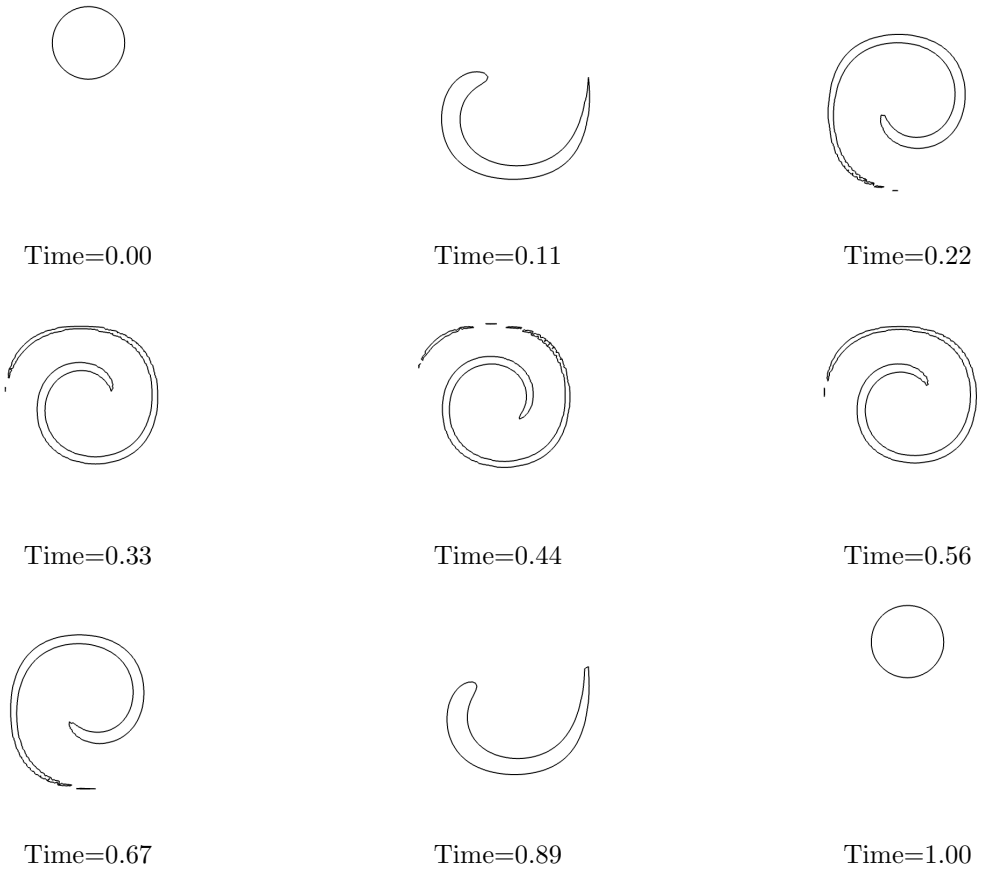
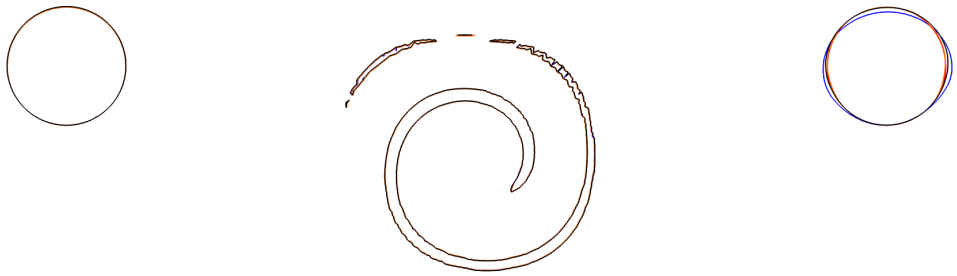
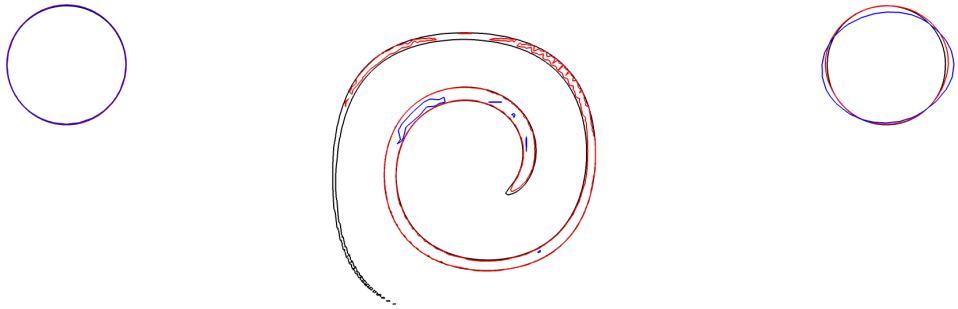


Figure 7. Two-dimensional deformation test case, interface identified using the zero level set iso-surface is plotted, results from simulation with six DOF



Time=0.0 Time=0.5 Time=1.0

Figure 8. Two-dimensional deformation test case, results from simulation with 64×64 grid points and varying DOF (Red: N=6, Orange: N=4, Black: N=3, Blue: N=2)



Time=0.0 Time=0.5 Time=1.0

Figure 9. Two-dimensional deformation test case, results from simulation with three DOF and varying mesh points (Black:128×128 points, Red: 64×64 points, Blue: 32×32 points)

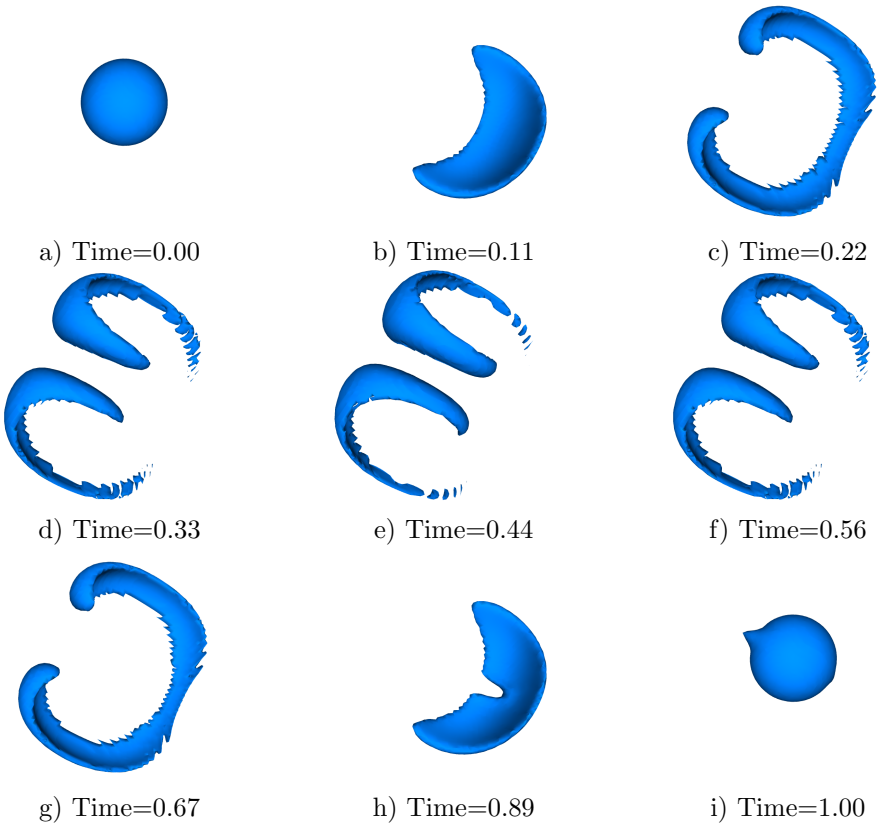


Figure 10. Three-dimensional deformation test case, results from simulation with four DOF. Plot of zero iso-surface shows how the surface changes with time.

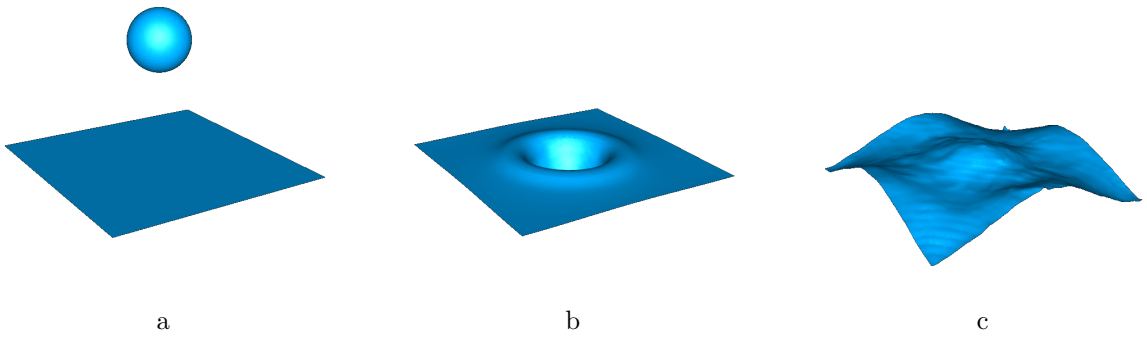


Figure 11. Water droplet colliding with quiescent body of water, interface plotted using zero iso-surface of level set