AN EFFICIENT, INTERFACE PRESERVING LEVEL SET RE-DISTANCING ALGORITHM AND ITS APPLICATION TO INTERFACIAL INCOMPRESSIBLE FLUID FLOW

MARK SUSSMAN* AND EMAD FATEMI†

This paper is dedicated to the memory of Dr. Emad Fatemi who was a very kind person and a truly original scientist.

Abstract. In Sussman, Smereka, & Osher (1994), A numerical scheme was presented for computing incompressible air-water flows using the level set method. Crucial to the above method was a new iteration method for maintaining the level set function as the signed distance from the zero level set. In this paper we implement a "constraint" along with higher order difference schemes in order to make the iteration method more accurate and efficient. Accuracy is measured in terms of the new computed signed distance function and the original level set function having the same zero level set. We apply our redistancing scheme to incompressible flows with noticeably better resolved results at reduced cost. We validate our results with experiment and theory. We show that our "distance level set scheme" with the added constraint competes well with available interface tracking schemes for basic advection of an interface. We perform basic accuracy checks and more stringent tests involving complicated interfacial structures. As with all level set schemes, our method is easy to implement.

Key words. Distance function, Incompressible, Level Sets

AMS subject classifications. 65M06, 76D05, 76T05

1. Introduction. Given an interface separating two regions in space, one would like to be able to efficiently compute the distance to the closest point on the interface from many points surrounding the interface. If the interface is moving, as in propagation of an air/water interface, one would like to use the previous distance function in order to speed up the computation of distance from the new interface. For applications of incompressible two-phase flow [15, 14], it is important to know the distance from an air/water interface in a small strip about the interface. This is needed for robustly computing with stiff surface tension effects. In [2], the distance from the zero level set, the interface, is needed in a small tube about the zero level set for determining which points need to be updated during propagation of the level set function. Other applications that need the shortest distance to a curve involve computer aided design (CAD, shape offsets see [7]), and computing minimal surfaces (see [3]). A brute force approach for finding the signed distance in a strip of k cells about the interface could be the following scheme:

1. Assume the computational domain is discretized into cells (i,j). We wish to assign the variable $d_{i,j}$ the value that is the signed distance between the closest point on the interface and the the center of cell (i,j).
2. Represent the interface as a collection of piecewise linear segments (a cell can contain a maximum of one linear segment).
3. Let $d_{i,j} = +\infty$ if the center of cell (i,j) lies on one side of the interface and let $d_{i,j} = -\infty$ if the center of cell (i,j) lies on the other side.

* Department of Mathematics, University of California Davis, Davis, CA 95616 (sussman@math.ucdavis.edu).
Work performed under the auspices of the U.S. Department of Energy by the Lawrence Livermore National Laboratory under contract No. W-7405-Eng-48. Support under contract No. W-7405-Eng-48 was provided by the Applied Mathematical Sciences Program of the Office of Energy Research. Center for Computational Sciences and Engineering, LLNL, Livermore, CA 94550

† work supported in part by NSF # DMS 94-04942, DARPA URI-ONR-N00014-92-J-1890. Dept. of Math, UCLA, Los Angeles, CA 90024-1555

4. For each cell (i,j) that contains a linear segment of the interface, update the values of $d_{i',j'}$ within the strip $|i-i'|<k$ and $|j-j'|<k$. The new value of $d_{i',j'}$ will be the signed distance that has a magnitude equal to the smaller of the original value of $|d_{i',j'}|$ and the distance from the line segment in cell (i,j) to the center of cell (i',j').

The above scheme has two drawbacks. First, the above process has to be repeated every time, even if the interface has moved a small amount. Secondly, the above method involves explicitly reconstructing the interface which can introduce error (see table 6.1, error for least squares reconstruction vs. L1 w/fix).

More elegant means for finding the signed distance can be found in the work of [4], [7], [6] and [12]. In the work of [4], an $O(N)$ method for redistancing the *whole* domain (N cells total) was devised in which the distance from "marked" cells (pixels) were computed. Unfortunately, this method would not yield sub-pixel information; thus the method is $O(\Delta x)$ accurate.

In [7, 6, 12], the levelset method is used to represent the interface between two regions. In level set methods [9], one represents an n-1 dimensional interface as the zero contour of an n-dimensional smooth function. For any interface, one can find a reasonably smooth level set function ϕ such that $\phi(x,y)=0$ on the interface. We note that a levelset function is a distance function if $|\nabla\phi|=1$. The redistance scheme in [7] would advect ϕ normal to itself for a specified time t. One then would know that all points on the zero level set of $\phi(x,y,t)$ are a distance t from the interface. This process can be repeated up to the required time. In the work done concurrently by [6, 9], "fast-marching" methods were presented in which the equation $|\nabla\phi|=1$ could be solved over the whole computational domain in O(N log N) operations. In "fast-marching" methods, a binary tree data structure is created that allows one to march through cells in a special order that allows only one pass per cell.

As opposed to the methods described above, the scheme presented in this paper is an iteration scheme designed to enforce $\mid \nabla\phi \mid = 1$. The advantage of an iteration scheme is the fact that if the interface moves a little, one can effectively use information from the previous value of distance to update the new values. Common to the levelset redistance methods described above [9, 6] is the fact that the zero levelset has to be "re-anchored" every time the redistance operation is called. This is done by piecewise linear reconstruction of the zero-contour or by other means (see [1] page 6). Every time the interface is "re-anchored" the error is accumulated.

In [11], an iteration method based on the level set method was used to solve the equation

$$\begin{aligned} \mid \nabla\phi \mid &= \lambda(x) \text{ in } \Omega^+ \\ \phi &= 0 \text{ on } \partial\Omega^+ \end{aligned}$$

The method presented in [11] was designed for shape from shading applications. If $\lambda(x)\equiv 1$, then the method can be applied to finding the distance from the zero level set on one side of the interface.

In [14], we presented a scheme in the same spirit as the scheme of [11], for maintaining the values of a level set function as the signed distance from the interface. The following equation was solved until $\mid \nabla\phi \mid = 1+O(h^2)$:

$$\begin{aligned} \phi(x,0) &= \phi_0 \\ \phi_t &= \text{sign}(\phi_0)(1- \mid \nabla\phi \mid) \end{aligned}$$

This enabled us to solve for the signed distance on *both* sides of the interface. In this paper, we improve the above scheme in the following ways:

- We formulate a constraint designed to prevent the straying of the zero level set from the initial position even after many iterations. This is very important, because we do not want errors to accumulate due to repeated redistance operations on a level set function that changes little (as when used in conjunction with the advection equation for ϕ) in between operations.

- We will demonstrate (see figure 9.1) that we only need to solve the above equation up to time $t = L$ where L is the thickness of the strip about the interface in which we need a distance function.
- We use high order methods in time (Runge-Kutta) as well as space for solving the above equation.

2. Reinitialization. We assume that we have an interface defined implicitly by the equation

$$\phi_0(x, y) = 0 \tag{2.1}$$

The function, ϕ_0 is a level set function, but it is not a distance function. In order to compute the distance function from the initial ϕ_0, we can solve the following time dependent PDE

$$\phi_t = \mathrm{sign}(\phi_0)(1 - |\nabla\phi|) \tag{2.2}$$

with the initial condition

$$\phi(x, y, 0) = \phi_0(x, y).$$

The steady state solution of this problem is the signed distance function from the boundary of the curve defined implicitly by equation 2.1.

To gain intuition about the above PDE, we can solve the above problem for small time using the method of characteristics. In the domain where $\phi_0(x, y)$ is positive, the solution is

$$\phi(x, y, t) = \begin{cases} t + \phi_0(x - tp/\sqrt{p^2+q^2}, y - tq/\sqrt{p^2+q^2}) & \text{if } t \le t_0 \\ t_0 & \text{if } t > t_0 \end{cases} \tag{2.3}$$

$$p = \partial_x \phi_0(x - tp/\sqrt{p^2+q^2}, y - tq/\sqrt{p^2+q^2})$$

$$q = \partial_y \phi_0(x - tp/\sqrt{p^2+q^2}, y - tq/\sqrt{p^2+q^2})$$

$$\begin{aligned} t_0 = {} & \text{Shortest distance from } (x, y) \text{ to the zero level set} \\ & \phi_0(x - t_0 p_0/\sqrt{p_0^2+q_0^2}, y - t_0 q_0/\sqrt{p_0^2+q_0^2}) = 0 \end{aligned}$$

In the domain where $\phi_0(x, y)$ is negative, we have

$$\phi(x, y, t) = \begin{cases} -t + \phi_0(x + tp/\sqrt{p^2+q^2}, y + tq/\sqrt{p^2+q^2}) & \text{if } t \le t_0 \\ -t_0 & \text{if } t > t_0 \end{cases} \tag{2.4}$$

$$p = \partial_x \phi_0(x + tp/\sqrt{p^2+q^2}, y + tq/\sqrt{p^2+q^2})$$

$$q = \partial_y \phi_0(x + tp/\sqrt{p^2+q^2}, y + tq/\sqrt{p^2+q^2})$$

$$\begin{aligned} t_0 = {} & \text{Shortest distance from } (x, y) \text{ to the zero level set} \\ & \phi_0(x + t_0 p_0/\sqrt{p_0^2+q_0^2}, y + t_0 q_0/\sqrt{p_0^2+q_0^2}) = 0 \end{aligned}$$

If we assume regularity of the initial condition ϕ_0 and its first and second derivatives, we can solve for (p,q) for small time, t, using the implicit function theorem for the last two equations. Near the boundary, defined by $\phi_0(x,y)=0$ the solution is extended from the point (x_0,y_0) by

$$\phi(x_0+tn_x(x_0,y_0),y_0+tn_y(x_0,y_0))=t$$

$$\phi(x_0-tn_x(x_0,y_0),y_0-tn_y(x_0,y_0))=-t$$

where

$$(n_x,n_y)=(\phi_{0x},\phi_{0y})/\sqrt{\phi_{0x}^2+\phi_{0y}^2}.$$

We can see that this defines a solution near the boundary for time small enough such that

$$t \quad \max|\kappa| < \quad 1$$

where κ is the curvature of the boundary defined by $\phi_0(x,y)=0$. If the boundary is parameterized by $x(s),y(s)$, then the map

$$x(s,t)=x(s)\pm tn_x(s)$$

$$y(s,t)=y(s)\pm tn_y(s)$$

$$\phi(s,t)\equiv\phi(s(x,y),t(x,y))=\pm t$$

can be defined as long as we can invert the transformation to solve for (s,t) in terms of (x,y). For $t|\kappa|\le 1$, the determinant of the Jacobian,

$$det(\frac{\partial(x,y)}{\partial(s,t)})=\pm\sqrt{x_s^2+y_s^2}(1\pm t\kappa),$$

is not zero. The Jacobian is computed using the fact that n_x, n_y and κ can be represented as:

$$\begin{aligned} n_x &= \frac{y_s}{\sqrt{x_s^2+y_s^2}} \\ n_y &= \frac{-x_s}{\sqrt{x_s^2+y_s^2}} \\ \kappa &= \frac{y_{ss}x_s-x_{ss}y_s}{(x_s^2+y_s^2)^{\frac{3}{2}}}. \end{aligned}$$

3. Gradient Projection . The evolution equation for the interface

$$\begin{aligned} \phi_t = L(\phi_0,\phi) &= \text{sign}(\phi_0)(1-|\nabla\phi|) \\ \phi(x,0) &= \phi_0(x) \end{aligned} \tag{3.1}$$

conserves the volume of the domain bounded by the curve defined implicitly by the equation $\phi_0(x,y)$. This is due to the fact that it does not change the position of the boundary. In numerical computations this is not true anymore. We conserve the volume of the domain by requiring that:

$$\partial_t\int_\Omega H(\phi)=0 \tag{3.2}$$

where H is a smoothed out approximation to the sign function and Ω is any fixed domain. We modify the evolution equation by

$$\phi_t = L(\phi_0, \phi) + \lambda f(\phi) \tag{3.3}$$

λ is a function of t only, determined by requiring

$$\partial_t \int_\Omega H(\phi) = \int_\Omega H'(\phi)\phi_t = \int_\Omega H'(\phi)(L(\phi_0, \phi) + \lambda f(\phi)) = 0. \tag{3.4}$$

Then λ is calculated to be

$$\lambda = \frac{-\int_\Omega H'(\phi)L(\phi_0, \phi)}{\int_\Omega H'(\phi)f(\phi)}. \tag{3.5}$$

In our calculations we choose

$$f(\phi) \equiv H'(\phi)|\nabla\phi|. \tag{3.6}$$

This insures that we only correct at the interface, without disturbing the distance function property away from the interface.

We note that if (3.1) is solved perfectly then λ will be zero. This is because $L(\phi_0, \phi)$ as it appears in (3.1) will be zero in regions where $H'(\phi)$ is not zero (the zero levelset of ϕ). Discretely, this is not true anymore, since the zero levelset of ϕ_0 may differ from that of ϕ due to numerical error.

Since the zero level set should be preserved by the reinitialization step, we require numerically that the mass remain unchanged in any subset of the domain Ω. For numerical purposes, when we discretize the above equation, we wish to preserve the value $\int H(\phi)$ in every grid cell $\Omega_{ij} = ((x,y)|x_{i-1/2} < x < x_{i+1/2}$ and $y_{j-1/2} < y < y_{j+1/2})$. In light of this, our new form for equations (3.3) and (3.5) is:

$$\phi_t = L(\phi_0, \phi) + \lambda_{ij} f(\phi) \text{ for } \vec{x} \text{ in } \Omega_{ij} \tag{3.7}$$

$$\lambda_{ij} = \frac{-\int_{\Omega_{ij}} H'(\phi)L(\phi_0, \phi)}{\int_{\Omega_{ij}} H'(\phi)f(\phi)} \tag{3.8}$$

λ_{ij} is assumed to be piecewise constant; constant in each cell Ω_{ij}. Thus, at a discrete level, λ is a function of both space and time, vanishing outside of a small neighborhood of the front.

4. Numerical Implementation . We describe how to discretize (3.7) and (3.8) for ϕ_0 already close to a distance function. That is, $| 1- | \nabla\phi_0 ||= O(\Delta t^{advect})$ near the interface ($|\phi_0| < \Delta x$). This will be the case for problems in which the levelset function ϕ is advected due to a velocity field, and one needs to maintain the levelset function as a distance function. If one starts off a problem with a levelset function that is far from a distance function (e.g. $\phi = 1$ in fluid 1 and $\phi = -1$ in fluid 2), then an initial "once-only" redistance step must be performed at $t = 0$. We refer the reader to methods presented by [4, 6, 12] and ourselves (see Appendix A) for doing the initial redistance step. While our method for doing the "once-only" redistance step may not be the most efficient, it requires the least extra programming. It is a slight modification of the method presented here for the case when ϕ_0 is already close to a distance function.

We assume $h = \Delta x = \Delta y$, where h is the spacing between grid points.

We first define the discretized version of the Heaviside function $H(\phi)$ and sign function $\mathrm{sign}(\phi_0)$ as they appear in (3.7) and (3.8):

$$H_{\Delta x}(\phi) \equiv \begin{cases} 1 & \text{if } \phi > \Delta x \\ 0 & \text{if } \phi < -\Delta x \\ \frac{1}{2}(1 + \frac{\phi}{\Delta x} + \frac{1}{\pi}\sin(\pi\phi/\Delta x)) & \text{otherwise} \end{cases} \tag{4.1}$$

$$\mathrm{sign}_{\Delta x}(\phi) \equiv 2(H_{\Delta x}(\phi) - 1/2) \tag{4.2}$$

If we want to recover the distance function a distance $\alpha\Delta x$ from the zero level set of ϕ, we need to solve equation (3.1) for $t = 0 \ldots \alpha\Delta x$. This is apparent if we put equation (3.1) in the form

$$\phi_t + \vec{w}\cdot\nabla\phi = \mathrm{sign}(\phi_0) \tag{4.3}$$

$$\vec{w} = (\nabla\phi / \mid \nabla\phi \mid)\mathrm{sign}(\phi_0) \tag{4.4}$$

The vector $\vec{w}$ has magnitude one and points away from the zero level set, hence the characteristics propagate away from the interface with speed one. This can be seen in figure 9.1 where an initially discontinuous levelset function becomes a distance function for points within $\alpha\Delta x$ of the initial levelset function after time $t = \alpha\Delta x$. We also deduce from equation (4.4) that a valid time-step obeying the CFL condition is $\Delta t = \Delta x/2$. Equation 4.4 has the form of an advection equation with velocity $\vec{w}$ so we use upwinded ENO type schemes (see [13, 5]) to approximate the spatial derivatives. Given ϕ_n we solve for $\tilde{\phi}_{n+1}$ using second or third order ENO plus second or third order Runge-Kutta.

4.1. First order discretization . We will present the first order method below and refer the reader to Appendix B for a description of the third order method. For first order we have:

1. Let ϕ_0 be initial data at time $t_0 = 0$. Repeat the following steps up to $t_N = \alpha\Delta x$.
2. compute an approximation to $\mid \nabla\phi_n \mid$.
   (a) Let

   $$D_x^+\phi \equiv \frac{\phi_{i+1,j} - \phi_{i,j}}{\Delta x} \tag{4.5}$$

   and

   $$D_x^-\phi \equiv \frac{\phi_{i,j} - \phi_{i-1,j}}{\Delta x}. \tag{4.6}$$

   (b)

   $$\frac{\partial\phi_n}{\partial x} \approx \begin{cases} D_x^+\phi_n & \text{if } D_x^+\phi_n\mathrm{sign}(\phi_0) < 0 \text{ and } (D_x^-\phi_n + D_x^+\phi_n)\mathrm{sign}(\phi_0) < 0 \\ D_x^-\phi_n & \text{if } D_x^-\phi_n\mathrm{sign}(\phi_0) > 0 \text{ and } (D_x^+\phi_n + D_x^-\phi_n)\mathrm{sign}(\phi_0) > 0 \\ 0 & \text{if } D_x^-\phi_n\mathrm{sign}(\phi_0) < 0 \text{ and } D_x^+\phi_n\mathrm{sign}(\phi_0) > 0 \end{cases}$$

   The approximation for $\frac{\partial\phi_n}{\partial y}$ is computed in a similar manner.
3. Let $L(\phi_0, \phi_n) = \mathrm{sign}(\phi_0)(1 - \mid \nabla\phi_n \mid)$.
4. Let $\tilde{\phi}_{n+1} = \phi_n + (\Delta t)L(\phi_0, \phi_n)$
5. Gradient projection step:

   $$\phi_{n+1} = \tilde{\phi}_{n+1} + \Delta t\lambda_{i,j}H'_{\Delta x}(\phi_0) \mid \nabla\phi_0 \mid \tag{4.7}$$

   where,

   $$\lambda_{i,j} = \frac{-\int_{\Omega_{ij}} H'_{\Delta x}(\phi_0)\frac{\tilde{\phi}_{n+1}-\phi_0}{\Delta t}}{\int_{\Omega_{ij}}[H'_{\Delta x}(\phi_0)]^2 \mid \nabla\phi_0 \mid}. \tag{4.8}$$

The numerical integration over the domain

$$\Omega_{ij} = ((x,y)|x_{i-1/2} < x < x_{i+1/2} \text{ and } y_{j-1/2} < y < y_{j+1/2})$$

is computed using a nine point stencil:

$$\int_{\Omega_{ij}} g \approx \frac{h^2}{24}(16 g_{ij} + \sum_{m,n=-1;(m,n)\neq(0,0)}^{1} g_{i+m,j+n}).$$

4.2. Discretization of the Constraint . In (4.7) and (4.8), we chose to discretize the constraint (see (3.7) and (3.8) for the non-discretized formulation) by discretizing $H'(\phi)$ as $H'_{\Delta x}(\phi_0)$, $L(\phi_0, \phi_n)$ as $\frac{\tilde{\phi}_{n+1}-\phi_0}{\Delta t}$ and $f(\phi)$ as $H'_{\Delta x}(\phi_0) \mid \nabla\phi_0 \mid$. For high order Runge-Kutta methods where we have multiple "predictor" steps per time-step (see Appendix B), we still only apply the gradient projection (constraint) once per timestep. Thus, for an r-th order Runge-Kutta scheme, we have:

$$\phi^{(i)} = \sum_{k=0}^{i-1} \alpha_{ik}\phi^{(k)} + \beta_{ik}\Delta t L(\phi_0, \phi^{(k)}), i = 1 \ldots r$$

$$\phi^{(0)} = \phi_n, \tilde{\phi}_{n+1} = \phi^{(r)}$$

The constraint is then applied as in (4.7) to $\tilde{\phi}_{n+1}$.

In this section we first give a justification as to why our constraint:

- maintains the zero levelset of ϕ_n to be very close to that of ϕ_0.
- Does not disturb the distance property of ϕ_n. That is to say, if $|\nabla\tilde{\phi}^n| = 1 + O(\Delta t^{advect})$, then $|\nabla\phi^n| = 1 + O(\Delta t^{advect})$.

We designed the discretization of the constraint in such a way as to remove the leading order term of the error in the quantity

$$\int_{\Omega_{ij}} (H_{\Delta x}(\phi_{n+1}) - H_{\Delta x}(\phi_0)). \tag{4.9}$$

If we write the Taylor expansion of (4.9) we have:

$$\begin{aligned} &\int_{\Omega_{ij}} H_{\Delta x}(\phi_{n+1}) - \int_{\Omega_{ij}} H_{\Delta x}(\phi_0) = \\ &\int_{\Omega_{ij}} H'_{\Delta x}(\phi_0)(\phi_{n+1} - \phi_0) + \int_{\Omega_{ij}} H''_{\Delta x}(\phi_0)(\phi_{n+1} - \phi_0)^2/2 + \ldots \end{aligned} \tag{4.10}$$

If we assume that $\lambda_{i,j}$ is piecewise constant in each cell $\Omega_{i,j}$, then the leading order term,

$$\int_{\Omega_{ij}} H'_{\Delta x}(\phi_0)(\phi_{n+1} - \phi_0)$$

can be written as,

$$\int_{\Omega_{ij}} H'_{\Delta x}(\phi_0)(\tilde{\phi}_{n+1} + \Delta t\lambda_{i,j}H'(\phi_0) \mid \nabla\phi_0 \mid - \phi_0) =$$

$$\Delta t[\int_{\Omega_{ij}} (H'_{\Delta x}(\phi_0)\frac{\tilde{\phi}_{n+1} - \phi_0}{\Delta t}) + \lambda_{i,j}\int_{\Omega_{ij}} ([H'_{\Delta x}(\phi_0)]^2 \mid \nabla\phi_0 \mid)]$$

If we plug in our expression for $\lambda_{i,j}$ (4.8), we have cancellation and the above leading order term is zero.

Since we assumed at the outset of this section that $\mid 1-\mid \nabla\phi_0 \mid\mid = O(\Delta t^{advect})$ near the interface ($|\phi_0| < \Delta x$), we can provide a justification of why the constraint does not disturb the distance property of ϕ_n. For this discussion, we only need to concern ourselves with points in which the constraint term is not zero; that is, $H'_{\Delta x}(\phi_0) \neq 0$. This will be the case if $|\phi_0| \leq \Delta x$. For our explanation, we assume that:

- $|\nabla\phi_{n-1}| = 1 + O(\Delta t^{advect})$
- Due to the fact that the constraint removes the leading order term of the mass error,

$$\int_{\Omega_{i,j}} (\phi_{n-1} - \phi_0) H'_{\Delta x}(\phi_0) = 0,$$

  we assume that we have the bound

$$|\phi_{n-1} - \phi_0| = O(h\Delta t^{advect}) \tag{4.11}$$

  for $|\phi_0| < \Delta x$.

We first note that since $|\nabla\phi_{n-1}| = 1 + O(h\Delta t^{advect})$ then we assume,

$$|\nabla\tilde{\phi}_n| = 1 + O(h\Delta t^{advect}).$$

According to [11] (see eqn. (11) of [11]), the first order discretization

$$\tilde{\phi}_n = \phi_{n-1} + \Delta t(1 - |\nabla\phi_{n-1}|), \tag{4.12}$$

is a monotone, consistent scheme that leads to convergence of $|\nabla\tilde{\phi}_n| - 1$ to zero. As pointed out in [14], this can carry over to the case when ϕ is positive or negative:

$$\tilde{\phi}_n = \phi_{n-1} + \Delta t S_{\Delta x}(\phi_0)(1 - |\nabla\phi_{n-1}|). \tag{4.13}$$

So, if $|\nabla\phi_{n-1}| = 1 + O(\Delta t^{advect})$, then we will also have $|\nabla\tilde{\phi}_n| = 1 + O(\Delta t^{advect})$.

We also gather from (4.13), that since $|\nabla\phi_{n-1}| = 1 + O(\Delta t^{advect})$, we have $|\tilde{\phi}_n - \phi_{n-1}| = O(h\Delta t^{advect})$. The assumption in (4.11) implies that

$$|\tilde{\phi}_n - \phi_0| < |\tilde{\phi}_n - \phi_{n-1}| + |\phi_{n-1} - \phi_0| = O(h\Delta t^{advect}). \tag{4.14}$$

In order to show that the constraint term does not disturb the distance function property near the interface, we will show that the constraint term is $O(h\Delta t^{advect})$.

From (4.7), we have:

$$\phi_n = \tilde{\phi}_n + \Delta t\lambda H'_{\Delta x}(\phi_0)|\nabla\phi_0|. \tag{4.15}$$

By making use of the fact that $|\nabla\phi_0| - 1 = O(\Delta t^{advect})$ and also making use of the fact that $\frac{\tilde{\phi}_n - \phi_0}{\Delta t} = O(\Delta t^{advect})$ (4.14), we have the resulting discretization of the constraint term (denoting $H'_{\Delta x}((\phi_0)_{i,j})$ by $H'_{i,j}$):

$$\Delta t|\lambda_{i,j} H'_{\Delta x}((\phi_0)_{i,j})|\nabla\phi_0|_{i,j}| \approx \Delta t|\lambda_{i,j} H'_{\Delta x}((\phi_0)_{i,j})| =$$

$$\Delta t \frac{\int_{\Omega_{ij}} H'_{\Delta x}(\phi_0)|\frac{\tilde{\phi}_n-\phi_0}{\Delta t}|}{\int_{\Omega_{ij}} H'_{\Delta x}(\phi_0)^2|\nabla\phi_0|} H'_{i,j} <$$
$$h \max_{\Omega_{ij}} |\frac{\tilde{\phi}_n-\phi_0}{\Delta t}| \frac{\int_{\Omega_{ij}} H'_{\Delta x}(\phi_0) H'_{i,j}}{\int_{\Omega_{ij}} H'_{\Delta x}(\phi_0)^2 (1+O(h))} =$$
$$O(h\Delta t^{advect}) \frac{16(H'_{i,j})^2 + \sum_{m,n=-1;(m,n)\neq(0,0)}^{1} H'_{i+m,j+n} H'_{i,j}}{16(H'_{i,j})^2 + \sum_{m,n=-1;(m,n)\neq(0,0)}^{1} (H'_{i+m,j+n})^2} =$$
$$O(h\Delta t^{advect}) \frac{16 + \sum_{m,n=-1;(m,n)\neq(0,0)}^{1} (H'_{i+m,j+n}/H'_{i,j})}{16 + \sum_{m,n=-1;(m,n)\neq(0,0)}^{1} (H'_{i+m,j+n}/H'_{i,j})^2} < 2[O(h\Delta t^{advect})]$$

So, using the fact that the constraint term is $O(h\Delta t^{advect})$, we see from (4.14) and (4.15) that

$$\frac{|\phi_n - \tilde{\phi}_n|}{\Delta x} = O(\Delta t^{advect})$$

and thus,

$$|\nabla\phi_n| = 1 + O(\Delta t^{advect})$$

4.2.1. Redistance step coupled to advection equation. In the discussion above, we showed that the constraint eliminated the leading order term in the mass error (4.11). We also showed that, so long as the following condition was met,

$$|1-|\nabla\phi_0|| < O(\Delta t^{advect}).$$

then the constraint would not disturb the distance function property of $\tilde{\phi}_n$. For applications involving a moving interface we can show that the above bound is satisfied after every advective time step using the following heuristic argument. We assume that we have calculated a distance function in the previous computational cycle, d_0. Then we compute the new level set by computing

$$\phi_t = -u\cdot\nabla\phi \quad \phi(x,0) = d_0(x)$$

The new level set is $\phi_0 = \phi(x, \Delta t^{advect})$.
We claim that $|\nabla\phi_0| = 1 + O(\Delta t^{advect})$. This can be seen by

$$\partial_t|\nabla\phi| = (\nabla\phi/|\nabla\phi|)\cdot\nabla(u\cdot\nabla\phi)$$

Now this is of order one, since $u\cdot\nabla\phi$ is the component of the velocity field normal to the interface and it is continuous even around the interface. Therefore its gradient is of order one and

$$\partial_t|\nabla\phi| = O(1).$$

This implies that $|\nabla\phi_0| = 1 + O(\Delta t^{advect})$. We note, that if Δt^{advect} (time step for the scheme that moves the interface) approaches zero, then our bound on the mass error gets stricter.

4.2.2. Alternative discretizations for the constraint. Other alternatives to our discretization exist. For example, we could choose to replace (4.7) and (4.8) with:

$$\phi_{n+1} = \tilde{\phi}_{n+1} + \Delta t \lambda_{i,j} H'_{\Delta x}(\tilde{\phi}_{n+1})|\nabla\tilde{\phi}_{n+1}|$$
$$\lambda_{i,j} = \frac{-\int_{\Omega_{ij}} H'_{\Delta x}(\tilde{\phi}_{n+1})\frac{\tilde{\phi}_{n+1}-\phi_n}{\Delta t}}{\int_{\Omega_{ij}} [H'_{\Delta x}(\tilde{\phi}_{n+1})]^2|\nabla\tilde{\phi}_{n+1}|}.$$

Unfortunately, the above discretization would not have the cancellation of the error term appearing in (4.10) as does the discretization of (4.7) and (4.8). We could also apply the redistance after each intermediate step of the Runge-Kutta iteration. We have found results in either of the above variations not to be as good as that described in (4.7) and (4.8).

5. Numerical Examples in one dimension. Our tests use a one dimensional version of the above re-distance scheme. We use second order ENO along with second order Runge-Kutta for discretizing the redistance equations. All our 1d tests take place on a periodic domain where $0 \leq x \leq 1$. We discretize the domain with a staggered grid. If n is the number of intervals in the domain, then we have:

$$\Delta x = \frac{1}{n}$$
$$x_i = \Delta x(i + \frac{1}{2}) \qquad i = 0 \ldots n-1$$

As will become apparent below, we find that the redistance scheme is very accurate in one-dimension, and the addition of the constraint does not help much. In fact, in 1d, we do not have the problem where repeated calls to the redistance function will cause the zero level set to stray. The reason that the 1d redistance scheme is so accurate is because the signed distance function near the zero levelset is represented exactly as a linear function of x. If one already has an exact distance function in 1d, then even a first order method would not disturb the distance function property. Thus the constraint is not really needed until we go to higher dimensions.

For the ensuing 1d tests, the L1 error will be measured as

$$\sum_{i=0}^{n-1} |\phi_i^{compute} - \phi_i^{actual}|/n$$

We will do two tests. The first test will involve the redistance of an initially parabolic levelset function

$$\phi_0 = -2(x - 1/4)(x - 3/4). \tag{5.1}$$

The second test will be a test of how the redistance procedure behaves when coupled with the advection of a levelset function.

For the first test we have $\phi(x, 0) = \phi_0(x)$ where ϕ_0 is specified in (5.1). We wish to apply our re-distance scheme such that ϕ will be a distance function on the whole domain. That is:

$$\phi(x)_{t=1} = 1/4 - |x - 1/2|$$

Tables 5.1 and 5.2 contain a convergence study of the L^1 error at $t = 1$. The data displayed in figure 5.2 use values for the number of cells that are one more than those used for the data in figure 5.1. This is to test for possible grid effects. We run cases with and without the constraint and compare the errors. Figure 9.3 contains a graph of the initial profile and the final profile for $\Delta x = 1/21$. As shown in our results, we have second order accuracy for both cases. The error using the constraint is about 3/5 that as without the constraint.

For our second test, we will couple the redistance scheme with the advection of a levelset function. Our main application of the redistance scheme is for maintaining a level set function ϕ as a distance function as the zero level set is advected around. So for our next tests, we have as initial data:

$$\phi_0 = \beta - |x - 1/2|$$
$$u(x) \equiv 1$$

Δx	L1 error no constraint	L1 error constraint	Order
1/20	2.4E-3	1.3E-3	N/A
1/40	6.2E-4	3.1E-4	2.0
1/80	1.6E-4	7.8E-5	2.0

TABLE 5.1
Convergence study: Redistance of 1d parabolic profile

Δx	L1 error no constraint	L1 error constraint	Order
1/21	7.6E-4	4.1E-4	N/A
1/41	2.0E-4	1.2E-4	1.8
1/81	5.2E-5	3.2E-5	1.9

TABLE 5.2
Convergence study: Redistance of 1d parabolic profile; points are slightly offset from those in figure 5.1 in order to check for grid effects.

β is effectively our "bubble radius". We solve the following equation:

$$\phi_t + \phi_x = 0 \text{ for } 0 \le t \le 1$$

We use the algorithm from [13] for discretization of the advection step:

$$\begin{aligned} \bar{\phi}_i^{n+1} - \phi_i^n &= -(k/h)(\phi_{i+1/2}^n - \phi_{i-1/2}^n) \equiv (k)L^{advect}(\phi^n) \\ \phi_i^{n+1} - \phi_i^n &= (k/2)(L^{advect}(\phi^n) + L^{advect}(\bar{\phi}^n)). \end{aligned} \tag{5.2}$$

$\phi_{i+1/2}$ is computed using third order ENO as described in [13] and the CFL number k/h is 1/4.

After each step of equation (5.2), we perform a redistance operation for $0 \le \bar{t} \le \alpha\Delta x$ where $2\alpha\Delta x$ ($\alpha = 1$ for our 1d tests) is the total thickness of the interface:

$$\begin{aligned} \phi^{0,n+1} &= \phi^{n+1} \\ \phi_{\bar{t}} &= \text{sign}(\phi^{0,n+1})(1 - |\phi_x|) + \lambda f(\phi) \\ \Delta\bar{t} &= \Delta x/2 \\ \phi^{n+1} &= \phi^{2\alpha,n+1} \end{aligned} \tag{5.3}$$

The second order discretization of (5.3) is described in appendix B. The discretization of the constraint is described in section 4.

Table 5.3 displays the errors for $\beta = 1/4, 1/8, 1/12$. We also show a diagram comparing the computed solution to the expected value for the case of $\beta = 1/8$ and $\Delta x = 1/21$ (see fig. 9.4). For $\beta = 1/4$, the results with and without the constraint are similar. Since the center of the "bubble" is far from the zero level set, the re-distance scheme without the constraint does ok. For $\beta = 1/8$ and $\beta = 1/12$, the constraint helps improve the results as compared to the non-constraint case ($\Delta x = 1/21$ for $\beta = 1/8$ and $\Delta x \ge 1/41$ for $\beta = 1/12$). The constraint appears to "correct" errors in the redistance algorithm when one is near the center of a "bubble" (where the flow can be underresolved). This is an important test because we would like good behavior of the re-distance scheme when the zero level set is near singularities of the level set function.

6. Numerical Examples in two dimensions. In this section we will show how the constraint added to the redistance scheme substantially helps the accuracy. Error will be measured in terms of

Δx	β	L1 no-redistance	L1 no constraint	L1 constraint
1/21	1/4	9.7E-3	3.0E-4	3.0E-4
1/41	1/4	3.5E-3	1.1E-4	1.1E-4
1/81	1/4	1.2E-3	2.9E-5	2.9E-5
1/21	1/8	9.7E-3	7.0E-3	4.6E-3
1/41	1/8	3.5E-3	1.1E-4	1.1E-4
1/81	1/8	1.2E-3	2.9E-5	2.9E-5
1/21	1/12	9.7E-3	2.7E-2	2.3E-2
1/41	1/12	3.5E-3	5.2E-4	3.3E-4
1/81	1/12	1.2E-3	2.9E-5	2.9E-5

TABLE 5.3
Convergence study: Advection of "triangle" function $\beta = 1/4, 1/8, 1/12$

how much the computed interface differs from the expected interface:

$$\int_{\Omega} \frac{1}{L} |H(\phi_{expect}) - H(\phi_{compute})| dxdy. \tag{6.1}$$

L is the perimeter size of the expected interface. The error is computed by:

1. partitioning the domain into many tiny pieces (e.g. 1000×1000)
2. interpolating the values of ϕ_{expect} and $\phi_{compute}$, onto the newly created pieces.
3. numerically integrating equation (6.1), where $H(x) \equiv 1$ if $x < 0$ and $H(x) \equiv 0$ otherwise.

We will also be comparing the "average mass error" for solutions using the redistance scheme with the constraint and without the constraint. The average mass error is defined as:

$$M_{error} = \int_{t=0}^{t_f} \frac{|M(t) - M(0)|}{t_f} dt \tag{6.2}$$

where

$$M(t) = \int_{\Omega} |H(\phi(x, y, t))| dxdy.$$

We shall discretize the domain of computation using a staggered grid. We assume the dimensions of our domain are $L_1 \times L_2$. If n represents the number of cells in the y-direction and m represents the number of cells in the x-direction, then we have:

$$\Delta x = \frac{L_1}{m} \qquad \Delta y = \frac{L_2}{n}$$
$$x_{i,j} = \Delta x(i + \frac{1}{2}) \qquad i = 0 \ldots m-1$$
$$y_{i,j} = \Delta y(j + \frac{1}{2}) \qquad j = 0 \ldots n-1$$

6.1. two-dimensional test . A previous problem with the redistance scheme was the fact that the more the iterations, the more the zero level set would stray from the expected position. This would pose a problem for applications involving incompressible flow with stiff surface tension effects. The time step of the advection routine would be quite small which means the redistance scheme would be called often even if the interface has not advected very far. Thus, the errors caused by the redistance scheme would override the errors derived from the advection scheme.

steps	Δx	L1 w/fix	L1 w/o fix	fast least squares reconstruction
8	1/4	5.6E-4	1.2E-3	2.0E-3
40	1/4	5.7E-4	5.1E-3	2.0E-3
16	1/8	1.8E-4	2.8E-4	5.2E-4
80	1/8	1.8E-4	9.4E-4	5.2E-4
400	1/8	1.8E-4	20.5E-4	5.2E-4
32	1/16	5.3E-5	7.3E-5	1.4E-4
160	1/16	5.3E-5	2.4E-4	1.4E-4
64	1/32	1.5E-5	1.8E-5	3.6E-5
320	1/32	1.5E-5	4.9E-5	3.6E-5
64	1/64	4.3E-6	5.4E-6	9.0E-6
320	1/64	4.3E-6	1.5E-5	9.0E-6

TABLE 6.1
error for repeated redistance of a circle

For the following test we shall use second order ENO and second order Runge Kutta for discretizing the redistance equations. We show how the repeated application of the redistance scheme does not cause the interface to stray when the constraint is active. In our tests, the redistance time step is $\Delta x/2$. The number of steps in our experiment varied in multiples of $M = 2/\Delta x$. After every M steps, we replace ϕ_0 with the current value ϕ_M. By doing this, we would verify that the error does not grow as the number of iterations grows or grow as the number of times that the redistance operation is called grows. We test our scheme on a 4×4 domain containing a circle of radius one:

$$\phi_0(x, y) = \sqrt{x^2 + y^2} - 1.$$

We refer the reader to table 6.1. It can be seen that the errors with the constraint are much smaller than the other errors, and more importantly, the error does not grow at all when the number of iterations is increased. We also note that the error is smaller than the error incurred by simply reconstructing the interface using a Volume of Fluid reconstruction (see [10]).

6.2. two dimensional advection. For moving an interface with specified velocity $\vec{u}$, we solve

$$\phi_t + \vec{u} \cdot \nabla\phi = 0. \tag{6.3}$$

This equation moves the zero level set of ϕ in accordance with the input velocity (see [9, 8, 14]). Equation (6.3) is discretized using third order Runge-Kutta for the temporal discretization and third order ENO for the spatial discretization:

1. set $\phi^{(0)} = \phi_n$, repeat steps (2) and (3) for $i = 1 \ldots r$ ($r = 3$, r is the order of the method):
2. Define $L^{advect}(\phi^{(i-1)})$ as

$$L^{advect}(\phi) \equiv -u_{ij}(\phi_{i+1/2j} - \phi_{i-1/2j})/\Delta x - v_{ij}(\phi_{ij+1/2} - \phi_{ij-1/2})/\Delta y.$$

   The fluxes $\phi_{i+1/2j}$ and $\phi_{ij+1/2}$ are computed using third order ENO as described in [13].
3.

$$\phi^{(i)} = \sum_{k=0}^{i-1} \alpha_{ik}\phi^{(k)} + \beta_{ik}\Delta t L^{advect}(\phi^{(k)})$$

   The coefficients α and β for third order Runge Kutta, are defined in appendix B.

Δx	error w/o constraint	error w/constraint	order w/constraint
1/4	1.44E-2	1.54E-3	N/A
1/8	7.71E-4	4.57E-4	1.8
1/16	1.29E-4	5.77E-5	3.0
1/32	2.15E-5	6.77E-6	3.1
1/64	5.44E-6	8.15E-7	3.1

TABLE 6.2
Convergence study: Diagonal translation of circle

4. $\phi^{n+1} = \phi^{(r)}$

After each time step, we apply the redistance operation for $0 \le \bar{t} \le \alpha\Delta x$ where $2\alpha\Delta x$ is the total thickness of the interface:

$$\begin{aligned}
\phi^{0,n+1} &= \phi^{n+1} \\
\phi_{\bar{t}} &= \text{sign}(\phi^{0,n+1})(1 - |\nabla\phi|) + \lambda f(\phi) \\
\Delta\bar{t} &= \Delta x/2 \\
\phi^{n+1} &= \phi^{2\alpha,n+1}
\end{aligned}$$

We shall also discretize the redistance equations using third order ENO and third order Runge-Kutta as described in Appendix B. The constraint is discretized as described in section 4.

6.2.1. Steady Advection . For steady advection, we specify a velocity field that doesn't change in time. We do convergence tests for the diagonal translation of a circle and also for Zalesak's (see [16]) problem. Zalesak's problem involves the rotation of a notched disc, which is a good test of how well we advect interfaces in the presence of high curvature.

For the translating circle, we compute the solution in a 4×4 periodic box. We initialize our domain with the following:

$$\begin{aligned}
u(x,y) = v(x,y) &= 1 \\
\phi(x,y) &= \sqrt{x^2+y^2} - 1.
\end{aligned}$$

The interfacial thickness parameter is $\alpha = 2$. We run the above problem up to $t = 4$ and then measure the error (error calculated according to (6.1)). In table 6.2 we measured third order accuracy when coupling the advection of the levelset function with the redistance step that uses the constraint:

We now test our advection scheme for computing "Zalesak's problem" (see [16]). The domain size is 100×100 and it contains a slotted circle centered at (50,75) with slot width 15. We initialize $\vec{u}$ and ϕ as follows:

$$\begin{aligned}
u_0 &= (\pi/314)(50 - y) \\
v_0 &= (\pi/314)(x - 50) \\
\phi_0 &= \text{signed distance from object}
\end{aligned}$$

We compute up to $t = 628$ (one full revolution) on a 100×100 grid; the same as that used in [16]. We then refine the grid in order to measure accuracy. In all cases, the time step Δt is equal to Δx. In table 6.3, we display the error when advecting the notched disk. The error is measured for the case when the redistance constraint is not used and also for the case when it is used. For the case when the constraint is used, we get an order of accuracy of ranging from 1.3 to 2.6 which is

Δx	error w/o constraint	error w/constraint	order w/constraint
1	1.22	2.62E-1	N/A
1/2	2.76E-1	4.18E-2	2.6
1/3	1.31E-1	2.43E-2	1.3

TABLE 6.3
Convergence study: Rotation of cutout circle (Zalesak's problem)

good considering the sharp corners in the initial data. We overlayed the coarse grid results with the expected solution in figure 9.5. The maximum mass fluctuation ranged from 1.3 percent on the coarse grid to 0.1 percent on the finest grid. In figure 9.8, we plot the area of the notched disc over time. The effectiveness of our constraint is demonstrated in figures 9.6 and 9.7. In figure 9.6, we overlay the computed results on the 200x200 grid with the expected results. The redistance constraint was used in this case. We see very good agreement. In figure 9.7, we display the corresponding 200x200 results when the redistance constraint is turned off. In this case, the corners become rounded.

6.2.2. Unsteady Advection: incompressible two-phase flow. The redistance iteration is necessary for computing incompressible two-phase flow using the level set formulation (see [14]). The purpose of the redistance scheme is to allow us to provide the interface with a thickness fixed in time such that flows with large density ratios and surface tension driven forces can be robustly computed.

We repeat some of the examples from [14] in order to show the improved accuracy and efficiency due to our redistance constraint. In [14] and in our "non-constraint" tests below the redistance time step is $\Delta x/10$. The redistance timestep when using the constraint is $\Delta x/2$. The constraint allows us to iterate with a larger timestep and with less error. In figure 9.7 it is clear that without the constraint, but yet leaving the redistance time step at $\Delta x/2$, fine features of the flow (corners) will be smeared. The non-constraint version we will be using is different in a few ways from [14]:

- we are using a third order accurate scheme in time (Runge-Kutta) and space (ENO) for the redistance iteration.
- When computing the error for determining the stopping criteria, we scale the error at points that have a large curvature.
- If the redistance time reaches a value equal to the thickness of the interface, then the iteration is stopped.

The improvements above will cause results using the non-constraint version to slightly outperform results from [14]. We will see below that results using the constraint will be considerably better than the result where the constraint was not used.

Briefly we describe how the velocity field is computed for the level set implementation of incompressible two-fluid flow:

We solve the following equations for $\vec{u}$ on the interior of the domain:

$$\begin{aligned} \vec{u}_t &= -\vec{u}\cdot\nabla\vec{u} - \nabla p/\rho(\phi) + \vec{g} + \\ &\quad (1/Re)\nabla\cdot(2\mu D)/\rho - (1/Bd)\kappa(\phi)\nabla H(\phi) \\ \nabla\cdot\vec{u} &= 0 \end{aligned}$$

We have free-slip solid wall boundary conditions:

$$\vec{u}\cdot\vec{n} = 0$$

Δx	mass w/o fix	error w/o fix	mass w/fix	error w/fix	order w/fix
6/50	3.27	N/A	3.12	N/A	N/A
6/100	3.13	1.15E-1	3.11	8.70E-2	N/A
6/200	3.13	3.87E-2	3.13	2.80E-2	1.6

TABLE 6.4
Convergence study: Rise of 2d gas bubble in liquid, t=4.4, $\rho_1/\rho_2 = 1000:1$, $Re = 100$, $Bd = 200$

Δx	mass w/o fix	error w/o fix	mass w/fix	error w/fix	order w/fix
6/50	2.94	N/A	3.08	N/A	N/A
6/100	3.05	1.32E-1	3.11	1.25E-1	N/A
6/200	3.06	5.90E-2	3.13	5.90E-2	1.1

TABLE 6.5
Convergence study: Rise of 2d gas bubble in liquid, t=4.8, $\rho_1/\rho_2 = 1000:1$, $Re = 100$, $Bd = 200$

$\vec{u} = (u, v)$ is the fluid velocity, $\rho = \rho(\phi)$ is the fluid density, $\mu = \mu(\phi)$ is the fluid viscosity, D is the viscous stress tensor, and $\vec{g}$ represents the gravitational force of magnitude 1. The surface tension term is considered to be a force concentrated on the interface. κ is the curvature of the front, and H is the Heaviside function. The equation is in non dimensional form where Re, Bd, ρ_1/ρ_2, and μ_1/μ_2 need to be specified.

We will do two examples. The first problems will involve the collision of two drops (see figure 9.9) and the second problem will involve the rise of a gas bubble in liquid (see figure 9.11). In both examples, we will assume the flow is 2d and symmetric about the line $x = 0$.

For our first problem, we have two drops collide (figure 9.9) and then undergo surface tension driven oscillations. The grid spacing is 22×44 with $\Delta x = \Delta y = 7/44$. This test is identical to a test done in [14] (figure 23). The density ratio is $\rho_1/\rho_2 = 14/1$ and the viscosity ratio is $\mu_1/\mu_2 = 333/1$. The Reynolds number is 20 and the Bond number is 2. In figure 9.10, we compare the mass for the version without the constraint to the mass for the version with the constraint. The average mass error for the version with the constraint is 0.03 (.5% error) and the error without the constraint is 0.08 (1.3% error).

For our second problem, we compute the rise of a gas bubble in liquid (see figure 9.11). We measure the relative error (6.1) on successively finer meshes in order to determine the convergence rate. As for the drop merge problem, we perform the computation first with the redistance constraint active and then with the redistance constraint turned off. The density ratio is $\rho_1/\rho_2 = 1000/1$ and the viscosity ratio is $\mu_1/\mu_2 = 100/1$. The Reynolds number is 100 and the Bond number is 200. In table 6.4 we display the error between successively finer grids at time $t = 4.4$. The order of accuracy here is 1.6. In table 6.5, we display the error at $t = 4.8$; the time after the bubble has begun to break up. The order accuracy here is 1.1. In figure 9.12, we compare the mass for the version without the constraint to the mass for the version with the constraint. As before, we see significant improvement in the results when using the redistance algorithm along with the new constraint. This is clearly seen in figure 9.13 where the version with the constraint has qualitatively the same results as without the constraint except that the fine features during pinch off remain for the case when using the constraint. The average mass error for the constraint version and the non-constraint version are 0.01 (0.3% error) and 0.03 (1%) respectively when comparing results on the finest grid $\Delta x = \Delta y = 6/200$.

7. Conclusion. We have shown that the addition of our "constraint" improves the accuracy of the redistance iteration. This in turn helps when applied to the basic advection of a level set function and when applied to interfacial incompressible flow. We would like to point out that, with the onset of our "constraint", mass conservation errors incurred during advection of a level set function when used in conjunction with the redistance iteration are now primarily due to the difference methods used in solving the equation

$$\phi_t + \vec{u} \cdot \nabla \phi = 0. \tag{7.1}$$

We demonstrated that higher order methods plus the use of the redistance iteration in conjunction with equation (7.1) will improve the accuracy of a (7.1), but will not conserve 100% mass. An open problem would be to find more effective schemes for handling (7.1). The advantage of our level set scheme still remains that we never have to explicitly reconstruct the interface and that quantities such as the gradient of the level set function and the curvature can be more accurately computed from a smooth level set function.

8. Appendix A; first-time only redistance step . We propose the following simple modification to the scheme presented in section 4 for redistancing a function ϕ_0 which is not close to a distance function. This can be the case when the initial interface for a fluid problem is much too complicated to come up with an initial guess for the level set function. This modification will only be invoked once for the duration of the problem. As shown in section 4.2, for ensuing time steps the level set function will stay within $O(h)$ of a distance function in-between redistancing operations. We start off with a ϕ_0 which is 1 in the first fluid and -1 in the second fluid.

1. We compute $\mid \nabla \phi_0 \mid$ exactly the same way as for the term appearing in section 4.1 (see Appendix B for higher order discretizations).
2. At points where $\mid \nabla \phi_0 \mid > 0$, we let $\phi_0 = \phi_0 / \mid \nabla \phi_0 \mid$.
3. We perform the iteration as in section 4 except with the constraint disabled. We iterate up to $t_n = L$ where L is the length of the domain.

8.1. Numerical examples of first-time only redistance step .

8.1.1. 1d examples. We start off with the following initial data,

$$\phi_0 = \begin{cases} -1 & \text{if } |x - 1/2| < 1/4 \\ 1 & \text{otherwise} \end{cases}$$

In this test, we convert the above initial data into a distance function using the scheme described in section 8. We display the results in figure 9.2. We had $\Delta x = 1/21$ and $\Delta t = \Delta x/2$. The number of iterations was 40.

8.1.2. 2d examples. In this section we display the effectiveness of the first-time only redistance step when applied to a circle. We test our scheme on a 7×7 domain containing a circle of radius one:

$$\phi_0(x, y) = \begin{cases} -1 & \text{if } \sqrt{x^2 + y^2} < 1 \\ 1 & \text{otherwise} \end{cases}$$

For this test, $\Delta x = 1/10$ and $\Delta t = \Delta x/2$. From equation (4.4), we claim that our redistance scheme "updates" expanding outward/inward from the zero level at a rate of one. This is apparent if one looks at figure 9.1 where one sees after time $t = 1$, the first ten level sets are updated on either side of the interface (level sets spaced Δx apart). After time $t = 2$, the first 20 level set are updated. Finally at time $t = 7$, ϕ represents a distance function over the whole domain. The ending area of the unit circle is 3.05 which is a %3 error that will only be incurred once during an advection problem. As shown in table 8.1, ensuing redistance operations using the constraint do not add to the error.

steps	Δx	L1 w/fix
0	1/10	3.1E-2
20	1/10	3.1E-2
100	1/10	3.1E-2

TABLE 8.1

error for repeated redistance of a circle where the initial level set function is the result of redistancing a color function. The initial error at steps = 0 is the error of our initial level set function as compared to the level set function $\phi(x,y) = (x^2+y^2)^{1/2} - 1$.

9. Appendix B; Details of high order accurate methods for discretizing the redistance step . In section 4.1, the first order discretization of the redistance scheme was presented. In our computations, we will be using third order methods in space and time. The first order time-discretization presented in steps (2) through (5) in section 4.1 will be replaced by a third order Runge-Kutta scheme [13]. The first order discretization of the spatial derivatives in equations (4.5) and (4.6) will be replaced by third order ENO (upwind) schemes [5, 13]. The resulting modifications to steps (2) through (5) in section 4.1 are:

2. set $\phi^{(0)} = \phi_n$, repeat steps (3) and (4) for $i = 1 \ldots r$ ($r = 3$, r is the order of the method):

3. Compute $L(\phi_0, \phi^{(i-1)})$ which will be a third order approximation to $\text{sign}(\phi_0)(1-|\nabla\phi^{(i-1)}|)$. The spatial derivatives of $|\nabla\phi^{(i-1)}|$ are computed using third order ENO as described below.

4.

$$\phi^{(i)} = \sum_{k=0}^{i-1} \alpha_{ik}\phi^{(k)} + \beta_{ik}\Delta t L(\phi_0, \phi^{(k)})$$

For 3rd order Runge-Kutta:
$\alpha_{10} = 1$ $\alpha_{20} = 3/4$ $\alpha_{21} = 1/4$ $\alpha_{30} = 1/3$ $\alpha_{31} = 0$ $\alpha_{32} = 2/3$
$\beta_{10} = 1$ $\beta_{20} = 0$ $\beta_{21} = 1/4$ $\beta_{30} = 0$ $\beta_{31} = 0$ $\beta_{32} = 2/3$

5. $\tilde{\phi}_{n+1} = \phi^{(r)}$

The high order discretization of the spatial derivatives are based on upwinded ENO schemes [5, 13]. The first order discretization in equations (4.5) and (4.6) are replaced as follows:

For $\frac{\partial\phi}{\partial x}$ we first compute the divided difference table:

$$\phi[x_i, x_i] = \phi_{ij}$$
$$\phi[x_{i-k}, x_{i+l}] = (\phi[x_{i-k+1}, x_{i+l}] - \phi[x_{i-k}, x_{i+l-1}])/(x_{i+l} - x_{i-k})$$

For a method of spatial accuracy r we have:

1. let $Q^0(x) = \phi[x_i, x_i]$
2. Do the following steps for $k_{\min}^{(1)}$ equal to $i-1$ and i:
   (a)
   $$Q^1(x) = \phi[x_{k_{\min}^{(1)}}, x_{k_{\min}^{(1)}+1}](x - x_i)$$
   (b) Repeat for $l = 1 \ldots r-1$:
      i.
      $$a^l = \phi[x_{k_{\min}^{(l)}-1}, x_{k_{\min}^{(l)}+l}]$$
      $$b^l = \phi[x_{k_{\min}^{(l)}}, x_{k_{\min}^{(l)}+l+1}]$$

ii.

$$c^l = \begin{cases} a^l & \text{if } |a^l| \le |b^l| \\ b^l & \text{otherwise} \end{cases}$$

$$k_{\min}^{(l+1)} = \begin{cases} k_{\min}^{(l)} - 1 & \text{if } |a^l| \le |b^l| \\ k_{\min}^{(l)} & \text{otherwise} \end{cases}$$

iii.

$$Q^{l+1}(x) = Q^l(x) + c^l \prod_{k=k_{\min}^{(l)}}^{k=k_{\min}^{(l)}+l} (x - x_k)$$

(c) if $k_{\min}^{(1)} = i - 1$, $D_x^- \phi = d(Q^r(x))/dx$ otherwise $D_x^+ \phi = d(Q^r(x))/dx$

3.

$$\frac{\partial \phi}{\partial x} \approx \begin{cases} D_x^+ \phi & \text{if } D_x^+ \phi \text{sign}(\phi_0) < 0 \text{ and } (D_x^- \phi + D_x^+ \phi)\text{sign}(\phi_0) < 0 \\ D_x^- \phi & \text{if } D_x^- \phi \text{sign}(\phi_0) > 0 \text{ and } (D_x^+ \phi + D_x^- \phi)\text{sign}(\phi_0) > 0 \\ 0 & \text{if } D_x^- \phi \text{sign}(\phi_0) < 0 \text{ and } D_x^+ \phi \text{sign}(\phi_0) > 0 \end{cases}$$

REFERENCES

[1] Adalsteinsson, D., Kimmel, R., Malladi, R., and Sethian, J.A., *Fast Marching Methods for Computing Solutions to Static Hamilton-Jacobi Equations*, report, Center for Pure and Applied Mathematics, PAM-667 (1996).

[2] Adalsteinsson, D., and Sethian, J., *A Fast Level Set Method for Propagating Interfaces*, J. Comp. Phys., 118, pp. 269-277 (1995).

[3] Chopp, D.L., *Computing Minimal Surfaces Via Level Set Curvature Flow*, J. Comp. Phys., 106, pp. 77-91 (1993).

[4] Danielsson, Per-Erik, *Euclidean Distance Mapping*, Computer Graphics and Image Processing, 14, pp. 227-248 (1980).

[5] Harten, A., J. Comp. Phys., 83, 148-184 (1989).

[6] Helmsen, J., Puckett, E.G., Colella, P. and Dorr, M., *Two new methods for simulating photolithography development in 3D*, SPIE Optical/Laser Microlithography IX, vol. 2726, March, 1996 in Santa Clara, CA.

[7] Kimmel, R. and Bruckstein, *Shape Offset via Level Sets*, CAD, vol. 25, number 3, pp. 154-162 (March 1993).

[8] Mulder, W., Osher, S., and Sethian, J.A., *Computing Interface Motion In Compressible Gas Dynamics*, J. Comp. Phys., 100, 209 (1992).

[9] Osher, S. and Sethian, J.A., *Fronts Propagating with Curvature-Dependent Speed: Algorithms Based on Hamilton-Jacobi Formulations*, J. Comp. Phys., 79,1, pp. 12-49, (1988).

[10] Pilliod, J. and Puckett, E.G., *Second Order Volume of Fluid Algorithms for Tracking Material Interfaces*, manuscript, to be submitted to J. Comput. Phys.

[11] Rouy, E. and Tourin, A., *A Viscosity Solutions Approach to Shape-From-Shading*, SIAM J. Numer. Anal., Vol. 29, No. 3, pp. 867-884, June 1992.

[12] Sethian, J.A., *A marching level set method for monotonically advancing fronts*, Proc. Nat. Acad. Sci., 93(4), 1996.

[13] Shu, C.W. and Osher,S., *Efficient Implementation of Essentially Non-Oscillatory Shock Capturing Schemes, II*, J. Comp. Phys., 83, pp. 32-78, (1989).

[14] Sussman, M., Smereka, P., & Osher, S.J., *A level set approach for computing solutions to incompressible two-phase flow*, J. Comp. Phys., 94, 146-159 (1994).

[15] Unverdi, S.O. and Tryggvason, G., *A Front-Tracking Method for Viscous, Incompressible, Multi-fluid Flows*, J. Comp. Phys., 100, pp. 25-37, (1992).

[16] Zalesak, S.T., J. Comp. Phys., 31, 335-362 (1979).

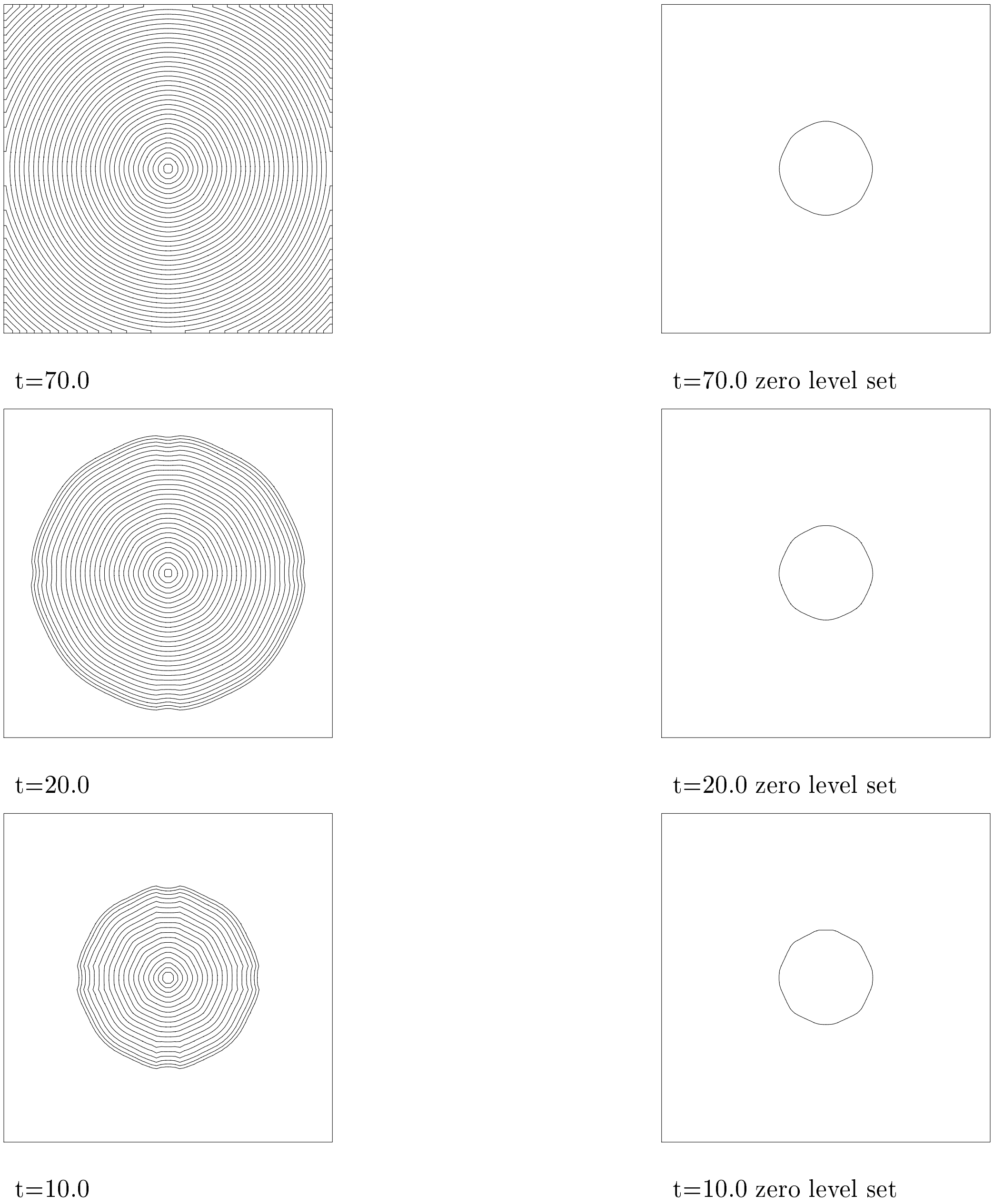

Fig. 9.1. *Result of redistancing an initially discontinuous 2d function $\phi_0(x, y)$ which is $+1$ outside of a unit circle and -1 inside. Contours are spaced $\Delta x = 1/10$ apart, pictures represent level set function after time $t = 10\Delta x, 20\Delta x, 70\Delta x$*

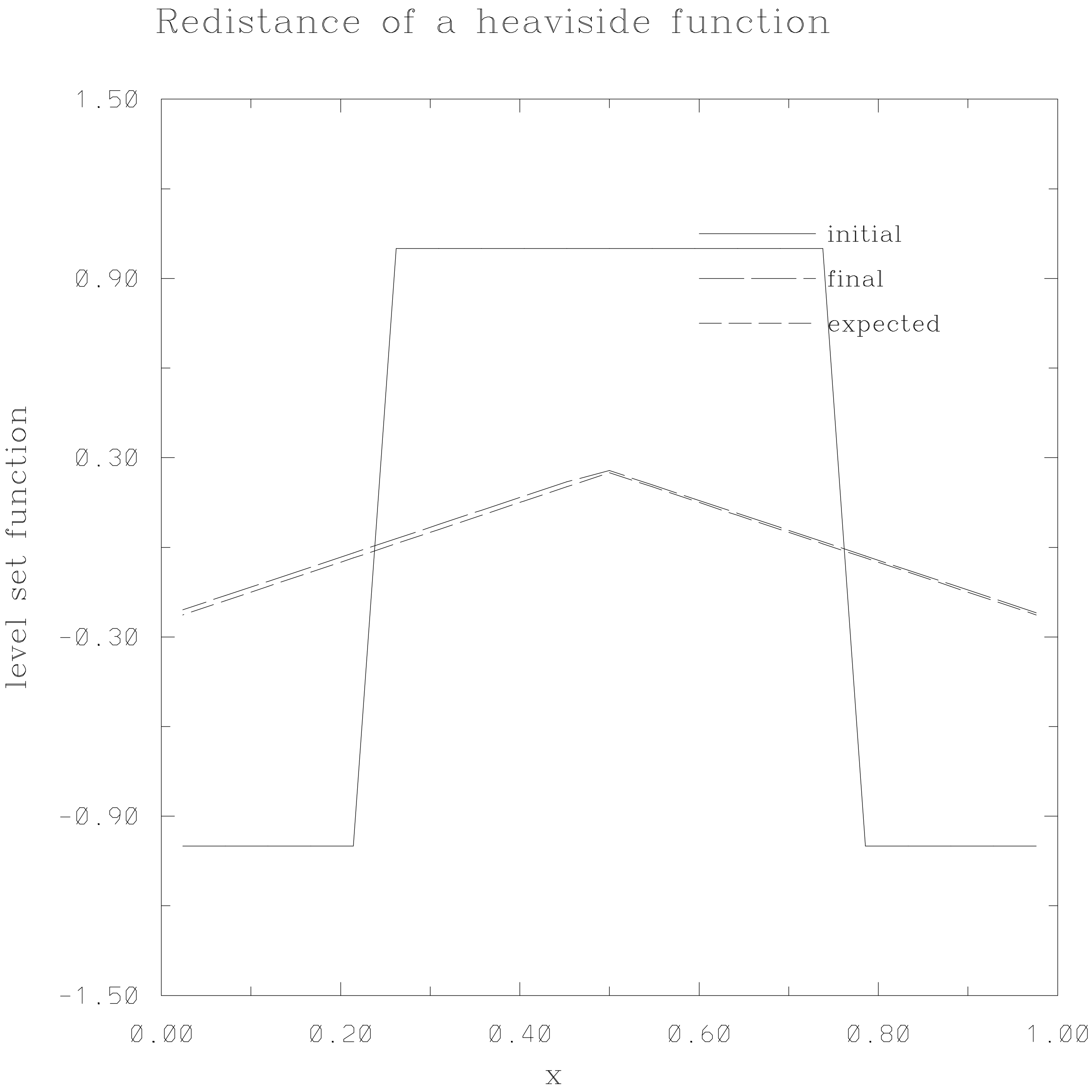

FIG. 9.2. *Result of redistancing an initially discontinuous 1d function $\phi_0(x) = \pm 1$ after time $t = 20\Delta x$, $\Delta x = 1/21, \Delta t = \Delta x/2$*

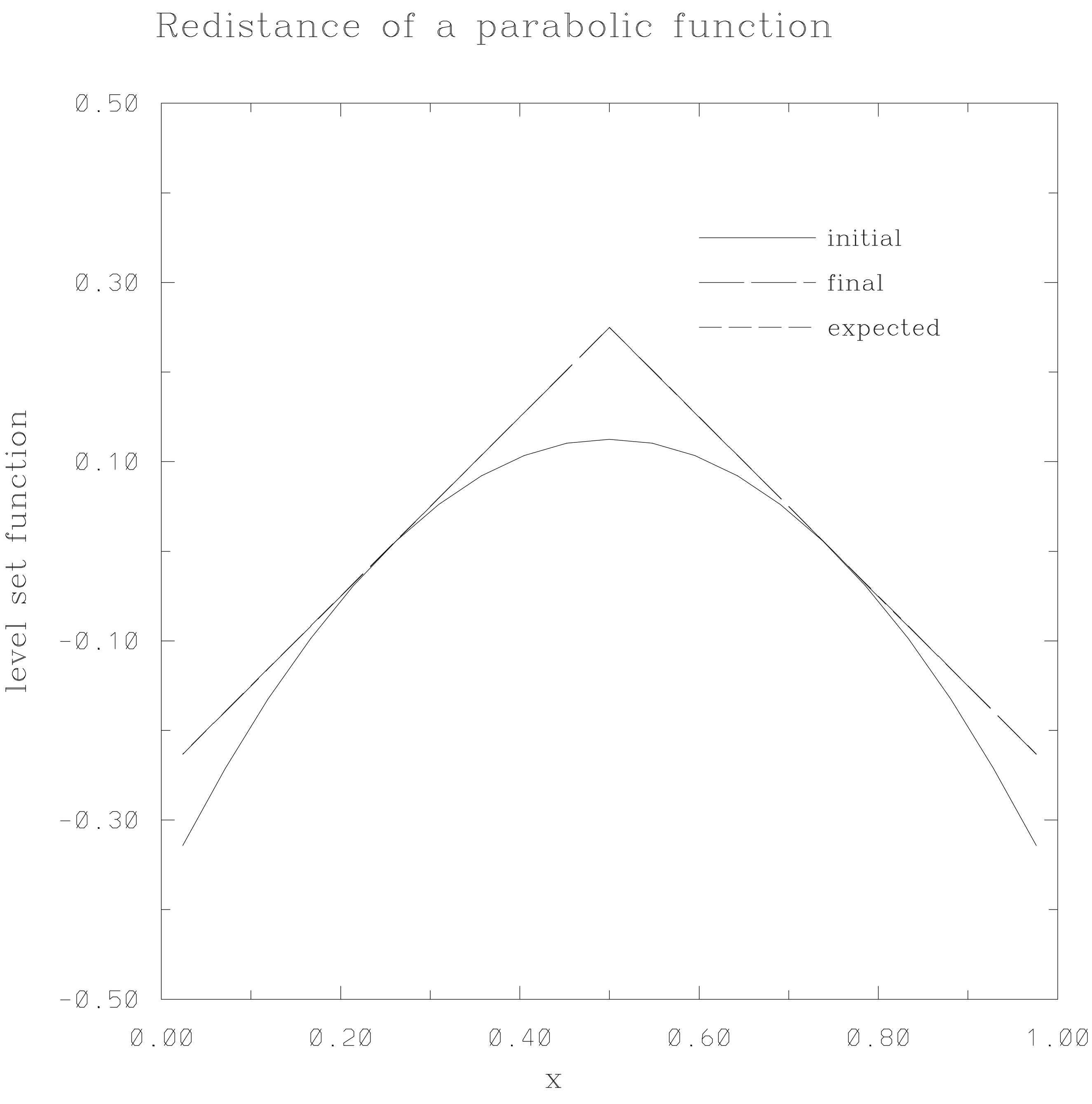

FIG. 9.3. *Result of redistancing an initially parabolic 1d function* $\phi_0(x) = -2(x-1/4)(x-3/4)$ *after time* $t=1$, $\Delta x = 1/21$, $\Delta t = \Delta x/2$

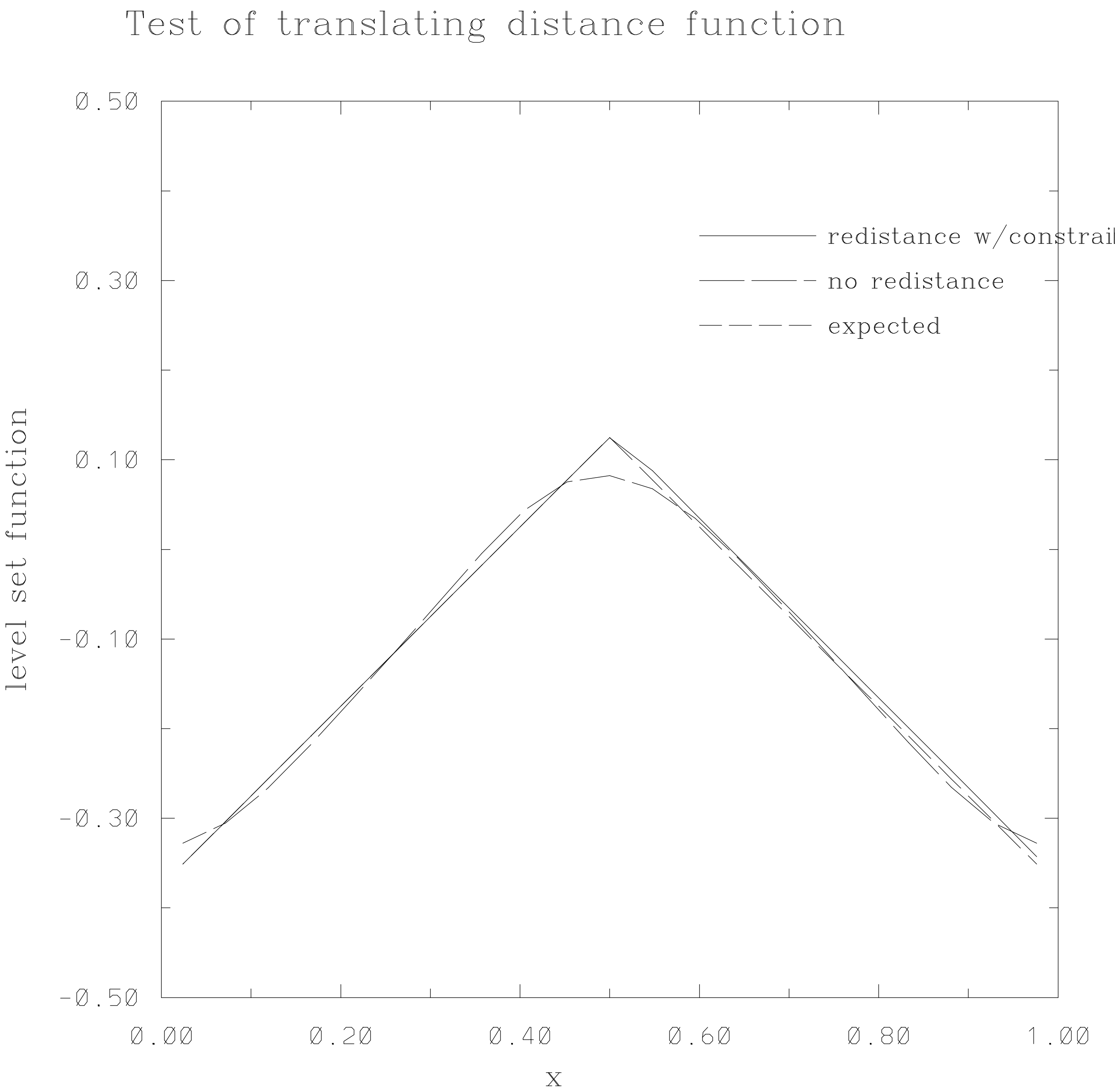

FIG. 9.4. *Results after $t = 1$ of a translating triangle function. The redistance scheme along with the constraint, helps maintain the initial profile even when it's close to the tip of the triangle. $\Delta x = 1/21$, $\Delta t = \Delta x/4$, $\bar{\Delta t} = \Delta x/2$, $\beta = 1/8$, the thickness of the interface is two points total ($2\Delta x$).*

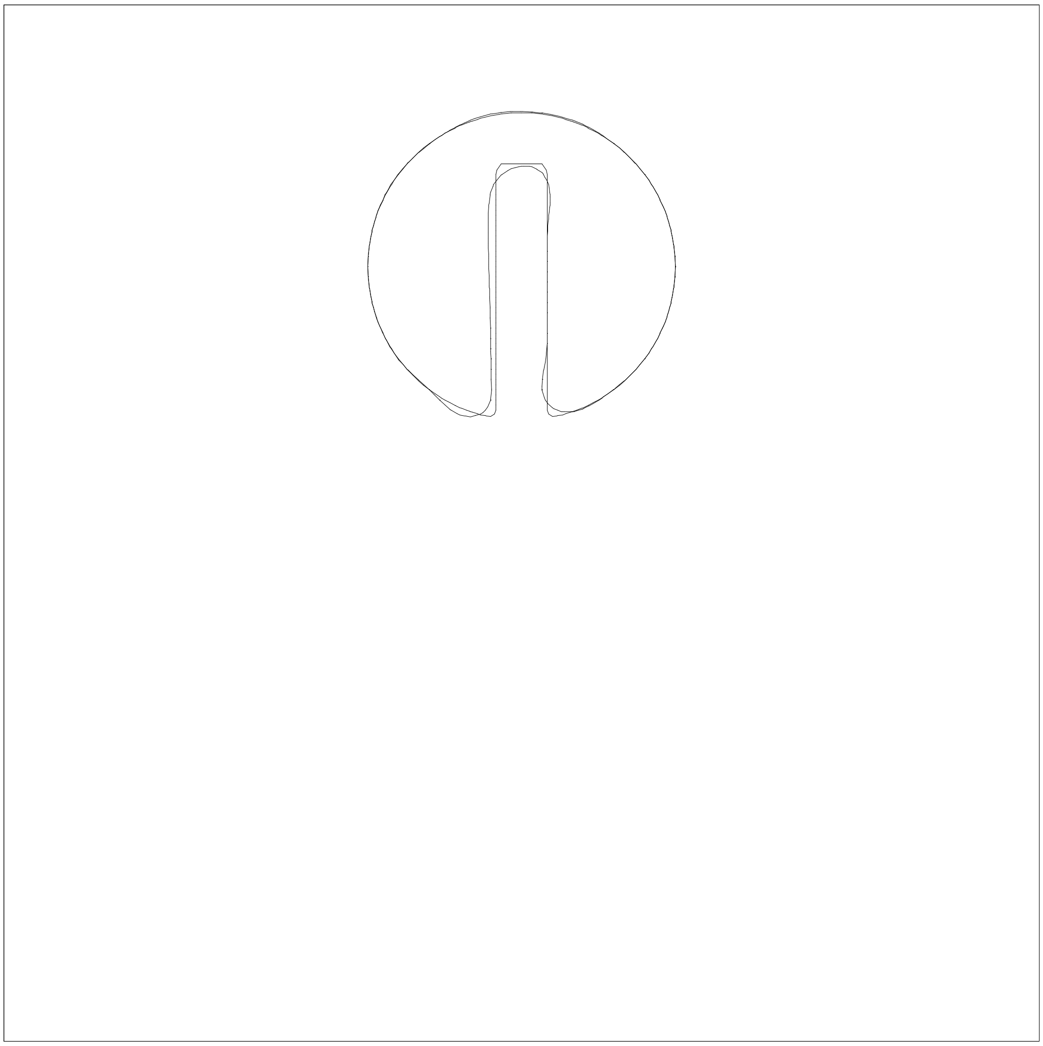

FIG. 9.5. *Zalesak's problem, Comparison of a notched disc that has been rotated one revolution about the center of the domain.* $\Delta x = \Delta t = 1$ *(100x100)*

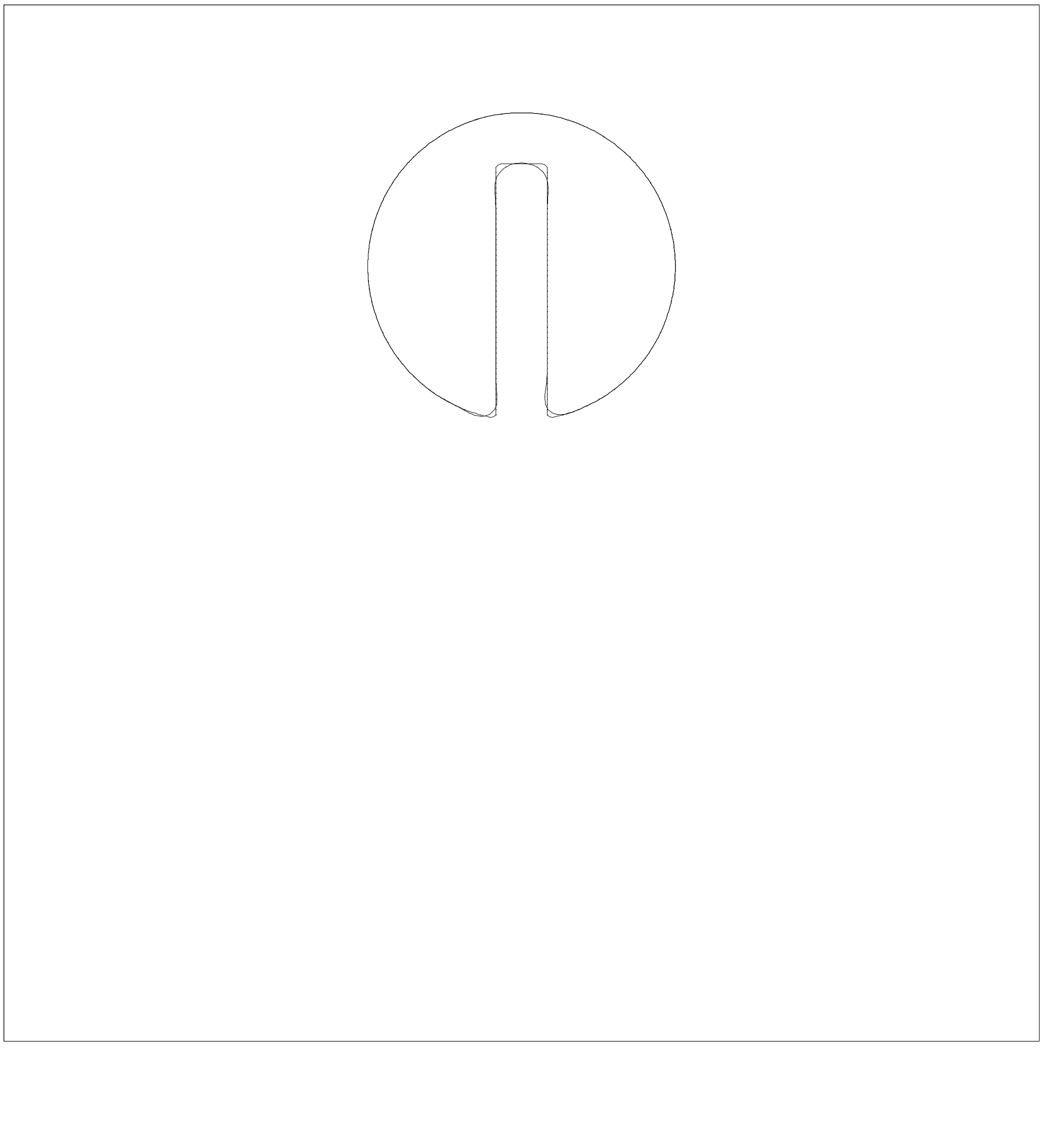

FIG. 9.6. *Zalesak's problem, Comparison of a notched disc that has been rotated one revolution about the center of the domain.* $\Delta x = \Delta t = 1/2$ *(200x200)*

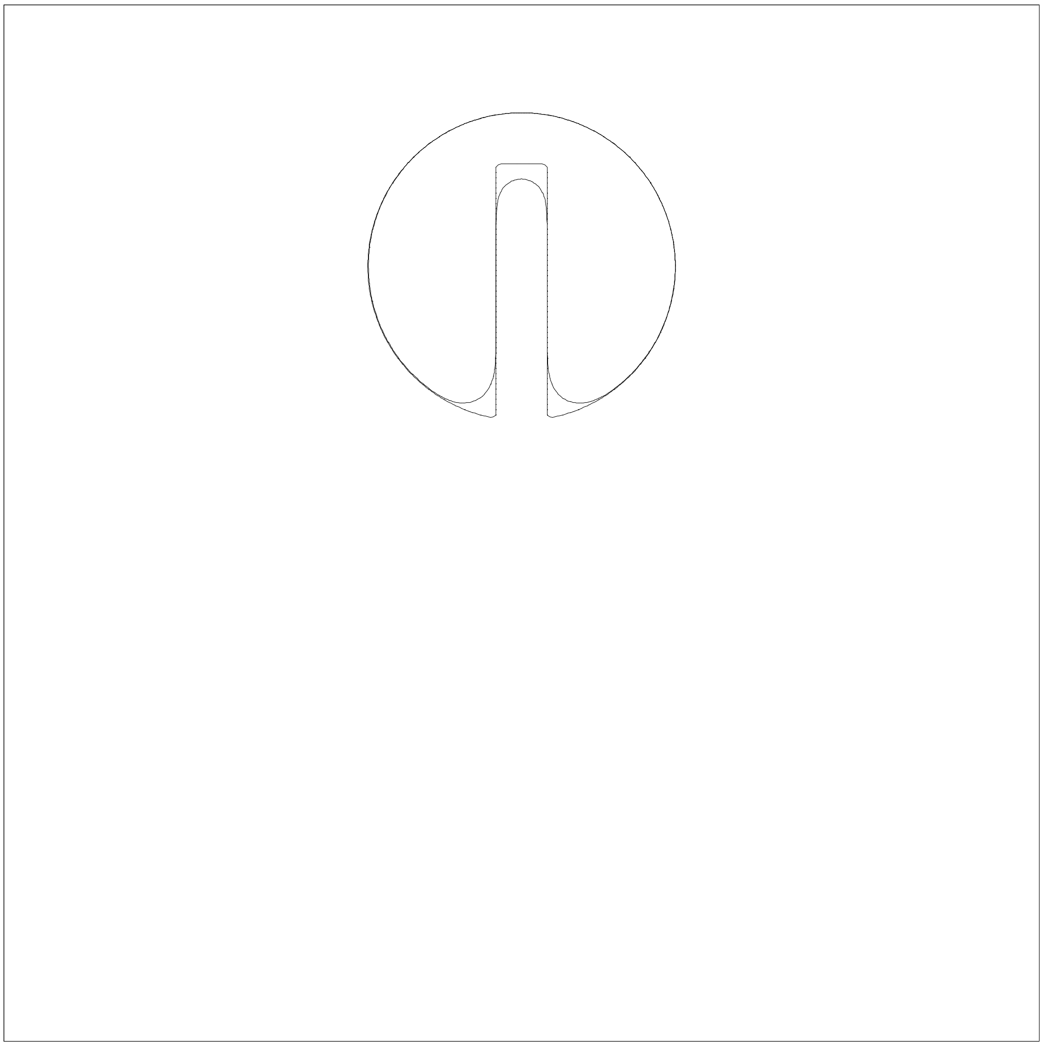

FIG. 9.7. *Zalesak's problem, Redistance constraint not implemented here, Comparison of a notched disc that has been rotated one revolution about the center of the domain.* $\Delta x = \Delta t = 1/2$ *(200x200)*

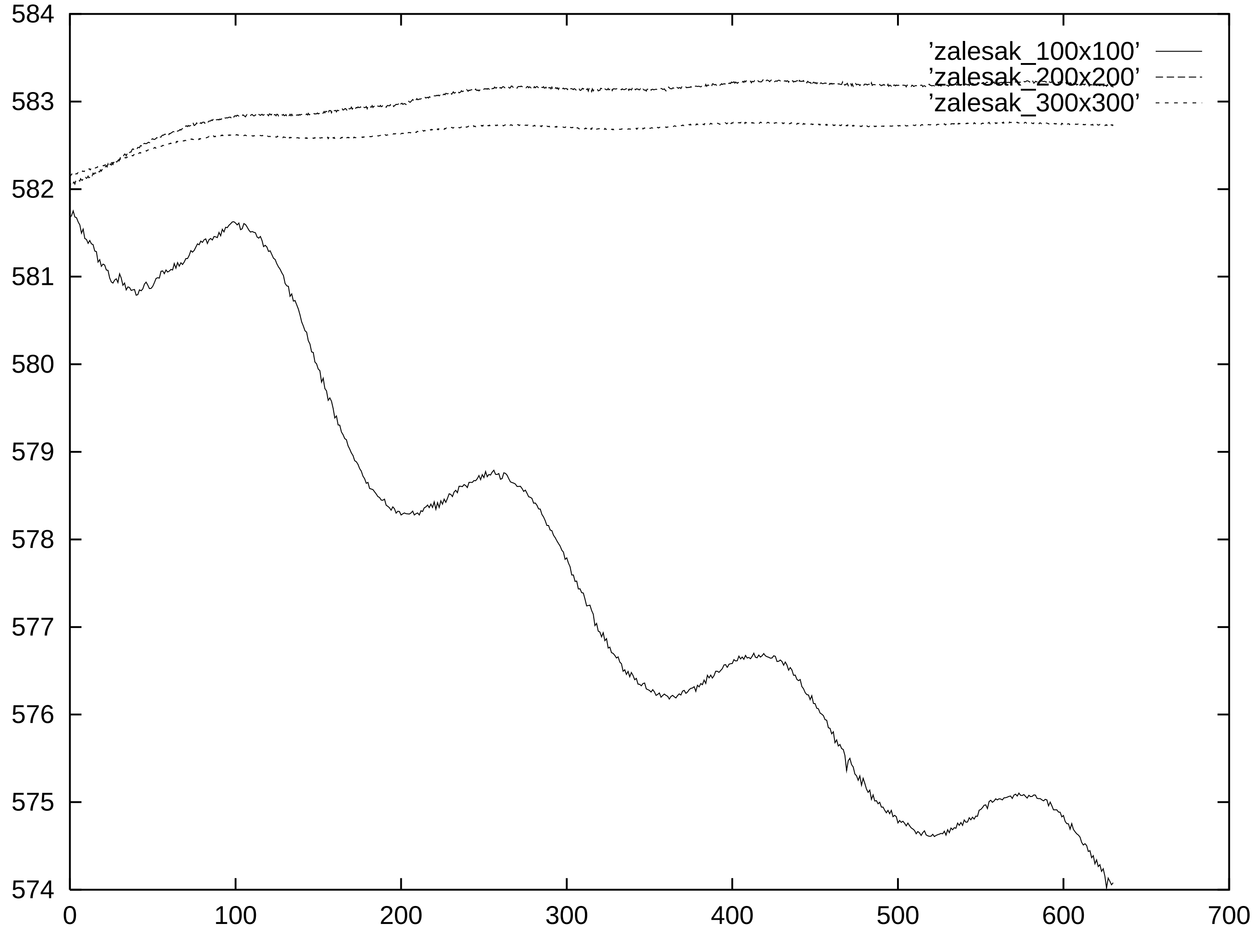

FIG. 9.8. *Zalesak's problem, Comparison of mass for different levels of resolution; x-axis is time and y-axis represents mass.*

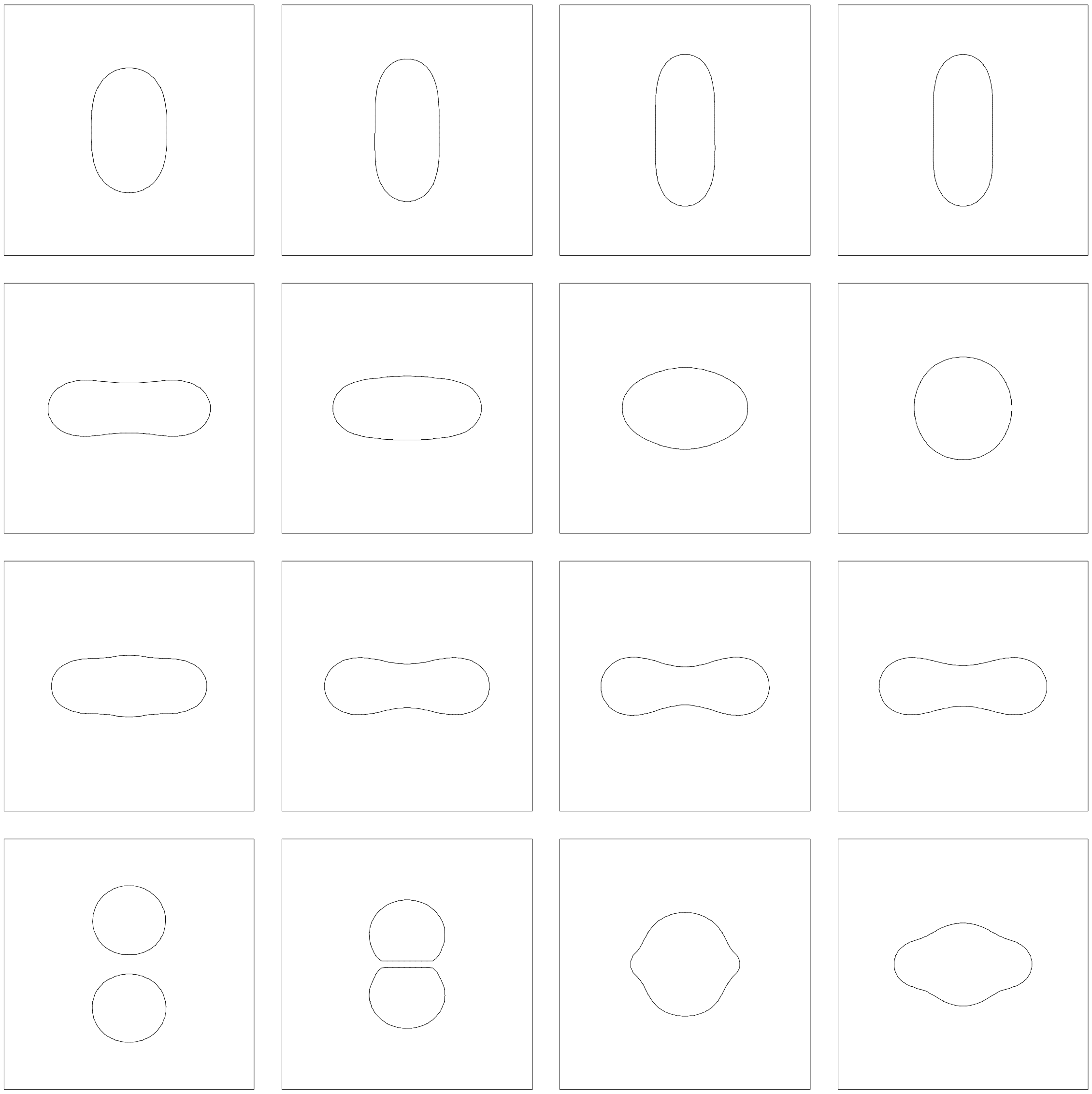

FIG. 9.9. *two-dimensional drop collision; redistance scheme with constraint was used. Re=20 Bd=2.0 density 14/1 grid 22x44 (symmetric boundary conditions). Time increases from left to right, bottom to top starting at t=2.0 and ending at t=9.5.*

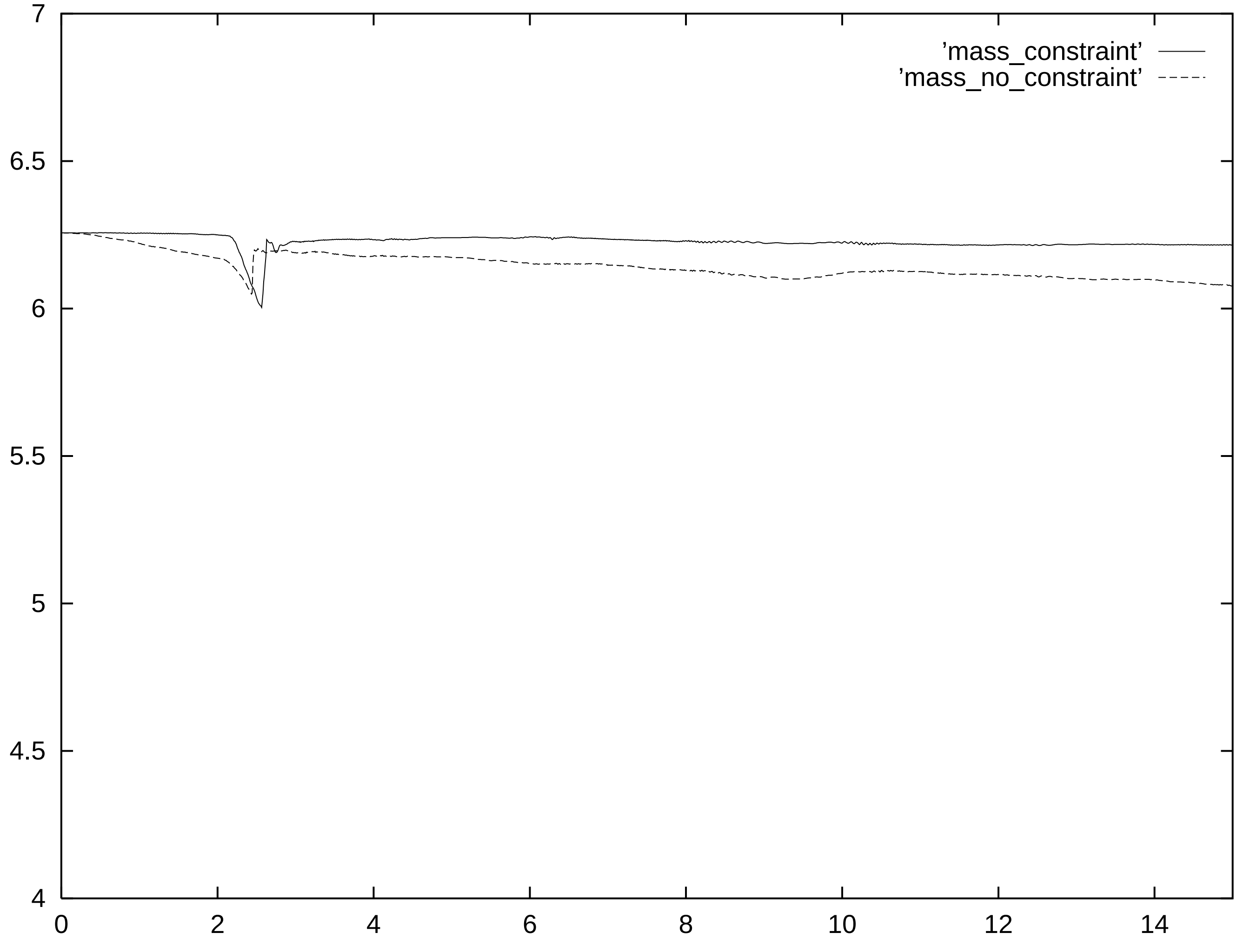

FIG. 9.10. *Mass conservation for drop collision problem. When the redistance scheme with the constraint was used, the average mass error was less than without the constraint; x-axis is time and y-axis represents mass.*

Fig. 9.11. *two-dimensional rising gas bubble; redistance scheme with constraint was used. Re=100 Bd=200 density 1000:1 grid 100x200,* $\Delta x = 6/200$ *(symmetric boundary conditions). Time increases from left to right, bottom to top starting at t=2.8 and ending at t=6.0.*

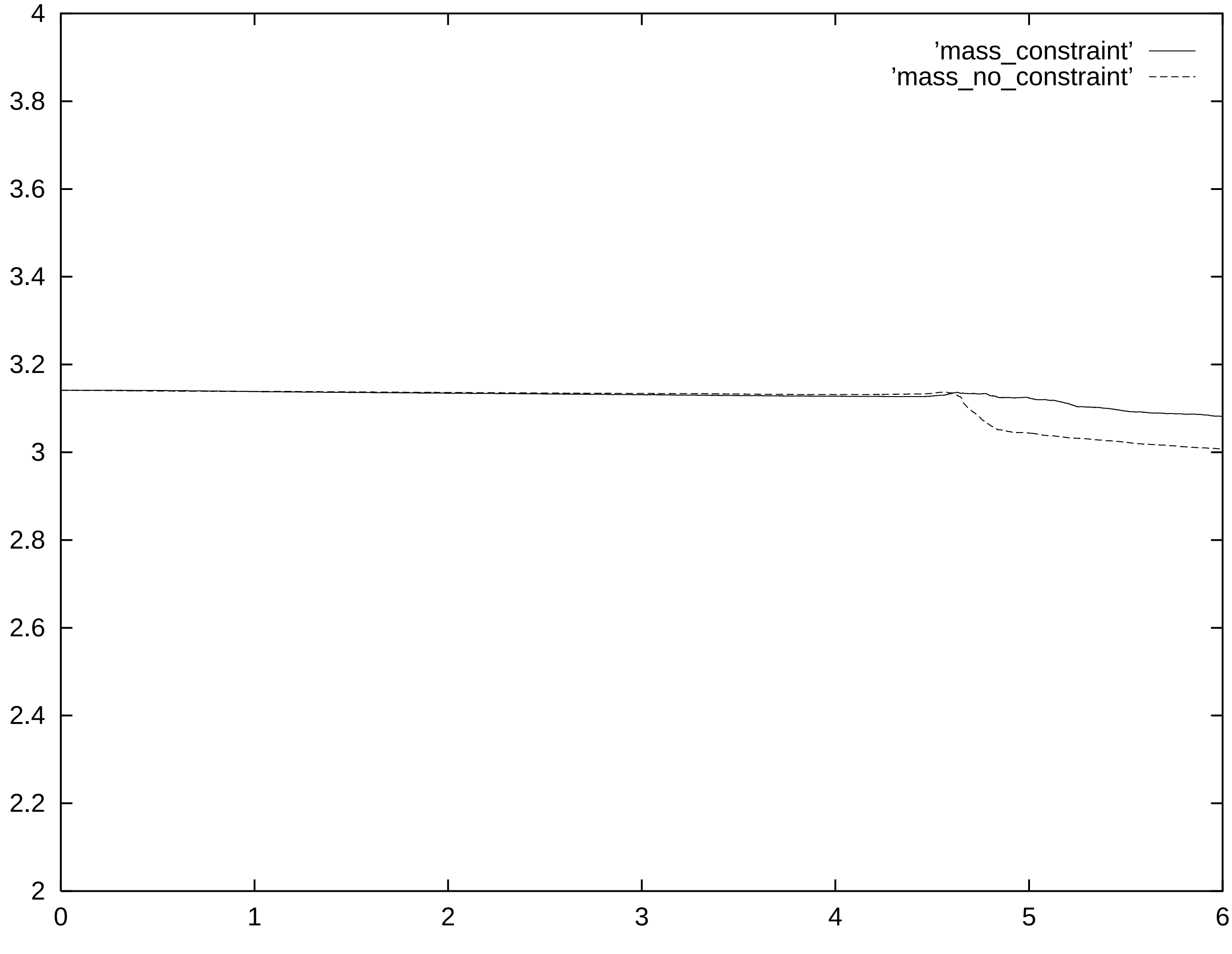

FIG. 9.12. *Mass conservation for rising gas bubble problem. When the redistance scheme with the constraint was used, the average mass error was significantly less than without the constraint; x-axis is time and y-axis represents mass.*

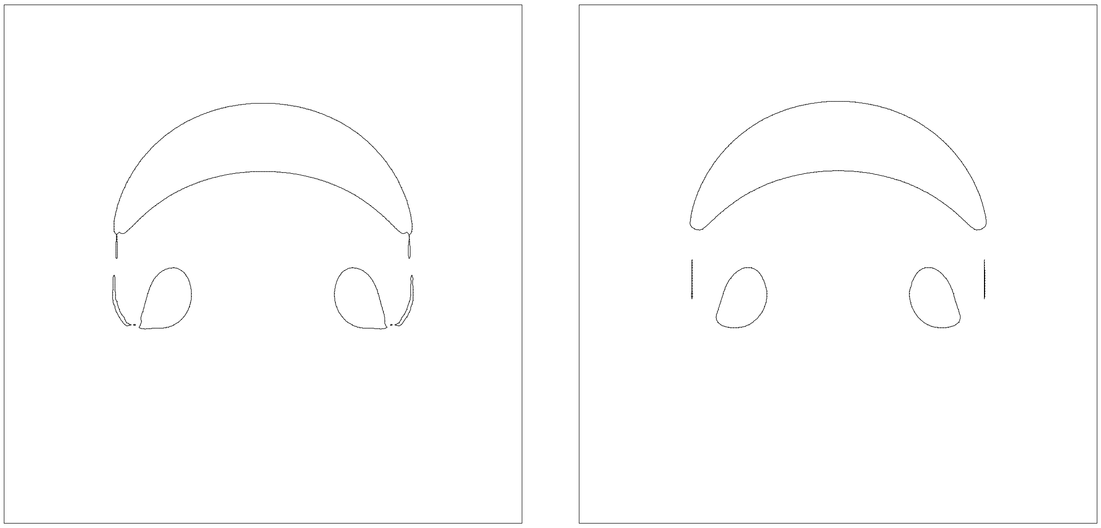

FIG. 9.13. *Comparison at* $t = 4.8$ *of rising gas computations: result on the left used the constraint version of the redistance scheme and the result on the right did not.* $\Delta x = 6/200$, *100x200 (symmetric boundary conditions)*