An Efficient Level Set Method for Constructing Wavefronts in Three Space Dimensions

Li-Tien Cheng [*]

March 22, 2006

Abstract

Wavefront construction in geometrical optics has long faced the twin difficulties of dealing with multivalued forms and resolution of wavefront surfaces. A recent change in viewpoint, however, has demonstrated that working in phase space on bicharacteristic strips using eulerian methods can bypass both difficulties. The success of the level set method in science and engineering makes it a suitable choice for such an eulerian method. Unfortunately, in three-dimensional space, the setting of interest for most practical applications, the advantages of this method are largely offset by a new problem: the high dimension of phase space. In this work, we present new types of level set algorithms that remove this obstacle and demonstrate their abilities to accurately construct wavefronts under high resolution. These results propel the level set method forward significantly as a competitive approach in geometrical optics under realistic conditions.

1 Introduction

In the high frequency regime, wave propagation can be simplified by recasting the wave equation into an eikonal equation for phase and transport equations for amplitude terms in what is known as the geometrical optics approximation. Solution of the phase has a geometric interpretation, involving solution of its level sets, called wavefronts, each of which can be viewed as manifestations of a surface flowing from an initial source through different traveltimes under a time-dependant eikonal equation. The right combinations of viewpoint, framework, and techniques to aid in capturing the generally multi-valued wavefront surfaces under their deformations has been the subject of immense study in applied mathematics.

Two approaches, lagrangian and eulerian, have traditionally dominated the subject of numerical wavefront construction. The algorithms arising from these two frameworks have diametrically opposite characteristics: a lagrangian method easily captures multi-valued behavior but encounters difficulties with the resolution of the surface while an eulerian

[*]Department of Mathematics, University of California San Diego, La Jolla, California 92093

method automatically resolves the surface but has trouble obtaining multi-valued forms. This trade-off has been the main obstacle to producing suitable numerical solutions. See, e.g., [15, 16] for more on lagrangian methods and [1, 3] for eulerian.

Recently, the work of [4] has advanced a different viewpoint that is able to bypass this obstacle. Since higher-dimensional phase space unravels multi-valued wavefronts, it noted that an eulerian method working in this setting can avoid the difficulties of multi-valued forms while having a handle on resolution. Numerical experiments using the segment projection method as the eulerian method provided early justification of these claims.

In light of the development, advances, and success of level set approaches in various fields of science (see, e.g., [11, 10]), the work of [9] continued with the same philosophy but substituted a level set method in place of the segment projection method. Given an initial wavefront, this method represented its form in phase space, the bicharacteristic strip, as the zeros of a vector-valued level set function. Wavefronts of other traveltimes were generated by evolving this function in phase space under the Liouville equation, which translates from the eikonal equation for wavefronts, up to the desired traveltime, then extracting the zeros and projecting them down into spatial space. Numerical experiments showed that multi-valued solutions could be captured and properly resolved in a variety of situations involving variable indices of refraction and reflection.

However, all but one of these experiments were conducting for two spatial dimensions. The reason for this is phase space has twice the dimension of spatial space. For two spatial dimensions, phase space is only four-dimensional, where one further dimension can be removed without complications. For three spatial dimensions, the natural and more physical environment for wave propagation, phase space is six-dimensional. In this setting, grid-based algorithms must place an emphasis on efficiency or face intolerably long computation times and unacceptably large memory requirements. Thus the algorithms presented in [9] showed level set methods could bypass traditional difficulties in geometrical optics, but were not efficient enough to directly extend for use in three spatial dimensions. The only result given for three spatial dimensions could only use 20 points in each dimension for its grid over a five-dimensional reduced phase space, hardly enough to resolve details in the wavefronts. Subsequent advances in [8] could only extend to grids with 64 points in each dimension.

In this paper, we modify the level set approach of [9] so that computations are performed only at gridcells near the bicharacteristic strip, the traditional goal of local level set methods [12]. This effectively reduces the complexity of the approach from the dimension of phase space to roughly that of the bicharacteristic strip itself. The essential technology used is a long-time semi-lagrangian approach (see, e.g., [5, 13]) for solving the Liouville equation that provides the flexibility needed to ignore the majority of gridcells (see [8, 6] for other semi-lagrangian level set methods). Based on this technology and the level set framework, we consider a theoretically fast multi-resolution algorithm, introduce a practically fast growing algorithm, and discuss advantageous combinations of the two. Section 2 introduces essential concepts and tools used in the algorithms presented in section 3. Section 4 then investigates the abilities of the algorithms through numerical experiments and compares results with the competitive approach introduced in [8]. Finally, section 5 reviews the contributions and content of this paper while section 6 acknowledges supporting factors in this paper.

2 Preliminary Tools

2.1 Solving the Liouville Equation

In this section, we summarize a flexible scheme for capturing the level set function as it flows under the Liouville equation. Let $\Phi(x, p, t)$ denote a four component vector-valued level set function existing in phase space, $\{(x, p)|x \in \mathbf{R}^3, p \in \mathbf{R}^3\}$. Suppose $\Phi(x, p, t = 0)$ is given so that its zeros match the two-dimensional bicharacteristic strip for a given wavefront of zero traveltime. The work of [9] gives details for finding such a level set function when the wavefront is single-valued and stable under representation by level set function $\tilde{\phi}(x)$ in spatial space, with the strategy consisting of setting the four components of Φ to be

$$\begin{aligned} (\phi_1, \phi_2, \phi_3) &= c\nabla_x \tilde{\phi}/|\nabla_x \tilde{\phi}| \\ \phi_4 &= \tilde{\phi}, \end{aligned}$$

where $c(x)$ denotes the local wave speed in the medium. Here, the first three components are dedicated to phase direction, which is orthogonal to the wavefront, and the last component to spatial location. The relationship between level set function and bicharacteristic strip is preserved for future traveltimes through the Liouville equation,

$$(\phi_i)_t + c\frac{p}{|p|} \cdot \nabla_x \phi_i - |p|\nabla_x c \cdot \nabla_p \phi_i = 0,$$

on the components ϕ_i of Φ. Thus for simpler notation, we use the term bicharacteristic strip when referring to the zero level set of Φ. An algorithm for capturing wavefronts then seeks to solve the Liouville equation for the level set function up to the desired traveltime, extract its zeros, and project them down into spatial space, $\{x|x \in \mathbf{R}^3\}$, to arrive at the wavefront at that traveltime. Most of the complexity in this procedure lies in solving the Liouville equation.

According to the method of characteristics, the Liouville equation is a transport equation moving values of Φ unchangingly along integral curves of the velocity field $v = (cp/|p|, |p|\nabla_x c)$ in phase space. The linear nature of the partial differential equation allows for application of a lagrangian strategy for solving for Φ at desired locations in phase space at desired times. Fix time T and position (x_T, p_T) in phase space. To find the point (x_0, p_0) at time 0 that flows to (x_T, p_T) along v after time T, we may solve the ordinary differential equation

$$(x(t), p(t))' = v$$

backwards from time T to time 0, with $x(T) = x_T, p(T) = p_T$ given and $x(0) = x_0, p(0) = p_0$ the solution. The value of Φ at (x_T, p_T) at time T can then be obtained from the value of the initial Φ at (x_0, p_0) at time 0 under the relationship

$$\Phi(x_T, p_T, T) = \Phi(x_0, p_0, 0).$$

This strategy becomes semi-lagrangian when the points (x_T, p_T) at time T fit into a grid in phase space. See [6] for more on this procedure in geometrical optics.

There are two advantages to solving the Liouville equation in this manner over using a finite differencing approach, as done in [9]. The first arises from the fact that values at different points and times are independently calculated. This means the calculation of a value at one point and time does not have the large domain of dependence of values needed by the stencils of finite differencing. Related to this, the collection of points where values can be calculated using the semi-lagrangian approach need not have a regular structure and can in fact be composed of a random assortment of points. Finite differencing is not as well-developed and loses many of its desirable abilities on such grids. The second advantage is the removal of the Courant-Friedrichs-Lewy stability restriction appearing in finite differencing approaches.

These advantages provide the flexibility needed to build efficient level set algorithms. The first advantage allows us to calculate values only at gridcells close to the bicharacteristic strip and also to use specialized grids to reduce the dimension of phase space, while the second advantage allows us to take larger steps in time, unhindered by stability constraints, for quicker computation.

2.2 Gridcells and Bicharacteristic Strips

In this section, we seek a method for checking whether a given gridcell intersects a bicharacteristic strip by using values of the level set function generated perhaps with the semi-lagrangian approach of the previous section. In the absence of simple analytical equations describing the bicharacteristic strip, obtaining an exact condition for this check is restrictively complicated; thus we turn to approximate conditions. Consider a grid over the six-dimensional phase space and, as notation, let B denote the set of gridcells that intersect the bicharacteristic strip. Fixing a gridcell of the grid, we check its acceptability using values of Φ at judiciously chosen locations. A natural choice for these locations, and the choice we make for simplicity, is at the gridpoints of the gridcell; however, we do note that restricting ourselves to these locations is actually a relic of grid-based schemes that is no longer needed with the semi-lagrangian method.

Before we list some simple options for checking gridcells, we make the general observation that the geometry of the bicharacteristic strip in the gridcell affects the ability to confirm its presence. If the surface enters and exits quickly out of a gridcell away from the gridpoints, it may be difficult to identify the gridcell as an element of B from the values of Φ there. This is well-known in the subject of interface dynamics. The common solution, refinement of the grid, works for two reasons: it is less likely for a surface to enter and exit quickly out of a smaller gridcell and even if it does, the effect is smaller. Thus we will always consider our gridcells to be sufficiently small so that undercounting in this situation will be negligible.

Our first checking scheme was first proposed in [2, 9]. It involves accepting a given gridcell as an element of B if, at the gridpoints, the values of each component of Φ change sign. This implies, by the intermediate value theorem, that the zero level set of each component passes through the gridcell. Such a check is fast, with only signs of values needed, but in practice, B can be significantly overcounted in many situations, when the zero level set of each component passes through the gridcell but they do not mutually intersect. Thus we seek different options.

Our second checking scheme is an improvement of the first that is ideal in specific situations. Whereas the first approach can be thought of as trying to determine the nature of the bicharacteristic strip by viewing each of its components individually, our second considers pairwise interactions between components. For any two components, we may determine if the values of one change sign at the points where the zero level set of the other intersects the one-dimensional lines composing the gridcell. If there is a sign change, then the intermediate value theorem implies that the zero level sets of the two components intersect. The approach then accepts the gridcell if this condition holds for each pair of components. One-dimensional linear interpolation can be used to determine both the needed points and values from values of Φ at gridpoints, making the checking process fast, though slower than the first approach. This method diminishes the overcounting of B, though overcounting still occurs when pairwise intersections of zero level sets do not mutually intersect. It is, however, ideal in the two-spatial dimension case, seen so often in the examples of [9], that uses a three-dimensional reduced phase space since the level set function has exactly two components. We now seek more accurate options.

Our third checking scheme is our most accurate but slowest one. While the second approach considers pairwise intersections of the zero level sets of the components of the level set function, this one considers the contributions of all components together. This essentially involves extraction of an approximation of the piece of bicharacteristic strip in the gridcell, if it exists. If it does exist, the gridcell is identified as being an element of B; otherwise, it is considered outside B. Note extraction of the bicharacteristic strip is also a necessary step for plotting the wavefront of interest. This checking process, since it arrives at an accurate, usually second-order, representation of the bicharacteristic strip, is the slowest of the three but will avoid serious overcounting or undercounting of B in all but the most degenerate cases.

For extraction of the bicharacteristic strip, we may consider the four-dimensional parts that compose the gridcell and determine whether and where the zero level sets of the four components of the level set function intersect these parts. By forming linear approximations of these components with normal vectors approximated using central differencing at the center of the gridcell, one can solve a linear system of equations with four equations and four unknowns for the intersection with a four-dimensional part. If an intersection lies approximately within the gridcell, then it represents an extracted point of the bicharacteristic strip. For checking, the existence of such a point for a four-dimensional part of the gridcell leads to the acceptance of the gridcell as an element of B. For plotting, the point can be projected to spatial space to obtain one lying on the wavefront. Small spheres drawn around these points then form a thicker version of the bicharacteristic strip of interest that can be visualized using surface plotting programs.

This simple approach, on the other hand, be replaced by more sophisticated options such as that of [7] which returns triangulations of the zero level set surface of vector-valued functions that have arbitrary numbers of components defined in arbitrary dimensions. When plotting is concerned, however, these triangulations tend to contain large numbers of triangles that lead to overly large memory usages in many surface plotting programs.

We note that in [8], properties of the level set function, specifically knowledge of Lipschitz constants, are used to help identify elements of B. Though overcounting occurs in this approach, and in fact is desired to a certain extent, it is another option that may be considered or improved upon.

3 Algorithms

3.1 Multi-Resolution Algorithm

The first approach for wavefront construction we consider, seen in varying degrees in [8, 6], makes use of the tools of the previous section to operate on a multi-resolution grid refining at the bicharacteristic strip of interest. The algorithm can be summarized as follows:

1. Lay down a coarse grid in phase space and fix a time T.
2. For each gridcell, check if the bicharacteristic strip of traveltime T passes through that gridcell.
3. If it does, furher check if the gridcell is smaller than a desired size; otherwise, halt further computations for this gridcell.
4. If it is, extract the piece of bicharacteristic strip and wavefront of traveltime T from the gridcell. Then return to the second step for another gridcell. Otherwise, refine the gridcell and, using recursion, apply the second step to each of the refined gridcells.

Expanding on the details of the steps involved: the grid in the first step can be chosen to be a fixed, regular grid composed of hypercube gridcells based in a fixed computational box assumed to include the bicharacteristic strip up to the time of interest; the check in the second step and extraction in the fourth step follow the discussions of the previous section; and the check in the third step stops the refinement sequence.

Altogether, this produces a theoretically fast and memory efficient algorithm for wavefront construction. To measure the computational complexity, we make two assumptions: first, that the finest level of refinement uses gridcells of a uniform grid with N points in each dimension, and the total number of these gridcells that intersect the two-dimensional bicharacteristic strip is $\mathcal{O}(N^2)$; and second, that the check performed in the third step indeed is able to exactly identify gridcells that intersect the bicharacteristic strip using $\mathcal{O}(1)$ values of Φ for each gridcell checked. With these assumptions, we can conclude that the number of gridcells touched by the algorithm is $\mathcal{O}(N^2)$, which is two-dimensional in nature. This means that though all quantities live in phase space, the algorithm works mainly around the two-dimensional bicharacteristic strip. The total complexity is dominated by the computation of values of Φ for these gridcells, $\mathcal{O}(MN^2)$, assuming the application of an ordinary differential equation solver in the semi-lagrangian scheme that produces values in M steps. For a further increase in speed, the algorithm can be parallelized by allowing separate processors to work on separate gridcells of the coarse grid. In addition, memory usage can be cut down to $\mathcal{O}(\log N)$, what is needed in recursion, by discarding values of gridcells immediately after use. Obviously, if the extracted wavefronts are remembered, this increases to $\mathcal{O}(N^2)$, the number of elements in B at the finest grid level.

On one computer, though, and in practical application with moderately sized N, the number of gridcells this algorithm needs to touch can be restrictively large. The two factors influencing this are the initial coarse grid and, to a lesser extent, the tree-based nature of the algorithm. For example, adopting a coarse grid with just 25 points in each dimension forces the algorithm to work on $64,000,000$ gridcells at the coarsest level, leading to slow computation times, as seen with even fewer gridcells in [9]. In addition, any grid coarser than this may not produce adequate results since the checking schemes of the previous section are not accurate for large gridcells. We thus discard this method, though reincarnate a version of it in section 3.3, and, as the main contribution of our work, achieve faster practical speed using a different approach.

3.2 Growing Algorithm

Consider a grid, allowably fine and uniform, placed in phase space. The philosophy of our new approach is to improve speed by confining computations to gridcells intersecting the bicharacteristic strip, i.e., lying in B. The main challenge will be in finding these gridcells. We begin by identifying one element of B. Let (x_0, p_0) be a point of the initial given bicharacteristic strip. Let (x_T, p_T) be the point after flowing along the velocity field of the Liouville equation for time T (i.e., ray tracing). The gridcell (x_T, p_T) lies in is an element of B. We call this gridcell the seed.

For more elements, we switch to a different strategy. We can, for example, check all gridcells neighboring the seed to see which of them lie in B and continue this process for the ones that do. However, to minimize the number of gridcells touched, which minimizes the number of times the semi-lagrangian method is used, we take a different approach. By checking which faces of the seed the bicharacteristic strip passes through, identifying neighboring gridcells sharing those faces as elements of B, and continuing this process for these newly identified elements, we avoid calculating additional values of Φ. Visually, this approach grows gridcells in B outwards from the initial seed.

The algorithm can be characterized as follows:

1. Lay down a fine grid in phase space and fix a time T.

2. Choose a point of the initial bicharacteristic strip and solve the ordinary differential equation along the velocity field of the Liouville equation to determine its position at time T.

3. Add the gridcell containing that point to the end of a list of recently identified elements of B and label it as the first generation of gridcells identified.

4. Proceeding through the list in order, consider one of its gridcells. Remove elements from the list whose generations are one or two less than that of the current gridcell.

5. Extract the piece of bicharacteristic strip and wavefront of traveltime T from the gridcell.

6. Check which faces of the gridcell the bicharacteristic strip exits out of and add the neighboring gridcells that share those faces, if not redundant, to the end of the list with generation label one above that of the current one. Then return to the fourth step for another gridcell.

This algorithm can be viewed as employing a breadth-first search strategy with constraints on inclusion. Elements of B are identified in an expansion out of the seed, ordered from least to furthest distance away under the taxicab metric. Thus, geometrically, the bicharacteristic strip is extracted as a surface expanding out of the seed. Due to this type of growth, note when there are disconnected components of bicharacteristic strips, the algorithm only constructs the specific component connected to the seed.

Expanding on the details of certain steps: the grid of the first step is fine enough to provide the desired resolution but need not physically exist in memory; the second and third steps seed an initial gridcell of B for iteration of the fourth, fifth, and sixth steps; the fourth step culls the list, removing elements that are not of recent origin and thus no longer needed for the redundancy search in the sixth step; and the extraction in the fifth step follows the discussions of section 2.2 to obtain a piece of the wavefront for plotting, as does the check in the sixth step when faces are viewed as lower-dimensional gridcells.

The number of gridcells the algorithm touches, if we can indeed determine which faces of a gridcell a bicharacteristic strip exits out of, is just the number of gridcells the bicharacteristic strip intersects. Thus, comparing with the multi-resolution algorithm that uses the same grid at its finest scale, our growing algorithm computes on fewer gridcells. The size of the list of recently identified elements, however, adds to the complexity of the approach. The gridcells needed in the list are those neighboring the boundary of the growing set of identified elements of B. This means the elements of the list lie near the one-dimensional boundary of the expanding extracted bicharacteristic strip surface. Thus, using the same notations and under the same assumptions made for the multi-resolution algorithm, we expect the number of elements in the list to be $\mathcal{O}(N)$. A direct consequence of this is that memory storage requirements for the list, and algorithm, when gridcells not in use are discarded, will be $\mathcal{O}(N)$, or $\mathcal{O}(N \log N)$ if the list is additionally stored as a tree. In terms of speed, each redundancy search will have complexity $\mathcal{O}(N)$ with linear searching through the list, or $\mathcal{O}(\log N)$ with tree-based searching. Thus the algorithm's total complexity will receive contributions of $\mathcal{O}(MN^2)$ from computing values of Φ at $\mathcal{O}(N^2)$ gridcells and either $\mathcal{O}(N^3)$ or $\mathcal{O}(N^2 \log N)$ from searching. In the case where M is of the same size as N, the complexity will be the same as for the multi-resolution approach.

From this analysis, we see that the growing algorithm may have the same complexity as the multi-resolution algorithm for large N; however, it performs much faster for N used in practical situations. We show experimental results of its speed, memory, and constructed wavefronts in section 4.

3.3 Variations

One variation of our algorithms involves replacing the six-dimensional phase space with the one-dimension lower reduced phase space $\{(x, p/|p|)|x \in \mathbf{R}^3, p/|p| \in \mathbf{S}^2\}$. In this setting, the level set function has three components and evolves under the appropriate translation of the Liouville equation. Good parametrizations of the sphere, however, are lacking. The attempts made in [9] were far from optimal due to use of spherical coordinates which are notably bad around the poles of the sphere. The semi-lagrangian method for solving the Liouville equation allows us to consider other options. Choosing a good triangulation of the sphere, we may work on a grid composed of gridcells that are products of hypercube gridcells of the spatial grid with triangles from the the triangulation of the sphere. Both of our algorithms carry through as usual using these new gridcells and the correct Liouville equation.

The advantage of using reduced phase space is an increase in speed due to reduction of dimension from six to five, which lowers the number of gridpoints from 64 to 24 and faces and neighbors from 12 to 9 for each gridcell. Furthermore, the level set function is reduced by one component. One disadvantage is the slight complication arising from the use of triangulations and how to generate, store, and access them. We currently just store the complete triangulation in memory, increasing memory requirements to $\mathcal{O}(N^2)$. Another disadvantage is related to initialization of the level set function. The approach of assigning the components ϕ_1 and ϕ_2 for phase direction and ϕ_3 for spatial location will often lead to the creation of a level set function encoding two bicharacteristic strips: one moving in the desired direction and the other in the opposite direction. This is not a problem for the growing algorithm, which will only pick out the correct one since the two are disconnected; however, the multi-resolution approach needs modification so that additional work in capturing a spurious surface is avoided.

Other variations involve hybrids of the multi-resolution and growing algorithms. The growing algorithm used first over a coarse grid can identify gridcells that intersect the bicharacteristic strip. The multi-resolution process can then continue on these gridcells, eliminating the need to initially try out all gridcells of the coarse grid. This removes the main obstacle to fast computations for practical N in the multi-resolution algorithm while preserving the same order of speed and memory storage requirements for large N. In addition, such a hybrid can apply smaller grids to regions of the bicharacteristic strip that deserve more attention, whether due to a desire for better resolution, accuracy, or to alleviate the symptoms of possible instability in the level set representation. We expand on the latter as it may represent a final hurdle for level set approaches in geometrical optics.

Stability refers to the change in the zero level set under perturbation of the level set function. If any of the components develops small derivatives at the bicharacteristic strip, a small perturbation may move its zero level set significantly. Furthermore, if the zero level set of one component is almost parallel to that of another, a small perturbation may move their intersection significantly. Both cases make it difficult to obtain accurate and reliable information from the level set function needed, for example, to extract and plot the zero level set. Better resolution at unstable locations fixes certain types of instabilities, possibly the ones that appear with frequency, though this is under investigation. Our current idea for removing all types of instabilities involves running the algorithms in intervals in time where, at the end of each interval, Φ is reinitialized to a more stable form (see [2, 14] for more on reinitialization). The added operations, however, will decrease the speed of our algorithms. Complexities when reinitialization is performed at each step of the ordinary differential equation solver of the semi-lagrangian method will be $\mathcal{O}(MN^2)$ for the multi-resolution algorithm, $\mathcal{O}(MN^3)$ for the growing algorithm with linear searching, and $\mathcal{O}(MN^2 \log N)$ for the growing algorithm with tree-based searching. In-depth studies of instability and approaches for its removal must be left to future research.

4 Numerical Results

4.1 Speed and Memory

We begin by testing the speed and memory usage of our algorithms. Consider the initial wavefront a point source at y in a medium of local wave speed $c \equiv 1$, given in phase space by the representation

$$\begin{aligned} (\phi_1, \phi_2, \phi_3) &= y \\ \phi_4 &= |p| - 1. \end{aligned}$$

Wavefronts of other traveltimes form spheres growing out of the initial point source. The constant local wave speed implies the identity

$$\Phi(x, p, t) = \Phi(x - pt, p, 0),$$

since the integral curves of the velocity field for the Liouville equation are straight lines. Use of this identity to obtain values of Φ allows us to determine the contributions of phase space independent of the traveltime. Note, also, this can be thought of as using Euler's method to jump to the solution in one giant step, and this is made possible because there is no stability constraint to satisfy between gridsize and time step.

We study the properties of three approaches: the growing algorithm with linear searching, the growing algorithm with tree-based searching, and a hybrid that follows the growing algorithm with multi-resolution refinement. All three use pairwise intersections of components of the level set function to check for intersections of gridcells with bicharacteristic strips, our second approach listed in section 2.2.

Our first two algorithms employ the growing algorithm but with different strategies in storing their lists of recently identified elements of B, with ramifications in the redundancy searches conducted through these lists prior to the adding of new elements. Linear searching proceeds one by one through the list to see if redundancy occurs while tree-based searching stores the list also as a tree to perform optimal searches. Table 1 counts several quantities

time	grid	cells	order	searches	order	list	order
0	50^6	$11,864$		$9,524,917$		543	
0	100^6	$47,168$	1.9912	$73,254,143$	2.9431	$1,032$	0.9264
0	200^6	$188,600$	1.9994	$581,551,384$	2.9889	$2,043$	0.9852
0.5	50^6	$26,750$		$31,793,290$		801	
0.5	100^6	$107,843$	2.0113	$256,756,224$	3.0136	$1,593$	0.9919
0.5	200^6	$437,505$	2.0204	$2,117,975,866$	3.0442	$3,204$	1.0081
1	50^6	$48,128$		$76,236,759$		$1,077$	
1	100^6	$195,249$	2.0204	$633,884,315$	3.0557	$2,199$	1.0298
1	200^6	$798,201$	2.0314	$5,311,800,799$	3.0669	$4,356$	0.9862

Table 1: This table counts quantities of importance to speed and memory in the growing algorithms.

of importance to speed and memory in growing the spherical wavefront when the computational domain in phase space is $[-1, 1]^6$ and $y = (0.001, 0.001, 0.001)$, chosen not to lie on a gridpoint.

In this table, the first column presents the traveltime of the wavefront constructed and the second the underlying fine grid. The third column counts the number of gridcells touched by the algorithms. This number contributes to speed since each gridcell requires several solves of the Liouville equation for values of the level set function. We make two comments on the numbers in this column. The first is that though we have not adopted the most accurate checking routine for determining intersections of gridcells with bicharacteristic strips, use of the third option of section 2.2 will only change the counts by 4% at most for this problem. Other problems, however, may need to settle for the slower but more accurate routine. The second comment is that because N is not large enough, the numbers in the column do not share a close relationship with the change in surface area of the bicharacteristic strip at different traveltimes, which can be calculated to be $4\pi(\rho^2 + 1)$ for a spherical wavefront of radius ρ. This relationship is satisfied, however, when N and traveltime are much larger than what is given in this table.

Continuing with the table, the fourth column determines the orders associated with these numbers from the use of finer grids and verifies the $\mathcal{O}(N^2)$ complexity corresponding to touched gridcells. The fifth column counts the number of elements in the algorithms' lists of recently identified elements. This number also contributes to speed, counting comparisons performed in the redundancy searches through the lists. The sixth column determines the order associated with these numbers and verifies the $\mathcal{O}(N^3)$ corresponding to searches required in linear searching. The seventh column counts the maximum number of elements attained in the lists. This number contributes to the memory usage of the algorithms in storing these quantities. The eighth column determines the order associated with these numbers and verifies the $\mathcal{O}(N)$ complexity corresponding to the maximium number of list elements.

Tables 2 and 3 give further verification of speed and memory estimates based on actual recorded times and memory usages for the growing algorithms on a computer. These num-

time	grid	runtime	order	memory
0	50^6	2.304		900
0	100^6	8.944	1.9568	924
0	200^6	37.109	2.0528	964
0.5	50^6	5.291		912
0.5	100^6	22.061	2.0599	944
0.5	200^6	95.512	2.1142	$1,004$
1	50^6	9.466		924
1	100^6	41.286	2.1248	964
1	200^6	178.848	2.1150	$1,056$

Table 2: This table collects runtimes and memory storages of the growing algorithm with linear searching.

time	grid	runtime	order	memory
0	50^6	2.222		992
0	100^6	8.872	1.9974	$1,112$
0	200^6	35.919	2.0174	$1,364$
0.5	50^6	5.372		$1,164$
0.5	100^6	21.749	2.0174	$1,496$
0.5	200^6	92.414	2.0872	$2,228$
1	50^6	9.383		$1,348$
1	100^6	40.110	2.0958	$1,936$
1	200^6	169.852	2.0822	$3,180$

Table 3: This table collects runtimes and memory storages of the growing algorithm with tree-based searching.

bers, however, are subject to randomness and programming style and so table 1 may be a better source of information on the speeds and memory usages of the algorithms.

In both these tables, the first column presents the traveltime of the wavefront constructed and the second the underlying fine grid. The third column gives the measured times in seconds, averaged over ten samples, for completion of the algorithms. The fourth column determines the order associated with these numbers. The orders of table 3 agree with the $\mathcal{O}(N^2 \log N)$ complexity of the growing algorithm with tree-based searching; however, those of table 2 are better than the $\mathcal{O}(N^3)$ complexity predicted for the growing algorithm with linear searching. This hints at the fact that only much larger N will produce that theoretical behavior in complexity. The fifth column reveals the amounts of memory used, in kilobytes, by the algorithms. This includes 796 kilobytes allocated by the programs in all cases before execution of the first steps of the algorithms. From these tables, we note tree-based searching gives slight benefits in speed for the grids used but larger memory usage due to overhead for

time	grid	cells	order	final	order
0	50^6	$188,890$		$11,840$	
0	100^6	$946,650$	2.3253	$47,144$	1.9934
0	200^6	$3,963,866$	2.0660	$188,576$	2.0000
0.5	50^6	$432,770$		$26,726$	
0.5	100^6	$2,143,234$	2.3081	$107,819$	2.0123
0.5	200^6	$9,043,650$	2.0771	$437,469$	2.0206
1	50^6	$764,400$		$48,002$	
1	100^6	$3,836,528$	2.3274	$194,961$	2.0220
1	200^6	$16,314,032$	2.0882	$797,268$	2.0319

Table 4: This table counts quantities of importance to speed and memory in the hybrid algorithm.

storing tree-structures.

Turning to tests of a hybrid algorithm, we consider a growing algorithm with linear searching strategy performed over a coarse grid with 25 points in each dimension followed by multi-resolution refinement on the resulting identified gridcells. Refinement in this case consists of subdividing hypercube gridcells in half in each dimension. Table 4 counts several quantities of importance to speed and memory in the same spherical wavefront setting.

In this table, the first column presents the traveltime of the wavefront constructed and the second the underlying fine grid. The third column counts the number of gridcells touched by the algorithm, with the growing algorithm contributing $2,906$ of these from the initial coarse grid. Note these numbers are much larger than those of our growing algorithms studied in table 1 but also much smaller from the $64,000,000$ needed just at the coarsest grid level if the multi-resolution algorithm were applied alone. The fourth column determines the order associated with the information of the third column from the use of finer grids and verifies the $\mathcal{O}(N^2)$ complexity corresponding to touched gridcells. The fifth column counts the number at the finest grid level identified to be in B. These numbers approximately coincide with those identified by the growing algorithms in table 1 but are not equal because the large size of gridcells in the coarse grid causes the growing algorithm to slightly undercount B. This explanation is verfied through the use of finer grids at the coarse level which results in matching numbers between the tables. The seventh column determines the order associated with the information of the fifth column and verifies $\mathcal{O}(N^2)$ complexity of identified elements of B.

Table 5 gives further verification of speed and memory estimates based on actual recorded times and memory usages for the hybrid algorithm on a computer. These numbers, again, are subject to randomness and programming style and so table 4 may be a better source of information on the speed and memory usage of the algorithm.

In this table, the first column presents the traveltime of the wavefront constructed and the second the underlying fine grid. The third column reveals the measured times in seconds, averaged over ten samples, for completion of the algorithm. We see that this algorithm is

time	grid	runtime	order	memory
0	50^6	7.849		900
0	100^6	33.018	2.0727	904
0	200^6	132.597	2.0057	904
0.5	50^6	18.046		908
0.5	100^6	80.631	2.1597	908
0.5	200^6	332.910	2.0457	908
1	50^6	29.726		912
1	100^6	140.280	2.2385	916
1	200^6	591.672	2.0765	916

Table 5: This table collects runtimes and memory storages of the hybrid algorithm.

slower than the growing algorithms studied earlier. The fourth column determines the orders associated with the numbers in the third column and verifies the $\mathcal{O}(N^2)$ complexity of our hybrid algorithm. The fifth column reveals the amounts of memory used, in kilobytes. This includes 800 kilobytes allocated in all cases before execution of the first step of the algorithm. Note the low amounts of memory used here throughout the experiments.

In total, we get a sense from these experiments of the speeds and memory usages of our algorithms and the sizes of the grids allowed. Compared with the results of [8], we have been able to run simulations on grids with 400^6 cells while that of [8] shows results on grids with only up to 64^5 cells. Furthermore, since it may not be fair to compare actual speeds and memory usages due to the randomness of the numbers and the different computers used, we merely say that we believe our algorithms are better in both respects. Thus we conclude that we have produced one of the first efficient level set algorithms for large-scale computations in wavefront construction.

4.2 Wavefronts

We now present plots of some of the wavefronts constructed by our algorithms in three dimensions. Starting with constant local wave speed, $c \equiv 1$, figure 1 shows a shrinking ellipse where two of the principal radii are the same, constructed by the growing algorithm with linear searching and a 100^6 point grid. The results correspond to surfaces of rotation of shrinking ellipses in two-dimensional space, such as those shown in [9]. From the figure, we see that not only are surface details resolved but multi-valued characteristics are correctly displayed. Figure 2 shows a shrinking ellipse where the principal radii are all different, constructed using the same algorithm. Again, resolution and multi-valued forms are satisfactorily handled by our approach. These two figures are in fact plotted by triangulating the bicharacteristic strip. When working with gridcells from the algorithm's list of recently identified elements, we may take a gridcell's center, that of a neighbor further down the list, and that of the neighbor's neighbor even further down the list and project them to the bicharacteristic strip to form a triangle lying on the surface. The number of triangles, though

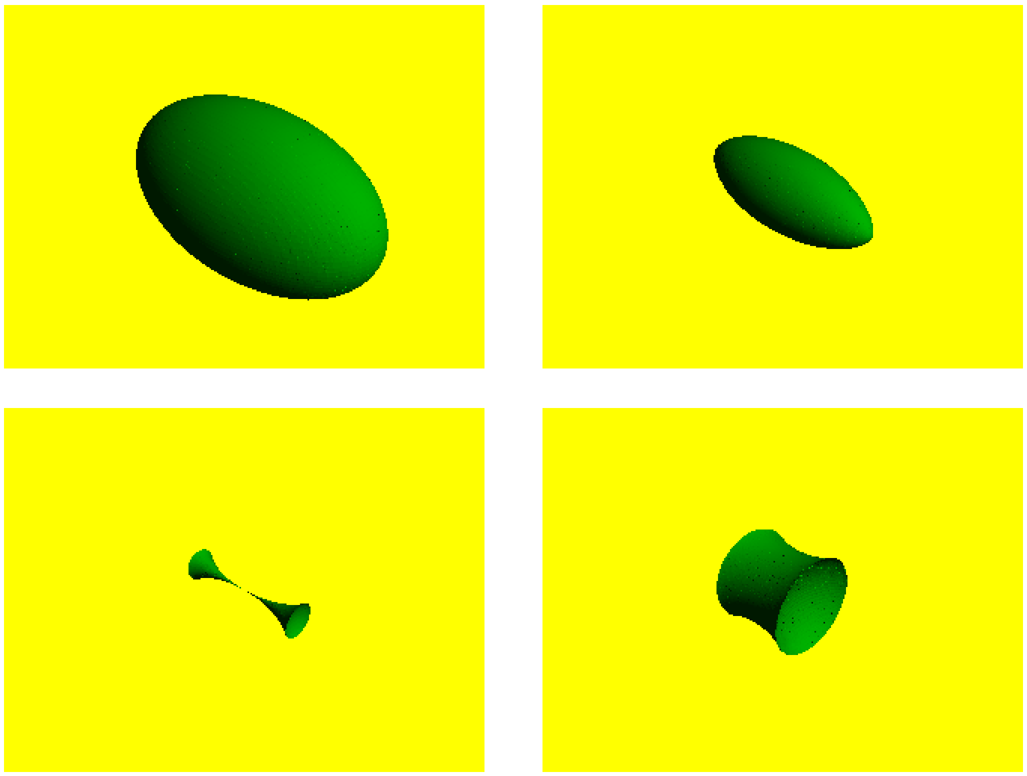

Figure 1: Ellipsoid with two equal principal radii shrinking into multi-valued forms.

smaller than that returned by application of [7], is large but for these two cases within a range that can be handled by the standard surface plotters.

To project the center of the gridcell onto the bicharacteristic strip, we may apply central differencing there to approximate derivatives of each component from values of Φ at grid-points. These can then be used to form linear approximations of each component. Assigning our point to be the one at the intersection of the hyperplane zero level sets of minimum distance from the center, computable using lagrange multipliers on this constrained quadratic minimization to form a linear system of equations, gives us our point on the bicharacteristic strip. Note drawing spheres around such points for each gridcell in B leads to another plotting scheme for the bicharacteristic strip. This is, in fact, the approach we use for our remaining figures.

Our next series of results keep a constant local wave speed, $c \equiv 1$, but introduce complexity through the presence of reflecting solids in the environment. The algorithm incorporates two modifications to treat this problem. The first involves changing the semi-lagrangian scheme at the boundaries of solids so that a point flowing along the velocity field of the Liouville equation, when it finds itself entering a solid, will be lifted to a direction exiting the solid according to reflection conditions. The discontinuity of this flow at these boundaries, however, will generate disconnected bicharacteristic strips forming the wavefront. Thus the second modification involves the use of multiple initial seed points in the growing algorithm,

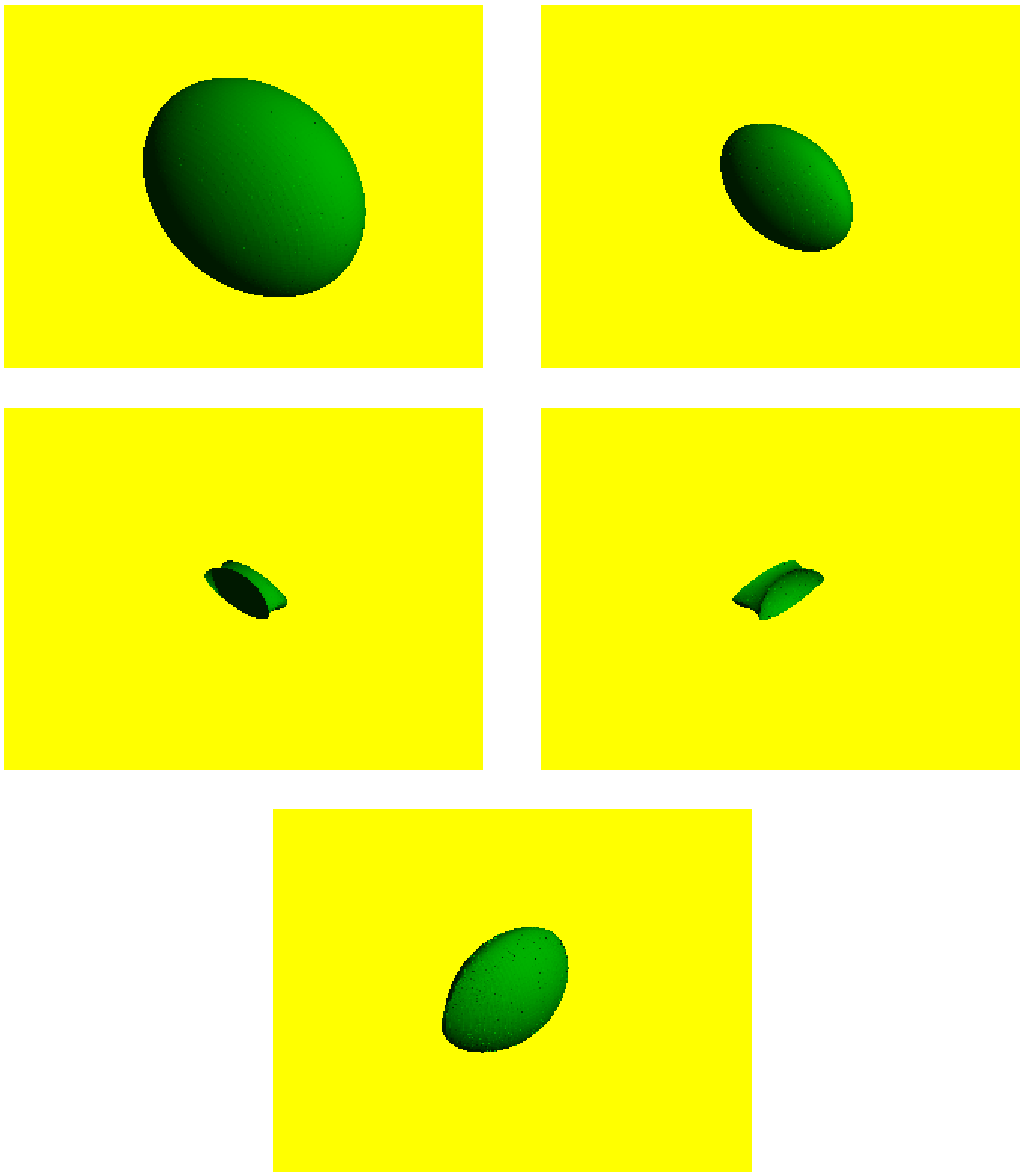

Figure 2: Ellipsoid with different principal radii shrinking into multi-valued forms.

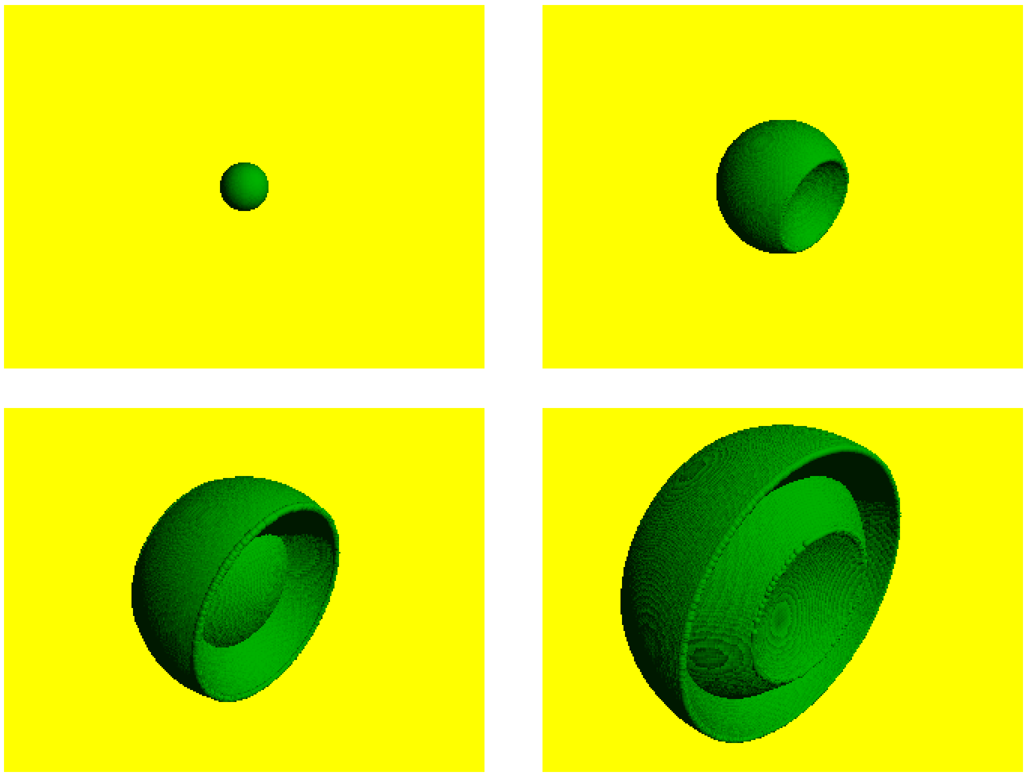

Figure 3: Growing sphere trapped between and reflecting off two walls.

one each for disconnected piece of reflected bicharacteristic strip. In this work, we assume that these seed points are given, while in future work, we will try to view the bicharacteristic strips as connected through the reflection conditions at the boundaries of the solids and use only one seed point. Other advances for reflection and refraction are also left to future work.

In figure 3, the reflecting boundaries are two walls and the wavefront grows from a point source located in the middle. The wavefront reflects off one wall, travels to the other, and again reflects off that one. In figure 4, the reflecting solid is an ellipsoid and the initial wavefront is a shrinking sphere surrounding it. The wavefront first reflects off the elongated ends of the ellipsoid. At later time, when all points have reflected off the ellipsoid, wavefront grows out of the encounter. In both examples, the multi-valued characteristics of the wavefronts are captured and the growing surfaces are properly resolved.

Our last series of results introduce complexity with a variable local wave speed,

$$c(x) = 0.5 \sin(2\pi x_1) \sin(2\pi x_2) \sin(2\pi x_3) + 1.$$

Starting with a spherical wavefront of radius 0.4, figure 5 shows the behavior of an initially shrinking sphere under our growing algorithm with linear searching. Runge-Kutta of third order is used to solve the ordinary differential equations of the semi-lagrangian scheme and second-order extraction techniques, our third approach in section 2.2, are employed for a more accurate test of the intersection of gridcells with bicharacteristic strips. Throughout

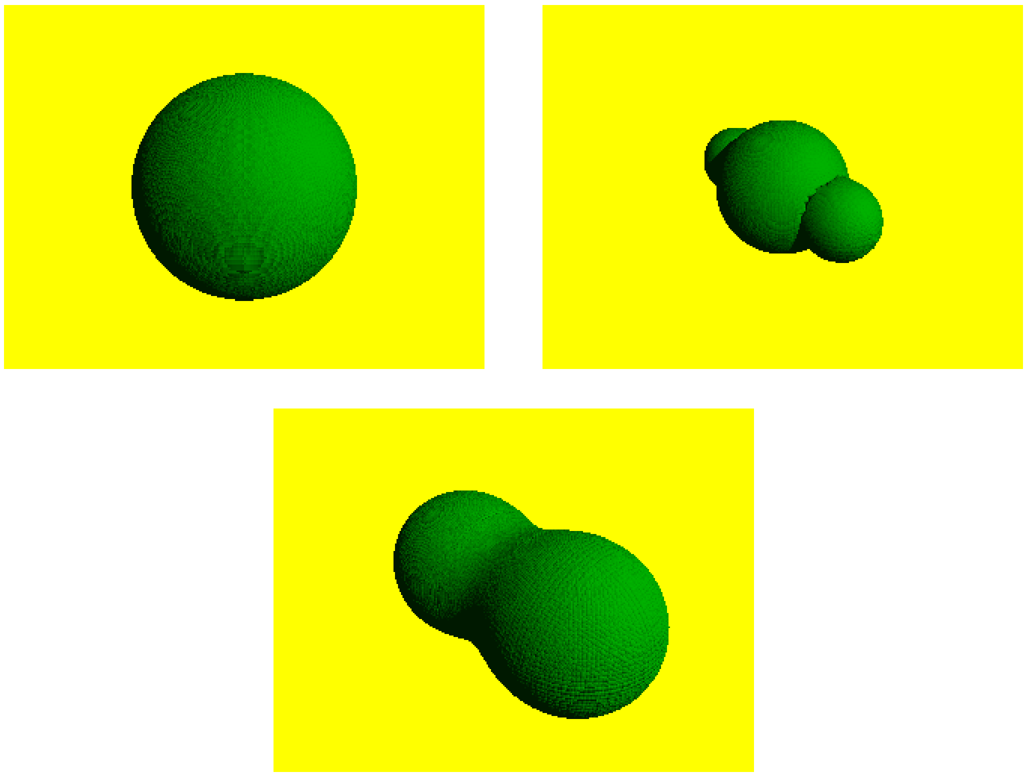

Figure 4: Shrinking sphere reflecting off an ellipsoid in the interior.

the evolution shown in the figure, the wavefronts remain well-resolved and their multi-valued characteristics are properly captured by the algorithm, as verified when compared to ray tracing results. Because the surfaces in the figures come from the plotting of spheres around points on the wavefront, corners on the multi-valued surface may not appear as sharp as they in reality are.

5 Conclusion

We introduced in this work a multi-resolution algorithm and a growing algorithm for efficient construction of wavefronts in three spatial dimensions. Both of these algorithms take advantage of the flexibility afforded by the long-time semi-lagrangian method for solving the Liouville equation to reduce the number of gridcells that require calculation. The multi-resolution algorithm has good speed and memory characteristics while the growing algorithm is faster in practice. Hybrids of the two improve on certain aspects of both approaches and can alleviate the effects of unstable level set representations. With three spatial dimensions now very much a reality, investigating better methods for removing instabilities is the last hurdle for the level set method in wavefront construction.

6 Acknowledgements

The author would like to thank Jianliang Qian, Stanley Osher, Chohong Min, and Yen-Hsi Richard Tsai for enlightening discussions and encouragement throughout this work. Research was supported in part by a Sloan Fellowship and NSF Grant #0511766.

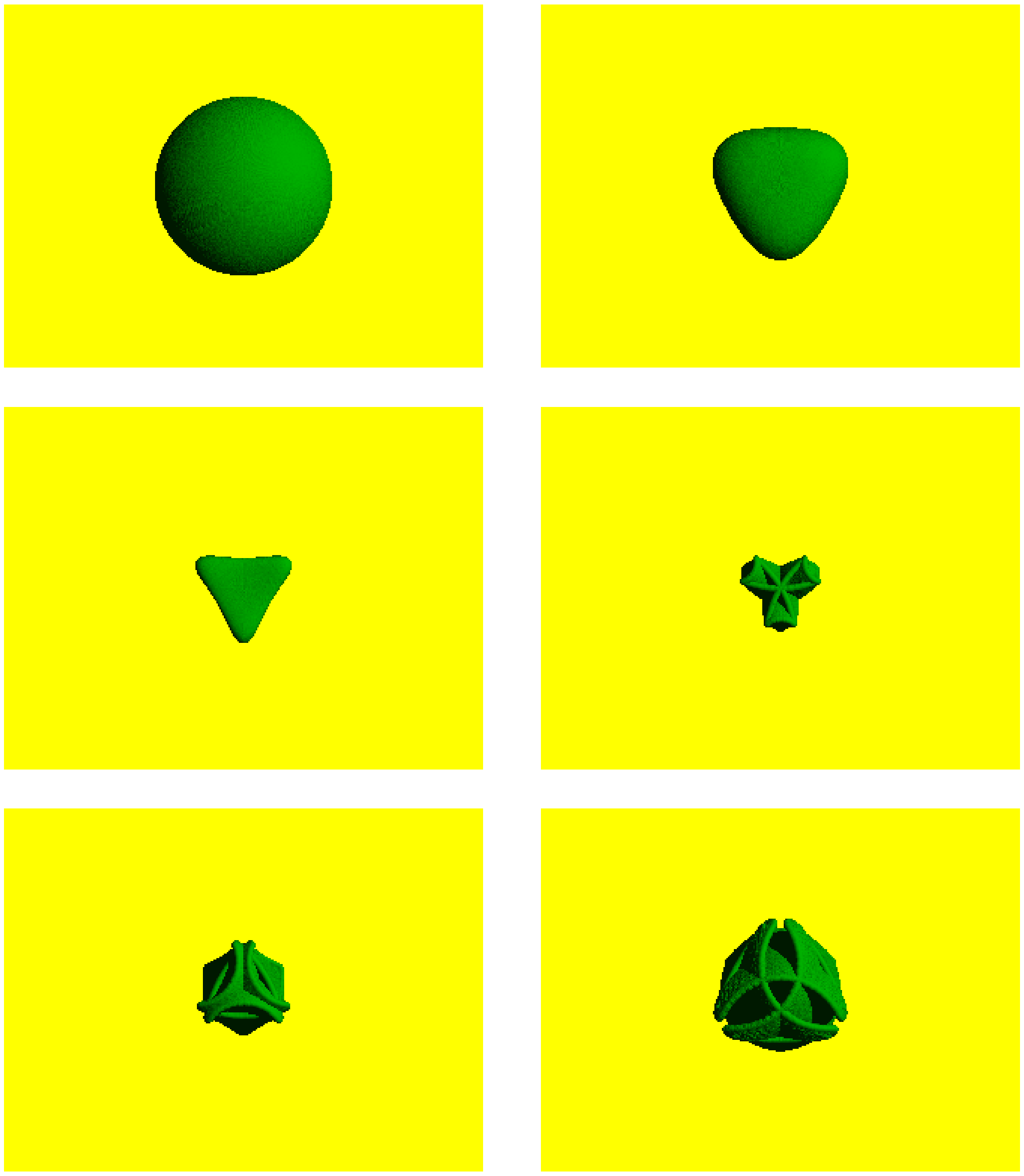

Figure 5: Shrinking sphere in medium with variable local wave velocity.

References

[1] J.-D. Benamou. An Introduction to Eulerian Geometrical Optics (1992–2002). *J. Sci. Comp.*, 19:63–93, 2003.

[2] P. Burchard, L.-T. Cheng, B. Merriman, and S. Osher. Motion of Curves in Three Spatial Dimensions Using a Level Set Approach. *J. Comput. Phys.*, 170(2):720–741, 2001.

[3] B. Engquist and O. Runborg. Computational High Frequency Wave Propagation. In *Acta Numerica*, pages 1–86. Cambridge University Press, Cambridge, United Kingdom, 2003.

[4] B. Engquist, O. Runborg, and A.-K. Tornberg. High Frequency Wave Propagation by the Segment Projection Method. *J. Comput. Phys.*, 178(2):373–390, 2002.

[5] M. Falcone and R. Ferretti. A Non-Oscillatory Eulerian Approach to Interfaces in Multimaterial Flows (The Ghost Fluid Method). *J. Comput. Phys.*, 152:457–492, 1999.

[6] S. Leung, J. Qian, and S. Osher. A Level Set Method for Three-Dimensional Paraxial Geometrical Optics with Multiple Point Sources. *Comm. Math. Sci.*, 2(4):657–686, 2004.

[7] C. Min. Simplicial Isosurfacing in Arbitrary Dimension and Codimension. *J. Comput. Phys.*, 190(1):295–310, 2003.

[8] C. Min. Local Level Set Method in High Dimension and Codimension. *J. Comput. Phys.*, 200(1):368–382, 2004.

[9] S. Osher, L.-T. Cheng, M. Kang, H. Shim, and Y.-H. Tsai. Geometric Optics in a Phase Space Based Level Set and Eulerian Framework. *J. Comput. Phys.*, 179(2):622–648, 2002.

[10] S. Osher and R. Fedkiw. Level Set Methods: An Overview and Some Recent Results. *J. Comput. Phys.*, 169:463–502, 2001.

[11] S. Osher and J.A. Sethian. Fronts Propagating with Curvature Dependent Speed: Algorithms Based on Hamilton-Jacobi Formulations. *J. Comput. Phys.*, 169(1):12–49, 1988.

[12] D. Peng, B. Merriman, S. Osher, H.K. Zhao, and M. Kang. A PDE-Based Fast Local Level Set Method. *J. Comput. Phys.*, 155(2):410–438, 1999.

[13] J. Strain. Semi-Lagrangian Methods for Level Set Equations. *J. Comput. Phys.*, 151:498–533, 1999.

[14] M. Sussman, P. Smereka, and S. Osher. A Level Set Method for Computing Solutions to Incompressible Two-Phase Flow. *J. Comput. Phys.*, 114:146–159, 1994.

[15] V. Vinje, E. Iversen, K. Astebol, and H. Gjøystdal. Estimation of Multivalued Arrivals in 3D Models using Wavefront Construction–Part I. *Geophysical Prospecting*, 44:819–842, 1996.

[16] V. Vinje, E. Iversen, K. Astebol, and H. Gjøystdal. Part II: Tracing and Interpolation. *Geophysical Prospecting*, 44:843–858, 1996.