Geometric Shock-Capturing ENO Schemes for Subpixel Interpolation, Computation and Curve Evolution

Kaleem Siddiqi [1] Benjamin B. Kimia [2] Chi-Wang Shu [3]

February 1995; Revised: July 1996.

[1]siddiqi@cim.mcgill.ca; Department of Electrical Engineering, McGill University, Montréal, Québec, Canada H3A 2A7.
[2]kimia@lems.brown.edu; Division of Engineering, Box D, Brown University, Providence, RI 02912, USA.
[3]shu@cfm.brown.edu; Division of Applied Math, Box F, Brown University, Providence, RI 02912, USA.

Geometric Shock-Capturing ENO Schemes for Subpixel Interpolation, Computation and Curve Evolution

Kaleem Siddiqi, Benjamin B. Kimia and Chi-Wang Shu

Abstract Subpixel methods that locate curves and their singularities, and that accurately measure geometric quantities, such as orientation and curvature, are of significant importance in computer vision and other applications. Such methods often use local surface fits or structural models for a local neighborhood of the curve, to obtain the interpolated curve. Whereas their performance is good in smooth regions of the curve, it is typically poor in the vicinity of singularities. Similarly, the computation of geometric quantities is often regularized to deal with noise present in discrete data. However, in the process, discontinuities are blurred over, leading to poor estimates at them, and in their vicinity. In this paper we propose a geometric interpolation technique to overcome these limitations by locating curves and obtaining geometric estimates while: 1) not blurring across discontinuities, and 2) explicitly and accurately placing them. The essential idea is to avoid the propagation of information across singularities. This is accomplished by a one-sided smoothing technique, where information is propagated from the direction of the side with the "smoother" neighborhood. When both sides are non-smooth, the two existing discontinuities are relieved by placing a single discontinuity, or shock. The placement of shocks is guided by geometric continuity constraints, resulting in subpixel interpolation with accurate geometric estimates. The interpolations are shown to be better than spline-like interpolations in smooth regions, and far better in discontinuous ones. Since the technique was originally motivated by curve evolution applications, we demonstrate its usefulness in capturing not only smooth evolving curves, but also discontinuous ones. In particular, the technique is shown to be far better than traditional methods when multiple or entire curves are present in a very small neighborhood.

Keywords: ENO interpolation, shock detection, geometric estimates, curve evolution.

1 Introduction

Hyperacuity, or the ability of the human visual system to detect features at resolutions an order of magnitude better than retinal resolution, is a remarkable phenomenon. This phenomenon is particularly impressive since locating curves, *e.g.*, for image registration [13, 46], as well as obtaining reliable estimates of geometric quantities, *e.g.*, orientation and curvature, from discrete data at normal resolution, have traditionally been difficult problems. Nevertheless, accurate estimates of orientation and curvature are important for a number of computer vision problems, *e.g.*, orientation for computing stereo disparity [41, 42], and boundary-based optical flow [64, 19]; accurately locating curvature extrema and inflection points [43, 4, 20, 7, 30]; and finally for curve evolution applications where the PDE governing the deformation of the curve is often a function of the curve's local geometry. In fact, while the latter application has been the main driving force behind the development of this paper, the findings have general applications.

We follow [22] in exploiting *two* main sources of information to obtain measurements of differential structure in images: 1) normal conditions to utilize information along profiles *orthogonal* to image curves, and 2) tangential conditions to take advantage of continuity *along* image curves. However, our goal is to achieve robust localization and accurate geometric estimates, not only at normal, but also at *subpixel* resolution. First, observe that image structure, *e.g.*, shape, is rarely present in binary form. Hyde and Davis considered the problem of locating image edges with subpixel accuracy, using "contrast" information orthogonal to the edges [21]. They showed that taking the intensities of the edge pixels into account yields little or no improvement over a least squares fit through edge pixel locations in smooth regions. They also observed that a major difficulty in subpixel estimation arises in the vicinity of discontinuities in the image, *e.g.*, corners. Kiryati and Bruckstein addressed the question of how the digitization process maps a binary shape onto grayscale [29]. Focussing on silhouettes of straight-edged planar shapes they demonstrated the importance of gray levels for accurate binary shape reconstruction. In particular, whereas error-free reconstruction is possible when the gray levels are not quantized and the spatial sampling resolution is sufficiently high, in the presence of quantization and limited spatial sampling resolution a low spatial-resolution gray-level digitizer can potentially introduce less ambiguity than a high spatial-resolution binary scanner. Recently, Kwok and Dong have extended these results, with an emphasis on curved edges [34]; see also [37, 45] for related research. In other work, curves have been viewed as level sets of a surface, *e.g.*, edges as the zero-crossings of the Laplacian of a Gaussian operator [40], isophotes of image intensity for scale [31], for shape from shading [9], *etc.* In

addition, more recently curve evolution applications have used an embedding surface to represent the evolving curve, thus utilizing the additional dimension to regularize computations [47, 15, 12], *e.g.*, shape representation [25, 27, 26], shape from shading [52, 28], image smoothing [11, 2, 24], affine invariant curve evolution [53, 1, 3], shape modeling [10, 39, 61], optical flow [32] and stereo [33]. In the above examples the process of locating the curve, and computing its geometric properties, *e.g.*, orientation and curvature, can benefit from the information contained in the embedding surface in the direction *orthogonal* to the curve.

Second, since slight variations in pixel data can cause large variations in computations of orientation and curvature, smoothing *along* the boundary is also often employed to make the estimation procedure more robust, *e.g.*, spline interpolation [14] with regularization [51]. A critical factor in such methods is the choice of the regularization parameter, which determines the tradeoff between the smoothness of the interpolation curve and its faithfulness to the data. In general it is desirable to have an objective data-driven method for estimating it. The technique of generalized cross-validation [16] has proven useful for this purpose, where the basic idea is to pick that parameter for which a model (*e.g.*, a polynomial spline), estimated from some of the data samples, gives the best least squares prediction of the the remaining samples. This method has been effectively used by Shahraray and Anderson for the estimation of geometric properties of contours. While in smooth regions the method gives excellent geometric estimates, as noted in [57] it produces inaccurate estimates in the presence of discontinuities, because these are smoothed over. Figure 2 (middle column) illustrates this effect this for a simple cubic spline interpolant. In such cases it is necessary to find the discontinuities *prior* to applying the algorithm to the smooth segments lying inbetween.

As argued by Terzopoulos [63], the application of regularization theory encounters difficulties in in a host of visual reconstruction problems, due to the deficiency of global smoothness constraints. What is required is a method for spatially controlling smoothness, so that visual discontinuities can be accomodated. This is accomplished by the introduction of *controlled-continuity stabilizers* where various order splines are blended using weighting functions that are allowed to make jump transitions to zero values. This powerful technique applies in arbitrary dimensions and has been successfuly used for a variety of visual reconstruction problems with discontinuities Terzopoulos:Computation:Visible. The stabilizers are related to geometric modelling primitives used in approximation theory and computer graphics: splines under tension [54, 44] and Beta-splines with varying bias and tension parameters [6].

In recent years, a large number of nonlinear approaches to smoothing shapes and images have been introduced, with the goal of preserving discontinuities, *e.g.*, see the articles in [62] for an overview.

The ideas expressed in this literature, which lead to a scale-space or a family of progressively blurred shapes or images, can also be used in the localization and geometric estimation procedure itself. In general surfaces of objects in the world, and hence their projections in retinal images, do not contain an arbitray number of discontinuities. Rather, these discontinuities are usually finite in number and are isolated, the object boundaries being smooth between them. In this paper we present a geometric interpolation technique that utilizes smoothnes constraints both *across* and *along* the boundaries to locate curves and obtain geometric estimates, with the following properties: 1) the technique does not blur across discontinuities, and 2) it explicitly and accurately places the discontinuities. The resulting geometric estimates are vastly improved and discontinuities, or shocks[1], which are often the salient features, are captured at subpixel resolution. The proposed interpolation technique is useful for: (i) traditional interpolation applications, (ii) the reliable computation of orientation and curvature, and (iii) accurate curve evolution and representation. As a preview of the results, and a comparison against the commonly used bilinear interpolation method, see Figure 1.

The paper is organized as follows. In Section 2 we review the *essentially non-oscillatory* (ENO) interpolation method, originally proposed for the purpose of obtaining accurate geometric estimates in regions *neighboring* discontinuities, in fluid dynamics applications. In the current context, the original technique suffers from two limitations: 1) the geometric estimates are not reliable in intervals *containing* a discontinuity, and 2) the scheme is developed for one dimensional (1D) data and it is not immediately clear how to extend it to two-dimensional (2D) data. In Section 3 we tackle the first problem, and suggest a modification of the ENO scheme to explicitly capture and represent the discontinuities, thus leading to highly accurate geometric estimates. In Section 4, we extend the technique for interpolating 1D functions to one for interpolating a curve. Specifically, we develop a geometric ENO style method to accurately interpolate the curve, and at the same time, explicitly capture and place curve discontinuities. The approach is in two stages. First a set of subpixel sample points are obtained and ordered along the shape's boundary, Section 4.1 and Appendix B. Second, we develop a *geometric* ENO (GENO) interpolation scheme using geometric basis functions consisting of lines, circular arcs, Euler spirals, *etc.*, Section 4.2. Several examples illustrate that discontinuities are preserved and detected with *subpixel accuracy*, and that the process is robust to Euclidean transformations of the underlying data. In Section 5, we analyze the computational errors associated with three methods to compute curvature, and numerically examine their accuracies. We find that numerical estimates from GENO are typically

[1]We informally use the word "shock", or "discontinuity", to refer to either a discontinuity in the function itself, or in its derivatives.

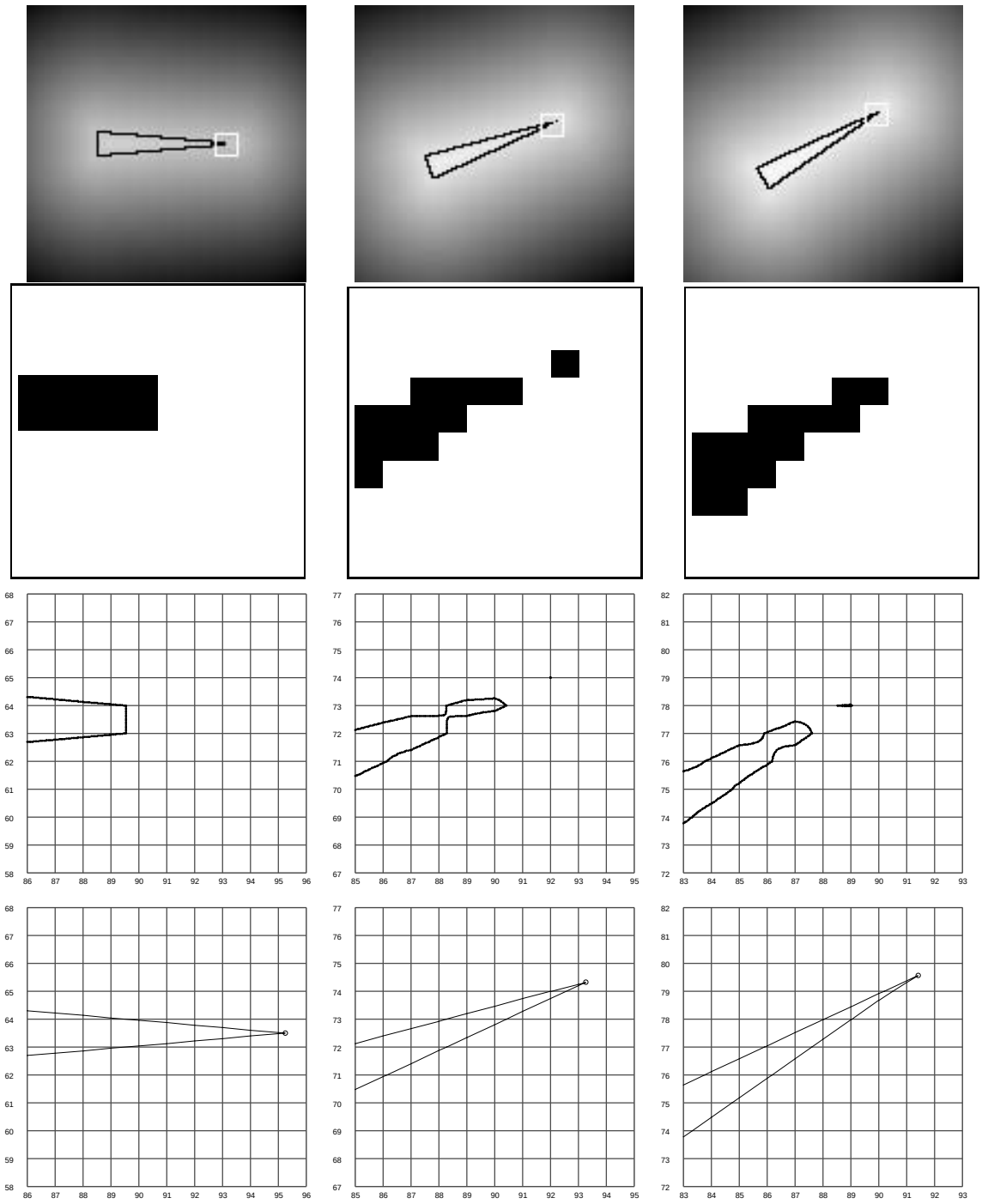

Figure 1: A preview comparison of interpolation techniques. TOP: The original 128x128 images are obtained by sampling an *exact* signed distance function of a triangle with a 10 degree right vertex, oriented at 0, 20, and 30 degrees respectively. The box depicts a 10x10 region under examination. SECOND ROW: The shape's interior at the resolution of the grid. THIRD ROW: Bilinear interpolation. Observe that the corner is truncated or rounded, and gross artifacts are introduced. FOURTH ROW: Geometric ENO interpolation; detected corners are marked with circles. The corners are preserved and accurately placed to subpixel resolution; the distance errors in pixels between detected and true corner locations are, from left to right, 0, 0.079, and 0.455.

better in smooth regions, and far better in regions containing discontinuities. Finally, we illustrate the advantages of GENO with a number of curve evolution examples in Section 6, and conclude with a summary of the results in Section 7.

2 Background: ENO

Splines or polynomials are commonly used to interpolate discrete data in a variety of computer vision and graphics applications. With low computational cost, they provide a continuous representation of the data, from which geometric estimates (orientation, curvature, *etc.*) can be obtained. However, whereas such estimates are generally reliable in regions where the data is smooth, they are prone to error in the vicinity of discontinuities. This follows because such fitting techniques blur over discontinuities by propagating information across them, Figure 2 (middle column)[2]. Recently a class of schemes have been proposed in the numerical analysis literature to address this problem, in application to the numerical solution of conservation laws and the propagation of fronts. These Essentially Non-Oscillatory schemes were introduced by Harten *et al.* [17, 18], and were later made more efficient by Shu and Osher [58]. The basic idea is to select between two contiguous sets of data points for interpolation the one which gives the lower variation, or coefficient of the highest derivative of the interpolation polynomial. At regions neighboring discontinuities, the smoothing is always from the side *not* containing the discontinuity. In this section we review the ENO interpolation algorithm in detail, and demonstrate its use in interpolating discrete data, while *preserving* discontinuities.

The key feature of ENO schemes is an adaptive stencil high order interpolation which tries to avoid shocks or high gradient regions whenever possible. Informally, to illustrate the idea of ENO interpolation, consider building up a polynomial approximation to the data which minimizes oscillations by not crossing over discontinuities, whenever possible. To find the polynomial approximation between the grid points x_j and x_{j+1}, start by interpolating a first-order polynomial between x_j and x_{j+1}. A second-order polynomial is constructed by adding either x_{j-1} or x_{j+2}, whichever produces the smoother polynomial. A third-order polynomial is interpolated by choosing an additional data point, and so on, for higher degree polynomial interpolations. Formally, the ENO procedure tries to obtain uniform high order accuracy by fixing the order of polynomial interpolation but varying the stencil according to

[2]It should be noted that more sophisticated spline interpolation techniques can give improved results, such as not requiring the interpolation curve to pass exactly through each data point [57], or using splines under tension [54, 44] or beta-splines [6]. However, the key point is that when discontinuities are present, their explicit placement must be considered either prior to or hand in hand with the interpolation process.

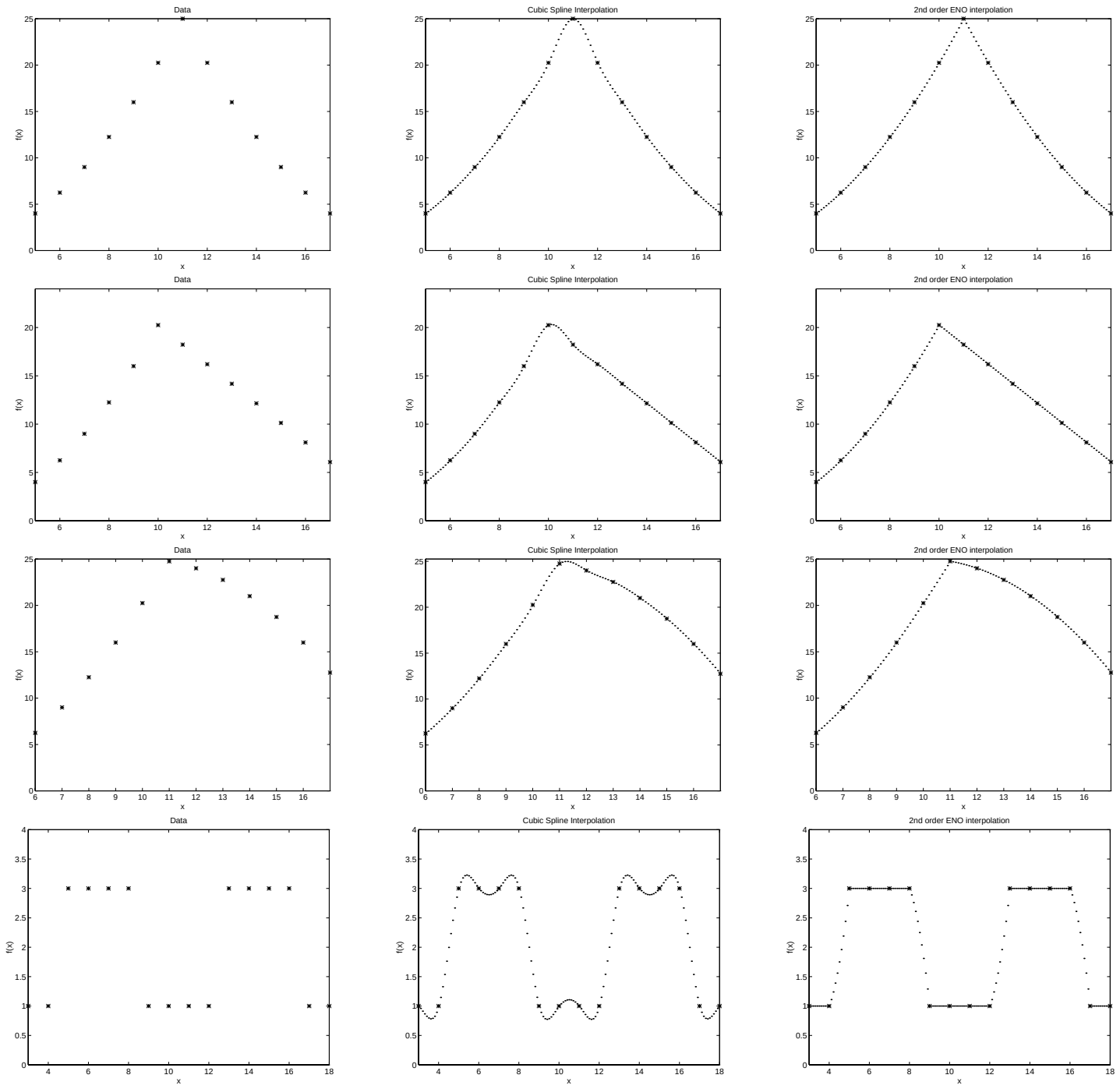

Figure 2: A comparison of cubic spline interpolation with second-order ENO interpolation for various 1D data sets. LEFT COLUMN: The data points are marked with "*"s. MIDDLE COLUMN: The cubic spline interpolation is overlaid on the data. RIGHT COLUMN: The second-order ENO interpolation is overlaid on the data. Observe that whereas ENO interpolation does not blur over singularities or introduce spurious oscillations, this is not the case for cubic spline interpolation.

the local smoothness of the solution. This interpolation procedure has been adapted to the numerical solution of Hamilton-Jacobi equations by Osher and Sethian [47] and Osher and Shu [48]. The procedure is known to give very good numerical results: sharp, non-oscillatory transitions at shocks and high order accuracy in regions where the data is smooth. For a guide to using ENO schemes for evolving PDE's that arise in computer vision see [60]. In the present paper, we wish to use the ENO interpolation algorithm for piecewise smooth interpolation of discrete 1D and 2D data, without blurring across discontinuities. We now summarize the interpolation algorithm.

ENO Interpolation Algorithm: The ENO Interpolation begins with a first degree polynomial $P^{f,1}_{j+1/2}(x)$ interpolating the function $f(x)$ between the two grid points x_j and x_{j+1}. If we stop here, we obtain the first-order monotone approximation. Whenever a higher order is desired, we add just one point to the existing stencil, chosen from the two immediate neighbors by the sizes of the two relevant divided differences, which measure the local smoothness of the function $f(x)$. Given point values $f(x_j)$, $j = 0, \pm 1, \pm 2, \cdots$ of a (usually piecewise smooth) function $f(x)$ at discrete nodes x_j, we associate an r-th degree polynomial $P^{f,r}_{j+1/2}(x)$ with each interval $[x_j, x_{j+1}]$, with the left-most point in the stencil as $x_{k^{(r)}_{min}}$, constructed inductively as described in Table 1.

Figure 2 compares ENO polynomial interpolation with cubic spline interpolation for various 1D data sets. Observe that no oscillations occur between data points, and accuracy is excellent for all intervals except the one which actually contains the discontinuity. Therefore, ENO polynomial fits can lead to vastly improved geometric estimates in the *vicinity* of singularities. However, by construction the ENO interpolation is smooth in the interval containing the singularity and does not capture the singularity itself, Figure 4 (middle column). Hence, the geometric estimates are still error prone in such an interval, and there are discontinuities at the interval's end points. This is the first limitation of the ENO approach, and suggests the following modification: in addition to the two choices of smoothing from the left and smoothing from the right, we might consider a third option, that of "breaking" the curve in the interval under consideration, and relieving the two end point discontinuities. This is in essence Harten's "sub-cell resolution" idea [17]; the difference here is that geometric reasoning rather than "conservation" is used to determine the sub-cell shock location. In the following section we develop this idea and illustrate its use.

1. Find the first-order polynomial interpolation $P^{f,1}_{j+1/2}(x)$ and initialize the first stencil point for the point j, $k^{(1)}_{min}$:

$$\begin{aligned} P^{f,1}_{j+1/2}(x) &= f[x_j] + f[x_j, x_{j+1}](x - x_j) \\ k^{(1)}_{min} &= j \end{aligned}$$

2. If $k^{(l-1)}_{min}$ and $P^{f,l-1}_{j+1/2}(x)$ are both defined, then

$$P^{f,l}_{j+1/2}(x) = P^{f,l-1}_{j+1/2}(x) + c^{(l)} \prod_{i=k^{(l-1)}_{min}}^{k^{(l-1)}_{min}+l-1} (x - x_i),$$

where

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

$$k^{(l)}_{min} = \begin{cases} k^{(l-1)}_{min} - 1 & \text{if } |a^{(l)}| \geq |b^{(l)}| \\ k^{(l-1)}_{min} & \text{otherwise} \end{cases}$$

and finally,

$$\begin{aligned} a^{(l)} &= f[x_{k^{(l-1)}_{min}}, \cdots, x_{k^{(l-1)}_{min}+l}] \\ b^{(l)} &= f[x_{k^{(l-1)}_{min}-1}, \cdots, x_{k^{(l-1)}_{min}+l-1}] \end{aligned}$$

Table 1: The ENO interpolation algorithm. In the procedure $f[\cdot, \cdots, \cdot]$ are the standard Newton divided differences; see Appendix A for a review of these.

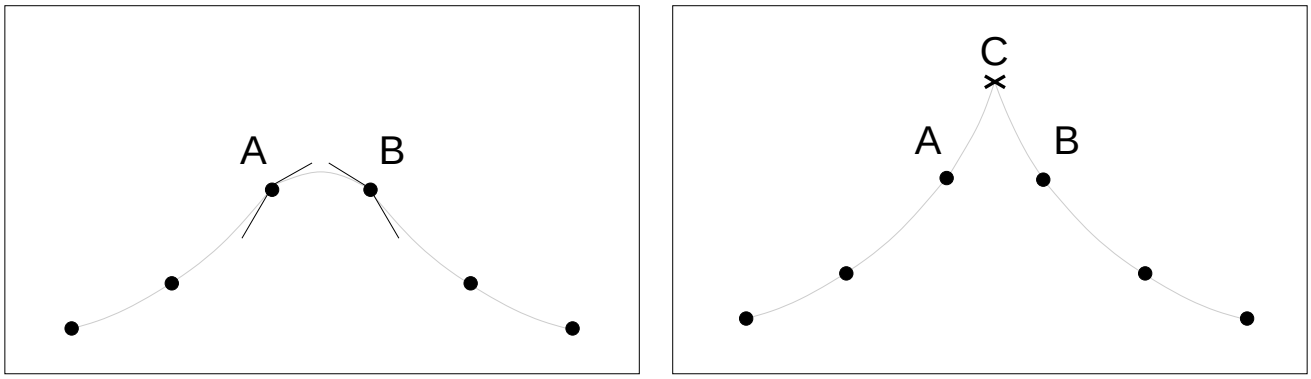

Figure 3: LEFT: The interval AB supports two discontinuities, one at A and one at B. RIGHT: As an alternative to interpolation, one may "relieve" these two shock points by extrapolating the neighboring interpolations and introducing a single singularity at C.

3 Shock Placing ENO

As illustrated in Section 2, while the ENO interpolation algorithm effectively deals with areas where the underlying data is smooth, and with areas neighboring discontinuities, it does not explicitly represent and interpolate singularities. Hence, the geometric estimates of an interval containing a singularity are still prone to error. In seeking a solution to this problem, two questions are posed: what constraints can be employed to signal the presence of a singularity, and, how should the singularity be placed? In the numerical simulation of a conservation law [17], Harten proposed the use of "conservation" to guide the placement of a shock at subcell resolution. In this section, we follow Harten's idea to explicitly place shocks, but rather than employ the conservation constraint, we suggest the use of geometric constraints for the applications we have in mind.

Observe that the placement of shocks can be signaled by the ENO interpolation algorithm: in smooth regions, ENO interpolation in one interval continuously follows to interpolations in neighboring intervals, with little or no change in orientation or curvature. However, in intervals appearing to contain discontinuities, neighboring interpolations lead to discontinuous estimates of orientation and curvature at the boundaries. To illustrate, consider Figure 3 (left), where the orientation and curvature limits at point A are different when approached from the left than when approached from the right. Therefore, at its boundaries, the interval AB typically supports two discontinuities, one on the left and one on the right. As an alternative to interpolation, one may "relieve" these two discontinuities (shock points) and introduce a single shock C by smoothly extrapolating neighboring interpolations, Figure 3 (right). Such a choice is guided by minimizing curvature and orientation variations. To formalize these ideas

we make two assumptions explicit:

Assumption 1 *The underlying curve is piecewise smooth with finite total curvature.*

In other words, the curve does not oscillate between data points, and is smooth on either side of a singularity. This smoothness can be used to guide the placement of singularities, provided that the following is also true:

Assumption 2 *The curve has a finite number of singularities, with the singularities being at least one interval apart.*

With the above two assumptions we are now in a position to develop a strategy for shock placement. First, recall that in inductively constructing the polynomial fit of degree n, the ENO interpolation algorithm selects between two data points the one that results in a lower highest derivative. Therefore, it is intuitively clear that in an interval that contains a singularity but neighbors smooth intervals on each side, the rightmost point will not contribute to the ENO interpolation in the interval to the left, and similarly the leftmost point will not contribute to the ENO interpolation in the interval to the right. This property can be used, as a first step, to "flag" intervals where singularities may occur. Second, the constraint of piecewise smoothness between singularities requires that the placement of a shock must not cause orientation or curvature discontinuities at either endpoint of the interval in which it is placed. This can be enforced by extrapolating the interpolations from the neighboring intervals, and placing the singularity at the point of their intersection. In such a strategy, the interpolating curve on one side of the shock is obtained solely from the neighboring interval on that side. This is logical since the very idea of placing the shock is to institute a break in the curve. Examples illustrating this strategy are shown in Figure 4 (right column). The resulting shocks, which are marked by crosses, appear to be intuitive. Observe also that for smooth data, no shocks are placed, and the interpolation agrees with the unmodified ENO interpolation, Figure 4 (top).

In summary, when the ENO interpolation gives rise to discontinuous measures of orientation or curvature, the option of placing a single shock is considered. We should mention that an exception arises when the boundary continuations do not intersect. In such cases, since placing a single shock is not an option *and* interpolation signals two discontinuities, we make *both* discontinuities *explicit*, Figure 5. Note that while this rarely occurs in our applications, we have included this possibility in our algorithm for completeness. Having modified the ENO scheme to explicitly place shocks, we are now in a position to extend our approach to one for 2D data.

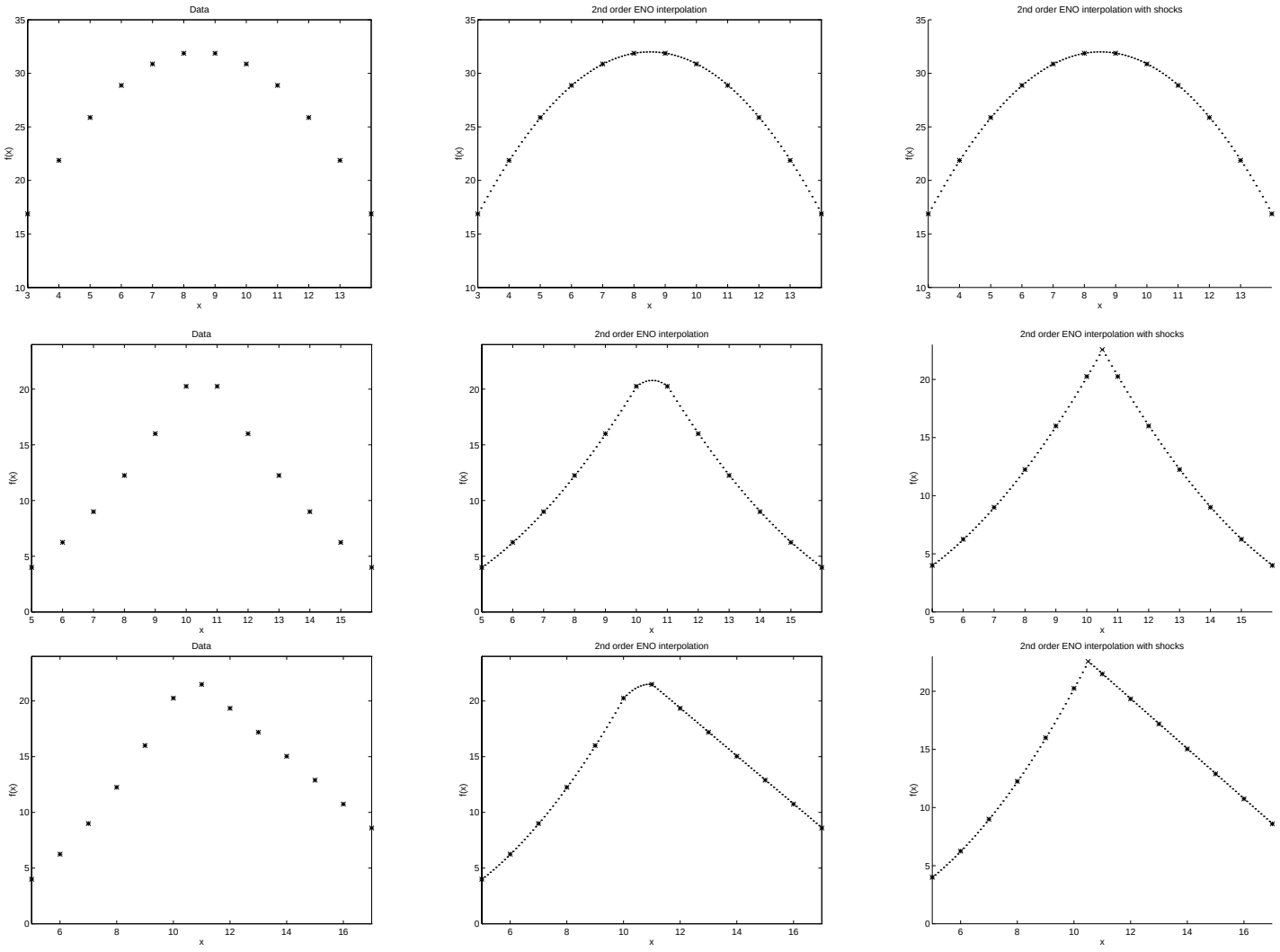

Figure 4: ENO interpolation versus ENO interpolation with shock placement. LEFT COLUMN: The data points are marked with "*"s. Since the discontinuities no longer coincide with nodes, as they did in Figure 2, subcell shock placement is required to capture them. MIDDLE COLUMN: The second-order ENO interpolation is overlaid on the data. Observe that singularities which lie within an interval are not captured. RIGHT COLUMN: In a modified ENO algorithm, using evidence from neighboring intervals the option of "breaking" the curve and placing a singularity is considered. The detected shocks are marked by crosses. Note that when the data is smooth (TOP ROW), no shocks are placed and the two interpolations are equivalent. However, when there are significant orientation and curvature changes at the end points of an interval, a shock is placed (MIDDLE AND BOTTOM ROWS).

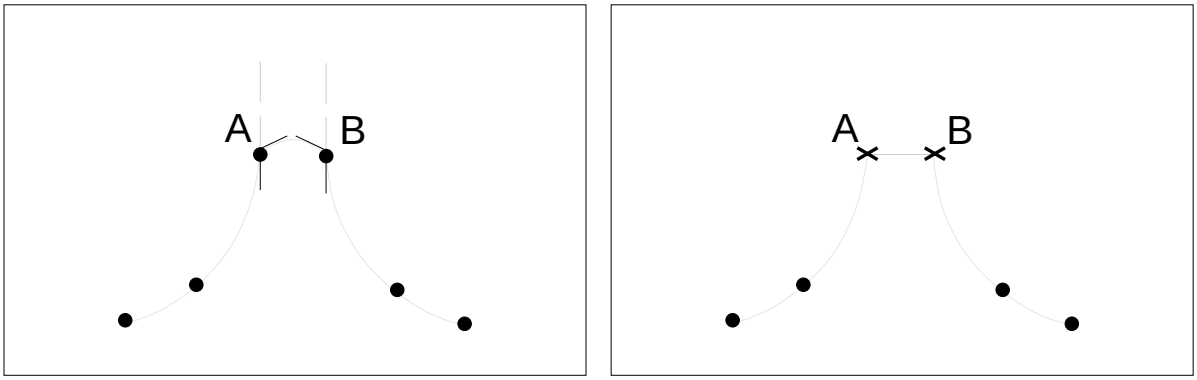

Figure 5: LEFT: The interval AB supports two discontinuities, one at A and one at B, however the continuations of the neighboring intervals do not intersect. RIGHT: Since placing a single shock is not an option, both discontinuities are made explicit.

4 Geometric ENO (GENO) Interpolation

In the previous section, 1D data was interpolated by using sampled values at a series of ordered points. It would appear then that these results should also apply to 2D data, since a shape's boundary can be locally represented as a function in some local coordinate system. Alternatively, the x and y coordinates of the boundary points can each be interpreted as functions of the arc length parameter $\tilde{s}$, each of which can be interpolated as a 1D function. Unfortunately, however, both approaches face significant difficulties. First, in both cases, the boundary points are not ordered. Such ordering is itself the result of tracing the boundary. Second, the exact measure of arc length $\tilde{s}$, which is necessary to serve as the independent variable for interpolation, is not known prior to interpolation; rather it is a result of it. Thus, we are faced with somewhat of a "chicken-and-egg" problem: order and arc-length can be obtained from accurate subpixel interpolation, but subpixel interpolation requires order and arc length![3] The solution, we propose, revolves around a two-stage interpolation scheme to: (i) obtain unordered sample points of the boundary with subpixel accuracy, and then order them, and (ii) interpolate between these points in 2D, using a *geometric* generalization of the 1D ENO scheme. We now describe each stage in turn.

4.1 First Stage: Subpixel Samples

The goal of the first stage is to obtain samples of the trace of the boundary with subpixel accuracy, and later to order these points. Subpixel sample points can be obtained by 1D ENO interpolation of the

[3] See [49] for a similar dilemma faced in the problem of curve detection from images.

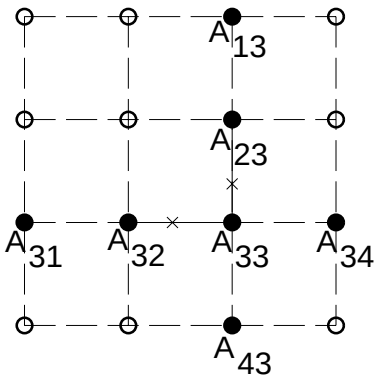

Figure 6: The 1D shock placing ENO interpolation is performed along gridlines. The zeros of the interpolation polynomials provide subpixel sample points, *e.g.*, the crosses in the intervals $A_{32}A_{33}$ and $A_{23}A_{33}$.

surface in the direction of any line passing through 2D grid points: for each interval along such a line, we apply the shock-placing ENO algorithm, Section 3, and store the coefficients of the interpolation polynomial. The sampled boundary points correspond to the zeros of these interpolation polynomials. Two cases must be handled. First, if no shock is placed there is a single interpolation polynomial to consider. Its zeros are found and those which lie *within* the interval under consideration are stored. Second, when a shock is placed, there are two polynomials to consider, *i.e.*, one from the left and one from the right. The zeros of each polynomial are found and those that lie *within* the interval under consideration *and* on the appropriate side of the shock are stored.

Now, since there are many lines that pass through 2D grid points, a question arises as to which lines to use. Not surprisingly, the most convenient choices are grid lines, or lines that pass through (normal resolution) grid points in both the horizontal and vertical directions. Of course, any other lines such as diagonal lines through grid points will also serve for our purpose, but in practice we have found that grid lines alone give excellent results. We note that underlying the 1D interpolation is the assumption that the surface is piecewise smooth along these lines. Hence, as we stated in Section 1, it is this step that implicitly takes advantage of geometric continuity in the direction *orthogonal* to level curves.

To illustrate, let us consider shock placing ENO interpolation using second-order polynomials, Figure 6. The interpolation in the interval $A_{32}A_{33}$ will use the points A_{32}, A_{33} and either A_{31} or A_{34}, depending upon which leads to the smaller coefficient for the second-order term. Similarly, The interpolation in the interval $A_{23}A_{33}$ will use the points A_{23}, A_{33} and either A_{13} or A_{43}. The detected, valid (lying within the interval under consideration) zeros will be stored for further processing, as indicated by the crosses in Figure 6. It is important to note that the subpixel boundary points are found *without explicitly representing* a high resolution sub-grid between any two neighboring grid points.

The set of boundary points obtained through the above procedure must now be ordered such that they lie consecutively along the boundary of the shape. This is done by generalizing a standard contour tracing algorithm, as proposed in Appendix B. It is important to note that while in our presentation we have separated the process of sampling boundary points from that of ordering them, our algorithm combines these into a single step, thus achieving significant computational savings. To elaborate, note that the contour tracer builds the ordered list of points sequentially, each time appending a new boundary point to the end of an existing list of points. The search for which boundary point to append is local and depends *only* on the last two boundary points stored in the list. Therefore, it is possible to merge the procedure for interpolating boundary points and the tracing process into one. In fact, this is what is done in practice; the ENO interpolation is only performed in intervals where the contour tracer searches for boundary points.

4.2 Second Stage: Geometric ENO interpolation

Having obtained a set of ordered subpixel sample points along the shape's boundary, our next task is to interpolate between them, while simultaneously placing singularities when required. Whereas it would appear that the shock placing ENO algorithm developed in Section 3 would apply directly to the ordered sample points, there are in fact a number of limitations. In the following we examine these limitations and propose a solution where, in place of extrinsic polynomials, a set of *geometric* interpolants are used.

Whereas the shape's boundary can be described locally by a function in some x-y coordinate system, *e.g.*, the *Frenet Frame*, this is only true in an infinitesimal sense. Since the data is discrete, in an interval where discontinuities or high curvature points are present, the function can become multivalued. In addition, the *Frenet Frame* is itself not computable prior to obtaining the curve, and in fact, if a discontinuity happens to coincide with a sample point, the tangent itself is not well-defined. Even if some coordinate system can be found in which the data is locally single-valued, the results are not guaranteed to be invariant to rotations of the data. Thus, using a local intrinsic coordinate frame appears to be infeasible.

Alternatively, the original (extrinsic) x and y coordinates of the boundary points may be expressed as separate functions of the arc length parameter $\tilde{s}$, $x(\tilde{s})$ and $y(\tilde{s})$, respectively. Unfortunately, however, as mentioned earlier, the measure of arc length is not available prior to interpolation, although it is needed to obtain the fit! One might proceed by assuming, for example, that the variation of $\tilde{s}$ corresponds with the Euclidean distance between successive sample points. However, the independent interpolation of $x(\tilde{s})$ and $y(\tilde{s})$ gives rise to a further difficulty: the selection of data points for ENO interpolation

Order	*Polynomial*	*Geometric*	
		Orientation	Curvature
First	$f(x) = a_1 x + a_0$ (linear)	$\theta(\tilde{s}) = \kappa_0$	$\kappa(\tilde{s}) = 0$ (line)
Second	$f(x) = a_2 x^2 + a_1 x + a_0$ (quadratic)	$\theta(\tilde{s}) = \kappa_1 \tilde{s} + \kappa_0$	$\kappa(\tilde{s}) = \kappa_1$ (circular arc)
Third	$f(x) = a_3 x^3 + a_2 x^2 + a_1 x + a_0$ (cubic)	$\theta(\tilde{s}) = \frac{\kappa_2}{2} \tilde{s}^2 + \kappa_1 \tilde{s} + \kappa_0$	$\kappa(\tilde{s}) = \kappa_2 \tilde{s} + \kappa_1$ (Euler spiral)

Table 2: A comparison between polynomial and geometric basis functions for interpolation. Note that geometric fits are *intrinsic*, or not dependent on the choice of a coordinate system.

may not coincide for $x(\tilde{s})$ and $y(\tilde{s})$. In other words, there is no guarantee that the choice between left and right points that results in a lower variation for the interpolation of $x(\tilde{s})$, will also result in a lower variation for the interpolation of $y(\tilde{s})$. In fact, a single joint criterion must be devised for the selection of data points to couple the x and y interpolations. An additional difficulty has to do with the *extrinsic* nature of the approach: as before, the results can vary with rotations of the data set. Therefore, an *intrinsic* scheme is required, as described next.

A solution to the above problems, we propose, rests in replacing extrinsic polynomials with a set of *geometric* basis "functions" for interpolation. The essential idea is to use fits that are not dependent on the choice of a coordinate system, but rather depend only on the geometry of the underlying curve. Recall that in first-order ENO interpolation using polynomials, all derivatives of the interpolant, with the exception of the first derivative, are set to zero. This gives first-order interpolation in linear form, $f(x) = a_1 x + a_0$. With second-order ENO, all derivatives except the first two are set to zero, and so on for higher order interpolation resulting in $f(x) = a_2 x^2 + a_1 x + a_0$, *etc.* What are the analogs of algebraic derivatives in a geometric sense? Here, the variables of interest are orientation and its derivatives, namely, curvature, curvature variation, *etc.* In analogy to the algebraic case, the basis functions can be found by setting various order derivatives to zero. For example, $\theta_{\tilde{s}} = 0$ yields a straight line, $\theta_{\tilde{s}\tilde{s}} = 0$ gives a circle, and $\theta_{\tilde{s}\tilde{s}\tilde{s}} = 0$ results in an Euler spiral [4]. These constitute our geometric interpolation bases, with an associated sense of order, Table 2.

The *Geometric Essentially Non Oscillatory* (GENO) interpolation begins, therefore, with a straight

[4]The Euler spiral is a curve with linear curvature variation, studied by Euler in 1781, which has found practical use in applications such as laying railroad tracks [35].

line, interpolating the boundary $C(s) = (x(s), y(s))$ between two consecutive sample points $C(s_n)$ and $C(s_{n+1})$. If we stopped here, we would have the first-order (line) interpolation. A second-order (circular arc) interpolation is constructed by adding either of the points $C(s_{n-1})$ or $C(s_{n+2})$, whichever provides the circular arc fit with lower curvature. A third-order (Euler spiral) interpolation is constructed by adding yet another sample point, choosing the one which results in a lower curvature variation; in general, for higher order interpolations, the sample point added is the one which provides the lower coefficient for the highest order term, Table 2 [5]. In this paper, we have limited ourselves to the use of second order (circular arc) interpolations, primarily because geometric fits using lines and circular arcs are straightforward to compute, and the accuracy of the results is generally excellent. In cases where curvature may in fact vary greatly within an interval, higher order interpolations will lead to improved accuracy.

The above geometric interpolation gives excellent accuracy both in smooth regions and in the vicinity of singularities. However, since by construction GENO interpolation is smooth in all intervals, including those containing singularities, it cannot capture the singularities themselves, thus leading to poor geometric estimates in such intervals. Fortunately, however, the same shock placing algorithm designed to overcome this limitation for the 1D case, Section 3, also applies here. The difference is that geometric basis functions are used instead of polynomials for interpolation. To repeat this argument for 2D data, in addition to the options of smoothing from one side or the other, we introduce the option of breaking the curve in the interval under consideration, and placing a discontinuity when required. Under assumptions 1 and 2, when the GENO interpolation in an interval AB leads to discontinuous estimates of orientation and curvature at is boundaries, Figure 3, the two discontinuities are relieved by smoothly continuing the neighboring interpolations and introducing a single shock (C) at the point of intersection. Finally, we note that underlying this second stage is the assumption that the level curves of the surface are piecewise smooth. Hence, as mentioned in Section 1, it is this step that takes advantage of geometric continuity in the direction *along* level curves.

In Figures 7 and 8 we illustrate GENO interpoplation on a variety of shapes, comparing it with the commonly used bilinear interpolation technique. Observe that: 1) the GENO fits are intuitive both in smooth regions, and in the vicinity of singularities, 2) in contrast to the bilinear interpolation,

[5] The set of data points used to obtain the fit of order n for an interval between two consecutive data points is a superset of that used to obtain fits of order less than n. However, the method does not predict the order of polynomial that is 'optimal' for a particular interval in the sense of a simplicity of model versus goodness of fit tradeoff. For this purpose, one could readily use a minimum description length (MDL) type approach [36] as a post-processing step. For example, with a second-order GENO fit using circular arcs, circular arc interpolants of large radius may later be replaced by straight lines.

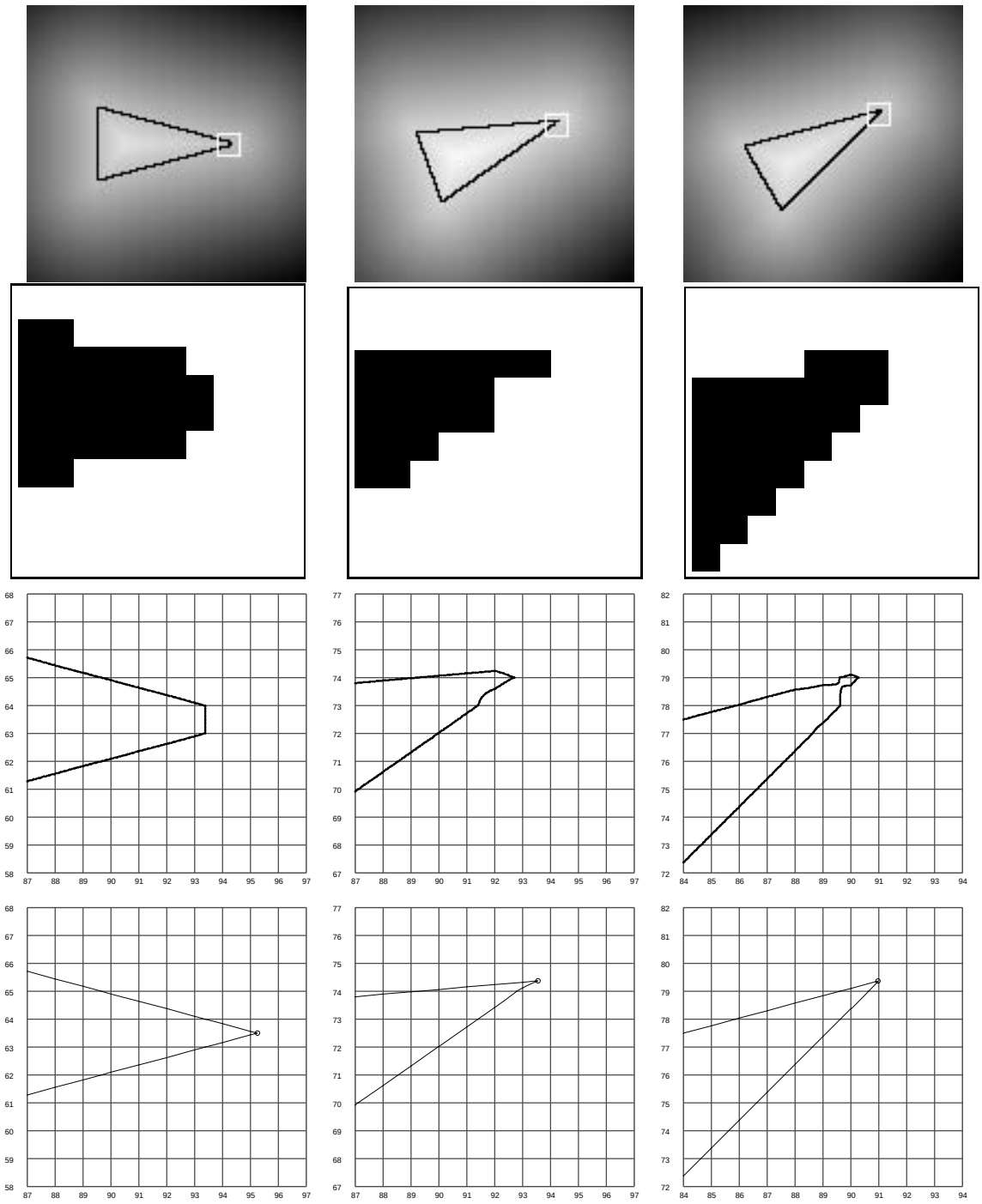

Figure 7: A comparison of interpolation techniques. TOP: The original 128x128 images are obtained by sampling an *exact* signed distance function of a triangle with a 30 degree right vertex, oriented at 0, 20, and 30 degrees respectively. The box depicts a 10x10 region under examination. SECOND ROW: The shape's interior at the resolution of the grid. THIRD ROW: Bilinear interpolation. Observe that the the corner is truncated or rounded, and gross artifacts are introduced. FOURTH ROW: Geometric ENO interpolation; detected corners are marked with circles. The corners are preserved and accurately placed to subpixel resolution; the distance errors in pixels between detected and true corner locations are, from left to right, 4.17E-8, 0.216, and 0.028.

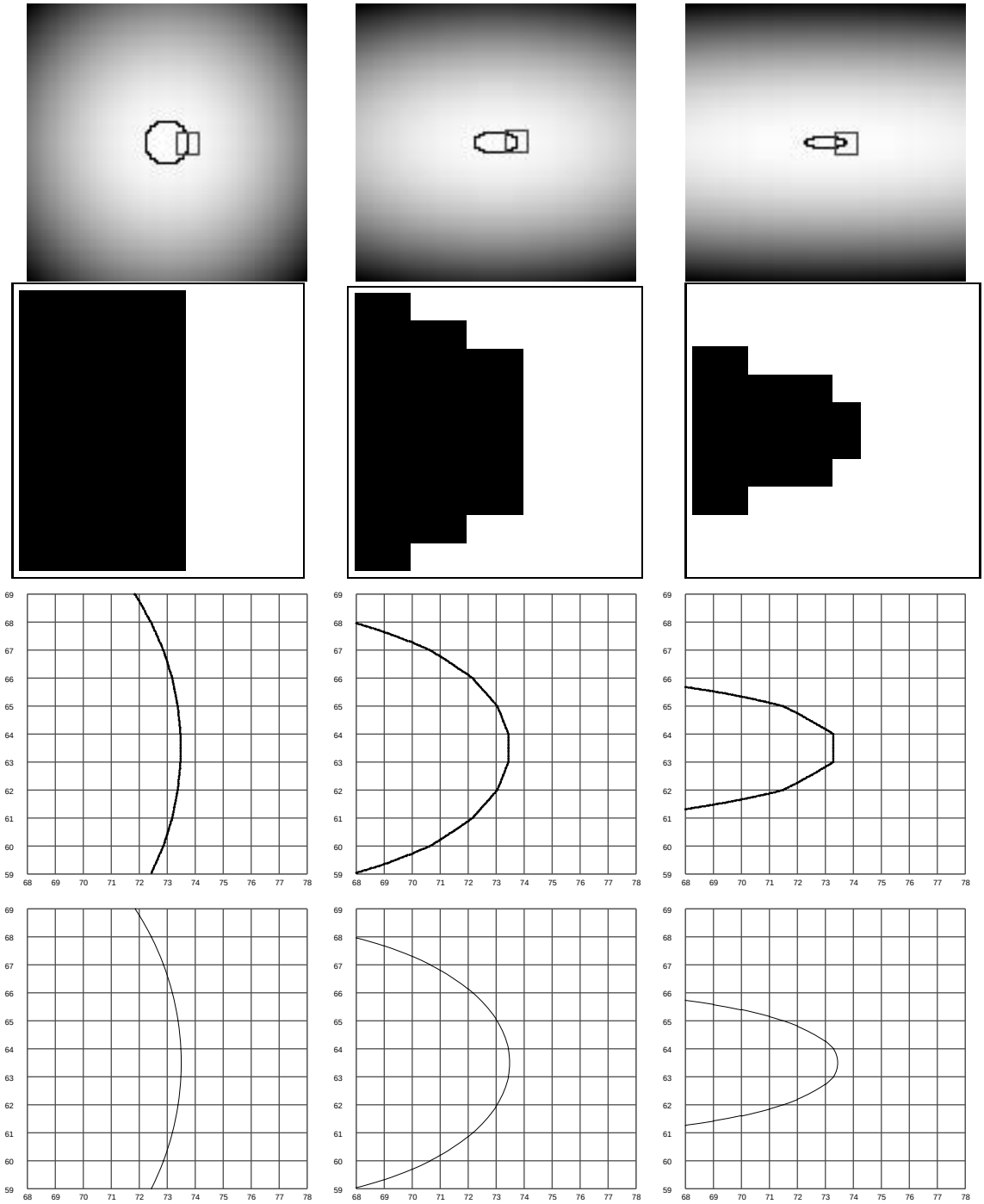

Figure 8: Bilinear versus ENO interpolation for 3 ellipses. TOP: The original images are 128x128 surfaces for an ellipse with major/minor axes equal to 20/20, 20/10, and 20/5 respectively. The box depicts a 10x10 region under examination. SECOND ROW: The shape's interior at the resolution of the grid. THIRD ROW: Bilinear interpolation. FOURTH ROW: Geometric ENO interpolation.

GENO explicity captures the singularities, and places them accurately, and 3) in contrast to the bilinear interpolation, GENO is robust to rotations and does not truncate corners or introduce artifacts in the vicinity of singularities. It would seem, therefore, that GENO interpolation should lead to improved geometric estimates in smooth regions, in the vicinity of discontinuities, and at discontinuities. In the following section we compare the accuracy of geometric estimates using GENO with estimates obtained from other methods.

5 Geometric Estimates

In this section we show how GENO interpolation allows for highly accurate geometric estimates. Whereas we illustrate its use for the computation of curvature, it provides accurate estimates of other geometric quantities as well, *e.g.*, orientation. Specifically, we compare the accuracy of three different methods for obtaining curvature estimates, namely, curvature computed from: 1) the derivatives of the embedding surface ϕ, 2) bilinear interpolation[6], and 3) GENO interpolation. We first present each method of computation, and then provide a theoretical analysis of the expected errors in each case, followed by numerical simulations on several shapes.

We begin by showing how curvature can be computed using each method. First, it can be shown that the curvature of a level set of ϕ, passing through a grid point (x, y), may be computed as:

$$\kappa = \frac{(\phi_{xx}\phi_y^2 - 2\phi_{xy}\phi_x\phi_y + \phi_{yy}\phi_x^2)}{(\phi_x^2 + \phi_y^2)^{3/2}}. \tag{1}$$

This method was used extensively by Sethian and Osher in numerical simulations of front propagation [47, 55]. Second, in any cell that contains the curve, bilinear interpolation takes the form:

$$y = -\frac{ax + b}{cx + d} \tag{2}$$

where a, b, c, and d are constants. Thus, curvature κ can be computed using the standard formula:

$$\kappa = \frac{y'' x' - x'' y'}{(x'^2 + y'^2)^{3/2}}. \tag{3}$$

Third, the estimation of curvature from GENO is directly obtained from the explicit representation of curvature using geometric interpolants, *e.g.*, circular arcs.

We now provide a brief analysis of errors in the computation of curvature for each of the above methods. We will use the notation $O(1)$ to denote an order one quantity, which is independent of the

[6]It should be noted that whereas we use bilinear interpolation in our comparisons, the results are generalizable since for any higher order polynomial interpolation, a higher order GENO interpolation may be used as well.

grid size Δ (taken for simplicity to be $\Delta = \Delta x = \Delta y$), but is in general dependent upon the function ϕ one approximates. The notation $O(\Delta^r)$ will denote an quantity which decays with the grid size Δ in r-th power. For example, if the error is $O(\Delta^2)$ and Δ is decreased by half (i.e. the number of grid points is increased by a factor of two), then the error should decrease by a factor of four. Our analysis examines two factors, namely, whether the curve lies in a smooth or discontinuous region of the embedding surface, and whether the curve has been localized exactly or approximately. First, assume that we are in a smooth region of ϕ, and that the curve has been located exactly. The expected errors are then as follows.

1. **Derivatives of ϕ:** If the values of the function ϕ are known at the grid points, either exactly, or at least to second order accuracy $O(\Delta^2)$, and each derivative in the analytic formula for curvature (1) is approximated by a second order divided difference, then κ is accurate up to $O(\Delta^2) \leq \rho\Delta^2$. The proportionality constant ρ depends directly on the magnitude of the higher derivatives of ϕ, *e.g.*, $\phi_{xxx}, \phi_{xxxx}, \phi_{xxyy}$, *etc.* This constant may be large or small depending upon the particular function ϕ; however it is fixed and is independent of the grid size Δ. Thus, to reduce the error (say by a factor of four), one only has to increase (double) the number of grid points in each direction. To see an example of how these error bounds are obtained, consider approximating ϕ_{xx} at the grid point $x = x_i$ by the central difference formula $\frac{\phi_{i+1}-2\phi_i+\phi_{i-1}}{\Delta^2}$. Simple Taylor expansion reveals that the error is given by
$$\phi_{xx}|_{x=x_i} = \frac{\phi_{i+1} - 2\phi_i + \phi_{i-1}}{\Delta^2} = \frac{1}{12}\phi^{(4)}(\xi)\Delta^2,$$
where ξ is some point between x_{i-1} and x_{i+1}. Clearly this error is of the size $O(\Delta^2)$, with the proportionality constant depending upon the fourth derivative of ϕ.

2. **Bilinear Interpolation:** When bilinear interpolation is used to compute curvature directly, the accuracy is at most first order, $O(\Delta)$. This is because the bilinear function (2) does not contain enough second order terms to approximate curvature; only the cross term xy exists. However, it is important to mention that it is always possible to use a wider stencil to construct a higher order polynomial interpolation, leading to more accurate curvature estimates, but *only in smooth regions* of ϕ.

3. **GENO:** When GENO interpolation is used to compute curvature and other geometric quantities, the accuracy is again the same as that of the interpolation itself, *e.g.*, of $O(\Delta^2)$ for the second-order interpolation. Again, higher order GENO interpolation will lead to higher order accuracy in the geometric estimates.

From the above it appears that in smooth regions of the function ϕ, and when the trace of the curve is exactly known, curvature from GENO is comparable to curvature from derivatives of ϕ, with $O(\Delta^2)$ errors. On the other hand, curvature computed directly from bilinear interpolation leads to errors of size at least $O(\Delta)$. Again, it is important to point out that the crucial factors are: i) whether the value of ϕ is accurate, *i.e.*, the curve has been accurately located, and ii) whether the interpolation region is a smooth region of ϕ. If both the above are true, then higher order estimates for curvature or other geometric quantities can be obtained by using a wider stencil interpolation. This is true for all the three cases above. The advantage of GENO is that it gives a much more accurate location of the curve, and it allows one to use a locally smooth region to perform the interpolation when close to or at a discontinuity in the curve.

Now, if we relax the assumption that the location of the curve is exactly known, *e.g.*, in method 1 typically one uses κ at grid points to estimate κ on the curve, then the additional error is first order, $O(\Delta)$. It is clear, therefore, that in order to obtain full second order accuracy, the location of the curve must be known either exactly or at least to second order accuracy, $O(\Delta^2)$. In other words, accurate subpixel placement of the curve is the crucial factor in preferring GENO over the other methods in smooth regions of ϕ.

Finally, if we also relax the assumption that we are in a smooth region of ϕ, the accuracy of methods 1 and 2 is entirely lost with central difference approximations. This is because the above analysis of errors is valid only when the interpolations have stencils *completely inside the smooth region of* ϕ. In regions containing discontinuities of ϕ (and hence singularities of the curve), one can expect errors of $O(1)$. Therefore, for accurate curvature computations in such regions, it is important that the discontinuities be explicitly placed.

We now present numerical experiments which examine curvature computations. In these numerical simulations, the surface in which the shape is embedded is exactly known, *e.g.*, as obtained by sampling values of an analytic signed distance function at grid points. The analytic representation allows for an *exact* computation of curvature at *any* point, which can then be used as a standard to measure numerical estimates against. For this purpose, we use shapes with corners having different angles (triangles), as well as smooth shapes with varying curvature (ellipses).

First, we compare curvature estimates from the above methods, on three triangle shapes, representing three different vertex angles, with the orientation of each triangle being varied over three values, 0, 20 and 30 degrees. Note that the expected curvature at each boundary point is 0, except at corners where it is infinite. For each of the three methods, we tabulate the maximum value of the error, namely,

Shape	Expected $\lvert\kappa\rvert$	Level Set $\lvert\kappa\rvert$ (max)	Bilinear $\lvert\kappa\rvert$ (max)	GENO $\lvert\kappa\rvert$ (max)
triangle-10, 0 degrees	0.0	3.38E-2	4.89E-1	7.34E-8
triangle-10, 20 degrees	0.0	1.96E-1	1.26E+1	8.96E-3
triangle-10, 30 degrees	0.0	2.44E-1	2.66E+0	3.33E-2
triangle-30, 0 degrees	0.0	9.22E-8	5.73E-1	4.94E-3
triangle-30, 20 degrees	0.0	1.50E-1	1.42E+0	1.96E-1
triangle-30, 30 degrees	0.0	2.13E-1	1.02E+1	1.93E-2
triangle-90, 0 degrees	0.0	1.22E-1	1.19E+0	4.21E-8
triangle-90, 20 degrees	0.0	1.90E-1	1.67E+0	1.09E-1
triangle-90, 30 degrees	0.0	2.17E-1	1.36E+0	1.11E-3

Table 3: A numerical comparison of curvature estimates for a 10 degree, 30 degree and 90 degree right vertex triangle, each at 3 different orientations. For each shape, we compute $|\kappa|_{max}$ over all boundary points, using three different methods. This provides a numerical estimate of the upper bound on the error in the estimation of curvature, since the expected curvature is 0.

$|\kappa|$, found over all non-corner boundary points, Table 3. This provides a numerical estimate of the maximum error in curvature estimation, for non-corner boundary points. Observe that in almost all cases, $|\kappa|_{max}$ using GENO with circular arcs is significantly less than $|\kappa|_{max}$ using the other two methods. While the performance of GENO is clearly better than the other methods in the smooth regions of the triangles, note that at *corners* the utility of GENO is indispensable: both the level set and bilinear methods completely miss the infinite curvature associated with corners. The GENO interpolation, on the other hand, explicitly places corners, and thus represents their infinite curvature.

We now examine curvature estimates for smooth shapes with varying curvature, using ellipses having different eccentricities. Specifically, we consider the three ellipse shapes in Figure 8, with major to minor axis ratios 20/20, 20/10, and 20/5, respectively. For each ellipse, we sample points lying at $\pi/2, 3\pi/8, \pi/4, \pi/8$ and 0 radians, Figure 9, and compare the numerical estimate of curvature, obtained using each of the above methods, with the expected curvature, Table 4. As expected, the bilinear interpolation method provides the poorest curvature estimates, and as mentioned earlier these could be substantially improved, at least in low curvature regions, by using higher order polynomials for interpolation. On the other hand, the GENO curvature estimates are typically the most accurate when compared with the bilinear and level set methods. In intervals where the GENO estimates lead to

Shape	Expected $\lvert\kappa\rvert$	Level Set $\lvert\kappa\rvert$	Bilinear $\lvert\kappa\rvert$	GENO (Circular Arcs) $\lvert\kappa\rvert$
ellipse-20-20				
P1: $\pi/2$	1.0E-1	9.492E-2, Error: 5.1%	0.0, Error: 100%	1.000E-1, Error: 0%
P2: $3\pi/8$	1.0E-1	9.871E-2, Error: 1.3%	2.894E-2, Error: 71.1%	9.991E-2, Error 0.1%
P3: $\pi/4$	1.0E-1	9.444E-2, Error: 5.6%	5.067E-2, Error: 49.3%	9.923E-2, Error: 0.8%
P4: $\pi/8$	1.0E-1	9.871E-2, Error: 1.3%	2.896E-2, Error: 71.0%	9.991E-2, Error: 0.1%
P5: 0	1.0E-1	9.492E-2, Error: 5.1%	0.0, Error: 100%	1.0E-1, Error: 0%
ellipse-20-10				
P1: $\pi/2$	5.000E-2	5.551E-2, Error: 11.0%	0.0, Error: 100%	5.035E-2, Error: 0.7%
P2: $3\pi/8$	5.953E-2	6.087E-2, Error: 2.3%	1.444E-2, Error: 75.7%	5.722E-2, Error: 3.9%
P3: $\pi/4$	1.012E-1	1.062E-1, Error: 4.9%	4.368E-2, Error: 56.8%	8.338E-2, Error: 17.6%
P4: $\pi/8$	2.316E-1	2.800E-1, Error: 20.9%	6.166E-2, Error: 73.4%	2.106E-1, Error: 9.1%
P5: 0	4.000E-1	3.617E-1, Error: 9.6%	0.0, Error: 100%	3.720E-1, Error: 7%
ellipse-20-5				
P1: $\pi/2$	2.500E-2	2.502E-2, Error: 0.1%	0.0, Error: 100%	2.521E-2, Error: 0.8%
P2: $3\pi/8$	3.120E-2	2.856E-2, Error: 8.5%	1.254E-2, Error: 59.8%	3.109E-2, Error: 0.4%
P3: $\pi/4$	6.456E-2	9.967E-2, Error: 54.4%	2.954E-2, Error: 54.2%	4.990E-2, Error: 22.7%
P4: $\pi/8$	2.799E-1	8.183E-1, Error: 192.4%	7.184E-2, Error: 74.3%	1.624E-1, Error: 42.0%
P5: 0	1.600E+0	8.156E-1, Error: 49.0%	0.0, Error: 100%	1.178E+0, Error: 26.4%
		Avg Error: 24.8%	Avg Error: 79%	Avg Error: 8.8%
		Max Error: 192.4%	Max Error: 100%	Max Error: 42.0%

Table 4: A numerical comparison of curvature estimates for the ellipses in Figure 8, and the associated errors. For each ellipse, the sample points P1, P2, P3, P4, and P5 lie at orientations of $\pi/2$, $3\pi/8$, $\pi/4$, $\pi/8$, and 0 radians respectively, Figure 9.

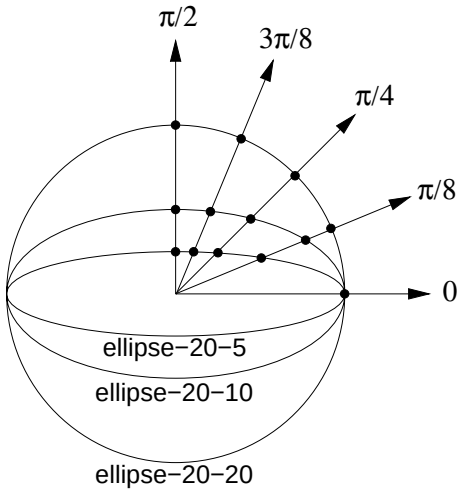

Figure 9: The sample points for the curvature estimates in Table 4 lie at orientations of $\pi/2$, $3\pi/8$, $\pi/4$, $\pi/8$, and 0 radians, respectively, from the center of each ellipse.

significant errors, *i.e.*, 20% or more, we see the limitation of using circular arc interpolants[7]. Typically the curvature is changing rapidly *within* such intervals, and higher order interpolants are required for improved accuracy. In summary, both theoretical and numerical considerations support the use of GENO in locating the curve and estimating its geometric properties.

6 Examples

In this section we illustrate the advantages of GENO by presenting examples of two popular applications of curve evolution in computer vision: shape segmentation and shape representation. In both these applications a 2D curve is evolved via the following partial differential equation:

$$\frac{\partial \mathcal{C}}{\partial t} = \beta(.)\vec{N},$$

where $\mathcal{C}$ is the boundary vector of curve coordinates, $\vec{N}$ is the outward normal, t is the time duration (magnitude) of the deformation, and β is an arbitrary function. It was shown that for numerical reasons [47, 55], as well as for theoretical ones [15, 12], it is preferable to embed the evolution in a higher dimension, where C is the zero level set of an evolving surface ϕ, *i.e.*, $\phi(x, y, t) = 0$. Typically, ϕ is taken to be the distance transform of the shape [5, 8]. However, other continuous functions may be used as well [15, 12]. The evolution of ϕ is then given by:

$$\frac{\partial \phi}{\partial t} + \beta(.)|\nabla \phi| = 0.$$

[7]It is worth noting that in such intervals, the level set method leads to even larger errors in the numerical estimates.

At each iteration, the recovery of $\mathcal{C}$ from ϕ is required, both for display purposes, and for obtaining accurate geometric estimates, Section 5. Since in the discrete domain the zero level set almost always passes between grid points, a straightforward discretization leads to a "jagged" appearance. As a remedy, Sethian and Strain [56] present a method to construct a piecewise linear approximation to the curve, using linear interpolation along gridlines to determine the line-segment end points. This method is very similar to bilinear interpolation and performs well in *smooth* regions, where at most one zero level set curve segment passes through each cell. However, it suffers from the same drawbacks: it cannot represent corners or multiple curve segments per cell, both of which can be significant aspects of the evolution. In the following we present numerical simulations that demonstrate the superiority of GENO.

6.1 Shape Segmentation

The first application we consider is that of shape segmentation, where a number of promising curve evolution approaches have recently been introduced [10, 38, 61]. In all these techniques the essential idea is to evolve a curve under the influence of image forces so that it is attracted to features of interest in an intensity image, *e.g.*, edges or areas of high gradient, combining ideas from active contours [23] with ones from level set modelling [47].

Figure 10 (top) illustrates the bubbles-based segmentation of a range image [61]. In this simulation three bubbles are initialized in homogenous regions of the image and they evolve to capture the outline of the screw. We focus on a 40x40 region of the original, depicted by the white box, and compare the recovery of the evolving fronts via bilinear interpolation of the embedding surface (middle left and bottom left) with GENO (middle right and bottom right). Observe that whereas bilinear interpolation results in topological splits when the fronts are within the same pixel, GENO is able to recover the fronts as distinct and to place discontinuities (circles) that lie along the threads of the screw.

6.2 Shape Representation

The second application we consider is of that of shape representation in the context of deformations of it [25, 27, 26]. The key to this representation is a classification of the shocks or discontinuities that occur in the course of evolution, into four types. The details of a numerical algorithm for shock detection based on GENO appear in [59]. The accurate representation of multiple curves per pixel, and the resulting discontinuities (shocks), are both significant aspect of their detection and classification.

In the following we compare GENO interpolation with discrete binarization, as well as with bilinear interpolation, for the recovery of the evolving curve in these four situations: (i) shapes evolving with

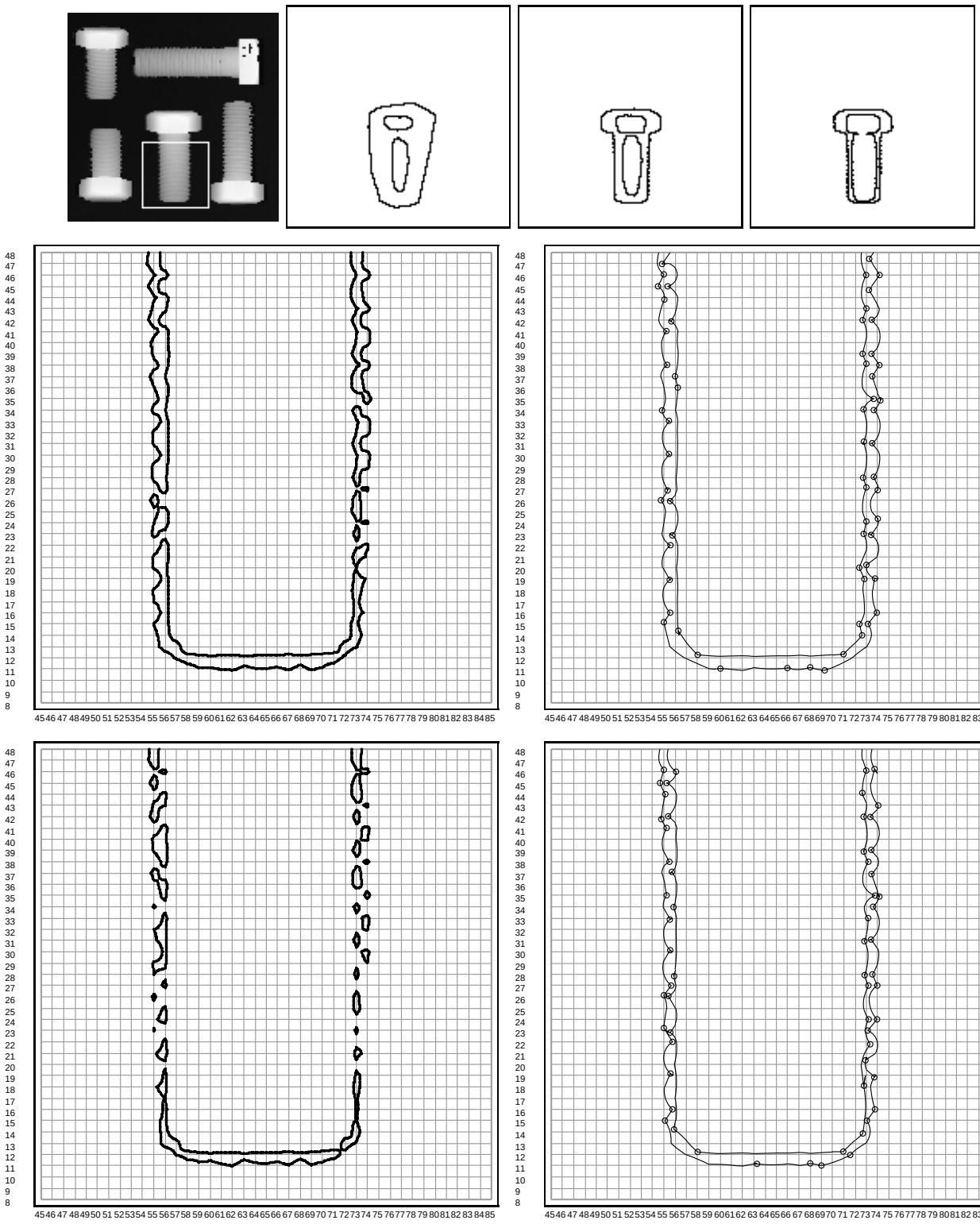

Figure 10: TOP: Three bubbles [61] are initialized in homogenous areas of a range image (left) and they evolve to capture the outline of a screw (left to right). MIDDLE AND BOTTOM: We focus on a 40x40 region, (the white box on the original), and compare bilinear interpolation (left) with GENO for recovery of the evolving fronts. Observe that whereas bilinear interpolation results in topological splits when the fronts are very close, GENO is able to recover the fronts as distinct and to place discontinuities (circles) that lie along the threads of the screw.

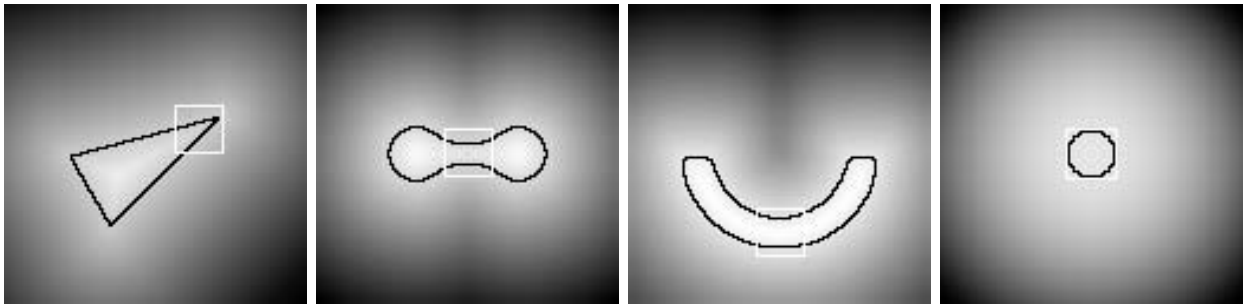

Figure 11: LEFT TO RIGHT: The embedding surfaces are 128 x 128 signed distance transforms of a "triangle", a "peanut", a "bend", and a "circle". In each case, the box depicts a 20 x 20 region under examination in Figures 12 and 13.

orientation discontinuities (first-order shocks), (ii) shapes developing topological splits (second-order shocks), (iii) the collapse of one curve segment onto another (third-order shocks), and finally (iv) the collapse of an entire shape onto a point (a fourth-order shock). Figure 11 depicts the original shapes, along with their embedding surfaces, which are then evolved using $\beta(.) = 1$, although other choices are possible as well.

First-order shocks: Figure 12 (top) displays the corner of an evolving triangle. Note that while the evolution preserves the discontinuity, this is not immediately evident in the discrete representation. The bilinear interpolation performs well in smooth regions. However, observe the artifacts near the corner, as well as the inaccuracies of corner placement. GENO interpolation, on the other hand, places both the corner and the adjacent line segments, to subpixel accuracy.

Second-order shocks: Figure 12 (bottom) depicts the central region of a "peanut" shape, which develops a topological split. Observe that both discrete binarization as well as bilinear interpolation truncate the evolving curve when it undergoes a topological change. The GENO interpolation, on the other hand, accurately represents the evolving front. Observe that the two approaching curve segments are recovered, even though both lie within the same cells. In addition, after the shape has split, the two cusps are accurately placed and represented with the appropriate orientation and curvature.

Third-order shocks: Figure 13 (top) depicts the evolution of two parallel curves which collide and subsequently annihilate. Observe that with discrete binarization the approximation to the curve boundaries is jagged, while bilinear interpolation leads to unwarranted topological splits. The GENO interpolation, on the other hand, accurately places the evolving curves as they are about to collide, even when portions of each curve lie within the same cell.

Fourth-order shocks: Figure 13 (bottom) displays an evolving circle which shrinks and collapses

onto a point. Note that the intermediate shapes should also be circles. However, both discrete binarization and bilinear interpolation lead to "squarish" representations as the circle becomes small. The GENO interpolation technique, on the other hand, is more faithful to the expected circle, even when it spans only slightly more than a cell.

7 Conclusion

To conclude, in this paper we have introduced a geometric interpolation scheme for subpixel recovery of level curves while respecting and explicitly placing discontinuities along them. The method relies on an extension of essentially non-oscillatory interpolation schemes to a particular subcell resolution implementation. Whereas subcell resolution for ENO was introduced by Harten [17], it applied to 1D conservation laws, and it was not clear how to extend the method to 2D conservation laws. In the current context we circumvent these limitations by: 1) using geometric continuity constraints (rather than conservation for conservation laws) and 2) using geometric building blocks (lines, circular arcs, etc.) rather than polynomial interpolants. As illustrated by several examples the method can be readily applied to the problem of recovering an evolving level set in a variety of curve evolution applications. In addition, the algorithm serves as a nice technique for postprocessing of image data to extract more accurate geometric information about level curves than regular ENO can do.

A Newton Divided Difference

In the following, we review the standard Newton divided difference, denoted $f[]$, where $f()$ denotes the function evaluated at its arguments. The standard Newton divided difference is inductively defined as $f[x_1, x_2, \cdots, x_{k+1}] = \frac{f[x_2,\cdots,x_{k+1}]-f[x_1,\cdots,x_k]}{x_{k+1}-x_1}$ with $f[x_1] = f(x_1)$. That is:

$$f[x_j] = f(x_j).$$

$$\begin{aligned} f[x_j, x_{j+1}] &= \frac{f[x_{j+1}] - f[x_j]}{x_{j+1} - x_j} \\ &= \frac{f(x_{j+1}) - f(x_j)}{x_{j+1} - x_j}. \end{aligned}$$

$$f[x_j, x_{j+1}, x_{j+2}] = \frac{f[x_{j+1}, x_{j+2}] - f[x_j, x_{j+1}]}{x_{j+2} - x_j}$$

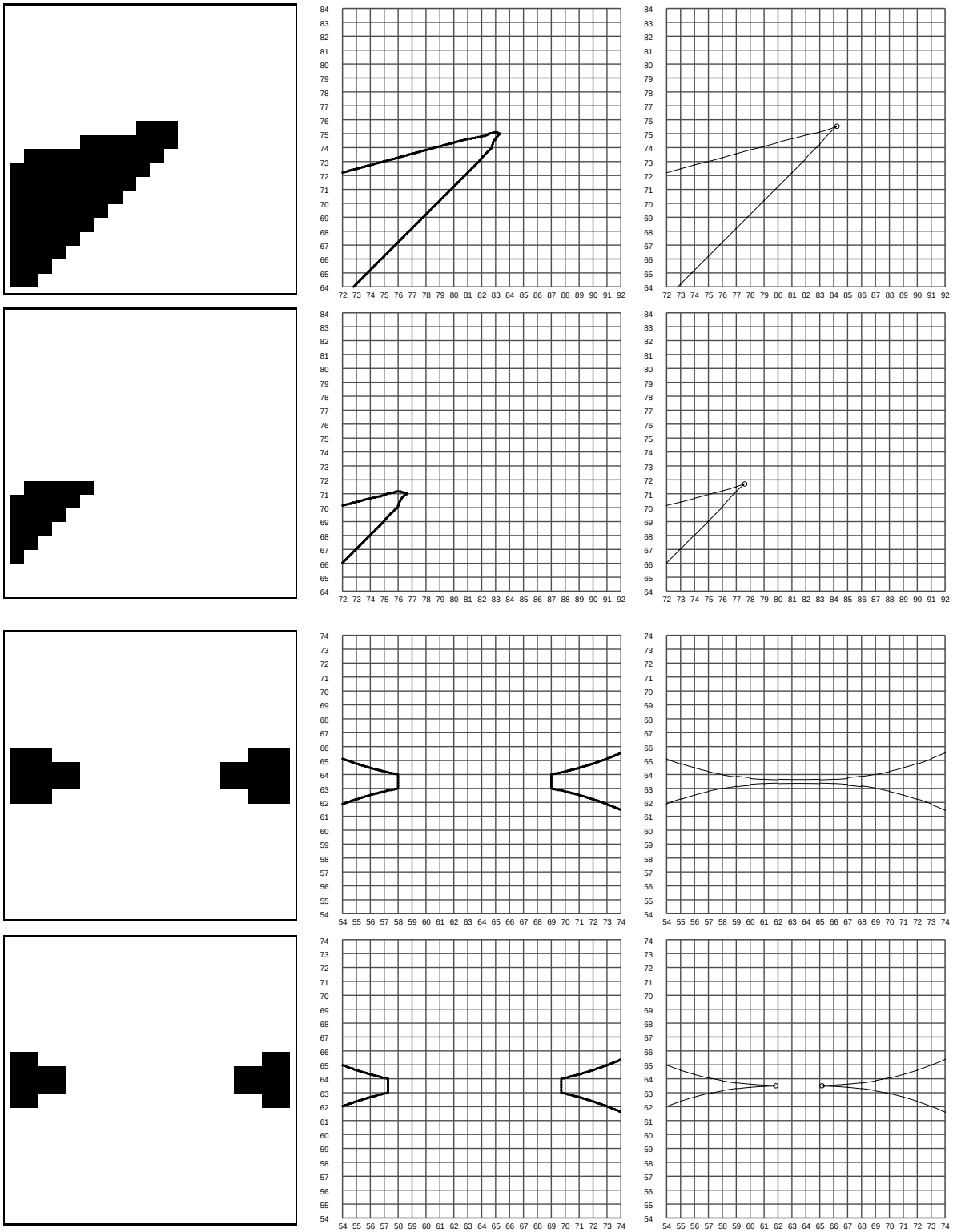

Figure 12: Each row depicts the interior of the shape at the resolution of the grid (LEFT), the bilinear interpolation of the boundary (MIDDLE) and the GENO interpolation of the boundary, with detected corners marked with circles (RIGHT). Note that in contrast to bilinear interpolation, GENO interpolation is able to capture the corner of the evolving triangle (first-order shock, TOP TWO ROWS) with subpixel resolution and without introducing artifacts, and to represent the topological split at the neck (second-order shock, BOTTOM TWO ROWS), followed by the formation of cusps on either side.

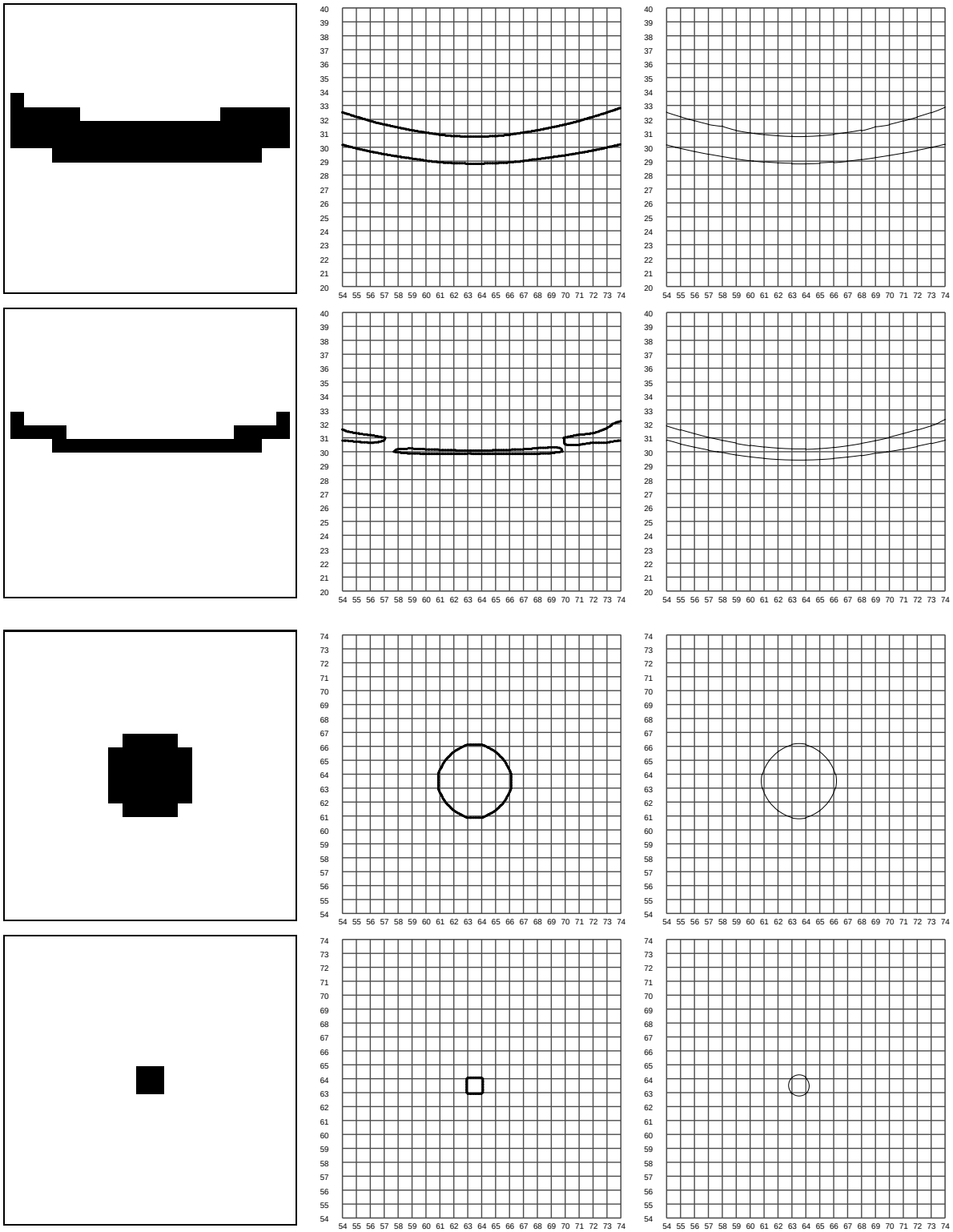

Figure 13: Each row depicts the interior of the shape at the resolution of the grid (LEFT), the bilinear interpolation of the boundary (MIDDLE) and the GENO interpolation of the boundary (RIGHT). Note that in contrast to bilinear interpolation, Geno interpolation is able to represent the collapse of the bend without introducing artifacts (third-order shocks, TOP TWO ROWS), and to preserve the circular shape of the circle, even as it shrinks to a point (fourth-order shock, BOTTOM TWO ROWS).

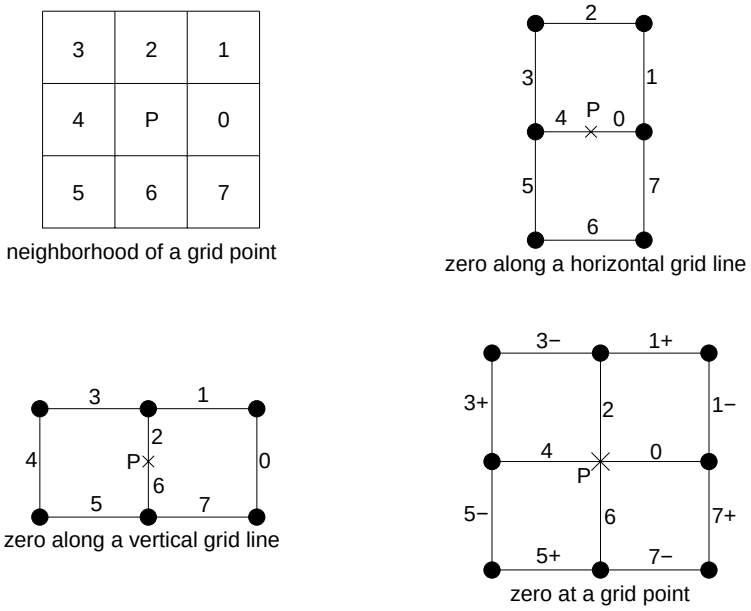

Figure 14: This figure illustrates the ENO contour tracer. Since a grid point has 8 neighbors, in the case of a discrete contour tracer there are 8 directions in which to search for the next contour point, 0 to 7, (TOP LEFT). For the ENO contour tracer, there are 8 intervals to search in for neighboring zero-crossings, but these differ depending upon whether the current point P lies along a horizontal gridline (TOP RIGHT), or a vertical gridline (BOTTOM LEFT). However, an exception arises when the current point P coincides exactly with a grid point (BOTTOM RIGHT). In this case, there are 12 intervals to search in.

$$\begin{aligned} &\vdots \\ f[x_j, ..., x_{j+k}] &= \frac{f[x_{j+1}, ..., x_{j+k}] - f[x_j, ..., x_{j+k-1}]}{x_{j+k} - x_j}. \end{aligned}$$

B ENO Contour Tracer

In the following, we propose an algorithm for obtaining a set of ordered high resolution sample points which lie along a shape's boundary, *without blurring across discontinuities*. The essential idea is to extend a standard (discrete) contour tracer such that instead of grid points, it uses high resolution boundary points obtained by ENO style interpolation, as discussed in Section 4.1. We begin by reviewing a standard contour tracer, and then show how it can be modified to accommodate ENO interpolation.

First, we summarize a standard contour tracing algorithm [50], adopting the convention that the shape's boundary is traced in a counter clockwise (CCW) direction around the shape. The essential idea behind this discrete contour tracer is to add, at each step, a single contour point to the end of an existing list of points; see [50] for how to initialize the tracer and further details. Given the last point

P added to the list, there are a fixed number of directions (8) in which to search for the next point to append, Figure 14 (top left). The search proceeds CCW around P, beginning in the direction of the next to last point in the list, P^-, and ending at the first point found that lies in the interior of the shape, P^+. This process is then repeated until the contour tracer returns to the point at which the trace started, and the trace is therefore complete.

Second, to see how the above tracer can be extended using ENO style interpolation, recall from Section 4.1 that high resolution boundary points can be found by locating zero-crossings of interpolation polynomials along grid lines, Figure 6. Observe that the local neighborhood of such a zero-crossing can be of two types, depending upon whether the crossing lies along a horizontal gridline, Figure 14 (top right) or a vertical gridline, Figure 14 (bottom left). However, in each case, there are still a fixed number of neighboring intervals (8) in which to search for neighboring zero-crossings. Thus, the same algorithm outlined above for the discrete case, also applies here. The difference is that instead of searching in a fixed number of directions to find the next grid point to store, one searches in a fixed number of intervals to find the next zero-crossing to store. However, since in general there can be more than one zero-crossing in the interval[8] care has to be taken to always select the *first* zero-crossing found, ordered in a CCW direction about the current zero-crossing. We should mention that there is one exception to the above algorithm, which arises when a zero-crossing lies *exactly* at a grid point, *i.e.*, both along a vertical and a horizontal gridline. In such a case, the search neighborhood is slightly more complicated, as shown in Figure 14 (bottom right).

Finally, we wish to stress two important properties of the ENO contour tracer, both of which lead to significant computational savings. First, whereas the ENO contour tracer provides an ordered set of high resolution boundary points, there is no explicit uniformly high resolution representation. Second, since at each step the search for which boundary point to append is local and depends *only* on the last two boundary points stored in the list, there is no need to first interpolate zero-crossings throughout the 2D grid, Section 4.1. Rather, the ENO interpolation can be performed hand in hand with the contour tracing procedure, and *only* in intervals where the contour tracer searches for boundary points.

Acknowledgements Many thanks to Hüseyin Tek for technical help with the 'bubble' simulations. The support of the following grants is gratefully acknowledged: NSERC research grant 0GP0183831, NSF grant IRI-9305630, NSF grant DMS-9211820, ARO grant DAAH04-94-G-0205, and AFOSR grant

[8] This of course depends on the order of the interpolation polynomial being used along the gridlines; in all our simulations we have found second-order polynomials to give excellent results.

93-0090.

References

[1] L. Alvarez, F. Guichard, P.-L. Lions, and J.-M. Morel. Axiomes et equations fundamentales du traitement d'images. (analyse multiechelle et e.d.p). *C. R. Acad. Sci. Paris*, 1:135–138, 1992.

[2] L. Alvarez, P.-L. Lions, and J.-M. Morel. Image selective smoothing and edge detection by nonlinear diffusion: II. *SIAM Journal of Numerical Analysis*, 29(3):845–866, June 1992.

[3] L. Alvarez and J.-M. Morel. Formalization and computational aspects of image analysis. *Acta Numerica*, pages 1–59, 1994.

[4] H. Asada and M. Brady. The curvature primal sketch. *IEEE PAMI*, 8:2–14, 1983.

[5] G. Barles. Remarks on a flame propagation model. Technical Report No 464, INRIA Rapports de Recherche, December 1985.

[6] B. A. Barsky and J. C. Beatty. Local control of bias and tension in beta-splines. *ACM Transactionson Graphics*, 2:109–134, 1983.

[7] I. Biederman. Recognition by components. *Psych. Review*, 94:115–147, 1987.

[8] G. Borgefors. Distance transformations in arbitrary dimensions. *Computer Vision, Graphics and Image Processing*, 27:321–345, 1984.

[9] A. M. Bruckstein. On shape from shading. *Computer Vision, Graphics, and Image Processing*, 44:139–154, May 1988.

[10] V. Caselles, F. Catte, T. Coll, and F. Dibos. A geometric model for active contours in image processing. Technical Report No 9210, CEREMADE, 1992.

[11] F. Catté, P.-L. Lions, J.-M. Morel, and T. Coll. Image selective smoothing and edge detection by nonlinear diffusion. *SIAM Journal of Numerical Analysis*, 29(1):182–193, February 1992.

[12] Y. Chen, Y. Giga, and S. Goto. Uniqueness and existence of viscosity solutions of generalized mean curvature flow equations. *Journal of Differential Geometry*, 33(3):749–786, 1991.

[13] I. J. Cox, J. B. Kruskal, and D. A. Wallach. Predicting and estimating the accuracy of a subpixel registration algorithm. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 12(8):721–734, 1990.

[14] C. De Boor. *A practical guide to Splines.* Springer Verlag, 1978.

[15] L. C. Evans and J. Spruck. Motion of level sets by mean curvature I. *Journal of Differential Geometry*, 33(3):635–681, May 1991.

[16] G. H. Golub, M. Heath, and G. Wahba. Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics*, 21(2):215–223, 1979.

[17] A. Harten. ENO schemes with subcell resolution. *Journal of Computational Physics*, 83:148–184, 1989.

[18] A. Harten, B. Engquist, S. Osher, and S. Chakravarthy. Uniformly high order accurate essentially non-oscillatory schemes, III. *Journal of Computational Physics*, 71:231–303, 1987.

[19] E. C. Hildreth. *The Measurement of Visual Motion*. MIT Press, Cambridge, MA, 1983.

[20] D. D. Hoffman and W. A. Richards. Parts of recognition. *Cognition*, 18:65–96, 1985.

[21] P. D. Hyde and L. S. Davis. Subpixel edge estimation. *Pattern Recognition*, 16(4):413–420, 1983.

[22] L. A. Iverson and S. W. Zucker. Logical/linear operators for image curves. Technical Report TR-CIM-92-12, McGill University, 1992.

[23] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1:321–331, 1988.

[24] B. B. Kimia and K. Siddiqi. Geometric heat equation and non-linear diffusion of shapes and images. *Computer Vision Graphics and Image Processing: Image Understanding*, In Press, 1995.

[25] B. B. Kimia, A. R. Tannenbaum, and S. W. Zucker. Toward a computational theory of shape: An overview. In *Proceedings of the First European Conference on Computer Vision*, Antibes, France, 1990. Springer Verlag.

[26] B. B. Kimia, A. R. Tannenbaum, and S. W. Zucker. Entropy scale-space. In C. Arcelli, editor, *Visual Form: Analysis and Recognition*, pages 333–344, New York, May 1991. Plenum Press.

[27] B. B. Kimia, A. R. Tannenbaum, and S. W. Zucker. Shapes, shocks, and deformations, I: The components of shape and the reaction-diffusion space. *International Journal of Computer Vision*, 15:189–224, 1995.

[28] R. Kimmel, K. Siddiqi, B. B. Kimia, and A. Bruckstein. Shape from shading: Level set propagation and viscosity solutions. *International Journal of Computer Vision*, In press, 1994.

[29] N. Kiryati and A. Bruckstein. Gray levels can improve the performance of binary image digitizers. *Computer Vision Graphics and Image Processing: Graphical Models and Image Processing*, 53(1):31–39, January 1991.

[30] J. J. Koenderink. The structure of images. *Biological Cybernetics*, 50:363–370, 1984.

[31] J. J. Koenderink and A. J. van Doorn. Dynamic shape. *Biological Cybernetics*, 53:383–396, 1986.

[32] A. Kumar, A. R. Tannenbaum, and G. J. Balas. Optical flow: A curve evolution approach. *IEEE Transaction on Image Processing*, to appear, 1994.

[33] A. Kumar, A. R. Tannenbaum, and S. W. Zucker. L^1 minimization for stereo disparity. Technical report, Dept. of Electrical Engineering, University of Minnesota, 1995.

[34] P. C. K. Kwok and C. Dong. The estimation of curves to subpixel accuracy. In *Proceedings of the International Workshop on Visual Form*, pages 324–333, Capri, Italy, May 1994. World Scientific.

[35] J. D. Lawrence. *A Catalog of Special Plane Curves*. New York, 1972.

[36] Y. G. Leclerc. Constructing simple stable descriptions for image partitioning. *International Journal of Computer Vision*, 3:73–102, 1989.

[37] P. J. MacVicar-Whelan and T. O. Binford. Line finding with subpixel precision. *Proceedings: Image Understanding Workshop*, pages 26–31, 1981.

[38] R. Malladi, J. A. Sethian, and B. C. Vemuri. Shape modelling with front propagation: A level set approach. to appear in IEEE Transactions on Pattern Analysis and Machine Intelligence.

[39] R. Malladi, J. A. Sethian, and B. C. Vemuri. Evolutionary fronts for topology-independent shape modelling and recovery. In *Proceedings of ECCV '94*, pages 3–13, 1994.

[40] D. Marr and E. Hildreth. Theory of edge detection. Technical Report MIT AI Memo 518, MIT AI Lab, 1979.

[41] D. Marr and T. Poggio. A theory of human stereo vision. *Proceedings of the Royal Society of London*, B 204:301–328, 1979.

[42] J. E. W. Mayhew and J. P. Frisby. Psychophysical and computational studies towards a theory of human stereopsis. *Artificial Intelligence*, 17:349–385, March 1981.

[43] F. Mokhtarian and A. Mackworth. Scale-based description of planar curves and two-dimensional shapes. *PAMI*, 8:34–43, 1986.

[44] G. M. Nielson. *Some piecewise polynomial alternatives to splines under tension*, pages 209–235. Academic Press, 1974.

[45] Y. Nomura, M. Sagara, H. Naruse, and A. Ide. Edge location to subpixel precision and analysis. *Systems and Computers in Japan*, 22(9):70–81, 1991.

[46] L. O'Gorman, A. M. Bruckstein, C. B. Bose, and I. Amir. Subpixel registration using a concentric ring fiducial. In *International Conference on Pattern Recognition*, pages 249–253, 1990.

[47] S. Osher and J. Sethian. Fronts propagating with curvature dependent speed: Algorithms based on Hamilton-Jacobi formulations. *Journal of Computational Physics*, 79:12–49, 1988.

[48] S. Osher and C.-W. Shu. High-order essentially non-oscillatory schemes for Hamilton-Jacobi equations. *SIAM J. Numer. Anal.*, 28:907–922, 1991.

[49] P. Parent and S. W. Zucker. Trace inference, curvature consistency and curve detection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 11(8):823–839, August 1989.

[50] T. Pavlidis. *Algorithms for Graphics and Image Processing.* Computer Science Press, Rockville, Maryland, 1982.

[51] T. Poggio, V. Torre, and C. Koch. Computational vision and regularization theory. *Nature*, 317:314–319, September 1985.

[52] E. Rouy and A. Tourin. A viscosity solutions approach to shape-from-shading. *SIAM. J. Numer. Analy.*, 29(3):867–884, June 1992.

[53] G. Sapiro and A. Tannenbaum. Affine invariant scale-space. *International Journal of Computer Vision*, 10:25–44, 1993.

[54] D. G. Schweikert. An interpolation curve using a spline in tension. *Journal of Mathematical Physics*, 45:312–317, 1966.

[55] J. A. Sethian. Curvature and the evolution of fronts. *Comm. Math. Physics*, 101:487–499, 1985.

[56] J. A. Sethian and J. Strain. Crystal growth and dendritic solidification. *Journal of Computational Physics*, 98:231–253, 1992.

[57] B. Shahraray and D. J. Anderson. Optimal estimation of contour properties by cross-validated regularization. *PAMI*, 11(6):600–610, 1989.

[58] C.-W. Shu and S. Osher. Efficient implementation of essentially non-oscillatory shock-capturing schemes. *Journal of Computational Physics*, 77:439–471, 1988.

[59] K. Siddiqi and B. B. Kimia. A shock grammar for recognition. In *CVPR'96 (IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Francisco, California, June 21–23, 1996)*, Washington, DC., June 1996. Computer Society Press.

[60] P. Stoll, C. Shu, and B. B. Kimia. Shock capturing numerical methods for viscosity solutions of certain pdes in computer vision: the Godunov, Osher-Sethian and ENO schemes. Technical Report 132, Brown University, LEMS, May 1994.

[61] H. Tek and B. B. Kimia. Image segmentation by reaction-diffusion bubbles. In *Fifth International Conference on Computer Vision*, Boston, Massachusetts, June 1995. IEEE Computer Society.

[62] B. M. ter Haar Romeny, editor. Kluwer, September 1994.

[63] D. Terzopoulos. Regularization of inverse visual problems involving discontinuities. *IEEE Trans. Patt. Anal. Mach. Intell.*, 8(4):413–424, 1986.

[64] S. Ullman. *The Interpretation of Visual Motion.* MIT Press, Cambridge, MA, 1979.