

# Narrow Band Methods for PDEs on Very Large Implicit Surfaces

Oliver Nemitz<sup>1</sup>, Michael Bang Nielsen<sup>2</sup>, Martin Rumpf<sup>3</sup>, Ross Whitaker<sup>4</sup>

<sup>1,3</sup>University of Bonn, Germany, <sup>2</sup>University of Århus, Denmark, <sup>4</sup>University of Utah, U.S.A.

Email: <sup>1,3</sup>{oliver.nemitz, martin.rumpf}@ins.uni-bonn.de,

<sup>2</sup>bang@daimi.au.dk, <sup>4</sup>whitaker@cs.utah.edu

## Abstract

Physical simulation on surfaces and various applications in geometry processing are based on partial differential equations on surfaces. The implicit representation of these eventually evolving surfaces in terms of level set methods leads to effective and flexible numerical tools. This paper addresses the computational problem of how to solve partial differential equations on level sets with an underlying very high-resolution discrete grid. These high-resolution grids are represented in a very efficient format, which stores only grid points in a thin narrow band. Reaction diffusion equations on a fixed surface and the evolution of a surface under curvature motion are considered as model problems. The proposed methods are based on a semi implicit finite element discretization directly on these thin narrow bands and allow for large time steps. To ensure this, suitable transparent boundary conditions are introduced on the boundary of the narrow band and the time discretization is based on a nested iteration scheme. Methods are provided to assemble finite element matrices and to apply matrix vector operators in a manner that do not incur additional overhead and give fast, cache-coherent access to very large data sets.

## 1 Introduction

This paper addresses the computational problem of how to solve partial differential equations (PDEs) on the level sets of smooth scalar functions that are approximated by very high-resolution discrete grids. The context for this work is the growing interest in computing PDEs on surfaces that are represented implicitly as the level sets of a smooth scalar function  $\phi$ . Starting with the pioneering paper by Osher and Sethian [31] this way has become increasingly important in a variety of fields such as

computational physics [3,4,6,7,20], scientific visualization [23], image analysis [5,8], and computer graphics [27,30]. Most of these applications rely on the efficient computation of partial differential equations on curves or surfaces implicitly represented by a level set function  $\phi$  resolved on a discrete, usually structured, grid. The attraction of solving problems with discretely sampled implicit surfaces is the relatively large number of degrees of freedom provided by the grid and the freedom of not having to choose an explicit surface parameterization, which often limits shape and topology.

There are in particular two scenarios in which such surface-based PDEs are interesting. The first is when the implicit surface serves as the domain and one would like to solve a PDE for a function  $u$  intrinsic on the surface. Projections of the derivatives in the ambient space onto the surface provide a mechanism for computing differential operators that live on the surface [5]. The other scenario is when the surface itself evolves according to a geometric PDE that depends on the shape. The most prominent example is motion by mean curvature [16]. For the discretization in space either finite difference [31,33] or finite element schemes [10] are considered. Semi-implicit time discretizations are suitable due to their stability properties also for large time steps, compared to explicit time discretization for diffusion type problems which require time steps of the grid size squared. This is particularly important when one is considering higher order PDEs [14,20].

Perhaps the greatest promise of level-set methods, for both moving interfaces and PDEs defined on static surfaces (codimension one), is their ability to deal with a wide variety of complicated shapes in an elegant manner within a single computational framework. However, the computation and memory requirements on the discrete grid that represents  $\phi$  become prohibitive as the grid resolution increases.

The complexity of the surface increases (roughly) as the grid resolution squared, but the overall grid size increases with the cube of the resolution.

Several technical advances have addressed different aspects of the problem associated with storing and computing level-set equations at high resolutions. The introduction of methods that solve PDEs on a small subset of grid points, that constitute a narrow band around the surface [2,32,37] provided significant advantages in computation time. As grid sizes become progressively larger the number of computations in the narrow band is not the limiting factor on performance. Rather, the performance of computations is limited by the very small fraction of the grid values that can fit into cache or random-access memory. To address this several authors have proposed memory-efficient data structures for storing the narrow bands associated with level sets that are represented with large grids.

The use of such narrow bands, which can encode many millions of degrees of freedom, gives the level-set approach to surface representation a *distinct computational advantage* relative to parametric representations, such as triangle meshes. The reason is that with careful attention to how grid points are stored and accessed, the grid-based, implicit method for processing surfaces provides regular, predictable access to memory in a way that allows for cache coherency (on conventional processors) and data streaming on more advanced architectures.

This very narrow computational domain presents a challenge for numerical schemes, however, because one must introduce a solution for the PDE along the boundary domain, whose shape can be quite irregular. As the resolution increases the boundaries of the computational domain become progressively closer to the level set of interest, and the so called *natural* boundary conditions allow artifacts from the grid (whose faces are aligned with the cardinal directions) to propagate into the PDE on the surface. Furthermore, when solving free boundary problems with finite differences, the time steps must be limited so that at each iteration the moving interface (level set of interest) is neither impeded by the boundary conditions nor allowed to pass outside of the computational domain (at which point its shape is lost).

This paper addresses these issues by introducing numerical schemes for finite element solutions

to PDEs on implicit surfaces that are appropriate for the Dynamic Tubular Grid (DT-grid) data structure [28] which is storage efficient and fast in practice. In contrast to the finite difference schemes already implemented in this context we consider here finite element methods with semi-implicit schemes in time and introduce the required suitable DT-grid based linear algebra operations on finite element matrices. These numerical schemes introduce transparent boundary conditions together with nested iterations in the time discretization that do not allow the irregularity of the narrow band to impact the solution. In the case of moving interfaces, this allows semi-implicit updates with larger time steps that do not restrict the updated solution to the computational domain from the previous time step. As applications we consider texture synthesis by systems of reaction diffusion equations and the evolution of surfaces by mean curvature motion on very large data sets that are appropriate for state-of-the-art applications in surface processing.

## 2 Related Work

There are several bodies of related work with respect to narrow band methods and corresponding sparse storage schemes. Narrow band methods for moving interface simulation were first proposed in [2] and proved their efficiency in various applications [18, 27, 38]. Narrow band techniques have been combined with boundary element methods [17] and with multiscale resolution techniques [39]. In [19] the reinitialization of the level set function on a narrow band is discussed and instabilities at the narrow band boundary are avoided by smoothing kernels applied to the level set function. Surface evolution based on the evolution of distance functions on narrow band domains is investigated theoretically in [11]. A heap sorted queue is applied for the administration of narrow band data in an active contour method [29]. Already in [12] Marc Droske proposed a finite element method for Willmore flow based on an iterative update scheme on narrow bands. We pick up and refine this type of iterative update scheme here on very thin narrow bands and high resolution background grids.

The work presented also builds on the research in computer science on efficient data structures for storing sparse computational domains associated with level sets. In recent years quadtrees (2D)

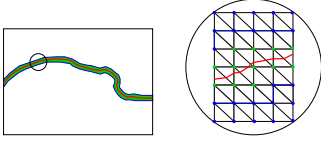


Figure 1: A sketch of a narrow band domain  $\Omega_n$  corresponding to a level set (plotted in red) is shown. In the zoom in on the right interior nodes are indicated by green dots, whereas the boundary  $\partial\Omega_n$  is represented by the blue lines.

and octrees (3D) [9] have been applied to level sets in numerous papers [13, 15, 25, 26, 34]. The quadtree and octree data structures reduce the storage requirements of level sets to  $O((d+1)n)$ , but also introduce an  $O(d)$  access time, where  $d$  is the depth of the quadtree or octree and  $n$  is the number of grid points in the narrow band. The Dynamic Tubular Grid [28] employs a hierarchical encoding of the topology of the narrow band, inspired by the storage-format of sparse matrices. Subsequent works employ a run-length encoding, and focus either on flexibility [21] or are tailored for a specific application in fluid simulation [22]. All of these data structures require  $O(n)$  storage and have  $O(1)$  access time to grid points in a local stencil during the sequential access typically required by level set methods. Also they perform faster in practice than recent narrow band and octree approaches due both to the lower memory footprint and the more cache coherent memory layout and access patterns [21]. In this work we utilize the DT-Grid since it has been shown to perform slightly faster than the run-length encoding alternatives.

### 3 Finite Elements on Narrow Bands

#### 3.1 Review of Level Set Finite Elements

Let us consider the finite-element formulation for reaction-diffusion equations on a fixed surface and a discretization of curvature motion, respectively. We deal with both as model problems for the cases of static and moving surfaces, respectively.

**A reaction-diffusion model on level sets.** We consider the following scalar initial value problem: The solution is a function  $u : \mathbb{R}^+ \times \mathcal{M} \rightarrow \mathbb{R}$ , such that  $\partial_t u - \Delta_{\mathcal{M}} u = f(u)$  with initial condition  $u(0) = u^0$ , where  $u^0$  is some initial value function on the surface  $\mathcal{M}$  and  $\Delta_{\mathcal{M}}$  is the Laplace-Beltrami

operator on  $\mathcal{M}$ . We represent the surface  $\mathcal{M}$  as the zero set of a smooth scalar function  $\phi : \Omega \rightarrow \mathbb{R}$ , so that  $\mathcal{M} = \{x | \phi(x) = 0\}$ , where  $\Omega$  is a box domain enclosing  $\mathcal{M}$ . The Laplace Beltrami operator of  $u$  is expressed in terms of derivatives of  $\phi$ , which gives  $\Delta_{\mathcal{M}} u = |\nabla\phi|^{-1} \text{div}(|\nabla\phi|P[\phi]\nabla u)$ , where  $P[\phi] = \mathbb{I} - \frac{\nabla\phi}{|\nabla\phi|} \otimes \frac{\nabla\phi}{|\nabla\phi|}$  is the projection onto the tangent space  $T_x\mathcal{M}$  of the surface  $\mathcal{M}$ . Now, we first discretize in time and introduce a time derivative  $\frac{u^{k+1} - u^k}{\tau}$  for time step functions  $u^k$ . Testing the equation with a smooth function  $\theta$  and applying integration by parts we derive the following time discrete weak formulation:

$$\begin{aligned} & \int_{\Omega} |\nabla\phi| \frac{u^{k+1} - u^k}{\tau} \theta + |\nabla\phi| P[\phi] \nabla u^{k+1} \cdot \nabla \theta \, dx \\ &= \int_{\Omega} |\nabla\phi| f(u^k) \theta \, dx \end{aligned} \quad (1)$$

for all test functions  $\theta \in C^1$ . Here the nonlinear right hand side  $f$  is evaluated on the old time step (forward differences). The operator  $P[\phi]$  ensures a decoupling of the reaction-diffusion process on different level sets  $[\phi = c]$ , which reflects the geometric nature of the problem. Thus, to identify the solution on  $\mathcal{M}$  it suffices to consider the weak formulation restricted to a small band around  $\mathcal{M}$ . Next, we discretize in space based on a finite element approximation. We denote discrete quantities with upper case letters to distinguish them from continuous quantities in lower case letters. The domain  $\Omega$  is supposed to be covered by a regular hexahedral grid and we denote the corresponding space of continuous, piecewise tri-linear functions by  $\mathcal{V}^h$ , where  $h$  indicates the grid size. Let  $\{\Phi_i\}_{i \in \mathbf{I}}$  be the canonical nodal basis of this finite element space for an index set  $\mathbf{I}$  corresponding to all grid nodes. A discrete function  $U$  is represented as a nodal vector  $\bar{U}$  corresponding to nodes of the spatial grid. We achieve the vector  $\bar{U} = (U_i)_{i \in \mathbf{I}}$ , where  $U = \sum_{i \in \mathbf{I}} U_i \Phi_i$  is the corresponding function. Given an approximation  $\Phi \in \mathcal{V}_h$  of the level set function  $\phi$ , we obtain an approximation  $\mathcal{M}^h := [\Phi = 0]$  of the continuous surface  $\mathcal{M}$  as one particular discrete level set represented by the function  $\Phi$ . Concerning the reaction-diffusion model, we replace all continuous quantities in (1) by their discrete counterparts and introduce mass lumping. Thus, we define the weighted lumped mass and stiffness matrix

$$\mathbf{M}[\Phi] = \left( \int_{\Omega} \mathcal{I}_h^0(|\nabla\Phi|) \mathcal{I}_h^1(\Phi_i \Phi_j) \, dx \right)_{i,j \in \mathbf{I}},$$

$$\mathbf{L}[\Phi] = \left( \int_{\Omega} |\nabla\Phi| P[\Phi] \nabla\Phi_i \cdot \nabla\Phi_j \, dx \right)_{i,j \in \mathbf{I}},$$

where  $\mathcal{I}_h^0, \mathcal{I}_h^1$  denote the piecewise constant and the piecewise multilinear Lagrangian projection, respectively. Furthermore, we introduce the right hand side vector  $\bar{F}[U] = (f(U_i))_{i \in \mathbf{I}}$  and end up with the system of linear equations

$$(\mathbf{M}[\Phi] + \tau \mathbf{L}[\Phi]) \bar{U}^{k+1} = \mathbf{M}[\Phi] \left( \tau \bar{F}[U^k] + \bar{U}^k \right).$$

Solving these systems we iteratively compute  $(U^k)_{k \geq 1}$  for a given approximation  $U^0$  of  $u^0$ .

**Curvature Motion of Level Sets.** The second application considered in this paper is the evolution of surfaces under mean curvature motion. Given an initial surface  $\mathcal{M}_0$  we ask for a family of surfaces  $\mathcal{M}(t)$  generated from the motion of points  $x(t)$  under the evolution  $\dot{x}(t) = -h(t)n(t)$  with initial condition  $x(0) = x_0$  with  $x_0 \in \mathcal{M}_0$ . Here  $n(t)$  is the normal and  $h(t)$  the mean curvature on  $\mathcal{M}(t)$ . The corresponding level set equation is given by  $\partial_t \phi - |\nabla\phi| \operatorname{div} \left( \frac{\nabla\phi}{|\nabla\phi|} \right) = 0$  on  $\mathbb{R}^+ \times \Omega$  with initial data  $\phi_0$ . Again discretizing in time and applying integration by parts we obtain the weak formulation

$$\int_{\Omega} \frac{\phi^{k+1} - \phi^k}{\tau |\nabla\phi^k|_{\epsilon}} \theta + \frac{\nabla\phi^{k+1}}{|\nabla\phi^k|_{\epsilon}} \cdot \nabla\theta \, dx = 0 \quad (2)$$

for test functions  $\theta \in C^1$ . Here, we take into account the old time step solution for the weight  $|\nabla\phi|^{-1}$  and the usual regularization  $|x|_{\epsilon} = \sqrt{\epsilon^2 + |x|^2}$ . Now, as in the case of the reaction-diffusion equation on a fixed surface we discretize in space and again end up with a sequence of linear systems of equations

$$\left( \mathbf{M}[\Phi^k] + \tau \mathbf{L}[\Phi^k] \right) \bar{\Phi}^{k+1} = \mathbf{M}[\Phi^k] \bar{\Phi}^k$$

for the nodal vector  $\bar{\Phi}^{k+1}$  of the discrete level set function at time  $t_{k+1} = \tau(k+1)$ . Here, the involved lumped mass and stiffness matrices are given by

$$\mathbf{M}[\Phi] = \left( \int_{\Omega} \mathcal{I}_h^0(|\nabla\Phi|_{\epsilon}^{-1}) \mathcal{I}_h^1(\Phi_i \Phi_j) \, dx \right)_{i,j \in \mathbf{I}},$$

$$\mathbf{L}[\Phi] = \left( \int_{\Omega} |\nabla\Phi|_{\epsilon}^{-1} \nabla\Phi_i \cdot \nabla\Phi_j \, dx \right)_{i,j \in \mathbf{I}}.$$

Solving these systems we iteratively compute  $(\Phi^k)_{k \geq 1}$  for a given approximation  $\Phi^0$  of  $\phi^0$  and obtain a sequence of discrete surfaces  $\mathcal{M}_h^k = [\Phi^k = 0]$ .

### 3.2 Transparent Neumann Boundary

The continuous formulation operates on the solution of each level-set separately, and thus, solutions from different level sets do not interact and we can truncate the computational domain to a narrow band around the zero set without affecting the solution. However, the discrete formulation introduces a coupling of nearby level sets through the finite extent of the test/basis functions and the natural boundary conditions induced by the weak formulation interfere strongly with the solution on  $\mathcal{M}_h$  in case of a thin narrow band. This interaction undermines the numerical convergence of the scheme on finer grids. In this section we will describe boundary conditions which avoid this interference.

Given a level set surface  $\mathcal{M}_h$  for a level set function  $\Phi \in \mathcal{V}^h$  we define a discrete narrow band as a union of supports of discrete basis functions. Hence, we consider a corresponding index set  $\mathbf{I}_{\text{int}} := \{i \in \mathbf{I} \mid \operatorname{supp} \Phi_i \cap \mathcal{M}^h \neq \emptyset\}$  and the resulting narrow band domain  $\Omega_n = \bigcup_{i \in \mathbf{I}_{\text{int}}} \operatorname{supp} \Phi_i$ . This is the smallest possible band to resolve the discrete surface  $\mathcal{M}_h$ . Let us denote by  $\mathcal{V}_{\text{int}}^h = \operatorname{span}\{\Phi_i \mid i \in \mathbf{I}_{\text{int}}\}$  the space of discrete functions on  $\Omega_n$  which vanish on  $\partial\Omega$  and by  $\mathcal{V}_{\text{bd}} = \operatorname{span}\{\Phi_i \mid i \in \mathbf{I} \setminus \mathbf{I}_{\text{int}}, \operatorname{supp} \Phi_i \cap \Omega_n \neq \emptyset\}$  the discrete function space corresponding to boundary values on  $\partial\Omega_n$ . Hence, the direct sum  $\mathcal{V}_n^h = \mathcal{V}_{\text{int}}^h \oplus \mathcal{V}_{\text{bd}}^h$  represents the discrete finite element space corresponding to the narrow band domain  $\Omega_n$ . Now, we replace the domain of integration in the weak formulations by the narrow band domain.

For the **reaction diffusion model** integration by parts leads to the boundary integral  $\int_{\partial\Omega_n} |\nabla\phi| P[\phi] \nabla u^{k+1} \cdot \nu \theta \, da$  on  $\partial\Omega_n$ , which gives rise to the Neumann boundary condition  $P[\phi] \nabla u^{k+1} \cdot \nu = 0$ . This condition is meaningless and for  $P[\phi] \nu \neq 0$  artificially couples the gradient of the solution  $u^{k+1}$  with the faceted, grid-aligned (jaggy) boundary of the narrow band domain. Let us suppose that a good estimate  $u_{\text{approx}}^{k+1}$  of the solution  $u^{k+1}$  is given. Then, we could compensate for this defect adding the above boundary with the unknown  $u^{k+1}$  replaced by the known approximation  $u_{\text{approx}}^{k+1}$  on the right hand side of the weak formula-

tion (cf. [12]). For the finite element discretization we obtain the corresponding correction vector

$$\bar{\Gamma}[U_{\text{approx}}] = \left( \int_{\partial\Omega_n} |\nabla\Phi| P[\Phi] \nabla U_{\text{approx}}^{k+1} \cdot \nu \Phi_j \, da \right)_{j \in \mathbf{I}}$$

for a given approximation  $U_{\text{approx}}^{k+1}$  of  $U^{k+1}$  on the right hand side of the modified system of linear equations

$$(\mathbf{M}[\Phi] + \tau \mathbf{L}[\Phi]) \bar{U}^{k+1} = \mathbf{M}[\Phi] \left( \tau \bar{F}[U^k] + \bar{U}^k \right) + \tau \bar{\Gamma}[U_{\text{approx}}^{k+1}].$$

Here  $\bar{F}$  is the nodal vector in  $\mathbb{R}^{\text{Iext}}$  corresponding to the right hand side  $f$ . A first possible choice for the approximation is given by the last time step, i. e. we may set  $U_{\text{approx}}^{k+1} = U^k$ .

For the discrete **mean curvature motion** we can proceed similarly. The natural boundary condition implied by the weak formulation is  $|\nabla\Phi^k|_{\epsilon}^{-1} \nabla\phi^{k+1} \cdot \nu\theta = 0$ . Hence, given an approximation  $\phi_{\text{approx}}^{k+1}$  of the unknown  $\phi^{k+1}$ , we again compensate for this defect adding the boundary integral  $\int_{\partial\Omega_n} |\nabla\phi^k|_{\epsilon}^{-1} \phi_{\text{approx}}^{k+1} \cdot \nu\theta \, da$  on the right hand side of the weak equation. For the finite element discretization we correspondingly consider the correction vector

$$\bar{\Gamma}[\Phi_{\text{approx}}^{k+1}, \Omega_n] = \left( \int_{\partial\Omega_n} |\nabla\Phi^k|_{\epsilon}^{-1} \nabla\Phi_{\text{approx}}^{k+1} \cdot \nu \Phi_j \, da \right)_{j \in \mathbf{I}}$$

and obtain the modified system to be solved:

$$(\mathbf{M}[\Phi] + \tau \mathbf{L}[\Phi]) \bar{\Phi}^{k+1} = \mathbf{M}[\Phi] \bar{U}^k + \tau \bar{\Gamma}[\Phi_{\text{approx}}^{k+1}, \Omega_n] \quad (3)$$

Again the old time step solution may serve as a first approximation of the unknown  $\Phi^{k+1}$ .

### 3.3 Transparent Dirichlet Boundary

So far, we have focused on Neumann type boundary conditions. In particular in case of mean curvature motion one might alternatively consider Dirichlet conditions. We propose boundary data which is coherent with a suitable, smooth extension of the unknown level set function  $\phi^{k+1}$  in time step  $k+1$ . Here a signed distance function from the current discrete surface is a good approximation. To present the discrete scheme in matrix vector notation, we exploit the introduced splitting of the finite element space  $\mathcal{V}_n^h = \mathcal{V}_{\text{int}}^h \oplus \mathcal{V}_{\text{bd}}^h$ . Thus, reordering degrees of freedom we obtain a splitting  $\bar{U}_n = (\bar{U}_{\text{int}}, \bar{U}_{\text{bd}})$ ,

where  $U_{\text{int}} \in \mathcal{V}_{\text{int}}^h$  and  $U_{\text{bd}} \in \mathcal{V}_{\text{bd}}^h$ . Correspondingly, we obtain a splitting of the stiffness with respect

to  $V_{\text{int}}$  and  $V_{\text{bd}}$  by  $\mathbf{L} = \begin{pmatrix} \mathbf{L}_{\text{int,int}} & \mathbf{L}_{\text{bd,int}} \\ \mathbf{L}_{\text{int,bd}} & \mathbf{L}_{\text{bd,bd}} \end{pmatrix}$ . Here

$\mathbf{L}_{\text{int,int}}$  is the actual stiffness matrix on  $V_{\text{int}}$ . Furthermore, let us introduce a trivial extension operator  $\mathbf{E} : \mathbb{R}^{\text{Iint}} \rightarrow \mathbb{R}^{\text{In}}$ ;  $\bar{U}_n \mapsto (\bar{U}_{\text{int}}, 0)$  and the corresponding restriction operator  $\mathbf{R} : \mathbb{R}^{\text{I}} \rightarrow \mathbb{R}^{\text{In}}$ ;  $\bar{U} \mapsto \bar{U}_{\text{int}}$ . Based on this notation we can rewrite  $L_{\text{int,int}} = RLE$  and hence the linear system to be solved in case of Dirichlet data  $\Phi_{\text{approx}}^{k+1}$  is  $\mathbf{R}(M[\Phi^k] + \tau \mathbf{L}[\Phi^k]) \mathbf{E} \bar{\Phi}_{\text{int}}^{k+1} = \mathbf{R}(M \bar{\Phi}^k - \tau \mathbf{L}[\bar{\Phi}^k] \bar{\Phi}_{\text{approx}}^{k+1})$ . The practical consequence is that we always work with the full matrix  $\mathbf{L}$  and do not explicitly extract  $L_{nn}$  from it. Boundary data and solution vector are stored in one vector in  $\mathbb{R}^{\text{In}}$ .

### 3.4 Solver Based on Nested Iterations

Here, we discuss the scheme for mean curvature motion, which requires special care because of the iterative update of the computational domain. The corresponding scheme for the reaction diffusion model on a fixed narrow band is a special case.

The inner iterations must modify the boundary conditions *and* the computational domain, as the solution moves toward the edge of the narrow band. Thus, the inner iterations compute the new  $\Phi^{k+1}$  relative to the old  $\Phi^k$  using Eq. 3. After each inner iteration we compute a new distance map to the zero set of the new  $\Phi^{k+1}$ . This redistancing rebuilds the DT-grid (to a specified width, as described in [28]). If the narrow band changes, we i) extend the old solution onto the new band using the signed distance transform (Eikonal equation) from the previous domain and ii) repeat the inner iteration with Dirichlet conditions using the distance field to  $\Phi^{k+1}$  on the boundary. That is, the domain extension  $\mathcal{E}[\Omega_n, \tilde{\Omega}_n] \bar{\Phi}$  from a narrow band  $\Omega_n$  onto a new band  $\tilde{\Omega}_n$  is the discrete solution of the Eikonal equation  $|\nabla\phi| = 1$  with boundary data  $\bar{\Phi}$  on the inner nodes of the band  $\Omega_n$ . If the computational domain does not change, we simply redistance, update the boundary conditions, and solve. These inner iterations repeat until the change from one iteration to the next falls below a threshold.

This scheme allows for large time steps and discrete surfaces propagating significantly outside the initial narrow band in this time step. In each time step we compute intermediate solutions  $\Phi^{k+1,j}$  and intermediate narrow band domains  $\Omega_n^{k+1,j}$ . The ex-

tension operator  $\mathcal{E}[\Omega_n^{k+1,j}, \Omega_n^{k+1,j+1}]$  ensures that the previous time step solution  $\Phi^k$  as well as the successively updated solution  $\Phi^{k+1,j}$  itself is extended onto the new band. In pseudo code notation the scheme looks as follows:

```

MeanCurvatureMotion( $\bar{\Phi}^0$ ) {
  initialize  $\Omega_n^0$ ;
  for ( $k = 1; k \leq K; k++$ ) {
     $\bar{\Phi}^{k+1,0} = \Phi^k; j = 0; \Omega_n^{k+1,0} = \Omega_n^k$ ;
    do {
      compute  $\bar{\Phi}^{k+1,j+1}$  on  $\Omega_n^{k+1,j}$  solving
         $\mathbf{R}(M[\Phi^k] + \tau\mathbf{L}[\Phi^k])\mathbf{E}\bar{\Phi}_{\text{int}}^{k+1,j+1}$ 
         $= \mathbf{R}(M[\Phi^k] - \tau\mathbf{L}[\Phi^k])\bar{\Phi}^{k+1,j}$ ;
      Define new band  $\Omega_n^{k+1,j+1}$  for  $\bar{\Phi}^{k+1,j+1}$ ;
      Apply  $\mathcal{E}[\Omega_n^{k+1,j}, \Omega_n^{k+1,j+1}]$  to  $\bar{\Phi}^{k+1,j+1}, \Phi^k$ ;
       $j = j + 1$ ;
    } while ( $|\bar{\Phi}^{k+1,j} - \bar{\Phi}^{k+1,j-1}| \geq \delta$ 
      or  $\Omega_n^{k+1,j} \neq \Omega_n^{k+1,j-1}$ );
     $\bar{\Phi}^{k+1} = \bar{\Phi}^{k+1,j}; \Omega_n^{k+1} = \Omega_n^{k+1,j}$ ;
  } }

```

The procedure in the case of Neumann conditions imposed on the boundary of the narrow band is completely analogous. We just exchange the linear system to be solved. The Dirichlet boundary conditions ensure that the new interface  $[\Phi^{k+1,j+1} = 0]$  is a subset of the current narrow band  $\Omega_n^{k+1,j}$ . This is no longer true in case of Neumann boundary conditions. Indeed, the discrete interface may cross  $\partial\Omega_n^{k+1,j}$ . Hence, before we are able to define the new band, we have to extend  $\Phi^{k+1,j+1}$  until we resolve the zero level set.

## 4 Narrow Band on the DT-Grid

In this section we describe how the proposed narrow band algorithms can be implemented on the Dynamic Tubular Grid (DT-Grid) data structure [28] in order to obtain a framework that is efficient both with regard to memory and time utilization. The DT-Grid is a data structure and set of algorithms designed for storing data of a subset of nodes or elements defined on a regular grid. Constant time access and cache performance for neighborhood operations are achieved through the careful use of *iterators*, which are used to build, store, and manipulate the solution and all of the associated matrices/vectors. We begin with a brief overview of the DT-Grid terminology required to comprehend the exposition of the implementation issues, and next we describe how to implement the proposed narrow band algorithms in the DT-Grid framework.

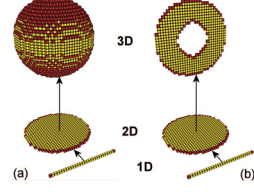


Figure 2: a) The 1D, 2D and 3D components of the DT-Grid encoding of a sphere. b) A slice of the narrow band of a DT-Grid encoding of a sphere.

### 4.1 Dynamic Tubular Grid Terminology

The nodes in the narrow band are stored in the DT-Grid in  $(x, y, z)$  lexicographic order which allows for a number of specific algorithmic constructs. In order to represent the topology of the narrow band, a 3D DT-Grid consists of 1D, 2D and 3D grid components as shown in Figure 2a. The 3D grid component consists of the nodes in the narrow band, the 2D grid component is the projection of the narrow band onto the XY-plane, and the 1D grid component is the projection of the 2D grid component onto the X-axis. For a full explanation of the DT-Grid we refer the reader to [28]. Here it is sufficient to say that each grid component has two constituents: *data* and *coord*. The *coord* constituent in the  $nD$  grid component stores the  $n$ th coordinate of the first and last node in each topologically connected component of grid points in a column of the  $nD$  grid component. These are colored red in Figure 2. As also depicted in Figure 2b, the *data<sub>1D</sub>* and *data<sub>2D</sub>* constituents link the 1D, 2D and 3D grid components together by storing indices that point to the first coordinate in a column in the *coord* constituent of the 2D and 3D grid components respectively.

We denote the *coord<sub>1D</sub>*, *coord<sub>2D</sub>*, *coord<sub>3D</sub>*, *data<sub>1D</sub>* and *data<sub>2D</sub>* constituents of the *topology* since they specify the topology of the narrow band. The *data<sub>3D</sub>* constituent contains the actual data values, e.g., a level set function, and is stored separately from the topology in a flat data vector of length equal to the number of nodes in the narrow band. Since a total (lexicographic) ordering, starting from zero, is imposed on the nodes in the narrow band, entry  $i$  in the data vector corresponds uniquely to node  $i$  in the narrow band. In fact, traversing the entries in the data vector sequentially from start to end corresponds to accessing the data of all nodes in the narrow band in lexicographic or-

der. This also means that storing multiple data items at each node in the narrow band can be done by allocating multiple separate data vectors. Entry  $i$  in each of these data vectors then identify the data stored at node  $i$  in the narrow band.

The DT-Grid utilizes the concept of *stencil iterators* to sequentially access each individual node of the narrow band and provide constant time access to the node's neighbors as defined by a stencil suited for some computational task. In particular a specific stencil iterator consists of  $M$  individual iterators, where  $M$  is the number of nodes in the stencil, and an iterator is simply a construct that sequentially visits all grid points of the narrow band in lexicographic order. Each iterator provides constant time access to the appropriate data items. Details are given in [28].

## 4.2 DT-Grid Implementation

The applications of the narrow band framework proposed in this paper require the definition of a narrow band level set function as well as a number of vectors and matrices defined over this narrow band. The vectors contain an entry for each node in the narrow band, and the matrices are defined over the cardinal product of the narrow band with itself. However, the matrices are sparse and banded due to the limited support of the nodal basis functions employed in the finite element method.

The narrow band, vectors and sparse matrices are represented as a single instance of a DT-Grid topology and a number of flat data vectors. We create a number of customized stencil iterators that compute boundary face integration on the narrow band, matrix-vector multiplication, and mass and stiffness matrix assembly with the stencil iterator framework.

The narrow band mesh used in our proposed framework is defined in terms of finite elements. Assembling the mass and stiffness matrices require an iteration over these elements, whereas the stencil iterators of the DT-Grid visit all the nodes. However, an iterator that sequentially visits all elements of the narrow band can be phrased as a stencil iterator with a stencil of eight nodes that form a finite element cell. The iterator of the lexicographically smallest node in the stencil (corner) dictates the movement of the stencil, and the stencil iterator skips a node whenever at least one of the seven remaining nodes in the stencil are outside the nar-

row band. Similarly an iterator that sequentially visits all boundary faces of the narrow band can be phrased as a stencil iterator.

## 5 Applications

We begin with the generation of a texture. Therefore we solve the initial value problem for two functions  $a, b : \mathbb{R}^+ \times \mathcal{M} \rightarrow \mathbb{R}$  where  $\mathcal{M}$  is the 0-isosurface of a level set function  $\phi$ . Reaction-diffusion equations describe a variety of biological and chemical phenomenon, but have been used in 3D graphics for the generation of interesting, natural-looking textures on surface [35, 36]. The form we use in this paper, as way of demonstrating the proposed numerical scheme, are the equations proposed by Turing:

$$\begin{aligned}\partial_t a &= c_s(\alpha - ab) + c_a \Delta_{\mathcal{M}} a \\ \partial_t b &= c_s(ab - b - \beta) + c_b \Delta_{\mathcal{M}} b\end{aligned}$$

Generally,  $c_s$ ,  $c_a$ ,  $c_b$ , and  $\alpha$  are parameters that determine the shapes, frequencies, sizes, etc. of the resulting texture (steady state) and  $\beta : \mathcal{M} \rightarrow \mathbb{R}$  is a stochastic function (e.g. generated through a pseudo-random number generator), that creates a degree of randomness in the texture.

We use the weak formulation combined with a forward difference scheme for the reaction terms and an implicit scheme for the diffusion.

Figure 3 (top) shows the solution of a reaction-diffusion equation on a 3D model of a dragon [1], which has been scan-converted to a DT-Grid, using the method in [21], with a volumetric representation of  $982 \times 695 \times 442$  grid points. With the reaction-diffusion quantities  $a$  and  $b$  and the mass and stiffness matrices, the full volumetric problem would not be solvable at this resolution on a conventional computer. The parameters, given in the caption, have been chosen to produce spots. The renderings at different levels of resolution demonstrate the difference in scale between the full model and the underlying grid.

Several authors have proposed anisotropic reaction-diffusion methods for anisotropic textures. For that purpose we replace the isotropic diffusion operator  $\Delta_{\mathcal{M}} u$  by an anisotropic version, defined by  $\tilde{\Delta}_{\mathcal{M}} u = |\nabla \phi|^{-1} \operatorname{div}(|\nabla \phi| DP[\phi] \nabla u) = |\nabla \phi|^{-1} \operatorname{div}(|\nabla \phi| (\tilde{\alpha}^2 P[v] + \tilde{\beta}^2 P[v^\perp]) P[\phi] \nabla u)$ . Figure 3 (bottom) shows solutions to the anisotropic

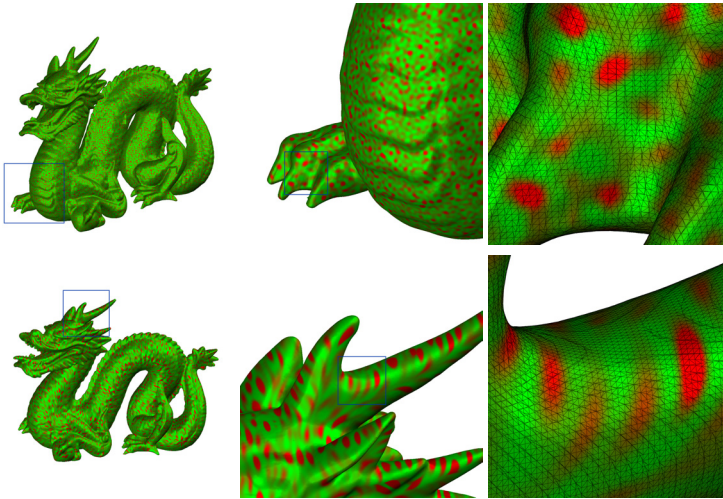


Figure 3: Results of the reaction-diffusion. (Top) Isotropic reaction-diffusion that generates a pattern with round spots after 150 iterations. (Left) A picture of the whole dragon of resolution  $982 \times 695 \times 442$ . Parameters are  $c_s = 0.05$ ,  $c_a = 2.5e - 07$ ,  $c_b = 6.3e - 08$ ,  $\alpha = 16$ , and  $\beta = 12 \pm 0.4$ , with grid spacing  $h = 0.00102$ . (Middle) A zoom to the feet region is shown, and (right) an even closer zoom to the foot overlaid with the mesh shows the fineness of the resolution. Bottom row: The same dragon surface with two zooms as result of an anisotropic reaction diffusion after 400 iterations with  $v = (0, 0, 1)^T$ , and diffusion in the orthogonal direction at one-fourth the main direction. (and otherwise the same parameters as the isotropic case).

reaction-diffusion equation with dominant diffusion in the  $z$  direction.

Another demonstration of the framework is the use of mean curvature motion on surfaces, which has been proposed for denoising (fairing) models that are derived from measured surface data. Various extensions of the approach, for both parametric and implicit surfaces, have been proposed to preserve high-curvature features. Figure 4 shows results for mean curvature motion for a scan converted model (DT-grid of size  $2471 \times 1439 \times 827$ ) of the *Lucy Statue* [1]. The full 3D grid of level-set data, stored as floats, would require almost 11 gigabytes of data, whereas the DT-Grid representation of the model requires roughly 159MB. These results demonstrate different levels of blurring rendered from different distances, in order to demonstrate the smoothing of small-scale features on very large models.

Because of the very large sizes of these models, the run times in the current version are still significant. The reaction-diffusion results required approximately 10 minutes (11.5 minutes in the anisotropic version) per timestep on an Intel Pen-

tium 3.6 GHz processor. The mean-curvature results required roughly 8 minutes per timestep for the *Lucy Statue* [1] in Figure 4 and about 5.5 minutes per timestep for the *Asian Dragon* [1] in Figure 5. However, these are very large models, which are not solvable without the very narrow band offered by the DT-grid and associated numerical schemes.

**Acknowledgements.** The narrow band approach for higher order geometric evolution problems in the PhD thesis of Marc Droske inspired this work. We thank Marc Droske and Martin Burger for many discussion and fruitful remarks on narrow band methods. We also thank Ken Museth and Ola Nilsson for permission to use their software. This work was partially supported by Deutsche Forschungsgemeinschaft through the Collaborative Research Center 611 *Singular phenomena and scaling in mathematical models*, the Hausdorff Center for Mathematics at Bonn University and the Danish Agency for Science, Technology and Innovation.

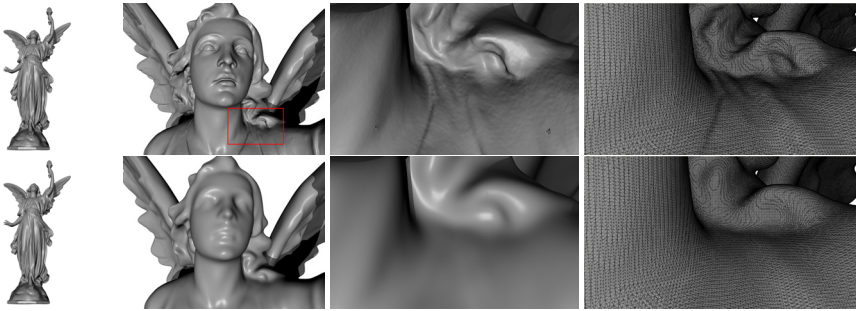


Figure 4: Results of the mean curvature motion on the Lucy statue (scan converted to a DT-grid of size  $2471 \times 1439 \times 827$ ). Initial surface (Top) and after 19 timesteps with  $\tau = h$  (Bottom). The first image shows the whole statue, the second a first zoom into the head region, the third depicts an even closer zoom to the neck and the last image shows the belonging mesh generated using the marching cubes algorithm [24].

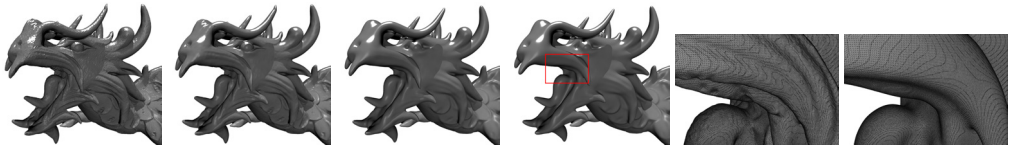


Figure 5: Evolution of the mean curvature motion on the Asian dragon surface ( $1986 \times 1323 \times 1104$ ). Here a closeup to the head with timesteps 0,6,30,44 ( $\tau = h^2$ ) is shown. The last two images show the mesh, generated using the marching cubes algorithm [24], of an even closer zoom to the tongue at timesteps 0 and 44.

## References

- [1] Stanford scanning repository. <http://graphics.stanford.edu/data/3Dscanrep/>.
- [2] D. Adalsteinsson and J. A. Sethian. A fast level set method for propagating interfaces. *Journal of Computational Physics*, 118(2):269–277, 1995.
- [3] D. Adalsteinsson and J. A. Sethian. A level set approach to a unified model for etching, deposition, and lithography III: Re-deposition, re-emission, surface diffusion, and complex simulations. *Journal of Computational Physics*, Volume 138, Issue 1:193–223, 1997.
- [4] H. Ben Ameer, M. Burger, and B. Hackl. Level set methods for geometric inverse problems in linear elasticity. *Inverse Problems*, 20(3):673–696, 2004.
- [5] M. Bertalmío, F. Ménil, L. T. Cheng, G. Sapiro, and S. Osher. *Geometric level set methods in imaging, vision, and graphics*, chapter Variational Problems and Partial Differential Equations on Implicit Surfaces: Bye Bye Triangulated Surfaces?, pages 381–397. Springer, New York, 2003.
- [6] M. Burger. A framework for the construction of level set methods for shape optimization and reconstruction. *Interfaces and Free Boundaries*, 5:301–329, 2003.
- [7] S. Chen, B. Merriman, M. Kang, R. E. Caffisch, C. Ratsch, C. L.-T., M. Gyure, R. P. Fedkiw, and S. Osher. A level set method for thin film epitaxial growth. *Journal of Computational Physics*, 167:475–500, January 2001.
- [8] D. Cremers, M. Rousson, and R. Deriche. A review of statistical approaches to level set segmentation: Integrating color, texture, motion and shape. *International Journal of Computer Vision*, 72(2):195–215, 2007.
- [9] M. de Berg. *Computational Geometry*. Springer, January 2000.
- [10] K. Deckelnick, G. Dziuk, and C. M. Elliott. Computation of geometric partial differential equations and mean curvature flow. *Acta Numerica*, 14:139–232, 2005.
- [11] M. Delfour and J. Zolésio. Oriented distance function and its evolution equation for initial sets with boundary. *SIAM Journal on Control and Optimization*, 42, No. 6:2286 – 2304, 2004.
- [12] M. Droske. *On Variational Problems and Gradient Flows in Image Processing*. Dissertation, University Duisburg, 2005.
- [13] M. Droske, B. Meyer, M. Rumpf, and C. Schaller. An adaptive level set method for medical image

- segmentation. *Lecture Notes in Computer Science*, pages 412–422, 2001.
- [14] M. Droske and M. Rumpf. A level set formulation for willmore flow. *Interfaces and Free Boundaries*, 6(3):361–378, 2004.
- [15] D. Enright, F. Losasso, and R. Fedkiw. A fast and accurate semi-lagrangian particle level set method. *Computers and Structures*, 83:479–490, Feb. 2005.
- [16] L. Evans and J. Spruck. Motion of level sets by mean curvature I. *Journal of Differential Geometry*, 33(3):635–681, 1991.
- [17] M. Garzon, D. Adalsteinsson, L. Gray, and J. A. Sethian. A coupled level set-boundary integral method for moving boundary simulations. *Interfaces and free boundaries*, 7 (3):277–302, 2005.
- [18] R. Goldenberg, R. Kimmel, E. Rivlin, and M. Rudzsky. Fast geodesic active contours. *IEEE Transactions on Image Processing*, 10:1467–75, October 2001.
- [19] P. Gomez, J. Hernandez, and J. Lopez. On the reinitialization procedure in a narrow-band locally refined level set method for interfacial flows. *International Journal for Numerical Methods in Engineering*, 63 (10):1478–1512, 2005.
- [20] J. B. Greer, A. L. Bertozzi, and G. Sapiro. Fourth order partial differential equations on general geometries. UCLA computational and applied mathematics reports, University of California Los Angeles, 2005. *Journal of Computational Physics*, Volume 216 , Issue 1, Pages: 216 - 246, Year of Publication: 2006.
- [21] B. Houston, M. Nielsen, C. Batty, O. Nilsson, and K. Museth. Hierarchical RLE Level Set: A Compact and Versatile Deformable Surface Representation. *ACM Transactions on Graphics*, 25(1):1–24, 2006.
- [22] G. Irving, E. Guendelman, F. Losasso, and R. Fedkiw. Efficient simulation of large bodies of water by coupling two and three dimensional techniques. *ACM Transactions on Graphics (SIGGRAPH)*, Aug. 2006.
- [23] A. E. Lefohn, J. M. Kniss, C. D. Hansen, and R. T. Whitaker. A streaming narrow-band algorithm: Interactive computation and visualization of level sets. *IEEE Transactions on Visualization and Computer Graphics*, Juli-August 2004, 10 (4):422–433, 2004.
- [24] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 163–169, New York, NY, USA, 1987. ACM Press.
- [25] F. Losasso, R. Fedkiw, and S. Osher. Spatially adaptive techniques for level set methods and incompressible flow. *Computers and Fluids*, 35:995–1010, 2006.
- [26] F. Losasso, F. Gibou, and R. Fedkiw. Simulating water and smoke with an octree data structure. *ACM Transactions on Graphics*, 23(3), Aug. 2004.
- [27] K. Museth, D. Breen, R. Whitaker, S. Mauch, and D. Johnson. Algorithms for interactive editing of level set models. *Computer Graphics Forum*, 24 (4):821–841, 2005.
- [28] M. B. Nielsen and K. Museth. Dynamic Tubular Grid: An efficient data structure and algorithms for high resolution level sets. *Journal of Scientific Computing*, 26(3):261–299, 2006. (submitted November, 2004; accepted January, 2005).
- [29] B. Nilsson and A. Heyden. A fast algorithm for level set-like active contours. *Pattern Recognition Letters*, 24(9-10):1331–1337, 2003.
- [30] S. J. Osher and R. P. Fedkiw. *Level Set Methods and Dynamic Implicit Surfaces*. Springer-Verlag, 2002.
- [31] S. J. Osher and J. A. Sethian. Fronts propagating with curvature dependent speed: Algorithms based on Hamilton–Jacobi formulations. *Journal of Computational Physics*, 79:12–49, 1988.
- [32] D. Peng, B. Merriman, S. Osher, H. Yhao, and M. Kang. A pde-based fast local level set method. *Journal of Computational Physics*, 155(2):410–438, 1999.
- [33] P. Smereka. Semi-implicit level set methods for curvature and surface diffusion motion. *Journal of Scientific Computing*, 19 (1 – 3):439 – 456, 2003.
- [34] J. Strain. Tree methods for moving interfaces. *Journal of Computational Physics*, 151(2):616–648, 1999.
- [35] A. M. Turing. The chemical basis of morphogenesis. *Phil. Trans. Roy. Soc. London*, B 237:37–72, 1952.
- [36] G. Turk. Re-tiling polygonal surfaces. *Computer Graphics (SIGGRAPH '92 Proceedings)*, 25, No. 4, 1991.
- [37] R. T. Whitaker. A level-set approach to 3D reconstruction from range data. *International Journal of Computer Vision*, 29(3):203–231, 1998.
- [38] M. H. Xu, P. M. Thompson, and A. W. Toga. An adaptive level set segmentation on a triangulated mesh. *IEEE Transactions on Medical imaging*, 23 (2):191–201, 2004.
- [39] G. Zheng, J. Feng, X. Jin, and Q. Peng. Adaptive level set method for mesh evolution. *Proceedings Lecture Notes in Computer Science*, 3942:1094–1097, 2006.