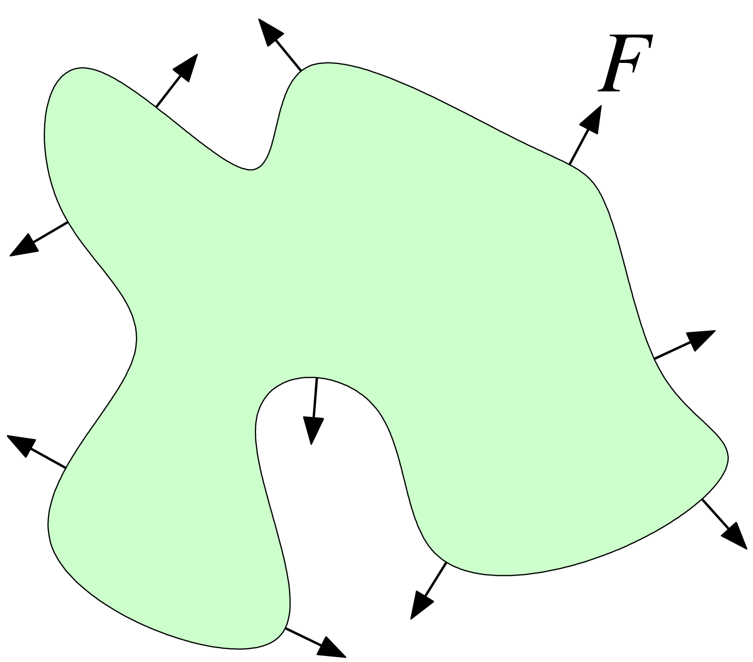
Modeling of moving interfaces

- Will talk about Levelset and Fast marching method
- Based on notes of Per-Olof Persson (see download link on course website)
- Problem statement: Moving interfaces



Propagate curve according to speed function $\boldsymbol{v} = F\boldsymbol{n}$

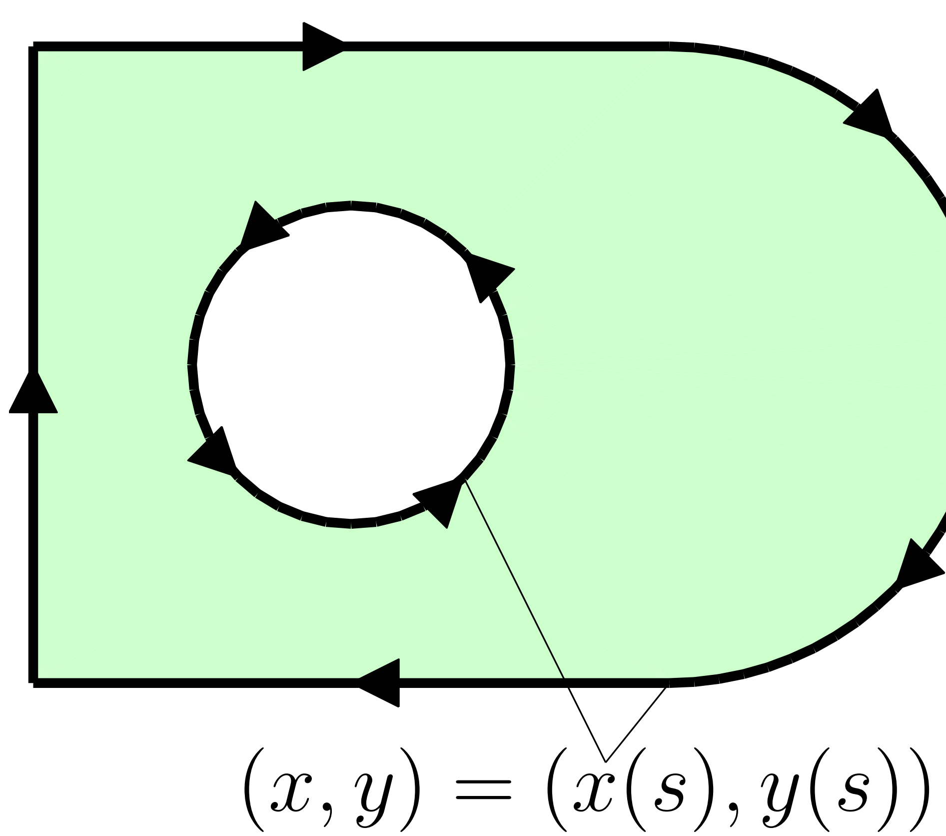
F depends on space, time, and the curve itself

Surfaces in three dimensions

Two different approaches

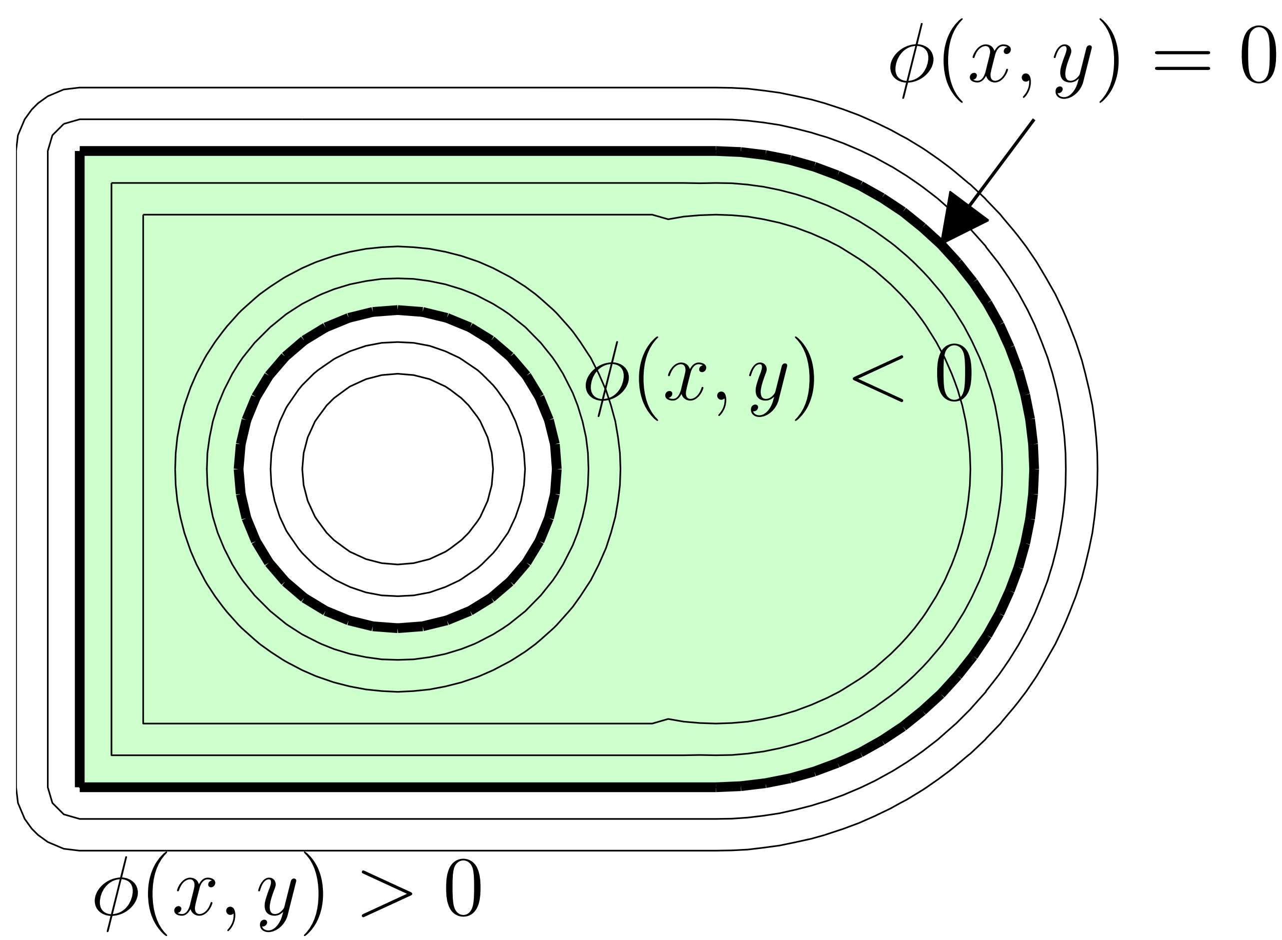
Explicit Geometry

- Parameterized boundaries



Implicit Geometry

- Boundaries given by zero level set



Explicit geometry modeling

- Simple approach: Model geometry explicitly by parametrized curves c(s) or surfaces $\Omega(s_1,s_2)$
- Discretized as a set of nodes $\boldsymbol{x}^{(i)}$, connected by edges (in 3D: "triangulated surface")
- Move nodes according to ODEs:

$$\frac{d\boldsymbol{x}^{(i)}}{dt} = \boldsymbol{v}(\boldsymbol{x}^{(i)}, t), \quad \boldsymbol{x}^{(i)}(0) = \boldsymbol{x}_0^{(i)}$$

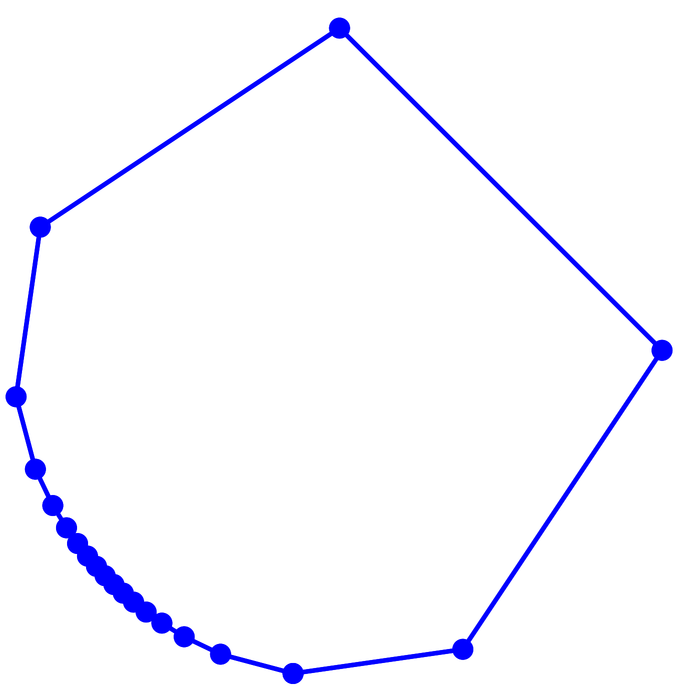
- Geometric properties (normal vector, curvature etc.) obtained by finite differences, e.g.

$$\frac{d\boldsymbol{x}^{(i)}}{ds} \approx \frac{\boldsymbol{x}^{(i+1)} - \boldsymbol{x}^{(i-1)}}{2\Delta s}$$

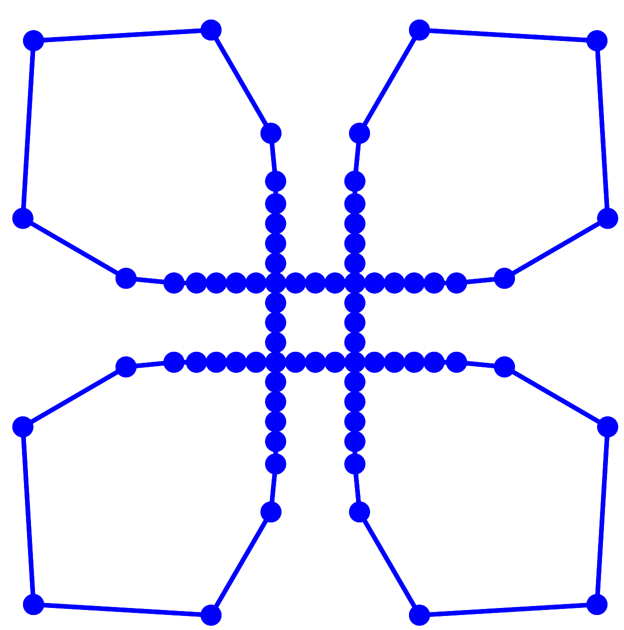
Disadvantages of explicit geometry modeling

- Shape deformation leads to distortion => need to redistribute nodes to accurately represent geometry!
- Incorrect behavior at corners
- Difficult to deal with topology changes!

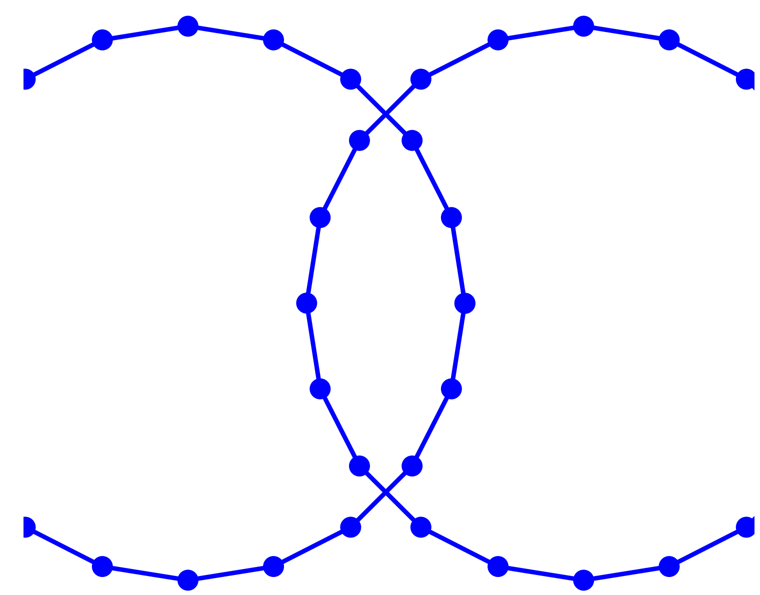
Node distribution



Sharp corners

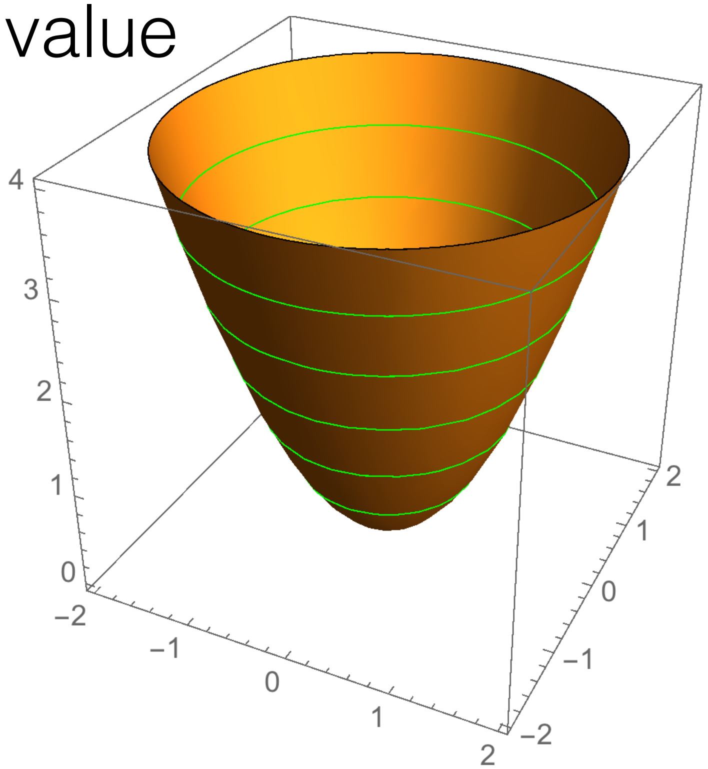


Topology changes

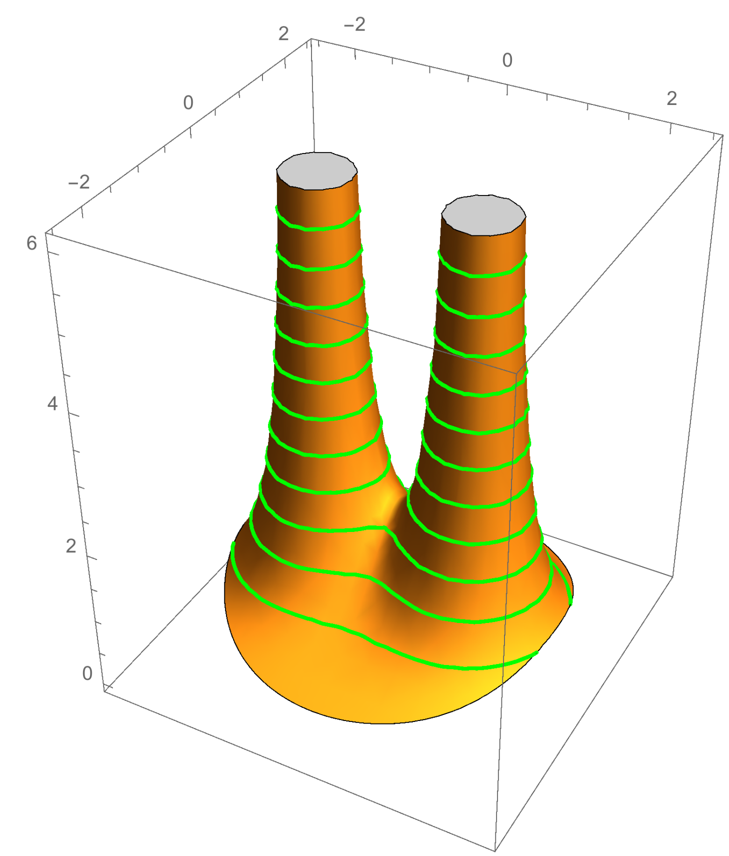


Implicit geometry modeling

- Sethian & Osher (1988)
- Represent curve by the *zero level set* of a function, $\phi(\boldsymbol{x})=\phi(x,y,\ldots)=0$
- Level set of a function $f(\boldsymbol{x})$: Values $\boldsymbol{x}$ along which f has constant value



Circle of radius r is in the level set of the function $f(\boldsymbol{x})=x^2+y^2$



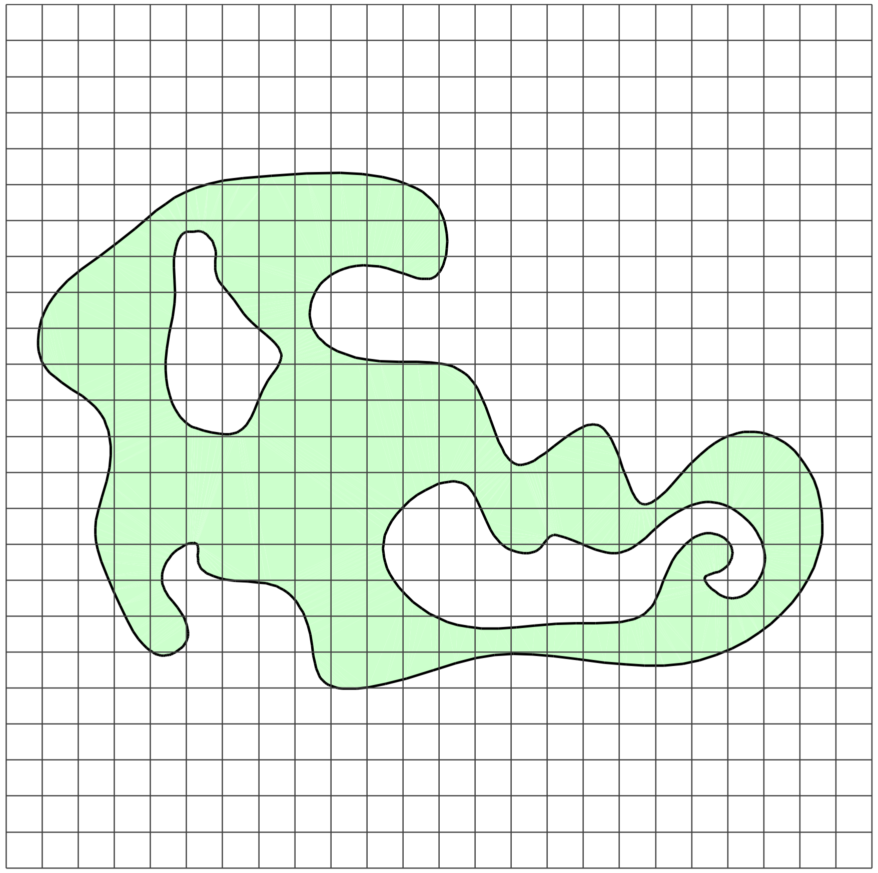
Topological changes easy to model!

Implicit geometry modeling

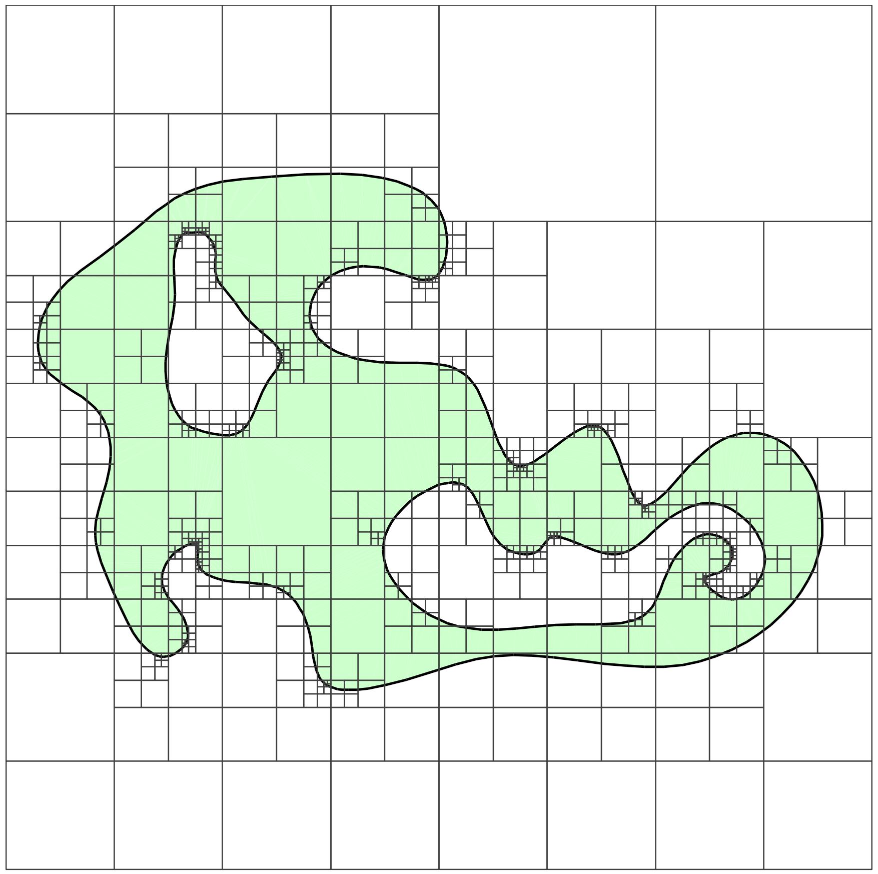
- Discretization (idea): Discretize implicit function $\phi(\boldsymbol{x})$ on background grid

- To find curve of zero level set $\phi(\boldsymbol{x})=0$, interpolate for general $\boldsymbol{x}$.

Cartesian

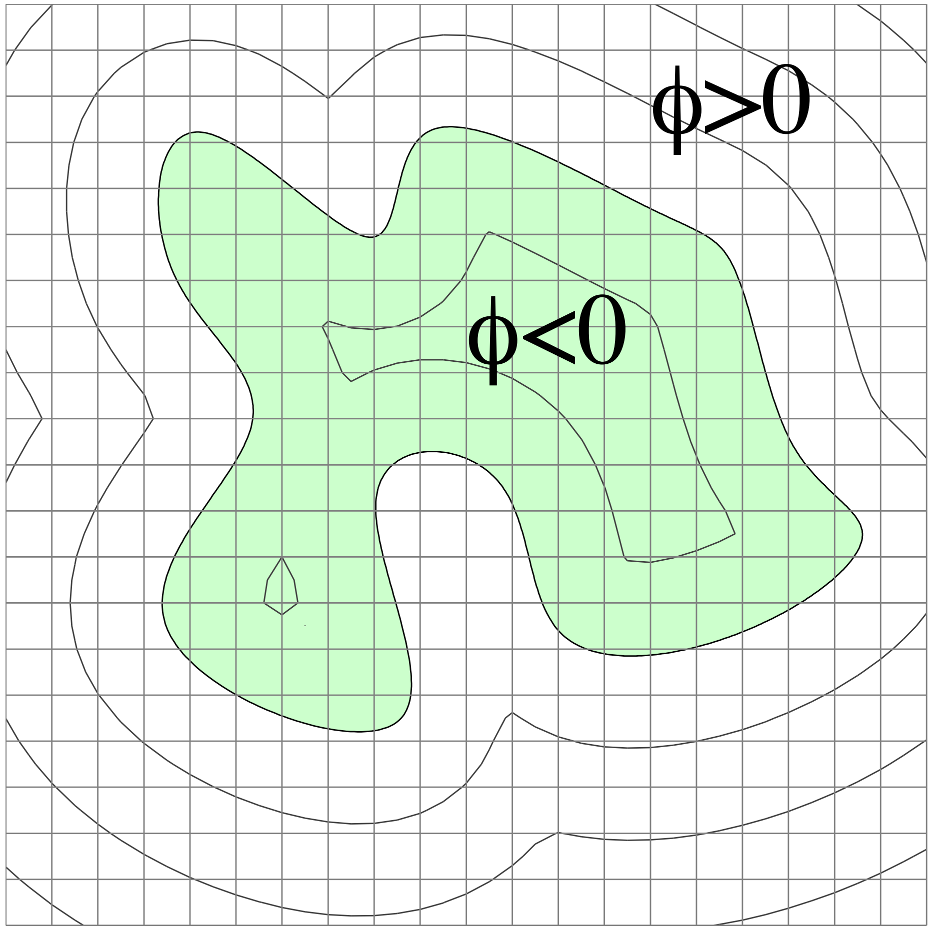


Quadtree/Octree



Implicit geometry modeling

- A special but very important choice for $\boldsymbol{\phi}(\boldsymbol{x})$ is the signed distance function. Then:
  - $|\nabla\phi| = 1$
  - $|\phi(\boldsymbol{x})|$ gives (shortest) distance from $\boldsymbol{x}$ to curve



Implicit geometry modeling

- When the curve is evolving, it's often necessary to know its geometric properties. For instance, a forest fire only moves *normal* to its current front

- Normal vector $\boldsymbol{n}$ (for general ϕ)

$$\boldsymbol{n} = \frac{\nabla\phi}{|\nabla\phi|}$$

- Curvature (for 2D curve):

$$\kappa = \nabla \cdot \frac{\nabla\phi}{|\nabla\phi|} = \frac{\phi_{xx}\phi_y^2 - 2\phi_y\phi_x\phi_{xy} + \phi_{yy}\phi_x^2}{(\phi_x^2 + \phi_y^2)^{3/2}}$$

- Material parameters (e.g. density inside vs. outside):

$$\rho(\boldsymbol{x}) = \rho_1 + (\rho_2 - \rho_1)\theta(\phi(\boldsymbol{x}))$$

where θ is the Heaviside step function (smoothened in the discretized case)

Level set equation

- Propagate ϕ (not only its zero level set!) by solving the advection equation

$$\phi_t + \boldsymbol{v} \cdot \nabla\phi = 0$$

- Since only normal part F=$\boldsymbol{v} \cdot \boldsymbol{n}$ of velocity changes shape of curve, we can simply assume $\boldsymbol{v}$=F$\boldsymbol{n}$. Then, we obtain, using

$$\boldsymbol{n} = \nabla\phi / |\nabla\phi|$$

$$\nabla\phi \cdot \nabla\phi = |\nabla\phi|^2$$

$$\boxed{\phi_t + F|\nabla\phi| = 0.}$$

level set equation

- Since F can depend on shape of curve, this is a nonlinear (hyperbolic) equation.
- Shape of curve is obtained by finding the zero level set

Discretization

- Discretize ϕ using upwind finite differences (schemes from conservation laws!)
- See online notes for details of discretization (change upwind direction depending on sign of F).
- Matlab demo

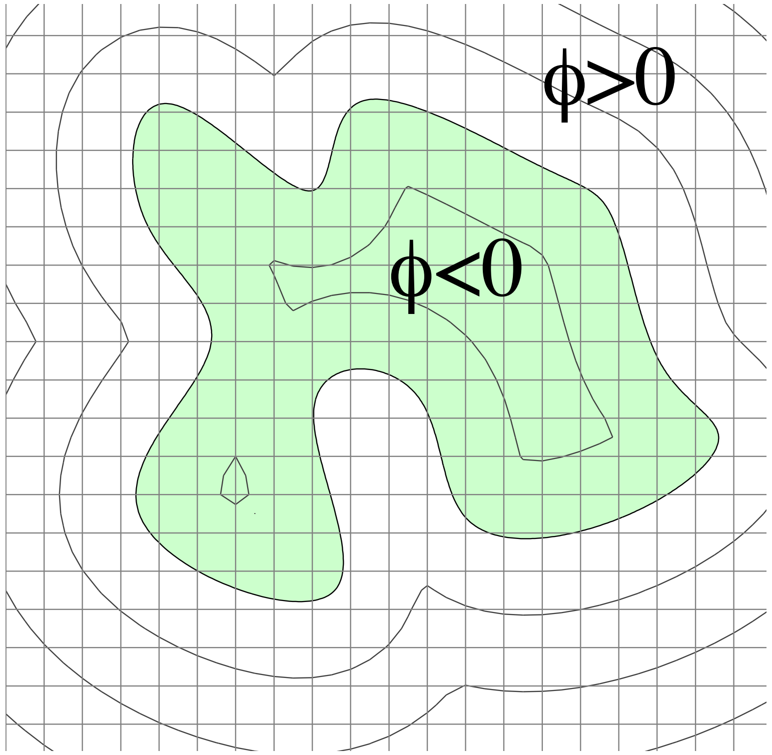
Reinitialization

- Consider the level set equation $\phi_t + F|\nabla\phi| = 0.$
- After few timesteps, $\nabla\phi$ varies from point to point.
- => requires small timestep for stable time integration!
- => Re-initialize by finding a new ϕ with the same zero level set, but with $|\nabla\phi|=1$
- Two different approaches:
  - Stop advection, instead integrate the *reinitialization equation* for a few timesteps:

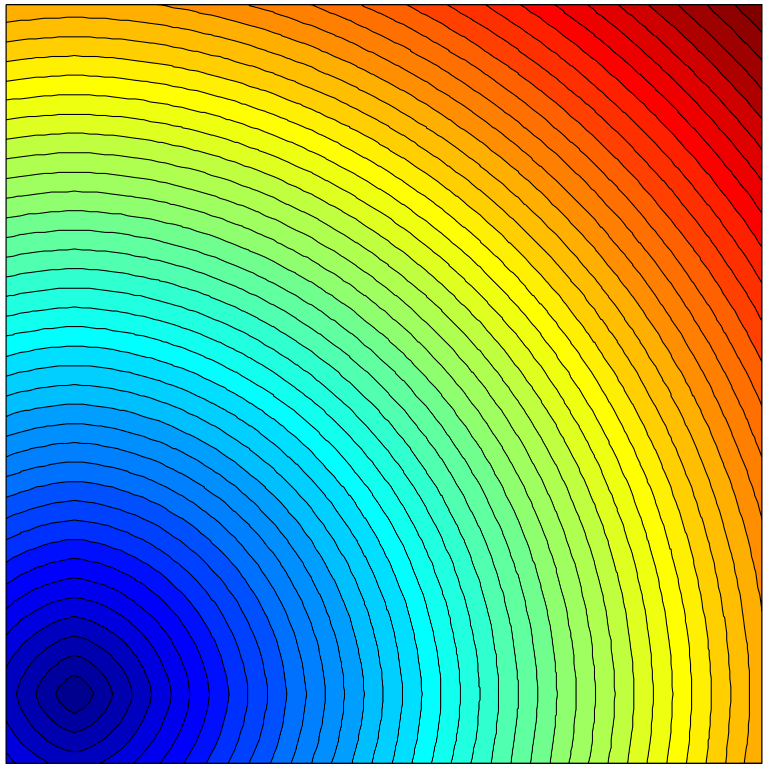
$$\phi_t + \text{sign}(\phi)(|\nabla\phi| - 1) = 0$$

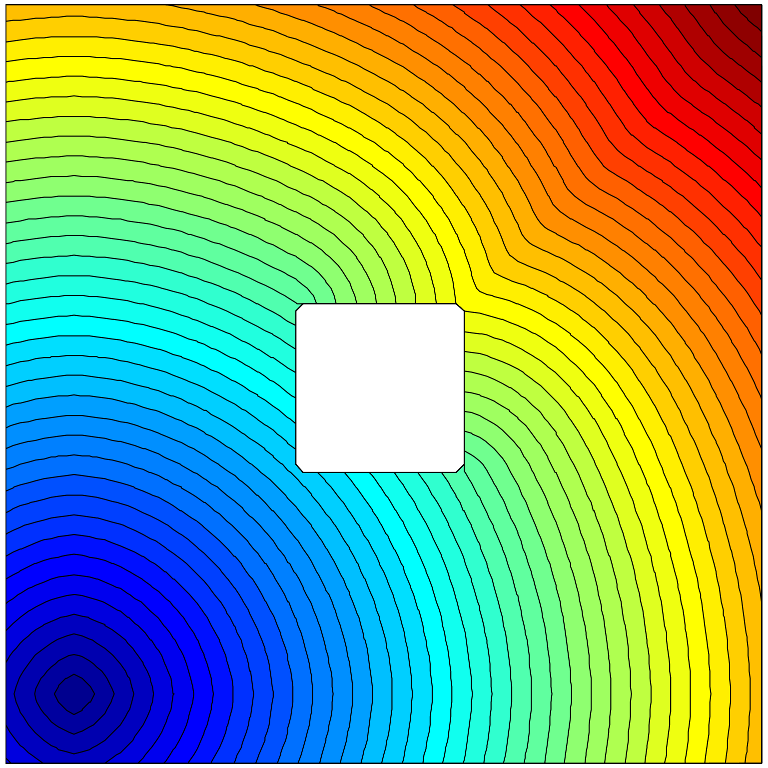
  - Find a completely new ϕ with $|\nabla\phi|=1$: Signed distance function!
    => Need to find signed distance of each mesh point from current curve, i.e. from $\phi=0$ (e.g. using the *fast marching method*)

Fast marching method



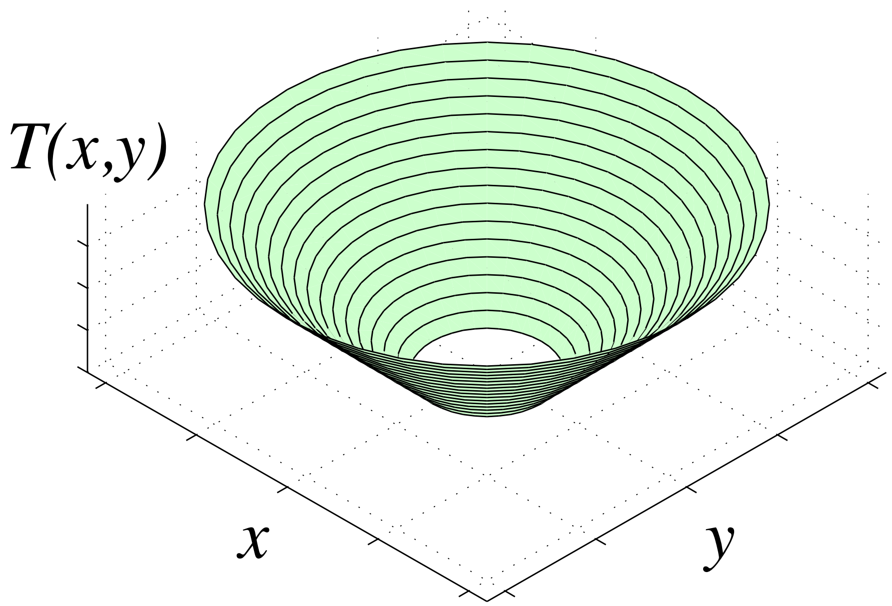
- Problem statement: Find distance of grid points from curve ϕ=0.
- Equivalent problem: Let the curve propagate in normal direction with uniform speed F>0 (outward), or F<0 (inward), and find the arrival time of the front for each grid point.
- Arrival time calculation is a very generic problem:





Fast marching method

- Problem statement:
  Find distance of grid points from curve ϕ=0 <=> arrival time T($\boldsymbol{x}$) for front propagating with speed F>0
- T($\boldsymbol{x}$) is the time needed to reach $\boldsymbol{x}$ from initial curve Γ



T(x,y) for Γ a circle

- Note: Since time * velocity = distance, we obtain the Eikonal equation

$$|\nabla T|F = 1, \quad T = 0 \text{ on } \Gamma$$

This is now an boundary value problem, not an initial value problem! In other words, the fast marching method actually solves an Eikonal equation…

- To obtain the distance function, we are interested in constant F=1

Fast marching method

- Rough sketch of method: First, identify curve and fixed values. Mark nearest neighbors as candidates and everything else as far-distance. Then:

1. Let Trial be the candidate point with smallest T.
2. Mark Trial as a fixed value (meaning T is known)
3. Move all far-distance neighbors of Trial as Candidates
4. Compute T for all neighbors of Trial

   this is done by solving the Eikonal equation using a FD scheme
   $$|\nabla T|F = 1$$
5. Repeat 1-4 until all grid points are fixed