Such equations appear in many applications. As pointed out in [111, 108], there is a close relation between HJ equations and hyperbolic conservation laws that in $\mathbb{R}^d$ take the form

$$u_t + \sum_{i=1}^{d} f_i(u)_{x_i} = 0, \qquad u(x,0) = u_0(x).$$

Actually, for the one-dimensional case ($d = 1$), the HJ equation is equivalent to the conservation law for $u = \phi_x$. This equivalence disappears when considering more than one dimension: $H(\cdot)$ is often a non linear function of its arguments ϕ_{x_i} and obviously does not have to be separable, so that we can no longer use the integration relation between ϕ and u. However, numerical methodologies that were successfully used for solving hyperbolic conservation laws are still useful for HJ equations.

2.4 Entropy Condition and Vanishing Viscosity

In general, the weak solution for a conservation law is not unique and an additional condition is needed to select the *physically correct* or *vanishing viscosity* solution. This additional condition is referred to as the *entropy condition.*

Consider the 'viscous' conservation law:

$$u_t + (H(u))_x = \epsilon\, u_{xx}.$$

The effect of the viscosity $\epsilon\, u_{xx}$ is to smear (or diffuse) the discontinuities, thereby, ensuring a unique smooth solution. Introducing the viscosity term turns the equation from a hyperbolic into a parabolic type, for which there always exists a unique smooth solution for $t > 0$. The limit of this solution as $\epsilon \to 0$ is known as the 'vanishing viscosity' solution. The entropy condition selects the weak solution of the conservation law

$$u_t + (H(u))_x = 0 \qquad u(x,0) = u_0(x),$$

that is the vanishing viscosity solution for u_0. Therefore, the vanishing viscosity solution is sometimes referred to as the entropy solution.

Satisfying the entropy condition guarantees meaningful and unique weak solutions. Moreover, there is a close duality between the entropy condition and the Eulerian formulation to curve evolution. Actually, the search for an entropy condition for the case of curve evolution [126] eventually led Osher and Sethian to the Eulerian formulation [111] that will be described in the following section.

2.5 The Eulerian Formulation

The Eulerian formulation for planar curve evolution first was proposed by Osher and Sethian in [111]. This formulation allows the developments of efficient and stable numerical schemes in which topological changes of the propagating curve are automatically handled.

Consider the family of planar curves given by $\mathcal{C}(s,t) : [0, L(t)] \times [0, T) \to \mathbb{R}^2$, where s is the arclength of the curve $\mathcal{C}$ at time t. Let the curve evolution equation describing the differential change of the curve in time be given by

$$\mathcal{C}_t = \mathcal{V}, \qquad \mathcal{C}(s,0) = \mathcal{C}_0(s),$$

where $\mathcal{V}(s,t) : [0, L] \times [0, T) \to \mathbb{R}^2$, is some velocity vector field that changes smoothly along the curve. The same evolution may be equivalently written by considering the normal $\mathcal{N} = \mathcal{C}_{ss}/|\mathcal{C}_{ss}|$ and tangential $\mathcal{T} = \mathcal{C}_s$ components of the velocity $\mathcal{V}$ along the curve:

$$\mathcal{C}_t = \langle \mathcal{V}, \mathcal{N} \rangle \mathcal{N} + \langle \mathcal{V}, \mathcal{T} \rangle \mathcal{T}, \qquad \mathcal{C}(s,0) = \mathcal{C}_0(s).$$

A basic result from the theory of curve evolution is that the geometric shape of the curve (often referred to as the trace or the image of the planar curve) is only affected by the normal component of the velocity. The tangential component affects only the parameterization, and not the geometric shape of the propagating curve:

Lemma 2.1 [Epstein-Gage [43]]: *The family of curves $\mathcal{C}(p,t)$ that solve the evolution rule*

$$\mathcal{C}_t = V_N \mathcal{N} + V_T \mathcal{T},$$

where V_N does not depend on the parameterization of the curve[1]*, can be converted into the solution of*

$$\mathcal{C}_t = V_N \mathcal{N}.$$

Proof. Given $\mathcal{C}(p,t) : S^1 \times [0, T) \to \mathbb{R}^2$ as the original family of curves, let $p = p(\omega, \tau)$ and $t = \tau$ with $\partial p / \partial \omega > 0$ be a reparametrization. By the chain rule

$$\begin{aligned} \mathcal{C}_\tau &= \mathcal{C}_\omega \omega_\tau + \mathcal{C}_t t_\tau \\ &= \mathcal{C}_\omega \omega_\tau + \mathcal{C}_t. \end{aligned}$$

For the arclength parametrization s we have that

$$\begin{aligned} \mathcal{C}_\omega &= \mathcal{C}_s s_\omega \\ &= \mathcal{T} s_\omega. \end{aligned}$$

Using these two expressions we calculate

$$\begin{aligned} \mathcal{C}_\tau &= \mathcal{C}_\omega \omega_\tau + \mathcal{C}_t \\ &= \mathcal{T} s_\omega \omega_\tau + V_T \mathcal{T} + V_N \mathcal{N} \\ &= (V_T + s_\omega \omega_\tau) \mathcal{T} + V_N \mathcal{N}, \end{aligned}$$

[1] V_N is thus called an 'intrinsic' or 'geometric' quantity

Choosing the parameter ω that solves the O.D.E.:

$$V_T + s_\omega \omega_\tau = 0,$$

and recalling the selection $t = \tau$ we arrive at:

$$\mathcal{C}_t = V_N \mathcal{N}.$$

$\blacksquare$

Therefore, since our interest is the shape of the curve we can consider the 'Lagrangian' form of the curve evolution:

$$\mathcal{C}_t = \langle \mathcal{V}, \mathcal{N} \rangle \mathcal{N}, \qquad \mathcal{C}(s,0) = \mathcal{C}_0(s),$$

and for $V_N = \langle \mathcal{V}, \mathcal{N} \rangle$,

$$\mathcal{C}_t = V_N \mathcal{N}, \qquad \mathcal{C}(s,0) = \mathcal{C}_0(s). \tag{2.1}$$

While implementing the evolution given by the Lagrangian formulation one should handle topological changes in the evolving curve by external procedures. Such a procedure should monitor the process and detect possible mergings and splittings of the curve. It was also shown [126, 111, 127] that such implementations are very sensitive to the formation of high curvature and sharp corners. The problems appear due to a time varying coordinate system (s,t) of the direct curve representation (where s is the parameterization, and t - the time). An initial smooth curve can develop curvature singularities. The question is how to continue the evolution after singularities appear. The natural way is to choose the solution which agrees with the *Huygens principle* [126]. Viewing the curve as the front of a burning flame, this solution states that *once a particle is burnt, it cannot be re–ignited* [127]. It can also be proved that from all the *weak* solutions of the Lagrangian formulation, the one derived from the Huygens principle is unique, and can be obtained by a constraint denoted as the "entropy condition for curve evolution [111]".

In order to overcome these difficulties the 'Eulerian formulation' was proposed in [111].

Let $\phi(x,y,t) : \mathbb{R}^2 \times [0,T) \rightarrow \mathbb{R}$ be an implicit representation of the curve $\mathcal{C}(s,t)$, so that the zero level set $\phi(x,y,t) = 0$ is the set of points constructing the curve $\mathcal{C}(s,t)$. In other words, the trace of the curve $\mathcal{C}$ at time t is given by the zero level set of the function ϕ at time t:

$$\mathcal{C}(t) = \phi^{-1}(0).$$

The demand of $\mathcal{C}$ being the zero level set is arbitrary, and actually any other level set may serve the same purpose. The problem is how to evolve the ϕ function in time so that its zero level set tracks the time varying curve $\mathcal{C}(t)$.

Denote by $\nabla \equiv (\partial/\partial x, \partial/\partial y)$ the gradient operator. Then, from basic calculus, we have

Lemma 2.2 *The planar unit normal of the curve* $\mathcal{C} = \phi^{-1}(c)$*, where* c *is an arbitrary constant selecting the level set, is given by* $\mathcal{N} = \nabla\phi/|\nabla\phi|$.

Proof. Let s be the arclength parameter of $\mathcal{C}$. Then, along the equal height contour $\mathcal{C}$ the change of ϕ is zero:

$$\phi_s = 0 = \phi_x x_s + \phi_y y_s.$$

This expression $\langle \nabla\phi, \mathcal{C}_s \rangle = 0$, determines that $\nabla\phi$ is orthogonal to $\mathcal{C}_s = \mathcal{T}$. ■

According to the chain rule,

$$\phi_t = \phi_x x_t + \phi_y y_t.$$

Then, the above equation may be written as:

$$\begin{aligned}
\phi_t &= \langle \nabla\phi, \mathcal{C}_t \rangle \\
&= \langle \nabla\phi, V_N \mathcal{N} \rangle \\
&= \langle \nabla\phi, V_N \frac{\nabla\phi}{|\nabla\phi|} \rangle \\
&= V_N \langle \nabla\phi, \frac{\nabla\phi}{|\nabla\phi|} \rangle \\
&= V_N |\nabla\phi|,
\end{aligned}$$

which is the Eulerian formulation for curve evolution. Given **any** smooth function $\phi_0(x,y)$ such that $\phi_0^{-1}(0) = \mathcal{C}_0$ we can rewrite the last result

$$\phi_t = V_N |\nabla\phi|, \qquad \phi(x,y,0) = \phi_0(x,y), \tag{2.2}$$

which is a Hamilton-Jacobi type of equation. This formulation of planar curve evolution processes frees us from the need to take care of the possible topological changes in the propagating curve. Sethian [127] named the above *Eulerian formulation* for front propagation, because it is written in terms of a fixed coordinate system.

The normal component V_N may be any smooth scalar function. An important observation is that any geometric property of the curve $\mathcal{C}$ may be computed from its implicit representation ϕ. The curvature, for example, plays an important role in many applications:

Lemma 2.3 *The curvature* κ *of the planar curve* $\mathcal{C} = \phi^{-1}(c)$ *is given by*

$$\kappa = -\frac{\phi_{xx}\phi_y^2 - 2\phi_x\phi_y\phi_{xy} + \phi_{yy}\phi_x^2}{(\phi_x^2 + \phi_y^2)^{3/2}} \tag{2.3}$$

Proof. Along $\mathcal{C}$, the function ϕ does not change its values. Therefore, $\partial^n\phi/\partial s^n = 0$, for any n. Particularly, for $n = 2$,

$$\begin{aligned}
0 &= \frac{\partial^2\phi}{\partial s^2} \\
&= \frac{\partial}{\partial s}(\phi_x x_s + \phi_y y_s) \\
&= \phi_{xx}x_s^2 + 2\phi_{xy}x_s y_s + \phi_{yy}y_s^2 + \phi_x x_{ss} + \phi_y y_{ss} \\
&= \phi_{xx}x_s^2 + 2\phi_{xy}x_s y_s + \phi_{yy}y_s^2 + \langle\nabla\phi, \mathcal{C}_{ss}\rangle. \qquad (2.4)
\end{aligned}$$

Recall that $\mathcal{N} = (-y_s, x_s) = \nabla\phi/|\nabla\phi|$, and that by definition $C_{ss} = (x_{ss}, y_{ss}) = \kappa\mathcal{N}$. Or explicitly

$$\left\{\begin{array}{lcl}
y_s &=& -\frac{\phi_x}{\sqrt{\phi_x^2+\phi_y^2}} \\
x_s &=& \frac{\phi_y}{\sqrt{\phi_x^2+\phi_y^2}},
\end{array}\right.$$

and

$$\left\{\begin{array}{lcl}
x_{ss} &=& \kappa\frac{\phi_x}{\sqrt{\phi_x^2+\phi_y^2}} \\
y_{ss} &=& \kappa\frac{\phi_y}{\sqrt{\phi_x^2+\phi_y^2}}.
\end{array}\right.$$

Introducing these two expressions into Equation (2.4) we conclude that

$$\begin{aligned}
0 &= \frac{\phi_{xx}\phi_y^2 - 2\phi_x\phi_y\phi_{xy} + \phi_{yy}\phi_x^2}{|\nabla\phi|^2} + \langle\nabla\phi, \mathcal{C}_{ss}\rangle \\
&= \frac{\phi_{xx}\phi_y^2 - 2\phi_x\phi_y\phi_{xy} + \phi_{yy}\phi_x^2}{|\nabla\phi|^2} + |\nabla\phi|\kappa.
\end{aligned}$$

■

2.6 Numerical Methodologies

We have seen that the curve evolution may be presented as a Hamilton-Jacobi equation. In one dimension, the HJ equation coincides with hyperbolic conservation laws. This close relation can be used to construct numerical schemes for our problems. Similarly to the continuous case, a finite difference method is in *conservation form* if it can be written in the form

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = -\frac{(g_{j+1/2}^n - g_{j-1/2}^n)}{\Delta x}, \qquad (2.5)$$

where $g_{j+1/2} = g(u_{j-p+1}, ..., u_{j+q+1})$ is called a *numerical flux*, is Lipschitz[2] and *consistent* (satisfies the consistency requirement)

$$g(u, ..., u) = H(u),$$

i.e. setting all the $p+q$ variables of the numerical flux function to u, the numerical flux becomes identical to the continuous flux.

Theorem 2.1 *Suppose that the solution $u(x, n\Delta t)$ of a finite difference method in conservation form converges to some function $v(x,t)$ as Δx and Δt approach zero. Then $v(x,t)$ is a weak solution of the continuous equation.*

The proof may be found in [133] page 286.

A numerical scheme is *monotone* if the function $F(u_{j-p}^n, ..., u_{j+q+1}^n)$ that defines the scheme

$$u_j^{n+1} = F(u_{j-p}^n, ..., u_{j+q+1}^n),$$

or equivalently (for a conservation form):

$$u_j^{n+1} = F(u_{j-p}^n, ..., u_{j+q+1}^n) = u_j^n - \frac{\Delta t}{\Delta x}(g_{j+1/2}^n - g_{j-1/2}^n),$$

is a non-decreasing function of all its $(p+q+1)$ arguments, that is,

$$F_j \equiv \frac{\partial F}{\partial u_{j+i}^n} \geq 0 \qquad \text{for} \qquad -p \leq i \leq q+1.$$

Theorem 2.2 [Harten-Hyman-Lax [54]] *Assume that the solution of a monotone finite difference method u_j^n that has a conservation form converges to some function $v(x,t)$ as Δx and Δt approach zero with $\Delta t/\Delta x$ fixed. Then $v(x,t)$ is a weak solution and the entropy condition is satisfied at all discontinuities of v.*

The *local truncation error* measures how well the finite difference method models the differential equation locally. It is defined by replacing the approximated solution in the difference method by the true solution $u(j\Delta x, n\Delta t)$. Let us replace for example u_j^{n+1} by the Taylor series about $u(x,t)$, *i.e.* $u + \Delta t u_t + (1/2)\Delta t^2 u_{tt} + ...$ We do the same for the spatial derivatives, and arrive at the error bound that is a function of Δx and Δt. A first order accurate scheme is a differential method with local truncation error (for $\Delta t/\Delta x = \text{constant}$) of $\mathcal{O}(\Delta t)$ (as $\Delta t \to 0$).

Satisfying the entropy condition is indeed a desired quality however these schemes are limited by the following theorem:

Theorem 2.3 *A monotone finite difference method in conservation form is first order accurate.*

[2] Observe that the numerical flux is a function $g : \mathbb{R}^{p+q} \to \mathbb{R}$, and thus maybe restricted as such to be Lipschitz.

For proof see [133] page 299.

Getting higher order accuracy for such equations by relaxing the monotonicity demand may be found in [111, 108]. One idea leads to the essentially non-oscillating (ENO) schemes, in which an adaptive stencil is used between the discontinuities. Thereby, piecewise smooth data may be handled with high accuracy.

The relation between the Hamilton-Jacobi equations and the conservation laws may be used to design first order finite difference methods for the HJ equations[111]. The relation between $\phi(x,t)$, the solution of an HJ equation, and $u(x,t)$, the solution of the corresponding differential conservation law that describes the change of u ='the slope of ϕ', for the one dimensional case, is given by integration, *i.e.* $\phi(x,t) = \int_{-\infty}^{x} u(\tilde{x},t)d\tilde{x}$. Thus by integrating over the monotone numerical scheme (and shifting form $j+1/2$ to j) we arrive at

$$\Phi_j^{n+1} = \Phi_j^n - \Delta t\, g(D_-\Phi_{j-p+1}^n, ..., D_+\Phi_{j+q}^n).$$

Definition 2.1 *An upwind finite difference scheme is defined so that*

$$g_{j+1/2} = \begin{cases} f(u_j) & H' > 0 \\ f(u_{j+1}) & H' < 0. \end{cases}$$

An upwind numerical flux in a conservation form results in a monotone method. The upwind monotone HJ scheme for the special case where

$$H(u) = h(u^2)$$

with $h'(u) < 0$, was introduced in [111]:

$$g_{HJ}(u_j^n, u_{j+1}^n) = h((\min(u_j^n, 0))^2 + (\max(u_{j+1}^n, 0))^2).$$

This scheme has the advantage of being easy to generalize to more than one dimension.

Motivated by the theory of mathematical morphology [13], we have found the following scheme to have same qualities (being upwind) as the HJ scheme under the same restrictions ($h'(u) < 0$):

$$g_M(u_j^n, u_{j+1}^n) = h((\max(-u_j^n, u_{j+1}^n, 0))^2).$$

The only difference between the g_{HJ} and the g_M is that at points where u changes form negative to positive magnitude, g_M selects the maximum between $(u_j)^2$ and $(u_{j+1})^2$, while g_{HJ} selects $(u_j)^2 + (u_{j+1})^2$. We have found that the g_M numerical flux produces better results in some cases.

Having the numerical flux, or numerical Hamiltonian in the HJ context, we can write the numerical approximation of the Hamilton-Jacobi formulation as

$$\Phi_j^{n+1} = \Phi_j^n - \Delta t\, g(D_-\Phi_j^n, D_+\Phi_j^n). \tag{2.6}$$

As we noted before, in some cases the requirements on the numerical scheme are relaxed to achieve higher order accuracy as well as handling more complicated flux functions. One useful example for our case is partial derivatives that are approximated by *slope limiters*. The idea is to keep the total variations of the evolving data under control, leading to the TVD (total variation diminishing) methods [87]. By selecting the smallest slope between the forward and backward derivatives, the estimated slope of the data is always limited by the continuous data. A simple example of a first order slope limiter is given by the *minmod* operation. Define the *minmod* selection function as

$$minmod\{a,b\} = \begin{cases} \text{sign}(a)\min(|a|,|b|) & \text{if } \; ab > 0 \\ 0 & \text{otherwise} \end{cases}$$

This can be used to approximate ϕ_x by the *minmod* finite derivative

$$\phi_x|_{x=i\Delta x} \approx minmod(D_+^x\Phi_i, D_-^x\Phi_i).$$

2.7 The CFL Condition

One of the earliest observations in the field of finite difference schemes was made by Courant, Friedrichs, and Lewy in [34, 35]. They observed that a necessary stability condition for any numerical scheme is that the *domain of dependence* of each point in the domain of the numerical scheme should include the domain of dependence of the PDE itself. This condition is necessary, but not necessarily sufficient, for the stability of the scheme. For hyperbolic PDEs the domain of dependence is known to be bounded.

Considering the $1D$ case, when refining the discretization grid by letting $\Delta x \to 0$ and $\Delta t \to 0$, the ratio $\Delta t/\Delta x$ should be limited. This limit, known as the CFL number or the Courant number, is determined by the maximal possible flow of information. The flow lines of the information obviously depend on the specific initial data and are known as the *characteristics* of the PDE. Collisions of characteristics form 'shocks' in the solution and therefore require additional conditions which determine how to handle the propagation of such a shock. A propagating shock in time may thus be defined as a sequence of colliding characteristics where the entropy condition defines the speed of this propagation.

As a simple example consider the $1D$ conservation law in which the the point $(x = \tilde{x}, t = \tilde{t})$ in the PDE domain can be influenced by the data bounded by the triangle $(x_0,0),(\tilde{x},\tilde{t}),(x_1,0)$. This means that any information at the interval (x_0,x_1) of the initial condition u_0 may influence the result at $(\tilde{x},\tilde{t})$, namely $u(\tilde{x},\tilde{t})$. Similarly, it may be asserted that the point $(\tilde{x},\tilde{t})$ is in the *domain of influence* of each point in the interval (x_0,x_1). Therefore, any finite difference approximating the PDE should take this fact into consideration, by limiting the ratio $\Delta t/\Delta x$. Taking this to a limit, for $u_t + (H(u))_x = 0$ the CFL restriction for a 3-point scheme can be shown to be

$$1 \geq \frac{\Delta t}{\Delta x}|H'|,$$

and in our case, where we have actually integrated a 3-point of Δx scheme of a conservation law into a 3-point HJ equation we arrive at the same CFL restriction.

As pointed out, the g_{HJ} and the g_M numerical flows may be easily generalized to several dimensions. The generalization is straightforward and for the specific case of $H(u,v) = f(u^2, v^2)$ we get the following form

$$g_M(u_i^n, u_{i+1}^n, v_j^n, v_{j+1}^n) = h((\max(-u_i^n, u_{i+1}^n, 0))^2, (\max(-v_j^n, v_{j+1}^n, 0))^2).$$

This yields an upwind monotone scheme with a CFL restriction of

$$1 \geq \left(\frac{\Delta t}{\Delta x}|H_u| + \frac{\Delta t}{\Delta y}|H_v|\right).$$

Consider the simple example of a planar curve propagating with constant velocity along its normal that obeys the following evolution law,

$$\mathcal{C}_t = \mathcal{N}.$$

It is easy to see that since $V_N = 1$ the Eulerian formulation for this case is

$$\phi_t = |\nabla \phi|,$$

thus, $H(u,v) = \sqrt{u^2+v^2}$. For the simple selection of $\Delta x = \Delta y = 1$, we arrive at the CFL restriction:

$$\Delta t \leq \frac{1}{\sqrt{2}}.$$

The following example presents offsets produced by two schemes, one with $\Delta t < 1/\sqrt{2}$, satisfying the CFL restriction, and another with $\Delta t > 1/\sqrt{2}$, violating the CFL restriction. The Eulerian formulation is implemented by the following numerical approximation:

$$\Phi_{ij}^{n+1} = \Phi_{ij}^n + \Delta t \sqrt{\left(\max(-D_-^x \Phi_{ij}^n, D_+^x \Phi_{ij}^n, 0)\right)^2 + \left(\max(-D_-^y \Phi_{ij}^n, D_+^y \Phi_{ij}^n, 0)\right)^2}.$$

Figure 2.1 is the data image I, given as initial condition to the evolution equation ($\Phi^0 = I$). The evolution of Φ in time for the scheme with $\Delta t = 0.7 < 1/\sqrt{2}$ is presented in Figure 2.2. The offsetting results of the two schemes with $\Delta t = 0.7$ and $\Delta t = 0.8$ are presented in Figure 2.3 on the left and right columns, respectively. The gray levels correspond to the height values of Φ_{ij}^n on the grid. Histogram equalization is applied to the last evolution step in order to strengthen the fact that violating the CFL restriction results in perturbations of the Φ function. The bottom row in Figure 2.3 shows the unstable result on the right compared with the stable one on the left. The zero level sets (every two time steps) are drawn as white contours on the original image (upper row). Since we have chosen only few iterations and selected a time step that is close to the CFL condition, the zero level sets are only slightly affected. More iterations or a larger time step will amplify the noise and distort the smoothness of the zero level sets.

Figure 2.1: The original image which is an implicit representation of the contours describing the outline of the letters in the image

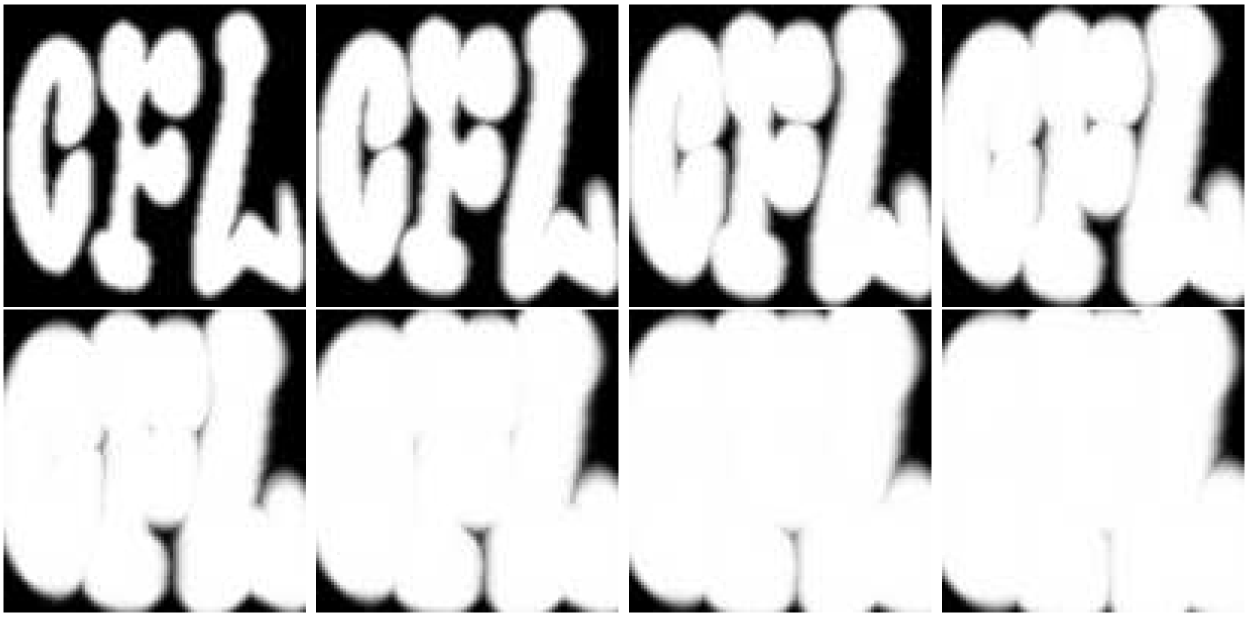

Figure 2.2: The images of the iterations (every two time steps) Φ^2 to Φ^{16}, left to right, upper to bottom, for the scheme with non-violating (satisfying the CFL restriction) time step $\Delta t = 0.7$.

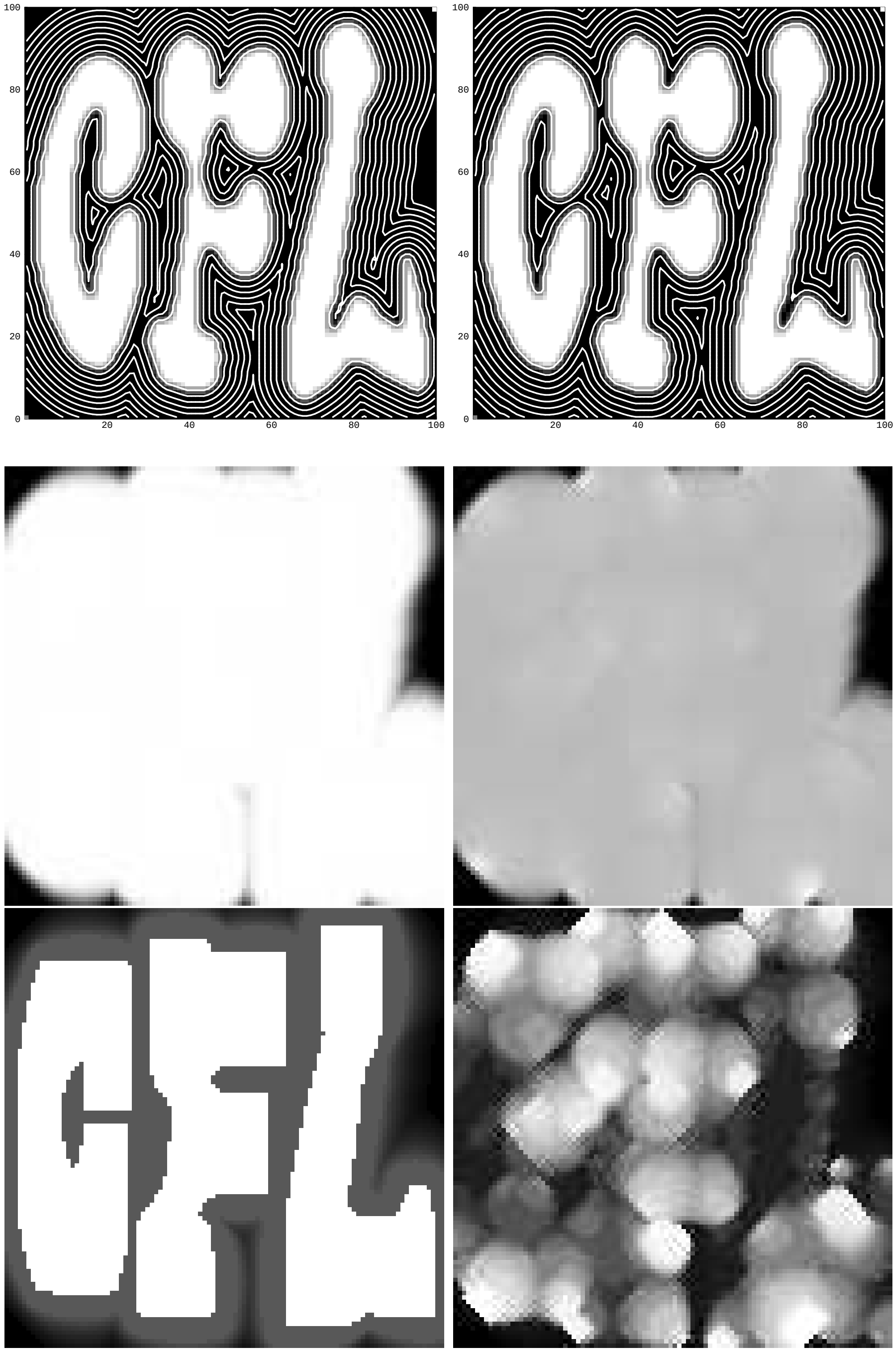

Figure 2.3: Left column: $\Delta t = 0.7$. Right column: $\Delta t = 0.8$, violating the CFL condition. Upper row: The offsets (zero level sets, of the propagating Φ every two time steps) are shown as white contours on the original image. Middle row: The images of Φ at $t = 11.2$, in which the heights Φ_{ij} are presented as gray levels. Bottom row: Φ images at $t = 11.2$ after histogram equalization that stresses the instability effects caused by violating the CFL condition.

2.8 Chapter Summary

In this chapter we reviewed the basic terminologies and methodologies in numerical analysis of conservation laws. Following Osher and Sethian, it was shown how planar curve evolution can be cast into the Eulerian formulation. This implicit formulation for curve evolution has the form of a Hamilton-Jacobi type of equation, for which there is a close relation to conservation laws. This relation was then explored and used to achieve efficient and stable numerical schemes.

The numerical schemes and limitations introduced in this chapter are used in subsequent chapters in the design of finite difference approximations to all the relevant PDEs. One important property of all the proposed numerical schemes is that when taking the discretization grid to a limit following the required limitations, the numerical schemes converge to the continuous case (the consistency property). As we shall see, this important property is lost for example when implementing graph search algorithms aimed at solving similar problems, since the metrics that the specific graphs induce inevitably lead to metrication errors.

Chapter 3

Weighted Distance Transforms

In this chapter a new framework for computing the Euclidean distance and weighted distance from the boundary of a given digitized shape is presented [69, 76]. The distance calculation with sub–pixel accuracy, is based on an equal distance contour evolution process. The relation between the classical shape from shading problem and the weighted distance transform is presented, as well as an algorithm that calculates continuous scale morphology.

3.1 Chapter Introduction

Distance maps on a pixel array picture are usually defined as discrete functions reflecting the minimal Euclidean distance of each pixel from the boundary pixels, see [37, 88, 143]. Algorithms that calculate such maps usually operate on segmented objects, *i.e.,* objects that have been transformed into a binary picture. In the process of mapping the shape into "White" interior and "Black" exterior pixels, information that may be crucial, if we desire to establish an accurate distance map, is usually lost.

A distance map may be found with the help of a propagating equal distance contour, or wave front, which starts from the boundary and propagates inwards (into the shape) with a unit velocity. The moving wave front assigns the distance to each pixel as it passes through it. See for example [8, 19], where the equal distance contour is referred to as the front of the flame in the *prairie–fire* model.

The problem of finding the distance becomes more interesting when considering a different cost or weight to each point in the domain. This leads to the idea of weighted distance transforms. We shall show that the weighted distance transform may be used to solve the classical shape from shading problem where one desires to reconstruct a three dimensional object from its gray level image. Considering other metrics than the Euclidean metric, introduces the continuous scale morphology, [13]. In [119] it was shown that morphological operations at any given scale may be performed via curve evolution, the 'structuring element' of the morphological operation being determined by the metric.

Taking off from planes to surfaces, we search for distance transform of a given area on a given surface, [63]. Calculating this *geodesic* distance map is performed by an equal distance curve propagating on the given three dimensional surface.

The solutions to all the above problems may be achieved by referring to the Osher–Sethian algorithm for propagating the equal distance contour, or by solving a Hamilton Jacobi type of equation that minimizes the difference between a given function gradient and the gradient of the desired solution. Two good examples of the last approach are the viscosity solutions to the shape from shading problem [117, 92], and [137] where such a formulation is used to find the distance map as part of a numerical algorithm that simulates incompressible two–phase flow, see also [69]. We shall study the relation between the two approaches.

This chapter presents a new framework for calculating all the above defined distance maps. In Section 3.2, the propagation rule of a planar curve evolving with constant velocity along its normal is introduced, as well as the level set formulation for that simple propagation rule. Section 3.3 introduces a cost function over the planar domain and studies the relation of the weighted distance transform and the shape from shading problem. In Section 3.4, the continuous scale morphology via level sets is used to achieve the distance maps for any given metric. In Section 3.5, propagation of equal distance contours on surfaces is used to calculate the *geodesic* distance map. Two possible applications are given in Section 3.6, followed by some examples, and summary of this chapter in Section 3.7.

In the next section we start by considering the evolution equation of a planar curve propagating with constant velocity along its normal direction. This simple planar case is reformulated to the evolution of an implicit representation of the curve. The new implicit formulation (the heart of the Osher–Sethian algorithm, called the *Eulerian* formulation) enables us to implement an efficient numerical scheme that approximates the evolution equation on a pixel grid. By tracking the propagating curve on the grid, the Euclidean distance transform is achieved.

The simplest planar curve evolution one may think of is given by the propagation rule of a curve evolving with constant velocity along its normal. This simple case is explored and the corresponding level set formulation is introduced. Next, a cost function over the planar domain is introduced in Section 3.3, and the relation of the weighted distance transform and the shape from shading problem is studied. In Section 3.4, the continuous scale morphology via level sets is used to achieve the distance maps for any given metric. In Section 3.5 propagation of equal distance contours on surfaces is used to calculate the *geodesic* distance map. The calculated distance maps may be used to find the path of minimal length in the Euclidean metric, the path of minimal cost in the weighted distance transform, and the minimal geodesics in the geodesic distance transform.

Using level sets of summation and subtraction of distance maps from different sources, two applications are given in Section 3.6, followed by some examples. The first is the solution to the Steiner problem, and the second is the Voronoi diagram calculated on surfaces.

3.2 Equal Distance Contour – Propagation Rule

Consider the propagating planar curve $\mathcal{C}(p,t) : [0,S] \times [0,T) \to \mathbb{R}^2$, evolving according to the propagation rule

$$\mathcal{C}_t(p,t) = \mathcal{N}(p),$$

where p parameterizes the curve, t is the time (which represents the distance in the simple constant velocity case,) and $\mathcal{N}(p)$ is the planar unit normal to $\mathcal{C}(p,t)$ at p.

As we have seen in Chapter 2, for $V_N = 1$, the Eulerian formulation to this evolution rule is given by

$$\phi_t = |\nabla\phi|.$$

Without any prior segmentation or edge detection procedures, let $\phi(x,y,0) \equiv I(x,y) - T$, where $I(x,y)$ is the gray level picture, and T is the threshold separating the object from the background, see [68] (T may also be adaptive, see *e.g.* [145]). We establish the following distance transform algorithm

1 Initialize $\phi(x,y,0) = I(x,y) - T$, see also [68].

2 Propagate ϕ according to $\phi_t = -|\nabla\phi|$, using a stable and accurate numerical algorithm (as provided by Osher and Sethian in [111], see Chapter 2).

3 Using a rectangular grid approximation (the pixel–grid), at each grid point: Check whether the zero level set has passed through the grid point. Assign the proper distance to the point if it does, using a simple linear interpolation.

This is a conceptually simple algorithm that calculates the distance map with sub–pixel accuracy. It can also solve the so-called *shape offsetting* problem in CAD, see [68]. The accuracy is governed by the pixel grid resolution and the distance (time) step of the iterative numerical scheme. When not considering any possible redundancy, the calculation effort is of order $\mathcal{O}(\frac{D}{\Delta t} m \cdot n)$, where D is the maximal distance from the boundary and $m \cdot n$ in the number of grid points used in the numerical scheme. A parallel implementation reduces the calculation complexity to $\mathcal{O}(\frac{D}{\Delta t})$.

3.3 Weighted Distance Transforms

In this section we argue that solutions of the classical *Shape from Shading* problem in which the viewer and the light source have the same location and the surface reflects the light with a "Lambertian" shading rule, see *e.g.* [17], also solve a problem of distance transforms with a given traversability map. We first introduce the weighted distance transform problem, and then present the approaches to its solution.

3.3.1 The path of minimal cost

Consider a problem where we search for a planar path from a source to a destination, so that the accumulation of a penalty function along that path is minimal. This minimization problem may be defined as follows. Let $a, b \in \mathbb{R}^2$ be the given source and destination points, and $f(x, y) : \mathbb{R}^2 \to \mathbb{R}^+$ be a given cost function. Then, find the planar curve $\mathcal{L}^o$ connecting a to b so that

$$|\mathcal{L}^o| \equiv \int_a^b f(x, y) d\mathcal{L}^o = \min_{\mathcal{L} \in L} \{ \int_a^b f(\mathcal{L}(s)) ds, \ \left| \frac{d\mathcal{L}}{ds} \right| = 1 \}, \tag{3.1}$$

where L is the set of all planar curves $\mathcal{L} \in \mathbb{R}^2$ connecting a to b, with arclength parameterization s.

The relation between the weighted distance transform and the shading from shape or contours from shading was dealt with in [141, 114]. The problem of finding the weighted distance may be shown to be equivalent to the well known *Shape from Shading* problem in Computer Vision, in the simple case where the light source and viewer have the same direction. In recent years, various solutions to this problem were proposed [6, 41, 56, 57, 58, 112, 113, 117]. In [17, 62, 66] a solution to the shape from shading problem via level sets is described. Motivated by this work, we propose to similarly solve the weighted distance transform problem and also to find the path of minimal cost from a to b.

As a first step propagate an equal cost contour (equal height contours according to [17]) that obeys the planar curve evolution rule

$$\frac{\partial \mathcal{C}}{\partial t} = \frac{1}{f(x, y)} \mathcal{N}.$$

Here $\mathcal{N}$ is the unit normal to the curve, and the initial curve is given as a very small contour around the source point $C(0) = \mathcal{C}_0$. Formulating this evolution in the Eulerian representation, see [127], yields the level set evolution version

$$\begin{cases} \phi_t = \frac{1}{f(x,y)} |\nabla \phi| \\ \\ \phi(0) = \phi_0, \end{cases}$$

see [66] as well as Chapter 2 for the numerical implementation. In the case of shape from shading, $I(x, y)$ – the gray level (shading) image – equals (under reasonable assumptions) to $\cos(\alpha)$, where α is the angle between the surface normal and the light source direction. When the viewing and the light source direction are the same, see Figure 3.1, the evolution of an equal height contour is given by

$$\frac{\partial \mathcal{C}}{\partial t} = \cot(\alpha) \mathcal{N},$$

or

$$\frac{\partial \mathcal{C}}{\partial t} = \frac{I}{\sqrt{1 - I^2}} \mathcal{N},$$

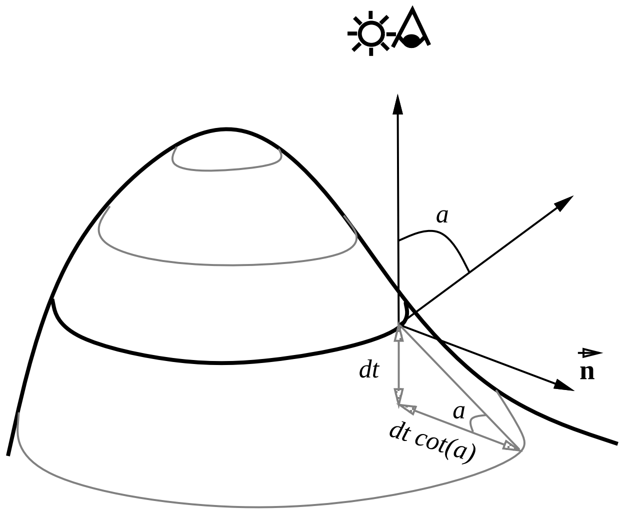

Figure 3.1: Descending dt in height, the equal height contour should evolve by $dt \cdot \cot(\alpha)$ along its normal direction

see [17, 66] for further details, and Chapter 4 for the case of oblique light source. Observe that defining the cost function as a function of $I(x, y)$ to be

$$f(x,y) \;=\; \tan(\alpha) \;=\; \frac{\sqrt{1-I^2(x,y)}}{I(x,y)},$$

the shape from shading problem coincides with the the problem of finding the weighted distance transform.

As it propagates, the equal cost contour assigns the proper time to each grid point it touches. The time assigned to every grid point therefore corresponds to the minimal cost it takes to reach that point (minimal height in the shape from shading case). Define $u_a(x, y)$ and $u_b(x, y)$ to be the weighted distance maps from the source and destination points, respectively. We can proceed and search for the minimal level of the function $u_a(x, y) + u_b(x, y)$, where $u_a(x, y)$ is the accumulated cost at point (x, y) when starting from a. This may be formally represented by

$$\mathcal{L}^o = \{(x,y)|u_a(x,y) + u_b(x,y) = \inf_{(x,y)} \{u_a(x,y) + u_b(x,y)\}\}.$$

This way of obtaining the paths of minimal cost may serve as a simple and direct approach for solving many energy or cost minimization problems.

Considering the problem in which the cost function is a traversability map, and the initial source is an area rather than a point, we are led to the "weighted distance transform" problem. In this problem a cost is assigned to each point in the continuous domain (the penalty function), and traveling from the source area to any point in the domain should accumulate the least possible cost.

We have shown in [66] a solution to the shape from shading problem, which also applies in this case. Another approach to this problem was recently presented by Rouy and Tourin in [117]. They propose to consider the "Hamiltonian" of this minimization problem

$$H(\nabla u(x,y)) = |\nabla u(x,y)| - f(x,y),$$

and present a viscosity solution to it which is implemented either via a numerical scheme based on Osher and Rudin [107], or via a simple, but longer procedure that evolves the discretization of the following PDE:

$$u_t = H(\nabla u(x,y)).$$

In the shape from shading problem $H(\nabla u(x,y)) = |\nabla u(x,y)| - \tan(\alpha)$, where u is the reconstructed surface and $\tan(\alpha)$ is the cost function dependent on the gray levels as before. This formulation explains the relation between the two approaches. In equal height contour evolution, the local propagation velocity of the curve is proportional to $\cot(\alpha)$, *i.e.,* when the gradient of the reconstructed shape is high, 'jumping' from one equal height contour to the next involves small steps. In the minimization problem, we try to modify u, so that the magnitude of its gradient will agree with $\tan(\alpha)$, as obtained from the shading image. Observe that the ϕ function introduced in the Eulerian approach is an arbitrary initialized auxiliary function used for an implicit representation of the propagating curve $\mathcal{C}$, while u is the reconstructed surface itself and may also be referred to as the "weighted distance function", "weighted distance map" or "weighted distance transform".

Note that this procedure might also be used to solve the simple distance transform, where $H(\nabla u(x,y)) = |\nabla u(x,y)| - 1$. It is easy to see that the above PDE-s' steady–state is for $|\nabla u(x,y)| = 1$ $(u_t = 0)$, which agrees with the distance function whose gradient is 1 almost everywhere. In [137] the distance is computed inward and outward simultaneously by modifying the above PDE to

$$u_t = S(u_0)(1 - |\nabla u|),$$

where $S(\cdot)$ is a smoothed version of a sign function, *e.g.*

$$S(u_0) = \frac{u_0}{\sqrt{u_0^2 + \epsilon^2}}.$$

This way the zero level set, that specifies the shape boundary in our case, is kept fixed while the rest converges to the proper distance from the boundary, negative distance in the interior and positive in the exterior of the shape.

3.4 Other Metrics: Continuous Scale Morphology

Let us consider other metrics than the simple Euclidean metric. Let $r(\theta)$ be the set of points of unit distance from the origin $(0,0)$, that is determined by a given metric. In Figure 3.2, an L_1 metric determines the set of points of a 'unit' distance from the origin as a diamond shape. The distance function imposed by a given metric may be determined by evolving the given shape boundary with a *structuring element* that is determined by the set $r(\theta)$ that characterizes the given metric.

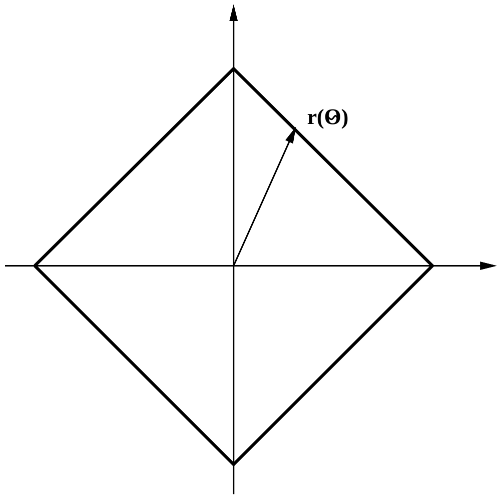

Figure 3.2: A structuring element is defined by the 'unit sphere' of the given metric. In this example the diamond is the unit sphere of the the L_1 *(city block)* metric.

It may be shown, see [119], that the above operation is the result of *dilating* the given shape by $r(\theta)$. In order to find the set of points of distance T from the boundary of the original shape, we can dilate the shape with a structuring element $r(\theta)T$. The same morphological operation may be achieved as a Minkovsky addition by successively applying an infinitesimal structuring element $\epsilon r(\theta)$, the number of successive dilation operations being given by T/ϵ. In the continuous case the structuring element must be taken to have an infinitesimal value. Observe a given point on the boundary, applying the infinitesimal dilation operation is the same as splitting the point into many points and moving or "propagating" each point to the boundary determined by the structuring element. The same procedure is carried out for each point along the curve. Our search is after the envelope (hull) of all the evolved points. This is a version of the *Huygens principle* form optics [126]. As shown in Chapter 2, the image (also known as trace, or geometric shape) of the evolving curve may be determined by only considering the normal component of the velocity at each point [120]. Using this fact we may determine only the maximal projection of the splitting point on the normal, and thereby get the evolution rule that agrees with the continuous scale morphology.

Formally, given the boundary of a shape $\mathcal{C}_0(s)$, the result of dilating the shape by $r(\theta)T$ is given by propagating the boundary according to:

$$\mathcal{C}_t = \sup_{\theta\in[0,2\pi]} \{\langle r(\theta), \mathcal{N}\rangle\}\mathcal{N},$$

starting with $\mathcal{C}(0) = \mathcal{C}_0$ and stopping the propagation at $t = T$, getting $\mathcal{C}(T)$ as the desired result. We have shown in [119] (see also [13]), that the above curve propagation rule in the level set formulation is given by

$$\phi_t = \sup_{\theta \in [0,2\pi]} \{\langle r(\theta), \nabla\phi \rangle\}.$$

In this framework we are after the distance imposed by the given metric. Using the above curve evolution we may assign each point a distance equal to t, which is the time the zero level set passed through that point. It was shown that the metric defines the structuring element that is used to dilate the boundary. The dilation operation may be performed in a continuous scale using the above curve evolution rule. The curve evolution is reformulated in the level set (Eulerian) formulation. And the distance is obtained by assigning the time of arrival of the zero level set to each point.

3.5 Surfaces: Geodesic Distance Map

Up to this point planar curves were propagated to calculate distances from given planar shapes. The above techniques still apply when considering more than two dimensions, that is, by propagating zero set $\mathbb{R}^n$ manifolds of functions in $\mathbb{R}^{n+1}$. A different kind of problem is calculating distance maps on surfaces in $\mathbb{R}^3$, (or manifolds in $\mathbb{R}^n$, in the general case).

Consider the simple case of a given boundary of a shape defined on a given surface $\mathcal{S} \in \mathbb{R}^3$. The boundary of the shape may be presented as a three dimensional curve $\alpha : [a,b] \in \mathbb{R} \to \mathbb{R}^3$, where $\alpha \in \mathcal{S}$. As proved in Chapter 6 [63] the evolution rule of an equal distance contour that propagates on a surface is given by

$$\alpha_t = \mathcal{T}^\alpha \times N,$$

where $\mathcal{T}^\alpha$ is the tangent vector of the curve $\mathcal{T}^\alpha = \alpha'/|\alpha'|$, and N is the normal of the surface $\mathcal{S}$. Propagating a $3D$ curve is hard to implement. We, therefore, project the evolution to the plane, and consider only the tangential component to the planar curve. Define the projection operation as π, *i.e* $\pi \circ (x,y,z) \equiv (x,y)$. The evolution rule of the planar curve, $\mathcal{C}$, that is the projection of the evolution of the surface curve α, is given by

$$\mathcal{C}_t = \langle \pi \circ (\mathcal{T}^\alpha \times N), \mathcal{N} \rangle \mathcal{N},$$

where $\mathcal{C} = \pi \circ \alpha$, and $\mathcal{N}$ is the normal of the planar curve $\mathcal{C}$.

A level set formulation of the above planar evolution rule is given by

$$\phi_t = \frac{1}{\sqrt{1+p^2+q^2}} \sqrt{(1+q^2)\phi_x^2 + (1+p^2)\phi_y^2 - 2pq\phi_x\phi_y},$$

where $p = dz/dx$, and $q = dz/dy$ are the derivatives of the surface $\mathcal{S}$, and $\mathcal{S}$ is assumed to be given as a function $(z(x,y),x,y)$.

The distance map on the surface may then be calculated as before, by propagating the planar curve (in its level set formulation) and assigning the distance to each point as the curve passes through that point. All the evolution rules presented here are implemented by numerical schemes that are consistent with the continuous propagation rule, and are motivated by numerical schemes that where developed in solving Hamilton Jacobi type of equations, [111, 108, 87]. As noted in Chapter 2, all the planar evolutions described here are of that type. The consistency condition, that is satisfied when constructing the numerical scheme, guarantees that the solution converges to the true one as the grid is refined and the time step in the numerical scheme is kept with the right proportion to the grid size. This is known not to be the case in general graph search algorithms (*e.g.* Dijkstra) that suffer from *digitization bias* due to the *metrication error* when implemented on a grid, see [99].

3.6 Applications and Results

In this section two examples of possible applications of using the sub–pixel distance maps based on the level set approach are presented. The first is the search for the Voronoi diagram between given points on surfaces (or weighted planar domains). The second deals with three point Steiner problems with cost function defined on the planar domain (the same procedure still apply when considering three points on a surface). Some examples of the distance and weighted distance transforms then demonstrate the capability of the algorithms devised.

3.6.1 Voronoi diagram on surfaces

It is possible to formulate a procedure that finds the Voronoi diagram on surfaces using the level set technique. Observe that the line that separates two points on the surface is given as the zero level set of the distance map from the first point, $\mathcal{M}_1$, subtracted from the distance map form the second point, $\mathcal{M}_2$. The projection of that curve is given by $\mathcal{L} = \{(x,y)|\mathcal{M}_1(x,y) - \mathcal{M}_2(x,y) = 0\}$. When more than two points are involved things become more complicated, and similar approaches as those that are used in the planar case should be followed. A possible procedure that finds the projection of the diagram is the following

$$\text{Voronoi Projection} = \{(x,y)|\forall i \forall j, i \neq j : \mathcal{M}_i - \mathcal{M}_j = 0 \cap [\cap_{k \neq i,j} \mathcal{M}_i < \mathcal{M}_k]\},$$

where $i,j,k \in \{1,2,...,N\}$ are indexes of the given points on the surface.

3.6.2 Solving the three point Steiner problem

The *Steiner* problem is defined as follows: given three points (or areas), find the graph of minimal cost connecting the three points. In this problem we still try to solve the problem as defined in Equation (3.1), but now L is defined to be the set of all planar curves connecting the three points that may have junctions (brunches). The cost is accumulated by integrating along all brunches. With uniform penalty, the solution is quite simple and looks like a "**Y**", connecting the three points, the junction in the "middle" usually forming three 120^o angles. Given three points and a cost function on the domain, it is easy to show that the situation is similar, but now there might be more than one solution.

In order to solve the problem we first calculate the weighted distance map from each of the given points, then find the infimum of the sum of the three maps. There might be more than one point achieving this infimum if there is more than one solution. These points will be junction points of the solution. The last stage involves computing the weighted distance map from the junction, and determining the path of minimal cost from the junction to each of the original regions using the technique described in the previous sections (that is, finding the minimal level set of the sum of the weighted distance map from the junction and the distance map from the point). Combining the paths to the junctions results in the desired solution. The "cost" of the graph is the infimum of the sum of the three weighted distance maps calculated on the first step.

This is only a simple example of a wide variety of possibilities for using calculated weighted distance maps.

3.6.3 Examples

The first example shows the sub–pixel distance map of a given gray level picture. The low resolution (64×64) gray level picture Figure 3.3 is used to initialize the ϕ function, *i.e.* $\phi(x, y, 0) = I(x, y) - T$, where the threshold $T = 128$, and each gray level pixel is represented by an 8-bit number taking the values 0 (white) to 255 (black). Figure 3.3.**b** shows $\phi(x, y, 0) = (I(x, y) - T)/256$. The outward distance map is shown in Figures 3.3.**c** as equal distance contours on a gray level indicating the distance, and 3.3.**d** as distance array. The same procedure applied to the negative picture to achieve the inwards distance map, is shown in Figure 3.4.

In the last example , Figure 3.5, the weighted distance transform is used to find the path of minimal cost between two points on a plane with two different traversability factors (two regions with different cost). The same result is obtained by using well known Snell's law in optics, and of course this is not surprising in view of the fact that what we are doing is solving efficiently an eikonal equation.

3.7 Chapter Summary

We have presented a new approach to find sub–pixel accuracy distance transforms, by efficiently propagating equal distance contours. The proposed numerical approximation uses an implicit representation of the evolving contour. The contour is propagated as a level set of bivariate function. This way numerical problems and topological changes, encountered in traditional direct wave propagation schemes are naturally overcome. A natural initialization to the algorithm was suggested, according to which the gray level image of the shape provides

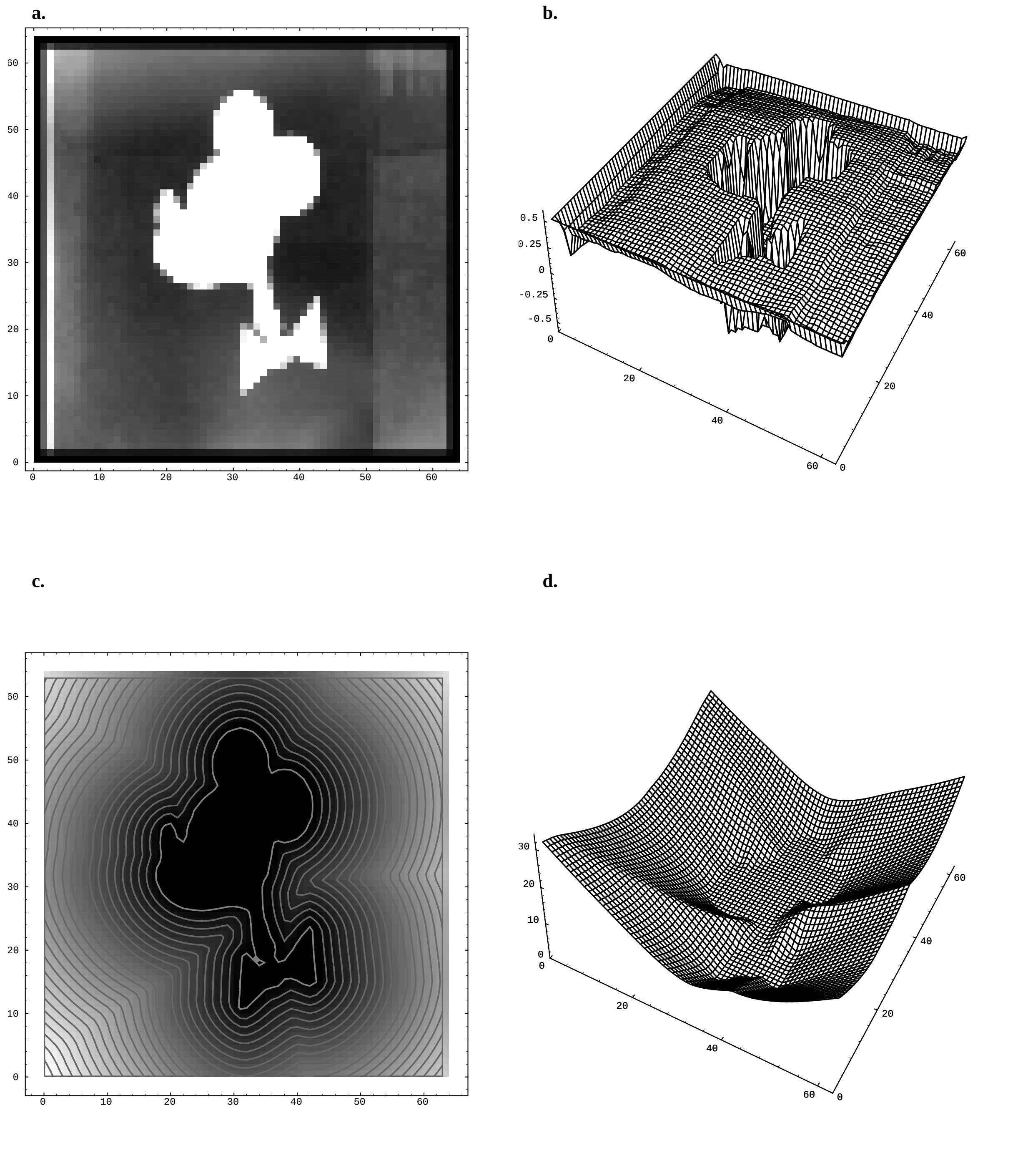

Figure 3.3: Sub–pixel distance map of a low resolution (64×64) gray level picture. **a.** The picture is given as 8-bit gray level numbers taking the values 0 (white) to 255 (black). **b.** $\phi(x, y, 0) = (I(x, y) - 128)/256$. **c.** The outward distance map as equal distance contours on gray levels indicating the distance. **d.** The distance map given as elevation array.

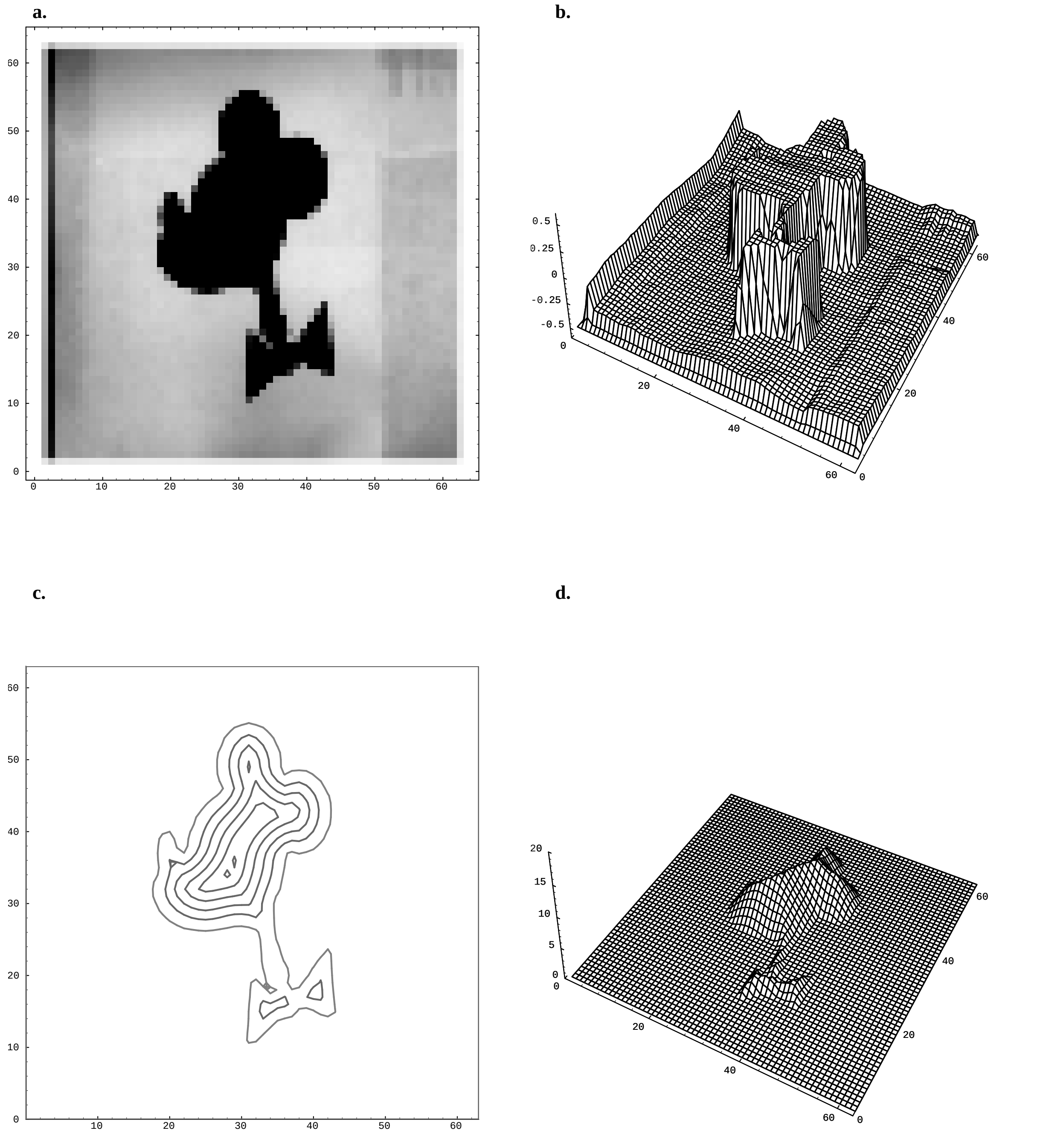

Figure 3.4: Sub–pixel distance map of a low resolution (64 × 64) gray level picture, the inverse of the previous one (**a**).

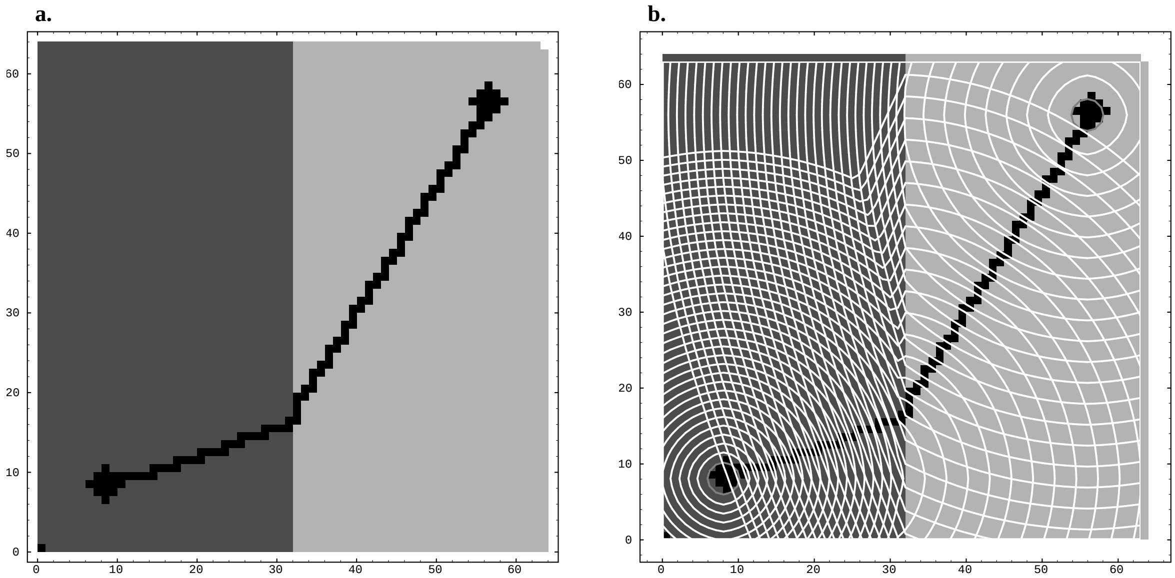

Figure 3.5: The weighted distance transform is used to find the path of minimal cost, between two points on two opposite sides of a low resolution picture (64×64). The picture displays the two different traversability factors as two gray levels.

the initial surface. This surface serves as the implicit representation of the initial contour.

A way to calculate the sub–pixel weighted distance transform was described. We have presented the relation of the weighted distance transform and the shape from shading problem, and used shape from shading solutions to calculate the path of minimal cost between two and then three points.

Continuous scale morphology via curve evolution was used to calculate distance maps for any given metric. Calculating the geodesic distance on surfaces was carried out by propagating a planar curve that is the projection of the three dimensional equal distance contour propagating on the surface.

Two examples of using the distance transforms were presented, the solution of the three points Steiner problem (on surfaces or on weighted domains), and the Voronoi diagram on surfaces.

Chapter 4

Shape from Shading

In this chapter a new algorithm for recovering depth to a *Lambertian* $\mathcal{C}^1$ smooth object given its gray–level image under uniform illumination from the viewing direction is presented. To recover depth, an almost arbitrarily initialized surface is numerically propagated on a rectangular grid, so that a level set of this surface tracks the height contours of the depth function. The image shading controls the propagation of the surface. When the light direction is tilted with respect to the viewing direction the problem is solved by tracking the projection of equal height contours defined with respect to the light source direction. This projection approach provides a solution that overcomes ambiguity problems encountered in previous work, while the level set approach of implementing the contour propagation overcomes numerical problems and some of the topology problems of the evolving contours.

4.1 Chapter Introduction

The problem of reconstructing three dimensional objects from their shaded images has been extensively investigated by computer vision researchers over the last two decades, starting from the classic "shape from shading" work of Horn [55]. Until recently, iterative schemes to solve this problem were derived by regularization versions of the direct problem formulation, see [56] for a survey on these methods. New iterative methods were recently suggested by Dupuis and Oliensis [105], and by Rouy, Tourin, Osher and Rudin based on viscosity solutions to partial differential equations, [117, 107]. They represented the shape from shading problem as a first order *Hamilton–Jacobi* equation. Viscosity solutions to the Hamilton–Jacobi equation guarantee uniqueness (up to the $+/-$ ambiguity of the shape from shading problem) and existence. The viscosity solutions are related to the smoothness of the Hamiltonian under consideration.

A direct approach to solving the shape from shading problem based on characteristic strip expansion method was provided by Horn in his original work. A different direct approach, using equal–height contour propagation, was suggested by Bruckstein [16, 17]. An accurate numerical scheme for implementing this method based on a level set propagation algorithm devised by Osher and Sethian [111] was then developed in [62, 66] and was also compared with viscosity solutions and other traditional methods in [81]. In [66], the numerical properties of the level set based approximation were explored, like the effects of noise, and its performance on real images. This novel level set based formulation for the equal height contour propagation method for the shape from shading problem turned out to be a general method for solving Dirichlet problems for Hamilton–Jacobi equations, see S. Osher [109].

In this chapter we first summarize the level set based numerical approach to the equal height contour propagation method for solving the shape from shading problem. Then, we derive the contour propagation equation for the case of general light source direction, where the contours are "equal–height" with respect to the light source direction instead of the height axis, see Figure 4.1. This new formulation solves some ambiguity problems that were encountered in our previous work [62, 66, 81]. We note that light source coordinates were used in [85] to improve the shape from shading method of [112, 57], and later in [42] as a natural coordinate system with respect to the singular points (the brightness points in image).

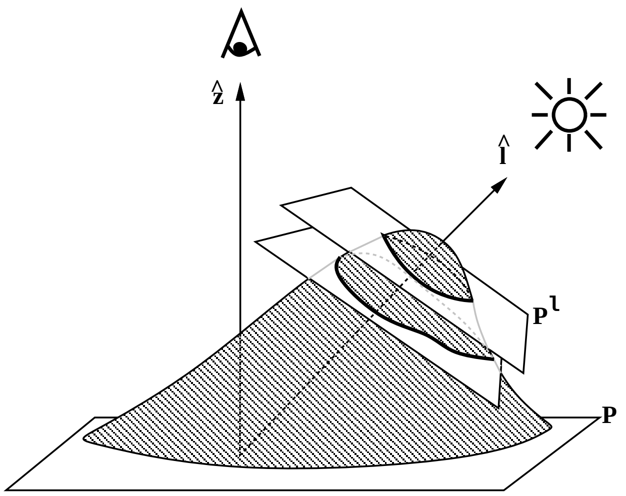

Figure 4.1: The case where the viewer and the light source are at different directions ($\hat{l} \neq \hat{z}$) is solved by propagating curves which are "equal height" contours with respect to the given light source direction ($\hat{l}$).

In the next section we give a formal definition of the shape from shading problem under Lambertian reflectance. In section 4.3, the equal height contour approach to the shape from shading problem is summarized and the new formulation to the general light source direction is presented. Section 4.4, presents the level set based algorithm for tracking the equal height contours. Section 4.5, presents results of applying the new algorithm to some synthetic as well as real shading images.

4.2 Problem Formulation and the Reflectance Map

Suppose we are given a continuous function of two variables, $z(x, y)$ describing the surface of an object. The shaded image of that surface is defined as a brightness distribution $I(x, y)$. The brightness values are defined by the properties of the surface, by its orientation at every point and by the illumination. The brightness $I(x, y)$ is determined via a so–called shading rule or *reflectance map*, which characterizes the surface properties and provides an explicit connection between the image and the surface orientation. The shape from shading problem is to recover the depth function $z(x, y)$, from the image $I(x, y)$.

Let us first specify the surface orientation using the components of the surface gradient $p = \partial z/\partial x$ and $q = \partial z/\partial y$. The surface unit normal can then be written as

$$N = \frac{1}{\sqrt{1 + p^2 + q^2}}(-p, -q, 1)^T.$$

In case of a surface with so–called Lambertian, or diffuse reflection properties and uniform illumination, $I(x, y)$ is proportional to the cosine of the angle between the surface normal N and the direction to the light source $\hat{l}$ (see Figure 4.2).

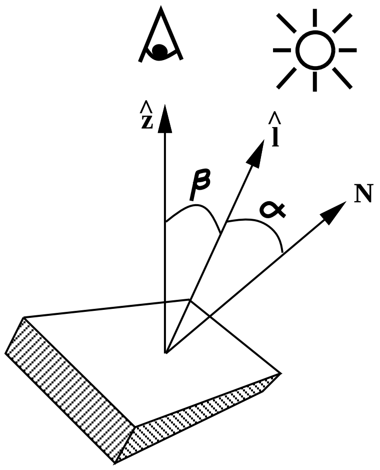

Figure 4.2: On a patch of a surface, the brightness under Lambertian shading rule is given by the cosine of the angle between the surface normal and the light source direction $I = cos\ \alpha = \langle N, \hat{l} \rangle$.

In the general case we define the light source direction as

$$\hat{l} = \frac{1}{\sqrt{1 + p_l^2 + q_l^2}}(-p_l, -q_l, 1)^T. \tag{4.1}$$

The dependence between brightness and surface orientation (the reflectance map) can be written as a function mapping the surface normal direction N to the brightness image $I(x, y)$ as follows

$$I(x, y) = \text{Function of}(N) = R(p(x, y), q(x, y)). \tag{4.2}$$

This is the image irradiance equation. For example in the Lambertian case we have

$$I(x,y) = \ R(p,q) = \langle \hat{l}, N \rangle = \frac{1 + p_l p + q_l q}{\sqrt{1+p^2+q^2}\sqrt{1+p_l^2+q_l^2}}$$

In this case the normal direction lies on an ambiguity cone whose main axis is directed towards the light source, see Figure 4.3b. There are some constant factors multiplying the R function due to the surface albedo, the beam luminance etc., in the above equation. It is, however, possible to factor them out by simply re-scaling the image gray levels.

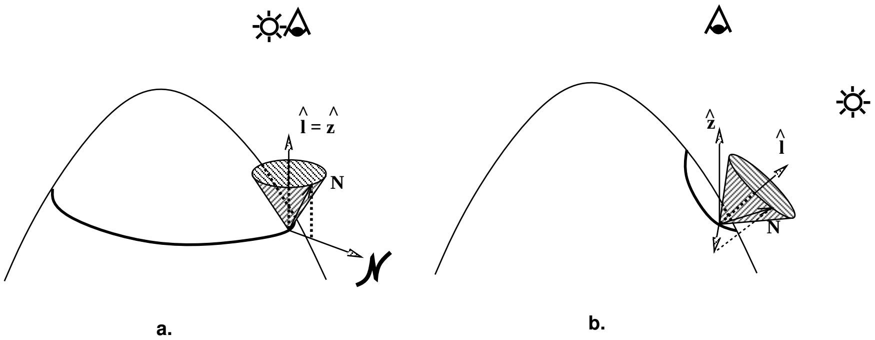

Figure 4.3: **a.** When the light source is from above ($\hat{l} = \hat{z}$) the ambiguity cone is directed upwards. **b.** For other light source directions the ambiguity cone is tilted towards the light source.

For the simple case where $\hat{l} = (0,0,1)^T$ we have

$$R(p,q) = \frac{1}{\sqrt{1+p^2+q^2}},$$

and the ambiguity of the surface normal direction is an upward directed cone, see Figure 4.3a.

Equation (4.2) is a nonlinear partial differential equation that has to be satisfied by the surface $z(x,y)$. Therefore, solving the shape from shading problem amounts to solving a nonlinear partial differential equation. Clearly boundary conditions are necessary. Any shape from shading scheme reconstructs a function $z(x,y)$ by implicitly or explicitly recovering its normal $N(x,y)$ everywhere. The surface normal at each point is represented by two numbers, and the only constraint we have so far is the image's irradiance equation (4.2). The two variables representing the surface normal direction at each point can only be computed by using more than one equation. The "art" of recovering a shape from its shaded image requires introducing boundary conditions and local constraints that follow from reasonable assumptions concerning the relation of each point on the surface to its neighborhood.

4.3 Shape from Shading via Equal–Height Contours

4.3.1 Light source from above

Let us first briefly review Bruckstein's shape from shading approach, (see [16, 17]). An equal–height contour or a *level curve* is a continuous curve in the (x, y)–plane on which the function $z(x, y)$ is constant. Defining $\{x(s), y(s)\}$, $s \in [0, L]$, as the parametric representation of the contour, we have

$$\frac{\partial z}{\partial s} = \frac{\partial z}{\partial x}\frac{\partial x}{\partial s} + \frac{\partial z}{\partial y}\frac{\partial y}{\partial s} = 0$$

or[1]

$$p\ x_s + q\ y_s = 0.$$

The unit normal to the equal height contour on the (x, y)–plane is given by

$$\mathcal{N}(s) = \frac{1}{\sqrt{x_s^2(s) + y_s^2(s)}}\left(y_s(s), -x_s(s)\right).$$

Clearly $\mathcal{N}$ is in the direction of the projection of N on the image plane. When considering the case of propagation from small circles around the singular points the unit normal $\mathcal{N}$ is directed outward, *i.e.* pointing outside the region bounded by the curve. Consider the case where $\hat{l} = \hat{z}$ (that is $\hat{l} = (0, 0, 1)$) and define dz as the height we climb while progressing a distance $\mathcal{D}$ in the normal direction $\mathcal{N}$ in the (x, y)–plane. From elementary geometry we have

$$\mathcal{D} = dz\ cot\ \alpha,$$

where α is the surface orientation angle, the angle between the surface normal N and the light source direction $\hat{l}$ ($\cos\alpha = \langle \hat{l}, N\rangle$). Under the simple Lambertian shading rule where $R(p, q) = cos\ \alpha = \frac{1}{\sqrt{1+p^2+q^2}}$, we have

$$\mathcal{D}\ =\ dz\ cot\ \alpha\ =\ dz\frac{1}{\sqrt{p^2 + q^2}} = dz\frac{I}{\sqrt{1 - I^2}}.$$

If, from the first contour, we uniformly climb dz, we get to the next equal height contour via

$$\{x(s, dz), y(s, dz)\} = \{x(s, 0), y(s, 0)\} + \mathcal{D}(s) \cdot \mathcal{N}$$

[1] The subscripts s, t, x, y, refer to partial derivatives, (*e.g.* $x_t \equiv \partial x/\partial t$), and the subscripts (and superscripts) l, n refer to the light source direction and the planar normal, respectively.

This yields the propagation of the equal height contours as a nonlinear initial value *P.D.E.* problem. Given $\{x(s,0), y(s,0)\}$ the evolution equations are [16, 17]):

$$\left\{ \begin{array}{rcl} x_t(s,t) & = & F(x,y)\frac{y_s}{\sqrt{x_s^2+y_s^2}} \\ \\ y_t(s,t) & = & F(x,y)\frac{-x_s}{\sqrt{x_s^2+y_s^2}} \end{array} \right. \tag{4.3}$$

where $t \equiv z$ and

$$F(x(s,t), y(s,t)) = \frac{I(x(s,t), y(s,t))}{\sqrt{1 - I^2(x(s,t), y(s,t))}}. \tag{4.4}$$

Note that z is replaced by t so that the equal height contour propagation in height is now referred to as a curve evolving in "time".

Define $\mathcal{C}(0) = (x(s,0), y(s,0))$ as the smooth (and, in some cases, closed) initial equal height curve, and $\mathcal{C}(t)$ as the one–parameter family of curves generated by moving $\mathcal{C}(0)$ along its normal vector field with a speed given by F. Here, F is a given scalar function of the brightness I. Using this notation, Equation (4.3) can be written as a planar curve evolution equation $\frac{\partial}{\partial t}\mathcal{C}(t) = F(x,y) \cdot \mathcal{N}$.

4.3.2 General light source direction

When the light source direction is $\hat{l}$ (as defined in (4.1)), the brightness map under the Lambertian shading rule is $I = \langle \hat{l}, N \rangle$. In this case the surface normal is on a tilted ambiguity cone as described in Figure 4.3b. In [62, 66] it was discussed how to propagate an equal height contour in this situation. It was also noted that there are possible ambiguities in recovering the normal direction, and these had to be solved by adding smoothness constraints along the equal height contour or along the surface. In the rest of this section we present a different kind of curve propagation that overcomes these ambiguities and frees us from the need to consider additional constraints while reconstructing the shape.

Consider the plane P^l defined by the light source direction $\hat{l}$, see Figure 4.1. The equal height contours with respect to this plane are given by the previous propagation rule

$$\mathcal{C}_t^l = F^l(x^l, y^l)\ \hat{\mathbf{t}} \times \hat{l},$$

where $\hat{\mathbf{t}}$ is the tangent vector to $\mathcal{C}^l$, and F^l is the velocity which is a function of the image as it would have been seen from the light source direction. This evolution may be projected to the (x,y)–plane by first projecting the velocity and then taking only the normal component of the projected velocity v_n, since in the case of planar curve propagation the tangent component of the velocity affects only the internal parameterization of the curve while the trace of the curve is determined only by the normal component (see [43], or Chapter 2 for justification). The projected evolution is then given by

$$\mathcal{C}_t = \tilde{F}\ v_n\ \mathcal{N},$$