社団法人　情報処理学会　研究報告
IPSJ SIG Technical Report

2006－CVIM－156（17）
2006／11／10

レベルセット法とその実装法について

倉爪 亮†

† 九州大学

〒819-0395 福岡県福岡市西区元岡 744
九州大学　システム情報科学研究院
kurazume@is.kyushu-u.ac.jp

あらまし　本稿では，Active Contour Model（動的輪郭モデル）の代表的な手法として，Kass らの Snakes と Osher，Sethian らの Level Set Method に焦点を当て，その理論と実装法を概説する．特に Level Set Method に対しては，その基本的な考え方から，Upwind Scheme, AOS, ADI などを用いた実装法，局所成長速度場と拡張成長速度場，Gemetric Active Contour と Geodesic Active Contour 等，LSM を利用したアプリケーションの構築に必要な知識と具体的な手法を解説する．また高速で安定な Level Set Method の実装法として，著者らの提案する Fast Level Set Method を紹介し，ビデオ画像上の移動物体のリアルタイム追跡，および 3 次元モデリングへの適用例を示す．

Introduction of level set method and its implementation

Ryo KURAZUME†

† Kyushu University

Graduate School of Information Science and Electrical Engineering, Kyushu University
744, Motooka, Nishi-ku, Fukuoka-shi, Fukuoka, 819-0395, JAPAN
kurazume@is.kyushu-u.ac.jp

Abstract　This paper introduces the theory and the implementation method of Active Contour Models including Snakes proposed by Kass and Level Set Method (LSM) by S. Osher and J. A. Sethian, LSM has attracted much attention as a topological-free active contour model. This method utilizes an implicit representation of a contour to be tracked, and is able to handle topological change of a contour naturally. This paper explains the basic idea of LSM, concrete implementation method using Upwind Scheme, AOS, and ADI, no extension velocity field and extension velocity field, and Geometric Active Contour and Geodesic Active Contour. This paper also introduces the efficient algorithm of LSM named Fast Level Set Method (FLSM) and some experimental results of realtime tracking of moving objects in video images and 3D modeling.
Keywords　Snakes, Level Set Method, Fast Marching Method, Geodesic Active Contour

1　はじめに

Kass の Snakes[11] に代表される Active Contour Model（動的輪郭モデル）は，ノイズに対して頑強な境界追跡法として現在広く用いられている。当初，2 次元画像の境界追跡手法として提案された Snakes は，その後 3 次元にも拡張され，3 次元点の集合（points of cloud）からそれを内包する閉曲面を安定に抽出するための deformable surface の手法 [7] [14] [10] として，幾何モデリングや CG 分野で研究が進められた．しかし従来の動的輪郭モデルに共通して，分離や結合など境界の位相変化への対応が困難であることが問題とされていた．この問題に対し，Terzopoulos ら [15] は位相変化可能な Snakes の拡張として T-Snakes (Topology-Adaptive Snakes) を提案しているが，位相変化の検出処理や結合される領域に対する処理の追加など，本質的な解決策とは言い難い．

これに対し，Osher，Sethian らによって提案された Level Set Method [17] [25] は，本質的に位相変化が可能な動的輪郭モデルとして注目を集め，現在，移動体追跡 [20] や 3 次元幾何形状モデリング [3] [33] [16]，半導体 [27] や結晶形成シミュレーション [34]，ノイズ除去 [26]，モデル成形 [16] など，様々な用途に応用され始めている．この手法は，検出する境界を一次元高い補助関数のゼロ等高面（zero level set）とみなし，境界の進行条件である偏微分方程式（PDE, Partial Differencial Equation）を数値的に解いて補助関数の形状を変更し，そのゼロ等高面を次々に検出することで境界形状を動的に制御する手法である．

本稿では，Active Contour Model に始めて触れる大学院修士レベルの学生を対象に，Snakes や Level Set Method など Active Contour Model の代表的な手法に対し，特に計算機への実装を目的とした具体的なアルゴリズムに焦点を当て解説する．

2 Active Contour Model とは

Active Contour Model とは，対象となる空間を領域の性質を現す指標により複数の領域に分け，その領域の間の境界線（あるいは境界面）を時連続的に移動させる手法である．例えば，図 1 左に示すように，2 次元平面内に閉領域 Ω が置かれた状態を考えよう．この閉領域 Ω と，これ以外の領域 Ω の間には，図に示すような閉じた境界線 C が定義できる．ここでこの閉領域 Ω が次時刻で図 1 右のように変形し，新たな閉領域 Ω' が生成されたとする．このとき，その境界も閉領域 Ω' に応じて境界線 C' のように変形するが，この境界線 C' を全く新たな境界として閉領域 Ω' から再度求めるのではなく，前時刻の境界線 C から変化したものとして連続的に求めるものが，Active Contour Model であり，この最も代表的な手法が，Kass らの Snakes[11] であり，Osher，Sethian らの Level Set Method[17] [25] である．次章以降では，これらの理論と具体的な実装法について概説する．

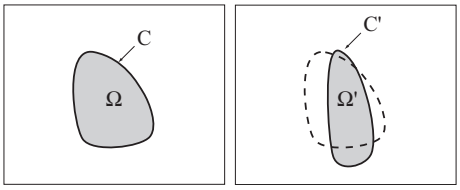


図 1: Active Contour Model

3 Snakes の理論と実装

3.1 Snakes の理論

2 次元画像 $I(x,y) \in R^2$ における Snakes の理論について述べる．Snakes は，陽 (explicit) な境界のパラメータ表現であり，次式で与えられるエネルギー関数 $\mathcal{E}(\mathbf{v})$ を最小化するように，閉曲線 C が決定される．

$$\mathcal{E}(\mathbf{v}) = \mathcal{S}(\mathbf{v}) + \mathcal{P}(\mathbf{v}) \tag{1}$$

ただし $\mathbf{v}(\mathbf{s}) = (x(s), y(s))$ は閉曲線 C のパラメータ表現であり，$s \in [0,1]$ は閉曲線 C の弧長パラメータである．

上式で $\mathcal{S}(\mathbf{v})$ は閉曲線 C の内部変形エネルギーであり，次式により定義される．

$$\mathcal{S}(\mathbf{v}) = \frac{1}{2}\int_0^1 w_1(s)\left|\frac{\partial \mathbf{v}}{\partial s}\right|^2 + w_2(s)\left|\frac{\partial^2 \mathbf{v}}{\partial s^2}\right|^2 ds \tag{2}$$

ただし，$w_1(s), w_2(s)$ はそれぞれ曲線の張力および剛性を決定する非負の関数である．例えば $w_1(s)$ を増やすと，無駄なループを減らし弧長を短くする効果が生じる．一方 $w_2(s)$ を増やすと，全体として曲率の小さい，すなわち硬い曲線が生成される．

これに対し，$\mathcal{P}(\mathbf{v})$ は外部ポテンシャルエネルギーであり，一般に以下のように定義される．

$$\mathcal{P}(\mathbf{v}) = \int_0^1 P(\mathbf{v}(s))ds \tag{3}$$

ここで $P(\mathbf{v}(s))$ は，曲線と画像との適合度を表すポテンシャル関数であり，例えば以下のように定義される．

$$P(x,y) = -c\left|\nabla[G_\sigma \otimes I(x,y)]\right| \tag{4}$$

ただし，c は定数，∇ は gradient，$G_\sigma \otimes I$ は Gaussian Filter である．これにより，曲線は画像中で濃度勾配の大きな領域で停留するように制御される．

3.2 Snakes の数値解法

Snakes の数値解法としては，Dynamic programming を用いたもの [32] や greedy アルゴリズムを用いたもの [30] が有名であるが，ここでは 8 ビット 256 階調の 2 次元濃淡画像に対する最も基本的なアルゴリズム [30] を示す．

まず，エネルギー関数 $\mathcal{E}$ を離散化する．このために，曲線 C に対し離散化した点の集合 $\mathbf{u} = \{\mathbf{v}_i\}$,$(i = 0 \sim N-1)$ を考え，曲線 C を点列 $\mathbf{u}$ を結ぶ多角形で近似する．また，点 $\mathbf{v}_i$ での局所エネルギー $\mathcal{E}_i$ を以下のように定義する．

$$\mathcal{E}_i = \alpha\mathcal{E}_{cont,i} + \beta\mathcal{E}_{curv,i} + \gamma\mathcal{E}_{image,i} \tag{5}$$

ただし，α, β, γ は適当な定数であり，

$$\mathcal{E}_{cont,i} = \left|\mathbf{v}_{i+1} - \mathbf{v}_i\right|^2 \tag{6}$$

または

$$\mathcal{E}_{cont,i} = (\bar{d} - \left|\mathbf{v}_{i+1} - \mathbf{v}_i\right|)^2 \tag{7}$$

また

$$\mathcal{E}_{curv,i} = \left|\mathbf{v}_{i+1} - 2\mathbf{v}_i + \mathbf{v}_{i-1}\right|^2 \tag{8}$$

$$\mathcal{E}_{image,i} = -\frac{I_{v_i} - I_{min}}{max(I_{max} - I_{min}, 5)} \tag{9}$$

とする．ただし，$\bar{d}$は頂点間距離の平均値，I_{v_i}は点$\mathbf{v}_i$での輝度値，I_{max},I_{min}は点$\mathbf{v}_i$の8近傍における最大および最小の輝度値である．また通常$\mathcal{E}_{cont,i}$,$\mathcal{E}_{curv,i}$は，点$\mathbf{v}_i$の8近傍における最大値で割り，正規化しておく．

これより，具体的なアルゴリズムは以下のようになる．

Step.1（変数の準備） 頂点の座標を格納する変数$\mathbf{v}_i = (x_i, y_i), (i = 0 \sim N-1)$と頂点の総移動量を表す変数$d_{total}$，平均頂点間距離を表す変数$\bar{d}$，および繰り返し回数を表す変数$n$を準備する．

Step.2（初期化） $n = 0$とし，適当な間隔で頂点を配置する．またそのときの頂点座標を$\mathbf{v}_i$に格納する．

Step.3 nに1を加え，$d_{total} = 0$とする．また平均頂点間距離$\bar{d}$を計算する．

Step.4 ある頂点$\mathbf{v}_i$を選び，その8近傍の点jで式(5)により局所エネルギー$\mathcal{E}_j$を計算する．

Step.5 最も局所エネルギーの小さな近傍点を新たな頂点として，頂点$\mathbf{v}_i$を移動させる．またそのときの頂点の移動量をd_{total}に加える．

Step.6 **Step.3** $\sim$ **5**を全ての頂点に対して行う．

Step.7 **Step.3** $\sim$ **6**を，頂点の総移動量d_{total}が閾値以下になるか，nが予め決められた繰り返し回数を超えるまで繰り返す．

4 Level Set Method

4.1 Snakesの限界

前章で説明したSnakesは，曲線上を線積分したエネルギー関数を最小化するため，局所的なノイズに対して頑強であり，移動体追跡や医療画像処理など多くのアプリケーションで用いられている．しかしSnakesには，曲線の分離や結合などが困難であるという本質的な問題がある．例えば，図2左の領域Ωが，次時刻で図2右のように2つの領域Ω_1,Ω_2に分離したとする．しかし上述したSnakesの実装法では，曲線は離散化した点を，予め定められた順に結んでできる多角形として求められるため，領域が分離しても境界線の数は1つであり，分離した領域それぞれ対応する複数の境界線は求められない．

4.2 Level Set Methodの考え方

Level Set Methodは，Osher，Sethianら[17] [25] [13] [4]によって提案された位相変化が可能な動的輪郭

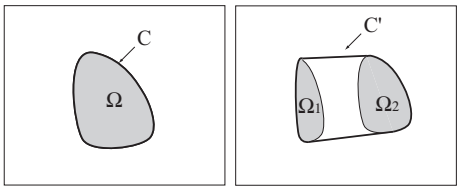


図2: Active Contour Modelの問題

モデルであり，Snakesと同じActive Contour Modelでありながら，領域の分離，結合を“自然な形”で表現できる．前章で説明したSnakesは，エネルギー最小化の考えに基づくものであったのに対し，Level Set Methodは，曲線の状態（収縮，膨張，曲率変化等）を偏微分方程式（PDE, Partial Differencial Equation）により表し，境界の進行を偏微分方程式の解として陰(implicit)に表現するものである．以下，Level Set Methodの基本的な考え方を示す．

Level Set Methodでは，対象としている空間に対して，1つ次元の高い空間を設定し，検出する境界をその高次元空間で定義された関数（補助関数）の断面と考える．例えば，図3(a)に示すように，2次元空間上で時刻tにおいて境界Cで囲まれた領域Ωを考えよう．このとき，Level Set Methodではこの領域を，図3(c)に示すような3次元空間で定義された補助関数ϕのゼロ等高面（zero level set）$\phi = 0$と考える．また通常，補助関数ϕの初期値には，境界Cからの符号付距離（境界の内側が負，外側が正）を与えておく．次に，次時刻$t + \Delta t$において，この補助関数ϕの形状をその形状を保ったままϕの正方向へ移動させ，同様にゼロ等高面$\phi = 0$を切り出す．補助関数ϕが図3(c)のように双峰的である場合，図3(d)のように，時刻$t + \Delta t$でのゼロ等高面は2つの領域Ω_1，Ω_2からなり，その結果，境界も図3(b)のようにC_1，C_2となる．このように，補助関数の形状を領域の特徴に応じて適切に設計し，制御することで，補助関数（すなわち境界）に対して滑らかな形状を保ちつつ，自然な形で境界の分離，結合が表現できる．

4.3 Level Set Methodの理論

例として，2次元画像$I(x, y) \in R^2$での境界線の検出問題を考える．まず時刻tでの境界線を$C(\mathbf{p}, t)$とする．ただし$\mathbf{p} = (p_x, p_y)$である．この境界に含まれる点$\mathbf{p}$は，移動速度$F(\kappa)$で境界線の法線方向$\mathbf{N}$に移動していると考える．ここでκはその点での境界線の曲

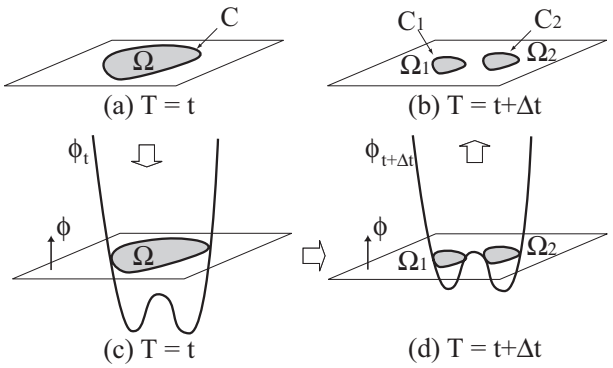


図 3: Level Set Method の考え方

率であり，$F(\kappa)$ を成長速度という．これを式で表すと，

$$C_t = F(\kappa)\mathbf{N} \tag{10}$$

$$C(\mathbf{p},0) = C_0(\mathbf{p}) \tag{11}$$

となる．ただし，C_t は境界 C の時間変化，$C_0(\mathbf{p})$ は初期曲線である．この問題は前章の Snakes のように，差分方程式を利用したラグランジェ法で解くことができるが，トポロジーの変化には対応できないという問題が残る．

そこで図 3 に示したように，新たな補助関数 $\phi(x,y,t)$ を導入し，境界線 $C(\mathbf{p},t)$ はその関数の一部，すなわち $\phi(x,y,t)=0$ を満たす ϕ で表されると考える．ここで，点 $\mathbf{p}(t)$ が境界線 $C(\mathbf{p},t)$ 上の点である場合，これが常に $\phi(x,y,t)$ のゼロ等高面上である条件は，

$$\phi(\mathbf{p}(t),t)=0 \tag{12}$$

で表される．これを偏微分すると，

$$\phi_t + \nabla\phi(\mathbf{p}(t),t)\mathbf{p}_t = 0 \tag{13}$$

となる．また曲線上の単位法線ベクトルは，

$$\mathbf{N} = \frac{\nabla\phi}{|\nabla\phi|} \tag{14}$$

で表され，さらに成長速度 $F(\kappa)$ は境界 $C(\mathbf{p},t)$ の法線方向速度であるから，

$$\mathbf{p}_t \cdot \mathbf{N} = F(\kappa) \tag{15}$$

となる．これにより式（13）は以下のように書くことができる．

$$\phi_t = -F(\kappa)|\nabla\phi| \tag{16}$$

$$\phi(C_0(p),0)=0 \tag{17}$$

このように境界 $C(\mathbf{p},t)$ を直接的に移動する代わりに，式 (16) によって補助関数 $\phi(x,y,t)$ を更新し，$\phi(x,y,t)=0$ を満たす線を新たな境界線とすることで，トポロジーの変化に対応した領域追跡が可能となる．また後述する式 (18) のように，画像 $I(x,y)$ の勾配などに応じて成長速度 $F(\kappa)$ を制御することで，例えば画像の一部の注目領域を取り囲むような複数の曲線を同時に得ることができる．

さて，Level Set Method の実装法は，検出する境界の移動方向の制約（全方向へ移動するか，一方向へ移動するか）によって，(一般的な) Level Set Method (4.4 ∼ 4.8) と Fast Marching Method (4.10) の 2 つの手法に分類できる．以下，それぞれの手法の具体的な実装法を述べる．

4.4 Upwind Scheme を用いた Level Set Method の数値解法

8 ビット 256 階調の 2 次元濃淡画像 $I(x,y)\in R^2$ に対する Level Set Method の具体的な数値解法を示す．

Step.1（変数の準備） まず，濃淡画像の各グリッド（ピクセル）毎に，そのグリッドの補助関数値を保持するスカラー量 $\phi_{i,j}$ と成長速度を保持するスカラー量 $F_{i,j}$ を割り当てる．また各時刻での境界の位置を格納する変数 $P_k=(p_{x,k},p_{y,k}),(k=0\sim N_{max}-1)$，および繰り返し回数を表す変数 n を準備する．ただし N_{max} は最大境界長である．後述する Narrow Band を用いる場合には，各グリッドが処理領域内かを表すフラグ $S_{i,j}$ も用意する．

Step.2（初期化） ある適当な初期閉曲線 $C(\mathbf{p},0)=C_0(p)$ を，通常は画像全体を取り囲むように設定し，その閉曲線上のグリッドに対しては補助関数値を $\phi_{i,j}=0$，その他のグリッドに対しては閉曲線からの符号付距離（境界の内側が負，外側が正）を与える．Narrow Band を用いる場合には，初期閉曲線からからの距離が δ 以内であるグリッドに，処理領域内であることを示すフラグを設定する．ただし δ は Narrow Band の幅である．

Step.3（成長速度の計算） n に 1 を加え，全グリッド（Narrow Band を用いる場合には，フラグの設定されたグリッド）において，成長速度 $F_{i,j}$ を計算する．これには，後述のように 1) 局所成長速度場と 2) 拡張成長速度場の 2 種類が考えられるが，両者とも閉曲線上のグリッド (i,j) での成長速度 $F_{i,j}$ は，例えば

$$F_{i,j} = k_{I,i,j}(a - b\kappa_{i,j}) \tag{18}$$

で与えられる．ただし $k_{I,i,j}$ は輝度勾配に関する項，$a, b \geq 0$ は定数であり，例えば $k_{I,i,j}$ として，

$$k_{I,i,j} = \frac{1}{1 + \nabla G \otimes I(i,j)} \tag{19}$$

などが考えられる．ただし，$I(i,j)$ はグリッド (i,j) での濃淡画像の輝度値である．また $\kappa_{i,j}$ は補助関数値の曲率であり，補助関数 $\phi_{i,j}$ を用いて以下のように計算できる．

$$\kappa_{i,j} = \nabla \left(\frac{\nabla \phi_{i,j}}{|\nabla \phi_{i,j}|} \right)$$

$$= \frac{\phi_{xx}\phi_y^2 - 2\phi_x\phi_y\phi_{xy} + \phi_{yy}\phi_x^2}{(\phi_x^2 + \phi_y^2)^{\frac{3}{2}}} \tag{20}$$

ただし

$$\phi_x = \frac{\phi_{i+1,j} - \phi_{i,j}}{h} \tag{21}$$

$$\phi_y = \frac{\phi_{i,j+1} - \phi_{i,j}}{h} \tag{22}$$

$$\phi_{xx} = \frac{\phi_{i+1,j} - 2\phi_{i,j} + \phi_{i-1,j}}{h^2} \tag{23}$$

$$\phi_{yy} = \frac{\phi_{i,j+1} - 2\phi_{i,j} + \phi_{i,j-1}}{h^2} \tag{24}$$

$$\phi_{xy} = \frac{\phi_{i+1,j+1} + \phi_{i,j} - \phi_{i+1,j} - \phi_{i,j+1}}{h^2} \tag{25}$$

であり，h は離散化幅である．これにより，輝度勾配が小さいと境界の進行速度が大きくなり，境界はその内側方向へ移動する．一方，輝度勾配が十分に大きい場所では速度が 0 に近くなり，境界はその場所で停止する．また曲率が $\kappa \geq \frac{a}{b}$ の場合には速度が負になり，境界は外側方向へ移動する．

Step.4（補助関数の更新） 全グリッド（Narrow Band を用いる場合には，フラグの設定されたグリッド）において，補助関数値を以下のいわゆる Upwind Scheme に従って更新する．

$$\phi_{ij} \leftarrow \phi_{ij} -$$

$$\Delta t(\max(F_{ij}, 0)\nabla^+ + \min(F_{ij}, 0)\nabla^-) \tag{26}$$

ただし，

$$\nabla^+ = (\max(D_{ij}^{-x}, -D_{ij}^{+x}, 0)^2 + \max(D_{ij}^{-y}, -D_{ij}^{+y}, 0)^2)^{\frac{1}{2}} \tag{27}$$

$$\nabla^- = (\max(D_{ij}^{+x}, -D_{ij}^{-x}, 0)^2 + \max(D_{ij}^{+y}, -D_{ij}^{-y}, 0)^2)^{\frac{1}{2}} \tag{28}$$

$$D_{ij}^{+x} = \frac{\phi_{i+1,j} - \phi_{i,j}}{h} \quad D_{ij}^{+y} = \frac{\phi_{i,j+1} - \phi_{i,j}}{h}$$
$$D_{ij}^{-x} = \frac{\phi_{i,j} - \phi_{i-1,j}}{h} \quad D_{ij}^{-y} = \frac{\phi_{i,j} - \phi_{i,j-1}}{h} \tag{29}$$

であり，Δt は積分間隔である．

Step.5（Zero level set の検出） $\phi_{ij} = 0$ となる位置を検出し，次時刻での閉曲線 $C(\mathbf{p}, t)$ とする．また，この位置を $P_k = (p_{x,k}, p_{y,k})$, $(k = 0 \sim N-1)$ に格納する．

Step.6（再初期化） n の適当な間隔（Narrow Band を用いる場合には，境界が Narrow Band の端に近づいた場合）で，4.7 で説明する再初期化処理を行い，P_k では補助関数値を $\phi_{p_{x,k},p_{y,k}} = 0$，その他のグリッドに対しては閉曲線からの符号付距離をセットする．Narrow Band を用いる場合には，現在の曲線からからの距離が δ 以内であるグリッドに処理領域内であることを示すフラグを再設定する．

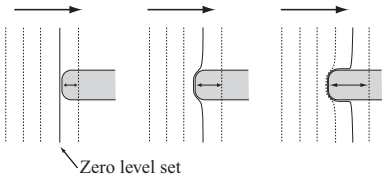
Step.7 Step.3 $\sim$ 6 を $\phi_{i,j}$ の変化量が閾値以下になるか，n が予め決められた繰り返し回数を超えるまで繰り返す．

なお，式（18）は κ を 3 次元曲率にすることで，3 次元空間にも拡張できる．また式（18）以外にも，指数関数を用いた成長速度の決定法 [19] なども提案されている．

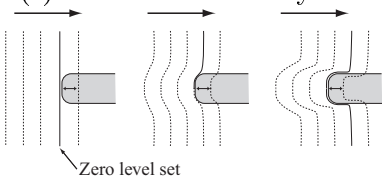
4.5 局所成長速度場と拡張成長速度場

前章の Step.3 において，各グリッドの成長速度 $F_{i,j}$ を決定するために，局所成長速度場（No extension velocity field）と拡張成長速度場（Extension velocity field）の 2 つの手法が提案されている．局所成長速度場とは，各グリッドごとに局所的に式 (18) を計算し，成長速度を決定するものである．一方，拡張成長速度場とは，まず zero level set での成長速度を式 (18) により決定し，その他のグリッドでは最も近い zero level set での成長速度をコピーして用いるものである．

両者の境界進行の様子を図 4 に示す．局所成長速度（図 4（a））は，zero level set が境界に到達し，その部分の進行速度が減少しても，その情報が周辺に伝わらず，周辺の速度には影響がない．そのため，進行を続けるうちに補助関数場が境界外側では密に，内側では粗になり，さらに境界周辺以外では関数場は変化し続け，停止することはない．一方，拡張成長速度（図 4（b））は，zero level set が境界に到達すると，その情報が周囲に伝わって周囲の進行速度も減少させるため，進行を続けても局所成長速度の場合のように補助関数場が不均一になることがない．Sethian[26] も多くの計算機実験から，拡張成長速度場のほうが解が安定に求まり，最終的に得られる zero level set の形状も高精度であることを示している．



(a) No extension velocity field



(b) Extension velocity field

図 4: 局所成長速度と拡張成長速度での境界進行の比較

4.6 Narrow Band

境界領域の追跡では，空間全体に対して補助関数を計算する必要はなく，ゼロ等高面に近い領域だけを対象にすればよい．そこで図 5 に示すように，ゼロ等高面に近い領域に細長い帯状の領域を設定し，その領域に含まれる格子点でのみ，補助関数の更新やゼロ投稿面の検出を行うことで，計算コストが削減できる．この手法は Narrow Band Method [6] [1] と呼ばれ，Level Set Method における計算コストの削減に対する最も代表的な手法である．さらにこの手法では，ゼロ等高面が Narrow Band 領域の境界に近づいたときのみ，Narrow Band 領域内で再初期化処理を行うことで，さらに計算コストが削減できる．

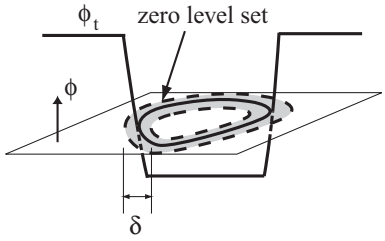


図 5: Narrow Band

4.7 再初期化

PDE を数値的に解き，Upwind Scheme により補助関数を更新する場合，更新とともに積分誤差も積算されるため，安定な解を得るには一定回数更新後に各グリッドごとに補助関数の値（一般には現在の zero level set からの符号付距離）を再計算し，以降の計算の初期値として設定する「再初期化」の処理 [25] が必要である．しかし再初期化処理で各グリッドにおいて現在の zero level set からの距離を求めるには，各グリッド点の最近傍 zero level set を探索する必要があり，計算コストが高い．これに対し，Adalsteinsson ら [2] は補助関数の計算が静的な境界追跡手法と等しいことを利用し，Fast Marching Method を用いた高速な補助関数の計算手法を提案している．また著者らも再初期化処理と拡張成長速度場の構築処理を統合することで，事実上計算量ゼロで再初期化を行う手法を提案している [36].

4.8 差分法を用いた Level Set Method

4.4 では，Upwind Scheme を用いた Level Set Method の実装法を紹介した．しかし流体数値シミュレーションなどの分野で広く用いられている差分法を利用した，より安定で効率のよい Level Set Method の実装手法が提案されている [28] [8]．本章では，それらの手法について説明する．

まず，式 (16),(18) を簡単化した，以下の PDE を考える．

$$\frac{\partial \phi}{\partial t} = \kappa \left|\nabla \phi\right| \tag{30}$$

ただし，

$$\kappa = \nabla \left(\frac{\nabla \phi}{\left|\nabla \phi\right|} \right) \tag{31}$$

である．式 (30) を以下のように離散化する．

$$\frac{\phi^n - \phi^{n-1}}{\Delta t} = \nabla \left(\frac{(\phi_x^n, \phi_y^n)}{\left|\nabla \phi^n\right|} \right) \left|\nabla \phi^n\right| \tag{32}$$

ここで，

$$\phi_x^n = \frac{\phi_{i+1,j}^n - \phi_{i,j}^n}{h} \tag{33}$$

$$\phi_y^n = \frac{\phi_{i,j+1}^n - \phi_{i,j}^n}{h} \tag{34}$$

とすると[1]，式 (32) は

$$\begin{aligned}
\frac{\phi_{i,j}^n - \phi_{i,j}^{n-1}}{\Delta t} &= \Bigg(\frac{\frac{\phi_{i+1,j}^n - \phi_{i,j}^n}{h\left|\nabla \phi_{i+\frac{1}{2},j}^n\right|} - \frac{\phi_{i,j}^n - \phi_{i-1,j}^n}{h\left|\nabla \phi_{i-\frac{1}{2},j}^n\right|}}{h} \\
&\quad + \frac{\frac{\phi_{i,j+1}^n - \phi_{i,j}^n}{h\left|\nabla \phi_{i,j+\frac{1}{2}}^n\right|} - \frac{\phi_{i,j}^n - \phi_{i,j-1}^n}{h\left|\nabla \phi_{i,j-\frac{1}{2}}^n\right|}}{h} \Bigg) \left|\nabla \phi_{i,j}^n\right| \\
&= \frac{2\left|\nabla \phi_{i,j}^n\right|}{\left|\nabla \phi_{i+1,j}^n\right| + \left|\nabla \phi_{i,j}^n\right|} \frac{\phi_{i+1,j}^n - \phi_{i,j}^n}{h^2} \\
&\quad + \frac{2\left|\nabla \phi_{i,j}^n\right|}{\left|\nabla \phi_{i-1,j}^n\right| + \left|\nabla \phi_{i,j}^n\right|} \frac{\phi_{i-1,j}^n - \phi_{i,j}^n}{h^2} \\
&\quad + \frac{2\left|\nabla \phi_{i,j}^n\right|}{\left|\nabla \phi_{i,j+1}^n\right| + \left|\nabla \phi_{i,j}^n\right|} \frac{\phi_{i,j+1}^n - \phi_{i,j}^n}{h^2}
\end{aligned}$$

[1]実際には，より精度の高い Crank-Nicolson 法などで 2 次近似するほうがよい

$$+\frac{2\left|\nabla\phi^n_{i,j}\right|}{\left|\nabla\phi^n_{i,j-1}\right|+\left|\nabla\phi^n_{i,j}\right|}\frac{\phi^n_{i,j-1}-\phi^n_{i,j}}{h^2}$$
$$=A_{i+1,j}(\phi^n_{i+1,j}-\phi^n_{i,j})+A_{i-1,j}(\phi^n_{i-1,j}-\phi^n_{i,j})$$
$$+A_{i,j+1}(\phi^n_{i,j+1}-\phi^n_{i,j})+A_{i,j-1}(\phi^n_{i,j-1}-\phi^n_{i,j})$$
$$=\sum_{k,l=\{\pm1\},k\neq l}A_{i+k,j+l}(\phi^n_{i+k,j+l}-\phi^n_{i,j}) \quad (35)$$

となる．ただし，

$$\left|\nabla\phi^n_{i+\frac{1}{2},j}\right|=\frac{\left|\nabla\phi^n_{i+1,j}\right|+\left|\nabla\phi^n_{i,j}\right|}{2} \quad (36)$$

$$A_{i+k,j+l}=\frac{2\left|\nabla\phi^n_{i,j}\right|}{\left|\nabla\phi^n_{i+k,j+j}\right|+\left|\nabla\phi^n_{i,j}\right|} \quad (37)$$

である．より一般的に，

$$\frac{\partial\phi}{\partial t}=-k_I(a-b\kappa)\left|\nabla\phi\right| \quad (38)$$

の場合には，同様の手順により，

$$A_{i+k,j+l}=k_Ib\frac{2\left|\nabla\phi^n_{i,j}\right|}{\left|\nabla\phi^n_{i+k,j+j}\right|+\left|\nabla\phi^n_{i,j}\right|} \quad (39)$$

とし，式 (35) の右辺に

$$-k_Ia\left|\nabla\phi^n_{i,j}\right| \quad (40)$$

を加えればよい．

式 (35) より，

$$(1+\sum_{k,l=\{\pm1\},k\neq l}A_{i+k,j+l}\Delta t)\phi^n_{i,j}$$
$$-\sum_{k,l=\{\pm1\},k\neq l}A_{i+k,j+l}\Delta t\phi^n_{i+k,j+l}=\phi^{n-1}_{i,j} \quad (41)$$

となる．ここで $\phi^n_{i,j}$ を一列に並べたベクトル $\mathbf{u}=\{\phi^n_{i,j}\}$ を考える．これより上式は

$$\mathbf{A}\mathbf{u}^n=\mathbf{u}^{n-1} \quad (42)$$

の形に整理でき，$\mathbf{A}$ が正則行列であれば，

$$\mathbf{u}^n=\mathbf{A}^{-1}\mathbf{u}^{n-1} \quad (43)$$

と求められる．この手法は implicit method と呼ばれ，積分間隔 Δt が大きい場合でも安定に解を求めることができるという特徴がある．しかし行列 A は一般に巨大であり，かつ大部分の要素は 0 であることから，計算の高速化，安定化を目的とした式 (43) の近似解法が多く提案されている．そこで本稿では代表的な手法である ADI(Alternating Direction Implicit) 法と AOS(Additive Operator Splitting) 法 [12] [28] [8] を紹介する．

4.8.1 ADI 法

ADI(Alternating Direction Implicit) 法 [21] は，式 (35) に対して，X 方向の解と Y 方向の解を逐次的に計算する方法である．

まず X 方向を考える．ただし，Y 方向は前時刻 $n-1$ の値を利用する．補助変数 $\phi^*_{i,j}$ に対して，

$$\frac{\phi^*_{i,j}-\phi^{n-1}_{i,j}}{\Delta t}=$$
$$A_{i+1,j}(\phi^*_{i+1,j}-\phi^*_{i,j})+A_{i-1,j}(\phi^*_{i-1,j}-\phi^*_{i,j})$$
$$+A_{i,j+1}(\phi^{n-1}_{i,j+1}-\phi^{n-1}_{i,j})+A_{i,j-1}(\phi^{n-1}_{i,j-1}-\phi^{n-1}_{i,j}) \quad (44)$$

より

$$\begin{pmatrix} a_{i-1,j} & a_{i,j} & a_{i+1,j}\end{pmatrix}\begin{pmatrix}\phi^*_{i-1,j}\\ \phi^*_{i,j}\\ \phi^*_{i+1,j}\end{pmatrix}=b_{i,j} \quad (45)$$

とする．ただし，

$$a_{i-1,j} = -A_{i-1,j}\Delta t \quad (46)$$
$$a_{i,j} = 1+A_{i-1,j}\Delta t+A_{i+1,j}\Delta t \quad (47)$$
$$a_{i+1,j} = -A_{i+1,j}\Delta t \quad (48)$$
$$b^{n-1}_{i,j} = \phi^{n-1}_{i,j}+A_{i,j+1}\Delta t(\phi^{n-1}_{i,j+1}-\phi^{n-1}_{i,j})$$
$$+A_{i,j-1}\Delta t(\phi^{n-1}_{i,j-1}-\phi^{n-1}_{i,j}) \quad (49)$$

である．

ここで $\phi^*_{i,j}$ を一列に並べたベクトルを $\mathbf{u}^*=\{\phi^*_{i,j}\}$ とする．これにより，式 (45) は以下の 3 重対角行列 $\mathbf{A}_x$ を用いた形で表現できる．

$$\mathbf{A}_x\mathbf{u}^*=\mathbf{b}^{n-1} \quad (50)$$

ただし，

$$\mathbf{A}_x=\begin{pmatrix}\ddots & \ddots & \ddots & & 0\\ & a_{i-1,j} & a_{i,j} & a_{i+1,j} & \\ 0 & & \ddots & \ddots & \ddots\end{pmatrix} \quad (51)$$

であり，$\mathbf{b}^{n-1}=\{b^{n-1}_{i,j}\}$ である．

これより，$\mathbf{A}_x$ が正則行列であれば，補助変数 $\phi^*_{i,j}$ の解は

$$\mathbf{u}^*=\mathbf{A}_x^{-1}\mathbf{b}^{n-1} \quad (52)$$

と求められる．

次に Y 方向も同様に，式 (45) に対応した 3 重対角行列 $\mathbf{A}_y$ を導く．ただし，ここで式 (50) の $\mathbf{b}^{n-1}$ の代わりに，X 方向に対して求められた補助変数 $\phi^*_{i,j}$ を用いてベクトル $\mathbf{b}^*$ を計算し，式 (35) の解を以下のように求める．

$$\mathbf{A}_y\mathbf{u}^n=\mathbf{b}^* \quad (53)$$
$$\mathbf{u}^n=\mathbf{A}_y^{-1}\mathbf{b}^* \quad (54)$$

4.8.2 AOS 法

AOS(Additive Operator Splitting) 法 [23] とは，式 (35) に対して，X 方向の解と Y 方向の解を別々に求め，その平均値を式 (35) の解とする手法である．具体的には，式 (50)(53) に対応する次式を導き，それぞれの解 $\mathbf{u}^*, \mathbf{u}^{**}$ を求める．

$$\mathbf{A}_x \mathbf{u}^* = \mathbf{u}^{n-1} \tag{55}$$

$$\mathbf{A}_y \mathbf{u}^{**} = \mathbf{u}^{n-1} \tag{56}$$

次にそれらの平均として，式 (35) の解を以下のように求める．

$$\mathbf{u}^n = \frac{\mathbf{u}^* + \mathbf{u}^{**}}{2} \tag{57}$$

4.9 Geodesic Active Contour

式 (10) および (18) で定義された成長速度を有する曲線は，Geometric Active Contour と呼ばれ，特に $C_t = \kappa \mathbf{N}$ で表される曲線は，以下の $\mathcal{L}$ を最小にする．

$$\mathcal{L} = \oint |C'(q)|\, dq = \oint ds \tag{58}$$

Level Set Method を用いた Geometric Active Contour は，Snakes と比較して位相変化に対応可能であるという優れた特徴を有するが，一方で対象によっては曲線の成長が不安定で，ノイズやギャップ (切れ) に対して敏感すぎるという欠点があった．これに対し Caselles ら [5] は，以下の $\mathcal{L}_R$ を最小にする Geodesic Active Contour を提案している [20]．[2]

$$\mathcal{L}_R = \oint g(C(q))\, |C'(q)|\, dq = \oint g(C(q)) ds \tag{59}$$

ただし，$g(C(q))$ は境界の情報を含む関数である．これにより Geodesic Active Contour は，式 (58) で表される単純な距離最小ではなく，境界の特徴も考慮した距離（測地距離）を最小化する．式 (59) で定義された $\mathcal{L}_R$ に最急降下法を適用すると次式が得られる．(詳細は [5] 参照)

$$\begin{aligned} C_t &= g(I)\kappa\mathbf{N} - (\nabla g(I)\cdot\mathbf{N})\mathbf{N} \\ &= -F'(\kappa)\mathbf{N} \end{aligned} \tag{60}$$

これより，式 (16) は

$$\phi_t = -F'(\kappa) \mid \nabla\phi \mid \tag{61}$$

$$= (g(I)\kappa - (\nabla g(I)\cdot\mathbf{N})) \mid \nabla\phi \mid \tag{62}$$

$$= g(I)\kappa \mid \nabla\phi \mid - \nabla g(I)\cdot\nabla\phi \tag{63}$$

となる．

式 (16),(18) で $a = 0, b = 1$ とした従来の Geometric Active Contour では，

$$\phi_t = g(I)\kappa \mid \nabla\phi \mid \tag{64}$$

ただし

$$g(I) = k_I \tag{65}$$

と書けることから，Geometric Active Contour と Geodesic Active Contour の差は，第 2 項の $\nabla g(I)\cdot\nabla\phi$ の有無である．従来の Geometric Active Contour では，この項がないため，境界は $g(I) = k_I = 0$ のときのみ停止する．しかし実際の画像には，ぼけやノイズ，ギャップが含まれるため，検出すべき全ての境界で正確に $g(I) = k_I = 0$ が成立することは考えられない．これに対し，$g(I)$ に式 (19) の $k_{I,i,j}$ を用いた場合，Geodesic Active Contour の第 2 項 $\nabla g(I)\cdot\nabla\phi$ の $\nabla g(I)$ は理想的な境界の中心を示しており，曲線はこの中心方向へ動く．このため，曲線は画像のぼけやノイズ，ギャップに対して頑強に境界を追跡できる．

4.10 Fast Marching Method

成長速度が常に $F > 0$，すなわち境界が外側にのみ動く場合を考える．このとき，ある初期時刻 $t = 0$ で定義された初期境界 C が，速度 F で移動したときの位置 x での到達時刻を $T(x)$ とする．ただし簡単のため，ここでは x は 1 次元とする．このとき，F と $T(x)$ には，距離=速度 $\times$ 時間より次式が成り立つ．

$$\Delta x = F \times \Delta T(x) \tag{66}$$

すなわち

$$\frac{dT(x)}{dx} F = 1 \tag{67}$$

より一般的には，上式は

$$\mid \nabla T(p) \mid F = 1 \tag{68}$$

となり，これは Eikonal 方程式と呼ばれる．このように境界の移動方向が一方向である場合には，境界の進行を到達時刻 $T(p)$ で置き換え，式 (68) の解として表現することができる．

Fast Marching Method は，この Eikonal 方程式の高速な数値解法として提案された．この方程式は通常，収束計算により解かれるため，膨大な計算時間が必要である．しかし，Fast Marching Method では，成長速度の符号が固定という条件を加え，到達時刻の小さい点から大きい点へ一方向に到達時刻を確定していくこと

[2] Geodesic curve とは，2 地点間を結ぶ測地距離が最短の曲線

で，収束計算を行うことなく高速にEikonal方程式の解を導くことができる．

Fast Marching Methodでは，まずEikonal方程式を次の差分方程式に置き換える．

$$(\max(D_{ij}^{-x}T, -D_{ij}^{+x}T, 0)^2 + \max(D_{ij}^{-y}T, -D_{ij}^{+y}T, 0)^2)^{\frac{1}{2}} = 1/F_{ij} \qquad (69)$$

次に，境界は到達時間が小さい場所から大きい場所へと一方向に伝播することから，到達時間の小さい場所から順に式（69）を解き，各点の到達時間を確定する．

具体的には以下のようになる．

Step.1（初期化） まず，次の処理によって全てのグリッドを3つのリストのいずれかに追加する．

1. 初期境界に属するグリッドを *known* のリストに追加し，到達時間を0にセットする（T=0）．
2. *known* の4近傍のグリッドのうち，*known* でないグリッドを *trial* に追加し，そのグリッドの到達時間を $T_{ij} = 1/F_{ij}$ により計算して仮登録する．また，これらのグリッドを到達時間に関する昇順のHeap形式で保存する．
3. 上記以外のピクセルを *far* に追加し，到達時間を無限大とする（T=∞）．

Step.2 Heapの先頭に置かれた，*trial* のリスト中で到達時間 T が最も小さいグリッド (i_{min}, j_{min}) を選択し，そのグリッドを *trial* のリストとHeapから除外し，*known* のリストに追加する．Heapから除外する際，同時にHeapを再構築（downheap）する．

Step.3 現在選択されたグリッド (i_{min}, j_{min}) の4近傍 $((i_{min}-1, j_{min})$,$(i_{min}+1, j_{min})$, $(i_{min}, j_{min}-1)$,$(i_{min}, j_{min}+1))$ のうち，*far* のリストに属しているグリッドを *trial* のリストに追加する．

Step.4 現在選択されたグリッド (i_{min}, j_{min}) の4近傍のうち，*trial* に属している近傍点の到達時間を式（69）を用いて計算して仮登録し，Heapを再構築（upheap）する．

Step.5 *trial* に属しているグリッドが存在すればStep.2へ戻る．それ以外の時は処理を終了する．

5 事例紹介

本章では著者らの提案した，高速で安定なLevel Set Methodの実装法であるFast Level Set Method(FLSM)を紹介する[31] [36] [35]．さらにFLSMを用いてビデオ画像上の移動物体のリアルタイム追跡実験を行った結果を紹介する．

5.1 Level Set Methodの高速計算法

これまでに示したLevel Set Methodの実装法では，再初期化処理や各更新ごとの拡張成長速度場の構築に多くの計算量が必要であり，計算の高速化が大きな課題となっている．この問題に対してこれまでに，Narrow Band Method[26] [1] やFast Marching Method[24] [2] [26]，SFA（Space Field Algorithm）[29]，Hermes[18] [20]，AOS（Additive Operator Splitting）[12] [28] [8] あるいは初期境界の最適設定[33]や多重解像度制御[22]などの手法が提案されている．

これに対し，著者らも高速で安定なLevel Set Methodの解法として，i) 高速な拡張成長速度場の構築（Fast Narrow Band Method, FNB[31]），ii) 補助関数の再初期化処理の高速化と頻繁な再初期化，を特徴とするFast Level Set Method[36] [35]を提案している．そこで本章ではこの手法について解説する．

5.1.1 高速な拡張成長速度場の構築法

まず，図6に示すような参照マップをあらかじめ作成する．これは，原点周辺にあるグリッドを原点からの距離に応じて分類したものである．つまり，原点からの2乗距離が r であるグリッドの集合を R_r とし，$r = 0 \sim \delta(\delta+1)$ に対するリスト $R_0, R_1, \cdots, R_{\delta(\delta+1)}$ をそれぞれ作成する．なお，ここで距離にはグリッド中心間のユークリッド距離を用いることとし，また $\delta > 0$ はNarrow Bandのバンド幅である．また，バンド幅 δ のNarrow Band領域は，各zero level setからの距離を小数点以下で四捨五入した整数値が δ 以下になるようなグリッドの集合と定義する．これは，$\delta(\delta+1) < (\delta+0.5)^2 < \delta(\delta+1)+1$ が常に成り立つことから，zero level setからの2乗距離が $\delta(\delta+1)$ 以下のグリッドの集合ともみなせる．一例として図6に，バンド幅 $\delta = 3$ における参照マップ（$R_0 \sim R_{12}$）を示す．グリッドに書き込まれている数字（r）は，属しているリスト（R_r）を示す．

次に，作成した参照マップを用いて拡張成長速度場を構築する．ただし，zero level setでの成長速度は式（18）等によりあらかじめ決定されているものとする．まず，リスト $R_{\delta(\delta+1)}$ を用いて，あるzero level setからの2乗距離が $\delta(\delta+1)$ であるようなグリッドを選択し，そのzero level setに格納されている成長速度の値

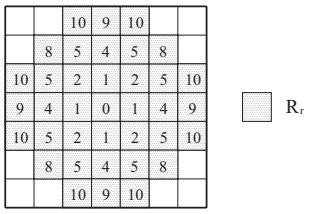


図 6: 参照マップ

を選択されたグリッドに仮登録する．この処理をすべての zero level set に対して行う．次に，添字の値を 1 小さくして同じ処理を行い，これを添字の値が 0 になるまで繰り返す．ただし，仮登録の際，異なる値がすでに仮登録されていた場合には，新たな値を上書きすることにする．これにより全ての処理が終了した時には，各グリッドには最も近い zero level set における成長速度の値が登録されている．このように，参照マップ内の距離に応じたリストを利用することで，距離比較を行うことなく代入処理だけで拡張成長速度場が構築できる．以上の手法を Fast Narrow Band Method （FNB）[31] と呼ぶ.

一例として，図 7 にバンド幅 $\delta = 3$ の場合の処理の流れ（(a) $\rightarrow$ (d)）を示す．図 7 (a) は，成長速度の値が F_1 である zero level set に対し，そこからの 2 乗距離が 10 であるグリッドをリスト R_{10} を用いて選択し，そのグリッドの成長速度を F_1 とした様子である．図 7 (b) では，全ての zero level set からの 2 乗距離が 10 であるグリッドに，各々の zero level set に格納されている成長速度の値を登録している．図 7 (c) は，成長速度の値が F_1 である zero level set に対し，そこからの 2 乗距離が $9(= 10 - 1)$ であるグリッドをリスト R_9 を用いて選択し，そのグリッドの成長速度を F_1 としている．ただし，図 7 (b) では，成長速度の値が F_1 でないグリッドに F_1 の値を上書きしている．この処理を繰り返すことで，図 7 (d) のように拡張成長速度場が構築できる.

5.1.2 高速で安定な Level Set Method

上述した拡張成長速度場の構築処理は，各グリッドで現在の zero level set からの距離に応じて成長速度を上書きするものである．そこで，その過程で距離値も同時に上書きすることで，各グリッドに zero level set からの距離を簡単に設定でき，これにより安定な計算に不可欠な再初期化処理が実現できる．また，この際追加される処理は単なるメモリーアクセスだけであり，全体の計算量はほとんど変化しない.

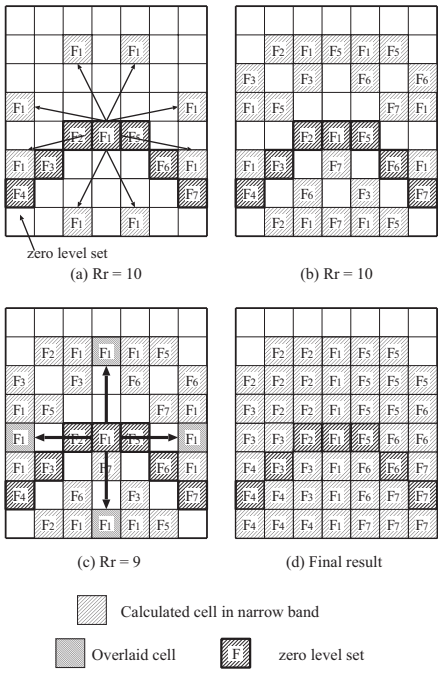


図 7: 拡張成長速度場の構築過程

さらに拡張成長速度場の計算は各更新時ごとに行われるので，再初期化の処理もほとんど計算量を増やすことなく，最大で各更新時ごとに行うことができる．これにより，大きな積分時間幅でも安定して解を求めることができ，高速で安定な Level Set Method (Fast Level Set Method[36]) が実現できる.

また上記の手法は，さらに参照マップの分割による適応的書き込み処理を行うことで，より一層の高速化が可能である [36] .

5.1.3 局所成長速度手法との比較

局所成長速度場は局所的な情報だけで成長速度を計算できるため，一般には最近傍点探索の必要な拡張成長速度場よりも高速に構築できると考えられる．しかし局所成長速度場の計算は各グリッドで境界かどうかの判定や補助関数の曲率などをもとに成長速度を計算する必要があり，実際には局所成長速度場の構築にもある程度の時間が必要である.

一方，提案した拡張成長速度場の構築法では，成長速度の計算は境界グリッドだけであり，その他のグリッドは境界グリッドで計算された成長速度をルールに従って上書きするだけである.

したがって提案した手法により最近傍点探索を単純な上書き処理に置き換え，拡張成長速度場の構築に必要な時間を短縮できれば，全体の処理時間を局所成長速度場の構築に近く，あるいは逆転できる可能性がある.

そこで従来の局所成長速度場を利用する手法と本論文で提案した Fast Level Set Method の計算量の比較を表 1 に示す．ただし，空間は 2 次元で大きさは $n \times n$ グリッド，Narrow Band の幅は δ とする．

まず再初期化に必要な計算量は，Fast Marching 法を用いた場合が $O(\delta n \log n)$ であるのに対し，提案手法では $O(\delta^2 n)$ である．ただし前述のように，提案手法ではこの再初期化処理を拡張成長速度場の構築に統合でき，このときの全体の計算量は再初期化の有無にはほぼ無関係，すなわち再初期化処理を見かけ上，計算量 0 で実行できる．

次に成長速度場の構築に必要な計算量について考える．まず局所成長速度場を用いた場合，成長速度の計算を全ての Narrow Band 領域で行うため，局所成長速度場の構築に $O(\delta n)$ の計算量が必要である．一方，提案した Fast Level Set Method では，成長速度の計算は zero level set だけで行うため，その計算量は $O(n)$ である．しかし，拡張成長速度場の構築には，zero level set で計算された成長速度を Narrow Band 領域に代入するため，これに加えて $\delta^2 n$ 回の代入処理が必要となる．

以上より，成長速度場の構築に対しては，個々の成長速度の計算が複雑で時間がかかる場合，局所成長速度場に比べて成長速度の計算回数の少ない拡張成長速度場の利用が有利である．しかしメモリーアクセスが遅く，代入に長い処理時間が必要であったり，バンド幅 δ が大きい場合には，局所成長速度場や Fast Marching 法を用いた再初期化が有利であると思われる．

表 1: 従来の Level Set Method と提案した Fast Level Set Method の計算量

	No Extension with Fast Marching (LSM)	Extension (FLSM)
Reinitialization	$O(\delta n \log n)$	$(O(\delta^2 n))$
Construction of velocity field	$O(\delta n)$	$O(\delta^2 n)$
Updating process	$O(\delta n)$	$O(\delta n)$
Detection of zero level set	$O(\delta n)$	$O(\delta n)$

() $\cdots$ The reinitialization process of FLSM can be integrated with the construction process of the extension velocity field.

5.2 ビデオ画像上の移動物体のリアルタイム検出と追跡（Level Set Tracking）

ビデオ画像に対して提案した Fast Level Set Method を適用し，移動物体の検出とリアルタイム追跡実験を行った．使用した画像のサイズ，入力速度はそれぞれ 320x240pixel，30Hz，使用した計算機は Pentium IV，2GHz であり，Fast Level Set Method の処理は約 60Hz で実行されている．また成長速度の最大値は 1 ピクセル，積分時間幅は 4 であり，Narrow Band の幅は 5 ピクセルとした．

実験ではまず背景差分により移動物体の領域を大まかに検出し，Fast Level Set Method により検出した移動物体領域の中心から外側へ zero level set を進行させて，濃淡値が急激に変化する領域を移動物体の境界として検出，追跡した．ただし式（18）で用いる輝度勾配項 k_I は，以下の式により背景差分画像 $D(x,y) = I(x,y) - I_{back}(x,y)$ と濃淡画像 $I(x,y)$ の両方から決定した．

$$k_I = \frac{1}{1 + \min(|\nabla I(x,y)|, |\nabla D(x,y)|)} \tag{70}$$

図 8 に複数の領域が交差し，分離している場合の追跡結果を示す．このように当初 2 つの閉曲線で表されていた移動物体の境界が，移動物体が交差したことで 1 つの閉曲線に統合され，次の時刻で再度 2 つの閉曲線に分離しており，境界の位相変化に柔軟に対応できている．

5.3 3 次元モデリングへの適用

提案した Fast Level Set Method は，3 次元空間における形状復元にも拡張できる．図 9 は，複雑な位相をもつ物体（バスケット）に本手法を適用した結果 [36] であり，境界の進行途中での位相変化に正確に対応できている．また図 10 は，複数ステレオカメラを用いたモーションキャプチャシステムへの適用例である [9]. Fast Level Set Method を用いることで，高速かつノイズや遮蔽に頑強に 3 次元形状が復元でき，また並列計算も容易である．

6 まとめ

本稿では，Active Contour Model（動的輪郭モデル）の代表的な手法として，Kass らの Snakes と Osher, Sethian らの Level Set Method に焦点を当て，その理論と実装法を概説した．また高速で安定な Level Set Method の解法として，著者らの提案する Fast Level

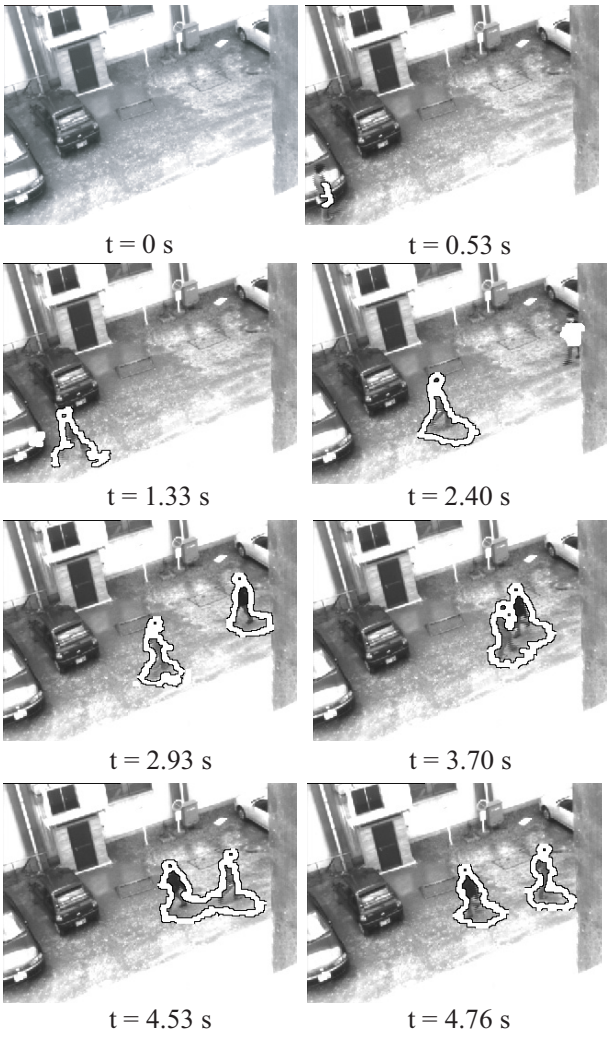


図 8: 複数物体のリアルタイム追跡

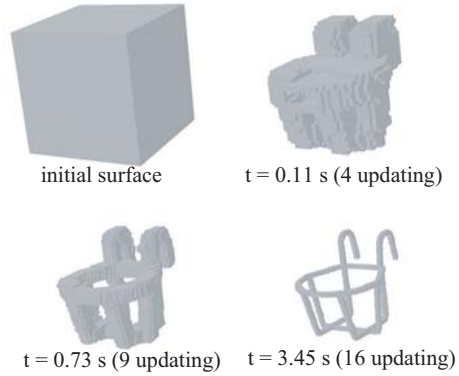


図 9: 複雑な位相をもつ物体の 3 次元形状復元

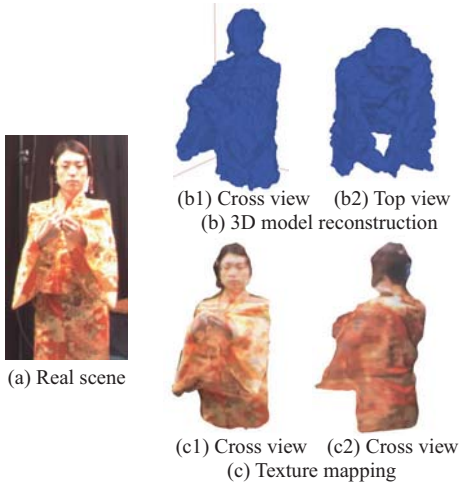


図 10: 複数ステレオカメラを用いたモーションキャプチャシステム

Set Method を紹介し，ビデオ画像上の移動物体のリアルタイム追跡へ適用した結果，および 3 次元モデリングへの適用例を示した．

謝辞

本稿の執筆に当たり御高閲と多くの御助言を賜りました九州大学 原健二助教授，内田誠一助教授に深甚なる感謝の意を表します．

参考文献

[1] Adalsteinsson, D. and Sethian, J.: A fast level set method for propagating interfaces, *Journal of Computational Physics*, Vol. 118, pp. 269–277 (1995).

[2] Adalsteinsson, D. and Sethian, J.: The fast construction of extension velocities in level set methods, *Journal of Computational Physics*, Vol. 148, pp. 2–22 (1999).

[3] Baillard, C., Hellier, P. and Barillot, C.: Cooperation between level set techniques and dense 3d registration for the segmentation of brain structure, *International Conference on Pattern Recognition*, Vol. I, pp. 991–994 (2000).

[4] Caselles, V., Catte, F., Coll, T. and Dibos, F.: A geometric model for active contours, *Numerische Mathematik*, Vol. 66, pp. 1–31 (1993).

[5] Caselles, V., Kimmel, R., Sapiro, G. and Sbert, C.: Geodesic Active Contours,, *International Journal of Computer Vision*, Vol. 22, No. 1, pp. 61–79 (1997).

[6] Chop, D.: Computing Minimal Surfaces via level Set Curvature Flow, *Journal of Computational Physics*, Vol. 106, pp. 77–91 (1993).

[7] Cohen, L. D. and Cohen, I.: Finite-Element Methods for Active Contour Models and Balloons for 2-D and 3-D Images, *IEEE Tranaction on Pattern Analysis and Machine Intelligence*, Vol. 15, No. 11, pp. 1131–1147 (1993).

[8] Goldenberg, R., Kimmel, R., Rivlin, E. and Rudzsky, M.: Fast Geodesic Active Contours, *IEEE Trans. on Image Processing*, Vol. 10, No. 10, pp. 1467–1475 (2001).

[9] Iwashita, Y., Kurazume, R., Hara, K. and Hasegawa, T.: Robust Motion Capture System against Target Occlusion using Fast Level Set Method, *Proc. IEEE International Conference on on Robotics and Automation,*, pp. 168–174 (2006).

[10] Johan, M., Herve, D., Nicolas, S. and Nicholas, A.: Representation, Shape, Topology and Evolution of Deformable Surfaces. Application to 3D Medical Image Segmentation, Technical Report RR-3954, INRIA report (2000).

[11] Kass, M., Witkin, A. and Terzopoulos, D.: Snakes, Active contour models, *International Journal of Computer Vision*, Vol. 1, No. 4, pp. 321–331 (1988).

[12] Lu, T., Neittaanmaki, P. and Tai, X. C.: A parallel splitting up method and its application to Navier-Strokes equation, *Applied Mathematics Letters*, Vol. 4, No. 2, pp. 25–29 (1991).

[13] Malladi, R., Sethian, J. and Vemuri, B. C.: Shape Modeling with Front Propagation: A level Set Approach, *IEEE Tranaction on Pattern Analysis and Machine Intelligence*, Vol. 17, No. 2, pp. 158–175 (1995).

[14] McInerney, T. and Terzopoulos, D.: Deformable models in medical image analysis, *Medical Image Analysis*, Vol. 1, No. 2, pp. 91–108 (1996).

[15] McInerney, T. and Terzopoulos, D.: T-Snakes: Topology adaptive snakes,, *Medical Image Analysis*, Vol. 4, No. 2 (2000).

[16] Museth, K., Breen, D., Whitaker, R. T. and Barr, A. H.: Level Set Surface Editing Operaters, (2002).

[17] Osher, S. and Sethian, J. A.: Fronts propagating with curvature dependent speed: Algorithm based on Hamilton-Javobi formation, *Journal of Computational Physics*, Vol. 79, pp. 12–49 (1988).

[18] Paragios, N. and Deriche, R.: A PDE-based Level Set approach for Detection and Tracking of moving objects, *International Conference on Computer Vision*, pp. 1139–1145 (1998).

[19] Paragios, N. and Deriche, R.: Geodesic Active Regions for supervised texture segmentation, *International Conference on Computer Vision*, pp. 926–932 (1999).

[20] Paragios, N. and Deriche, R.: Geodesic Active Contours and level sets for detection and tracking of moving objects, *IEEE Tranaction on Pattern Analysis and Machine Intelligence*, Vol. 22, No. 3, pp. 266–280 (2000).

[21] Peaceman, D. W. and Rachford, H. H.: The numerical solution of parabolic and elliptic differential equations, *J. Soc. Indust, Appl. Math*, Vol. 3, pp. 28–41 (1955).

[22] Peng, D., Merriman, B., Osher, S., Zhao, H. K. and Kang, M.: A PDE-based fast local level set method, *Journal of Computational Physics*, Vol. 155, pp. 410–438 (1999).

[23] Perona, P. and Malik, J.: Scale-space and edge detection using anisotropic diffusion, *IEEE Trans. on Pattern. Anal. Machine Intell.*, Vol. 12, pp. 629–639 (1990).

[24] Sethian, J.: A fast marching level set method for monotonically advancing fronts, *Proceedings of the National Academy of Science*, Vol. 93, pp. 1591–1595 (1996).

[25] Sethian, J.: *Level Set Methods, 1st ed.*, Cambridge University Press, New York (1996).

[26] Sethian, J.: *Level Set Methods and Fast Marching Methods second ed.*, Cambridge University Press, UK (1999).

[27] Sethian, J. and Adalsteinsson, D.: An Overview of Level Set Methods for Etching Deposition, and Lithograpy Development, *IEEE Tranaction on Semiconductor Devices*, Vol. 10, No. 1, pp. 167–184 (1996).

[28] Weickert, J., ter Haar, B. M., Romeny and Viergever, M. A.: Efficent and reliable schemes for nonlinear diffusion filtering, *IEEE Transactions on Image Processing*, No. 3, pp. 398–410 (1998).

[29] Whitaker, R.: A level-set approach to 3D reconstruction from range data, *International Journal of Computer Vision*, Vol. 29, No. 3, pp. 203–231 (1998).

[30] Williams, D. J. and Shah, M.: A Fast Algorithm for Active Contours and Curvature Estimation, *CVGIP: Image Understanding*, Vol. 55, No. 1 (1992).

[31] Yui, S., Hara, K., Zha, H. and T.Hasegawa: A fast narrow band method and its application in topology-adaptive 3-D modeling, *International Conference on Pattern Recognition*, Vol. IV, pp. 122–125 (2002).

[32] Yuille, A. Y., Cohen, D. S. and P.W.Hallinan: Feature Extraction from Faces Using Deformable Templates, *Computer Vision and Pattern Recognition*, pp. 104–109 (1989).

[33] Zhao, H. K., Osher, S. and Fedkiw, R.: Fast surface reconstruction using the level set method., *Proceedings of 1st IEEE Workshop on Variational and Level Set Methods* (2001).

[34] 小林景, 杉原厚吉: 情報重みつき結晶成長ボロノイ図の近似構成法とその応用, 電子情報通信学会論文誌, Vol. Vol.J83-A, No. 12 (2000).

[35] 岩下友美, 倉爪亮, 辻徳生, 原健二, 長谷川勉: Fast Level Set Method を用いた複数移動物体の 3 次元追跡, 日本ロボット学会誌, Vol. 23, No. 7, pp. 813–820 (2005).

[36] 倉爪亮, 由井俊太郎, 辻徳生, 岩下友美, 原健二, 長谷川勉: Fast Level Set Method の提案とビデオ画像の移動物体のリアルタイム追跡, 情報処理学会論文誌, Vol. 44, No. 8, pp. 2244–2254 (2003).