コンピュータビジョン

担当: 井尻 敬

Contents

01. 序論 ：イントロダクション
02. 特徴検出1 ：テンプレートマッチング、コーナー検出、エッジ検出
03. 特徴検出2 ：ハフ変換、 DoG，SIFT特徴
04. 領域分割 ：領域分割とは，閾値法，領域拡張法，グラフカット法，
05. オプティカルフロー：領域分割残り，Lucas-Kanade法
06. パターン認識基礎1 ：パターン認識概論，サポートベクタマシン
07. パターン認識基礎2 ：ニューラルネットワーク、深層学習
08. パターン認識基礎3 ：主成分分析, オートエンコーダ
09. プログラミング演習 1：PC室
10. プログラミング演習 2：PC室
11. プログラミング演習 3：PC室
12. プログラミング演習 4：PC室
13. プログラミング演習 5：PC室
14. プログラミング演習 6：PC室

Contents：

画像領域分割の続き
- 動的輪郭モデル
- 曲面再構成法

モーフォロジー演算

オプティカルフロー

概要のみ紹介

Contents：画像領域分割

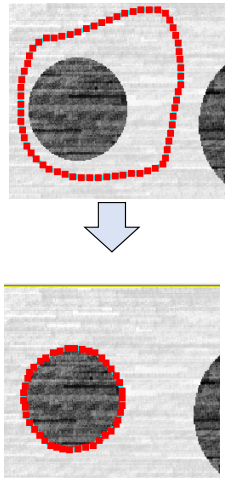
動的輪郭モデル
Active Contours

TActiveContour.exe

動的輪郭モデル（Active Contours）

領域境界を徐々に変形する手法
- 境界形状を滑らかに
- 境界が画像のエッジを通るよう

2手法に分類できる
- 境界を陽的に表現する手法：**Snakes**法
- 境界を陰的に表現する手法：**Level Set**法

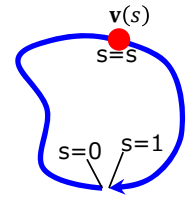


画像のCTデータは理研・画像情報処理研究チームより

参考資料

Snakes法のコスト関数

前提1. 曲線はパラメータ表現される　$\mathbf{v}(s) = \begin{pmatrix} x(s) \\ y(s) \end{pmatrix}$　$s \in [0,1]$



前提2. 曲線のエネルギー

$$E(\mathbf{v}(s)) = \alpha E_{len} + \beta E_{curv} + \gamma E_{img}$$

弧長に対応する項 (長い → 大)

$$E_{len} = \int_0^1 \left\| \frac{d\mathbf{v}(s)}{ds} \right\|^2 ds$$

曲率に対応する項 (ガタガタ → 大)

$$E_{curv} = \int_0^1 \left\| \frac{d^2\mathbf{v}(s)}{ds^2} \right\|^2 ds$$

画像(勾配強度)に対応する項

$$E_{img} = -\int_0^1 \|\nabla(G \otimes I(\mathbf{v}(s)))\| ds$$

画像
ガウスフィルタ(ぼかす)
勾配強度を計算

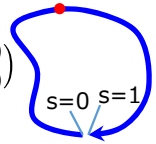
問題 Eを最小化する曲線$\mathbf{v}(s)$ を探す
（動的計画法 / 貪欲法など）

参考資料

Snakes法のコスト関数

最適化計算のため**折れ線近似**する

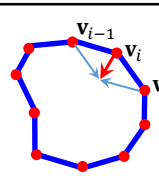
$\mathbf{v}(s) = \begin{pmatrix} x(s) \\ y(s) \end{pmatrix}$
$s \in [0,1]$



$$E_{len} = \int_0^1 \left\| \frac{d\mathbf{v}(\mathbf{s})}{ds} \right\|^2 ds$$

$$E_{curv} = \int_0^1 \left\| \frac{d^2\mathbf{v}(\mathbf{s})}{ds^2} \right\|^2 ds$$

$$E_{img} = -\int_0^1 \|\nabla(G \otimes I(\mathbf{v}(s)))\| ds$$



$i = 1,2, \ldots N$

$$E_{len} \simeq \sum_{i=1}^{N} (\mathbf{v}_i - \mathbf{v}_{i-1})^2$$

$$E_{curv} \simeq \sum_{i=1}^{N} (\mathbf{v}_{i+1} + \mathbf{v}_{i-1} - 2\mathbf{v}_i)^2$$

$$E_{img} \simeq -\sum_{i=1}^{N} \|\nabla I(\mathbf{v}_i)\|$$

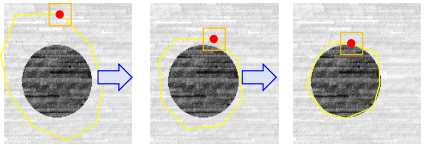
※左式は連結部分 $\mathbf{v}_1$と$\mathbf{v}_N$を考慮してないので注意（見やすくするためこうしました）

※ ∇Iは勾配強度画像

参考資料

Snakes法の最適化

- 色々な最適化法が考えられるがここでは貪欲法を紹介



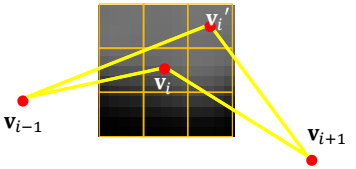
頂点$\mathbf{v}_i$が寄与するエネルギー

$$E = \frac{\alpha}{2}(\mathbf{v}_i - \mathbf{v}_{i-1})^2 + \frac{\alpha}{2}(\mathbf{v}_{i+1} - \mathbf{v}_i)^2 \quad (E_{len})$$
$$+\beta\,(\mathbf{v}_{i+1} + \mathbf{v}_{i-1} - 2\mathbf{v}_i)^2 \quad (E_{curv})$$
$$-\gamma\, I'(\mathbf{v}_i) \quad (E_{img})$$

Snakes (貪欲法)

入力：画像と初期曲線（折れ線）

1. 頂点をひとつずつ訪問する
2. 頂点$\mathbf{v}_i$に訪問中，局所コストが最小となる近傍画素へ$\mathbf{v}_i$を移動する
3. 頂点の移動量が閾値以下になるまで，又は，決められた回数，(1) を繰り返す



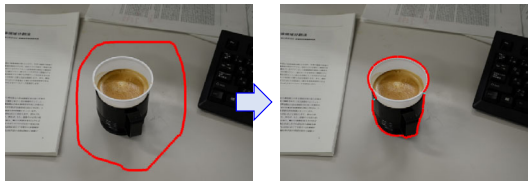
1-1 $\mathbf{v}_i$を近傍画素中心に移動してみる
1-2 局所エネルギー最小の画素に移動

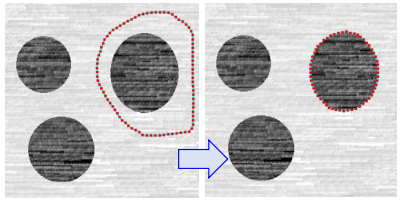
※この手法だと接線方向に頂点が動いてしまう問題はあります

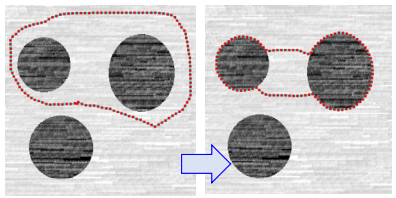
Snakes法の特徴

利点
- ノイズに強い領域分割が可能
- 高速かつ実装が簡単

欠点
- パラメータに強く依存
- 初期輪郭線に強く依存
- トポロジー変化が困難



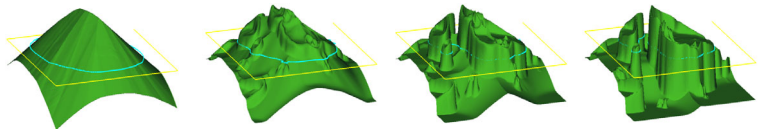




Level set法

TActiveContour.exe

- 輪郭線を符号付きスカラー場（内挿関数）のゼロセットで表現
- スカラー場の初期値は初期境界からの距離場を利用
- スカラー場を変化させることで輪郭線を間接的に変形する
- → トポロジー変化に対応できる

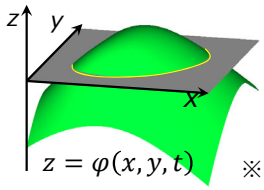


Level set法

参考資料

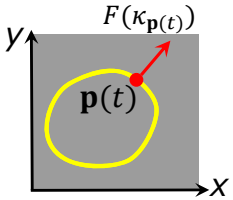
前提1. 境界曲線を内挿関数$\varphi(x,y,t)$のゼロセットで表現

曲線： $\varphi\left(p_x(t),p_y(t),t\right)=0$



※tは時刻

前提2. 曲線上の点$\mathbf{p}(t)$ は
法線方向に速度$F(\kappa_{\mathbf{p}(t)})$で動く



※$\kappa_{\mathbf{p}(t)}$は曲率

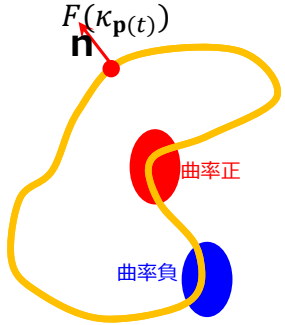
課題: 収束するまで曲線を変形する
※ 実際に変化させるのは$\varphi(x,y,t)$
※ 移動速度$F\left(\kappa_{\mathbf{p}(t)}\right)$をうまく指定すると、輪郭の進展が境界で止まる

速度Fを曲率$\kappa_{p(t)}$の関数にする理由

参考資料

曲線の凸部・凹部で挙動を変化できる
例) 凹部は外へ、凸部は内へ向かって移動

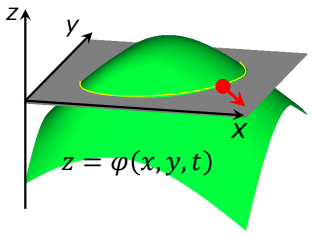
曲がり具合に応じて挙動を変化できる
例) 曲がり具合が大きいところは速く移動



※曲率の正負は
曲線の向き・定義に依存

参考資料

Level Set法



問題の前提

点**p**が曲線に乗る　$\varphi(p_x(t), p_y(t), t) = 0$

点**p**は法線**n**方向に速度Fで移動　$\frac{\partial \mathbf{p}(t)}{\partial t} = F(\kappa_{\mathbf{p}(t)})\mathbf{n}$

法線**n**はφの勾配　$\mathbf{n} = \frac{\nabla\varphi}{||\nabla\varphi||}$

内挿関数φの時間進展に関する方程式を導く

整理する　$\frac{\partial \varphi(p_x(t), p_y(t), t)}{\partial t} = -F(\kappa_{\mathbf{p}(t)})||\nabla\varphi||$　※この式は曲線上で成立 ※変形の詳細は付録へ

全体に拡張　$\frac{\partial \varphi(x, y, t)}{\partial t} = -F(\kappa)||\nabla\varphi||$　※κは内挿関数φの曲率

位置(x, y)と時刻tについて離散化　$\frac{\varphi(i, j, t+1) - \varphi(i, j, t)}{h} = -F(\kappa_{i,j})||\nabla\varphi(i, j, t)||$　※ i,jは画素位置 ※ hは微小時刻

参考資料

- 1. 初期曲線から初期スカラー場 $\varphi(i, j, 0)$ を構築

  (i,jは画素位置,$\varphi(i, j, 0)$は初期曲線からの距離場)
- 2. 変化が十分小さくなるまで時刻tを進めφを更新

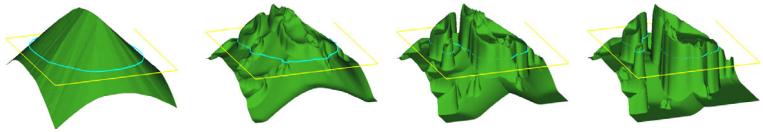
$$\frac{\varphi(i, j, t+1) - \varphi(i, j, t)}{h} = -\frac{a - b\kappa_{i,j}}{1 + ||\nabla(G \otimes I(i, j))||}||\nabla\varphi||$$

a, bはパラメータ

$\kappa_{i,j}$ > a/bなら正方向に進む

エッジ部分では変化が遅い

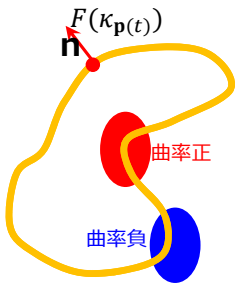
- 3. $\varphi(i, j, t) = 0$ である画素を境界として出力する



参考資料

曲率の定義

曲線上の定義

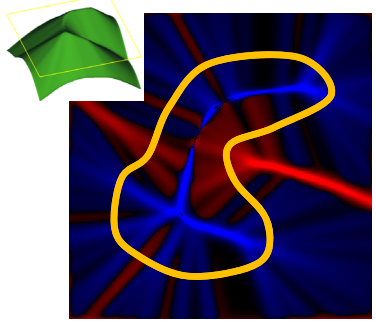


$$\varphi(p_x(t), p_y(t), t) = 0$$

$$\frac{\partial \mathbf{p}(t)}{\partial t} = F(\kappa_{\mathbf{p}(t)})\mathbf{n}$$

$$\mathbf{n} = \frac{\nabla\varphi}{||\nabla\varphi||}$$

内挿関数(空間全体)へ拡張

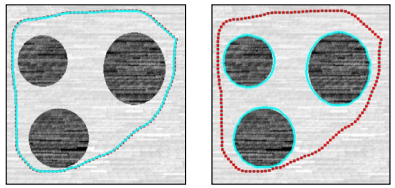


$$\frac{\partial \varphi(x, y, t)}{\partial t} = -F(\kappa)||\nabla\varphi||$$

$\kappa = \nabla\left(\frac{\nabla\varphi}{||\nabla\varphi||}\right)$ ：内挿関数の曲率

Level set法の特徴

輪郭線をスカラー場のゼロセットで表現し，スカラー場を変更することで輪郭線を動かす

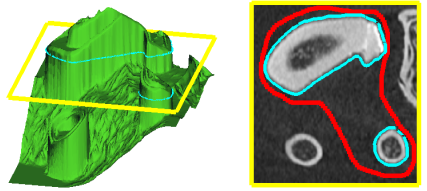


利点

- ノイズに対し堅固・高速
- **トポロジー変化に対応**

欠点

- 速度$F(\kappa_{\mathbf{p}(t)})$の適切な選択が難しい
- パラメータ調節が大変

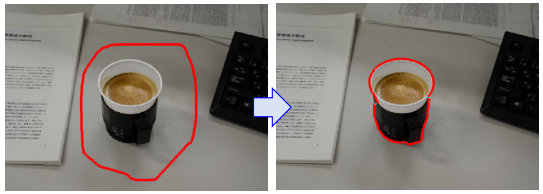


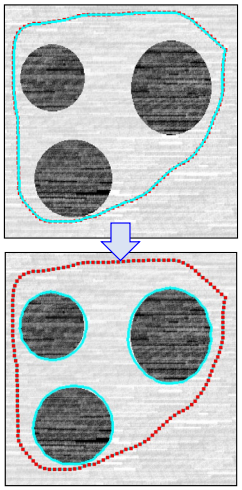
まとめ: 動的輪郭モデル

境界曲線を『形状が滑らかになるように』『画像のエッジを通る様に』変形する手法

- 境界を陽的に表現する：**Snakes** 法
- 境界を陰的に表現する：**Level Set** 法

変形モデル(速度の定義)・最適解の計算法・高次元化など、関連研究は多い





概要のみ紹介

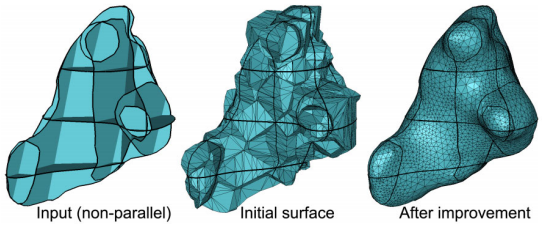
Contents：画像領域分割

曲面再構成法

曲面再構成法による領域分割

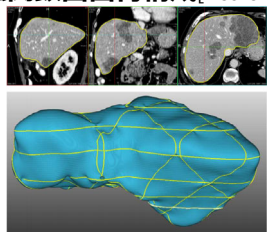
- 形状モデリングのための曲面再構成法を転用
- **輪郭線制約から三次元境界曲面を生成**する

陽的曲面再構成法[Lie et. al. 2008]



図は論文[Lie et. al. 2008]より

陰関数曲面再構成[Heckel et. al. 2011]

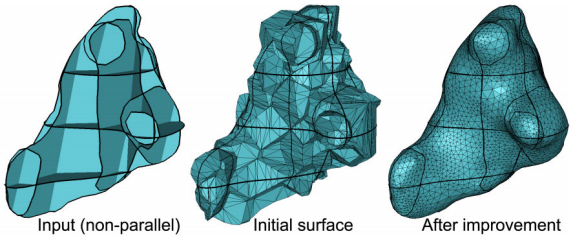


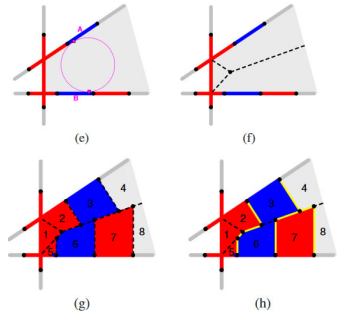
図は論文[Heckel et. al. 2011]より

Heckel F., et. al. : Interactive 3D medical image segmentation with energy-minimizing implicit function. VCBM 35, 2(2011),275–287.
Liu L. et. al. : Surface reconstruction from non-parallel curve networks. *CGF* 27, 2(2008), 155–163.

陽的曲面再構成による領域分割

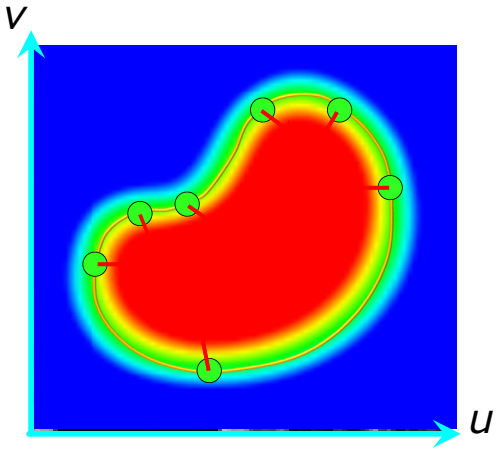
図は[Liu L. et. al. : Surface reconstruction from non-parallel curve networks. *CGF* 27, 2(2008), 155–163.]より





入力：複数の輪郭線（ポリライン）
+ 輪郭線頂点をうまくつなぐことで初期メッシュモデルを生成
+ 初期メッシュモデルを平滑化することで滑らかな境界面を取得する

曲面再構成法：陰関数曲面再構成 [Turk and O'Brien 02]

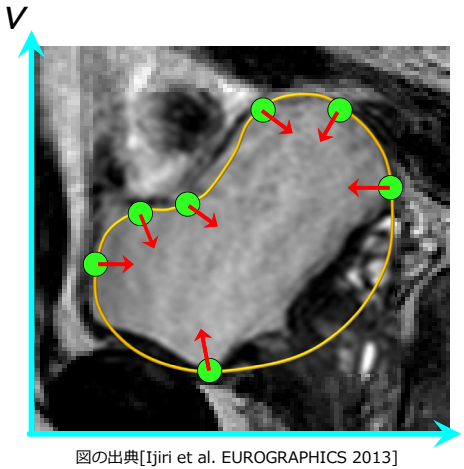


図は[Ijiri et al. EUROGRAPHICS 2013]より

1. 境界の通る点・法線を指定
2. uv空間にスカラー場fを構築
   $f(\mathbf{x}) = 0$、 $\nabla f(\mathbf{x}) = \mathbf{n}$
3. スカラー場のゼロセットを抽出

参考資料

陰関数曲面再構成による領域分割



図の出典[Ijiri et al. EUROGRAPHICS 2013]

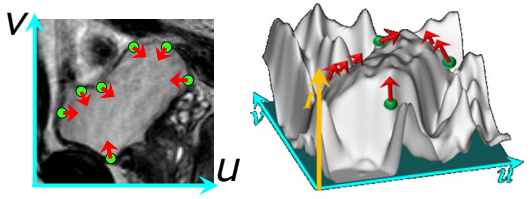
<u>問題：画像のエッジを追従しない</u>

→動的輪郭モデルで境界を動かす
[Aliroteh M et. al. 2007]

→Bilateral空間への拡張
[Ijiri et al 2013]

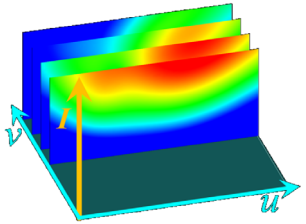
参考資料

陰関数曲面再構成による領域分割 [Ijiri et al 2013]

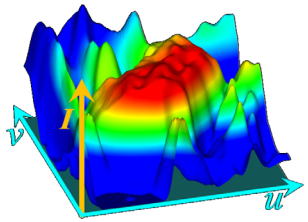


1) Bilateral空間の画像多様体を構築

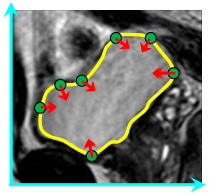
画素値を高さとみなす
と画像は曲面をなす



2) Bilateral空間に
スカラー場構築



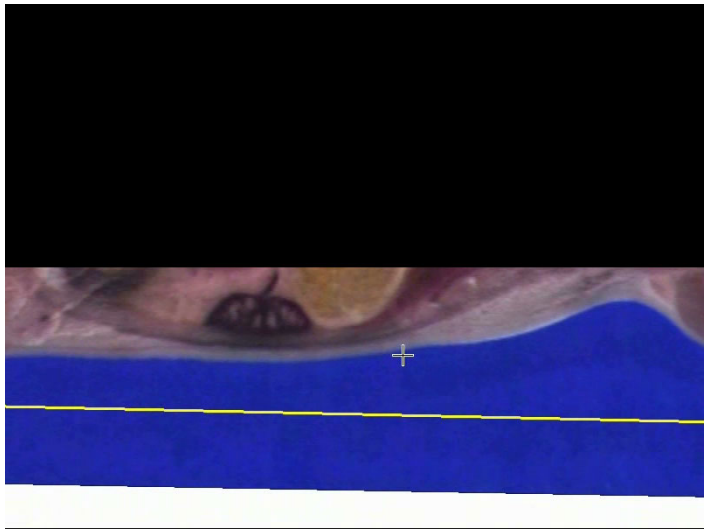
3) 画像多様体上でスカラー値
をサンプリング



4)ゼロ等値面を
境界として出力

参考資料

陰関数曲面再構成による領域分割 [Ijiri et al 2013]

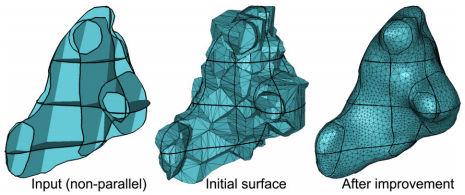


まとめ：曲面再構成法を応用した領域分割

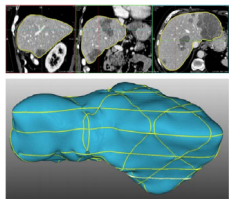
輪郭線制約から境界面を生成する

陽的曲面再構成：輪郭線頂点を直接つなぎ境界面を構築

陰関数曲面再構成：輪郭線から滑らかなスカラー場を構築しそのゼロ等値面を出力



直接的再構成 [Lie et. al. *CGF 2008*]



陰関数曲面再構成 [Heckel et. al. VCBM 2011]

TURK G., O'BRIEN J. F.: Modelling with implicit surfaces that interpolate. *ACM TOG* 21, 4(2002), 855–873.
Heckel F., et. al. : Interactive 3D medical image segmentation with energy-minimizing implicit function. VCBM 35, 2(2011),275–287.
Liu L. et. al. : Surface reconstruction from non-parallel curve networks. *CGF* 27, 2(2008), 155–163.
Takashi Ijiri, et. al. Bilateral Hermite Radial Basis Functions for Contour-based Volume Segmentation. CGF, 2013.

まとめ：画像領域分割

- ここでは多様な領域分割法を広く浅く紹介した
  - 閾値法、領域成長法、クラスタリング、グラフカット法、動的輪郭モデル、曲面再構成法
- 任意の画像（写真, CT, MRI, 顕微鏡）、任意の関心領域(人物、臓器、腫瘍、細胞内小器官)に対し良い結果を出せる『オールマイティ』な領域分割法は未だ実現されていない
- ユーザは、対象に応じて手法を注意深く選択することが大切(選択肢が多くあることを知っておくだけでも)

モーフォロジー演算

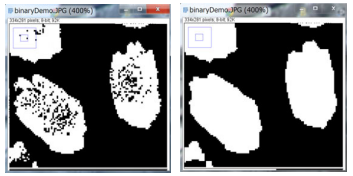
モーフォロジー演算

集合論の概念を利用した画像変換法

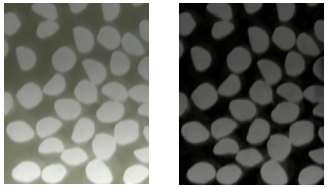
空隙/ノイズ除去・背景グラデーション除去などに利用可能

Opening: 細かなごみを除去

Dilation-ErosionでEdge抽出



Closing: 領域内の穴を除去

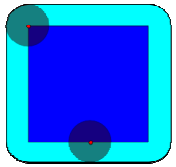


Top-hat: グラデーションを除去

モーフォロジー演算

集合**A** (入力2値画像)

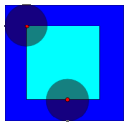
集合**B** (Structure Element)



Dilation (膨張)

$\mathbf{A} \oplus \mathbf{B} = \{\mathbf{c} | \mathbf{c} = \mathbf{a} + \mathbf{b}, \mathbf{b} \in \mathbf{B}, \mathbf{a} \in \mathbf{A}\}$

Bの原点を**A**内で動かしたとき
Bが描く図形



Erosion (収縮)

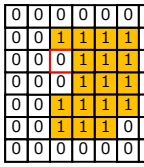
$\mathbf{A} \ominus \mathbf{B} = \{\mathbf{c} | \mathbf{c} + \mathbf{b} \in \mathbf{A}, \forall \mathbf{b} \in \mathbf{B}\}$

B全体が**A**に含まれるよう
Bを動かしたときBの原点が描く図形

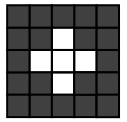
図はwikipediaより [右下の図 : CC-BY-SA, BY Renato Keshet]

2値画像のMorphological operation (1/3)

Basic operations

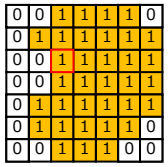


入力画像 $I(\mathbf{x})$



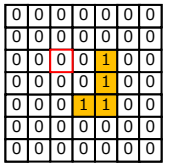
Str. Elem: B

Dilation



$(I \oplus B)(x)$
$= \max_{t \in B}(I(x-t))$

Erosion

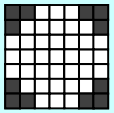


$(I \ominus B)(x)$
$= \min_{t \in B}(I(x-t))$

Structure Element
+ 2値の線形フィルタのようなもの
+ 円形のものが良く用いられる

4近傍

8近傍



半径3pixelの円

※細かいことだが、max(I(x-t))の『マイナス -』は、(ほとんどないけど)左右/上下非対称なStructure elementを利用するとき大切.
計算時は注目画素の周囲の領域の max / minを見るため Structure Elementをひっくり返す必要が有る。

2値画像のMorphological operation (2/3)

入力画像 $I(\mathbf{x})$

Dilate(I, 3)

Dilate(I, 6)

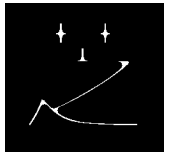
Dilate(I, 9)

Structure Element
Radius : r-pixel

r=1,

r = 3

Erode(I, 3)



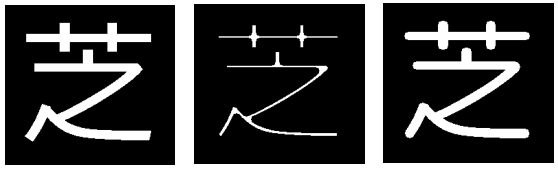
Erode(I, 6)

Erode(I, 9)

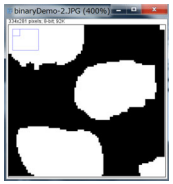
※ Dilate(画像, 半径), Erode(画像, 半径),
※ Dilateでは、Structure elementが円なので角が取れて膨張する
※ Erodeでは、Structure element半径より細い構造はすべて消える

2値画像のMorphological operation (3/3)

Opening （穴あけ） - 収縮させて →膨張させる
Open(I,r) = Dilate(Erode (I,r), r)



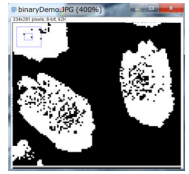
r = 5



前景の小さな構造（線・点）を除去

Closing （穴うめ） - 膨張させて →収縮する
Close(I,r) = Erode(Dilate (I,r), r)

r = 5

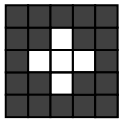


背景の小さな構造（穴）を除去する効果

グレースケール画像のMorphological operation

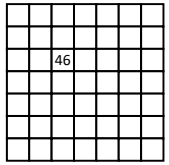
52	2	3	90	34	53	42
42	20	46	24	12	3	32
23	15	10	1	23	4	5
24	3	4	2	42	2	66
45	14	58	4	2	23	13
2	12	6	5	2	2	12
11	23	56	75	67	7	12

入力画像 $I(\mathbf{x})$



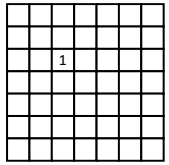
Str. Elem: B

Dilation



$(I \oplus B)(x)$
$= \max_{t \in B}(I(x-t))$

Erosion

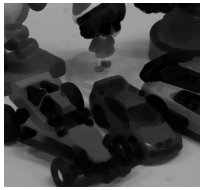


$(I \ominus B)(x)$
$= \min_{t \in B}(I(x-t))$

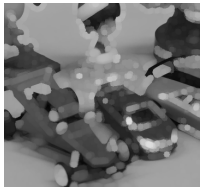
注目画素にStructure Elementを重ね、
周囲の最大値/最小値を新たな画素値とする

グレースケール画像のMorphological operation

元画像



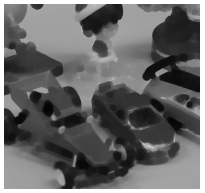
Erode: 明るい領域が収縮



Dilate: 明るい領域が膨張

Dilate – Erode: edgeのようなものが抽出出来る

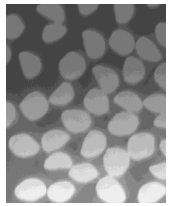
Openinig: 細かい明領域が閉じる(無くなる)



Closing: 細かい暗領域が閉じる(無くなる)

Structure elementは、すべてr =10の円

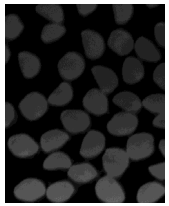
Top-hat transform による背景除去



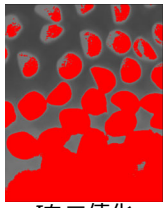
元画像I

Erode(I)

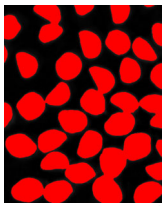
Open(I)
=Dilate(Erode(I))



I - Open(I)



Iを二値化



TopHat(I)を二値化

TopHat(I) = I - *Open*(I)
*Open*で消えた部分を強調
背景のShadeを消す効果がある
(暗い背景に有向)

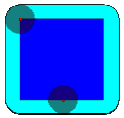
BottomHat(I) = *Close*(I) - I
*Open*で消えた部分を強調
背景のShadeを消す効果がある
(明るい背景に有向)

まとめ：**モーフォロジー演算**

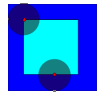
集合理論に基づく画像処理法

入力画像

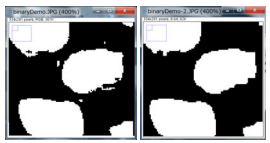
Structure Element



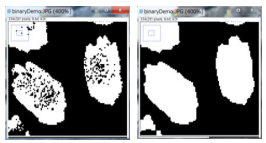
Dilate



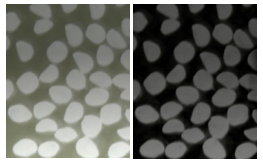
Erode



Opening
Dilate(Erode(・))



Closing
Erode(Dilate(・))



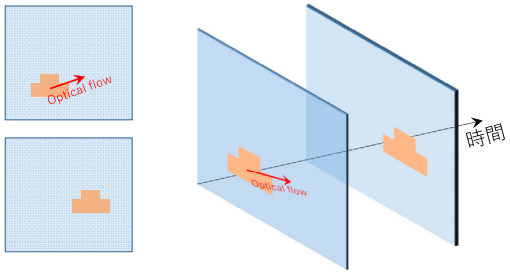
TopHat
I - Dilate(Erode(I))

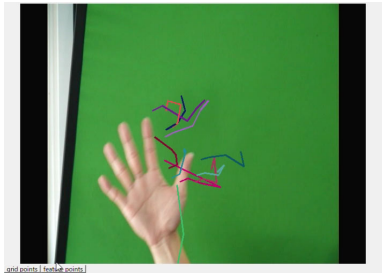
オプティカルフロー

OpticalFlowvis.py

オプティカルフロー

- 2枚の画像内におけるある物体の移動量をベクトルとして表現したもの
- 動画内の『物体追跡』や『物体の動きの解析』などに利用される
- ブロックマッチング法，Lucas-Kanade法など





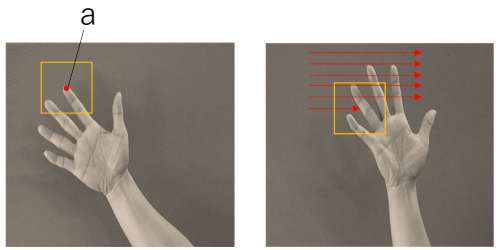
ブロックマッチング法

テンプレートマッチングにより
オプティカルフローを求める手法

入力：２枚の画像A・B、画像A内の点a
出力：点aの移動先
手法：

- 点aを中心とする窓領域を作成
- 窓領域をテンプレートとし、画像Bに対し、テンプレートマッチングを実施
- SSDが最小となる場所を移動先として決定する

※ 移動距離が大きくないと仮定できる場合は、点aの周囲のみでテンプレートマッチングを実施する



テンプレート

追跡したい点ごとにテンプレートマッチング
を実施するため計算量が大きくなる

復習：連立一次方程式の行列を利用した表現

2変数の連立1次方程式を考える

$$\begin{aligned} 1x + 2y &= 1 \\ 3x + 4y &= 2 \end{aligned}$$

行列を使って以下の通り書き表せる

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

両辺に逆行列をかけることで解が得られる

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}^{-1} \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

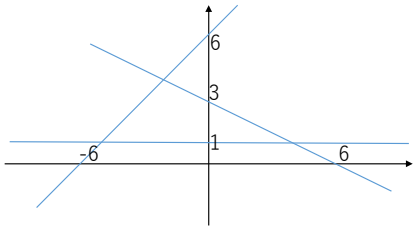
※ 逆行列が存在しない場合、この
連立方程式は不定or不能

復習：一般逆行列（擬似逆行列）

2変数の連立1次方程式を考える

$$\begin{aligned} 1x + 2y &= 6 \\ -1x + 1y &= 6 \\ 0x + 1y &= 1 \end{aligned}$$

変数が２個、方程式が３本なので解なし



3本の直線の交点はない

復習：一般逆行列（擬似逆行列）

$$\begin{aligned} 1x + 2y &= 6 \\ -1x + 1y &= 6 \\ 0x + 1y &= 1 \end{aligned}$$

$$\begin{pmatrix} 1 & 2 \\ -1 & 1 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 6 \\ 6 \\ 1 \end{pmatrix}$$ 行列を使って書き表した

$$\mathbf{A}\begin{pmatrix} x \\ y \end{pmatrix} = \mathbf{b},$$ ただし$\mathbf{A} = \begin{pmatrix} 1 & 2 \\ -1 & 1 \\ 0 & 1 \end{pmatrix}$, $\mathbf{b} = \begin{pmatrix} 6 \\ 6 \\ 1 \end{pmatrix}$として整理した

$$\mathbf{A}^{\mathrm{T}}\mathbf{A}\begin{pmatrix} x \\ y \end{pmatrix} = \mathbf{A}^{\mathrm{T}}\mathbf{b}$$ 両辺にA^{T}をかけた

$$\begin{pmatrix} x \\ y \end{pmatrix} = \left(\mathbf{A}^{\mathrm{T}}\mathbf{A}\right)^{-1}\mathbf{A}^{\mathrm{T}}\mathbf{b}$$ 両辺に$\left(\mathbf{A}^{\mathrm{T}}\mathbf{A}\right)^{-1}$をかけた

この$\left(\mathbf{A}^{\mathrm{T}}\mathbf{A}\right)^{-1}\mathbf{A}^{\mathrm{T}}$を一般逆行列とよび、
今回の例ではすべての方程式を“なるべく”満たす解が得られます

かなり雑な説明です。詳しくは線形代数の教科書等を参考にしてください。

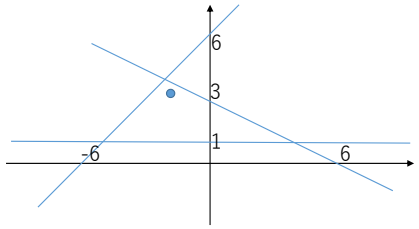
復習：一般逆行列（擬似逆行列）

$$\begin{pmatrix} 1 & 2 \\ -1 & 1 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 6 \\ 6 \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} x \\ y \end{pmatrix} = \left(\mathbf{A}^{\mathrm{T}}\mathbf{A}\right)^{-1}\mathbf{A}^{\mathrm{T}}\begin{pmatrix} 6 \\ 6 \\ 1 \end{pmatrix}$$

$$= \begin{pmatrix} 2 & 1 \\ 1 & 6 \end{pmatrix}^{-1}\begin{pmatrix} 0 \\ 19 \end{pmatrix}$$

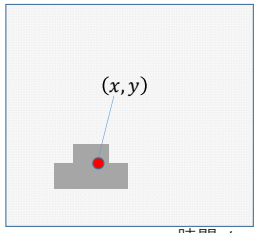
$$= \frac{1}{11}\begin{pmatrix} -19 \\ 38 \end{pmatrix}$$



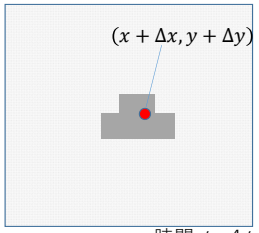
Lucas-Kanade法 0

時刻tにおける画像、画像内の点(x,y)、
および 時刻$t+\Delta t$における画像が与えられる

→ このもとで点(x,y)の移動先(x+Δx, y+Δy)を求めたい



時間 t



時間 $t+\Delta t$

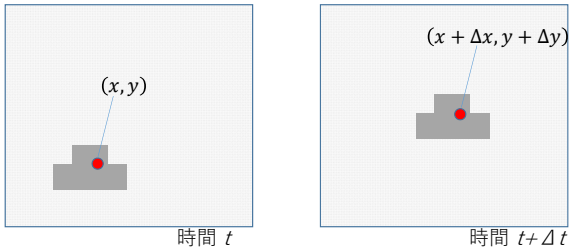
Lucas-Kanade法 1

- 仮定1. 追跡対象の物体は移動後も同じ輝度値を持つ
- 仮定2. 移動量は非常に小さい

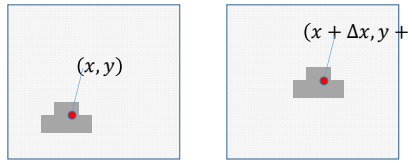
時刻tにおける点(x,y)が、時刻$t+\Delta t$で$(x+\Delta x,y+\Delta y)$に移動したとすると以下が成り立つ

$$I(x,y,t) = I(x+\Delta x, y+\Delta y, t+\Delta t)$$

$I(x,y,t)$は時刻tにおける画素（x,y）の輝度値を表す



Lucas-Kanade法 2



$$I(x,y,t) = I(x+\Delta x, y+\Delta y, t+\Delta t)$$

$$I(x,y,t) \approx I(x,y,t) + \frac{\partial I(x,y,t)}{\partial x}\Delta x + \frac{\partial I(x,y,t)}{\partial y}\Delta y + \frac{\partial I(x,y,t)}{\partial t}\Delta t$$

右辺をテーラー展開
2次以上の後を無視

$$I_x \Delta x + I_y \Delta y + I_t \Delta t = 0$$

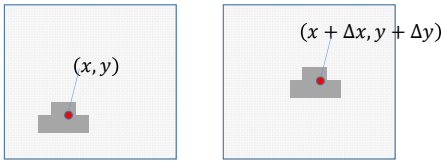
画像のx, y方向微分をI_x, I_yと，
時間方向微分をI_tと表記

$$I_x u + I_y v + I_t = 0$$

両辺を Δtで除算し、$\frac{\Delta x}{\Delta t} = u, \frac{\Delta y}{\Delta t} = v$とおいた

Lucas-Kanade法 3

$$I_x u + I_y v + I_t = 0$$



- この式はオプティカルフローの拘束式と呼ばれる

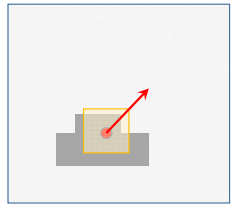
$I_x = \frac{\partial I(x,y,t)}{\partial x}$ は、画像Iの (x,y)における横方向微分値 (Sobelフィルタでもとまる)

$I_y = \frac{\partial I(x,y,t)}{\partial y}$ は、画像Iの (x,y)における縦方向微分値 (Sobelフィルタでもとまる)

$I_t = \frac{\partial I(x,y,t)}{\partial t}$画像$I$の (x,y)における時間方向微分値，$\big(I(x,y,t+\Delta t) - I(x,y,t)\big)/\Delta t$ で計算できる

- 求めたいのは、追跡したい画素 (x,y)の速度ベクトル（ u, v ）
- 未知変数2個、方程式1本なのでこの問題は解けない（不定）

Lucas-Kanade法 4



仮定3 : 近傍画素は似た速度ベクトルを持つ

- 物体は複数画素から構成されるため

注目点(x,y)を中心とする窓領域内の画素(x_1,y_1), (x_2,y_2),…(x_n,y_n)
について，すべてが同じ速度ベクトル(u,v)を持つとすると…

$$\begin{aligned} I_x(x_1,y_1)u + I_y(x_1,y_1)v &= -I_t(x_1,y_1) \\ I_x(x_2,y_2)u + I_y(x_2,y_2)v &= -I_t(x_2,y_2) \\ &\vdots \\ I_x(x_n,y_n)u + I_y(x_n,y_n)v &= -I_t(x_n,y_n) \end{aligned}$$

が成り立つ

Lucas-Kanade法 5

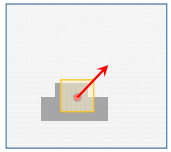
$$
\begin{aligned}
I_x(x_1,y_1)u + I_y(x_1,y_1)v &= -I_t(x_1,y_1)\\
I_x(x_2,y_2)u + I_y(x_2,y_2)v &= -I_t(x_2,y_2)\\
&\vdots\\
I_x(x_n,y_n)u + I_y(x_n,y_n)v &= -I_t(x_n,y_n)
\end{aligned}
$$

$$
\Rightarrow \begin{pmatrix} I_x(x_1,y_1) & I_y(x_1,y_1) \\ I_x(x_2,y_2) & I_y(x_2,y_2) \\ \vdots & \vdots \\ I_x(x_n,y_n) & I_y(x_n,y_n) \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = - \begin{pmatrix} I_t(x_1,y_1) \\ I_t(x_2,y_2) \\ \vdots \\ I_t(x_n,y_n) \end{pmatrix} \cdots (1)
$$

この連立方程式は、変数2個、制約式n本なので解なし（不能）
→ 一般逆行列により、なるべく全ての方程式を満たす解が得られる

Lucas-Kanade法の利点は高速であること

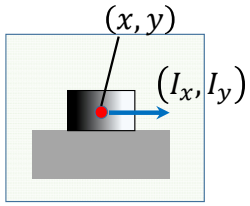
必要な計算は
- 着目点周囲のx/y/時間方向微分
- 一般逆行列の計算 のみ

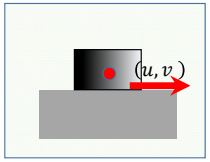


オプティカルフローの拘束式の定性的な理解

$$I_x u + I_y v = - I_t$$

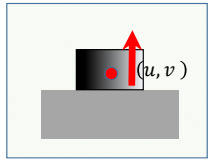
(I_x, I_y)：点(x,y)における輝度勾配
(u, v)：点(x,y)にある物体の速度
I_t：点(x,y)の輝度の時間変化





物体が右に動いていたら点(x,y)
はこれから暗くなりそう

$I_t = -I_x u - I_y v$ だけ暗くなる



物体が上に動いていたら点(x,y)
の輝度値は変化しなさそう

$I_t = -I_x u - I_y v = 0$ なので変化なし

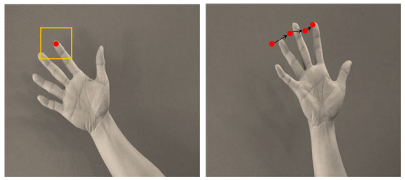
Lucas-Kanade法 6

ここまでの説明したものは実際の動画に対してはあまりうまく動かない
これは『仮定2: 移動量は非常に小さい』が実世界データでは成り立たないため
※ 移動量は1画素程度であって欲しいが、追跡対象が１フレーム間に数画素分動いてしまうことも多い

逐次更新

先の一般逆行列による解法を複数回繰り返すことで、移動ベクトルを更新

更新時の移動ベクトルにより，前頁(1)の右辺を計算する



画像ピラミッドを利用した拡張

対象画像に対しガウシアンピラミッドを作成

最も高い層からはじめ、ひとつ上の層の計算結果を利用してある層の移動ベクトルを計算

まとめ：オプティカルフロー

- 2枚の画像内におけるある物体の移動量をベクトルとして表現したもの
- 動画内の『物体追跡』や『物体の動きの解析』などに利用される
- **ブロックマッチング法：**
  → 着目点を中心とする窓領域をテンプレートとしてテンプレートマッチング
- **Lucas-Kanade法：**
  → オプティカルフローの拘束式を立て、隣接画素の動きが似ている（空間的な連続性）と仮定し、複数の拘束式よりオプティカルフローを計算