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. 筆記試験
10. プログラミング演習 1：PC室
11. プログラミング演習 2：PC室
12. プログラミング演習 3：PC室
13. プログラミング演習 4：PC室
14. プログラミング演習 5：PC室

Contents：画像領域分割

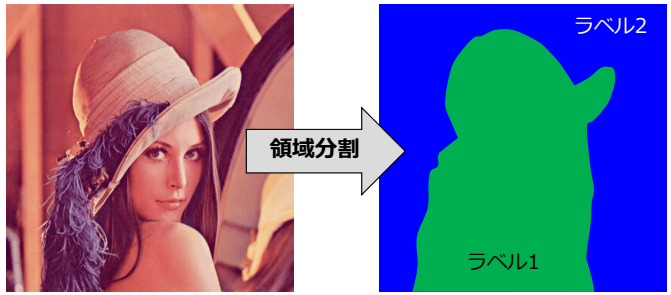
- 画像領域分割とは
- 閾値法
- 領域成長法
- クラスタリング
- 識別器 (後半)
- グラフカット法
- 動的輪郭モデル
- 曲面再構成法
- 深層学習 (後半)

今回と次回は多様な領域分割法を
広く浅く紹介します

教科書10章に対応しますが
井尻の専門分野なので参考書からは
だいぶ外れた内容も紹介します

画像領域分割（Image Segmentation）とは

- 『画像領域分割』『画像領域抽出』『画像ラベリング』とも呼ばれる
- Vision/Graphics/Image Processing 分野において重要な課題
- デジタル画像の各画素にラベルをつける作業
  （ラベル画像を作る作業）

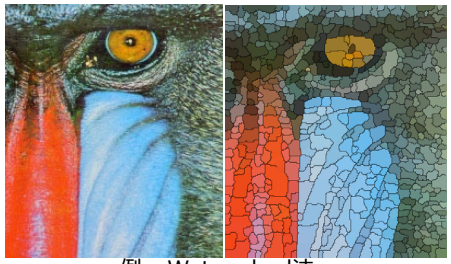


Low-level と High-level segmentation

Low-level segmentation
画像を特徴（色等）が一様な
局所領域に分割する作業

High-level segmentation
画像内の目標物の領域を切り
抜く作業

※両者の境界は曖昧で両者の意味を込めて
『画像領域分割』と呼ぶのが一般的

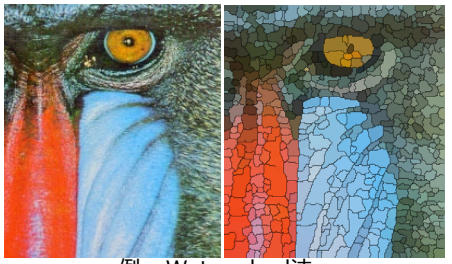


例：Water shed法

例：Graph Cut法

Low-level と High-level segmentation

Low-level segmentation
意味のある固まりを抽出
画像を圧縮
処理の高速化
(画素は直接処理するのに小さすぎる)



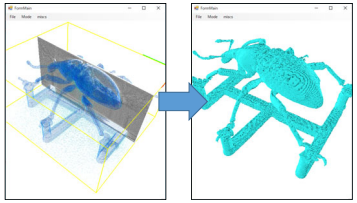
例：Water shed法

High-level segmentation
画像編集（エフェクト適用）
コラージュ
シミュレーション用モデルの構築(3D)

例：Graph Cut法

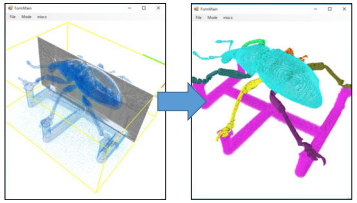
二値化と多値化

二値化
画像を前景・背景に分割

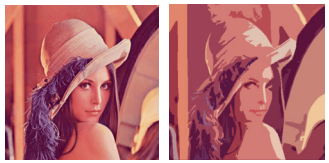


例)濃淡画像の白黒二値化

多値化
画像を複数領域に分割



例) ポスタリゼーション
（階調数を削減し特殊効果を得る）

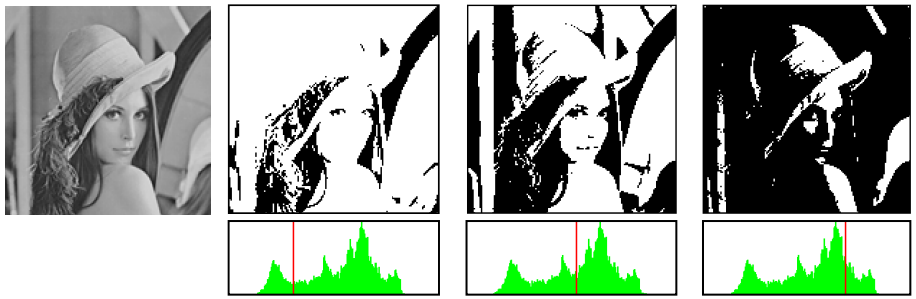


Contents : 画像領域分割

閾値法
thresholding

TThresholding.exe

閾値法とは



閾値により画素に前景・背景ラベルを付ける
閾値を自動的に計算する方法が研究される
→ Pタイル法[1], 大津法[2], Sauvola法[3]…etc…

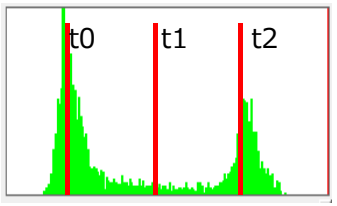
[1] CG-Arts協会. ディジタル画像処理
[2] Ostu N.: A threshold selection method from gray-level histo-grams. IEEE SMC, 9, 1979, 62-66.
[3] J. Sauvola et. al., "Adaptive document image binarization," Pattern Recognition 33(2), 225–236, 2000.

TThresholding.exe

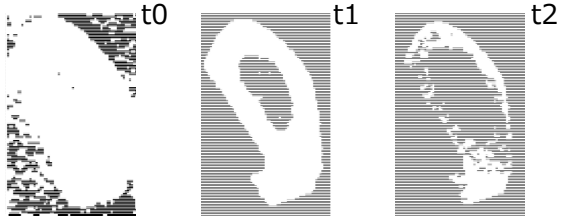
閾値法：大津法

入力画像

ヒストグラム



閾値をどこに置けばいい？
二峰を真ん中で分割するのが理想



大津法

二峰性ヒストグラムを仮定し，
二峰を最も良く2分割する閾値を
自動計算する手法

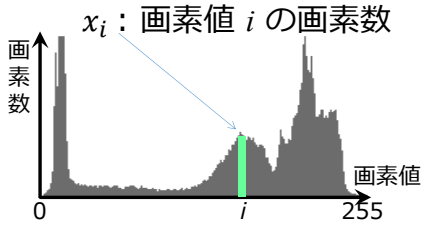
引用数49kを超える論文

TThresholding.exe

閾値法：大津法

ヒストグラムの**分離度**を定義しこれを最大化する閾値を探す

入力画像



画素数

$$\omega = \sum_{i=0}^{255} x_i$$

平均

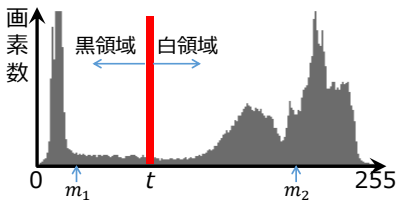
$$m = \frac{1}{\omega}\sum_{i=0}^{255} x_i \times i$$

分散

$$\sigma^2 = \frac{1}{\omega}\sum_{i=0}^{255} x_i \times (i-m)^2$$

閾値法：大津法

ある閾値tで2領域に分離したとき…



	全体	黒領域	白領域
画素数	ω	ω_1	ω_2
平均	m	m_1	m_2
分散	σ^2	σ_1^2	σ_2^2

黒領域

$$m_1 = \frac{1}{\omega_1}\sum_{i=0}^{t-1} x_i \times i$$

$$\sigma_1^2 = \frac{1}{\omega_1}\sum_{i=0}^{t-1} x_i \times (i-m_1)^2$$

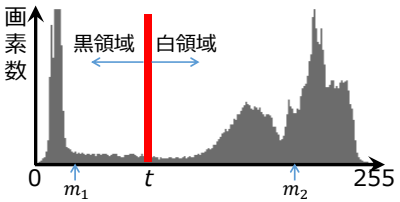
白領域

$$m_2 = \frac{1}{\omega_2}\sum_{i=t}^{255} x_i \times i$$

$$\sigma_2^2 = \frac{1}{\omega_2}\sum_{i=t}^{255} x_i \times (i-m_2)^2$$

閾値法：大津法

ある閾値tで2領域に分離したとき…



	全体	黒領域	白領域
画素数	ω	ω_1	ω_2
平均	m	m_1	m_2
分散	σ^2	σ_1^2	σ_2^2

クラス内分散

$$\sigma_w^2 = \frac{\omega_1\sigma_1^2 + \omega_2\sigma_2^2}{\omega_1 + \omega_2}$$

2領域の分散の平均値
小さい方が良い分割

クラス間分散

$$\sigma_b^2 = \frac{\omega_1(m_1 - m)^2 + \omega_2(m_2 - m)^2}{\omega_1 + \omega_2}$$

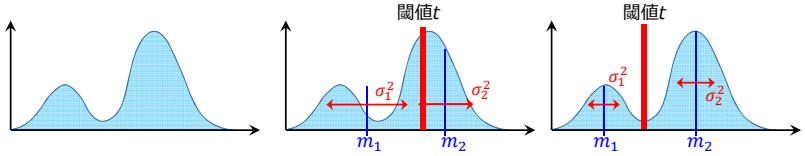
2領域の平均値の距離（注）
大きい方が良い分割

閾値法：大津法

分離度 $= \dfrac{\text{クラス間分散}}{\text{クラス内分散}} = \dfrac{\sigma_b^2}{\sigma_w^2}$

$$\sigma_b^2 = \frac{\omega_1(m_1 - m)^2 + \omega_2(m_2 - m)^2}{\omega_1 + \omega_2}$$

$$\sigma_w^2 = \frac{\omega_1\sigma_1^2 + \omega_2\sigma_2^2}{\omega_1 + \omega_2}$$

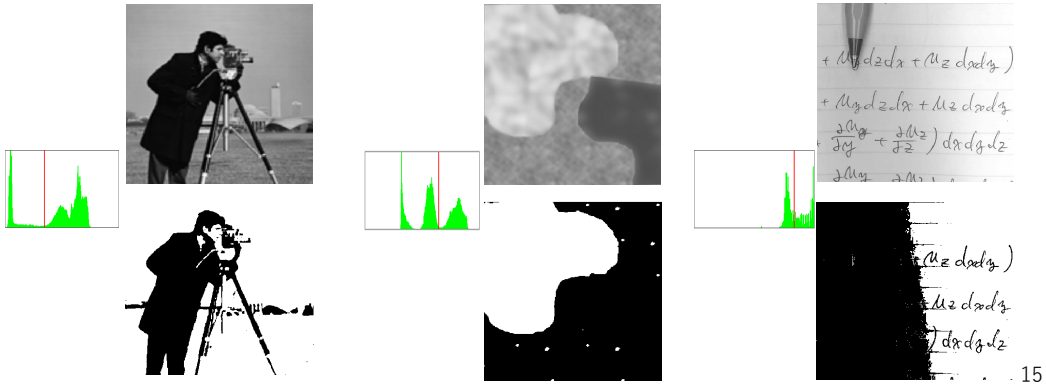


大津法

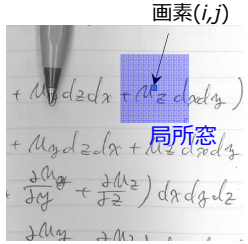
1) 入力画像のヒストグラムを構築
2) 閾値tを 1 から254まで動かし分離度を計算
3) 分離度が最大になる閾値t_{max}で画像を分割

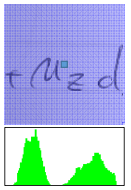
閾値法：大津法

双峰性の高いヒストグラムを持つ画像には強い（そうでない画像には使えない）
グラデーションに弱い

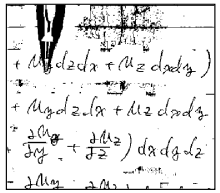


閾値法：Adaptive thresholding

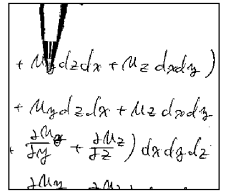




各画素x_{ij}周囲の局所窓を考える
窓内のヒストグラムからその画素用の閾値t_{ij}を計算



大津法：局所窓のヒストグラムから大津法を計算



$$t_{i,j} = m_{i,j}\left(1 + k\left(\frac{\sigma_{i,j}}{R} - 1\right)\right)$$

$t_{i,j}$ ：画素i,jの閾値
$m_{i,j}$：窓内の平均
σ^{i,j^2}：窓内の分散
R ：最大標準偏差 (= 128)
k ：パラメータ (= 0.2)

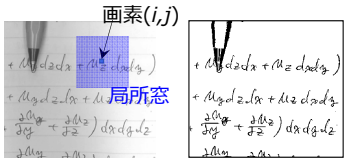
Sauvola法：局所窓の平均値と分散$\sigma 2$から閾値tを計算

まとめ：閾値法

大津法

分離度 $= \frac{\text{クラス間分散}}{\text{クラス内分散}} = \frac{\sigma_b^2}{\sigma_w^2}$

Adaptive thresholding



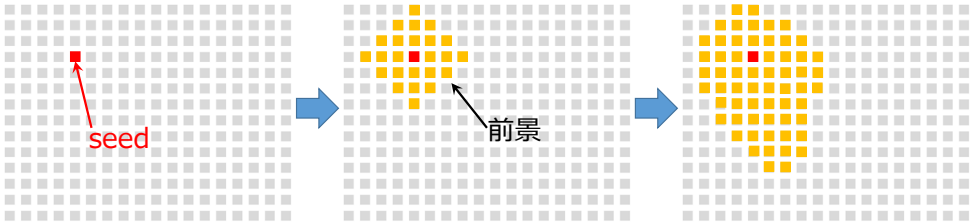
局所窓の情報を利用して閾値計算
大津法 や Sauvola法

閾値により画素に前景ラベル・背景ラベルを付ける
閾値を自動計算する手法（Pタイル法, 大津法, Sauvola法）を紹介した

Contents：画像領域分割

領域成長法
Region Growing

領域成長法の概要



- Seed画素から領域を徐々に成長させる
  (Seedは手で与えるか自動生成する)
- 局所的な規則に従って成長を止める
- Seed配置・成長規則について多くの研究・開発がされている

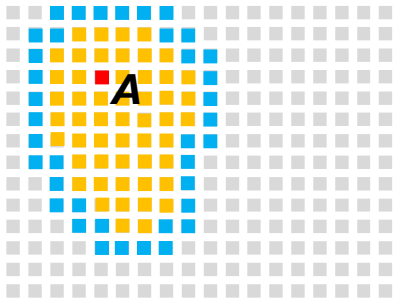
Adams R. *et. al.*: Seeded region growing. *IEEE PAMI* 16, 641-647, 1994.
Roerdink J.B.T.M., *et. al.*: The Watershed Transform: Denitions, Algorithms and Parallelization Strategies, 2000.

領域成長法: 二値化

TRegionGrowing.exe

■ Seed　■ 境界画素 T

■ 現在の領域 *A*



*k*ステップ後の状態

領域成長法(二値化)

入力：複数のseed画素

1. Seed画素を前景領域に追加
2. 前景領域に隣接する画素xのうち
   次式を満たすもの前景領域に追加
   $|c(seed) - c(x)| < r$　※
3. 成長が止まるまで(2)を繰り返す

$c(seed)$ ：seedの画素値
$c(x)$　　：画素xの画素値
r　　：パラメータ

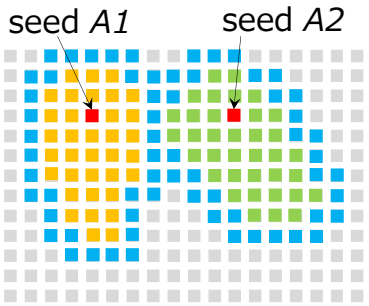
※条件には様々なものが考えられる

TRegionGrowing.exe

領域成長法: 多値化

■ Seed　■領域境界画素 T

■領域 *A1*　■領域 *A2*



図は*k*ステップ後の状態

Seeded Region Growing

入力: 領域ID(*A1,…, An*)の付いたSeed

1. 各Seedを領域*A1,…, An*の要素とする
2. 境界画素 x とその隣接領域 *Ai* のうち、次式が最小となるxを*Ai* に追加
   $$\delta(x) = |c(x) - c(Ai)|$$
3. 全画素を追加するまで(2)を繰り返す

$c(x)$: xの色

$c(Ai)$: xが隣接する領域*Ai*の平均色

※2.においてxが複数領域に隣接する場合は境界ラベル（-1など）をつける

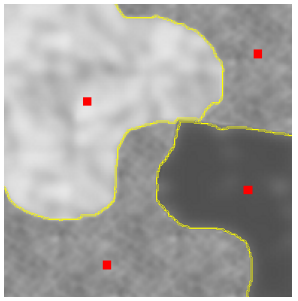
[Adams *et. al.* 1994]

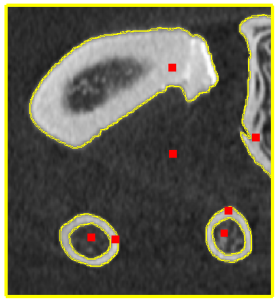
TRegionGrowing.exe

領域成長法の特徴

一様な画素値を持つ領域の分割に適する

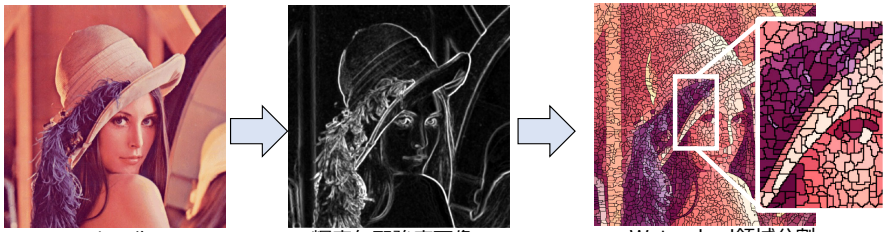
ぼけた境界では成長が止まりにくい





領域成長法: Watershed Algorithm

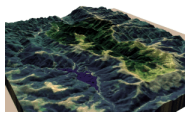
[Roerdink J.B.T.M., *et. al.*: 2000.]



入力画像　輝度勾配強度画像　Watershed領域分割

勾配強度を高さと見なすと、勾配強度画像を地形と見なせる

この地形の分水界を境界とする領域分割法



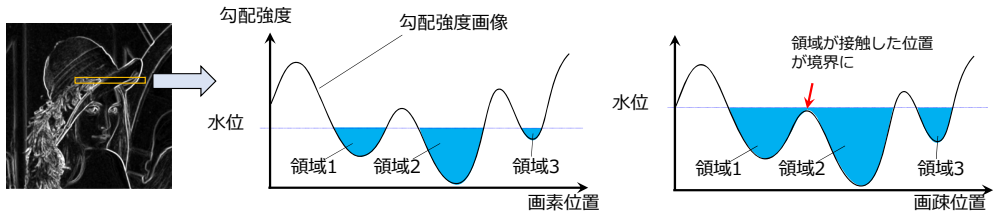
Watershed：地形学における『分水界』を表す用語:

ある地形のある点に落ちた雨がどこに溜まるかを考える隣接しながらも溜まる先が異なる2点間を領域の境界に

左図はwikipediaより（Public Domain）

領域成長法: Watershed Algorithm

[Roerdink J.B.T.M., *et. al.*: 2000.]



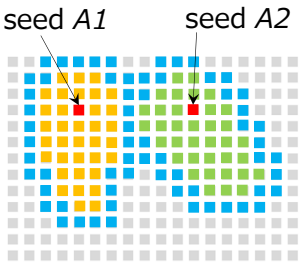
勾配強度を高さと考え水を下から満たしていく

徐々に水位を上げ隣接領域が接した部分を境界とする

領域成長法の言葉で言うと….

- 勾配強度の全ての局所最小点にSeedを配置
- 全領域を同時に成長させ，異なる領域が接した部分を境界にする

領域成長法: まとめ



- 局所的な規則に従って領域を成長させる手法
- Seed配置・成長規則に関する研究がなされている
- 単純な二値化 / Seeded Region Growing(多値化)/Watershed法』を紹介

Adams R. *et. al.*: Seeded region growing. *IEEE PAMI* 16, 641-647, 1994.
Roerdink J.B.T.M., *et. al.*: The Watershed Transform: Denitions, Algorithms and Parallelization Strategies, 2000.

練習問題

(1) 0-7の値を持つ画像に付いてヒストグラムを計算したところ、右図が得られた。画素値0-4と5-7のグループに分ける場合の、クラス内分散・クラス間分散・分離度を求めよ

※Google スプレッドシート や excelを使うとよいかもしれません

値	0	1	2	3	4	5	6	7
個数	3	10	25	8	4	5	20	5

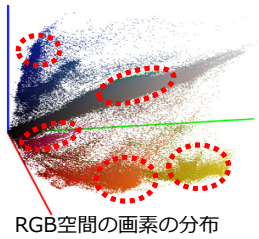
(2) 0-7の値を持つ10x10画像に対して、赤い画素をシードとした領域拡張を行う。結果として得られる全景画素を示せ。ただし、領域成長には上下左右4近傍を利用し、値が6以上の画素が隣接する場合に領域を成長させるものとする

6	1	3	3	5	7	1	6	7	3
5	5	6	1	4	5	3	3	6	2
5	7	7	7	7	7	7	7	7	3
4	3	7	7	3	6	6	3	2	4
4	6	5	5	5	6	6	4	2	5
2	6	6	3	2	6	6	6	7	5
4	7	6	2	3	7	7	5	6	2
5	7	6	3	1	2	3	5	1	1

Contents : 画像領域分割

クラスタリング
clustering

クラスタリングによる領域分割


RGB空間の画素の分布


12領域

画素を**特徴空間**に配置し、特徴空間内で**密集する画素集合（クラスタ）**を発見し分割する

特徴空間　：色空間, Bilateral空間, テクスチャ空間, etc…

有名な手法：K-mean法[1], Mean shift法[2], Normalized Cut法[3], etc…

[1] 高木幹雄ら, 新編画像解析ハンドブック. 東京大学出版会, 2004.
[2] Comaniciu D. *et. al.*: Mean shift: A robust approach toward feature space analysis, *IEEE PAMI*, 24, 5(2002), 603-619.
[3] Shi J. *et. al.*: Normalized cuts and image segmentation. *IEEE PAMI*, 22, 8(2002), 888–905.

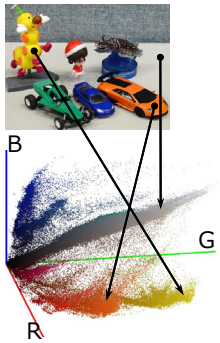
TClustering.exe

特徴空間とは

特徴空間：画像の局所的な特徴が張る空間のこと

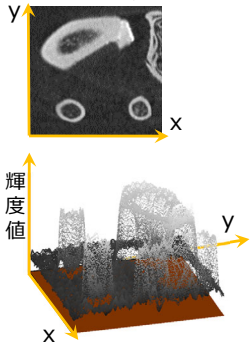
RGB空間

$p_i \rightarrow (R_i \ G_i \ B_i)^T$



Bilateral空間

$p_i \rightarrow (p_{i,x} \ p_{i,y} \ I_i)^T$



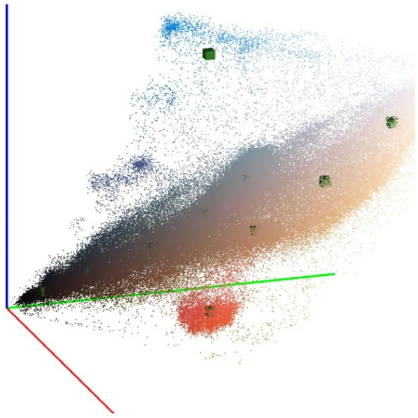
Bilateral空間（Color）

$p_i \rightarrow (p_{i,x} \ p_{i,y} \ R_i \ G_i \ B_i)^T$

※ R_i・G_i・B_i・I_iは画素p_iのR・G・B・輝度値

他にも…
テクスチャ特徴/HOG/SIFT/HLAC/CHLAC/等

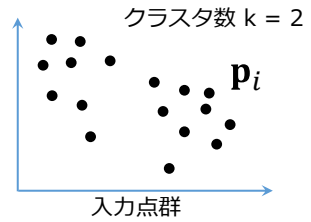
k-means clustering（k-平均法）



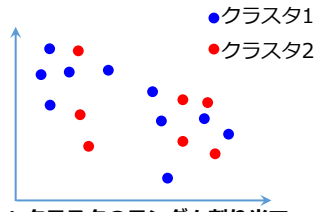
◆利点
- アルゴリズムが単純で実装が楽
- kを変化させることで、発見できるクラスタ数を変更できる

◆欠点
- 初期割り当てに結果が依存
- 多様なクラスタ形状を扱えない
- クラスタ数kが既知の必要あり

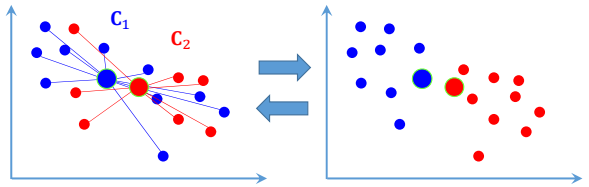
k-means clustering（k-平均法）



入力：特徴空間の点群$\mathbf{p}_i$, クラスタ数k
1. 各点$\mathbf{p}_i$にクラスタIDをランダムに割り当てる
2. クラスタ中心$\mathbf{C}_j$をクラスタの重心に移動
3. 各点$\mathbf{p}_i$を中心$\mathbf{C}_j$が最も近いクラスタに割り当てる
4. 変化がなくなるまで2,3を繰り返す



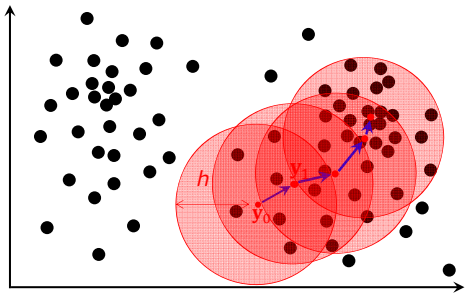
1.クラスタのランダム割り当て



2. クラスタ中心C_j計算　　**3. クラスタ割り当て更新**

Mean-Shift Clustering（平均シフト法）

点群 $\mathbf{p}_i \ \ i = 1,2,\dots,N$



Mean Shift Procedure (MSP)

点 $\mathbf{y}_0$ 付近の点群密度の局所最大点を発見
入力：点群 $\mathbf{p}_i$, 初期点$\mathbf{y}_0$, バンド幅 h

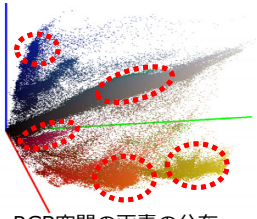
1. $\mathbf{y}_{k+1} \leftarrow \frac{\sum_{i=1}^{N} g_i \mathbf{p}_i}{\sum_{i=1}^{N} g_i}$ $\quad g_i = \begin{cases} 1 & \|\mathbf{p}_i - \mathbf{y}_k\| \le h \\ 0 & \|\mathbf{p}_i - \mathbf{y}_k\| > h \end{cases}$
2. $\|\mathbf{y}_{k+1} - \mathbf{y}_k\| <$ 閾値 まで 1 を繰り返す

Mean Shift Clustering

方法1. 各画素位置$\mathbf{p}_i$からMSPを行い、近い点に収束した画素を同一クラスタにする

方法2. 特徴空間内に格子状に配置した点群$\mathbf{x}_i$にMSP行う．同じ点に収束するカーネルが通った画素を同一クラスタにする

クラスタリングによる領域分割： まとめ



RGB空間の画素の分布

12領域

画素を特徴空間に射影し、その特徴空間内で密度の濃い部分を同一領域として分割する

- **特徴空間の選択**と**クラスタの発見法**が大切
- 教師無し（正解データセット無し）学習の一種
- k-平均法, Mean Shift法 を紹介した

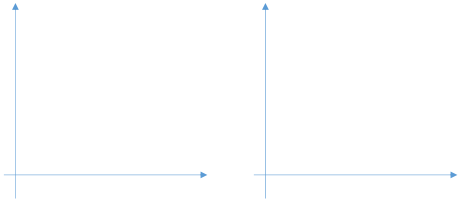
Contents : 画像領域分割

識別器

SKIP
後半にて解説

練習問題

(1)クラスAに属するサンプル点10個と、クラスBに属するサンプル点10個が2次元の特徴空間に分布している。このデータに対して、k=2としてk-meansクラスタリングを行った際、うまくクラスタを発見できない分布の例を２つ挙げ、それぞれについてうまくいかない理由を述べよ。



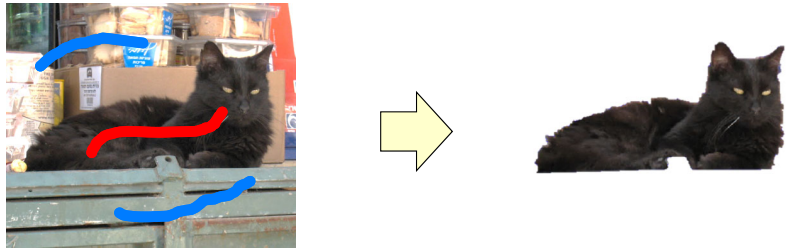
(2)k-meansクラスタリングを適用可能な、画像処理以外の課題を一つ挙げ、k-meansクラスタリングがどのように活用できるか簡潔に説明せよ。

Contents : 画像領域分割

グラフカット法
Graph Cut

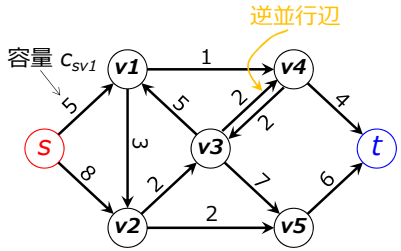
グラフカット領域分割

前傾制約 背景制約



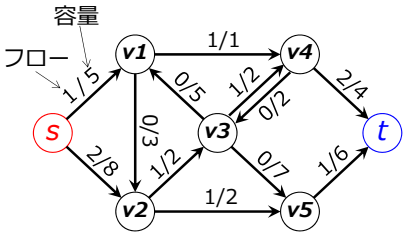
前景・背景に属する画素を適当に入力すると，これを制約に画像を二値化
二値化をエネルギー最小化問題として定式化し、フローネットワークの最小カットにより最適解を計算する
臓器などの塊状領域に対してはかなり強力な領域分割法

準備：フローネットワーク



フローネットワーク

- 容量付き有向グラフ $G = (V, E)$
- 頂点集合 V と辺集合 E から成る
- 辺 (p, q) は容量 $c_{pq} > 0$ を持つ
- 始点 s と終点 t を含む



フロー

- 各辺には容量を超えない範囲でフローが流れる
- ある頂点vについて，流入するフローと流出するフローは等しい
- 総流量：始点から出るフローの総和
- 最大フロー: ネットワークに流せる最大の総流量

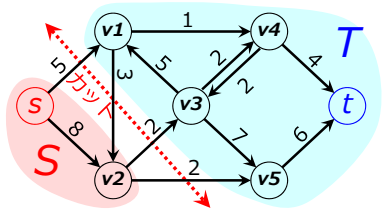
準備：フローネットワーク

カット:
頂点を『sを含む部分集合S』と
『tを含む部分集合T』に分割する

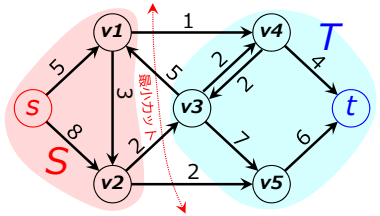
カットセット:
部分集合SとTの間をつなぐ辺の集合

カットの容量:
カットセットのうち S→T方向の辺の容量総和（逆向きの辺は無視する）

最小カット
容量が最小となるカット



カット　　：S={s, v2}, T={v1,v3,v4,v5,t}
カットセット：{(s,v1), (v1,v2), (v2,v3), (v2,v5)}
カット容量　：5 + 2 + 2 = 9



最小カット： S={s,v1,v2}, T={v3,v4,v5,t}
カット容量： 1 + 2 + 2 = 5

準備：フローネットワーク

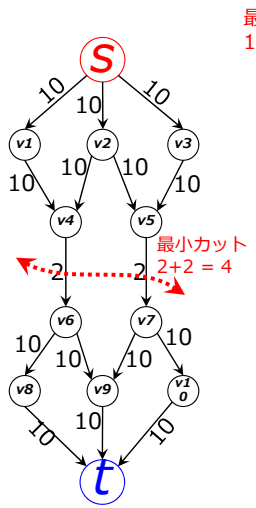
最大フロー最小カットの定理

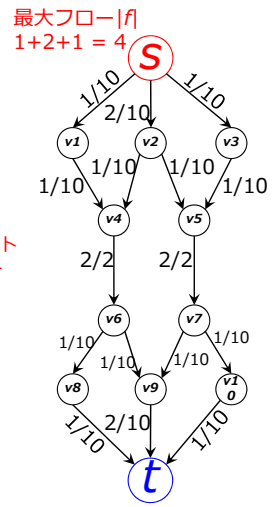
任意のフローネットワークにおいて、
その**最大フローと最小カットは等しい**

最大フローはネットワークの一番細い部分(最小カット)によって決定される

※ 最大フローが流れているとき、始点sから不飽和辺のみを使って到達できる頂点群をSとし、$T = V - S$ とすると、S-Tは最小カットをなす

※ 最大フロー・最小カットの探索には様々なアルゴリズムが存在し、比較的高速に“解ける”ことを知っておいてください



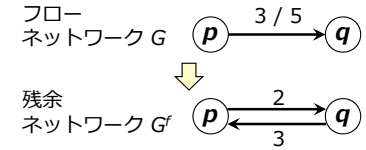


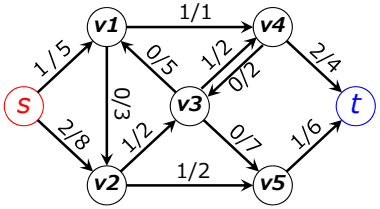
補足資料

最大フローアルゴリズム

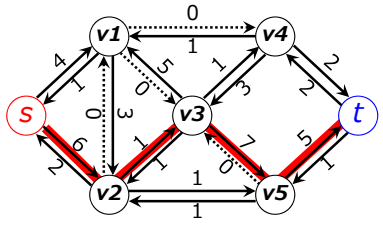
残余ネットワーク G^fとは，フローネットワークGにフローが流れているとき、フローの可変範囲を表すネットワークのこと

増加可能経路とは，残余ネットワーク内のs→t経路のこと





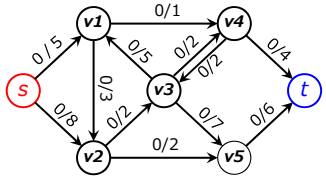
フローネットワーク G



残余ネットワーク G^f

補足資料

最大フローアルゴリズム

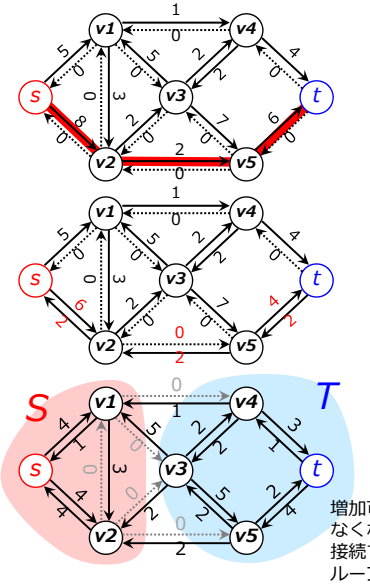


最大フローアルゴリズム（単純なもの）

入力：フローネットワークG

出力：最大フローと最小カット

1. フローを0で初期化
2. 残余ネットワークを構築
3. 増加可能経路Pが無くなるまで下を繰り返す
   3-1) 増加可能経路 Pの探索
   3-2) 経路Pに沿ってフロー追加



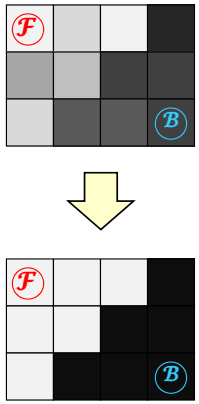
増加可能経路Pがなくなったら，Sと接続する全頂点をグループSに．

グラフカット領域分割：コスト関数

入力：画像、制約画素集合（前景F・背景B）

出力：以下を満たす二値化

- 制約画素pは必ず制約を満たす
- 非制約画素pは、その特徴(色など)が前景画素Fに近ければ前景に、Bに近ければ背景になる
- 境界は特徴の異なる画素間を通る



$$\underset{\{L_p|\,p\in\Omega\}}{\operatorname{argmin}} \sum_{p\in\Omega} E_1(L_p) + \lambda \times \sum_{(p,q)\in\mathcal{N}} E_2(L_p, L_q)$$

Ω ： 全画素集合
$\mathcal{N}$ ： 近傍画素集合
L_p ： 画素$p\in\Omega$につくラベル値 {*fore, back*}のどちらか

グラフカット領域分割：コスト関数

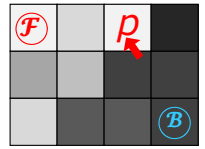
$$\underset{\{L_p|\,p\in\Omega\}}{\operatorname{argmin}} \sum_{p\in\Omega} E_1(L_p) + \lambda \times \sum_{(p,q)\in\mathcal{N}} E_2(L_p, L_q)$$

Ω ： 全画素集合
$\mathcal{N}$ ： 近傍画素集合
L_p ： ラベル値 {*fore, back*}

$E_1(L_p)$：『データ項』画素pのラベル付の不正確さに反応する項

画素pが前景制約画素に似ているなら…

$E_1(L_p = fore)$ ← 小
$E_1(L_p = back)$ ← 大



定義は論文によって色々
右は一例

$$E_1(L_p = fore) = \begin{cases} 0 & p \in \mathcal{F} \\ \infty & p \in \mathcal{B} \\ t_p^{fore} & other \end{cases}$$

$$E_1(L_p = back) = \begin{cases} \infty & p \in \mathcal{F} \\ 0 & p \in \mathcal{B} \\ t_p^{back} & other \end{cases}$$

$$t_p^{fore} = \frac{d_p^{\mathcal{F}}}{d_p^{\mathcal{F}} + d_p^{\mathcal{B}}}, t_p^{back} = \frac{d_p^{\mathcal{B}}}{d_p^{\mathcal{F}} + d_p^{\mathcal{B}}}$$

$$d_p^{\mathcal{F}} = \min_{k\in\mathcal{F}} ||\mathbf{c}_p - \mathbf{c}_k||$$

$$d_p^{\mathcal{B}} = \min_{k\in\mathcal{B}} ||\mathbf{c}_p - \mathbf{c}_k||$$

※$\mathbf{c}_p$は画素pの画素値

※ t_p^{fore}は画素pの色が前景画素に似るほど小さくなるよう指定する

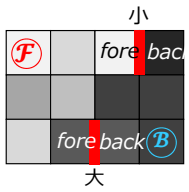
グラフカット領域分割：コスト関数

$$\underset{\{L_p|\,p\in\Omega\}}{\operatorname{argmin}} \sum_{p\in\Omega} E_1(L_p) + \lambda \times \sum_{(p,q)\in\mathcal{N}} E_2(L_p, L_q)$$

Ω ： 全画素集合
$\mathcal{N}$ ： 近傍画素集合
L_p ： ラベル値 {*fore, back*}

$E_2(L_p, L_q)$：『平滑化項』 隣り合う画素が似た特徴（色）を持つときは、なるべく同じラベルをつける

☆隣接画素 *p,q*に同じラベルをつける
→ $E_2 = 0$
☆隣接画素 p,qに違うラベルをつける
*p,q*が似た色を持つ → E_2は大
*p,q*が遠い色を持つ → E_2は小



定義は論文によって色々
右は一例

$$E_2(L_p, L_q) = \begin{cases} 0 & L_p = L_q \\ \dfrac{1}{1+\|\mathbf{c}_p - \mathbf{c}_q\|} & other \end{cases}$$

$\mathbf{c}_p$は画素*p*の画素値

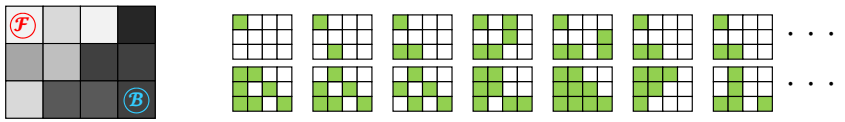
グラフカット領域分割：コスト関数

$$\underset{\{L_p|\,p\in\Omega\}}{\operatorname{argmin}} \sum_{p\in\Omega} E_1(L_p) + \lambda \times \sum_{(p,q)\in\mathcal{N}} E_2(L_p, L_q)$$

Ω ： 全画素集合
$\mathcal{N}$ ： 近傍画素集合
L_p ： ラベル値 {*fore, back*}

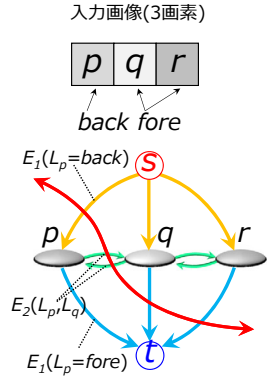
この問題は解くのが難しい

- 局所最小解が多い
- 全通り検索する↓？ さすがに組み合わせが多すぎ（3×4画像でも2^{12} = 4096通り）



『この問題の大域最小化解は、フローネットワークの最小カットにより高速に求められる』 [Boykov Y., Jolly M-P. *MICCAI*, 276-286, 2000.]
※L_pが二状態をとる場合（二値化）に限る
※グラフカット発見以前はやきなまし法（ランダムウォーク）で解いていた

グラフカットを用いた最適化



$$E_1(L_p = back) + E_1(L_q = fore) + E_1(L_r = fore) + E_2(L_p, L_q)$$

$$\underset{\{L_p|\,p\in\Omega\}}{\operatorname{argmin}} \sum_{p\in\Omega} E_1(L_p) + \lambda \times \sum_{(p,q)\in\mathcal{N}} E_2(L_p, L_q)$$

画像からフローネットワークを構築

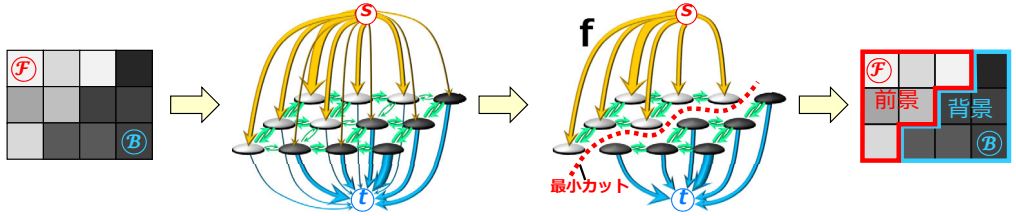
- 頂点*V*：全画素，始点 *s*, 終点 *t*
- 辺 *E* ：右図

フローネットワークをカットし

- *s* に連結する画素にラベル *fore* をつける
- *t* に連結する画素にラベル *back* をつける

→ カット容量 がコストに対応する
→最小カットを求めればコスト最小化できる

グラフカットを用いた最適化



1) 画像からフローネットワークを構築
- 頂点：全画素，始点 *s*, 終点 *t*
- 辺 *E*: 図の通り
- 辺の容量：コスト関数を利用

2) ネットワークの最小カットを計算
- *s* に連結する画素にラベル *fore* をつける
- *t* に連結する画素にラベル *back* をつける

カットされた辺の容量がコスト関数に対応
→ 最小カット容量がコスト関数に対応する

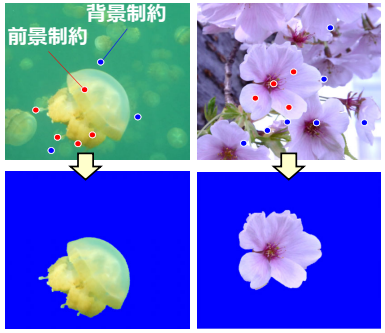
グラフカット領域分割

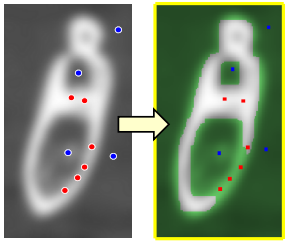
利点

高速・高精度
高次元化が容易
UIと相性がよい

欠点

境界が不明瞭な領域には利用し難い
血管・筋膜等、細い・薄い形状には不向き





まとめ：画像領域分割

- 画像領域分割とは
- 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 (次回)
- 曲面再構成法　(次回)