クラスタリングの手法を用いた
画像処理に関する研究

2000年3月

春日 秀雄

①

クラスタリングの手法を用いた画像処理に関する研究

2000年3月

春日 秀雄

目 次

第 1 章

序論

1.1 はじめに

ハードウェアの進歩に伴い，コンピュータ上での表現技術も大幅に進歩した．特に，画像に関する表現力は，10 年前とは比べ物にならないレベルに達している．それは，とりもなおさず，情報の大容量化をも意味している．実際，グラフィック描画用のメモリ（VRAM）の大きさもこの 10 年で 10 倍以上になっており，表示される色数，解像度も大幅に上がっている．特に色数では，1 ピクセルあたり 3 バイト（24 ビット）のデータが与えられ，人間の認識力を越える 16,777,216 もの色が表現されるまでに至った．

画像処理の手法に関する研究は，古くから行なわれてきた分野である．だが，画像を扱うハードウェアが次々と新しくなるにつれて，計算機上で表示される画像も大きく様変わりした．それら表現技術の進歩だけでなく，計算機の処理能力も大幅に上がっている．だが，大容量の画像データを扱うには，より効果的な手法が求められるようになってきた．また，画像の多様化により，従来の手法を単純に拡張しただけでは対応が難しい分野も生まれてきた．

そのため本論文では，近年の多彩なカラー画像を処理する手法のひとつとして，クラスタリングアルゴリズムを用いた手法を本論文では紹介する．本研究では特に，色情報に対するクラスタリングを効果的に利用する画像処理の手法を研究している．

1.2 本研究の意義

クラスタリングアルゴリズムは，データ圧縮，量子化，パターン認識，学習といった様々な分野で利用されている．クラスタリングアルゴリズムとは，多数のサンプルをその類似性によって，いくつかの集合に分類するアルゴリズムである．与えられたサンプルをいくつかの集合に分類することをクラスタリング，またはクラスタ化という．その集合はクラスタと呼ばれ，そこに含まれるサンプルは類似した特徴を示すことになる．クラスタリングの目的は，類似するサンプル同士をまとめることにある．そのため，類似性を評価するパターン認識などの分野で多く利用される．また，大きすぎるデータを処理する際に，データをいくつかのクラスタに分け，クラスタ単位で処理を行うことによって大量のデータを容易に扱えるようにするといったデータ圧縮的な目的も存在する．これまでに多く行われてきたクラスタリングを用いた画像処理の研究としては，カラー画像の領域分割の研究があげられる．これは，色および位置の情報を元にクラスタリングを行い，画像をいくつかの領域に分割する手法についての研究である．本研究でも，カラー画像の色情報をクラスタリングして利用する手法を研究しているが，結果の評価の難しい領域分割ではなく，カラー画像の色量子化と，カラー文書画像からの文字抽出についての研究を行っている．

カラー画像の色量子化も，カラー文書画像からの文字抽出も，類似色をクラスタリングによってまとめることが重要なポイントである．色の差は一般に，色空間における距離で表され，容易に計算することができ，色の近い遠いはそれによって判断できる．しかし，与えられる画像データのパターンによっては，使われている色が大きく偏っていたりして，どこまでを類似色とするかが異なる．カラー画像では，個々の画像データごとに，どの色までをを類似色とするかが異なるので，色の類似性を利用して画像処理を行う場合には，本論文で用いているようなクラスタリングアルゴリズムを用いて類似色を求めることが有効である．このテクニックを用いることによって，カラー文書画像からの文字抽出では，さまざまな条件の画像から文字色と背景色を分離することが可能になる．また，1 ピクセルあたり 24 ビットのデータを持つカラー画像の色情報は，多分に冗長性を備えている．そこで，カラー画像の色量子化によって，冗長な情報を取り除き，複数の類似性の高い色をま

とめることは画像処理の効率化につながる．

1.3 本論文の内容と構成

本論文では，カラー画像の色情報をクラスタリングした結果を利用した，いくつかの画像処理技術について述べる．しかし，クラスタリングアルゴリズムといってもさまざまな手法が存在する．そこで，まずクラスタリングアルゴリズムの種類について簡単に説明すると，クラスタリングアルゴリズムには，大きく分けて階層的クラスタリングと非階層的クラスタリングの2種類が存在する．階層的なクラスタリングとは，類似性の高いサンプル同士を結合させてクラスタを作り，更にそのクラスタ同士の類似性からクラスタをまとめたクラスタを作るといったことを繰り返す，階層的な手法である．非階層的なクラスタリングとは，階層的なクラスタリングのよう一つ一つのサンプルをまとめていくのではなく，最初から存在するクラスタの形を変えることによって最終的な解を求める手法である．本論文で用いているクラスタリングアルゴリズムは，非階層的なクラスタリングアルゴリズムである．

第2章では，カラー画像の色量子化について述べる．ここでは，24ビット（1667万色）の色情報を持つフルカラー画像の色を256に量子化するアルゴリズムを提案する．フルカラーの画像を色量子化するにあたっては，量子化した代表値と実際の画素値の誤差の2乗の和が最小になるような解を求めた．なお，量子化の評価としては，信号対雑音比（SNR）を用いている．そのための手法として，K-means法と呼ばれるクラスタリングアルゴリズムを高速化したアルゴリズムを考案し，利用することにした．この高速K-means法は，K個のクラスタをクラスタ中心（重心）をもとに大まかに分類することにより，頻繁に行われる距離計算の回数を減らし，実行時間を短縮したアルゴリズムである．この高速K-means法はクラスタ数Kが大きいときに特に有効で，K=256の場合，実験的に4分の1以下の実行時間になった．

第3章では，カラー文書画像からの文字抽出について述べる．ここでは，雑誌等のカラー印刷物から文字認識を行う場合に必要な，文字と背景の切り離しに関するアルゴリズムを

提案している．本研究では，カラー印刷物中に印刷されている文字の色が単色で書かれいることを前提として，色情報をクラスタリングすることによって文字色を背景色から離させている．この時，色の類似性をより正確に捉えるために，サンプルのクラスタへ所属の程度が分かるファジイクラスタリングを用いている．ファジイクラスタリングによて得られる帰属度という値を用いることにより，クラスタリング後にそれぞれの色がと程度の類似性を持っているか分かるようになり，微妙な色の差などを有効に識別できるうになる．

第4章では，まとめとして，第2章，第3章での結果を要約し，今後の課題等につい述べる．

第 2 章

カラー画像の色量子化

2.1 序言

色量子化問題は，ディスプレイの性能が不十分であった時期に多く研究された問題である．フルカラーの画像を，ルックアップテーブルを用いたディスプレイで表示するためには，適切な色量子化が必要である（一般的な色量子化については，文献 [1]，[2] 等に紹介されている）．本章では，24 ビットの色情報を持つフルカラー画像の色を 256 に量子化するアルゴリズムについて述べる．色量子化アルゴリズムとしては，頻度法 [3]，Median Cut Algorithm（MCA）[3]，渡辺のアルゴリズム [4] などがある．頻度法は，代表となる色を出現頻度により選ぶアルゴリズムである．頻度法の欠点は，使用されている色数が多く，まんべんなく使われている場合に不適切な色が代表色に選ばれることが多いことである．MCA は，RGB 色空間を次々と軸に垂直に分割し，いくつかの直方空間を作り，その領域の平均値をその空間内の色の代表値とするアルゴリズムである．色空間の分割に際しては，データの分布の幅を元に分割空間を選び，中央値で空間を分割する．MCA は頻度法よりも高速であるが，色空間の分割の仕方が余り適切ではなく，量子化精度はそれほど良くない．渡辺のアルゴリズムは MCA の改良アルゴリズムである．色空間を分割するという考え方は MCA と同じであるが，分割に際しては，分散を元に分割空間を選び，平均値で空間を分割している．渡辺のアルゴリズムは MCA と同じく高速で，量子化精度も頻度法や MCA よりは良い．しかし，従来の手法は処理時間の点では優れているが，量子化の精度の点ではまだ改良の余地がある．限られた色で，再現性の高いカラー画像を表示することに関し

ては，ディザリングを用いた手法も存在する [5] が，ディザ化した画像では本来の画像の
報を十分に読みとることは難しい．そこで，本論文では従来の手法より量子化精度の高
アルゴリズムを紹介する．アルゴリズムの量子化精度の評価として，信号対雑音比（SN
を評価値として使っている．この量子化精度を上げるのには，K クラスタリング問題て
いられる K-means 法を利用することが考えられる（文献 [1] 参照）．

色量子化問題は，評価基準を距離二乗和最小とする K クラスタリング問題と考えるこ
ができる．しかし，K-means 法は，K や要素数（本件では色数）が多い場合には処理に時
がかかる．そこで，K 個のクラスタを大まかに分類することにより高速化をはかる K-mea
アルゴリズムを考案した．本アルゴリズムは，通常の K-means 法と全く同じ解を求める
とができ，実行時間は通常のものの 4 分の 1 程度になる（K=256 の時）．本章では，こ
アルゴリズムを使用して，従来の手法より量子化精度の点で優れたアルゴリズムを提案
る．なお，本章で提案する高速 K-means 法は，K-means 法を利用する他の応用分野へも
用可能である．

2.2 従来の色量子化アルゴリズム

色量子化問題は過去に多く研究されてきた．そこで，これまでに研究された，従来の色量子化アルゴリズムとして，頻度法，Median Cut Algorithm，渡辺のアルゴリズムの3つのアルゴリズムについて簡単に述べる．

2.2.1 頻度法

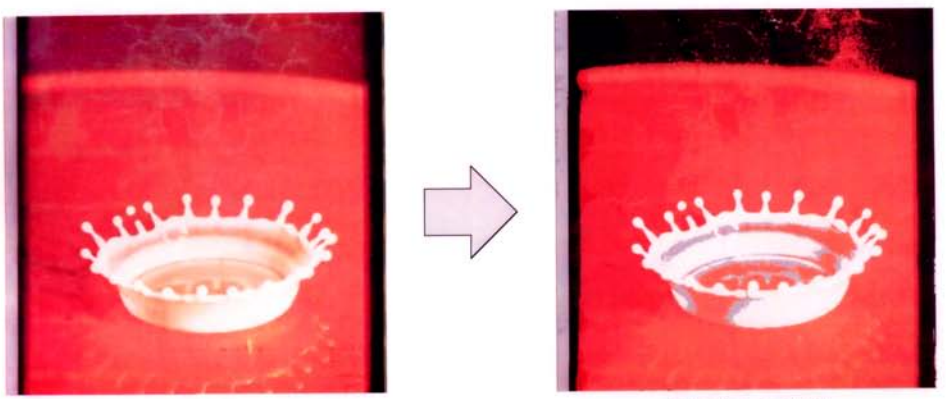
STEP1: 原画像のR，G，Bの色ヒストグラムを作成する

STEP2: ヒストグラムより，出現頻度の高い順に256色を選び代表色とする

STEP3: 各色に対し，距離の最も近い代表色を割り当てる

頻度法とは最もシンプルな色量子化アルゴリズムである．代表色は色の出現頻度により選ばれる．このアルゴリズムの欠点は，色の分布が広い場合（色数が多い場合），出現頻度の小さい色には適切な色が割り当てられないことが多いことである．なお，ヒストグラムの作成の際には，RGB各8ビットのうち，その上位4〜5ビットのみを用いてヒストグラムを作るのが普通である．もし，7ビット以上を用いてヒストグラムを作成した場合，類似色の中に高いヒストグラム値を持つものが多く存在する可能性が高くなり，図2.1のように同系色の色ばかりが大量に代表色に選ばれてしまう．ヒストグラムを作成する際のRGBの各有効ビット数を少なくすれば，色はまんべんなく広い範囲から取られることになるが，RGBの各階調が落ちてしまうので，微妙な色差を表現できなくなってしまう．このように，頻度法は量子化精度についてはいくつかの問題を持っている．また，このアルゴリズムは，色数が多い場合にはヒストグラムのソート処理に時間がかかるという欠点も持つ．

なお，本章の後半における比較実験では，上位5ビットからヒストグラムを作成した頻度法についてと比較している．



原画像　　量子化した画像

図 2.1: 上位 7 ビットからヒストグラムをつくった時の頻度法

2.2.2 Median Cut Algorithm

STEP1: 原画像の R，G，B の各上位 5 ビットに対する色ヒストグラムを作成する

STEP2: 次の処理を領域の数が 256 になるまで繰り返す

1. ヒストグラムより，各領域の分布の広がり（0 ではないヒストグラム値を持つ，最もはなれた 2 点間の長さ）を軸ごとに求め，その長さの最も長い軸を持つ領域を見つける
2. 分布の広がりの最も長い軸を持つ領域を，その軸の中央値で分割する

STEP3: 各領域の平均色を計算し，それを領域内の色の代表色とする

Median Cut Algorithm(MCA) は RGB 色空間を図 2.2のように 256 の直方体状の空間に分け，その領域の内部にある色をその領域の平均色で代表させるアルゴリズムである．分割に際しては，分布の広がり（0 ではないヒストグラム値を持つ，最もはなれた 2 点間の長さ）の最も長い軸が分割される．分割に際しては，その領域内の要素の中央値を用いる．このアルゴリズムの特徴としては，処理が高速であることが挙げられる．処理にかかる時間は頻度法と違い，色の数で変わることはない．この手法の欠点は，データによっては分割の仕方が適切とは限らないために不自然な色が割り当てられることがあることである．特に，分布は広くてもその領域内に含まれる色数が少ない場合には，分割は不適当である．分割する軸を分布の幅だけで決めているので，その範囲に含まれる色数や色の散らばり具合を無視していることになる．また，分割値が中央値であることもデータによっては適切とは限らない．色数が少ない場合や，その領域内の色が偏って存在している場合には，生成される領域が不適切になりやすい．

このように，MCA は高速であるが，量子化精度がデータによって大きくことなる可能性がある．

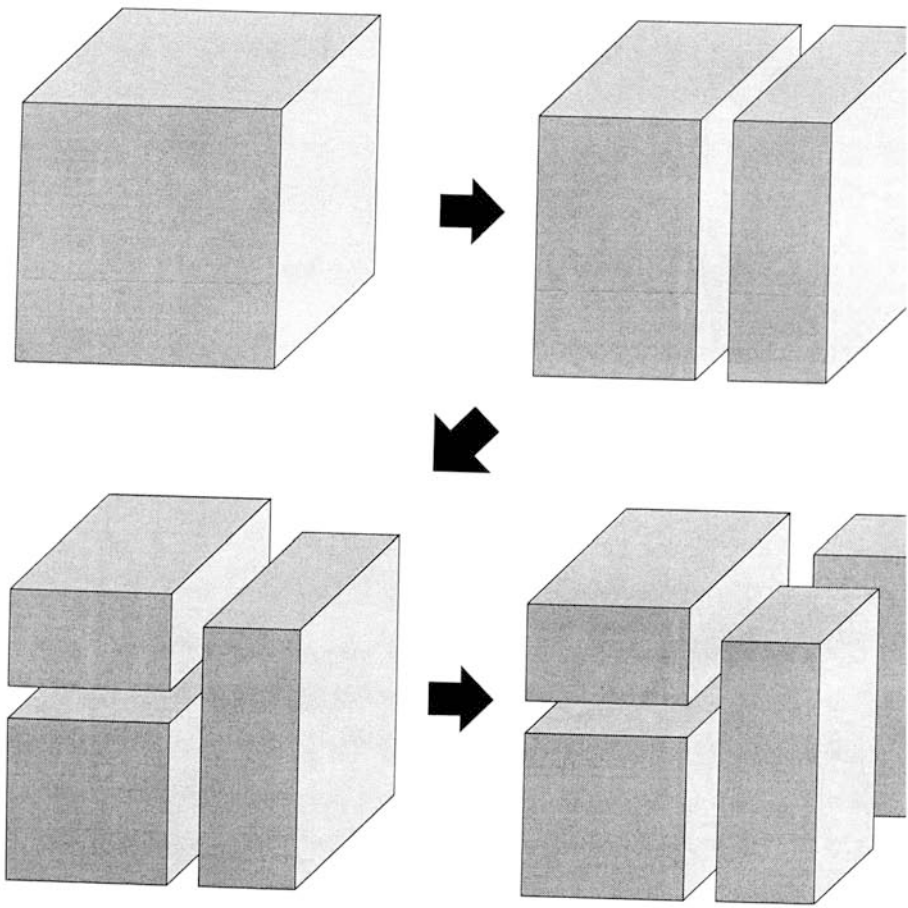


図 2.2: MCA における色空間の分割例

2.2.3 渡辺のアルゴリズム

STEP1: 原画像のR，G，Bの各上位5ビットに対する色ヒストグラムを作成する

STEP2: 色空間を均等に64（全軸を4等分にする）の領域に分ける

STEP3: 次の処理を5回繰り返す

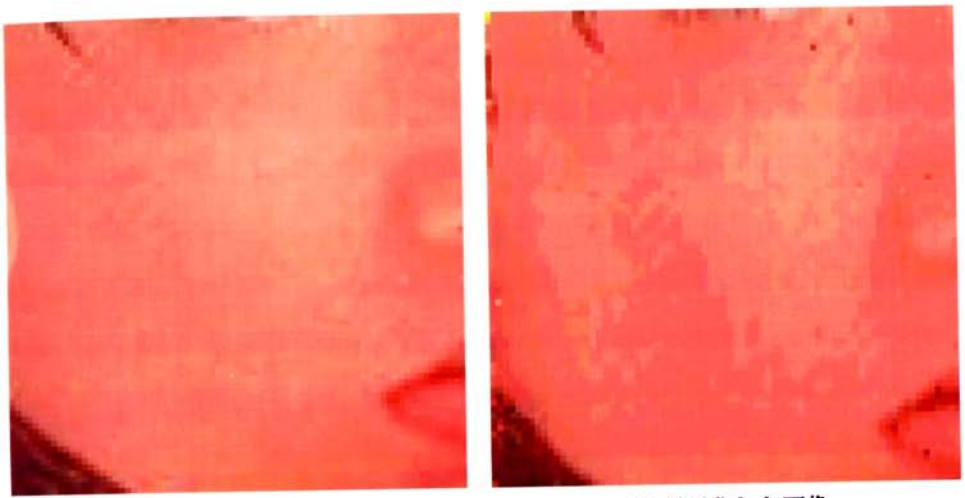
1. ヒストグラムより，各領域の各軸方向の分散が最大となる軸を求める
2. 領域を分散の大きさでソートする
3. 分散の大きなものから32の領域を選び，分散が最大となる軸をその軸の平均値で分割する

STEP4: 空である領域を取り除いたうえで，領域の数が256に満たない場合，分散の最も大きな領域を分割する処理を領域の数が256になるまで繰り返す

STEP5: 各領域の平均色を計算し，それを領域内の色の代表色とする

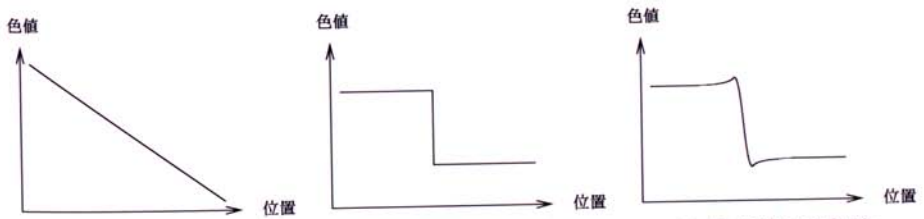
渡辺のアルゴリズムはMCAの改良型であり，処理の流れは似ている．このアルゴリズムも非常に高速である．MCAとの大きな違いは分割する領域を分散によって決めていることである．また，分割値は平均値になっている．色の分布を考えているため，このアルゴリズムは非常に優秀である．処理は高速であるし，近似精度も前記の2つより優れている．ただ，多数の色が連続的に緩やかに変化している画像では，図2.3のようなマッハ効果と呼ばれる疑似輪郭が発生するという欠点がある．マッハ効果とは，本来段階的に変化しているはずの色が急激に変化することによって，その境界が帯状にくっきり浮き出してしまう効果である（図2.4参照）．この画像は人の肌の部分だが，図2.3（b）では割り当てられた色数が少ないために，肌色のグラデーションがうまく表現されず，色の変り目が特に目立つ．これは，分割領域を分散の大きさで決めていることから起る．分散によって分割領域を決めた場合，各領域間の色数の差を考えなくなってしまう．それによって，色数は少ないが，分散が大きい領域が優先的に分割されることになる．疑似輪郭の発生を押えるためには，多数の色が連続的に変化している部分により多くの色を割り当てる必要がある．

この，マッハ効果の発生というわずかな欠点を除けば，渡辺のアルゴリズムは高速で秀なアルゴリズムである．



(a) 原画像　　(b) 量子化した画像

図 2.3: 渡辺のアルゴリズムを用いた際のマッハ効果



(a) 原画像の色値　(b) 処理後の色値　(c) 見かけ上の色値

図 2.4: マッハ効果による見かけ上の色値

2.3 K クラスタリング問題

K クラスタリング問題とは，与えられた n 個のデータをある評価基準に従い K 個のグループ（クラスタ）に分類する問題である．色量子化問題も，色を K 個のクラスタに分類し，各色をそのクラスタの重心で代表させると考えれば，K クラスタリング問題とみなすことできる．本研究における評価基準は，原画像の各画素値と量子化した色の，RGB 色空間におけるユークリッド距離の 2 乗の平均を最小化するというものである．この評価値を Cost とすると，これは次のように表すことができる．

$(M \times N)$ ピクセルの画像で (i,j) 座標の原画像の R，G，B の画素値を $r_1(i,j), g_1(i,j), b_1(i,j)$ とし，量子化した色で表される画像の R，G，B の画素値を $r_2(i,j), g_2(i,j), b_2(i,j)$ とする．

$$\Delta r(i,j) = r_1(i,j) - r_2(i,j) \tag{2.1}$$

$$\Delta g(i,j) = g_1(i,j) - g_2(i,j) \tag{2.2}$$

$$\Delta b(i,j) = b_1(i,j) - b_2(i,j) \tag{2.3}$$

$$X(i,j) = (\Delta r(i,j))^2 + (\Delta g(i,j))^2 + (\Delta b(i,j))^2 \tag{2.4}$$

$$\mathrm{Cost} = \frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} X(i,j) \tag{2.5}$$

2 次元以上の空間での一般的な K クラスタリング問題は NP 完全であり，最適解を求める効率的な解法は存在しない．一般的には，2 クラスタリングを再帰的に繰り返すことによって K 個のクラスタを作る手法が用いられる．しかし，最適な 2 クラスタリングを求めるのにも，要素数が多い時には多大な時間がかかる．ランダムに選んだ少数の要素からクラスタリングを行なうランダマイズドクラスタリング [6][7] というものによって，ある程度高速な処理が可能であるが，本研究では適当な初期点に K-means 法を適用するという手法を使っている．というのも，最適な 2 クラスタリングを再帰的に繰り返しても最適な K クラスタリングができるとは限らず，初期点がそれなりに適切ならば，多少の誤差があっても K-means 法によって十分な精度の局所最適解が見つかるからである．そこで，本研究

では，K クラスタリング問題を解くにあたり，K-means 法を改良し，高速にしたものをいている．

2.4 アルゴリズム

本手法の大まかな流れは K-means 法と同じである．

K-means 法のアルゴリズムを簡単に述べると次のようになっている．

STEP1: K 個の初期クラスタ中心（図 2.5（a）の△）を適当に決める

STEP2: すべてのデータを最も近いクラスタ中心のクラスタに分類する

STEP3: 新たにできたクラスタの重心（図 2.5の×）をクラスタ中心とする

STEP4: 新たなクラスタ中心がすべて以前と同じ（図 2.5（d）のような状態）であれば了し，そうでなければ STEP2 に戻る

図 2.5の流れのように，徐々に局所最適解に収束していくアルゴリズムである．本手法特徴である高速 K-means 法は，通常の K-means 法の STEP2 を改良したものである．

高速 K-means 法の紹介の前に，初期点の求め方を説明する．

2.4.1 初期点の選定

大まかな流れは次のようになる．

1. 色ヒストグラムを作成する
2. 色ヒストグラムを元に色空間を分割する
3. 各空間の代表値を決定する

色ヒストグラムの作成に際しては，RGB 各 8 ビットのデータの四捨五入した上位 5 トのみを使う．K-means 法はデータ数（色数）が多ければ多いほど処理に時間がかか

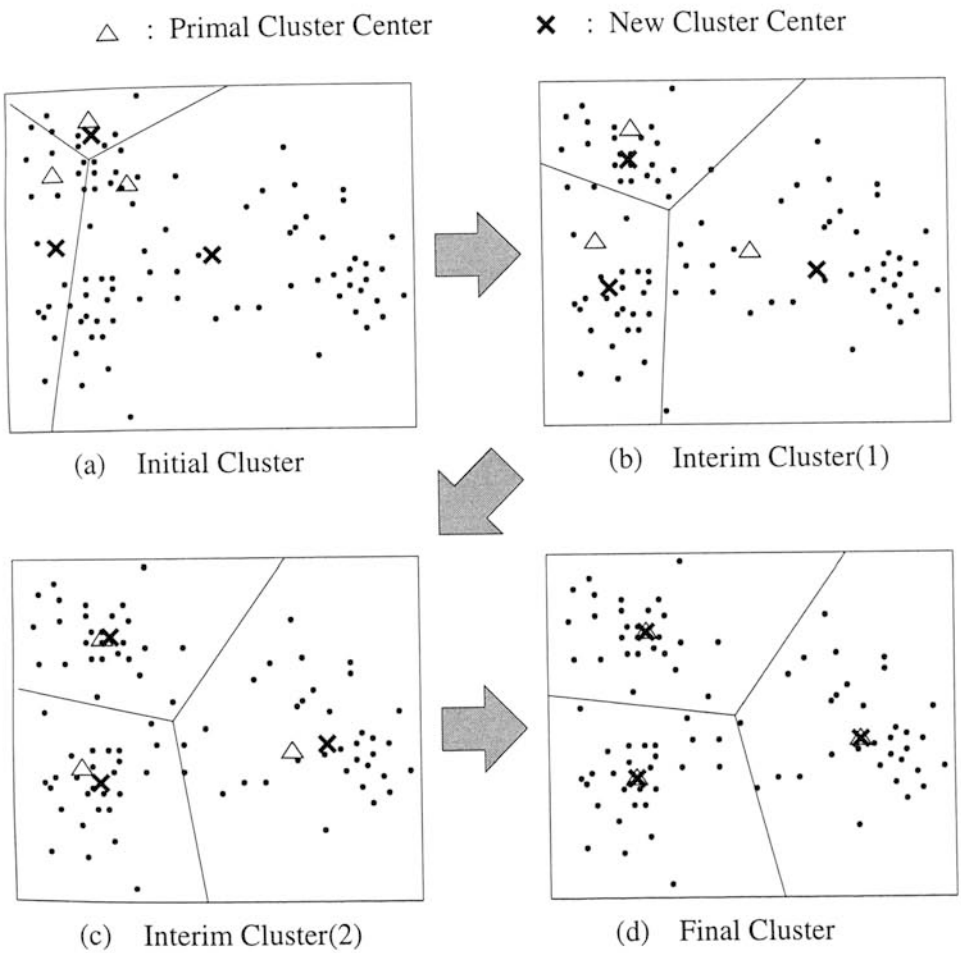

△ : Primal Cluster Center
× : New Cluster Center
(a) Initial Cluster
(b) Interim Cluster(1)
(c) Interim Cluster(2)
(d) Final Cluster

図 2.5: K-means 法

め，データ数をある程度押える必要がある．四捨五入した上位5ビットのみを使うことよる下位ビットの分の誤差は，256に量子化する時にはほとんど無視できるものである．

この色ヒストグラムを元に，色空間を256に分割する．この時，色空間の分割は，R，GBの各軸方向に垂直な面で行なう．

色空間を256の領域に分割するにあたり，まずはデータの散らばりから最も分割に適た領域を選ぶ．最初は領域が1つしかないが，次々と領域分割を繰り返すうちに領域のは増えてくる．分割する領域は次のようにして決める．

各領域のR，G，Bの各軸方向のデータの分布の大きさSを調べる．各軸方向のデーの分布の大きさSは，次の式で与えられる値である．

$$S_r = \sum_{i=1}^{n}(r_i - \bar{r})^2 \quad (2.$$

$$S_g = \sum_{i=1}^{n}(g_i - \bar{g})^2 \quad (2.$$

$$S_b = \sum_{i=1}^{n}(b_i - \bar{b})^2 \quad (2.$$

$r_1, r_2, ..., r_n$はその領域の含まれるn個のデータのそれぞれのRの値．$\bar{r}$は$r_1, ..., r_n$のR平均値．g_i,$\bar{g}$, b_i,$\bar{b}$もそれぞれG，Bに関して同様に定義される．

S_r, S_g, S_bのうち最も大きな値をその空間のデータの分布の大きさとする．全領域のうその値が最も大きな領域を分割する領域とする．

次にその領域を2つに分割する．分割は，最もSの大きかった軸に対して垂直に行な軸上の分割面のある値のことを分割値と呼ぶことにする．

分割値は，最初の8回はヒストグラム値が最小となる所とする（最小値が複数ある場もっとも平均値に近い所とする．また，最小値がその空間の端であった場合は，その空のもっとも小さな極小値をとる所とする）．以後，分割した空間が256になるまではデタの平均値とする．最初の数回の分割値を最小値とすることによって，偏った分布のデタに対して，より的確な分割ができることが多い．

このようにして空間を256に分割し，各領域のデータの平均値をK-means法で用いるラスタ中心の初期点とする．

この初期点の求め方は，MCA や渡辺のアルゴリズムに近い．

2.4.2 高速 K-means 法

本手法では，K-means 法と同様な局所最適解が求まり，K-means 法より高速なアルゴリズムを用いる．これは，K 個あるクラスタ中心を大まかに分類することによって，頻繁に行なわれるデータとクラスタ中心との距離計算の回数を減らすものである．このアルゴリズムは次のようになっている．

STEP1: n 個の要素からなる集合をクラスタリングするに当たり，K 個の初期クラスタ $S_j(j = 1, 2, ..., K)$ と，その中心 z_j を適当に決める．

STEP2: K 個の初期クラスタ中心 z_j を要素として次のようなクラスタリングを行ない，m 個のクラスタを作る．このクラスタリングによって生じるクラスタをマクロクラスタ $T_i(i = 1, 2, ..., m)$ と呼ぶことにする．マクロクラスタの中心 w_i は，T_i に含まれる z_j の平均値．

1. 各領域の各軸方向の要素 z_j の分布の広がりを式 (2.6) (2.7) (2.8) のようにして求める．
2. 分布の広がりが最大となる軸を持つ空間を，その軸に垂直な面で分割する．分割値は要素 z_j の平均値．
3. 領域の数が m 以下ならば STEP2.1 に戻る．

STEP3: K 個のクラスタ中心 z_j を最も近いマクロクラスタに分類する．

STEP4: m 個のマクロクラスタのマクロクラスタ中心 w_i を更新する．

STEP5: K 個のクラスタ中心 z_j を再び最も近いマクロクラスタに分類する．その際に，マクロクラスタの半径 $r_1, r_2, ..., r_m$ を計算する．半径は，マクロクラスタに所属する要素 z_j の中で，マクロクラスタ中心から最も離れた要素までの距離とする．

STEP6: n 個ある要素 x を次の方法で K 個のクラスタに分類する．

1. 要素 x の所属していたクラスタを S_j とするとき，仮の最も近いクラスタ中心までの距離

$$mindist(x) = \|x - z_j\| \tag{2.9}$$

を求める．なお，$\|x-z_j\|$ は，要素 x からクラスタ中心 z_j までの距離である

2. 要素 x からマクロクラスタ T_i の外縁までの距離

$$distT(i,x) = \|x - w_i\| - r_i \tag{2.10}$$

を計算する（図 2.6参照）．

3. 次の条件

$$distT(i,x) < mindist(x) \tag{2.11}$$

を満たすならば，x とマクロクラスタ T_i に含まれるすべてのクラスタ中心までの距離を計算する．

4. STEP6.3 で求めたクラスタ中心までの距離が $mindist(x)$ より小さければ，その値を $mindist(x)$ とし，その時のクラスタに要素 x を所属させる．

5. $i = 1$ から m になるまで STEP6.2，STEP6.3，STEP6.4 を繰り返すことにより，最も近いクラスタに x を所属させる．

STEP7: STEP6 で新たに得られたクラスタ S_j の新しいクラスタ中心を求める．クラスタ中心 z_j は，そのクラスタに所属する要素 x の平均値である．

STEP8: すべてのクラスタ中心が以前のクラスタ中心と一致すればアルゴリズムは収束したものとして終了する．そうでなければ STEP3 に戻る．

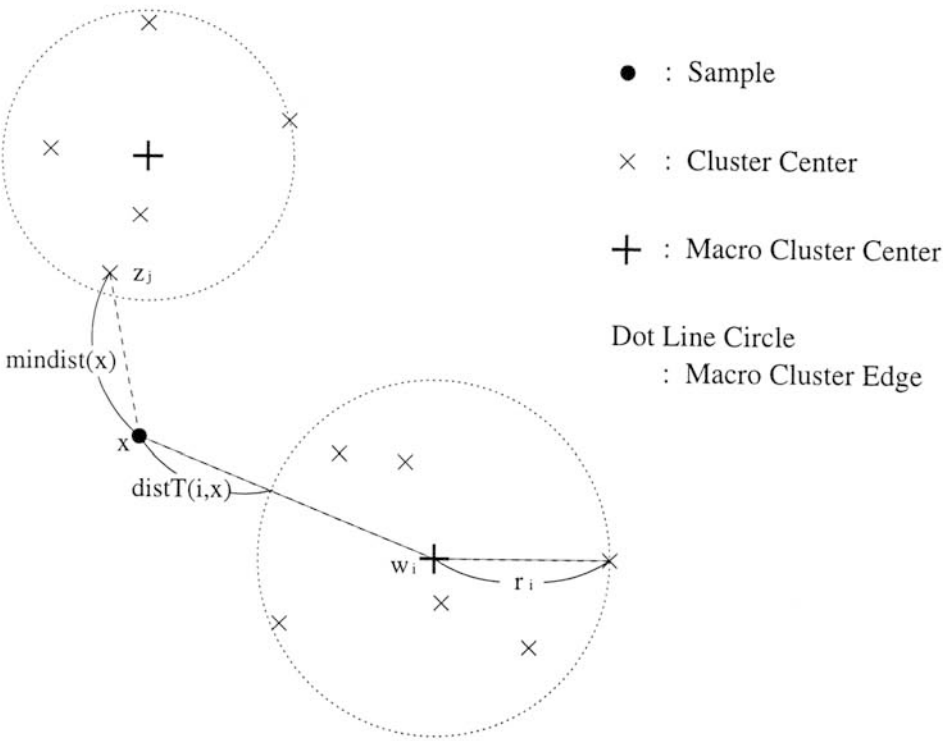

● : Sample
× : Cluster Center
+ : Macro Cluster Center
Dot Line Circle
: Macro Cluster Edge
z j
mindist(x)
x
distT(i,x)
w i
r i

図 2.6: マクロクラスタ

2.5 検証

高速 K-means 法が通常の K-means 法と同じ結果になり，なおかつ計算回数も少なくなるという検証をする．

ここではマクロクラスタの作成方法が最適であるかは検証しない．マクロクラスタの作成アルゴリズムについては今だ研究の余地があり，原時点での方法が最適とはいえないが，STEP2 で作られる初期マクロクラスタが適切であるという仮定の元に検証を進める．

計算回数を調べるにあたっては，最も頻繁に行なわれる要素とクラスタ中心のユークリッド距離の計算の回数を調べることにする．

まずは，本アルゴリズムが通常の K-means 法と同様の結果を得ることを証明する．ポイントとなるのは STEP6 である．これは，通常の K-means 法の簡単なアルゴリズム紹介における STEP2 である．この操作で，要素から最も近いクラスタが求められることを示す．

STEP6.1 で仮の最も近いクラスタ中心までの距離 $mindist(x)$ を求める．マクロクラスタ T_i を中心 w_i，半径 r_i の球と考えた時，STEP6.2 では要素 x とマクロクラスタ T_i の外縁部までの距離 $distT(i,x)$ を求めている．そして，ユークリッド空間では次の式が成り立つ．

$$\|x - w_i\| \leq \|x - z\| + \|z - w_i\| \tag{2.12}$$

式（2.12）の両辺から r_i を引くと次のようになる．

$$\|x - w_i\| - r_i \leq \|x - z\| + \|z - w_i\| - r_i \tag{2.13}$$

式（2.13）の左辺は $distT(i,x)$ であるので，次の式が成り立つことになる．

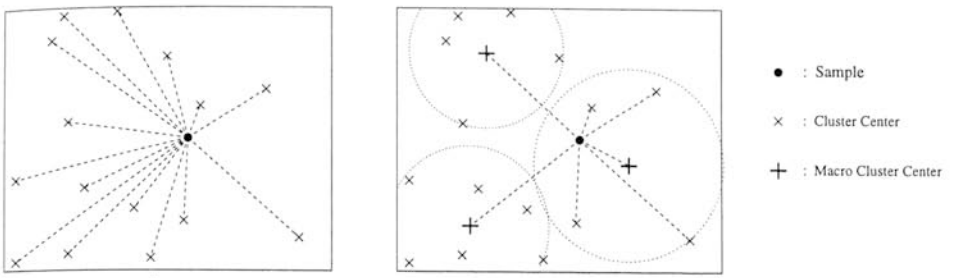
$$distT(i,x) \leq \|x - z\| + \|z - w_i\| - r_i \tag{2.14}$$

z を T_i に含まれる任意のクラスタ中心とすると次の関係が成り立つ．

$$\|z - w_i\| \leq r_i \tag{2.15}$$

式（2.15）を次のようにする．

$$c = \|z - w_i\| - r_i \leq 0 \tag{2.16}$$



(a) K-means法
K-means Algorithm

(b) 高速K-means法
Fast K-means Algorithm

図 2.7: 距離計算の回数

ここで，次の条件について考える．

$$distT(i,x) \geq mindist(x) \tag{2.17}$$

式（2.14）と式（2.16）より

$$mindist(x) \leq distT(i,x) \leq \|x-z\| + c \tag{2.18}$$

よって

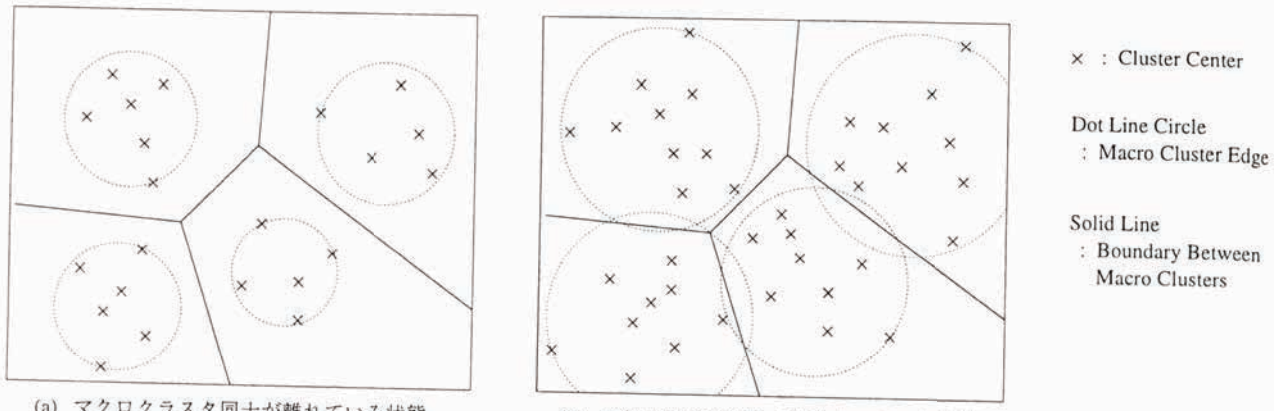
$$mindist(x) \leq \|x-z\| + c \tag{2.19}$$

式（2.16）より $c \leq 0$ であるから，式（2.19）より式（2.17）の成り立つマクロクラスタ T_i には

$$mindist(x) > \|x-z\| \tag{2.20}$$

となるようなクラスタ中心 zは存在しない．よって，条件式（2.11）を満たすマクロクラスタ T_iに含まれるクラスタ中心との距離を計算するだけでも最も近いクラスタを見つけることが可能である．

高速 K-means 法は，クラスタを大まかに分類することにより，余分なデータの距離計算回数を減らし高速化をはかっている．通常ならば図 2.7（a）のように，すべてのクラスタとの距離計算を行なわなければならない所を，図 2.7（b）のように全マクロクラスタと，



(a) マクロクラスタ同士が離れている状態
Macro Clusters are apart from each other

(b) マクロクラスタ同士が重なっている状態
Macro Clusters overlap each other

図 2.8: マクロクラスタの密度

特定のマクロクラスタ内のクラスタとの距離計算を行なうだけで最も近いクラスタを発見できるようになっている．次に，この計算量がどの程度になるか考える．

通常の K-means 法では，あるデータが所属クラスタ（クラスタ中心までの距離が最も小さいクラスタ）を見つけるのに，すべてのクラスタとの距離を調べ，所属クラスタを決めるため，距離計算は K 回行なうことになる．この高速 K-means 法ではデータの距離計算は，1 つのデータにつき STEP6.2 で m 回，STEP6.3 で条件を満たすマクロクラスタに含まれるクラスタの数（<K）回行なわれるだけである．

これは，$m = 1$ や K といった特殊な場合を除いて，K より小さい．m をいくつに設定するかによって計算量の減少率は変わるが，m の値の決定については次のように考えることができる．

図 2.8（a）のように，マクロクラスタ同士が十分にはなれており，各マクロクラスタに所属するクラスタの数がほぼ一定である理想的なデータがあったならば，条件式（2.11）を満たすマクロクラスタはほぼ 1 つになり，そこに含まれるクラスタの数はほぼ$\frac{K}{m}$になる．この時，1 つのデータが所属クラスタを見つけるのに距離計算の回数は約 $m + \frac{K}{m}$ 回になる．これが最小になるのは $m = \sqrt{K}$ の時である．実際には，マクロクラスタ同士が十分に離れていることは少なく，図 2.8（b）のようにマクロクラスタ同士が重なり，条件式（2.11）を満たすマクロクラスタが複数出てくる要素が多くあることが多い．この時，m の数を大きくすればマクロクラスタの半径は小さくなり，マクロクラスタ同士の重なりが少なくなり，

STEP6.3 で行なわれる計算回数を減らすことができる．このことより，m の値は$\sqrt{K}$よりわずかに大きいくらいが良いと考えられる．本研究における条件（次元数 3，クラスタ数 256）の場合，実験的に m の値は $2\sqrt{K}$ が良いとなった．

この高速 K-means 法は，通常の K-means 法よりほぼあらゆる条件で高速であるが，特に K が大きい時ほど効果的である．通常の K-means 法は K に比例した実行時間がかかる（実際にはクラスタ数 K と要素数 n，繰り返し数 t に比例する）．高速 K-means 法は$\sqrt{K}$に近い値で比例する．そのため，このアルゴリズムは K が大きい時ほど効率良く計算時間を減らすことができる．

2.6 実験結果

表 2.1: 実験に使用したデータ

	ファイル名	サイズ	色数 (8bit)	色数 (5bit)
1	aerial1.ppm	256x256	58,600	2,235
2	auto1.ppm	800x600	192,128	3,186
3	bear1.ppm	800x600	272,120	5,180
4	city1.ppm	800x600	197,853	2,008
5	city2.ppm	800x600	193,845	4,337
6	couple.ppm	256x256	26,565	1,685
7	desk1.ppm	800x600	187,554	3,612
8	face1.ppm	800x600	68,098	1,598
9	girl1.ppm	256x256	32,519	2,950
10	home3.ppm	800x600	170,222	3,206
11	home4.ppm	800x600	160,815	3,075
12	mandrill.ppm	512x512	230,427	8,597
13	milkdrop.ppm	512x512	86,727	1,691
14	room1.ppm	800x600	62,909	1,156
15	room7.ppm	800x600	96,585	1,886

実験は 15 枚のフルカラーの ppm ファイルに対して行なった．MCA は Median Cut Algorithm，渡辺は渡辺のアルゴリズムである．

表 2.1が実験に使ったファイルである．色数（8bit）とは RGB 各 8 ビット 16,777,216 色中に使用されている色数である．色数（5bit）とは RGB 各 5 ビット 32,768 色中に使用されているの色数である．本手法では，色ヒストグラム作成の際に色データの下位 3 ビットを無視するので，色数（5bit）の数がクラスタリングの際の要素の数である．

アルゴリズムの評価は信号対雑音比（SNR）で表している．ここで用いている評価値の PSNR は次の式で表される値である．

$$PSNR = 10 \times \log_{10} \frac{3 \times 255^2}{\mathrm{Cost}} \tag{2.21}$$

ここでの Cost とは式（2.5）の Cost である．RGB の値は，それぞれ 0〜255 として計算する．

図 2.9は，m をいろいろと変えた時の，要素とクラスタ中心（マクロクラスタ中心）の距離計算の回数を通常の K-means 法と比較した値のグラフである．縦軸の値は，通常の K-means 法の距離計算の回数と比較しての%表示である．横軸のファイルナンバーは，表 2.1のファイルの番号である．理論的には，前章で述べた通り，m は$\sqrt{k}$より幾らか大きな値が良いはずである．実験では $m = 8, 16, 32, 64$ の 4 つの場合を調べた．

実験では表 2.2より，$m = 32$ の時 19.9%と最も計算回数を減らすことができた．

図 2.10は実行時間を通常の K-means 法と比較した値のグラフである．この値も，通常の K-means 法の実行時間と比較しての%表示である．計算回数が減れば実行時間も減少する．実際の実行時間は，マクロクラスタの計算等にかかる時間のせいで計算量ほどには減少しない．表 2.3より，高速 K-means 法では K=256，$m = 32$ の時，実行時間が通常の K-means 法の 4 分の 1 以下になることがわかる．実際の実行時間は，ファイル 1（aerial1.ppm）の場合，通常の K-means 法で約 17 秒，高速 K-means 法で約 4 秒である（使用したマシンは，Axil320．メモリ 64MB）．ただし，今回の比較実験に用いた K-means 法では，距離計算をすべてユークリッド距離 $d(x_1, x_2) = \sqrt{(r_1 - r_2)^2 + (g_1 - g_2)^2 + (b_1 - b_2)^2}$で行なっている．実際には $d(x_1, x_2)^2$で距離の比較ができ，高速 K-means 法のような平方根の計算が不要となり，計算時間が短縮されるが，K-means 法，および，本アルゴリズムは，距離の定義をユークリッド距離に限定しなくても良いという特徴があり，他の距離の定義でも，どの程度，本アルゴリズムが高速になるか分かるように，あえて，平方根計算を含んだ距離計算を行なった．本アルゴリズムをユークリッド距離での適用に限定し，不要な平方根計算を省いた実行時間を比較すると，K=256，$m = 32$ の条件で，実行時間は約 40%程になる．

K を変化させたときの高速 K-means 法の実行時間を確かめるために，K=64，$m = 16$ の時の実験も行なってみた．この高速 K-means 法は，K の時間のかかるアルゴリズムを，約 $m + \frac{K}{m}$の時間で処理するアルゴリズムである．そのため，K が小さいときには処理時間の減少率も少ないはずである．実際に実験を行ってみた結果，この時の計算量は通常の 40%程であった．通常の K-means 法の 2 分の 1 以下の計算量になってはいるが，K=256 の時と比べると K が小さいため効率は低下している．

量子化精度，PSNR を表した図 2.11を見てみると，本手法（Original）は頻度法（Frequency

），MCA，渡辺（WCA）のアルゴリズムのどれよりも実験に用いた全データで高い値を示している．平均値では，表2.6のように，37dbになった．PSNRの値だけでなく，ディスプレイ上で表示した画像も，より原画像に近く表示されているように見える．

実際に，いくつかの画像についての処理結果を示す．

図2.12は，ファイル1，aerial1.ppmである．図2.13は，図2.12を4種類のアルゴリズムで量子化した画像である．図2.13（a）の頻度法では，いくつかの小さな点の色値が大きく違っている（大きく色の違う点が固まっていないので判別は難しい）．図2.13（b）のMCAでは，中央の赤い色の部分の色がうまく量子化されていない．図2.13（c）の渡辺のアルゴリズムでは，ほとんど原画像と同じに見えるが，若干色が暗くなっている．図2.13（d）の本手法では，原画像との違いはほとんどわからない．この画像の場合，4つのアルゴリズムの違いはそれほどはっきり出てこないが，PSNRでは31.3，29.8，32.8，36.2（それぞれ頻度法，MCA，渡辺のアルゴリズム，本手法を表す）となっており，明らかに本手法における量子化が優れている．

図2.14は，ファイル9，girl1.ppmである．図2.15は，図2.14を4種類のアルゴリズムで量子化した画像である．図2.15（a）の頻度法では，画像の右側（窓の外）の色の再現性が低い．また，ハイライト部分の色（人物の髪の毛など）が暗くなっているので，コントラストが悪く感じられる．これは，出現頻度の低い明るい色に代表色が割り当てられなかったからである．図2.15（b）のMCAでは，肌の色，窓の外等，明らかに原画像の色と掛け離れた色が割り当てられている部分がある．図2.15（c）の渡辺のアルゴリズムでは，若干コントラストが悪く感じられるほかは，ほとんど原画像と同じに見える．図2.15（d）の本手法では，やはり原画像との違いはほとんどわからない．

図2.16は，ファイル12，mandrill.ppmである．図2.17は，図2.16を4種類のアルゴリズムで量子化した画像である．図2.17（a）の頻度法では，多くの部分で不適切な色が出現している．特に目立つのは，右目，眉間，鼻の頭等のハイライトである．この画像のように，使われている色の多い画像では，頻度法では出現頻度の低い色には極めて不適切な色が割り当てられることがある．図2.17（b）のMCAでは，鼻の頭のハイライト部分と，画像左端，中央より若干下の黄色い部分が，不適切な色として目立つ．図2.17（c）の渡辺の

アルゴリズムでは，青系の色の割り当ての少なさから，マッハ効果のような不自然さが多少出ている．図2.17（d）の本手法でも，赤系の色で若干違いはあるが，原画像との違いはわかりにくい．

他の画像に対しても実際に表示させた結果を見たが，本手法ならば原画像との違いがはっきりとわかるようなことはなかった．PSNRの値どおり，全画像で高い再現性があった．

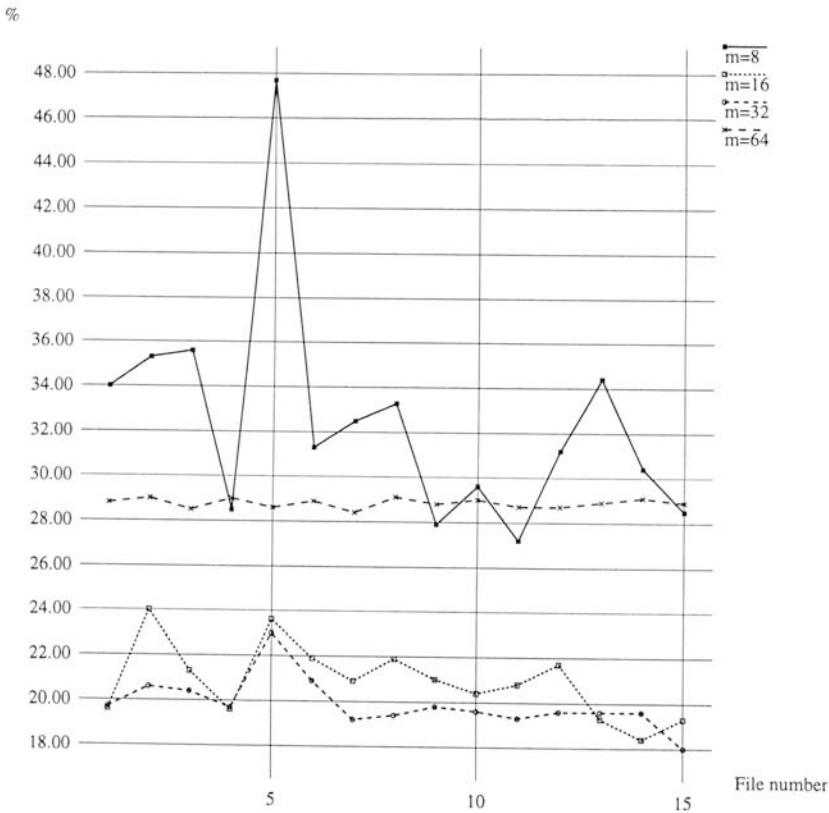


図 2.9: 通常の K-means 法と比較しての距離計算の回数（%）

表 2.2: 通常の K-means 法と比較しての距離計算の回数の平均（%）

	m=8	m=16	m=32	m=64
平均	32.5	20.9	19.9	28.8

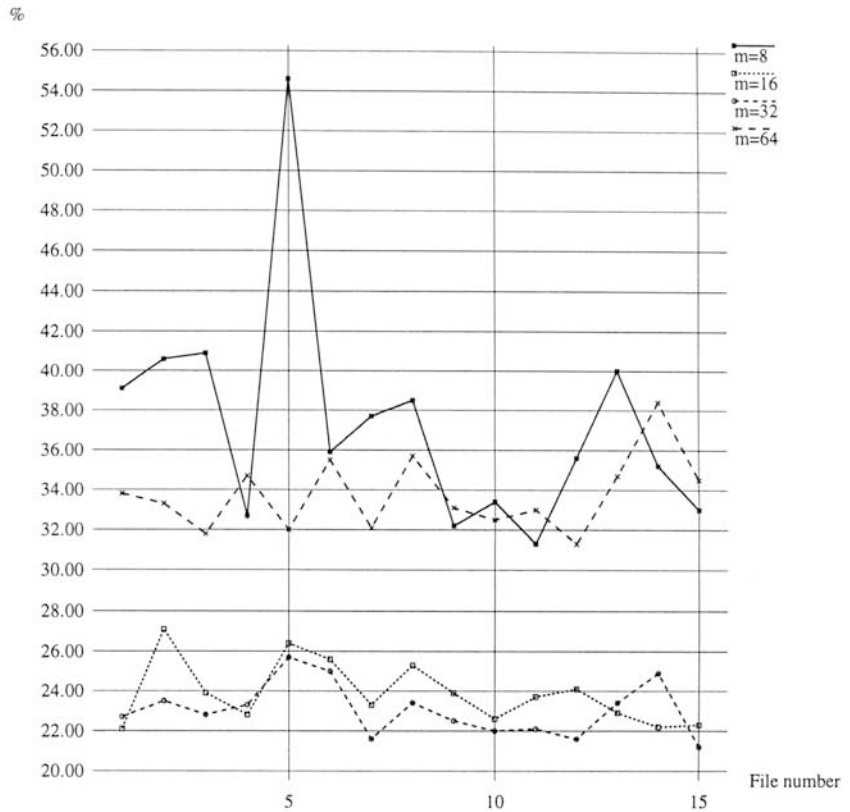


図 2.10: 通常の K-means 法と比較しての実行時間（%）

表 2.3: 通常の K-means 法と比較しての実行時間の平均（%）

	m=8	m=16	m=32	m=64
平均	37.4	23.9	23.0	33.8

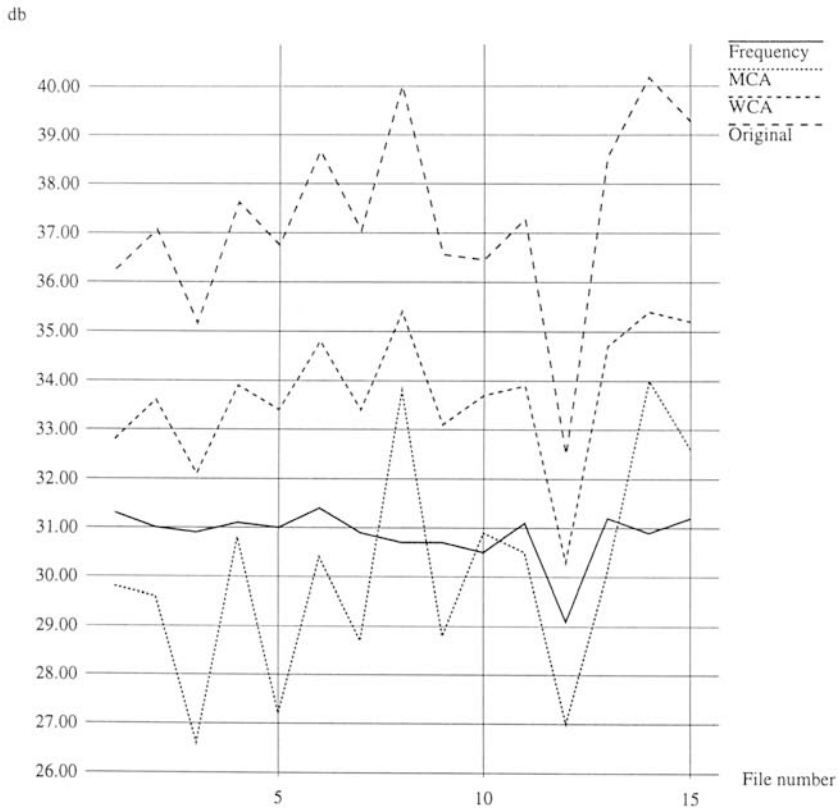
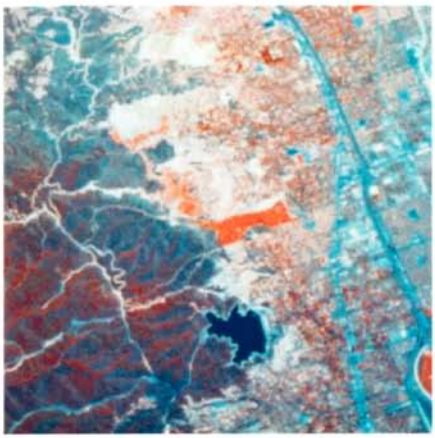


図 2.11: PSNR(db)

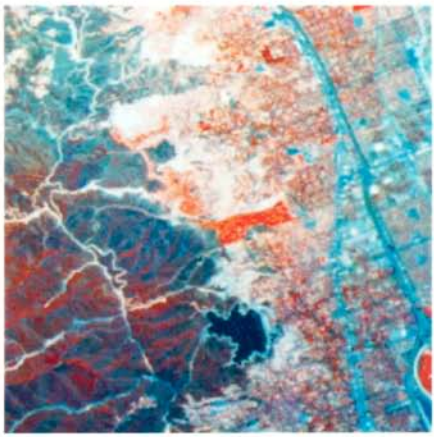
表 2.4: PSNR の平均 (db)

	頻度法	MCA	渡辺	本手法
平均	30.9	30.1	33.7	37.4

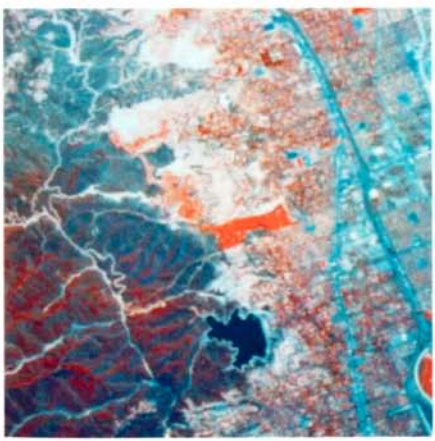
図 2.12: 原画像（aerial1.ppm）



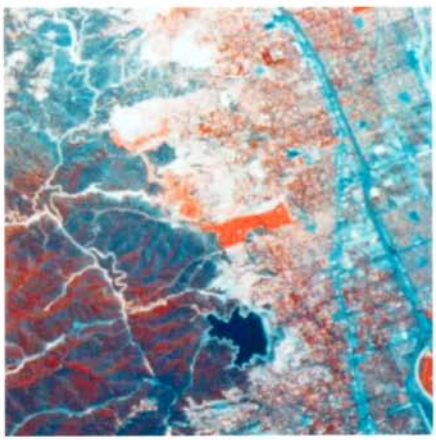
(a) 頻度法



(b)MCA



(c) 渡辺のアルゴリズム



(d) 本手法

図 2.13: 実験結果 1：各アルゴリズムごとに量子化した画像

図 2.14: 原画像（girl1.ppm）

(a) 頻度法

(b)MCA

(c) 渡辺のアルゴリズム

(d) 本手法

図 2.15: 実験結果 2：各アルゴリズムごとに量子化した画像

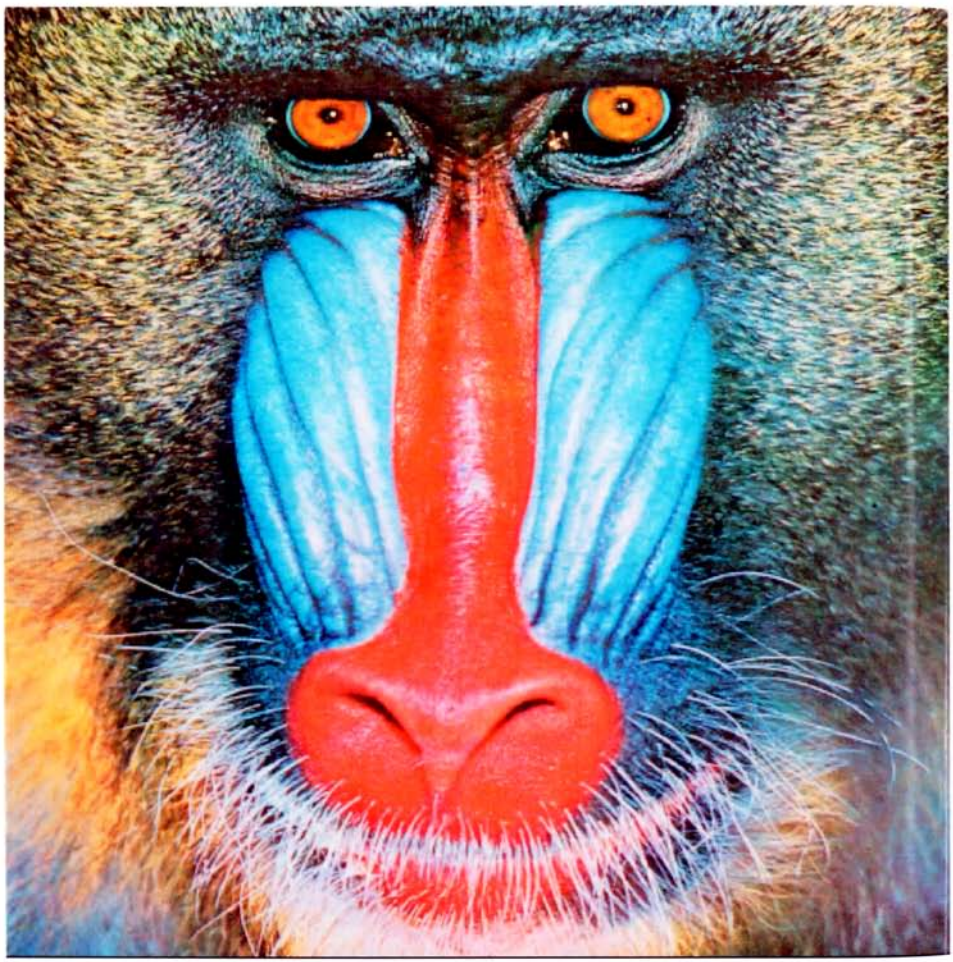
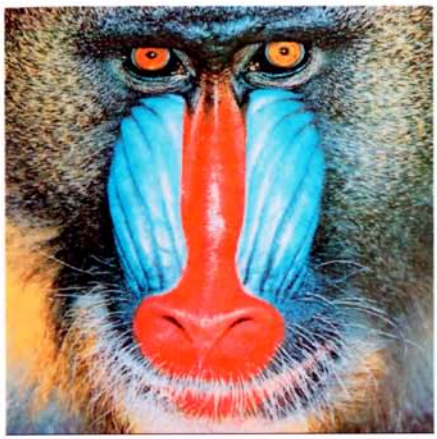


図 2.16: 原画像（mandrill.ppm）



(a) 頻度法

(b)MCA

(c) 渡辺のアルゴリズム

(d) 本手法

図 2.17: 実験結果 3：各アルゴリズムごとに量子化した画像

2.7 結言

すべての画像ファイルで本手法は最も良い結果を出した．これは，K-means 法が与えられた初期点より求まる局所最適解を求めるアルゴリズムであるからである．ただし，K-means 法の処理時間は早いとはいえず，リアルタイムで処理を行なわなければならないようなシステムには向いていない．しかし，本章で提案した高速 K-means 法ならば通常の K-means 法と同様な結果が得られ，K=256 といった場合で処理時間も 4 分の 1 以下にすることができる．

しかし，高速 K-means 法でも通常の K-means 法と同じ問題点が存在する．それは，初期点によって求まる解が大きく違うということである．本手法で用いた初期点の選定法は，Median Cut Algorithm や渡辺のアルゴリズムと同じような考え方のアルゴリズムである．そのため，従来の Median Cut Algorithm，渡辺のアルゴリズムで求まる値を初期点として用いて K-means 法をおこなってもそれほど差がでない．今後の課題として，より適切な初期点を求めることが考えられる．

また，高速化されたとはいえ，MCA や渡辺のアルゴリズム等と比較すると，まだ処理に時間がかかる．K-means 法を高速化させるにあたっては，繰り返し回数を制限することも考えられる．表 2.5は，本手法によるおおよその実行時間（Axil320 上で複数回試行し，その平均をとった）と，本手法の STEP3 から STEP8 までの繰り返しの回数である．この表 2.5より，ファイルによって繰り返しの回数が大きく違うことがわかる．本章のアルゴリズムの紹介では，K-means 法は完全に解が収束してから終了するようになっている．しかし，実行時間は繰り返しの回数に比例するので，繰り返しの回数が大きいものは非常に多くの実行時間を必要とする．K-means 法は繰り返しによって解を徐々に局所改良していくアルゴリズムであるが，その局所改良は，最初の数回は有効であるが，後半になるにつれて徐々に改良率が落ちてくる．そのため，ある程度解が収束したら，完全に収束しなくてもアルゴリズムを終了させることが考えられる．ファイル 12，mandrill.ppm を用いて，繰り返しを一定数でやめた場合の PSNR を調べたのが図 2.18である．これより，局所改良の有効性は，繰り返しの 10 回以降には落ちてくることが分かる．また，今回実験に用いた画

表 2.5: 実行時間と繰り返し数の関係

	ファイル名	実行時間 (秒)	繰り返し数
1	aerial1.ppm	4	10
2	auto1.ppm	9	10
3	bear1.ppm	21	21
4	city1.ppm	5	8
5	city2.ppm	16	17
6	couple.ppm	3	8
7	desk1.ppm	15	21
8	face1.ppm	4	7
9	girl1.ppm	9	17
10	home3.ppm	8	10
11	home4.ppm	9	13
12	mandrill.ppm	64	43
13	milkdrop.ppm	4	10
14	room1.ppm	3	5
15	room7.ppm	6	12

像は，ほとんどが 10 回前後で収束している．ごく一部に 20 回以上の繰り返しを必要としたものがあるが，繰り返しの後半では K-means 法の局所改良があまり有効でなくなることが実験より分かっている．よって，実行時間の短縮のために，最大繰り返し数を設定する方法は有効であると考えられる．

本手法の高速 K-means 法は，ユークリッド距離以外の距離を評価基準に使うことができる．例えば，ユークリッド距離は $d_2(x,y) \equiv (\sum_{i=1}^{n} |x_i - y_i|^2)^{\frac{1}{2}}$ で表されるが，$d_1(x,y) \equiv \sum_{i=1}^{n} |x_i - y_i|$ で表される様な距離でも高速 K-means 法は利用できる．実際にこの距離 d_1 を利用した時には，距離計算が簡単になるため処理が高速になる．

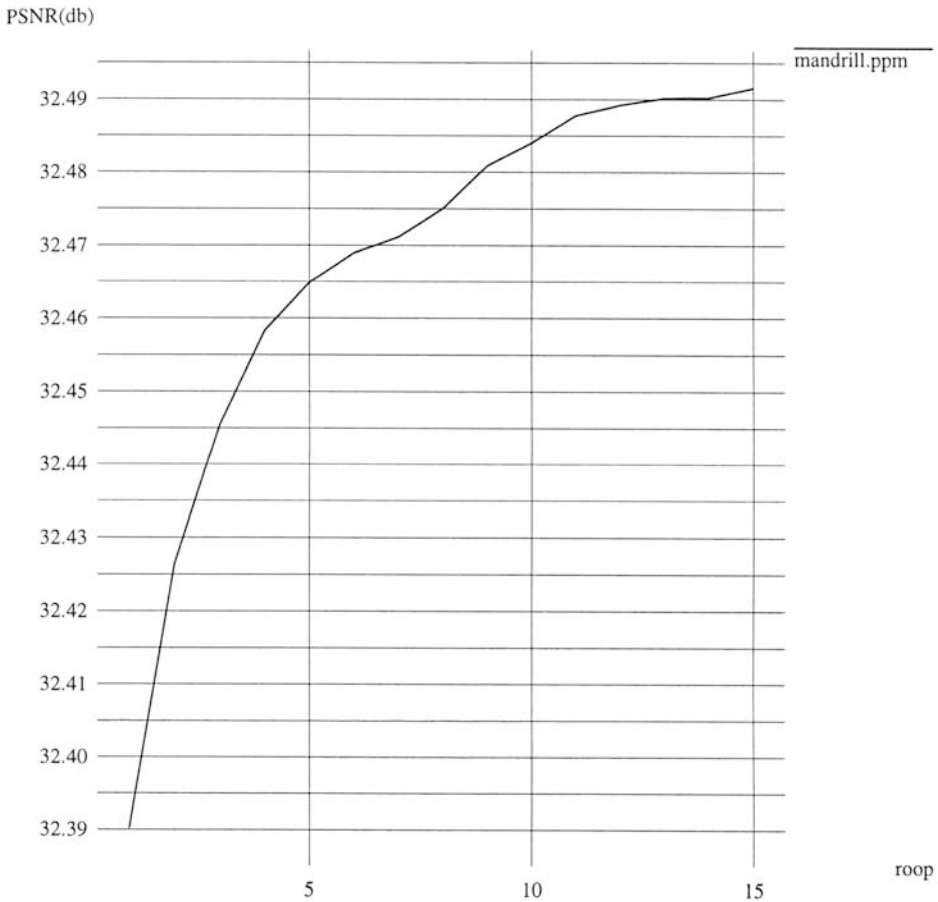


図 2.18: 繰り返し数と PSNR の関係

第 3 章

カラー文書画像からの文字抽出

3.1 序言

近年，多くの光学的文字読み取り装置（OCR）が研究されてきた．その結果，印刷物からの文字認識では，非常に高い精度の認識が行なわれるようになってきている．だが，そのシステムのほとんどは 2 値（白黒）の印刷物を対象にしたものである．現在出版されている印刷物の中には，多彩な色を持ち，複雑な背景の上に書かれた，雑誌のような出版物も存在する．そのような雑誌等のカラー印刷物から文字認識を行なう場合，白地に黒で印刷されているだけの印刷物と違い，背景の中から文字色を見つけ出す作業が必要となる．本研究は文字列が単色で書かれていることを前提として，色情報をクラスタリングすることによって画像の背景色と文字色を分離し，どのような背景からでも文字を抽出できることを目的としている．

カラー画像の領域分割に関しては，過去にいくつかの研究がある [8][9][10]．また，情景画像からの文字抽出に関しても，文献 [11][12] のような研究がある．しかし，文献 [11] はカラー画像の色の情報を利用しておらず，文献 [12] は限定された状況（画像中に文字列が 1 列のみ）の研究である．文字色を推定し，それを利用して文字を抽出する手法としては，文献 [13][14] が存在する．文献 [13] は，色ヒストグラムから K 平均法 (K-means 法）によるクラスタリングによって色を分解しており，本章で提案する手法に近いアルゴリズムになっている．しかし，本章で提案する手法は，ファジイクラスタリングを用いた手法であり，要素の所属が 2 値的（所属するか，しないか）に決まってしまうハードクラスタリン

グと異なり，所属の程度まで判別できるクラスタリングを使用している．このファジイクラスタリングによって得られる帰属度を用いることによって，2 値的な分類だけでは判別の難しかった画像についての微妙な色の識別ができるようになる．そして，色ごとに分解された画像 (2 値画像) から，文字パターンを抽出する手法についても述べる．

本手法では，2 値化された色分解画像をラベリングし，そのラベルの外接矩形から文字の並びと思われる特徴を見つけ出している．本手法は，同じ文字列内の各文字の色が同一で単色であれば，複数行で複数色の文字列があっても抽出可能である．また，様々なサイズの文字が混じっていても抽出が可能である．本手法は，私の所属した研究室の修士の学生の過去の研究である，文献 [15][16] が元になっている．

3.2 処理の概要

大まかな流れとしては，色分解画像を作成し，その 2 値化したデータをラベリングし，そのラベルを囲む外接矩形より文字列を抽出する，といった流れになっている．本手法の特徴は，色分解画像を作成するにあたり，入力画像の色情報に対してファジイクラスタリングを用いて色分割を行なっていることである．

細かな処理の流れは以下の通りである．

1. スムージングによるディザの除去
2. 色値の RGB から $L^*u^*v^*$ への変換とヒストグラム作成
3. 色情報のファジイクラスタリング
4. 帰属度を元に色分解画像（2 値画像）作成
5. 2 値画像のノイズ除去
6. 黒画素および白画素のラベリング
7. 文字抽出に適した 2 値画像の選択

8. 文字行の抽出

それぞれの処理における細かなアルゴリズムについては，次節より説明する．

3.3 スムージングによるディザの除去

3色もしくは4色の色を周期的に配置することで，見た目にはそれらの色を混ぜ合わせたような効果を期待するのがディザ法である．ディザ法によって印刷されている文字があると，図3.1（a）のように文字の単色性が満たされにくい．文献[13]では，スムージングを行なうことによって，ディザの影響を軽減する方法が提案されている．そこでは，文字の輪郭がぼやけてしまわないように，画像のエッジの弱い部分のみを平均化する手法を提案している．

本手法では，エッジであるかどうかの判定に，次の式を用いている．

$$d_i max = max(\|x_j - x_k\|) \tag{3.1}$$

$$j,k \in N = \{\text{座標 } i \text{ とその8近傍の座標}\}$$

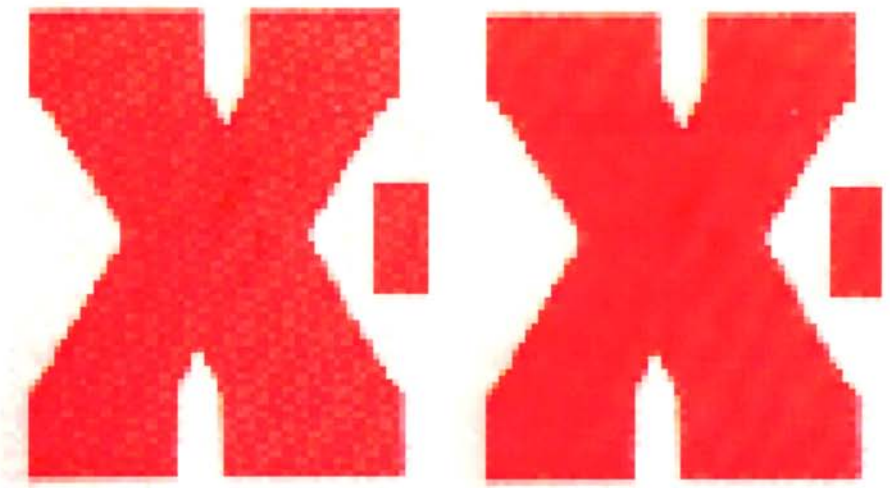
なお，x_iは座標 i の色値を表す

式（3.1）は，目標画素とその8近傍の画素の計9つの画素の色値のうち，最も離れた2つの色値の距離である．$d_i max$ が一定値以内ならば，エッジではないと考えて，式（3.2）で表されるフィルタで8近傍の画素との平均化を行なう．

$$x_i = \frac{\sum_{j \in N} x_j}{9} \tag{3.2}$$

$d_i max$ が一定値以上ならば，エッジを含んでいると考えて平均化を行なわない．

このようなスムージング処理を行なうことによって，図3.1（a）が図3.1（b）のようになる．



(a) スムージング前　　(b) スムージング後

図 3.1: スムージング処理

3.4 色値のRGBから$L^*u^*v^*$への変換とヒストグラム作成

CIEの$L^*u^*v^*$色空間は，人間の色感覚に近いとされる均等知覚色空間である．そこで，RGBで表される色値を$L^*u^*v^*$に変換する．変換式は以下の通りである．

$$X = 0.478R + 0.299G + 0.175B \tag{3.3}$$

$$Y = 0.263R + 0.655G + 0.051B \tag{3.4}$$

$$Z = 0.020R + 0.160G + 0.908B \tag{3.5}$$

$$L^* = 25(\frac{100Y}{Y_0})^{\frac{1}{3}} - 16 \tag{3.6}$$

$$u^* = 13L(u' - u'_0) \tag{3.7}$$

$$v^* = 13L(v' - v'_0) \tag{3.8}$$

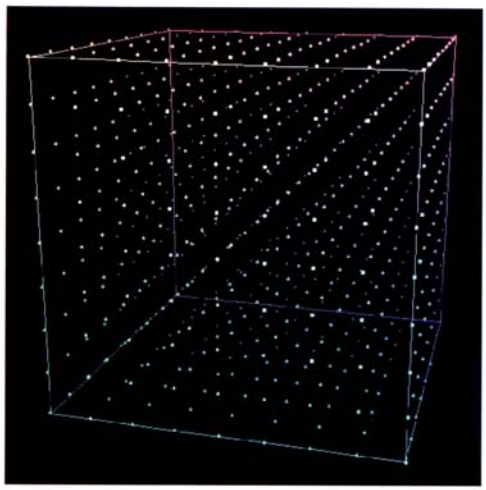
$$u' = \frac{4X}{(X + 15Y + 3Z)} \tag{3.9}$$

$$v' = \frac{9Y}{(X + 15Y + 3Z)} \tag{3.10}$$

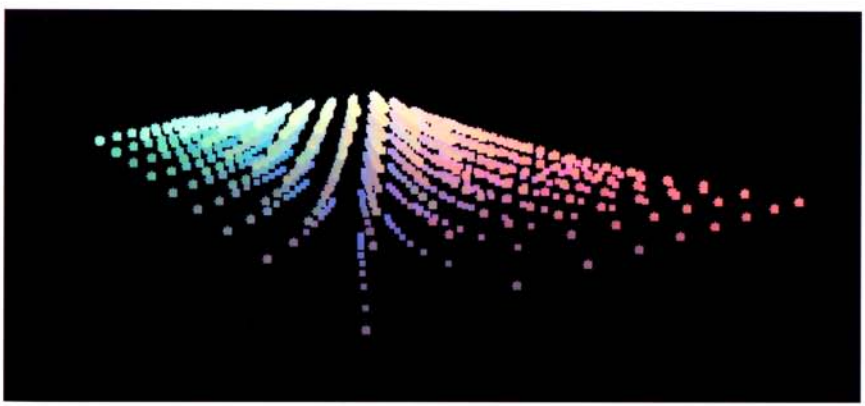
なお，RGBは本来0～255の値を持つデータだが，それを0～1の値に変換したものを上記の式に適用する．また，$Y_0 = 1, u_0 = 0.201, v_0 = 0.461$となっている．

RGBからXYZへの変換式についてはいくつかのバリエーションが存在するが，本研究では上記の式を用いている．なお，この$L^*u^*v^*$色空間は立方体をしておらず，L^*, u^*, v^*のとる値の範囲もそれぞれ違っている．イメージとしては図3.2（b）のような形状をしている．ちなみに，L^*は0～116の値をとり，u^*は-97～171の値をとり，v^*は-128～109の値をとる．

このようにして得られた色値からヒストグラムを作成し，クラスタリングを行なうが，本システムでは文字がすべて横書きで書かれているものとして考え，その位置情報も利用してクラスタリングを行なう．そこで，$L^*u^*v^*$の3次元だけでなく，画像の垂直方向の座標（y座標）も加えた4次元の空間に対してクラスタリングを行なう．この4次元の空間は



(a)RGB 色空間



(b)$L^*u^*v^*$色空間

図 3.2: 色空間の形状

$17 \times 45 \times 40 \times 10$（$L^*$が17, u^*が45, v^*が40, y座標が10）に分けられ，ヒストグラムが作成される．

3.5 色情報のファジイクラスタリング

入力された画像の色情報に対してファジイクラスタリングを行ない，色を分解する．ファジイクラスタリングが，ハードクラスタリングといわれる通常のクラスタリングと異なる点は，要素が複数のクラスタに少しずつ所属する事を認めている点である．ハードクラスタリングにおける，要素のクラスタへの所属の有無を1と0で表すと，図3.3のようになる．これがファジイクラスタリングでは，各クラスタ中心までの距離の比で決まる0〜1の値の帰属度という数値で表される（図3.4参照）．帰属度は，要素にそのクラスタがどれだけ強い影響を与えているかを示している．これにより，あるクラスタに所属するかしないかといった2値的な判別ではなく，どのクラスタにどの程度所属しているかといった程度の違いまで判断できるようになる．

3.5.1 FCM

代表的なファジイクラスタリングとしては，Bezdekらによるファジイc-means法（FCM）[17]というアルゴリズムが存在する．これは，第2章でも述べたK-means法に帰属度の考えを付け加えて拡張したアルゴリズムである．このFCMのアルゴリズムを簡単に述べると次のようになっている．

STEP1: c個の初期クラスタ中心$v_i, (i = 1, 2, ..., c)$を適当に決める

STEP2: すべての要素$x_k, (k = 1, ..., n)$の帰属度

$$u_{ik} = \frac{1}{\sum_{j=1}^{c} \left(\frac{\|x_k - v_i\|}{\|x_k - v_j\|} \right)^{2/(m-1)}}, (1 \leq i \leq c, 1 \leq k \leq n) \tag{3.11}$$

を求める

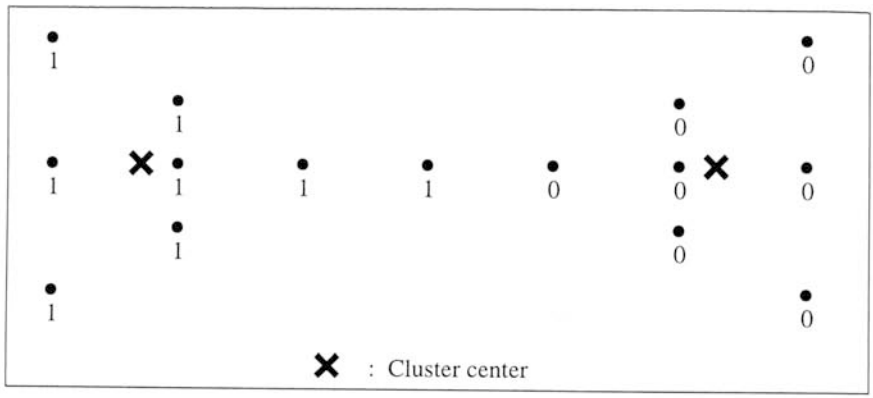


図 3.3: ハードクラスタリングの結果

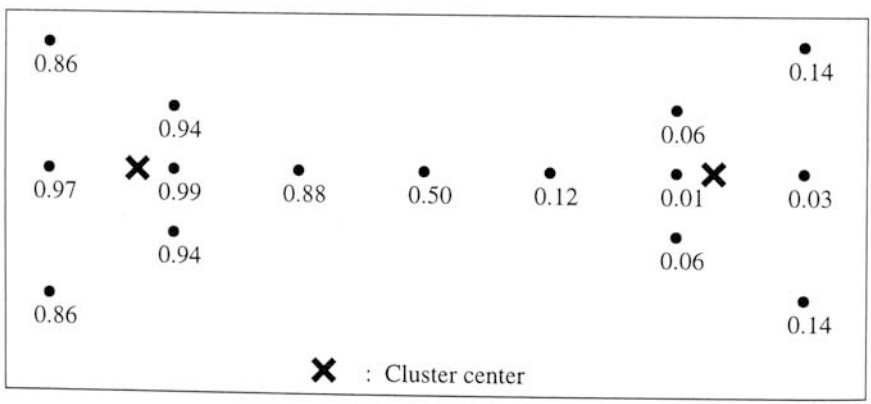


図 3.4: ファジイクラスタリングの結果

STEP3: 新たなクラスタ中心

$$v_i = \frac{\sum_{k=1}^{n} (u_{ik})^m x_k}{\sum_{k=1}^{n} (u_{ik})^m}, (1 \leq i \leq c) \tag{3.12}$$

を計算する.

STEP4: 前回の繰り返しにおけるクラスタ中心を$\hat{v}_i$としたとき,
すべてのiに対して$\|\hat{v}_i - v_i\| \leq \epsilon$が満たされれば終了する.
さもなくばSTEP2に戻る.

ここでのmは帰属度に対する重み値で,$1 < m < \infty$の値をとり,mが大きくなればなるほど,帰属度の大きな要素x_kのクラスタ中心v_iに対する影響が大きくなる.なお,式(3.11),(3.12)で,$m \to 1, u_{ik} \in \{0, 1\}$と置けば,このアルゴリズムは通常のK-means法と同じものとなる.

3.5.2 自己収束型ファジイクラスタリング

本研究では,最も一般的なファジイクラスタリングであるFCMにクラスタの分割,統合,消滅といった処理を加えた,自己収束型ファジイクラスタリングを使用している.この自己収束型ファジイクラスタリングのアルゴリズムは以下のようになっている.

STEP1: クラスタ数の初期値c,帰属度の重みm,収束判定値ϵ,最大繰り返し数I,クラスタ分割条件の閾値θ_S,クラスタ統合の閾値θ_d,クラスタ消滅条件の閾値θ_cを決める.

STEP2: 初期クラスタ中心$v_i, (i = 1, 2, ..., c)$を決める.

STEP3: 要素$x_k = (x_{kL}, x_{ku}, x_{kv}, x_{ky}), (k = 1, ..., n)$の帰属度

$$u_{ik} = \frac{1}{\sum_{j=1}^{c} \left(\frac{\|x_k - v_i\|}{\|x_k - v_j\|} \right)^{2/(m-1)}}, (1 \leq i \leq c, 1 \leq k \leq n) \tag{3.13}$$

を求める.

STEP4: 各クラスタの分散

$$\sigma_i = \frac{\sum_{k=1}^{n} (u_{ik})^m \|x_k - v_i\|^2}{\sum_{k=1}^{n} (u_{ik})^m}, (1 \leq i \leq c) \tag{3.14}$$

を求め，ソートする．

STEP5: σ_iがθ_S以上のクラスタを分割する．なお，分割後の新しいクラスタ中心は，

$$v_1 = \frac{\sum_{x_k \in S_1} (u_{ik})^m x_k}{\sum_{x_k \in S_1} (u_{ik})^m}, v_2 = \frac{\sum_{x_k \in S_2} (u_{ik})^m x_k}{\sum_{x_k \in S_2} (u_{ik})^m}, \tag{3.15}$$

$(S_1 = \{x_k | x_{kX} > v_{iX}\}, S_2 = \{x_k | x_{kX} \leq v_{iX}\}, X$は$L, u, v, y$のうち最も分散の大きいもの)

STEP6: クラスタ間の距離がθ_d以下のクラスタを統合する．なお，統合後の新たなクラスタ中心は，２つのクラスタ中心の中間点である．クラスタを統合したならば，すべての要素の帰属度を求め直す．

STEP7: 各クラスタの所属要素数（重みつき帰属度合計）

$$\rho_i = \sum_{k=1}^{n} (u_{ik})^m, (1 \leq i \leq c) \tag{3.16}$$

がθ_c以下のクラスタを消滅させ，再度すべての要素の帰属度を計算する．

STEP8: 新たなクラスタ中心

$$v_i = \frac{\sum_{k=1}^{n} (u_{ik})^m x_k}{\sum_{k=1}^{n} (u_{ik})^m}, (1 \leq i \leq c) \tag{3.17}$$

を計算する．

STEP9: 前回の繰り返しにおけるクラスタ中心を$\hat{v}_i$としたとき，
すべてのiに対して$\|\hat{v}_i - v_i\| \leq \epsilon$が満たされるか，繰り返し数が$I$であるならば終了する．さもなくばSTEP4 に戻る．

クラスタ数が増減し，自己収束するアルゴリズムは，K-means 法を元にしたものではいくつも存在する．本手法は，ハードクラスタリングで使われる自己収束化手法を，ファジイクラスタリングに拡張したものである．本来のファジイクラスタリングと異なる点は，STEP4，STEP5 のクラスタ分割処理，STEP6 のクラスタ統合処理，そして，STEP7 のクラスタ消滅処理である．クラスタ分割処理は，クラスタの分散の大きさを見て行われる．分散の大きなクラスタは 2 つに分割される．次に，クラスタ同士が接近していた場合（クラスタ中心間の距離が小さかった場合），その 2 つのクラスタを 1 つにする．そして，このクラスタ数の変化などによって，所属する要素の数（重みつき帰属度合計）が少なくなったクラスタは消滅させる．

ただし，このようなクラスタ数決定アルゴリズムは，解が振動してしまい，収束しないことがある．そこで，解の振動を押さえるために，繰り返しが進むにつれて閾値や収束条件を緩めることが考えられる．

このような処理を加えた，適切なクラスタ数で自己収束するファジイクラスタリングを用いて，各クラスタおよび帰属度が計算される．

なお，実験結果のところで示してあるデータは以下の初期値，閾値で計算された結果である．

$$
\begin{aligned}
c &= 1 \\
m &= 1.5 \\
I &= 10 \\
\epsilon &= 1.0 \\
\theta_S &= 8.0 \\
\theta_d &= 3.0 \\
\theta_c &= \text{全画素数}/100
\end{aligned}
$$

なお閾値に関しては，解が振動することが多いことから，繰り返しが進むにつれて閾値を緩める手法を用いた．この値は，あくまで実験的に求めた値である．

3.6 帰属度を元に色分解画像（2 値画像）作成

ファジイクラスタリングによって得られる各色の帰属度から，そのクラスタへの所属の程度が分かる．その帰属度を元に，2 値画像を作成する．その際の閾値 t_i は，式（3.16）そのクラスタに所属する要素の数 ρ_i に応じて次の式で決定する．

$$t_i = 0.7\rho_i / \sum_k \rho_k + 0.3 \tag{3.18}$$

所属する要素の数が多いクラスタは画像の広い範囲で使われる背景の色であると考えられ，なるべく文字部分を含まない 2 値画像を作成するために閾値を高くする．逆に，所属する要素の数が少ないクラスタは文字色である可能性が高く，色のにじみも含めて抽出するために閾値は低くする．

なお，この 2 値化処理の際には，y 座標（画素の垂直方向の座標）が違うのみで L^*，u^*，v^* の値がほぼ同じクラスタに対しては，それぞれの帰属度を足して 1 つのものとして閾値処理を行っている．画像によっては，背景色のように広い範囲で使われている色が，y 座標で 2 つに分けられることがある．このような場合，2 つのクラスタを統合して処理しても，文字色と背景色の分離性にはほとんど影響しないので，2 値画像の数を減らし，処理の効率化を図るために，クラスタを統合処理する必要がある．

このようにして，図 3.5の画像からは図 3.6中に示してあるような 2 値画像が作成される．


The High-Performance
Computing
Continuum
Identifying Software Project Risks
The Virtual Design Team

図 3.5: 入力画像

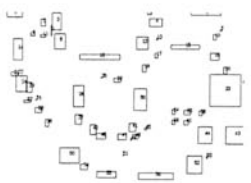
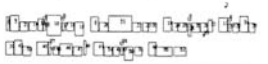
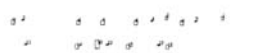
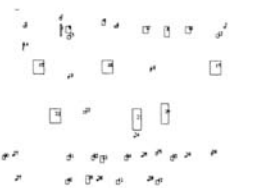
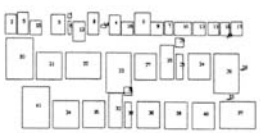
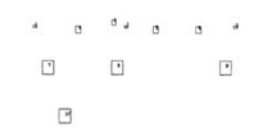
	2 値画像	黒画素外接矩形	白画素外接矩形
Cluster 1			
Cluster 2			
Cluster 3	Identifying Software Project Risks The Virtual Design Team		
Cluster 4	The High-Performance Computing Continuum Identifying Software Project Risks The Virtual Design Team		
Cluster 5	The High-Performance Computing Continuum		

図 3.6: 色分解画像

3.7　2 値画像のノイズ除去

作成された 2 値画像には，スムージング処理で吸収しきれなかったディザや，背景上の孤立点等がノイズとして出現する．具体的には，画素の連結数が少ないものである．画像の大きさによって，いくつの連結数のものまでをノイズとするかは変わる．これらは，次のラベリング処理やそれに続く文字行の抽出で，メモリや処理速度の効率化の妨げになるので除去する．

3.8　黒画素および白画素のラベリング

ラベリングとは，同じ連結成分に属するすべての画素に同じラベル (番号) を割り当て，異なった連結成分には異なったラベルを割り当てる操作である．2 値化された画像の黒画素，白画素両方に対してラベリングを行ない，そのラベルに対して外接矩形を求める．図 3.7のような 2 値画像には，図 3.8 (a) (b) のようにラベルが付けられる．ここでは 4 連結でラベリングを行なっている．そして，その 1 つのラベルをつけられた連結成分を囲う矩形（外接矩形）を元に文字の抽出を行なう．図 3.9 (a) (b) が図 3.8の黒画素，白画素のラベルに対してのそれぞれの外接矩形となる．この時，大き過ぎる（入力画像の 4 分の 1 以上の大きさ）あるいは小さ過ぎる矩形（幅と高さの両方が 3 画素以下），縦横比が大きく違う（幅/高さが 15 以上，または 15 分の 1 以下）矩形は文字矩形ではないとして除外する．

図 3.5の画像から得られる外接矩形は，図 3.6のようになっている．

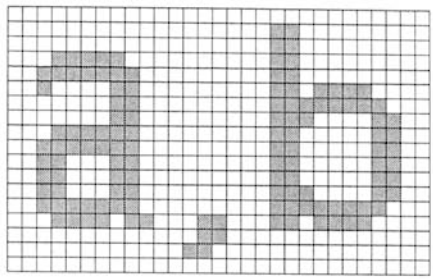
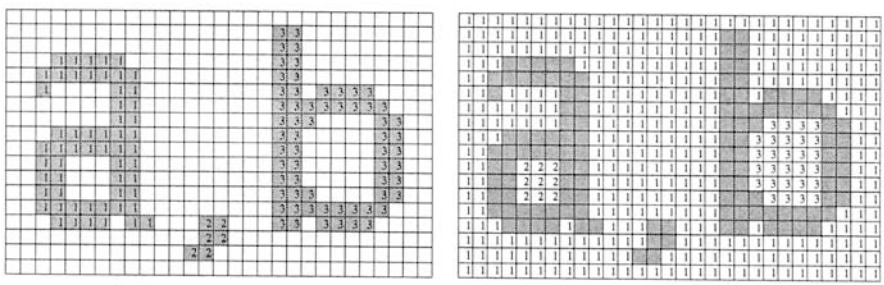
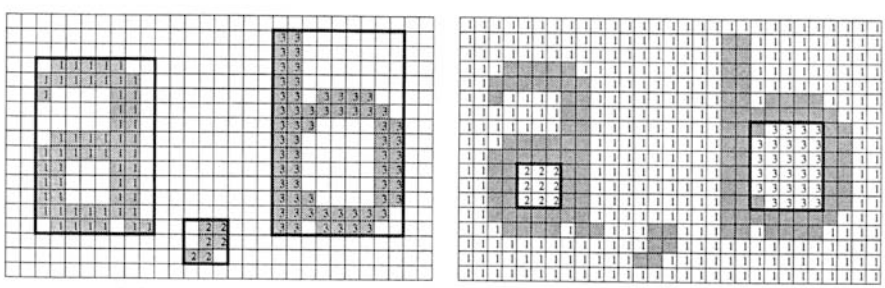


図 3.7: 2 値画像



(a) 黒画素のラベル　　(b) 白画素のラベル

図 3.8: ラベリング



(a) 黒画素外接矩形　　(b) 白画素外接矩形

図 3.9: 外接矩形

3.9 文字抽出に適した2値画像の選択

以後の処理の高速化とメモリの節約のため文字抽出に適した画像を選ぶ．その際には次の4点を考慮する．

- 矩形の分散
- 矩形数
- 画素密度
- 平均矩形サイズ

文字行は同じくらいの大きさの矩形からできているはずなので，各2値画像中の矩形の幅と高さの分散を計算し，その値が大き過ぎる画像については文字行を含んでいないと考え，以後の処理を行なわない．また，矩形数が多過ぎるものも，文字色を含んだ画像である可能性が低いので棄却する．各矩形ごとの黒及び白の画素密度が高過ぎたり低過ぎたりするものも，文字ではない可能性が高いので棄却する．なお，画素密度 $density$ とは

$$
\begin{aligned}
density &= \frac{N}{w+h} \qquad (3.19)\\
N &: \text{ラベルのついた画素数}\\
w &: \text{矩形の幅}\\
h &: \text{矩形の高さ}
\end{aligned}
$$

で表される値である．また，平均矩形サイズが小さ過ぎるものも文字を含んでいないとして棄却する．

3.10 文字行の抽出

文字矩形が含まれていると思われる2値画像が求まったら，文字列としての特徴を持った矩形を抽出するために，各矩形ごとに隣接する（対象矩形から一定距離内にある）矩形

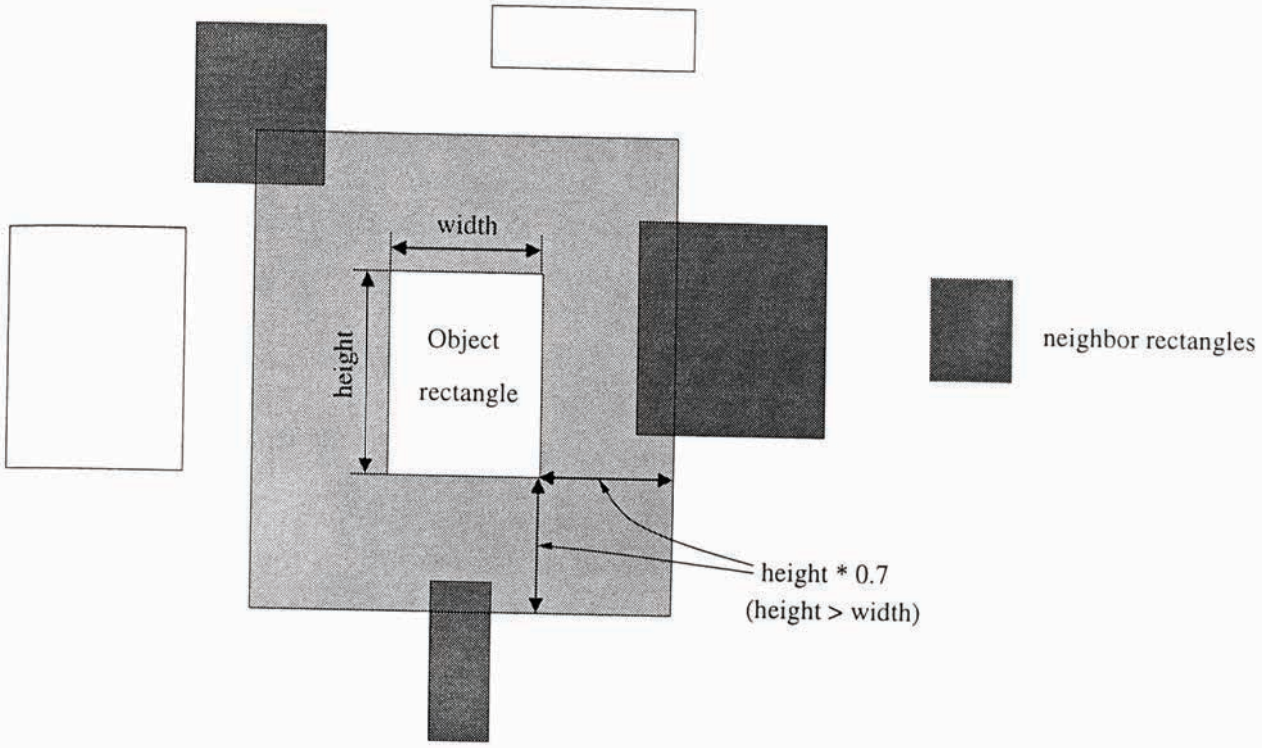


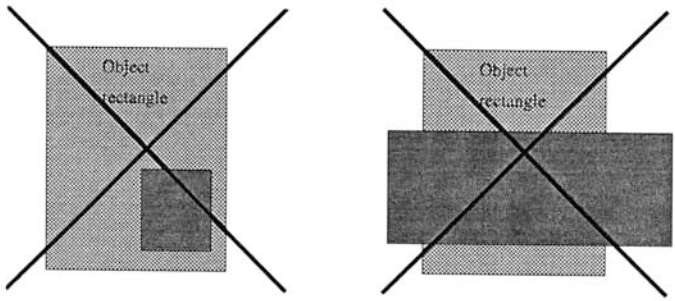
図 3.10: 隣接矩形

を求め，それをもとに矩形の連結（文字行として推定されるもの）を求める．この隣接矩形は，対象矩形から見て次の条件を満たすものである．

- 対象矩形をその高さ（もしくは幅のどちらか大きい方）の 0.7 倍，上下左右に拡大した範囲に，一部，または全部含まれている（図 3.10参照）
- 対象矩形の中に完全に含まれてしまっていない（一部は含まれていても良い）（図 3.11（a）参照）
- 対象矩形と交差していない（図 3.11（b）参照）

この条件を満たす矩形を対象矩形の隣接矩形とし，それらを次々と連結させて矩形の連なり，連結矩形を求める．この，隣接矩形の連結条件は以下の通りである．

- 矩形の大きさが同じくらいである（幅および高さが対象矩形の 4 分の 1 以上 4 倍以下，かつ，幅もしくは高さのどちらかが対象矩形の 2 分の 1 以上 2 倍以下）



(a) 対象矩形の中に完全に含まれている　(b) 対象矩形と交差している

図 3.11: 隣接矩形の条件

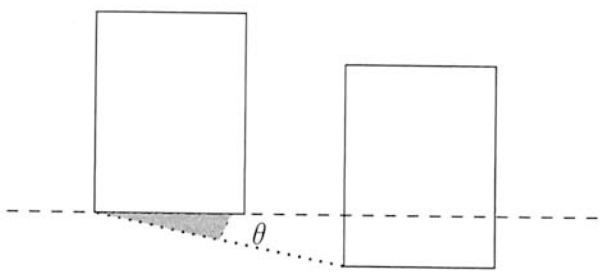


図 3.12: 矩形同士の角度

- それぞれの矩形の代表色が近い色である
- 矩形がほぼ水平にならんでいる（矩形の左下の座標同士をみて，その角度θが水平から10°以内．図3.12参照）

上記の条件を満たす矩形を次々に連結させて，連結矩形を求める．

このようにして求めた連結矩形の連結数が4以上であれば，その連結矩形は文字行の一部であるとみなす．そして，その連結矩形から文字行の幅と高さを推定する．推定される文字行の高さは，連結矩形の最上部と最下部を，もっとも高さの大きい矩形の高さの2分の1だけ広げた範囲である（図3.13参照）．幅は，先に求めた高さの範囲内で，連結矩形

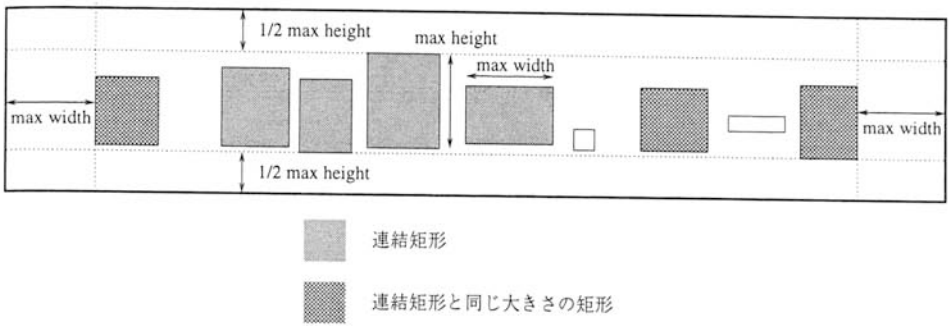


図 3.13: 文字行の推定範囲

と同じくらいの大きさで近い色の矩形を求め，そのような矩形の存在する終端を図3.13のように延長した範囲とする．推定された文字行の範囲内に矩形の中心が存在し，その矩形の色が連結矩形の色に近いならば，それを文字の要素として抽出する．

3.11 実験結果と考察

いくつかのカラー文書画像に対して実験を行なった結果，その多くでほとんどの文字を抽出できた．図3.14は図3.5から文字を抽出した結果である．他にもいくつか実験結果を示す．図3.15，3.16では，すべての文字が抽出されている．図3.17，3.18では，一部に文字以外のパターンが抽出されている．これは，背景パターンに文字色に近い色のパターンがあり，そのサイズが文字に近かったからである．このように，背景に文字色を含んだパターンが存在する場合には，完全に文字だけを抽出するのは難しい．

画像によっては，図3.19のように，白に近い色の文字色の抽出に失敗することがあった．この画像の場合，肉眼では文字色と背景色が同系色とはいえ，十分な違いが感じられるのに，色の分離が行なわれなかった．これは$L^*u^*v^*$色空間が，輝度の高い色の違いについて鈍感であるためと思われる．

他の文字抽出の失敗した画像には，画質が悪く文字の単色性が満たされ難かったもの（文字のにじみが大きく，背景との境界がはっきりしないためにうまくクラスタリングが行わ

れなかった)，文字色のディザがスムージングで吸収しきれなかったものなどがある．これらは元になる入力画像の質の問題であり，極端に品質の悪いものに対してはどうしても難しくなってしまう．また，縁取りのある文字や影付きの文字について，縁や影を文字として抽出してしまうという失敗があった．これは，矩形の画素密度を見ることによって，画素密度の小さい縁や影を文字候補からはずすことでほとんど対処できる．

難しい問題としては，文字が文字色に似た色の背景と接触しているもの（図 3.20）がある．この場合は，文字だけをくくる外接矩形が作成されないので，文字列が抽出できない．この問題は，今後も研究が必要である．

また，現在の文字抽出アルゴリズムでは，矩形の連結数が 3 以下の場合は文字列と判断されず，見出しのように，使用されている文字が少ない場合（図 3.21の「入門」の部分など）は文字として抽出されない．これは難しい問題で，2 文字程度の極めて少ない文字列の場合には，どうしても判断がつけられない．文字数の少ない文字列を抽出するための手法を考えることも今後の課題である．

3.11.1 ファジイクラスタリングの効果

ファジイクラスタリングを用いることにより，ハードクラスタリングを用いた時よりもいくつかの点で効果的に文字を抽出できるようになった．まず，全画素中でも少数しか使われていない文字色は，ハード，ファジイの両クラスタリングでも背景との分離が困難である．この時，ハードクラスタリングではその文字を抽出することはまず不可能であるが，ファジイクラスタリングでは帰属度の違いにより背景より分離することが可能になることが多い．

また，色空間の広い範囲に渡って色が使われている画像では，ハードクラスタリングは必要以上に色空間を分割してしまい，処理効率の悪化や背景パターンの誤抽出が起こりやすくなっていたが，ファジイクラスタリングではそれらは改善されている．逆に，使われている色数が少なく偏っていた場合には，ファジイクラスタリングの方が多くクラスタを生成する傾向がある．しかし，ファジイクラスタリングでは極端にクラスタ数が多くなることはなかったので，処理効率の点でも十分有用性があると言える．

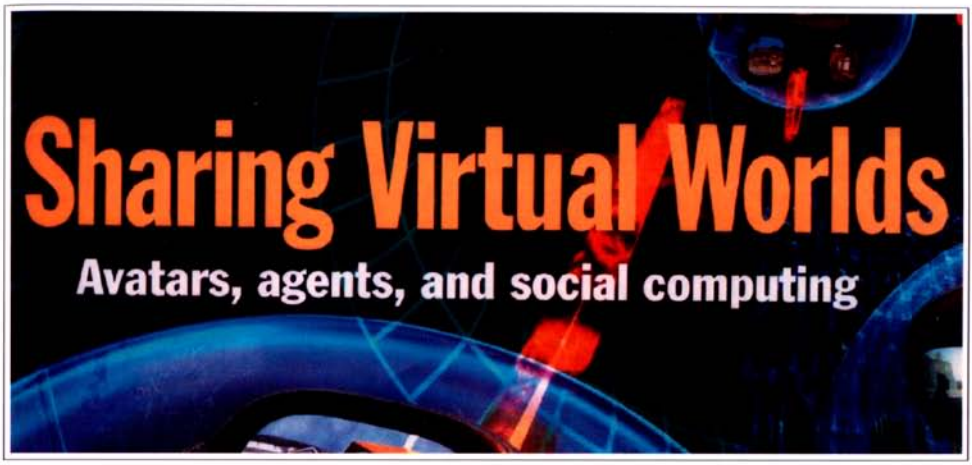
3.12 結言

本手法による文字抽出は，背景に複雑で多様な色が使われている時に特に有効である．シンプルな背景（模様がなく，単一色）の場合には，ハードクラスタリングや，従来の他の手法を用いても文字抽出が容易であり，メモリや処理速度の効率の点で本手法よりすぐれているものは多い．

本手法では，色ヒストグラムと画像の垂直座標を元にクラスタリングを行ない色分解画像を作成したが，本手法だけでは背景と文字色を完全に分離することは困難な場合もある．また，今後は横書きの文字だけでなく，縦書きの文字にも対応させていくことが考えられるので，その場合には本手法のような垂直座標だけではうまく処理できない．そのような場合には，画像中の位置情報を現在以上にうまく利用したクラスタリングを考えることが必要である．


The High-Performance
Computing
Continuum
Identifying Software Project Risks
The Virtual Design Team

図 3.14: 文字抽出結果

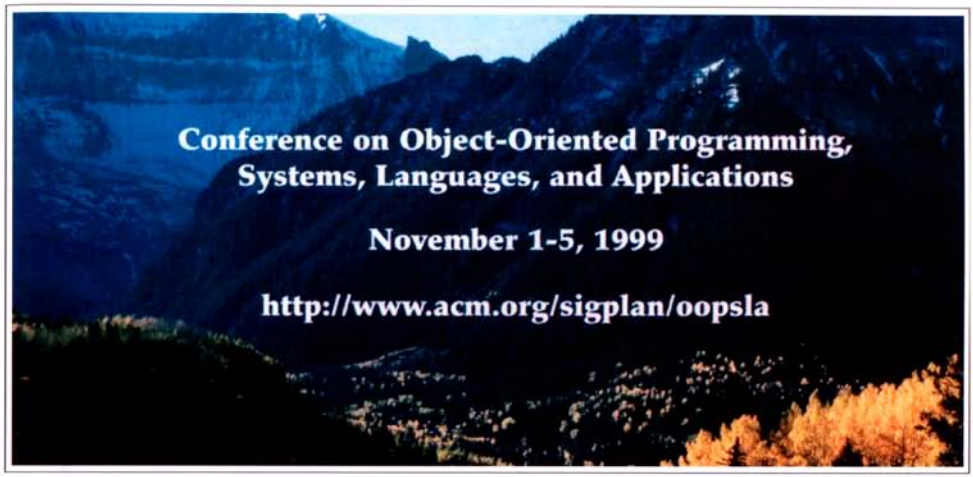


(a) 入力画像

Sharing Virtual Worlds
Avatars, agents, and social computing

(b) 文字抽出結果

図 3.15: 実験結果 1



(a) 入力画像

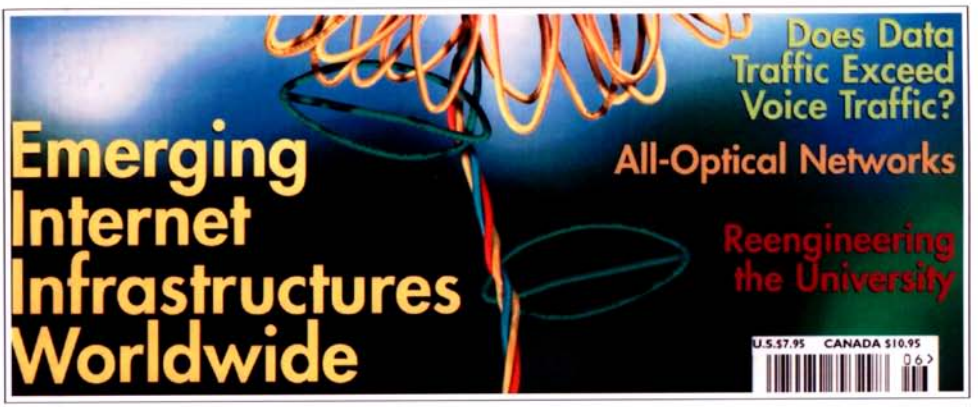
Conference on Object-Oriented Programming,
Systems, Languages, and Applications

November 1-5, 1999

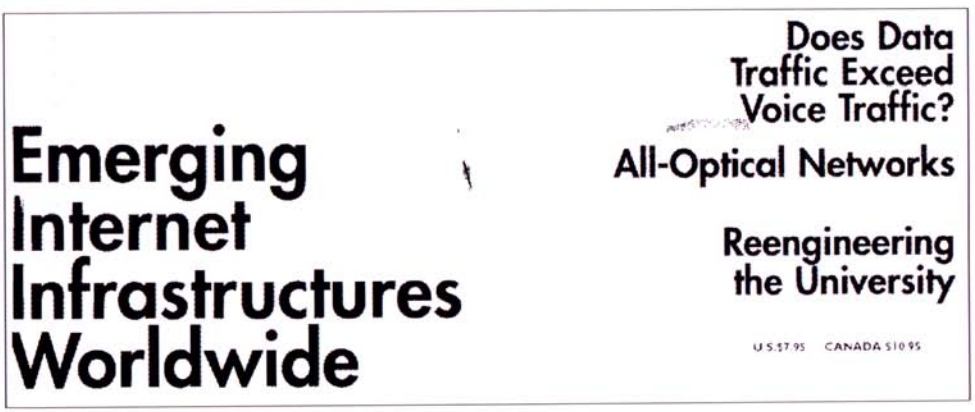
http://www.acm.org/sigplan/oopsla

(b) 文字抽出結果

図 3.16: 実験結果 2

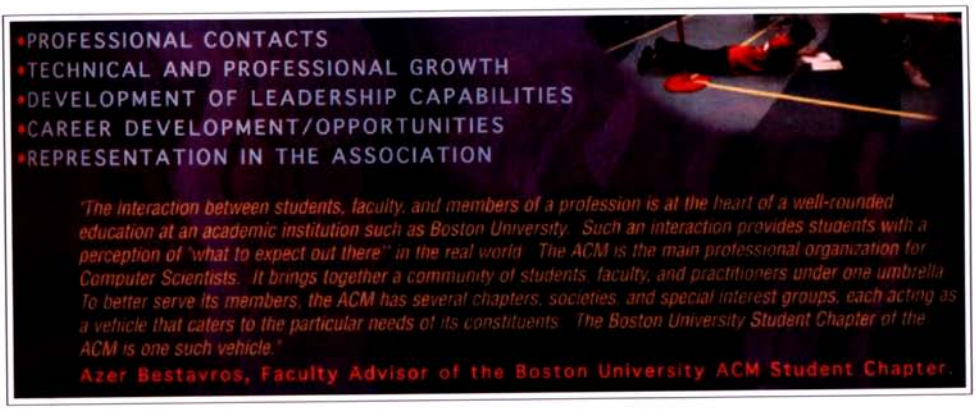


(a) 入力画像



(b) 文字抽出結果

図 3.17: 実験結果 3



(a) 入力画像

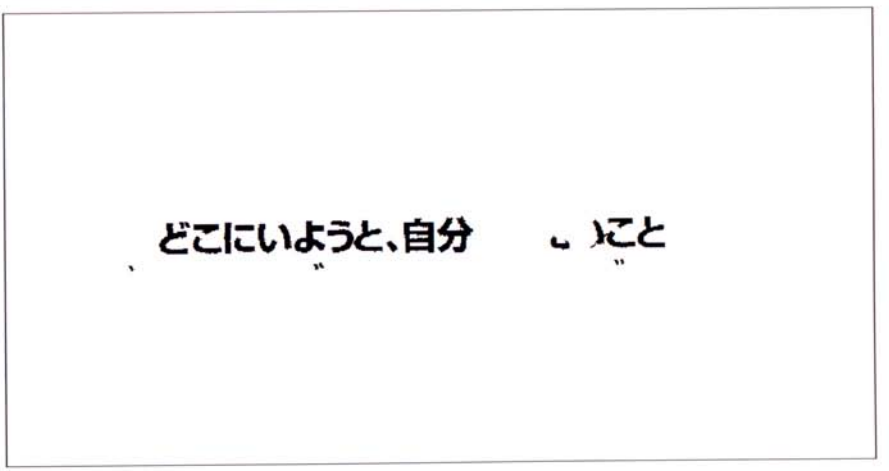
PROFESSIONAL CONTACTS
TECHNICAL AND PROFESSIONAL GROWTH
DEVELOPMENT OF LEADERSHIP CAPABILITIES
CAREER DEVELOPMENT/OPPORTUNITIES
REPRESENTATION IN THE ASSOCIATION

"The interaction between students, faculty, and members of a profession is at the heart of a well-rounded education at an academic institution such as Boston University. Such an interaction provides students with a perception of "what to expect out there" in the real world. The ACM is the main professional organization for Computer Scientists. It brings together a community of students, faculty, and practitioners under one umbrella. To better serve its members, the ACM has several chapters, societies, and special interest groups, each acting as a vehicle that caters to the particular needs of its constituents. The Boston University Student Chapter of the ACM is one such vehicle."
Azer Bestavros, Faculty Advisor of the Boston University ACM Student Chapter.

(b) 文字抽出結果

図 3.18: 実験結果 4

(a) 入力画像



(b) 文字抽出結果

図 3.19: 実験結果 5

(a) 入力画像

(b) 文字抽出結果

図 3.20: 実験結果 6

(a) 入力画像

(b) 文字抽出結果

図 3.21: 実験結果 7

第 4 章

まとめ

4.1 本研究で得られた成果

本論文では，色情報をクラスタリングして利用する画像処理アルゴリズムについて，2つの有益な手法を提案した．

第2章では，カラー画像の色量子化アルゴリズムについて述べたが，ここでは特に，高速な K-means 法という，通常よりも極めて高速な K-means 法を用いた手法を提案している．色量子化アルゴリズムに K-means 法を利用することは，量子化精度をあげるために有効である．これは K-means 法が，与えられた初期点からサンプルとその所属するクラスタ中心までの距離平均を最小にする局所最適解を求めるアルゴリズムであるからである．これによって，15 種類のファイルに対して実験を行なった結果，平均 PSNR で 37.4db という結果が出た．この結果は，他に比較した頻度法，MCA，渡辺のアルゴリズムより 3db 以上高い値である．実際，本手法を用いて色量子化した画像は，24 ビットのフルカラー画像と比べても，ほとんど違いの分からないものが多かった．しかし，K-means 法はその処理時間の長さが問題としてあげられる．第2章の実験時の条件（3次元，サンプル数=色数が数千で，クラスタ数が 256）で，当研究室で最も高速な計算機を用いても，数十秒から数分という時間がかかった．これを，今回提案した高速 K-means 法を用いることにより，約 4 分の 1 の時間で処理できるようになる．従来のアルゴリズムの中でも高速な MCA，渡辺のアルゴリズムなどと比べると，まだ若干遅くなるが，通常の K-means 法を用いた時よりは大幅に早くなっている．これによって，リアルタイムで処理を行なうようなシステムに

も応用する道が開ける．

なお，ここで提案した高速 K-means 法は，カラー画像の色量子化だけでなく，他の K-means 法を用いるアルゴリズムにも適用可能である．高速 K-means 法はとくに，クラスタ数 K が大きい場合に有効である．そこで，カラー画像の領域分割などにこの高速 K-means 法を利用することが考えられる．他にも，極めて大量のサンプルを多数のグループに分ける処理に利用することも可能である．

第 3 章では，カラー文書画像からの文字抽出について述べた．ここでは，ファジイクラスタリングを利用し，類似色をまとめることによって，複雑で多様な背景を持ったカラー文書画像から，文字色部分のみを抜き出す手法を紹介している．ファジイクラスタリングは，要素のクラスタへの所属の程度を表す帰属度という値を持つクラスタリングアルゴリズムである．これにより，微妙な色（背景と文字の中間色など）の所属の程度が分かるようになり，2 値的な分類をするハードクラスタリングでは判断の難しい低画質の画像や，色彩の豊富な画像についてもそれなりに良好な結果が得られるようになった．また，画素数の少ない色の分離が有効にできるようになったり，色彩の豊富な画像で必要以上にクラスタを生成しなくなった．本手法は，特に，複雑な背景で多彩な色を持つ画像に対して有効性が認められた．

4.2 今後の課題

第 2 章の色量子化問題では，高速化の観点からアルゴリズムを考案した．しかし，量子化精度の点では若干劣るが，本手法より高速なアルゴリズムはいくつか存在する．高い量子化精度が求められる時には本手法は有効であるが，純粋に早い処理時間が求められた場合，本手法の有効性は劣る．そこで，より高速な処理を行なうことが課題であるといえる．本手法で用いている K-means 法は与えられた解を局所改良していくアルゴリズムである．しかし，改良率は，繰り返しが進むにつれて落ちてくる．そこで，K-means 法が局所最適解に完全に収束する前の一定の繰り返し数でアルゴリズムを終了させることも考えられる．実験的に，繰り返し数を最大で 15 以下にした方が良いことが分かった．なお，高速化に際して

は，距離の計算式を簡略化することも考えられる．本手法では，すべての距離計算をユークリッド距離 $d_2(x,y) \equiv (\sum_{i=1}^{n} |x_i - y_i|^2)^{\frac{1}{2}}$ で計算しているが，それを，$d_1(x,y) \equiv \sum_{i=1}^{n} |x_i - y_i|$ と変えることにより高速化する方法もある．この場合，距離＝色差の定義が若干変わることになるが，肉眼で確認できる量子化精度にはほとんど大差がない．よって，$d_1(x,y)$ 距離の方がより実用的とも考えられる．また，章中では初期点の有効性を十分に考察していないので，量子化精度をあげるための，より有効な初期点の導出アルゴリズムを考えることも必要であると考えられる．

第 3 章のカラー文書画像からの文字抽出は，まだまだ多くの点で改良の余地が残されている．文字色抽出の手法は，本手法のようにヒストグラムを元にクラスタリングを行なう手法以外にもいくつか考案されており，それぞれに特徴があり，本手法が圧倒的に有利とは言い切れない．そして，本手法のファジイクラスタリングを用いても，必ずしも常に最適なクラスタリングが行なわれるとは限らず，画像によっては文字と背景を分離できないこともある．なお，処理対象のカラー画像の画質が高ければ，もっとシンプルな手法で効果的に文字色を抽出できることが多い．本手法を改良するにあたり考えられることとして，まず，クラスタリングを行なう色空間をより適当なものにすることが考えられる．本手法では，色空間を RGB から $L^*u^*v^*$ に変えてクラスタリングを行なったが，実験を行なうにつれて，$L^*u^*v^*$ 色空間が必ずしもクラスタリングに最適とは限らないことが分かった．つまり，色値を RGB から $L^*u^*v^*$ 色空間に変換しても，必ずしも印刷してある色差を十分に捉えることができないということである．また，色情報だけでなく，位置情報をより効果的に利用するクラスタリングを考える必要がある．位置情報の利用として，本研究では画像の垂直座標のみ利用したが，この処理を追加しても処理時間を多く必要とするだけで，必ずしも効果が上がらなかった．色情報だけでなく，位置情報も利用するというアイディアは有効であると思われるが，その利用法には今後も研究が必要である．また，色分解画像を作成した後も，外接矩形から文字列を識別するアルゴリズムには改良の余地が多い．現在は外接矩形だけを見て判断しているが，特に日本語のように，文字がいくつもの部分からなることが多い場合，外接矩形を見ても文字かどうかの判断が難しい場合がある．また，縁取り文字や，文字色と同じ色の背景との接触のある文字を抽出しようと考えた場合も，よ

り複雑なアルゴリズムが必要である．また，現在はある程度長い文字列でなければ文字を抽出できないので，見出し等の文字数の少ない文字列からの文字抽出も今後の課題である．なお，今回は横書きの文字のみを対象にした．そのため，画素の垂直座標の情報をクラスタリングに生かすことができたが，今後は縦書きの文字列や，傾きを持った文字列に対しても抽出が行えるようにすることが考えられる．クラスタリングのアルゴリズムだけでなく，2 値化した色分解画像からの文字抽出のアルゴリズムについても，あらゆる傾きの文字列に対応できるように拡張が必要である．

謝辞

本論文は筆者が信州大学大学院に在学中にクラスタリングアルゴリズムと画像処理に関して行った研究をまとめたものです．本研究を進めるにあたり，多数の方々のご指導，援助をいただきました．論文を締めくくるにあたり，ここに慎んで感謝いたします．

長年ご指導いただきました山本博章助教授，岡本正行教授に深く感謝いたします．また，有益な助言をいただきました中野康明教授，そして，他の情報工学科の先生方に深く感謝いたします．

最後に，筆者が研究をすすめ論文を作成する際に，様々な角度から協力していただいた岡本・山本研究室の学生の方々に深く感謝いたします．

参考文献

[1] 田島譲二，“カラー画像複製論” 丸善株式会社（1996）

[2] 高木幹雄，下田陽久，“画像解析ハンドブック” 東京大学出版会（1991）

[3] Heckbert, “Color Image Quantization for Frame Buffer Display”, *ACM Computer Graphics*, Vol.16, No.3 pp.297–307（1982）

[4] 渡辺孝志，“カラー画像を 256 色で近似表示するための高速なアルゴリズム” 電子情報通信学会論文誌 (D) Vol.J70-D No.4 pp.720–726（1987）

[5] 渡辺孝志，“色彩量子化画像のための改良ディザリング法” 電子情報通信学会論文誌 D-II Vol.J72-D-II No.7 pp.985–992（1989）

[6] 稲葉真理，今井 浩，加藤直樹，“Randomized Algorithms for Variance-Based k-Clustering” 冬の LA シンポジウム論文集（京都大数理解析研究所講究録 871 pp.124–129）（1994）

[7] 稲葉真理，今井 浩，加藤直樹，“ランダマイズドクラスタリングアルゴリズムに関する実験結果について” 情処学アルゴリズム研報 53-12（1996）

[8] M. Celenk, “A Color Clustering Technique for Image Segmentation”, *Comput. Vision Graphics Image Process. 52*, pp. 145–170（1990）

[9] J. Liu and Y. H. Yang, “Multiresolution Color Image Segmentation”, *IEEE Trans. Pattern Anal. Mach. Intell. 16, 7*, pp. 689-700（1994）

[10] S. H. Park, I. D. Yun and S. U. Lee, "Color Image Segmentation based on 3-D Clustering:Morphological Approach", *Pattern Recognition 31, 8*, pp. 1061–1076 (1998)

[11] J. Ohya, A. Shio and S. Akamatsu, "Recognition Characters in Scene Images", *IEEE Trans. Pattern Anal. Mach. Intell. 16, 2*, pp. 214–220 (1994)

[12] 松尾賢一，梅田三千雄，"濃淡及び色情報による情景画像からの文字列抽出" 電子情報通信学会技術研究会報告 PRU92-121 (1992)

[13] 仙田修司，美濃導彦，池田克夫，"文字列の単色性に着目したカラー画像からの文字パタン抽出法" 電子情報通信学会技術研究報告 PRU94-29 (1994)

[14] Y. Zhong, K. Karu and A. K. Jain, "Locating Text in Complex Color Images", *Pattern Recognition 28, 10*, pp. 1523–1535 (1995)

[15] 長井晋司，"カラー画像からの文字抽出" 信州大学大学院工学系研究科 平成 8 年度修士学位論文 (1997)

[16] 河尻寛之，"色情報による文字抽出の研究" 信州大学大学院工学系研究科 平成 10 年度修士学位論文 (1999)

[17] J.C.Bezdek, " Pattern Recognition with Fuzzy Objective Function Algorithms" Plenum Press (1981)

研究業績

論文

[1] 春日秀雄，山本博章，岡本正行，“高速K-means法を用いたカラー画像の色量子化”，電子情報通信学会論文誌，vol.J82-D-II，No.7，pp.1120–1128（1999）

[2] Hideo Kasuga, Masayuki Okamoto, Hiroaki Yamamoto, “Extraction of characters from color documents”, IST/SPIE's 12th Annual International Symposium Electronic Imaging 2000

口頭発表

[1] 春日秀雄，山本博章，岡本正行，“高速K-meansクラスタリングを用いたカラー画像の色量子化”，電子情報通信学会技術研究報告，IE97-1811（1998）

[2] 春日秀雄，山本博章，岡本正行，“カラー画像からの色抽出法としてのファジイクラスタリング”，電子情報通信学会総合大会講演論文集，D-11-169（1999）