自己組織化マップを用いた絵画を想像する人工知能のデータベース部分の開発

迎山　和司

公立はこだて未来大学

1. はじめに

本研究では人間がイメージを理解し表象できる仕組みをプログラムによって実現する。この研究は「人工知能画家・静」と名づけたプログラムを作品として発表するために、2000 年より続けている研究の一部である。これはプログラムに人間より上手な絵を描かせることを目的にしているのではなく、人間が絵を描く手順を生徒のようにプログラムに教えていくことによって、逆に普段自分が意識していなかった絵を描くことの意義を発見することを目的としている。これまでの研究によって、このプログラムは、経験として学習した線描から自律的に独創的な線描を描けるようになった。例えば、細かくプログラムをしたわけでないのに、単純な手描きの円を与えればそこに目や鼻や口に当たるものを描き加えることが実現できている。これは、獲得した過去のイメージからプログラムが自律的に選択した結果である。しかし、現在はこのプログラムは不安定な挙動が多く、期待する鑑賞に堪えるだけの絵画を生成できるまでには至っていない。よって今後は学習するイメージの多様性を増やすために、膨大なデータベースの自動作成、より妥当といえる選択をする決定ルールの検証、線描以上のより鑑賞に堪える洗練された表現の実現を考えている。このうち今回の課題では、膨大なデータベースの自動作成を中心にプログラムを開発する。すなわち、入力としての「目」の部分をより強化し、それによって得られた膨大なイメージを処理する部分を深く考察して、期待する挙動を行うようにする。このプログラムの特徴は、人間の幼児のように周りの情報を自らが経験して法則を作り出すことに注目している。以下に本研究の概要と今回作成したデータベース部分についての目的と結果を報告する。

2. 人工知能画家・静

美術とは何であろうか？という疑問に決定的な答えはない。しかし、コンピュータに絵を描かせることは、この疑問に対する理解の一助として非常に有意義である。なぜなら、コンピュータプログラムによって絵を作り出すことは、何も知らない生徒に一つ一つ絵の描き方を教えることに近いので、普段自分がなにげなく行っている描画行為がいかに簡単ではないかを逆に知ることが出来るからである。筆者はそのために「静（しずか)」と名付けた絵を描くコンピュータプログラムの制作を続けている[1]。現状の静は人間の想像力に注目し、スマートボードと呼ばれる電子黒板上に鑑賞者が何かを描くと、それに関連した絵を描き加えることができる。（図１、２）この仕組みの実現のために、静にあらかじめいくつかの絵を読み込ませその絵をばらばらのパーツにして記憶させている。このパーツを Primitive とよぶ。各 Primitive はお互いに相手の Primitive の位置関係を情報として持っており、関連する Primitive を集合すると意味のわかる絵になる。この Primitive の集合を Figure とよぶ。（図３）ある Primitive に似たような形が描かれるとその形に関連した Primitive の形を描き加える。例えば、静は顔を多く覚えているの

で、円を描くとその円の目や鼻や口にあたる位置にしかるべき形を描き加えることが出来る。この仕組みの特徴とするところは、たくさんの絵を憶え込ませるほど豊かな絵を描けることにある。このような研究あるいは作品制作には、他にはCohenのAARON[2] や Burton のThoughtful Drawings[3] などがある。

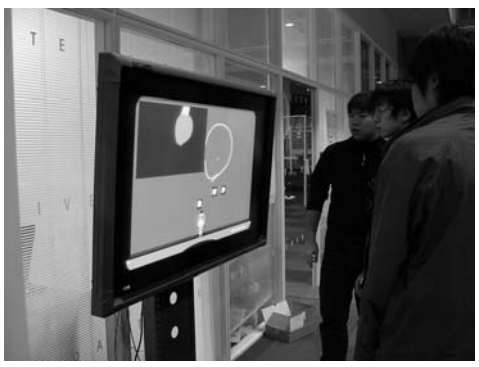

図1 スマートボードによるデモンストレーション

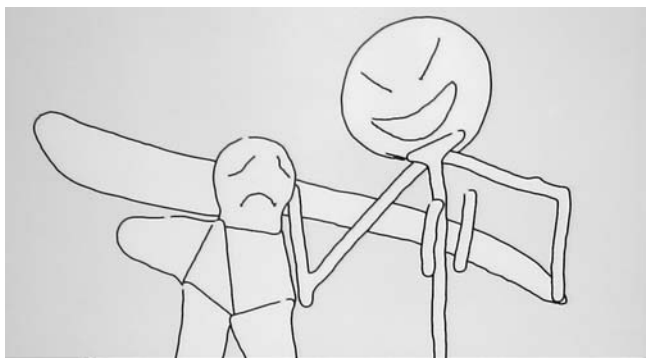

図2 静(しずか)と鑑賞者が共同で描いた絵

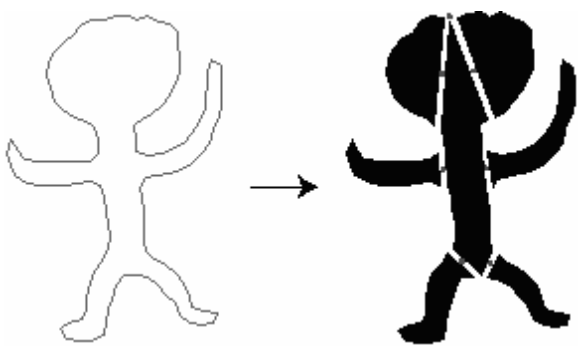

図3 PrimitiveとFigure

3. 絵画の原理

イメージができあがる様子を直接他人の脳から観察する方法を、我々はまだ知らない。したがって、イメージの生成を知るには脳の機能を考えるのではなく、描かれた絵画の共通性を考えるほうが妥当である。ある絵画を描くとき、人間は平面空間上になんらかの意味を付加している。ある絵画を見たとき、人間は平面空間からなんらかの意味を読み取り評価をする。つまり、絵画を作成し鑑賞するということは、意味の「記録」と「読取」の相互行為である。しかし、描く人間の意図が見る人間に正確に伝わる保証はどこにもない。なぜなら、描く人間の意味は描く人間の知識により描かれるし、見る人間は自身の知識に基づいて意味を解釈する。両者の知識はまったく同じではないからである。にもかかわらず、顔の絵を描けばその絵は多くの人間が「それは顔である」と読み取ることが出来ることは事実である。すなわち、平面空間上に記録された図形には多くの人間に同じような意味の喚起を促すパターンがあるといえる。これはおそらく、多くの人間は近しい群集にいるためにおおよその知識を文化として共有しているからであろう。たとえば、ネイティブアメリカンの洞窟壁画は 1500 年前から800 年前に描かれたものであるが、作者はもはや存命していなくても我々はその壁画に描かれたものが人を意味していることが理解できる。このことに注目して、これまでの研究で明らかになったことは、意味を生成するイメージには

・意味を生成しているのは形状ではなく配置
・同じ意味を共有している配置は似ている
・配置を守れば図形を変えても意味は保持される

という原則があることである。たとえば人の顔のイメージの配置は、おおよそ似ている。この配置をくずさないで目の部分の形を○から×に変えたとしても、それは顔であるという意味は保持されている。さまざまな組み合わせを瞬時に行うことは、コンピュータの得意とするところである。（図４、５）

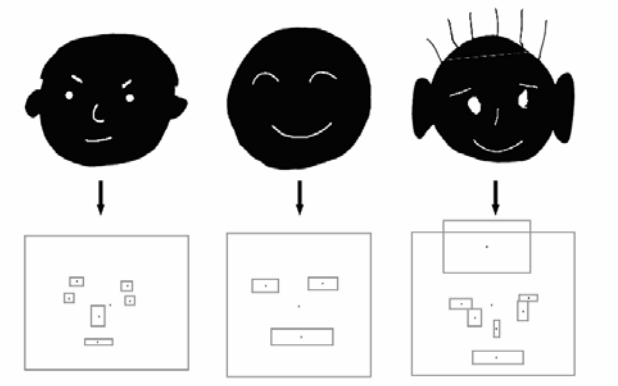

図4 顔のイメージの配置関係

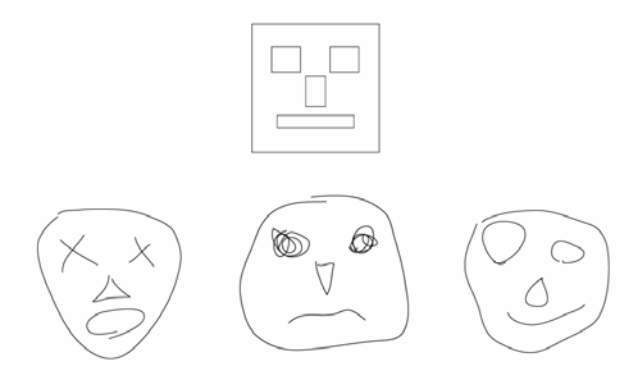

図5 配置が同じ状態での図形変更例

4. 静の動作原理

静は鑑賞者が描いた線描などの任意の画像を取り込み、その任意の画像が意味を表現できないレベルにまで Primitive に分解してそれらをデータとして記憶する。任意の画像からデータは理論上無限に獲得できるが、データが増えて選択肢が多くなっても、もっともらしい連想候補を自律的に選び出すためにはなにかしら選択のためのルールが必要である。現在は、人間が長い紐を見たときに蛇をイメージする、といったようなことを実現するために描かれた図形の形から、判断している。この実現のために SOM を採用しており、これによってもっともらしい連想候補が選ばれやすくなっている。先に述べたとおり、絵画において意味を生成しているのは形状ではなく配置なので、この規則を守っていれば、ある部品を構成する図形を別の図形に交換しても、意味の表現は保持されることがわかっている。よって、ある形が選択されて連想候補が選ばれると、その連想候補が持つ Primitive データをすでに描かれている画像に付け加える。このようなプロセスを知識として与えると、人の形をした線描の中に顔をつけたり、顔の周りに花びらをつけたりするようになった。静の持つ知識はデータの「取り扱い方法」のみである。顔はこのような形、花はこのような形という意味を辞書としては持っていない。にもかかわらず、教えてはいないのに人面花のような絵を描くことができた。これは先に述べた絵画の原理を踏まえた結果であり方法として有効である。しかしながら、実際は人面花のような連想がでてくることはまれで、描き進めるうちに意味のわからない絵になることのほうが多い。また、現段階では線描を誰かに描いてもらわなければいけないので、画像を収集するのが大変である。

5. 開発目的

我々が普段目にするコンピュータは、そのままでは手も足も目もない。また、プログラムを与えないと自分からは動かない。よって、 Primitive データを蓄えるための最大の問題点は、こちらから画像を収集しなければならないことだ。現段階では人が描いた線描の画像に特化しているた

めに、収集する画像が制限される。そこで、カメラを使って撮影された実写画像でも対応できるようにする。さらに、集めた大量の画像を分類し共通する部分を求めて、ある任意の概念を代表する画像テンプレートが得られるようにする。このとき、あらかじめ画像をこちらで整理して、静に「顔」などの言語と連結した辞書のデータベースを用意しないのには理由がある。なぜなら、言語は異なる文化圏によって指し示す範囲が異なるので、自律的にルールを決定するモデルとして妥当ではないと考えたからである。筆者は、意味は言語によって先天的に概念化されているのではなく、何らかの試行の末に概念が生まれそれが言語によって概念化されると仮定している。つまり、たくさんものを見て、その中から繰り返し見たものは意味のあるもので、それがわかるように分類しクラスタリングすれば概念の形成が自律的にされたと思えるのだ。この仮定の成否はともかく、プログラムとしては後者のほうが柔軟なデータベースモデルになる。よって今回開発するデータベースの特徴はカメラを使って撮影された実写画像を分類し、さらにこの画像から Primitive の集まりである Figure データを作成してテンプレートする。

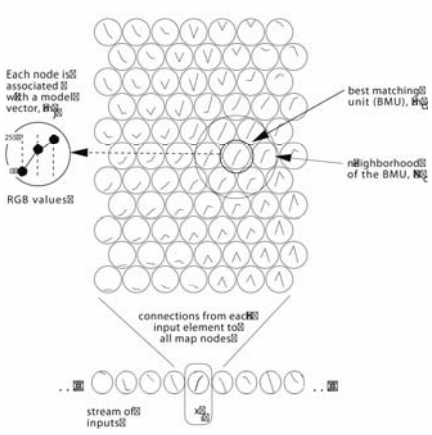

図6 SOM の仕組みの図解[5]

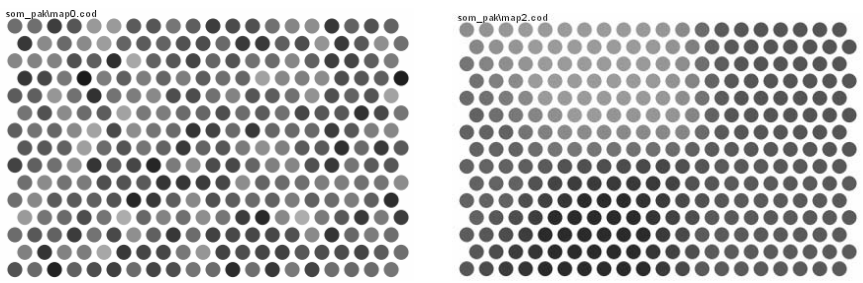

図7 SOM による色の認識例

6. 自己組織化マップ

自己組織化マップ[4] (Self-Organized Map: 以下 SOM と呼ぶ) はフィンランド国立アカデミーの名誉教授であるコホネンによって考え出されたデータを分類するアルゴリズムで、いわゆるニューラルネットと呼ばれる動物や、人間のニューロンと呼ばれる脳(とくに大脳皮質)の神経細胞網を参考にしたコンピュータモデルである。SOM は網のような状態で構成された2 次元のマップで、網の結び目一つ一つがニューロンに相当する。結び目は、近傍の結び目のデータと自分のデータが均衡するように修正しながら学習していく。学習によって網の均衡状態が収束すると、結果は入力されたデータの近似する分類マップになっている。図6のように、SOM のデータは数値の並びで構成されたもので、これをベクトルデータという。数値の個数は柔軟で理論上制限はないが、ここでは説明のために三つの数値の並びで表現できる「色」を扱う。初めにランダムなベクトルデータを持つ各ノードは、学習回数を繰り返すごとにデータを修正して整理していく。10000 回の学習の結果、それぞれの特徴に分類された SOM が出来上がる。学習された SOM に新しいベクトルデータをテストさせると、すでに学習されたノードのなかから一番近いノードが一致する。このため、完全な正解ではなく一致したノードにある程度近いものであるという判断をすることに向いている。この特性を生かして、SOM は文字認識、音声認識や指紋照合などの分野で広く使われている。また SOM には、ベクトルデータであればどんなデータでも柔軟に扱えるという特徴がある。例えば図7は5 色の色データに対して、学習を行った結果である。

20 × 15 の平面空間にある結び目（ノード）は、初めばらばらのデータを持っているのでまったく異なる色をしている。しかし、5 色のデータを与えるとノード間で学習が起こり、およそ五つの領域が自己組織的に出来上がる。

7. データベース・プログラムの紹介

今回のデータベース作成に当たり 4 つのプログラムを作成した。すべてのプログラムは Java [5] で作成されている。以下にそれぞれのプログラムの説明をする。

7.1. SOM Data File Manager

SOM Data File Manager は後に述べるSOM Viewerで扱うためのデータを作成するプログラムである。JPEG形式[6] で保存された縦256ピクセル、横256ピクセルのカラー画像から、1024 の配列データを特徴成分として抽出する。特徴成分は原画像をWavelet変換[7] することによって得られる。画像をWavelet変換すると、LL 成分、HL 成分、LH 成分、HH成分という四つに 4 分割されるこのうち最も低周波である LL 成分について 3 回Wavelet変換を繰り返すと、最終的に32x32の矩形ピクセルが得られる。今回の画像の特徴としては、この部分を使っている。データは拡張子.datで任意の名前で保存できる。これをサンプルデータと呼ぶ。（図８）サンプルデータはテキストファイルでできており、一画像あたり一行で表現される。画像から取り出した特徴成分は配列数値ごとにスペースで区切られ、行の最後にその画像のファイル名を付加する。また一行目には配列の次元数を表す数字が入る。今回の場合は 1024 と数字になっている。任意の指定されたディレクトリにあるすべてのJPEG形式画像を一度にサンプルデータとしてまとめるため、サンプルデータの行数は（ディレクトリに入っているJPEG形式画像数 +1 ）となる。仕様上データ数の制限はない。

```
1024
 0.24705882352941178 0.24705882352941178 0.25098039215686274 ...0.1843137254901961 015AD093
0.23529411764705882 0.23921568627450981 0.23921568627450981 ...0.17647058823529413 015AD218
…
```

図8 サンプルデータ

7.2. SOM Viewer

SOM Viewer（図 10 ）は SOM Data File Manager で作られたサンプルデータによって似ている画像データを分類するプログラムである。基本的な機能はコホネンによって公開されているSOM_PAKに従っている。特徴としてGUI（グラフィカル・ユーザ・インターフェース）によってマウスで操作できるようになっており、ただ結果を出力するだけでなくその様子を視覚的に確認しながら表示することができる。また、学習回数なども任意に指定できるため、どのようなデータの時には学習が適当なのかを試行錯誤できる。学習によって分類された結果は、保存し再度読み込むことができる。保存できる結果には 2 種類あり、SOM が学習したマップデータとそのマップデータのどの部分に実際のサンプルデータが属しているのかを示した配置データがある。マップデータは拡張子.codで任意の名前で保存できる。データの構造は基本的にはサンプルデータとほぼ同じである。一行あたりサンプルデータと同じ次元数の配列データ

が書かれている。サンプルデータの違いは SOM の学習結果を記録したものなので、行の最後にはファイル名が付加されていない。また、マップの縦横の次元数が一行目に付加されている。たとえば例にあげたマップデータ（図９）では１ノードあたり 1024 次元の配列データがあり、ノードが縦 40 、横 40 の２次元に配置されている。この場合、全行数は40x40+1となる。配置データは拡張子 .vis で任意の名前で保存できる。他の２つのデータと違い一行あたり必ず３次元の配列データで、左から順にマップの x 座標、マップの y 座標、誤差である。これらはサンプルデータの順に参照し、マップデータから最も近いノードの座標を求める。近さはユークリッド距離として求められるので、その差が最も小さいものが求めているマップ上のノードである。距離の差は誤差として扱われる。その求めた座標と誤差、さらに自分のファイル名を付加して一行分とする。一行目にはかならず３が付加されている。構造はサンプルデータと同じであるので、これをサンプルデータとして利用することもできる。

```
1024 40 40
 0.24564926 0.24919987 0.24644664 ...0.20510937
0.25161803 0.2550185 0.2544072 ...0.20087527
...
```

図9 マップデータ

図 10 SOM Viewer 画面

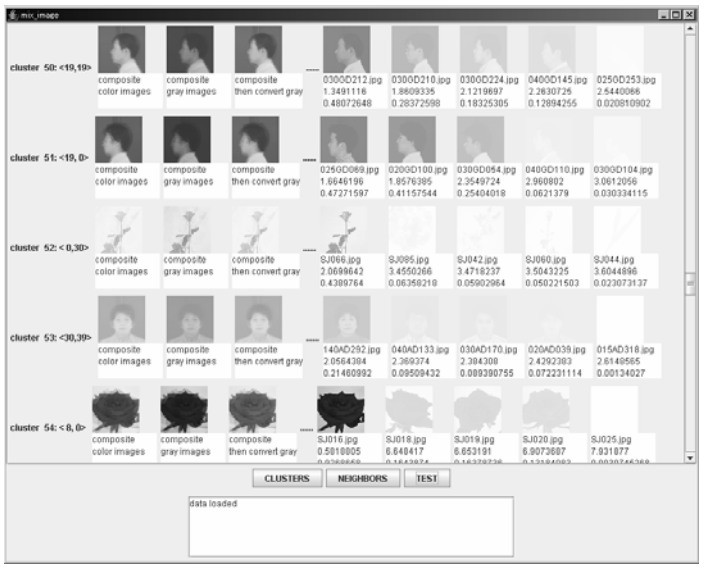

図 11 Cluster Viewer 画面

7.3. Cluster Viewer

Cluster Viewer（図 11 ）はSOM Viewerで得られたマップデータをクラスタリングし、得られたクラスタに属する画像データの表示や合成をするプログラムである。上記二つのプログラムによって得られた各データから、２つの方法で実際の画像を合成する。クラスタリングの方法は田中らの方法[8] に従った。この方法は、まず SOM のマップの各ノードに近傍する４つのノードからの、各ユークリッド距離の平均をもとめる。これを距離マップとし、さらに近傍の距離の差が近いものを、同じクラスタに所属するものとして番号をつけていく。番号がつけられたクラスタのうち、初めにクラスタ番号がつけられたものをクラスタの中心とする。中心となったノードがそのクラスタを代表するデータとなる。（図 12 ）クラスタを代表とするノードが得られたということは、それがあるものをたくさん見た場合のテンプレートと言える。し

かしながらテンプレートのデータをそのまま後に述べる Primitive Maker を使って Figure を作成するには不都合がある。ノードの次元数は 1024 であり、画像としては 32x32 の解像度の低い画像である。これを 256x256 の画像として拡大して Figure 作成を行っても望ましい結果にはならない。かといって 256x256 の画像を直接 SOM で分類するには膨大な計算時間がかかる。そこでテンプレートノードのデータとサンプルデータから誤差を再計算し、誤差が少ない順にソートした画像を、その誤差の量を透明度にして合成した。全画像を合成することは、コンピュータのメモリの限界もあり、クラスタに属さない画像も含めるのはあまり有効ではないので、試行錯誤の結果、先頭から 5 個分を合成することにした。合成に関してカラー画像を直接合成する以外にグレイスケール画像もテストした。なぜなら Wavelet 変換をする際に、カラー画像をグレイスケール画像にしているため、SOM の分類時には色情報は利用されていないためである。

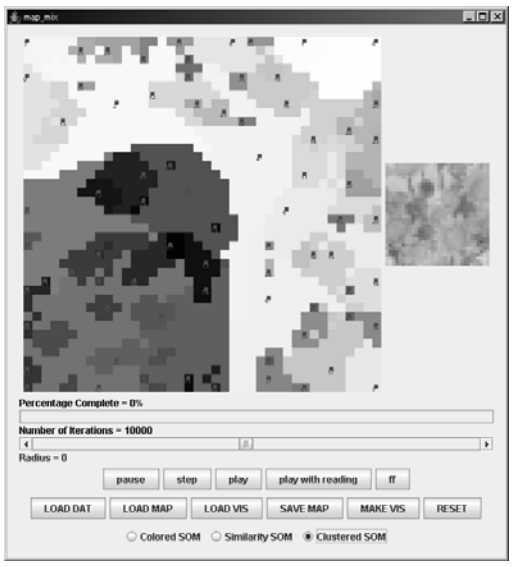

図 12 クラスタリング表示

図 13 画像から変換された Figure

7.4. Primitive Maker

Primitive Maker は画像から第 2 章で説明した Figure を作成するプログラムである。任意のカラー画像が与えられると k-mean クラスタリング[9] によって画像を Primitive という Primitive の集合である Figure に変換する。単独でも動作するが、今回は Cluster Viewer のなかに組み込んでいる。Cluster Viewer で表示されている画像をマウスポインタでクリックすると、画像データが Primitive Maker に送られる。Primitive Maker は処理が終了すると、結果を新しいウィンドウに表示する。ウィンドウ内では、作られた Figure を自由に動かすことができる。（図 13 ）

8. 実験と結果

顔[10] 、風景、花、動物など[11] 様々な画像について、このプログラムを使って実験を行った。この内、正面顔、横顔および花の画像 300 枚（図 14 ）について説明する。まず用意した 256x256 の JPEG 形式画像から、SOM Data File Manager によってサンプルデータを作成した。次に SOM Viewer を用いて、得られたサンプルデータに対して学習を 10000 回繰り返した結果、マップデータと配置データを得た。図 15 は学習した後の各ノード間の距離マップである。花と顔で大きく濃度が違い、ここで大局的に 2 つの領域に分かれていることがわかる。またノードを画像に復元してみると、顔部分の領域でも正面顔と横顔で分かれている。さらに、この距離マップに

基づいてクラスタリングを行った結果が図 16 である。クラスタリング個数は、距離マップを計算するための近傍をいくつとるかによって変わる。ある程度離れたものまでとると大まかなクラスタになり、近いものだけとると細かいクラスタになる。図 16 は最も近いものだけをとった 4 近傍でのクラスタリング結果である。この場合、クラスタ数は全部で 88 個となった。この結果をCluster Viewerを用いて、得られた結果が図 17 である。右側の 5 個の画像が実際の画像にグレイスケール処理と透明度を施したものである。画像の下にある情報は上から順に、ファイル名、テンプレートからの誤差、誤差による透明度である。透明度はテンプレートを 1.0（不透明）6 番目に近い画像を 0.0（透明）として、その範囲で画像の透明度を決めている。左側の 3 個の画像が合成した結果である。それぞれの違いは左から原画像の合成、原画像をグレイスケール画像にしてからの合成、原画像の合成の後のグレイスケールである。印刷ではカラーではないのでわかりにくいかもしれないが、原画像の合成と原画像をグレイスケール画像にしてからの合成とでは後者のほうの濃度が高い。実験で用いた画像は大局的には正面顔、横顔および花という 3 つの意味を持つものであるが、最も細かいクラスタリングではクラスタ数は 88 個となっている。このことについて各クラスタであげられた 5 個の画像を調べてみると、さらに詳細に傾向が分類されていることがわかった。たとえば図 18 のように顔に関しては正面、横といった向きだけではなく、年齢や性別でも分類されていることがわかる。ただし、特徴成分はあくまでも画像のみで年齢や性別を入れているわけではない。にもかかわらずこの結果が得られるのは、髪型や顔のつくりなどが見た目において年齢や性別に無関係ではないことを表していると思う。得られた合成画像から Primitive Maker を使って Figure を作成すると、図 19 の結果が得られた。合成することによって細部の情報がぼやけて大局的な配置情報を得ることができている。

図 14 正面顔、横顔および花の画像 300 枚

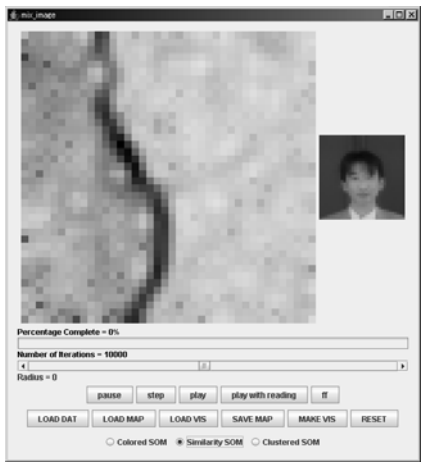

図 15 SOM 学習結果

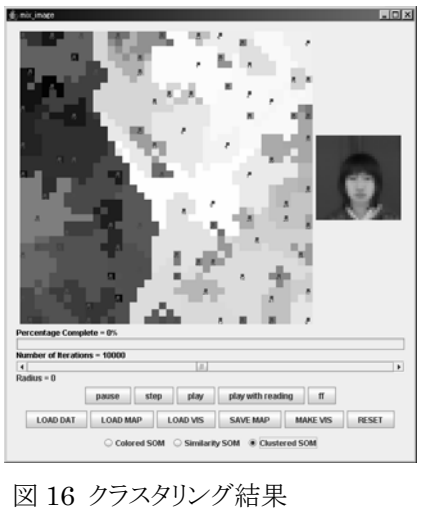

図 16 クラスタリング結果

図 17 クラスタリングの内容表示

図 18 各クラスタの分類傾向

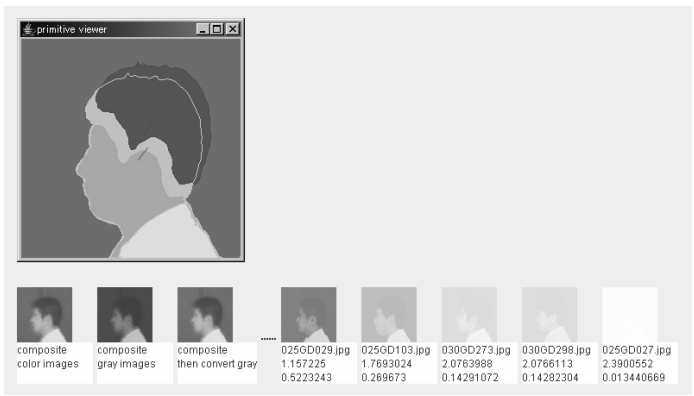

図 19 Figure 抽出結果

9. 問題点と限界

正面顔、横顔および花の画像 300 枚（図 14 ）ではある程度期待する結果が得られた。しかし、さまざまな要素を含む画像 5000 枚（図 20 ）では期待できる効果が得られなかった。とりわけ問題だったのが風景画像である。風景画像には複数の対象物が写っているために、もともと意味をひとつに特定しにくい。そのような画像は単体の画像と比べて画像の特徴量を煩雑に含んでしまう。このおかげで無関係な意味同士をクラスタリングすることが多くなり、結果的に意味の読み取れる画像でないものが多くなった。そこで、この 5000 枚の画像から、分類に問題と思われる画像、つまり風景画像や背景が一様ではない画像を除外した 1345 枚の画像について再度検討した結果、ある程度まとまった結果になった。また意味としては正面の顔画像だとしても異なる照明条件の画像が混在すると最終的な分類はうまくいかないことがわかった。したがって、うまく分類させるためにはあらかじめ入力する画像を考えなければならない。それは均一照明で無背景であることが条件である。また風景画像は Figure を作成する際にはふさわしい画像でないことがわかった。もともと複数の対象物が写っている風景画像では、輪郭をトレースして形を出す際に非常に時間がかかり、得られた画像も複雑すぎて入力画像としてはふさわしくなかった。（図 21 ）これらの問題は画像認識でも難解な問題で、どのような状況でも対応する汎用的なモデルを作ることは難しい。

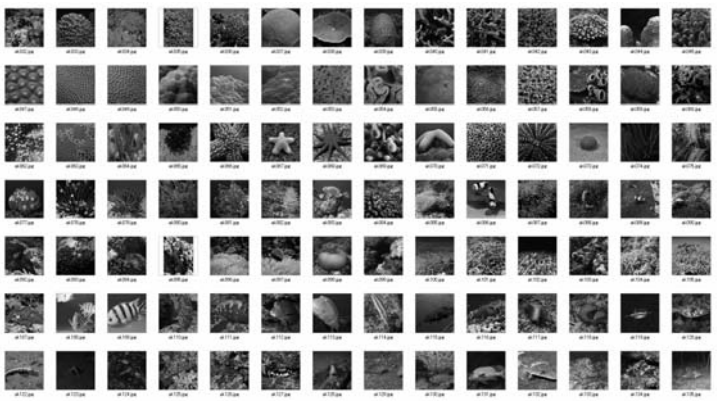

図 20 様々な画像 5000 枚

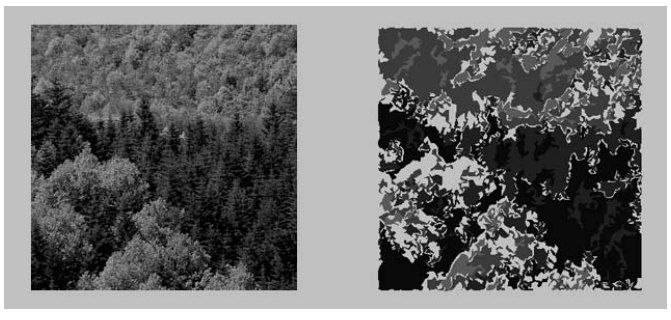

図 21 Figure 化に向かない画像

10. 今後の展望

本研究の目的は、繰り返し見たものは意味のあるものであるという仮定の下に、大量の画像データを自動的に分類にクラスタに分けて、その代表データ（テンプレート）を抽出するデータベースの開発であった。実際、開発したデータベースを用いる場合、物体をスタジオ撮影に限定して、

撮影条件を同じにすれば求める目的には近いものは得られる。しかしながら、実のところ整然とした画像から分類するならば、初めから顔ならば顔の画像一枚を元に、手作業でテンプレートを得ればよい。今回の実験で SOM が柔軟に分類できるとしても、異なる条件化で撮影された実写画像からでも分類がおおむね得られる汎用性の高いシステムを作成することは、困難を伴うことがわかった。以上からデータベース開発はここで区切りとして、今後は手本 Figure を写真画像から手作業で作成する。ただし、条件が限定していればうまくいくということは、分類プロセスそのものは間違いとはいえないと考えている。よって手作業で得た Figure がある程度多く集まったとき、さらに同じ分類プロセスが適用できるかどうかを、試行錯誤で探していく予定である。同時にこれらの手本データを使い、絵画生成プログラムの開発を始める予定である。

参考文献

1 迎山 和司：“人工知能画家 静 第3 版”, 情報処理学会, 東京, 2004, 2004-HI-108 Vol.2004 No.51

2 Harold Cohen: http://crca.ucsd.edu/hcohen/, 人工知能画家 AARON の製作者

3 Ed Burton: “Thoughtful Drawings: A Computational Model of the Cognitive Nature of Children's Drawing”, Computer Graphics Forum 14(3) Eurographics Association, 1995, p159-170

4 Timo Honkela: “Self-Organizing Maps in Natural Language Processing”, Helsinki University of Technology, Espoo, 1997

5 Java テクノロジ：http://jp.sun.com/java/, Java 言語の入手先

6 Joint Photographic Experts Group: http://www.jpeg.org/, 静止画像データの圧縮方式の一つ

7 Yves Meyer: “Wavelets: Algorithms and Applications”, Society for Industrial and Applied Mathematics, Philadelphia, 1993, pp. 13-31, 101-105.

8 田中 雅博, 古河 靖之, 谷野 哲三, “自己組織化マップを利用したクラスタリング,” 電子情報通信学会論文誌, vol.J79-D- Ⅱ, no.2, pp.301-304, Feb.1996

9 IJ Plugins: Clustering: http://ij-plugins.sourceforge.net/plugins/clustering/index.html, ImageJ k-means clustering pluginの入手先

10 財団法人ソフトピアジャパン IT 研究センター: "顔画像データベース ", http://www.softopia.or.jp/rd/facedb/outline01.html

11 Datacraft: "素材辞典 " http://www.sozaijiten.com/