DBSJ Journal, Vol.7, No.3
December 2008

色に応じた異なるハッチパターンを描画するプラグインの開発

Development of Web Browser Plug-in that Draws Different Hatch Patterns According to Color

下田 雅彦♠　　横田 一正♦

Masahiko SHIMODA　Kazumasa YOKOTA

地下鉄の路線図や気象情報を表す地図など，特定の情報が色によって表現されたイメージがインターネット上には数多く存在する．近年，色覚バリアフリーの意識が一般的に向上してきており，高齢者や色覚異常の人々に配慮された配色のものが増えてきた．しかし，色で塗り分けられたイメージから彼らが必要な情報を得るのは困難な場合が多い．この問題を解決するために，我々は，ウェブブラウザに表示されているページに，色に応じて異なるハッチパターンを描画するプラグインを開発した．このプラグインにより，色の区別を補助する目的は十分に果たせることが確認できた．しかし，同時にいくつかの問題点も明らかになった．この結果から，我々は，この研究における今後の課題とその解決方法について考察した．

On the internet, there are many images in which each information is represented by a different color, such as a subway map and a weather map. In recent years, peoples' awareness of barrier-free color vision is rising, and the images with consideration to the elderly and color-blind people are being increased. However, in many cases, it is difficult for them to obtain necessary information from those color-coded images. To solve the problem, we developed a web browser plug-in that draws different hatch patterns onto different color regions shown in a browser. We could confirm that the purpose to assist distinguishing one color from another is accomplished enough by using the plug-in. At the same time, however, some new problems have appeared. Based on the results, we discussed about future problems in this study and their solutions.

1. はじめに

インターネットのウェブサイトには，複数の情報がそれぞれに対応した色によって表現されている地図やグラフなどのイメージが数多く見られる．例えば，地下鉄の路線図では，各路線がそれぞれ異なる色で描画されている．また，気象情報を表す地図では，警報・注意報などの種類や程度が色によって表現されている．近年，ウェブアクセシビリティの意識が一般的に向上してきており，ウェブページやその中で使われるイメージの配色についても，高齢者や色覚異常の人々への配慮が為されるようになってきた．しかし，区別すべき情報の数が多い場合には，多くの色を使う必要があり，すべての人にとって区別しやすい色を選ぶことは難しくなる．また，仮に色覚異常の人々の区別可能な色を選んでも，それらは彼らにとっては類似色である可能性が高い[1], [2]．空間的に離れた位置にある類似色を区別することは，正常色覚の人であっても困難である．以上のように，色で塗り分けられたイメージから，高齢者や色覚異常の人々が必要な情報を得るのは困難な場合が多い．現在，色覚異常の人々にとって区別が難しい色を区別しやすい色に変換するソフトウエアDaltonize[3]が公開されている．また，同じ目的の色変換手法も提案されている[4]．しかし，これらを使用しても，色覚異常の人々が一度に多くの色を正確に区別することは不可能である．

このような問題の解決方法の１つは，イメージ中の異なる色で塗られた領域に，それぞれ異なるハッチパターンを描画することである．我々は，現在投稿中の論文[5]において，そのようなハッチパターンの描画を効率良く行う方法を提案した．今回，その方法を使用して，ウェブブラウザに表示されているページに，色に応じて異なるハッチパターンを描画するプラグインを開発した．このプラグインにより，ページ中の地図イメージなどの色の区別を補助する目的は十分に果たせることを確認した．また，ハッチパターンの描画に要する時間は，Celeron(Coppermine) 700MHz，128MB RAMのコンピュータを使用し，1024×768のスクリーンでブラウザを最大化した状態でも約１秒程度であり，現在，広く使われているコンピュータであればストレスなく実行できる．

一方，前記のハッチパターンの描画方法における以下のような潜在的な問題点は，このプラグインでもそのまま残されている．

(1) イメージの内容によって，自動的に描画するハッチパターンのサイズ（目の細かさ）を見やすいサイズに変えることはできない．例えば，目の細かいパターンでは，パターン間の区別が難しくなる．逆に，目の粗いパターンを面積の小さい領域に描画すると，パターン自体が認識できなくなる．

(2) ある情報を示す領域が複数の類似色を含む場合（例えば，路線図などがJPEGで保存されている場合など）は，正確にハッチパターンを描画できない場合がある．

ただし，これらの問題は，使用するハッチパターンのサイズや，区別すべき色のおおよその数などをユーザが実行時に指定することにより回避することはできる．しかし，これらのほかに，ユーザの操作では回避できない新たな問題が明らかになった．

前記のハッチパターンの描画方法では，入力されるイメージはすべてハッチパターンの描画対象の領域として処理を行っている．しかし，ブラウザに表示されているページには，色で塗り分けられた図やグラフ以外にも，文字が書かれている領域や風景写真イメージなどのハッチパターンの描画を必要としない領域が含まれている場合が多い．当然，このようなページを入力イメージとした場合，文字の背景にパターンを描画して文字が読みにくくなったり，写真イメージにノイズを加えたような結果になってしまったりということが起こる．この問題を解決するためには，ハッチパターンの描画処理を行う前に，ページ全体のイメージから描画が不要な

♠ 非会員　（株）両備システムソリューションズ　shimoda@ryobi-sol.co.jp
♦ 正会員　岡山県立大学情報工学部　yokota@c.oka-pu.ac.jp

領域を除外する必要がある．本論文で我々は，このハッチパターンの描画が不要な領域を除外する方法について考察した．

2. プラグインの概要

今回，我々が開発したのは，ウインドウズ版インターネットエクスプローラのプラグインであり，インストールするとエクスプローラのツールバーとして表示できるようになる．一般的なツールバーの開発方法は，マイクロソフト社のウェブサイト[6]で公開されている．

2.1 処理手順

我々のツールバーは１つのボタンのみを持ち，そのボタンが押されるとエクスプローラウインドウ内のクライアント領域（ウインドウ内のタイトルバー，メニュー，及びウインドウ枠を除いた領域）のイメージをメモリ内に取得する．このイメージは，R，G，Bの値がそれぞれ８ビットで表されたフルカラーイメージである．そして，そのメモリ内のイメージに[5]で提案した方法によりハッチパターンを描画した後，それを元のクライアント領域に描画する．このとき，ハッチパターンを描画したイメージが表示されていることをユーザに示すために，カーソルの形状を通常の矢印とは異なるものに変える．その後，マウスクリックなどのイベントが発生すると，エクスプローラウインドウに再描画を要求するメッセージを送り，カーソルの形状を元に戻した後，処理を終了する．ユーザは，ツールバーのボタンを押してハッチパターンを描画させ，ウインドウ内のどこかをクリックすることにより表示を元に戻すことができることになる．

2.2 ハッチパターンの描画方法

我々のプラグインで使用している，[5]で提案したハッチパターンの描画方法の概要を以下に示す．

2.2.1 ハッチパターンの定義

処理の前に，あらかじめ描画するすべてのハッチパターンを定義しておく．例えば，図1(a)に示した４ピクセル間隔の斜線のパターンは，描画領域に図1(b)のような４×４ピクセルのイメージを縦横に並べたものとなる．図1(b)の黒色の正方形はパターンを描画するピクセル，白色の正方形は元のイメージがそのまま見えるピクセルを示す．以下，このパターンを構成する最小イメージを単位パターンと呼ぶ．このような単位パターンを，必要な数だけ定義しておく（それぞれが幅，高さ，及び図1(b)に相当するON/OFF情報を持つ）．

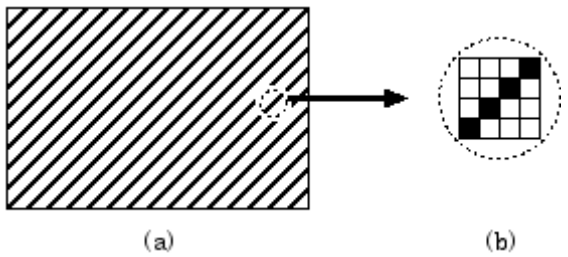

図１　ハッチパターンの例

Fig.1　An example of hatch pattern.

2.2.2 色とハッチパターンの対応付け

Median Cut Algorithm[7]を使用し，入力されたフルカラーイメージからインデックスカラーイメージを生成する．この処理により，カラーマップと各ピクセルの色番号が記録されたピクセルデータが作成される．ただし，カラーマップの各エントリの情報として，R，G，B 値に加え，処理の過程で得られるその色を参照するピクセル数（以下，参照数とする）も保持しておく．また，カラーマップの最大エントリ数，すなわち量子化後の最大色数は，実行時に任意の値を指定できる．そして，カラーマップの参照数を降順にソートし，使用頻度の高い色（エントリ）から順に単位パターンを割り当てる．

2.2.3 ハッチパターンの描画

前記のピクセルデータを走査し，各ピクセルについて次の処理を行う．以下，処理対象ピクセルの入力イメージにおける位置（行番号，列番号）を（r, c）とする．ただし，r, c はともに0から始まるものとする．処理対象ピクセルに対応するカラーマップのエントリ情報から，その色に対応付けされた単位パターンの情報を取得する．（その色に単位パターンが対応付けられていない場合は，処理対象ピクセルはハッチパターンの描画対象とせず，次のピクセルを処理する．）ここで，取得した単位パターンの幅を Wp, 高さを Hp とする．そして，取得した単位パターンを入力イメージ全体に並べて描画すると仮定した場合，処理対象ピクセル（r, c）には単位パターン内の位置（$r \bmod Hp$, $c \bmod Wp$）のピクセルが対応する．したがって，この単位パターン内の位置（$r \bmod Hp$, $c \bmod Wp$）のピクセルが，図1(b)に示した黒あるいは白色の正方形に相当するかによって，処理対象ピクセルの位置にハッチパターンの一部を描画すべきか否かが判断できる．そして，描画すべきと判断した場合は，処理対象ピクセルのR，G，B値をハッチパターンの描画色に変更する．以上の処理を入力イメージの全ピクセルについて行えば，すべてのハッチパターンの描画は終る．

この方法により，全ピクセルの２度の走査（インデックスカラーイメージを生成するときと，ハッチパターンを描画するときにそれぞれ１度）で，効率良くハッチパターンの描画が行える．

2.2.4 細い線への描画の抑止

イメージ中の文字や境界線などの細い線にハッチパターンを描画した場合，描画したパターンが認識できないだけでなく，それらの細い線を部分的に消してしまうような結果になる．これを回避するため，前記のハッチパターンの描画の際に次のような処理を行っている．処理対象ピクセルに対応するカラーマップのエントリ情報を取得するとき，ピクセルデータに記録された色番号をそのまま使用するのではなく，周囲の８ピクセルを合わせた合計９ピクセル中の５ピクセル以上が同じ色番号を持っている場合のみ，それを処理対象ピクセルの色番号とする．同じ色番号を持つピクセルが５ピクセル未満の場合は，処理対象ピクセルはハッチパターンの描画対象としない．この処理には，ノイズの影響を軽減する効果もある．

2.2.5 オプション機能

入力イメージ中の最も参照数の多い色，または白，黒，灰色などの無彩色へのハッチパターンの描画を抑止することができる．これらの色は地図やグラフのイメージにおいては背景色である場合が多く，通常，背景にはハッチパターンを描画しない方が良い．この機能は，色とハッチパターンの対応付けの際に，カラーマップ中の参照数の最も多い色や無彩

色のエントリに単位パターンを割り当てないことで実現できる．

3. 実行結果と問題点

プラグインの実行結果を示すために作成したテスト用ウェブページを図 2 に示す．ブラウザに表示させたときのクライアント領域のイメージを切り出したもので，これがプラグインの入力イメージとなる．このページの構成は，以下のようになっている．上部のタイトルは白色の文字で記述しており，その背景は風景写真イメージから切り出した深緑色の海面のイメージである．左部のインデックスは，深緑色を背景色として指定し，白色と薄黄色の文字が書かれている．残りの部分の背景は白色で，左側に色で塗り分けられた地図イメージ（PNG），右側に風景写真イメージ（JPEG）があり，それらの下部に黒色の文字が書かれている．そして，右端と下端にブラウザのスクロールバーがある．このページの中で，ハッチパターンを描画する必要があるのは，中央部の地図イメージだけである．

図 2 のページをブラウザに表示させ，プラグインを実行させた結果が図 3 である．ただし，無彩色へのハッチパターンの描画は抑止した．図 3 では少し分かりにくいが，色の区別が必要な地図イメージには正しくハッチパターンが描画されており，本来の目的は十分に果たせている．しかし，ハッチパターンを描画すべきではない，以下のような箇所に描画が行われている．

(1) 上部のタイトルの背景イメージ
(2) 右端と下端のスクロールバー
(3) 中央部右側の風景写真イメージ
(4) 左部のインデックスの文字の背景

ユーザは，マウスボタンのクリックだけでハッチパターンの描画，オリジナルイメージの再描画が瞬時に実行できる．したがって，色の区別が必要な箇所に常に正しくハッチパターンが描画されるのであれば，不要な箇所にハッチパターンが描画され，見やすさが損なわれたとしても，大きな問題ではないかもしれない．しかし，風景写真イメージなどに多くの異なる色が使用されている場合は，ハッチパターンの描画処理の中のインデックスカラーイメージを生成する際に，地図中の本来は異なる色が同一色に量子化されてしまう可能性がある．この場合，本来は区別すべき異なる色の領域に，同じハッチパターンが描画される．また，異なる色が同一色に量子化されること自体は，カラーマップの最大エントリ数に大きな値を指定することで回避できるが，そのエントリ数がハッチパターンの定義数（現在は 128）を超えた場合は区別が必要な色にハッチパターンが描画されない可能性が生じる．ハッチパターンはプラグイン実装時に人手により定義しておく必要があり，またそれぞれが容易に識別できるパターンでなければならないので，定義可能な数には限界がある．

以上のことから，ブラウザに表示されているページ内の色の区別が必要な領域に，色に応じて異なるハッチパターンを常に正しく描画することを考えた場合，入力イメージ中の風景写真など色の区別が不要な領域を検出して除外しておくことが不可欠である．

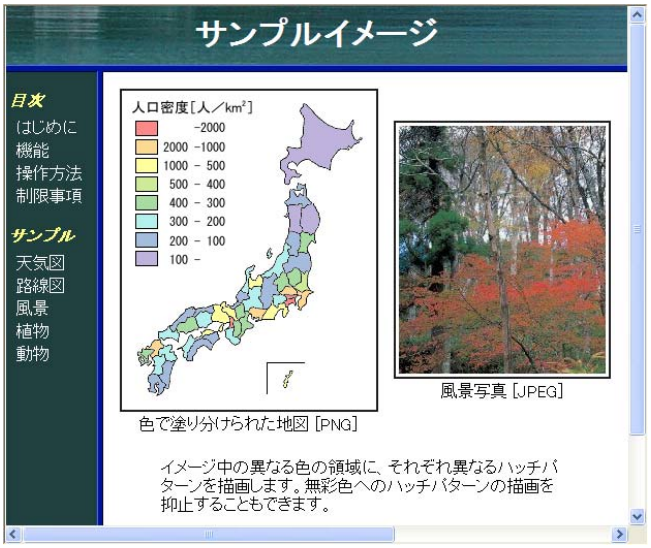

図 2　テスト用ウェブページ

Fig.2　Test web page.

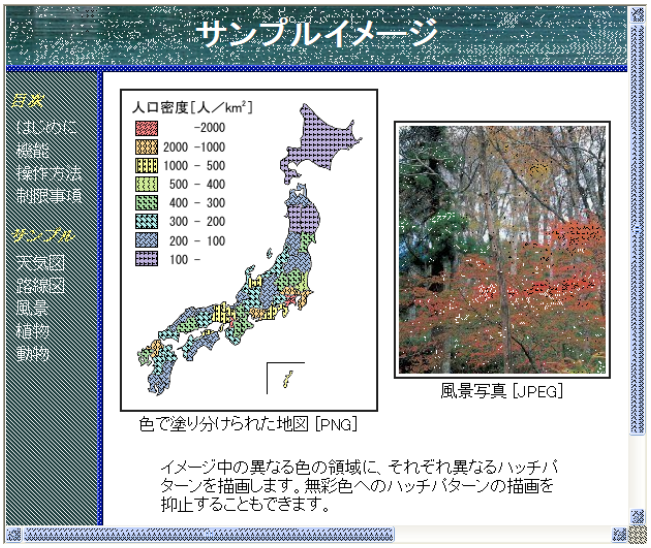

図 3　プラグインの実行結果

Fig.3　Application result of our plug-in.

4. 解決方法に関する考察

4.1 ハッチパターン描画対象領域の限定

前述のように，図 3 では，色の区別が不要な 4 つの領域にハッチパターンが描画されている．しかし，(4)の文字の背景は単一の色で塗られた領域であり，多くの類似色で構成される他の 3 つの領域とは性質が異なる．このような有彩色の背景に文字が書かれた領域は，円グラフなどにも現れる場合があり，その領域の色を区別する必要があるか否かは簡単には判断できない．また，背景に使われている色は 1 色であるため，前記のような色の区別が必要な領域へのハッチパターンの描画に悪影響を与える可能性も極めて低い．したがって，ここでは風景写真などのように多くの類似色が使われた，色

の区別が不要な領域（以下，問題領域と呼ぶ）を除外する方法を考えてみる．

4.1.1 エッジ要素抽出処理の利用

地図イメージと風景写真イメージの特徴を比較したときの違いの１つとして，前者は大多数のピクセルが隣接するピクセルと同色であるのに対し，後者では隣接ピクセルが同色である場合は少ないことがあげられる．このことから，画像処理で広く知られているエッジ要素抽出処理の利用を検討してみた．隣接ピクセル間の色の変化が大きい部分をエッジ要素として抽出する処理であれば，その過程で，少しでも色の変化がある部分を抽出することもできるはずである．

エッジ要素を抽出するための最も簡単な方法は，局所オペレータを用いる方法である．このオペレータには，隣接ピクセル間の濃度勾配に基づく差分型の Sobel, Prewitt オペレータ，テンプレート型の Robinson，Prewitt，Kirsch オペレータ，及び２次微分に基づく Laplacian オペレータなどがある[8]．また，入力イメージにウェーブレット変換を適用し，算出されたウェーブレット展開係数を参照することによってもエッジを検出できる[9]．我々の目的は，風景写真などの多くの類似色が使われている領域，すなわち隣接ピクセル間の色が異なる領域を抽出して除外することである．したがって，前記のオペレータにより濃度勾配を持つと判断されるピクセル，またウェーブレット変換により高周波成分を持つと判断されるピクセルをハッチパターンの描画対象から除外してみる．

まず，図 2 のテスト用ウェブページのイメージにおいて，前記のオペレータ，及びウェーブレット変換を適用し，描画対象から除外すべきと判断されるピクセルを確認してみた．図 4 は，前記の７種類（６種類の局所オペレータとウェーブレット変換）の中で最も良い結果が得られたウェーブレット変換により高周波成分を持つと判断されたピクセル（黒色）を示したものである．そして，それらのピクセルを実際に除外してハッチパターンの描画を実行させた結果が図 5 である．

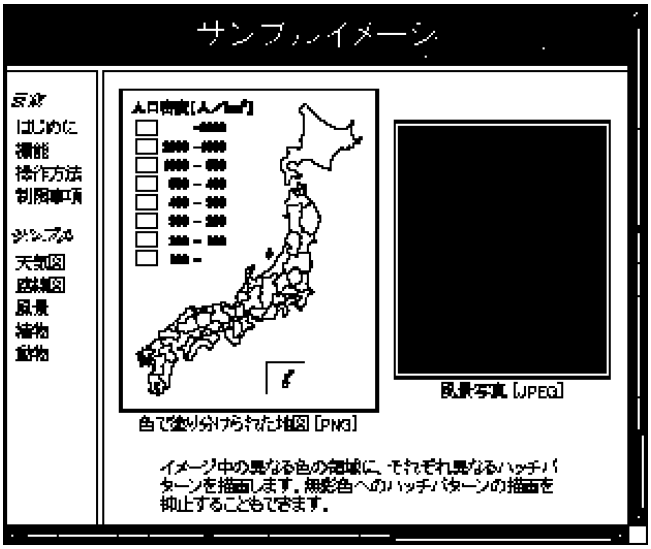

図 4　ウェーブレット変換による高周波成分を持つ領域の検出

Fig.4　Detection result of regions with high-frequency components by using wavelet transform.

ただし，ここでのウェーブレット変換では，ハールウェーブレットを使用して２次元ウェーブレット変換を１度だけ行い，イメージ内の各ピクセルについて，対応する３つのウェーブレット展開係数のいずれかが 0 ではないとき高周波成分を持つと判断した．図 4 を見ると，ほぼ完全に問題領域は抽出できている．また，その問題領域を除外してハッチパターンを描画させた図 5 では，ほぼ期待通りの結果が得られている．ただし，地図中の県境に近い部分が，わずかではあるが必要以上に描画対象から除外されている．

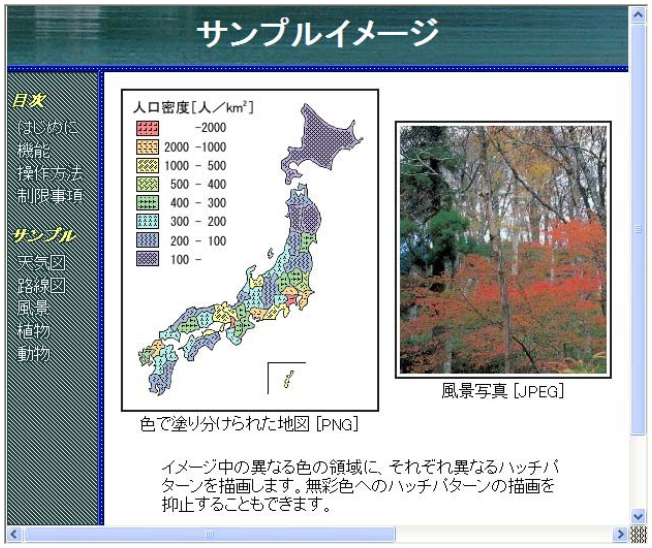

図 5　高周波成分を持つ領域を除外したハッチパターンの描画

Fig.5　Drawing hatch patterns after filtering out regions with high-frequency components.

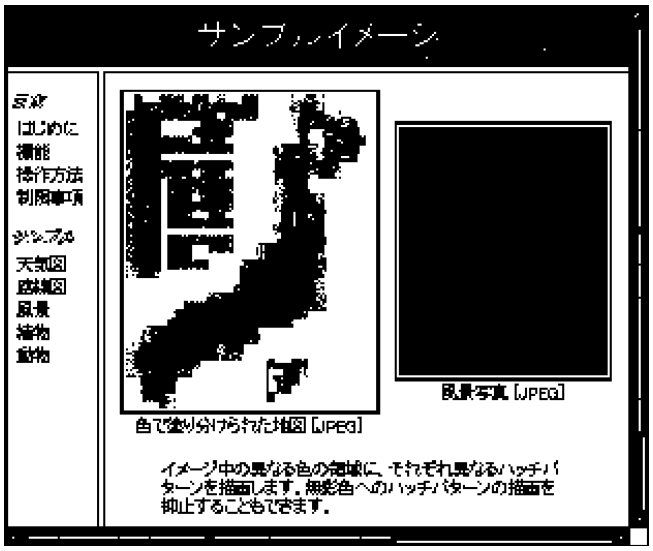

図 6　JPEG の図を含んだイメージの高周波成分を持つ領域の検出

Fig.6　Detection result of regions with high-frequency components in the image contains JPEG map.

しかし，この方法では，地図イメージが JPEG であった場合，それが描画対象から除外されてしまうことが予想される．そこで，テスト用ウェブページ内の地図イメージを JPEG で保存したものに置き換え，ウェーブレット変換により描画対象から除外すべきと判断されるピクセルを確認してみた（図 6）．図 6 を見ると，地図イメージの色が塗られた領域はほとんどすべて除外すべきと判断されている．JPEG イメージにハッチパターンを描画する必要がある場合には，エッジ要素抽出処理の利用は難しい．

4.1.2 Laplacian オペレータの利用

図 7 は，図 6 で使用した，地図イメージを JPEG に置き換えたページのイメージに，前記の Laplacian オペレータを適用し，計算結果が正になるピクセルのみを黒色で描画したものである．具体的には，入力イメージにおける行番号 r, 列番号 c のピクセルの明度を $b(r, c)$ としたときに，以下の式を満たすピクセルである．

$$\nabla^2 b(r,c) = b(r-1,c) + b(r,c-1) + b(r,c+1) + b(r+1,c) - 4b(r,c) > 0 \qquad (1)$$

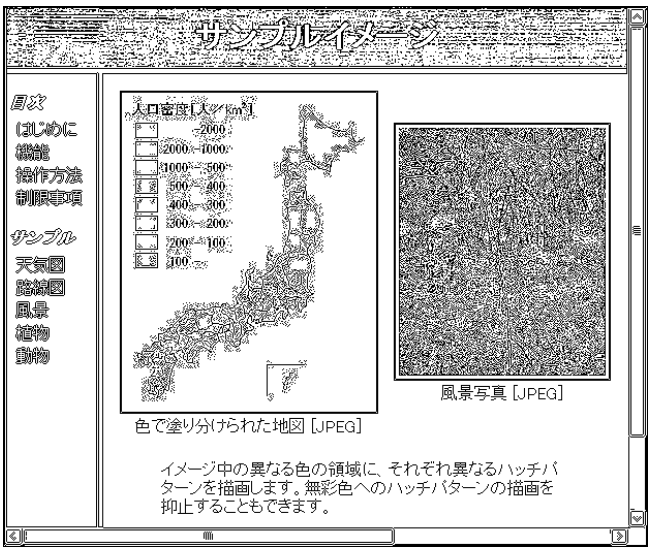

図 7　ラプラシアンオペレータ適用結果が正となるピクセル

Fig.7　The pixels whose value derived from applying Laplacian operator becomes positive.

図 7 を見ると，風景写真やタイトルの背景の領域では，黒色のピクセルは元のイメージの内容がわずかに推測できる程度の不規則な並びになっている．しかし，JPEG の地図イメージの領域では，文字や県境などのエッジ要素と，それらの付近のノイズを示すような並びになっている．そして，これらのノイズは，その付近のエッジ要素の向きには依存せず，大部分が短い斜め方向の線分によって構成されているように見える．図 8 は，図 7 の地図イメージの部分を拡大したものである．ただし，エッジ要素とノイズを区別しやすくするため，元の地図イメージの文字と県境を灰色で図 7 に上書きした．いくつかの地図やグラフの JPEG イメージに対して前記と同じ処理を行ってみたが，いずれの場合も同様の特徴を示すノイズが見られた．これらのノイズは，何らかの規則性を持っているように思われる．仮に，前記のような Laplacian オペレータによる処理を行って，JPEG の地図イメージなどに風景写真イメージなどでは現れない，規則性を持ったノイズが現れるとすると，それらをテクスチャとみなして領域を抽出できる可能性がある[10]，[11]．抽出した領域はハッチパターンの描画が必要な領域なので，前項で抽出した問題領域から削除すれば良いことになる．

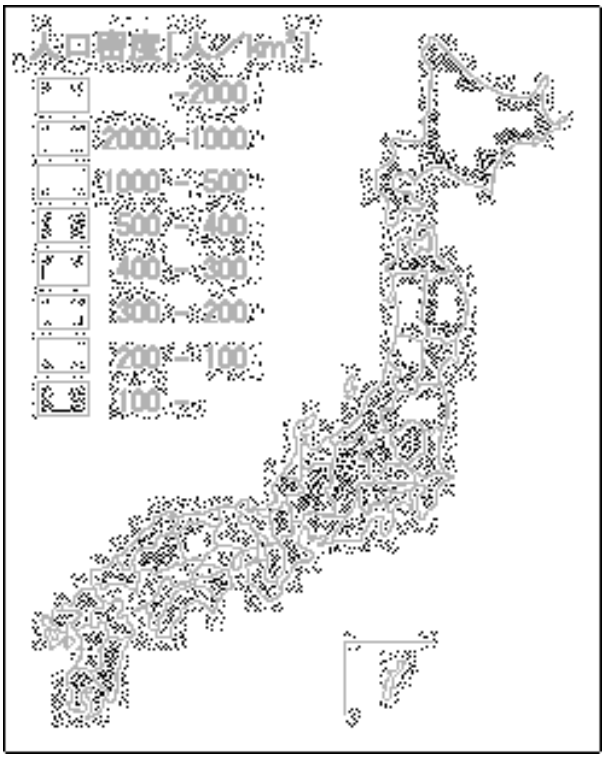

図 8　ラプラシアンオペレータ適用結果が正となるピクセル（拡大図）

Fig.8　Enlarged view of the pixels whose value derived from applying Laplacian operator becomes positive.

しかし，この方法が実現可能か否かを判断するためには，式 (1) により抽出したピクセルと JPEG のエンコード方法との関連性，JPEG の地図イメージに現れるノイズの規則性，及びテクスチャ領域の抽出方法などについて更なる研究が必要である．

4.2 ハッチパターン描画対象色の限定

区別が必要な色にハッチパターンが描画されない可能性があるという問題を解決するために，前項では問題領域を描画処理の前に除外することを考えた．一方，描画対象の色を限定することによっても，この問題の発生する可能性は抑えられる．現在は，地図やグラフなどのイメージ中の区別する必要のあるすべての色をハッチパターンの描画対象としていたが，その中にはハッチパターンがなくても容易に識別できる色も含まれている．したがって，これらの色は描画対象から除外し，識別が容易ではないと判断される色のみを描画対象とすることで，使用するハッチパターンの数を抑えることができる．その結果，カラーマップの最大エントリ数に大きな値を指定しても，区別が必要な色にハッチパターンが描

画されない可能性は極めて低くなることが予想される．

これを実現するためには，前記のハッチパターンの描画方法の中の，色とハッチパターンの対応付けの際に，次の処理を実行すれば良い．カラーマップ中の各色について，ユーザの色覚特性に応じたシミュレーションを実行し，シミュレーション後の各色間の均等色空間（Lab や Luv など）における色差を計算する．そして，色差が一定の値以下となる別の色が存在する場合のみ，その色に単位パターンを割り当てる．ユーザの色覚特性については，年齢別の高齢者の色覚[12]，及び３タイプの二色型色覚[1]，[2]のシミュレーションが可能であり，それらの中から選択されたタイプを，コンピュータのログオンユーザ毎にプロファイルとして保持しておくことが可能である．

5. まとめ

我々は，ウェブブラウザに表示されているページに，色に応じて異なるハッチパターンを描画するプラグインを開発した．これにより，高齢者や色覚異常の人々が，ウェブページ内の色で塗り分けられた地図やグラフのイメージから必要な情報を得ることを補助できると思われる．しかし，現時点では，ページ内に風景写真などのイメージが含まれている場合や，それらとともに JPEG の地図やグラフが含まれている場合には，必要な領域にハッチパターンが正しく描画できない可能性がある．今後，更に研究を進め，これらの問題を解決していく必要がある．

また，データベースのマルチメディア情報を検索するシステムでは，そのデータの多様性から，検索結果を単純な表などでユーザに提示することは稀で，様々な工夫が為された提示方法をとる場合が多い．当然，その中では特定の属性値や分類結果などが色によって表現される可能性が高くなり，そのような場合にも我々のプラグインは有効に利用できると思われる．

我々は，今回，開発したプラグインを含んだ色覚異常の人々のための補助ソフトウエアを，以下のサイトで公開している．

http://www.ryobi-sol.co.jp/visolve/

現在，そのダウンロード数は 20,000 件を超えており，多くの方々から良い評価をいただいている．

［文献］

[1] Brettel, H., Viénot, F., and Mollon, J.D. : “Computerized simulation of color appearance for dichromats", the Optical Society of America A, vol.14, no.10, pp.2647-2655 (1997).
[2] Viénot, F., Brettel, H., and Mollon, J.D. : “Digital video colourmaps for checking the legibility of displays by dichromats", COLOR research and application, vol.24, no.4, pp.243-252 (1999).
[3] http://www.vischeck.com/daltonize/
[4] 目黒光彦，高橋知紘，古閑敏夫: “混同色線理論と色覚モデルに基づくカラー画像からの弁別困難色の検出と弁別しやすい色への変換", 電子情報通信学会技術研究報告. IE，画像工学, vol.104, no.367, pp.19-24 (2004).
[5] Shimoda, M., Okabe, K., and Yokota, K. : “An Efficient Method for Drawing Different Hatch Patterns According to Color", (2008).
[6] http://msdn.microsoft.com/en-us/library/cc144099(VS.85).aspx
[7] Heckbert, P.S.: “Color image quantization for frame buffer display”, ACM SIGGRAPH '82 Proceedings, vol.16, no.3, pp.297-307 (1982).
[8] 高木幹雄，下田陽久: “画像処理ハンドブック”，東京大学出版会，東京（1991).
[9] 中野宏毅，山本鎭男，吉田靖夫: “ウェーブレットによる信号処理と画像処理”，共立出版株式会社，東京（1999).
[10] 狩野芳正，大町真一郎，阿曽弘具: “特徴選択によるテクスチャ画像の教師なし領域分割”，電子情報通信学会論文誌(D-II), vol.J86-D-II, no.7, pp.988-995 (2003).
[11] 田中秀郎，吉田靖夫，深見公彦，中野宏毅: “ガボールフィルタの振幅及び位相情報を用いたテクスチャ画像の領域分割”，電子情報通信学会論文誌(D-II)，vol.J84-D-II, no.12, pp.2565-2576 (2001).
[12] Pokorny, J., Smith, V.C., and Lutze, M. : “Aging of the human lens", Applied Optics, vol.26, no.8, pp.1437-1440 (1987).

下田 雅彦 Masahiko SHIMODA
（株）両備システムソリューションズ所属．岡山県立大学情報系工学研究科博士後期課程在学中．1989 神戸大学工学部電子工学科卒業．データベースシステム，画像処理システムの研究・開発に従事．ACM SIGMOD，SIGGRAPH，各会員．電子情報通信学会学生会員．

横田 一正 Kazumasa YOKOTA
岡山県立大学情報工学部情報通信工学科教授．データベースシステムの研究・開発に従事．日本データベース学会，情報処理学会，電子情報通信学会，人工知能学会， ACM，IEEE 等正会員．