統計数理 第34巻 第1号（1986）

コンピュータ・グラフィクスのための カラー・モデル変換プログラム

統計数理研究所　大　隅　　昇

（1986年2月　受付）

1. コンピュータ・グラフィクスとカラー・モデル

統計情報を視覚的に伝達する手段として，“グラフィカル表現”や“統計的グラフ解析法”の研究報告が多くみられる．また，こうした統計手法の支援ツールとしてハードウェア，ソフトウェア両面の進展には著しいものがある．たとえば，ハードウェアの面ではXYプロッタ，カラー・プロッタ，グラフィクス・プリンタ，グラフィクス・ディスプレイ，イメージ・リーダなどの性能は一昔前に比べて格段に向上し，しかも比較的廉価で購入できるようになってきている．一方，ソフトウェアの面でも，ハードウェアの性能の向上にあわせて，汎用的でかつ規模の大きいグラフィクス・システムが次々に登場している（たとえば，ERDAS, MIMS, ODYSSEY）．さらに，SAS, SPSS-Xなどの統計システムもグラフィクス機能の強化を図る傾向にある．この他，DATAPLOT, STATGRAPHなど統計グラフ専用のソフトウェアも無数に現れている．

くわえて最近は，データ解析の各種分析結果を色彩により表現するという試みもあり，これにあわせてカラー・グラフィクスの利用法が注目されだした（Becker (1983), Nicholson (1983), Trumbo (1981), Wainer (1980)）．しかし，実際に色彩の利用がデータ解析にどのように有用であるかという点で慎重な議論がさらに必要であるように思われる．単に円グラフや棒グラフに彩色を行うという以上の何かがあって初めて，色彩を用いるデータ解析が意味をもつものと思われるが，この点での議論がいまだ十分ではない．現状の色彩科学の知識では，色彩の利用にあたって様々の問題が山積しているといってよいが，主なものとして次のようなことが挙げられよう．

ハードウェアの側面から

1)　価格がまだ高い（色彩が自由に操作できる機器はかなり高価である）

2)　利用可能な色彩の範囲（色相の種類）が限られている．あるいは発色数は多くとも，実際に利用できる色の数に制限がある．

3)　使用するカラーメディアによって，色域の条件が異なる．たとえば，ディスプレィ上での発色，スライドやカラー印刷などのハードコピーとしての記録，カラープロッタへの出力など，色彩処理の技術がそれぞれ異なる．したがって，これらのカラー・メディア間の対応が十分でない．また，色彩処理に高度の技術を必要とすることが多く，身近なものとして利用することが難しい．

ソフトウェアの側面から

1)　色彩操作のうえでユーザフレンドリなソフトウェアが少ない，または，あっても非常に

高価である．しかも，ユーザが必要とする色彩を自由に表示するために適したソフトウェアを利用できる環境が整っているとはいいがたい．これは，後述するカラーモデルにもとづくソフトウェアが少ないことによるものと思われる．

2） 機種の依存度が高い．

3） アルゴリズム等の標準化が十分ではない．

すでに，CORE システム，GKS (Graphical Kernel System) などがあるが，色彩操作についての規定は未だ十分とはいえない．

ところで，グラッフィクス・モニタ上で色彩を用いる場合の重要な要素として，色彩操作の理論的根拠をどこにおくかということがある．現在，カラー・グラフィクスを用いる場合のよりどころとされている考え方は，“カラー・システム（表色系）”または，“カラー・モデル（色彩モデル）”といわれるものである．

もちろん，色覚の心理学的あるいは生理学的・生物学的見地からの研究は古くから行われており，様々の色覚モデルやカラー・システムの提案がある（たとえば，マンセルのカラー・システム，オストワルトのモデル（朝日新聞社編（1983））．しかしこうした，“色覚のカラー・システム”の研究には未解決の問題が依然として残されており，コンピュータ・グラフィクスに必要とされる“アルゴリズム的カラー・システム”との間には大きな差異があるといわれている．現状は，こうした事実を踏まえたうえでなお，カラー・グラフィクスを操作するために有用なソフトウェアのあり方を模索するという段階にあるといえよう．

この報告の目的は，データ解析において色彩を利用する場合に有用と思われる，カラー・モデルに依拠したアルゴリズムを前提とするマイクロコンピュータ向きの実用的なプログラムを提供することにある．

2. カラー・モデルとその変換ソフトウェア

カラー・モデルとハードウェアの形態や性能とは大いに関係がある．最近は，従来主流であったストレージ・チューブ型のディスプレイに代わって，高品位・高精細度のラスター・グラフィクス・ディスプレイが比較的廉価で購入できるようになり，これにともなって，カラー・モデルの考え方にも変化がみられるようになってきた．また，多くのマイクロコンピュータではラスタ・スキャン方式のモニタを使用している．

こうしたラスター・スキャン型のグラフィクスを操作するうえで有用な（ハードウェア向きの）カラー・モデルと，利用者が色彩操作を行ううえで都合のよい，あるいは，色彩表現を必要とするデータ解析のプログラム開発に適したカラー・モデルとがある．通常利用されている代表的なカラー・モデルについてこの関係を要約すると図1のようになる（Enderle et al. (1984), Foley, et al. (1982)）．

ハードウェア対応モデル，たとえば，RGB モデルは，3原色（R : red, G : green, B : blue）とその色域（ディスプレイが発色可能な色彩の範囲），発色チューブの蛍光面の特性などを工学的な見地から扱う際に適したモデルである（Beatty (1983), Foley (1982)）．一方，人の色覚に対応させやすいソフトウェア対応モデルは，色覚をある記述的な（あるいは人工的な）モデルに対応させるものであり，いわゆる“色の3要素”（色相（H）: Hue, 明度（L）: Lightness またはV値 : Value, 飽和度または彩度（S）: Saturation）により規定される．従ってここに，ハードウェアの操作面で優れている RGB モデルと，ソフトウェアの開発に適した，あるいは人の色覚に対応させやすい，HSV, HLS モデルとの間を関連づけるカラー・モデル間の変換のアルゴ

リズム（色彩空間の変換アルゴリズム）を配慮することとそれを具体的にプログラム化することが必要となる．これを模式的に示すと次の図2のようになる．

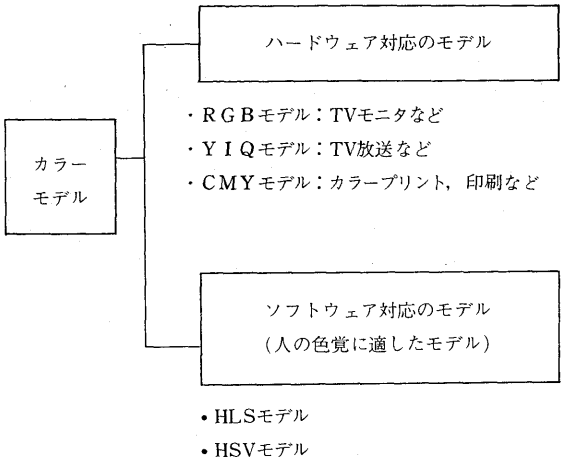


図1.

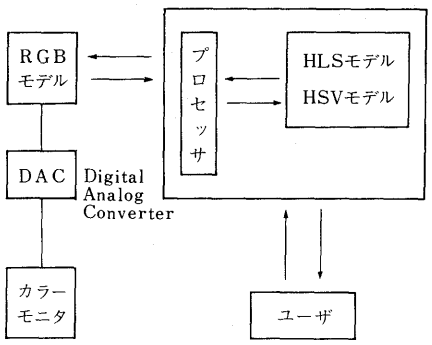


図2.

たとえば，HLSモデルからRGBモデルへの変換を考える．これは，カラーシステムとして図3のような双円錐形の色立体（Double-cone model）を考え，これに色の3要素を対応させる(HSVモデルの場合には，図3の下半分の円錐を考える．また，明度をV値として，0≦v≦1として考える)．つまり明度（L），飽和度（S），色相（H）をそれぞれ与える．

図3は，垂直方向が明度に対応し，上端が白（L＝1），下端が黒（L＝0）に対応する．また，上から下に向かって中心を貫く線上(S＝0)が，グレイレベル，すなわち，無彩色域を示す．この中心線から円錐の表面に向かって遠ざかるほど飽和度が高くなり，この方向の変化は色調

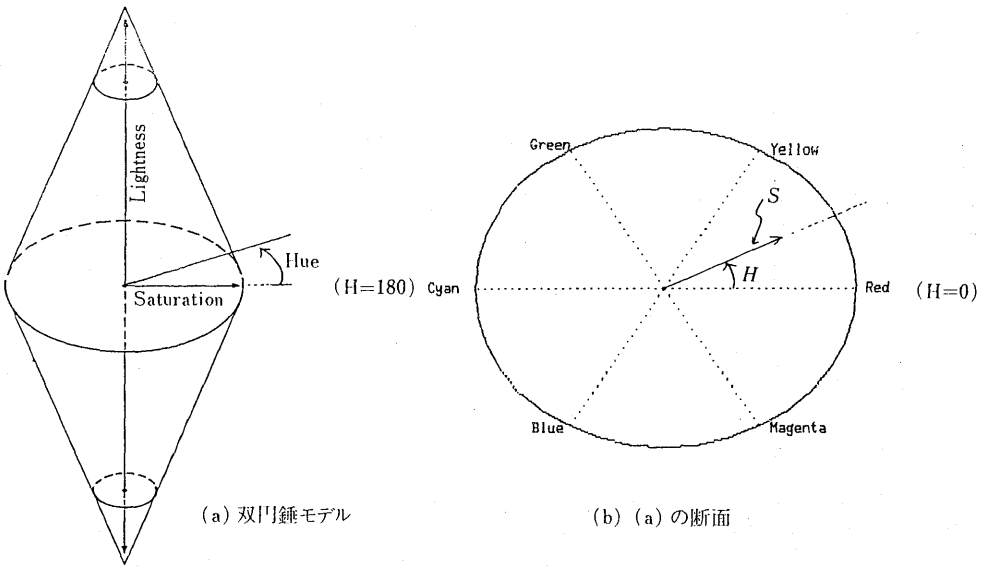


図3．HSLカラー・モデル

(tint) を変えることに相当する．また，色相（H）と飽和度（S）を固定して，円錐の中を，上または下に移動することは，色の明暗を変えることになる．さらに，ある明度で，水平に切断した断面上で(図3の(b))，中心から角度0度を赤とし，反時計方向に移動する角度に対応して色相（H）が定まる．なお，機種によっては，たとえばTektronixのように，基線（H＝0）を赤ではなく青にとるものもある．

こうした，カラー・モデルを考えると，利用者が望む色，たとえば，「明るい薄紫」とか「赤みがかった茶色」などをカラー・モデル（HLS）上に対応させることができて，人の直感に対応させやすいということがある．すなわち，RGBモデルで，赤，緑，青をそれぞれどのような割合で混ぜると何色になるのか，といったことを考える煩わしさがない．

実際の変換手順としては，まず指定したH, L, Sの各値（h, l, s）に対するRGB値，(r, g, b) をそれぞれ算出してこれをディスプレイの機能に合わせてRGBの強度（intensity）に変換する．次にこれを，電気信号としてモニタに送ると発色する．たとえば，4ビットの強度段階をもつモニタの場合，$0 \leqq r, g, b \leqq 15$ となる整数を対応させる．

一方，RGB系からHLS系への変換の場合，たとえば，モニタの画面上の色彩映像のある位置（画素）に置かれた色は“何色”であるかを知るには，画素上からひろった強度（r, g, b）値がHLSモデル上でどの位置にあるかを知ることが必要になる．

ところで，これらの変換アルゴリズムの提案や簡単な紹介は断片的にはあっても，実際にすぐに利用できるようなプログラムが報告された例は少ないようである．また，カラー・モデルをグラフィクス・ディスプレイ表示した例はあってもそこで使われたプログラムは多くの場合非公開であったり，言語の制約から，かならずしも移植が容易であるとはかぎらない．さらに大型コンピュータやグラフィクス専用の高価なコンピュータを必要とすることが多い．本報告の目的はこの点を補うために，カラー・モデルの変換の基本的な機能について，フォートラン言語を用いた汎用的な，またマイクロコンピュータ向きの変換プログラムを作成し利用に供することにある．とにかく，こうした変換プロセスを可能にするプログラムがあってはじめて，データ解析において色彩を用いる意義や価値が生まれるといってよい．

3. フォートラン・プログラムの概略

初めに指摘したように，カラー・グラフィクスを利用する場合，そのハードウェアの制約が無視できないということがある．プログラムを作成する際にこの事実を十分に念頭においてとりかかる必要があるが，それでもなお，できるかぎり応用範囲の広いプログラムであることが望ましい．この点を配慮して，何通りかのプログラムを用意した．また，プログラムの開発にあたって，できるだけ小規模のマイクロコンピュータでも利用できてしかも改編や移植が容易であるように，次の計算機環境のもとに開発を進めた．

本　　体：DEC社，MINC−11/23
　　（PDP−11/23相当，主記憶容量は128 KB）
　　これに，グラフィクス・ビデオ・ディスプレイ（VT-125）が接続
カラーモニタ：NEXUS−5800相当
　　解像度＝256×240（画素）
　　色彩強度＝R（赤），G（緑），B（青）の各色4ビット
　　（したがって，各色0〜15の16段階）
使用言語：FORTRAN言語（OSはRT-11）

以上の環境下で開発したプログラムの概要を次に述べる．

(1) RGBCNV

RGB 系から HLS, HSV 系へ，あるいは，その逆の変換を行うプログラムである．

RGB を(r, g, b)値として指定すると(ここでは，0～15 の範囲の整数)，HLS あるいは HSV カラー・モデル上の位置，(h, l, s)，または (h, s, v) が算出される．あるいは，その逆変換が行われる．ここで，$0 \leqq l, v, s \leqq 1$, $0 \leqq h < 360$ とするが，これは機種にあわせて適当に変えることができる．これが，カラー・モデル間の変換を行う基本プログラムである．

(2) HLSRGB

HLS モデルから RGB モデルへの変換を行う．このとき，ディスプレイ上に，HLS モデルのグラッフィクス表示があるので，カーソルを使ってまず明度（L）を，次に色相（H），飽和度（S）を指定すると，やはり画面上に変換された RGB 値が表示される．

(3) HLSNEX

(2) の機能に加えて，得られた RGB 値を，カラーモニタ・ディスプレイにカラー・スワッチ（色相見本）として表示する．

(4) RGBHLS

RGB モデルから HLS モデル上の位置を求めるところは (1) と同じであるが，さらに，グラフィクス・ディスプレイ（VT-125）に表示された HLS モデルの図の中に変換位置が与えられる．これで，およその色の感じを把むことができる．

(5) COLGEN

カラー・モニタ・ディスプレイが表示可能な色の数（ここでは，16 の 3 乗＝4096 色）すべての（r, g, b）値および対応する HLS モデルの（h, l, s）値を算出してファイルとして格納するプログラムである．このプログラムにより，カラーモニタの特性を確かめることができる．同時に他の色彩利用のプログラム中で，COLGEN により作成した色彩情報をルックアップ・テーブルとして用いることが可能となる．

また，発色数の多い機種を用いる場合，たとえば，各色 8 ビットとすると 256 の 3 乗色となるが，これだけを常時参照したり生成することは大変であるから，必要とする数だけテーブルとして格納しておく，という処理を行うためにも利用できる．こうした操作に必要なプログラムである．たとえば，次にあげる COLPLT ではこれを利用して処理速度の節約を図っている．

(6) COLPLT

(5) のプログラムで生成し格納したテーブルを使って，発色可能な色の HLS モデル上での位置をグラフィクス・ディスプレイ（VT-125）上で確認するためのプログラムである．

(7) COLNEX

HLS モデルの明度を指定すると，その明度における色立体の断面上にある発色可能な色をカラー・モニタ上に表示する．このプログラムを用いて HLS カラー・モデル上の必要とする位置に相当する色を色彩としてカラー・モニタ上に具体的に観察することができる．

表1.

明 度	色の数	明 度	色の数	明 度	色の数
0.0000	1	0.4000	253	0.8000	73
0.0333	6	0.4333	294	0.8333	54
0.0667	13	0.4667	337	0.8667	37
0.1000	24	0.5000	384	0.9000	24
0.1333	37	0.5333	337	0.9333	13
0.1667	54	0.5667	294	0.9667	6
0.2000	73	0.6000	253	1.0000	1
0.2333	96	0.6333	216		
0.2667	121	0.6667	181		
0.3000	150	0.7000	150	(合計して4,096色)	
0.3333	181	0.7333	121		
0.3667	216	0.7667	96		

(8) LINGET/DANMEN

カラー・モニタ上の任意の色彩映像（たとえば，ビデオテープあるいはビデオカメラから入力した映像）の任意のある点から別の点までを，カーソルによりなぞって得られる直線上のRGBの強度の分布をグラフィクス表示する．これは，モニター上の表示色彩を確認するプログラムを応用したユーティリティ・プログラムの一つである．

ところで，HLSモデルを考えるときに注意すべき点として，われわれが実際に利用可能な色は，ハードウェアの制約から，HLSモデルの円錐の内部に稠密に分布しているわけではなく，利用するカラー・モニタの特性に依存して定まるある有限個の色である，ということがある．つまり，円錐の中に離散的に分布していると考えればよい（これを確認するプログラムがCOLPLTである）．

このことは，利用者が色彩を自由に使うことを望んでもそこで用いるハードウェアの環境に左右されるということを意味する．われわれがここで用いたカラー・モニタの場合，明度は31段階（区間 [0, 1] を30等分），その各水準で発色できる色の数はHLSモデルを仮定すると，表1のようになっている（明度 $L=0.5$ で発色数がもっとも多く，双円錐に対して対称となる）．とくに，明度 $L=0$ で黒色，$L=1$ で白色に相当する．

4. プログラムの操作例

ここで，プログラムRGBCNV, HLSRGB, RGBHLS, HLSNEX, LINGET/DANMENについて簡単な例を示す．

4.1. RGBCNVの実行例

まず，RGBモデルとHLS, HSVモデルの間の変換例を示す．なお，[例1]～[例4]までの色彩を示すためのカラー・パターンとして図Aを用意した．

［例 1］ (r, g, b) を与えて (h, l, s) または (h, s, v) を求める.

```
〔例1〕
.RUN RGBCNV

 <<< Color model converter >>>

          RGB -> HLS    (1)
          HLS -> RGB    (2)
          RGB -> HSV    (3)
          HSV -> RGB    (4)
          END           (0)

 COMMAND : 1

 INPUT R,G,B : 12 10 8

 H =  30.00
 L =   0.67
 S =   0.40
```

←メニューから必要とする変換を選ぶ

←1を選ぶ

←(r,g,b)値を入力

←算出した(h,l,s)値

図 4. (r, g, b) から (h, l, s) への変換

［例 2］ (h, l, s) または (h, s, v) を与えて (r, g, b) を求める.

```
〔例2〕
(1)<<< Color model converter >>>

          RGB -> HLS    (1)
          HLS -> RGB    (2)
          RGB -> HSV    (3)
          HSV -> RGB    (4)
          END           (0)

 COMMAND : 2

 INPUT H,L,S : 275.0 0.5 1.0

 R =     9
 G =     0
 B =    15
(h,l,s)を与えて(r,g,b)を求める例である
```

```
(2)<<< Color model converter >>>

          RGB -> HLS    (1)
          HLS -> RGB    (2)
          RGB -> HSV    (3)
          HSV -> RGB    (4)
          END           (0)

 COMMAND : 4

 INPUT H,S,V : 90. 1.0 1.0

 R =     8
 G =    15
 B =     0
```

(黄緑の純色に相当)

図 5. (h, l, s), (h, s, v) から (r, g, b) への変換

4.2. RGBHLS, HLSRGB の実行例

［例 3］ RGBHLS の例

(r, g, b) 値を与えて，それに対応する (h, l, s) 値を算出しこれを HLS カラー・モデルの図の中に表示し，その位置を教える．ここでは，［例 1］と逆の変換を指定し，確かに［例 1］に同じ (h, l, s) 値が得られることを確かめた．

この例で得られる色は，明るい金色がかった灰色である．なお，円錐の断面が正円でない理由はプリンタのアスペクト比がグラフィクス・モニタ画面のそれと異なるためである．すなわち，画面上では正円となっている．

〔例3〕

```
Enter R,G,B
   R,G,B :12 10 8 ←――(r,g,b)値を入力する
        try again ?(y/n) :N
```

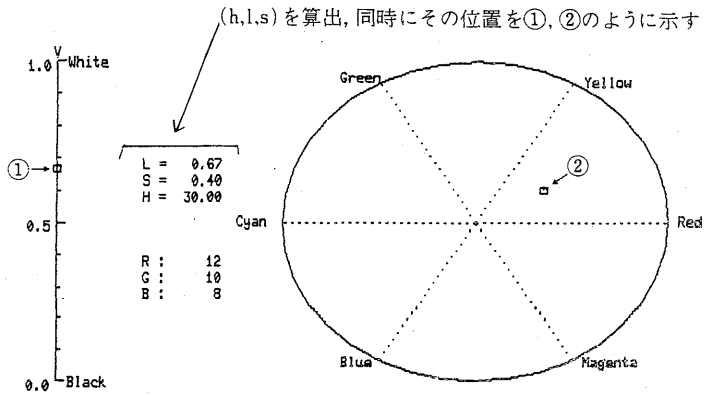


図6. RGBHLSの出力例

［例4］ HLSRGBの例

(h, l, s) 値をカラー・モデルの図の中で，カーソルにより指定し，それに対応する (r, g, b) 値を求める例を示す．この例にみるように，それぞれ発色される色のイメージがカラー・モデルの上で予想される．また，実際に，こうした色に近いものが図Aのように観察される．また図7の (2) の例における (r, g, b)＝(10, 11, 8) を，赤のみ強調して (15, 11, 8) と変えてみると図Aの右上のように輝度がやや増した赤の側に寄った色となる．

4.3. COLNEXの実行例

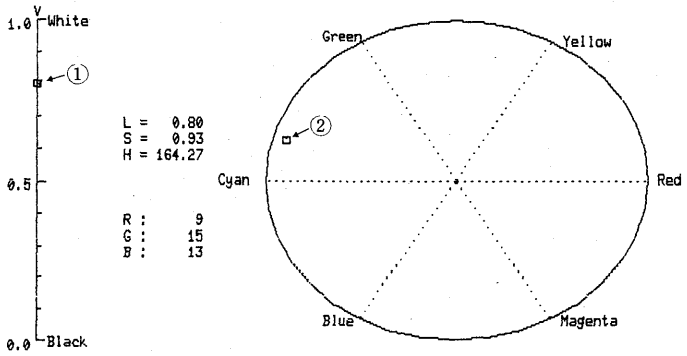
プログラムCOLGENで生成しファイルに格納済みのカラー・ルックアップ・テーブルを用いて，HLSモデルの任意の明度における発色可能な色を色彩テスト・パターンとしてモニタ上に表示する例を示す．

ここでは，L＝0.37およびL＝0.5と与えたときの色彩パターンを図Bの (a), (b) として示した．図Bの (a) は216色あり，(b) は384色ある (表1を参照)．これらの図は，HLSモデルの断面の円を丁度扇子をひろげたように考えて，赤の部分で切断し，次に中心の要をはずして長方形に展開したという場面を想像すればよい．また，ある明度で切断したHLSモデルの断面上の色の数は離散的に分布し，飽和度が小さくなるほど (無彩色レベルに近づくほど) 色の数は減少する．反対に，円錐の外面に近づくほど色の数は増えるので，それらの位置と数の関係を色彩の帯 (短冊) の縦と横の巾で表すようにした．なお図Bのカラー写真の横に，明度をL＝0.37, L＝0.5と与えたときに得られるHLSモデル内の色の分布を，プログラムCOLPLTを使って出力し併記した．

また，各色すなわち各短冊の境界が滑らかにみえないが，これは類似色を連続的に並べたときにみられる“境界効果 (boundary effect)”による一種の錯視であり，隣り合う色の変化が階段状に変化していることに起因している．これを防ぐには，各短冊の間を切り離して空けると

〔例4〕

(1)〈シアンにやや緑を混ぜる〉



(2)〈黄緑の色調を落として，灰色に近づける〉

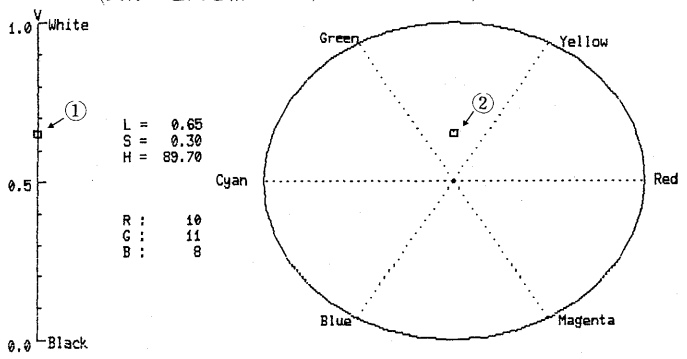


図 7.　HLSRGB の出力例(①, ② の順に指定すると (r, g, b) が算出される)

か，適当な補間法で境界を平滑化するなどの操作を必要とする．たとえば，図Cのように，長方形を大きくとり，また間を切り離すと，この影響がみられない．

4.4.　HLSNEX の実行例

図CはプログラムHLSNEXを用いて得られる色相見本のうち，とくにL＝0.5, S＝1.0(完全飽和，かつ色調が最大) と与えた色の中から12の基本色を選んで(色相を15度づつに12等分した)，カラー・モニタ上に出力させた例である．もちろん，このプログラムによればこれらの色に限らず任意の強度の色を選ぶことができる．図Cは，左下から左上，中段の下から上，…の順に，赤，黄橙，黄，黄緑，緑，…と対応している．

4.5.　LINGET, DANMEN の実行例

色彩画像の，ある点から別の点までの軌跡上にある画素の色彩の分布を調べるプログラムの適用例の一つである．

まず，プログラムLINGETを用いて直線上の画素の色彩情報をサンプリングする．この抽出した (r, g, b) 値をファイルに格納し，次にプログラムDANMENにより軌跡上のRGB強度の変化を視覚的に観察する．

いま，図Dのような色彩画像に対して図中に示す直線に沿って画素の (r, g, b) 値を抽出し

〔LINGETの実行〕

```
Enter file name : TRYA.DAT ← 抽出データ
Plot line (y/n) ? : Y          格納用ファイルの名称
Please set first point !!
Please set last point  !!

( 19 ,132 ) - (164 ,150 ) ← サンプリングを開始した点と
                             終了時の点の位置座標
   4     6     8
   5     5     4
   5     5     5
   6     7     9
                    } 抽出した線上の各点の(r,g,b)値
  10     9     7
   9     8     7
   9    10     9
   9     9    11
   9     9    10

( 19 ,132 ) - (164 ,150 ) ← 位置の確認

COUNT =   146 ← サンプリングした点の数(画素数)

try again (y/n) ? : N
```

図8. LINGETの実行例

さらにその強度分布を出力すると図8, 9となる. 図8はプログラムLINGETによる画素の抽出過程を示し, 図9は(r, g, b)の各値のヒストグラムおよび抽出開始点から終了点までの強度の変化を示すグラフである. 図Dと図8, 9との比較により抽出線上の色彩変化を視覚的に観察することができる.

6. プログラム・リスト

上に述べたプログラムのうち, RGBCNV, RGBHLS, およびHLSRGBの3つについてプログラム・リストを挙げる.

なおここで示したプログラムのうちRGBCNVを除く他のプログラムは, グラフィクス・ディスプレイ(VT-125)あるいはカラー・グラフィクス・モニタ(NEXUS-5800)を必要とする. VT-125を用いる場合にはMINC-11/23用のグラフィクス・ディスクリプタ(グラフィクス制御コマンド集合)であるREGIS (Remote-Graphics Instruction Set)および, その機能をサブルーチン化したRGL/FEPライブラリ(RGLLIB)を必要とする. たとえば, プログラム中で現れるサブルーチン名, INTGR, CLRSCR, SWINDO, MOVE, TEXT, RELLIN, SLNPAT, MARKERなどがそれである.

これらのサブルーチンの機能の詳細については文献(RGL/FEP Programmer's Reference Manual, VT-125 USER Guide)を参照されたい. また, 他の機種でプログラムを利用する場合は, これらのルーチンを同等の機能のものと置き換える必要がある.

さらに, カラー・モニタを使う場合には, これを制御するシステム・ライブラリとしてMINC-11/23のOS下で機能するRGBインターフェース制御用のシステム・ライブラリIBLIBを必要とする(ただし, ここで掲載のプログラムでは用いていない).

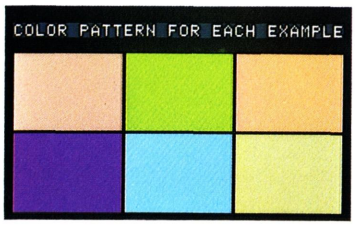
〔DANMENの実行〕

```
Enter file name ? : TRYA.DAT
count =   146
PUSH RETURN !!
```

(*) (r,g,b) 値それぞれの分布

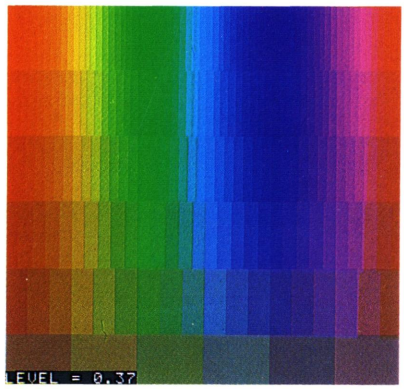
```
<<< R >>>

 0 :     0 :
 1 :     0 :
 2 :     0 :
 3 :     4 : *****
 4 :    18 : ******************************
 5 :    24 : ****************************************
 6 :    23 : ***************************************
 7 :    28 : ************************************************
 8 :    19 : ******************************
 9 :    19 : ******************************
10 :     7 : **********
11 :     3 : ****
12 :     1 :
13 :     0 :
14 :     0 :
15 :     0 :

<<< G >>>

 0 :     0 :
 1 :     0 :
 2 :     0 :
 3 :     0 :
 4 :     5 : *******
 5 :    18 : ******************************
 6 :    36 : ************************************************************
 7 :    35 : ***********************************************************
 8 :    30 : **************************************************
 9 :    18 : ******************************
10 :     4 : *****
11 :     0 :
12 :     0 :
13 :     0 :
14 :     0 :
15 :     0 :

<<< B >>>

 0 :     0 :
 1 :     0 :
 2 :     0 :
 3 :     1 :
 4 :     1 :
 5 :     6 : *********
 6 :    24 : ****************************************
 7 :    34 : ********************************************************
 8 :    32 : *******************************************************
 9 :    30 : **************************************************
10 :    12 : ********************
11 :     5 : *******
12 :     1 :
13 :     0 :
14 :     0 :
15 :     0 :
```

図 9.　DANMEN の実行例



例1	例2 (2)	例4-(2) (赤を強調)
例2 (1)	例4 (1)	例4 (2)

図 A.



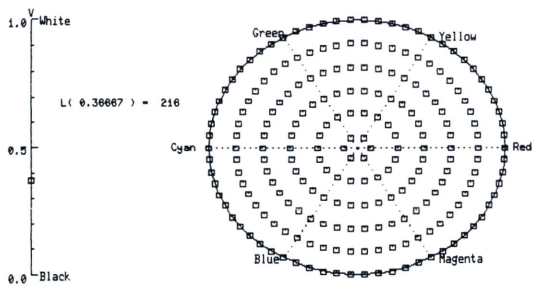
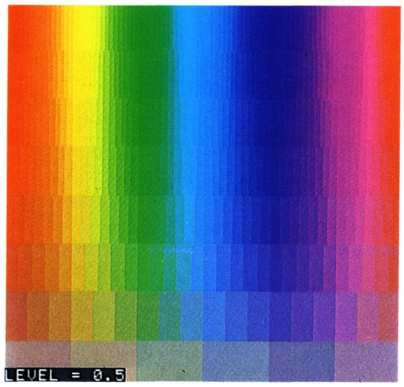


図 B(a)



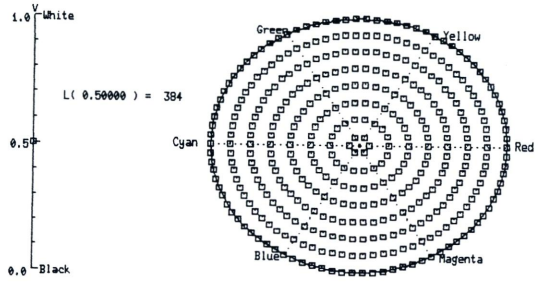


図 B(b)

図 C.

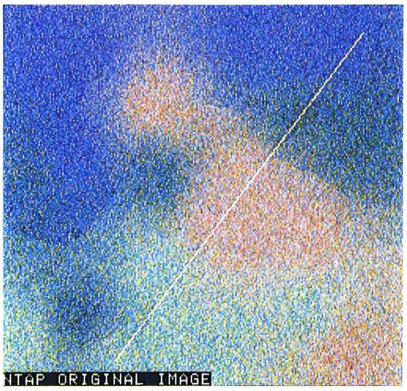


図 D.

(*) (r,g,b) 値のサンプリングした軌跡上の強度の変化

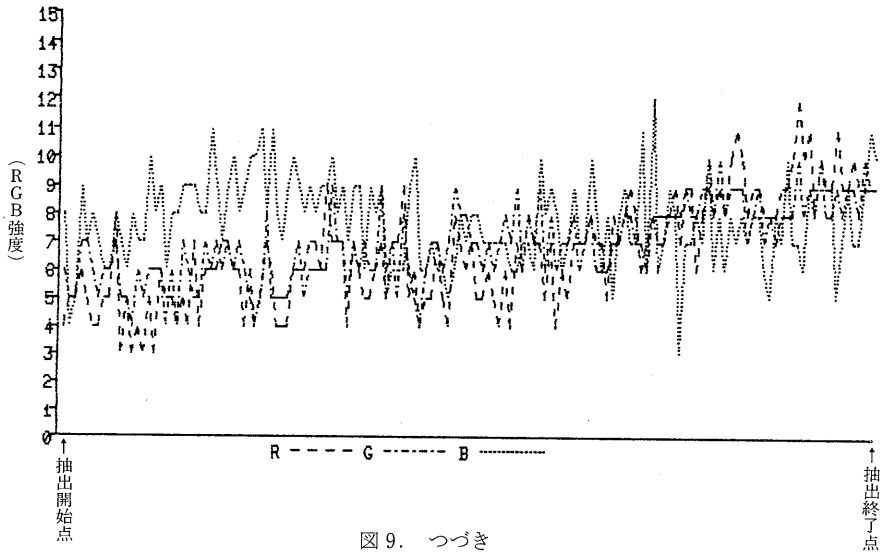


図9. つづき

謝 辞

適切なコメントをいただいた査読者の方々に心から感謝いたします.

参 考 文 献

Beatty, J.C. (1983). Raster graphics and color, *Amer. Statistician,* **37**, 1, 60-75.

Becker, R.A. (1983). Integrating color into statistical software, Comp. Sciences and Statistics, The Interface, 14th Symposium.

Cleveland, W.S., McGill, R. (1983). A color-caused optical illusion on a statistical graphs, *Amer. Statistician,* **37**, 2, 101-105.

Enderle, G., Kansy, K., Pfaff, G. (1984). Computer Graphics Programming-GKS The Graphical Standard, Springer Verlag.

Foley, J.D., Van Dam, A. (1982). *Fundamentals of Interactive Computer Graphics,* Addison Wesley.

Hosmer, T., Edwards, R. (1984). Imaging: A tool for data reduction, *Amer. Statistician,* **38**, 4, 322-327.

Nicholson, W.L., Littlefield, R.J. (1983). Interactive color graphics for multivariate data, Comp. Science and Statistics, The Interface, 14th Symposium.

Trumbo, B.E. (1981). A theory for coloring bivariate statistical maps, *Amer. Statistician,* **35**, 4, 220-226.

Wainer, H., Francolini, C.M. (1980). An empirical inquiry concerning human understanding of two-variable color maps, Amer Statistician **34**, 2, 81-93.

リー・ボールドウィン (1985). コンピュータ・グラフィックスにおける色彩の考察, 日経バイト, March, 163-174.

朝日新聞社編 (1983). 色の彩時記.

RGL/FEP Programmer's Reference Manual, Digital Equipment Cooporation (1981). (ReGIS Graphics Library for the FORTRAN Enhance Package, Version 2)

VT-125 USER Guide, Digital Equipment Cooperation (1981).

(1) プログラム RGBCNV

```
c-----------------------------------------------------------------------
      program RGBCNV
c.......................................................................
c
c     This is a program for providing specifications of colors within
c     some color samut and performs conversions one color model to the
c     other among three color models;RGB,HLS and HSV.
c      RGB = RGB color model;the color intensities are specified
c            as (R,G,B) triples.
c      HLS = HLS(hue,lightness,saturation) color model;
c            the double cone model.
c            H / ranging from 0 to 360 degrees as angle.
c            L / ranging from 0 to 1 as a ratio.
c            S / ranging from 0 to 1 as a ratio.
c      HSV = HSV(hue,saturation,value) color model;
c            the single cone model.
c            H / ranging from 0 to 360 degrees as angle.
c            S / ranging from 0 to 1 as a ratio.
c            V / ranging from 0 to 1 as a ratio.
c
c      Authors and Correspondence :
c       Ichiro Kaneko and Noboru Ohsumi
c       4-6-7, Minami-Azabu, Minato-ku, Tokyo 106
c       The Institute of Statistical Mathematics
c       Date : April, 1985/ Revised January, 1986
c
c-----------------------------------------------------------------------
c
      data CLEVEL/15.0/                          ! color intensity levels
c
   10 continue
      write (6,600)
  600 format (//1h ,' <<< Color model converter >>>'//
     $            11x,'RGB -> HLS    (1)'/
     $            11x,'HLS -> RGB    (2)'/
     $            11x,'RGB -> HSV    (3)'/
     $            11X,'HSV -> RGB    (4)'/
     $            11X,'END           (0)'//)
c
      write (6,610) 7
  610 format (1h ,' COMMAND : ',a1,$)
      read (5,*) num
      if (num .eq. 1) call rtohl(CLEVEL)         ! RGB to HLS
      if (num .eq. 2) call hltor(CLEVEL)         ! HLS to RGB
      if (num .eq. 3) call rtohs(CLEVEL)         ! RGB to HSV
      if (num .eq. 4) call hstor(CLEVEL)         ! HSV to RGB
      if (num .ne. 0) goto 10                    ! error
c
      stop
      end
c-----------------------------------------------------------------------
      subroutine rtohl(CLEVEL)                   ! RGB to HLS
c
      call rersb(ir,is,ib)                       ! read R,G,B
c
      call rsbhls(ir,is,ib,h,al,s,CLEVEL)        ! convert
c
      write (6,600) h,al,s
  600 format (//1h ,' H =',f7.2,/,'  L =',f7.2,/,'  S =',f7.2)
c
      return
      end
c-----------------------------------------------------------------------
      subroutine hltor(CLEVEL)                   ! HLS to RGB
c
      write (6,600)
  600 format (//1h ,' INPUT H,L,S : ',$)
      read (5,*) h,al,s                          ! read H,L,S
c
      call hlsrsb(ir,is,ib,h,al,s,CLEVEL)        ! convert
c
      call prrsb(ir,is,ib)                       ! print R,G,B
c
      return
      end
c-----------------------------------------------------------------------
      subroutine rtohs(CLEVEL)                   ! RGB to HSV
c
      call rersb(ir,is,ib)                       ! read R,G,B
c
      call rsbhsv(ir,is,ib,h,s,v,CLEVEL)         ! convert
c
      write (6,600) h,s,v
  600 format (//1h ,' H =',f7.2,/,'  S =',f7.2,/,'  V =',f7.2)
c
      return
      end
c-----------------------------------------------------------------------
      subroutine hstor(CLEVEL)                   ! HSV to RGB
c
      write (6,600)
  600 format (//1h ,' INPUT H,S,V : ',$)
      read (5,*) h,s,v                           ! read H,S,V
c
      call hsvrsb(ir,is,ib,h,s,v,CLEVEL)         ! convert
c
      call prrsb(ir,is,ib)                       ! print R,G,B
c
      return
      end
c-----------------------------------------------------------------------
      subroutine rersb(ir,is,ib)                 ! read R,G,B
c
      write (6,600)
  600 format (//1h ,' INPUT R,G,B : ',$)
      read (5,*) ir,is,ib
c
      return
      end
c-----------------------------------------------------------------------
      subroutine prrsb(ir,is,ib)                 ! print R,G,B
c
      write (6,600) ir,is,ib
  600 format (//1h ,' R =',i7,/,'  G =',i7,/,'  B =',i7)
c
      return
      end
```

```
c-----------------------------------------------------------------------
C     procedure RGB to HLS convert
      subroutine rgbhls(ir,ig,ib,h,al,s,CLEVEL)
c
c     ir,ig,ib in[0..CLEVEL]
c     al,s in[0.0..1.0]
c     h in[0.0..360.0]
c
      fc(i,max,min) = float(max - i) / float(max - min)
c
      max = max0(ir,ig,ib)
      min = min0(ir,ig,ib)
      al = (max + min) / CLEVEL / 2.0
c
      s = 0.0
      h = 0.0                                   ! undefined
      if (max .eq. min) return
c
      s = float((max - min)) / float((max + min))
      if (al .gt. 0.5) s = float((max - min)) /
     $                              (CLEVEL * 2.0  - max - min)
c
      rc = fc(ir,max,min)
      gc = fc(ig,max,min)
      bc = fc(ib,max,min)
c
      if (ir .eq. max) h = bc - gc
      if (ig .eq. max) h = 2.0 + rc - bc
      if (ib .eq. max) h = 4.0 + gc - rc
c
      h = amod(h * 60.0 + 360.0,360.0)
c
      return
      end
c-----------------------------------------------------------------------
C     procedure HLS to RGB convert
      subroutine hlsrgb(ir,ig,ib,h,al,s,CLEVEL)
c
c     ir,ig,ib in[0..CLEVEL]
c     al,s in[0.0..1.0]
c     h in[0.0..360.0]
c
      ira(x) = ifix(x + 0.5)
c
      am2 = al * (1.0 + s)
      if (al .gt. 0.5) am2 = al + s - al * s
c
      am1 = 2.0 * al - am2
c
      if (s .ne. 0.0) goto 10
c
      if (h .eq. 0.0) goto 20
       write (6,600) ir,ig,ib,h,al,s
  600  format (ih ,' ?-HLSRGB error ',3i5,3f10.5)
       call exit
c
   20 ir = ira(al * CLEVEL)
      ig = ira(al * CLEVEL)
      ib = ira(al * CLEVEL)
      return
c
   10 continue
      ir = ikhlrg(am1,am2,h + 120.0,CLEVEL)
      ig = ikhlrg(am1,am2,h        ,CLEVEL)
      ib = ikhlrg(am1,am2,h - 120.0,CLEVEL)
c
      return
      end
c-----------------------------------------------------------------------
      function ikhlrg(an1,an2,hue,CLEVEL)       ! VALUE
c
      ira(x) = ifix(x + 0.5)
c
      hue = amod(hue + 360.0,360.0)
c
      akhlrg = an1
      if (hue .lt. 240.0)
     $  akhlrg = an1 + (an2 - an1) * (240.0 - hue) / 60.0
      if (hue .lt. 180.0) akhlrg = an2
      if (hue .lt. 60.0) akhlrg = an1 + (an2 - an1) * hue / 60.0
      ikhlrg = ira(akhlrg * CLEVEL)
c
      return
      end
c-----------------------------------------------------------------------
C     procedure RGB to HSV convert
      subroutine rgbhsv(ir,ig,ib,h,s,v,CLEVEL)
c
c     ir,ig,ib in[0..CLEVEL]
c     h in[0.0..360.0]
c     s,v in[0.0..1.0]
c
      fc(i,max,min) = float((max - i)) / float((max - min))
c
      max = max0(ir,ig,ib)
      min = min0(ir,ig,ib)
      v = max / CLEVEL
      s = 0.0
      if (max .ne. 0) s = float((max - min)) / float(max)
c
      h = 0.0                                   ! undefined
      if (s .eq. 0.0) return
c
      rc = fc(ir,max,min)
      gc = fc(ig,max,min)
      bc = fc(ib,max,min)
c
      if (ir .eq. max) h = bc - gc
      if (ig .eq. max) h = 2.0 + rc - bc
      if (ib .eq. max) h = 4.0 + gc - rc
c
      h = amod(h * 60.0 + 360.0,360.0)
c
      return
      end
```

```
c-----------------------------------------------------------------------
C     procedure HSV to RGB convert
      subroutine hsvrgb(ir,ig,ib,h,s,v,CLEVEL)
c
c     ir,ig,ib in[0..CLEVEL]
c     s,v in[0.0..1.0]
c     h in[0.0..360.0]
c
      ira(x) = ifix(x + 0.5)
c
      h1 = h
c
      if (s .ne. 0.0) goto 10
c
      if (h .eq. 0.0) goto 20
       write (6,600) ir,ig,ib,h,s,v
  600  format (1h ,' ?-HSVRGB error ',3i5,3f10.5)
       call exit
c
   20 ir = ira(v * CLEVEL)
      ig = ira(v * CLEVEL)
      ib = ira(v * CLEVEL)
      return
c
   10 if (h1 .eq. 360.0) h1 = 0.0
      h1 = h1 / 60.0
      i = ifix(h1)
      f = h1 - i
      p = v * (1.0 - s)
      q = v * (1.0 - (s * f))
      t = v * (1.0 - (s * (1.0 - f)))
c
      if (i .ne. 0) goto 1
       ir = ira(v * CLEVEL)
       ig = ira(t * CLEVEL)
       ib = ira(p * CLEVEL)
       return
    1 if (i .ne. 1) goto 2
       ir = ira(q * CLEVEL)
       ig = ira(v * CLEVEL)
       ib = ira(p * CLEVEL)
       return
    2 if (i .ne. 2) goto 3
       ir = ira(p * CLEVEL)
       ig = ira(v * CLEVEL)
       ib = ira(t * CLEVEL)
       return
    3 if (i .ne. 3) goto 4
       ir = ira(p * CLEVEL)
       ig = ira(q * CLEVEL)
       ib = ira(v * CLEVEL)
       return
    4 if (i .ne. 4) goto 5
       ir = ira(t * CLEVEL)
       ig = ira(p * CLEVEL)
       ib = ira(v * CLEVEL)
       return
    5 continue
       ir = ira(v * CLEVEL)
       ig = ira(p * CLEVEL)
       ib = ira(q * CLEVEL)
       return
      end
```

(2) プログラム RGBHLS

```
c-----------------------------------------------------------------------
      program RGBHLS  ! convert RGB to HLS models
c.....................................................................
c
c     This program is for transforming the RGB color system to the
c     HLS color model and the resulting is displayed as a point
c     within a HLS-double-cone model on the VT-125 graphic terminal.
c     The followins library is used : RGLLIB (for VT-125)
c
c     Authors and Correspondence :
c      Ichiro Kaneko and Noboru Ohsumi
c      4-6-7, Minami-Azabu, Minato-ku, Tokyo 106
c      The Institute of Statistical Mathematics
c      Date April, 1985/ Revised January, 1986
c
c-----------------------------------------------------------------------
c
      byte yes,word(80)
c
      call initgr(5)
   10 call clrscr
      write (6,600)
  600 format (///1h ,' Enter R,G,B',//,'       R,G,B :',$)
      read (5,*) ir,ig,ib
c
      call initgr(5)
      call clrscr
      call clrtxt
c
      call swindo(-0.7,-0.1,1.1,1.1)
c
      call linev
c
      call cirsh
c
      call prgb(ir,ig,ib)
c
      call rgbhls(ir,ig,ib,h,al,s)
c
      call phls(h,al,s)
c
      call move(-0.4,0.7)
      encode (11,900,word) ' L = ',al
  900 format (a5,f6.2)
      word(12) = 0
      call text(word)
c
      call move(-0.4,0.65)
      encode (11,900,word) ' S = ',s
      word(12) = 0
      call text(word)
c
      call move(-0.4,0.60)
      encode (11,900,word) ' H = ',h
      word(12) = 0
      call text(word)
c
      write (6,630)
  630 format (1h ,10x,'try again ?(y/n) :',$)
      read (5,500) yes
  500 format (a1)
      if (yes .eq. 'y' .or. yes .eq. 'Y') goto 10
c
      call cpyscr
c
      call clrscr
      stop
      end
```

```
c-----------------------------------------------------------------------
      subroutine rgb(ir,ig,ib)
c
      byte word(12)
c
      call move(-0.4,0.4)
      encode (11,900,word) ' R : ',ir
  900 format (a5,i6)
      word(12) = 0
      call text(word)
      call move(-0.4,0.35)
      encode (11,900,word) ' G : ',ig
      word(12) = 0
      call text(word)
      call move(-0.4,0.3)
      encode (11,900,word) ' B : ',ib
      word(12) = 0
      call text(word)
c
      return
      end
c-----------------------------------------------------------------------
      subroutine linev
c
      call move(-0.6,1.0)
      call line(-0.6,0.0)
c
      call rellin(0.02,0.0)
      call move(-0.6,0.1)
      call rellin(0.01,0.0)
      call move(-0.6,0.2)
      call rellin(0.01,0.0)
      call move(-0.6,0.3)
      call rellin(0.01,0.0)
      call move(-0.6,0.4)
      call rellin(0.01,0.0)
      call move(-0.6,0.5)
      call rellin(0.02,0.0)
      call move(-0.6,0.6)
      call rellin(0.01,0.0)
      call move(-0.6,0.7)
      call rellin(0.01,0.0)
      call move(-0.6,0.8)
      call rellin(0.01,0.0)
      call move(-0.6,0.9)
      call rellin(0.01,0.0)
      call move(-0.6,1.0)
      call rellin(0.02,0.0)
c
      call move(-0.61,1.05)
      call text('V')
      call move(-0.574,1.02)
      call text('White')
      call move(-0.574,0.02)
      call text('Black')
c
      call move(-0.68,1.01)
      call text('1.0')
      call move(-0.68,0.51)
      call text('0.5')
      call move(-0.68,0.01)
      call text('0.0')
c
      return
      end
c-----------------------------------------------------------------------
      subroutine cirsh
c
      call move(0.5,0.5)
      call circle(0.5)
c
      call slnpat(6,)
      call move(0.5,0.5)
      call rellin(0.5,0.0)
      call move(0.5,0.5)
      call rellin(0.25,-0.433)
      call move(0.5,0.5)
      call rellin(-0.25,-0.433)
      call move(0.5,0.5)
      call rellin(-0.5,0.0)
      call move(0.5,0.5)
      call rellin(-0.25,0.433)
      call move(0.5,0.5)
      call rellin(0.25,0.433)
      call slnpat(1,)
c
      call move(0.775,0.96)
      call text('Yellow')
      call move(1.025,0.525)
      call text('Red')
      call move(0.775,0.087)
      call text('Masenta')
      call move(0.15,0.087)
      call text('Blue')
      call move(-0.13,0.525)
      call text('Cyan')
      call move(0.146,0.972)
      call text('Green')
c
      return
      end
c-----------------------------------------------------------------------
C     Procedure RGB to HLS
      subroutine rgbhls(ir,ig,ib,h,al,s)
c     ir,ig,ib in[0..15]
c     al,v in[0.0..1.0]
c     h in[0.0..360.0]
c
      fc(i,max,min) = float(max - i) / float(max - min)
c
      max = max0(ir,ig,ib)
      min = min0(ir,ig,ib)
      al = (max + min) / 30.0
c
      s = 0.0
      h = 0.0                                   ! undefined
      if (max .eq. min) return
c
      s = float((max - min)) / float((max + min))
      if (al .gt. 0.5) s = float((max - min)) / (30.0 - max - min)
c
      rc = fc(ir,max,min)
      gc = fc(ig,max,min)
      bc = fc(ib,max,min)
c
      if (ir .eq. max) h = bc - gc
      if (ig .eq. max) h = 2.0 + rc - bc
      if (ib .eq. max) h = 4.0 + gc - rc
c
      h = amod(h * 60.0 + 360.0,360.0)
c
      return
      end
```

```
C-----------------------------------------------------------------------
      subroutine phls(h,al,s)
c
      call marker(1,-0.6,al)
c
c
      al = al
      if (al .gt. 0.5) al = 1.0 - al
      if (al .le. 0.001) al = 0.001
c
      x = s * cos(h * 3.14159265 / 180.0) / 2.0 + 0.5
      y = s * sin(h * 3.14159265 / 180.0) / 2.0 + 0.5
c
      call marker(1,x,y)
c
      return
      end
```

(3) プログラム HLSRGB

```
c-----------------------------------------------------------------------
      program HLSRGB ! convert HLS models to RGB ( without NEXUS)
c.....................................................................
c
c     This program performs conversion from the HLS color model
c     to the RGB one.
c     The followins library is used : RGLLIB (for VT-125)
c
c     Authors and Correspondence :
c      Ichiro Kaneko and Noboru Ohsumi
c      4-6-7, Minami-Azabu, Minato-Ku, Tokyo 106
c      The Institute of Statistical Mathematics
c      Date : April, 1985/ Revised January, 1986
c
c-----------------------------------------------------------------------
c
      byte yes
c
   10 call initgr(5)
      call clrscr
      call clrtxt
c
      call swindo(-0.7,-0.1,1.1,1.1)
c
      call linev
c
      call cirsh
c
      call inp(h,al,s)
c
      call hlsrgb(ir,ig,ib,h,al,s)
c
      call prgb(ir,ig,ib)
c
      write (6,600)
  600 format (1h ,10x,'try again ?(y/n) : ',$)
      read (5,500) yes
  500 format (a1)
      if (yes .eq. 'y' .or. yes .eq. 'Y') goto 10
c
      call cpyscr
c
      call clrscr
c
      stop
      end
```

```
c-----------------------------------------------------------------------
      subroutine inp(h,al,s)
c
      byte word(12)
c
   10 x = -0.59
      y = 0.5
c
      call cur(x,y)
      if (y .lt. 0.0 .or. y .gt. 1.0) goto 10
c
      call marker(1,-0.6,y)
c
      al = y
c
      call move(-0.4,0.7)
      encode (11,900,word) ' L = ',al
  900 format (a5,f6.2)
      word(12) = 0
      call text(word)
c
   20 x = 0.5
      y = 0.5
c
      call cur(x,y)
      x = x - 0.5
      y = y - 0.5
c
      if (sqrt(x ** 2 + y ** 2) .gt. 0.5) goto 20
c
      call marker(1,x + 0.5,y + 0.5)
c
      s = sqrt(x ** 2 + y ** 2) * 2.0
      h = amod(atan2(y,x) * 180.0 / 3.14159265 + 360.0,360.0)
c
      call move(-0.4,0.65)
      encode (11,900,word) ' S = ',s
      word(12) = 0
      call text(word)
      call move(-0.4,0.60)
      encode (11,900,word) ' H = ',h
      word(12) = 0
      call text(word)
c
      return
      end
c-----------------------------------------------------------------------
      subroutine prgb(ir,ig,ib)
c
      byte word(12)
c
      call move(-0.4,0.4)
      encode (11,900,word) ' R : ',ir
  900 format (a5,i6)
      word(12) = 0
      call text(word)
      call move(-0.4,0.35)
      encode (11,900,word) ' G : ',ig
      word(12) = 0
      call text(word)
      call move(-0.4,0.3)
      encode (11,900,word) ' B : ',ib
      word(12) = 0
      call text(word)
c
      return
      end
```

```
c-----------------------------------------------------------------------
      subroutine cur(x,y)
c
      call move(-0.5,-0.05)
      call text(' PUSH return !')
c
   10 call locate(x,y,key)
      if (key .ne. 15) goto 10
c
      call swmode('RE')
      call move(-0.5,-0.05)
      call text('                ')
      call swmode('OV')
c
      return
      end
c-----------------------------------------------------------------------
      subroutine linev
c
      call move(-0.6,1.0)
      call line(-0.6,0.0)
      call rellin(0.02,0.0)
      call move(-0.6,0.1)
      call rellin(0.01,0.0)
      call move(-0.6,0.2)
      call rellin(0.01,0.0)
      call move(-0.6,0.3)
      call rellin(0.01,0.0)
      call move(-0.6,0.4)
      call rellin(0.01,0.0)
      call move(-0.6,0.5)
      call rellin(0.02,0.0)
      call move(-0.6,0.6)
      call rellin(0.01,0.0)
      call move(-0.6,0.7)
      call rellin(0.01,0.0)
      call move(-0.6,0.8)
      call rellin(0.01,0.0)
      call move(-0.6,0.9)
      call rellin(0.01,0.0)
      call move(-0.6,1.0)
      call rellin(0.02,0.0)
      call move(-0.61,1.05)
      call text('V')
      call move(-0.574,1.02)
      call text('White')
      call move(-0.574,0.02)
      call text('Black')
      call move(-0.68,1.01)
      call text('1.0')
      call move(-0.68,0.51)
      call text('0.5')
      call move(-0.68,0.01)
      call text('0.0')
c
      return
      end
c-----------------------------------------------------------------------
      subroutine cirsh
c
      call move(0.5,0.5)
      call circle(0.5)
      call slnpat(6,)
      call move(0.5,0.5)
      call rellin(0.5,0.0)
      call move(0.5,0.5)
      call rellin(0.25,-0.433)
      call move(0.5,0.5)
      call rellin(-0.25,-0.433)
      call move(0.5,0.5)
      call rellin(-0.5,0.0)
      call move(0.5,0.5)
      call rellin(-0.25,0.433)
      call move(0.5,0.5)
c
      call move(0.775,0.96)
      call text('Yellow')
      call move(1.025,0.525)
      call text('Red')
      call move(0.775,0.087)
      call text('Masenta')
      call move(0.15,0.087)
      call text('Blue')
      call move(-0.13,0.525)
      call text('Cyan')
      call move(0.146,0.972)
      call text('Green')
c
      return
      end
c-----------------------------------------------------------------------
C     Procedure HLS to RGB
      subroutine hlsrgb(ir,ig,ib,h,al,s)
c     ir,ig,ib in[0..15]
c     al,s in[0.0..1.0]
c     h in[0.0..360.0]
c
      ira(x) = ifix(x + 0.5)
c
      am2 = al * (1.0 + s)
      if (al .gt. 0.5) am2 = al + s - al * s
c
      am1 = 2.0 * al - am2
c
      if (s .ne. 0.0) goto 10
c
      if (h .eq. 0.0) goto 20
       write (6,600) ir,ig,ib,h,al,s
  600  format (1h ,' ?-HLSRGB error ',3i5,3f10.5)
       call exit
   20 ir = ira(al * 15.0)
      ig = ira(al * 15.0)
      ib = ira(al * 15.0)
      return
c
   10 continue
      ir = ikhlrg(am1,am2,h + 120.0)
      ig = ikhlrg(am1,am2,h)
      ib = ikhlrg(am1,am2,h - 120.0)
c
      return
      end
c-----------------------------------------------------------------------
      function ikhlrg(an1,an2,hue)
c
      ira(x) = ifix(x + 0.5)
c
      hue = amod(hue + 360.0,360.0)
c
      akhlrg = an1
      if (hue .lt. 240.0)
     $  akhlrg = an1 + (an2 - an1) * (240.0 - hue) / 60.0
      if (hue .lt. 180.0) akhlrg = an2
      if (hue .lt. 60.0) akhlrg = an1 + (an2 - an1) * hue / 60.0
      ikhlrg = ira(akhlrg * 15.0)
c
      return
      end
```

Transformation Programs between Color Models for Computer Graphics

Noboru Ohsumi

(The Institute of Statistical Mathematics)

Techniques for using color graphics systems in data analysis and a design policy for user-friendly interface software are described. An implementation of these concepts in several new programs for color-model transformations are provided. These can be fully realized with a microcomputer and a low-cost graphics display monitor and form the basis for an integrated color-handling system. Examples of the results obtained with these programs are also provided.