KONVERSI WARNA RGB KE HLS MENGGUNAKAN C++

Ina Agustina, Fauziah
Jurusan Sistem Informasi Universitas Nasional
Jl. Sawo Manila No. 61 Pasar Minggu Jakarta Selatan
E-Mail : ina_agustina2007@yahoo.com,fauziah_z2@yahoo.com

Abstrak
Pada Grafik Komputer, banyak dipelajari tentang konversi warna, misalnya dari RGB ke CIE, dari RGB ke HSV, dan dari RGB ke HLS. Dimana RGB merupakan model warna berorientasi hardware, sedangkan CIE, HSV, dan HLS merupakan model warna berorientasi software. Maka dari itu di makalah ini dibahas konversi model warna yang berorientasikan hardware ke model warna yang berorientasikan software.
Makalah ini membahas konversi warna dari RGB Ke HSL menggunakan bahasa Pemrograman C++ dengan IDE Micosoft Visual C++.

Keyword : Model Warna RGB, Model Warna HLS, Bahasa Pemrograman C++

1. PENDAHULUAN

Di makalah ini hanya membahas tentang konversi warna dari RGB ke HLS. Dimana RGB singkatan dari *Red*, *Green*, dan *Blue*. Sedangkan HLS singkatan dari *Hue*, *Lightness*, dan Saturation. Bahasa pemrogram yang dipakai adalah C++ dengan menggunakan IDE Microsoft Visual C++. Makalah "Konversi Warna RGB ke HLS" ini berisi penjelasan materi beserta contoh program, algoritma program, dan output program guna menunjang penguasaan dan pemahaman atas materi yang disajikan. Disini kami membuat program mengenai konversi warna dari RGB ke HLS, dimana dalam nilai RGB, angka yang dimasukkan harus antara 0 sampai dengan 1. Kemudian kami konversikan angka tersebut ke dalam HLS. Dimana maksimal percampuran untuk warna-warna dominan pada HLS adalah sampai dengan S= 1 dan L = 0,5.

2 TINJAUAN PUSTAKA

Warna sebenarnya merupakan presepsi kita terhadap pantulan cahaya dari benda-benda di depan mata. Mata manusia dibagi menjadi beberapa bagian, yaitu cornea, iris, lensa, retina, dan syaraf mata yang menghubungkan retina dengan otak.
Sel-sel mata di bagian retina dibagi menjadi 2 macam:

1. *Rods* → peka terhadap perbedaan intensitas warna terang – gelap.
2. *Cones* → peka terhadap spektrum cahaya dengan panjang gelombang 390mm – 720mm. Cones dibagi menjadi 3, yaitu: *red cones*, *green cones*, dan *blue cones*.

Pada teori warna terdapat 2 model warna, yaitu model warna berorientasi *hardware* dan berorientasi *software*. Model warna yang banyak digunakan saat ini berorientasi *hardware* (contoh monitor dan printer) atau aplikasi dimana manipulasi warna menjadi tujuannya (kreasi warna grafik untuk animasi).
Berikut ini adalah model warna yang berorientasi *hardware*:

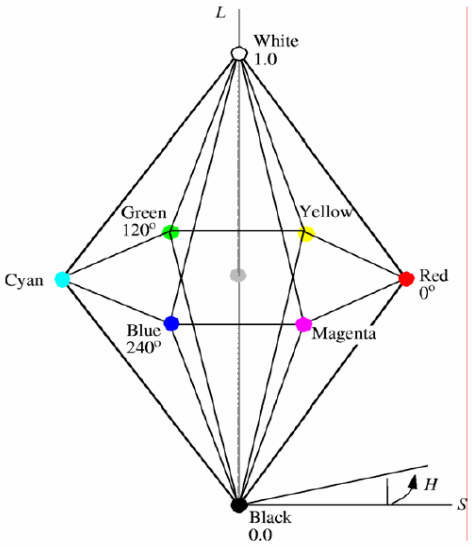
1. Model RGB (*red, green, blue*) untuk warna monitor dan warna pada kamera video.
2. Model CMY (*cyan, magenta, yellow*) untuk model printer.
3. Model YIQ , digunakan untuk standard televisi. Y berkoresponden dengan luminasi, I dan Q adalah dua komponen kromatik yang disebut *inphase* dan *quarature* .

Berikut ini adalah model warna berorientasi *Software* (*hue, saturation,brightness*) adalah manipulasi :

1. Model HSV (*hue, saturation, value*).
2. Model HSI (*hue, saturation, intensity*).
3. Model HLS (*hue, lightness, saturation*).

Karena dalam makalah ini kami hanya membahas tentang konversi warna dari RGB ke HLS, maka disini kami hanya menjelaskan tentang RGB dan HLS. RGB merupakan singkatan dari *Red, Green, Blue*. Model warna RGB didasarkan pada teori dimana mata manusia peka terhadap panjang gelombang 630nm (merah), 530nm (hijau) dan 450nm (biru). Dengan campuran warna *red, green, blue* akan didapat campuran warna aditif. Warna aditif terjadi karena sumber cahaya memancarkan sejumlah energi pada panjang gelombang tertentu dan penjumlahan energi pada berbagai panjang gelombang yang dipancarkan sumber cahaya menentukan warna akhir, dengan rumus: $W = R_R + G_G + B_B$. HLS merupakan singkatan dari *Hue, Lightness, Saturation.* Hue merupakan warna dominan dalam *angle* (warna dominan yang dinyatakan dengan sudut). Saturation adalah banyaknya campuran warna yang dominan dengan putih (*purity*). Sedangkan *Lightness* adalah terang atau

gelapnya intensitas warna. Maksimal percampuran untuk warna-warna dominan sampai dengan S= 1 dan L = 0,5. Berikut adalah penggambaran model HLS:



Gambar 1. Model Warna HLS

3. .HASIL DAN PEMBAHASAN

Algoritma Program Konversi Warna RGB ke HLS :

1. Tentukan masing-masing nilai warna RGB yaitu *Red, Green* dan *Blue* yang ditampung pada variabel r untuk merah, g untuk hijau dan b untuk biru. Karena dalam nilai RGB, maka angka yang dimasukkan harus antara 0 sampai 1.
2. Cari nilai terbesar antara tiga bilangan yang telah diinput dan beri nama dengan mx.
3. Cari nilai terkecil antara tiga bilangan yang telah diinput dan beri nama dengan mn .
4. Hitung l= (mx+mn) / 2.
5. Hitung nilai s sebagai *saturation* dan nilai h sebagai *hue*.
   - Jika mx=mn, maka s=0 dan h=0, yang berarti abu-abu.
   - Jika mx tidak sama dengan mn, maka warna berupa *chromatic*. Kemudian nilai *saturation* dan *hue* di*huting* dengan cara:
     a.) Hitung nilai s sebagai saturation dengan rumus:
     ⇒ Jika l kurang dari atau sama dengan 0.5, maka s = (mx -mn) / (mx + mn)
     ⇒ Jika l lebih besar dari 0.5, maka s= (mx-mn) / (2 - mx+mn)
     b.) Kemudian hitung nilai h sebagai hue, dengan rumus:
     ⇒ Pertama ubah nilai r,g,dan b dengan rumus:
     r = (mx - r) / (mx - mn)
     g = (mx - g) / (mx - mn)
     b = (mx - b) / (mx - mn)
     ⇒ Adakan kondisi, yaitu
     - Jika r = mx, maka h = (g-b)/(mx-mn)
     - Jika g = mx, maka h = 2.0 + (b-r)/(mx-mn);
     - Jika b = mx , maka h = 4.0 + (r-g)/(mx-mn);

     ⇒ Hitung h= h*60 untuk mendapatkan besar dalam derajat karena h berupa sudut.
     ⇒ Jika h< 0, maka h=h+360.

LISTING PROGRAM

```
#include<iostream.h>
#include<ctype.h>
#include<conio.h>


class RGBColor{
public:
        double  r, g, b;
};
class HLSColor{public: double  h, l , s;};

double maks(double r,double g,double b)
{
        double m;
        if (r>g) m=r;
        else m=g;
        if (m>b) m=m;
        else m=b;
        return m;
};

double min(double r,double g,double b)
{
        double m;
        if (r<g) m=r;
        else m=g;
        if (m<b) m=m;
        else m=b;
        return m;
};


void RGB_to_HLS(RGBColor rgb, HLSColor& hls)
{
        // convert (r,g,b), each in[0, 1], to h,l,s
        double    mx, mn;
        RGBColor          tmp;
        mx = maks(rgb.r,rgb.g,rgb.b);
        mn = min(rgb.r,rgb.g,rgb.b);
        hls.l = (mx +mn) / 2.0;       //compute lightness
        if (mx == mn) // compute saturation
        {
          hls.s = 0.0; //color is gray
          hls.h=0.0;
        }
        else
        {           //color is chromatic
                    if(hls.l <= 0.5)  hls.s = (mx -mn) / (mx + mn);
                    else hls.s = (mx-mn) / (2 - mx+mn);
                    //compute hue
                    tmp.r = (mx   -   rgb.r)  /  (mx - mn);
                    tmp.g = (mx  -  rgb.g)  /  (mx - mn);
                    tmp.b = (mx  -  rgb.b) /  (mx - mn);
                    if (rgb.r == mx) hls.h = tmp.b -tmp.g;
                    else if (rgb.g == mx) hls.h = 2 +tmp.r - tmp.b;
                    else if (rgb.b == mx) hls.h = 4 + tmp.g - tmp.r;
                    hls.h *= 60;
                    if(hls.h < 0.0)
```

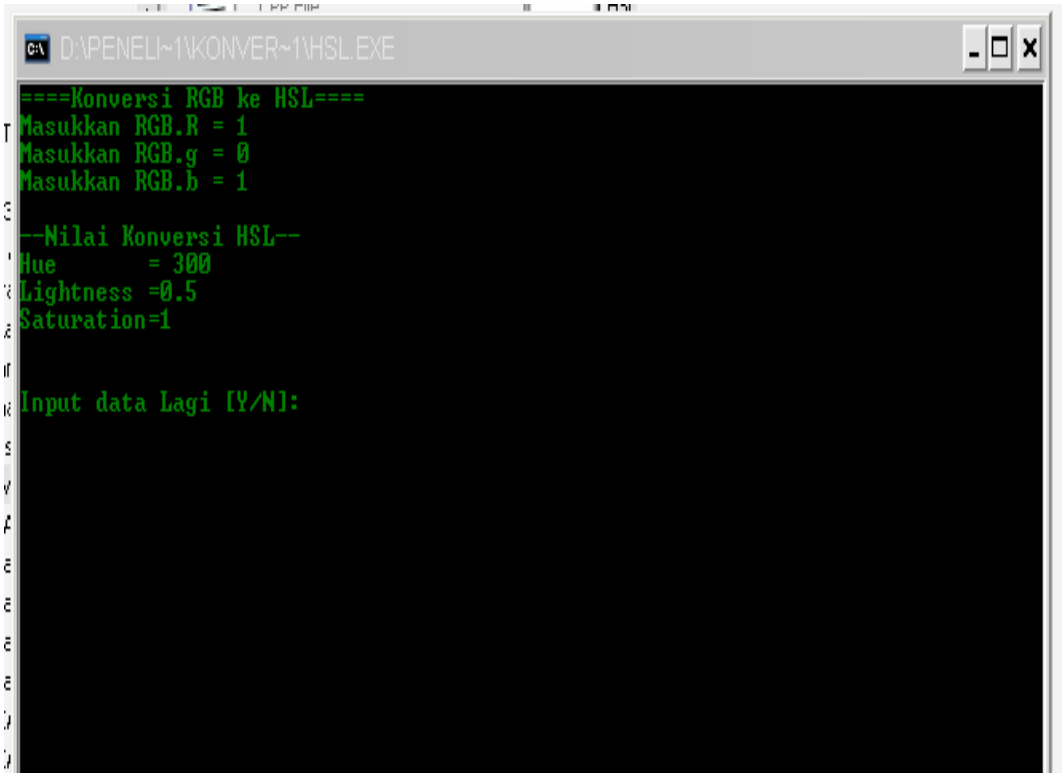
```
                        hls.h +=360;
        }

}

void main()
{
        char pil='N';
        clrscr();
        do{
        HLSColor hlsc;
        RGBColor rgbc;
        cout<<endl<<"====Konversi RGB ke HSL===="<<endl<<endl;
        cout<<"Masukkan Nilai RGB antara 0 sampai 1: "<<endl;
        cout<<"Red =";
        cin>>rgbc.r;
        cout<<"Green =";
        cin>>rgbc.g;
        cout<<"Blue =";
        cin>>rgbc.b;
        if ((rgbc.r < 0) || (rgbc.r > 1)||(rgbc.g < 0) || (rgbc.g > 1)||(rgbc.b < 0) || (rgbc.b > 1))
        {
                cout <<"Maaf anda salah input!!!"<<endl;
        }
        else
        {
                RGB_to_HLS(rgbc,hlsc);
                //if (hlsc.h<0 || hlsc.h>360)
                  //        cout<<"hlsc.h tidak terdefinisi"<<endl;
                //else
                cout<<endl;
                cout<<"Hue        = "<<hlsc.h<<endl;
                cout<<"Lightness  = "<<hlsc.l<<endl;
                cout<<"Saturation = "<<hlsc.s<<endl;
        }
        cout<<"Input data Lagi [Y/N]: ";
        pil=toupper(getch());
        }while (pil=='Y');
        //RGB_to_HLS(rgbc,hlsc);

}
```

OUTPUT PROGRAM

```
====Konversi RGB ke HSL====

Masukkan Nilai RGB antara 0 sampai 1:
Red =1
Green =0
Blue =1

Hue        = 300
Lightness  = 0.5
Saturation = 1
Input data Lagi [Y/N]:

====Konversi RGB ke HSL====
```

Masukkan Nilai RGB antara 0 sampai 1:
Red =0
Green =0
Blue =1

Hue = 240
Lightness = 0.5
Saturation = 1
Input data Lagi [Y/N]:



Gambar 2. Output

4. PENUTUP

Dalam grafik komputer terdapat pembahasan tentang konversi model warna yang berorientasi *hardware* ke model warna yang berorientasi software. Berikut ini adalah model warna yang berorientasi hardware:

- Model RGB (red, green, blue) untuk warna monitor dan warna pada kamera video.
- Model CMY (cyan, magenta, yellow) untuk model printer.
- Model YIQ model, digunakan untuk standard televisi. Y berkoresponden dengan luminasi, I dan Q adalah dua komponen kromatik yang disebut inphase dan quarature .

Berikut ini adalah model warna berorientasi *Software* (hue, saturation,brightness) adalah manipulasi :

- Model HSV (hue, saturation, value).
- Model HSI (hue, saturation, intensity).
- Model HLS (hue, lightness, saturation).

HLS merupakan singkatan dari Hue, Lightness, Saturation. Hue merupakan warna dominan dalam angle (warna dominan yang dinyatakan dengan sudut). Saturation adalah banyaknya campuran warna yang diminan dengan putih (purity). Sedangkan Lightness adalah terang atau gelapnya intensitas warna. Maksimal percampuran untuk warna-warna dominan sampai dengan S= 1 dan L = 0,5.

Makalah di atas membahas konversi warna dari RGB ke HLS dan sebaliknya menggunakan bahasa penrograman C++. Ke depan kami akan kembangkan makalah ini ke bahasa pemrograman berbasis web atau dengan teknologi mobile.

5. DAFTAR PUSTAKA

1. A.M. Tannenbaum, Yediyah Langsam, Meshe J. Augenstein,1955, *Data Structures Using C*, Prentice Hall International Editions
2. Braun, 1986, *grafik Programmierung*, Data Becker GmbH, Duesseldorf.
3. Johson,Nelson,1987, *Graphics in C Programming and Techniques,* McGraw-Hill, Internasional Edition.
4. http://www.opengl.org
5. http://www.planetsourcode.com
6. http://www.acm.org/fcg/graphicsgems/index.html