

PROTÓTIPO DE VISUALIZADOR PARA MODELOS DE COR PARA MEDIÇÕES DE OBJETOS EM ESPECTROFOTÔMETROS POR REFLECTÂNCIA

Antônio Carlos Fernandes

Prof. Dalton Solano dos Reis
Orientador

Roteiro

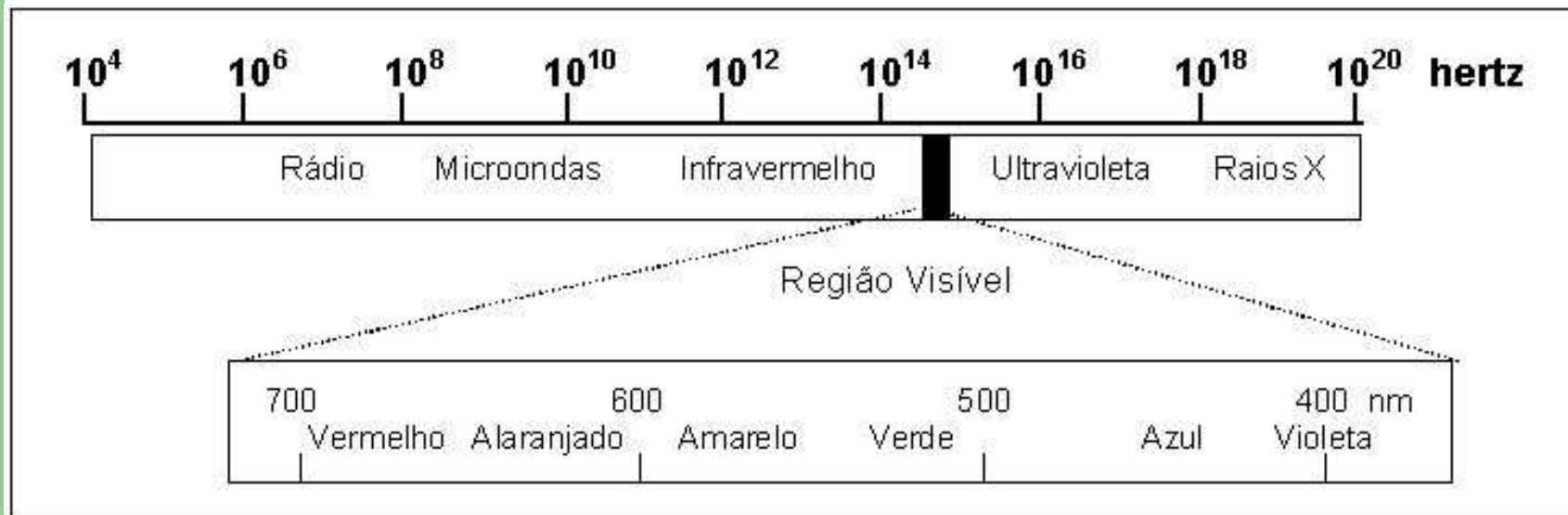
- Introdução
- Conceitos Básicos de Cores
- Atlas de Cores
- Valores Colorimétricos Normais
- Protótipo: especificação e implementação
- Conclusão
- Extensão

Introdução

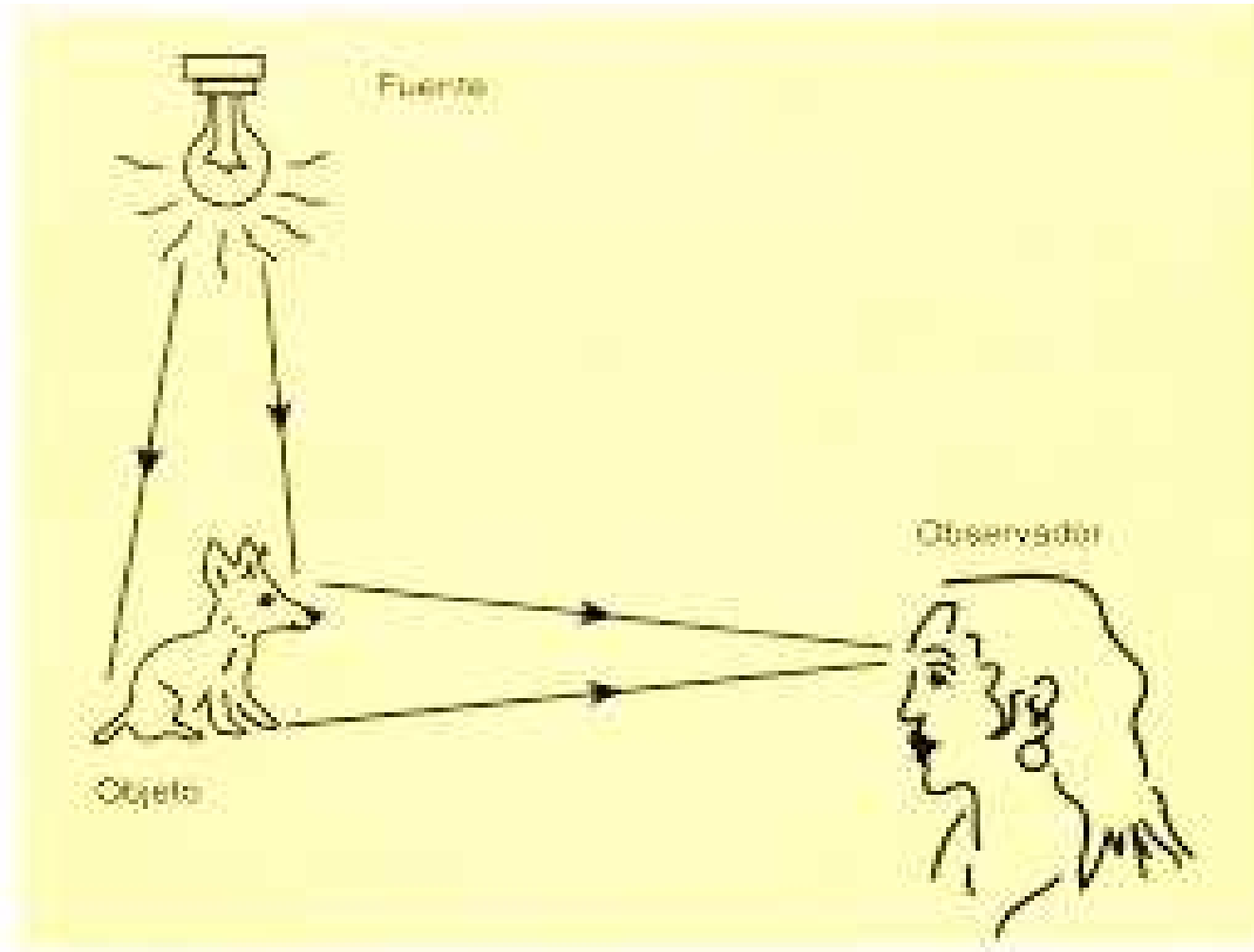
- Colorimetria
- Atlas de Cores
- Modelos de Representação de Cores

Conceitos Básicos de Cores

- Princípio do Processo da Percepção Visual



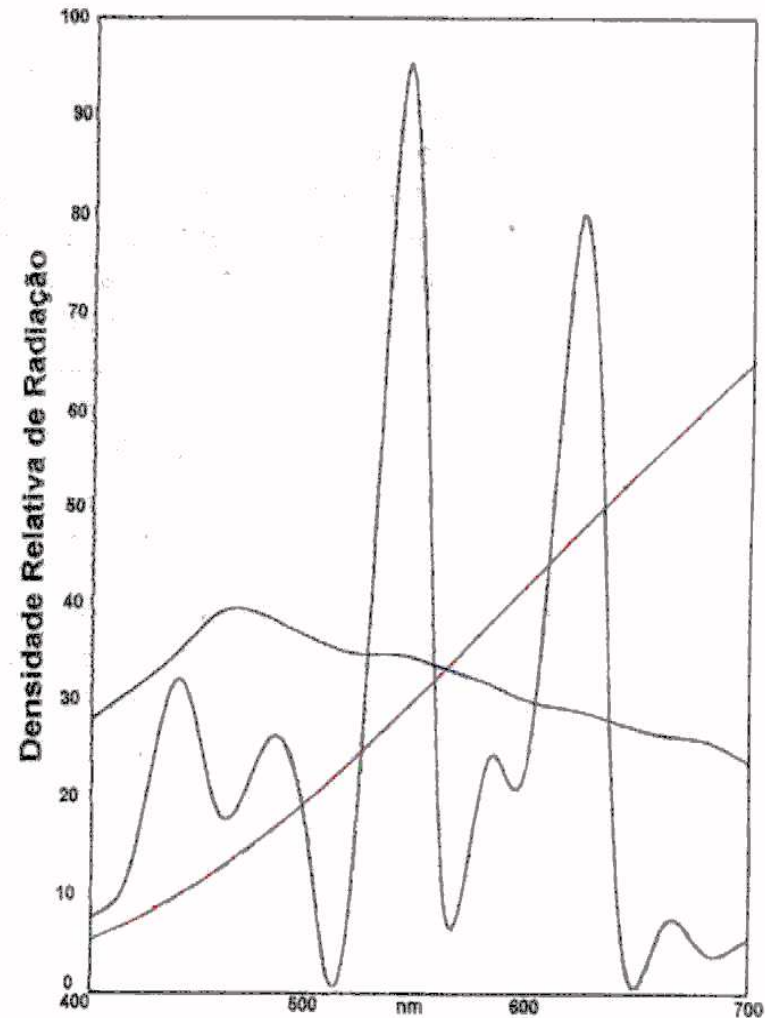
Conceitos Básicos de Cores



Conceitos Básicos de Cores

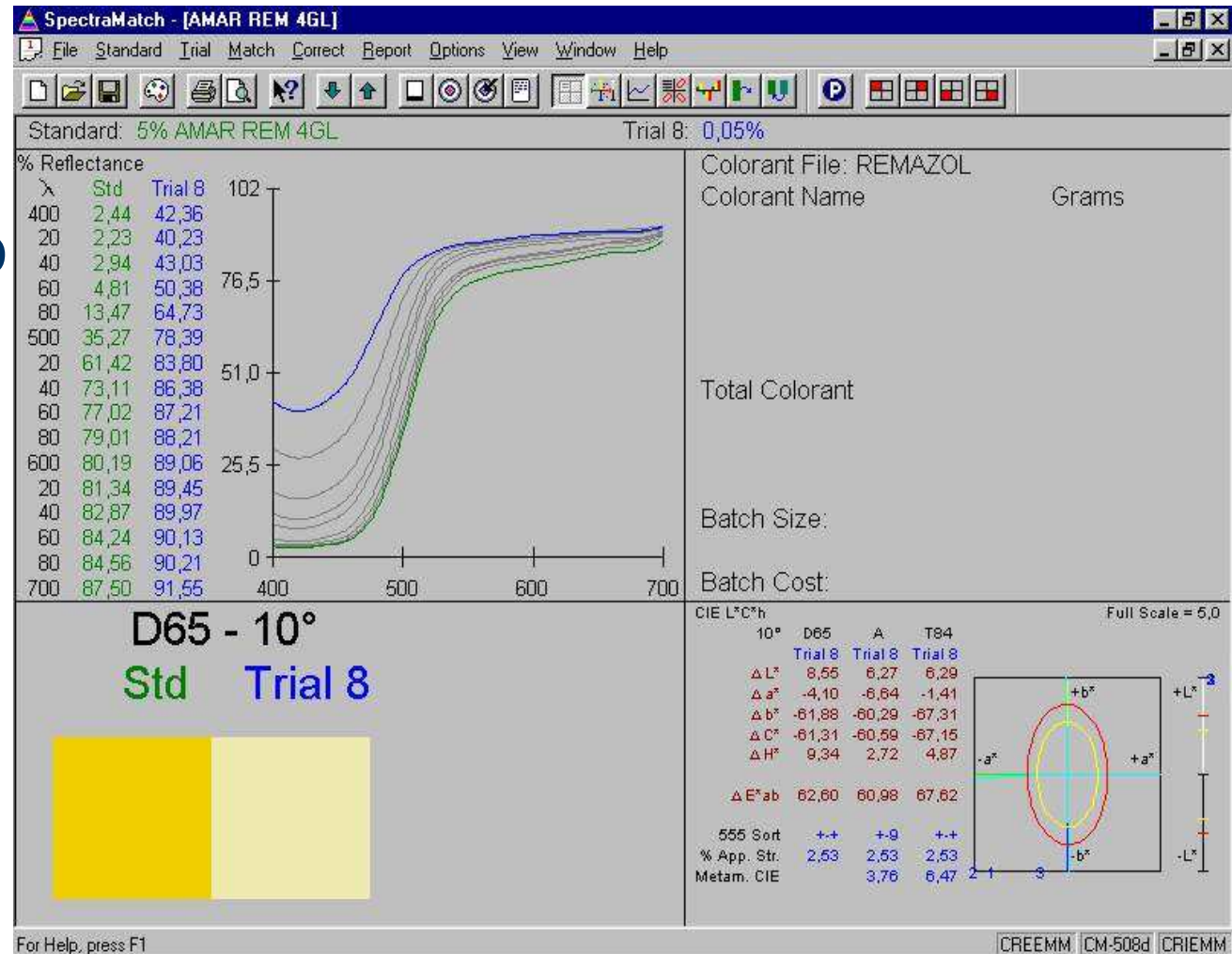
- Fontes Luminosas

— Luz do Dia D65
— Luz Incandescente A
— Luz Fluorescente Philips TL84



Conceitos Básicos de Cores

- Objeto Observado



Conceitos Básicos de Cores

- Observador

The image displays two side-by-side screenshots of the 'CIE Chromaticity Diagrams' software interface, illustrating the difference between two standard observer models.

Left Screenshot (1931 2-degree Observer):

- Observer:** CIE Standard Observer, 1931 2-degree (selected).
- Viewing Distance [mm]:** 450 (10 mm).
- Field of View:** 2-degree Field of View Circle, 15.7 mm diameter (65.9 pixels).
- Calibration Data:** Screen Diagonal Measure [mm] = 305. Resolution: 1024 pixels wide by 768 pixels high (244.0 mm wide by 183.0 mm high). Assumes square pixels.
- Notes:** The experiments leading to the 1931 standard observer were performed using only the fovea, which covers about a 2-degree angle of vision. According to [Foley96, p. 580], "the original 1931 tabulation is normally used in work relevant to computer graphics." CIE standard observers are averages.

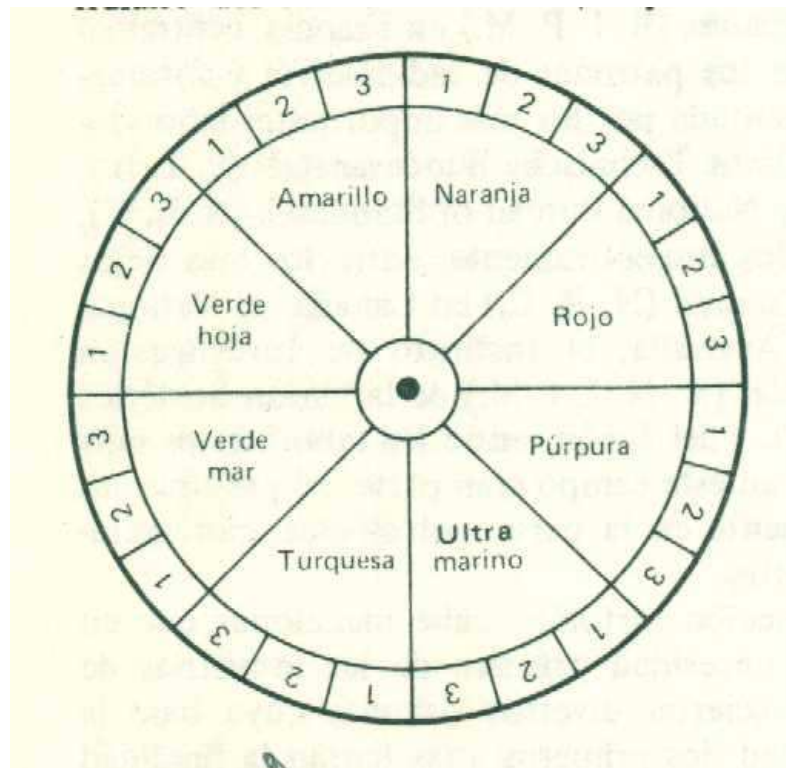
Right Screenshot (1964 10-degree Observer):

- Observer:** CIE Standard Observer, 1964 10-degree (selected).
- Viewing Distance [mm]:** 450 (10 mm).
- Field of View:** 10-degree Field of View Circle, 78.3 mm diameter (328.8 pixels).
- Calibration Data:** Screen Diagonal Measure [mm] = 305. Resolution: 1024 pixels wide by 768 pixels high (244.0 mm wide by 183.0 mm high). Assumes square pixels.
- Notes:** The 1964 supplementary standard observer was based on color-matching experiments using a 10-degree area on the retina. The observers were instructed to ignore the central 2-degree spot. The supplementary 1964 standard observer is recommended when visual perception of more than about 4-degrees.

Both screenshots are from 'efg's Computer Lab' and include the URL www.efg2.com/.

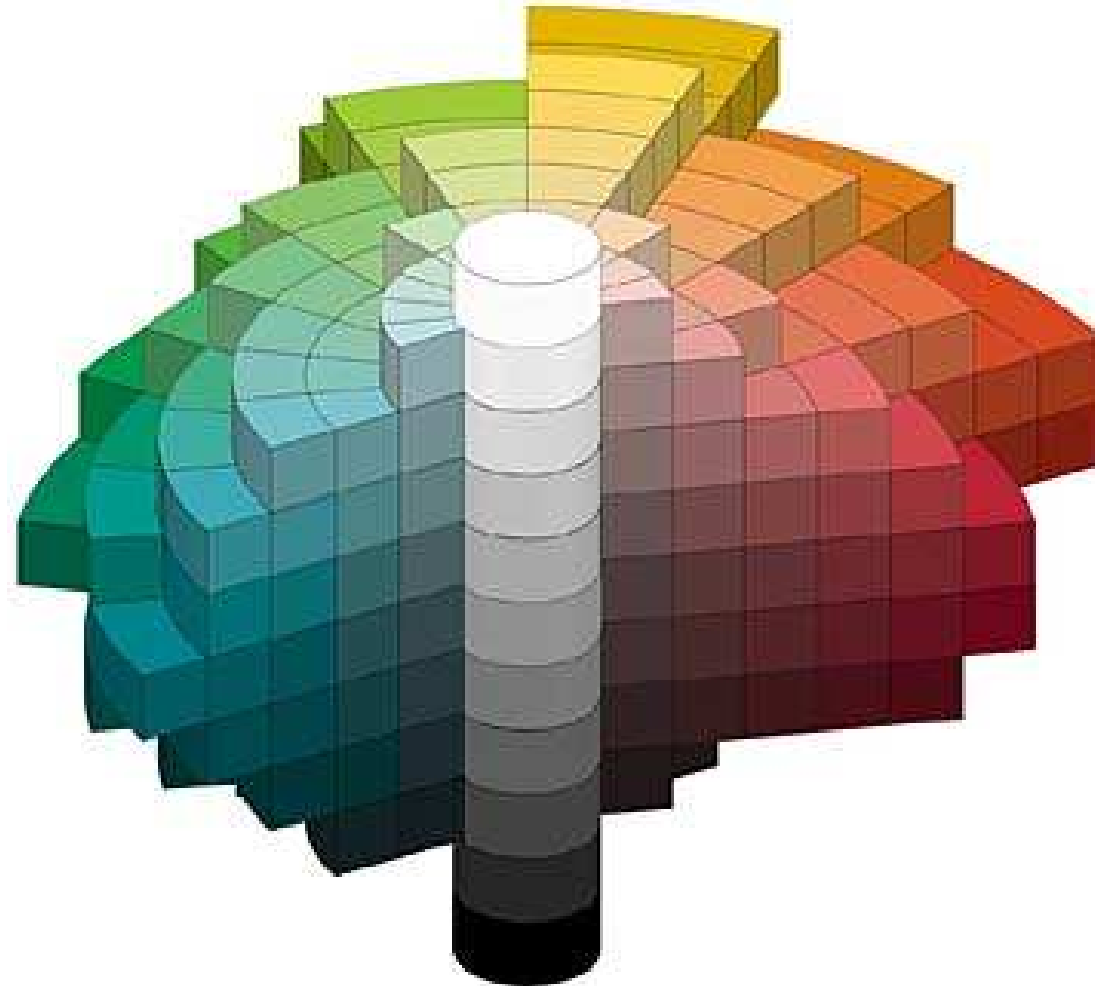
Atlas de Cores

- Sistema Ostwald



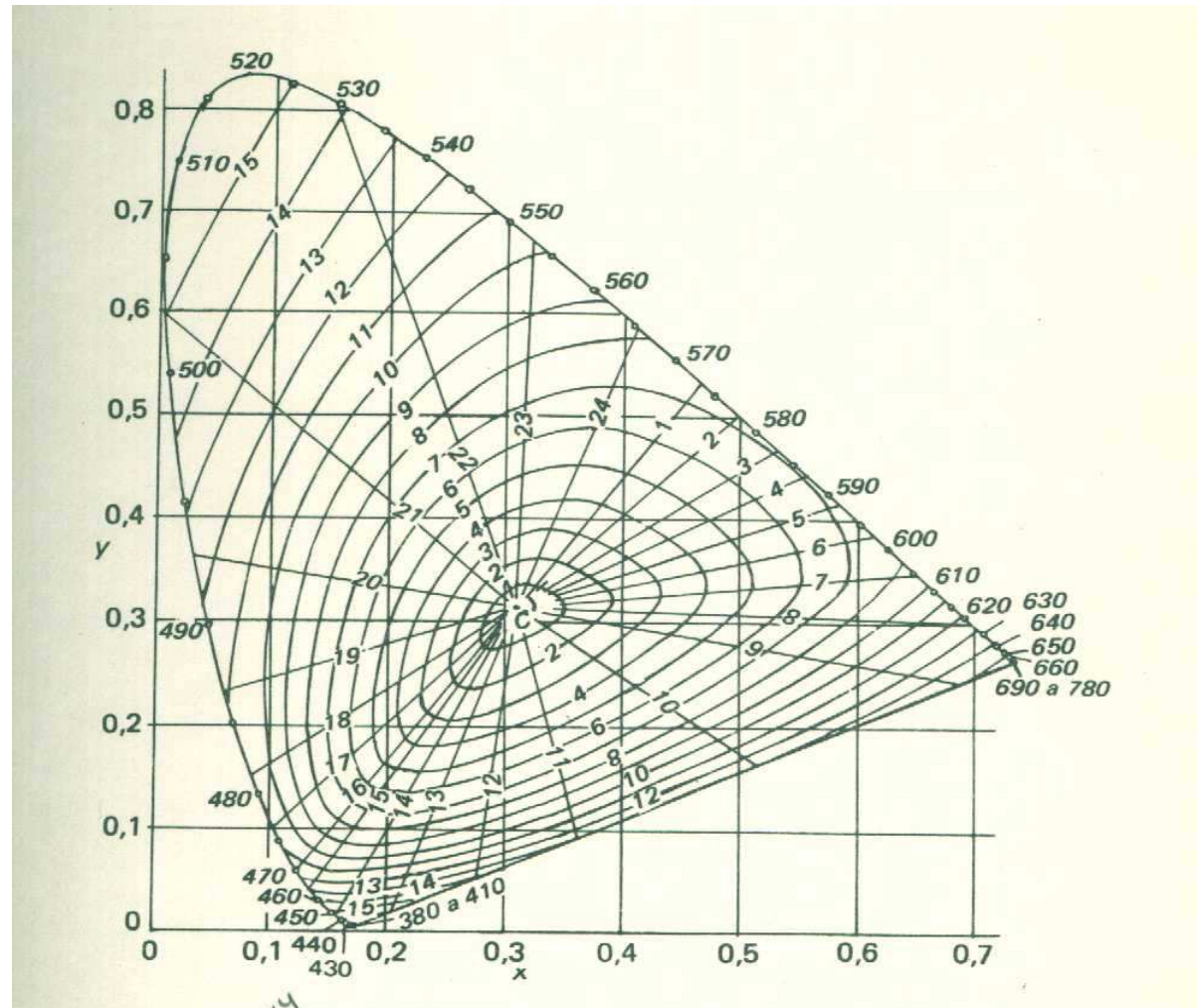
Atlas de Cores

- Sistema Munsell



Atlas de Cores

- Sistema DIN



Valores Colorimétricos Normais

- Valores Cromáticos Normais

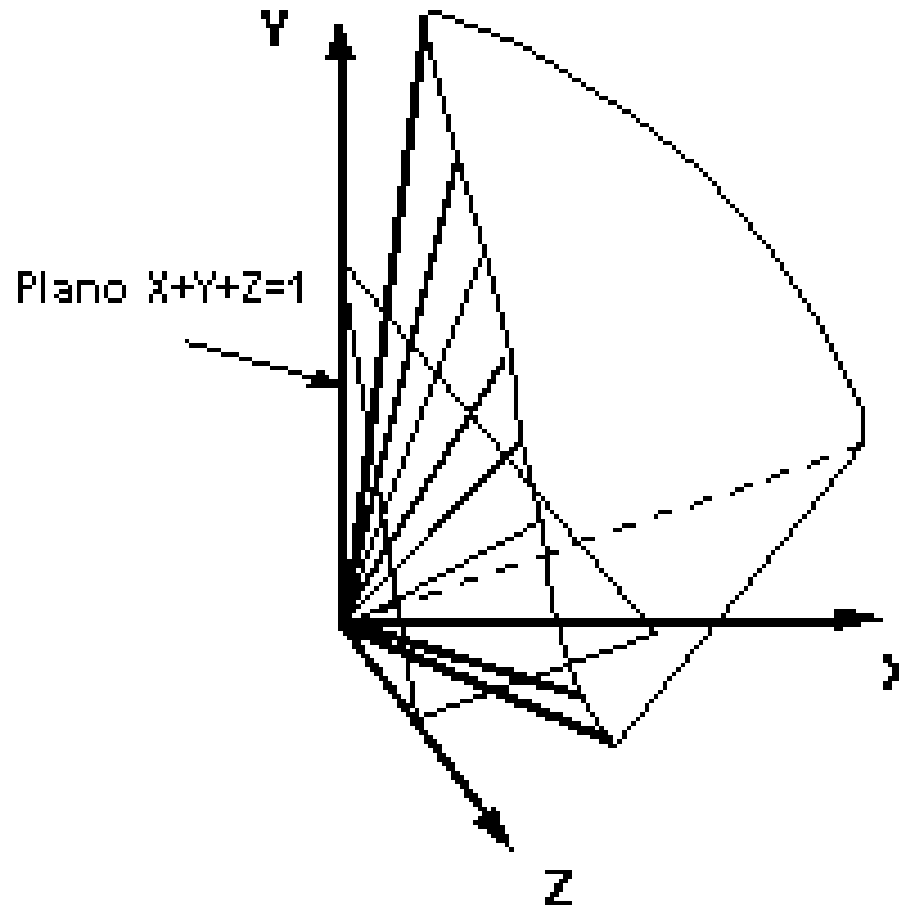
$$\rightarrow X = \sum E_{\lambda} x_{\lambda} R_{\lambda}$$

$$\rightarrow Y = \sum E_{\lambda} y_{\lambda} R_{\lambda}$$

$$\rightarrow Z = \sum E_{\lambda} z_{\lambda} R_{\lambda}$$

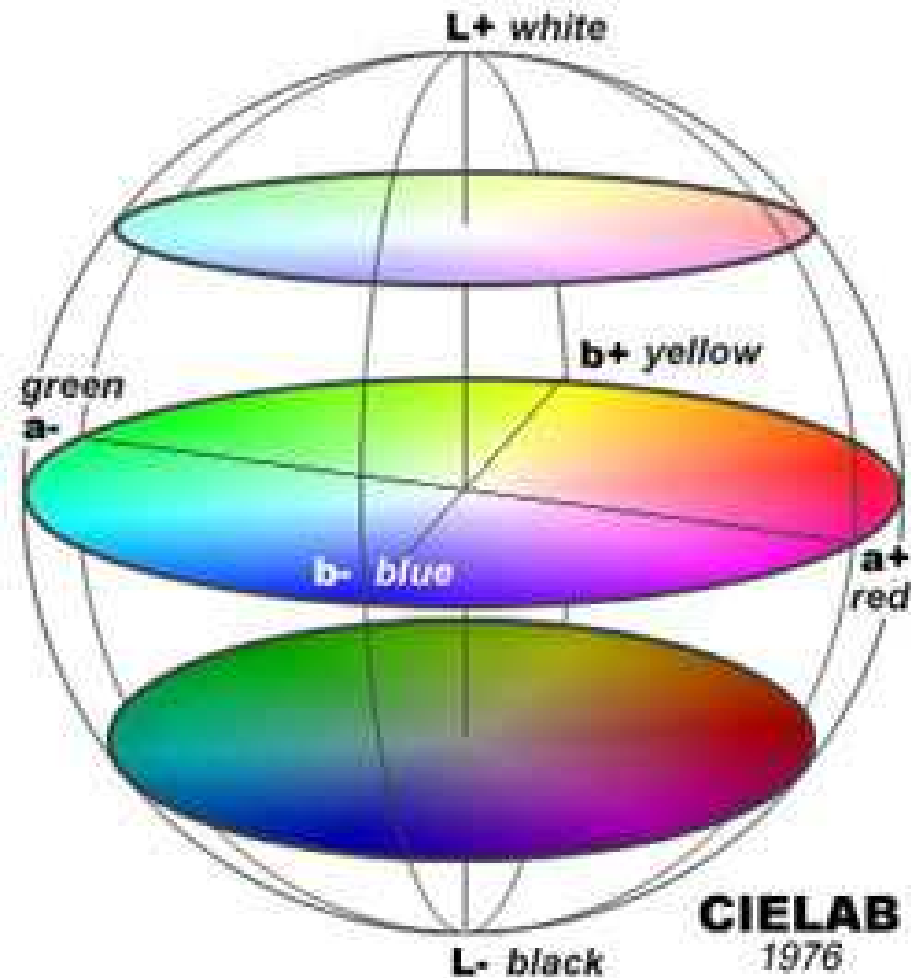
Valores Colorimétricos Normais

- Modelo XYZ



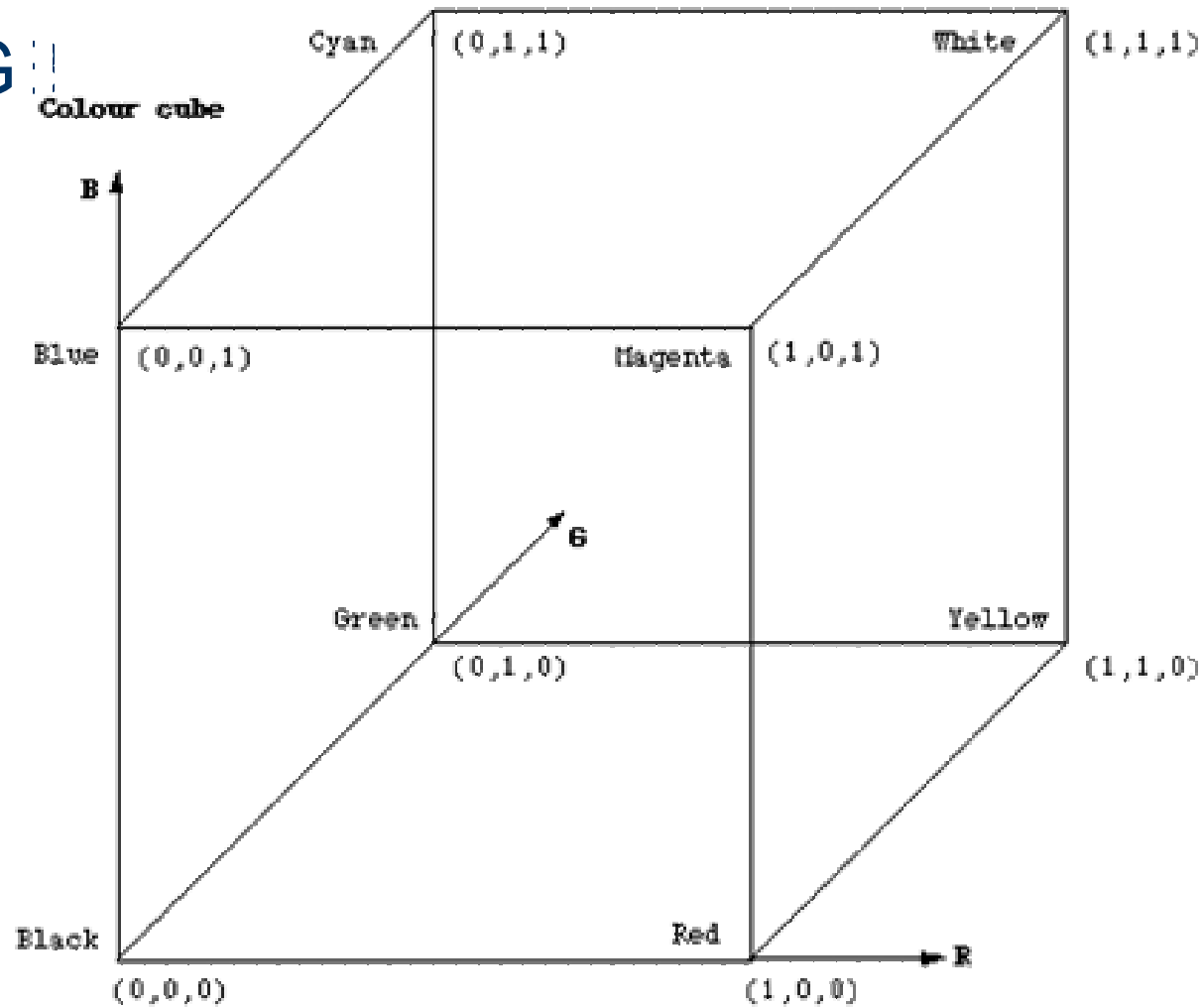
Valores Colorimétricos Normais

- Modelo $L^*a^*b^*$



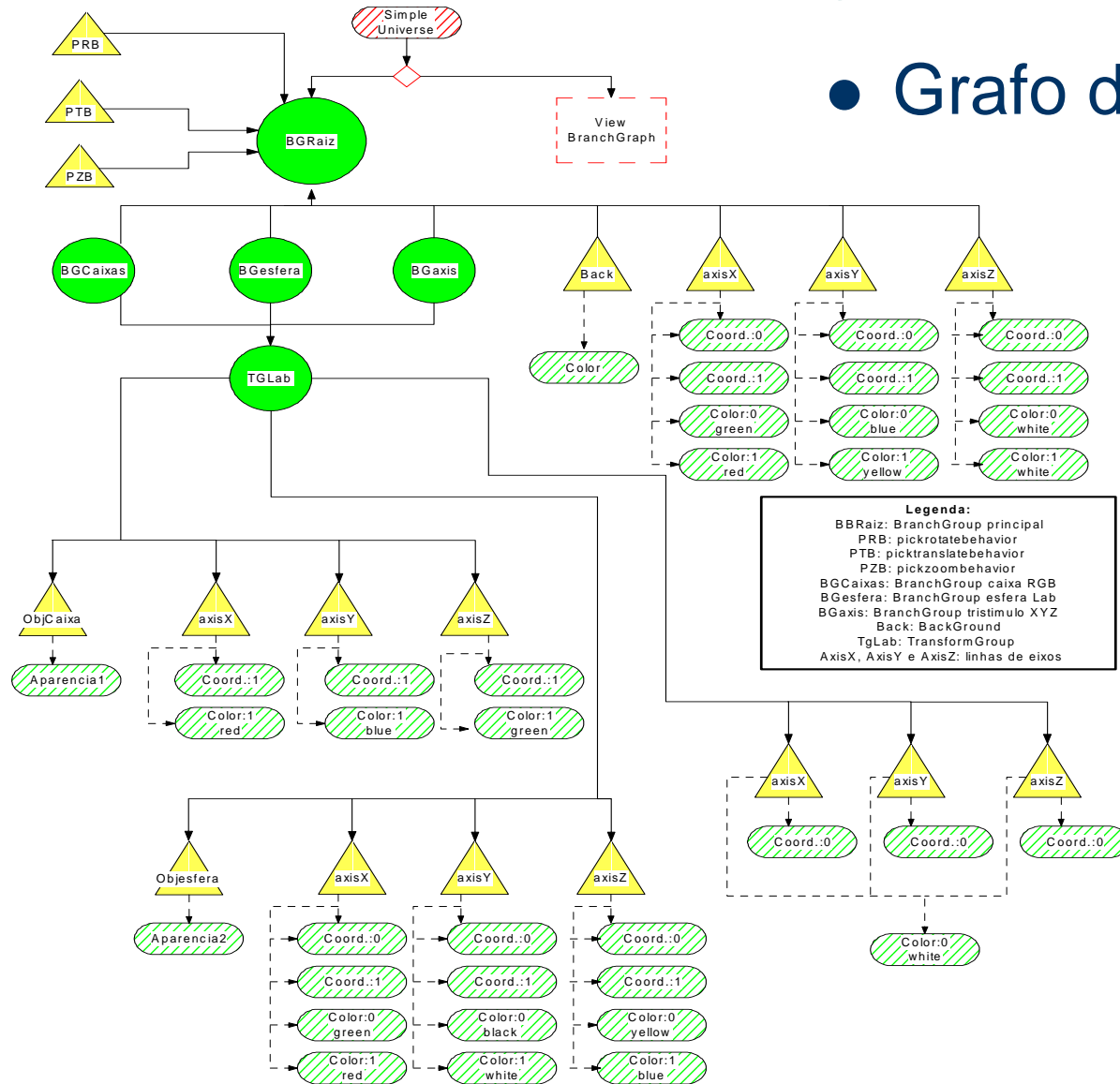
Valores Colorimétricos Normais

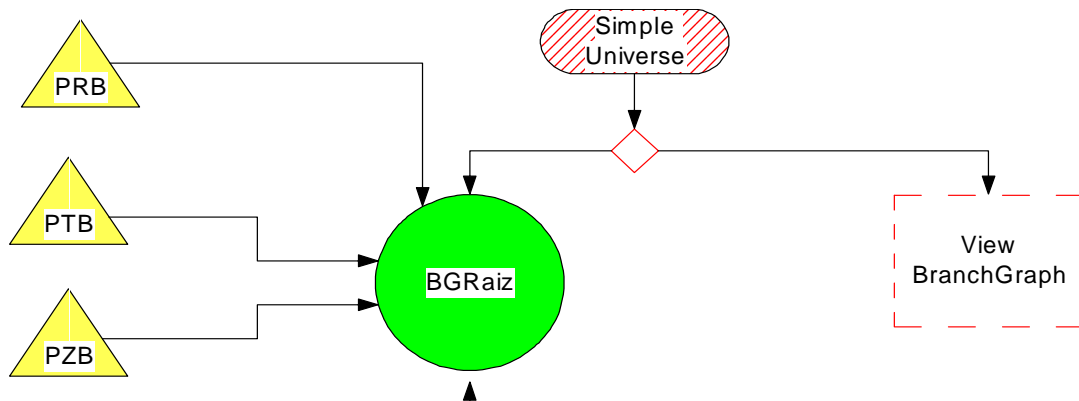
- Modelo RG



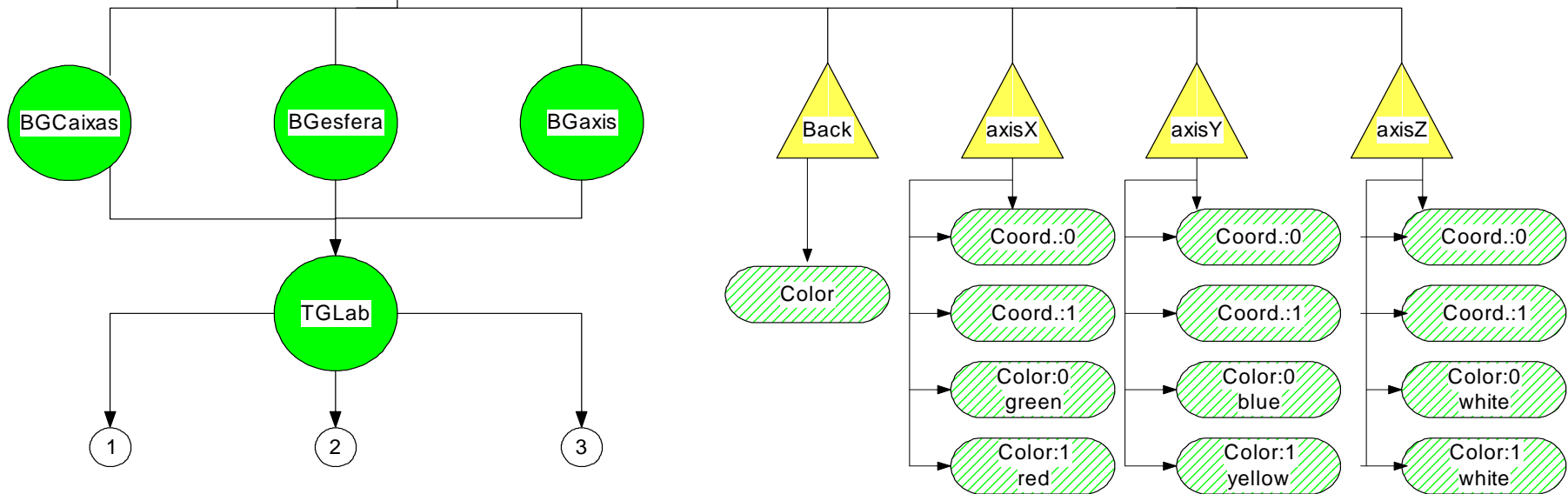
Protótipo - Especificação

● Grafo de Cena

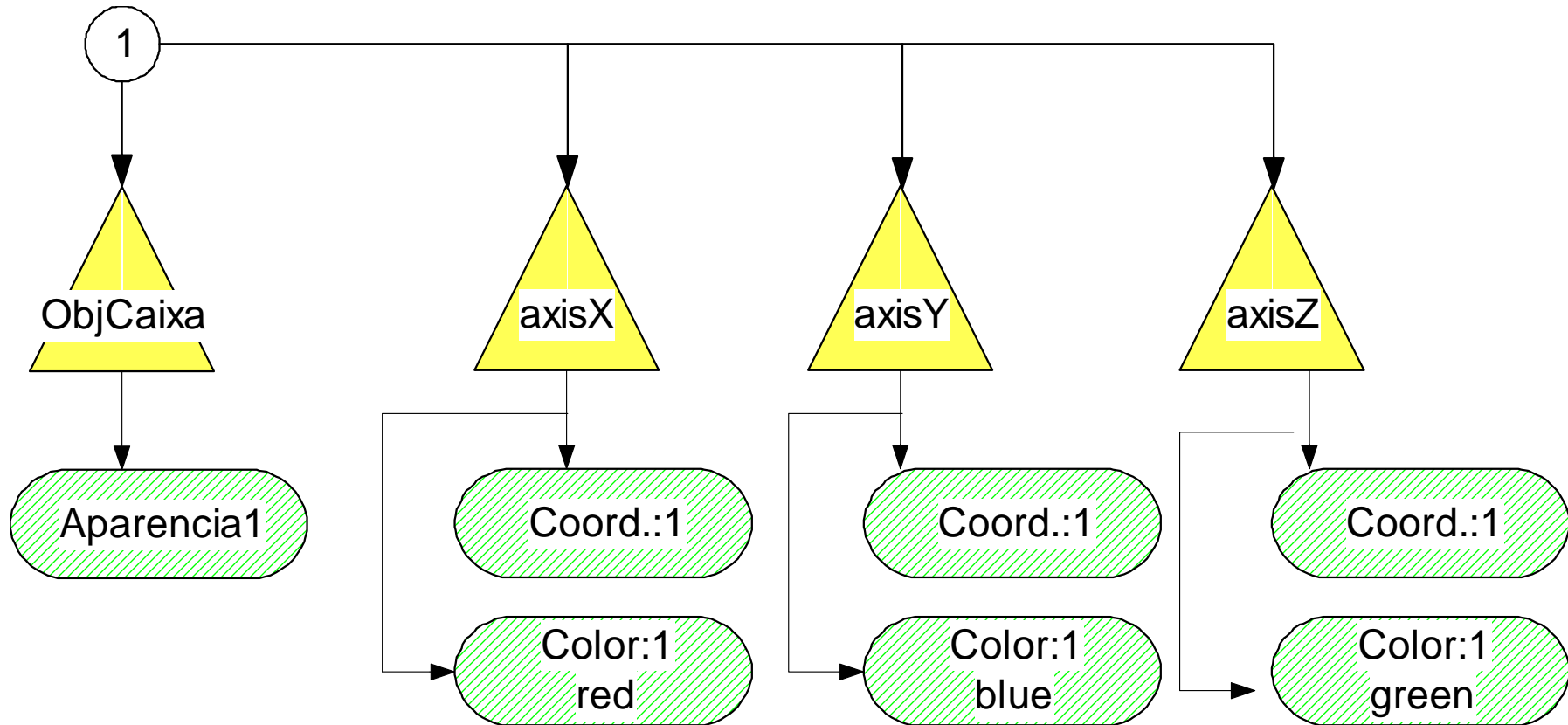




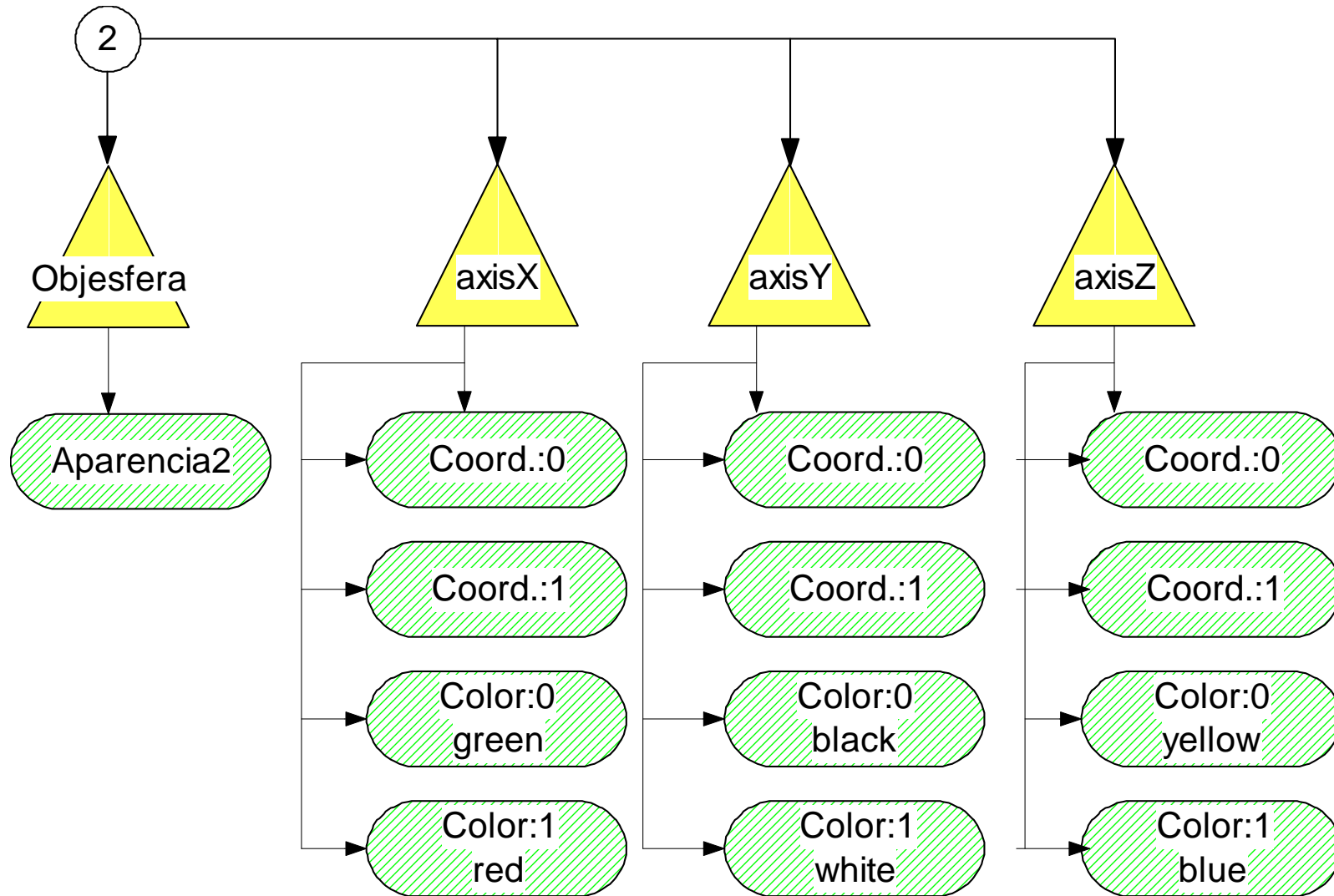
● Grafo de Cena -1



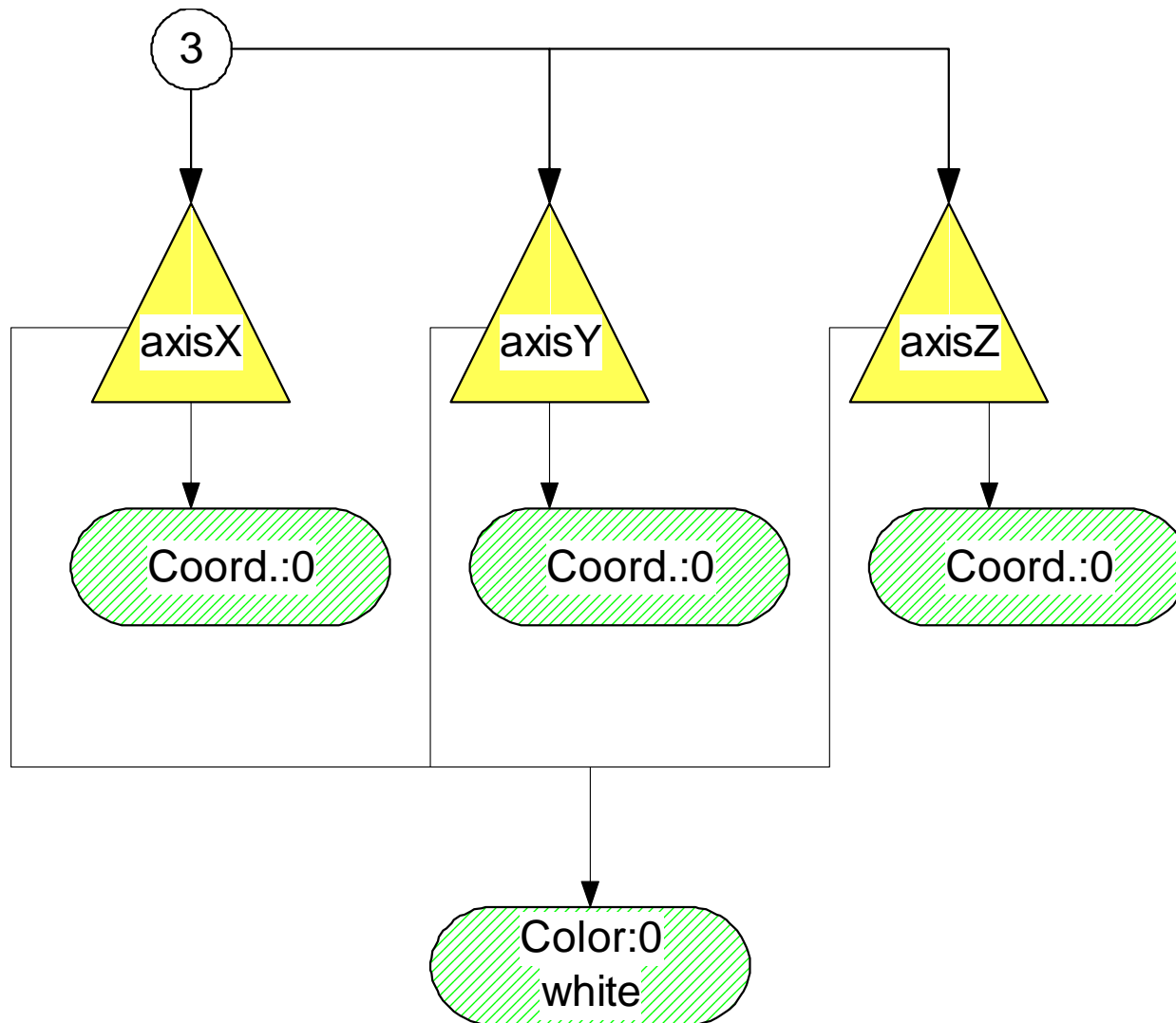
- Grafo de Cena – conector 1



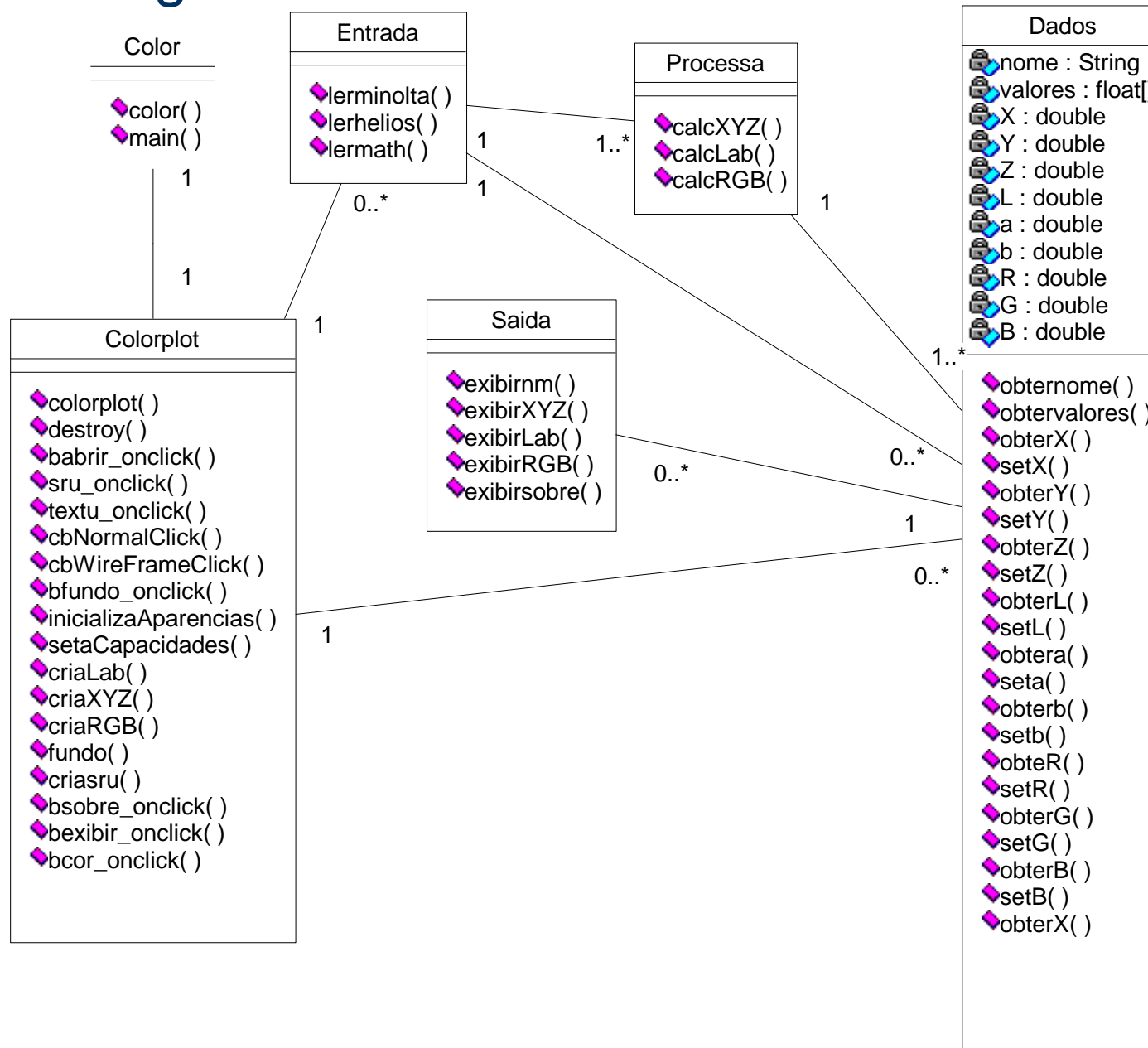
- Grafo de Cena – conector 2



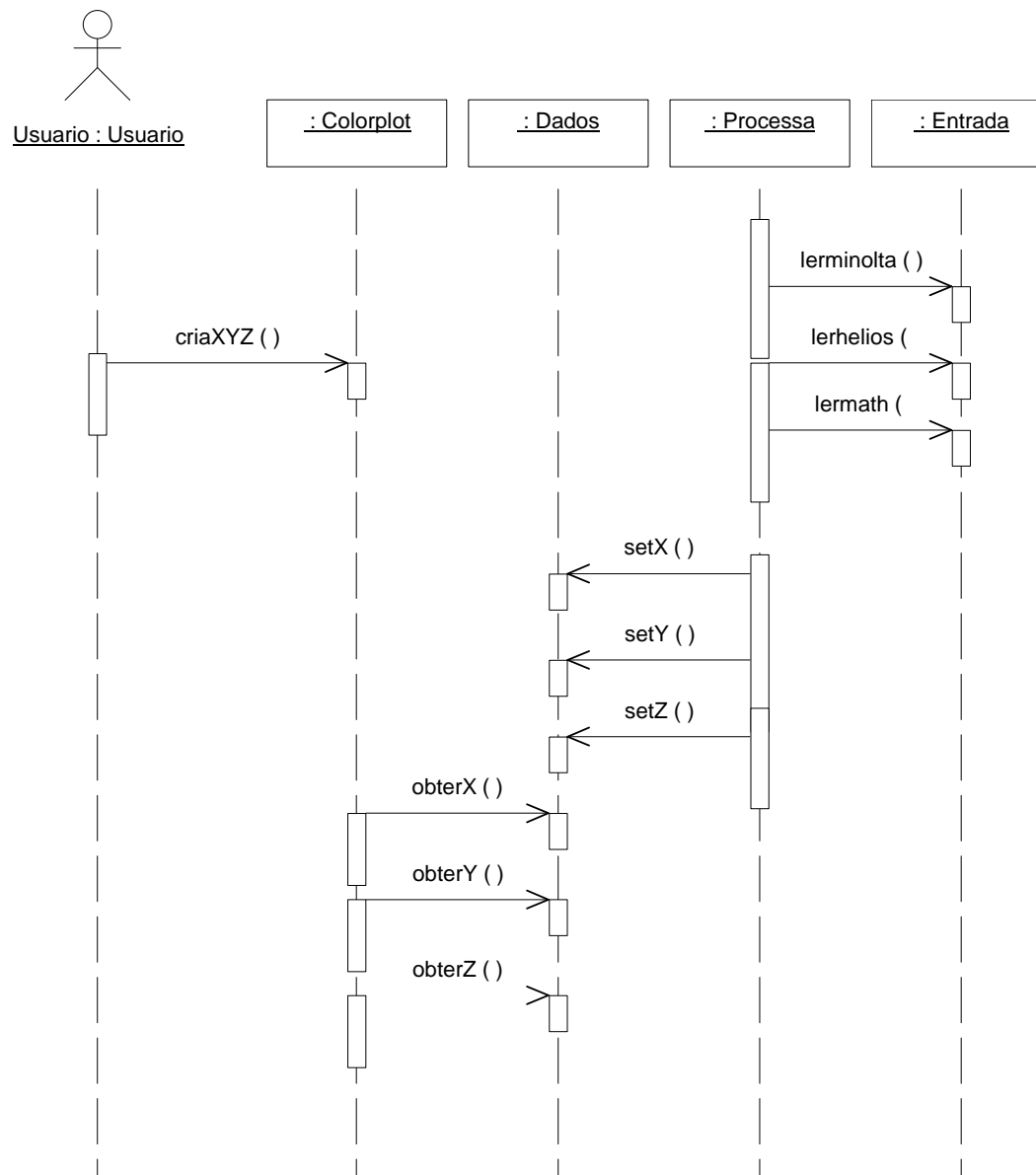
- Grafo de Cena – conector 3



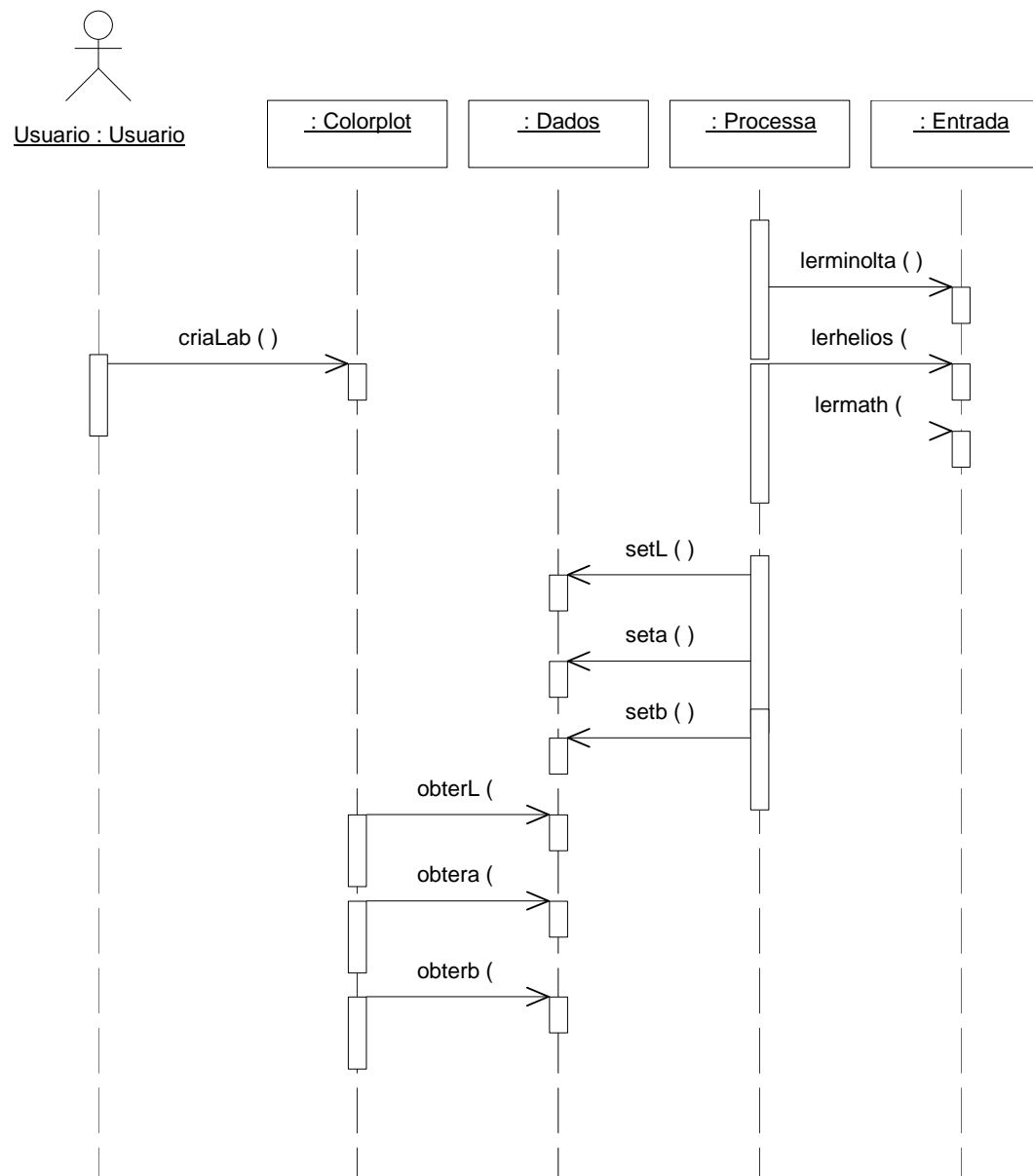
UML – Diagrama de Classes



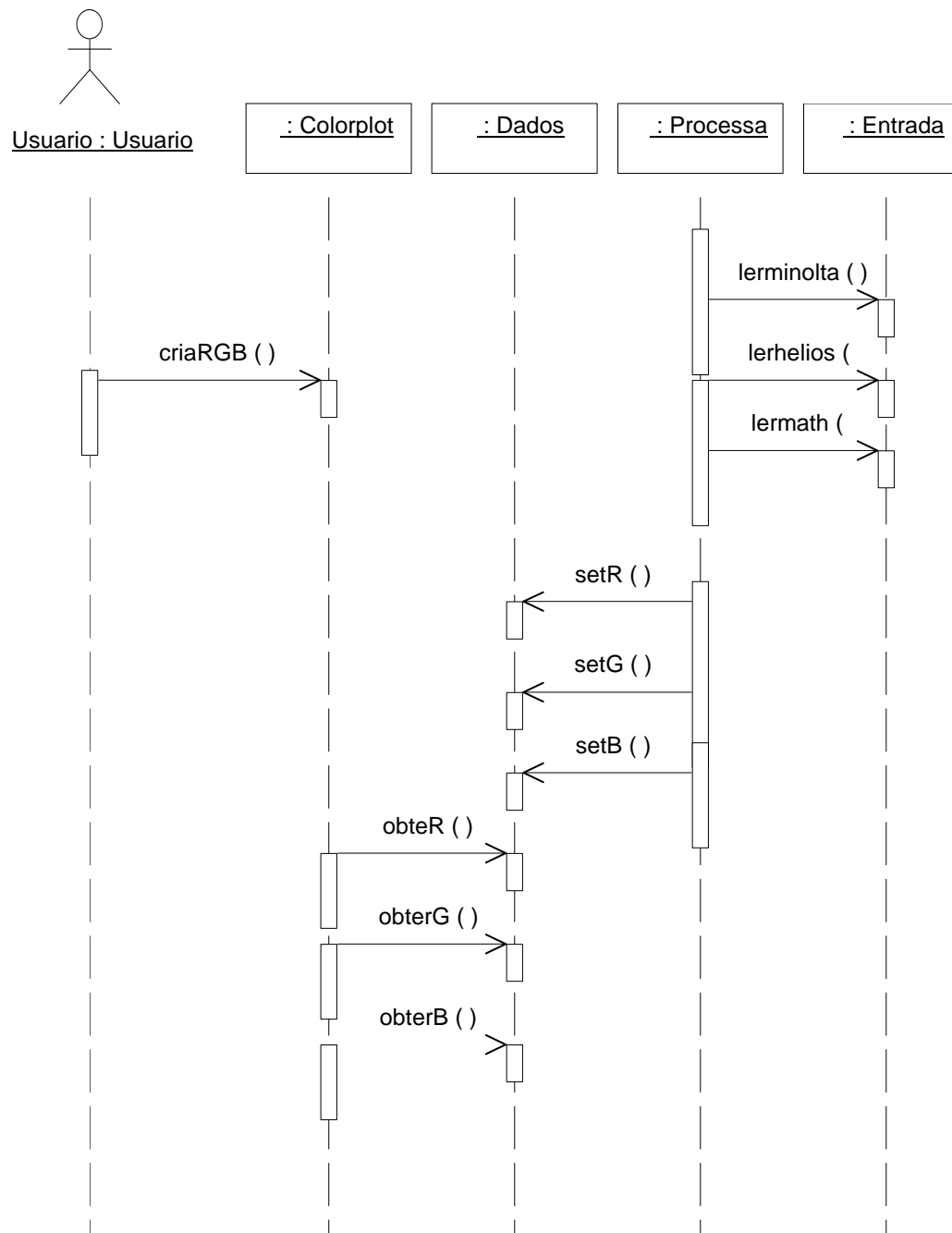
- UML – Diagrama de Sequência – método criaXYZ



- UML – Diagrama de Sequência – método criaLab



- UML – Diagrama de Sequência – método criaRGB



Protótipo - Implementação

- Método criaXYZ

```
void criaXYZ(Dados dados[]) {
    bgRaiz.detach();
    ...
    axisXLines.setCoordinate(1,
new Point3f( 1.5f, 0.0f,
0.0f));
axisXLines.setColor(0,whitec);
...
p.setCoordinate(0, new
Point3f(px, py, pz));
tglab.addChild(new Shape3D(p));
...
}
```

Protótipo - Implementação

- Método
criaLab

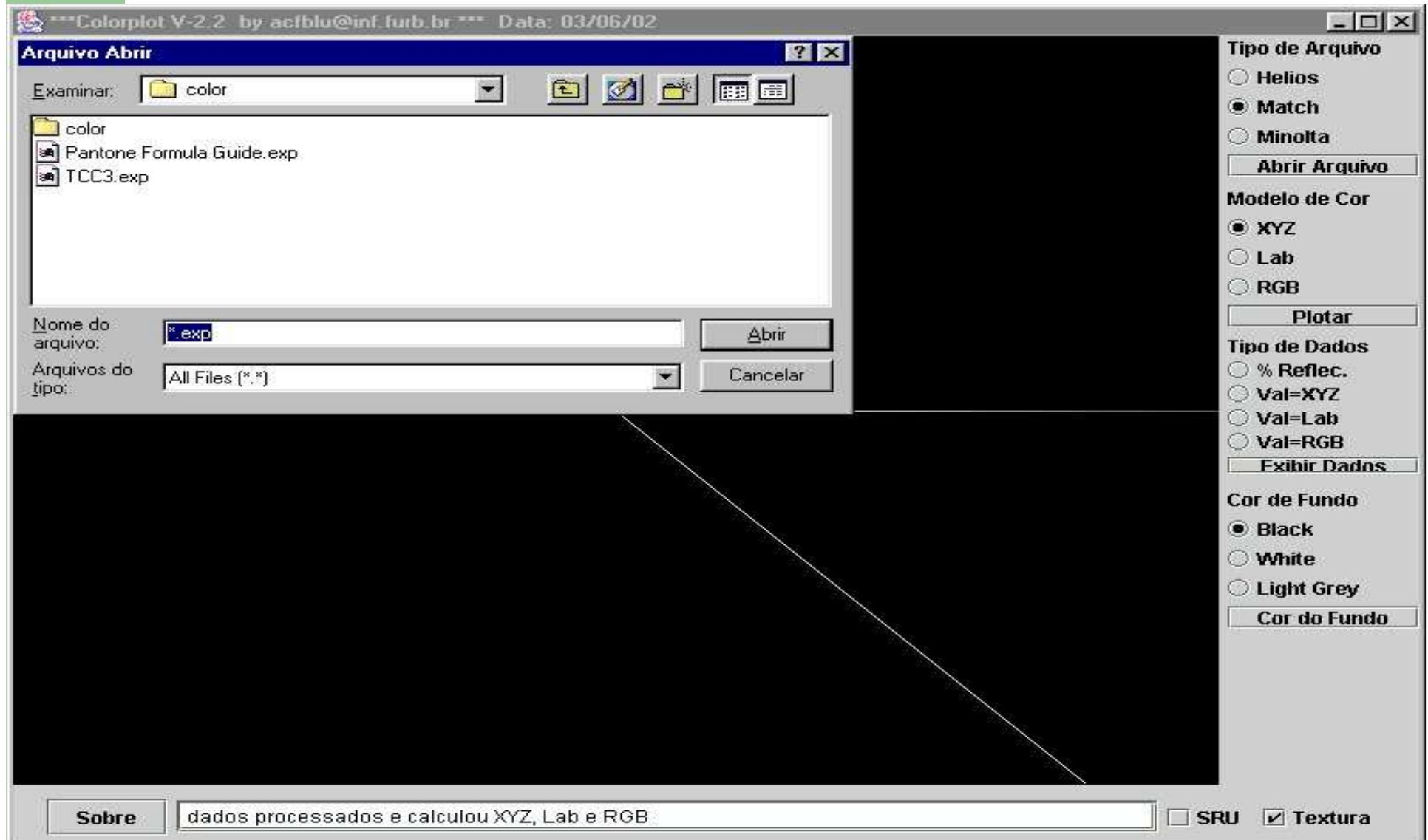
```
void criaLab(Dados dados[]) {  
    Sphere objEsfera = new Sphere(0.7f,  
    ap2);  
    ...  
    axisXLines.setCoordinate(0, new  
    Point3f(-1.0f, 0.0f, 0.0f));  
    axisXLines.setCoordinate(1, new  
    Point3f( 1.0f, 0.0f, 0.0f));  
    axisXLines.setColor(0,greenc);  
    axisXLines.setColor(1, redc);  
    ...  
    p.setCoordinate(0, new  
    Point3f(pa,pele,pb));  
    tglab.addChild(new Shape3D(p));  
}
```

Protótipo - Implementação

- Método
criaRGB

```
void criaRGB(Dados dados[]) {  
    com.sun.j3d.utils.geometry.Box  
    objCaixa = new  
    com.sun.j3d.utils.geometry.Box(0.5f,  
    0.5f, 0.5f, ap1);  
    ...  
    axisXLines.setCoordinate(1, new  
    Point3f( 1.0f, 0.0f, 0.0f));  
    axisXLines.setColor(0,redc);  
    ...  
    p.setCoordinate(0, new Point3f(pr, pb,  
    pg));  
    tglab.addChild(new Shape3D(p));  
    ...  
}
```

Protótipo - Implementação



Conclusão

- Foi possível a execução do protótipo da área em estudo
- Importância das cores e sua representação
- Interdisciplina no estudo científico e aplicação
- API Java3D

Extensões

- Outras características do ambiente 3D
- Outros modelos de cores
- Criação de arquivos de imagens
- Leitura de dados de outros softwares
- Implementação de interação gráfica



Fim

Protótipo - Implementação

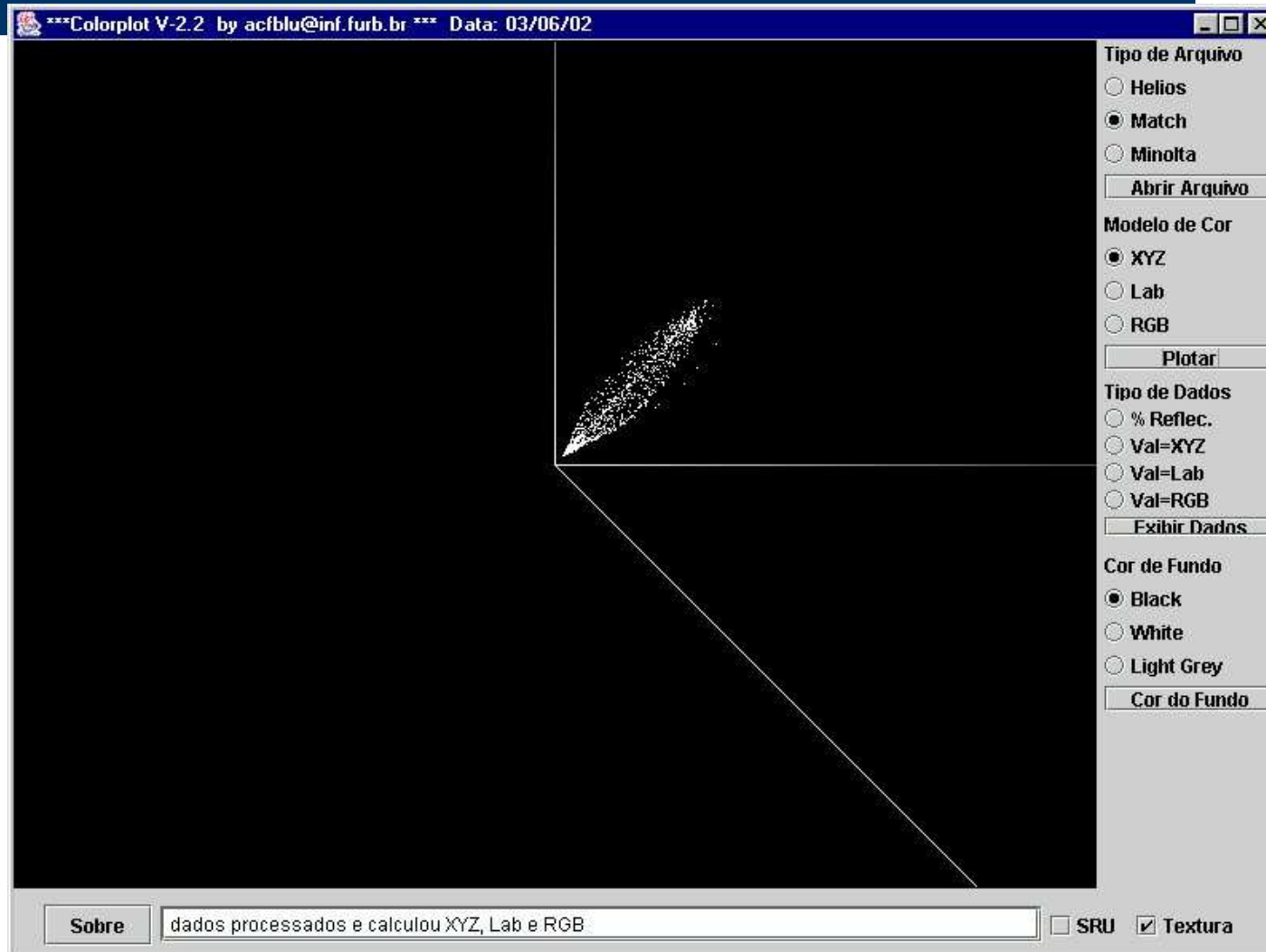
- Chamada para os modelos de cor

```
if (qtdeXYZ == -1) { } else {  
    bgRaiz.detach();  
    bgRaiz.removeChild(qtdeXYZ);  
    tglab.removeAllChildren();  
    universo.addBranchGraph(bgRaiz);  
    qtdeXYZ--;  
}
```

```
...  
if (cbrgb.isSelected()) {  
    criaRGB(dadose);  
}  
else if (cbxyz.isSelected()) {  
    criaXYZ(dadose);  
}
```

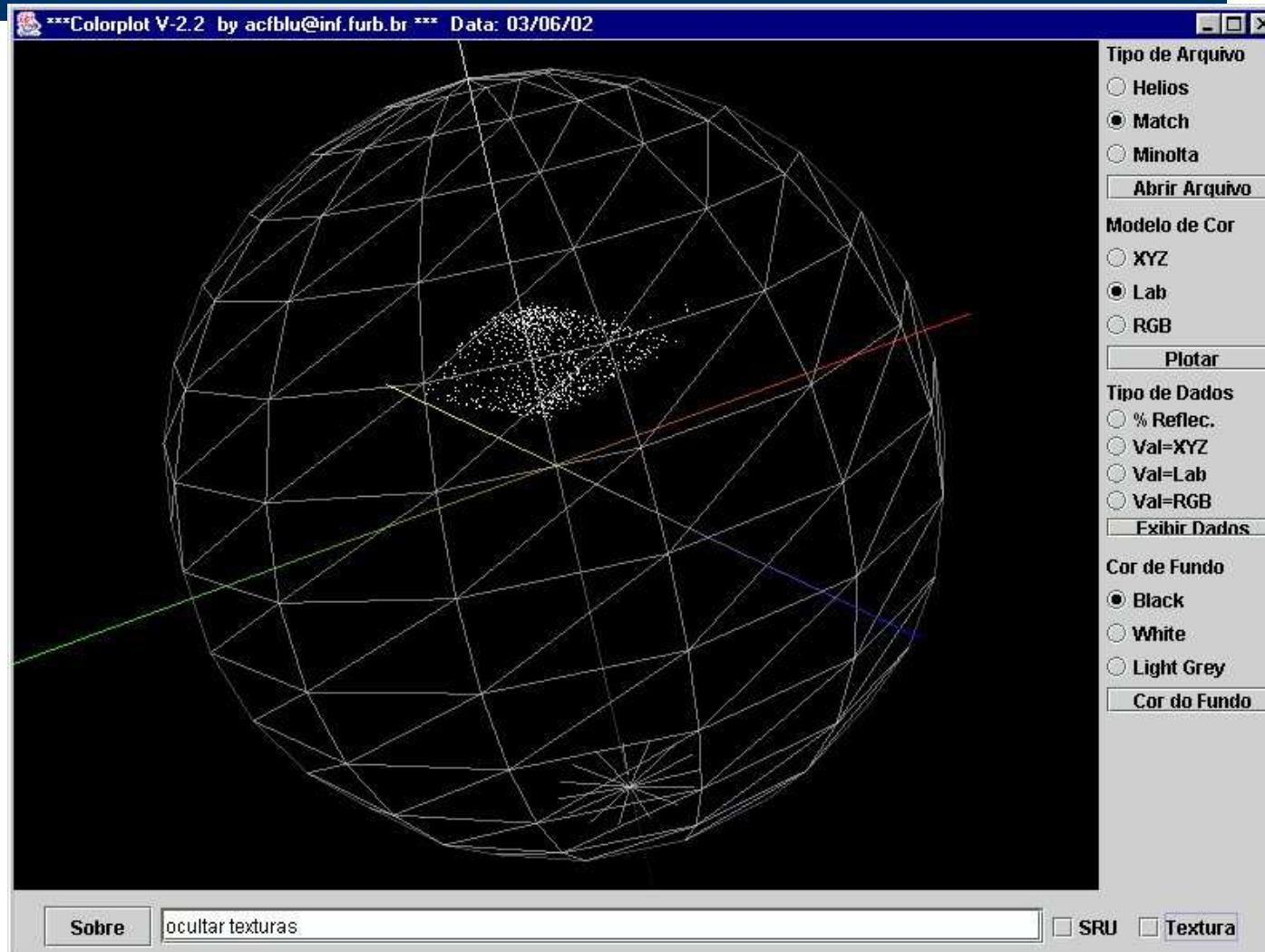

Protótipo - Implementação

- Modelo XYZ



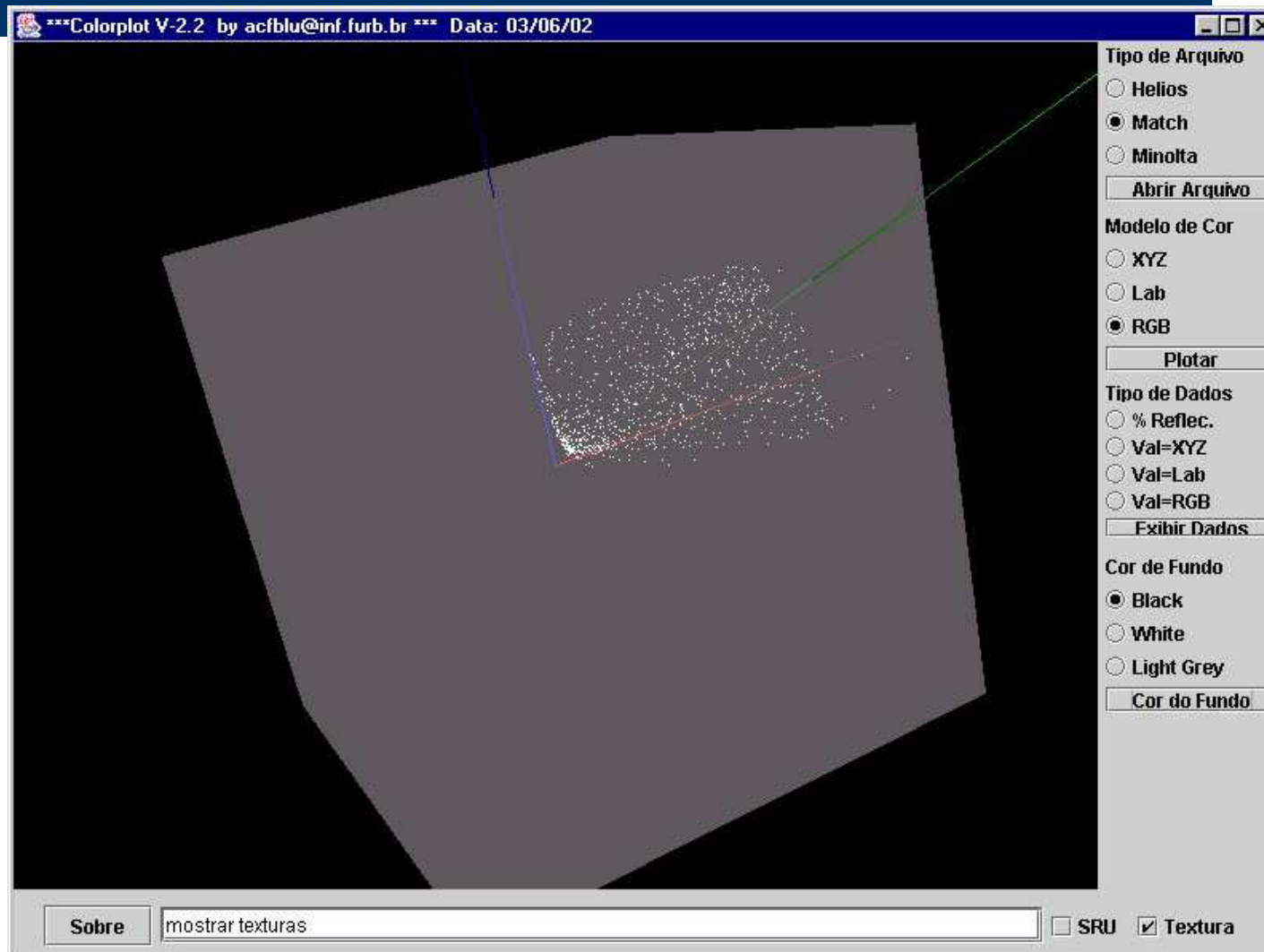
Protótipo - Implementação

- Modelo L*a*b*



Protótipo - Implementação

- Modelo RGB



Protótipo - Implementação

- calcXYZ

```
double Ex [] =
{0.136f,0.667f,1.645f,2.349f,3.464f,3.734f,3.066f
,1.934f,0.803f,0.152f,0.036f,0.348f,1.062f,2.192f
,3.386f,4.744f,6.069f,7.285f,8.36f,8.536f,8.706f,
7.945f,6.462f,4.641f,3.109f,1.808f,1.053f,0.576f,
0.276f,0.119f,0.059f};

.....
for (j=0;j<31;j++)          sumEx = sumEx
+(valores[j] * Ex[j]);
X = sumEx;
dados[i].setX(X);
```

Protótipo - Implementação

- calcLab

```
g[0] = dados[i].obterX()/94.827 ;
g[1] = dados[i].obterY()/100.0;
g[2] = dados[i].obterZ()/107.414;
.....
for (int j=0;j<3 ;j++)
    {if (g[j] <= 0.008856)  {
        f[j] = 7.787 * g[j] + (16.0/116.0);}
    else {
        f[j] = (double)Math.pow(g[j],0.333333333333333);}
    }
ele = 116.0*(f[1]-(16.0/116.0));
dados[i].setL(ele);
```

Protótipo - Implementação

- calcrgb

```
double[] mat1 = {3.240479, -1.53715, -0.498535};  
double[] mat2 = {-0.969256, 1.875992, 0.041556};  
double[] mat3 = {0.055648, -0.204043, 1.057311};  
X = dados[i].obterX();  
  
.....  
matxyz[0] = X;  
for (int j=0; j<3; j++) R = R + (mat1[j]*matxyz[j]);  
dados[i].setR(R);
```