

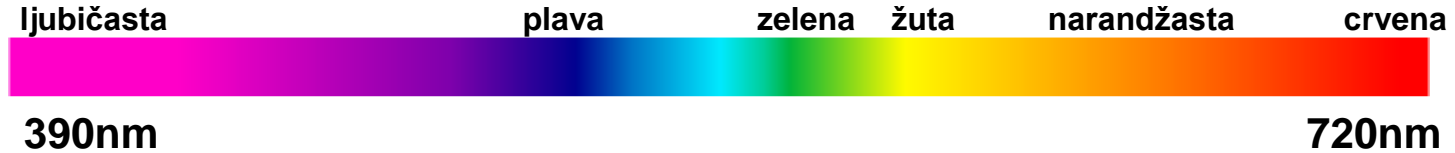
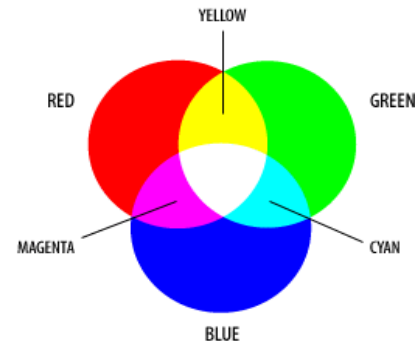
Računarska grafika
(2OER7O02)

OpenGL – Osvetljenje

Auditivne vežbe



Boje

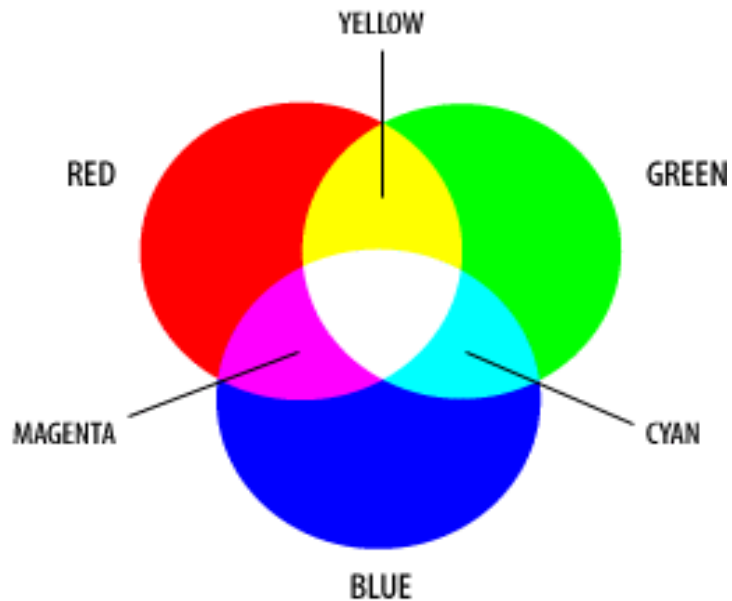


Boja je senzacija koju svetlosna energija izaziva na retini i koji se interpretira u mozgu.

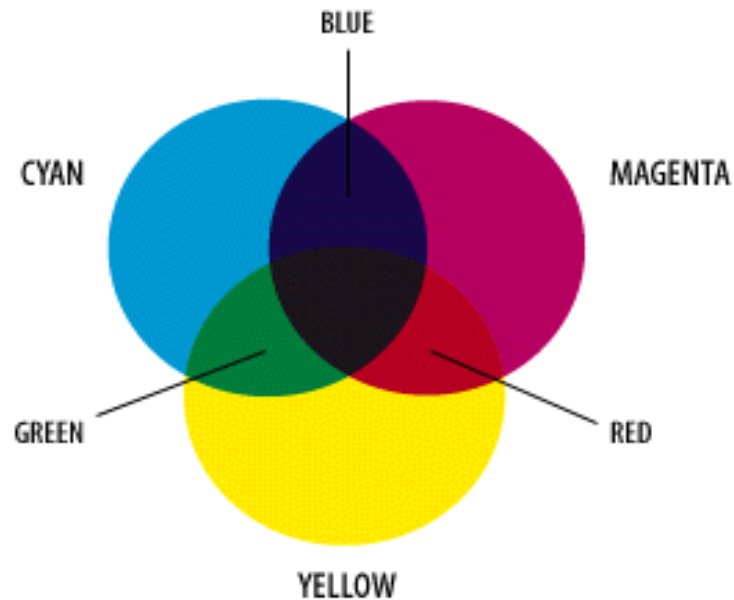
Mozak formira sliku na osnovu mešavine fotona različitih frekvencija.

Ljudi poseduju 3 tipa konusnih ćelija, koje reaguju na kratke, srednje i duge talasne dužine (crvena, zelena i plava).

Mešanje boja



Svetlosni (aditivni) model



Pigmentni (subtraktivni) model

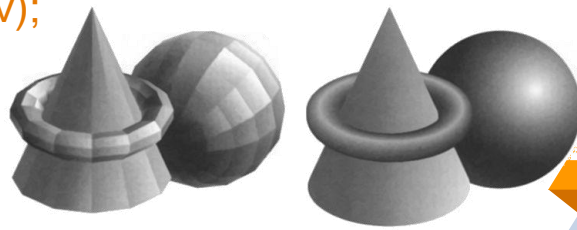
Dodavanje osvetljenja u scenu

Da bi se formirala scena sa osvetljenjem, potrebno je uraditi sledeće:

- definisati normale u svakom od temena objekata,
- definisati svojstva materijala objekata u sceni,
- definisati model osvetljenja i
- kreirati, aktivirati i postaviti jedan ili više izvora svetlosti.

Definisanje normala

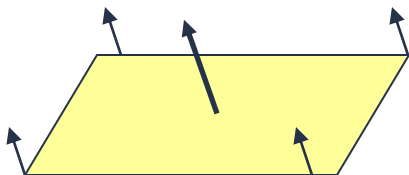
- Da bi se moglo izračunati koliko je osvetljena neka površina objekta, potrebno je znati kako je ona orijentisana u odnosu na izvor svetlosti.
- Orijentacija površine određuje se vektorom normale u svakoj tački te površine.
- Normale definišemo u temenima modela i to pre poziva **glVertex*()** funkcije.
- Normala je trodimenzionalni vektor, i postavlja se jednom od **glNormal3*()** funkcija.
 - ▷ `void glNormal3{bsidf}(TYPE nx, TYPE ny, TYPE nz);`
 - ▷ `void glNormal3{bsidf}v(const TYPE *v);`



Definisanje normala

Ako se koriste transformacije koje narušavaju normalizaciju normale (Scale, Shear, ...), treba omogućiti automatsku normalizaciju pozivom:

glEnable(GL_NORMALIZE)



```
glBegin (GL_POLYGON);  
    glNormal3fv(n0);  
    glVertex3fv(v0);  
    glNormal3fv(n1);  
    glVertex3fv(v1);  
    glNormal3fv(n2);  
    glVertex3fv(v2);  
    glNormal3fv(n3);  
    glVertex3fv(v3);  
glEnd();
```

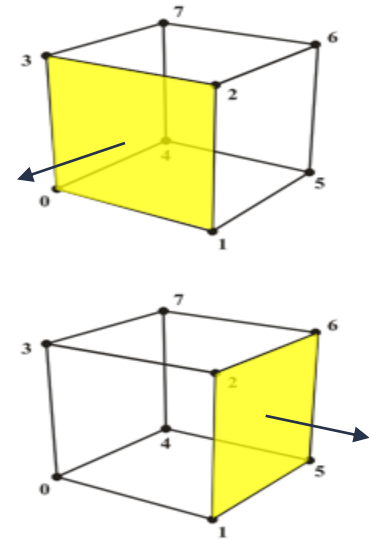
Vektor normala

- Definicija: void **glNormalPointer**(GLenum type, GLsizei stride, const GLvoid *pointer); - nema size jer se podrazumeva 3.
- Aktiviranje:
glEnableClientState(**GL_NORMAL_ARRAY**);
- Deaktiviranje:
glDisableClientState(**GL_NORMAL_ARRAY**);

Primer zadavanje polja

```
// Zajednicko polje
vert[0] = -a/2; vert[1] = -a/2; vert[2] =  a/2; // vert0
vert[3] =  0.0; vert[4] =  0.0; vert[5] =  1.0; // norm0
vert[6] =  a/2; vert[7] = -a/2; vert[8] =  a/2; // vert1
vert[9] =  0.0; vert[10]=  0.0; vert[11]=  0.0; // norm0
vert[12]=  a/2; vert[13]=  a/2; vert[14]=  a/2; // vert2
vert[15]=  0.0; vert[16]=  0.0; vert[17]=  1.0; // norm0
vert[18]= -a/2; vert[19]=  a/2; vert[20]=  a/2; // vert3
vert[21]=  0.0; vert[22]=  0.0; vert[23]=  1.0; // norm0

vert[24]=  a/2; vert[25]= -a/2; vert[26]=  a/2; // vert1
vert[27]=  1.0; vert[28]=  0.0; vert[29]=  0.0; // norm1
vert[30]=  a/2; vert[31]= -a/2; vert[32]= -a/2; // vert5
vert[33]=  1.0; vert[34]=  0.0; vert[35]=  0.0; // norm1
vert[36]=  a/2; vert[37]=  a/2; vert[38]= -a/2; // vert6
vert[39]=  1.0; vert[40]=  0.0; vert[41]=  0.0; // norm1
vert[42]=  a/2; vert[43]=  a/2; vert[44]=  a/2; // vert2
vert[39]=  1.0; vert[40]=  0.0; vert[41]=  0.0; // norm1
//...
```



Is crtavanje preko polja temena

void **glDrawArrays**(GLenum mode, GLint first, GLsizei count);

Parametar *mode* definiše koji tip primitiva se formira (GL_POINTS, GL_LINES, GL_TRIANGLES, itd.), a primitive se formiraju od elemenata svih aktivnih polja, počev od indeksa *first*, a zaključno sa indeksom *first+count-1*. Svako teme se multiplicira, tako da se sekvencijalnim obilaskom polja iscrtava čitav objekat. U slučaju kocke to znači da će se svako teme ponoviti tačno tri puta (obzirom da učestvuje u formiranju tri stranice).

```
void CGLRenderer::DrawVACube2 ()
{
    glVertexPointer(3, GL_FLOAT, 6*sizeof(float), &vert[0]);
    glNormalPointer(GL_FLOAT, 6*sizeof(float), &vert[3]);

    glEnableClientState(GL_VERTEX_ARRAY);
    glEnableClientState(GL_NORMAL_ARRAY);

    glDrawArrays(GL_QUADS, 0, 24);

    glDisableClientState(GL_VERTEX_ARRAY);
    glDisableClientState(GL_NORMAL_ARRAY);
}
```

Normala parametarskih površi

Ako je površ zadata analitički:

gde su X , Y i Z diferencijabilne funkcije po s i t .

$$V(s, t) = [X(s, t) \ Y(s, t) \ Z(s, t)]$$

Treba naći $\frac{\partial V}{\partial s}$ i $\frac{\partial V}{\partial t}$ a zatim $\frac{\partial V}{\partial s} \times \frac{\partial V}{\partial t}$

Pacijalni izvodi po s i t daju vektore tangente na površ, po s i t pravcu, vektorski proizvod je upravan na oba, pa time i na površinu.

Vektorski proizvod dva vektora $V(v_x, v_y, v_z)$ i $W(w_x, w_y, w_z)$, računa se po formuli:

$$V \times W = [v_x \ v_y \ v_z] \times [w_x \ w_y \ w_z] = [(v_y w_z - v_z w_y) \ (v_z w_x - v_x w_z) \ (v_x w_y - v_y w_x)].$$

Normalizacija (svođenje na jedinični vektor): deljenje svake komponente sa dužinom

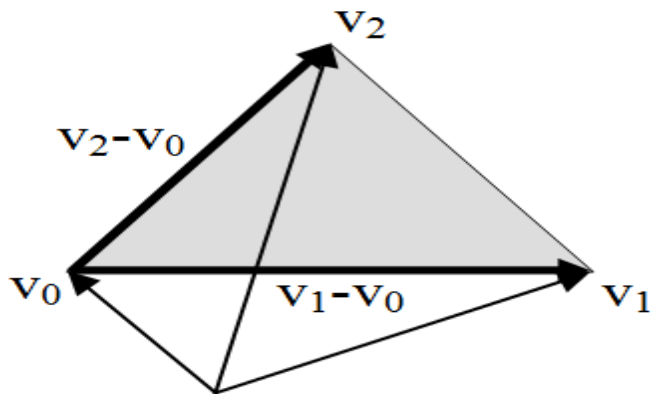
Primer: Ako je $V(s,t) = [s^2 \ t^3 \ 3-st]$

$$\frac{\partial V}{\partial s} = [2s \ 0 \ -t], \quad \frac{\partial V}{\partial t} = [0 \ 3t^2 \ -s], \quad \text{and} \quad \frac{\partial V}{\partial s} \times \frac{\partial V}{\partial t} = [-3t^3 \ 2s^2 \ 6st^2] \quad len = \sqrt{x^2 + y^2 + z^2}$$

Za $s = 1$ i $t = 2$, tačka na površi ima koordinate $(1,8,1)$, a vektor normale je $(-24,2,24)$

Posle normalizacije: $(-24/34, 2/34, 24/34) = (-0.70588, 0.058823, 0.70588)$

Normale poligonalnih površi



$$\mathbf{N} = [\mathbf{V}_1 - \mathbf{V}_0] \times [\mathbf{V}_2 - \mathbf{V}_0]$$

```
void normcrossprod(float v1[3], float v2[3], float out[3])
{
    GLint i, j;
    GLfloat length;
    out[0] = v1[1]*v2[2] - v1[2]*v2[1];
    out[1] = v1[2]*v2[0] - v1[0]*v2[2];
    out[2] = v1[0]*v2[1] - v1[1]*v2[0];
    normalize(out);
}

void normalize(float v[3]) {
    GLfloat d = sqrt(v[1]*v[1]+v[2]*v[2]+v[3]*v[3]);
    if (d == 0.0) {
        error("zero length vector");
        return;
    }
    v[1] /= d; v[2] /= d; v[3] /= d;
}
```

$$[v_x \ v_y \ v_z] \times [w_x \ w_y \ w_z] = [(v_y w_z - v_z w_y) \ (v_z w_x - v_x w_z) \ (v_x w_y - v_y w_x)]$$

















Definisanje materijala

Materijal određuje:

- koji deo vidljivog spektra objekat najviše reflektuje (pa time i boju objekta),
- da li je objekat sjajan ili mat,
- da li je i koliko objekat providan i
- da li emituje svetlost.



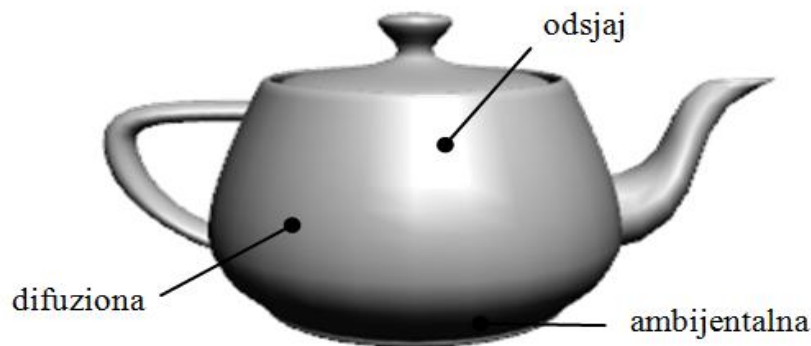
Klasa CGLMaterial

CGLMaterial	
	 m_vAmbient[4] : float
	 m_vDiffuse[4] : float
	 m_vSpecular[4] : float
	 m_vEmission[4] : float
	 m_fShininess : float
 Select()	
 SetAmbient()	
 SetDiffuse()	
 SetSpecular()	
 SetEmmision()	
 SetShininess()	

Komponente materijala

Materijal ima pet komponenti, i to:

- ambijentalnu boju (*ambient*),
- difuzionu boju (*diffuse*),
- boju odsjaja (*specular*),
- emisionu boju (*emission*) i
- sjaj (*shininess*).



Definisanje karakteristika materijala

```
void glMaterial{if}[v](GLenum face, GLenum pname, TYPE param);
```

Parameter *face* može biti: **GL_FRONT**, **GL_BACK**, ili **GL_FRONT_AND_BACK** i određuje stranu (lice) objekta na koju se materijal primenjuje.

Pname je svojstvo materijala koje se menja, i može biti: **GL_AMBIENT**, **GL_DIFFUSE**, **GL_SPECULAR**, **GL_EMISSION**, **GL_SHININESS** ili **GL_AMBIENT_AND_DIFFUSE** ,

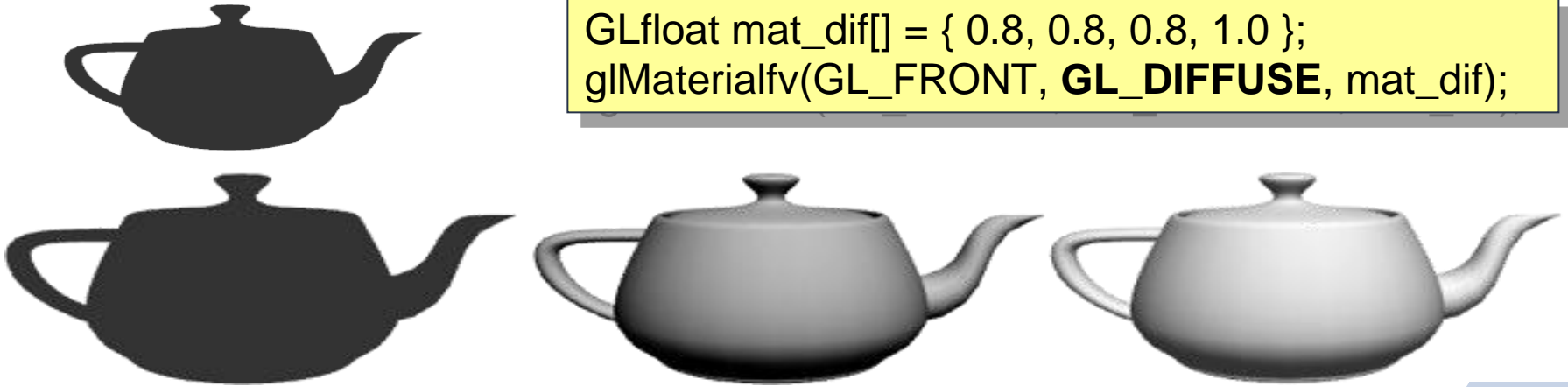
a *param* vrednost na koju se postavlja.

Ambijentalna i difuziona komponenta

```
GLfloat mat_amb[] = { 0.5, 0.5, 0.5, 1.0 };  
glMaterialfv(GL_FRONT, GL_AMBIENT, mat_amb);
```



```
GLfloat mat_dif[] = { 0.8, 0.8, 0.8, 1.0 };  
glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_dif);
```



Odsjaj

```
GLfloat mat_specular[] = { 1.0, 1.0, 1.0, 1.0 };  
GLfloat mat_shininess = 64.0;  
glMaterialfv(GL_FRONT, GL_SPECULAR, mat_specular);  
glMaterialf(GL_FRONT, GL_SHININESS, mat_shininess);
```



Efekat samo odsjaja



Kombinovani efekat odsjaja, ambijentalne
i difuzije komponente materijala



$s = 8$



$s = 64$



$s = 128$

Emisiona komponenta

```
GLfloat mat_emission[] = {0.5, 0.5, 0.5, 1.0};  
glMaterialfv(GL_FRONT, GL_EMISSION, mat_emission);
```



Podrazumevane vrednosti materijala

Parametar	Vrednost	Značenje
GL_AMBIENT	(0.2, 0.2, 0.2, 1.0)	Ambijentalna boja materijala
GL_DIFFUSE	(0.8, 0.8, 0.8, 1.0)	Difuziona boja materijala
GL_AMBIENT_AND_DIFFUSE	-	Ambijentalna i difuziona boja materijala
GL_SPECULAR	(0.0, 0.0, 0.0, 1.0)	Odsjaj materijala
GL_SHININESS	0.0	Ekspozet odsjaja
GL_EMISSION	(0.0, 0.0, 0.0, 1.0)	Emisiona boja materijala

Na providnost materijala (*alpha* vrednost) utiče samo *alpha* kanal difuzione komponente materijala.

Definisanje modela osvetljenja

- Pod definisanjem modela osvetljenja podrazumeva se
 - ▷ određivanje jačine globalnog ambijentalnog osvetljenja
 - ▷ izbor lokalnog ili udaljenog posmatrača,
 - ▷ izbor jednostranog ili dvostranog osvetljenja i
 - ▷ izbor da li se efekat refleksije primenjuje nezavisno od ambijentalne i difuzione boje i nakon primene teksture.
- Sve što je vezano za definisanje modela osvetljenja postavlja se funkcijom:
 - ▷ void **glLightModel**{fi}[v](GLenum pname, TYPE param)

Definisanje modela osvetljenja

- **Globalno ambijentalno osvetljenje** predstavlja osvetljenje koje ne potiče od postavljenih izvora svetlosti.

```
GLfloat lmodel_ambient[] = { 0.2, 0.2, 0.2, 1.0 };  
glLightModelfv(GL_LIGHT_MODEL_AMBIENT, lmodel_ambient);
```

- **Izbor lokalnog ili udaljenog posmatrača** utiče na način izračunavanja osvetljenosti objekata. Lokalni posmatrač zahteva mnogo složenije izračunavanje, jer je potrebno za svako teme svakog od objekata izračunati ugao pod kojim ga vidi posmatrač.

```
glLightModeli(GL_LIGHT_MODEL_LOCAL_VIEWER, GL_TRUE);
```

Definisanje modela osvetljenja

- Izbor jednostranog ili **dvostranog osvetljenja** vrši se parametrom `GL_LIGHT_MODEL_TWO_SIDE`. Postavljanjem ovog parametra na `GL_FALSE`, samo strana poligona okrenuta izvoru svetlosti biće osvetljena.

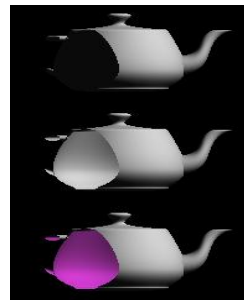
```
glLightModeli(GL_LIGHT_MODEL_TWO_SIDE, GL_FALSE);
```

- Podrazumevano stanje je – isključeno osvetljenje. Uključivanje osvetljenja ostvaruje se pozivom sledeće funkcije

```
glEnable(GL_LIGHTING);
```

- a isključivanje

```
glDisable(GL_LIGHTING);
```



Definisanje izvora svetlosti

- OpenGL omogućava postavljanje do 8 izvora svetlosti. Inicijalno, svi izvori su isključeni. Uključivanje i -tog izvora ostvaruje se pozivom funkcije
 - ▷ glEnable(**GL_LIGHT*i***);
- Ako želimo izbeći delovanje izvora svetlosti na neki objekat, pozivamo funkciju **glDisable()** sa identifikatorom datog izvora.
 - ▷ glDisable(**GL_LIGHT*i***);

Postavljanje boje svetlosnog izvora

Osnovne karakteristike svakog izvora svetlosti su boja i intenzitet svetlosti koju emituje. Obzirom da materijal ima tri komponente na koje deluje svetlost (na emisiju komponentu ne utiče svetlost), i svetlosni izvori definišu tri komponente:

- **ambijentalnu**,
- **difuzionu** i
- **refleksionu**.

```
float light_ambient[] = { 0.1, 0.1, 0.1, 1.0 };  
float light_diffuse[] = { 1.0, 1.0, 1.0, 1.0 };  
float light_specular[] = { 1.0, 1.0, 1.0, 1.0 };  
glLightfv(GL_LIGHT0, GL_AMBIENT, light_ambient);  
glLightfv(GL_LIGHT0, GL_DIFFUSE, light_diffuse);  
glLightfv(GL_LIGHT0, GL_SPECULAR, light_specular);
```

Sve komponente izvora svetlosti postavljaju se funkcijom:

```
void glLight{if}[v](GLenum light, GLenum pname, TYPE param)
```

gde argument *light* definiše izvor svetlosti čiji se parametar postavlja (može imati vrednost **GL_LIGHT0** do **GL_LIGHT7**), *pname* naziv parametra, a *param* vrednost na koju se postavlja.

Za postavljanje boje i intenziteta svetlosnih izvora koriste se parametri: **GL_AMBIENT**, **GL_DIFFUSE** i **GL_SPECULAR**.

Pozicija i slabljenje

Izvori mogu biti:

- direkcion GLfloat light_position[] = { 1.0, 1.0, 1.0, 0.0 };

Svi zraci koji dopiru do objekata smatraju se paralelnim (npr. sunce je takav izvor)

- pozicioni GLfloat light_position[] = { 1.0, 1.0, 1.0, 1.0 };

Ugao zraka zavisi od položaja svetla u sceni (npr. lampa). Zahteva mnogo više izračunavanja, pa ukoliko nije neophodno treba koristiti direkciono svetlo.

Kod realnih izvora, intenzitet svetla opada sa rastojanjem. Ovo ima smisla samo kod pozicionih svetala. Faktor slabljenja za poziciona svetla računa se po formuli:

$$faktor_atenuacije = \frac{1}{k_c + k_l d + k_q d^2}$$

d – rastojanje između izvora svetla i temena objekta

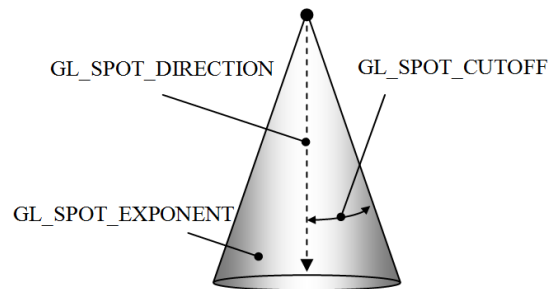
k_c = GL_CONSTANT_ATTENUATION

k_l = GL_LINEAR_ATTENUATION

k_q = GL_QUADRATIC_ATTENUATION

Usmereni izvor svetlosti

Prostor koji osvetljava poziciono svetlo može se omeđiti kupom definisanjem **GL_SPOT_CUTOFF** parametra koji definiše ugao između pravca svetla i zraka koji putuje ivicom kupe. Ugao kupe je 2 puta veći. Po defaultu, ovo je isključeno, obzirom da je vrednost postavljena na 180 (kupa je sfera!!!). **GL_SPOT_CUTOFF** parametra mora biti u opsegu [0.0, 90.0], da bi kupa postojala.



```
GLfloat spot_direction[] = { -1.0, -1.0, 0.0 };  
glLightfv(GL_LIGHT0, GL_SPOT_DIRECTION, spot_direction);  
glLightf(GL_LIGHT0, GL_SPOT_CUTOFF, 45.0);
```

Distribucija intenziteta svetlosti u okviru kupe definiše (osim atenuacije i) **GL_SPOT_EXPONENT**. Intenzitet opada ka obodu kupe sa kosinusom rastojanja od ose kupe. Što je **GL_SPOT_EXPONENT** veći, to je svetlo usmerenije i brže opada ka rubu kupe. Ako je 0 (po default-u), nema slabljenja.

Primer usmerenog izvora

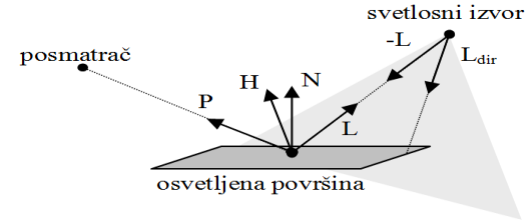
```
GLfloat light1_ambient[] = { 0.2, 0.2, 0.2, 1.0 };
GLfloat light1_diffuse[] = { 1.0, 1.0, 1.0, 1.0 };
GLfloat light1_specular[] = { 1.0, 1.0, 1.0, 1.0 };
GLfloat light1_position[] = { -2.0, 2.0, 1.0, 1.0 };
GLfloat spot_direction[] = { -1.0, -1.0, 0.0 };

glLightfv(GL_LIGHT1, GL_AMBIENT, light1_ambient);
glLightfv(GL_LIGHT1, GL_DIFFUSE, light1_diffuse);
glLightfv(GL_LIGHT1, GL_SPECULAR, light1_specular);
glLightfv(GL_LIGHT1, GL_POSITION, light1_position);
glLightf(GL_LIGHT1, GL_CONSTANT_ATTENUATION, 1.5);
glLightf(GL_LIGHT1, GL_LINEAR_ATTENUATION, 0.5);
glLightf(GL_LIGHT1, GL_QUADRATIC_ATTENUATION, 0.2);

glLightf(GL_LIGHT1, GL_SPOT_CUTOFF, 45.0);
glLightfv(GL_LIGHT1, GL_SPOT_DIRECTION, spot_direction);
glLightf(GL_LIGHT1, GL_SPOT_EXPONENT, 2.0);

glEnable(GL_LIGHT1);
```

Matematika osvetljenja



$$\begin{aligned}
 boja_temena &= emission_{material} + \\
 &ambient_{light_model} * ambient_{material} + \\
 &\sum_{i=0}^{n-1} \left(\frac{1}{k_c + k_l d + k_q d^2} \right)_i * (spotlight_effect)_i * \\
 &[ambient_{light} * ambient_{material} + \\
 &(\max\{L \cdot n, 0\}) * diffuse_{light} * diffuse_{material} + \\
 &(\max\{s \cdot n, 0\})^{shininess} * specular_{light} * specular_{material}]_i
 \end{aligned}$$

L - Jedinični vektor usmeren od temena ka izvoru svetlosti

n – Vektor normale

s – Normalizovani zbir dva jedinična vektora (1. teme-izvor svetlosti, 2. teme-posmatrač) – “half” vektor