Základy počítačové grafiky
Přednáška 7

Martin Němec

VŠB-TU Ostrava

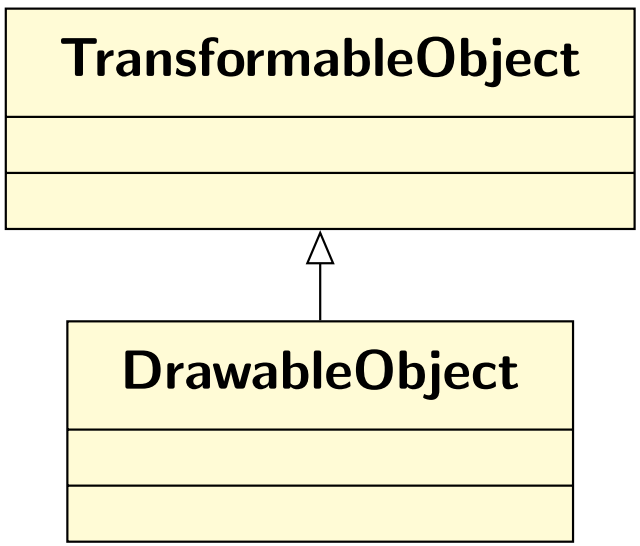
2024

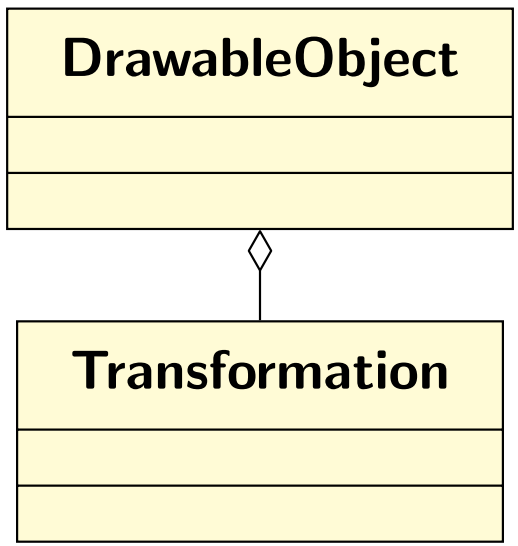
Použití transformaci v projektu

Použit transformace v projektu lze více způsoby. Zamyslete se nad:

- DrawableObject **je transformovatelný**
- DrawableObject **má transformace**

Obě varianty jsou použitelné (reprezentují jiný přístup) a mají své výhody a nevýhody.

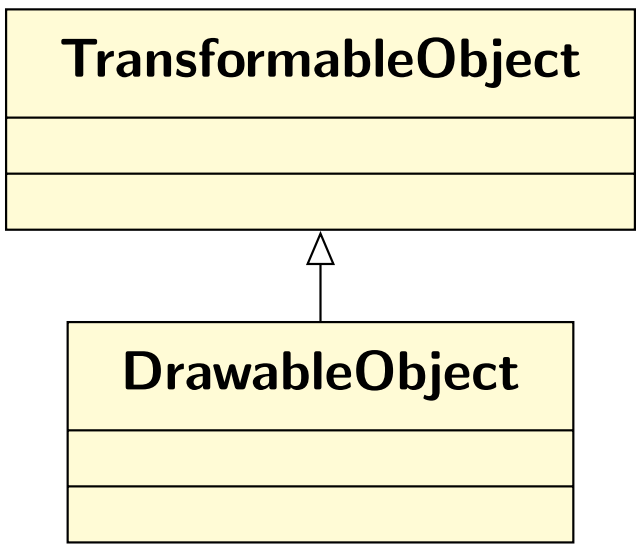




Základní rozdíly

DrawableObject **je transformovatelný**

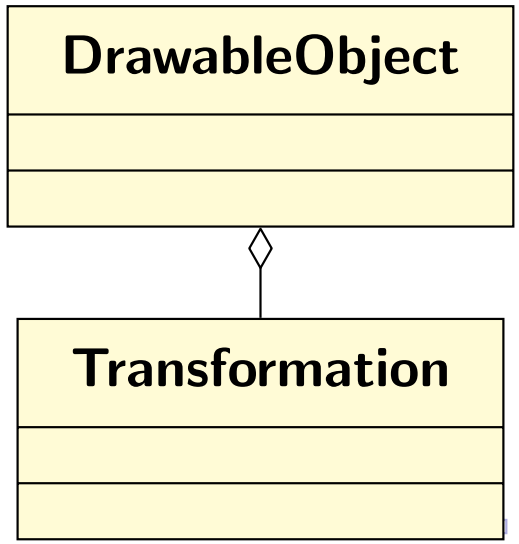
- Transformace jsou vlastnost objektu.
- Transformace umožní zdědit metody, které se provádí na objektu, ale objekt nemá transformace samostatně uložené.
- Pokud se nemusí transformace ukládat, tak se jedná o snazší implementaci.
- Při potřebě transformace ukládat, analyzovat, kombinovat apod. se jedná o méně flexibilní řešení.



Základní rozdíly

DrawableObject **má transformace**

- Transformace jako trvalé součásti objektu, které mohou být sledovány, modifikovány a kombinovány (např. složené transformace).
- Objekt má atribut Transformation, který uchovává všechny potřebné informace o transformacích a jejich kombinacích.
- Jasně oddělená zodpovědnost pro transformace.
- Vhodnější u složitějších aplikacích, jako jsou grafické nebo herní enginy,



glViewport()

Převádí objekty z NDC (Normal Device Coordinates) do souřadnic okna (Window Coordinates). Definuje obdélník pro vykreslení v okně.

glViewport(GLint x, GLint y, GLsizei width, GLsizei height)

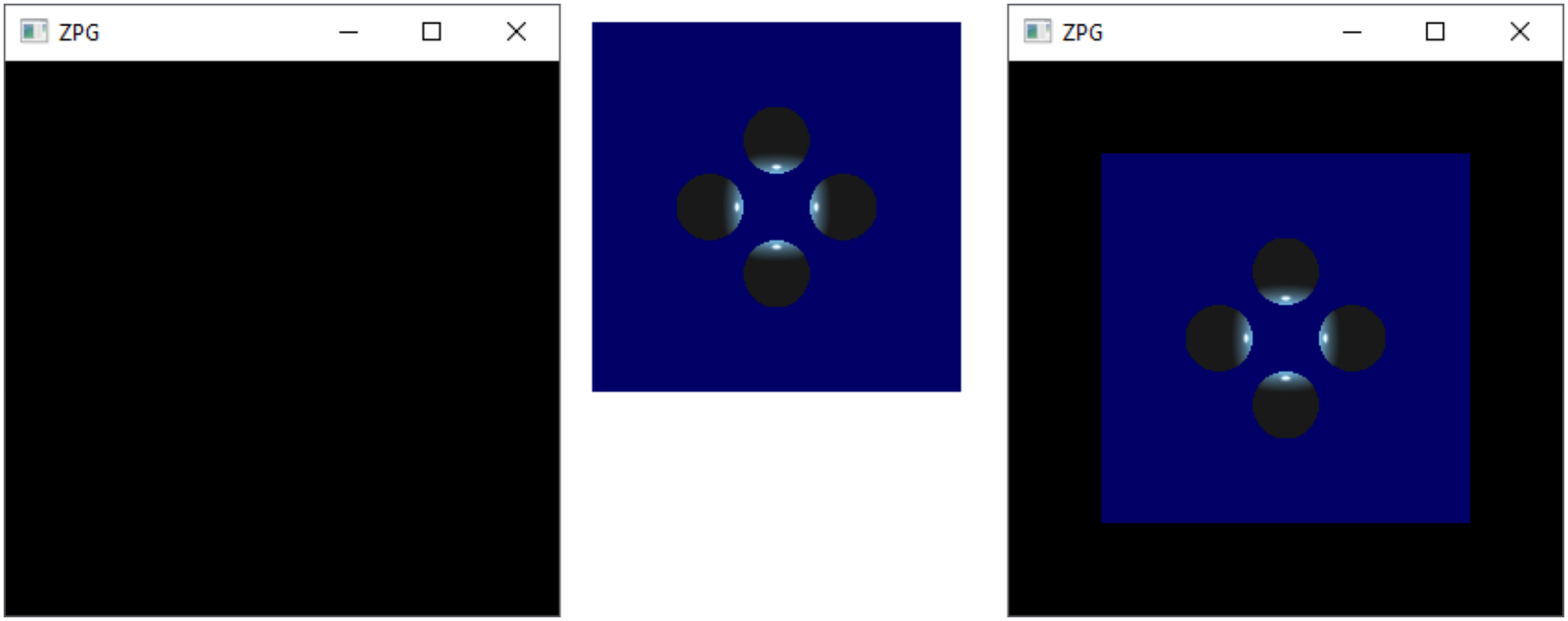
- x, y - pozice levého dolního rohu;
- *width*, *height* - šířka a výška vykreslovaného obdélníku.

$$x_w = (x_{nd} + 1)\left(\frac{width}{2}\right) + x\,,$$

$$y_w = (y_{nd} + 1)\left(\frac{height}{2}\right) + y\,.$$

Jaké souřadnice bude mít bod $A = [-0.5, 1]$, pokud ho převedeme z NDC do okna s následujícím nastavením $(0, 0, 800, 600)$?

glViewport()



glViewport(50, 50, width-100, height-100);

Light attenuation

Pro světlo platí, že intenzita klesá s druhou mocninou vzdálenosti. V grafice ale nejsou hodnoty předem dané, upravujeme je podle potřeby a požadavků na výslednou scénu.

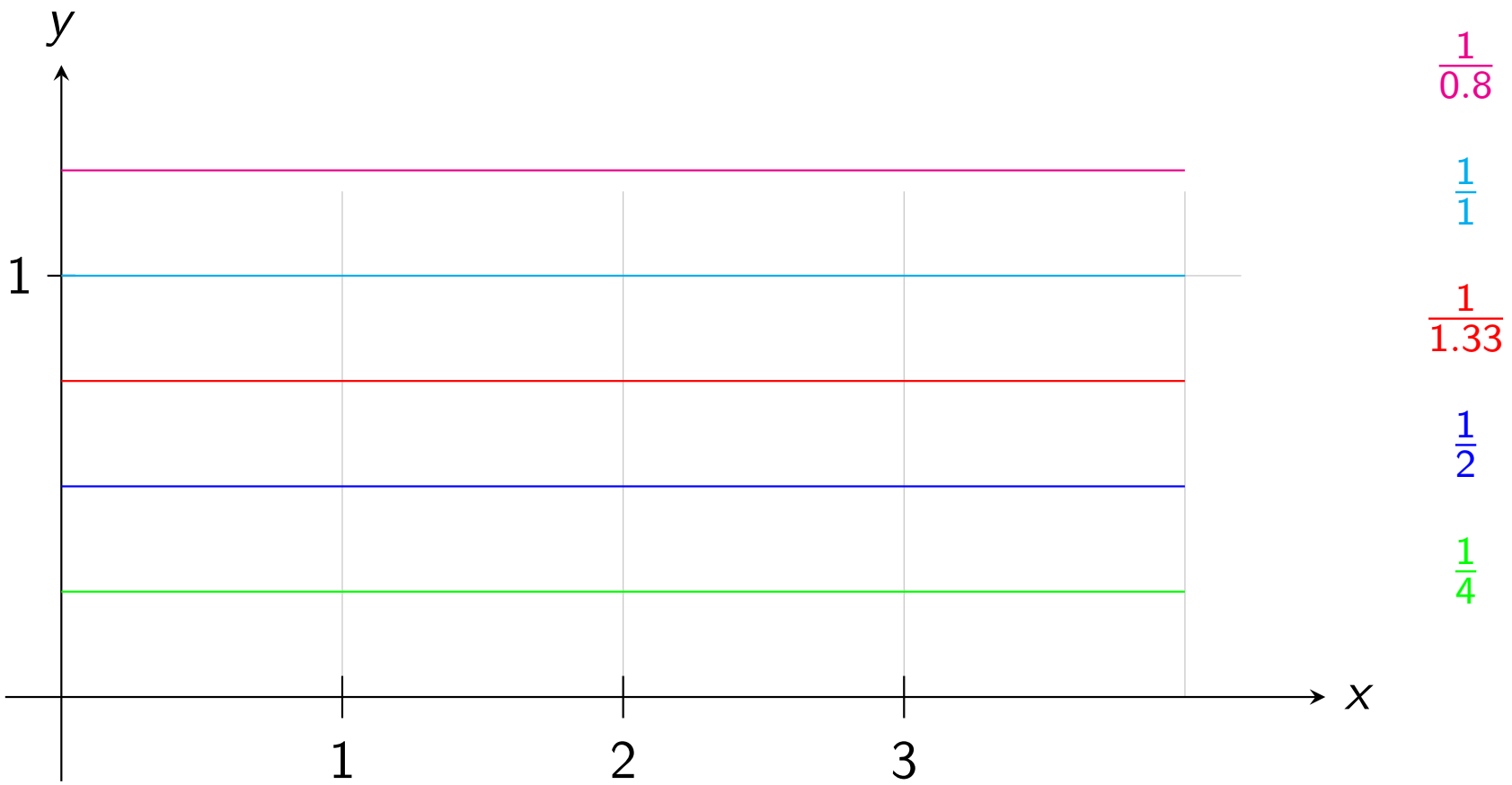
$$f_{att}(d) = \frac{1}{k_c + k_l d + k_q d^2}$$

Příklad použití.

```
float attenuation(float c,float l,float q,float dist){
  float att=1.0/(c+l*dist+q*dist*dist);
  return clamp(att, 1.0, 0.0);
}
//napr. c=1.0, l=0.1  q=1.0
```

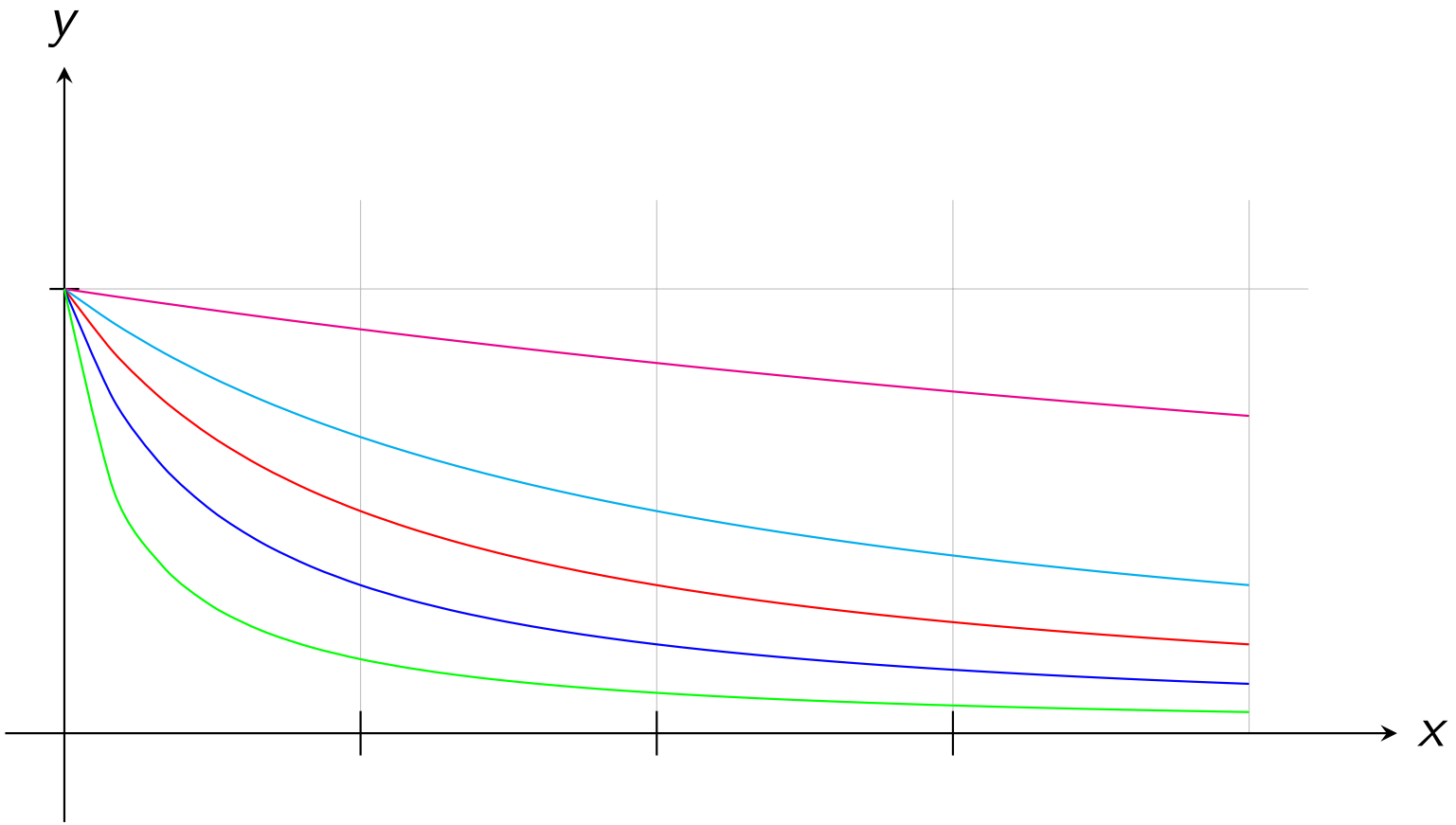
Light attenuation

$$f_{att}(d) = \frac{1}{k_c}$$



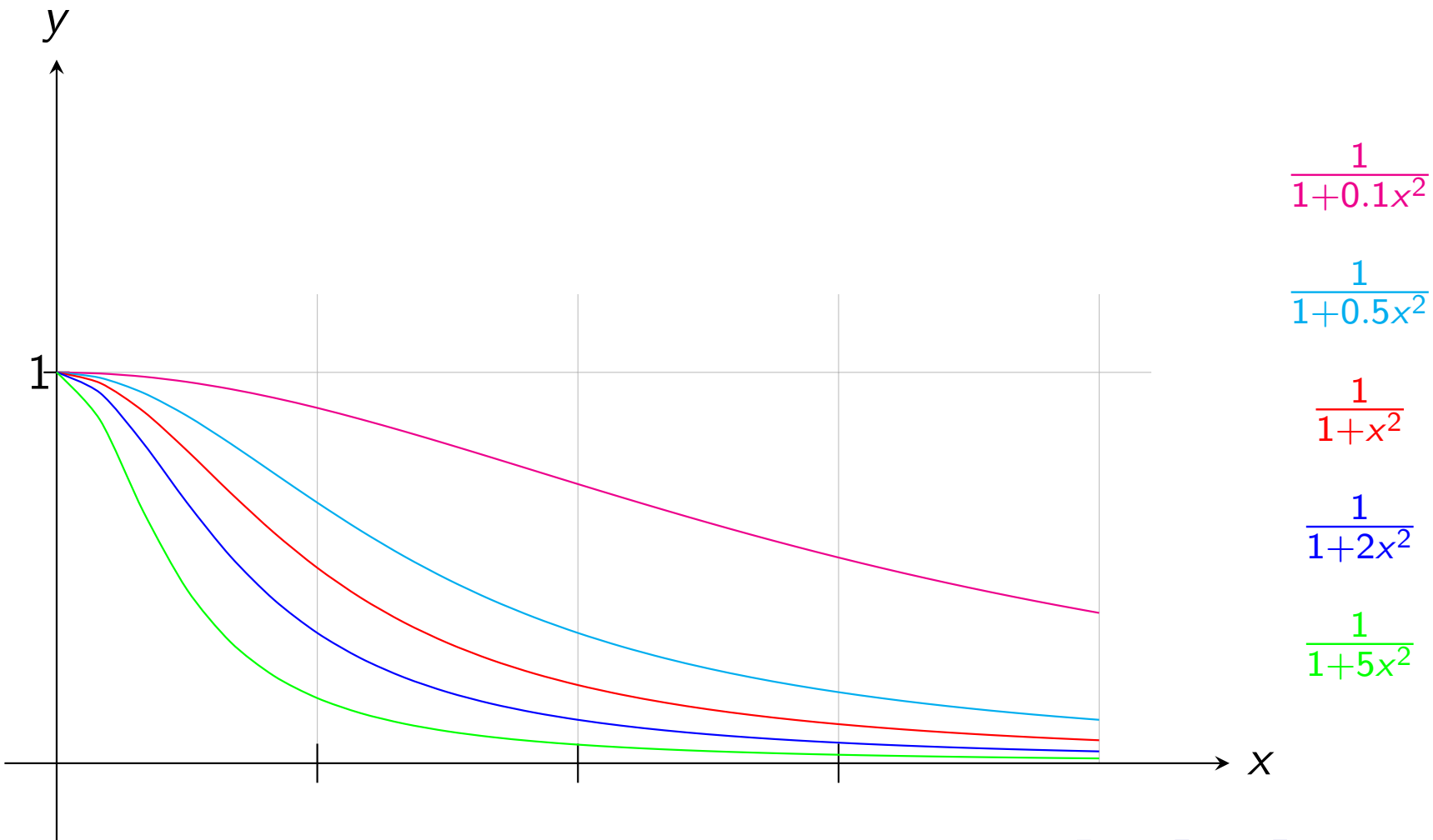
Light attenuation

$$f_{att}(d) = \frac{1}{1 + k_l d}$$



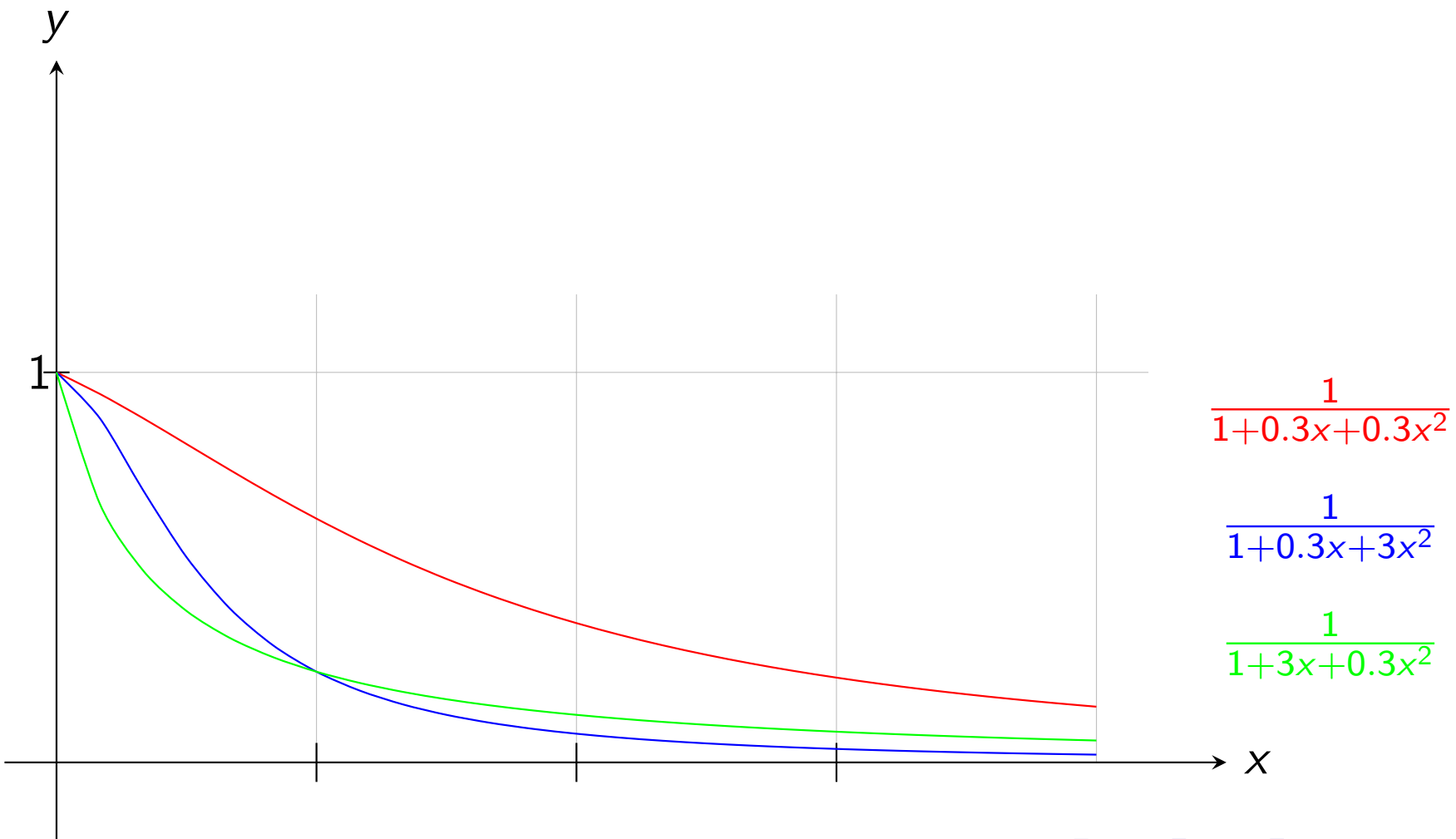
Light attenuation

$$f_{att}(d) = \frac{1}{1 + k_q d^2}$$



Light attenuation

$$f_{att}(d) = \frac{1}{1 + k_l d + k_q d^2}$$

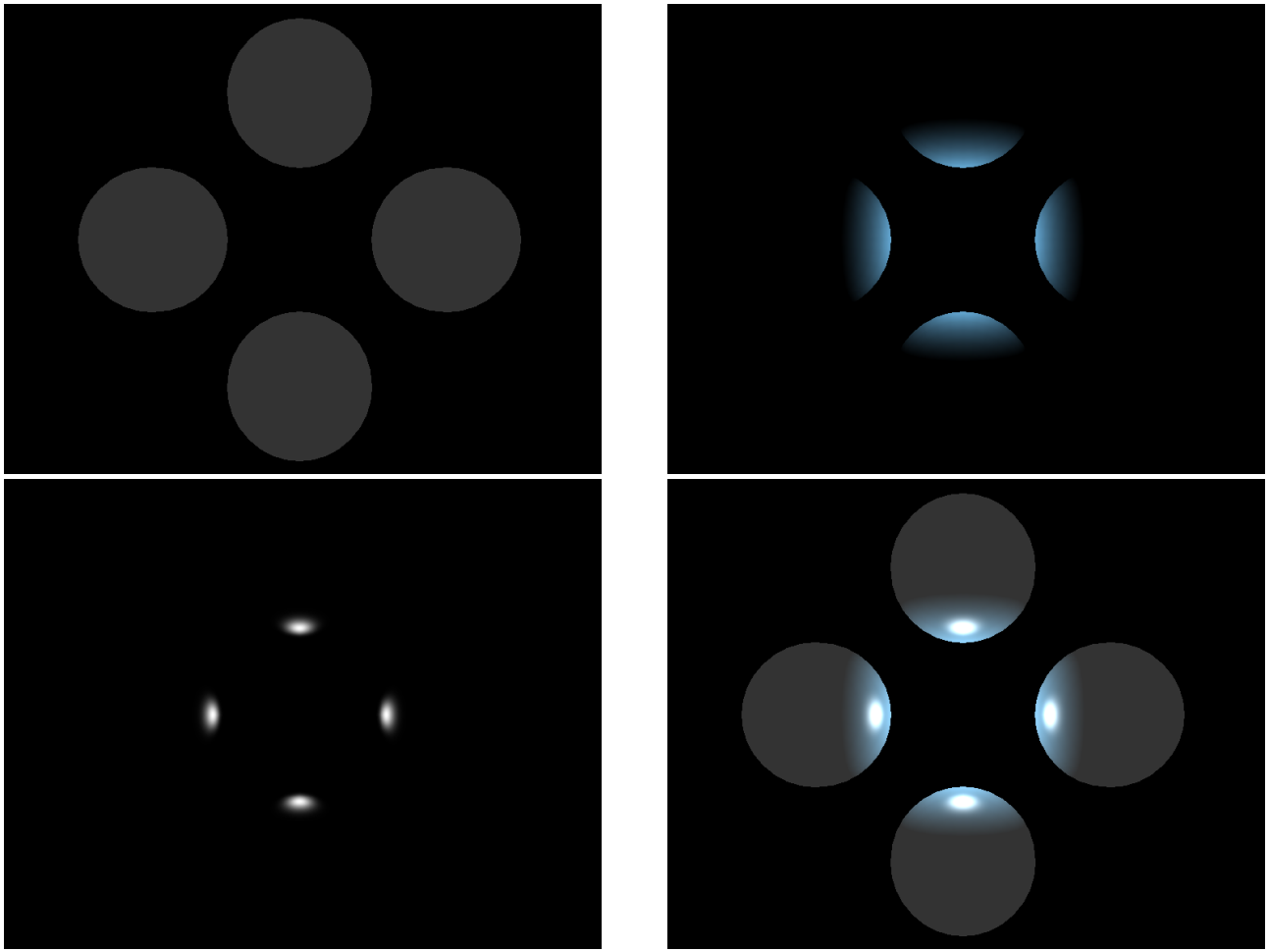


Phong reflection model

Empirický osvětlovací model, kde je výsledná intenzita dána součtem ambientní (obvykle jedna), difúzních a zrcadlových složek ze všech světel.

$$I_v = I_a r_a + \sum_{i=0}^{m} (I_{d,i} r_d \cos \alpha_i + I_{s,i} r_s \cos^h \phi_i)$$

Phongův osvětlovací model



Světla

Pro výpočet osvětlení bude důležité nastavení vlastností jednotlivých světel.

- Fixní řetězec byl omezen na maximálně 8 světel GL_LIGHT0 až GL_LIGHT7.
- Moderní OpenGL závislé na naprogramování shaderů.

Základní typy světel:

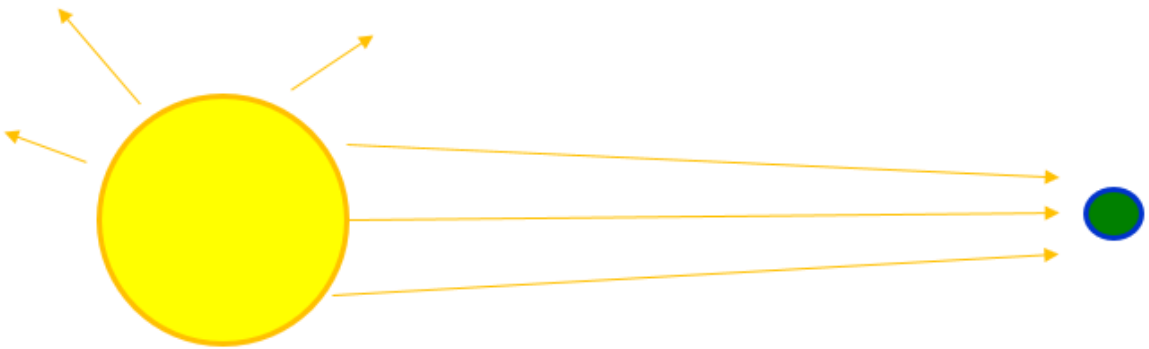
- bodové světlo (point light);
- směrové světlo (directional light);
- reflektor(spot light);
- plošné světlo (area light).

Bodové světlo

- Bodové světlo (point light) vyzařuje světlo do všech směrů (žárovka).
- Důležitá pozice světla.
- Potřeba vypočítat útlum – bodové světlo slábne vzhledem ke vzdálenosti od světla.

Směrové světlo

- Směrové světlo (directional light) vyzařuje světlo pouze daným směrem. Příkladem takového světla může být slunce, které vzhledem k zemi má paprsky "rovnoběžné".
- Světlo se zadává pouze směrem paprsků, poloha v tomto případě není důležitá (vektor směru paprsků).
- Nepočítá se útlum.



Reflektor

- Reflektor (spot light) vyzařuje světlo z jednoho bodu, avšak směr světla je omezen (nejčastěji kuželem).
- Příkladem může být lampa se stínítkem, baterka apod.
- Světlo je dáno svou pozicí, a směrem, ve kterém je obvykle intenzita největší (střed kužele).
- Slábnutí světla můžeme ovlivňovat podle úhlu svírajícího mezi směrem světla a směrem ke zkoumanému bodu.
- Potřeba vypočítat útlum.

Plošný zdroj světla

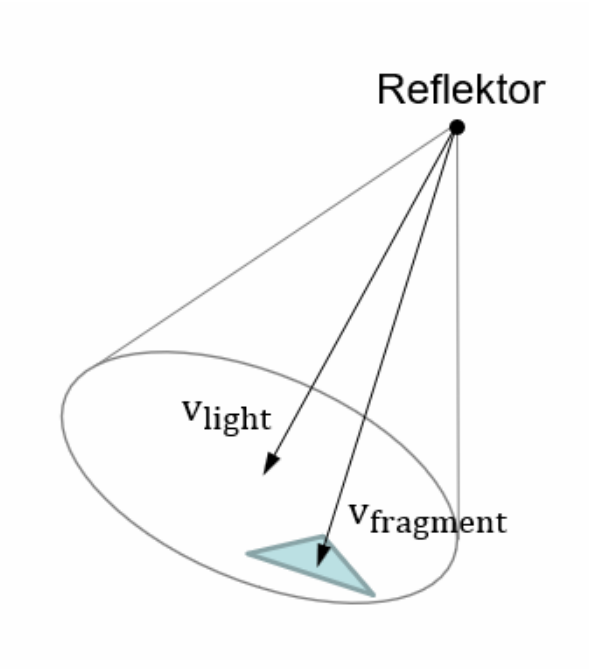
- Plošné světlo (area light) nebyl ve fixním řetězci podporován, náročné na výpočet.
- Příkladem může být třeba zářivka.
- Obvykle se používá až při výpočtu metodami jako je radiační metoda (radiosity) .
- V OpenGL se obvykle nahrazuje plošný zdroj více bodovými.
- V případě osvětlení se počítají světelné mapy (lightmapy). Ukládání do textur (bake).

Reflektor

Fixní vykreslovací řetězec

```
spot_direction[]={-1.0, -1.0, 0.0};  //smer
glLightfv(GL_LIGHT0,GL_SPOT_DIRECTION,spot_direction);
glLightfv(GL_LIGHT0,GL_SPOT_CUTEOF, 45.0f);
```

Moderní OpenGL $dot(\vec{v}_{light}, \vec{v}_{frag}) > \cos\alpha$



Multiple Lights

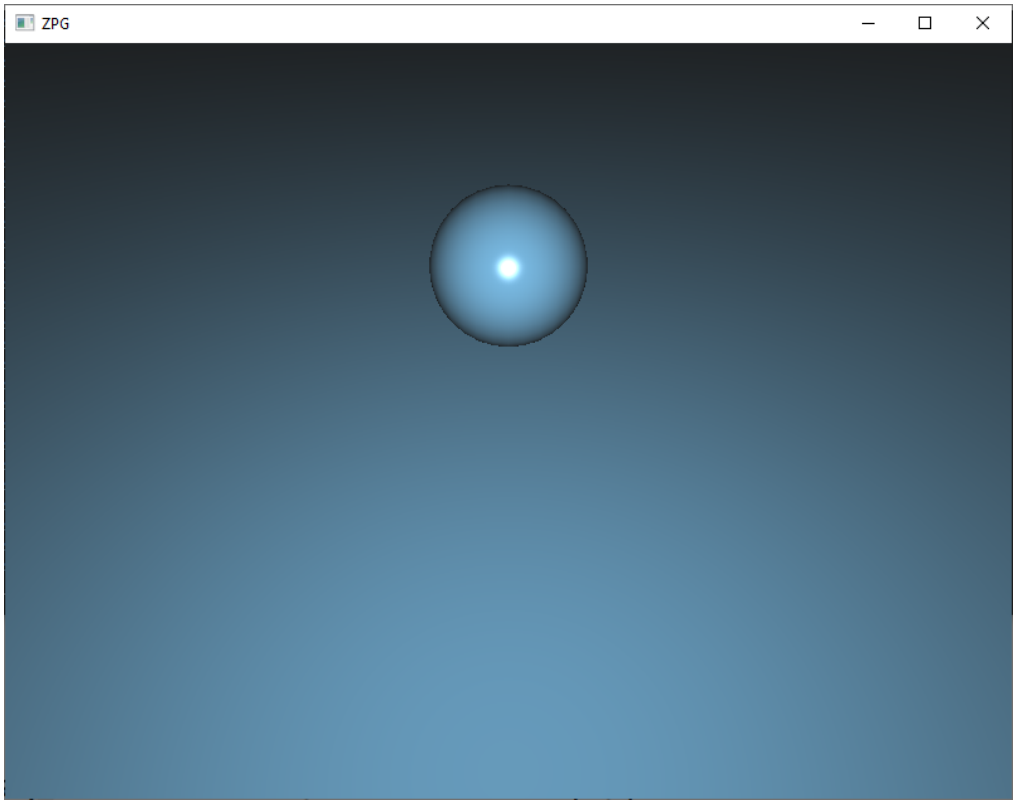
Jak poslat více světel do shaderu?.

```
struct lightSource{
  vec3 position;
  vec3 diffuse;
  vec3 specular;
};

lightSource light = lightSource(
  vec3(0.0,  1.0,  2.0),
  vec3(1.0,  1.0,  1.0),
  vec3(1.0,  1.0,  1.0),
);

uniform lightSource lights[5];

//Sending a variable
GLuint loc = glGetUniformLocation(
    shader,"light[0].position");
glUniformxx(loc, value);
```

Typy světel

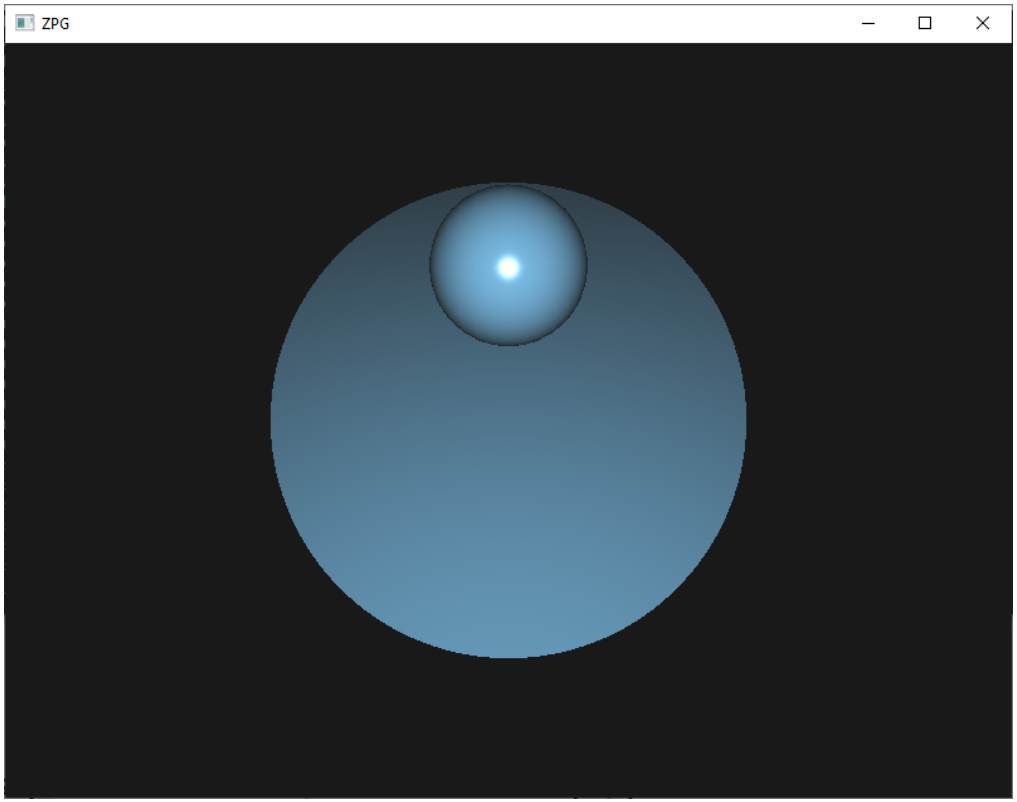
Point Light

Máme již naimplementováno.



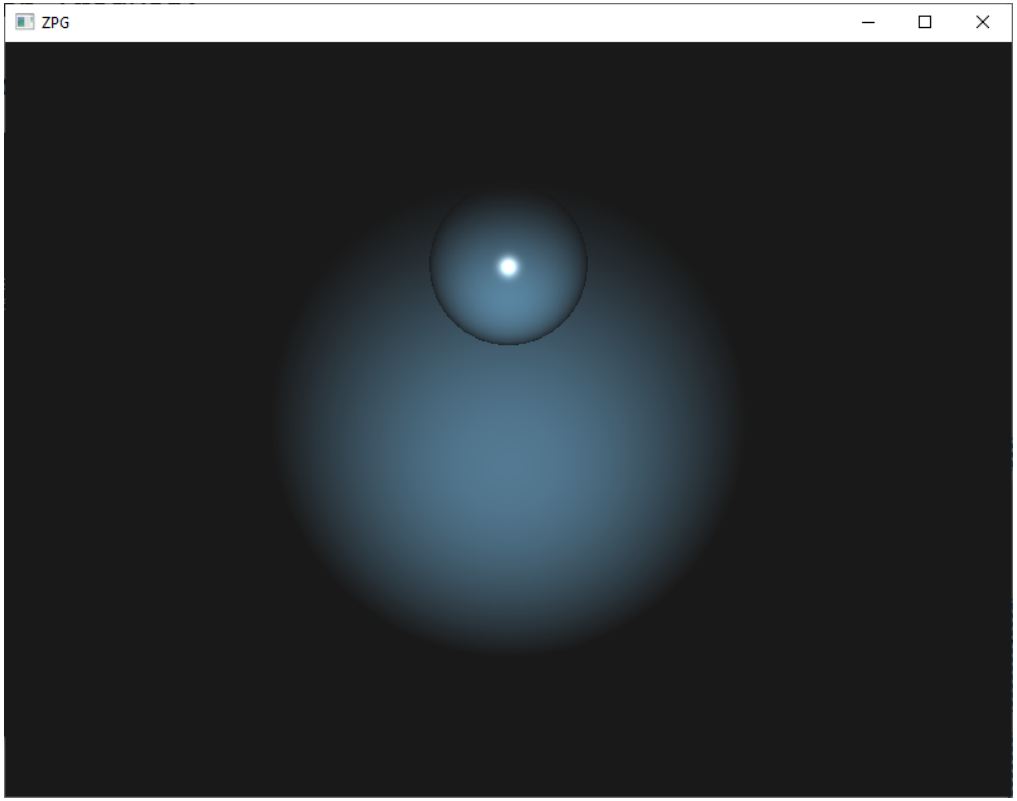

```
// for all light sources
for (int index = 0; index < numberOfLights; index++)  {
     finalColor += lightIntensity;
  }
```

Reflektor



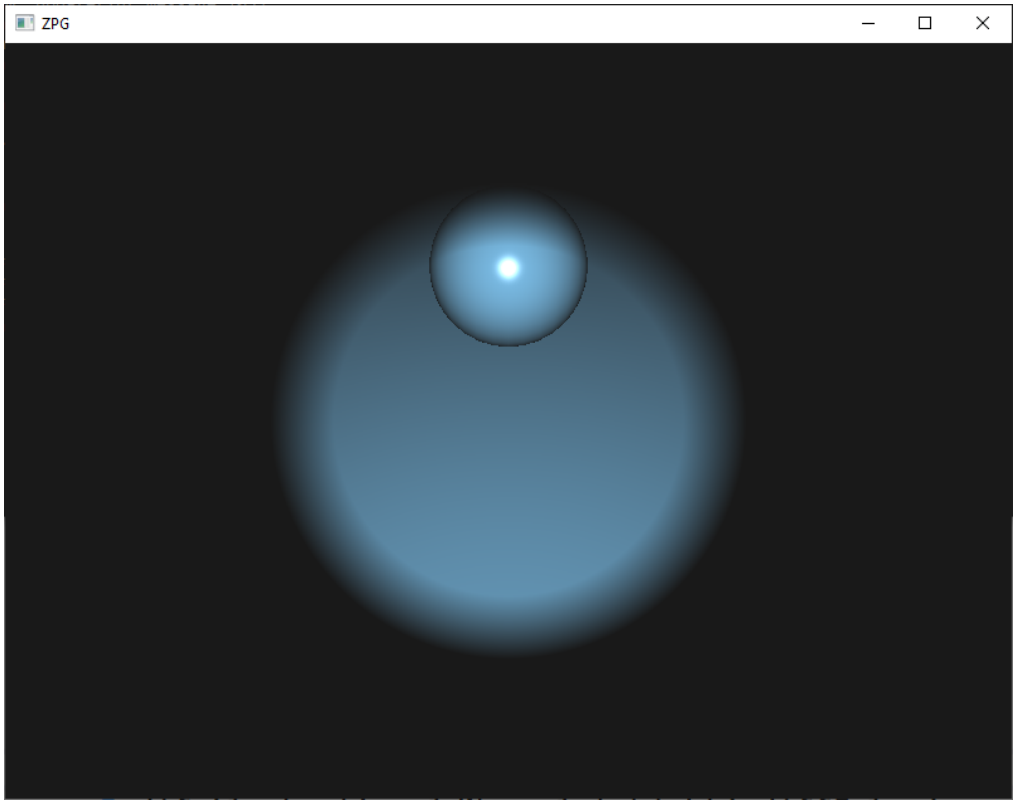

```
alpha = cos(radians(angle));
if (dotLF > alpha)
    finalColor += lightIntensity;
```

Reflektor - různé úpravy



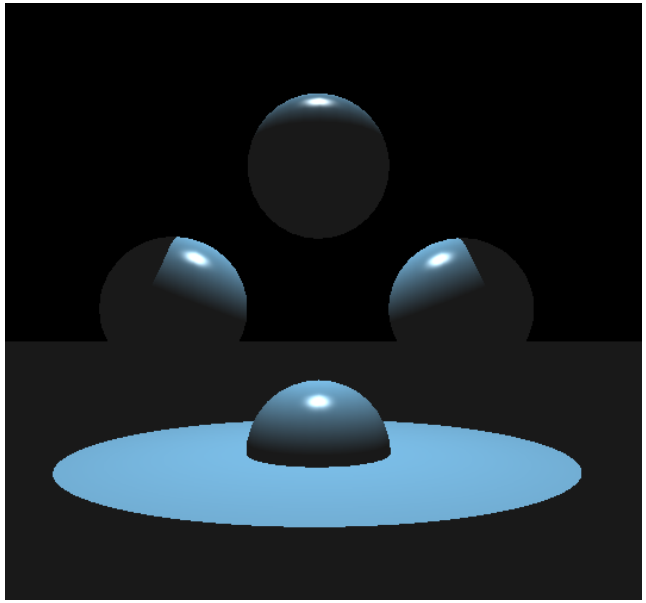

```
// Normalization Formula  xn = ( x - Min )/( Max - Min )
 float intens = (dotLF - alpha)/( 1-alpha);
 //(0.9 - 0.5) / (1 - 0.5) = 0.8
 //(0.6 - 0.5) / (1 - 0.5) = 0.2
```

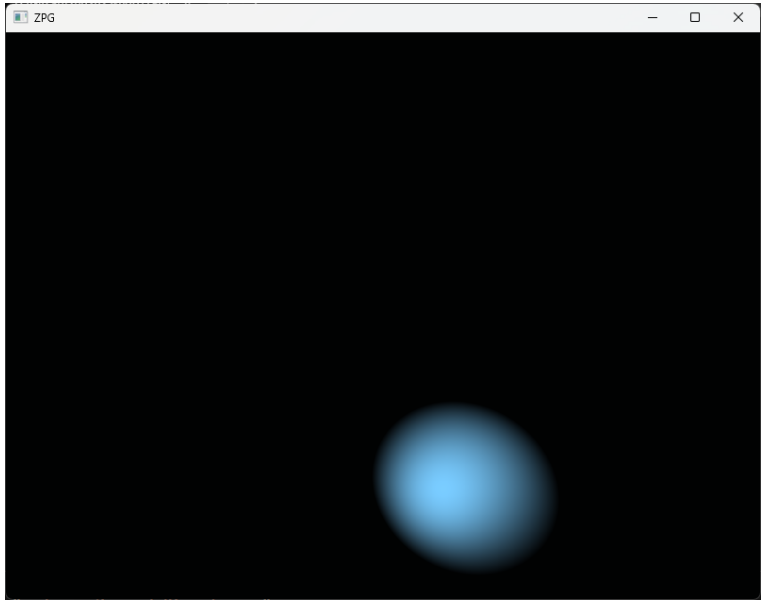
Reflektor - různé úpravy




```
float intens = (dotLF - alpha)/( beta-alpha);
intens = clamp(intens, 0.0,1.0); //oreze na <0,1>
//(0.9 - 0.5) / (0.6 - 0.5) = 4
//(0.6 - 0.5) / (0.6 - 0.5) = 1
//(0.55 - 0.5) / (0.6 - 0.5) = 0.5
```

Reflektor





Závěr

Dotazy?