Virtualna Okruženja

Laboratorijska vježba 1

Izvještaj

1. Preslikavanje neravnina

Vježbu sam riješila koristeći Microsoft DirectX Runtime i Visual Studio Code.

Za ostvarivanje preslikavanja neravnina potreban je jedan prolaz kroz cjevovod. Potrebni su odgovarajući programi za sjenčanje fragmenata i vrhova. Na izlazu su potrebni vektori prema oku i prema svjetlu

U program trebalo je dodati sljedeće linije koda:

1.) Računanje normale:

```
float3 n = normalize(float3(-1,-1,-1) + 2*tex2D(normalMap,IN.texCoord));
```

Normalu n je bilo potrebno pretvoriti iz intervala [0,1] u interval [-1,1]. To sam napravila preko formule:

$$f(t) = c + \left(\frac{d-c}{b-a}\right)(t-a)$$

Gdje je:

a=0

b=1

c=-1

d=1

t= vrijednost koja se pretvara

f(t) = -1 + (1-(-1))/(1-0)*(t-0)

f(t) = -1 + 2/1 * t

f(t) = -1 +2*t

2.) Računanje polu-vektora h:

```
float3 h = normalize(IN.lightDir + IN.viewDir);
```

Uz dodanu liniju, u programu su već postojale potrebne linije za računanje vektora l i v.

```
float3 l = normalize(IN.lightDir);
float3 v = normalize(IN.viewDir);
```

3.)Računanje veličine difuzne komponente:

```
float nDotL = dot(n, l);
```

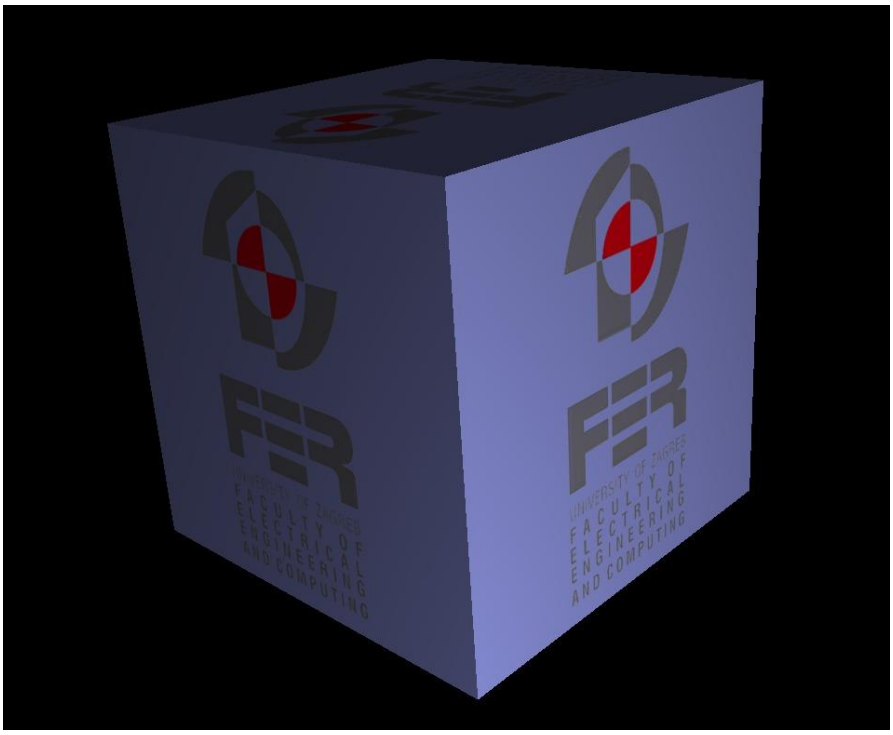
4.)Računanje veličine reflektirajuće komponente

```
float nSpecL = pow(dot(n, h), material.shininess);
```

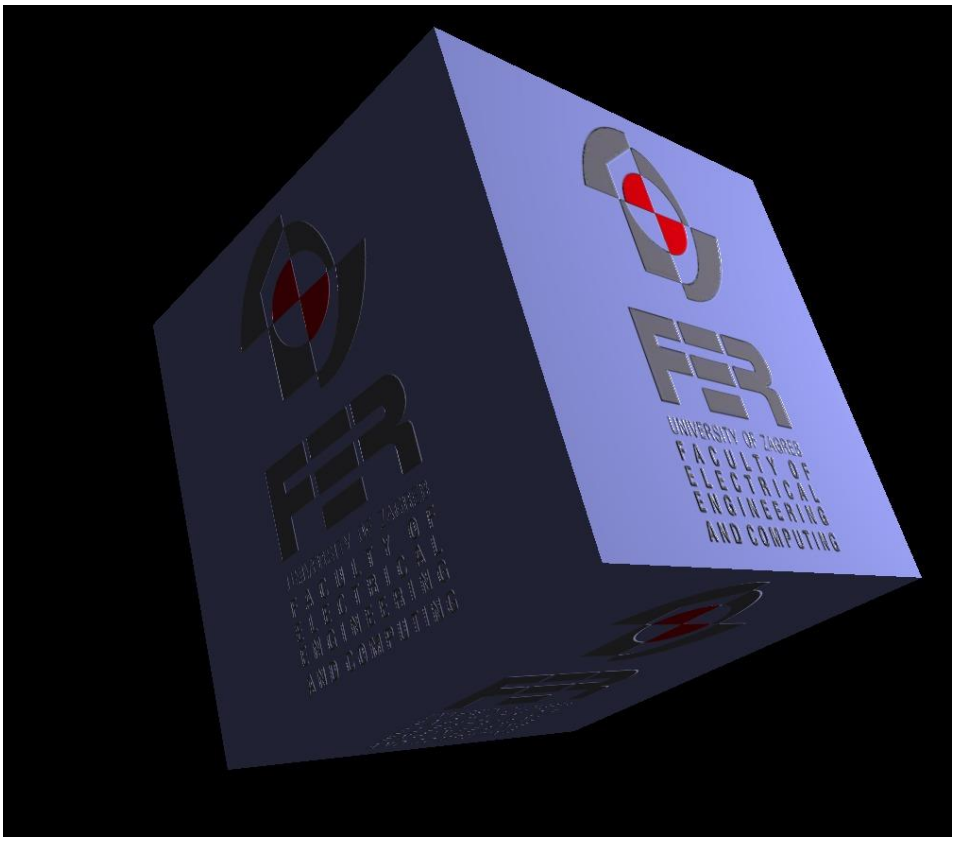
5.)Ukupni efekt osvjetljenja (dodavanje reflektirajuće komponente u formulu)

```
float4 color = (material.ambient * light.ambient) +
                    ( material.specular * light.specular * nSpecL ) +
               (material.diffuse * light.diffuse * nDotL );
```

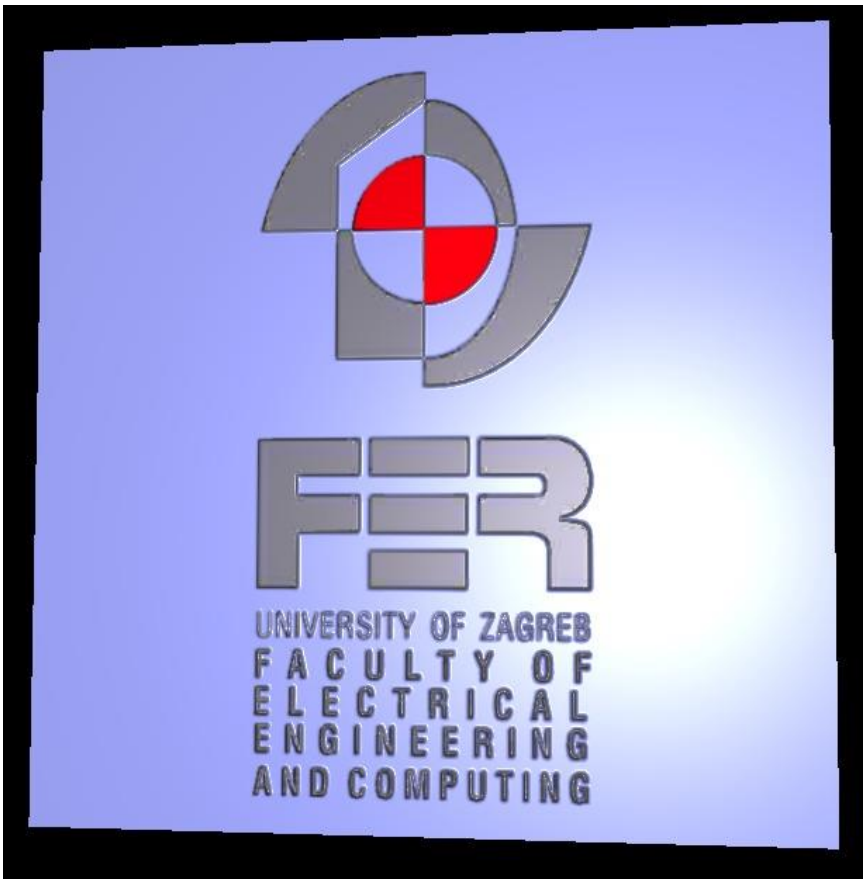
Na slici 1. nalazi se scena bez uključenog preslikavanja neravnina. Slika 2 i 3 prikazuju scenu sa preslikavanjem neravnina.



Slika 1: Scena bez preslikavanja neravnina



Slika 2: Scena s preslikavanjem neravnina



Slika 3: Scena s preslikavanjem neravnina

2. Preslikavanje sjena

Vježbu sam riješila koristeći Microsoft DirectX Runtime i Visual Studio Code.

Za preslikavanje sjena potrebna su dva prolaza kroz cjevovod.

U prvom prolazu se iscrtava iz perspektive svjetla u sceni. Kao rezultat se dobiva tekstura sjena koja sadrži podatke o fragmentima vidljivih iz pozicije svjetla.

U drugom prolazu se iscrtava iz perspektive kamere. Ponovno se računaju udaljenosti fragmenata od svjetla. Uspoređuje se novo dobivena udaljenost sa udaljenosti u teksturi sjena. Ako je udaljenost veća, onda postoji fragment koji je bliže svjetlu i trenutni se fragment nalazi u sjeni i treba ga potamniti

U program trebalo je dodati sljedeće linije koda:

1.)Transformacija vektora:

```
float4 transVPL = (float4(1,1,1,1)+(IN.viewPosLight / IN.viewPosLight.w))/2;
transVPL.y = 1.0 - transVPL.y;
```

Vektor transVPL je bilo potrebno pretvoriti iz intervala [-1,1] u interval [0,1]. To sam napravila preko formule:

$$f(t) = c + \left(\frac{d-c}{b-a}\right)(t-a)$$

Gdje je:

a=-1

b=1

c=0

d=1

t= vrijednost koja se pretvara

f(t) = 0 + (1-0)/(1-(-1))*(t-(-1))

f(t) = 0 + 1/2 * (t+1)

f(t) = 1/2 * (t+1)

2.)Učitavanje dubine:

```
float shadowDepth = tex2D(shadowMap, float2(transVPL.x, transVPL.y)).x;
```

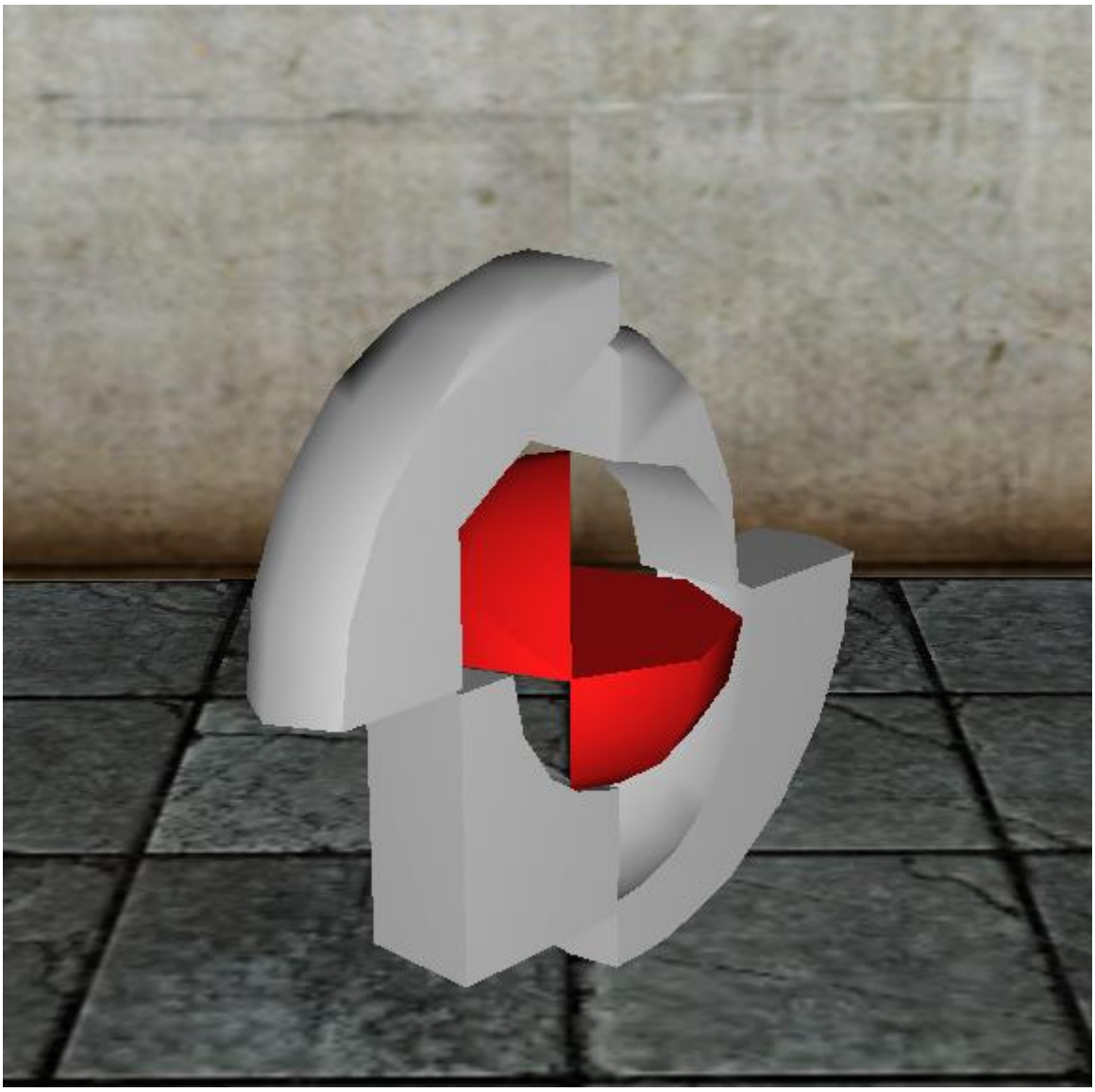
3.)Provjera udaljenosti preko zadane formule:

```
float flag = 1.0;
if( shadowDepth < (IN.viewPosLight.z/xMaxDepth - 0.01) ){
    flag = 0.0;
}
```

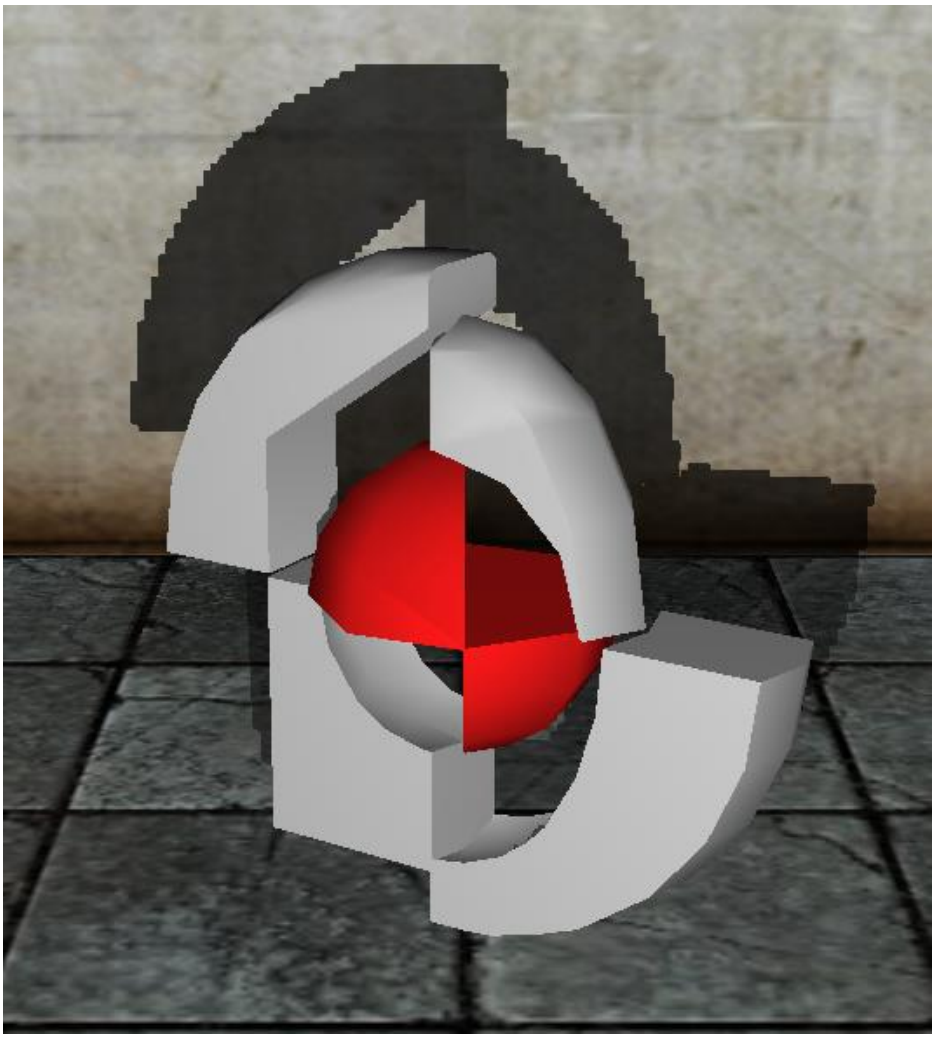
4.)Potpuno osvjetljenje:

```
color =(light.ambient + flag*light.diffuse * nDotL) * material;
```

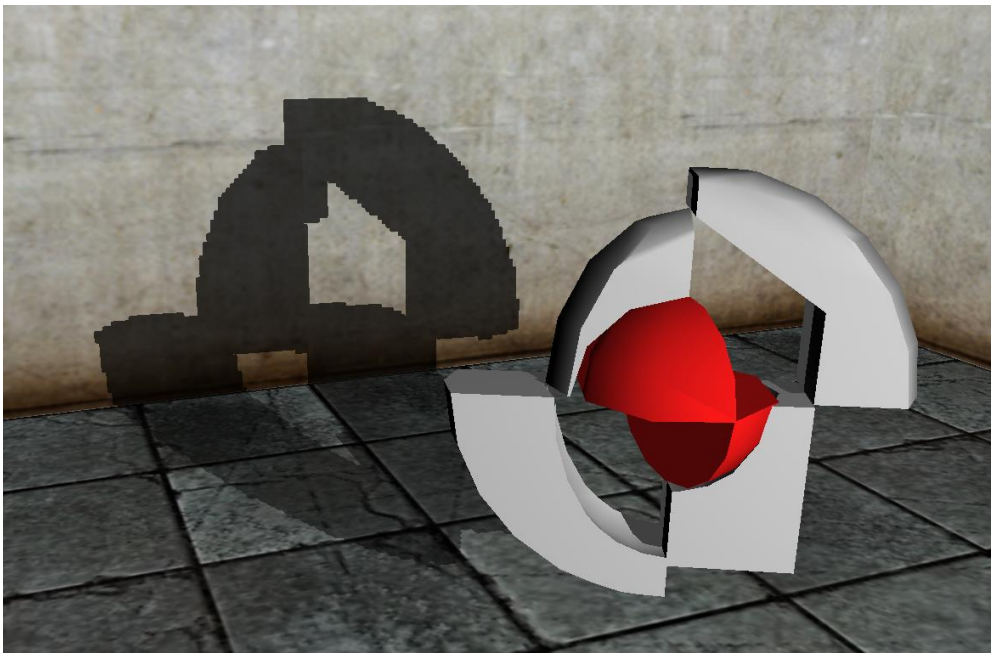
Na slici 4. nalazi se scena bez uključenog preslikavanja sjena. Slika 5 i 6 prikazuju scenu sa preslikavanjem sjena.



Slika 4: Scena prije preslikavanja sjena



Slika 5: Scena s preslikavanja sjena



Slika 6: Scena s preslikavanja sjena