WebGL – Projekcie, nastavenie kamery, textúrovanie a osvetľovanie

doc. Ing. Branislav Sobota, PhD.
Ing. Marián Hudák, Ing. Lenka Bubeňková
Katedra počítačov a informatiky, FEI TU v Košiciach

C 04

Fakulta elektrotechniky a informatiky

Počítačová Grafika

Ciele cvičenia

- WebGL - Príprava balíčka
- WebGL - Projekcie a zobrazovací reťazec
- WebGL - Práca s parametrami kamery, jednoduché GUI
- WebGL - Ovládanie transformácií kamery pomocou klávesnice
- WebGL – Textúrovanie a technika mapovania textúry
- WebGL – Osveľovanie a technika osvetľovania 3D objektov

2. WebGL – projekcie, kamera, textúrovanie a osvetľovanie – príprava balíčka

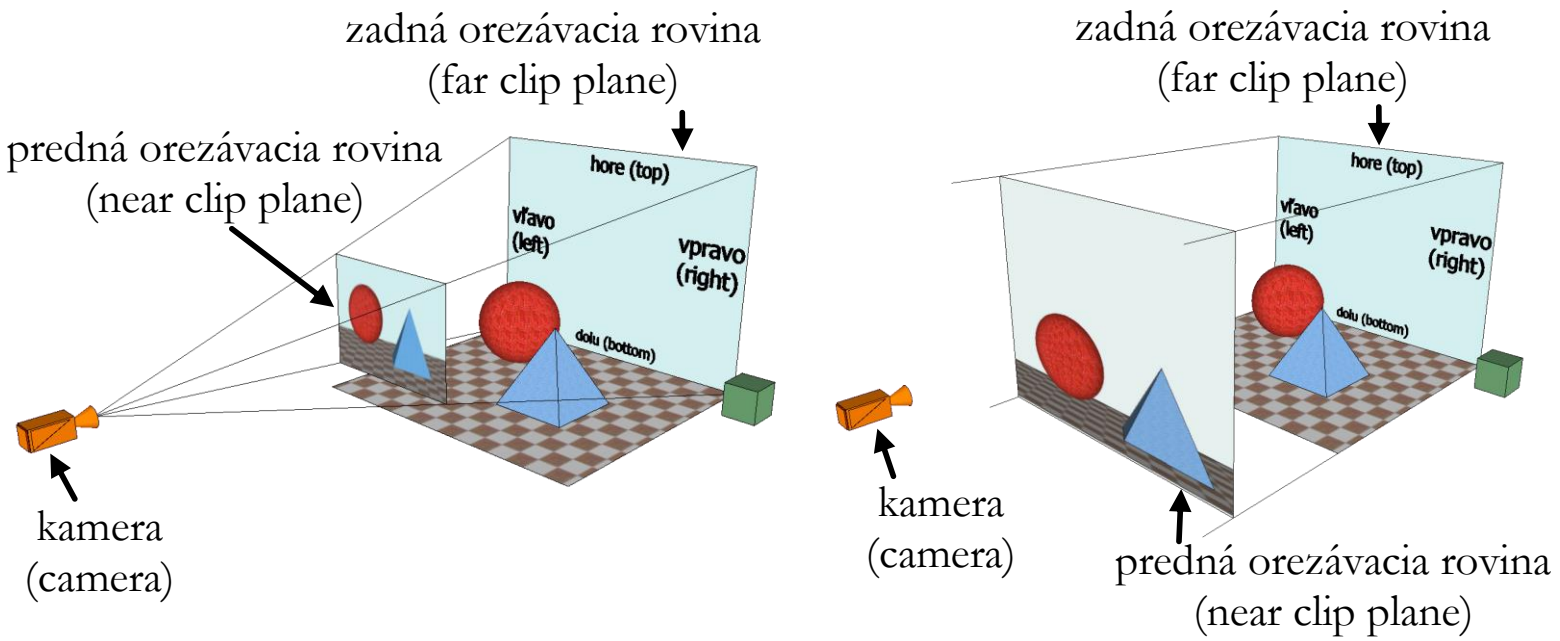
- **Úloha:** Stiahnite si balíček „**WebGL_Projekcie_texturovanie_Osvetlovanie.zip**“ z portálu Moodle KPI a predmetu Počítačova grafika.
- Obsah balíčka skopírujte do vášho projektu aby štruktúra vyzerala nasledovne:
  - `WebGL_getStart`
    - `> css`
    - `>> style.css`
    - `> js`
    - `>> datGUI`
    - `>> glmatrix`
    - `>> GUI_Cube_sliders_proj.js`
    - `>> Cube_Text_Light.js`
    - `models`
    - `> texture`
    - `>> box.jpg`
    - `>> lirkis.jpg`
    - `Index.html`

Súbor „*GUI_Cube_sliders_proj.js*“ je obmenou súboru „*GUI_Cube_sliders.js*“.

1. WebGL – Projekcie a zobrazovací reťazec

Typy projekcií

- Pri práci s trojrozmerným 3D priestorom nesmieme zabúdať, že vykresľovanie bude vo väčšine prípadov do roviny t.j. 2D (napr. obrazovka).

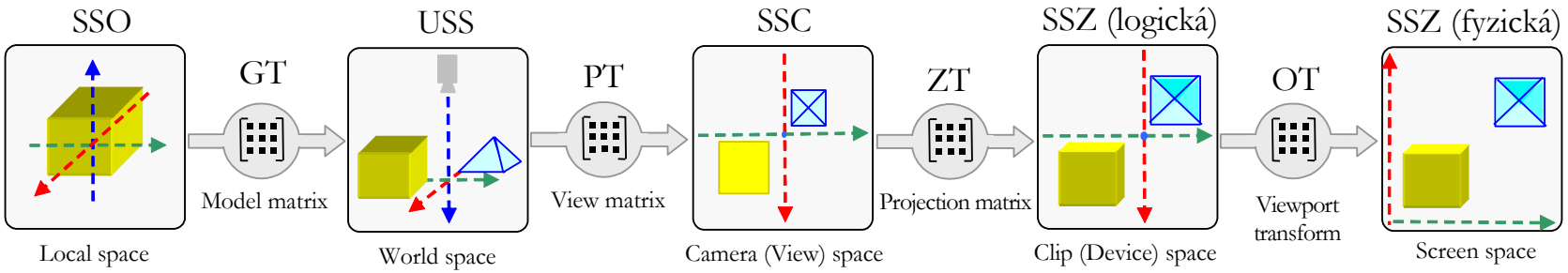


Perspektívna projekcia

Ortografická projekcia

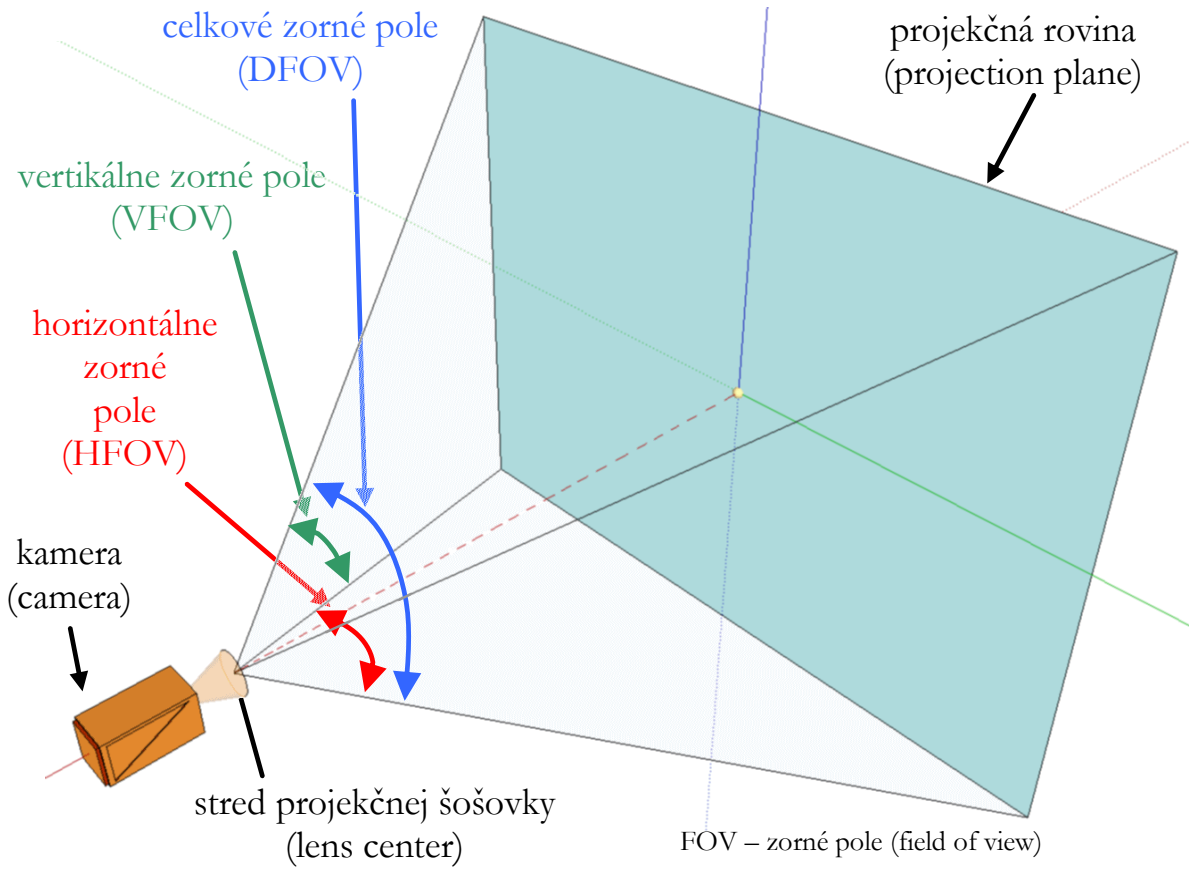
1. WebGL – Projekcie a zobrazovací reťazec
Zobrazovací reťazec

- Pri zobrazovaní 3D priestoru WebGL sa používa nasledujúci zobrazovací reťazec.



3. WebGL – Práca s parametrami kamery

- Zorné pole kamery (Field of view - FOV)



3. WebGL – Práca s parametrami kamery Zorné pole kamery (Field of view - FOV)

- **Úloha:** Vo vývojárskom prostredí si otvore súbor „index.html“ kde použijete cestu pre otvorenie skriptu :

```
<script src="js/GUI_Cube_sliders_proj.js"></script>
```

Odkaz na knižnicu *glmatrix*

```
<script src="js/glmatrix/gl-matrix.js"></script>
```

nemažte, bude potrebný pre prácu s transformáciami.

3. WebGL – Práca s parametrami kamery Zorné pole kamery (Field of view - FOV)

1. Pridajte štýl pre css:

```
<link rel="stylesheet" type="text/css" href="css/style.css">
```

2. Otvorte si súbor **„index.html“** a vložte doň nasledujúce riadky. **Vkladajte ich za ukončeným elementom canvas.**

```
<h3> Camera FOV </h3>
<div class="slidecontainer">
<input type="range" min="0" max="120" value="60" class="slider"
    id="camFOV_value" oninput="setCamFov(this.value)" >
</div>
```

3. WebGL – Práca s parametrami kamery Zorné pole kamery (Field of view - FOV)

3. Otvorte si javascript „**GUI_Cube_sliders_proj.js**“.
4. Pozrite si, že parameter „fieldOfView“ na riadku kódu 255 je komentovaný. Miesto neho je vytvorená globálna premenná s rovnakým názvom (riadok 5) a vstupnou hodnotou 60 prepočítanou na radiány.

```
var fieldOfView = 60 * Math.PI / 180; //nast. počiatočnej hodnoty uhla zorného poľa
```

5. Prezrite si metódu function setCamFov(val), ktorá za globálnu premennú „fieldOfView“ dosadzuje hodnoty načítané zo *slider* elementu (riadok 10).

```
function setCamFov(val) {
     fieldOfView = val *  Math.PI / 180;
}
```

alebo

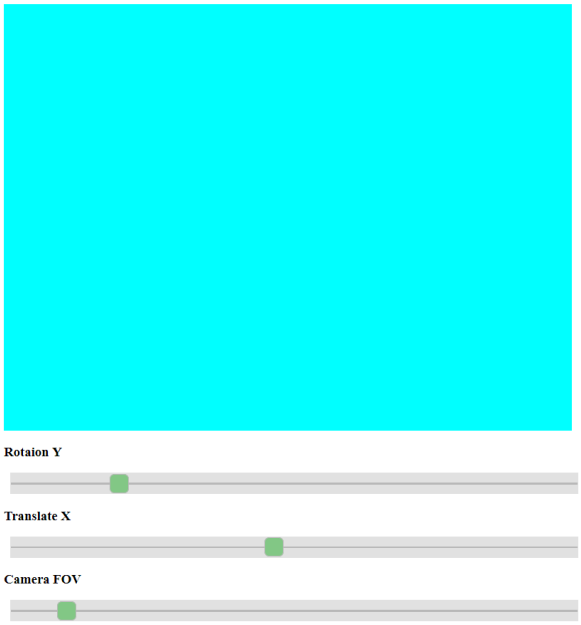
```
function setCamFov(val) {
    document.getElementById('camFOV_value').innerHTML = val;
    fieldOfView = document.getElementById('camFOV_value').innerHTML * Math.PI / 180;
}
```

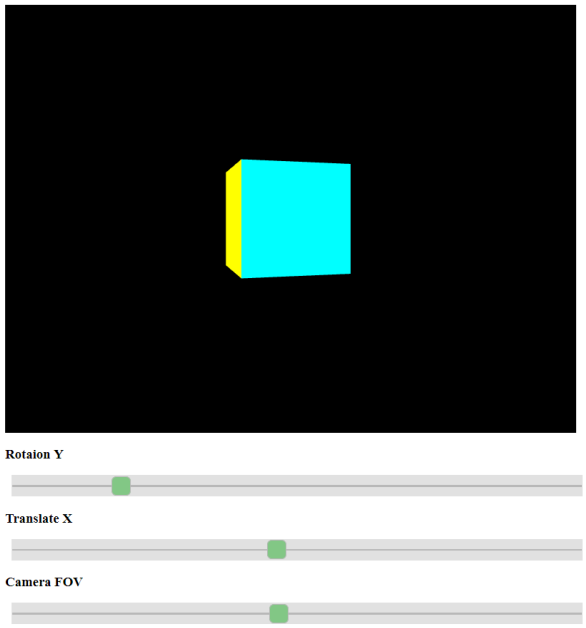
3. WebGL – Práca s parametrami kamery Zorné pole kamery (Field of view - FOV)

6. Prezrite si implementáciu riadkov 263-2671

```
mat4.perspective(projectionMatrix, fieldOfView, aspect, zNear, zFar);
```

7. Spustite si projekt a vyskúšajte meniť hodnoty elementu *bežec* (slider) jeho posúvaním.





3. *WebGL – Práca s parametrami kamery* Ovládanie ostatných parametrov kamery.

- Okrem FOV, medzi ďalšie parametre kamery vytvorenej knižnicou WebGL patria :
- ***Aspect*** *– rozlíšenie kamery*
- ***zNEar*** *– najbližšia hranica vykreslenia*
- ***zFar*** *– najvzdialenejšia hranica vykreslenia*
- ***Projectionmatrix*** *– projekčná matica pre uloženie informácii o vykonávanej projekcii a kamery*

3. *WebGL – Práca s parametrami kamery* Ovládanie ostatných parametrov kamery.

- **Úloha:** Skúste pridať vlastné ovládanie parametrov kamery:

1. Vyberte si niektorý z parametrov v riadoch (256 – 259) v súbore „**GUI_Cube_sliders_proj.js**“.
2. Zakomentujte parameter a vytvorte jeho globálnu premennú s definovanou hodnotou (hodnota určí typ).
3. Vytvorte si metódu, ktorá bude do globálnej premennej parametra dosadzovať hodnoty získané z elementu *slider*. (inšpirujte sa metódami predošlých cvičení).
4. V súbore “**index.html**” vytvorte vlastný *slider* ktorý bude obsahovať :
   - vlastný parameter **`id`** (dosaďte vlastné)
   - volanie metódy **`oninput`** (dosaďte vlastné – metóda, ktorú ste vytvorili v súbore “**GUI_Cube_sliders_proj.js**”) (inšpirujte sa predošlými *slider* elementmi).
5. Spustite vizualizáciu projektu.

4. WebGL – Ovládanie transformácií kamery pomocou klávesnice

- **Úloha:** Implementujte rotáciu kamery okolo osi y pomocou klávesnice.

1. Pozrite si riadky kódu 269 – 272 v súbore „**GUI_Cube_sliders_proj.js**“.
2. Transformácia rotácie okolo osi y je aplikovaná na objekt kamery, pričom sa uhol mení podľa globálnej premennej „**`rotateCamY`**“. Globálna premenná rotácie kamery „**`rotateCamY`**“ je zapísaná v riadku 6.
3. Skúste do javascriptu vložiť nasledujúce riadky kódu tak, aby boli umiestnené na konci skriptu.

```
document.addEventListener('keypress', (event) => {
 const keyName = event.key;
 if(keyName == "w" ) {rotateCamY = rotateCamY + 0.01;}
});
document.addEventListener('keypress', (event) => {
 const keyName = event.key;
 if(keyName == "s" ) {rotateCamY = rotateCamY - 0.01;}
});
```

4. WebGL – Ovládanie transformácií kamery pomocou klávesnice

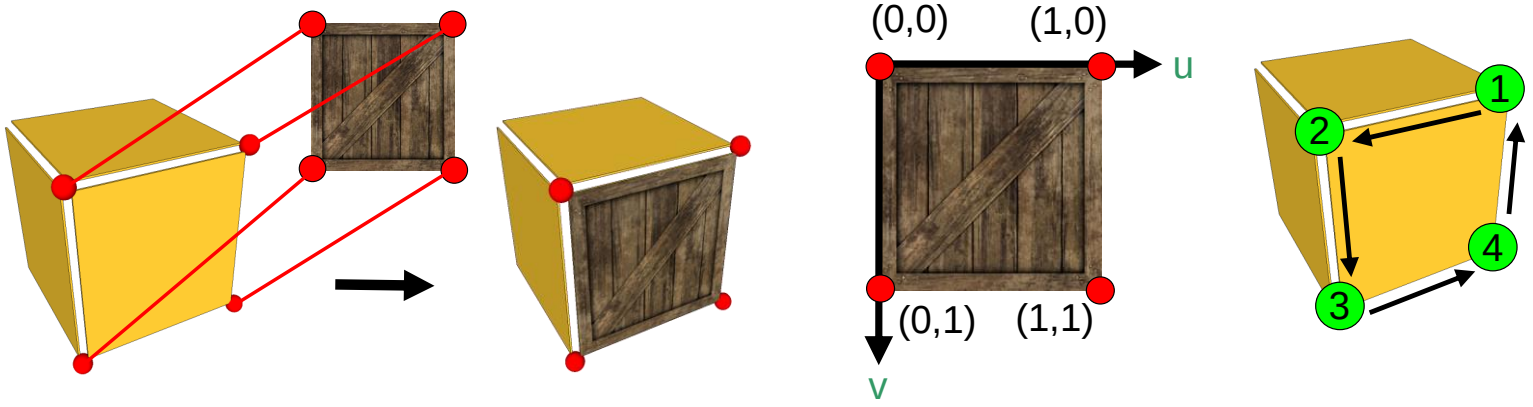
4. Spustite vizualizáciu a ovládajte rotáciu kamery klávesami "**w**" a "**s**" okolo osi *y*.

5. WebGL – Textúrovanie a technika mapovania textúry

- Textúrovanie je proces nanášania obrazových vzoriek (textúr, tapiet) na povrch objektov za účelom získania vizuálneho dojmu, že objekt je z istého materiálu (napr. drevo, kameň, kov a pod.).

5. WebGL – Textúrovanie a technika mapovania textúry

- Pri tomto procese nanášania obrazového formátu na plochy trojrozmerného objektu sa najčastejšie používa ako vzor obrázok (textúra).
- Obrazový formát (obrázok, textúra) využíva dvojrozmernú súradnicovú sústavu **SST [u,v]**, ktorej hodnoty súradníc **u,v** zodpovedajú jednotkovej miere obrazového formátu. **Súradnica „u“** zodpovedá **x-ovej osi obrázka (textúry)**, **súradnica „v“** zodpovedá ***y*-ovej osi textúry.**
- **Jedna jednotka** = dĺžka/šírka obrázku textúry. (u/v)



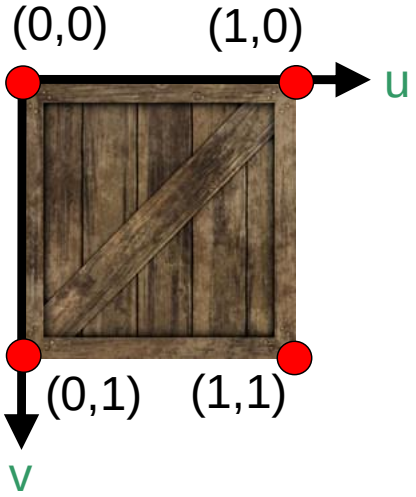
5. *WEBGL* – TEXTÚROVANIE – IMPLEMENTÁCIA TEXTÚROVANIA KOCKY

- **Úloha:** Implementujte textúru na objekt kocky

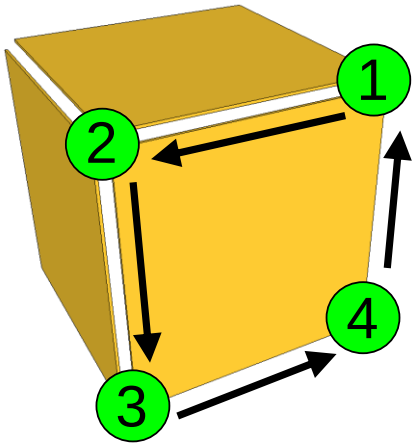
1. **Vo vývojárskom prostredí si otvorte súbor** „Cube_Text_Light.js“.
2. **Prezrite si implementáciu na riadkoch kódu :**
   - Riadok 7 s premennou pre zápis cesty k súboru textúry.
   - Riadok 101 Priradenie textúry/cesty súboru textúry vykreslenej na plochách objektu kocky.
   - Riadky 233-266 `const textureCoordinates` pre priradenie súradníc textúry na plochy objektu kocky.

5. WebGL – TEXTÚROVANIE – IMPLEMENTÁCIA TEXTÚROVANIA KOCKY

```
const textureCoordinates = [
 // Predná stena (Front)
 0.0,  0.0,
 1.0,  0.0,
 1.0,  1.0,
 0.0,  1.0,
. . .
];
```




```
const positions = [
 // Predná stena (Front face)
 -1.0, -1.0,  1.0,
  1.0, -1.0,  1.0,
  1.0,  1.0,  1.0,
 -1.0,  1.0,  1.0,
. . .
];
```



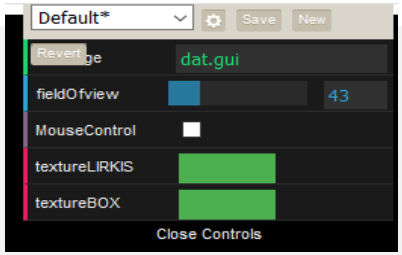
5. *WEBGL* – TEXTÚROVANIE
– IMPLEMENTÁCIA TEXTÚROVANIA KOCKY

3. Spustite vizualizáciu. Pozrite si nanesenú textúru. Vizualizáciu objektu kocky pri stlačení **tlačidiel *textureLIRKIS*** a ***textureBOX*** ukazuje nasledujúci obrázok :

5. WebGL – TEXTÚROVANIE – GRAFICKÉ POUŽÍVATEĽSKÉ ROZHRANIE DAT.GUI

- Pozrite si použité grafické rozhranie, ktoré je vstavané pre potreby vizualizácie. Pre vytvorenie rozhrania je použitá knižnica ***dat.GUI***, ktorá poskytuje rôzne rozšírenia.
- Grafické používateľské rozhranie je implementované v súbore „**Cube_Text_Light.js**“ v riadkoch kódu **610-638**
- **Implementácia dátovej štruktúry GUI Text :**

```
var GUItext =  {
 message:'dat.gui',
 fieldOfview:60,
 MouseControl:false,
 textureLIRKIS: function () {
                texturePath = './texture/lirkis.jpg';
                main();
              },
 textureBOX: function () {
              texturePath = './texture/box.jpg';
              main();
            },
}
```



Vytvorenie objektu rozhrania :

```
window.onload = function() {
 //var text = new GUItext();
 var gui = new dat.GUI();
 gui.remember(GUItext);
 gui.add(GUItext, 'message');
 gui.add(GUItext, 'fieldOfview', 20, 120);
 gui.add(GUItext, 'MouseControl');
 gui.add(GUItext, 'textureLIRKIS');
 gui.add(GUItext, 'textureBOX');
 gui.open();
};
```

6. WebGL – Textúrovanie

– Implementácia textúrovania kocky

- **Úloha:** Prepíšte hodnoty pre mapovanie textúry tak, aby ste textúru namapovali otočenú o 90°.

(riadok 137)

```
const positions = [
 // Predná stena (Front face)
  1.0,  1.0,  1.0,
  1.0, -1.0,  1.0,
 -1.0, -1.0,  1.0,
 -1.0,  1.0,  1.0,
. . .
];
```

alebo

(riadok 233)

```
const textureCoordinates = [
 // Predná stena (Front)
 1.0,  0.0,
 1.0,  1.0,
 0.0,  1.0,
 0.0,  0.0,
. . .
];
```

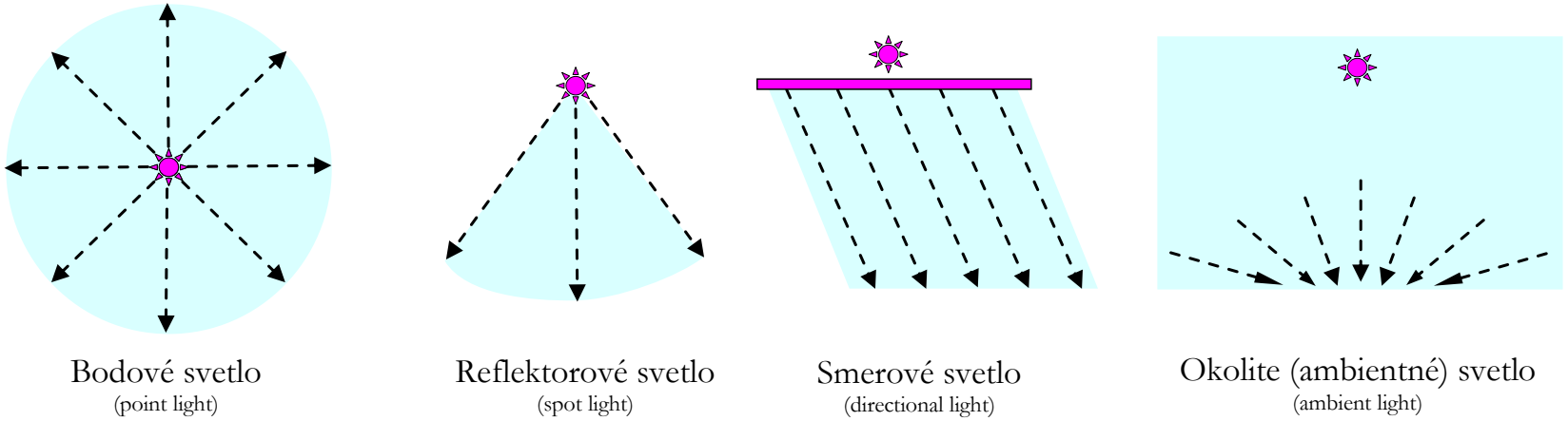
- **Úloha:** Nastavte textúru aplikovanú na plochy kocky tak aby 1 stena zobrazovala ½ textúry obrázka

```
const textureCoordinates = [
 // Predná stena (Front)
 0.0,  0.0,
 0.5,  0.0,
 0.5,  0.5,
 0.0,  0.5,
. . .
];
```

6. WebGL – Osvetľovanie

– Osvetľovanie a technika osvetľovania objektov

- Väčšina grafických knižníc a rámcov disponuje štandardnou množinou svetelných prvkov.
- **Medzi štandardné svetelné prvky sú radené :**
  - svetlo prostredia (okolia, ambient light)
  - bodové svetlo (point light)
  - reflektorové svetlo (spot light)
  - smerové svetlo (directional light)



Bodové svetlo (point light)

Reflektorové svetlo (spot light)

Smerové svetlo (directional light)

Okolite (ambientné) svetlo (ambient light)

6. WebGL – Osvetľovanie

– Osvetľovanie a technika osvetľovania objektov

- **Úloha:** Implementujte svetelné prvky vo WebGL

1. Vo vývojárskom prostredí preštudujte v súbore „*Cube_Text_Light.js*“ nasledujúce riadky kódu :
2. Riadky 48-50 pre vytvorenie svetla prostredia (*ambientLight*) a smerového svetla (*directionalLight*). Dôsledne preštudujte parametre, ktorými je možné ovplyvniť farbu svetla s hodnotami R,G,B. Pozrite si akým spôsobom je implementovaný smer svietenia smerového svetla (*directionalVector*).

```
                    ambientLight = vec3(R, G, B);
               directionalLightColor= vec3(R, G, B);
directionalVector = normalize(vec3(0.85, 0.8, 0.75));
```

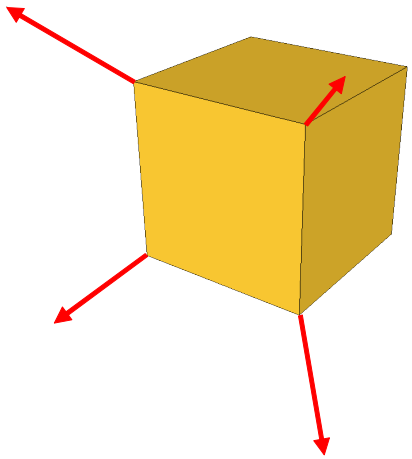
pozícia bodu, do ktorého smeruje lúč

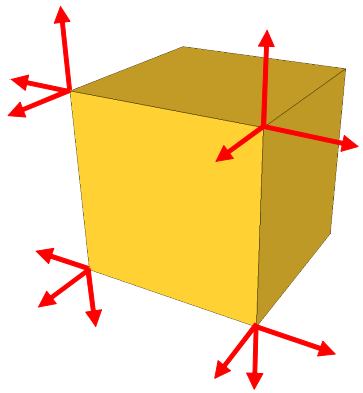
6. WebGL – Osvetľovanie

– Osvetľovanie a technika osvetľovania objektov

3. **Riadky 186-222** obsahujú súradnice bodov kocky, z ktorých sú vedené vektory ich normál. Vektor každej normály je kolmý na plochu kocky z ktorej vychádza.

```
const vertexNormals = [
 // Predná stena (Front face)
 0.0,  0.0,  1.0,
 0.0,  0.0,  1.0,
 0.0,  0.0,  1.0,
 0.0,  0.0,  1.0,
 . . .
]
```

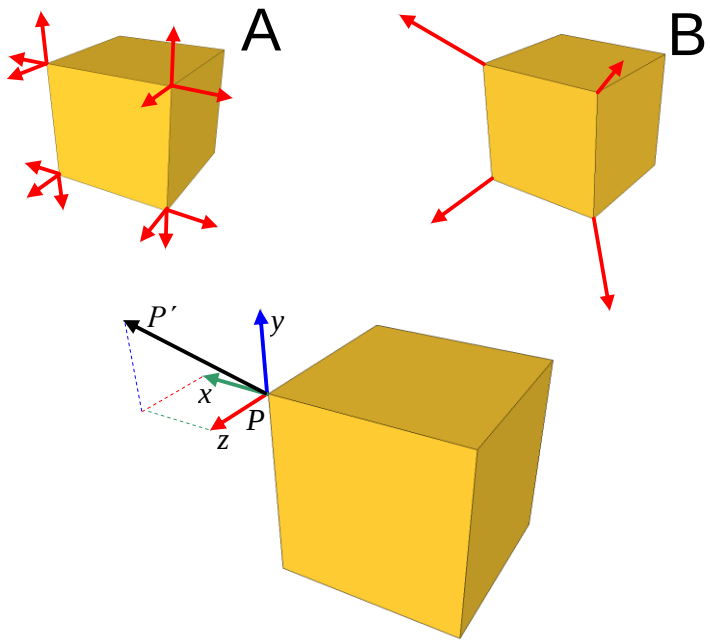




6. WebGL – Osvetľovanie

– Osvetľovanie a technika osvetľovania objektov

- Vektory normál sú v implementácii zapísané spôsobom, ktorý je uvedený na obrázku nižšie (A) pre každý smer v smere osi x,*y*,z a zároveň pre každý bod *P* (vertex) kocky (pole vertexNormals).
- Vo finálnej fáze sa pre každý vrchol (vertex) sčítajú koncové súradnice vektorov jeho normál.
- Sčítaním koncových súradníc vznikne koncový bod *P´*, ktorý určí nový smerový vektor. Jeho počiatok bude začínať v bode kocky *P* (vertex) a smerovať bude do koncového vypočítaného bodu *P´* (B).



6. WebGL – Osvetľovanie

– Osvetľovanie a technika osvetľovania objektov

- **Úloha:** Prepíšte hodnoty RGB pre zmenu farby (riadok 49) svetelného prvku tak, aby ste získali nasledujúce výsledky (orientácia textúry nie je podstatná):

```
directionalLightColor= vec3(0, 1, 1);
```

```
directionalLightColor= vec3(1, 0, 1);
```

```
directionalLightColor= vec3(1, 0, 0);
```

DOPLŇUJÚCE ÚLOHY

- **V rámci práce s kamerou vyskúšajte samostatne implementovať :**
  - Vlastný HTML prvok v súbore „index.html“
  - Vlastnú metódu v súbore „*GUI_Cube_sliders_proj.js*“
  - Naviazať prvok s metódou.
  - Vytvoriť globálnu premennú v súbore „*GUI_Cube_sliders_proj.js*“
  - Dosadiť globálnu premennú za niektorý z parametrov projekcie kamery.
  - Predveďte hotovú implementáciu cvičiacemu.
- **Nastavte textúru aplikovanú na plochy kocky tak aby :**
  - 1 stena obsahovala dlaždice 2x2 (4 dlaždice)
  - 1 stena zobrazovala ¼ textúry obrázka
  - Predveďte hotovú implementáciu cvičiacemu.

ÚLOHY NA SAMOSTATNÉ RIEŠENIE

- Implementujte funkcionalitu takú, aby pri stlačení klávesov "**w**", "**a**", "**s**" a "**d**" sa vykonával pohyb kamery v súradnicovom systéme **x,y,z**.
- Nastavte textúru aplikovanú na plochy kocky tak aby 2 steny boli korektne osadené logom LIRKIS
- **Implementujte funkcionalitu** takú, **aby pri stlačení klávesov** "**w**", "**a**", "**s**" a "**d"** sa vykonával pohyb (posun alebo rotácia) vybraného svetelného zdroja v súradnicovom systéme x,y,z.

Q & A

branislav.sobota@tuke.sk
lenka.bubenkova@tuke.sk

Katedra počítačov a informatiky, FEI TU v Košiciach