Nepovinné úkoly na procvičování po 2. cvičení

Úkoly jsou pouze pro procvičení látky, není to písemka, nebudou kontrolovány. Případné dotazy směřujte na diskuzní fórum nebo pošlete e-mail cvičícím.

Poznámka: V jednotlivých sekcích jsou úkoly odstupňovány podle náročnosti.

Procvičování modelové matice

Úkol 1: Pokud jste na cvičeních nestihli stůl, vytvořte jej. Na stole vytvořte pyramidu nebo věž z kostiček.

Úkol 2: Vytvořte model sluneční soustavy: Slunce, okolo něj obíhající planety a Měsíc obíhající okolo Země.

Úkol 3: Vytvořte jednoduchou scénu s podlahou a několika objekty nad ní, např. kostka, konvička, a po stisku klávesy G nechte objekty na podlahu spadnout. Pohyb objektů k podlaze simulujte jako působení gravitační síly Země.

Procvičování pohledové matice

Úkoly 4: Pomocí klávesnice a/nebo myši upravujte pozici pozorovatele tak, aby bylo možné si scénu se stolem prohlédnout z libovolného místa. Můžete například pomocí šipek měnit pozici oka pozorovatele (místo, na které se díváte, a směr nahoru zůstane stejný), ale možnosti necháváme na Vás, úkolem je pouze to, aby bylo možné si scénu prohlédnout.

Nápověda: Např. je možné pohybovat s kamerou po kružnici okolo scény.

Úkol 5: Vytvořte kameru jako z first-person-shooter hry, to znamená:

- směr, kterým se uživatel dívá, se ovládá pohybem myši,
- kamera se pohybuje, dokud držíte klávesu na klávesnici, bez trhání,
- směr pohybu dopředu/dozadu/vlevo/vpravo je dán směrem, kterým se uživatel dívá.

Nápověda: Pozici kamery nemusíte nastavovat jen pomocí `Matrix4f.lookAt`, její pozici a natočení lze definovat i pomocí matic translace a rotace, jen si dobře uvědomte, v jakém pořadí se mají tyto matice znásobit a jaké pozice/úhly musíte použít.

Úkol 6: Umístěte kameru na povrch některé z planet sluneční soustavy z minulého odstavce, kamera samozřejmě musí být stále na stejném místě povrchu této planety, i když se planeta pohybuje.

Procvičování projekční matice

Úkol 7: Ve scéně se stolem nechte uživatele přepínat mezi ortografickou (`Matrix4f.ortho`) a perspektivní projekcí.

Úkol 8: Pomocí `glViewport` rozdělte okno na 4 části a ve všech částech vykreslete tutéž scénu se stolem: v jedné části použijte pohled zepředu (ortografickou projekci), ve druhé pohled shora (taky orto. p.), ve třetí pohled zboku (taky orto. p.) a ve čtvrté perspektivní projekci.

Úkol 9: Vytvořte projekční matici pro kabinetní projekci (viz https://cs.wikipedia.org/wiki/Kosoúhlé_promítání). Kabinetní projekce se často používá na středních školách, je to ta projekce, kde se osa z zobrazuje pod úhlem 45 stupňů a všechny velikosti v této ose se zkracují na polovinu.

- Zobrazte v této projekci jednoduchou kostku a ověřte, že je projekce opravdu správně, třeba pravítkem a úhloměrem :-).
- Použijte tuto projekci na scénu se stolem, umožněte uživateli dívat se z různých míst.

Vzorové řešení vybraných příkladů

Úkol 1: Na stole vytvořte pyramidu nebo věž z kostiček.

Řešení:

- vykreslíme pyramidu ve `Cv2 → render()`
- nový kód (Cv2.java):

```
...
private void drawCube(Matrix4f mv, Matrix4f p, Vector3f pos, Vector3f color) {
    Matrix4f scaledMV = new Matrix4f(mv)
            .translate(pos)
            .scale(0.125f);
    drawModel(scaledMV, p, cubeArray, 36, color);
}
...
...
private void render() {
    ...
    Matrix4f cubesMV = new Matrix4f(view)
                .translate(0f, 1.6f, 0f);

    // bottom row
    drawCube(cubesMV, projection, new Vector3f(-0.35f, 0f, 0f), new Vector3f(1f, 0f, 0f));
    drawCube(cubesMV, projection, new Vector3f( 0f,    0f, 0f), new Vector3f(0f, 1f, 0f));
    drawCube(cubesMV, projection, new Vector3f( 0.35f, 0f, 0f), new Vector3f(0f, 0f, 1f));

    // middle row
    drawCube(cubesMV, projection, new Vector3f(-0.175f, 0.25f, 0f), new Vector3f(1f, 1f, 0f));
    drawCube(cubesMV, projection, new Vector3f( 0.175f, 0.25f, 0f), new Vector3f(0f, 1f, 1f));

    // top
    drawCube(cubesMV, projection, new Vector3f(0.0f, 0.5f, 0f), new Vector3f(1f, 1f, 1f));
    ...
}
...
```

Úkol 4: Pomocí klávesnice a/nebo myši upravujte pozici pozorovatele tak, aby bylo možné si scénu se stolem prohlédnout z libovolného místa.

Řešení:

- do třídy Cv2 přidáme nový atribut *angle* typu `float`
- přidáme/rozšíříme obsluhu kláves o stisk a opakování kláves E a Q pro úvpravu *angle*
- nový/upravený kód (Cv2.java):

```
// new class variable
private float angle;
...
...
// new code in KeyCallback or equivalent in GLUT
if (action == GLFW_PRESS || action == GLFW_REPEAT) {
    switch (key) {
        case GLFW_KEY_E:
            angle += 0.1;
            break;
        case GLFW_KEY_Q:
            angle -= 0.1;
            break;
    }
}
...
...
// modifed view matrix
Matrix4f view = new Matrix4f()
        .lookAt(5f * (float) Math.sin(angle), 4f, 5f * (float) Math.cos(angle),
                0f, 0f, 0f,
                0f, 1f, 0f);
```