Підготовка:

Робота на машині:

Захист: Лабораторна робота №4

Тема: Побудова проекцій тривимірних об’єктів

Завдання

# Розробити програму, що будує точкову (перспективну) проекцію тривимірного об’єкта. Об’єкт для побудови вибрати самостійно та затвердити у викладача. Об’єкт повинен мати понад 50 базових точок і складатися із 20 або більше багатокутників (полігонів) .

**Теоретичні відомості**

Процес виведення тривимірної графічної інформації по суті є складнішим, ніж відповідний двовимірний процес. Складність, характерна для тривимірного випадку, ґрунтується на тому, що поверхня виведення не має графічного третього виміру. Така невідповідність між просторовими об’єктами і пласкими зображеннями усувається шляхом введення проекцій, які відображують тривимірні об’єкти на двовимірній проекційній картинній площині. В процесі виведення тривимірної графічної інформації ми задаємо видимий об'єм в світовому просторі, проекцію на картинну плоскість і поле виводу на видовій поверхні. У загальному випадку об'єкти, визначені в тривимірному світовому просторі, відсікаються по межах тривимірного видимого об'єму і після цього проектуються. Те, що потрапляє в межі вікна, яке само є проекцією видимого об'єму на картинну плоскість, потім перетвориться в полі виводу і відображується на графічному пристрої. У загальному випадку операція проекції перетворює точки, задані в системі координат розмірності n, в точки системи координат розмірності меншій, ніж n. У нашому випадку точка тривимірного простору відображується в двовимірний простір. Проекція тривимірного об’єкта будується за допомогою прямих променів, що його проецирують. Ці промені називаються проекторами і вони виходять з центру проекції, проходять через кожну точку об’єкта і, при перетинанні картинної площини, створюють проекцію. На рис. 10 представлені дві різні проекції одного й того самого відрізку і проектори, які проходять через його кінцеві точки.

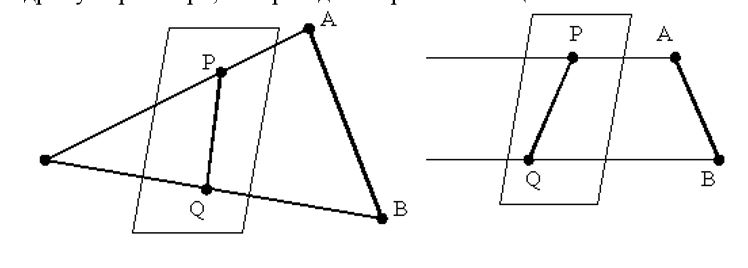


Рис. 10 – Побудова проекцій

Визначений таким чином клас проекцій відомий під назвою плоских геометричних проекцій, оскільки проектування здійснюється на площину, а не на викривлену поверхню і як проектори використовують прямі лінії. Пласкі геометричні проекції можна розділити на два основні класи: центральні (перспективні) і паралельні (ортогональні). Відмінність між ними визначається співвідношенням між центром проекції і проекційною плоскістю. Так, якщо відстань між ними, звичайна, то проекція буде центральною, якщо ж вона нескінчена, то – паралельною. При описі центральної проекції ми явно задаємо її центр, тоді як для паралельної проекції ми вказуємо лише напрям проектування. Центр проекції породжує візуальний ефект, аналогічний тому, до якого приводять фотографічні системи і використовується у випадках, коли бажано досягти деякої міри реалізму. Слід зауважити, що

розмір центральної проекції об'єкту змінюється зворотно пропорційно до відстані від центру проекції до об'єкту. Паралельна проекція породжує менш реалістичне зображення, оскільки відсутнє перспективне "укорочення" об'єкту. Проекція фіксує дійсні розміри об'єкту, і паралельні лінії залишаються паралельними. У загальному випадку завдання побудови центральної проекції полягає в тому, щоб визначити проекцію точки об'єкту, розташованої в довільному місці тривимірного простору, на деяку площину в цьому ж просторі. Цю площину називають картинною. Знаходження центральної проекції є окремим випадком завдання визначення перетинання променя L з площиною α в тривимірному просторі (рис. 11)

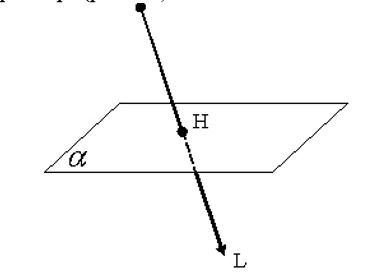


Рис.11 – Перетинання променя з площиною

У комп’ютерній графіці завдання обчислення центральної проекції, як правило, сильно спрощене. В даному випадку центр проекції, який також називають точкою зору, знаходиться на одній з осей системи координат, картинна (проекційна) площина перпендикулярна оптичній осі. Як правило, точку зору (центр проекції) розташовують на осі OZ, тоді картинна площина буде паралельна площині OXY системи координат (рис. 12).

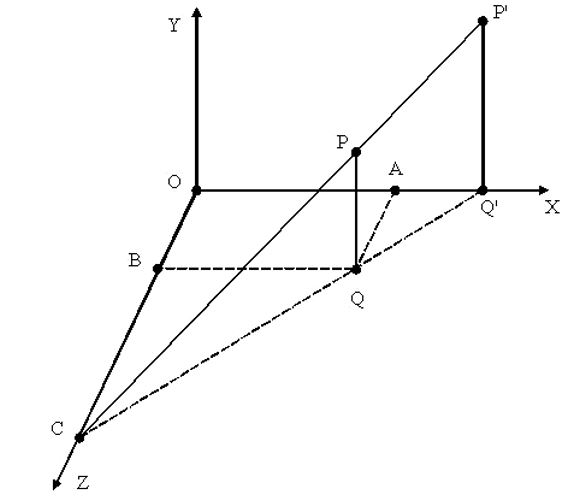


Рис.12 – Схема побудови центральної проекції

В нашому випадку точка C=(0,0,c) - центр проекції (положення спостерігача), площина z=0 - картинна площина. Нехай точка P=(x,y,z) має проекцію P'=(x',y',0). Розглянемо два подібних трикутника CPQ та CP'Q', і запишемо відношення катетів:

Таким чином, крок проектування можна описати в термінах матричної операції добутку. В результаті цього ми можемо поєднати разом операції перетворення об’єкта (перенесення, масштабування, обертання які потрібно буде реалізувати в наступній лабораторній роботі) і операцію проектування в одну загальну матрицю перетворення. Аналогічно можна зробити з приведенням точок, які проектуються, до екранних координат:

Код програми представлений на лістингу 4

**public class** Main **extends** Application  
{  
 **private** PerspectiveCamera **camera** = **new** PerspectiveCamera(**false**);  
 **private** BorderPane **borderPane** = **new** BorderPane();  
 **private** MeshView **meshView** = **this**.createMeshView();  
 **private** PointLight **light** = **new** PointLight();  
 **private** AmbientLight **ambientLight** = **new** AmbientLight();  
   
  
 **public static void** main(String[] args)  
 {  
 Application.*launch*(args);  
 }  
  
 @Override  
 **public void** start(Stage stage)  
 {  
 Group root = **new** Group(**borderPane**, **meshView**, **light**, **ambientLight**);  
 Scene scene = **new** Scene(root, 800, 700, **true**);  
 stage.setScene(scene);  
  
 **borderPane**.setCenter(**camera**);

**borderPane**.setLeft(**vBox**);  
  
 **meshView**.setTranslateX(-100);  
 **meshView**.setTranslateY(250);  
 **meshView**.setTranslateZ(300);  
  
 **ambientLight**.setTranslateX(0);  
 **ambientLight**.setTranslateY(250);  
 **ambientLight**.setTranslateZ(-200);  
  
 stage.setTitle(**"Fool"**);  
 stage.show();  
  
 }  
  
 **public** MeshView createMeshView()  
 {  
 **float**[] points =  
 {  
 420, 0, 20, *// A* 500, 130, 0, *// D* 400, 130, 0, *// C* 480, 0, 20, *// B* 420, 0, 80, *// J* 500, 130, 100, *// o* 400, 130, 100, *// K* 480, 0, 80, *// L* 430, -20, 30,  
 430, -20, 70,  
 470, -20, 70,  
 470, -20, 30,  
  
 415, -50, 15,  
 415, -50, 85,  
 485, -50, 85,  
 485, -50, 15,  
  
 415, -80, 15,  
 415, -80, 85,  
 485, -80, 85,  
 485, -80, 15,  
  
 410, 160, 10,  
 410, 160, 90,  
 490, 160, 90,  
 490, 160, 10,  
  
 410, 180, 10,  
 410, 180, 90,  
 490, 180, 90,  
 490, 180, 10,  
  
 410, 180, 130,  
 490, 180, 130,  
  
 386, 87, 37,  
 386, 87, 64,  
 406, 90, 64,  
 406, 90, 37,  
  
 400, 24, 39,  
 400, 24, 61,  
 415, 27, 61,  
 415, 27, 39,  
  
 386, 87, 64,  
 386, 87, 104,  
 407, 90, 104,  
 407, 90, 64,  
  
 390, 67, 64,  
 390, 67, 104,  
 407, 70, 104,  
 407, 70, 64,  
  
 514, 87, 63,  
 514, 87, 36,  
 494, 90, 36,  
 494, 90, 63,  
  
 504, 24, 61,  
 504, 24, 39,  
 484, 27, 39,  
 484, 27, 61,  
  
 514, 87, 104,  
 514, 87, 63,  
 494, 90, 63,  
 494, 90, 104,  
  
 510, 67, 104,  
 510, 67, 62,  
 490, 70, 62,  
 490, 70, 104  
 };  
  
 **float**[] texCoords =  
 {  
 1.0f, 1.0f, 1.0f, 1.0f, 1.0f, 1.0f, 1.0f, 1.0f,  
 1.0f, 1.0f, 0.5f, 1.0f, 0.0f, 5.0f, 0.0f, 0.0f,  
 1.0f, 1.0f, 0.5f, 1.0f, 0.0f, 5.0f, 0.0f, 0.0f,  
  
 1.0f, 1.0f, 0.5f, 1.0f, 0.0f, 5.0f, 0.0f, 0.0f,  
 1.0f, 1.0f, 0.5f, 1.0f, 0.0f, 5.0f, 0.0f, 0.0f,  
 1.0f, 1.0f, 0.5f, 1.0f, 0.0f, 5.0f, 0.0f, 0.0f,  
 1.0f, 1.0f, 0.5f, 1.0f, 0.0f, 5.0f, 0.0f, 0.0f,  
 0.0f, 5.0f, 0.0f, 0.0f, 0.5f, 1.0f, 0.0f, 5.0f,  
 0.0f, 0.0f, 0.0f, 5.0f, 0.5f, 1.0f, 0.0f, 5.0f,  
 0.0f, 0.0f, 0.0f, 5.0f, 0.5f, 1.0f, 0.0f, 5.0f,  
 0.0f, 0.0f, 0.0f, 5.0f, 0.5f, 1.0f, 0.0f, 5.0f,  
 0.0f, 0.0f, 0.0f, 5.0f, 0.5f, 1.0f, 0.0f, 5.0f,  
 0.0f, 0.0f, 0.0f, 5.0f, 0.5f, 1.0f, 0.0f, 5.0f,  
 0.0f, 0.0f, 0.0f, 5.0f, 0.5f, 1.0f, 0.0f, 5.0f,  
 0.0f, 0.0f, 0.0f, 5.0f, 0.5f, 1.0f, 0.0f, 5.0f,  
 0.0f, 0.0f, 0.0f, 5.0f  
 };  
  
 **int**[] faces =  
 {  
 0, 0, 2, 2, 1, 1, *// front* 1, 1, 3, 3, 0, 0,  
 0, 0, 6, 6, 2, 2,  
 6, 6, 0, 0, 4, 4,  
 5, 5, 6, 6, 4, 4,  
 5, 5, 4, 4, 7, 7,  
 1, 1, 5, 5, 7, 7,  
 1, 1, 7, 7, 3, 3,  
  
 0, 0, 8, 8, 9, 9,  
 4, 4, 0, 0, 9, 9,  
 4, 4, 9, 9, 10, 10,  
 7, 7, 4, 4, 10, 10,  
 3, 3, 11, 11, 8, 8,  
 0, 0, 3, 3, 8, 8,  
 7, 7, 10, 10, 11, 11,  
 3, 3, 7, 7, 11, 11,  
  
 8, 8, 12, 12, 13, 13,  
 9, 9, 8, 8, 13, 13,  
 9, 9, 13, 13, 14, 14,  
 10, 10, 9, 9, 14, 14,  
 10, 10, 14, 14, 15, 15,  
 11, 11, 10, 10, 15, 15,  
 11, 11, 15, 15, 12, 12,  
 8, 8, 11, 11, 12, 12,  
  
 12, 12, 16, 16, 17, 17,  
 13, 13, 12, 12, 17, 17,  
 13, 13, 17, 17, 18, 18,  
 14, 14, 13, 13, 18, 18,  
 14, 14, 18, 18, 19, 19,  
 15, 15, 14, 14, 19, 19,  
 15, 15, 19, 19, 16, 16,  
 12, 12, 15, 15, 16, 16,  
 16, 16, 19, 19, 18, 18,  
 17, 17, 16, 16, 18, 18, *//32* 20, 20, 1, 1, 2, 2,  
 21, 21, 20, 20, 2, 2,  
 21, 21, 2, 2, 6, 6,  
 22, 22, 21, 21, 6, 6,  
 22, 22, 6, 6, 5, 5,  
 23, 23, 22, 22, 5, 5,  
 23, 23, 5, 5, 1, 1,  
 20, 20, 23, 23, 1, 1,  
  
 24, 24, 20, 20, 21, 21,  
 25, 25, 24, 24, 21, 21,  
 25, 25, 21, 21, 22, 22,  
 26, 26, 25, 25, 22, 22,  
 26, 26, 22, 22, 23, 23,  
 27, 27, 26, 26, 23, 23,  
 27, 27, 23, 23, 20, 20,  
 24, 24, 27, 27, 20, 20,  
  
 25, 25, 21, 21, 28, 28,  
 26, 26, 29, 29, 22, 22,  
 21, 21, 22, 22, 28, 28,  
 22, 22, 29, 29, 28, 28,  
 24, 24, 26, 26, 27, 27,  
 25, 25, 26, 26, 24, 24,  
  
 30, 30, 34, 34, 35, 35,  
 31, 31, 30, 30, 35, 35,  
 31, 31, 35, 35, 36, 36,  
 32, 32, 31, 31, 36, 36,  
 32, 32, 36, 36, 37, 37,  
 33, 33, 32, 32, 37, 37,  
 33, 33, 37, 37, 34, 34,  
 30, 30, 33, 33, 34, 34,  
  
 34, 34, 37, 37, 36, 36,  
 35, 35, 34, 34, 36, 36,  
 30, 30, 32, 32, 33, 33,  
 31, 31, 32, 32, 30 ,30,  
  
 38, 38, 42, 42, 43, 43, *//front* 39, 39, 38, 38, 43, 43,  
 39, 39, 43, 43, 44, 44,  
 40, 40, 39, 39, 44, 44,  
 40, 40, 44, 44, 45, 45,  
 41, 41, 40, 40, 45, 45,  
 41, 41, 45, 45, 42, 42,  
 38, 38, 41, 41, 42, 42,  
 42, 42, 45, 45, 44, 44,  
 43, 43, 42, 42, 44, 44,  
 38, 38, 40, 40, 41, 41,  
 39, 39, 40, 40, 38, 38,  
  
  
 46, 46, 50, 50, 51, 51,  
 47, 47, 46, 46, 51, 51,  
 47, 47, 51, 51, 52, 52,  
 48, 48, 47, 47, 52, 52,  
 48, 48, 52, 52, 53, 53,  
 49, 49, 48, 48, 53, 53,  
 49, 49, 53, 53, 50, 50,  
 46, 46, 49, 49, 50, 50,  
 50, 50, 53, 53, 52, 52,  
 51, 51, 50, 50, 52, 52,  
 46, 46, 48, 48, 49, 49,  
 47, 47, 48, 48, 46, 46,  
  
 54, 54, 58, 58, 59, 59, *// front* 55, 55, 54, 54, 59, 59,  
 55, 55, 59, 59, 60, 60,  
 56, 56, 55, 55, 60, 60,  
 56, 56, 60, 60, 61, 61,  
 57, 57, 56, 56, 61, 61,  
 57, 57, 61, 61, 58, 58,  
 54, 54, 57, 57, 58, 58,  
 58, 58, 61, 61, 60, 60,  
 59, 59, 58, 58, 60, 60,  
 54, 54, 56, 56, 57, 57,  
 55, 55, 56, 56, 54, 54,  
 };  
  
 **int**[] facesmoothes =  
 {  
 0,0, 1,1, 0,0, 1,1, 0,0, 1,1, 1,1, 0,0,  
 1,1, 0,0, 1,1, 0,0, 0,0, 1,1, 0,0, 1,1,  
 0,0, 1,1,  
 2,3, 3,3, 0,0, 3,3, 2,3, 3,3, 2,2,  
 1,1, 0,0, 1,1, 0,0, 1,1, 0,0, 1,1, 0,0,  
 9,9, 0,0, 1,1, 2,2, 3,3, 4,4, 5,5,  
 0,0, 1,1, 2,2, 3,3, 4,4, 5,5, 6,6, 7,7, 10,10, 9,9,  
 8,8, 8,8  
 };  
  
 TriangleMesh mesh = **new** TriangleMesh();  
 mesh.getPoints().addAll(points);  
 mesh.getTexCoords().addAll(texCoords);  
 mesh.getFaces().addAll(faces);  
 PhongMaterial material = **new** PhongMaterial();  
 material.setDiffuseColor(Color.***GREEN***);  
 MeshView meshView = **new** MeshView();  
 meshView.setMaterial(material);  
 mesh.getFaceSmoothingGroups().addAll(facesmoothes); *//108* meshView.setMesh(mesh);  
 **return** meshView;  
 }  
}

Лістинг 4 – код программи

Результат роботи

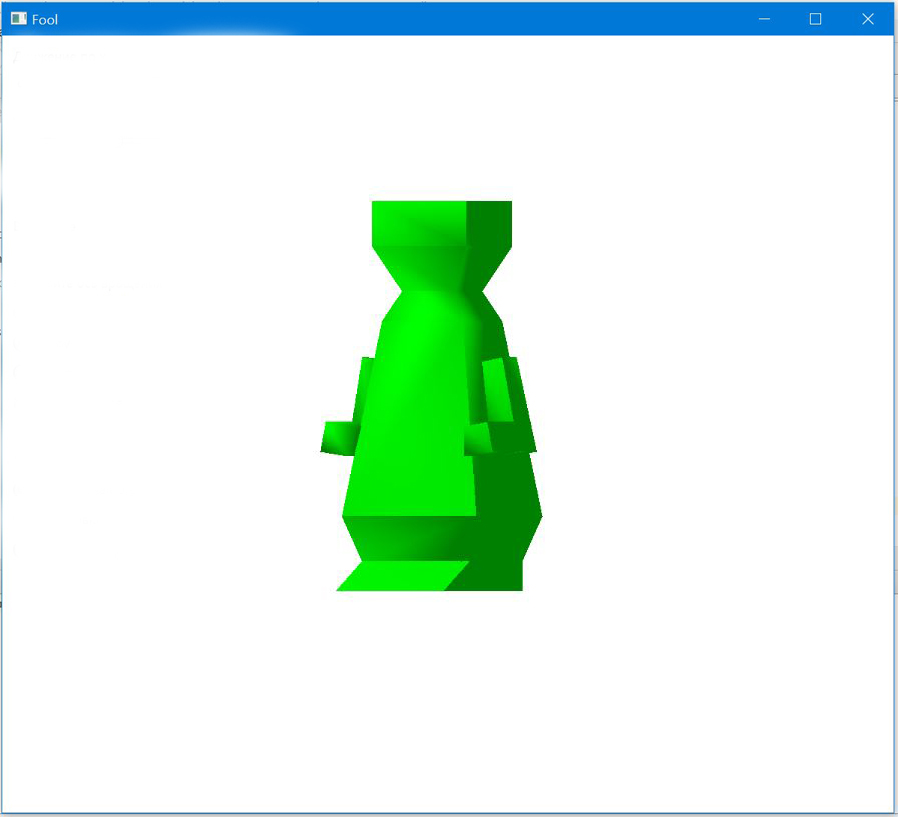


Рисунок 13 – результат роботи