Лекція №6

Тема: Тривимірні координати. Тривимірне векторне зображення

Мета: Реалізувати каркасне зображення тривимірних об’єктів.

ПЛАН

1. Матричний запис перетворення тривимірних координат

2. Тривимірна сцена

3. Камера

4. Тест глибини, відсікання невидимих пікселів

1. Матричний запис перетворення тривимірних координат

В п’ятій лекції показано виведення матриці перетворення для двовимірного простору:

$$\begin{pmatrix} x_1 \\ y_1 \\ 1 \end{pmatrix} = \begin{pmatrix} m_x\cos(\alpha) & m_y\sin(\alpha) & dx \\ -m_x\sin(\alpha) & m_y\cos(\alpha) & dy \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_0 \\ y_0 \\ 1 \end{pmatrix} .$$

Відтворення тривимірних об’єктів вимагає роботи з трьома просторовими координатами, і для врахування третьої координати, якої перетворення не стосується матриця перетворюється так:

$$\begin{pmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{pmatrix} = \begin{pmatrix} m_x\cos(\alpha) & m_y\sin(\alpha) & 0 & dx \\ -m_x\sin(\alpha) & m_y\cos(\alpha) & 0 & dy \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{pmatrix} .$$

В наведеній матриці використане перетворення масштабування розмірів по осям OX, OY а також повороту на кут відносно осі OZ. Однак більш вигідно комбінувати перетворення окремо:

$$\begin{pmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{pmatrix} = \begin{pmatrix} m_x & 0 & 0 & dx \\ 0 & m_y & 0 & dy \\ 0 & 0 & m_z & dz \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\gamma) & \sin(\gamma) & 0 \\ 0 & -\sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} \cos(\beta) & 0 & \sin(\beta) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\beta) & 0 & \cos(\beta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} \cos(\alpha) & \sin(\alpha) & 0 & 0 \\ -\sin(\alpha) & \cos(\alpha) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{pmatrix} .$$

За цим перетворенням істотне значення має порядок виконаних перетворень, бо у

випадку переносу точки з наступним обертанням, обертатиметься вже перенесена точка. Напроти, комбінація обертання та перенесення дасть інше положення фігури в просторі. Тому в наведеному прикладі повного перетворення використано послідовність обертання навколо осей, а потім операцію перенесення, що дозволить зберегти очікуване місце перенесення з вказаним масштабуванням.

Зазначена схема перетворень, дозволяє будувати перетворення обертання навколо заданої точки в просторі. Нехай точкою обертання є точка (*a,b,c*). Тоді обертання можна позначити так:

$$\begin{pmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & c \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} \cos(\alpha) & \sin(\alpha) & 0 & 0 \\ -\sin(\alpha) & \cos(\alpha) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & -a \\ 0 & 1 & 0 & -b \\ 0 & 0 & 1 & -c \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{pmatrix} .$$

Сенс у такому перетворенні є у порядку перетворень: 1) пересуваємо об’єкт до центру обертання; 2) обертаємо на заданий кут; 3) повертаємо фігуру до попереднього положення. Цей приклад наочно показує перевагу матричного запису перетворення, де обертання навколо заданої точки за допомогою виразу записати досить важко.

2. Тривимірна сцена

Тривимірна сцена є сукупність тривимірних об’єктів, які задані переліком координат точок та переліком пласких полігонів за порядковими номерами зазначених точок. Також на об’єкт, як загальна величина визначається колір та правило обробки освітленості, правила фарбування. Для визначення положення всіх об’єктів задаються матриці перетворення координат. При цьому, для одного й того ж об’єкту може бути декілька матриць перетворення. Наприклад, таке можна використати при наявності декількох копій об’єкту: за столом розташувати декілька стільців. Істотно, що завантажувати багато об’єктів-стільців не обов’язково, достатньо до кожного з стільців вказати потрібний набір вершин та полігонів та матрицю перетворення для потрібного розташування стільця в просторі.

З вище згаданого можна зробити висновок, що тривимірна сцена є сукупність об’єктів, які є сукупностями вершин та полігонів, а також сукупність посилань на об’єкти з матрицями перетворення для правильного їх розташування в просторі. Один об’єкт може бути на сцені розташований декілька раз.

3. Камера

Вигляд тривимірної сцени залежить від місця спостереження та напрямку погляду. Сукупність інформації, яка задає місце та напрямок спостереження називають “камера”, що є асоціацією з кінематографічною камерою, яка знімає сцену з фільму. Також на вигляд сцени впливає й “фокусна відстань” “об’єктиву” камери, або перспективне масштабування віддалених об’єктів. Якщо зменшення об’єктів зі зміною відстані не відбувається, то така проекція називається “паралельною”. Всі проекції є своєрідними перетвореннями координат, після яких до уваги беруться лише дві координати (x,y). Розглянемо на початок паралельну проекцію.

Для створення паралельної проекції потрібно всю сцену повернути таким чином, щоб центр екрану збігався з початком координат, а напрямок зору був по напрямку осі глибини OZ. Задамо положення ока за допомогою трьох координат (*a,b,c,*1). Створимо матрицю, яка пересуне всю сцену так, щоб точка спостереження була в точці (0,0,0,1):

$$\begin{pmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & -a \\ 0 & 1 & 0 & -b \\ 0 & 0 & 1 & -c \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{pmatrix} \quad .$$

Тепер, після перенесення, сцену потрібно повернути на кут $-\gamma$, який визначає кут на який “піднято голову”. Знак мінус означає, що сцену потрібно повернути так, щоб компенсувати нахил голови:

$$\begin{pmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\gamma) & -\sin(\gamma) & 0 \\ 0 & \sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & -a \\ 0 & 1 & 0 & -b \\ 0 & 0 & 1 & -c \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{pmatrix} \quad .$$

Поворот камери “ліворуч-праворуч” задає кут повороту навколо вертикальної осі OY $-\beta$:

$$\begin{pmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{pmatrix} = \begin{pmatrix} \cos(\beta) & 0 & -\sin(\beta) & 0 \\ 0 & 1 & 0 & 0 \\ \sin(\beta) & 0 & \cos(\beta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\gamma) & -\sin(\gamma) & 0 \\ 0 & \sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & -a \\ 0 & 1 & 0 & -b \\ 0 & 0 & 1 & -c \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{pmatrix}$$

Іноді використовують додатково перетворення зміни масштабу та зсуву координат, щоб перетворити власні координати до координат сумісних з координатами пристрою виведення графічних примітивів. Як зазначено вище, для створення зображення на екрані, після перетворення камери використовують лише дві координати: $(x_1; y_1)$.

Для врахування перспективних спотворень використовується співвідношення:

$$x_e = F\,x/(F+z),$$
$$y_e = F\,y/(F+z).$$

Тут в якості умовної фокусної відстані об'єктиву використано величину F, чим менша фокусна відстань, тим є більш значними перспективні спотворення. Також це перетворення можна записати за допомогою матричного добутку:

$$\begin{pmatrix} F\,x \\ F\,y \\ F\,z \\ z+F \end{pmatrix} = \begin{pmatrix} F & 0 & 0 & 0 \\ 0 & F & 0 & 0 \\ 0 & 0 & F & 0 \\ 0 & 0 & 1 & F \end{pmatrix} \cdot \begin{pmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{pmatrix} .$$

В цьому виразі перспективної проекції використано однорідність тривимірних координат, коли остання координата є дільником трьох попередніх. Тобто, для переходу від однорідних координат до декартових потрібно скористатися перетворенням:

$$\begin{pmatrix} F\,x \\ F\,y \\ F\,z \\ z+F \end{pmatrix} \rightarrow (\textit{Перехід до декартових координат}) \rightarrow \begin{pmatrix} x_e = F\,x/(z+F) \\ y_e = F\,y/(z+F) \\ z_e = F\,z/(z+F) \end{pmatrix} ,$$

що і відповідає поставленій задачі отримання перспективної проекції.

4. Тест глибини, відсікання невидимих пікселів

Це один з найпростіших алгоритмів видалення невидимих поверхонь. Вперше він був запропонований Кетмул. Застосовний цей алгоритм для малювання тривимірних сцен, де об'єкти складаються з пласких полігонів. Для кращої наочності розглядатимемо в якості полігонів лише трикутники.

Формальний опис алгоритму z-буфера:

1) заповнити буфер кадру фоновим значенням інтенсивності чи кольору;

2) заповнити z-буфер мінімальним значенням z;

3) перетворити кожен багатокутник в растрову форму в довільному порядку;

4) для кожного пікселю (x, y) в тлі багатокутника обчислити його глибину координату-глибину z, порівняти глибину z(x, y) зі значенням яке вже знаходиться в буфері Z(x, y);

5) якщо z(x, y)>Z(x, y), то записати атрибут цього багатокутника (інтенсивність, колір і т. п.) в буфер кадру і замінити Z(x, y):=z(x, y);

6) в іншому випадку залишити колір пікселю як є (там вже намальований більш близький піксель).

Для оптимізації виведення в z-буфер і для зниження кількості обчислень використовується той факт, що чим раніше видима межа буде виведена в z-буфер, тим більше невидимих граней, які закривають нею, буде відкинуто. Для цього будується список тих граней, які були виведені в попередньому кадрі. В подальшому кадрі ці межі виводяться в z-буфер найпершими, так як майже напевно вони будуть залишатися видимими і в цьому кадрі.

Цей алгоритм вирішує завдання про видалення невидимих поверхонь і робить тривіальною задачу візуалізації перекриття складних поверхонь. Сцени можуть бути будь-якої складності. Оскільки габарити простору зображення фіксовані, оцінка обчислювальної трудомісткості алгоритму є лінійною. Оскільки елементи сцени чи картинки можна заносити в буфер кадру або в z-буфер в довільному порядку, їх не потрібно попередньо сортувати по пріоритету глибини. Це економить обчислювальний час, який витрачається на сортування по глибині.

Основний недолік алгоритму — великий обсяг необхідної пам'яті. Якщо сцена піддається видовому перетворенню і відсікається до фіксованого діапазону координат z значень, то можна використовувати z-буфер з фіксованою точністю. Щодо глибині потрібно обробляти з більшою точністю, ніж координатну інформацію на площині (x, y); зазвичай буває достатньо 20 біт. Буфер кадру розміром 512 * 512 * 24 біт в комбінації з z-буфером розміром 512 * 512 * 20 біт вимагає майже 1.5 мегабайт пам'яті.

Інший недолік алгоритму z-буфера полягає в трудомісткості і високої вартості усунення сходового ефекту, а також реалізації ефектів прозорості і просвічування. Оскільки алгоритм заносить пікселі в буфер кадру в довільному порядку, то нелегко отримати інформацію, необхідну для методів усунення сходового ефекту, що ґрунтуються на попередньої фільтрації. При реалізації ефектів прозорості і просвічування пікселі можуть заноситися в буфер кадру в некоректному порядку, що веде до локальних помилок.