Linearna algebra u računalnoj grafici

Pokos, Mihael

Sveučilište u Rijeci

Fakultet za matematiku

Sveučilišni prijediplomski studij Matematika

Mihael Pokos

Linearna algebra u računalnoj grafici

Završni rad

Rijeka, rujan 2023.

Sveučilište u Rijeci

Fakultet za matematiku

Sveučilišni prijediplomski studij Matematika

Mihael Pokos

Linearna algebra u računalnoj grafici

Mentor: doc. dr. sc. Nevena Jurčević Peček

Završni rad

Rijeka, rujan 2023.

Sadržaj

Poglavlje 1

Sažetak

U ovome radu predstavljamo matematičke strukture i alate linearne algebre koji se koriste u modernim računalnim sustavima renderiranja trodimenzionalne grafike. Potrebe za njihovim uvođenjem motivirane su primjerima problema s kojima se računalni programi susreću pri implementaciji sustava. Predstavljeni su prostori u koje ćemo smjestiti predmete koje želimo renderirati i alati koji omogućuju manipuliranje njima.

Ključne riječi: **vektorski prostori**, **euklidski prostori**, **renderiranje**, **afine transformacije**, **homogene koordinate**

Poglavlje 2

Uvod

Tema ovog završnog rada je primjena alata linearne algebre u području računalne grafike. Kako bismo ih mogli primijeniti, potrebno je poznavati objekte nad kojima će oni djelovati (točke i vektori u prostoru).

Ukratko ćemo predstaviti jedan od osnovnih načina generiranja trodimenzionalne računalne grafike (rasterizacija) i zatim povezati potrebe takve metode sa matematičkim strukturama. Te strukture ćemo redom definirati i objasniti različite prostore u kontekstu grafike koje oni modeliraju. Navest ćemo definiciju samih transformacija i neka njihova poželjna svojstva.

Na kraju ćemo definirati sve transformacije (translacija, rotacija, skaliranje, refleksija, smicanje, projekcije) i predstaviti probleme koji se s njima javljaju. Kod translacije je to problem množenja njene matrica sa ostalima (što rješavamo uvođenjem homogenih koordinata), kod rotacije je to rotacija oko proizvoljnog pravca, a kod projekcija je to projiciranje neželjenih predmeta.

Poglavlje 3

Objekti u renderiranju

Renderiranje 3D grafike je prikazivanje trodimenzionalne scene koja se sastoji od čvrstih predmeta (prozirnih ili neprozirnih), izvora svjetlosti, fluida, i drugih elemenata na dvodimenzionalnom zaslonu korisnika. Metoda renderiranja koju ću predstaviti pokušava modelirati fizičke procese ljudskog vida - slika scene koju osoba može vidjeti nastaje kao posljedica refleksije zraka svjetlosti od predmeta u prostoru prema ljudskom oku. Stoga, kao polazni element stvaranja sustava renderiranja potrebna nam je kamera, odnosno ljudsko oko, koju ćemo apstrahirati kao jednu točku u euklidskom prostoru i smjestiti u ishodište trodimenzionalnog realnog koordinatnog sustava. U ovaj prostor ćemo također smjestiti i predmete u sceni, koje ćemo morati matematički zapisati.

Najčešći način zapisa predmeta je korištenjem trokuta. Osnovni oblici kojima izražavamo predemete u prostoru nazivamo primitivama (engl. *primitives*). Naime, modeli koje ćemo smjestiti u scenu, neovisno o njihovoj kompleksnosti (zakrivljenosti, broju diskontinuiteta), sastavljeni su od mreže trokuta. Oni mogu biti sastavljeni i od drugih poligona ili čak biti izraženi matematičkim jednadžbama, ali tema ovog rada biti će ograničena na trokute. Tako izražene predmete tada možemo smjestiti u koordinatni sustav gdje će vrhovima svakog trokuta biti pridružene trodimenzionalne koordinate.

Potrebno je još modelirati i leću kamere: u koordinatni sustav smjestit ćemo ravninu koja će služiti kao podloga na koju ćemo projicirati predmete. Kako je leća kamere ograničene površine, kasnije ćemo ograničiti i površinu na koju projiciramo. Projekciju objekata možemo vršiti na dva načina: kao ortografsku projekciju i kao perspektivnu projekciju. Sliku na zaslonu dobivamo kao presjek pravaca koji prolaze kroz ishodište i kroz točke objekta sa ravninom na koju ih projiciramo.

3.1 Osnovni matematički objekti

Kao što je prethodno bilo navedeno, predmeti u sceni sastavljeni su od mreže trokuta, a svaki trokut sastavljen je od tri točke. U ovom dijelu rada uvodimo matematičku podlogu potrebnu za razumijevanje tih elemenata.

Definicija 3.1.1. *Neka je* $\mathbf{F}$ *polje. Neprazan skup* $\mathbf{V}$ *na kojem su definirane dvije operacije*

$+ : \mathbf{V} \times \mathbf{V} \to \mathbf{V}$ *(zbrajanje vektora) i*

$\cdot : \mathbf{F} \times \mathbf{V} \to \mathbf{V}$ *(množenje vektora skalarom)*

naziva se ***vektorskim prostorom*** *ako vrijede sljedeća svojstva:*

(VP1) $(\forall a, b, c \in \mathbf{V})\, (a + b) + c = a + (b + c)$.

(VP2) Postoji element $0 \in \mathbf{V}$ *takav da je* $a + 0 = 0 + a = a$*, za svaki* $a \in \mathbf{V}$.

(VP3) Za svaki $a \in \mathbf{V}$ *postoji jedinstven element* $a' \in \mathbf{V}$ *takav da je* $a + a' = a' + a = 0$.

Element a' *je suprotni element od* a *i označava se sa* $-a$.

(VP4) $(\forall a, b \in \mathbf{V})\, a + b = b + a$.

(VP5) $(\forall a, b \in \mathbf{V})(\forall \lambda \in \mathbf{F})\, \lambda(a + b) = \lambda a + \lambda b$.

(VP6) $(\forall a \in \mathbf{V})(\forall \lambda, \mu \in \mathbf{F})\, (\lambda + \mu)a = \lambda a + \mu a$.

(VP7) $(\forall a \in \mathbf{V})(\forall \lambda, \mu \in \mathbf{F})\, (\lambda\mu)a = \lambda(\mu a)$.

(VP8) $(\forall a \in \mathbf{V})\, 1 \cdot a = a$.

Elemente vektorskog prostora nazivamo ***vektorima****, a elemente polja* $\mathbf{F}$ ***skalarima****.[2]*

Za potrebe osnovne računalne grafike, za polje skalara $\mathbf{F}$ uzimamo polje $\mathbb{R}$ realnih brojeva te promatramo realne vektorske prostore.

Definicija 3.1.2. *Neka je* $\mathbf{A}$ *neprazan skup,* $\mathbf{V}$ *vektorski prostor nad poljem* $\mathbf{F}$*, a*

$$v : \mathbf{A} \times \mathbf{A} \to \mathbf{V}$$

preslikavanje sa sljedećim svojstvima:

(A1) $(\forall P \in \mathbf{A})(\forall a \in \mathbf{V})(\exists! Q \in \mathbf{A})\, v(P, Q) = a$.

(A2) $(\forall P, Q, R \in \mathbf{A})\, v(P, Q) + v(Q, R) = v(P, R)$.

Uređenu trojku $(\mathbf{A}, \mathbf{V}, v)$ *nazivamo* ***afinim prostorom*** *nad* $\mathbf{V}$*, a njegove elemente nazivamo* ***točkama****. [3]*

Svaki vektorski prostor $\mathbf{V}$ je afini prostor nad samim sobom shvatimo li elemente od $\mathbf{V}$ kao točke i definiramo preslikavanje $v : \mathbf{A} \times \mathbf{A} \to \mathbf{V}$ s

$$v(a,b) = b - a \tag{3.1}$$

Svojstva (A1) i (A2) iz definicije afinog prostora lako se provjere.

Definicija 3.1.3. *Neka je* $(\mathbf{A}, \mathbf{V}, v)$ *afini prostor i* $P, Q \in \mathbf{A}$*. Uređeni par točaka* (P, Q) *nazivamo* ***orijentiranom dužinom*** *i označavamo s* $\overrightarrow{PQ}$*. Pritom je* P ***početna***, *a* Q ***krajnja točka*** *orijentirane dužine* $\overrightarrow{PQ}$*.*

Definicija 3.1.4. *Neka je* $\mathbf{F}$ *polje realnih brojeva,* $\mathbf{V}$ *vektorski prostor nad poljem* $\mathbf{F}$*, a*

$$s : \mathbf{V} \times \mathbf{V} \to \mathbf{F}$$

preslikavanje sa sljedećim svojstvima:

(S1) $s(a,b) = s(b,a)$.

(S2) $s(\lambda a, b) = \lambda s(a,b)$.

(S3) $s(a+b, c) = s(a,c) + s(b,c)$.

(S4) $a \neq 0 \implies s(a,a) > 0$.

Prethodne tvrdnje vrijede za svaki izbor vektora $a, b, c \in \mathbf{V}$ *i skalara* $\lambda \in \mathbf{F}$*. Preslikavanje* s *nazivamo* ***skalarnim množenjem****.[3]*

Definicija 3.1.5. *Uređeni par* $(\mathbf{V}, s)$ *vektorskog prostora* $\mathbf{V}$ *i skalarnog množenja na tom prostoru nazivamo* ***unitarnim prostorom****.[3]*

Sada konačno možemo navesti definiciju euklidskog prostora.

Definicija 3.1.6. *Afini prostor* $\mathbf{E}$ *kojemu je pridružen unitaran prostor* $\mathbf{U}$ *i preslikavanje* $v : \mathbf{E} \times \mathbf{E} \to \mathbf{U}$ *sa svojstvima:*

(E1) $(\forall T \in \mathbf{E})(\forall v \in \mathbf{U})(\exists! Q \in \mathbf{E})\, v(P,Q) = v$.

(E2) $(\forall P, Q, R \in \mathbf{E})\, v(P,Q) + v(Q,R) = v(P,R)$.

nazivamo ***euklidskim prostorom****.*

3.1.1 Afine transformacije

Prije definicije afinih transformacija, potrebna nam je definicija linearnih operatora.

Definicija 3.1.7. *Neka su* $\mathbf{U}$ *i* $\mathbf{V}$ *vektorski prostori nad istim poljem* $\mathbf{F}$*, a* $f : \mathbf{U} \to \mathbf{V}$ *neko preslikavanje. Kažemo da je to preslikavanje* ***linearni operator*** *ako ima sljedeća*

svojstva:

(LO1) $f(a+b) = f(a) + f(b), \forall a, b \in \mathbf{U}$.

(LO2) $f(\alpha a) = \alpha f(a), \forall \alpha \in \mathbf{F}, \forall a \in \mathbf{U}$.*[3]*

Prije navođenja afinih transformacija koje se koriste u računalnoj grafici, istaknimo neka njihova svojstva. Linearne operatore i afine transformacije zapisujemo koristeći matrice, stoga ćemo poistovjetiti svojstva operatora i svojstva matrica. U nastavku podrazumijevamo da su matrice kvadratne (imaju isti broj redaka i stupaca).

Sljedeći teorem opravdava nam mogućnost komponiranja više transformacija u jednu i time uštedu na brzini renderiranja.

Propozicija 3.1.1. *Kompozicija linearnih operatora je linearni operator.*

Dokaz 3.1.1. *Neka su* $f : \mathbf{V} \to \mathbf{W}$ *i* $g : \mathbf{U} \to \mathbf{V}$ *linearni operatori i* $h = f \circ g$. *Tada za proizvoljne* $x, y \in \mathbf{V}$ *i* $\alpha, \beta \in \mathbf{F}$ *vrijedi:*

$h(\alpha x + \beta y) = (f \circ g)(\alpha x + \beta y) = f[g(\alpha x + \beta y)] = f[\alpha g(x) + \beta g(y)] = \alpha f[g(x)] + \beta f[g(x)] = \alpha(f \circ g)(x) + \beta(f \circ g)(y) = \alpha h(x) + \beta h(y)$.

Stoga, operator h je linearan, pa je kompozicija linearnih operatora opet linearni operator.□

Često se javlja potreba za vraćanjem iz jednog prostora u drugi (iz prostora kamere u prostor svijeta, i iz prostora svijeta u prostor modela), pa da bismo to mogli učiniti, potrebna nam je sljedeća definicija.

Definicija 3.1.8. *Za matricu* $\mathbf{A} \in \mathbf{M}_n$ *kažemo da je* ***regularna*** *(invertibilna) matrica ako postoji matrica* $\mathbf{B} \in \mathbf{M}_n$ *takva da je* $\mathbf{AB} = \mathbf{BA} = \mathbf{I}$.

Ovdje $\mathbf{M}_n$ *označava skup kvadratnih matrica dimenzija* $n \times n$ $(n \in \mathbb{N})$, *a* $\mathbf{I}$ *označava jediničnu matricu istih dimenzija.[2]*

Sljedeće dvije definicije daju nam brze i efikasne načine pronalaska inverza određenih tipova matrica.

Definicija 3.1.9. ***Transponirana*** *matrica matrice* $\mathbf{A} = (a_{ij})$, $\mathbf{A} \in \mathbf{M}_{m,n}$, *je matrica* $\mathbf{B} = (b_{ji}), \in \mathbf{M}_{n,m}$ *za koju je* $b_{ij} = a_{ji}, \forall i \in \{1, ..., m\}, j \in \{1, ..., n\}$, *gdje su n i m prirodni brojevi. Matricu* $\mathbf{B}$ *označavamo s* $\mathbf{A}^\mathsf{T}$.

Definicija 3.1.10. *Matrica* $\mathbf{A} \in \mathbf{M}_n$ *je* ***ortogonalna*** *ako vrijedi* $\mathbf{AA}^\mathsf{T} = \mathbf{A}^\mathsf{T}\mathbf{A} = \mathbf{I}$, *tj. ako je njen inverz jednak njenoj transponiranoj matrici.*

Ukoliko znamo da su matrice s kojima radimo ortogonalne, tada umjesto računanja njihovih inverza, što može biti dugotrajan proces, možemo uzeti njihove transponirane matrice.

Definicija 3.1.11. *Neka su* $\mathbf{V}$ *i* $\mathbf{V}'$ *vektorski prostori and istim poljem* $\mathbf{F}$*, a* $\mathbf{A}$ *i* $\mathbf{A}'$ *afini prostori nad* $\mathbf{V}$ *i* $\mathbf{V}'$*, redom. Preslikavanje* f_a *je* ***afino preslikavanje*** *ako postoji točka* $O \in \mathbf{A}$ *i linearno preslikavanje* $f : \mathbf{V} \to \mathbf{V}'$ *tako da vrijedi* $(\forall T \in \mathbf{A})\, f(\overrightarrow{OT}) = \overrightarrow{f_a(O)f_a(T)}$*.[4]*

Propozicija 3.1.2. *Afino preslikavanje ne ovisi o izboru točke* O*.*

Dokaz. Neka je O' točka različita od O. Tada vrijedi:
$\overrightarrow{f_a(O')f_a(T)} = \overrightarrow{f_a(O')f_a(O)} + \overrightarrow{f_a(O)f_a(T)} = -\overrightarrow{f_a(O)f_a(O')} + \overrightarrow{f_a(O)f_a(T)} = -f(\overrightarrow{OO'}) + f(\overrightarrow{OT}) = f(\overrightarrow{OT} - \overrightarrow{OO'}) = f(\overrightarrow{O'T})$.□

Propozicija 3.1.3. *Svaki linearni operator je afino preslikavanje.*

Dokaz. Neka su $\mathbf{V}$ i $\mathbf{V}'$ vektorski prostori nad poljem $\mathbf{F}$ i $f : \mathbf{V} \to \mathbf{V}'$. Po (3.1), svaki vektorski prostor je afini prostor nad samim sobom. Tada je f afino preslikavanje s pridruženim linearnim operatorom f jer vrijedi:
$\overrightarrow{f(x_1)f(x_2)} = f(x_2) - f(x_1) = f(x_2 - x_1) = f(\overrightarrow{x_1x_2}), \forall x_1, x_2 \in \mathbf{V}$.□[5]

Propozicija 3.1.4. *Neka su* $\mathbf{A}$*,* $\mathbf{A}'$ *i* $\mathbf{A}''$ *afini prostori nad vektorskim prostorima* $\mathbf{V}$*,* $\mathbf{V}'$ *i* $\mathbf{V}''$*, redom. Neka su* $f_a : \mathbf{A} \to \mathbf{A}'$ *i* $g_a : \mathbf{A}' \to \mathbf{A}''$ *afina preslikavanja. Tada je kompozicija* $g_a \circ f_a : \mathbf{A} \to \mathbf{A}''$ *također afino preslikavanje i* $g \circ f$ *je njemu pridružen linearni operator.*

Dokaz. Sa

$$h(\overrightarrow{XY}) = \overrightarrow{(g_a \circ f_a(X))(g_a \circ f_a(Y))}$$

za sve $\overrightarrow{XY} \in \mathbf{V}$ definiran je operator $h : \mathbf{V} \to \mathbf{V}''$.
Za sve $X, Y \in \mathbf{A}$ vrijedi

$$h(\overrightarrow{XY}) = \overrightarrow{g_a(f_a(X))g_a(f_a(Y))} = g(\overrightarrow{f_a(X)f_a(Y)}) = g(f(\overrightarrow{XY})) = (g \circ f)(\overrightarrow{XY}),$$

pa je $h = g \circ f$. Stoga je h, kao kompozicija linearnih operatora, linearni operator. Preslikavanje $g_a \circ f_a$ je tada prema (3.2) afino preslikavanje s pridruženim operatorom h, odnosno $g \circ f$.□[5]

Ako želimo transformirati objekt na zaslonu, na svaki od vrhova tog predmeta primijenit ćemo jednu od željenih afinih transformacija, odnosno koordinate točke T pomnožit ćemo matricom afine transformacije.

3.1.2 Homogene koordinate

Kada želimo transformirati predmet u prostoru korištenjem afinih transformacija, na sve njegove točke djelujemo istim transformacijama. Kako bismo skratili vrijeme i broj potrebnih izračuna i operacija računala, asocijativnost množenja matrica nam omogućava da množenjem svih afinih transformacija niz množenja svedemo na jedno, umjesto da na svaku točku djelujemo redom svakom matricom. Ovdje nam činjenica da je translacija operacija vektorskog zbrajanja onemogućava uštedu vremena. Kako bi riješili ovaj problem, umjesto da koristimo trodimenzionalne koordinate točke u prostoru, proširimo ih do četverodimenzionalnih tako da za zadnju, dodatnu koordinatu točke postavimo 1. Ukoliko neka od transformacija promijeni posljednju koordinatu točke, normalizirat ćemo cijeli vektor tako da tu koordinatu vratimo na 1. Ovakav matematički objekt opisuje sljedeća definicija.

Definicija 3.1.12. *Definirajmo na skupu točaka prostora* $\mathbf{E}^4$ *relaciju ekvivalencije* $\sim$ *na sljedeći način:*

$$(x_1, x_2, x_3, x_4) \sim (y_1, y_2, y_3, y_4) \iff (\exists \lambda \in \mathbb{R} - \{0\})\,(\forall i \in 1, 2, 3, 4)\, x_i = \lambda y_i.$$

Klasu ekvivalencije (x_1, x_2, x_3, x_4) *nazivamo skupom* ***homogenih koordinata*** *vektora predstavnika i označavamo s* $[x_1, x_2, x_3, x_4]$*.[4]*

Bitno je naglasiti da ćemo posljednju koordinatu pozicije vrha normalizirati na 1, dok ćemo posljednju koordinatu vektora normale vrha (čiju ćemo ulogu kasnije opisati) normalizirati na 0. Razlog tome je to što želimo razlikovati poziciju i vektor normale vrha i to što na vektore normale ne djelujemo istim transformacijama s kojima djelujemo na pozicije.

3.2 Računalni objekti u renderiranju

S prethodnim definicijama sada imamo potreban alat za opisivanje čvrstih tijela koje želimo renderirati. Svaki trodimenzionalan predmet sastoji se od mreže trokuta, i svaki trokut od tri vrha, stoga je svaki predmet opisan pomoću niza trodimenzionalnih točaka, tj. svaki predmet je smješten u $\mathbf{E^3}$, trodimenzionalni realni euklidski prostor. Međutim, same koordinate vrhova ne pružaju nam dovoljno podataka za realan prikaz slike pa svakom vrhu pridružujemo i trodimenzionalni vektor normale. Taj vektor normale koristimo

na sljedeći način: on nam daje tangencijalnu ravninu na predmet u tom vrhu. U vidu diferencijalne geometrije, pridruživanje vektora normale vrhu predmeta nije moguće pošto je to točka gdje ploha predmeta nije diferencijabilna, ali mrežu trokuta ćemo promatrati kao apstrakciju stvarne topologije predmeta. Pojedinačni vektor normale koristimo zajedno s dva preostala vektora normale trokuta, pa korištenjem interpolacije možemo doći do novog vektora normale u bilo kojoj točci unutrašnjosti i ruba trokuta. Tako bismo npr. sferni trokut mogli opisati korištenjem samo tri točke i tri njihova odgovarajuća vektora normale (umjesto aproksimiranja plohe sfere korištenjem velikog broja točaka).

3.2.1 Sjenčanje

Ovaj način apstrahiranja plohe predmeta opravdan je tehnikom generiranja grafike koja se zove sjenčanje (engl. *shading*). Naš konačan cilj je bojanje piksela od kojih je sastavljena slika, a to postižemo algoritmima i jednadžbama koje modeliraju refleksiju svjetlosti od površine predmeta. Vektori normala nam tada omogućuju izračunati kut pod kojim zraka svjetlosti pogađa površinu predmeta, te s obzirom na odabranu jednadžbu sjenčanja, odrediti boju piksela koji odgovara točki u kojoj je predmet pogođen.

Poglavlje 4

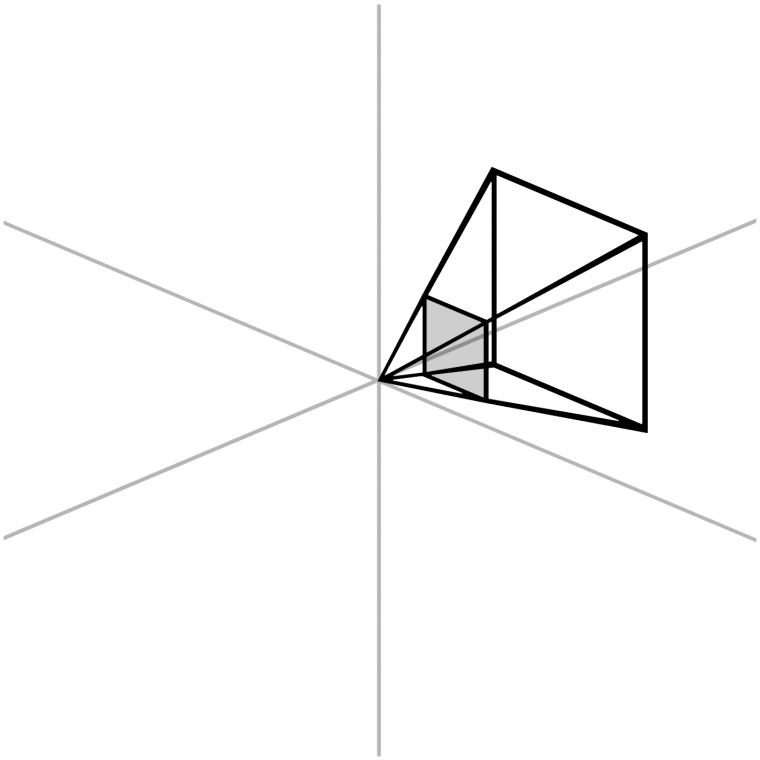
Rasterizacija

Uz poznavanje osnovnih pojmova (kamera, ravnina zaslona i objekti u sceni) sada možemo objasniti principe rasterizacije.

Pri rasterizaciji, prvo ćemo odrediti sve objekte koji se nalaze u sceni koju želimo prikazati, pa ćemo te objekte projicirati na zaslon. Kod ovog koraka je potrebno uvesti ograničenu površinu zaslona - ne želimo prikazivati objekte koji se nalaze iza kamere. Ograničimo li površinu zaslona na pravokutnik, sada kroz vrhove tog pravokutnika i kroz ishodište možemo definirati lijevu, desnu, gornju i donju ravninu. Istovremeno bismo trebali ograničiti udaljenost do koje ćemo renderirati objekte (nije potrebno renderirati objekte koji se nalaze "daleko") što možemo učiniti tako da ispred ravnine zaslona (engl. *near plane*) stavimo još jednu, daleku ravninu (engl. *far plane*). Geometrijsko tijelo dobiveno s tih šest ravnina je krnja četverostrana piramida (engl. *frustum*).

Ukoliko se predmet ili dio predmeta nalazi unutar frustuma, projicirati ćemo odgovarajuće trokute na ravninu zaslona. Nakon određivanja vidljivih predmeta, podaci o vrhovima mreže trokuta predmeta procesuiraju se pojedinačno. U ovom koraku možemo primijeniti jednadžbe sjenčanja. One će tada koristiti samo podatke vezane uz same vrhove, ne uz točke unutar trokuta, te će mijenjati te iste podatke (npr. možemo koristiti vektore normale tih vrhova kako bismo odredili mjeru u kojoj svjetlost doprinosi vidljivosti i boji vrhova). Nakon procesuiranja vrhova, slijedi procesuiranje trokuta. U ovom koraku također možemo koristiti jednadžbe sjenčanja, međutim sada ćemo promatrati i unutrašnje i rubne točke trokuta te procesuirati svaku od njih zasebno. Njihove vrijednosti vektora normala dobivamo prethodno spomenutim metodama interpolacije. [1]

[1] Ovdje možemo koristiti sustav baricentričnih koordinata svakog pojedinačnog trokuta.



Slika 4.1: Frustum

Poglavlje 5

Afine transformacije

5.1 Prostori modela, svijeta i kamere

U uobičajenom modelu trodimenzionalnog renderiranja, gotovo nikada nećemo konstruirati cijelu scenu sa svim objektima u istom koordinatnom sustavu. Grafički dizajneri će u računalnim programima prvo u zasebnom koordinatnom sustavu dizajnirati modele predmeta - taj ćemo ortonormirani koordinatni sustav nazivati prostorom modela. Ortonormirani koordinatni sustav u koji ćemo smjestit sve predmete u našoj sceni nazvat ćemo prostorom svijeta ili prostorom scene. Svaki pojedinačni predmet stavit ćemo u ishodište tog koordinatnog sustava, prilagoditi njihovu veličinu, po želji rotirati, te zatim staviti na mjesto gdje želimo. Između ova dva koraka prirodno se javlja potreba za nizom transformacija koje izvršavamo nad mrežom trokuta modela predmeta. Te transformacije izražavamo matricom koju nazivamo matricom modela (engl. *model matrix*) i to je najčešće prva matrica kojom ćemo djelovati na predmet.

Konačan prostor koji imenujemo je prostor kamere. Naime, kameru i njen odgovarajući frustum također promatramo kao predmet u prostoru svijeta. Predmetu kamere pridružena su i tri vektora: vektor usmjeren prema naprijed, prema gore i prema desno, i oni tvore ortonormirani koordinatni sustav. Zbog jednostavnosti koju nam to pruža kod projiciranja, sve predmete u sceni želimo smjestiti u koordinatni sustav gdje je "oko" kamere u središtu, a koordinatne osi odgovaraju trima vektorima pridruženih predmetu kamere. Ovaj proces također prirodno motivira potrebu za niz transformacija nad mrežama trokuta predmeta, a rezultirajuću matricu nazivamo matricom pogleda (engl. *view matrix*).

Konačna transformacija koju ćemo vršiti nad mrežama trokuta predmeta je projekcija, čiju odgovarajuću matricu nazivamo matricom projekcije (engl. *projection matrix*). Kako je redoslijed množenja matrica bitan, konačan niz operacija glasi: vrhove prvo množimo matricom modela, zatim matricom pogleda i zatim matricom projekcije.

5.2 Translacija

Translacija je transformacija koja svakoj komponenti točke daje određeni pomak. U računalnoj grafici ona prenosi objekt na novu poziciju na ekranu. Neka je $T \in \mathbf{E}^3$ točka s koordinatama $T(x_0, y_0, z_0)$. Tada njoj jednoznačno pridružujemo točku T' danu s koordinatama $T'(x_0, y_0, z_0, 1)$. Neka je sada $\vec{tr} \in \mathbb{R}^3$ vektor translacije s koordinatama $\vec{tr} = (tr_x, tr_y, tr_z)$. Njemu jednoznačno pridružujemo matricu $\mathbf{T}(\vec{tr})$:

$$\mathbf{T}(\vec{tr}) = \begin{bmatrix} 1 & 0 & 0 & tr_x \\ 0 & 1 & 0 & tr_y \\ 0 & 0 & 1 & tr_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

. Pomnožimo li sada radijus-vektor točke T' matricom $\mathbf{T}(\vec{tr})$ dobivamo:

$$\begin{bmatrix} 1 & 0 & 0 & tr_x \\ 0 & 1 & 0 & tr_y \\ 0 & 0 & 1 & tr_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{bmatrix} = \begin{bmatrix} x_0 + tr_x \\ y_0 + tr_y \\ z_0 + tr_z \\ 1 \end{bmatrix}.$$

Promotrimo li sada točku u $\mathbf{E}^3$ kojoj je pridružena dobivena točka, vidimo da je to $(x_0 + tr_x, y_0 + tr_y, z_0 + tr_z) = (x_0, y_0, z_0) + (tr_x, tr_y, tr_z) = T + \vec{tr}$. Prikažemo li sada matrice preostalih afinih transformacija kao njihove tipične trodimenzionalne matrice kojima je dodana jedinica na koordinatu $(4, 4)$ i 0 na preostale koordinate u četvrtom retku i četvrtom stupcu, omogućili smo množenje matrica svih afinih transformacija, uključujući i translacije.

5.3 Rotacija

Transformacija rotacije djeluje na vektor ili točku (djeluje na radijus-vektor točke) tako da ju rotira s obzirom na pravac koji prolazi kroz ishodište. Ovdje ćemo redom navesti

matrice rotacije oko x, y, i z-osi. Neka je $T \in \mathbf{E}^3$ točka. Tada su matrice rotacije oko odgovarajućih osi za kut ϕ:

$$\mathbf{R}_x(\phi) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(\phi) & -sin(\phi) & 0 \\ 0 & sin(\phi) & cos(\phi) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{R}_y(\phi) = \begin{bmatrix} cos(\phi) & 0 & sin(\phi) & 0 \\ 0 & 1 & 0 & 0 \\ -sin(\phi) & 0 & cos(\phi) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{R}_z(\phi) = \begin{bmatrix} cos(\phi) & -sin(\phi) & 0 & 0 \\ sin(\phi) & cos(\phi) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Lako je provjeriti da su sve ove matrice ortogonalne.

5.4 Skaliranje

Matrica skaliranja (engl. *scaling matrix*) $\mathbf{S}(\vec{s}), \vec{s} \in \mathbb{R}^3, \vec{s} = (s_x, s_y, s_z)$ skalira objekt faktorima s_x, s_y i s_z redom duž osi x, y i z. Nju koristimo kako bismo objekt povećali ili smanjili: što je $s_i, i \in \{x, y, z\}$ veći, to se objekt više skalira u tom smjeru. Ako je bilo koja od komponenti jednaka 1, ne dolazi do promjene u tom smjeru.

$$\mathbf{S}(\vec{s}) = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Koeficijenti skaliranja mogu biti bilo koji realni brojevi, no u računalnoj grafici posebno kod izrade računalnih igara, koristi se skaliranje pozitivnim brojevima.
Ukoliko vrijedi da je $s_x = s_y = s_z$, kažemo da je skaliranje uniformno ili izotropno. U suprotnom kažemo da je neuniformno ili anizotropno.

Refleksiju možemo promatrati kao specijalan slučaj skaliranja kada je jedna ili su sve tri komponente vektora $\vec{s}$ negativne.
Ako su dvije komponente negativne, dobivamo matricu rotacije.

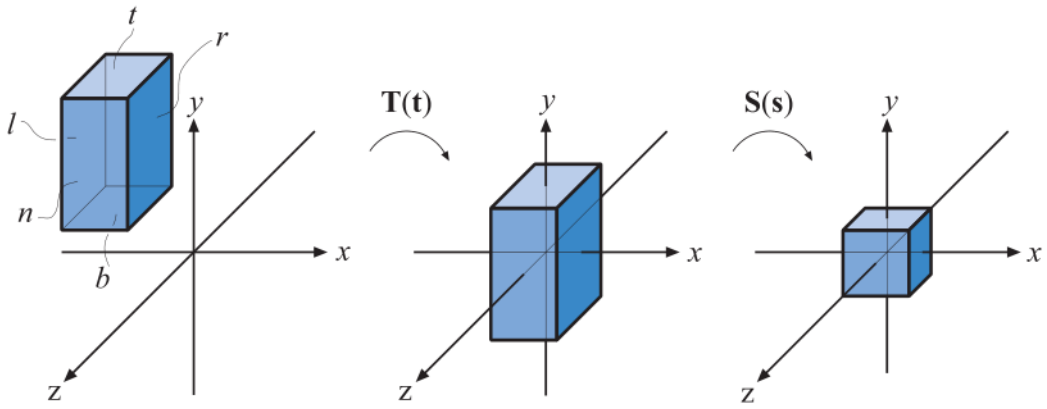
5.5 Smicanje

Posljednja klasa afinih transformacija koju ćemo spomenuti je smicanje (engl. *shear*) i to je jedna od glavnih transformacija u računalnoj grafici. Postoji šest matrica smicanja: $\mathbf{H}_{xy}(\vec{s})$, $\mathbf{H}_{xz}(\vec{s})$, $\mathbf{H}_{yx}(\vec{s})$, $\mathbf{H}_{yz}(\vec{s})$, $\mathbf{H}_{zx}(\vec{s})$ i $\mathbf{H}_{zy}(\vec{s})$. Prvi indeks označava koordinatu koja će biti izmijenjena matricom, dok drugi indeks označava koordinatu u čijem smjeru se vrši smicanje. Primjer matrice $\mathbf{H}_{xz}(s)$:

$$\mathbf{H}_{xz}(\vec{s}) = \begin{bmatrix} 1 & 0 & s & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

5.6 Projekcije

Prije nego možemo prikazati objekte u sceni na zaslonu potrebno ih je projicirati na blisku stranu frustuma. Dvije vrste projekcije koje ćemo spomenuti su perspektivna projekcija i ortografska projekcija. U ovome poglavlju, pretpostavit ćemo da kamera gleda u smjeru negativne z-osi, da je y-os usmjerena prema gore, a x-os prema desno. [1]



Slika 5.1: Ortografska projekcija

5.6.1 Ortografska projekcija

Kod ortografske projekcije, frustum kamere mijenjamo običnim kvadrom. Ortografska projekcija $\mathbf{P}$, dano matricom

$$\mathbf{P} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

ne mijenja x i y koordinate točke i postavlja njenu z koordinatu na 0, odnosno projicira na ravninu z = 0.

Međutim, ova transformacija djeluje na sve predmete u prostoru svijeta, uključujući i one koji se nalaze "iza" kamere, tj. one koje ne bismo željeli vidjeti. Ono što bismo tada željeli je ograničiti vrijednosti x, y, i z koordinata vrhova koje ćemo projicirati na neke intervale. Točnije, z koordinate ćemo ograničiti na vrijednosti između udaljenosti bliske ravnine od ishodišta, što ćemo označiti s n (od engl. *near*), i daleke ravnine od ishodišta, što ćemo označiti s f (od engl. *far*). Primijetimo, kako kamera gleda u negativnom smjeru z-osi, vrijedit će $n > f$. Dodatno ćemo označiti i donju ravninu frustuma s b (od engl. *bottom*), gornju ravninu s t (od engl. *top*), lijevu ravninu s l (od engl. *left*) i desnu ravninu s r (od engl. *right*). Tada matricu ortografske projekcije možemo izraziti koristeći uređenu šestorku (l, r, b, t, n, f). Ovdje trojke (l, b, n) i (r, t, f) označavaju dijagonalno suprotne

[1]Dogovor o orijentaciji kamere ovisi o *APIju* (engl. *Application Programming Interface*) korištenom za komunikaciju s *GPUom* (engl. *Graphics Processing Unit*). U ovome radu koristimo orijentaciju kamere korištenu u *OpenGLu* danu pravilom desne ruke.

vrhove kvadra unutar kojeg se nalaze predmeti koje želimo renderirati.

U *APIevima* se projekcija vrši tako da se (u slučaju ortografske projekcije) kvadar kamere preslika u *AABB* (engl. *Axis-Aligned Bounding Box*). *AABB* je kvadar čiji se normalni vektori stranica poklapaju sa kanonskim vektorima baze koordinatnog sustava. Tu kocku možemo opisati pomoću dvije trojke koordinata dijagonalno suprotnih vrhova, kao što smo prethodno učinili i sa kvadrom kamere. Te trojke *AABBa* ovise o korištenom *APIu*: kod *OpenGLa*, to su $(-1,-1,-1)$ i $(1,1,1)$, dok su kod *DirectXa* to $(-1,-1,0)$ i $(1,1,1)$. Nakon transformacije kvadra kamere u *AABB*, dobivene koordinate vrhova unutar kvadra nazivamo normaliziranim koordinatama uređaja (engl. *normalized device coordinates*). Razlog ovoj transformaciji je brže i točnije određivanje crhova koji se moraju projicirati.

Dvije transformacije koje tada vršimo su translacija kvadra u ishodište koordinatnog sustava, i skaliranje za prilagođavanje veličina. U konačnici, matrica ortografske projekcije kakva se koristi u računalnim sustavima $\mathbf{P_0}$ glasi:

$$\mathbf{P_0} = \mathbf{S}(\vec{s})\,\mathbf{T}(\vec{tr}) = \begin{bmatrix} \frac{2}{r-l} & 0 & 0 & 0 \\ 0 & \frac{2}{t-b} & 0 & 0 \\ 0 & 0 & \frac{2}{f-n} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -\frac{l+r}{2} \\ 0 & 1 & 0 & -\frac{t+b}{2} \\ 0 & 0 & 1 & -\frac{f+n}{2} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \frac{2}{r-l} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & \frac{2}{f-n} & -\frac{f+n}{f-n} \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

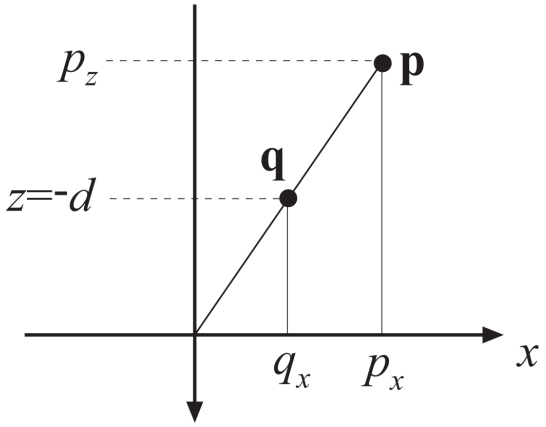
Strogo matematički gledano, ovo nije matrica projekcije (lako je vidjeti da je invertibilna što matrica ortografske projekcije nije). U modernim sustavima, projekcija se vrši na *GPUu* automatski tako da se vrhovi van dobivenog *AABB* odbacuju, pa se preostali vrhovi projiciraju na ravninu koja prolazi točkom $(0,0,-1)$ paralelnu s xy-ravninom (to se sada lako postiže postavljanjem z koordinate vrhova na -1). Rezulat te projekcije je ono što će biti renderirano.

5.6.2 Perspektivna projekcija

Za razliku od ortografske projekcije koja se koristi u slučajevima gdje je potreban točan prikaz udaljenosti i odnosa između predmeta u scena, perspektivna projekcija modelira način na koji vidimo svijet - objekti koji su udaljeni od kamere izgledaju manje.

Izvedimo sada matricu perspektivne projekcije na ravninu $z = -d, d > 0$. Koordinatni sustav orijentiran je kao i prije, dok se kamera nalazi u središtu koordinatnog sustava.

Neka je s $\vec{p} = (p_x, p_y, p_z)$ dana točka koju ćemo projicirati, te neka je s $\vec{q} = (q_x, q_y, -d)$ označena rezultantna točka.



Slika 5.2: Perspektivna projekcija

Tada sa slike 5.2 možemo vidjeti slične trokute $\mathbf{Oqq_x}$ i $\mathbf{Opp_x}$ (gdje je točka $\mathbf{O}$ ishodište koordinatnog sustava) i dobiti jednadžbe

$$\frac{q_x}{p_x} = \frac{-d}{p_z} \Longleftrightarrow q_x = -d \cdot \frac{p_x}{p_z}.$$

Izrazi za ostale koordinate od $\vec{q}$ glase:

$$q_y = -d \cdot \frac{p_y}{p_z}$$

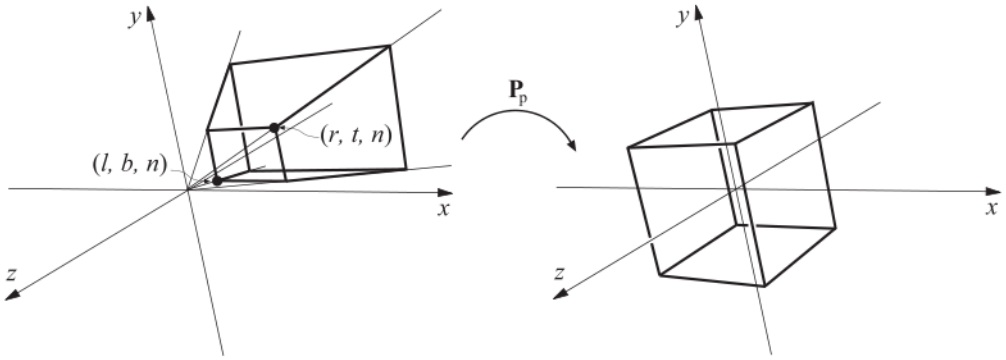
$$q_z = -d.$$

Konačno, dobivamo izraz matricu perspektivne projekcije $\mathbf{P}_p$:

$$\mathbf{P}_p = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -\frac{1}{d} & 0 \end{bmatrix}.$$

Ova matrica će djelovati na dodatnu, četvrtu koordinatu pozicije vrhova, i vjerojatno ju promijeniti iz dogovorene jedinice, ali kako koristimo homogene koordinate to nam ne predstavlja problem - jednostavno podijelimo sve koordinate s tom novom vrijednosti:

$$\begin{bmatrix} T_x \\ T_y \\ T_z \\ T_w \end{bmatrix} \Rightarrow \begin{bmatrix} \frac{T_x}{T_w} \\ \frac{T_y}{T_w} \\ \frac{T_z}{T_w} \\ 1 \end{bmatrix}$$



Slika 5.3: Transformacija frustuma perspektivne projekcije

U praksi, kako smo kod ortografske projekcije transformirali kvadar u standarizirani $AABB$, ovdje ćemo transformirati prethodno spomenuti frustum, te ćemo opet koristiti oznake šestorke (l, r, b, t, n, f). Praktična matrica perspektivne projekcije $\mathbf{P}_p$, kakva se koristi u računalnim sustavima, tada glasi:

$$\mathbf{P}_p = \begin{bmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & -\frac{f+n}{f-n} & -\frac{2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{bmatrix}.$$

Nakon primjene ove transformacije na vrh, četvrta koordinata pozicije bit će promijenjena pa ćemo opet morati podijeliti sve koordinate s novom vrijednosti.

Poglavlje 6

Zaključak

Definiranje matematičkih struktura i povezivanje računalnih objekata s njima omogućuje nam korištenje naprednih alata (transformacija) i s time ostvarivanje naprednijih grafičkih efekata. Od velike koristi je sposobnost uočavanja kada nam te strukture omogućuju uvelike pojednostaviti i ubrzati cijeli proces renderiranja, kao što nam to omogućuju vektori normala vrhova. Umjesto da koristimo veliki broj vrhova za opisivanje površine predmeta, to možemo učiniti s malim brojem. Ključno je i proširenje trodimenzionalnih prostora na četverodimenzionalne: ono nam omogućuje veliku uštedu u brzini izračuna složenih matrica transformacija. Uvođenje dodatne koordinate također nam omogućuje i lakši rad sa matricama projekcija.

Veliki dio matematičkih procesa prilikom renderiranja, tijekom godina razvoja i istraživanja, je standardiziran i implementiran u samom hardveru tako da sam programer često neće imati doticaja sa dijelovima sustava. Međutim, poznavanje korištenih alata je od ključne važnosti jer tada možemo donijeti odluke koje će ubrzati i poboljšati naše sustave te omogućiti daljnje istraživanje kompleksnijih struktura i alata.

Literatura

[1] Akenine-Möller, Tomas - Haines, Eric - Hoffman, Naty: *Real-Time Rendering*, 3. izdanje, CRC Press Taylor & Francis Group, 2008.

[2] Kurepa, Svetozar: *Uvod u linearnu algebru : vektori, matrice, grupe*, 4. izdanje, Školska knjiga, Zagreb, 1985.

[3] Horvatić, Krešimir: *Linearna algebra*, 9. izdanje, Golden Marketing - Tehnička knjiga, Zagreb, 2004.

[4] Audin, Michèle: *Geometry*, 1. izdanje, Springer, Heidelberg, 2002.

[5] Polonijo, M. - Crnković, D. - Ban Kirigin, T. - Bombardelli, M. - Franušić, Z. - Sušanj, R.: *Euklidski prostori*, Sveučilište u Zagrebu, PMF-Matematiški odjel, Sveučilište u Rijeci, Filozofski fakultet u Rijeci