Fakulteta za računalništvo in informatiko
Univerza v Ljubljani

Premikanje po okolju na podlagi vidnih informacij

Seminarska naloga

Jernej Vičič

Ljubljana, 2001

Kazalo

Premikanje po okolju na podlagi vidnih informacij 1
Kazalo 2
  Level Of Detail - LOD 3
  Uvod 3
  Pristop z napovedovanjem 4
  Računanje dodane vrednosti 4
  Izboljšava osnovne metode 5
  Dodana vrednost glede na pogled 5
  Dodana vrednost gruč 6
  Ogrodje sistema 6
  Objekti 6
  Izbira nastopača 7
  Opis nove metode 7
    Gradnja modela 8
    Prehod drevesa 9
Literatura 10

Level Of Detail - LOD

Osnovna predstavitev nekega objekta je njegova najnatančnejša predstavitev v določenem sistemu. Včasih je takšna predstavitev predraga za prikaz (računalnik potrebuje preveč časa za izris tega objekta). Predvsem je hitrost pomembna predvsem pri sistemih v realnem času kot so prikazi pokrajine, določenega objekta,... in najbolj poznan problem prehoda prek scene. Problem prehoda prek scene je sestavljen iz kompleksne scene sestavljene iz množice objektov ter uporabnika, ki se premika po sceni. Uporabnik določa pogled in hitrost premikanja, glede na ti dve komponenti pa sistem izračuna pogled na posamezne objekte oziroma na skupine. Za hitrejši izris objektov se uporablja različne tehnike, najpomembnejša pa je določanje različnih natančnosti predstavitve objekta glede na oddaljenost, pogled, pomembnost objekta na sliki ter še dodatnih parametrov, ki so lastni posameznim implementacijam. Predstavitve različnih natančnosti imenujemo LODi (Level Of Detail), v nadaljevanju bomo uporabili kar okrajšavo. Pred pregledom metod predstavimo še nov izraz nastopač (impostor), ki obsega osnovno idejo LOD ter dodatne lastnosti kot so cena, kvaliteta prikaza in drugo.

Uvod

To delo predstavlja novo rešitev problema prehoda scene. Gledalec se premika skozi statično sceno z veliko in konstantno hitrostjo.
Tradicionalne rešitve uporabljajo strojni cevovod in skušajo minimizirati število poligonov poslanih sistemu. Minimizacija je dosežena z obrezovanjem scene in geometrijskim osiromašenjem posameznih objektov (nivojev natančnosti – levels of detail LOD). Delo temelji na predpostavki, da so tudi pogledi, sestavljeni iz najbolj osiromašenih objektov, preveč zahtevni za tekoči prikaz. Osnovna ideja je v uporabi poenostavljenih predstavitev namesto resničnih objektov, te predstavitve bomo poimenujmo z nastopači. Nastopač je umetni objekt, ki ga lahko hitreje narišemo, vsebuje pa še vedno vse osnovne vizualne lastnosti. Tradicionalni LODi so posebna različica nastopačev. Ključno opravilo je določiti kateri nadomestni objekt naj se generira, tako da bo slika čimbolj resnična, ob upoštevanju časovne meje, ki jo določi uporabnik. Sistem, ki je v bistvu razširitev pristopa z napovedovanjem, si bomo natančneje ogledali v nadaljevanju, osnovne lastnosti pa predstavimo:

Celotna podatkovna baza je predstavljena kot drevo z nastopači (tudi pravimi LOD) in grozdi, ki jih lahko narišemo. To je osnovna posplošitev koncepta LOD.
Sistem uporablja strojne grafične podenote za avtomatično generiranje drevesa, generacijo nastopačev in izračun cen (časa za prokaz) posameznih objektov. Prav tako vnaprej izračuna vse statične izračune sistema.

Pristop z napovedovanjem

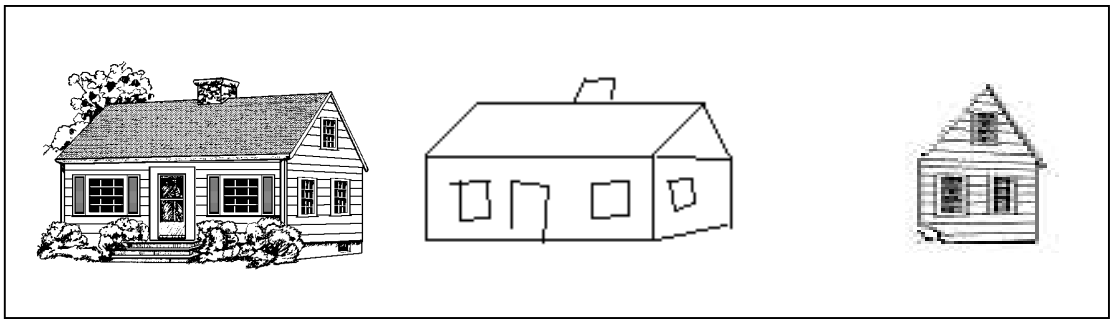
Ta pristop predvideva, da lahko določimo ceno (čas) za generiranje nekega objekta. Cena objektov je določena z empiričnimi podatki o strojni opremi, ki gosti naš sistem. Ti podatki se uporabijo kot parametri v formulah, ki opisujejo ceno objektov. Uporabni sistemi prehodov scen iščejo najboljše razmerje kvalitete izdelane scene v omejenem časovnem obdobju. V ta namen uporabljajo hevristike za določitev koliko določena natančnost generiranja nekega objekta prispeva k izgledu celotne scene oziroma trenutnega pogleda na sceno. Hevristika upošteva velikost objekta pri določenem pogledu, fokus, relativna hitrost glede na žarišče pogleda, semantiko, natančnost LOD in histezo v obziru na preklapljanje med različnimi LODi. Inkrementalni optimizacijski algoritem izbira objekte za generiranje glede na količnik prispevek/cena, dokler vsota cen ne preseže izbrane časovne meje. Torej obstaja možnost, da objekti z nižjim količnikom ne bodo nikoli izbrani. Na scenah, ki imajo veliko objektov se lahko zgodi, da algoritem zavrže mnoge objekte in nastanejo velika prazna polja, kar je precej nezaželjen efekt. Za zmanjšanje števila objektov, ki jih je potrebno generirati se uporablja tudi iskanje zakritih objektov, to iskanje se izvrši že v fazi predprocesiranja scene. Tako lahko enostavno definiramo objekte, ki so popolnoma ali delno zakriti z bližjimi objekti. Metoda lepo deluje pri natančno in ostro definiranih objektih kot so hiše z vrati in okni, ki predstavljajo nova vidna polja za objekte v ozadju, ni pa uporabna za objekte v naravi, saj je slabo definirana pokritost.

Računanje dodane vrednosti

Sistemi za vizualno navigacijo uporabljajo različne predstavitve objektov (LOD) za izboljšanje hitrosti generiranja scene. Vsak LOD doda določen del k celotni kvaliteti reprodukcije generirane scene, ki je določena s hevristiko. Pri računanju te vrednosti smo soočeni z dvema težavama: določanje vrednosti objekta glede na pogled oziroma kot pogleda na objekt ter določanje vrednosti v odvisnosti od grupiranja, okolja v katerem objekt nastopa (semantika).

Izboljšava osnovne metode

Poleg osnovnih LOD lahko objektu določimo še vse druge predstavitve, ki se lahko narišejo in so podobne objektu. Na slika1 si lahko ogledamo hišo v popolni predstavitvi, LOD predstavitev, brez tekstur ter tretjo predstavitev, ki je odvisna od pogleda na objekt in je primerna samo za poglede, ki so pravokotni na pročelje.



Slika1: Osnovna predstavitev objekta z največjo natančnostjo (levo)
od pogleda neotdvisen LOD (sredina)
LOD, ki je odvisen od pogleda (desno)

Dodana vrednost objekta je skupek lastnosti, ki jih lahko razdelimo na dve množici: lastnosti objekta in lastnosti določene predstavitve objekta.
V prvo skupino sodijo lastnosti kot je relativna velikost objekta glede na ostale ter na celotno sceno (seveda je odvisna od oddaljenosti), večji objekti so pomembnejši od manjših, oddaljenost objekta od premice pogleda. Dodana vrednost je utežena vsota vrednosti teh lastnosti, uteži so empirično določene in se lahko spreminjajo.
Druga skupina pa je natančnost predstavitve pogleda glede na celoten objekt, koliko je predstavitev podobna resničnemu objektu oziroma določenemu pogledu na resničen objekt.
Razen semantike so vse lastnosti prve skupine določljive šele med izvajanjem, vrednosti lastnosti druge skupine pa so statične in jih lahko vnaprej izračunamo.

Dodana vrednost glede na pogled

Poglejmo ponovno slika1, najvišjo oceno dodatne vrednosti ima gotovo prva slika, ki pa je verjetno preveč zahtevna in časovno potratna za generiranje. Dodana vrednost ostalih slik pa je odvisna od kota pogleda glede na objekt ter od oddaljenosti (določeni koti zelo dvignejo ceno tretjemu pogledu, oddaljeni pogledi pa drugemu).
Način dodajanja odvisnosti pogleda v hevristiko je merjenj natančnosti predstavitev za vse močne poglede.Seveda se omejimo na diskretne porazdelitve prostora in tem

ustrezne poglede. Izdelamo tabelo parov predstavitev/pogled z vrednostjo približka k idealnimi predstavitvi objekta, to tabelo uporabljamo pri sprehodu skozi sceno. Določitev kakovosti določene predstavitve scene bi morala temeljiti na perceptualni primerjavi originalne slike ter generirane slike. To pa je nemogoče, ker se kakovost določa avtomatično, zatekli se bomo k algoritmom, ki simulirajo človeško percepcijo. Enostaven algoritem, ki služi samo kot primer je pretvorba slike v sivinsko (akromatska percepcija je najpomembnejša), uporabljen je filter za iskanje robov, ki jih človeško oko najprej zapazi, nato pa sledi zameglitev slike. Tako obdelani sliki primerjamo z istoležnimi piksli.

Dodana vrednost gruč

Ljudje ne dojemamo objektov neke scene kot posamezne elemente ampak kot dele celotne scene ali vsaj podcelot. Tako tudi vse do sedaj opisane in uporabljane tehnike za določanje cene posameznih objektov ne odražajo pravega človekovega pogleda na neko sceno. Samo vsota posameznih cen objektov ne odraža prave vrednosti določenih podcelot, določanje vrednosti mora biti odvisno od objektove okolice in od načina vključevanja v to okolico. Tukaj uvedemo pojem gruče objektov, ki bolj natančno določa dodano vrednost.

Ogrodje sistema

Velikokrat so konvencionalni LODi, predstavljeni v članku, predragi, preveč časovno požrešni ali pa jih sploh ni. Ker pa so LODi samo predstavitve objekta, ki se hitreje izrišejo, ni obvezno, da so to samo osnovni objekti z manj grafičnimi primitivami, lahko so drugačne predstavitve objekta, na primer drugačna izbira senčenja. Tako lahko objektu določimo čimveč predstavitev, vsako za določeno množico pogledov in določen namen ter z določeno vrednostjo in ceno generiranja.

Objekti

Kot objekt pojmujemo nek model, ki ima dobro definiran pomen kot na primer avto aki hiša. Takšnemu objektu določimo množico predstavitev, ki jih lahko narišemo. Predstavitev, ki jo lahko narišemo je lahko primitiva, ki jo generira strojna oprema, skupek takšnih primitiv ali pa celo nastopač. Nastopač pomeni določeno predstavitev objekta in je lahko odvisen od pogleda ali pa neodvisen.

Poglejmo si nekaj primerov razdeljenih v dve skupini glede na odvisnost od pogleda. Prva skupina:

- Tekstura, ki jo prilepimo na steno zgradbe
- Plošča, ki je vedno obrnjena k gledalcu. S pomočjo takšne predstavitve lahko enostavno prikažemo rotacijsko simetrične objekte kot so drevesa, premiki samo menjajo predstavitve glede na pogled

Druga skupina:

- Tradicionalni LODi (geometrijsko oskubljene različice dejanskega objekta)
- Škatle, ki imajo stranice povrečno enakih barv in obli kot škatle, ki obkrožajo dejanski objekt
- Škatle s teksturami, ki so nalepljene na stranice. Primerne za velike škatlaste oblike kot so stolpnice

Izbira nastopača

Obstajajo slučaji, ko lahko programu priporočimo izbiro nastopača. Če je zahtevana velikost objekta le nekaj pikslov (vnaprej definirano število) lahko uporabimo škatlo s povprečno barvo, večji objekt pa narišemo v največjem merilu. Pri večjem številu LODov lahko povečamo število mej.

Objekte kvadrastih oblik lahko predstavimo s testurami lepljenimi na kvadre, - simetrične objekte pa z napisnimi ploščami (oblepljenimi s teksturami). Izbira tekstur je odvisna od uporabnikovega pogleda.
Oglejmo si uporabo na primeru: objekt v obliki kvadra razdelimo po diagonalah. Vsaka diagonala razdeli svet na dva dela, obe diagonali pa na štiri regije. Vsaka stranica objekta je predstavljena z lastno teksturo, ki jo nalepimo na stranico objekta. Ta del lahko opravimo že v predobdelavi scene. Ko se uporabnik sprehaja se z regijami spreminjajo teksture (vsaka regija ima lastno teksturo).

Opis nove metode

Objekte scene, ki jo obdelujemo, pripravljamo za posredovanje uporabniku, definiramo z meta – objekti. Le ti so sestavljeni iz ene ali več predstavitev osnovnega objekta. Te predstavitve pa vsebujejo poleg osnovne ga grafičnega opisa, ki ga lahko naša grafična strojna oprema nariše, tudi čas potreben za izris določene predstavitve ter vrednost, ki jo celotni sceni prinaša določena predstavitev. Torej pri izrisu nekega objekta izbiramo med nekaj različnimi predstavitvami, iščemo predstavitev, ki v določenem pogledu prinese čimveč k večji kakovosti scene, samo izrisovanje pa še ne preseže časovnih meja. Tako predstavitev celotne scene bomo poimenovali model.

Hierarhija modela je drevo. Vozlišča drevesa so meta – objekti, vsako vozlišče predstavlja skupno predstavitev vseh svojih otrok. Tako je korensko vozlišče najbolj grob prikaz celotne scene, listi drevesa pa najnatančnejše in seveda tudi najdražje

(časovno potratne) predstavitve posameznih objektov (na primer hiše z vsemi okni in rožami na oknih).

Tako podano drevo za predstavitev neke scene nam problem sprehoda po sceni prevede na prehod drevesa (tree traversal). Izberemo niz vozlišč drevesa, ki omogočajo uporabniku dovolj dober pogled na prizorišče ob upoštevanju naslednjih omejitev:

- Vsota cen (čas generiranja) vseh izbranih vozlišč mora biti manjši kot definiran največji čas (ponavadi ga definira uporabnik)
- Samo eno vozlišče je lahko izbrano na vsaki poti od korenskega vozlišča do lista (vsako vozlišče že vsebuje predstavitev vseh nižje ležečih vozlišč v hierarhiji)

Tako opisan algoritem je posebna različica 1/0 nahrbtnika in spada med NP polne probleme. Nizi kandidatov iz vsake poti so elementi, ki jih dodajamo v nahrbtnik. Velikost nahrbtnika je časovni limit generiranja celotne scene, cena vsakega elementa pa njegova cena definirana v našem problemu (scena grafičnega generiranja), profit pa je prav tako naš profit oziroma ocena koliko določena predstavitev prinese k natančnosti prikaza celotne scene.

Ogledali si bomo algoritma za postavitev hierarhičnega modela scene na prikazan način ter hevristični algoritem prehoda drevesa hierarhičnega modela.

Gradnja modela

Drevo sestavljamo iz vozlišč, ki vsebujejo najcenejše predstavitve svojih otrok. Drevo je različica 8 – drevesa, ki predstavlja hierarhijo z omejevanjem volumna, tako lahko oskubimo objekte. Gradimo od listov k korenu, tako lahko nadziramo izdelovanje predstvitev in njihovo grupiranje v gruče. Kriterij za združevanje objektov je enostavno razdalja, objekti oddaljeni za določen radij sodijo k isti gruči.
Glede na uporabnikovo nastavitev določimo število razdelitev po x, y in z oseh in tako določimo globino drevesa. Začnemo z listo otrok, ki vsebuje vse konceptualne objekte modela z omejitvenimi škatlami. Ta lista predstavlja liste drevesa, uporabimo jo za gradnjo višjih nivojev. Za vsako celico ugotovimo kateri objekti so popolnoma v njej, izračunamo omejujočo škatlo in če je še nimamo v listi otrok, generiramo nastopača.
Gruče tvorimo samo v škatlah, ki imajo več kot en celoten objekt, če so objekti na meji ali pa je samo en objekt v škatli, grupiramo na višjem nivoju. Tvorimo teksture, ki predstavljajo nize objektov – gruče. Ko so generirane impostor predstavitve postavimo kazalce s celice na njene otroke (te prikazuje nova predstavitev) in otroke umaknemo s seznama otrok, na seznam pa dodamo to novo celico.

Prehod drevesa

Osnovni problem prehoda drevesa je NP poln, potrebujemo hitrejši algoritem, ki bo v določenem času našel dovolj dobro rešitev (potrebujemo nek hevristični algoritem, ki bo veliko hitreje našel še dovolj dobro rešitev). Algoritem se bo sprehodil od korena proti listom, izbiral elemente za generanje tes sešteval čas potreben za generacijo izbranih predstavitev objektov. Problem takšnega algoritma je, da upošteva dodano vrednost objektov in ne gruč, objekti pa so šele na najnižjem nivoju drevesa, v listih. Na višjih nivojih še ne vemo kateremu objektu pripada največja dodana vrednost, iskanje najboljših vozlišč tako razdelimo na dve fazi.

1. faza: Določanje začetne predstavitve, dodatka k kvaliteti predstavitve, cene izrisa ter vidljivosti

Po drevesu se premikamo rekurzivno v globino in določamo začetne vrednosti parametrov za vsako vozlišče. Listi vsebujejo po en sam objekt in enostavno izračunamo vrednost hevristične funkcije kot uteženo povprečje kvalitete ter cene in pogleda na objekt. Dodatek k kvaliteti za neko vozlišče izračunamo kot maksimum dodatkov vseh otrok. Neko vozlišče določimo kot vidno, če je viden vsaj en otrok. V določeni točki simulacije lahko izbiramo med predstavitvijo, ki je odvisna od pogleda in neodvisno predstavitvijo objekta. Katero predstavitev bomo izbrali kot začetno določimo z izbiro manjšega količnika cena/dodatek k kvaliteti vsake predstavitve.

2. faza: Prehod drevesa v načinu Best first

V tej fazi uporabimo vrednosti zapisane v prvi fazi za prehod drevesa v načinu best first. Rezultat je lista predstavitev objektov, ki jih pošljemo grafičnemu sistemu za izris. Lista meta-objektov sortiranih po padajočem vrstnem redu glede na donos k kvaliteti predstavitve nam služi za postavitev poti. Ti meta-objekti so vozlišča z vrednostmi, ki smo jih izračunali v prvi fazi. Izbiramo objekte iz liste in če vsota cen izbranih objektov in novega objekta ne presega postavljene meje, objekt sprejmemo, drugače pa se premaknemo na naslednji objekt v listi.
Najprej izberemo korensko vozlišče in ga uvrstimo v listo, ceno pa postavimo kot začetno ceno vozlišča. Nato nadaljujemo s pregledom liste, če je vozlišče list drevesa pogledamo ali lahko izberemo še boljšo predstavitev od že izbrane (ali nam čas to dovoljuje). Dodamo čas za generiranje objekta k skupnemu času. Vozlišču, ki vsebuje predstavitev gruče, pregledamo spisek otrok ali imamo dovolj časa za generiranje vseh sinov. Če imamo dovolj časa, uvrstimo v listo vse sinove, drugače pa predstavitev, ki jo hrani to vozlišče (primerno popravimo vrednost celotnega časa).

Literatura

[1] P. W. C. Maciel, P. Shirley: Visual navigation of large environments using textured clusters

[2] T. Frankhouser, C. H. Sequin: Adaptive display algorithm for interactive frame rates during visualisation..., Proceedings of SIGGRAPH '93

[3] A. V. Aho, J. E. Hopcroft, J. D. Uhlmann: The design and analisys of computer algorithms, Addison – Westley, 1974