Programski vmesnik za zvezno večstensko prostorsko projekcijo navideznega prostora na osebnem računalniku

I. Bajec, N. Zimic, I. Lapanja, M. Mraz
Fakulteta za računalništvo in informatiko
Univerza v Ljubljani
Tržaška 25, SI-1000 Ljubljana, Slovenija
ibajec@lotus.fri.uni-lj.si

Povzetek

Sledeči članek predstavlja opis postopkov, uporabljenih pri realizaciji programskega vmesnika za večstenski prostorski prikaz navideznega prostora na osebnem računalniku. Programski vmesnik je bil zasnovan na konkretnem primeru sistema za prvoosebno popolno potopitev na osebem računalniku - PC-CAVE, zgrajenega na osnovi "off-the-shelfšestavin.

Abstract

The following article is a review of the approaches used when developing a library for multipipe stereoscopic rendering of a virtual environment on a personal computer. The library was needed when developing a first person full immersion virtual reality system on a personal computer - PC-CAVE, which was based on off-the-shelf products.

1 Uvod

Leta 1838 je Whetstone s pomočjo naprave imenovane *stereoskop* razložil širsim množicam, da z vsamim očesom vidimo rahlo različno sliko. Rahla različnost slik je posledica fizičnega razmika naših oči, ki je pri odraslem človeku približno 64mm. To spoznanje je privedlo do prvih naprav, ki so opazovalcu uspešno nudile občutek globine - *prostorskosti*. Ena izmed prvih je bila naprava z imenom *View-Master*. Ta je s pomočjo dveh sistemov leč opazovalcu prikazala dve rahlo različni sliki (*prostorski par*) - vsako ustreznemu očesu opazovalca. Njegovi možgani so ti dve sliki združili v eno - celovitejšo - sliko, kar je imelo za posledico opazovalčev občutek prostorskosti [1].

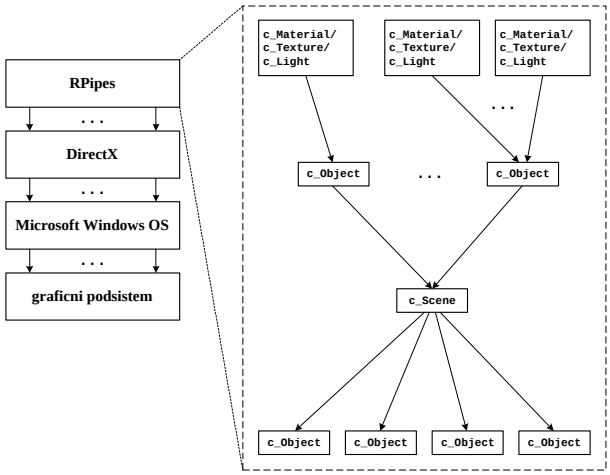
Kasneje, leta 1992, so raziskovalci Laboratorija za vizualizacije z Univerze Illinois v ZDA razvili prvi sistem za navidezno resničnost, ki opazovalcu nudi občutek popolne potopitve. Poimenovali so ga CAVE (ang. *CAVE automatic virtual environment*) [2]. CAVE je prostor v obliki kocke s stranico 3m, kjer na tri stene in tla projiciramo zvezno *prostorsko sliko*. Prostorska slika predstavlja kodiran prostorski par, ki se s pomočjo ustreznih dekoderjev (*stereo očala*) prikaže uporabnikovim očem. Tak pristop omogoča prekritje uporabnikovega vidnega polja v celoti, s tem pa dosežemo njegov občutek popolne potopitve v umetno zgrajeni - *navidezni* - prostor. Pristop velikih projekcij in dokaj velikega prostora omogoča tudi večuporabniško delovanje sistema. Temelj sistema CAVE predstavlja splošnonamenska knjižnica za izgradnjo navideznih svetov CAVELib. Ta temelji na namenskem programskem vmesniku za 3D grafiko OpenGL. Težava je le v tem, da obstoja le za delovne postaje oz. grafične supperračunalnike.

Vse hitrejši mikroprocesorji, predvsem pa zadnje čase vse boljši 3D grafični pospeševalniki za osebne računalnike, so nas pripeljali do ideje o izgradnji sistema za navidezno resničnost. Prvi poiskus v tej smeri je predstavljal umetniški projekt Virtualne Sanje. Ta je temeljil na osebnem računalniku, čeladi za navidezno resnišnost in posebnem vmesniku - slončku [3, 4]. Zaradi kasnejšega prodora še hitrejših grafičnih pospeševalnikov, predvsem pa njihove večje kvalitete, smo se odločili sestaviti sistem razreda CAVE. Poimenovali smo ga preprosto PC-CAVE (ang. *personal computer CAVE*).

2 Programska oprema

Časovne in cenovne omejitve med snovanjem sistema so nas privedle do odločitve o realizaciji sistema v obliki *multipipe* rešitve. Multipipe je izvedba v obliki enega računalnika z več grafičnimi karticami sposobnimi pospeševanja 3D operacij. Vsaka izmed njih izračunava in izrisuje svojo prostorsko sliko. Ta izvedba nam je omogočila tudi lažjo izgradnjo programskega vmesnika, saj so v tem primeru vsa sredstva v enem računalniku. Z drugimi besedami, ni potrebe po medsebojni sinhronizaciji večjega števila procesov, kot je to potrebno v primeru *multicomputer* rešitve. Multicomputer izvedba namreč pred-

videva uporabo večjega števila računalnikov, medsebojno povezanih v lokalno omrežje. Grafična kartica vsakega izmed njih pa izračunava in izrisuje svojo prostorsko sliko.



Slika 1: modularna zasnova vmesnika.

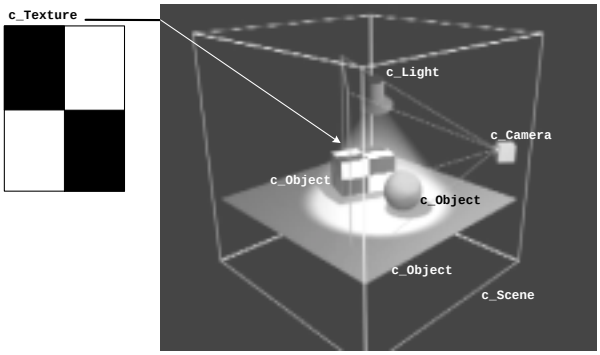
Kot osnovo smo vzeli računalnik razreda Pentium II 400Mhz z 256MB RAM. Njegov grafični podsistem predstavljajo štiri grafične kartice, ki temeljijo na procesorju nVidia Riva TNT, vsaka s po 16MB pomnilnika. Kot operacijski sistem smo, zaradi izbire osebnih računalnikov, izbrali Windows 98 z možnostjo nadgradnje na Windows 2000. Izbira Windows 98/2000 ni naključna, saj v tem primeru operacijski sistem sam dopušča uporabo več grafičnih kartic v enem samem računalniku - ti. *multimonitor* način - kar pomeni veliko prednost, saj ni potrebno razviti posebnih gonilnikov za neposredno krmiljenje večjega števila grafičnih kartic v enem sistemu. Kljub temu pa se je pojavila potreba po medsebojni sinhronizaciji grafičnih kartic. Rešili smo jo z manjšim trikom. Trem katicam smo odvzeli kristale in jih povezali z glavno grafično kartico. Tako so vse kartice delovale z istim kristalom - v popolnoma enakem taktu. V povezavi s tem smo uporabili kodiranje prostorskega para vrste zgoraj-spodaj. Ta deluje po principu prikaza prostorskega para v eni sliki, kjer je zgornja polovica kodiranje slike podslika namenjena enemu očesu, spodnja pa drugemu.

2.1 Temelji vmesnika

Pri snovanju smo se, zaradi želje po vmesniku, ki bi omogočal dokaj hitro izgradnjo navideznih svetov in hkrati bil transparenten glede načina prikaza, odločili za vmesnik na osnovi prostorskega opisa. Za pomoč pri rutinskih operacijah v 3D grafiki, kjer programer večino časa porabi za izbiro materialov, vzorcev ter luči kot osnovnih gradnikov navideznega okolja (v nadaljevanju samo prostora), smo razvili module `c_Light`, `c_Material` in `c_Texture` (slika 2). Osnovne operacije `c_Light` so izbira modela in nastavitve parametrov luči, s čimer programerju omogočimo njihovo preprostejšo in hitrejšo postavitev v prostor. Med drugim s takšnim pristopom preusmerimo skrb za preverjanje večkratnega obstoja luči v prostoru s programerja na modul. `c_Material` predstavlja skupek operacij z materiali, kot je npr. izbira materiala geometrijske predstavitve predmeta v prostoru. Tudi tu, zaradi takšnega pristopa, skrb za učinkovito izrabo razpoložljivih sredstev in večkratno uporabo že uporabljenih materialov preide s programerja na modul sam. `c_Texture` pa predstavlja skupek osnovnih operacij nad vzorci, s katerimi lahko programer izbere vzorec, ki se pojavi na prikazovanem predmetu [5].

2.2 Moduli prostorskega opisa

Prostorski opis je podatkovna struktura, ki opisuje vsebino prostora (slika 2). Zamišljamo si jo lahko kot spisek uporabljenih rekvizitov in njihovih lokacij pri opisu vsebine gledališkega odra. Navadno vsebuje opis geometrijske predstavitve predmetov v prostoru, kakor tudi opis njihove lokacije, materiala ter njihovega vzorca. Pri večini visokonivojskih vmesnikov za 3D grafiko ima takšna struktura hierarhično obliko drevesa, v katerem so podatki večinoma razvrščeni prostorsko. Primer: predmeti, ki so dejansko bližje skupaj, so blizu tudi v drevesni predstavitvi. Moduli, namenjeni rokovanju s prostorkim opisom, so sledeči:



Slika 2: prostor in njegovi gradniki.

- `c_Object` je osnovni nadrazred, iz katerega programer izpeljuje svoje razrede. Uporablja ga za opis geometrijske predstavitve predmeta v prostoru ter njegovih lastnosti. Osnovni operaciji, ki ju `c_Object` pozna, sta določitev pozicije ter orientacije v prostoru. Programerjeva naloga je izpeljava novega razreda s spremembo

funkcij `SetupState()` ter `Render()`, pri čemer s pomočjo prve nastavi ustrezne lastnosti geometrijske predstavitve, kot sta uporabljen material ter vzorec, z drugo pa izvede dejanski izris geometrije. S takšnim pristopom programerju ostaja odprta pot v nadgradnjo razreda z dodatnimi lastnostmi.

- `c_Scene` predstavlja razred, ki skrbi za prostorski opis. Trenutna implementacija zaradi časovnih omejitev ne vsebuje optimizacijskih posegov, vendar programerju dopušča vso svobodo realizacije le-teh. Predvideno razširitev predstavlja implementacija večstopenjske optimizacije, kjer se izris slike dogaja v treh fazah. Najprej iz prostora izločimo predmete, ki s strani opazovalca niso vidni. Zatem pri preostalih izberemo ustrezen nivo natančnosti. Nazadnje trikotnike, ki so nujni za izris slike, razvrstimo tako, da zmanjšamo število potrebnih sprememb parametrov izrisa. Seveda takšna optimizacija predvideva tesno sodelovanje vseh modulov vmesnika.

- `c_Camera` je razred, ki nudi osnovne operacije pozicioniranja opazovalca v prostoru. S pomočjo funkcije `SetLookAt()` programer pove mesto opazovalca in smer njegovega pogleda. `c_Camera` poskrbi za osvežitev notranjih struktur in izračuna projekcijske matrike celotnega grafičnega podsistema.

```
izračunaj pozicijo oči:  levoOko, desnoOko.
for vsak prijavljen RPipe
    rotiraj levoOko in desnoOko za RPipe.FOV.
    izračunaj projekcijsko matriko levegaOčesa.
    izračunaj projekcijsko matriko desnegaOčesa.
```

sprememba pozicije gledalca.

2.3 Glavni modul vmesnika

Zasnova modula `c_RPipe` predstavlja glavni del vmesnika. Ta mora ob inicializaciji sistema vzpostaviti kontakt z grafičnim podsistemom in izbrati njegov najboljši način delovanja. Po uspešni vzpostavitvi parametrov delovanja posamezne kartice se mora prijaviti moduloma `c_Scene` ter `c_Camera`.

Pri izbiri grafičnega vmesnika na katerem temelji celotni sistem smo se odločili za DirectX. Zanj smo se odločili na podlagi pregleda razmerij cena/hitrost grafičnih pospeševalnikov OpenGL in DirectX za osebne računalnike. K takšni odločitvi je pripomogla tudi podpora ti. *multimonitor* načina delovanja vmesnika DirectX.

Multimonitor arhitektura (imenovana tudi MultiMon) omogoča operacijskemu sistemu uporabo

```
preštej grafične kartice:  RPipe.
postavi način izrisa v RPIPES_STEREO_ZGORAJ_SPODAJ.
if le en RPipe
    odpri RPipe kot glavni.
else
    odpri vse RPipe, kjer naj bo prvi glavni.

nova Camera.
nova Scena.

for vsak RPipe
    priključi Camero na RPipe.
    priključi Sceno na RPipe.
    nastavi parametre projekcije RPipe.
```

inicializacija grafičnega podsistema.

prikazne površine dveh ali več grafičnih kartic kot eno skupno prikazno površino. Vzpostavitev kontakta z grafičnim podsistemom predstavlja vzpostavitev takšnega načina delovanja vmesnikov DirectDraw in Direct3D (v nadaljevanju DirectX).

Kot že prej omenjeno; DirectX že v sami zasnovi omogoča uporabo multimonitor načina delovanja, v kolikor pri vzpostavitvi kontakta z grafično kartico, tj. klicu funkcije `DirectDrawCreateEx()`, navedemo namesto GUID (ang. *Globally Unique Identifier*) grafične kartice vrednost null. To pomeni, da lahko okno aplikacije, ki uporablja DirectX, poljubno premikamo po celotni skupni prikazni površini grafičnega podsistema.

V tem primeru bo DirectX izkoriščal strojno pospeševanje 3D operacij le v primeru, ko bo okno aplikacije v celoti na prikazni površini glavne grafične kartice. DirectX bo izkoriščal strojno pospeševanje na celotni skupni prikazni površini le v primeru, če progamer vzpostavi ti. ekskluzivni celozaslonski način delovanja vsake posamezne kartice grafičnega podsistema.

Tako PC-CAVE sistem zaradi svoje zasnove zahteva ekskluzivni celozaslonski način delovanja, pri čemer pa je potrebna vzpostavitev popolne kontrole nad grafičnim podsistemom in natančna določitev režima delovanja posamezne grafične kartice. Vzrok za to je nezmožnost hardverskih preslikav med medpomnilniki različnih grafičnih kartic.

Vzpostavitev eksluzivnega celozaslonskega delovanja DirectX aplikacije predstavlja nastavitev fokusnega okna in okna enote.

Vsaka DirectX aplikacija, ki deluje v celozaslonskem ekskluzivnem načinu delovanja, mora imeti eno fokusno okno. Fokusno okno je tisto, ki skrbi za sprejem podatkov posredovanih prek tipkovnice.

Vsaka grafična kartica, ki deluje v celozaslonskem načinu, mora biti DirectX predstavljena z DirectDraw objektom in oknom enote. Okno enote predstavlja okno, v katerem izrisujemo sliko - nahaja se na vrhu vseh ostalih oken.

Pri aplikacijah, ki ne delujejo v načinu multimonitor, predstavljata okno enote in fokusno okno isto okno. Pri aplikacijah, ki delujejo v tem načinu, je za vsako grafično kartico potrebno nastaviti okno enote. Med drugim je potrebno tudi vsakemu DirectDraw objektu predstaviti fokusno okno, ki je samo eno - aplikacijsko.

```
postavi aplikacijsko okno kot fokusno.
if glavni RPipe
    okno enote je fokusno okno.
else
    novo okno enote.
```

nastavitev fokusnega okna in okna enote.

Ko `c_RPipe` vzpostavi kontakt z grafičnimi karticami, pri vsaki nastavi fokusno okno ter okno enote in se uspešno prijavi moduloma `c_Scene` in `c_Camera`, je vse nared za izris. Opozoriti je potrebno, da programer sam določa vsebino izrisanega prostora (glej 2.1). Modulova naloga je predvsem izris prostora v obliki zvezne prostorske slike.

```
for vsak RPipe
    vprašaj Camero za perspektivo levega očesa.
    vzpostavi parametre vidnega polja.
    izriši Sceno.
    vprašaj Camero za perspektivo desnega očesa.
    vzpostavi parametre vidnega polja.
    izriši Sceno.
```

izris prostorske slike.

Programer s pomočjo modulov zgradi prostorski opis, vzpostavi kontakt z grafičnim podsistemom, ter prepusti kontrolo izrisa vmesniku. Le-ta zna z večkratnimi zahtevami izrisa prostora z več različnih očišč prikazati zvezno prostorsko sliko prostorskega opisa, ki ga je posredoval programer.

S takšnim pristopom izgradnje vmesnika smo dosegli hitrejši razvoj aplikacij ter neodvisnost programskega dela od načina prikaza.

3 Sklep

Sistem smo zasnovali tako, da smo predvideli tudi možne razširitve. Če pogledamo najprej s strani stopnje potopitve. Tako kot sistemi CAVE bi tudi ta lahko uporabljal smer pogleda uporabnika - vodiča, za aktivno obnavljanje smeri pogleda in morda celo ZPS (ang. *zero parallax setting*) za dosego še boljšega prostorskega občutka uporabnika. Možno razširitvev predstavlja tudi podpora različnih taktilnih vmesnikov, npr. podatkovnih rokavic, s čimer omogočimo uporabnikovo interakcijo z navideznim okoljem. Nenazadnje je tu še, verjetno tudi najnujnejša, nadgradnja v obliki optimizacije vseh gradnikov. Tu je mišljena aktivna optimizacija števila trikotnikov, posredovanih v obdelavo grafičnemu podsistemu, s poudarkom na tehnikah, predstavljenih v delih Hoppa in Xie [6, 7].

Aplikativna uporabnost vmesnika se je pokazala med izgradnjo projekta vizualizacije simulacije širjenja požarov v homogenem okolju na osnovi mehke logike Laboratorija za računalniške strukture in sisteme na Fakulteti za računalništvo in informatiko v Ljubljani.

Sistem PC-CAVE je bil tudi uporabljen pri izgradnji dveh umetniških projektov: BARCODE - II. potopitev, ki je bil razstavljen v galeriji Kapelica na Kersnikovi 4 v Ljubljani med 14.5.1999-28.5.1999, ter BARCODE - III. potopitev, ki je bil razstavljen na Štajerski jeseni v galeriji Avl_ArtGate v Gradcu med 24.9.1999-24.10.1999 in na Manifesti[3] v Cankarjevem Domu v Ljubljani med 23.6.2000-24.9.2000.

Literatura

[1] StereoGraphics Corporation, *StereoGraphics Developers Handbook*, 1997.

[2] Fakespace systems,
A Complete Set of Tools to Let You Work,
`http://www.fakespacesystems.com/products.html`.

[3] I. Bajec, N. Zimic, I. Lapanja, M. Mraz, *Postopek računanja pozicije sprehajalca v prostoru s pomočjo vozička*, Zbornik ERK'98, str. B: 171-174, Portorož 1998.

[4] I. Bajec, *Virtualne sanje - uporaba navidezne resničnosti v moderni umetnosti*, Zbornik ERK'98, str. B: 465-466, Portorož 1998.

[5] A. H. Wat, *Fundamentals of Tree-Dimensional Computer Graphics*, Addison-Wesley, 1989.

[6] H. Hoppe, *Smooth View Dependant Level-of-Detail Control and its Application to Terrain Rendering*, Microsoft Research.

[7] J. C. Xia, J. El-Sane, A. Varshney, *Adaptive Real-Time Level-of-Detail Rendering of Poligonal Models*, University of New York at Stony Brook.