Szakdolgozat

Miskolci Egyetem

Itt jelenik meg a szakdolgozat címe, akár több sorban is

Készítette:

Ide kerül a hallgató neve

Évfolyam. szak-szak

Témavezetõ:

Egyik konzulens neve

Másik konzulens neve...

Miskolc, 2016

Miskolci Egyetem
Gépészmérnöki és Informatikai Kar
Alkalmazott Matematikai Intézeti Tanszék **Szám:**

Szakdolgozat Feladat

Árva Zoltán (B5X5NT) mérnökinformatikus jelölt részére.

A szakdolgozat tárgyköre: számítógépi grafika, optimalizálás, C++ programozás

A szakdolgozat címe: Számítási modellek bemutatása egy FPS játék készítése kapcsán

A feladat részletezése:

Témavezetõ: Piller Imre (egyetemi tanársegéd)

A feladat kiadásának ideje:

. . . . . .
szakfelelõs

Eredetiségi Nyilatkozat

Alulírott .....; Neptun-kód: .....
a Miskolci Egyetem Gépészmérnöki és Informatikai Karának végzõs .....
szakos hallgatója ezennel büntetõjogi és fegyelmi felelõsségem tudatában nyilatkozom és aláírásommal igazolom, hogy .....
címû szakdolgozatom/diplomatervem saját, önálló munkám; az abban hivatkozott szakirodalom felhasználása a forráskezelés szabályai szerint történt.

Tudomásul veszem, hogy szakdolgozat esetén plágiumnak számít:

- szószerinti idézet közlése idézõjel és hivatkozás megjelölése nélkül;
- tartalmi idézet hivatkozás megjelölése nélkül;
- más publikált gondolatainak saját gondolatként való feltüntetése.

Alulírott kijelentem, hogy a plágium fogalmát megismertem, és tudomásul veszem, hogy plágium esetén szakdolgozatom visszautasításra kerül.

Miskolc, ...év ..... .hó ..... .nap

.....
Hallgató

1.

szükséges (módosítás külön lapon)

A szakdolgozat feladat módosítása

nem szükséges

....

dátum témavezetõ(k)

2. A feladat kidolgozását ellenõriztem:

témavezetõ (dátum, aláírás): konzulens (dátum, aláírás):

....

....

....

3. A szakdolgozat beadható:

....

dátum témavezetõ(k)

4. A szakdolgozat ..... szövegoldalt

..... program protokollt (listát, felhasználói leírást)

..... elektronikus adathordozót (részletezve)

.....

..... egyéb mellékletet (részletezve)

.....

tartalmaz.

....

dátum témavezetõ(k)

5.

bocsátható

A szakdolgozat bírálatra

nem bocsátható

A bíráló neve: .....

....

dátum szakfelelõs

6. A szakdolgozat osztályzata

a témavezetõ javaslata: .....

a bíráló javaslata: .....

a szakdolgozat végleges eredménye: .....

Miskolc, ...

a Záróvizsga Bizottság Elnöke

Tartalomjegyzék

1. fejezet

Bevezetés

Egy FPS játék elkészítése különféle számítási- és optimalizálási problémát felvet.

Sorravenni, hogy milyen jellegűek ezek, hogyan kapcsolódnak egyáltalán egy témához.

Ez a rész elég ha 1-2 oldal. Bőven elég lesz majd csak a végén összerakni.

2. fejezet

Problémakör

Játékmotorok

- Quake, Unreal, Unity, Cry Engine említés szintjén - A játékmotorok fő funkciói - Miért volt szükség új játékmotor implementálására?

Problémafelvetés

- Leírni, hogy melyek azok a funckiók, amelyekre a játékmotorban szükségesek.

Milyen konkrét problémát old meg?

- Felsorol néhány gyakori optimalizálási problémát.

Miért volt szükség az elkészítésére?

- Lehetőséget adott az optimalizálási problémák vizsgálatára. - Aktívan kutatott terület. - Az eredmények szemléletesek.

2.1. Mesterséges intelligencia

Egyjátékos FPS játékról van szó, így mindenféleképp kellett egy, az ellenfelek mozgását irányító mesterséges intelligencia. Ez az egész az A* algoritmusra alapszik. A játék véletlenszerűen, különböző helyekre kirak x mennyiségű ellenfelet, amiknek közeledni kell a játékos felé, különböző kritériumoknak megfelelve. Ezek a kritériumok azért kellenek, mert a pályán vannak játékelemek, amiken nem lehet átmenni, illetve az ellenfelek sem mehetnek egymásba.

Az alapötlet az, hogy a pályán le lesznek rakva pontok, ún. waypointok, amik csak az ellenfelek számára lesznek láthatók. Ha kikerül egy adott ellenfél a pályára, az első feladata az lesz, hogy megkeresse a hozzá legközelebb eső waypointot. Ezt egy sima pitagorasz tétellel számolja ki. Ezután az A* algoritmus segítségével meghatározza a játékoshoz legközelebb eső waypointhoz vezető legrövidebb utat, végigmegy rajta, majd ha elérte, akkor onnantól a játékos lesz a közvetlen célpontja. Mivel a játékos folyamatosan mozog, mindezt másodpercenként legalább 15-30x kell kiszámolni.

2.2. Hangok

Egy játék alapelemeihez hozzátartoznak a hangeffektek, zenék is, amik élvezetessé teszik azt. Egy jól megválasztott zene például nagyon sokat tud javítani a játékélményen. Ezek megszólalásáért az SDL_mixer felelős, ami lehetővé teszi azt is, hogy több hang egyidőben, átfedésben megszólaljon, és ne várják meg egymást. Ezt a Sound osztályban

valósítottam meg. Lehetőség van hangcsatornákat megadni, folyamatos újrajátszást, illetve a hangok egymáshoz viszonyított hangerejét beállítani.

2.3. Magasságmező

A játékteret adó domborzatot, a játék egy fekete-fehér, max 400x400pixeles képből számolja ki. Az adott képen minél magasabb egy pont, annál fehérebb, így a fehér adja a legmagasabb, a fekete szín a legalacsonyabb pontot. Ezekből az adatokból vissza lehet számolni a játékos függőleges pozícióját is, illetve az átmeneteket nézve, meredekségnek megfelelően a visszacsúszást. Mindez tehát megadja a játékteret minden szempontból (vizualitás, játékos mozgástere).

2.4. Konkrét megjelenítés módja

A játéktér adatait tehát képből kinyertük, de ez még nem jelenti azt, hogy látjuk is a monitoron. Ehhez a VBO-s (Vertex Buffer Object) kirajzolási módszert választottam, ami a videókártyának a legoptimálisabb adatstruktúrában adja át azt a lehető leggyorsabb kirajzoláshoz.

2.5. A csapda mint játékelem

Mindenképp szerettem volna olyan elemet a játékba, ami megkülönbözteti a többi hasonló, aréna jellegű, túlélős játéktól. Az egyik ilyen elem, hogyha a játékos megáll x ideig (az x majd tesztelés során derül ki mi a legoptimálisabb), akkor megjelenik egy csapda, majd leesik, és meghal. Vannak olyan játékok, ahol szimplán az ellenfelek ösztönzik a játékost a mozgásra, de itt konkrétan ki is van kényszerítve. Jelenlegi beállítás: Ha megáll a játékos 5mp-ig, akkor megjelenik a csapda modell egy hang kíséretében, illetve onnantól nem lehet már pozíciót változtatni, csak az egérrel körbe nézni. Majd még 2mp leteltével már az egeret sem érzékeli, a föld felé fordul a kamera, majd leesik, ekkor ismét egy hangot lehet hallani, és értesül a játékos arról, hogy miért halt meg.

2.6. A lövés

Egy ilyen játékban alapelem az is, hogy tudjunk lőni, mivel ez nélkül az egész értelmét veszti. A textúrázott játéktéren ugyan nem látni, de az egész több százezer háromszögből áll. Adott egy irányvektor, ami azt határozza meg, merre fordul épp a játékos, azaz mire néz. Ennek a vektornak a metszéspontját kell vizsgálni a háromszögekkel, hogy találatot meg tudjuk jeleníteni, ami jelen esetben a falon egy lövésnyom. Szóval ki számolja a program a metszéspont x, y, z koordinátáját, majd odailleszt egy lövésnyom textúrát. Az ellenfeleknél szintén metszéspontot kell számolni, csak ott hengerre. Ki lehetne számolni a modellre is közvetlen, de az nagyon lassú lenne, így egy leegyszerűsített alakzatot választottam. Ez az ellenfél ún. hitboxa.

3. fejezet

Komponensek

- Felhasznált technológiák (C++, SDL) - Nagyobb komponensek

Le kellene írni, hogy hogyan épül fel a program. Milyen nagyobb modulokból épül fel.

A játék indításának a folyamatát is lehetne részletezni.

4. fejezet

Ütközésvizsgálat

Ütközéssel, metszéspontokkal kapcsolatos számítások

- A try bullet-es példa részletes kifejtése, matematikai része

Egyszerűbb változatoktól a bonyolultabbak felé

- Először az intuitív, egyszerűbb megoldások, még ha kevésbé hatékony is. - Felsorolni több megoldást is. - Optimalizálási módszerek részletezése - Összehasonlítani őket pontosság és számításigény szempontjából.

5. fejezet

Útvonalkeresés

Gráf adatstruktúra bemutatása a waypoint-ok kezeléséhez

- Milyen adatok tartoznak egy waypoint-hoz - A waypoint-ok kezeléséhez szükséges műveletek (például távolságszámítás) - Leírni, hogy hogyan menne el az egyik pontból a másikba.

A*-algoritmus Waypointok generálása: Delaunay triangulation

6. fejezet

Animáció

A csontváz kezelésének és számítógépen való ábrázolásának bemutatása

- Kulcsképkocka animáció - Csontváz alapú animáció - Blender-es animációs formátum - Egyszerűbb görbék bemutatása, amelyeket animációkhoz használnak (interpolációs módszerek)

A mesterséges intelligencia és az evolúciós algoritmusok szerepe a probléma megoldásában

- Megnézni azt a cikket, amihez az animációs videó készült. - Felvetni néhány módszert, például ANN vagy evolúciós algoritmus. (Itt még nem kell majd implementálni, egyelőre csak legyen összegyűjtve.)

7. fejezet

Ellenfelek viselkedésének modellezése

Ágens alapú viselkedés

- Leírni, hogy milyen bemenetek és kimenetek szükségesek ahhoz, hogy az ellenfelek viselkedését modellezzük.

Klasszikus, állapotgép alapú modell bemutatása Véletlenszerű tényezők a játék érdekesebbé tételéhez

8. fejezet

Tesztek

A játékmotor vizsgálata egy az egyben Review jellegű észrevételek Profilozás

9. fejezet

Összegzés

A bevezetéshez hasonlóan, csak itt már múltidőben.

Irodalomjegyzék

[1] Bujdosó Gyöngyi, Fazekas Attila: *TEX kezdõlépések*, Tertia Kiadó, Budapest, 1997.

[2] Házy Attila: *Lineáris függvényegyenletek megoldása számítógéppel*, Doktoranduszok fóruma 2005, Miskolc, 2005. november 9., Gépészmérnöki Kar szekciókiadványa, Miskolc, ME ITTC, 2006., 108–113.

[3] Hettl, Mayer, Szabó: *LATEX kézikönyv*, Panem Könyvkiadó, Budapest, 2004.

[4] M. E. Hohmeyer, B. A. Barsky: Rational continuity: parametric, geometric and Frenet frame continuity of rational curves, *ACM Transactions on Graphics*, **8** (1989), 335–359.

[5] TEX Catalogue, `www.ctan.org/tex-archive/help/Catalogue/catalogue.html`