3D modell és játékfejlesztés

Naprendszer szimuláció (pygame és modern OpenGL)

.

Készítette:
Becskeházi Róbert THE LPTI Bsc
Neptun kód: NKOB5W
Kurzuskód: THE_00747_L_5_K

Tartalom

A Projekt Célja és Általános Felépítése

A projekt alapvető célja egy valós időben működő, interaktív 3D-s Naprendszer megvalósítása volt, amely Python környezetben, a Modern OpenGL 3.3-as magprofiljára építve jeleníti meg az égitestek mozgását és textúráját. A program a Pygame segítségével kezeli az ablakot és a felhasználói eseményeket, míg a transzformációkhoz szükséges matematikai műveleteket a PyGLM végzi. A szimuláció teljes képernyős módban indul, és minden grafikai számítást a GPU-n futó GLSL shaderek végeznek, biztosítva a folyamatos és valósághű megjelenítést.

A rendszer a Nap köré szerveződik, amely egyben a szimuláció fényforrása is, és a nyolc bolygó mozgását követi, beleértve a Hold Föld körüli keringését, valamint a Szaturnusz gyűrűinek külön kezelést igénylő textúráját. A bolygók a térben folyamatosan keringenek a Nap körül, miközben a saját tengelyük körüli forgás is érvényesül, így a modell egyszerre jelenít meg többféle dinamikus mozgást.

A felhasználó egy teljesen szabad nézőpontú, első személyű kamerával járhatja be a Naprendszert. A térbeli elmozdulást a WASDQE billentyűk, a nézet irányítását pedig az egér mozgatása határozza meg, amely a Yaw és Pitch értékek módosításával állítja be a kamera irányvektorát. A szimuláció sebessége menet közben is változtatható: a plusz és mínusz billentyűkkel növelhető vagy csökkenthető az égitestek mozgásának üteme, szükség esetén pedig a folyamat bármikor szüneteltethető.

A tájékozódást a pályavonalak segítik, amelyek finom fehér ívekként jelennek meg a bolygók körül, jól láthatóvá téve a térbeli elrendezést és a keringések útvonalát. Az így kialakított rendszerben a felhasználó egyszerre élheti át a 3D-s, interaktív világ felfedezésének élményét és követheti a fizikai mozgások szemléletes modelljét.

Geometria és Logikai Megvalósítás

A program architektúrája a `main.py` fájlban szerveződik egységgé, ahol a geometriai objektumok előállításáért és a szimuláció működéséért felelős osztályok egy koherens rendszert alkotnak. A teljes modell működését olyan objektumok vezérlik, amelyek külön feladatköröket látnak el, kezdve a geometriák előállításától a bolygók mozgásán át egészen a kameravezérlésig.

A geometria generálását végző osztályok gondoskodnak arról, hogy a GPU számára minden szükséges adat rendelkezésre álljon. A gömb hálozatát létrehozó Sphere osztály állítja elő a vertex pozíciókat, a normálvektorokat és a textúra koordinátákat, így ez adja a Nap, a bolygók és a Hold megjelenítésének alapját. A Szaturnusz gyűrűjének megjelenítését a `Ring` biztosítja, amely egy belső és egy külső sugár alapján hoz létre egy lapos, körgyűrűs geometriát. A bolygók keringési pályáit az Orbit osztály rajzolja ki finom, kör alakú vonalláncok formájában, míg a háttér csillagmezőjét a StarField hozza létre nagyszámú véletlenszerű ponttal, amelyek egyszerű fényforrásként jelennek meg a tér mélyén. A dinamika megvalósításának központi eleme a Planet osztály, amely nemcsak a bolygók fizikai jellemzőit tárolja, hanem gondoskodik a mozgásuk időbeli frissítéséről is. Az `update` függvény minden képkocka során növeli a bolygó keringési és tengelyforgási szögét, a felhasználó által meghatározható sebességi faktorral skálázva ezeket. Ily módon a szimuláció tempója bármikor módosítható, ami különösen látványos, amikor a bolygók mozgása felgyorsul vagy lelassul. A térbeli elhelyezés a get_model_matrix metódus feladata, amely pontosan meghatározza a bolygó pozícióját és orientációját a világban. A transzformációk sorrendje itt létfontosságú: először a saját tengely körüli forgás történik meg, ezt követi a bolygó méretezése, majd a pályán való eltolás, végül a Nap körüli keringéshez szükséges forgatás. Ez a sorrend biztosítja, hogy a bolygó mozgása és valódi helyzete a világban valósághű és következetes legyen.

A kamera működése ugyancsak a fő logikai kör része. A felhasználó egérmozgása határozza meg a Yaw és Pitch értékeket, amelyekből kiszámítható a kamera irányát leíró vektor. A pozíció és az irány alapján áll elő a nézőpontot leíró View mátrix, amely a világ objektumait a kamera szemszögébe transzformálja. A billentyűzeten megadott irányok segítségével a nézőpont szabadon mozgatható, így a felhasználó tetszőleges nézőpontból járhatja be a Naprendszer teljes modelljét.

Ez az architektúra egységes rendszerré áll össze: a geometria adja a világ formáját, a bolygók logikája biztosítja a mozgást, a kamera pedig megteremti az interaktív felfedezés élményét. Az egész együtt egy valós idejű, dinamikus 3D-s környezetet alkot, amelyben a Naprendszer működése valóban átélhetővé válik.

Renderelési Pipeline és Shaderek

A szimuláció vizuális megjelenítését egy többfázisú renderelési folyamat irányítja, amely négy különálló shader programra épül. Ezek együtt biztosítják a 3D objektumok valósághű

megvilágítását, a Nap emisszióját, a pályavonalak rajzolását és a HUD réteg kezelését. A teljes rendszer közös alapja az MVP (Model–View–Projection) transzformáció, amely minden geometriai objektumot a világkoordinátákból a képernyőre vetít.

A bolygók shader programja végzi a legösszetettebb munkát. A vertex shader itt nem csupán a koordinátákat alakítja át, hanem előkészíti a megvilágítási számításhoz szükséges adatokat is. A világkoordinátába transzformált pozíciókat, valamint a normálvektorokat továbbítja a fragment shader felé, amely meghatározza a bolygók felületének megjelenését. A megvilágítás két komponensből áll: egy állandó környezeti fényből, amely biztosítja, hogy a felszín árnyékos részei se legyenek teljesen sötétek, valamint egy szórt fényből, amely a Nap irányából érkező megvilágítást adja vissza a normálvektorok alapján. A diffúz fény számítása határozza meg, hogy a felszín mely részei kapnak több vagy kevesebb fényt, így alakul ki a valósághű árnyékolás. A textúra színe és a kiszámolt fényerő kombinációja adja meg végül a bolygók megjelenését.

A Nap shader programja ettől eltérő működést követ. Mivel a Nap a szimulációban fényforrásként viselkedik, nincs szükség megvilágítási számításokra. A fragment shader egyszerűen felerősíti a textúra értékét, ezzel keltve azt a hatást, hogy a Nap saját fényt bocsát ki. Ez különbözteti meg vizuálisan a bolygóktól, és ezzel válik a jelenet központi fényforrásává.

A speciális objektumok megjelenítéséhez külön shader programok segítik a renderelést. A Szaturnusz gyűrűje például átlátszó textúrával rendelkezik, ezért a rajzolás előtt szükséges az alfa-keverés engedélyezése. Így a gyűrű finoman simul rá a háttérre, és a megfelelő helyeken áttetsző marad. A pályavonalak ezzel szemben nagyon egyszerű geometriák: a shaderük mindössze a pozíció átalakítását végzi, majd egyszínű, kontrasztos fehér színnel rajzolja ki őket.

A HUD kezeléséhez teljesen különálló shader páros működik. Ezek a shaderek a 3D világ helyett a képernyő síkjában dolgoznak, ezért ortografikus projekciót alkalmaznak. A HUD elemei — például az FPS vagy a sebesség kijelzése — így minden kameramozgástól függetlenül rögzítve jelennek meg. A megjelenítéskor kikapcsolásra kerül a mélységteszt, és engedélyeződik az alfa-keverés, hogy a 2D-s szövegréteg akadálytalanul kerüljön a 3D jelenet elemei fölé.

Az egész renderelési folyamat úgy épül egymásra, hogy a bolygók, a Nap, a gyűrűk, a pályavonalak és a HUD egyetlen képkockán belül összeállnak, mégis mindegyik külön shader programmal éri el a számára szükséges vizuális hatást. Ez teszi lehetővé, hogy a szimuláció egyszerre legyen látványos, valósághű és technikailag stabil.

Összegzés és továbblépési lehetőségek

A projekt végére egy olyan rendszer jött létre, amelyben a geometriai modellezés, a shader-alapú renderelés és az objektumorientált logika egységben működik, valós időben megjelenítve a Naprendszer dinamikus mozgásait. A Modern OpenGL által biztosított vizuális minőség és a Python rugalmassága együtt tette lehetővé, hogy a szimuláció egyszerre legyen látványos, interaktív.

A program nem csupán egy kész modell, hanem egy olyan fejlesztési alap, amely továbbfejleszthető – részletesebb fizikai működés, új égitestek, bővített vizuális effektek vagy akár komplexebb kamera- és felhasználói interakciók irányába. A mostani verzió tehát egyszerre lezárt munka és egy következő szint kiindulópontja, amely tovább építhető, finomítható és még valósághűbbé tehető.