Görbe ábrázolása Lagrange-polinom használatával

Féléves beadandó Számítógépi grafika tantárgyból

Együd Vivien

C11M1L

2023. március

### Megvalósítandó feladat

Görbe kirajzolása Lagrange-polinom használatával 10-20 megadott ponttal OpenGL grafikus megjelenítés segítségével.

Program leírása

A feladat bemutatása a GLUT keretrendszerre épül, mellyel egyszerűen inicializálható adott platformon az OpenGL ablakozó rendszer.

A program indításakor 3 opcióból lehet választani: két előre rögzített demó rajzolás, valamint saját bevitt pontokra rajzolás. A bevitt és az előre rögzített demó pontokat egy Data nevű struct-ban tároljuk.

struct Data {  
 float x, y;  
};

A görbe adatainak tárolására a következő változókat használjuk:

* numberOfPoints: int típus, a pontok darabszáma
* *controlPoints[PMSIZE][3]*: a képernyőn zölddel megjelenített pontok mátrixa, ezekre rajzolja a függvényt a program

A darabszám és a mátrix is a program kezdetén inicializálásra kerül, majd a menü opciók választása után adatokkal töltjük fel.

for (int i = 0; i <= n; i++) {  
 controlPoints[i][0] = function[i].x / 10;  
 controlPoints[i][1] = function[i].y / 10;  
}

Ezután a megadott pontokat felhasználva a Lagrange-polinom segítségével kiszámoljuk a függvény pontjait. Ehhez -1 és 1 között 200 lépésben minden xi-re meghívjuk az interpolációt végző függvényt, a kapott pontpárokat elmentjük a *curvePoints* mátrixban. A ciklus során szükséges az adatok módosítása a számoláshoz, mert a megjelenítéskor csak -1 és 1 közé eső értékek lesznek láthatóak a képernyőn.

float xi = -1.0;  
for (int i = 0; i < 200; i++) {  
 curvePoints[i][0] = xi;  
 float yi = interpolate(function, xi \* 10, n);  
 curvePoints[i][1] = yi / 10;  
 xi = xi + 0.01;  
}

Az interpoláció számításához bemenő paraméterként megadjuk az ismert pontokat, az xi-t, és a pontok darabszámát. Két ciklus segítségével kiszámoljuk az egyes tagokat, behelyettesítve az ismert pontokat és xi-t a képletbe. Vizsgálni kell, hogy a két ciklusváltozó (i és j) nem lehet egyenlő. A kiszámolt tagok összege megadja az xi-hez tartozó yi-t.

float interpolate(Data function[], float xi, int n) {  
 float result = 0;  
 for (int i = 0; i < n; i++) {  
 float term = function[i].y;  
 for (int j = 0; j < n; j++) {  
 if (j != i) {  
 term = term \* (xi - function[j].x) / float(function[i].x - function[j].x);  
 }  
 }  
 result += term;  
 }  
 return result;  
}

Az OpenGL futásához szükséges beállításokat a *main* függvényben futtatjuk. Képernyőbeállítások, ablakmegjelenítéshez tartozó függvényparancsok ezek.

glutInit(&argc, argv);  
glutInitWindowSize(1024, 768);  
glutInitWindowPosition(250, 250);  
glutInitDisplayMode(**GLUT\_RGB**);  
  
glutCreateWindow("me-grafika-beadando");  
psInit();  
glutDisplayFunc(psDisplay);  
glutKeyboardFunc(psKey);  
  
glutMainLoop();

A görbe megjelenítése több részletben történik. Első lépésben felrajzoljuk a koordinátarendszert, majd zöld pontokban jelölve a görbe ismert pontjait. Ezután rajzoljuk rá a pontokra a kiszámolt görbét.

static void psDisplay() {  
 glClear(**GL\_COLOR\_BUFFER\_BIT**);  
 */\*  
 \* Felrajzoljuk a koordinata rendszert  
 \*/* glLineWidth(0.2);  
 glColor3f(0.5, 0.5, 0.5);  
 glBegin(**GL\_LINE\_STRIP**);  
 glVertex3f(-1.0, 0.0, 0.0);  
 glVertex3f(1.0, 0.0, 0.0);  
 glEnd();  
 glLineWidth(0.2);  
 glColor3f(0.5, 0.5, 0.5);  
 glBegin(**GL\_LINE\_STRIP**);  
 glVertex3f(0.0, -1.0, 0.0);  
 glVertex3f(0.0, 1.0, 0.0);  
 glEnd();  
  
  
 */\*  
 \* Megjelenitjuk a pontmatrix elemeit  
 \*/* glPointSize(6);  
 glColor3f(0.0, 1.0, 0.0);  
 glBegin(**GL\_POINTS**);  
 for (int i = 0; i < numberOfPoints; i++) {  
 glVertex3fv((GLfloat \*) &controlPoints[i][0]);  
 }  
 glEnd();  
  
  
 */\*  
 \* A megadott pontokbol kiszamoljuk és abrazoljuk a fuggveny tobbi pontjat is  
 \*/* glLineWidth(0.2);  
 glColor3f(1.0, 0.0, 0.0);  
 glBegin(**GL\_LINE\_STRIP**);  
 for (int i = 0; i < 200; i++) {  
 glVertex3f(curvePoints[i][0], curvePoints[i][1], 0.0);  
 }  
 glEnd();  
 glFinish();  
}

A program indulásakor több opció közül választhatunk, ezt a konzolban beadott inputokkal lehet kiválasztani. Az első demó egy 5 pontra ábrázolt görbét mutat, a második demó 14 pontra mutat görbét. Az egyéni pontok felvételére csak ilyenkor van lehetőség, később már nem, azonban a két demó görbét bármikor kirajzolja futás közben is a menü sorszámok választásával.

void printMessage() {  
 cout << "Grafika beadando 2023 - Gorbe rajzolas Lagrange polinom hasznalataval\n\r";  
 cout << "Egyud Vivien - C11M1L\n\r";  
 for (int k = 0; k < 51; k++) {  
 cout << "\*";  
 }  
 cout << "\n\r \* 0 - Kilepes";  
 cout << "\n\r \* 1 - Demo elore beallitott pontokkal (5)";  
 cout << "\n\r \* 2 - Demo elore beallitott pontokkal (14)";  
 cout << "\n\r \* 7 - Gorbe rajzolas sajat pontokra\n";  
 for (int k = 0; k < 51; k++) {  
 cout << "\*";  
 }  
 cout << "\n";  
}

Teljes forráskód

A teljes forráskód elérhető GitHub-on.

URL:

<https://github.com/vivoca/grafika>

QR kód:

Qr code

Description automatically generated