IV. gyakorlat - Jegyzőkönyv

Ekart Csaba - ZWPMKP

1. feladat

Az első feladatban egy olyan 2D-s OpenGL animációt kellet készíteni, melyben egy asztalon felülnézetben mutatott labda egyenes vonalú egyenletes mozgást végez, míg a képernyő széleiről tökéletesen rugalmasan visszapattan.

A megoldás során készítettem két két segédváltozót, melyek az x és y irányú sebességkomponensek voltak. Ezeknek adtam egy kezdeti értéket, hogy ferdén induljon el a labda, majd a Step függvényben megírtam a visszapattanás kódját.

Amennyiben valamelyik szélhez elér a megfelelő sebességkomponense a negatívjára változik.

```
79  void Step()
80  {
81      /*Move the ball, and response for collision*/
82
83      if ( ( (posX + radius) > sizeX ) ||  ( (posX - radius) < 0 ) )
84      {
85          speedX = -speedX;
86      }
87      if ( ( (posY + radius) > sizeY ) ||  ( (posY - radius) < 0 ) )
88      {
89          speedY = -speedY;
90      }
91
92      posX += speedX;
93      posY += speedY;
94      glutPostRedisplay();   //display scene
95  }
```

2. feladat

A feladatban az előzőekben elkészítettek alapján kellett ferde hajítást modellezni. A megoldás nem sok módosítást igényelt, csupán be kellett vezetni egy új változót a gravitációs gyorsulást, amellyel minden pillanatban csökken az y sebességkomponens.

```
30  double speedX = 0.04;
31  double speedY = 0.05;
32  double gravity = 0.0001;
33
```

3. feladat

A harmadik feladat során egy rugó mozgást kellett modellezni állítható rugóállandóval. Ehhez létrehoztam egy külön változót a rugoAllandot.

Az y irányú sebességet a példa kedvéért nullára állítottam, majd egy olyan elágazást írtam a megfelelő helyre, hogy amennyiben a gömb az ablak feléhez képest balra van jobb irányba gyorsuljon, amennyiben a jobbra a bal irányba. Így rugómozgást kapunk.

```
84      if ( posX < sizeX / 2 )
85      {
86          speedX += rugoAllando;
87      }
88      else
89      {
90          speedX -= rugoAllando;
91      }
92
93      posX += speedX;
94      posY += speedY;
```

Kiegészítésként maximum és minimum rugóállandó értéket állítottam be, hogy ne legyen probram futása közbe a rugállandó tetszőleges változtatásával (u és d gomb).

```
111 void Keyboard(unsigned char key, int x, int y) {
112
113     switch(key) {
114         case 'u':
115             if(rugoAllando < 0.001) rugoAllando += 0.00001;
116             break;
117         case 'd':
118             if(rugoAllando > 0.00002) rugoAllando -= 0.00001;
119             break;
120         case 27:
121             exit(0);
122             break;
123        default:
124        break;
125     }
126     glutPostRedisplay();
127 }
```

4. feladat

A negyedik feladatban egy a lagrange görbén futó autót kellett készíteni. A megfelelő változók létrehozása után, és a CalcLagrange() fv. meghívása utána az animáció elkészítésénél a szöget kelllett kiszámolni, hogy az autó mindig abba az irányba nézzen, ami nekünk megfelelő. Az animáció az “s” gomb leütésével indítható.

Az alábbi kódrészlez a fontos a projektben:

```
if (animate) {

    /*Draw car in correct position*/
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();

    int e = 0;
    int M_PI = 3;

    if ( L_autoX <= autoX && L_autoY >= autoY) {
        e = 2 * M_PI;
    } else if ( L_autoX >= autoX && L_autoY >= autoY) {
        e =  M_PI;
    } else if ( L_autoX >= autoX && L_autoY <= autoY) {
        e =  M_PI;
    }
    double a = L_autoY-autoY;
    double b = L_autoX-autoX;
    int angleAuto = ( atan( a / b) + e ) / M_PI * 180;

    glTranslated( autoX, autoY, 0 );
    glRotated( angleAuto, 0, 0, 1 );
```