به نام خدا

دانشگاه شهید بهشتی

دانشکده علوم ریاضی

گزارش تمرین سوم گرافیک (کار با کتابخانه openGL)

زهرا دهقانی تفتی (۹۶۲۲۲۰۳۷)

تاریخ تحویل :۱۹ اردیبهشت ۱۴۰۰

فهرست مطالب

تمرین سوم–ربات با ۳ بازو

یک Robot بسازید که بتواند یک شی را (که در فاصله دسترس قرار دارد) برداشته و در نقطه دیگر رها کند. Robot شما حداقل ۳ بازو دارد.

- شی مورد نظر می تواند یک کره باشد که درمکان مناسب قرار گرفته است.
- در ساده ترین حالت، بازوی شما فقط می تواند در یک صفحه حرکت کند.
- انتهای بازو باید برخورد با جسم مورد نظر را تشخیص دهد (به این معنی که مختصات بازو انتهایی با شی مورد نظر برابر شود) .سپس جسم توسط بازو به مکان دیگری منتقل شود و رها شود.
- اگر بازوی شما می خواهد در فضای ۳ بعدی در بیشتر از یک جهت چرخش داشته باشد، باید از دستور چرخش های متفاوت استفاده کنید که هر کدام جسم را در یک جهت می چرخاند. (همیشه چرخش دور یکی از محورهای مختصات انجام می شود)
- طراحی مساله بر عهده شما می باشد.
- برای کنترل حرکت بازو می تواند از TIMER یا KEYBOARD استفاده کنید.
- اضافه کردن جزئیات به طراحی (نور پردازی – اضافه کردن بافت) نمره مثبت خواهد داشت.

پاسخ:

در این تمرین ما بازویی متشکل از سه جز در نظر می‌گیریم. برای رسم هر قسمت از تابع آماده Glut که برای رسم مکعب است استفاده می کنیم سپس با تابع Scale آن را به اندازه مورد نظر برای خود در می‌آوریم. در این تمرین استفاده درست از تابع های glpushmatrix و glpopmatrix بسیار مهم است و در صورت استفاده‌ی غلط، هر جز از بازو حرکت مجزایی خواهد داشت و بهم متصل نیستند. نکته ای که در مورد این سوال وجود دارد آن است که با رسم هر مکعب، مرکز مکعب در مبدا مختصات قرار می‌گیرد و محور چرخش این مکعب نیز همین مرکز مکعب می شود در صورتی که ما می‌خواهیم چرخش در انتهای هر جز ایجاد شود پس باید با استفاده از تابع translate مرکز چرخش برای هر مکعب را تنظیم کنیم و برای این کار هر مکعب را به اندازه‌ی نصف طول خودش جابجا کنیم. بعد از رسم قسمت بالایی بازو، برای رسم قسمت میانی و قسمت پایینی بازو نیز باید مرکز چرخش تنظیم شود ونیز باید با استفاده از translate این اجزا طوری کنار هم قرار بگیرند که حالت مفصل ایجاد شود و برای مثال انتهای بازوی اول روی ابتدای بازوی میانی قرار بگیرد و فیکس شود تا موقع حرکت این اجزا از هم جدا نشوند. برای کنترل انیمیشن در قسمت اول این تمرین یعنی فقط جابجایی بازوها، برخلاف تمرین های قبل که از تایمر استفاده شد،از صفحه کلید کمک میگیریم. اما برای قسمت دوم سوال و برای جایجایی جسم با بازو از تایمر استفاده می‌کنیم و فرض می‌کنیم فقط بازوی پایینی تکان داده شود. برای برداشتن جسم توسط بازو باید بازو توانایی تشخیص تصادف با جسم را داشته باشد و هرگاه مختصات انتهای بازوی پایینی با مختصات جسم یکی شد آن را برداشته و در جای دیگر رها کند.

ابتدا کتابخانه های مورد نیاز را فراخوانی می کنیم.

```
#include <windows.h>  // for MS Windows
#include <GL/glut.h>  // GLUT, include glu.h and gl.h
#include <math.h>
```

سپس متغیرهای مورد نیاز را تعریف می‌کنیم که این متغیرها به ترتیب زاویه بازوی بالایی با محورx ، زاویه‌ی بازوی میانی با بازوی بالایی، زاویه بازوی پایینی با بازوی میانی، مختصات انتهای بازوی آخر، مختصات مرکز کره، فاصله ی انتهای بازوی آخر و مرکز کره از هم و تایم رفرش انیمیشن هستند.

```
/* Global variables */
char title[] = "Articulated Arm";
int partOneRotation = 0, partTwoRotation = 0, partThreeRotation = 0,n = 0;
float x_partThree, y_partThree, z_partThree, x_sphere = 0.68, y_sphere = 0, z_sphere =
0, distance;
int refreshMills = 415;        // refresh interval in milliseconds
```

سپس تابع initGl را برای تنظیماتی که فقط یکبار و در ابتدا باید انجام شوند مثل پاکسازی پنجره خروجی، تنظیمات مربوط به نورپردازی و متریال تعریف میکنیم.

```
void initGL() {
    glClearColor(0.0f, 0.0f, 0.0f, 1.0f); // Set background color to black and opaque
    glClearDepth(1.0f);                   // Set background depth to farthest
    glEnable(GL_DEPTH_TEST);   // Enable depth testing for z-culling
    glDepthFunc(GL_LEQUAL);    // Set the type of depth-test
    glShadeModel(GL_SMOOTH);   // Enable smooth shading


    GLfloat light_ambient[] = { 0.0f, 0.0f, 0.0f, 1.0f };
    GLfloat light_diffuse[] = { 1.0f, 1.0f, 1.0f, 1.0f };
    GLfloat light_specular[] = { 1.0f, 1.0f, 1.0f, 1.0f };
    GLfloat light_position[] = { 1.0f, 1.0f, 1.0f, 0.0f };

    GLfloat mat_ambient[] = { 0.0f, 1.0f, 0.3f, 1.0f };
    GLfloat mat_diffuse[] = { 0.8f, 0.8f, 0.8f, 1.0f };
    GLfloat mat_specular[] = { 0.5f, 1.0f, 1.0f, 1.0f };
    GLfloat high_shininess[] = { 100.0f };

    glLightfv(GL_LIGHT0, GL_AMBIENT, light_ambient);
    glLightfv(GL_LIGHT0, GL_DIFFUSE, light_diffuse);
    glLightfv(GL_LIGHT0, GL_SPECULAR, light_specular);
    glLightfv(GL_LIGHT0, GL_POSITION, light_position);

    glMaterialfv(GL_FRONT, GL_AMBIENT, mat_ambient);
    glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_diffuse);
    glMaterialfv(GL_FRONT, GL_SPECULAR, mat_specular);
    glMaterialfv(GL_FRONT, GL_SHININESS, high_shininess);

    glEnable(GL_LIGHT0);          //Enabling the light
    glEnable(GL_LIGHTING);
    glEnable(GL_COLOR_MATERIAL);

}
```

سپس در تابع display اجزای صحنه را مشخص می کنیم. این صحنه متشکل از جز ثابت که در سمت چپ بازو قرار دارد و بازوی بالایی به آن متصل است، بازوی بالایی، بازوی میانی، بازوی پایینی و کره است که کد مربوط به هرکدام را در زیر

می‌بینیم. فعلا در این کد نمی توانیم جسم را با استفاده از بازو جابجا کنیم و جسم ثابت است. کل کد مربوط به قسمت بازوها بین glpushmatrix و glpopmatrix قرار گرفته زیرا اگر اینکار را نکنیم هربار که صحنه از نو رسم می شود این تغییرات باعث بهم ریختن مکان بازوها و دوربین می شود.

```
// Displays the arm in its current position and orientation.  The whole
// function is bracketed by glPushMatrix and glPopMatrix calls because every
// time we call it we are in an "environment" in which a gluLookAt is in
// effect.  (Note that in particular, replacing glPushMatrix with
// glLoadIdentity makes you lose the camera setting from gluLookAt).
void display_Articulated_Arm() {

    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT); // Clear color and depth
buffers

    ////// Fixed part //////
    glPushMatrix();
    glTranslatef(-0.1, 0, 0);
    glScalef(0.2, 0.4, 1.0);
    glColor3f(1, 0, 0);
    glutSolidCube(1);
    glPopMatrix();

    // Draw the upper arm, rotated shoulder degrees about the z-axis.  Note that
    // the thing about glutWireBox is that normally its origin is in the middle
    // of the box, but we want the "origin" of our box to be at the left end of
    // the box, so it needs to first be shifted 0.15 (half of its length) unit in the
x direction, then
    // rotated.

    ////// First part //////

    glPushMatrix();

    glRotatef(partOneRotation,0,0,1);
    glTranslatef(0.15,0,0);         //set the pivot point of part one

    glPushMatrix();
    glScalef(0.3, 0.1, 1.0);
    glColor3f(1, 0, 1);
    glutSolidCube(1);
    glPopMatrix();

    ////// second part //////


    glTranslatef(0.15, 0, 0);          // we have to position middle arm at the end of
upper arm
    glRotatef(partTwoRotation, 0, 0, 1);
    glTranslatef(0.1, 0, 0);          //set the pivot point of part two

    glPushMatrix();
    glScalef(0.2, 0.06, 1.0);
    glColor3f(1, 1, 0);
    glutSolidCube(1);
    glPopMatrix();

    ////// last part //////
```

```
    glTranslatef(0.1, 0, 0);           // we have to position lower arm at the end of
middle arm
    glRotatef(partThreeRotation, 0, 0, 1);
    glTranslatef(0.09, 0, 0);         //set the pivot point of part three

    glPushMatrix();
    glScalef(0.18, 0.035, 1.0);
    glColor3f(0, 1, 0);
    glutSolidCube(1);
    glPopMatrix();

    glPopMatrix();

    ////// sphere //////

    glPushMatrix();

    glTranslatef(x_sphere, y_sphere, z_sphere); هربار کره در جایی که مختصات انرا می دهیم رسم می شود که
باعث تکان خوردن کره می شود
    glScalef(0.05,0.05,0.05);
    glColor3f(1,1,1);
    glutSolidSphere(1, 16, 16);

    glPopMatrix();


    glutSwapBuffers();  // Swap the front and back frame buffers (double buffering)

}
```

در صورتی که بخواهیم برای کنترل انیمیشن از تایمر استفاده شود(در قسمت دوم سوال برای تکان دادن جسم با بازوی انتهایی) از تابع زیر استفاده می کنیم ولی ما در این تمرین از صفحه کلید استفاده می‌کنیم.این تابع به اینصورت است که ابتدا مختصات انتهای بازوی آخر را با استفاده از طول آن بازو و زاویه ای که آن بازو تشکیل می دهد محاسبه می کنیم، سپس فاصله نقطه انتهای بازو را با مرکز کره بدست می آوریم اگر برابر با صفر بود و بر هم منطبق بودند که بازو این جسم را حرکت می دهد و مختصات مرکز کره برابر با مختصات بازو می شود در طی چرخش های مختلف تحت زاویه های مختلف و پس از ۳ بار تکرار دیگر کره را جابجا نمی کنیم و در همان جا رهایش می کنیم. در غیرایصورت و مساوی نشدن مختصات بازو و کره، کره تغییر مکان نمی دهد. (البته این کار نمی کند و کره تکان نمی خورد و تنها بازوی انتهایی میچرخد.)

```
void timer(int value) {

    //Coordinates of the end of the last part
    x_partThree = 0.5 + 0.18 * cos(partThreeRotation);
    y_partThree = 0.18 * sin(partThreeRotation); //doroste????
    z_partThree = 0;

    distance = sqrt(pow(x_partThree - x_sphere, 2) + pow(y_partThree - y_sphere, 2) +
    pow(z_partThree - z_sphere, 2));

    if (distance == 0 && n <=3) {
        x_sphere = x_partThree;
        y_sphere = y_partThree;
        z_sphere = z_partThree;
    }

    partThreeRotation = (partThreeRotation + 2) % 360;
    n += 1;
```

```
    glutPostRedisplay();      // Post re-paint request to activate display()
    glutTimerFunc(refreshMills, timer, 0); // next timer call milliseconds later
}
```

سپس تابع reshape که حاوی اطلاعات دوربین و تغییر اندازه صفحه است را تعریف می‌کنیم.

```
void reshape(GLsizei width, GLsizei height) {  // GLsizei for non-negative integer

    glViewport(0, 0, (GLsizei)width, (GLsizei)height);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
}
```

سپس برای کنترل انیمیشن از تابع زیر برای استفاده از صفحه کلید بهره می‌بریم!(برای قسمت اول سوال و جابجا شدن هر سه بازو)

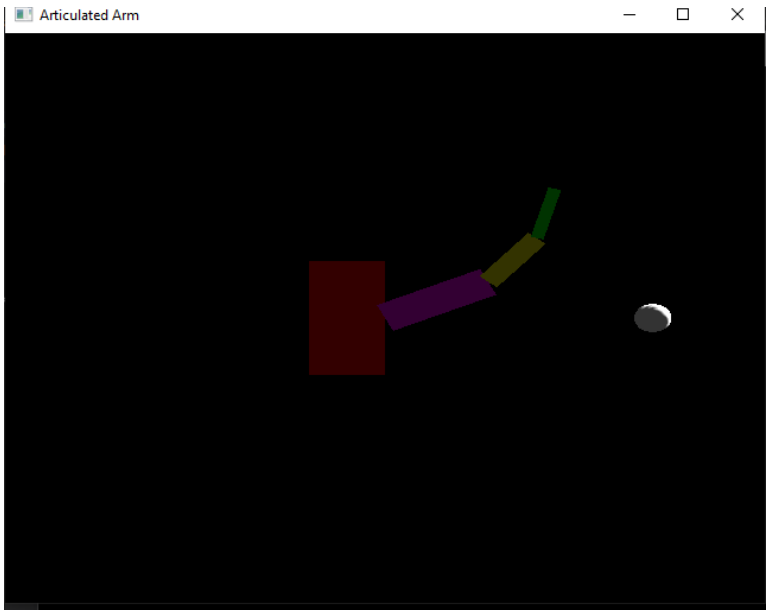
```
void keyboard(unsigned char key, int x, int y) {
    switch (key) {
        //Shoulder
    case 's':
        partOneRotation = (partOneRotation + 5) % 360;
        glutPostRedisplay();
        break;
    case 'S':
        partOneRotation = (partOneRotation - 5) % 360;
        glutPostRedisplay();
        break;
        //Elbow
    case 'e':
        if (partTwoRotation < 180)
            partTwoRotation = (partTwoRotation + 5) % 360;
        glutPostRedisplay();
        break;
    case 'E':
        if (partTwoRotation > 0)
            partTwoRotation = (partTwoRotation - 5) % 360;
        glutPostRedisplay();
        break;
        //Hand
    case 'h':
        if (partThreeRotation < 90)
            partThreeRotation = (partThreeRotation + 5) % 360;
        glutPostRedisplay();
        break;
    case 'H':
        if (partThreeRotation > 0)
            partThreeRotation = (partThreeRotation - 5) % 360;
        glutPostRedisplay();
        break;
    }


}
```

سپس تابع main را تعریف می‌کنیم. در تابع main اطلاعات مربوط به پنجره خروجی را داریم. در این تابع باید از تابع هایی که در بدنه برنامه معرفی کردیم استفاده کنیم.برای هموارتر کردن انیمیشن از دو بافر استفاده می شود

```
int main(int argc, char** argv) {
    glutInit(&argc, argv);            // Initialize GLUT
    glutInitDisplayMode(GLUT_DOUBLE); // Enable double buffered mode
    glutInitWindowSize(640, 480);   // Set the window's initial width & height
    glutInitWindowPosition(50, 50); // Position the window's initial top-left corner
    glutCreateWindow(title);          // Create window with the given title
    glutDisplayFunc(display_Articulated_Arm);       // Register callback handler for
window re-paint event
    glutReshapeFunc(reshape);       // Register callback handler for window re-size
event
    initGL();                       // Our own OpenGL initialization
    //glutTimerFunc(0, timer, 0);     //درصورتی که بخواهیم از تایمر استفاده کنیم از حالت کامنت خارج میکنیم.
    glutKeyboardFunc(keyboard);          //Getting input from keyboard !
    glutMainLoop();                 // Enter the infinite event-processing loop
    return 0;
}
```

خروجی:

مربوط به قسمت اول سوال که هر سه بازو میچرخند: همانطور که انتظار داشتیم انتهای بازوها بهم متصل اند و در هنگام حرکت همگی باهم جابجا می‌شوند . در این تمرین از نورپردازی و متریال استفاده شده. و می توانیم با استفاده از صفحه کلید حرکت هرکدام از بازوها را کنترل کنیم.



شکل زیر مربوط به قسمت دوم سوال است که فقط بازوی انتهایی جابجا می شود و باید بتواند توپ را جابجا کند ولی کار نمی کند.