作业二 绘制一个沿固定线路运动的机器人

18340012 陈晨

1 实验题目

- **必要：**

  a) 线路可以是圆或任意其它闭合路径

  b) 机器人在任意时刻应面向运动曲线的切线方向

  c) 机器人应该有头、躯干、四肢等基本部分

  d) 机器人在运动过程中应具有摆臂及抬脚两个基本动作

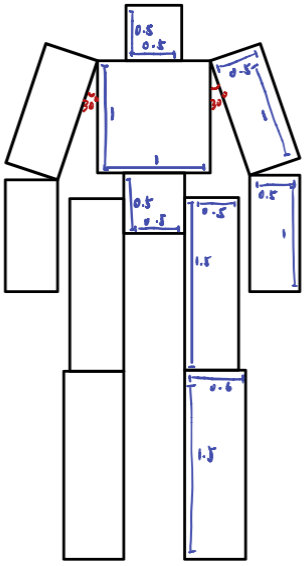
  可选：

  e) 使用 mesh 模型（如 obj 文件）载入机器人模型

  f) 或载入其他 mesh 模型围绕机器人运动

2 实验过程和核心代码

设计一个机器人构造如下。机器人的每部分都由立方体组成。身体的厚度为1，其余身体部分厚度为0.5（除了小腿为0.6）。



首先，设置投影矩阵。为了直观地显示机器人的远近，这里用gluPerspective创建一个透视型视景体。

```
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluPerspective(60.0f, (GLfloat)width() / (GLfloat)height(), 0.1f,
200.0f);
```

对于机器人的整体，这里设计为沿着一个圆运动。由于每过一段时间调用一次paintGL()，因此可以声明一个静态变量来记录当前旋转角度，每次调用paintGL()，该变量的值随之发生改变。

```
    //每次调度函数，角度更新
    static float angle = 0.0f;
    float delta = 1.2f;
    angle += delta;
    if (angle >= 360.0f) angle = 0.0f;
```

对于沿着圆运动，可以先将机器人平移，再沿着y轴旋转。这里为了观察方便，将旋转后的机器人又往z轴负方向平移了-15.0f。

```
    //绘制沿着圆移动的机器人
    glPushMatrix();
    glTranslatef(0.0f, 0.0f, -15.0f);
    glRotatef(angle, 0.0f, 1.0f, 0.0f);
    glTranslatef(3.0f, 0.0f, 0.0f);
    drawRobot(angle);
    glPopMatrix();
```

实现drawRobot函数之前，先实现一个画单个立方体的函数如下。

```
void drawBox() {
    glPushMatrix();
    glColor3f(1.0f, 1.0f, 1.0f);
    glBegin(GL_QUAD_STRIP);
    glVertex3f(0.0f, 0.0f, 0.0f);
    glVertex3f(1.0f, 0.0f, 0.0f);
    glVertex3f(0.0f, 0.0f, 1.0f);
    glVertex3f(1.0f, 0.0f, 1.0f);
    glVertex3f(0.0f, 1.0f, 1.0f);
    glVertex3f(1.0f, 1.0f, 1.0f);
    glVertex3f(0.0f, 1.0f, 0.0f);
    glVertex3f(1.0f, 1.0f, 0.0f);
    glVertex3f(0.0f, 0.0f, 0.0f);
    glVertex3f(1.0f, 0.0f, 0.0f);
    glVertex3f(0.0f, 0.0f, 0.0f);
    glVertex3f(0.0f, 1.0f, 0.0f);
    glVertex3f(0.0f, 0.0f, 1.0f);
    glVertex3f(0.0f, 1.0f, 1.0f);
    glVertex3f(1.0f, 0.0f, 0.0f);
    glVertex3f(1.0f, 1.0f, 0.0f);
    glVertex3f(1.0f, 0.0f, 1.0f);
    glVertex3f(1.0f, 1.0f, 1.0f);
    glEnd();
    glPopMatrix();
}
```

然后实现画机器人整体的函数drawRobot。通过调用drawBox函数，并且进行一些变换操作，可以组成机器人的身体的各个部分。考虑到机器人的摆臂、抬腿角度可以和机器人整体的旋转角度相关联，这里需要传入参数angle。除此之外，考虑到动作的自然性，这里把上半身考虑为一个整体，随着机器人的移动添加了左右旋转的变换。

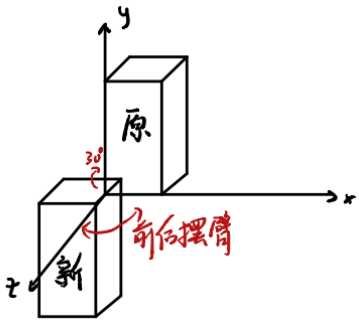
对于机器人的身体，只调用drawBox函数，并以此为坐标参考画出其他部分。对于上半身的整体，左右转动即沿y轴旋转，由于身体的中心在(0.5f, 0.5f, 0.5f)，这里需要先平移(-0.5f, 任意, -0.5f)，旋转，再复原。这里声明一个变量angle1，angle1的变化随着angle变化而变化，变化方式是从0°到90°，再从90°到0°。angle1用于接下来的所有旋转。

```
    float angle1 = angle;
    if (angle1 > 180.0f) angle1 -= 180.0f;
    if (angle1 > 90.0f)angle1 = 180.0f - angle1;

    //上半身左右移动，故头、身体、左右手看成一个整体
    glPushMatrix();
    glTranslatef(0.5f, 0.0f, 0.5f);
    glRotatef((-angle1 + 45.0f) * 0.8f, 0.0f, 1.0f, 0.0f);
    glTranslatef(-0.5f, 0.0f, -0.5f);

    //头
    ...

    //身体
    glPushMatrix();
    drawBox();
    glPopMatrix();

    //左手
    ...

    //右手
    ...

    glPopMatrix();//上半身结束
```

头的部分实现如下。先进行缩放，再根据坐标关系平移即可。由于立方体本身是过原点的，所以先进行缩放会比较方便计算坐标。

```
    //头
    glPushMatrix();
    glTranslatef(0.25f, 1.0f, 0.25f);
    glScalef(0.5f, 0.5f, 0.5f);
    drawBox();
    glPopMatrix();
```

手臂的部分实现如下（这里以左臂为例）。首先左大臂和身体相隔30°，整个左手作为一个整体进行前后的摆臂（沿x轴旋转），小臂在此基础之上，沿(1.0f, 0.0f, 1.0f)摆臂。对于左手的整体，先平移至如下图所示的位置，再进行所有的旋转，最后先平移恢复，再平移到以身体为参照的对应位置。这里为了代码可读性，将最后两步平移分开写。



小臂的旋转实现与上述方式类似。

```
    //左手
    glPushMatrix();
    glTranslatef(-0.5f, -0.25f, 0.25f);
```

```
    glTranslatef(0.5f, 1.0f, 0.0f);
    glRotatef(angle1 - 45.0f, 1.0f, 0.0f, 0.0f);
    glRotatef(-30.0f, 0.0f, 0.0f, 1.0f);
    glTranslatef(-0.5f, -1.0f, 0.0f);
    //左手大臂
    glPushMatrix();
    glScalef(0.5f, 1.0f, 0.5f);
    drawBox();
    glPopMatrix();
    //左手小臂
    glPushMatrix();
    glTranslatef(0.0f, -1.0f, 0.0f);
    glTranslatef(0.0f, 1.0f, 0.0f);
    glRotatef(angle1, 1.0f, 0.0f, 1.0f);
    glTranslatef(0.0f, -1.0f, 0.0f);
    glScalef(0.5f, 1.0f, 0.5f);
    drawBox();
    glPopMatrix();
    glPopMatrix();//左手结束
```

而对于下半身，并不需要左右旋转，因此逐个实现即可。腿的实现方式和手类似，腿作为一个整体前后旋转(沿x轴)，小腿在此基础上也前后旋转。

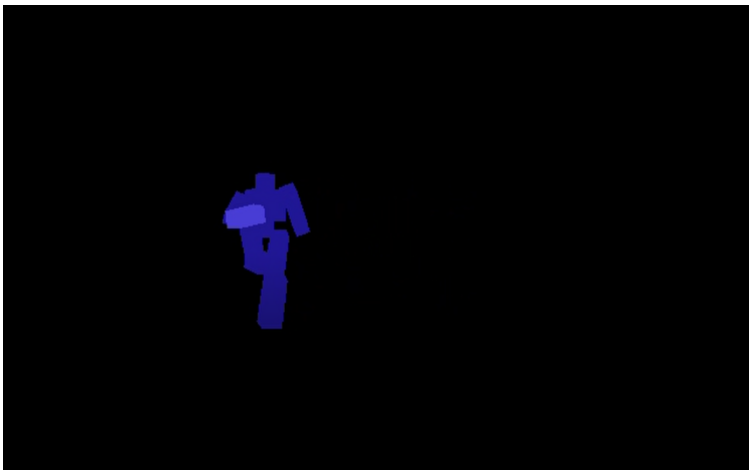
```
    //胯
    glPushMatrix();
    glTranslatef(0.25f, -0.5f, 0.25f);
    glScalef(0.5f, 0.5f, 0.5f);
    drawBox();
    glPopMatrix();

    //左腿
    glPushMatrix();
    glTranslatef(-0.25f, -1.75f, 0.25f);
    glTranslatef(0.0f, 1.5f, 0.0f);
    glRotatef((-angle1 + 45.0f) * 0.75f, 1.0f, 0.0f, 0.0f);
    glTranslatef(0.0f, -1.5f, 0.0f);
    //左腿大腿
    glPushMatrix();
    glScalef(0.5f, 1.5f, 0.5f);
    drawBox();
    glPopMatrix();
    //左腿小腿
    glPushMatrix();
    glTranslatef(-0.1f, -1.5f, 0.0f);
    glTranslatef(0.5f, 1.5f, 0.0f);
    glRotatef(angle1 * 0.5f, -1.0f, 0.0f, 0.0f);
    glTranslatef(-0.5f, -1.5f, 0.0f);
    glScalef(0.6f, 1.5f, 0.6f);
    drawBox();
    glPopMatrix();
    glPopMatrix();//左腿结束

    //右腿
    ...//与左腿类似
```

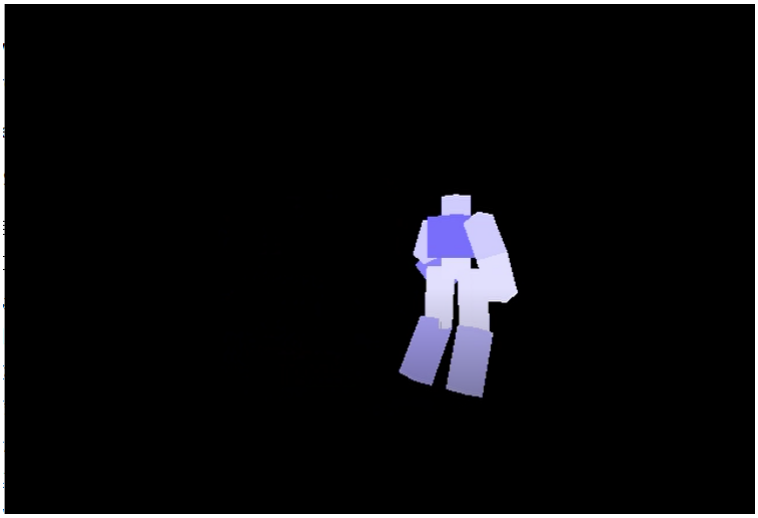
最后，为了方便观察，设置光照、材质。

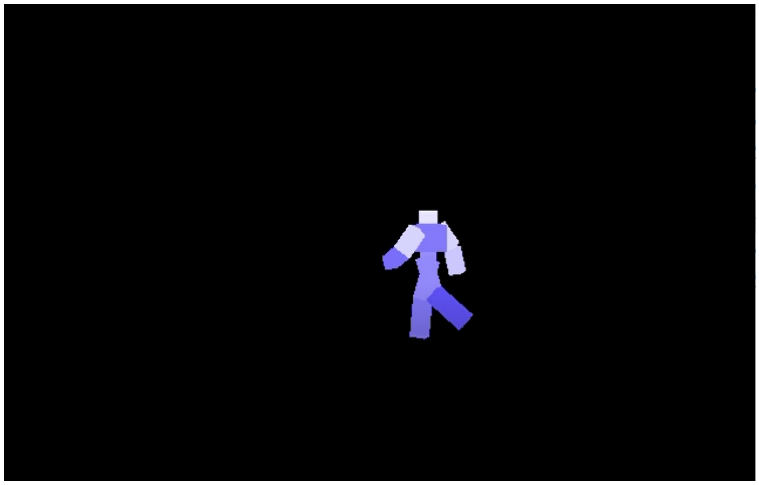
```
    glEnable(GL_LIGHTING);
    glEnable(GL_LIGHT0);

    GLfloat ambient[] = { 1.0f, 1.0f, 1.0f, 1.0f };
    GLfloat diffuse[] = { 1.0f, 1.0f, 1.0f, 1.0f };
    GLfloat specular[] = { 0.5f, 0.5f, 0.5f ,1.0f };

    glLightfv(GL_LIGHT0, GL_AMBIENT, ambient);
    glLightfv(GL_LIGHT0, GL_DIFFUSE, diffuse);
    glLightfv(GL_LIGHT0, GL_SPECULAR, specular);

    GLfloat position[] = { 0.0f, 0.0f, 10.0f, 0.0f };
    glLightfv(GL_LIGHT0, GL_POSITION, position);

    GLfloat blue[] = { 0.1f, 0.1f, 0.5f, 1.0f };
    GLfloat white[] = { 1.0f, 1.0f, 1.0f, 1.0f };

    glMaterialfv(GL_FRONT, GL_AMBIENT, blue);
    glMaterialfv(GL_FRONT, GL_DIFFUSE, blue);
    glMaterialfv(GL_FRONT, GL_SPECULAR, white);
    glMaterialf(GL_FRONT, GL_SHININESS, 1.0f);
```

3 实验结果

实验结果如下图所示（文件夹内附视频）。机器人具有头、躯干、四肢，沿着闭合圆运动，在任意时刻面向圆的切线方向运动，且运动过程中具有摆臂及抬脚的动作。







4 实验感想

在本次实验中，主要的难点在于如何对机器人的每个部件进行变换。由于机器人的每个部分是沿着不同的轴进行旋转的，这就需要先平移到合适的位置，进行旋转，再平移到应到的地方。除此之外，也要有整体和部分的概念，有一些变换是基于整体的，而有些是基于部分的，这中间就需要通过矩阵的出入栈来实现。