山东大学计算机图形学实验报告

实验一 walking stick man

郭奚源 201400301235

1. 实验要求
   1. 使用OPENGL库函数 画一个由矩形立方体组成的小人（手臂和腿部均由两个立方体组成）
   2. 通过translate和rotate 操作，实现小人的行走（行走过程中不能出现立方体之间散架的情况）
   3. 行走过程中手和腿的摆动通过调参数显得自然
2. 完成情况
   1. 由于时间关系，手臂用一个立方体表示，所以在行走过程中，**腿部可以出现弯曲，而手臂只能直摆。**最终小人由**头，躯干，左右臂，左右大腿，左右小腿，左右脚**8个部分组成
   2. **实现了通过W,S,A,D键盘操作控制小人移动（包括360转向以及向前向后走），**并且通过将Z轴的值和translate和rotate关联**实现了近小远大的透视效果。**
   3. 通过调参数，**实现了左右臂向相反方向摆动，并且摆动到30度自动换向；大腿和左右臂摆动同步，左右大腿分别对应右左手臂；大腿向前抬的时候小腿适当往相反方向抬，符合运动规律，显得更加自然。**

三．具体实验过程。

1.配置opengl库函数（就不赘述过程）：耗时1小时左右

2．学习具体需要使用的相关函数（下面是例子）：glPushMatrix();glPopMatrix();

**画小人**：glScalef(0.5, 0.5, 0.5); glutSolidCube(1.0); glColor3f(1.0, 0.6, 0.4);

**运动：**glTranslatef(0.0, 1.25, 0.0); glRotatef((GLfloat)turn, 0.0, 1.0, 0.0);

耗时1小时左右

3.构思需要的变量：1）x,y,z坐标；2）转向:turn（当前方向）和turn1（转向）；3）向前：forward；4） 控制关节：shoulder1, shoulder2, elbow,tag(用来记录左右臂是否已经

摆动到30度);5)由于有键盘事件，所以添加isStop变量（boolean）来控制小人的运动和停止

4.实现小人的构建和相应的运动

a).实现机器人前进

glPushMatrix();

glTranslatef(forward, 0.0, **z**);//Z实现近小远大的透视，后面的z将不再注释

glRotatef((GLfloat)turn, 0.0, 1.0, 0.0);

b)构建小人各部分以及实现相应运动（下面以右大腿，右小腿，躯干为例子 其他的实现在完整代码里，由于其余部分代码基本相同，在此就不赘述）

右大腿的运动和实现

glTranslatef(0.375, 0.0, 0.0)；

glRotatef(**(GLfloat)shoulder2**, 1.0, 0.0, 0.0);//右大腿的旋转和左手臂相对应

glTranslatef(0.0, -0.5, 0.0);

glColor3f(0.8, 1.0, 0.2);//各个部分颜色不同更易区分

glPushMatrix();

glScalef(0.5, 1.0, 0.5);//将手臂拉成长矩形立方体

glutSolidCube(1.0);

glPopMatrix();

右小腿的运动和实现

glTranslatef(0.0, -0.5, 0.0)；

glRotatef((GLfloat)elbow, 1.0, 0.0, 0.0);//用elbow实现小腿与大腿反向运动，后面会给出具体的elbow和shoulder的关系

glTranslatef(0.0, -0.5, 0.0);

glColor3f(0.5, 0.1, 0.8);

glPushMatrix();

glScalef(0.5, 1.0, 0.5);

glutSolidCube(1.0);

glPopMatrix();

躯干的运动和实现（并无特殊之处）

glTranslatef(0.0, 1.0, 0.0);

glColor3f(0.2, 0.2, 0.2);

glPushMatrix();

glScalef(1.4, 2.0, 0.5);

glutSolidCube(1.0);

glPopMatrix();

5．实现Keyboard控制函数：

**思想：捕捉键盘事件，通过switch判断WSAD，进行相应运动。**

下面以前进W和左转A为例子。

前进W:

case 'w':

turn1 = turn;// W不改变方向

forward = forward - 0.04\*sin((GLfloat)turn1 / 360 \* 3.14 \* 2);

**z = z - 0.05\*cos((GLfloat)turn1 / 360 \* 3.14 \* 2);//根据此时偏转的角度，得到z轴变换函数**

**if (tag == 0) { //右手向前，左手向后**

shoulder1 = (shoulder1 + 1);

shoulder2 = (shoulder2 – 1);

**if (shoulder1 >= 0) { elbow = elbow – 1.2; }//如果右手向前摆，则左小腿向后摆**

else { elbow = elbow + 1.2; }

}

else

{

shoulder1 = (shoulder1 - 1);

shoulder2 = (shoulder2 + 1);

if (shoulder1 >= 0) { elbow = elbow + 1.25; }

else { elbow = elbow - 1.2; }

}

**if (shoulder1>30) {**

**tag = 1;**

**}**

**if (shoulder1<-30) {**

**tag = 0;**

**}//通过tag记录双手摆动是否应该换向**

IsStop = true;**// 键盘事件结束，人物停止**

glutPostRedisplay();

break;

**PS:再向后走S的时候，有一个唯一不一样的地方在于，向后的时候小腿是不动的（跟大腿是一个整体，不单独旋转），所以不需要改变elbow的值**

左转A:

case 'a'://左转

**turn = (turn + 5) % 360;//，每次转5度**

glutPostRedisplay();

IsStop = true;

break;

6.实现最后的reshape函数和main函数

**思想：由于小人的大小及其移动范围的大小需要与最后的创建的窗口大小匹配所以设置参数w,h作为最后窗口大小的输入，实现对于小人的reshape。**

由于是第一次写关于reshape的函数，很多方法都不熟悉，所以借鉴了CSDN上的经验，设置了VIEWPOINT等。因为reshape的代码多为借鉴，所以就不放在实验报告中，具体的实现在具体代码中。

Main:

int main(int argc, char\*\* argv)

{

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);

glutInitWindowSize(800, 600);

glutInitWindowPosition(100, 100);

glutCreateWindow("walking robots!!");

glutDisplayFunc(display);

glutReshapeFunc(reshape);

glutKeyboardFunc(keyboard);

**glutMainLoop();//实现连续键盘操作**

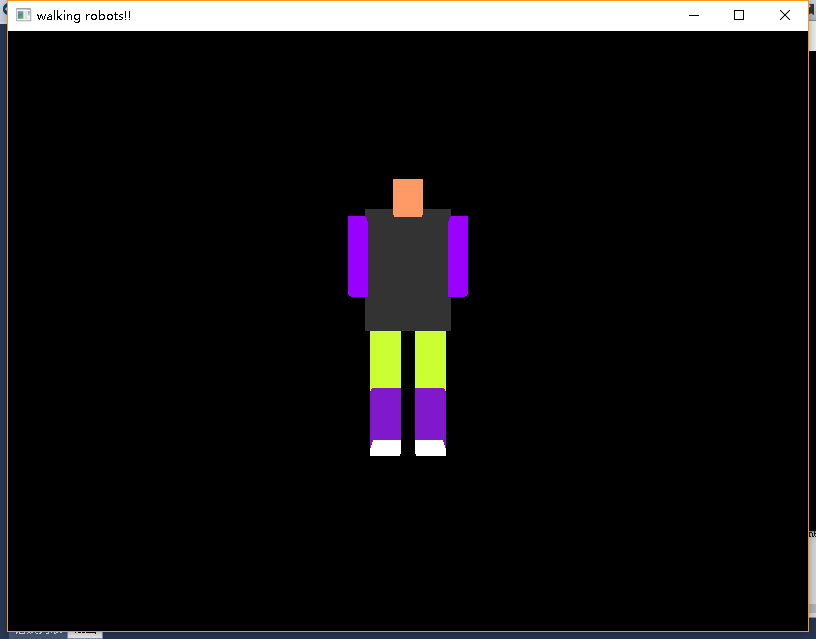
return 0;

}

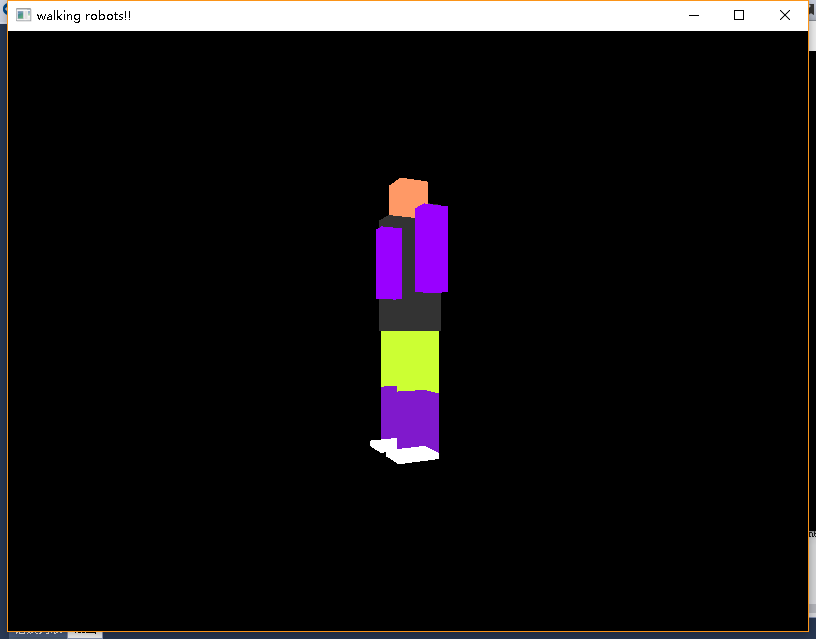
**到这里编程任务就全部结束了，整个编程任务包括之前的变量设计，之后的方法实现，以及过程中变量的修改和添加还有最后的debug过程，持续将近三天。**

1. 实验结果（报告中结果以图片形式展示，在GitHub中另有视频展示）：

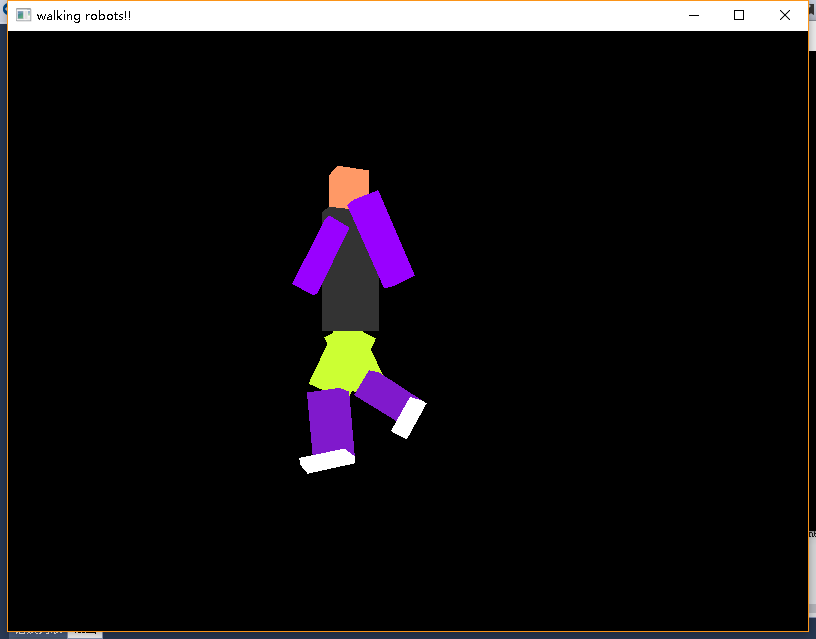
1.小人正面



2.小人转向后的侧面



3.前进过程中的运动状态



4.近小远大效果

