void Diffuse\_color(float mat\_diffuse,float light\_diffuse,float \*\*\* object\_color,int \* lightpos )

//此函数用来计算漫反射的颜色，输入为材料漫反射系数、光源漫反射系数、物体的位置，物体每一个像素对应的基本颜色，光源的位置

//通过修改全局变量数组中的每一个变量来实现函数作用的实现

//调用这个方法的时候先将整个diffuse\_color存储到颜色数组中，后两个函数将分别的将颜色值加上去

{

float diffuse\_color = 0;

//这里假定base\_color是一个全局变量且物体是一个基本颜色为这个数值的匀色物体

diffuse\_color=base\_color\*mat\_diffuse\*light\_diffuse;

int \*vec\_pot\_light=new int[3];

int \*nor\_vec=new int[3];

float coefficient;

for(int j =0;j<10;j++)

{

for(int k=0;k<10;k++)

{

//计算x=10的那个平面

//计算该店法线点乘该点到光源的单位向量，如果该值小于0，那么就用0代替此次结果

//ijk分别是这个点的三个维度上对用的坐标

vec\_pot\_light[0]=lightpos[0]-9;

vec\_pot\_light[1]=lightpos[1]-j;

vec\_pot\_light[2]=lightpos[2]-k;

//vet\_pot\_light数组现在存储的是方向向量，但是不是单位向量

//如果投影的是长方体的话，那么法向量就很容易的可以到，这里假设已经得到了 该点的法向量并且法向量为nor\_vec,

nor\_vec[0]=1;

nor\_vec[1]=0;

nor\_vec[2]=0;

coefficient=vec\_pot\_light[0]\*nor\_vec[0]+vec\_pot\_light[1]\*nor\_vec[1]+vec\_pot\_light[2]\*nor\_vec[2];

coefficient=coefficient/sqrt(pow(vec\_pot\_light[0],2)+pow(vec\_pot\_light[1],2)+pow(vec\_pot\_light[2],2));

if (coefficient<=0)

coefficient=0;

object\_color[9][j][k]=diffuse\_color\*coefficient;

//计算y=10的那个平面

vec\_pot\_light[0]=lightpos[0]-j;

vec\_pot\_light[1]=lightpos[1]-9;

vec\_pot\_light[2]=lightpos[2]-k;

//vet\_pot\_light数组现在存储的是方向向量，但是不是单位向量

//如果投影的是长方体的话，那么法向量就很容易的可以到，这里假设已经得到了 该点的法向量并且法向量为nor\_vec,

nor\_vec[0]=0;

nor\_vec[1]=1;

nor\_vec[2]=0;

coefficient=vec\_pot\_light[0]\*nor\_vec[0]+vec\_pot\_light[1]\*nor\_vec[1]+vec\_pot\_light[2]\*nor\_vec[2];

coefficient=coefficient/sqrt(pow(vec\_pot\_light[0],2)+pow(vec\_pot\_light[1],2)+pow(vec\_pot\_light[2],2));

if (coefficient<=0)

coefficient=0;

object\_color[j][9][k]=diffuse\_color\*coefficient;

//计算z=10的那个平面

vec\_pot\_light[0]=lightpos[0]-j;

vec\_pot\_light[1]=lightpos[1]-k;

vec\_pot\_light[2]=lightpos[2]-9;

//vet\_pot\_light数组现在存储的是方向向量，但是不是单位向量

//如果投影的是长方体的话，那么法向量就很容易的可以到，这里假设已经得到了 该点的法向量并且法向量为nor\_vec,

nor\_vec[0]=0;

nor\_vec[1]=0;

nor\_vec[2]=1;

coefficient=vec\_pot\_light[0]\*nor\_vec[0]+vec\_pot\_light[1]\*nor\_vec[1]+vec\_pot\_light[2]\*nor\_vec[2];

coefficient=coefficient/sqrt(pow(vec\_pot\_light[0],2)+pow(vec\_pot\_light[1],2)+pow(vec\_pot\_light[2],2));

if (coefficient<=0)

coefficient=0;

object\_color[j][k][9]=diffuse\_color\*coefficient;

}

}

}