第三章作业 王晨曦

Created @February 27, 2022 9:07 PM

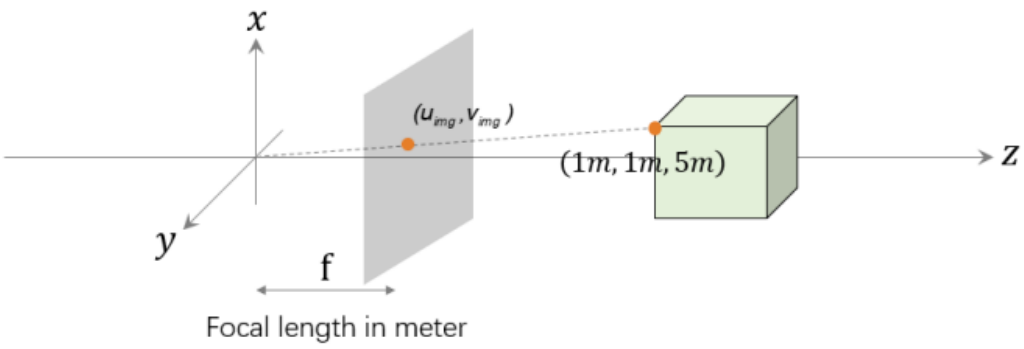
姓名：王晨曦

班级：计算机96

学号：2196123413

得分：

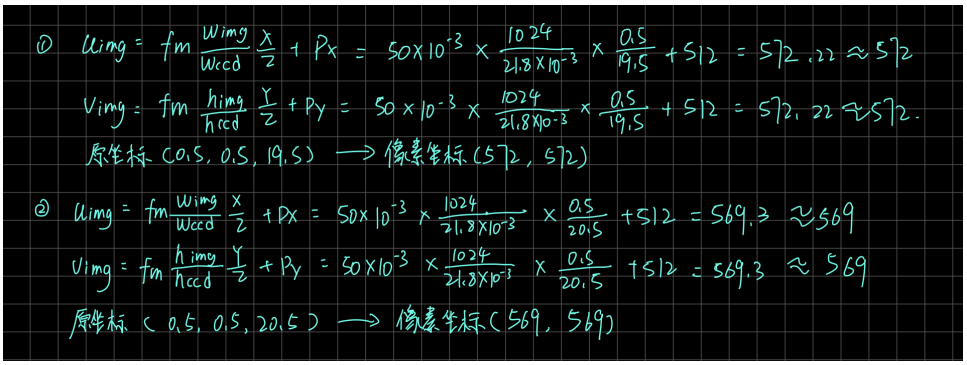
1. **如果摄像机的有效焦距是f=50mm，CCD的尺寸为21.8mm*21.8mm，所成的图像大小是1024*1024，光心在图像中的位置为512*512，试问在20m外放置一个边长为1m的立方体，8个顶点的坐标分别为**



$$(0.5,0.5,19.5),(0.5,0.5,20.5)$$
$$(-0.5,0.5,19.5),(-0.5,0.5,20.5)$$
$$(-0.5,-0.5,19.5),(-0.5,-0.5,20.5)$$
$$(0.5,-0.5,19.5),(0.5,-0.5,20.5)$$

- **计算这8个顶点在图像平面的像素值**

  答：我先按照公式手算了两个坐标



由上图显而易见，像素计量的焦距fx和fy的值均为2348.624左右

然后我发现其实我可以写个代码批量处理坐标，不用挨个手算，于是有了下面的结果：

In [20]:
```
def Entity2pixel(entity):
    pixel=[]
    pixel.append(entity[0] * 2348.624 / entity[2]+512)
    pixel.append(entity[1] * 2348.624 / entity[2]+512)
    print(pixel)
```

In [21]:
```
data = [[0.5, 0.5, 19.5], [0.5, 0.5, 20.5],
        [-0.5, 0.5, 19.5], [-0.5, 0.5, 20.5],
        [-0.5, -0.5, 19.5], [-0.5, -0.5, 20.5],
        [0.5, -0.5, 19.5], [0.5, -0.5, 20.5]]
```

In [22]:
```
for i in data:
    Entity2pixel(i)
```

```
[572.2211282051283, 572.2211282051283]
[569.283512195122, 569.283512195122]
[451.7788717948718, 572.2211282051283]
[454.71648780487806, 569.283512195122]
[451.7788717948718, 451.7788717948718]
[454.71648780487806, 454.71648780487806]
[572.2211282051283, 451.7788717948718]
[569.283512195122, 454.71648780487806]
```

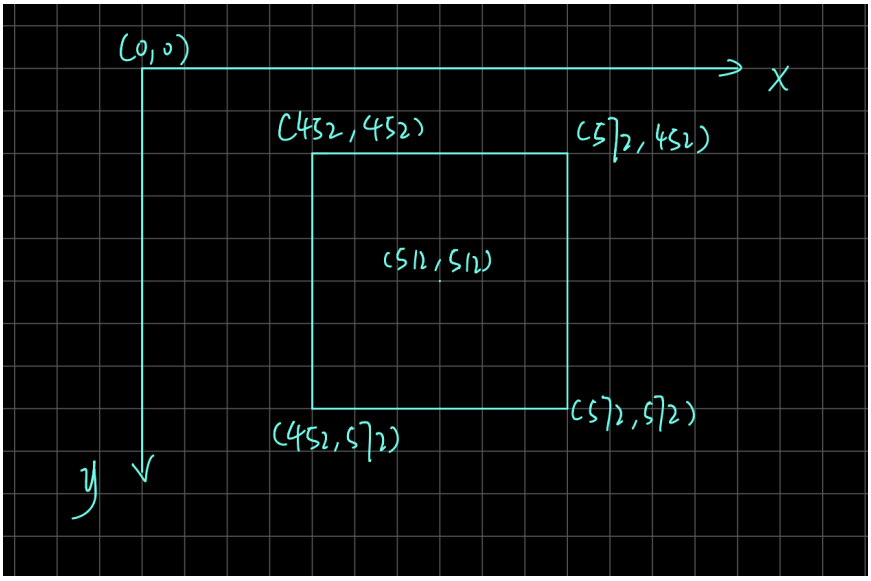
由于像素是离散的，因此需要对上面的坐标四舍五入一下，因此可以得到最终答案：

(572, 572), (569, 569), (452, 572), (455, 569)

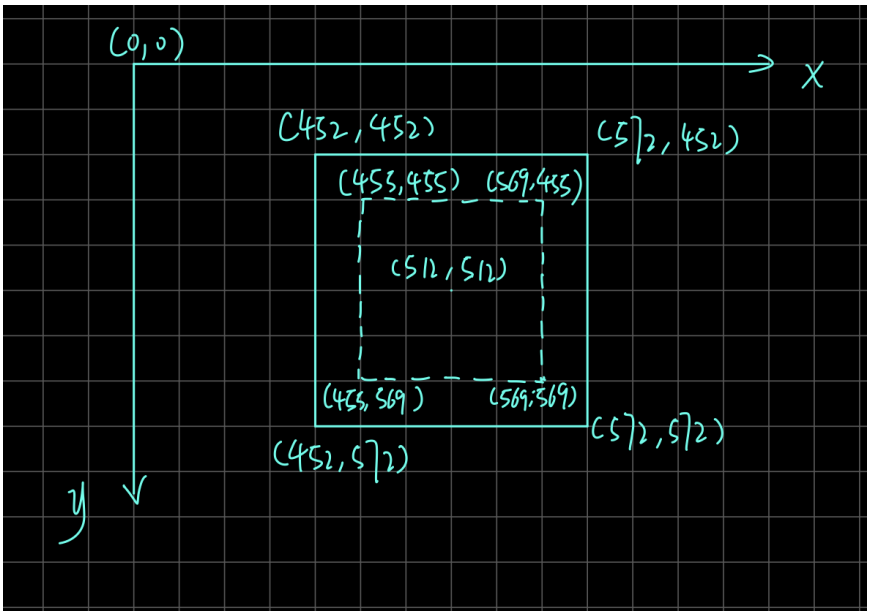
(452, 452), (455, 455), (572, 452), (569, 455)

- **分析这8个顶点的遮挡关系**

  答：首先在像素平面上画出立方体上Z轴坐标为19.5的四个点的位置：



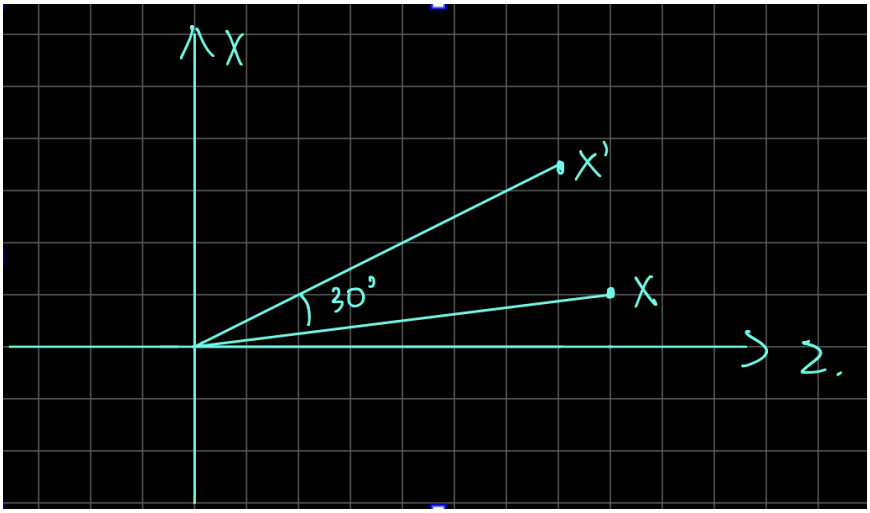
  接下来画出立方体上Z轴坐标为20.5的四个点的位置：



  显而易见，虚线连接的四个点构成的四边形小于实线连接的四个点构成的四边形，由此可知，Z轴坐标为20.5的四个点被Z轴坐标为19.5的四个点遮挡。

- **如果这立方体绕着y轴旋转30°，尝试计算它们在三维空间的新的坐标和成像**

  答：首先假设是由于是绕y轴旋转，默认右手系法则，因此对每个点有如下图的坐标变换：



$$x = r\cos\alpha, \quad y = r\sin\alpha ,$$

$$x_1 = r\cos(\alpha-\theta), \quad y_1 = r\sin(\alpha-\theta),$$

由此得到坐标轴旋转的坐标变换公式：

$$\begin{cases} x_1 = x\cos\theta + y\sin\theta, \\ y_1 = y\cos\theta - x\sin\theta. \end{cases}$$

将8个顶点的原坐标带入，算得新坐标如下：

```
[569.2835121951122,  454.71648780487806]
```

In [15]:
```
def newcoordinate(old):
    new = []
    new.append(old[0]*1.73/2 + old[2]*1/2)
    new.append(old[1])
    new.append(old[2]*1.73/2 - old[0]*1/2)
    print(new)
```

In [16]:
```
for i in data:
    newcoordinate(i)
```

```
[10.1825, 0.5, 16.6175]
[10.6825, 0.5, 17.482499999999998]
[9.3175, 0.5, 17.1175]
[9.8175, 0.5, 17.982499999999998]
[9.3175, -0.5, 17.1175]
[9.8175, -0.5, 17.982499999999998]
[10.1825, -0.5, 16.6175]
[10.6825, -0.5, 17.482499999999998]
```

接下来将新坐标代入（1）问中的实体坐标→像素坐标的公式，得新的像素坐标为：

In [17]:
```
data1=[[10.1825, 0.5, 16.6175],
[10.6825, 0.5, 17.482499999999998],
[9.3175, 0.5, 17.1175],
[9.8175, 0.5, 17.982499999999998],
[9.3175, -0.5, 17.1175],
[9.8175, -0.5, 17.982499999999998],
[10.1825, -0.5, 16.6175],
[10.6825, -0.5, 17.482499999999998]]
```

In [18]:
```
for i in data1:
    Entity2pixel(i)
```

```
[1951.137287799007, 582.6671882052053]
[1947.1022954382954, 579.1707135707136]
[1790.4170655761648, 580.6030086169126]
[1794.2252812456557, 577.3030446267204]
[1790.4170655761648, 443.3969913830875]
[1794.2252812456557, 446.6969553732796]
[1951.137287799007, 441.3328117947947]
[1947.1022954382954, 444.8292864292864]
```

由于像素点是离散的，故可知新的像素坐标为：(1951, 583), (1947, 579), (1790, 581), (1794, 577), (1790, 443), (1794, 447), (1951, 441), (1947, 445)

2. **尝试解释 dolly zoom 现象。**

   答：滑动变焦（Dolly Zoom），是电影拍摄中一种很常见的镜头技法，它的特点是镜头中的主体大小不变，而背景大小改变。

   所谓滑动变焦有三大要素：摄像机的方向、推拉的速度、摄像机镜头的焦距，即在改变镜头焦距的同时，让摄像机也动起来，使得镜头中的物体在画面中的大小始终保持不变，只改变背景的大小。滑动变焦一般分为两种：外退内进（Dolly-out & Zoom-in）和外进内退（Dolly-in & Zoom-out）。外退内进指的是摄像机往后退，同时焦距往前进；外进内退则反之，摄像机往前进，同时焦距往后退滑动。

   滑动变焦能产生一种连续的透视变形，让背景看起来突然变大或变小，有一种“扑面而来”或“倏然离去”的感觉。很适合营造恐怖、扭曲、压迫、紧张的镜头氛围。相对静止的画面主体，和远近拉动的背景之间，会形成一种空间错位的观感，把画面主体推向视觉的中心。