获取视差图

首先读取两幅图片，然后得到图片的通道数，设置窗口的大小。然后利用 opencv 提供的全局匹配函数 cv2.StereoSGBM_create()函数创建 stereo，然后再利用 stereo.compute()函数计算得到视差图。

相关的参数设置在下方代码的 cv2.StereoSGBM_create()函数中体现。

```
def getDisparity():
    #读取彩色图片
    imgr = cv2.imread("RIma-000023.png")
    imgl = cv2.imread("LIma-000023.png")
    #设置参数
    window_size = 7
    channels = imgr.shape[2]
    #计算视差图
    stereo = cv2.StereoSGBM_create(
        #参数设置
        minDisparity = 0, #最小视差
        blockSize = block_size, #窗口大小
        #以下参数设置为官方推荐
        #P1 与 P2 为动态规划有关参数
        P1 = 8*channels*block_size*block_size,
        P2 = 32*channels*block_size*block_size,
        disp12MaxDiff = 1, #左视差图与右视差图的最大差异
        uniquenessRatio = 10, #次低代价>=最低代价*(1+uni/100)
        speckleWindowSize = 100, #平滑区域最大尺寸
        speckleRange = 10 #视差变化的阈值
    )
    disparity = stereo.compute(imgl,imgr)
    return disparity
```

计算深度图

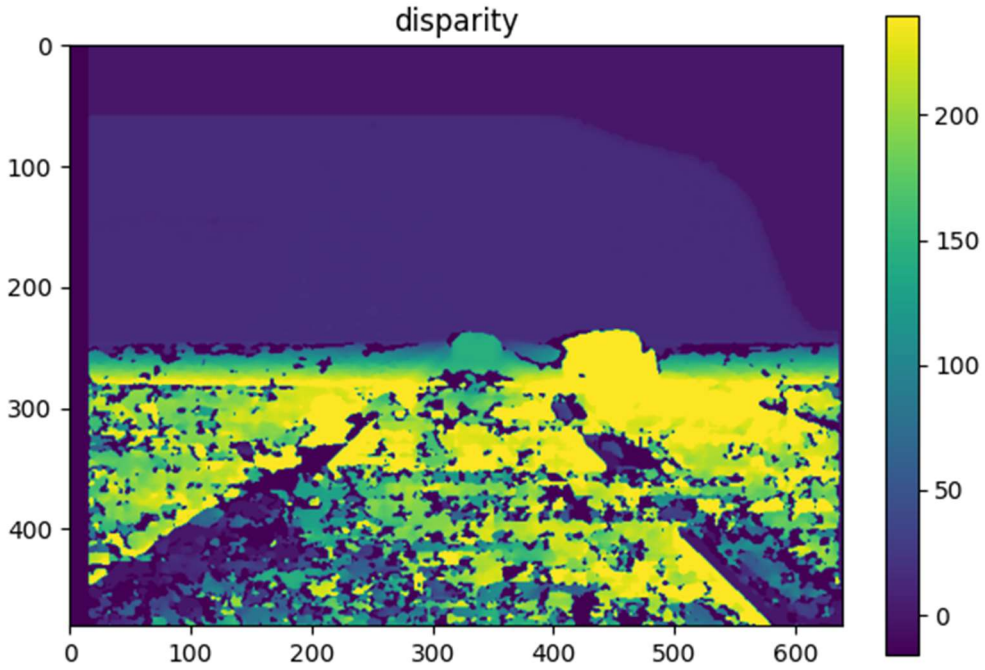
得到视差图之后，便可以根据公式得到深度，公式如下

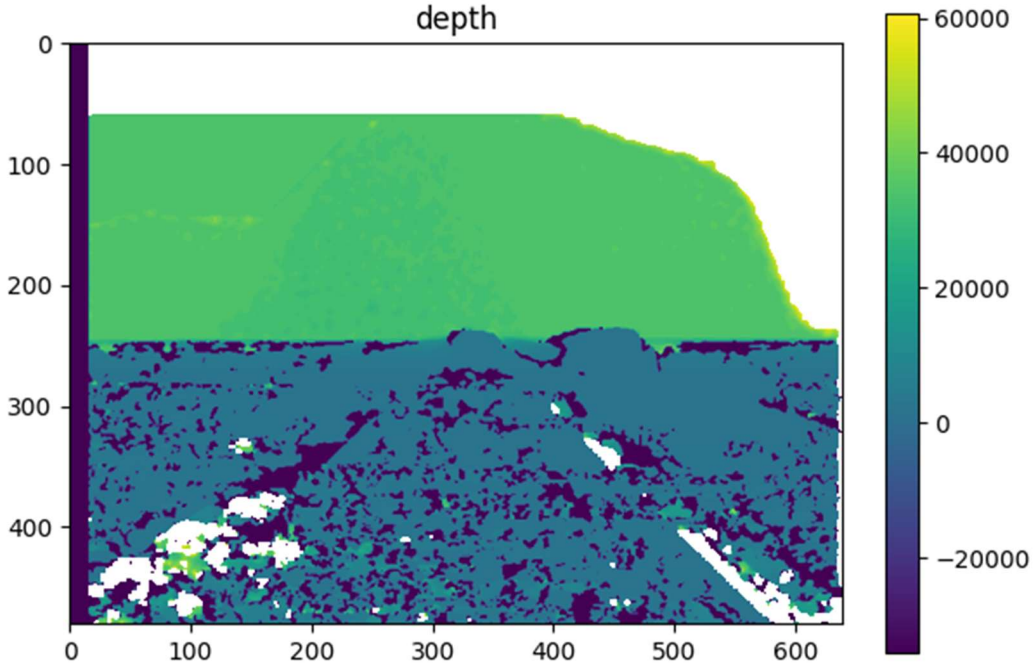
$$Z = \frac{B \cdot f}{D}$$

对此，遍历视差图中每个像素，计算每个像素对应的深度，最终获得视差图。

```
def getDepth(dis):
    f = 7 #焦距，单位为毫米
    B = 500 #基线， 单位为毫米
    w_of_pixel = 0.0064 #每个像素的宽度，单位为毫米
    [row, col] = dis.shape #获取形状
    #定义空图片
    depth = np.zeros((row, col))
    for r in range(row):
        for c in range(col):
            #计算视差实际值
            D = dis[r][c] * w_of_pixel
            #如果视差为0，说明深度无穷大
            if D == 0.0:
                Z = float("inf")
            else:
                Z = (B * f) / D
            depth[r][c] = Z
    return depth
```

实验结果


disparity
0
100
200
300
400
0
100
200
300
400
500
600
200
150
100
50
0


depth
0
100
200
300
400
0
100
200
300
400
500
600
60000
40000
20000
0
−20000