图像视差匹配分析报告

目录

一、实验目的

图像视差匹配，通过立体匹配得到两张图像的视差图。

二、实验原理

1、立体图像

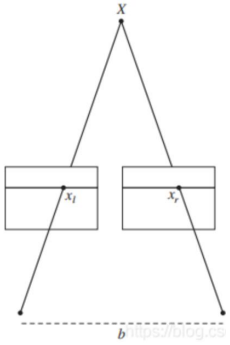
一个多视图成像的特殊例子是立体视觉（或者立体成像），即使用两台只有水平（向一侧）偏移的照相机观测同一场景。当照相机的位置如上设置，两幅图像具有相同的图像平面，图像的行是垂直对齐的，那么称图像对是经过矫正的。该设置在机器人学中很常见，常被称为立体平台。

通过将图像扭曲到公共的平面上，使外极线位于图像行上，任何立体照相机设置都能得到矫正（我们通常构建立体平台来产生经过矫正的图像对）。

假设两幅图像经过了矫正，那么对应点的寻找限制在图像的同一行上。一旦找到对应点，由于深度是和偏移成正比的，那么深度（Z 坐标）可以直接由水平偏移来计算：

$$Z = \frac{fb}{x_l - x_r}$$

其中，f 是经过矫正图像的焦距，b 是两个照相机中心之间的距离，x_l 和 x_r 是左右两幅图像中对应点的 x 坐标。分开照相机中心的距离称为基线。矫正后的立体照相机设置如图，对应点位于两幅图像的同一行。立体重建（有时称为致密深度重建）就是恢复深度图（或者相反，视差图），图像中每个像素的深度（或者视差）都需要计算出来。



2、视差图

2.1 视差

左右双目图像中，两个匹配块中心像素的水平距离。视差越大越靠近摄像头，而视差小的像素离摄像机远。

2.2 窗口计算视差

构造一个小窗口，用窗口覆盖左边的图像,选择出窗口覆盖区域内的所有像素点，同样用窗口覆盖右边的图像并选择出覆盖区域的像素点，左边覆盖区域减去右边覆盖区域，并求出所有像素点灰度差的绝对值之和。此算法常用于图像块匹配，将每个像素对应数值之差的绝对值求和，用来评估两个图像块的相似度。

2.3 影响视差的因素

1） 光学失真和噪声（亮度、色调、饱和度等失衡）
2） 平滑表面的镜面反射
3） 投影缩减
4） 透视失真
5） 低纹理
6） 重复纹理
7） 透明物体
8） 重叠和非连续

2.4 视差计算步骤

1）首先要对相机进行标定：标定的目的为获得相机的内外参，其基本原理是基于相机成像平面与空间的映射对应关系。
2）立体标定：获取左右相机的空间转换关系。
3）极线校正：同一个点在两幅图像上的映射，校正后两幅图片的点的坐标 y 值是一样的，可以减少待匹配的点数量。
4）立体匹配：求点跟点的视差，每个点的视差都求出来，就有了点的三维坐标了。

3、NCC 视差匹配

NCC 是用于归一化待匹配目标之间的相关程度，这里比较的是原始像素。对于原始的图像内任意一个像素点(Px,Py)构建一个 n×n 的邻域作为匹配窗口。然后对于目标相素位置(Px+d,Py)同样构建一个 n×n 大小的匹配窗口，对两个窗口进行相似度度量，这里的 d 有一个取值范围。对于两幅图像来说，在进行 NCC 计算之前要对图像处理，也就是将两帧图像校正到水平位置，光心处于同一水平线上，此时极线是水平的，否则匹配过程只能在倾斜的极线方向上完成，会消耗更多的计算资源。

当密集地应用在图像中时，归一化的互相关值可以很快地计算出来。我们使用每个像素周围的图像（根本上说，是局部周边图像）来计算归一化的互相关。对于这里的情形，我们可以在像素周围重新写出 NCC 计算公式：

$$\mathrm{ncc}(I_1, I_2) = \frac{\sum_x (I_1(\mathbf{x}) - \mu_1)(I_2(\mathbf{x}) - \mu_2)}{\sqrt{\sum_x (I_1(\mathbf{x}) - \mu_1)^2 \sum_x (I_2(\mathbf{x}) - \mu_2)^2}}$$

归一化将匹配结果限制在 [-1,1]的范围内，可以非常方便得到判断匹配窗口相关程度：

若 NCC = -1，则表示两个匹配窗口完全不相关，相反，若 NCC = 1 时，表示两个匹配窗口相关程度非常高。

匹配步骤：

1）采集图像：通过标定好的双目相机采集图像，也可用两个单目相机组合成双目相机。

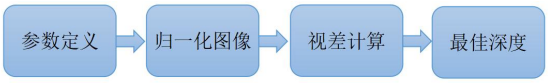
2）极线校正：校正的目的是使两帧图像极线处于水平方向，或者说是使两帧图像的光心处于同一水平线上。通过校正极线可以方便后续的 NCC 操作。实验中用的是已经矫正过的左图片和右图片。

3）特征匹配：利用 NCC 做匹配，右视图中与左视图待测像素同一水平线上相关性最高的即为最优匹配。完成匹配后，我们需要记录其视差 d，即待测像素水平方向 xl 与匹配像素水平方向 xr 之间的差值 d = xr - xl，最终可以得到一个与原始图像尺寸相同的视差图 D。

4）深度恢复：通过上述匹配结果得到的视差图 D，可以很简单的利用相似三角形反推出以左视图为参考系的深度图。

三．实验测试

1、代码流程



（1）设置参数：读取左目和右目图像，设置开始偏移、步长、窗口。

```
im_l = array(Image.open('/home/hyq/CV/4/script/data/1.png').convert('L'), 'f')
im_r = array(Image.open('/home/hyq/CV/4/script/data/2.png').convert('L'), 'f')
# 开始偏移，并设置步长
steps = 50
start = 4

# ncc 的宽度
wid = 12
```

（2）计算图像块的平均值，进行图像归一化。

```
# 计算图像块的平均值
uniform_filter(im_l, wid, mean_l)
uniform_filter(im_r, wid, mean_r)
# 归一化图像
norm_l = im_l - mean_l
norm_r = im_r - mean_r
```

（3）视差图计算。

```
# 尝试不同的视差
for displ in range(steps):
    # 将左边图像移动到右边，计算加和
    uniform_filter(roll(norm_l, -displ - start) * norm_r, wid, s)  # 和归一化
    uniform_filter(roll(norm_l, -displ - start) * roll(norm_l, -displ - start), wid, s_l)
    uniform_filter(norm_r * norm_r, wid, s_r)  # 和反归一化
    # 保存 ncc 的分数
    dmaps[:, :, displ] = s / sqrt(s_l * s_r)
# 为每个像素选取最佳深度
return argmax(dmaps, axis=2)
```

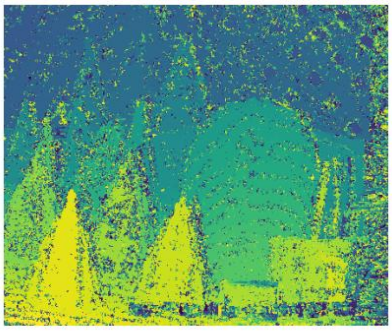
2、结果展示

输入左右两张图像，设置窗口值为 3、6、9、12，分别得到以下不同视差结果图。当 wid=3 时，噪声较多，图像的匹配结果比较模糊，因为 wid 值过小，窗口值为 3*3，所以导致匹点不够精准，在极线上的相似匹配点过多造成。当 wid=6 和 wid=9 的情况下，随着窗口

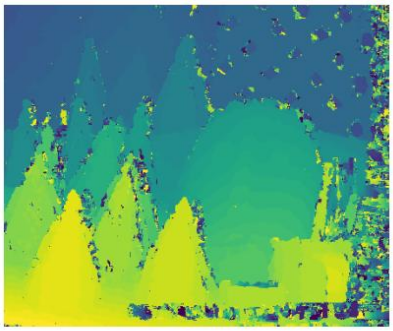
的扩大，包含的数据量变大，匹配点变的更加精准，图像边缘逐渐变的清晰，产生的噪点也少了很多。而 wid=12 时因为窗口值过大，导致丢失了一些边缘信息，图像边缘过于平滑，轮廓模糊。

左图

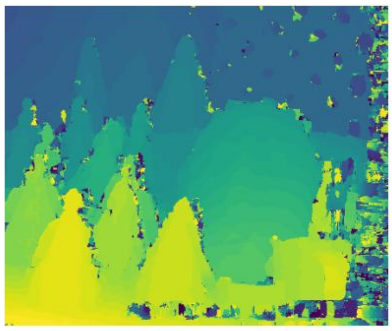
右图



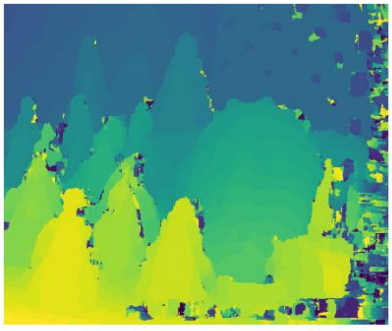
wid=3



wid=6



wid=9



wid=12