**图像视差匹配——实验报告**

# 姓名：于海煊 学号：123106222860 学院：计算机科学与工程学院

# 1 实验目的

本实验旨在通过立体匹配技术计算两张图像之间的视差图。实验采用了经典的块匹配方法，包括归一化互相关（NCC）、平方差（SSD）和绝对差（SAD）作为相似性度量。立体匹配是计算机视觉中的一项重要技术，广泛应用于3D重建、机器人导航和增强现实等领域。

## 1.1 立体视差

立体视差，亦称立体视像、立体知觉。基于双眼视差所获得的深度知觉。立体视差的测量包括三个步骤：

(1)必须从一幅图像中选出位于场景中一个表面上的某一特定位置；

(2)必须在另一幅图像中鉴别出同一个位置；

(3)测出这两个对应像点之间的视差。

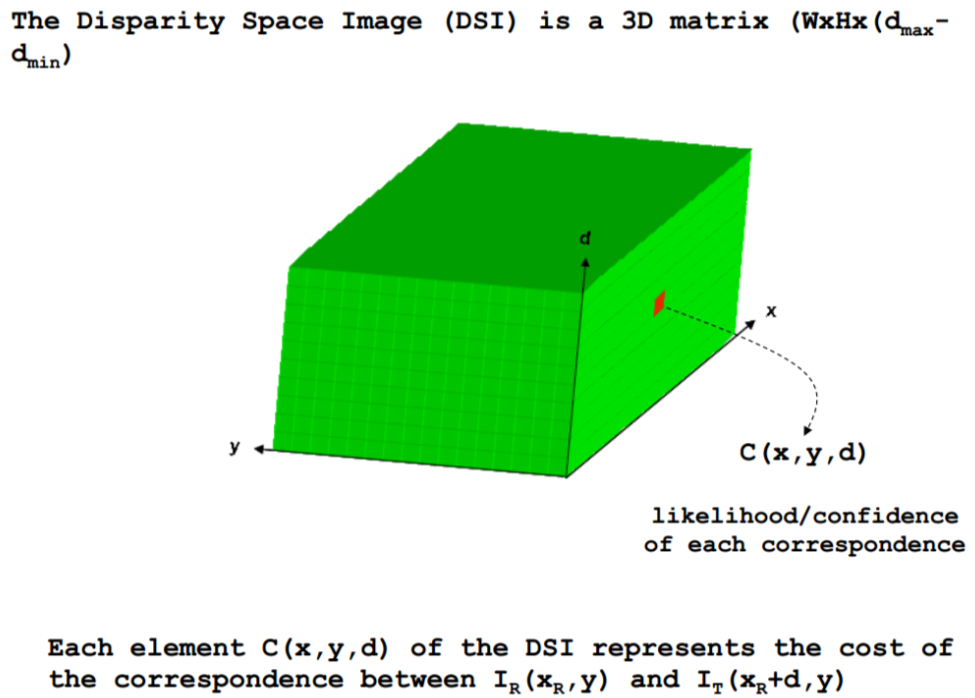
在一组重复摄影的两张照片上(立体模式)，同一地物的影象，沿着摄影基线(摄影地点和下一个摄影地点之间的飞行方向线)方向位置变换，这个变化量叫“立体视差”。

立体视差在每张照片上，从那点到照片基线(临接摄影地点与照片的摄影地点的像连结的直线)，下垂线的足和主点(相片中心)之间叫距离之和。这是把相对两个照片的基线方向，使其成为一直线，按摄影顺序排列时，和相应的同一地物影象间的两个照片主点之间的间隔差。

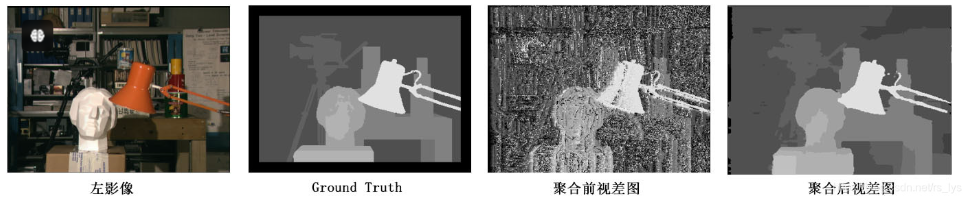
## 1.2 立体匹配算法

立体匹配算法通常由四个部分组成，包括：匹配代价计算，代价聚合，视差计算和视差优化。

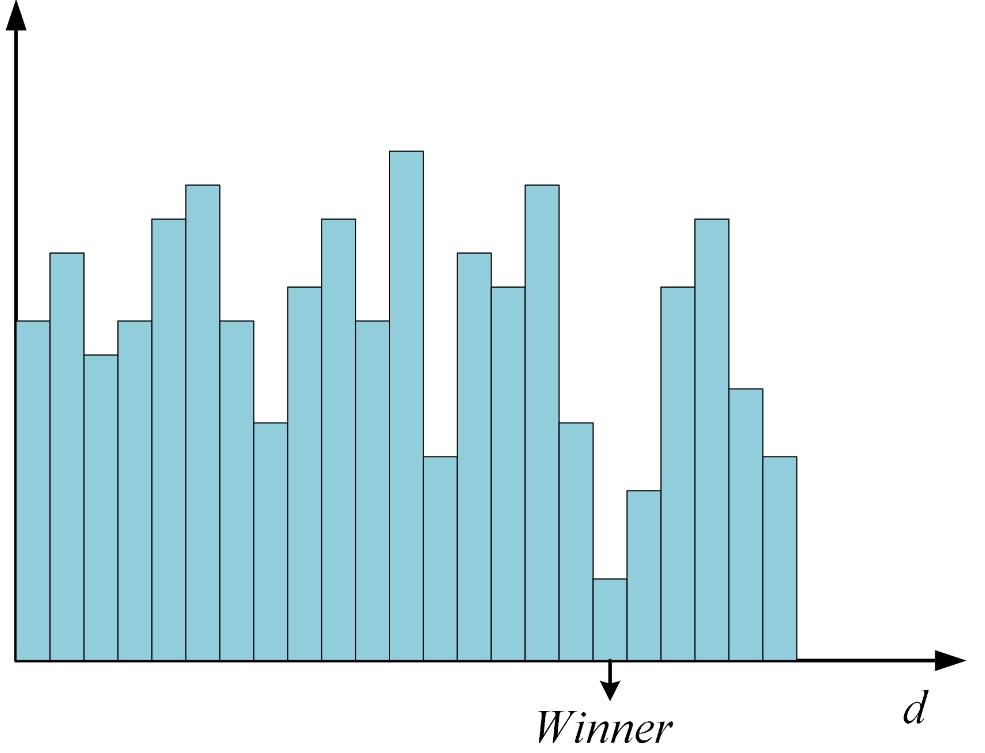
匹配代价计算的目的是衡量待匹配像素与候选像素之间的相关性。两个像素无论是否为同名点，都可以通过匹配代价函数计算匹配代价，代价越小则说明相关性越大，是同名点的概率也越大。每个像素在搜索同名点之前，往往会指定一个视差搜索范围D（Dmin ~ Dmax）,视差搜索时将范围限定在D内，用一个大小为W×H×D（W为影像宽度，H为影像高度）的三维矩阵C来存储每个像素在视差范围内每个视差下的匹配代价值。矩阵C通常称为DSI（Disparity Space Image）。DSI示意图如下：



代价聚合的根本目的是让代价值能够准确的反映像素之间的相关性。上一步匹配代价的计算往往只会考虑局部信息，通过两个像素邻域内一定大小的窗口内的像素信息来计算代价值，这很容易受到影像噪声的影响，而且当影像处于弱纹理或重复纹理区域，这个代价值极有可能无法准确的反映像素之间的相关性，直接表现就是真实同名点的代价值非最小。



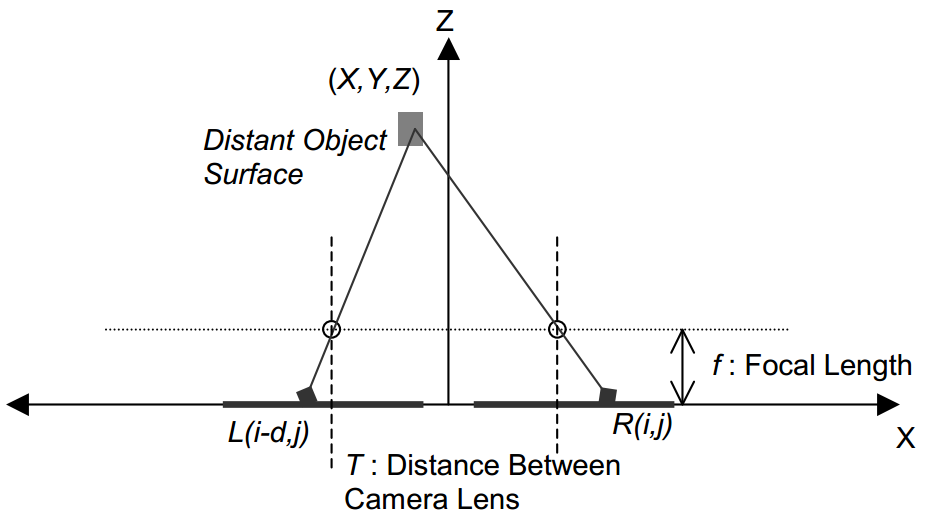
视差计算即通过代价聚合之后的代价矩阵S来确定每个像素的最优视差值，通常使用赢家通吃算法（WTA，Winner-Takes-All）来计算，如图2所示，即某个像素的所有视差下的代价值中，选择最小代价值所对应的视差作为最优视差。这一步非常简单，这意味着聚合代价矩阵S的值必须能够准确的反映像素之间的相关性，也表明上一步代价聚合步骤是立体匹配中极为关键的步骤，直接决定了算法的准确性。



视差优化的目的是对上一步得到的视差图进行进一步优化，改善视差图的质量，包括剔除错误视差、适当平滑以及子像素精度优化等步骤，一般采用左右一致性检查（Left-Right Check）算法剔除因为遮挡和噪声而导致的错误视差；采用剔除小连通区域算法来剔除孤立异常点；采用中值滤波（Median Filter）、双边滤波（Bilateral Filter）等平滑算法对视差图进行平滑；另外还有一些有效提高视差图质量的方法如鲁棒平面拟合（Robust Plane Fitting）、亮度一致性约束（Intensity Consistent）、局部一致性约束（Locally Consistent）等也常被使用。

双目立体匹配一直是双目视觉的研究热点，双目相机拍摄同一场景的左、右两幅视点图像，运用立体匹配匹配算法获取视差图，进而获取深度图。而深度图的应用范围非常广泛，由于其能够记录场景中物体距离摄像机的距离，可以用以测量、三维重建、以及虚拟视点的合成等。

为了模拟人眼对立体场景的捕捉和对不同景物远近的识别能力，立体匹配算法要求采用两个摄像头代替人眼，通过获取两幅非常接近的图片以获取景深(视差：Disparity），从而计算出不同景物与摄像头的距离,得到景深图。



立体投影表示

# 2 实验方法

编程语言：Python

算法核心步骤：

1、图像读取：以灰度模式读取左右视图的图像。

2、相似性度量函数：实现了NCC、SSD和SAD三种相似性度量方法。

3、视差图计算：通过滑动窗口在左右图像间搜索最佳匹配块，并计算视差值。

4、结果展示：将计算得到的视差图进行缩放并展示。

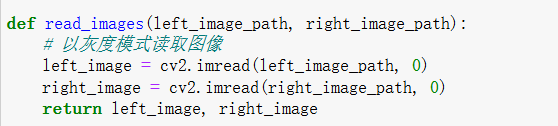
# 3 实验过程

## 3.1 数据准备

本次实验用的是2003 Stereo datasets，它包含高分辨率的双目图像序列，以及精确到像素水平的实际真值视差图。实际真值视差图由结构光这种新技术采集获得，因而不需要矫正光照映射。

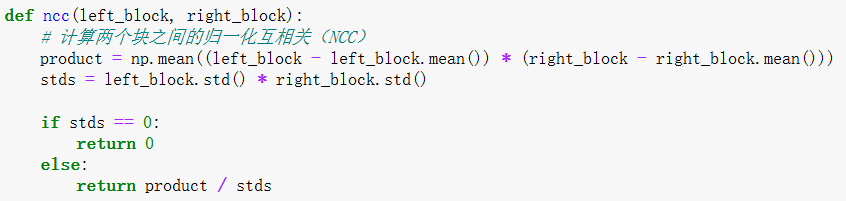
## 3.2 代码实现

### 3.2.1 读取图像函数

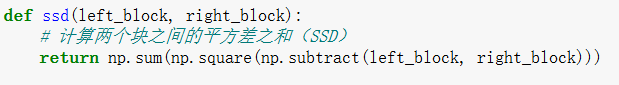


read\_images函数接收两个参数，分别是左右图像的文件路径。使用cv2.imread函数以灰度模式读取图像，并返回这两个图像。

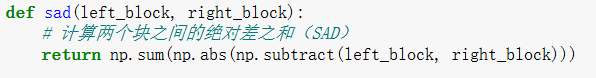
### 3.2.2 相似性度量函数



ncc:归一化互相关，用于测量两个图像块之间的相关性。

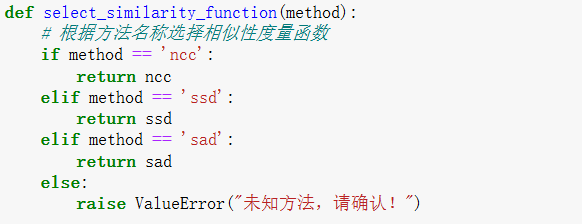


ssd:平方差，计算两个图像块之间的平方差总和。



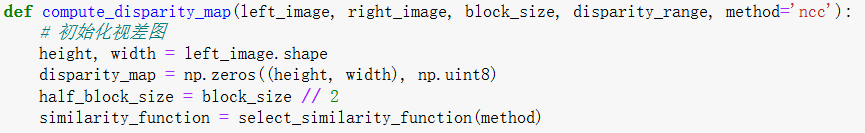
sad:绝对差，计算两个图像块之间的绝对差总和。

### 3.2.3 选择相似性度量函数

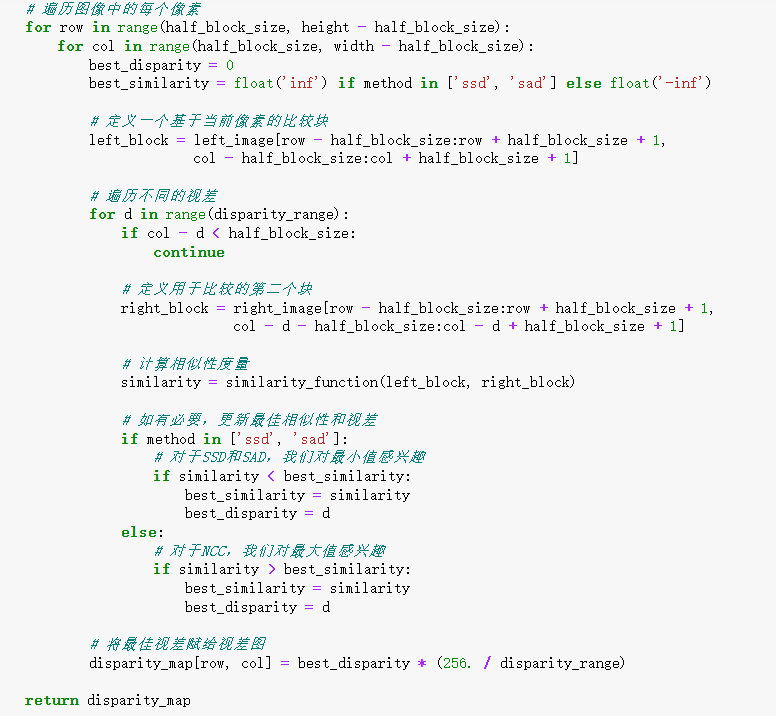


select\_similarity\_function函数根据输入的字符串参数（'ncc', 'ssd', 'sad'）返回相应的相似性度量函数。

### 3.2.4 计算视差图函数



初始化视差图



compute\_disparity\_map函数是程序的核心，它接收左右图像、块大小、视差范围和相似性度量方法作为参数。函数内部初始化视差图，然后通过双层循环遍历图像中的每个像素，对于每个像素，它搜索最佳匹配的视差值，并更新视差图。

### 3.2.5 主函数部分



# 4 结果分析

测试的是以下四组图像，报告中仅展示个别结果，全部结果请参考outputs文件夹：





## 4.1 相同方法下不同窗口大小的影响

本次实验在SSD方法下测试不同窗口大小的影响。

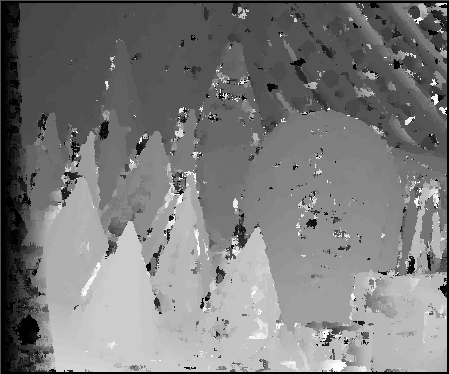
block\_size = 3时，输出的视差图：



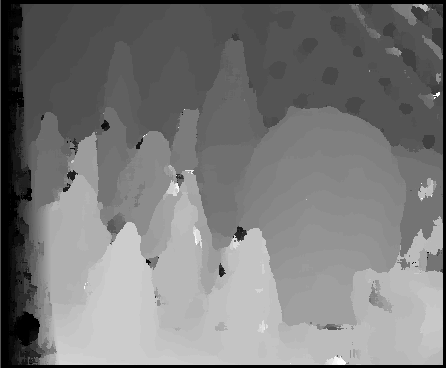
block\_size = 5时，输出的视差图：



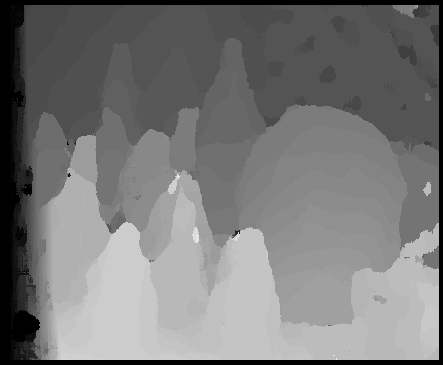
block\_size = 7时，输出的视差图：



block\_size =15时，输出的视差图：



block\_size =21时，输出的视差图：



### 分析

窗口大小的影响：

首先使用了一个非常小的窗口大小(block\_size=3)。观察到虽然物体的轮廓清晰可见，但视差图却明显受到噪声的影响。当窗口大小稍微增加到5或7时，噪声显著减少。随着窗口大小的进一步扩大，达到15或21，噪声几乎减少到不可察觉的水平，图像不仅更平滑，而且具有可辨识的深度感，允许明确区分近处和远处的物体。

大窗口大小的失真：

然而，这种尺寸调整也带来了一个意外的后果。在较大的窗口大小，特别是15和21时，观察到物体形态开始出现失真，尤其是较近的物体。此外，一些细节，如远处栅栏的孔洞，被模糊或完全丢失。

|  |  |
| --- | --- |
| block\_size | Time(s) |
| 3 | 48.70 |
| 5 | 53.18 |
| 7 | 56.90 |
| 15 | 60.20 |
| 21 | 63.82 |

时间观察：

在计算时长方面，实验显示出不一致性，处理时间在45到65秒之间波动，没有明显的模式与窗口大小或其他参数相关。

小结：

a.小窗口的缺点：

极小的窗口大小会导致每个点基本独立工作，类似于仅使用左右图像中单个像素的灰度值进行立体匹配，这已知是不准确的。

b.大窗口的复杂性：

过大的窗口包含了过多的像素，稀释了任何单个像素位移的影响。因此，当右侧窗口稍微移动时，窗口内容的微小变化会导致算法错误地假设相邻点的视差是均匀的。这导致了一个虽然平滑但缺乏关键细节的视差图。

c.最优窗口大小：

实验明确地展示了窗口大小对生成的视差图保真度的深远影响。从较小到较大的尺寸过渡，揭示了噪声减少、图像平滑度和层次感增强的趋势。对于这个实验，考虑到[0,64]的视差搜索范围，一个最优的窗口大小可能在7到15之间，平衡了细节与噪声减少以及整体图像质量。

## 4.2 不同方法在相同窗口大小的情况下对结果的影响

block\_size=3,SAD 方法：



block\_size=3,SSD方法 ：



block\_size=3，NNC方法：



### 分析：

SAD算法效果不佳。当使用3x3这样的小窗口大小时，所有三种算法（SAD、SSD和NCC）都倾向于在结果中产生大量噪声。SSD（平方差之和）比SAD显示出较少的噪声点，而NCC（归一化互相关）产生的噪声点最多，显著影响了远处物体（如栅栏）的清晰度。

误差表现：

在SAD和SSD的结果中，深度不准确通常呈现为离散点或聚集区域。相比之下，NCC输出中的错误更均匀分布，表现为密集、分散的点，聚集较少。

块大小变化：

当窗口大小增加到7x7或15x15这样的尺寸时（输出的视差图在results文件夹中），SSD显示出较少的不准确，特别是在接近视点的物体上，尽管一些区域显示出夸张的错误。相比之下，NCC提供了更远物体更平滑的距离表示，尽管视差图中的粒度更明显。当窗口大小扩大到21x21时，SAD比SSD和NCC产生更少的噪声点，尽管所有三种算法都引入了一些物体形状的失真。

NCC中的不准确：

NCC图中特别在大三角形周围的显著错误区域，可能是由于搜索范围不足。当算法从左到右处理左图像时，它最初定位了绿色三角形的最左点。然而，搜索范围的限制阻止了它找到最优匹配，导致过早选择了匹配点。这个误算转化为一个低估了实际视差的视差，由表示较小视差的暗点证明。

最优算法选择：

基于重复试验，推荐使用SSD或NCC算法，配合适中的7x7到15x15块大小，以获得更精确的结果，或者使用较小的3x3窗口大小的SAD算法，以较低的计算需求获得可接受的结果。

# 5 结论

实验结果分析表明，窗口大小对视差图的质量有显著影响。较小的窗口（如3x3）会产生噪声，而较大的窗口（如15x15或21x21）则可以减少噪声并提高图像的平滑度和深度感。然而，窗口过大也会导致物体形态失真和细节丢失。实验中还观察到，处理时间在不同窗口大小下波动，但没有明显的时间与窗口大小的关联模式。

综合考虑，实验推荐使用SSD或NCC算法，配合适中的7x7到15x15的窗口大小，以获得更精确的视差图结果。或者，在计算需求较低的情况下，可以使用3x3窗口大小的SAD算法，以获得可接受的结果。