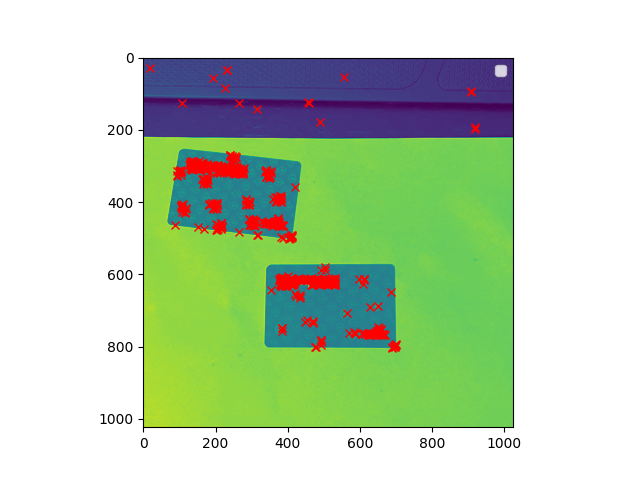
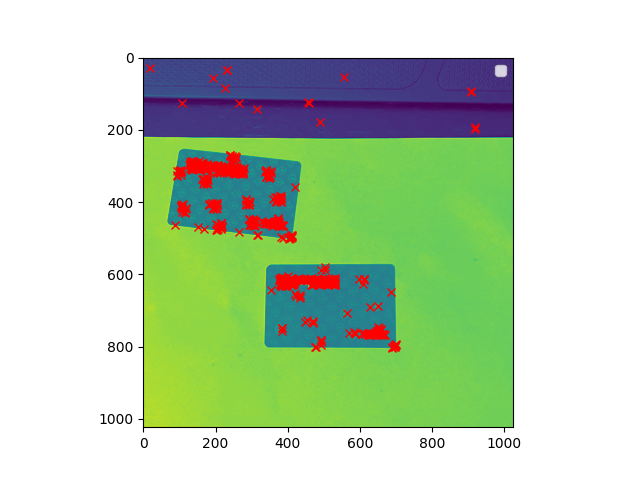
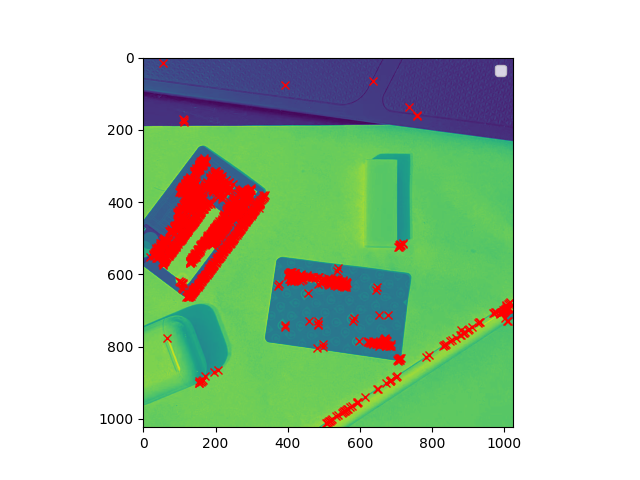
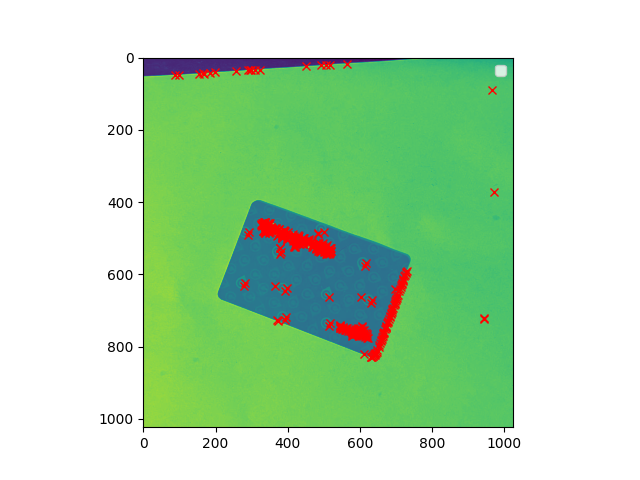
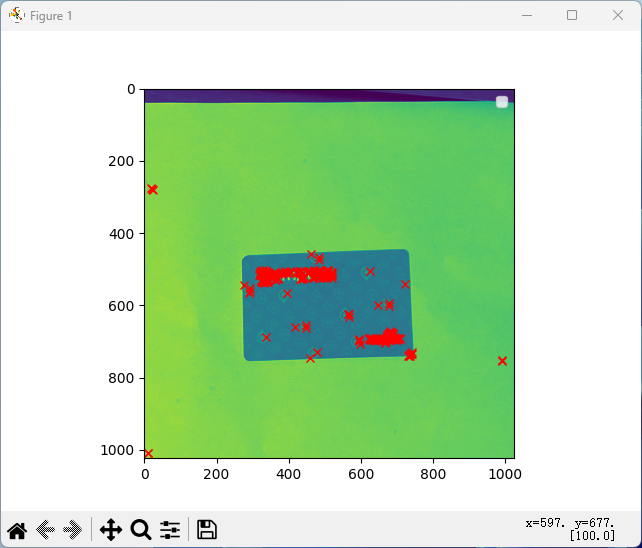
**DIP作业3**

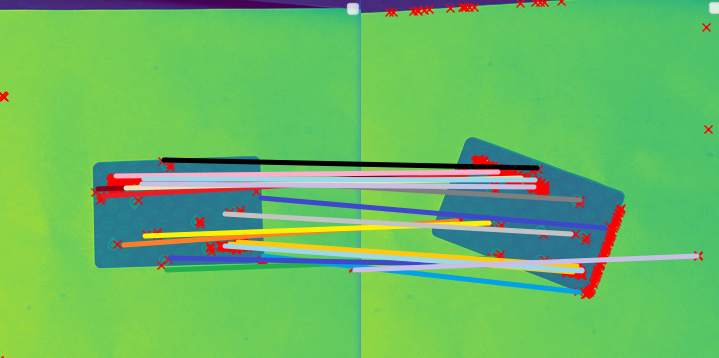
课程作业 1：(1月12日前完成)

作业要求见实验3 PPT。

可以使用任何一种编程语言，需要提交(C/C++，python等源代码，课程作业2报告文档（方法，实验载图等））将源代码和作业文档压缩在一起提交到QQ群作业，压缩包名称按照“学号姓名DIP1.zip”格式。

实验结果：

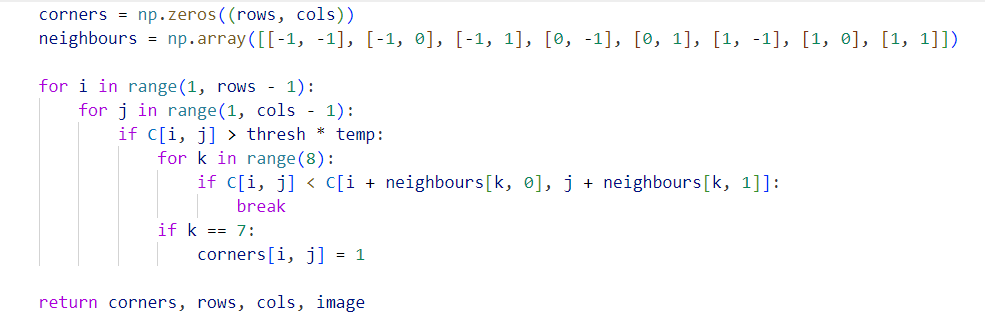




作业流程：

1、Harris角点检测算法实现





该代码实现了Harris角点检测算法，用于在图像中识别角点，并返回一个二值图像，突出显示检测到的角点。

1. **输入参数:**

img: 待进行Harris角点检测的输入图像。

dim1 和 dim2: 用于对输入图像进行中心裁剪的维度。

thresh: 角点响应的阈值。具有响应大于此阈值的角点被认为是有效的。

1. **图像处理:**

将输入图像转换为PyTorch张量 (img\_torch)。

使用由 dim1 和 dim2 指定的维度对图像进行中心裁剪。

生成用于平滑的高斯核 (gaus)。

生成用于计算图像梯度的导数核 (deriv\_gaus\_x 和 deriv\_gaus\_y)。

1. **梯度计算:**

使用生成的导数核计算图像在x和y方向的梯度 (ix 和 iy)。

计算梯度的平方 (ix2 和 iy2) 以及梯度的乘积 (ixiy)。

1. **梯度平滑:**

对梯度的平方和梯度的乘积使用高斯核进行卷积以进行平滑 (ix2g、iy2g、ixiyg)。

Harris角点响应函数:

对于图像中的每个像素（排除边缘8像素），从平滑梯度构建一个2x2结构张量 M。

在每个像素处计算Harris角点响应 C，其计算方式为M的行列式减去M的迹的平方的缩放倍数。

1. **角点检测:**

应用阈值处理来识别潜在的角点。具有响应大于图像中最大响应的某个分数 (thresh) 的像素被视为角点。

通过检查每个像素及其8个邻居来执行非极大值抑制。如果一个像素的响应高于其所有邻居，则将其标记为角点。

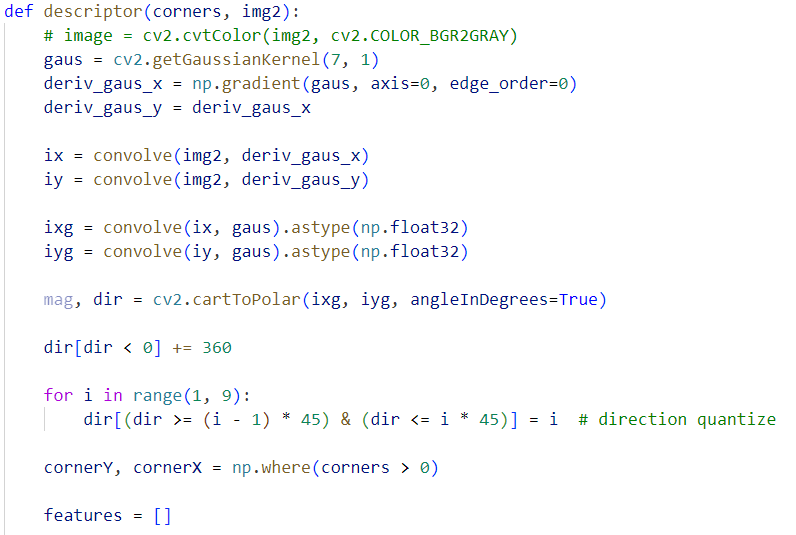
1. **输出:**

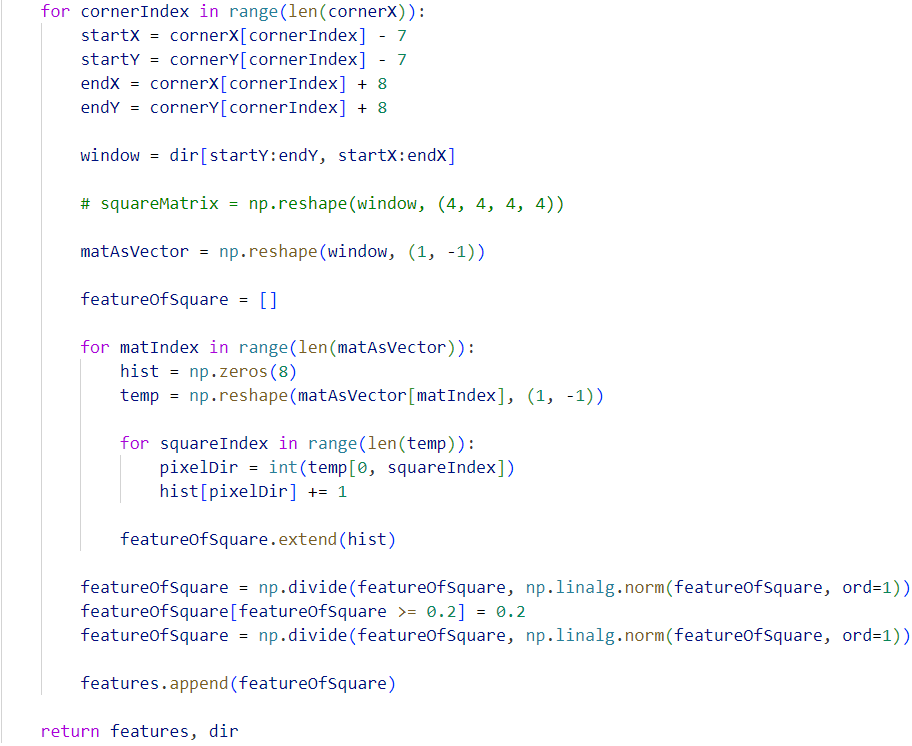
函数返回一个二值图像 (corners)，表示检测到的角点。

rows 和 cols 表示输入图像的维度。

image 是中心裁剪的输入图像。

2、Harris角点周围区域的局部图像描述符计算





该代码计算了Harris角点周围局部区域的图像描述符，描述符是在每个角点周围窗口中梯度方向的分布的直方图。特征向量被归一化并进行了剪切，以保证其稳健性。

1. **梯度计算:**

生成高斯核 (gaus)。

使用numpy的梯度函数计算x方向上的高斯核的导数 (deriv\_gaus\_x)。

y方向上的高斯核的导数 (deriv\_gaus\_y) 等于x方向的导数。

1. **图像梯度计算:**

通过将输入图像 (img2) 与x和y方向上的高斯导数进行卷积，得到图像的梯度值 (ix 和 iy)。

1. **梯度平滑:**

利用高斯核对梯度值进行卷积，得到平滑的梯度值 (ixg 和 iyg)。

1. **梯度幅值和方向:**

使用 cv2.cartToPolar 函数计算梯度的幅值 (mag) 和方向 (dir)。

1. **方向量化:**

将方向进行量化，将角度分为8个方向。

1. **循环处理每个角点:**

遍历检测到的Harris角点的坐标 (cornerX 和 cornerY)。

1. **描述符计算:**

定义每个角点周围的局部窗口。

将方向窗口展平为一维数组 (matAsVector)。

对于 matAsVector 中的每个元素，计算直方图 (hist) 来表示窗口中方向的分布。

将直方图连接在一起形成局部窗口的特征向量。

1. **归一化:**

使用L1范数对特征向量进行归一化。

1. **剪切和再次归一化:**

将特征向量中大于等于0.2的值截断为0.2。再次进行L1范数归一化。

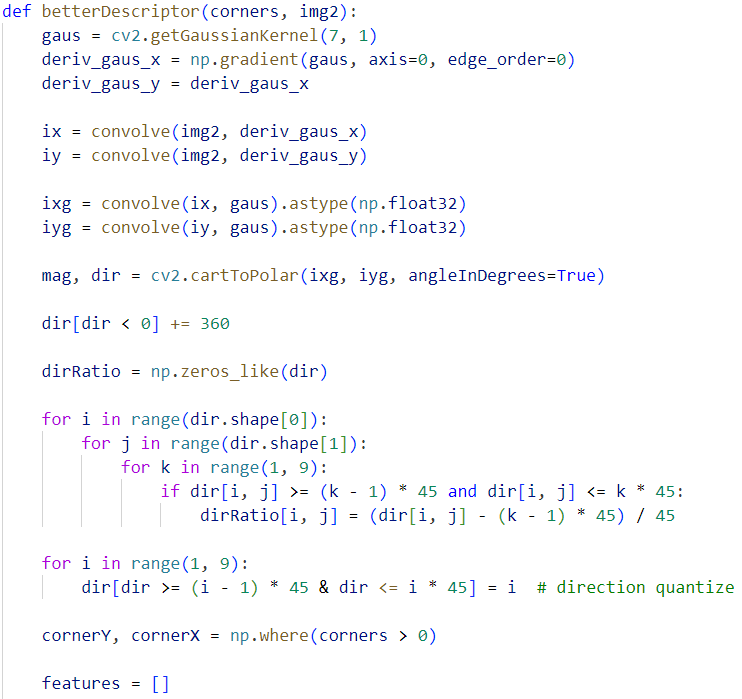
1. **特征存储:**

将每个角点周围的特征向量存储在一个列表中 (features)。

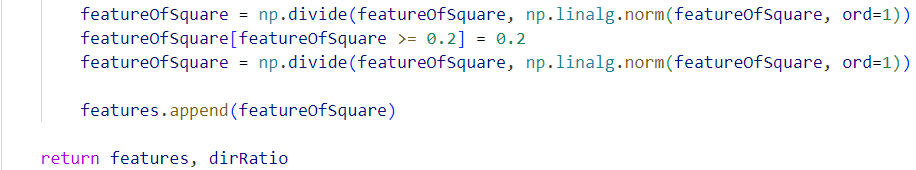
1. **输出:**

函数返回特征向量列表 (features) 和方向信息 (dir)。

3、Harris角点周围区域的局部图像描述符的改进版本







这段代码改进了对Harris角点周围区域的图像描述符计算，引入了方向比例信息，从而更好地描述了梯度在不同方向上的分布情况。

1. **梯度计算:**

生成高斯核 (gaus)。使用numpy的梯度函数计算x方向上的高斯核的导数 (deriv\_gaus\_x)。

y方向上的高斯核的导数 (deriv\_gaus\_y) 等于x方向的导数。

1. **图像梯度计算:**

通过将输入图像 (img2) 与x和y方向上的高斯导数进行卷积，得到图像的梯度值 (ix 和 iy)。

1. **梯度平滑:**

利用高斯核对梯度值进行卷积，得到平滑的梯度值 (ixg 和 iyg)。

1. **梯度幅值和方向:**

使用 cv2.cartToPolar 函数计算梯度的幅值 (mag) 和方向 (dir)。

1. **方向量化和比例计算:**

对方向进行量化，将角度分为8个方向。计算角度的相对比例 (dirRatio)，用于后续计算梯度在不同方向的分布。

1. **循环处理每个角点:**

遍历检测到的Harris角点的坐标 (cornerX 和 cornerY)。

1. **描述符计算:**

定义每个角点周围的局部窗口。

将方向窗口、方向比例窗口和梯度幅值窗口分别展平为一维数组。

对于每个元素，计算梯度方向在不同方向的分布直方图。

1. **归一化:**

使用L1范数对特征向量进行归一化。

1. **剪切和再次归一化:**

将特征向量中大于等于0.2的值截断为0.2。再次进行L1范数归一化。

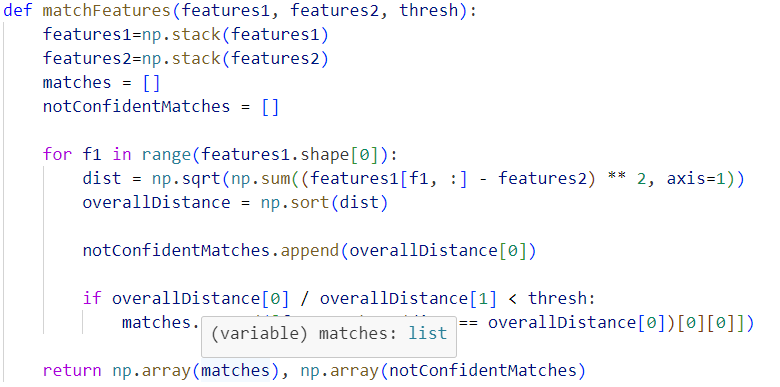
1. **特征存储:**

将每个角点周围的特征向量存储在一个列表中 (features)。

1. **输出:**

函数返回特征向量列表 (features) 和方向比例信息 (dirRatio)。

4、特征匹配函数



这段代码实现了一种简单的特征匹配方法，遍历了两组特征向量，计算了它们之间的欧氏距离，并通过阈值检测确定最终匹配结果。

1. **准备特征向量:**

将输入的两组特征向量 (features1 和 features2) 转换为NumPy数组。

1. **初始化匹配列表:**

初始化两个列表，matches 用于存储最终的匹配结果，notConfidentMatches 用于存储未经过阈值检测的匹配距离。

1. **循环匹配:**

遍历 features1 中的每个特征向量。

计算当前特征向量与 features2 中所有特征向量之间的欧氏距离。

对每个特征向量的距离进行排序，找到最小距离和次小距离。

1. **距离比较:**

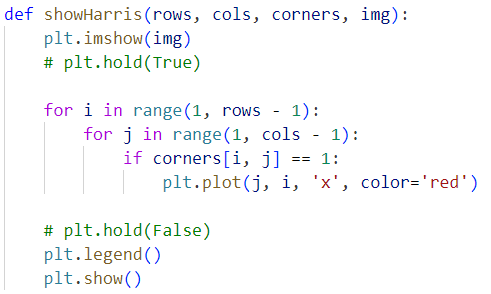
将最小距离与次小距离的比值与预设的阈值 (thresh) 进行比较。

如果比值小于阈值，将这一对匹配的特征向量索引加入 matches 列表。

1. **返回结果:**

将最终的匹配结果以及未经过阈值检测的匹配距离列表返回。

5、Harris角点检测结果可视化



可视化Harris角点检测的结果，通过在图像上用红色交叉符号标记检测到的角点。

1. **导入matplotlib库:**

假设在其他地方已经导入了 matplotlib.pyplot 库。

1. **图像显示:**

使用 plt.imshow(img) 将输入图像 img 显示。

1. **标记Harris角点:**

遍历角点图像 (corners)，当检测到一个角点时，使用 plt.plot(j, i, 'x', color='red') 在图上标记红色的交叉符号。

1. **显示图例:**

虽然代码中有 plt.legend()，但在这里并没有添加图例说明，因此这行代码可能是不必要的。

1. **显示图像:**

使用 plt.show() 显示包含标记的图像。