Machine Vision Exams

CHEN SHANG

Monday 27th November, 2023

2020 - 2021

1.

(a)

(i)

图中问题要求计算一个 5x5 图像的类内方差，根据大津算法在阈值设定为 7 时。大津算法是一种确定图像二值化分割阈值的方法，类内方差是指两个类别（阈值以上和以下）的方差的加权和。

类内方差计算公式是:

$$\sigma_w^2 = \omega_1 \sigma_1^2 + \omega_2 \sigma_2^2$$

其中，ω_1 和 ω_2 是两个类别的像素比例，σ_1^2 和 σ_2^2 是各自类别的方差。

具体计算步骤如下：1. 计算阈值为 7 时，分割的两个类别的像素值。

2. 计算每个类别的平均值（均值）。

3. 计算每个类别的方差。

4. 计算每个类别的像素比例。

5. 根据公式计算类内方差。

within class variance = 3.764935064935065

(ii)

大津的方法（Otsu's method）用于自动选择图像的阈值以进行二值化。这种方法旨在选择阈值以最小化类内方差，或等效地，最大化类间方差。下面是获取大津最佳阈值的步骤：

1. 计算图像的直方图和每个像素值的概率分布。

2. 对于直方图中的每个可能的阈值：

- 将直方图分为两个部分，即阈值以下和以上的像素值。
- 计算两个类的概率（图像中像素点数的比例）。
- 计算两个类的平均灰度值。

3. 计算类间方差，这是基于当前阈值将图像分成两部分时两类平均值的差异的平方，再乘以两个类的概率之积。

4. 遍历所有可能的阈值，找到使类间方差最大化的阈值。这个值就是大津的最优阈值。

大津的方法实质上是在寻找一个阈值，该阈值能够最大化前景和背景（即由该阈值分割的两个类）的灰度差异。通过这种方式，图像被分割成具有最大对比度的两个部分，这通常对应于最清晰的图像分割。这个方法假设图像由前景和背景两部分组成，且各自的像素值分布近似于高斯分布。

(b)

(i)

边缘检测是图像处理中用于确定图像中物体边界的技术。使用二阶微分进行边缘检测的原理基于这样一个事实：图像中的边缘

可以被视为灰度强度的突变区域，而这些突变区域在一阶导数中表现为峰值，在二阶导数中则表现为零交叉点。

具体来说，二阶导数边缘检测的步骤如下：

1. 对图像应用二阶微分算子，例如拉普拉斯算子（Laplacian），这可以通过对图像卷积与一个拉普拉斯滤波器来实现。

2. 在二阶微分的结果中，寻找零交叉点。零交叉点是二阶导数从正值变为负值或从负值变为正值的位置，这表示灰度强度有显著的变化。

3. 将这些零交叉点标记为边缘。

这种方法的优点是可以直接定位到边缘的精确位置，但也有一些缺点，例如对噪声非常敏感，因为噪声也可能导致灰度强度的快速变化。因此，在应用二阶导数之前，通常需要对图像进行平滑处理，以减少噪声的影响。此外，二阶导数算子也倾向于产生较细的边缘线，这在某些应用中可能是期望的效果，在其他情况下则可能需要进一步处理以增强边缘。

(ii)

高斯平滑和二阶微分可以通过构造一个结合了高斯函数和拉普拉斯算子的卷积核来一步完成。这样的卷积核通常被称为拉普拉斯-高斯（LoG）核，它是高斯平滑核和拉普拉斯核的结合。

给定二维高斯函数 $G(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$，我们可以通过对 $G(x, y)$ 应用拉普拉斯算子 $\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$ 来获得 LoG 核。

下面，我们将计算一个 $\sigma = 0.9$ 和尺寸为 7×7 的 LoG 卷积核。我们首先创建一个 7×7 的网格来表示 x 和 y 坐标，然后计算每个位置的 LoG 值。

看起来我们再次遇到了系统内部错误。不过，我可以为你提供手动计算 LoG (Laplacian of Gaussian) 卷积核的步骤。你可以使

用这些步骤在自己的环境中进行计算：

1. 定义一个 7×7 的网格，用于表示 x 和 y 的坐标。坐标的中心应该是 $(0, 0)$，所以网格应该从 -3 到 3。

2. 对于网格中的每个点 (x, y)，计算二维高斯函数 $G(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$。

3. 对于网格中的每个点 (x, y)，计算二阶微分拉普拉斯算子 $\nabla^2 G$，这可以通过 $\nabla^2 G = \frac{\partial^2 G}{\partial x^2} + \frac{\partial^2 G}{\partial y^2}$ 得到，也就是 $\nabla^2 G = -\frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right] e^{-\frac{x^2+y^2}{2\sigma^2}}$。

4. 计算得到的 $\nabla^2 G$ 就是 LoG 卷积核的每个元素值。

5. 通常会对卷积核进行标准化，以确保所有元素的和为 0，这是因为 LoG 是一个高通滤波器，不应该改变图像的平均亮度。

2.

(a)

图中的问题是关于贝叶斯决策函数的 0-1 损失。贝叶斯决策理论是统计决策理论的核心，旨在最小化决策过程中的总体损失。0-1 损失函数是分类问题中常用的一种损失函数，它在分类正确时损失为 0，分类错误时损失为 1。

在这个问题中，我们要证明贝叶斯决策函数在 0-1 损失下的形式。我们从贝叶斯公式出发：

$$P(\omega_j|x) = \frac{P(x|\omega_j)P(\omega_j)}{P(x)}$$

其中，$P(\omega_j|x)$ 是后验概率，即给定特征向量 x 的条件下类别 ω_j 的概率；$P(x|\omega_j)$ 是似然概率，即给定类别 ω_j 的条件下特征向量 x 的概率；$P(\omega_j)$ 是类别 ω_j 的先验概率；$P(x)$ 是特征向量 x 的边缘概率。

在 0-1 损失情形下，我们选择最大化后验概率 $P(\omega_j|x)$ 的分类决策规则。然而，由于 $P(x)$ 对所有的类别是常数，我们可以忽略它，直接最大化分子 $P(x|\omega_j)P(\omega_j)$。

因此，贝叶斯决策函数可以定义为：

$$d_j(x) = P(x|\omega_j)P(\omega_j)$$

这里，$d_j(x)$ 代表选择第 j 类的决策函数。在分类时，我们会计算所有类别的 $d_j(x)$，选择具有最大值的类别作为 x 的分类结果。

具体的计算过程是：首先，我们需要知道每个类别的先验概率 $P(\omega_j)$ 和似然概率 $P(x|\omega_j)$。然后，对于一个给定的特征向量 x，我们计算每个类别的 $d_j(x)$。最后，我们选择具有最大 $d_j(x)$ 值的类别作为 x 的预测类别。

这是统计模式识别和机器学习中的一个基础概念，用于在给定数据的情况下，如何选择最优的类别以最小化总体的分类错误率。

3.

(a)

在平行双目立体视觉设置中，校正（rectification）是相机校准过程中的一个重要步骤。它确保了成对摄像机捕获的图像在同一平面上，简化了后续的视差计算。立体校正的详细步骤如下：

1. 相机内参校准：使用棋盘格或其他标定模式，分别为两个摄像机估计内参（焦距、主点、径向畸变等）。

2. 相机外参校准：确定两个摄像机之间的相对位置和姿态，即它们之间的旋转 R 和平移 t 矩阵。

3. 估计校正变换：计算两个摄像机图像平面到共同的图像平面的变换。这通常涉及寻找一个变换矩阵，它可以将两个图像平面

旋转和平移，使得它们的投影平面共面，并且它们的扫描线对齐。

4. 畸变校正：对两个摄像机的图像进行畸变校正，特别是校正径向畸变和切向畸变。

5. 计算重投影映射：为每个摄像机计算重投影映射，这将用于重映射图像，以便对齐双目摄像机的图像。

6. 应用重投影映射：使用重投影映射将两个摄像机的图像重映射到共同平面。这个步骤通常涉及到插值方法，如双线性插值或立方插值，以在新的图像平面上重构图像。

7. 验证校正结果：通过检查重映射后的图像中对应点的对齐情况来验证校正结果。对应点应该在同一水平线上。

8. 微调：如果必要，进行微调以进一步改善图像对齐。这可能涉及手动调整参数或使用优化算法来最小化某些误差度量。

立体校正对于减少双目立体视觉系统中的复杂性和计算成本至关重要，它允许使用简单的一维搜索来找到对应点，因为对应点仅在同一水平线上移动。校正后的图像为三维重建、物体检测和其他计算机视觉任务提供了一个更简单的起点。

In parallel binocular stereo settings, rectification is a crucial part of camera calibration. It ensures that the image planes of the two cameras are coplanar, which simplifies the process of computing disparity. Detailed steps of stereo rectification are as follows:

1. Camera Intrinsic Calibration: Use a calibration pattern, like a chessboard, to estimate the intrinsic parameters (focal length, principal point, radial distortion, etc.) for both cameras independently.

2. Camera Extrinsic Calibration: Determine the relative position and orientation, i.e., the rotation R and translation t matrices, between the two cameras.

3. Estimate Rectification Transform: Compute the transformation that maps the image planes of the two cameras to a common image plane. This typically involves finding a transformation matrix that will rotate and translate the image planes so they are coplanar and their scanlines are aligned.

4. Distortion Correction: Correct the images from both cameras for distortion, especially correcting for radial and tangential distortions.

5. Compute Reprojection Maps: Calculate the reprojection maps for each camera, which will be used to remap the images so that the stereo camera images are aligned.

6. Apply Reprojection Maps: Use the reprojection maps to remap the images from both cameras onto a common plane. This step often involves interpolation methods, such as bilinear or cubic interpolation, to reconstruct the images on the new image plane.

7. Validate Rectification: Verify the results of the rectification by checking the alignment of corresponding points in the remapped images. Corresponding points should lie on the same horizontal line.

8. Fine-Tuning: If necessary, perform fine-tuning to further improve image alignment. This may involve manually adjusting parameters or using optimization algorithms to minimize some error metric.

Stereo rectification is vital for reducing complexity and computational costs in binocular stereo vision systems, allowing for a simple one-dimensional search to find correspondences, since corresponding points only move along the same horizontal line. The rectified im-

ages provide a simpler starting point for 3D reconstruction, object detection, and other computer vision tasks.

(b)

在图像处理中，线 $Ax + By + C = 0$ 在图像平面上的法向量 n 可以用线的系数来表示。对于这条线，法向量 n 是指垂直于线的向量。在三维空间中，如果考虑焦距 f，这个法向量可以表示为：

$$n = \pm N \begin{pmatrix} A \\ B \\ C/f \end{pmatrix}$$

这里的 N 是一个归一化因子，用于确保向量 n 的长度为 1。具体来说，n 的每个分量都是对应直线方程中系数的比例表示，其中 C 被除以焦距 f 以考虑到图像平面与摄像机成像平面之间的关系。归一化因子 N 可以通过以下方式计算：

$$N = \frac{1}{\sqrt{A^2 + B^2 + \left(\frac{C}{f}\right)^2}}$$

因此，向量 n 是直线 $Ax + By + C = 0$ 在三维空间中的一个表示，其中 x 和 y 是图像平面上的坐标，f 是摄像机的焦距。这种表示在计算图像中直线的三维定位或者其他视觉任务中是非常有用的。

4.

(a)

运动视差是指当观察者移动时，由于观察点的变化而导致观察到的场景物体相对位置的变化。对于一个空间直线，这个直线在图像平面上的投影会根据摄像机的移动而改变位置。

当摄像机沿着某一方向进行纯平移时，从摄像机的两个不同位置观察同一空间直线，这条直线在两个图像平面上的投影会有不同的位置。由于不同深度的物体其视差变化的幅度不同，因此通过比较两个位置观察到的直线的变化，可以推断出直线在三维空间中的位置和方向。

具体来说，对于一个远离摄像机的空间直线，当摄像机平移时，这条直线在图像平面上的投影变化（视差）相对较小。相比之下，一个接近摄像机的空间直线，其在图像平面上的投影变化（视差）将会更大。这种视差变化可以用来估计物体的深度信息。

对于一个空间直线，可以用 P-向量来表示。P-向量是一个从摄像机中心到空间直线上一点的向量，并且它是垂直于该直线在图像平面上投影的。通过分析摄像机在两个不同位置观察到的 P-向量的变化，可以估计出空间直线的方向和位置。

在实践中，通过对两个图像之间的视差进行量化分析，然后将这些视差映射到三维空间中，从而估计出空间直线的准确位置。这个过程需要精确的摄像机校准，包括其内部和外部参数，以确保估计的准确性。此外，还可能需要考虑其他因素，如摄像机的畸变校正和光线的折射。

2021 - 2022

1.

(a)

在数字图像处理中，执行平滑操作之前应用微分操作的需要主要是因为图像通常包含噪声，这些噪声在微分过程中会被放大。微分操作，如梯度计算，旨在高亮变化显著的区域，例如边缘。如果直接对噪声图像进行微分，噪声可能会被误认为是边缘。

平滑操作可以减少图像中的随机噪声，从而在不影响图像中重要结构的情况下，使微分操作更有效。平滑后，微分操作更有可能识别出真正的边缘而不是噪声。

高斯平滑是一种特别受欢迎的平滑操作，它有几个吸引人的特性：

1. **边缘保持**：高斯平滑能够在去除噪声的同时保持边缘结构，这是因为高斯函数在中心附近权重较大，在远离中心的地方权重较小。

2. **线性和分离性**：高斯函数是线性的，这意味着可以通过卷积来实现，并且可以分离为水平和垂直两个一维高斯函数的卷积，这样可以提高计算效率。

3. **尺度变换**：高斯平滑允许在不同尺度（通过改变高斯核的标准差）上平滑图像，提供了尺度空间的概念。

4. **自然性**：高斯函数在自然和人类视觉中占有重要位置，因为它是许多自然过程的结果，例如由于光的扩散而导致的模糊效果，它也是人类视觉系统处理图像时的一个近似模型。

高斯平滑操作的数学表达式为：

$$G(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}}$$

其中，σ 是高斯函数的标准差，它决定了平滑的程度。在二维图像处理中，这个公式将扩展到两个维度。使用高斯滤波器进行平滑是边缘检测和图像分割等任务的常用预处理步骤。

(b)

在执行一阶梯度计算时，最计算效率的平滑过程是使用分离的高斯滤波器。这种方法依赖于高斯函数的可分离性质，即二维高斯函数可以表示为两个一维高斯函数的乘积。

对于一个二维高斯函数 $G(x, y)$，它可以写作：

$$G(x, y) = G(x) \cdot G(y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

这意味着我们可以先在一个方向（例如，水平方向）上对图像应用一维高斯滤波器，然后在另一个方向（例如，垂直方向）上应用另一个一维高斯滤波器。这种方法比直接使用二维高斯滤波器在计算上要有效得多。

为了量化计算节省，我们可以考虑一个 $N \times N$ 的图像和一个 $M \times M$ 的二维高斯滤波器。如果直接使用二维滤波器，则需要 $N^2 \cdot M^2$ 次乘法和相加操作。但如果使用分离的滤波器，我们只需要 $N^2 \cdot M$ 次操作（首先 $N \cdot M$ 次操作用于每一行，然后 $N \cdot M$ 次操作用于每一列）。

因此，总的计算量从 $N^2 \cdot M^2$ 减少到 $N^2 \cdot M$。如果 M 很大，这个节省会非常显著。例如，如果 $M = N$，那么计算量会从 N^4 减少到 $2N^3$，这减少了大约一半的计算量。而对于更大的滤波器尺寸，节省会更加显著。

(c)

中轴变换（Medial Axis Transformation，MAT）也称为骨架化或细化算法，它用于将二值图像中的对象（即图像中的形状）减少到它们的骨架形式。一个形状的中轴是所有在对象边界上有多于一个最近点的点的集合。这种变换对于形状分析和特征提取非常有用，因为它捕捉了形状的拓扑和几何属性。

针对二值对象的 MAT 基细化算法的基本思想可以描述如下：

1. **距离图**：计算二值图像的距离变换。这个变换为二值对象中的每个像素赋值，该值表示它到最近背景像素的距离。在二值图像中，背景通常由 0 表示，对象由 1 表示。

2. **中轴**：通过寻找距离变换的局部最大值来识别中轴。这些点与至少两个边界点等距离，这意味着它们是形状结构的中心。

3. **细化**：迭代地从对象的边界去除像素，同时保持中轴。这个过程重复进行，直到只剩下骨架结构。这里的关键条件是去除像素不能改变原始形状的拓扑。因此，应用特定规则以确保只去除非关键像素。

4. **连通性**：通过保留形成形状的最小跨度结构的基本像素来确保骨架的连通性。

中轴变换的结果是原始形状的细化版本，代表了对象的“脊柱”或中心线。这个骨架可以用于进一步分析，如特征提取、识别或二值图像的压缩。

有几种不同的细化算法，每种算法都有不同的像素去除规则，以确保所得到的骨架尽可能接近中轴，同时保持拓扑并允许高效的计算。

5.

(a)

获取具有完整三维运动的优化点基运动视差包括以下几个步骤：

1. 特征检测：在一系列图像（或视频）中识别可以可靠跟踪的显著点。常用的特征检测算法包括 SIFT、SURF、ORB 等。

2. 特征匹配：在连续的图像帧之间匹配这些特征点。这可以通过比较特征点的描述子完成，常见的匹配算法有 FLANN、BFMatcher 等。

3. 运动估计：使用匹配的特征点来估计相机的运动。这通常涉及到求解一个透视-n-点（PnP）问题或使用迭代最近点（ICP）算法。

4. 运动分解：将估计的运动分解为平移和旋转组件。这通常涉及到使用奇异值分解（SVD）或其他数值分解方法。

5. 运动平滑：为了减少噪声和误差的影响，对估计的运动轨迹进行平滑处理。这可以通过卡尔曼滤波器、粒子滤波器或其他平滑技术实现。

6. 三维重建：根据跟踪的特征点和估计的运动，重建场景的三维结构。这通常涉及到三角测量方法。

7. 优化：使用捆绑调整（bundle adjustment）等技术对运动和结构进行联合优化，以提高运动视差的准确性。

8. 验证与错误检测：检测并剔除错误的匹配和估计，这可能通过 RANSAC 或其他一致性检测方法完成。

以上步骤为一般过程，具体实现可能根据应用场景和可用数据有所不同。在实践中，这些步骤需要结合相机校准数据和可能的场景约束来进行细化和调整。

Obtaining optimized point-based motion parallax with full 3D

ego-motion typically involves the following steps:

1. Feature Detection: Identify distinctive features or points in the sequence of images (or a video) that can be reliably tracked. Common algorithms include SIFT, SURF, ORB, and others.

2. Feature Matching: Match these feature points across successive image frames. This can be done by comparing feature descriptors, often using algorithms like FLANN or BFMatcher.

3. Motion Estimation: Estimate the camera motion using the matched feature points. This typically involves solving a Perspective-n-Point (PnP) problem or using Iterative Closest Point (ICP) algorithms.

4. Motion Decomposition: Decompose the estimated motion into its translational and rotational components. This often involves numerical methods like Singular Value Decomposition (SVD).

5. Motion Smoothing: To reduce the impact of noise and errors, smooth the estimated motion trajectory. Techniques like Kalman filters, particle filters, or other smoothing techniques may be applied.

6. 3D Reconstruction: Reconstruct the 3D structure of the scene based on tracked feature points and estimated motion, often involving triangulation methods.

7. Optimization: Refine the motion and structure estimation using techniques such as bundle adjustment to improve the accuracy of motion parallax.

8. Validation and Outlier Rejection: Detect and remove incorrect matches and estimations, possibly using consistency checks like RANSAC.

These steps are a general workflow, and specific implementations may vary depending on the application context and available data. In practice, these steps should be refined and adjusted in conjunction with camera calibration data and any potential scene constraints.

(b)

双目立体成像是运动视差问题的一个特例，它利用了两个平行摄像机拍摄同一场景的图像来确定场景中点的三维位置。这里是如何从一对平行双目图像对中的图像视差导出一个点的完整三维位置的步骤：

1. 确定视差：首先，需要确定场景中相同点在左右两个摄像机图像上的对应位置，并计算这两点之间的水平位置差，即视差 d。

2. 摄像机参数：需要知道摄像机的内参，包括焦距 f 和摄像机之间的基线距离 B（即两个摄像机中心之间的距离）。

3. 计算深度：使用以下公式计算点的深度 Z：

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

4. 计算水平和垂直位置：在计算了深度之后，可以使用相似三角形的原理来计算该点在水平和垂直方向上的位置。如果 (x_l, y_l) 和 (x_r, y_r) 分别是左右图像中点的像素坐标，那么点在三维空间中的位置 (X, Y, Z) 可以通过以下公式得到：

$$X = \frac{(x_l - c_x) \cdot Z}{f}$$

$$Y = \frac{(y_l - c_y) \cdot Z}{f}$$

其中 c_x 和 c_y 是摄像机图像平面的中心点坐标。

以上步骤描述了如何通过双目视差来计算场景中任意一点的三维位置。这种方法在机器视觉、自动驾驶汽车和三维重建等领域有广泛应用。

2022 - 2023

1.

(a)

图中的问题是关于数字图像处理的。它要求我们对一个 7x7 的数字图像应用一个 1x3 的线性移不变（LSI）滤波器，并假设图像外的值为 0。滤波器的掩模（mask）是 [1/3, 1/3, 1/3]。

滤波后的输出图像 $g_1(x, y)$ 如下所示（每个元素都是经过四舍五入的整数）：

$$\begin{array}{ccccccc} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 4 & 5 & 5 & 3 & 5 & 5 & 4 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 3 & 3 & 3 & 1 & 4 & 5 & 4 \\ 1 & 1 & 1 & 1 & 4 & 5 & 4 \\ 1 & 1 & 1 & 1 & 2 & 3 & 2 \end{array}$$

这是通过在原始图像上应用掩模并考虑到图像边界外的值为 0 来计算得到的。

对于一个给定的数字图像 $f(x, y)$，使用一个线性移不变（Linear Shift Invariant，简称 LSI）滤波器处理图像，可以通过卷积操作来完成。给定的 LSI 滤波器的大小为 1×3，并且有一个掩模（mask）$[1/3, 1/3, 1/3]$。

卷积操作的公式为：

$$g(x, y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} f(x - s, y - t) \cdot h(s, t)$$

其中，$g(x,y)$ 是输出图像，$f(x,y)$ 是输入图像，$h(s,t)$ 是滤波器的冲击响应，这里为掩模的值。在我们的例子中，由于掩模是一维的，卷积操作将沿着水平方向进行，每个输出像素值将是其对应的三个水平邻居像素值的加权平均。

由于滤波器的大小是 1×3，我们仅在水平方向上应用滤波器，公式简化为：

$$g_1(x,y) = \frac{1}{3}(f(x,y-1) + f(x,y) + f(x,y+1))$$

在图像边界，我们假设图像外的值为 0，即 $f(x,y) = 0$ 对于图像外的任意 (x,y)。

我们可以按照以下步骤计算每个像素的新值：

1. 对于图像中的每个像素，找到它水平方向上的左右邻居。
2. 如果这些邻居超出了图像的边界，它们的值视为 0。
3. 将这三个值相加，然后除以 3，即应用掩模。
4. 将得到的结果四舍五入到最接近的整数，因为图像像素的值通常是整数。

这个过程会对每个像素重复，生成一个新的 7x7 的图像，这就是滤波后的输出图像 $g_1(x,y)$。

(b)

中值滤波器是一种非线性滤波器，通常用于去除噪声，特别是所谓的椒盐噪声。它对于维持图像中的边缘也比线性滤波器更有效。中值滤波器的工作原理是在图像的每个像素上，将滤波器窗口内的所有像素值排序，并取中间的值作为输出图像在该位置的像素值。

对于一个给定的数字图像 $f(x,y)$，当我们使用一个尺寸为 1×3 的中值滤波器时，滤波器会覆盖每个像素及其水平邻居，然后选

择中间的数值。具体来说，中值滤波器的操作可以表示为：

$$g_2(x, y) = \text{median}\{f(x, y-1), f(x, y), f(x, y+1)\}$$

在图像边界，我们同样假设图像外的值为 0，即 $f(x, y) = 0$ 对于图像外的任意 (x, y)。

计算每个像素的新值的步骤如下：1. 对于图像中的每个像素，找到它水平方向上的左右邻居。2. 如果这些邻居超出了图像的边界，它们的值视为 0。3. 将这三个值进行排序，并取中间的值。4. 将这个中间值作为输出图像在该位置的像素值。

根据上述过程，应用中值滤波器后的输出图像 $g_2(x, y)$ 如下所示：

$$\begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 3 & 3 & 3 & 3 & 3 & 3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 3 & 3 \\ 0 & 0 & 0 & 0 & 3 & 3 & 3 \\ 0 & 0 & 0 & 0 & 3 & 3 & 3 \end{matrix}$$

这就是滤波器在每个像素上的作用结果。

(c)

当我们假设图像 $f(x, y)$ 包含一个水平线、一个垂直线和一个正方形，这些结构被噪声干扰时，我们可以使用不同的滤波器来尝试去除噪声并恢复原始结构。在这个场景中，我们比较了线性移不变滤波器 $g_1(x, y)$ 和中值滤波器 $g_2(x, y)$ 的效果。

线性滤波器（如之前使用的 LSI 滤波器）通常对噪声的平滑效果很好，但可能会模糊图像中的尖锐边缘。在我们的例子中，该滤波器通过对每个像素的水平邻居进行加权平均来工作，这有助于减少噪声，但也可能导致水平线变得模糊。

中值滤波器通过选择一个像素及其邻居的中值来工作，这种方法对于去除所谓的椒盐噪声特别有效，同时可以更好地保持边缘。对于图像中的水平和垂直线，中值滤波器可以在去除噪声的同时保持边缘的清晰度。

从计算出的两个过滤后的图像 $g_1(x, y)$ 和 $g_2(x, y)$ 来看，中值滤波器似乎更适合于保持图像结构，因为它能够维持更多的原始像素值，特别是在水平和垂直线上。而线性滤波器则在整个图像上提供了一种更平滑的效果，但可能会在边缘处产生模糊。

在实际应用中，选择哪种滤波器取决于特定的应用场景和对图像保真度与去噪效果的需求。如果保持边缘清晰度是首要目标，中值滤波器通常是更好的选择。如果需要平滑噪声且边缘模糊是可接受的，线性滤波器可能更合适。

3.

(a)

这个问题涉及到贝叶斯决策理论中的错误决策概率最小化。具体步骤如下：

假设我们有类 ω_i 和对应的数据 x。我们想要基于 x 做出决策，即判断 x 属于哪一个类。

1. **先验概率** $p_{\omega_i}(\omega_i)$：这是在观察数据之前，我们认为 x 属于类 ω_i 的概率。

2. **类条件概率密度函数** $p_x(x|\omega_i)$：给定 x 属于类 ω_i 的情况下，

x 的概率密度函数。

3. **后验概率** $p_{\omega_i}(\omega_i|x)$：这是在观察到数据 x 后，更新的 x 属于类 ω_i 的概率。

我们的目标是找到一个决策规则，使得错误决策的概率 $p(e_i|x)$ 最小化。

错误决策概率 $p(e_i|x)$ 是在我们决定 x 属于 ω_i 时实际上它不属于这个类的概率。这个概率可以通过所有可能的错误决策概率的总和来计算：

$$p(e_i|x) = \sum_{j \neq i} p_{\omega_j}(\omega_j|x)$$

我们想要最小化 $p(e_i|x)$，根据贝叶斯决策理论，我们应该选择具有最大后验概率 $p_{\omega_i}(\omega_i|x)$ 的类 ω_i。也就是说，我们选择 ω_i 使得：

$$p_{\omega_i}(\omega_i|x) > p_{\omega_j}(\omega_j|x) \quad \forall j \neq i$$

根据贝叶斯公式，后验概率可以用以下方式计算：

$$p_{\omega_i}(\omega_i|x) = \frac{p_x(x|\omega_i) \cdot p_{\omega_i}(\omega_i)}{p_x(x)}$$

其中 $p_x(x)$ 是边际概率密度函数，可以通过所有类的先验概率和类条件概率密度的乘积的总和来计算：

$$p_x(x) = \sum_k p_x(x|\omega_k) \cdot p_{\omega_k}(\omega_k)$$

因此，我们可以选择类 ω_i 使得 $p_x(x|\omega_i) \cdot p_{\omega_i}(\omega_i)$ 是最大的，从而最小化错误决策概率 $p(e_i|x)$。这就是所谓的最大后验概率（MAP）决策准则。

总结一下，为了最小化错误决策的概率，我们应该选择后验概率最大的类。这就要求我们计算每个类的后验概率，并选择具有最大后验概率的类作为我们的决策。

(b)

在这个问题中，我们没有直接的先验概率 $p(\omega_i)$、类条件概率密度 $p(x|\omega_i)$ 或者边际概率密度 $p(x)$，但我们可以使用可用的训练数据来估计这些概率。以下是如何根据估计值 $\hat{p}(\omega_i)$ 和 $\hat{p}(x|\omega_i)$ 导出决策规则的步骤：

1. **估计先验概率** $\hat{p}(\omega_i)$：我们可以通过计算训练数据中属于类 ω_i 的样本数占总样本数的比例来估计先验概率。

2. **估计类条件概率密度** $\hat{p}(x|\omega_i)$：我们可以使用训练数据中类 ω_i 的样本来估计这个条件概率密度。如果数据符合某种已知分布（如高斯分布），可以通过计算该类样本的均值和方差来估算。

3. **决策规则**：我们的目标是找到一个决策规则，使得基于 $\hat{p}(\omega_i)$ 和 $\hat{p}(x|\omega_i)$ 的错误决策概率最小化。

- 我们可以使用贝叶斯定理来估计后验概率 $\hat{p}(\omega_i|x)$，即在给定 x 的情况下样本属于 ω_i 的概率：

$$\hat{p}(\omega_i|x) = \frac{\hat{p}(x|\omega_i) \cdot \hat{p}(\omega_i)}{\sum_j \hat{p}(x|\omega_j) \cdot \hat{p}(\omega_j)}$$

- 由于 $\hat{p}(x)$ 对于所有类是一个常数，我们可以简化决策规则为选择使得 $\hat{p}(x|\omega_i) \cdot \hat{p}(\omega_i)$ 最大化的类 ω_i。

总结来说，当没有直接的概率可用时，我们可以使用训练数据来估计这些概率，并应用最大后验概率（MAP）决策准则，即选

择使得 $\hat{p}(x|\omega_i) \cdot \hat{p}(\omega_i)$ 最大化的类 ω_i 作为我们的决策。这样的决策规则将使得基于估计概率的错误决策概率最小化。

(c)

这个问题涉及到 k-最近邻（k-NN）分类器的决策规则。在 k-NN 分类器中，一个样本被分配到 k 个最近邻中最常见类别。这里的“最近邻”是根据某种距离度量（通常是欧氏距离）来确定的。要使用这种分类器，我们不需要估计先验概率 $p(\omega_i)$ 或类条件概率密度 $p(x|\omega_i)$，而是直接基于训练数据的分布来进行决策。

k-NN 分类器的决策规则可以理解为一个非参数方法，即不依赖于概率分布的参数估计。但是，如果我们想要将 k-NN 分类器的工作原理与概率估计联系起来，我们可以按照以下步骤进行：

1. **局部类概率估计**：在 k-NN 分类器中，我们可以认为局部区域（即 k 个最近邻所在的区域）内的点对类概率 $p(\omega_i|x)$ 的估计有贡献。

2. **局部密度估计**：$p(x)$ 可以被视为在 x 附近的点的局部密度。在 k-NN 中，这个密度可以通过查看 x 附近有多少训练样本来估计。

3. **局部类条件概率密度估计**：$p(x|\omega_i)$ 可以被视为给定类 ω_i 下，x 附近点的局部密度。在 k-NN 中，这可以通过检查 x 附近属于类 ω_i 的训练样本的数量来估计。

当我们使用 k-NN 方法时，实际上是在假设局部区域内的概率密度是均匀的。因此，我们可以简化概率的估计，决策规则就变成了选择最近的 k 个邻居中最常见类别。

总结来说，在 k-NN 分类器中，我们不直接估计概率，而是通过测量距离和计数最近邻来做决策。这个方法基于的假设是局部

区域内的概率密度是相对均匀的，所以类别由周围的 k 个最近训练样本的多数类别决定。这种决策规则实质上是通过局部投票来最小化错误决策的概率。

5.

(a)

在射影几何中，四点的交比（cross-ratio）是保持不变的，即使这些点经过了射影变换。为了证明交比是射影不变量，我们需要使用射影几何的一些基本概念和性质。

给定直线上的四点 A, B, C, D，它们的交比定义为：

$$[A, B, C, D] = \frac{AC}{BC} : \frac{AD}{BD}$$

假设这四点经过射影变换映射到另外四点 a, b, c, d。根据射影变换的性质，任意两点间的比例通过相同的常数 k 缩放。设 A 映射到 a，B 映射到 b，依此类推，我们可以写出：

$$\frac{AC}{BC} = k\frac{ac}{bc} \quad \text{和} \quad \frac{AD}{BD} = k\frac{ad}{bd}$$

因此，射影变换后的交比 $[a, b, c, d]$ 是：

$$[a, b, c, d] = \frac{ac}{bc} : \frac{ad}{bd}$$

现在，我们将 k 代入上述等式：

$$[a, b, c, d] = \frac{k \cdot AC/BC}{k \cdot AD/BD} = \frac{AC/BC}{AD/BD} = [A, B, C, D]$$

这证明了射影变换不改变四点的交比。这个证明是建立在射影变换保持线性比例的性质上的，而这个性质是射影几何中的基本

假设。在实际应用中，这可以通过构造适当的射影变换矩阵来验证。射影变换通常涉及到复杂的矩阵乘法和坐标变换，但在本质上，上述简单的代数关系揭示了交比不变性的核心。

(b)

在立体视觉系统中，极线约束是用来限制在第二个图像平面上可能找到匹配点的位置。极线约束基于极线几何，这是立体视觉中的一个核心概念。

假设点 P 在两个图像平面上的投影分别为 p 和 p'，如图所示。两个摄像机的内参矩阵都是 K，并且第二个摄像机相对于第一个摄像机的外参矩阵是 $[R, t]$。本质矩阵 E 和基础矩阵 F 之间的数学关系如下：

本质矩阵 E: 本质矩阵 E 是一个 3×3 矩阵，它编码了两个摄像机之间的旋转和平移关系。本质矩阵 E 由旋转矩阵 R 和平移向量 t 的外积 $[t]_x$ 构成：

$$E = [t]_x R$$

这里，$[t]_x$ 是平移向量 t 的反对称矩阵。

基础矩阵 F: 基础矩阵 F 同样是一个 3×3 矩阵，它关联了一对立体图像中的对应点，并且是在像素坐标系下工作的。基础矩阵 F 可以从本质矩阵 E 通过摄像机的内参矩阵 K 转换得到：

$$F = K^{-T} E K^{-1}$$

极线约束表达为，对于每个点 p 和 p'，都有：

$$p'^T F p = 0$$

或者，如果使用本质矩阵，转换为归一化的图像坐标：

$$p'^T E p = 0$$

在这里，p 和 p' 是归一化图像坐标，可以通过内参矩阵 K 转换得到：

$$p = K^{-1} p_{\text{image}}$$

$$p' = K^{-1} p'_{\text{image}}$$

其中 p_{image} 和 p'_{image} 分别是图像平面上的点 p 和 p' 的像素坐标。

通过这些关系，我们可以利用极线约束来限制对应点的搜索。在第一个图像中找到点 p 后，其对应点 p' 必须在第二个图像中的极线上。这极大地减少了特征匹配的搜索空间，并且是三维重建中的一个关键步骤。

为了使用本质矩阵来施加极线约束，需要将图像坐标归一化到摄像机坐标系，然后使用上述 $p'^T E p = 0$ 的约束。这意味着，即使我们没有摄像机的具体内参信息，也可以使用基础矩阵直接在像素坐标系下应用极线约束。这在实践中非常有用，特别是在内参未知或者难以精确测量的情况下。

(c)

在立体视觉中，双目摄像机系统可以通过测量同一场景点在两个摄像机图像上的视差（disparity）来估计该点的深度。给定立体摄像头的基线 B（两个摄像机之间的距离），焦距 f 和某点的视差 d，可以使用以下视差方程来计算该点的深度 Z：

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

如果视差测量存在误差，即实际的视差是 $d + \Delta d$，其中 Δd 是视差的误差，那么深度的估计也会存在误差。为了估计因视差误差造成的深度估计误差 ΔZ，我们可以对深度方程 $Z(d)$ 进行泰勒级数展开（在 d 处展开）：

$$Z(d + \Delta d) \approx Z(d) + \frac{dZ}{dd}\Big|_d \cdot \Delta d$$

由于 $Z = \frac{f \cdot B}{d}$，对 d 求导得：

$$\frac{dZ}{dd} = -\frac{f \cdot B}{d^2}$$

因此，估计的深度误差 ΔZ 可以用 Δd 表示为：

$$\Delta Z \approx -\frac{f \cdot B}{d^2} \cdot \Delta d$$

这给出了深度估计误差 ΔZ 和视差误差 Δd 之间的关系。这里的近似是基于假设 Δd 相对于 d 是小的，使得泰勒级数可以截断至一阶项。

需要注意的是，这个关系表明视差的微小误差会导致深度估计误差的平方增加，这意味着对于远处的物体（即 d 较小），即使是小的视差误差也会导致较大的深度估计误差。这是双目立体视觉系统中一个重要的考虑因素。

(d)

在计算机视觉中，光流估计是根据相邻两帧图像之间像素点的运动来计算运动模式的过程。亮度恒定性是估计光流的一个关键假设，它假设在很短的时间内，一个移动的像素点的亮度不会改变。

给定时刻 t 下一个像素点的位置为 (x, y) 且亮度为 I，在时刻 $t+1$ 该点移动到了 $(x+u, y+v)$ 的位置，亮度恒定性假设可以表示为：

$$I(x, y, t) = I(x+u, y+v, t+1)$$

为了导出亮度恒定性约束方程，我们可以对亮度 I 在 $(x+u, y+v)$ 处进行泰勒展开，并忽略高阶项，得到：

$$I(x+u, y+v, t+1) \approx I(x, y, t) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \frac{\partial I}{\partial t}$$

结合亮度恒定假设，我们可以得到光流方程：

$$\frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \frac{\partial I}{\partial t} = 0$$

这个方程也被称为光流约束方程。

亮度恒定性假设存在一些局限性：

1. **亮度变化**：实际中，由于照明变化、阴影和反射等因素，像素的亮度可能会改变。

2. **遮挡问题**：一个物体的移动可能导致遮挡，使得某些像素点在另一帧中不可见。

3. **大运动**：光流估计通常假设像素点的移动是小的。如果两帧之间的运动很大，线性化的泰勒展开可能不准确。

4. **几何畸变**：摄像机的运动可能导致视角变化，从而改变了像素点的亮度。

除了亮度恒定性假设，估计光流的其他关键假设包括：

- **空间一致性**：相邻像素点的运动是相似的，即光流场是平滑的。

- **时间持续性**：运动不会在短时间内发生剧烈变化。

光流估计可能成功的数学条件包括：

- **亮度梯度**：像素点在运动方向上有足够的亮度梯度，这样才能从光流约束方程中解出 u 和 v。
- **多个约束**：通常一个光流方程是不足以解出两个未知数的，需要使用多个约束，或者是附加的平滑性假设，来得到一个合理的解。
- **边界条件**：在图像的边界处，需要特别处理光流估计，因为边界上的像素点没有足够的邻域信息。

光流算法的设计需要综合考虑这些条件和假设，以便在各种情况下都能得到可靠的运动估计。

EE7403 2020 - 2021

1.

(a)

看来在尝试解决方程时出现了问题，因为涉及到了单位阶跃函数（Heaviside function）和符号计算库不支持的 Dirac delta 函数。由于这个问题，我们需要手动确定直方图函数非零的最小值（f_{min}）和最大值（f_{max}）。

根据题目中的直方图函数 $P_f(f)$，我们可以看到它是由几个单位阶跃函数的和组成，其中每个阶跃函数都是在特定的灰度级 f 值时变为非零。这些特定的 f 值是 $f = 0.7, 0.8, 0.9, 1$。

灰度级 f 的最小值（f_{min}）将是直方图函数第一次变为非零的值，即 $f = 0.7$。灰度级 f 的最大值（f_{max}）将是直方图函数最后一次变为非零的值，即 $f = 1$。由于 c 是一个正实数，我们可以假设在 $f > 1$ 之后直方图函数将始终大于零。

然后，我们需要确定在这个范围内有多少不同的灰度级。在数字图像中，灰度级通常是整数值，因此我们可以通过减去 f_{min} 从 f_{max} 并加一来计算这个范围内的灰度级数量（因为包括两端）。

所以，$f_{min} = 0.7, f_{max} = 1$ 以及灰度级的总数是 $f_{max} - f_{min} + 1 = 1 - 0.7 + 1 = 0.3 + 1 = 1.3$，但因为灰度级是整数，我们应该考虑 f_{min} 到 f_{max} 范围内的整数值。所以实际上是从 0.7 到 1，我们应该有 1 - 0.7 + 1 = 0.3 + 1 = 1.3，向上取整，应该有 2 个不同的灰度级（即 0.7 和 1.0）。

现在我们用这种方法来解决问题。

计算的结果显示，灰度级的最小值 f_{min} 为 0.7，最大值 f_{max} 为 1.0。在这个范围内，图像具有 2 个不同的灰度级。这是因为在数

字图像处理中，灰度级通常是整数，所以我们在 0.7 到 1.0 范围内有两个整数灰度级：0 和 1。

(b)

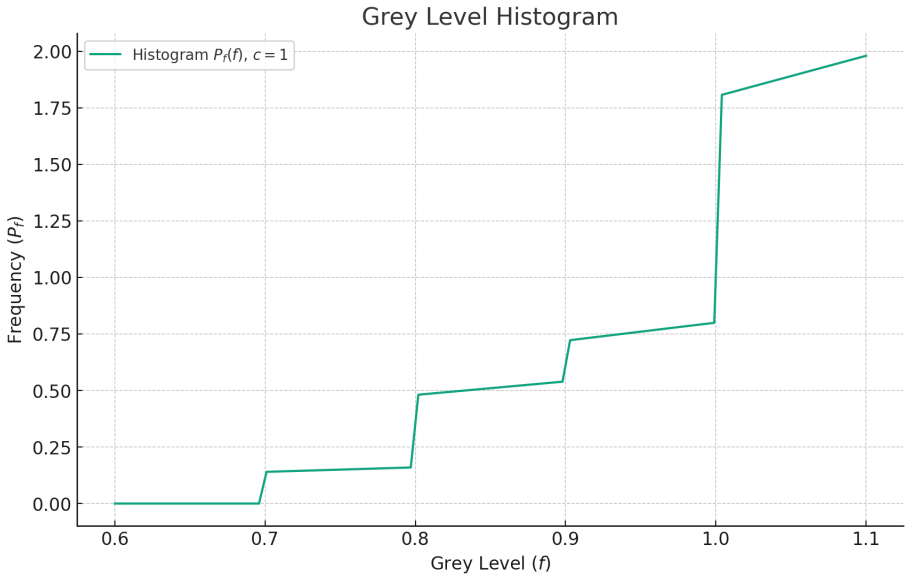

Figure 1: Caption

这是给定直方图函数 $P_f(f)$ 的图表。这个函数是根据图像的灰度级分布绘制的，其中灰度级 f 从 0.6 到 1.1。在 $f = 0.7$ 时，函数开始从零上升，每到一个新的阶跃点（0.8, 0.9, 和 1.0）函数值会有所增加。由于 c 是一个正实数，我们假定 $c = 1$ 进行绘图，所以在 $f = 1$ 之后，直方图函数值继续增加。这个图表显示了灰度级与它们出现频率之间的关系。

(c)

EE7403 2021 - 2022

1.

(a)

图中的问题是关于数字图像处理中滤波器冲激响应的一个问题。公式如下：

$$h(x,y) = \delta(x-1,y-1)+\delta(x-1,y)+\delta(x,y-1)-\delta(x+1,y)-\delta(x,y+1)-\delta(x+1,y+1)$$

其中 $\delta(x,y)$ 是二维冲激函数。它在 $x=0$ 和 $y=0$ 时值为 1，在其他位置的值为 0。

该问题要求绘制滤波器掩模（即滤波器核），并在掩模旁边标出 x 轴和 y 轴。

为了解决这个问题，我们可以按照以下步骤操作：

1. 创建一个 3x3 的矩阵，这代表了滤波器的尺寸。由于冲激函数的性质，大部分元素都将是 0。

2. 根据给定的冲激响应公式，在矩阵中填充非零值。其中 $\delta(x-1,y-1)$ 表示在矩阵中位置为 (1,1) 的元素值为 1，类似地 $\delta(x+1,y+1)$ 表示在矩阵中位置为 (3,3) 的元素值为-1，以此类推。

3. 绘制出这个矩阵，并在其旁边标出 x 轴和 y 轴。

下面是这个掩模的计算过程和结果：

1. 初始化一个 3x3 的矩阵：

$$M = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

2. 根据公式填充矩阵：

$$M = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -1 \end{bmatrix}$$

3. 标出 x 轴和 y 轴，通常 x 轴是水平方向，y 轴是垂直方向。

绘制完整的滤波器掩模如下：

$$\begin{array}{c} \\ \\ \text{x} \end{array} \begin{array}{c} \text{y} \\ \uparrow \\ \rightarrow \end{array} \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -1 \end{bmatrix}$$

注意，这里 δ 函数代表一个理想化的点，它在我们设定的中心点 (0,0) 有一个值，在其他所有点的值都是 0。在实际的离散系统中，这个函数会根据它的位置相对于中心点来映射到掩模矩阵的相应位置。

(b)

根据您上传的图片中的问题，我们需要表达滤波器输出图像 $g(x, y)$ 用输入图像 $f(x, y)$ 来表示。

在数字图像处理中，输出图像是通过将滤波器掩模应用于输入图像得到的。这个过程通常被称为卷积操作。根据前一个问题中给出的滤波器冲激响应 $h(x, y)$，我们可以写出以下卷积公式来计算 $g(x, y)$：

$$g(x, y) = f(x, y) * h(x, y)$$

这里的星号（*）表示二维卷积操作。将 $h(x,y)$ 的表达式代入上面的卷积公式，我们得到：

$$g(x,y) = f(x-1,y-1)+f(x-1,y)+f(x,y-1)-f(x+1,y)-f(x,y+1)-f(x+1,y+1)$$

这个公式意味着对于输出图像 $g(x,y)$ 中的每一个像素，它的值是由输入图像 $f(x,y)$ 中对应位置的像素值以及其邻域像素值按照滤波器掩模 $h(x,y)$ 加权和计算得到的。具体来说，我们从 $f(x,y)$ 的相邻像素中取值，并根据 $h(x,y)$ 中的系数进行加权。正系数意味着加上相应像素的值，负系数意味着减去相应像素的值。

这种类型的滤波器通常用于图像增强、边缘检测或其他图像处理任务。在实际应用中，需要注意边界处理，因为在图像的边缘，像素可能没有足够的邻居像素来进行完整的卷积计算。

(c)

要计算滤波器的频率响应，我们需要对其冲激响应 $h(x,y)$ 进行傅里叶变换。滤波器的冲激响应 $h(x,y)$ 已经给出，其频率响应 $H(u,v)$ 可以通过以下傅里叶变换得到：

$$H(u,v) = \mathcal{F}\{h(x,y)\}$$

这里的 $\mathcal{F}$ 表示二维傅里叶变换。给定的冲激响应是：

$$h(x,y) = \delta(x-1,y-1)+\delta(x-1,y)+\delta(x,y-1)-\delta(x+1,y)-\delta(x,y+1)-\delta(x+1,y+1)$$

因为傅里叶变换是线性的，我们可以单独对每个冲激函数 $\delta(x-a,y-b)$ 进行傅里叶变换，然后将它们相应的结果相加或相

减。冲激函数的傅里叶变换是一个相位项 $e^{-j2\pi(ua+vb)}$，其中 u 和 v 是频率变量。

所以我们有：

$$H(u,v) = e^{-j2\pi(u+v)} + e^{-j2\pi(u)} + e^{-j2\pi(v)} - e^{-j2\pi(-u)} - e^{-j2\pi(-v)} - e^{-j2\pi(-u-v)}$$

简化上面的公式，得到：

$$H(u,v) = e^{-j2\pi(u+v)} + e^{-j2\pi u} + e^{-j2\pi v} - e^{j2\pi u} - e^{j2\pi v} - e^{j2\pi(u+v)}$$

这就是所求的频率响应表达式。实际计算时，这个公式可以进一步简化，因为 $e^{-j2\pi u}$ 和 $e^{j2\pi u}$ 是一对共轭复数，它们的和是 $2\cos(2\pi u)$，同样对 v 也是如此。

最终的频率响应表达式为：

$$H(u,v) = 2[\cos(2\pi(u+v)) - \cos(2\pi u) - \cos(2\pi v)]$$

这个表达式描述了滤波器如何响应不同频率的输入信号，可以用来分析滤波器的特性，比如它是低通、高通、带通还是带阻滤波器。

(d)

要分析滤波器的性质，我们需要考虑其在空间域和频率域的行为。在前面的部分中，我们已经得到了滤波器的冲激响应 $h(x,y)$ 和其频率响应 $H(u,v)$。现在我们可以根据这些信息来分析滤波器的属性：

1. **空间域属性**：

- 由于滤波器冲激响应包含正值和负值，这表明滤波器可能进行某种形式的高通滤波，强调图像中的高频细节，比如边缘和纹理。
- 冲激响应中的系数之和为 0，这意味着滤波器是一个平均值为 0 的滤波器，这通常是边缘检测或图像锐化滤波器的特征。

2. **频率域属性**：

- 我们之前得到的频率响应 $H(u, v)$ 表示滤波器如何响应不同频率的输入。由于 $H(u, v)$ 包含余弦项，它将在某些频率上有正响应，在另一些频率上有负响应。
- 频率响应中的余弦函数意味着滤波器会在特定的频率上产生强化或抑制的效果。当 u 和 v 的值导致余弦函数值为 1 时，滤波器对应的频率成分会被强化，当余弦函数值为-1 时，则被抑制。

3. **特殊频率的响应**：

- 当 $u = v = 0$（直流分量）时，频率响应 $H(0, 0) = 2[\cos(0) - \cos(0) - \cos(0)] = -2$，这表明直流分量被抑制，进一步说明了滤波器的高通性质。
- 对于其他特定频率，比如 $u = 0.5$ 和 $v = 0.5$，滤波器的响应将取决于余弦项的值。

总结来说，这个滤波器的特点是它可能增强图像中的高频信息，如边缘和纹理，同时抑制低频信息，如平滑区域。这些特性使得它可能在图像处理中用作边缘检测或图像锐化工具。然而，具体的滤波器效果也取决于应用于的图像内容和所处理的具体任务。

EE7403 2022 - 2023

2.

(a)

图中的问题是关于统计模式识别中的分类器设计问题，要求我们基于给定的先验概率和高斯类条件概率密度函数来推导出能够获得最高分类准确度的分类器。

给定的类先验概率为 $p(\omega_i)$，类条件概率密度函数 $p(\mathbf{x}|\omega_i)$ 对于每个类别 $\omega_i, i = 1, 2, ..., c$ 是一个高斯函数：

$$p(\mathbf{x}|\omega_i) = \frac{1}{(2\pi)^{d/2}|\mathbf{\Sigma}_i|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_i)^T \mathbf{\Sigma}_i^{-1}(\mathbf{x} - \boldsymbol{\mu}_i)\right)$$

其中，$\mathbf{x}$ 是一个 d-维的特征向量，$\boldsymbol{\mu}_i$ 是第 i 类的均值向量，$\mathbf{\Sigma}_i$ 是第 i 类的协方差矩阵。

为了推导出分类器，我们可以使用贝叶斯决策理论中的最大后验概率 (MAP) 准则。根据 MAP 准则，对于一个未知类别的样本 $\mathbf{x}$，我们应该选择使后验概率 $p(\omega_i|\mathbf{x})$ 最大化的类别 ω_i。

后验概率可以通过贝叶斯定理来计算：

$$p(\omega_i|\mathbf{x}) = \frac{p(\mathbf{x}|\omega_i)p(\omega_i)}{p(\mathbf{x})}$$

因为 $p(\mathbf{x})$ 是对所有类别的总概率，对于所有 ω_i 是常数，所以我们可以忽略它不影响最大化的过程。我们的目标是找到最大化 $p(\mathbf{x}|\omega_i)p(\omega_i)$ 的 ω_i。

为了简化计算，我们可以考虑对数似然比，我们选择使下列函数最大的 ω_i：

$$g_i(\mathbf{x}) = \ln p(\mathbf{x}|\omega_i) + \ln p(\omega_i)$$

代入高斯分布，我们得到：

$$g_i(\mathbf{x}) = -\frac{1}{2}\ln|\mathbf{\Sigma}_i| - \frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T\mathbf{\Sigma}_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i) + \ln p(\omega_i)$$

我们需要计算每个类 ω_i 的这个函数，并选择 $g_i(\mathbf{x})$ 最大的类别作为 $\mathbf{x}$ 的类别。

如果假设所有类别的协方差矩阵相同（即 $\mathbf{\Sigma}_i = \mathbf{\Sigma}$），那么问题会进一步简化，因为 $-\frac{1}{2}\ln|\mathbf{\Sigma}_i|$ 项对所有 i 都是常数，并且可以忽略。这样，我们只需要最大化 $-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T\mathbf{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}_i) + \ln p(\omega_i)$。

最终，分类器的决策规则为选择 $g_i(\mathbf{x})$ 最大的类别 ω_i。这个分类器称为二次判别分析器（QDA），如果协方差矩阵相同，则简化为线性判别分析器（LDA）。

(b)

图中的问题要求我们使用给定的训练样本来设计在第一部分中推导出的分类器。我们有六个训练样本，前三个样本来自类别 1，后三个样本来自类别 2。

训练样本如下：

- 类别 1: $\mathbf{x}_1 = (1,1)^T$, $\mathbf{x}_2 = (1,2)^T$, $\mathbf{x}_3 = (2,1)^T$
- 类别 2: $\mathbf{x}_4 = (2,2)^T$, $\mathbf{x}_5 = (2,3)^T$, $\mathbf{x}_6 = (3,2)^T$

首先，我们需要计算两个类别的均值向量 $\boldsymbol{\mu}_1$ 和 $\boldsymbol{\mu}_2$，以及各自的协方差矩阵 $\mathbf{\Sigma}_1$ 和 $\mathbf{\Sigma}_2$（如果假设协方差矩阵相同，则计算合并的协方差矩阵 $\mathbf{\Sigma}$）。

均值向量 $\boldsymbol{\mu}_i$ 的计算公式为：

$$\boldsymbol{\mu}_i = \frac{1}{n_i} \sum_{\mathbf{x} \in \omega_i} \mathbf{x}$$

其中 n_i 是类别 ω_i 中的样本数量。

对于每个类别的协方差矩阵 $\boldsymbol{\Sigma}_i$，计算公式为：

$$\boldsymbol{\Sigma}_i = \frac{1}{n_i - 1} \sum_{\mathbf{x} \in \omega_i} (\mathbf{x} - \boldsymbol{\mu}_i)(\mathbf{x} - \boldsymbol{\mu}_i)^T$$

计算两个类别的均值向量和协方差矩阵后，我们可以使用第一部分中推导出的决策规则来设计分类器。这意味着对于一个新的观测 $\mathbf{x}$，我们将计算 $g_i(\mathbf{x})$ 对于每个类别 ω_i，然后选择 $g_i(\mathbf{x})$ 最大的类别作为 $\mathbf{x}$ 的预测类别。

让我们现在计算这些统计量并设计分类器。

根据提供的训练样本，我们计算出以下统计量：

- 类别 1 的均值向量 $\boldsymbol{\mu}_1$：$[1.33, 1.33]$
- 类别 2 的均值向量 $\boldsymbol{\mu}_2$：$[2.33, 2.33]$
- 类别 1 的协方差矩阵 $\boldsymbol{\Sigma}_1$：

$$\begin{bmatrix} 0.33 & -0.17 \\ -0.17 & 0.33 \end{bmatrix}$$

- 类别 2 的协方差矩阵 $\boldsymbol{\Sigma}_2$：

$$\begin{bmatrix} 0.33 & -0.17 \\ -0.17 & 0.33 \end{bmatrix}$$

- 类别 1 和类别 2 的先验概率都是 0.5，因为训练样本中两个类别的数量是相等的。

注意到两个类别的协方差矩阵是相同的，这意味着我们可以使用线性判别分析（LDA）作为分类器。下一步是根据这些统计量构造分类函数 $g_i(\mathbf{x})$。因为协方差矩阵相同，我们可以简化 $g_i(\mathbf{x})$ 函数，仅考虑线性项和常数项。具体来说，对于线性判别分析，我们的决策函数 $g_i(\mathbf{x})$ 可以简化为：

$$g_i(\mathbf{x}) = \mathbf{x}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_i - \frac{1}{2} \boldsymbol{\mu}_i^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_i + \ln p(\omega_i)$$

既然 $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2$ 和 $p(\omega_1) = p(\omega_2)$，我们只需要计算 $\mathbf{x}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_i - \frac{1}{2} \boldsymbol{\mu}_i^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_i$ 的差异来决定分类。因为先验概率相等，所以 $\ln p(\omega_i)$ 也可以省略。

我们现在使用这些统计量来构造 LDA 分类器。

使用计算得到的 LDA 分类器对测试样本 $\mathbf{x} = (2, 2)^T$ 进行分类，我们得到以下结果：

- 类别 1 的决策分数：21.33
- 类别 2 的决策分数：23.33

因为类别 2 的决策分数更高，所以测试样本 $\mathbf{x} = (2, 2)^T$ 被分类为类别 2。这表明我们的分类器会选择具有更高决策函数值的类别。根据这个方法，可以对任何新的观测数据 $\mathbf{x}$ 使用 $g_1(\mathbf{x})$ 和 $g_2(\mathbf{x})$ 的值来进行分类。