

第 1 周教案：计算机视觉导论与图像基础

计算机视觉课程组

1 基本信息

周次	第 1 周
主题	计算机视觉导论与图像基础
学时	3 学时（160 分钟）
故事问题	机器是怎么“看见”试卷的？
OBE 目标	A1-图像处理：能对图像进行读取、显示、保存操作

2 教学目标

1. 知识目标：

- 理解计算机视觉的定义与应用领域
- 掌握图像的数字表示方法（像素、RGB、灰度图）

2. 能力目标：

- 能够使用 OpenCV 读取、显示、保存图像
- 能够访问和修改图像的像素值
- 能够实现基础的图像滤镜效果

3. 素养目标：

- 建立对计算机视觉课程的兴趣和期待
- 理解从图像到数字的转换思维

3 教学重点与难点

教学重点

- 图像的数字表示：像素与 RGB 色彩空间
- OpenCV 基础操作：imread、imshow、imwrite
- 图像数据结构：numpy 数组

教学难点

- RGB 通道的理解与分离
- 图像数组索引与切片操作

4 教学过程设计

4.1 环节一：课程导论（40 分钟）

4.1.1 1.1 什么是计算机视觉？（10 分钟）

引入问题：

- 人眼是如何看到东西的？
- 机器能不能像人一样“看见”世界？

CV 定义：

计算机视觉是一门研究如何使机器“看”的科学，指用摄影机和电脑代替人眼对目标进行识别、跟踪和测量等机器视觉，并进一步做图形处理。

4.1.2 1.2 CV 应用案例展示（20 分钟）

领域	应用案例
安防监控	人脸识别、行为分析、异常检测
自动驾驶	车道检测、障碍物识别、交通标志识别
医疗影像	肿瘤检测、病灶定位、影像辅助诊断
工业检测	缺陷检测、尺寸测量、质量分拣
教育领域	自动阅卷系统、作业批改

演示：播放 CV 应用短视频，展示人脸识别、自动驾驶等实际应用

4.1.3 1.3 课程介绍（10 分钟）

课程目标：造一个能改卷子的 AI 助教

课程路线图：

1. 装上眼睛——图像基础
2. 学会识别——三种题型（选择、判断、简答）
3. 成为助教——完整阅卷系统

4.2 环节二：图像的数字表示（50 分钟）

4.2.1 2.1 图像是什么？（10 分钟）

物理图像 → 数字图像

- 物理图像：连续的光信号分布
- 数字图像：离散的像素点阵

类比：像马赛克拼图，每个小格子就是一个像素

4.2.2 2.2 像素与分辨率（10 分钟）

概念：

- 像素（Pixel）：图像的最小单位
- 分辨率：图像的尺寸（宽 × 高）
 - 例：1920×1080 表示宽 1920 像素，高 1080 像素

互动问题：手机拍照的像素越高越好吗？

4.2.3 2.3 RGB 色彩空间（20 分钟）

三原色原理：

- R（Red）：红色通道
- G（Green）：绿色通道
- B（Blue）：蓝色通道

数值范围：0-255（8 位无符号整数）

R	G	B	颜色
255	0	0	红色
0	255	0	绿色
0	0	255	蓝色
255	255	255	白色
0	0	0	黑色
255	255	0	黄色

- 0 = 最暗
- 255 = 最亮

混合示例：

灰度图：单通道图像，0= 黑，255= 白

4.2.4 2.4 Jupyter 实验：查看图像数据（10 分钟）

Listing 1: 查看图像基本信息

```
import cv2
import numpy as np

# 读取图像
img = cv2.imread('test.jpg')

# 查看图像信息
print(f"图像尺寸: {img.shape}") # (height, width, channels)
print(f"数据类型: {img.dtype}") # uint8
print(f"总像素数: {img.size}")

# 访问单个像素
pixel = img[100, 100] # [B, G, R]
print(f"像素值: {pixel}")
```

4.3 环节三：OpenCV 入门（40 分钟）

4.3.1 3.1 OpenCV 简介（5 分钟）

- 全称：Open Source Computer Vision Library
- 特点：开源、跨平台、功能强大

- 语言支持：C++、Python、Java 等

4.3.2 3.2 环境搭建（10 分钟）

安装 OpenCV:

```
pip install opencv-python
pip install opencv-python-headless # 无GUI环境
pip install numpy matplotlib
```

Jupyter 环境配置:

```
# 在Jupyter中显示图像
import cv2
import matplotlib.pyplot as plt

# OpenCV读取的是BGR格式，需要转换为RGB
img = cv2.imread('test.jpg')
img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

plt.imshow(img_rgb)
plt.axis('off')
plt.show()
```

4.3.3 3.3 基础操作（25 分钟）

1. 读取图像（10 分钟）:

```
# 读取彩色图像
img_color = cv2.imread('test.jpg', cv2.IMREAD_COLOR)

# 读取灰度图像
img_gray = cv2.imread('test.jpg', cv2.IMREAD_GRAYSCALE)

# 读取原始图像（包含alpha通道）
img_alpha = cv2.imread('test.png', cv2.IMREAD_UNCHANGED)
```

2. 显示图像（10 分钟）:

```
# 方法1：使用cv2.imshow（本地运行）
cv2.imshow('Image', img)
cv2.waitKey(0) # 等待按键
cv2.destroyAllWindows()
```

```
# 方法2: 使用matplotlib (Jupyter 推荐)
import matplotlib.pyplot as plt

plt.figure(figsize=(10, 6))
plt.subplot(1, 2, 1)
plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
plt.title('Color Image')

plt.subplot(1, 2, 2)
plt.imshow(img_gray, cmap='gray')
plt.title('Gray Image')
plt.show()
```

3. 保存图像 (5 分钟):

```
# 保存图像
cv2.imwrite('output.jpg', img)
cv2.imwrite('output.png', img) # PNG格式支持透明通道
```

4.4 环节四：动手实验——给试卷加滤镜 (30 分钟)

4.4.1 4.1 任务说明 (5 分钟)

任务目标:

1. 读取一张试卷图片
2. 实现 3 种滤镜效果
3. 保存并展示结果

4.4.2 4.2 滤镜实现 (20 分钟)

滤镜 1: 灰度化

```
def to_gray(image):
    """将图像转换为灰度图"""
    return cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
```

滤镜 2: 反色效果

```
def invert_color(image):
    """反色效果: 255 - 原值"""
    return 255 - image
```

滤镜 3：亮度调整

```
def adjust_brightness(image, value=50):  
    """调整亮度：加上固定值"""  
    # 确保值在0-255范围内  
    new_image = image.astype(int) + value  
    new_image = np.clip(new_image, 0, 255).astype(uint8)  
    return new_image
```

完整示例：

```
import cv2  
import numpy as np  
import matplotlib.pyplot as plt  
  
# 读取试卷图像  
img = cv2.imread('exam_paper.jpg')  
  
# 应用滤镜  
gray = to_gray(img)  
inverted = invert_color(img)  
bright = adjust_brightness(img, 50)  
  
# 展示结果  
fig, axes = plt.subplots(2, 2, figsize=(12, 10))  
  
axes[0, 0].imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))  
axes[0, 0].set_title('原图')  
axes[0, 0].axis('off')  
  
axes[0, 1].imshow(gray, cmap='gray')  
axes[0, 1].set_title('灰度图')  
axes[0, 1].axis('off')  
  
axes[1, 0].imshow(cv2.cvtColor(inverted, cv2.COLOR_BGR2RGB))  
axes[1, 0].set_title('反色')  
axes[1, 0].axis('off')  
  
axes[1, 1].imshow(cv2.cvtColor(bright, cv2.COLOR_BGR2RGB))  
axes[1, 1].set_title('提亮')  
axes[1, 1].axis('off')
```

```
plt.tight_layout()
plt.savefig('filter_results.png', dpi=150)
plt.show()
```

4.4.3 4.3 学生练习（5 分钟）

练习任务：

1. 修改亮度调整的值，观察效果变化
2. 尝试实现”变暗”滤镜

5 课后作业

5.1 作业内容

题目：用 OpenCV 实现 3 种图像滤镜效果

要求：

1. 必须包含：灰度化、反色
2. 自选一种：亮度调整、对比度调整、模糊等
3. 提交：代码 + 处理前后对比图

5.2 评分标准

评分项	标准	分值
代码正确性	能正常运行无错误	30 分
滤镜实现	实现 3 种滤镜，效果明显	40 分
代码规范	有注释、结构清晰	15 分
结果展示	对比图清晰可见	15 分
合计		100 分

5.3 提交方式

- 截止时间：下周上课前
- 提交格式：学号 _ 姓名 _week1.zip
- 包含内容：Python 代码文件 + 结果图片