回顾：第 1 周的挑战

还记得这些痛点吗？

- `cv2.imread()` 返回 `None`
- 图像显示颜色异常
- 数组越界、维度不匹配
- API 参数太多，记不住

典型错误代码

[language=Python, basicstyle=]

```
死循环！while img is not None:

cv2.imshow('image', img) if cv2.waitKey(1) ==

'q': break
```

传统解决方式：

1. 翻文档 (`docs.opencv.org`)
2. 搜索 StackOverflow
3. 问同学/老师
4. 试错（耗费大量时间）

AI 辅助的新方式

- 直接问 AI
- 解释错误原因
- 给出修改建议
- **降低学习门槛！**

大模型写代码的本质

Token 预测

+

模式匹配

为什么 AI 擅长语法但不一定懂逻辑?

- **擅长**：基于海量代码库的模式复刻
- **不擅长**：理解你的具体业务逻辑

幻觉 (Hallucination) 专题

什么是 AI 幻觉?

AI 自信满满地生成看似合理但实际上错误或不存在的信息。

案例：AI 发明了一个不存在的 OpenCV 函数

AI 的" 创意" 代码

[language=Python, basicstyle=] `import cv2`
`img = cv2.imread('exam.jpg')`

不存在的函数！ `fixed = cv2.auto`$_{f}$$ix_{e}xposure(img)$

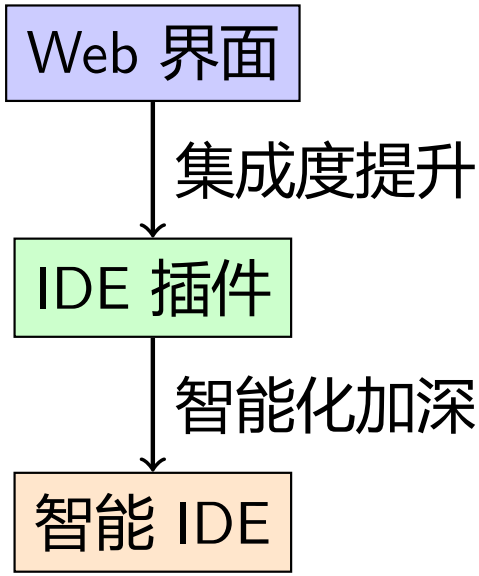
如何识别幻觉?

- 查官方文档 (docs.opencv.org)
- 在 Python 中 `dir(cv2)` 查看所有可用函数
- 多问一句：" 这个函数真的存在吗？"

核心结论：程序员的核心竞争力转变

从 ”**记忆语法**” 转向 ”**问题定义**” 与 ”**代码审查**”

AI 编程工具全景图



选择建议

- **学习理解：** ChatGPT/Claude（对话深入）
- **实时编码：** GitHub Copilot/Cursor（IDE 集成）
- **国内使用：** 通义千问/DeepSeek/CodeGeeX

主流工具详细对比

工具	特点	适用场景	推荐指数
ChatGPT	对话能力强，代码生成准确	学习、调试、解释	
Claude	代码分析深入	代码审查、架构设计	
GitHub Copilot	IDE 集成，实时补全	日常编码、快速开发	
Cursor	AI 原生 IDE	项目开发、重构	
通义千问	中文友好，国内可用	中文问题咨询	
DeepSeek	编程能力强，开源	代码生成、算法实现	

Cursor = VS Code + AI 原生集成

https://cursor.sh

核心快捷键：

- Ctrl+K：原地编辑代码
- Ctrl+L：上下文对话
- @Files：引用项目文件
- @Code：引用代码片段
- @Docs：引用文档

强大功能：

- **代码生成**：根据注释直接生成代码
- **代码解释**：选中代码，一键解释
- **代码重构**：智能重命名、提取函数
- **错误修复**：自动检测并修复 bug
- **项目理解**：全局搜索，跨文

Cursor 实战演示

场景 1：原地编辑 (Ctrl+K)

1. 选中要修改的代码
2. 按 `Ctrl+K`
3. 输入指令：”添加异常处理”
4. AI 原地修改代码

场景 2：上下文对话 (Ctrl+L)

1. 按 `Ctrl+L` 打开聊天面板
2. 输入：”解释这段代码的作用”
3. AI 结合上下文给出解释

实战技巧

- 多轮对话细化需求
- 结合 @Files 提供上下文
- 选中代码后再提问
- 不满意可以要求重写

本地大模型与隐私保护

真实场景

企业/学校要求：代码不能上传到外部服务器！

解决方案：本地部署

1. Ollama（推荐）

- 一行命令部署本地模型
- 支持 Llama、DeepSeek、Qwen 等
- 完全离线运行

2. LM Studio

- 图形界面，易于使用
- 支持多种模型格式
- 内置聊天界面

3. vLLM

- 高性能推理引擎
- 适合企业级部署
- 支持多卡并行

不同模型的对比测试

测试任务：用 Python 和 OpenCV 实现答题卡边界检测

模型	代码质量	中文理解	运行成功率
GPT-4o			95%
Claude 3.5			93%
DeepSeek-Coder			90%
通义千问 2.5			88%
GPT-3.5			75%

GPT-4o/Claude 3.5 优势：

- 代码逻辑最严谨
- 边界情况处理完善
- 参数解释最详细

DeepSeek/通义千问优势：

- 中文理解更自然
- 国内访问更稳定
- 价格更实惠

RTF 模板示例

Role (角色)：你是一个资深计算机视觉算法专家，精通 OpenCV 和图像处理。

Task (任务)：实现试卷图像的二值化，要求能自适应处理光照不均的情况。

Format (格式)：

- 返回带有详细 Docstring 的 Python 函数
- 包含输入输出示例
- 列出关键参数的调优建议

RTF 框架实战对比

不好的 Prompt

帮我写个代码处理图像。

问题分析：

- 没有定义角色
- 任务模糊
- 没有格式要求

好的 Prompt (RTF)

Role：你是一位有 10 年经验的计算机视觉工程师，精通 OpenCV。

Task：请编写一个 Python 函数，实现答题卡图像的自适应二值化。

Format：

- 提供完整的、可运行的代码
- 包含详细的函数文档
- 解释关键参数的设置依据

进阶技巧：Chain-of-Thought (思维链)

什么是思维链?

让 AI ”**一步一步思考**”（Step by Step），而不是直接给出答案。

直接给答案

Prompt： 用 OpenCV 实现透视变换矫正倾斜的试卷。

问题：

- AI 直接扔出代码
- 学生不理解原理
- 换个场景就不会了

思维链引导

Prompt：
请帮我实现试卷透视变换矫正。
请按以下步骤思考：
Step 1： 透视变换的数学原理是什么？需要哪些参数?
Step 2： 如何从图像中自动找到试卷的四个角点?
Step 3： 请写出完整的 Python 实现代码

思维链实战：准确率对比

测试任务：用 OpenCV 实现自适应阈值处理

Prompt 类型	**直接提问**	**思维链引导**
代码完整度	70%	95%
参数解释清晰度	一般	详细
边界情况处理	很少提及	全面考虑
实际运行成功率	60%	90%

思维链 Prompt 模板

请帮我解决 [问题]。请按以下步骤思考：
Step 1：分析问题的核心要点是什么？
Step 2：有哪些可能的解决方案？各自的优缺点？
Step 3：请给出推荐的实现代码

进阶技巧：Few-shot (少样本提示)

什么是 Few-shot?

给 AI 几个 **正确的例子**，让它模仿你的风格或格式处理新任务。

示例场景： OpenCV 图像转换

Few-shot Prompt

请模仿以下示例的代码风格和注释规范：

示例 1 - 灰度化：

```
# 读取图像并转换为灰度图
# 参数: image_path: 图像文件路径
# 返回: gray_image: 灰度图像数组
def load_and_gray(image_path):
    img = cv2.imread(image_path)
```

待处理任务：

继续 Prompt

示例 2 - 高斯模糊：

```
# 对图像进行高斯模糊
# 参数: image: 输入图像
# 返回: blurred: 模糊后的图像
def gaussian_smooth(image):
    blurred = cv2.GaussianBlur(image, (5,5), 0)
    return blurred
```

Few-shot 效果对比

Prompt 类型	无 Few-shot	有 Few-shot
代码风格一致性	随机，不稳定	与示例高度一致
注释完整度	较简略	详细，符合示例
参数说明	缺失或不清晰	完整的 Docstring
错误处理	经常遗漏	按示例模式添加
代码可读性	一般	优秀

Few-shot 使用技巧

1. **示例数量：** 2-3 个示例通常足够
2. **示例质量：** 确保示例是正确的、高质量的
3. **格式一致：** 示例之间保持风格一致
4. **明确指令：** 告诉 AI ”请按照示例的风格”

上下文窗口管理

问题：为什么代码太长了 AI 就会" 忘掉" 前面的内容？

- 大模型有 **上下文窗口限制**（通常 4K-128K tokens）
- 超出限制后，模型会" 遗忘" 最早的内容
- 导致前后文不一致、回答质量下降

各模型上下文限制：

模型	上下文
GPT-4	8K/32K
GPT-4o	128K
Claude 3.5	200K
DeepSeek	64K
通义千问	128K

精简 Prompt 的技巧：

1. **只提供必要代码：** 不要贴整个文件
2. **使用摘要：** " 前面我们讨论了 X，现在要解决 Y"
3. **分段处理：** 长任务拆分成多个短任务
4. **定期总结：** 让 AI 总结当前进度

任务背景：试卷图像的噪声处理

故事问题

手机拍摄的试卷照片总是有噪点、光照不均，如何自动处理这些问题？

常见图像质量问题：

- **噪声**：手机传感器产生的随机噪点
- **光照不均**：拍摄环境光线不均匀
- **模糊**：手抖或对焦不准
- **透视变形**：拍摄角度倾斜
- **阴影**：遮挡造成的暗区

对阅卷系统的影响：

- 二值化效果差 $\rightarrow$ 填涂识别错误
- 边缘检测不准 $\rightarrow$ 答题卡定位失败
- 文字识别率低 $\rightarrow$ 主观题评分困难

本周目标

用 **AI 辅助**实现：

1. 图像去噪（高斯/中值滤波）
2. 透视矫正（透视变换）
3. 边缘检测优化

实战 1：引导 AI 写出滤波代码

任务：对比高斯滤波、中值滤波对试卷扫描件的效果

第一轮 Prompt：

基础 Prompt

Role: 你是一位图像处理专家
Task: 请用 Python 和 OpenCV 实现图像去噪
要求：

- 分别实现高斯滤波和中值滤波
- 对比两种方法的效果
- 适用于试卷扫描件的去噪

第二轮 Prompt（优化）：

优化 Prompt

上面的代码很好，但还需要：

1. 添加详细的函数文档 (Docstring)
2. 解释两种滤波器的适用场景
3. 添加可视化对比图
4. 处理可能的错误（如文件不存在）

关键技巧

- 不要期望一次 Prompt 就完美
- 多轮迭代，逐步细化
- 明确告诉 AI 哪里需要改进

实战 2：引导 AI 解决透视变换

挑战

透视变换 (Warp Perspective) 的数学原理复杂，涉及齐次坐标、变换矩阵等概念。**如何通过 AI 快速实现，而不需要深入数学细节?**

第一轮：概念解释

概念 Prompt

请解释 OpenCV 中的透视变换原理：

1. 什么是透视变换?
2. getPerspectiveTransform 做了什么?
3. warpPerspective 做了什么?
4. 用简单的比喻解释

AI 的比喻解释：

第二轮：代码实现

实现 Prompt

请根据上面的解释，实现一个函数：

- 输入：任意角度拍摄的试卷图像
- 输出：矫正为正面视角的图像
- 自动或手动确定试卷的四个角点
- 使用 OpenCV 实现透视矫正
- 添加详细的注释说明

实战 3：自动识别答题卡边缘（迭代优化）

目标：通过 3 轮迭代 Prompt，让 AI 从写出简单 Canny 算子到写出鲁棒的轮廓提取逻辑。

第一轮：基础实现

基础 Prompt

请用 OpenCV 实现答题卡边缘检测。

AI 输出：

- 简单的 Canny 边缘检测
- 查找轮廓
- 可能的问题：

第二轮：优化改进

优化 Prompt

上面的代码检测效果不好，请改进：

1. 先进行图像预处理（去噪）
2. 使用自适应阈值
3. 筛选出最大的四边形
4. 处理检测失败的情况

AI 输出：

第三轮：鲁棒完善

完善 Prompt

请进一步优化，使其更鲁棒：

1. 处理极端拍摄角度
2. 处理光照不均和阴影
3. 添加透视变换矫正
4. 添加可视化调试信息
5. 封装成可复用的类

实战任务：用 AI 辅助实现人脸检测

Prompt 示例：

人脸检测 Prompt

请用 Python 和 OpenCV 实现一个人脸检测程序：
功能：从图片中检测所有人脸，并用矩形框标注
输入：图片文件路径
输出：标注了人脸框的图片
要求：

- 使用 OpenCV 的 Haar 级联分类器
- 在每个人脸周围绘制绿色矩形框
- 显示检测到的人脸数量
- 代码有详细中文注释

预期输出：完整的可运行代码，包含：

1. 详细的函数文档（Docstring）
2. 中文注释
3. 使用示例
4. 参数说明

AI 生成的人脸检测代码示例

[language=Python, basicstyle=] import cv2
def detect$_{f}$aces($image_{p}ath$) : """
参数: image$_{p}$ath : : img : """ Haar face$_{c}$ascade = $cv2.CascadeClassifier(cv2.data.haarcascades +' haarcascade_{f}rontalface_{d}efault.xml')$
读取图像 img = cv2.imread(image$_{p}$ath)$if img is None : raise ValueError(f" : image_{p}ath")$
转换为灰度图（人脸检测需要）gray = cv2.cvtColor(img, cv2.COLOR$_{B}GR2GRAY)$
检测人脸 faces = face$_{c}$ascade.$detectMultiScale(gray, scaleFactor = 1.1, minNeighbors = 5, minSize = (30, 30))$
在检测到的人脸周围绘制绿色矩形框 for (x, y, w, h) in faces: cv2.rectangle(img, (x, y), (x+w, y+h), (0, 255, 0), 2) cv2.putText(img, 'Face', (x, y-10), cv2.FONT$_{H}ERSHEY_{S}IMPLEX, 0.5, (0, 255, 0), 2)$
print(f" 检测到 len(faces) 个人脸") return img

使用示例 result = detect$_{f}aces('test.jpg')cv2.imwrite('output.jpg', result)$

第 1 轮：基础

- 简单 Canny
- 查找轮廓
- 基本功能

问题：

- 噪点干扰
- 参数固定
- 不完整边缘

第 2 轮：优化

- 高斯模糊
- 自适应阈值
- 面积筛选

改进：

- 抗噪性增强
- 自动调参
- 鲁棒性提升

第 3 轮：完善

- 完整类封装
- CLAHE 预处理
- 透视变换

特性：

- 可复用性强
- 调试可视化
- 生产级可用

第 1 步：提供 Traceback

完整复制报错信息：

- 错误类型
- 错误位置（行号）
- 完整的堆栈跟踪

示例：

```
cv2.error: ...
Invalid number of
channels
```

第 2 步：说明环境

提供运行环境信息：

- 操作系统
- Python 版本
- OpenCV 版本
- 其他相关库版本

示例：

```
Windows 11
Python 3.9.7
OpenCV 4.8.0
```

第 3 步：贴出输入数据

提供输入数据信息：

- 数据类型和格式
- 数据的维度/大小
- 期望的输出格式
- 实际的输出（如果有错误）

示例：

```
图像尺寸：
1920x1080
通道：3（RGB）
```

调试 Prompt 模板

完整的调试 Prompt 示例

我的 OpenCV 代码报错了，请帮我分析：

错误信息：

```
cv2.error: OpenCV(4.8.0) ...
Invalid number of channels in input image
```

运行环境：

```
Windows 11, Python 3.9.7, OpenCV 4.8.0
```

相关代码：

```
img = cv2.imread('test.jpg')
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
```

输入数据：

图像文件 test.jpg，尺寸 1920x1080

期望行为：正常转换为灰度图

重构：把 AI 当成导师

重构的意义

代码能运行只是第一步，**优雅的代码**才能长期维护。

重构 Prompt 模板：

重构请求

请帮我重构这段代码，提高代码质量：

[粘贴代码]

要求：

1. 提高运行效率
2. 添加类型提示
3. 改进命名规范
4. 减少重复代码
5. 添加异常处理
6. 改进代码结构

AI 可能的重构建议：

1. **性能优化：**
   - 向量化运算替代循环
   - 避免重复计算
   - 使用生成器替代列表
2. **可读性提升：**
   - 变量名更有意义
   - 函数拆分更合理
   - 添加类型提示
3. **健壮性增强：**
   - 输入参数校验
   - 异常处理完善
   - 边界情况考虑

安全与伦理：红线警告

重要提醒

使用 AI 辅助编程时，必须注意以下安全与伦理问题！

1. **数据安全**
   - **不要上传：**
     - API Key、密码
     - 个人隐私数据
     - 商业机密代码
     - 学生个人信息
   - **安全做法：**
     - 使用脱敏示例数据
     - 本地部署模型
     - 检查企业合规政策

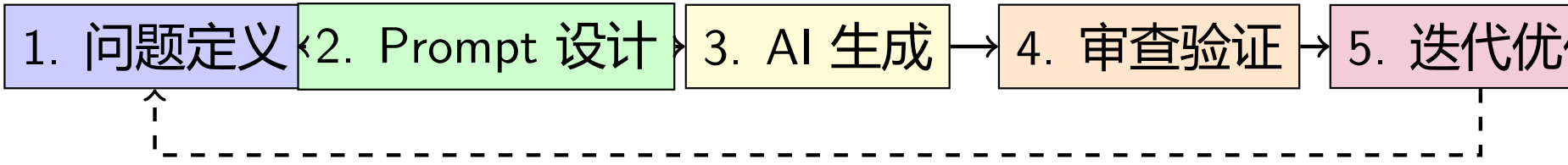
2. **版权与责任**
   - **代码归属：**
     - AI 生成代码的版权复杂
     - 检查所用工具的许可协议
     - 商业项目需谨慎
   - **责任界定：**
     - AI 生成的代码可能有错误
     - 最终责任人是你自己
     - 务必测试验证

3. **学术诚信**
   - 作业/论文中使用 AI 需声明
   - 了解学校的 AI 使用政策

建立自己的"AI 协作 SOP"

什么是 SOP?

Standard **O**perating **P**rocedure（标准操作流程）
一套可重复、可优化的工作流程



1. 问题定义

- 明确要解决的问题
- 确定输入输出
- 列出约束条件

2. Prompt 设计

- 应用 RTF 框架

3. AI 生成

- 运行 Prompt
- 获取初始输出
- 记录生成时间

4. 审查验证

- 检查代码逻辑
- 运行测试用例
- 验证输出结果

课后作业：答题卡边界检测

作业题目

用 AI 辅助实现 **答题卡边界检测**程序

项目关联

这是 **AI 阅卷助手**的第一步：

1. 图像采集与预处理
2. **答题卡定位（当前任务）**
3. 填涂检测与识别
4. 手写文字 OCR
5. 成绩统计与输出

作业要求

1. **AI 对话：** 至少 3 轮交互
   - 第 1 轮：基础实现
   - 第 2 轮：优化改进

提交内容

1. **AI 对话记录**（必交）
   - 截图或复制文本
   - 展示完整的交互过程
   - 标注关键的 Prompt 设计
2. **最终代码**（必交）
   - 完整可运行的 Python 代码
   - 包含必要的注释
3. **测试结果**（必交）
   - 测试用例图片
   - 处理结果图片
   - 标注检测到的边界
4. **反思报告**（必交）
   - 如何设计 Prompt

谢谢！

第 3 周预告：图像预处理与增强

故事问题：试卷拍照模糊怎么办？

- 图像去噪（高斯/中值滤波）
- 图像二值化（全局/Otsu/自适应）
- 透视矫正（透视变换）