3022207128-杨宇鑫-Dehazing-图像去雾-实验报告

目录

[一. 暗通道先验去雾算法实现报告 3](#_Toc4906)

[1. 算法原理 3](#_Toc19444)

[1.1 暗通道先验理论 3](#_Toc1891)

[1.2 大气散射模型 3](#_Toc23455)

[2. 代码实现 3](#_Toc25473)

[2.1 暗通道计算 3](#_Toc13704)

[2.2 大气光值估计 4](#_Toc23587)

[2.3 透射率估计 4](#_Toc17714)

[3. 关键参数说明 4](#_Toc6803)

[3.1 omega (去雾强度) 4](#_Toc13106)

[3.2 sky\_thresh (天空阈值) 5](#_Toc32708)

[3.3 sky\_trans (天空透射率) 5](#_Toc29874)

[3.4 window\_size (窗口大小) 5](#_Toc27692)

[4. 改进与优化 5](#_Toc4192)

[4.1 天空区域处理 5](#_Toc7053)

[4.2 引导滤波优化 5](#_Toc2969)

[5. 实验结果 6](#_Toc23532)

[5.1 效果分析 6](#_Toc18947)

[5.2 参数影响 6](#_Toc24893)

[二. 图像去雾处理系统使用教程 6](#_Toc15103)

[1. 安装依赖包 6](#_Toc17306)

[2. 启动应用 6](#_Toc19347)

[2.1 运行Flask应用 6](#_Toc25594)

[2.2 访问Web界面 6](#_Toc2238)

[1. 打开浏览器 6](#_Toc5051)

[3. 使用说明 6](#_Toc32084)

[3.1 基本操作流程 6](#_Toc32138)

[3.2 参数说明 7](#_Toc28629)

[• 去雾强度：控制整体去雾程度，值越大去雾效果越强 7](#_Toc611)

[4. 常见问题解决 7](#_Toc2009)

[4.1 无法启动应用 7](#_Toc5061)

[4.2 图片上传失败 7](#_Toc24586)

[4.3 处理结果不理想 7](#_Toc7244)

[5. 注意事项 7](#_Toc28870)

[三. 去雾效果 7](#_Toc21066)

# 暗通道先验去雾算法实现报告

## 1. 算法原理

### 1.1 暗通道先验理论

暗通道先验(Dark Channel Prior)是一种基于统计观察的图像去雾方法。其核心观察是:在大多数非天空区域的局部区域中,至少有一个颜色通道的值很低。

数学表达式:

Jdark(x) = min(min(Jc(y))) # c ∈ {r,g,b}, y ∈ Ω(x)

其中:

* J 是无雾图像
* Jc 是图像的RGB三个通道
* Ω(x) 是以像素x为中心的局部区域
* Jdark 是暗通道图像

### 1.2 大气散射模型

图像形成模型:

I(x) = J(x)t(x) + A(1-t(x))

其中:

* I 是有雾图像
* J 是场景辐射(要恢复的无雾图像)
* t 是透射率图
* A 是大气光值

## 2. 代码实现

### 2.1 暗通道计算

def get\_dark\_channel(self, img):  
 """优化暗通道计算"""  
 b, g, r = cv2.split(img)  
 min\_rgb = cv2.min(cv2.min(r, g), b)  
 kernel = cv2.getStructuringElement(cv2.MORPH\_RECT,   
 (self.window\_size, self.window\_size))  
 dark\_channel = cv2.erode(min\_rgb, kernel)  
 return dark\_channel

### 2.2 大气光值估计

def estimate\_atmospheric\_light(self, img, dark\_channel):  
 """优化大气光值估算"""  
 h, w = dark\_channel.shape  
 flat\_img = img.reshape(h \* w, 3)  
 flat\_dark = dark\_channel.ravel()  
   
 # 取暗通道前0.1%最亮的点  
 max\_pixels = int(h \* w \* 0.001)  
 indices = np.argpartition(flat\_dark, -max\_pixels)[-max\_pixels:]  
   
 # 在最亮的点中选择RGB均值最大的点  
 candidate\_pixels = flat\_img.take(indices, axis=0)  
 bright\_values = np.mean(candidate\_pixels, axis=1)  
 atmospheric\_light = candidate\_pixels[np.argmax(bright\_values)]  
   
 return atmospheric\_light

### 2.3 透射率估计

def estimate\_transmission(self, img, atmospheric\_light):  
 """更新天空区域处理"""  
 normalized = img / atmospheric\_light  
 dark\_channel = self.get\_dark\_channel(normalized)  
   
 # 计算初始透射率  
 transmission = 1 - self.omega \* dark\_channel  
   
 # 区分天空和非天空区域  
 intensity = np.mean(img, axis=2)  
 blue\_channel = img[:,:,0]  
 sky\_mask = (intensity > self.sky\_thresh) & (blue\_channel > np.mean(img, axis=2))  
   
 # 差异化处理  
 transmission[sky\_mask] = np.maximum(transmission[sky\_mask], self.sky\_trans)  
 transmission[~sky\_mask] = transmission[~sky\_mask] \* 0.9  
   
 return transmission

## 3. 关键参数说明

### 3.1 omega (去雾强度)

* 范围: 0.5-1.0
* 默认值: 0.75
* 作用: 控制去雾程度,值越大去雾效果越强

### 3.2 sky\_thresh (天空阈值)

* 范围: 0.3-1.0
* 默认值: 0.7
* 作用: 控制天空区域的识别范围

### 3.3 sky\_trans (天空透射率)

* 范围: 0.5-1.5
* 默认值: 0.85
* 作用: 控制天空区域的去雾程度和亮度

### 3.4 window\_size (窗口大小)

* 范围: 5-25
* 默认值: 5
* 作用: 控制暗通道计算的局部区域大小

## 4. 改进与优化

### 4.1 天空区域处理

为了避免过度去雾,对天空区域进行特殊处理:

# 天空区域使用较高的透射率  
transmission[sky\_mask] = np.clip(transmission[sky\_mask], self.sky\_trans, 0.98)

### 4.2 引导滤波优化

使用引导滤波改善透射率图的边缘保持性:

def guided\_filter(self, img, p, r, eps):  
 mean\_i = cv2.boxFilter(img, cv2.CV\_64F, (r,r))  
 mean\_p = cv2.boxFilter(p, cv2.CV\_64F, (r,r))  
 mean\_ip = cv2.boxFilter(img\*p, cv2.CV\_64F, (r,r))  
 cov\_ip = mean\_ip - mean\_i \* mean\_p  
   
 mean\_ii = cv2.boxFilter(img\*img, cv2.CV\_64F, (r,r))  
 var\_i = mean\_ii - mean\_i \* mean\_i  
   
 a = cov\_ip / (var\_i + eps)  
 b = mean\_p - a \* mean\_i  
   
 mean\_a = cv2.boxFilter(a, cv2.CV\_64F, (r,r))  
 mean\_b = cv2.boxFilter(b, cv2.CV\_64F, (r,r))  
   
 return mean\_a \* img + mean\_b

## 5. 实验结果

### 5.1 效果分析

* 天空区域保持自然
* 暗部细节得到恢复
* 整体对比度适中
* 色彩保持自然

### 5.2 参数影响

* 窗口大小影响细节保持
* 去雾强度影响整体效果
* 天空参数影响天空区域的自然度

# 图像去雾处理系统使用教程

## 1. 安装依赖包

pip install -r requirements.txt

## 2. 启动应用

### 2.1 运行Flask应用

python app.py

成功启动后会看到类似输出：

\* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)  
\* Restarting with stat  
\* Debugger is active!  
\* Debugger PIN: xxx-xxx-xxx

### 2.2 访问Web界面

1. 打开浏览器
2. 访问 http://127.0.0.1:5000 或 http://localhost:5000

## 3. 使用说明

### 3.1 基本操作流程

1. 点击”选择图片”按钮上传有雾图片
2. 调整参数（可选）：
   * 去雾强度 (0.5-1.0)
   * 天空阈值 (0.3-1.0)
   * 天空透射率 (0.5-1.5)
   * 窗口大小 (5-25)
3. 点击”处理图片”按钮
4. 等待处理完成，查看结果
5. 点击图片可放大查看
6. 可以使用”重置参数”恢复默认设置

### 3.2 参数说明

* **去雾强度**：控制整体去雾程度，值越大去雾效果越强
* **天空阈值**：控制天空区域的识别范围，值越大识别的天空区域越少
* **天空透射率**：控制天空区域的去雾程度和亮度，值越大天空越亮
* **窗口大小**：控制局部区域大小，较大的值会使去雾更均匀，但可能丢失细节

## 4. 常见问题解决

### 4.1 无法启动应用

* 检查Python版本是否符合要求
* 确认所有依赖包安装成功
* 检查端口5000是否被占用

### 4.2 图片上传失败

* 确认图片格式是否支持
* 检查图片大小是否超过限制(16MB)
* 确认浏览器版本是否过旧

### 4.3 处理结果不理想

* 尝试调整参数
* 使用”重置参数”恢复默认值
* 确保原图清晰度足够

## 5. 注意事项

1. 建议使用虚拟环境安装依赖，避免包冲突
2. 处理大图片时可能需要等待较长时间
3. 建议使用最新版本的现代浏览器
4. 保持网络连接稳定

# 去雾效果













