

姓名：王旭 学号：3020001267

---

## 校园图像



## 噪声

### 高斯噪声

#### 实现细节

主要使用opencv提供的RNG类的成员函数fill完成,

```
Mat gaussian_noise(Mat& img) {  
    Mat imgGaussianNoise(img.size(), img.type());  
    Mat gaussianNoise(img.size(), img.type());  
  
    RNG rng(seed);  
    rng.fill(gaussianNoise, RNG::NORMAL, 0, 30);  
    cv::add(img, gaussianNoise, imgGaussianNoise);  
  
    imshow("高斯噪声图像", gaussianNoise);  
    imshow("加上高斯噪声后的图像", imgGaussianNoise);  
    return imgGaussianNoise;  
}
```

fill为gaussianNoise填充了均值为0，方差为30的正态分布噪声

效果



可以看到图像被加上了高斯噪声

## 椒盐噪声

### 实现细节

指定一个椒盐的数量nums, 遍历这个nums, 每一次都随机生成一个坐标 $(x, y)$  其中  $x, y$  不超过图像的范围, 并且如果当前轮数是奇数的话添加白点, 否则添加黑点。

```
Mat saltPepper_noise(Mat& img, int nums) {
    Mat imgSaltPepperNoise = img.clone();

    RNG rng(seed);
    for (int i = 0; i < nums; i++) {
        int x = rng.uniform(0, img.cols);
        int y = rng.uniform(0, img.rows);
        if (i % 2 == 1) {
            imgSaltPepperNoise.at<Vec3b>(y, x) = Vec3b(255, 255, 255);
        }
        else {
            imgSaltPepperNoise.at<Vec3b>(y, x) = Vec3b(0, 0, 0);
        }
    }
    imshow("加上椒盐噪声后的图像", imgSaltPepperNoise);
    return imgSaltPepperNoise;
}
```

### 效果





可以看到图像被加上了椒盐噪声

## 高斯滤波

### 实现细节

#### opencv函数调库实现

使用opencv库提供的GaussianBlur，可以设置高斯核大小，和方差

```
Mat gaussian_filter(Mat& img,int kernelSize,int sigma) {  
    Mat ImgGaussianFilter;  
    GaussianBlur(img, ImgGaussianFilter, Size(kernelSize, kernelSize), sigma);  
    imshow("高斯滤波处理后的图像", ImgGaussianFilter);  
    return ImgGaussianFilter;  
}
```

#### 手工实现

算法流程：

1. 首先按照kernelSize创建一个二维数组，表示高斯核，按照二维正态分布公式实现

$$G(u, v) = \frac{1}{2\pi\sigma^2} e^{-(u^2+v^2)/(2\sigma^2)}$$

式中 $u, v$ 是离核中心的归一化距离

2. 随后用高斯核对图像逐kernelSize\*kernelSize大小的正方形和逐通道的做一对一相乘即可

```
Mat gaussian_filter_manual(Mat& img, int kernelSize, double sigma) {
    Mat ImgGaussianFilter(img.size(), img.type());
    vector<vector<double>> kernel(kernelSize, vector<double>(kernelSize));
    double sum = 0;
    double pi = acos(-1);
    double step = 1.0 / kernelSize;
    for (int i = 0; i < kernelSize; i++) {
        for (int j = 0; j < kernelSize; j++) {
            double u = -0.5 + (0.5 + j) * (step);
            double v = -0.5 + (0.5 + i) * (step);
            kernel[i][j] = exp(-(u * u + v * v) / (2 * sigma * sigma)) / (2 * pi *
sigma * sigma);
            sum += kernel[i][j];
        }
    }
    for (int i = 0; i < kernelSize; i++) {
        for (int j = 0; j < kernelSize; j++) {
            kernel[i][j] /= sum;
        }
    }

    int border = kernelSize / 2;

    for (int i = border; i < img.rows - border; i++) {
        for (int j = border; j < img.cols - border; j++) {
            for (int c = 0; c < 3; c++) {
                double sum = 0, weight_sum = 0;
                for (int k = -border; k <= border; k++) {
                    for (int l = -border; l <= border; l++) {
                        double weight = kernel[k + border][l + border];
                        sum += weight * img.at<Vec3b>(i + k, j + l)[c];
                        weight_sum += weight;
                    }
                }
                ImgGaussianFilter.at<Vec3b>(i, j)[c] = saturate_cast<uchar>(sum
/ weight_sum);
            }
        }
    }
    imshow("高斯滤波处理后的图像", ImgGaussianFilter);
    return ImgGaussianFilter;
}
```

## 高斯滤波处理原图



可以看到图像变模糊了很多

## 高斯滤波处理高斯噪声



虽然图像变模糊了，但是噪声也不明显了

**高斯滤波处理椒盐噪声**





可以看到高斯滤波对椒盐噪声处理的效果比较差

**手工实现高斯滤波处理高斯噪声**





可以看到效果与opencv调库实现的基本一致，只是在手工实现版本中图像边框处有灰边。

## 中值滤波

### 实现细节

#### opencv调库实现

使用opencv 提供的medianBlur函数即可

```
Mat median_filter(Mat& img, int kernelSize) {  
    Mat ImgMedianFilter;  
    medianBlur(img, ImgMedianFilter, kernelSize);  
    imshow("中值滤波处理后的图像", ImgMedianFilter);  
    return ImgMedianFilter;  
}
```

#### 手工实现

算法流程：

1. 对图像逐kernelSize\*kernelSize的正方形，逐通道地把像素点值存到一个vector中，然后对vector排序取中位数即可。

```

Mat median_filter_manual(Mat& img, int kernelSize)
{
    // 对每个像素进行中值滤波
    int border = kernelSize / 2;
    Mat ImgMedianFilter(img.size(), img.type());
    for (int i = border; i < img.rows - border; i++) {
        for (int j = border; j < img.cols - border; j++) {
            for (int c = 0; c < 3; c++) {
                vector<uchar> values;
                for (int k = -border; k <= border; k++) {
                    for (int l = -border; l <= border; l++) {
                        values.push_back(img.at<Vec3b>(i + k, j + l)[c]);
                    }
                }
                sort(values.begin(), values.end());
                ImgMedianFilter.at<Vec3b>(i, j)[c] = values[kernelSize *
kernelSize / 2];
            }
        }
    }

    imshow("中值滤波处理后的图像", ImgMedianFilter);
    return ImgMedianFilter;
}

```

## 中值滤波处理原图



图像变得有种印象派油画的感觉

**中值滤波处理高斯噪声**





中值滤波处理椒盐噪声



椒盐噪声被基本消除

**手工实现中值滤波处理椒盐噪声**



效果与调库实现的基本一致，只是图像边框处有灰边。