山东大学 计算机科学与技术 学院

计算机视觉 课程实验报告

学号：201800130086	姓名： 徐鹏博	
实验题目：图像匹配 1		

实验过程中遇到和解决的问题：

Harris 角点检测

实现 Harris 角点检测算法，并与 OpenCV 的 cornerHarris 函数的结果进行比较

1. 计算图像$I(x,y)$在X和Y两个方向的梯度I_x、I_y。

$$I_x=\frac{\partial I}{\partial x}=I\otimes(-1\ 0\ 1),\ I_y=\frac{\partial I}{\partial x}=I\otimes(-1\ 0\ 1)^T$$

2. 计算图像两个方向梯度的乘积。

$$I_x^2=I_x\cdot I_y,\ I_y^2=I_y\cdot I_y,\ I_{xy}=I_x\cdot I_y$$

3. 使用高斯函数对I_x^2、I_y^2和I_{xy}进行高斯加权（取$\sigma=1$），生成矩阵M的元素A、B和C。

$$A=g(I_x^2)=I_x^2\otimes w,\ C=g(I_y^2)=I_y^2\otimes w,\ B=g(I_{x,y})=I_{xy}\otimes$$

4. 计算每个像素的Harris响应值R，并对小于某一阈值t的R置为零。

$$R=\{R:detM-\alpha(traceM)^2<t\}$$

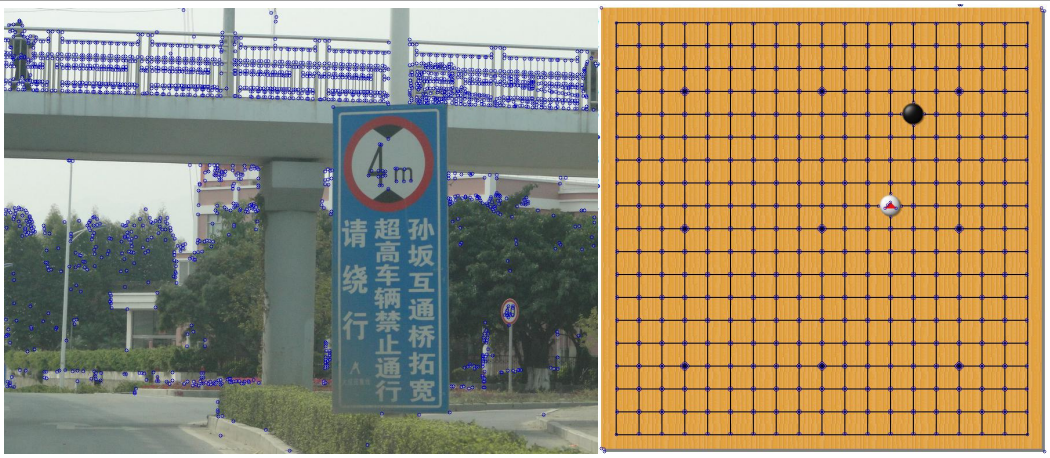
5. 在3×3或5×5的邻域内进行非最大值抑制，局部最大值点即为图像中的角点。

```
Mat Harris_Corner(const Mat& input,double alpha) {
    Mat gray;
    cvtColor(input, gray, COLOR_RGB2GRAY);
    gray.convertTo(gray, CV_64F);
    Ix,Iy 梯度计算
    Mat Ix, Iy;
    Sobel(gray, Ix, -1, 1, 0);
    Sobel(gray, Iy, -1, 0, 1);
    梯度乘积计算
    Mat Ix2, Ixy, Iy2;
    Ix2 = Ix.mul(Ix);
    Ixy = Ix.mul(Iy);
    Iy2 = Iy.mul(Iy);
    高斯加权
    GaussianBlur(Ix2, Ix2, cv::Size(7, 7), 1, 1);
    GaussianBlur(Ixy, Ixy, cv::Size(7, 7), 1, 1);
    GaussianBlur(Iy2, Iy2, cv::Size(7, 7), 1, 1);
```

```
    计算每个像素的 Harris 响应值 R
    Mat Corner(gray.size(), gray.type());
    for (int i = 0; i < Corner.rows; ++i) {
        for (int j = 0; j < Corner.cols; ++j) {
            double det_M = Ix2.at<double>(i, j) * Iy2.at<double>(i, j)- Ixy.at<double>(i,j) * Ixy.at<double>(i,j);;
            double trace_M = Ix2.at<double>(i, j) + Iy2.at<double>(i, j);
            Corner.at<double>(i, j) = det_M - alpha * trace_M * trace_M;
        }
    }
    非最大值抑制
    double MaxValue;
    minMaxLoc(Corner, NULL, &MaxValue, NULL, NULL);

    Mat dilated, localMax;
    dilate(Corner, dilated, Mat());
    compare(Corner, dilated, localMax, CMP_EQ);
    Mat CornerMAP;
    double Val = 0.01;
    double threshold = Val * MaxValue;
    cout<<"max:"<<MaxValue<<endl;
    CornerMAP = Corner > threshold;
    bitwise_and(CornerMAP, localMax, CornerMAP);
    return CornerMAP;
}
    画出角点
Mat DrawCircle(Mat& input,const Mat& Bin) {
    Mat output = input.clone();
    Mat_<uchar>::const_iterator it = Bin.begin<uchar>();
    Mat_<uchar>::const_iterator it_end = Bin.end<uchar>();
    for (int i = 0; it != it_end; it++, i++) {
        if (*it)
            circle(output, Point(i % output.cols, i / output.cols), 3, Scalar(200, 0, 0), 1);
    }
    return output;
}
```

我的代码效果：



opencv 效果：

结果分析与体会：

$$R = \lambda_2\lambda_2 = \alpha(\lambda_2+\lambda_2)^2 = \lambda^2(k-\alpha(1+k)^2)\ 0 \le \alpha \le \frac{k}{(1+k)^2} \le 0.25$$

增大α的值，将减小角点响应值 R，降低角点检测的灵性，减少被检测角点的数量；减小α值，将增大角点响应值 R，增加角点检测的灵敏性，增加被检测角点的数量。

Harris 角点检测算子对亮度和对比度的变化不敏感。