数字图像处理

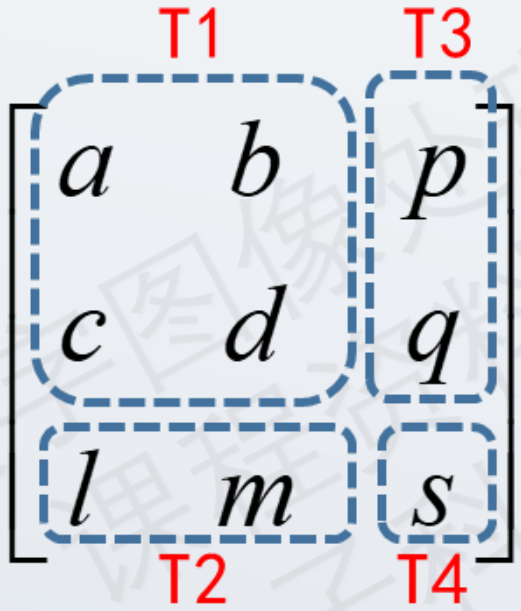
第七周课堂练习

李竹

杭州电子科技大学

电子信息学院

仿射矩阵



T1：比例、旋转、对称、错切

T2：平移

T3：投影

T4：整体缩放

OpenCV中为3x2矩阵

$$\begin{bmatrix} a & c & l \\ b & d & m \end{bmatrix}$$

思考

通过OpenCV进行图像的旋转后，超出原尺寸的部分，会被自动裁剪，如何实现不自动裁剪的图像旋转

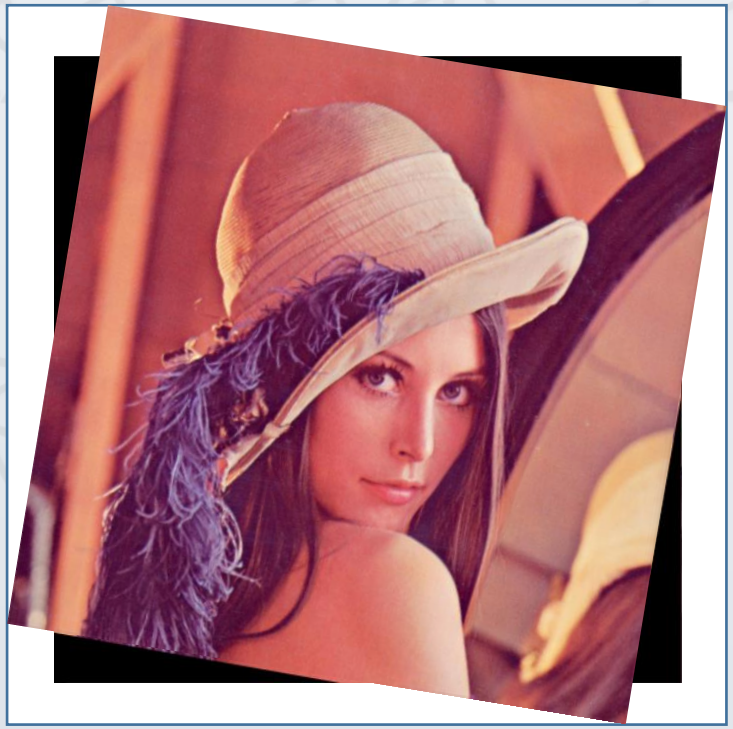
原图

原旋转

改进后的旋转

思考

```
cv::Point2f center(src.cols / 2.0, src.rows / 2.0);
cv::Mat rot = cv::getRotationMatrix2D(center, angle, scale);
// 获取外界四边形
cv::Rect bbox = cv::RotatedRect(center, src.size(), angle).boundingRect();
// 调整仿射矩阵参数
rot.at<double>(0, 2) += bbox.width / 2.0 - center.x;
rot.at<double>(1, 2) += bbox.height / 2.0 - center.y;
//仿射变换
cv::warpAffine(src, dst, rot, bbox.size(), interMethod);
```



OpenCV中为3x2矩阵

$$\begin{bmatrix} a & c & l \\ b & d & m \end{bmatrix}$$

练习1

通过OpenCV进行图像的旋转后，超出原尺寸的部分，会被自动裁剪，如何实现不自动裁剪的图像旋转

原图

原旋转

改进后的旋转

练习2

```
void HoughLines( InputArray image, OutputArray lines,
                 double rho, double theta, int threshold,
                 double srn = 0, double stn = 0,
                 double min_theta = 0, double max_theta = CV_PI );
```

绘制直线参考代码

0.输入：8bit二值图

1.输出：直线的结果

2.极坐标rho的步长

3.极坐标角度的步长

4.投票阈值

```
std::vector<cv::Vec2f>::iterator it = lines.begin();
for (; it != lines.end(); ++it) {
    float rho = (*it)[0], theta = (*it)[1];
    cv::Point pt1, pt2;
    double a = cos(theta);
    double b = sin(theta);
    double x0 = a*rho;
    double y0 = b*rho;
    pt1.x = cv::saturate_cast<int>(x0 + 1000 * (-b));
    pt1.y = cv::saturate_cast<int>(y0 + 1000 * (a));
    pt2.x = cv::saturate_cast<int>(x0 - 1000 * (-b));
    pt2.y = cv::saturate_cast<int>(y0 - 1000 * (a));
    cv::line(src, pt1, pt2, cv::Scalar(0, 0, 255), 1, CV_AA);
}
```

练习2

输出方式推荐使用Mat

```
std::vector<cv::Vec2f> lines;
cv::HoughLines(canny, lines, 1, CV_PI / 180, [redacted]);

cv::Mat lineMat;
cv::HoughLines(canny, lineMat, 1, CV_PI / 180, [redacted]);
```

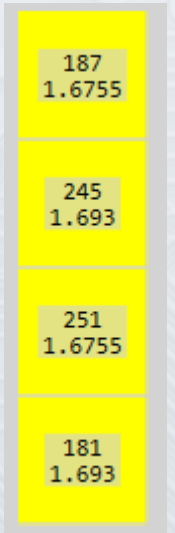
vector输出

第0条直线的rho

第0条直线的theta

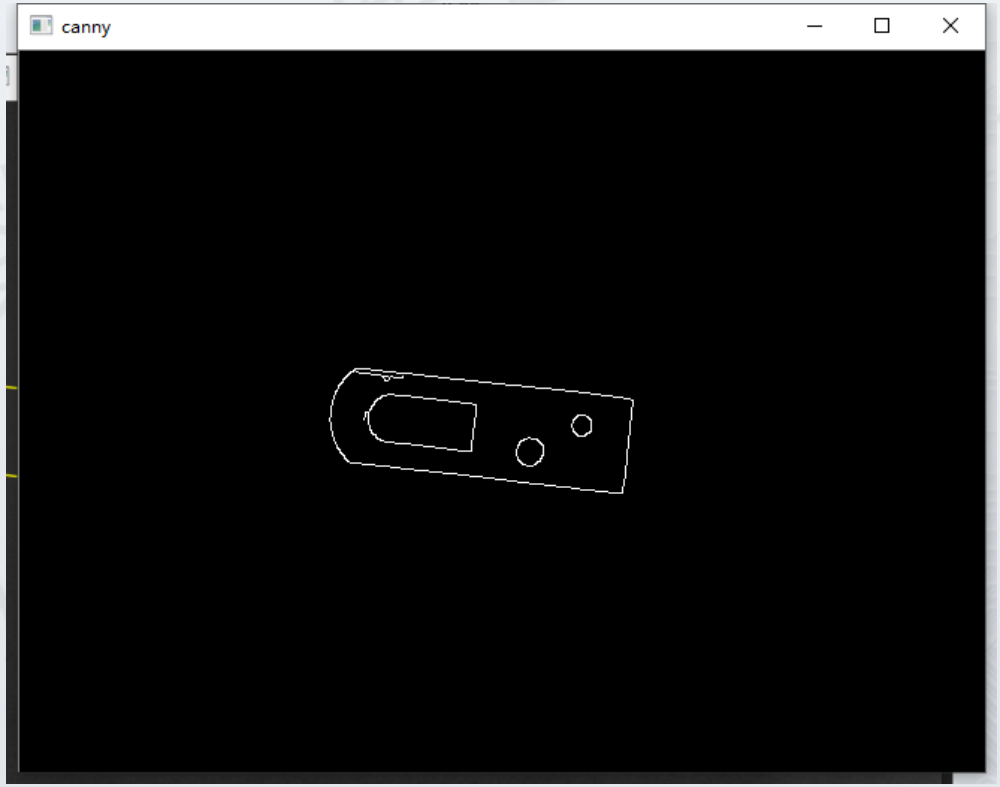
```
float r = lines[0][0];
float t = lines[0][1];
```

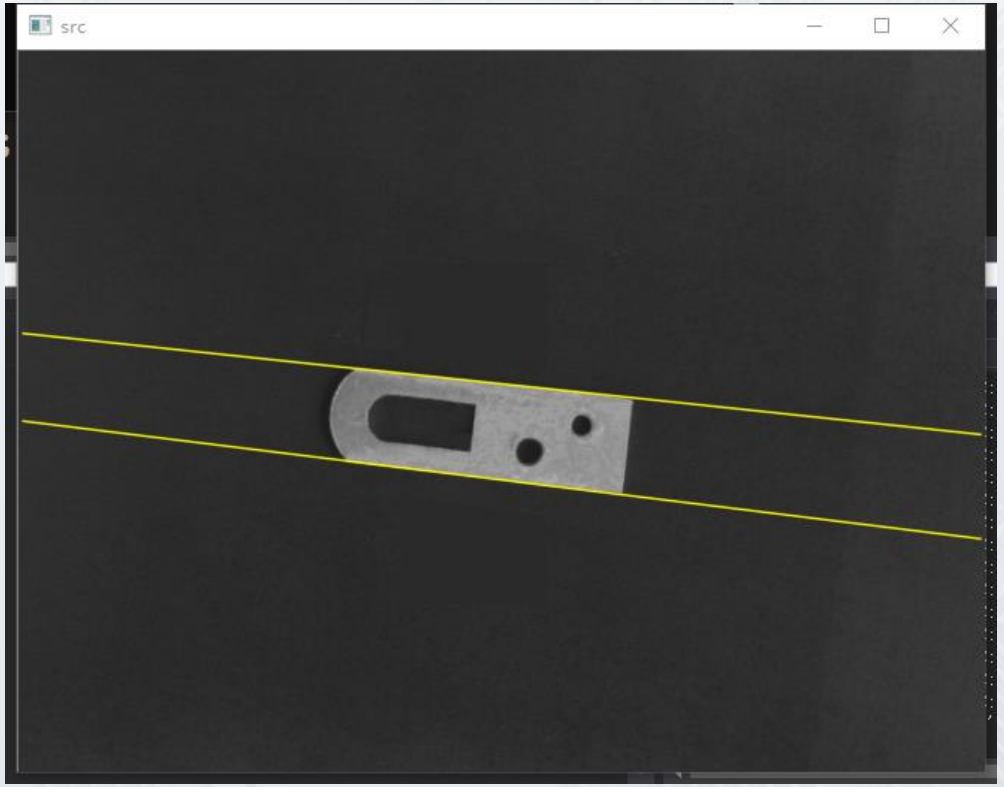
Mat输出



练习2

参考结果





练习3

```
void HoughLinesP( InputArray image, OutputArray lines,
                  double rho, double theta, int threshold,
                  double minLineLength = 0, double maxLineGap = 0 );
```

0.输入：8bit二值图

1.输出：直线的结果

2.极坐标rho的步长

3.极坐标角度的步长

4.投票阈值

5.最小直线长度

6.最大间隔

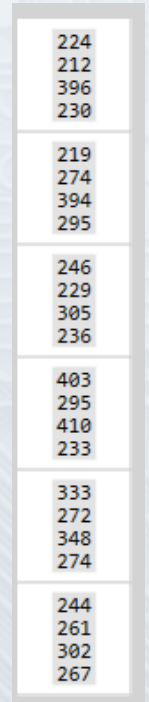
vector输出

声明

```
std::vector<cv::Vec4i> lines;
```

```
int x1 = lines[0][0];
int y1 = lines[0][1];
int x2 = lines[0][2];
int y2 = lines[0][3];
```

Mat输出



练习3

参考结果