영상처리 실제 10주차 실습_영상분할

2023254015 장욱진

```
#include <opencv2/opencv.hpp>

using namespace std;
using namespace cv;

void setLabel(Mat& img, const vector<Point>& pts, const String& label)
{
	Rect rc = boundingRect(pts);
	rectangle(img, rc, Scalar(0, 0, 255), 1);
	putText(img, label, rc.tl(), FONT_HERSHEY_PLAIN, 1, Scalar(0, 0, 255));
}

Mat preprocessing(Mat img)
{
	Mat gray, th_img;
	cvtColor(img, gray, CV_BGR2GRAY);
	GaussianBlur(gray, gray, Size(7, 7), 2, 2);

	threshold(gray, th_img, 130, 255, THRESH_BINARY | THRESH_OTSU);
	morphologyEx(th_img, th_img, MORPH_OPEN, Mat(), Point(-1, -1), 1);

	return th_img;
}

vector<RotatedRect> find_coins(Mat img)
{
	vector<vector<Point> > contours;

	findContours(img.clone(), contours, RETR_EXTERNAL, CHAIN_APPROX_SIMPLE);

	vector<RotatedRect> circles;
	for (int i = 0; i < (int)contours.size(); i++)
	{
		RotatedRect mr = minAreaRect(contours[i]);
		mr.angle = (mr.size.width + mr.size.height) / 4.0;

		if (mr.angle > 18) circles.push_back(mr);
	}

	return circles;
}

void page13()
{
	Mat src = imread("./lenna.jpg", IMREAD_GRAYSCALE);
```

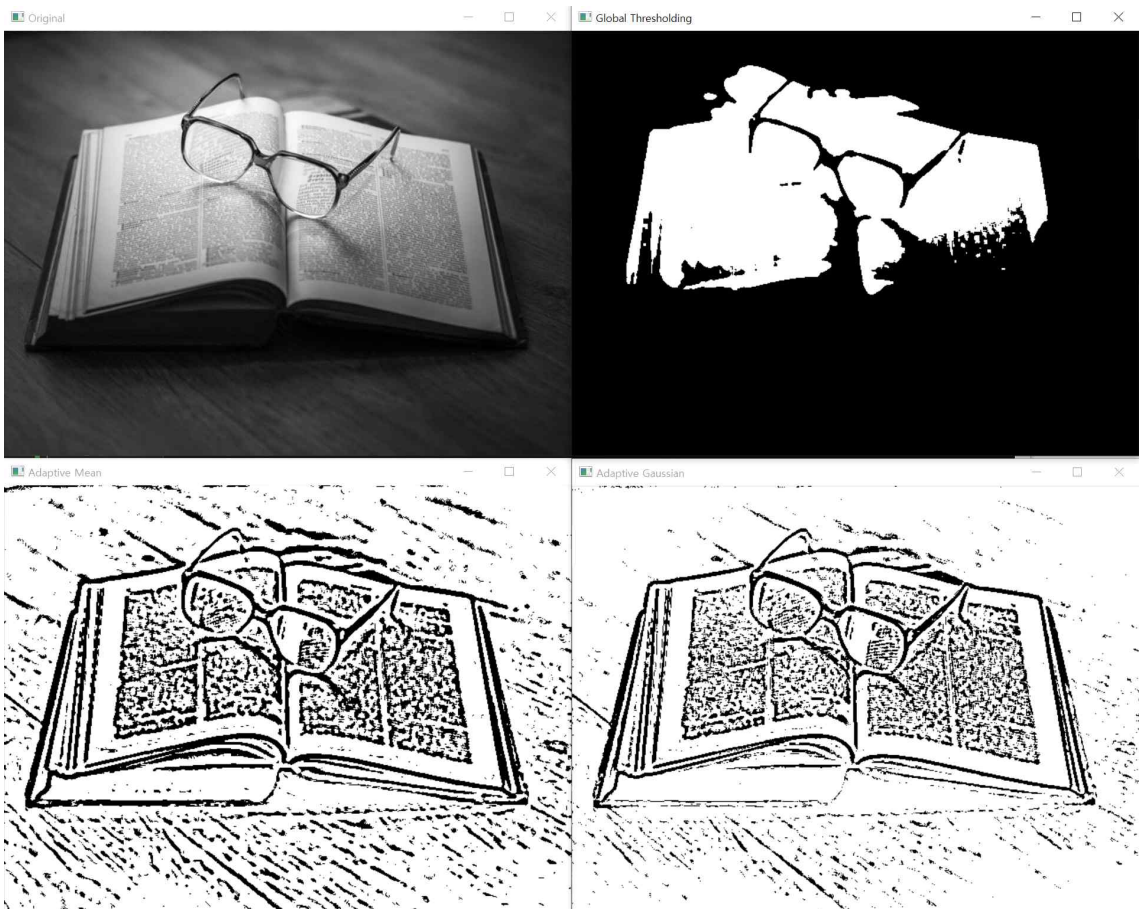
```
	Mat blur, th1, th2, th3, th4;
	threshold(src, th1, 127, 255, THRESH_BINARY);
	threshold(src, th2, 0, 255, THRESH_BINARY | THRESH_OTSU);
	Size size = Size(5, 5);
	GaussianBlur(src, blur, size, 0);
	threshold(blur, th3, 0, 255, THRESH_BINARY | THRESH_OTSU);
	imshow("Original", src);
	imshow("Global", th1);
	imshow("Ostu", th2);
	imshow("Ostu after Blurring", th3);
	waitKey();
}

void page19()
{
	Mat src = imread("./book1.jpg", IMREAD_GRAYSCALE);
	Mat img, th1, th2, th3, th4;
	medianBlur(src, img, 5);
	threshold(img, th1, 127, 255, THRESH_BINARY);
	adaptiveThreshold(img, th2, 255, ADAPTIVE_THRESH_MEAN_C, THRESH_BINARY,
11, 2);
	adaptiveThreshold(img, th3, 255, ADAPTIVE_THRESH_GAUSSIAN_C,
THRESH_BINARY, 11, 2);
	imshow("Original", src);
	imshow("Global Thresholding", th1);
	imshow("Adaptive Mean", th2);
	imshow("Adaptive Gaussian", th3);
	waitKey();
}

void page28()
{
	Mat img, img_edge, labels, centroids, img_color, stats;
	img = cv::imread("./coins.png", IMREAD_GRAYSCALE);
	threshold(img, img_edge, 128, 255, THRESH_BINARY_INV);
	imshow("Image after threshold", img_edge);
	int n = connectedComponentsWithStats(img_edge, labels, stats, centroids);
	vector<Vec3b> colors(n + 1);
	colors[0] = Vec3b(0, 0, 0);
	for (int i = 1; i <= n; i++) {
		colors[i] = Vec3b(rand() % 256, rand() % 256, rand() % 256);
	}
	img_color = cv::Mat::zeros(img.size(), CV_8UC3);
	for (int y = 0; y < img_color.rows; y++)
		for (int x = 0; x < img_color.cols; x++)
		{
			int label = labels.at<int>(y, x);
			img_color.at<cv::Vec3b>(y, x) = colors[label];
		}
```

```
	cv::imshow("Labeled map", img_color);
	cv::waitKey();
}


void page34()
{
	int coln_no = 0;
	String fname = format("./%d.png", coln_no);
	Mat image = imread(fname, 1);
	CV_Assert(image.data);

	Mat th_img = preprocessing(image);

	vector<RotatedRect> circles = find_coins(th_img);

	for (int i = 0; i < circles.size(); i++)
	{
		float radius = circles[i].angle;
		circle(image, circles[i].center, radius, Scalar(0, 255, 0), 2);
	}

	imshow("전처리영상", th_img);
	imshow("동전영상", image);
	waitKey();
}

int page39()
{
	Mat img = imread("polygon.bmp", IMREAD_COLOR);

	if (img.empty())
	{
		cerr << "Image load failed!" << endl;
		return -1;
	}

	Mat gray;
	cvtColor(img, gray, COLOR_BGR2GRAY);

	Mat bin;
	threshold(gray, bin, 200, 255, THRESH_BINARY_INV | THRESH_OTSU);

	vector<vector<Point> > contours;
	findContours(bin, contours, RETR_EXTERNAL, CHAIN_APPROX_NONE);

	for (vector<Point>& pts : contours)
	{
		if (contourArea(pts) < 400) continue;
```

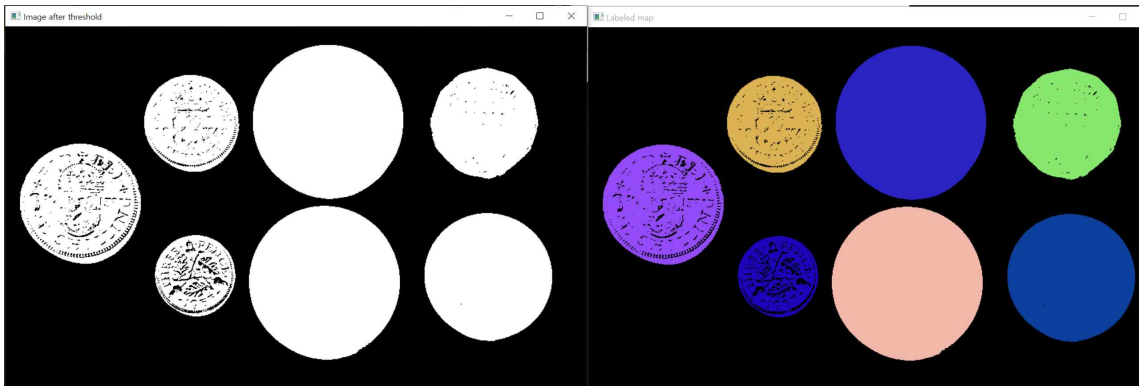
```
                vector<Point> approx;
                approxPolyDP(pts, approx, arcLength(pts, true) * 0.02, true);

                int vtc = (int)approx.size();

                if (vtc == 3)
                        setLabel(img, pts, "TRI");
                else if (vtc == 4)
                        setLabel(img, pts, "RECT");
                else if (vtc > 4)
                {
                        double len = arcLength(pts, true);
                        double area = contourArea(pts);
                        double ratio = 4. * CV_PI * area / (len * len);

                        if (ratio > 0.8)
                                setLabel(img, pts, "CIR");
                }
        }

        imshow("img", img);
        waitKey();
}

int main()
{
        page13();
        page19();
        page28();
        page34();
        page39();

        return 0;
}
```

결과화면

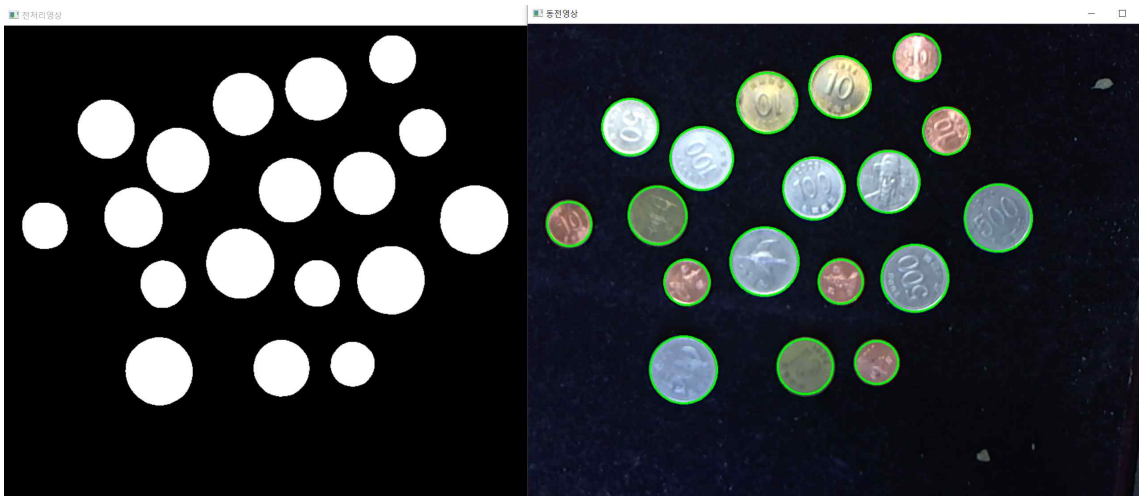
<page13 결과화면>



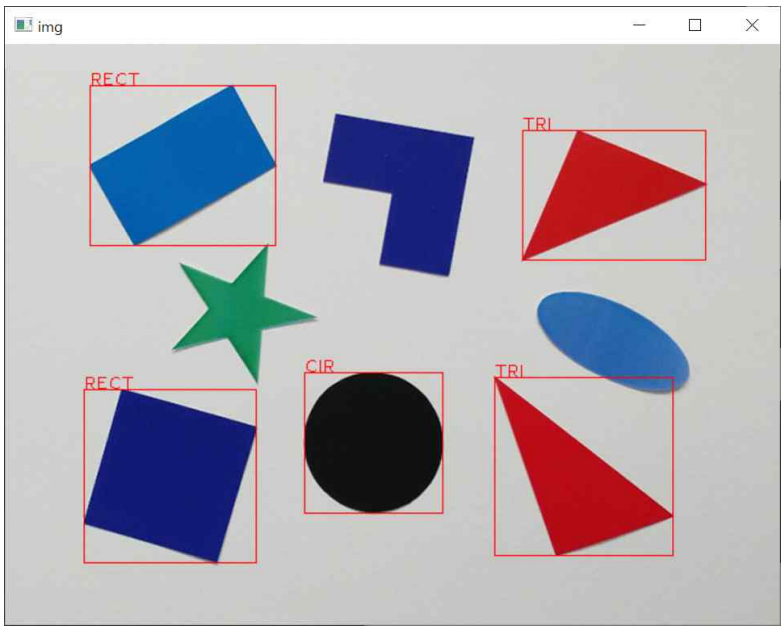
<page19 결과화면>



<page28 결과화면>



<page34 결과화면>



<page39 결과화면>