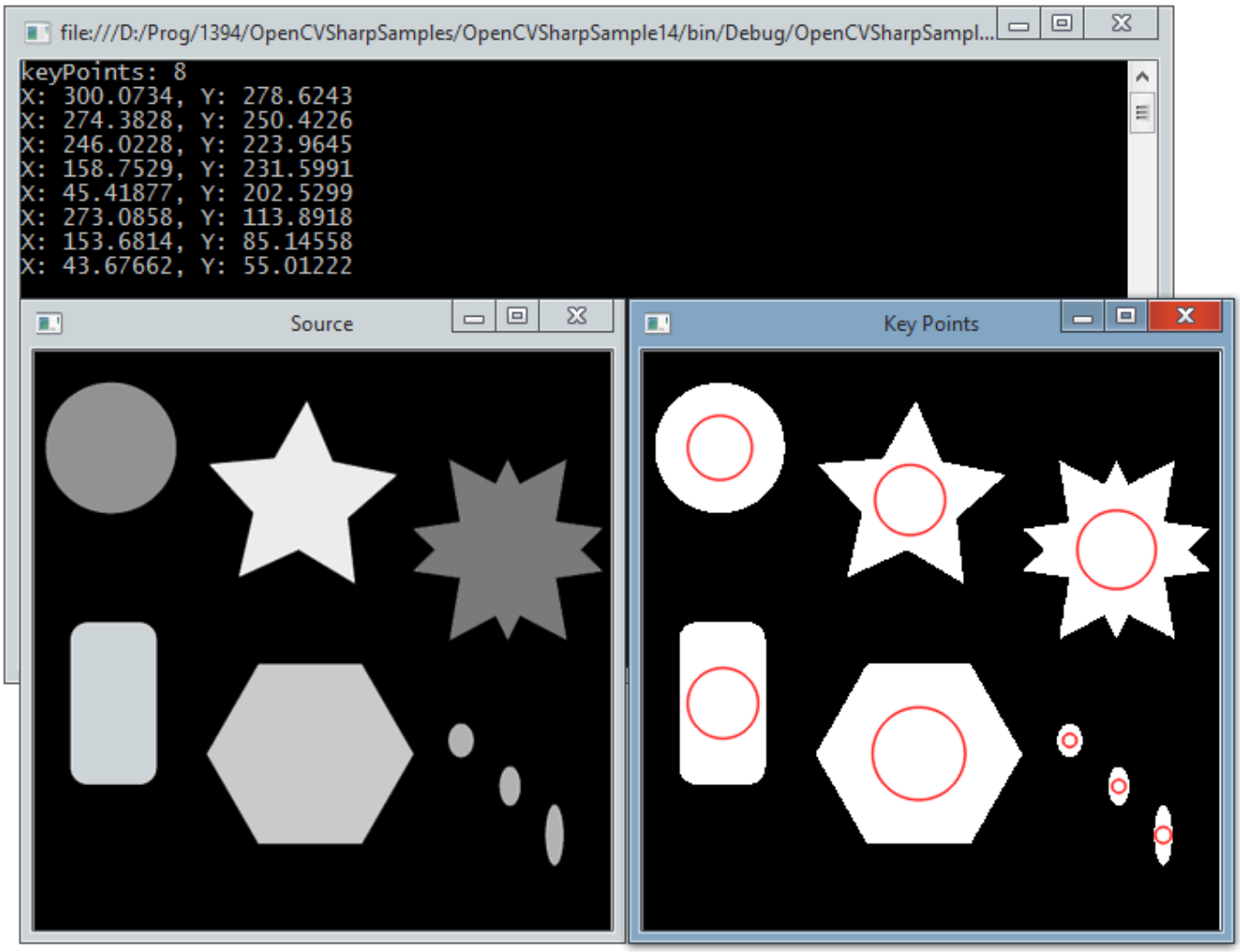
عنوان: OpenCVSharp #14
نویسنده: وحید نصیری
تاریخ: ۱۳۹۴/۰۳/۲۸ ۱۲:۲۰
آدرس: www.dotnettips.info
گروه‌ها: OpenCV

تشخیص BLOBs در تصویر به کمک OpenCV

BLOB یا **B**inary **L**arge **OB**ject به معنای گروهی از نقاط به هم پیوسته‌ی در یک تصویر باینری هستند. Large در اینجا به این معنا است که اشیایی با اندازه‌هایی مشخص، مدنظر هستند و اشیاء کوچک، به عنوان نویز درنظر گرفته خواهند شد و پردازش نمی‌شوند.
برای نمونه در تصویر ذیل، تصویر سمت چپ، تصویر اصلی است و تصویر سمت راست، نمونه‌ی باینری سیاه و سفید آن. سپس عملیات تشخیص Blobs بر روی تصویر اصلی انجام شده و نتیجه‌ی نهایی که مشخص کننده‌ی اشیاء تشخیص داده شده‌است، با دوایری قرمز در تصویر سمت راست، مشخص هستند.



معرفی کلاس SimpleBlobDetector

در کدهای ذیل، نحوه‌ی کار با کلاس SimpleBlobDetector را مشاهده می‌کنید. این کلاس کار تشخیص Blobs را در تصویر اصلی انجام می‌دهد:

```
var srcImage = new Mat(@"..\..\Images\cvlbl.png");
Cv2.ImShow("Source", srcImage);
Cv2.WaitKey(1); // do events


var binaryImage = new Mat(srcImage.Size(), MatType.CV_8UC1);

Cv2.CvtColor(srcImage, binaryImage, ColorConversion.BgrToGray);

Cv2.Threshold(binaryImage, binaryImage, thresh: 100, maxval: 255, type: ThresholdType.Binary);


var detectorParams = new SimpleBlobDetector.Params
{
    //MinDistBetweenBlobs = 10, // 10 pixels between blobs
    //MinRepeatability = 1,

    //MinThreshold = 100,
    //MaxThreshold = 255,
    //ThresholdStep = 5,

    FilterByArea = false,
    //FilterByArea = true,
    //MinArea = 0.001f, // 10 pixels squared
    //MaxArea = 500,

    FilterByCircularity = false,
    //FilterByCircularity = true,
    //MinCircularity = 0.001f,

    FilterByConvexity = false,
    //FilterByConvexity = true,
    //MinConvexity = 0.001f,
    //MaxConvexity = 10,

    FilterByInertia = false,
    //FilterByInertia = true,
    //MinInertiaRatio = 0.001f,

    FilterByColor = false
    //FilterByColor = true,
    //BlobColor = 255 // to extract light blobs
};
var simpleBlobDetector = new SimpleBlobDetector(detectorParams);
var keyPoints = simpleBlobDetector.Detect(binaryImage);

Console.WriteLine("keyPoints: {0}", keyPoints.Length);
foreach (var keyPoint in keyPoints)
{
    Console.WriteLine("X: {0}, Y: {1}", keyPoint.Pt.X, keyPoint.Pt.Y);
}

var imageWithKeyPoints = new Mat();
Cv2.DrawKeypoints(
        image: binaryImage,
        keypoints: keyPoints,
        outImage: imageWithKeyPoints,
        color: Scalar.FromRgb(255, 0, 0),
        flags: DrawMatchesFlags.DrawRichKeypoints);


Cv2.ImShow("Key Points", imageWithKeyPoints);
Cv2.WaitKey(1); // do events


Cv2.WaitKey(0);

Cv2.DestroyAllWindows();
srcImage.Dispose();
imageWithKeyPoints.Dispose();
```

توضیحات

در اینجا ابتدا تصویر منبع به صورتی متداول باگذاری شده‌است. سپس توسط متد CvtColor به نمونه‌ای سیاه و سفید و به کمک متد Threshold به یک تصویر باینری تبدیل شده‌است.

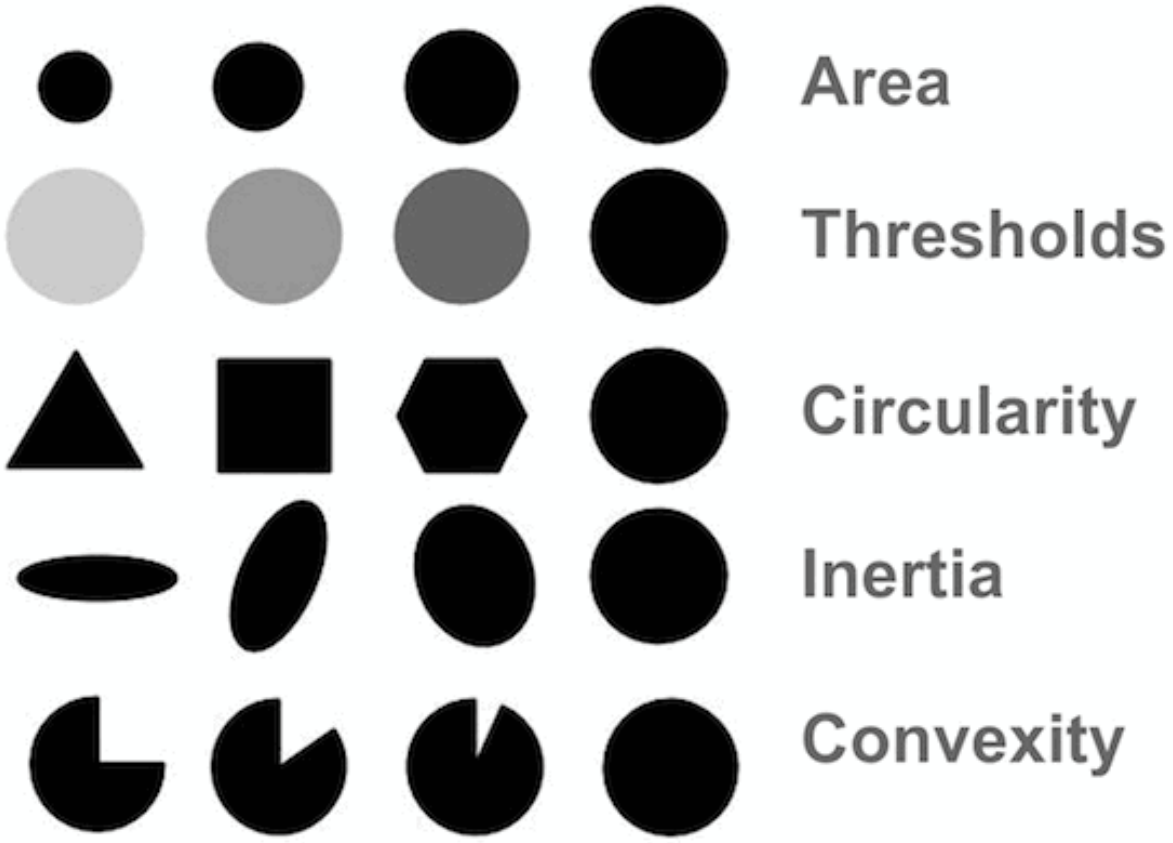
اگر به تصویر ابتدای بحث دقت کنید، اشیاء موجود در منبع مفروض، رنگ‌های متفاوتی دارند؛ اما در تصویر نهایی تولید شده، تمام

آن‌ها تبدیل به اشیایی سفید رنگ شده‌اند. این کار را متد Cv2.Threshold انجام داده‌است، تا کلاس SimpleBlobDetector قابلیت تشخیص بهتری را پیدا کند. تشخیص اشیایی یک دست، بسیار ساده‌تر هستند از تشخیص اشیایی با رنگ‌ها و شدت نور متفاوت.
اگر بخواهیم الگوریتم Threshold را بیان کنیم به pseudo code زیر خواهیم رسید:

```
if src(x,y) > thresh
    dst(x,y) = maxValue
else
    dst(x,y) = 0
```

در اینجا رنگ نقطه‌ی x,y با مقدار thresh (پارامتر سوم متد Cv2.Threshold) مقایسه می‌شود. اگر مقدار آن بیشتر بود، با پارامتر چهارم جایگزین خواهد شد. مقدار 255 در اینجا روشن‌ترین حالت ممکن است؛ اگر خیر با صفر یا تیره‌ترین رنگ، جایگزین می‌شود. به همین دلیل است که در تصویر سمت راست، تمام اشیاء را با روشن‌ترین رنگ ممکن مشاهده می‌کنید.

سپس پارامتر اصلی سازنده‌ی کلاس SimpleBlobDetector مقدار دهی شده‌است. این تنظیمات در تشخیص اشیاء موجود در تصویر بسیار مهم هستند. برای درک بهتر واژه‌هایی مانند Circularity، Convexity، Inertia و امثال آن، به تصویر ذیل دقت کنید:



در اینجا می‌توان حداقل و حداکثر تقعر و تحدب اشیاء و یا حتی حداقل و حداکثر اندازه‌ی اشیاء مدنظر را مشخص کرد.
اگر سازنده‌ی کلاس SimpleBlobDetector به نال تنظیم شود، از مقادیر پیش فرض آن استفاده خواهند شد که در اینجا به معنای تشخیص اشیاء کدر دایره‌ای شکل هستند. به همین جهت در تنظیمات فوق FilterByColor به false تنظیم شده‌است تا اشکال سفید رنگ تصویر اصلی را بتوان تشخیص داد.

در ادامه متد simpleBlobDetector.Detect بر روی تصویر باینری بهبود داده شده، فراخوانی گشته‌است. خروجی آن نقاط کلیدی اشیاء یافت شده‌است. این نقاط را می‌توان توسط متد DrawKeypoints ترسیم کرد که حاصل آن دوایر قرمز رنگ موجود در مرکز اشیاء یافت شده‌ی در تصویر سمت راست هستند.

کدهای کامل این مثال را از اینجا **می‌توانید دریافت کنید.**