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：基于直方图的目标跟踪

实验过程中遇到和解决的问题：

1. 需要计算直方图

这里直接使用了 opencv 内置的直方图计算

将 RGB 图像转换为 HSV，然后计算直方图

```
int h_bins = 50; int s_bins = 60;
int histSize[] = { h_bins, s_bins };
float h_ranges[] = { 0, 180 };
float s_ranges[] = { 0, 256 };
const float* ranges[] = { h_ranges, s_ranges };
int channels[] = { 0, 1 };
cvtColor(RectImg, RectHSV, COLOR_BGR2HSV);
calcHist(&RectHSV, 1, channels, Mat(), RectHist, 2, histSize, ranges, true, false);
normalize(RectHist, RectHist, 0, 1, NORM_MINMAX);
```

2. 需要比较相似度

这里使用巴氏距离:

```
curValue = compareHist(CurHist, RectHist, HISTCMP_BHATTACHARYYA);
```

$$d(H_1,H_2)=\sqrt{1-\frac{1}{\sqrt{\bar{H}_1\bar{H}_2N^2}}\sum_I\sqrt{H_1(I)\cdot H_2(I)}}$$

3. 鼠标交互截取目标图像

```
void onMouse(int event, int x, int y, int flags, void *)                //鼠标操作截取目标图像
{
    if (event == EVENT_LBUTTONDOWN)//左键取点 X
    {
        ButtonFlag=true;
        PointX = Point(x, y);
        PointY = PointX;
    }
    if (event == EVENT_MOUSEMOVE && ButtonFlag)//鼠标移动取点 Y
    {
        CopyImg = CurImg.clone();
        PointY = Point(x, y);
        if (PointX != PointY)
        {
            rectangle(CopyImg, PointX, PointY, Scalar(0, 0, 255), 2);
        }
```

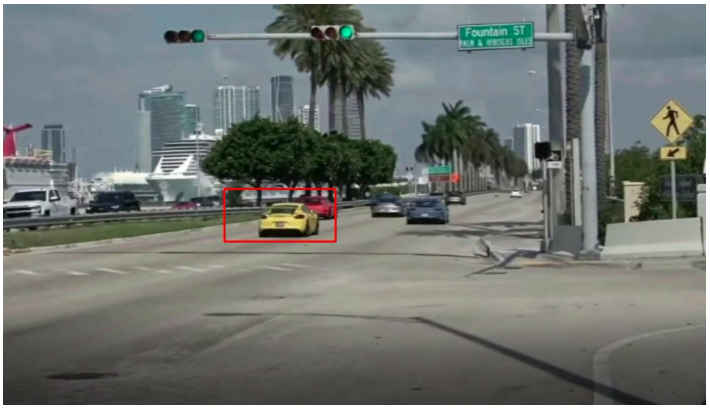
```
        imshow("Target", CopyImg);
    }
    if (event == EVENT_LBUTTONUP)//左键点击结束计算目标区域直方图和归一化
    {
        ButtonFlag=false;
        RectImg = CopyImg(Rect(PointX, PointY));
        cvtColor(RectImg, RectHSV, COLOR_BGR2HSV);
        calcHist(&RectHSV, 1, channels, Mat(), RectHist, 2, histSize, ranges, true, false);
        normalize(RectHist, RectHist, 0, 1, NORM_MINMAX);
        GetTarget=true;
    }
}
```

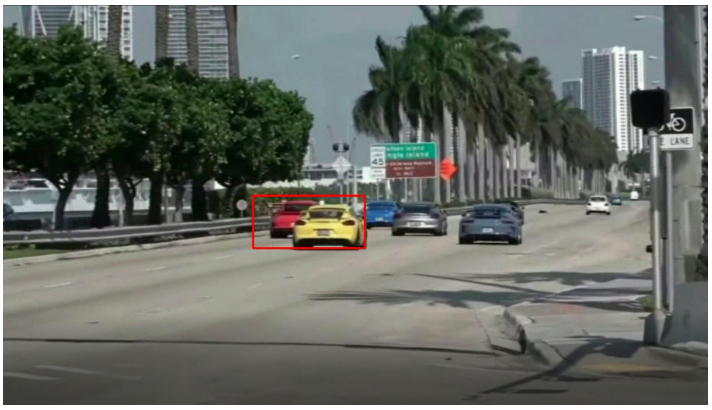
4. 寻找当前图像中与目标图像最近似的截图

```
while (true) {                                                                          //目标跟踪
    VideoIn >> CurImg;
    if (CurImg.empty() || waitKey(DisTime) == 27) break;
    int rateX = 10, rateY = 10;
    x1 = (x1 > 0) ? x1 : 0;
    y1 = (y1 > 0) ? y1 : 0;
    BestValue = 1.0;
    Point Start, End;
    for (int y = y1; y < y2; y += rateY) {
        for (Start.x = x1, Start.y = y; Start.x < x2; Start.x += rateX) {
            End.x = (Start.x + w) < W ? Start.x + w : W - 1;
            End.y = (Start.y + h) < H ? Start.y + h : H - 1;
            if (Start.x < End.x && Start.y < End.y) {
                CurRect = CurImg(Rect(Start, End));
                cvtColor(CurRect, CurHSV, COLOR_BGR2HSV);
                calcHist(&CurHSV, 1, channels, Mat(), CurHist, 2, histSize, ranges, true, false);          //直方图
                normalize(CurHist, CurHist, 0, 1, NORM_MINMAX);                                            //归一化
                curValue = compareHist(CurHist, RectHist, HISTCMP_BHATTACHARYYA);                          //计算巴氏距离
                if (curValue < BestValue) {
                    BestValue = curValue;
                    BestRect = CurRect;
                    A = Start;
                    B = End;
                }
            }
        }
    }
    x1 = A.x - w;
    y1 = A.y - h;
    x2 = A.x + w;3
    y2 = A.y + h;
```

```
        rectangle(CurImg, A, B, Scalar(0, 0, 255), 2);
        if(BestValue<0.15){                                                    //如果相似度较高就更新目标图像
            RectImg=BestRect;
            cvtColor(RectImg, RectHSV, COLOR_BGR2HSV);
            calcHist(&RectHSV, 1, channels, Mat(), RectHist, 2, histSize, ranges, true, false);
            normalize(RectHist, RectHist, 0, 1, NORM_MINMAX);
        }
        imshow("Video", CurImg);
        VideoOut << CurImg;
}
```

效果视频截图：




结果分析与体会：

-直方图相似计算使用巴氏距离，巴氏距离的值越小越相似。

-为了减少计算量，设定移动步长为 5-10，减少计算量，使得视频更加流畅。

-循环中需要更新计算区域的坐标位置，检测区域和目标图像很接近时便更新最佳区域到该位置。

-另外当最终计算的最佳区域与目标区域非常相似时，可以考虑更新目标图像以及对应直方图。