一、实验要求

利用课上学习的meanshift算法对视频中的目标进行跟踪。

具体要求包括:

(1)输入一段短视频载入一段视频，选择其中一个物体并对其进行跟踪(使用opencv库函数即可)。

(2)课下学习camshift目标跟踪方法，使用camshift方法对目标物体进行跟踪（使用opencv库函数即可)。

(3)比较camshift方法与meanshift方法的异同，并在报告中进行分析。

提交内容:

a.源码。

完成(1)(30分)完成(2)(30分)

使用c++，提交cpp文件。

b.报告。

包含实验结果分析及运行结果截图。

原视频截图+(1）中跟踪结果+(2）中跟踪结果+camshaft和meanshift的方法比较（30分)
Pdf文件，10+(30)分

二、实验步骤

1. 查看OpenCV官方文档，了解MeanShift和CamShift函数的使用方法
   MeanShift函数说明

   CamShift函数说明

   根据官方的文档所说，CamShift会调用MeanShift函数，优化了自适应窗口。其次，二者的返回也不同，MeanShift返回的是迭代的次数，而CamShift返回的是一个RotatedRect。
2. 读取视频，并对视频逐帧处理
   为了方便，这里直接调用摄像头读取视频流。

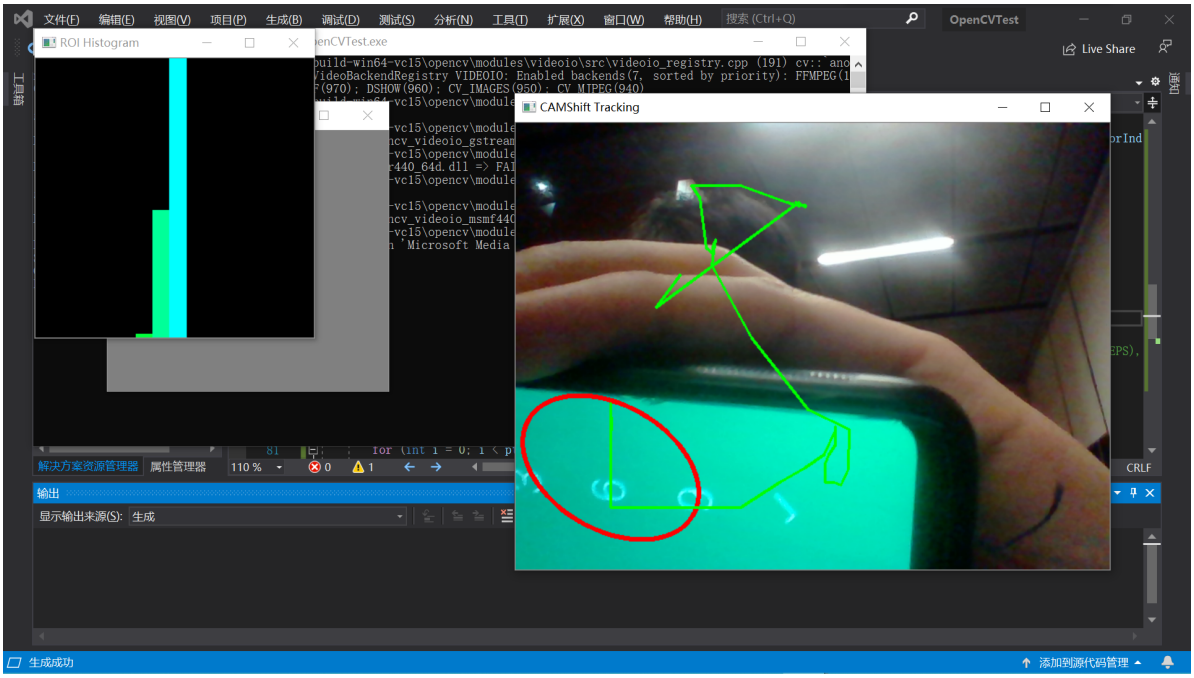
   ```
   capture.open(0);
   ```

3. 将图片转换为HSV色彩空间
   由于RGB色彩空间有3条通道，如果要处理RGB色彩空间的数据，那么数据就是5维的，不容易处理，所以转换为HSV色彩空间，只有H一个通道是色调，数据变为了3维，就容易处理多了。
4. 计算HSV空间H通道直方图
   这一步骤，要对值进行正则化，范围是0到255，这样可以避免曝光带来的影响。
5. 直方图反向映射
   为了更好地找到我们要找的目标，所以要进行直方图的反向映射。
6. MeanShift或者CamShift 位置跟踪
   这里调用了Meanshift或CamShift函数，对图片中的目标进行了追踪

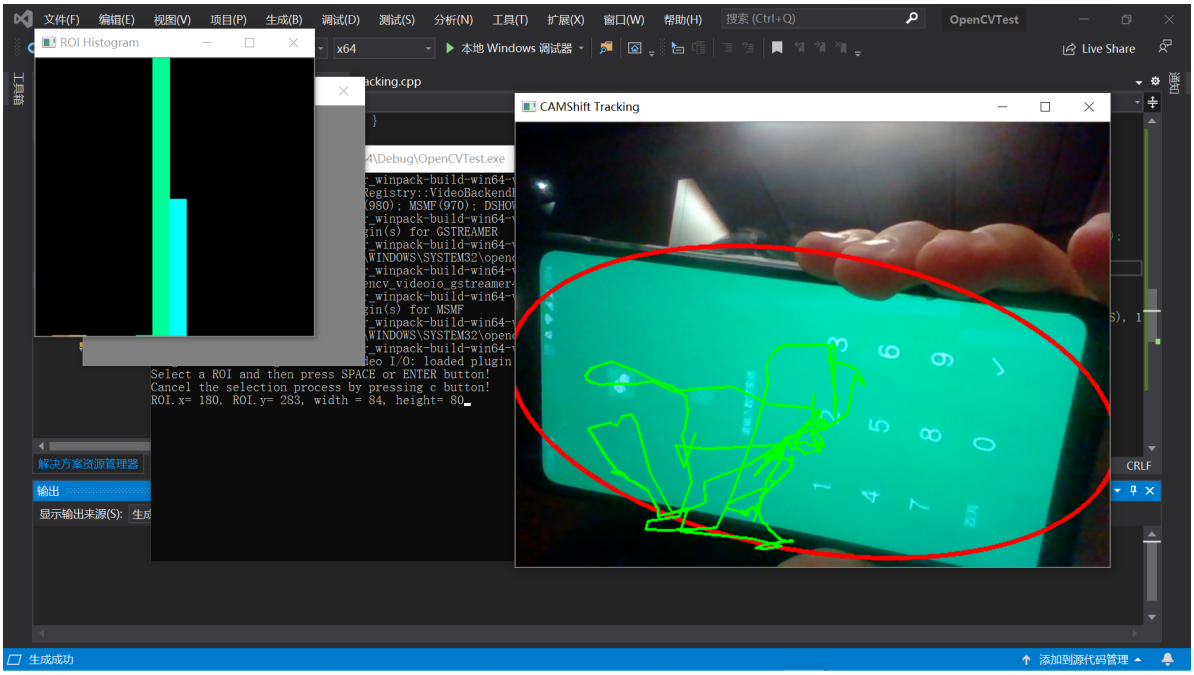
7. 绘制新位置以及画质心线
   将目标标记出来，显示在图片上

三、结果分析

在我的运行效果中，两种算法的效果差别不是很大，唯一的区别就在于Meanshift算法的识别结果不会随着距离摄像头的距离而改变，但是CamShift会随距离而放大和缩小。
MeanShift：如图，开始我距离较远，所以手机屏幕较小，靠近后椭圆大小不变。



CamShift：而CamShift一直能够完全选中我的手机屏幕。



四、源码

tracking.cpp

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

using namespace cv;
```

```
using namespace std;

int smin = 15;
int vmin = 40;
int vmax = 256;
int bins = 16;
int main(int argc, char** argv) {
    VideoCapture capture;
    capture.open(0);
    if (!capture.isOpened()) {
        printf("could not find video data file...\n");
        return -1;
    }
    namedWindow("MeanShift/CAMShift Tracking", WINDOW_AUTOSIZE);
    namedWindow("ROI Histogram", WINDOW_AUTOSIZE);

    vector<Point> pt;
    bool firstRead = true;
    float hrange[] = { 0, 180 };
    const float* hranges = hrange;
    Rect selection;
    Mat frame, hsv, hue, mask, hist, backprojection;
    Mat drawImg = Mat::zeros(300, 300, CV_8UC3);
    while (capture.read(frame)) {
        if (firstRead) {
            Rect2d first = selectROI("CAMShift Tracking", frame);
            selection.x = first.x;
            selection.y = first.y;
            selection.width = first.width;
            selection.height = first.height;
            printf("ROI.x= %d, ROI.y= %d, width = %d, height= %d", selection.x,
selection.y, selection.width, selection.height);
        }
        // convert to HSV
        cvtColor(frame, hsv, COLOR_BGR2HSV);
        inRange(hsv, Scalar(0, smin, vmin), Scalar(180, vmax, vmax), mask);
        hue = Mat(hsv.size(), hsv.depth());
        int channels[] = { 0, 0 };
        mixChannels(&hsv, 1, &hue, 1, channels, 1);

        if (firstRead) {
            // ROI 直方图计算
            Mat roi(hue, selection);
            Mat maskroi(mask, selection);
            calcHist(&roi, 1, 0, maskroi, hist, 1, &bins, &hranges);
            normalize(hist, hist, 0, 255, NORM_MINMAX);

            // show histogram
            int binw = drawImg.cols / bins;
            Mat colorIndex = Mat(1, bins, CV_8UC3);
            for (int i = 0; i < bins; i++) {
                colorIndex.at<Vec3b>(0, i) = Vec3b(saturate_cast<uchar>(i * 180
/ bins), 255, 255);
            }
            cvtColor(colorIndex, colorIndex, COLOR_HSV2BGR);
            for (int i = 0; i < bins; i++) {
                int  val = saturate_cast<int>(hist.at<float>(i) * drawImg.rows /
255);
```

```
                rectangle(drawImg, Point(i * binw, drawImg.rows), Point((i + 1)
* binw, drawImg.rows - val), Scalar(colorIndex.at<Vec3b>(0, i)), -1, 8, 0);
            }
        }

        // back projection
        calcBackProject(&hue, 1, 0, hist, backprojection, &hranges);
        // CAMShift tracking
        backprojection &= mask;
        //使用MeanShift算法
        //int num = meanShift(backprojection, selection,
TermCriteria((TermCriteria::COUNT | TermCriteria::EPS), 10, 1));
        //Point2f center = Point2f(selection.x + selection.width / 2,
selection.y + selection.height / 2);
        //RotatedRect trackBox = RotatedRect(center, Point2f(selection.width,
selection.height), 30);

        //使用CamShift算法
        RotatedRect trackBox = CamShift(backprojection, selection,
TermCriteria((TermCriteria::COUNT | TermCriteria::EPS), 10, 1));
        Point2f center = Point2f(selection.x + selection.width / 2, selection.y
+ selection.height / 2);

        // draw location on frame;
        ellipse(frame, trackBox, Scalar(0, 0, 255), 3, 8);
        pt.push_back(center);

        for (int i = 0; i < pt.size() - 1; i++)
        {
            line(frame, pt[i], pt[i + 1], Scalar(0, 255, 0), 2.5); //画质心线
        }
        if (firstRead) {
            firstRead = false;
        }
        imshow("MeanShift/CAMShift Tracking", frame);
        imshow("ROI Histogram", drawImg);
        char c = waitKey(50);// ESC
        if (c == 27) {
            break;
        }
    }

    capture.release();
    waitKey(0);
    return 0;
}
```