

OpenCV中MeanShift算法视频移动对象分析

原创：gloomyfish OpenCV学堂 昨天

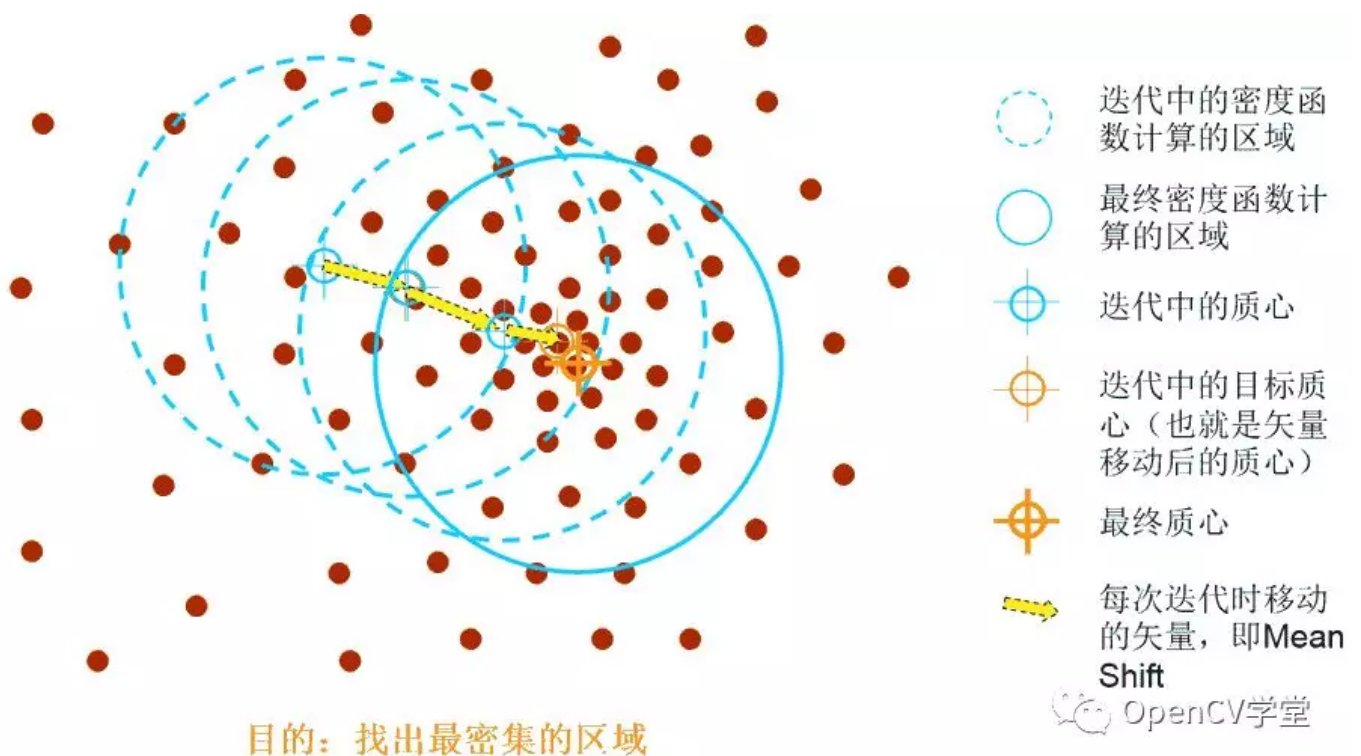


点击上方蓝字关注我们

星标或者置顶【OpenCV学堂】
干货教程第一时间送达！

MeanShift算法

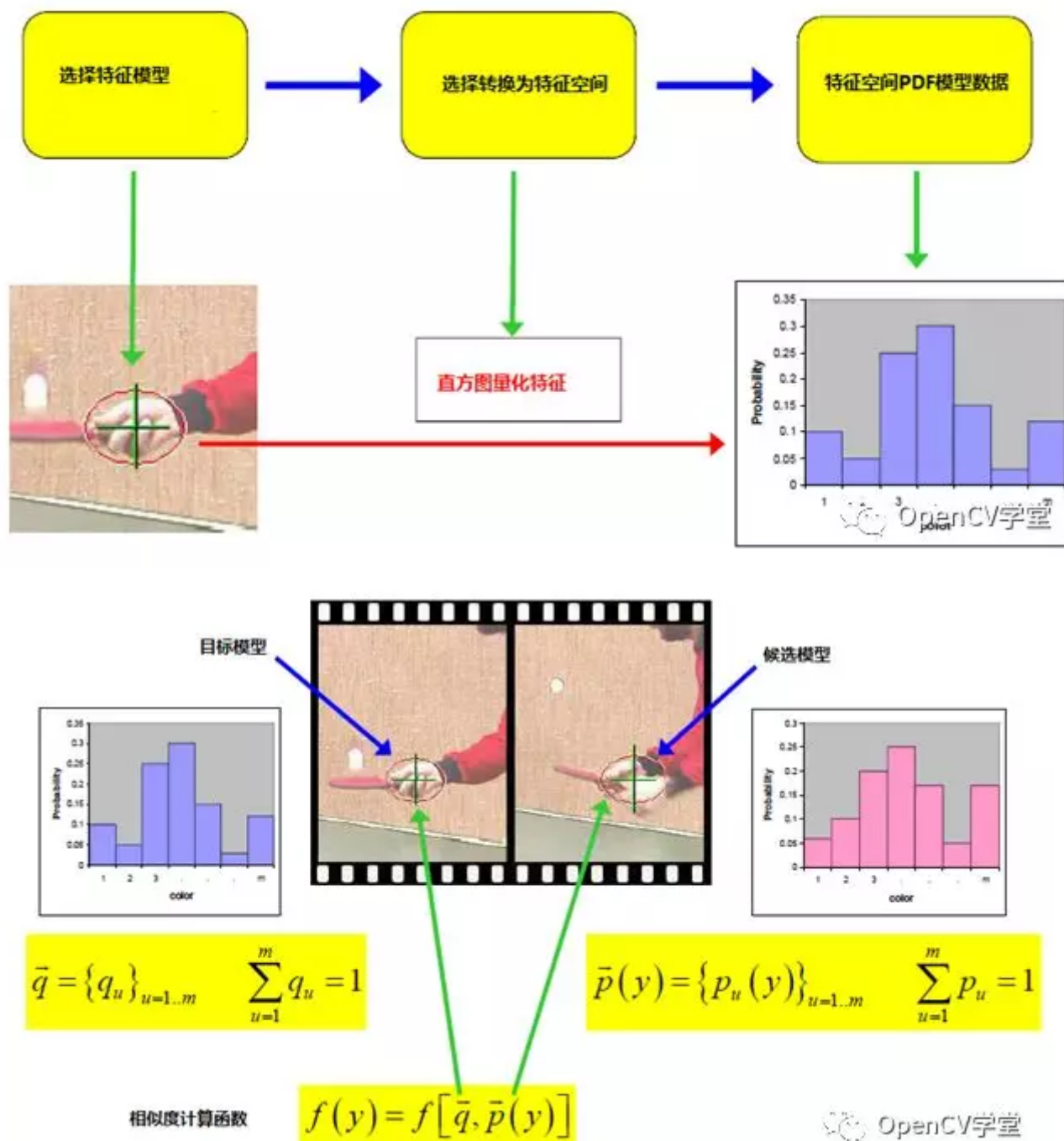
Mean Shift是一种聚类算法，在数据挖掘，图像提取，视频对象跟踪中都有应用。OpenCV在图像处理模块中使用均值迁移可以实现去噪、边缘保留滤波等操作。在视频分析模块中使用均值迁移算法结合直方图反向投影算法实现对移动对象分析，是一种非常稳定的视频移动对象跟踪算法。其核心的思想是对反向投影之后的图像做均值迁移（meanshift）从而发现密度最高的区域，也是对象分布最大的区域，均值迁移的原理可以通过下面这张图来解释：



会从初始化的中心位置，通过计算生成新的中心位置坐标， dx 与 dy 就是均值迁移每次移动的步长，移动到新的中心之后，会基于新的分布进行中心位置计算，如此不断迭代，直到中心位置处于最大分布为止。

MeanShift移动对象分析，首先会读取视频第一帧，选择ROI区域，生成直方图。然后对视频中的每一帧执行如下操作：

- 1.直方图反向投影该帧
- 2.基于前一帧的窗口位置，使用means shift寻找新的最大分布密度，生成新位置窗口
- 3.更新窗口直至最后一帧



OpenCV中meanshift的API函数如下：

```
int cv::meanShift(
    InputArray probImage,
    Rect & window,
```

```
TermCriteria criteria  
)
```

参数解释

probImage 输入图像，是直方图反向投影的结果
window 搜索窗口，ROI对象区域，每帧会自动更新窗口
criteria 均值迁移停止条件

代码演示

代码实现分为如下几个部分

1. 通过VideoCapture读取视频文件

```
VideoCapture cap("D:/images/video/balltest.mp4");
```

2. 对第一帧进行ROI选择，绘制直方图

```
// Object has been selected by user, set up CAMShift search properties once  
Mat roi(hue, selection), maskroi(mask, selection);  
calcHist(&roi, 1, 0, maskroi, hist, 1, &hsize, &phranges);  
normalize(hist, hist, 0, 255, NORM_MINMAX);  
  
trackWindow = selection;  
trackObject = 1; // Don't set up again, unless user selects new ROI  
  
histing = Scalar::all(0);  
int binW = histing.cols / hsize;  
Mat buf(1, hsize, CV_8UC3);  
for (int i = 0; i < hsize; i++)  
    buf.at<Vec3b>(i) = Vec3b(saturate_cast<uchar>(i*180. / hsize), 255, 255);  
cvtColor(buf, buf, COLOR_HSV2BGR);  
  
for (int i = 0; i < hsize; i++)  
{  
    int val = saturate_cast<int>(hist.at<float>(i)*histing.rows / 255);  
    rectangle(histing, Point(i*binW, histing.rows),  
        Point((i + 1)*binW, histing.rows - val),  
        Scalar(buf.at<Vec3b>(i)), -1, 8);  
}
```

3. 执行反向投影

```
calcBackProject(&hue, 1, 0, hist, backproj, &phranges);
```

4. 执行MeanShift均值迁移分析，得到每帧移动位置信息，并完成绘制

```
meanShift(backproj, trackWindow, TermCriteria(TermCriteria::EPS | TermCriteria::COUNT, 1  
rectangle(image, trackWindow, Scalar(0, 0, 255), 3, LINE_AA);
```

完整的演示代码如下

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

using namespace cv;
using namespace std;

Mat image;
bool selectObject = false;
int trackObject = 0;
bool showHist = true;
Point origin;
Rect selection;
int vmin = 10, vmax = 256, smin = 30;

int main(int argc, const char** argv)
{
    // VideoCapture cap(0);
    VideoCapture cap("D:/images/video/balltest.mp4");
    Rect trackWindow;
    int hsize = 16;
    float hranges[] = { 0,180 };
    const float* phranges = hranges;

    if (!cap.isOpened())
    {
        printf("could not open camera...\n");
        return -1;
    }

    namedWindow("Histogram", WINDOW_AUTOSIZE);
    namedWindow("CamShift Demo", WINDOW_AUTOSIZE);

    Mat frame, hsv, hue, mask, hist, histimg = Mat::zeros(200, 320, CV_8UC3), backproj;
    bool paused = false;
    cap.read(frame);
    Rect selection = selectROI("CamShift Demo", frame, true, false);

    while(true)
    {
        bool ret = cap.read(frame);
        if (!ret) break;
        frame.copyTo(image);

        cvtColor(image, hsv, COLOR_BGR2HSV);

        int _vmin = vmin, _vmax = vmax;
        inRange(hsv, Scalar(26, 43, 46), Scalar(34, 255, 255), mask);
        int ch[] = { 0, 0 };
```

```

hue.create(hsv.size(), hsv.depth());
mixChannels(&hsv, 1, &hue, 1, ch, 1);

if (trackObject <= 0)
{
    // Object has been selected by user, set up CAMShift search properties once
    Mat roi(hue, selection), maskroi(mask, selection);
    calcHist(&roi, 1, 0, maskroi, hist, 1, &hsize, &phranges);
    normalize(hist, hist, 0, 255, NORM_MINMAX);

    trackWindow = selection;
    trackObject = 1; // Don't set up again, unless user selects new ROI

    histimg = Scalar::all(0);
    int binW = histimg.cols / hsize;
    Mat buf(1, hsize, CV_8UC3);
    for (int i = 0; i < hsize; i++)
        buf.at<Vec3b>(i) = Vec3b(saturate_cast<uchar>(i*180. / hsize), 255, 255)
    cvtColor(buf, buf, COLOR_HSV2BGR);

    for (int i = 0; i < hsize; i++)
    {
        int val = saturate_cast<int>(hist.at<float>(i)*histimg.rows / 255);
        rectangle(histimg, Point(i*binW, histimg.rows),
            Point((i + 1)*binW, histimg.rows - val),
            Scalar(buf.at<Vec3b>(i)), -1, 8);
    }
}

// Perform meanShift
calcBackProject(&hue, 1, 0, hist, backproj, &phranges);
backproj &= mask;
meanShift(backproj, trackWindow, TermCriteria(TermCriteria::EPS | TermCriteria::
rectangle(image, trackWindow, Scalar(0, 0, 255), 3, LINE_AA);

imshow("CamShift Demo", image);
imshow("Histogram", histimg);
char c = (char)waitKey(50);
if (c == 27)
    break;
}

return 0;
}

```

显示效果如下：



欢迎扫码加入【OpenCV研习社】

- 学习OpenCV+tensorflow开发技术
- 与更多伙伴相互交流、一起学习进步

- 每周一到每周五分享知识点学习（音频+文字+源码）
- 系统化学习知识点，从易到难、由浅入深
- 直接向老师提问、每天答疑辅导




OpenCV研习社

星主: gloomyfish



长按扫码预览社群内容

和星主关系更近一步

 OpenCV学堂

推荐阅读

[OpenCV学堂-原创精华文章](#)

[《tensorflow零基础入门视频教程》](#)

[基于OpenCV与tensorflow实现实时手势识别](#)

[tensorflow风格迁移网络训练与使用](#)

[使用tensorflow layers相关API快速构建卷积神经网络](#)

[基于OpenCV Python实现二维码检测与识别](#)

[OpenCV+Tensorflow实现实时人脸识别演示](#)

[教程 | Tensorflow keras 极简神经网络构建与使用](#)

关注【OpenCV学堂】

长按或者扫码即可关注

