计算机视觉综合实验报告

8063 班 朱丹彤 3118305459

1 手机相机标定与校正

1.1 实验目的

要进行手机相机的标定与校正，首先需要建立模型。通过拍摄到的图像信息获取到物体在真实三维世界里相对应的信息，建立物体从三维世界映射到相机成像平面这一过程中的几何模型。

由于相机透镜的制造工艺，会使成像产生多种形式的畸变，例如近大远小，在世界坐标系中的直线转化到其他坐标系不在是直线等。在矫正过程中，利用畸变系数来矫正这种像差。

1.2 实验原理

1.2.1 相机标定模型与方法

定义如下的四个坐标系来建立模型：

世界坐标系（三维）：用户定义的三维世界的坐标系，描述目标物在真实世界里的位置。单位为 m。

相机坐标系（三维）：在相机上建立的坐标系，从相机的角度描述物体位置，作为沟通世界坐标系和图像/像素坐标系的中间一环。单位为 m。

图像坐标系（二维）：描述成像过程中物体从相机坐标系到图像坐标系的投影透射关系，方便进一步得到像素坐标系下的坐标。 单位为 m。

像素坐标系（二维）：描述物体成像后的像点在数字图像上（相片）的坐标，是我们真正从相机内读取到的信息所在的坐标系。单位为个（像素数目）。

通过单应性变化实现像素坐标系与世界坐标系之间的映射，假定标定棋盘位于世界坐标系中 $Z_w = 0$ 的平面，两者间坐标映射关系如下：

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = s \begin{bmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_1 & r_2 & t \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ 1 \end{bmatrix}$$

其中，u、v 表示像素坐标系中的坐标，s 表示尺度因子，X.Y 表示世界坐标系中的坐标。

如下矩阵为相机的内参矩阵，其中 $f_x = f/dx, f_y = f/dy$ 为分别在 x 轴和 y 轴上对焦距进行归一化所求得的值，dx、dy 为像元尺寸。(u_0, v_0) 为图像中心坐标。

$$\begin{bmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

张氏相机标定法利用单应性矩阵来描述世界坐标系与像素坐标系之间的映射关系，将尺度因子、内参矩阵和外参矩阵的乘积定义为单应性矩阵，如下所示：

$$H = s\begin{bmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} r_1 & r_2 & t \end{bmatrix}$$

确定角点后，可通过下述公式求得单应性矩阵 H，进一步求得内参矩阵以及外参矩阵：

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

展开化简，化为 AX=0 的齐次方程组形式，多组对应点形成超定方程组，可以运用最小二乘法求解：

$$\begin{cases} x' = \dfrac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}} \\ \mathrm{y}' = \dfrac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}} \end{cases} \rightarrow \begin{array}{l} x'(h_{31}x + h_{32}y + h_{33}) = h_{11}x + h_{12}y + h_{13} \\ \mathrm{y}'(h_{31}x + h_{32}y + h_{33}) = h_{21}x + h_{22}y + h_{23} \end{array}$$

将单应性矩阵化为 $H = sM\begin{bmatrix} r_1 & r_2 & t \end{bmatrix}$，M 为内参矩阵。通过元素对应关系以及旋转向量的约束关系求解，即可得到内参矩阵及外参矩阵。

1.2.2 相机校正原理

由于光线在远离透镜中心的地方比靠近中心的地方更加弯曲以及透镜的工艺等原因，相机会产生径向畸变和切向畸变两种类型的畸变，如图 1 所示。

通过泰勒级数展开可以对两种畸变进行描述，径向畸变可以通过下面的方程组进行纠正：

$$x_{corrected} = x(1 + \mathrm{k}_1\mathrm{r}^2 + \mathrm{k}_2\mathrm{r}^4 + k_3\,\mathrm{r}^6)$$
$$y_{corrected} = y(1 + \mathrm{k}_1\mathrm{r}^2 + \mathrm{k}_2\mathrm{r}^4 + k_3\,\mathrm{r}^6)$$

图 1 相机畸变示意图

切向畸变会造成图像中的某些点看上去的位置会比我们认为的位置要近一些，它可以通过下列方程组进行校正：

$$x_{corrected} = x + \left[2\,\mathrm{p}_1 \mathrm{xy} + \mathrm{p}_2(\mathrm{r}^2 + 2x^2)\right]$$

$$y_{corrected} = y + \left[2\,\mathrm{p}_1 \mathrm{xy} + \mathrm{p}_2(\mathrm{r}^2 + 2x^2)\right]$$

1.3 实验内容

1.3.1 相机模型建立与参数求取

张氏相机标定法采用黑白棋盘标定板作为拍摄对象，通过关联棋盘所在平面与相机成像平面，建立相机模型。首先，通过角点检测找到棋盘中各个角点在图像中的像素位置，如图 2 所示。假定棋盘在 XY 平面是静止的，Z 总是等于 0，传入角点位置(0, 0), (1, 0), (2, 0)...已知每个角点在实际世界坐标系中的位置，建立对象点与图像点之间联系，即可根据标定图得到单应性矩阵。

图 2 棋盘格角点检测

输入棋盘内角点规格 8*6，通过函数 cv2.findChessboardCorner()返回的角点是近似值，实际位置的精度受限于图像设备的精度，小于 1 个像素。利用 cv2.cornerSubPix()函数，用近似位置以及图像作为输入，计算角点的精确位置，确定亚像素角点。

部分角点的世界坐标系三维坐标以及相机坐标系二维坐标如图三所示：

```
array([[[0., 0., 0.],
        [1., 0., 0.],
        [2., 0., 0.],
        [3., 0., 0.],
        [4., 0., 0.],
        [5., 0., 0.],
        [6., 0., 0.],
        [7., 0., 0.],
```

```
array([[[499.9785 , 161.52307]],

       [[513.50214, 209.96275]],

       [[525.87115, 256.6397 ]],

       [[538.26074, 301.64148]],

       [[550.2502 , 344.89737]],

       [[561.6343 , 386.78903]],

       [[572.61646, 426.8051 ]],

       [[583.93536, 465.68988]],
```

图 3 部分角点坐标

通过已知的对象点与图像点，即可求得单应性矩阵 H，通过计算可求得相机的内参矩阵以及畸变系数矩阵，最终求得的畸变系数矩阵形式为$[k_1, k_2, p_1, p_2, k_3]$。在求得全部的内外参矩阵参数以及畸变系数后，还需要对内外参矩阵以及畸变系数矩阵进行优化。调用 cv2.getOptimalNewCameraMatrix()函数，通过其得到的自由缩放系数进行优化。如果缩放系数等于 0，表示尽可能裁剪不想要的像素，参数 1 表示保留所有像素点，同时可能引入黑色像素。最终计算的内参矩阵以及畸变系数矩阵如下所示：

```
内参矩阵：array([ [972.88494873,                0, 366.30673431]
                  [               0, 957.01281738, 514.73849197]
                  [               0,               0,               1]])
畸变系数矩阵：array([[ 0.1376898, -1.35990126, 0.01781483 , -0.01621212, 2.94943391]])
```

1.3.1 矫正结果分析

通过 1.2.2 中的校正方程式，运用内参矩阵以及畸变系数矩阵对所拍摄的图片进行校正。图 4 左边为已矫正图像，右边为原始图像。可以看出，图像中的边界相比原图更贴合直线。

图 5 为利用 matlab 相机标定工具箱对同一图片进行矫正所得结果，对比可得，矫正效果一致。

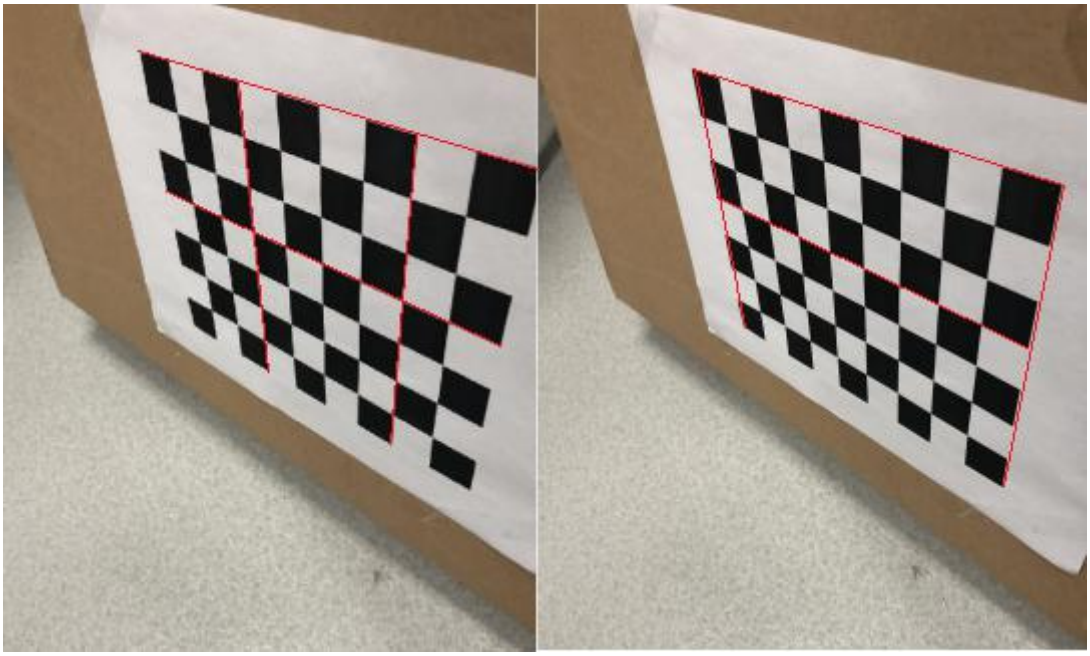

图 4 校正后图像（左）原始图像（右）

图 5 matlab 工具箱矫正结果

2 光流估计

在处理一段视频时，可以利用像素亮度的变化来追踪光线移动的方向，从而确定物体运动的方向。L-K 运动估计方法和 H-S 光流估计方法都可以实现关键点的跟踪，这两种方法分别代表了稀疏光流和稠密光流。

将图像中的每个像素与速度关联，或等价地与表示像素在连续两帧之间的位移关联，得到的是稠密光流。H-S 方法计算的就是稠密光流的速度场。稀疏光流的计算需要在被跟踪前指定一组具有明显特征的角点，使得跟踪相对稳定可靠。

2.1 实验原理

2.1.1 L-K 运动估计方法

L-K 运动估计方法是一种基于特征点的跟踪算法，首先探测当前帧的特征点，通过当前帧和下一帧灰度比较，估计当前帧特征点在下一帧的位置，实现目标跟踪。

L-K 运动估计方法基于以下三个假设：

(1)亮度恒定。对于灰度图像，假设其像素被逐帧跟踪时亮度不发生变化。

(2)小运动。即图像的运动随时间的变化比较缓慢。

(3)空间一致，一个场景上邻近的点投影到图像上也是邻近点，且邻近点速度一致。

用 $I(\mathrm{x},\mathrm{y})$ 来表示视频中某点的亮度，在假设亮度恒定的条件下，可以得到如下公式：

$$\frac{\partial I}{\partial x}\Delta x+\frac{\partial I}{\partial y}\Delta y+\frac{\partial I}{\partial t}\Delta t=0$$

$$\frac{\partial I}{\partial x}v_x+\frac{\partial I}{\partial y}v_y+\frac{\partial I}{\partial t}=0$$

其中 $v_x=u,v_y=v$ 就是光流的值。

另外，由于 L-K 运动估计方法假设是小运动，为了解决运动速度变化较快的情况，引入了图像金字塔，从图像金字塔的最高层（细节最少）开始向金字塔的底层（丰富的细节）进行跟踪。

根据假设，光流的值 (u,v) 在大小为 15*15 的窗口中为常数，那么从中即可得到一组超定方程组，通过最小二乘法求解。检测到第一帧的角点后，使用 L-K 算法迭代地跟踪这些特征点，最终实现目标跟踪。

2.1.2 H-S 光流估计方法

H-S 光流估计方法的假设条件与 2.1.1 中 L-K 运动估计方法的假设基本相同，由亮度恒定可得出如下光流约束方程：

$$\frac{\partial I}{\partial x}v_x+\frac{\partial I}{\partial y}v_y+\frac{\partial I}{\partial t}=0$$

方程中存在两个未知数，要求解光流的值(u,v)，还需引入光流的全局平滑约束条件，即(u,v)随着像素点移动而发生的改变是缓慢的，通过对光流速度分量的二阶导数进行规则化获得：

$$\frac{\partial}{\partial x}\frac{\partial v_x}{\partial x}-\frac{1}{\alpha}\frac{\partial I}{\partial x}(\frac{\partial I}{\partial x}v_x+\frac{\partial I}{\partial y}v_y+\frac{\partial I}{\partial t})=0$$
$$\frac{\partial}{\partial y}\frac{\partial v_y}{\partial y}-\frac{1}{\alpha}\frac{\partial I}{\partial y}(\frac{\partial I}{\partial x}v_x+\frac{\partial I}{\partial y}v_y+\frac{\partial I}{\partial t})=0$$

其中α是不变的权重系数，称为规则化常数，一般为 1。α的值较大可以获得更加局部一致的运动流向量。

联立即可求得(u, v)，通过迭代完成对光流的估计。计算两次迭代光流值的误差，如果小于给定误差阈值，则计算过程结束。

2.2 实验内容

读取视频文件后，首先选取第一帧，通过函数 cv2.goodFeaturesToTrack 在第一帧图像中检测角点，作为光流追踪起始点，得到 L-K 运动估计方法所跟踪的特征点，如图 2-1 所示:

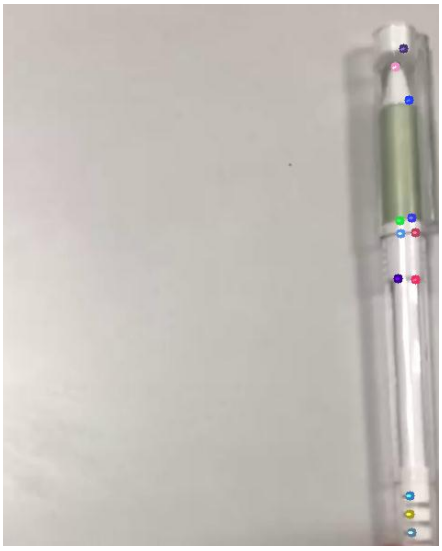

图 2-1 光流追踪起始点

函数的返回值为角点的位置数组，展示于图 2-2 中：

```
array([[[525., 709.]],

       [[516., 383.]],

       [[500., 332.]],

       [[494., 152.]],

       [[516., 331.]],

       [[509., 637.]],

       [[512., 315.]],

       [[509., 617.]],

       [[497., 381.]],

       [[530., 703.]],

       [[499., 318.]],

       [[511., 657.]],

       [[509., 188.]],

       [[503., 132.]]], dtype=float32)
```

图 2-2 光流追踪起始点位置

得到这些角点后，使用 L-K 算法来迭代地跟踪这些特征点。在 L-K 算法的特征参数中，需要设置窗口的尺寸以及图像金字塔的层数。窗口的尺寸是计算局部连续运动的窗口尺寸。根据假设，光流的值（u,v）在大小为 15*15 的窗口中为常数，由此可列出方程求得特征点位置。

通过调用函数 cv2.calcOpticalFlowPyrLK 来实现迭代。函数的输入值为上一帧的图片、上一帧中的特征点、当前帧的图片以及参数。返回值包含一个二维点的向量、标志位以及向量中的每个特征对应的错误率。其中二维点的向量可以是用来作为光流算法的输入特征点，也可以是光流算法在当前帧找到特征点的新位置，取决于标志位的值。标志位为 1 或 0，如果在当前帧找到了上一帧中的点，那么这个点的状态就是 1，否则就是 0。通过算法逐步更新特征角点的位置，最终通过选取跟踪成功的点，估计物体在视频流中的运动轨迹。

结果如下列图片所示：

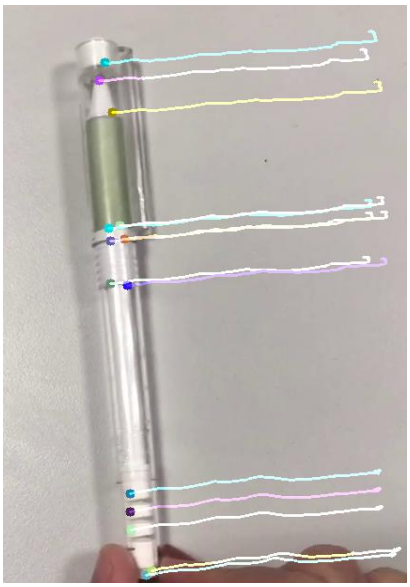

图 2-3 L-K 算法物体追踪结果 1

图 2-4 L-K 算法物体追踪结果 2

由图片可以看出，算法实现了较为准确的目标跟踪，且对于不同视频也具有一定的适应性。

3　静止背景下的多目标追踪

随着计算机技术以及智能汽车行业的发展，多目标的检测与追踪的实用性与研究价值逐渐提高。

静止背景下的多目标追踪可分为两步来实现，第一步是在视频文件的每帧中检测出移动的目标，第二步是将检测到的目标与跟踪轨迹实时匹配。在本次实验中，利用混合高斯模型进行背景减除，使用形态学操作消除噪声，通过卡尔曼滤波预测目标位置，最后利用匈牙利算法进行匹配，实现静止背景下的多目标追踪。

3.1 实验原理

3.1.1 混合高斯模型

利用混合高斯模型(GMM)可以进行背景减除，将前后景分离，得到移动的目标。对每个像素点建立由 k 个单高斯模型线性叠加而成的模型，在这些混合高斯背景模型中，认为像素之间的颜色信息互不相关，对各像素点的处理都是相互独立的。单个像素点在 t 时刻服从混合高斯分布概率密度函数：

$$p(\mathrm{x}_t)=\sum_{i=1}^{k} w_{i,t}\eta(\mathrm{x}_t,\mu_{i,t},\tau_{i,t})$$

其中 k 为分布模式总数，$\eta(\mathrm{x}_t,\mu_{i,t},\tau_{i,t})$ 为 t 时刻第 i 个高斯分布，$\mu_{i,t}$ 为其均值，$\tau_{i,t}$ 为其协方差矩阵。

在获得新一帧图像后更新混合高斯模型。用图像中的每个像素点与混合高斯模型匹配,如果成功则判定该点为背景点, 否则为前景点。当背景更新完成后，高斯模型与背景相关程度越大，标准差就越小，权值越大。按照权值与标准差的比值从大到小进行排序，取前 B 个模型作为背景。

$$B=\arg\left(\min\left(\sum_{k=1}^{b} w_k>T\right)\right)$$

3.1.2 卡尔曼滤波

试验中利用卡尔曼滤波计算并预测每个轨迹在下一帧中的位置，建立帧间轨迹的关系。卡尔曼滤波将跟踪分为 5 种状态：新目标出现、目标匹配、目标遮挡、目标分离以及目标消失。其状态方程如下所示:

$$X(\mathrm{k}+1)=A(\mathrm{k}+1,k)X(\mathrm{k})+w(\mathrm{k})$$

$$X(\mathrm{k})=[\mathrm{x(k)},\mathrm{y(k)},\mathrm{w(k)},\mathrm{h(k)},\mathrm{v(k)}]$$

其中 x,y,w,h 分别表示目标外接矩形的横纵坐标以及长、宽， $\mathrm{w(k)},v(\mathrm{k})$ 为不相关的高斯白噪声。

定义其观测方程为 $Z(\mathrm{k})=\mathrm{H(k)}\,\mathrm{X(k)}+v(\mathrm{k})$ 。

定义好了观测方程与状态方程之后就可以用卡尔曼滤波器实现运动目标的跟踪，步骤如下：

1）计算运动目标的特征信息（运动质心，以及外接矩形）。

2）用得到的特征信息初始化卡尔曼滤波器。

3）用卡尔曼滤波器对下一帧中对应的目标区域进行预测，当下一帧到来时，在预测区域内进行目标匹配。

4）如果匹配成功，则更新卡尔曼滤波器。

3.1.3 匈牙利匹配算法

匈牙利匹配算法是一种利用增广路径求取二分图最大匹配的算法。在实验中，用于将新一帧图片中检测到的运动物体匹配到对应的轨迹。匹配的过程是通过最小化卡尔曼预测得到的质心与检测到的质心之间的欧氏距离之和实现的。

通过卡尔曼滤波计算并预测每个轨迹在下一帧中的位置，然后计算预测的轨迹位置和每个新检测到的目标之间的欧几里得距离，将度量结果作为损失函数矩阵。损失矩阵的大小为（M,N），其中 M 是轨迹数目，N 是检测到的运动物体数目。

3.2 实验内容

3.2.1 目标检测

要实现目标检测，首先利用混合高斯模型区分前景背景。通过调用函数 `vision.ForegroundDetector` 设置检测子为混合高斯模型，其中参数分别为高斯核数目、训练背景帧数以及背景阈值。函数的返回值为一个二进制掩码，其中 1 的像素值对应前景，0 的像素值对应背景。

完成背景减除后，通过设置 blob 分析子来寻找连通域，函数设置的参数为最小区域面积，返回值为目标面积、质心和边界框。

在检测目标的过程中，利用形态学运算中的开运算以及闭运算可以消除噪声，使目标检测更为准确。开运算是通过先腐蚀再膨胀，去除孤立的像素点、总的位置和结构不变。而闭运算是先膨胀再腐蚀，弥合小裂缝，而总的位置和形状不变，通过填充图像的凹角来滤波图像。其效果如图 3.1 所展示：

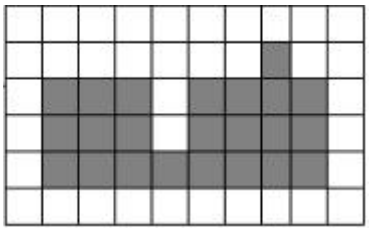

图 3.1.1 原始像素点

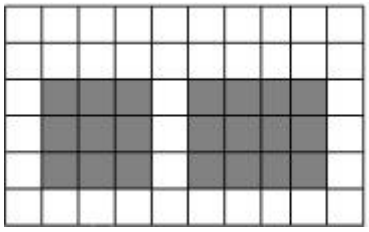

图 3.1.2 开运算效果图

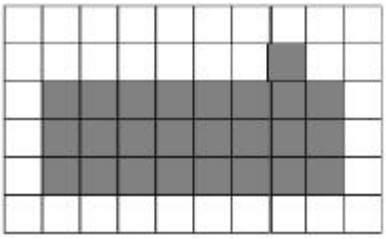

图 3.1.3 闭运算效果图

图 3.1　形态学运算效果图

滤除噪声后，使用 blob 分析得到所有连通域的中心以及边界框的大小。

视频中对移动目标的检测结果如图 3.2,3.3 所示：

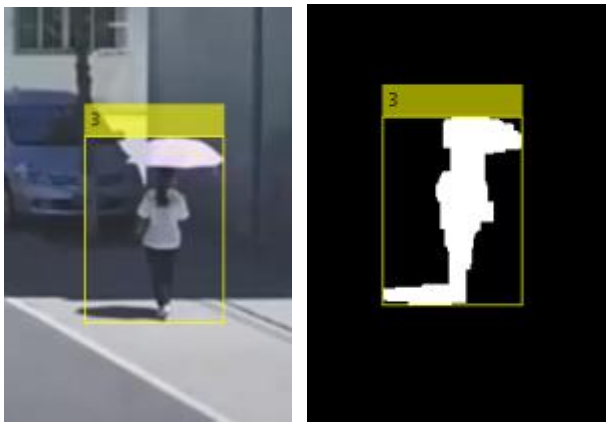

图 3.2 移动目标检测结果 1

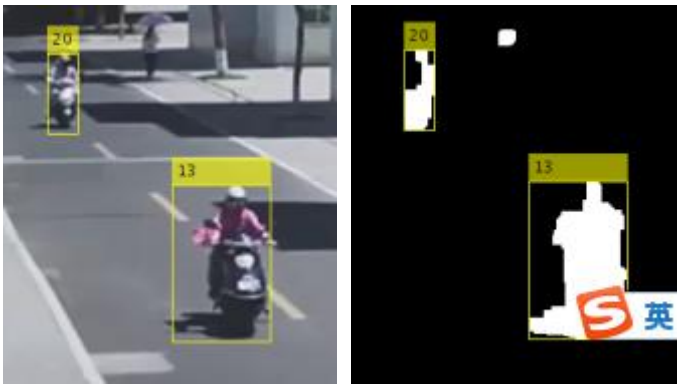

图 3.3 移动目标检测结果 2

3.2.2 目标跟踪

要进行目标跟踪，首先需要进行轨迹初始化，通过函数 initializeTracks()来进行初始化，每一个轨迹代表视频中一个移动的目标。轨迹的结构包含如下信息：

1) ID，轨迹编号；
2) Bbox，目标的边界框；
3) kalmanFilter，用于预测目标位置的卡尔曼滤波器；
4) Age，目标被检测到的总帧数；
5) totalVisibleCount，目标可被检测到的全部帧数；
6) consecutiveInvisibleCount：连续未检测到目标的帧数。

为消除噪声对目标追踪的影响，仅在 totalVisibleCount 超过阈值时才显示目标的轨迹。当连续几帧没有检测到与跟踪相关的信息时，则假设该对象已经离开了可视图画面。通过参数 consecutiveinvisiblecount 可判断这种情况，当其超过阈值时，删除跟踪轨迹。如果跟踪时间较短，并且在大多数帧中标记为不可见，那么轨迹也可能作为噪声被删除。

轨迹初始化完成后，通过卡尔曼滤波计算并预测每个轨迹在下一帧的位置，并调整目标边界框的位置，使其中心到达预测位置。

完成位置预测后，计算预测的轨迹位置和每个新检测到的目标之间的欧几里得距离，将度量结果作为损失函数矩阵。损失矩阵的大小为（M,N），其中 M 是轨迹数目，N 是检测到的运动物体数目。

通过函数 assignDetectionsToTracks(cost, costOfNonAssignment)利用匈牙利匹配算法将新一帧图片中检测到的运动物体匹配到对应的轨迹。其中输入的参数为损失矩阵以及阈值，低于阈值时，取消匹配。返回值为匹配的结果以及未匹配成功的轨迹以及目标。

完成匹配后，对已分配的轨迹，将其更新至当前帧目标所在位置，对未分配的轨迹，增加其连续不可见帧数。设置两个阈值，invisibleForLong 代表当连续不可见帧数大于它时，删除轨迹；ageThreshold 代表当总出现帧数小于它时，当该参数与总可见帧数的比值小于 0.6 时，删除轨迹。

目标检测与轨迹跟踪的操作循环进行，直至视频结束，显示最终跟踪结果。

3.3 实验结果分析

在本次试验中，通过混合高斯模型背景减除，形态学操作消除噪声，利用卡尔曼滤波预测每个轨迹在下一帧中的位置，最后通过匈牙利匹配算法完成目标与轨迹之间的匹配。实现了在静止背景下的多目标检测与跟踪。

如图 3.4 所示可见，对于匀速移动的目标，检测效果较好，可以实现较好的目标检测与跟踪。

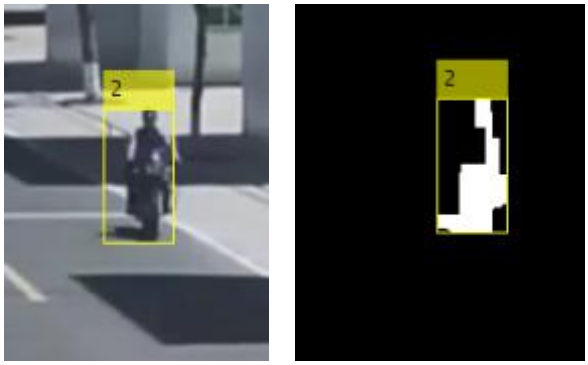

图 3.4 匀速移动目标检测结果

但静止背景下，基于动态的多目标追踪，很容易受到环境的影响。当检测目标为行人时，风吹动树叶或是有车辆经过，甚至是光照导致的影子变化都会很大程度地影响跟踪效果。图 3.5 展示了明显的失败样例：

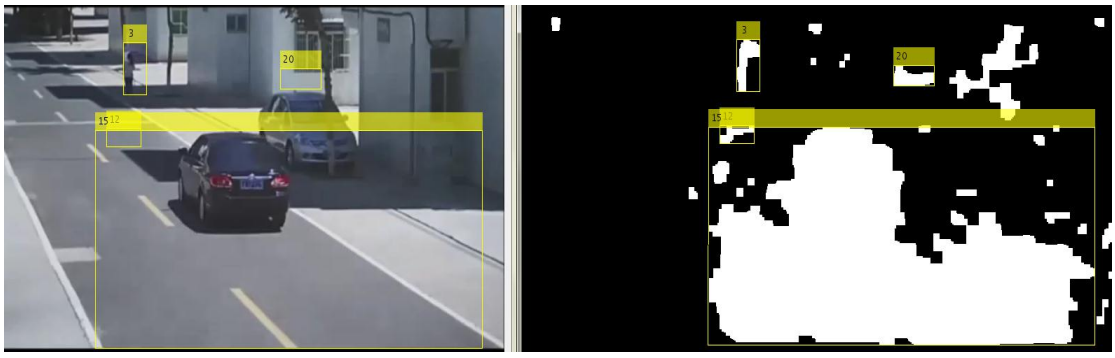

图 3.4 目标追踪失败样例

在实验中使用的卡尔曼滤波预测目标下一帧所在位置的模型，只适用于匀速变化，而汽车存在加速运动。从右边的二值图像可以看出，汽车的车灯造成像素点大量变化，只使用帧间相减以及形态学操作来得到检测目标的本实验并不适用于此。造成了较大的误差，导致检测以及跟踪目标失败。要规避由前景中不同物体造成的实验误差，可使用其他方法在前景检测中对检测到的物体进行分类。

通过多次试验发现，程序参数的鲁棒性较差，在进行形态学计算时，不同的视频调整开运算以及闭运算的结构元素，目标检测的效果差异很大。

在后续的学习过程中可以继续对本次实验进行更新修正，得到更好的实验效果。