Project 4:基于 MeanShift 的目标跟踪

1.实验目的

（1）利用 Car_Data 文件夹中的视频序列实现基于 Mean Shift（均值漂移）目标跟踪。

（2）实时性问题的分析（每秒能处理多少帧）

2.实验原理

（1）Mean Shift

Mean Shift 向量定义为

$$\mathrm{M}_h = \frac{1}{k}\sum_{x_i \in S_k}(x_i - x)$$

即基准点 x 的偏移向量为中心点指向区域内各点的向量之和再求平均。因为样本点更多的落在沿着概率密度梯度的方向上，因此$\mathrm{M_h}$指向概率密度函数的梯度方向。经过多轮迭代，质心的位置最终收敛，落在概率密度最大的地方。

从上式我们可以看出，只要是落在区域内的点，对于 x 偏移量的贡献或影响是一样大的（权重均为 1/N），但在现实跟踪过程中，当跟踪目标出现遮挡等影响时，由于外层的像素值容易受遮挡或背景的影响，所以目标模型中心附近的像素比靠外的像素更可靠。因此，对于所有采样点，每个样本点的重要性应该是不同的，离中心点越远，其权值应该越小。故引入核函数和权重系数来提高跟踪算法的鲁棒性并增加搜索跟踪能力。

（2）核函数

核函数$\mathrm{K}: R_d \to \mathrm{R}, \mathrm{K(x)} = \mathrm{k}(\left|\left|\mathrm{x}\right|\right|^2)$也叫窗口函数，在核估计中起到平滑的作用。本次实验用到的核函数为高斯核函数，它的表达形式为

$$\mathrm{K(x)}= \frac{1}{\sqrt{2\pi}} e^{\frac{-||x||^2}{2}}$$

（3）基于 mean shift 的目标跟踪法

分别计算目标区域和候选区域内像素的特征值概率，得到关于目标模型和候选模型的描述（概率密度直方图），然后利用相似度函数度量初始帧目标模型和当前帧的候选模型的相似度，选择使相似度最大的 MeanShift 向量进行移动。通过不断的迭代计算，算法最终将收敛到目标的真实位置，达到跟踪的目的。

3.实验步骤

step 1：初始化参数

变量	含义	初始值
nbin	直方图中将灰度值划分为 nbin 个 bin	16
w	矩形框的宽度	21
l	矩形框的长	21
pos	第一帧矩形框左上角的坐标	[35,132]
h	带宽	(w/2)^2+(h/2)^2
threshold	迭代的误差精度	0.2
maxiter	最大迭代次数	5
dis_k	矩形框内每个点到中心点的距离权重	dis_k=$\frac{1}{\sqrt{2\pi}}e^{\frac{-\|\|X\|\|^2}{2}}$ X=$\frac{x-x_c}{h}$
dis_g	g(x)=-K'(x)	dis_g=X*dis_k

Step 2：目标模型的q_u概率密度（u=1…nbin）可以表示为

$$q_u = C\sum_{i=1}^{n} K\left(\left\|z_i^*\right\|^2\right)\delta[b(z_i)-u]$$

$$C = 1/\sum_{i=1}^{n} K(\left\|z_i^*\right\|^2)$$

$$z_i^* = (\frac{(x_i-x_0)^2+(y-y_0)^2}{x_0^2+y_0^2})^{\frac{1}{2}}$$

即对于第一帧中目标所在的矩形区域进行遍历，根据灰度值判断当前像素点属于直方图中的第 i 个 bin，将当前点的 s 距离权重加入第 i 个 bin。遍历完成后，整个直方图 q 乘以归一化系数 C。

Step 3：对前一次（或前一帧）标定的区域进行和 step2 中同样操作，得到概率直方图 p。

Step 4：目标模型和候选模型的相似度

$$\rho(p,q) = \sum_{u=1}^{nbin}\sqrt{q_u p_u}$$

$$\approx \frac{1}{2}\sum_{u=1}^{nbin}\sqrt{q_u p_u} + \frac{C}{2}\sum_{i=1}^{n} w_i K(||\frac{f-z_i}{h}||^2)$$

只有第二项随 f 变化，其极大化过程就可以通过候选区域中心向真实区域中心的 meanshift 迭代方程完成。

$$\mathrm{f}_{k+1} = f_k + \frac{\sum_{i=1}^{n} w_i (f_k - z_i) g(||\frac{f - z_i}{h}||^2)}{\sum_{i=1}^{n} w_i g(||\frac{f - z_i}{h}||^2)}$$

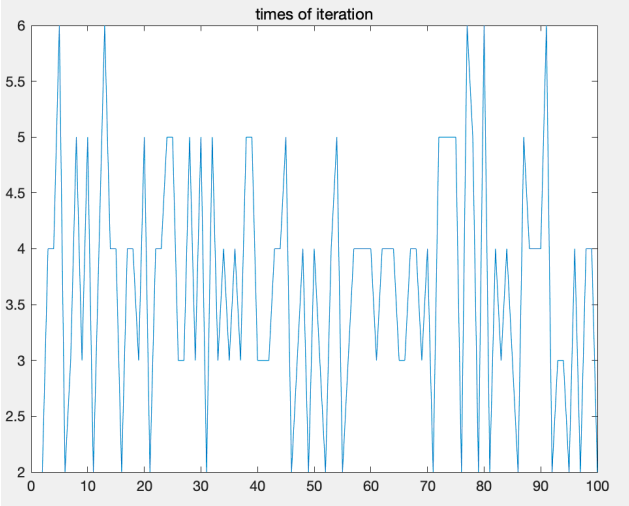
如果此时的迭代变化量小于迭代的误差精度，即当前中心坐标为真实区域中心坐标，进入下一帧的目标跟踪。否则转入 step2，直至算法收敛。

4.实验结果

运行文件夹中 KernelTrack.m 即可得到结果。变量 pos_seq 存储了第一帧到最后一帧标定目标矩形框的左上角坐标。

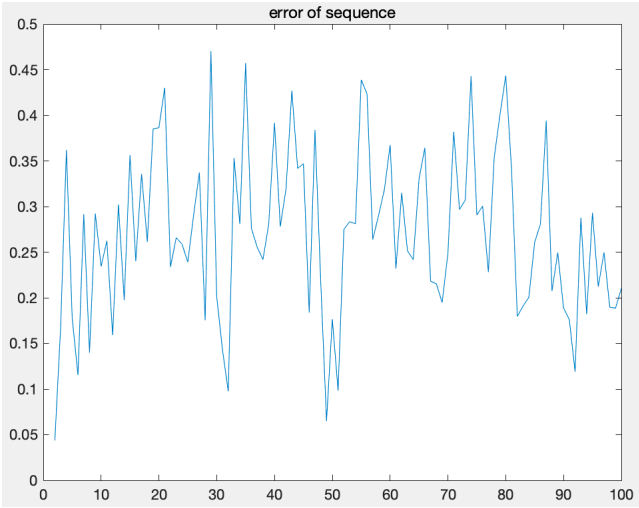
5.结果分析

（1）迭代速度和误差。算法在每一帧图片上的迭代次数如下图所示，



可以看出平均迭代 3.5 次左右算法收敛。

每一帧的精度误差为



可以看出平均误差约为 0.25。

算法的收敛速度很快，精度也较好。

（2）实时性分析。程序处理 100 帧图片需要用到的时间为 0.206535 秒。即每秒钟可以处理约 484 张图片。远高于每秒 24 帧的实时性要求。原因在于算法的时间复杂度较小，每次只对前一次标定区域内的像素点进行计算，且有些变量可以储存起来重复使用，无需多次计算，如像素点和灰度直方图中 bin 的隶属关系。有些变量的计算可以放在同一个循环里进行，如 g(x)和 k(x)。

（3）基于 meanshift 目标跟踪方法的优点在于：

算法计算量不大，在目标区域已知的情况下完全可以做到实时跟踪；

采用核函数直方图模型，对边缘遮挡、目标旋转、变形和背景运动不敏感。

缺点：

跟踪过程中由于窗口宽度大小保持不变，框出的区域不会随着目标的扩大（或缩小）而扩大（或缩小）；

当目标速度较快时，跟踪效果不好;

直方图特征在目标颜色特征描述方面略显匮乏，缺少空间信息。

6.参考资料

[1] https://blog.csdn.net/qq_22562949/article/details/49591205