# 动态识别期末报告

摘要

我实现了基于opencv图像处理的动态物体追踪标记。为保证鲁棒性，我首先通过背景建模法进行前景检测，进而得到差分图以进行后续的识别，在之后我又进行了形态学转换、寻找轮廓、以及我重点处理的前后帧的信息对比，从而最终实现动态物体的追踪和标记。

## 1、资料简介

本项目的初衷是实现动态物体的识别以及追踪标记，这对于人来说轻而易举，对机器来说却不容易办到，为了克服这个问题，首先要了解机器是怎么看一张“图片”的，在机器眼中，图片就是一个矩阵，视频本身就是一个个矩阵根据时间先后的顺序连续播放。那么该如何从矩阵中提取出我们需要的信息，这就需要一系列的算法。

在如何处理图像问题的选择上，我选择了Python+Opencv的组合，Python语言的简洁和Opencv的轻量高效使得问题较为容易解决。

**1.1、前景检测**

为了完成这项任务，我先后尝试过两种方法进行前景检测：平均背景法、帧差法和背景建模法。经过对比后发现，三者中的背景建模法最为准确有效，鲁棒性最佳。背景建模法里我选用了基于高斯混合模型（GMM）的cv2.createBackgroundSubtractorMOG2（）

其原理是在不知道图像历史的时候，假设每个像素点的值都是可以分解为一组adaptive Gaussian，跟随光照条件的变化而变化。像素值的历史由一组合适数量的高斯分布进行建模，包括每个分布的权重。

每次新图像输入的时候都会用这一组高斯分布进行评估，如果像素匹配上其中一个分布就会认为这个像素属于背景，而高斯分布的均值和方差等参数会用当前像素的值进行更新。

标记为前景的像素通过connected component analysis进行分组

**1.2、形态学转换**

在图像的操作中，我对已经二值化的图片先后进行了开运算和闭运算。以排除诸多干扰。

1.2.1、对二值图像开运算

定义：先腐蚀再膨胀，相当于先用结构元素B对A腐蚀，再对腐蚀结果用同样的结构元进行膨胀操作。

等幂性：对同样的A，做多次开运算的结果与做一次是一样的。

结果：开运算的边界是由这样一些点组成的，就是当B沿A的内部边界滚动时，B中所能达到的A的内部边界的最远的点。

1.2.2、对二值图像闭运算

定义：先膨胀再腐蚀，相当于先用结构元B对A进行膨胀，再对膨胀结果用同样的结构元进行腐蚀操作，过程与开运算正好相反。

等幂性：对同样的A，做多次闭运算的结果与做一次是一样的。

就像腐蚀和膨胀的关系一样，开和闭也是关于集合补和反转的对偶。

1.2.3、基本作用

从开、闭运算的基本定义和运行过程可以看出，这两种集合操作的所能导致的大致效果如下：开运算通常对图像轮廓进行平滑，使狭窄的“地峡”形状断开，去掉细的突起。闭运算也是趋向于平滑图像的轮廓，但于开运算相反，它一般使窄的断开部位和细长的沟熔合，填补轮廓上的间隙。

**1.3、寻找轮廓以及前后帧的信息对比**

在这一步处理上主要用到Opencv的findContours（）轮廓查找，以及后续对轮廓信息的保存和处理所用的面向对象编程的内容。

## 2、实现细节

**2.1、开闭运算细节**

由于是对二值图进行操作，所以开运算选取元素全为1的3\*3的矩阵作为结构元素，闭运算选取11\*11的元素为1 的矩阵。

**2.2、对轮廓信息的处理**

对轮廓信息的处理，以及如何将前后帧的信息联系起来是解决问题的重点。我分以下几步进行处理。

2.2.1、初步筛选

在找到轮廓后，简单不做处理是不行的，由于受到亮度变化、自然震动等因素的干扰，会产生许多干扰轮廓。因此需要对各个轮廓进行筛选，解决方法是直接根据轮廓大小先排除一部分。轮廓大小小于一定值的不会参与到后续筛选中。

2.2.2、前后帧联系

经过分析得到，前后帧的轮廓如果位置相近，那么可以将其看作同一物体，重点在于如何判断其是否相近以及该如何处理干扰项，为此，我用到了Opencv的moment（）用以提取轮廓的特征位置，用Python的list保存该帧得到的所有轮廓的位置。但在存入前先检查是否与已经保存的物体位置相近，若相近则不向list中添加新对象而是更新那个对象的位置信息。

2.2.3、创建新对象以及对象的list以保存结果

轮廓的有效性信息有：轮廓的位置，该轮廓出现的次序，该轮廓出现的时间和其寿命。于是，我顺其自然地想到将这些和类的要素一一对应联系起来。

2.2.4、干扰项处理

为防止已经超出镜头检测范围的物体干扰检查，我设置了两个个类中的bool变量用以随时获取该物体的状态（是否检测完毕/在最新一次中检查是否改变过位置），每当当前帧进行完所有轮廓检测时，如若发现这一次没有变动过位置或超出一定范围则将其结束（获得结束时刻和寿命且不会参与下一轮检测），根据寿命（即出现时间长短）判定其是否为干扰项，若为干扰项则将其移除出list。

2.2.5、序号问题

在解决问题的途中，实现编号的稳定是一个难点，上文提到的干扰项处理之后，如果有元素被移除，则需要进行物品编号的再排序，这里我使用两层嵌套循环实现剔除干扰项和id的重新排序。

## 3、结论

我的物体动态识别系统实现了动态物体的识别、跟踪、标记，应对亮度变化的抗干扰能力较强，在识别速度和准确度上达到了较好的效果。