第三次实验报告

赵博瑄 2015201907 樊树霖 2015201919 牟善磊 2015201909 王宇睿 2015201906

一、前期回顾

在第一次实验中，小组利用超声波测距，根据两组超声波配件的距离差判断小车的方向；其次，小组将手机通过蓝牙向小车发送指令，改变小车的运动方向。在第二次实验中，小组将第一次实验中发送指令的发送端由手机改为电脑，通过 ip camera 在手机端建立服务器将手机摄像头的内容以图片流的形式上传到服务器，电脑端通过访问服务器获取图像，实现实时跟踪小车视角，在电脑端通过 OpenCV 模块识别特定颜色的物体位置，进而实现跟踪功能；第二次实验的另一个功能是语音控制，通过使用端到端的语音识别程序实现语音控制小车的前进、后退、左转、右转、停车的状态转换。

二、本阶段内容

2.1 功能介绍

本阶段是对上次实验的改进，以及加入深度学习后的功能强化。首先小组对上次的 OpenCV 方法进行优化，通过减小视频传输流的分辨率以优化控制逻辑实现视频帧数的增加，使得物体识别更加平滑。第二个功能是物体识别与语音控制，这次实验将两种功能合并到一起，同时加入深度学习的部分。通过语音识别改变小车的寻找目标，然后通过物体识别部分识别物体在视角中的位置，然后通过调整行进到物体所在位置。

2.2 功能背景与意义

物体识别是机器智能的基本功能之一，它是任何一个以图像或视频作为输入的实际应用系统中的核心问题和关键技术。这类系统的性能和应用前景都依赖于其中物体的知识表示和分类识别所能达到的水平。物体识别技术无论是在军事还是在民用中都有着广泛需求和应用。如智能视频监控、视觉导航、人机交互、计算机取证、各类身份识别和认证系统、数字图书馆和 Internet 中的在海量图像库和视频中的基于内容的检索、编码与压缩等等。

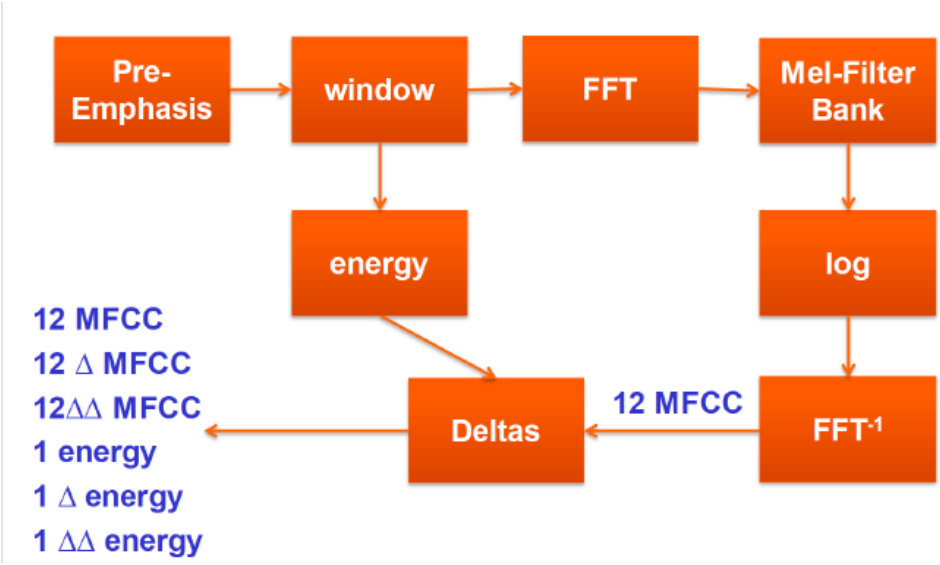
三、功能实现

3.1 语音控制扩展

在第二阶段中，我们构建了自己的简单语音识别系统，识别包括“左”，“右”，“前”，“后”，“停”五个语音命令。在本阶段中我们加入了“umbrella”，“laptop”，“backpack”，“bottle”，“apple”等物体的语音命令服务于我们的目标检测功能，考虑“物体”不像“前”、“后”、“左”、“右”这样可以用中文语音较为简单的控制，由于同一物体可能有多种叫法，且中文的语音较为复杂，于是对于“物体”的语音命令采用英文构建，构建和识别的主要过程为：

• 数据采集及特征提取

我们通过录制音频和开源数据中采集两种方式获得了用于输入大量的语音数据和其对应的文字标签 。对获得的原始语音信号进行处理（由于录制时在较为安静的环境中，因此不需进行降噪处理），对语音信号进行分帧（近似认为在10-30ms 内是语音信号是短时平稳的，将语音信号分割为一段一段进行分析）以及预加重（提升高频部分）等处理。用kaldi 提取出能够反映语音信号特征的关键参数MFCC 特征。提取MFCC 的计算过程如下：



按照这种方法对其中对每一段音频，提取出 39 维的向量。

• 模型训练

将整理好的 MFCC 特征和对应的类标签输入到构建的分类器中进行训练，利用交叉验

证和网格搜索调整分类器的超参数，获得构建好的语音识别模型。

在第二阶段的语音识别模型中，我们的模型存在着速度慢，识别精确不高的问题，在这一阶段中，我们从声学模型出发，提高系统性能，我们的主要策略有以下几个方面：

1） 增加训练数据。不同的训练数据也会对模型有一定的影响。通过调整采集数据的 channel 和数据的男女均衡性。

2） 采用比较好的模型训练方法。本阶段中采用了给予 EM 和 Baum-Welch 算法的最大似然估计 MLE(Maximum Likelihood Evaluation)方法，为期能得到较优的分类器。

3） 声学模型表示。采用 GMM-HMM 模型。

- **识别命令**

利用 python 中的 pyaudio 包调用电脑的麦克风，从麦克风中获取声音。每隔一定的秒数将获取到的声音文件实时保存到本地。在 linux 环境中调用 shell 脚本利用现有的工具对音频文件进行 MFCC 特征提取，将得到的特征结果结果以文件形式保存，然后程序从文件中读取数据，输入到语音识别模型中进行分类，得到对应的物体标签（umbrealla、backpack、laptop、bottle、apple）。

然而，我们仍没有解决第二阶段存在的问题：从麦克风中获得声音后进行的操作较多，速度慢，且经过改进的语音识别系统正确率仍然不够，无法满足控制小车的实时性要求，因此我们决定采用第二阶段所使用的现有模型向其中增加了“umbrella”、“backpack”、“laptop”、“bottle”、“apple”等语音命令对小车进行语音控制。

3.2 目标检测

目前常见的目标检测模型有以下几种：

Faster R-CNN《Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networkds》

YOLO 《You Only Look Once: Unified, Real-Time Object Detection》

SSD 《SSD: Single Shot MultiBox Detector》

我们的目标检测模型是基于 SSD 模型实现的，接下来介绍一下 SSD 模型。

SSD 算法是一种直接预测目标类别和 bounding box 的多目标检测算法。与 Faster R-CNN

相比，该算法没有生成 proposal 的过程，这就极大提高了检测速度。针对不同大小的目标检测，传统的做法是先将图像转换成不同大小（图像金字塔），然后分别检测，最后将结果综合起来（NMS）。

而 SSD 算法则利用不同卷积层的 feature map 进行综合也能达到同样的效果。算法的主网络结构是 VGG16，将最后两个全连接层改成卷积层，并随后增加了 4 个卷积层来构造网络结构。对其中 5 种不同的卷积层的输出（feature map）分别用两个不同的 3×3 的卷积核进行卷积，一个输出分类用的 confidence，每个 default box 生成 21 个类别 confidence；一个输出回归用的 localization，每个 default box 生成 4 个坐标值（x, y, w, h）。

此外，这 5 个 feature map 还经过 Prior Box 层生成 prior box（生成的是坐标）。上述 5 个 feature map 中每一层的 default box 的数量是给定的(8732 个)。最后将前面三个计算结果分别合并然后传给 loss 层。

本模型的核心之一是同时采用 lower 和 upper 的 feature map 做检测。

如图 Fig 1 所示，这里假定有 8×8 和 4×4 两种不同的 feature map。第一个概念是 feature map cell，feature map cell 是指 feature map 中每一个小格子，如图中分别有 64 和 16 个 cell。另外有一个概念：default box，是指在 feature map 的每个小格(cell)上都有一系列固定大小的 box，如下图有 4 个（下图中的虚线框，仔细看格子的中间有比格子还小的一个 box）。

假设每个 feature map cell 有 k 个 default box，那么对于每个 default box 都需要预测 c 个类别 score 和 4 个 offset，那么如果一个 feature map 的大小是 m*n，也就是有 m*n 个 feature map cell，那么这个 feature map 就一共有（c+4）*k * m*n 个输出。这些输出个数的含义是：采用 3×3 的卷积核对该层的 feature map 卷积时卷积核的个数，包含两部分（实际 code 是分别用不同数量的 3*3 卷积核对该层 feature map 进行卷积）：数量 c*k*m*n 是 confidence 输出，表示每个 default box 的 confidence，也就是类别的概率；数量 4*k*m*n 是 localization 输出，表示每个 default box 回归后的坐标）。训练中还有一个东西：prior box，是指实际中选择的 default box（每一个 feature map cell 不是 k 个 default box 都取）。

也就是说 default box 是一种概念，prior box 则是实际的选取。训练中一张完整的图片送进网络获得各个 feature map，对于正样本训练来说，需要先将 prior box 与 ground truth box 做匹配，匹配成功说明这个 prior box 所包含的是个目标，但离完整目标的 ground truth box 还有段距离，训练的目的是保证 default box 的分类 confidence 的同时将 prior box 尽可能回归到 ground truth box。

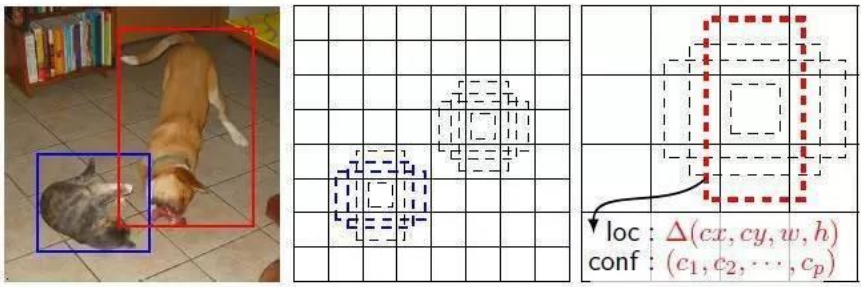


Fig. 1: **SSD framework.** (a) SSD only needs an input image and ground truth boxes for each object during training. In a convolutional fashion, we evaluate a small set (e.g. 4) of default boxes of different aspect ratios at each location in several feature maps with different scales (e.g. 8×8 and 4×4 in (b) and (c)). For each default box, we predict both the shape offsets and the confidences for all object categories ($(c_1, c_2, \cdots, c_p)$). At training time, we first match these default boxes to the ground truth boxes. For example, we have matched two default boxes with the cat and one with the dog, which are treated as positives and the rest as negatives. The model loss is a weighted sum between localization loss (e.g. Smooth L1 [6]) and confidence loss (e.g. Softmax).

四、实现效果

（1）小车能够以更快的速度和精度追踪红色电烙铁，随着电烙铁的移动而随之移动。

（2）小车能够识别雨伞、笔记本电脑等物品，当物品出现在小车视野范围内时，小车会朝其移动，直至到达物品所在位置。

（3）结合语音功能，小车能够跟随语音指令连续识别多个物体，并朝物体所在位置移动。

五、问题及解决方案

在实验过程中，小组遇到以下问题：

1. 小车速度控制。小车速度过快，则延迟较大，使得当识别指令传输到小车的时候物体已经不在小车的视角中，小车处于找到物体的状态时间很短，无法有效的完成物体识别的功能；如果小车速度过慢，由于地面及电机阻力较大，小车无法启动。
2. 小车马达接触不良，左轮经常失控
3. 小车动力不足，从右转到直行状态转换经常卡住

为解决第一个问题，小组使用了双向延时的方法，具体实现方法为：最初的小车行进的策略是状态化的，即小车接收到特定指令后会持续的保持小车的运动状态——直行、左转、右转等，如果后续的指令因为延迟或连接断开等原因没有被小车接收，则小车会持续直行上

一次的指令。在新的策略中，小车接收到某一条指令后，只会执行 t_1 时间，之后会停止等待下一条指令，在电脑端每发送一条指令，会延迟 t_2 时间，考虑到指令的发送延迟以及小车执行的延迟，将时间设定为 $t_1<t_2$。最终的实现效果为小车会间隔前进，最终达到指定物体的位置。

剩下的两个问题，由于是硬件问题，无法在软件层面进行优化，当小车无法前进的时候，通过外力推动小车，给小车初速度继续前进。当小车因为左右速度配比不均导致偏离前进路线的时候，小车会无法找到目标物体而进入无法识别的状态，会在原地转圈寻找待测物体，当重新找到的时候会继续前进。