# 上手效果

需要准备两个可执行程序，分别为

服务端：卡牌识别服务（card\_detcet）

客户端：桌面控制程序（card222222）

先运行桌面控制程序，然后你就可以看到开机logo（图片），然后桌面就会启动起来。桌面框架使用的LVGL轻量级嵌入式显示框架。这里使用了韦东山老师的LVGL桌面显示的框架。由于LVGL框架的特殊性，桌面的分辨率不能随意调节，均是由图片组成，所以大家看到的桌面效果就会由这个大白边。改进方法在后面教程中会附上。

桌面中有很多APP的图标，同时还有时间等状态栏的显示。其他APP的效果100ASK视频中有所展示，这里就不过多介绍，直接启动我们的APP（calc），可以看到初始化的过程，此时会停止在41%，因为客户端需要和服务端连接，如果此时服务端启动，双方连接成功，则会进入APP界面（图片）。APP顶部是APP名称，又或者是状态栏。

正中间为卡牌识别到的卡牌图片，屏幕上有两个按键和两个箭头QWQ，箭头可以调整卡牌属性，按键用来拍照+识别。因为性能有限无法做到实时识别，所以采用手动识别的方式。

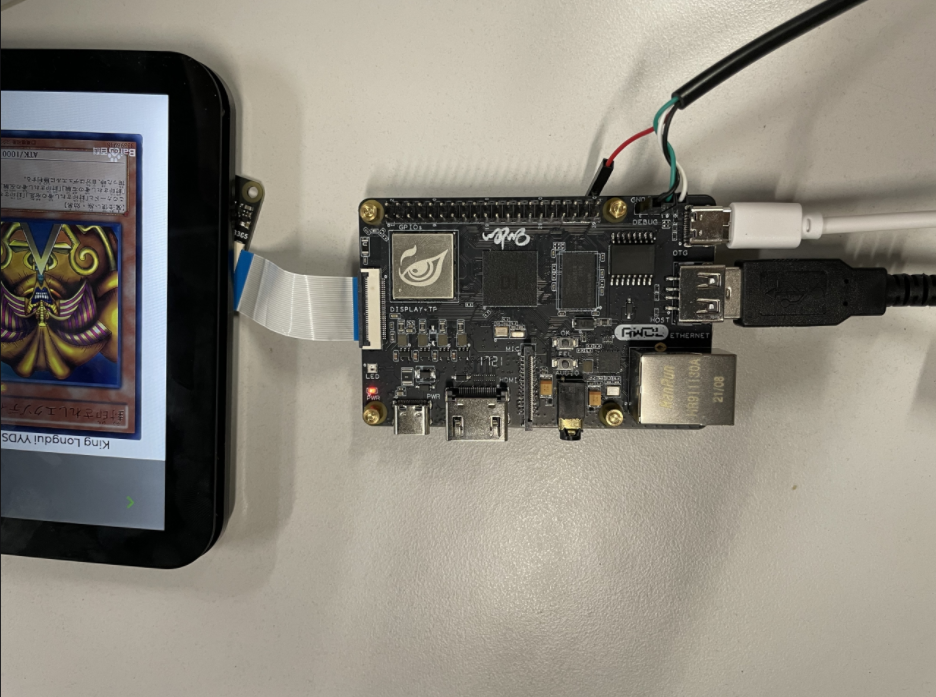
DETECT按键可以实现拍照并保存。FIND按键可以实现对拍照的识别，识别成功后会将卡牌的图片显示在屏幕上。基本的演示效果就是这些。

视频播放说明，视频的播放可以使用tpayerdemo来进行播放。其函数的接口文档会一并上传至github。

# 硬件准备

* 哪吒开发板一块
* USB免驱摄像头
* MIPI显示屏

其连接如图：



# 代码获取

Github链接：[jie2311260561/Collector: Card\_detect and show animation (github.com)](https://github.com/jie2311260561/Collector)

ZIP包：（百度网盘）

# 编译及运行

其中可执行文件为：card22222

carddetect

下载来可执行文件装载上SD卡，在SD卡中装入模型。就可以直接运行。

## ncnn编译

参考ncnn作者nihui的知乎来进行编译。<https://github.com/Tencent/ncnn/wiki/how-to-build#build-for-linux>。编译后得到ncnn的静态库，记住当前的路径一会要用。

## demo编译

服务端代码编译：

更改cardddetectdemo中的CMakeLists.txt文件的第20行：

目录更改为上面的ncnn编译后安装的目录。（一定要改啊啊！！！）然后执行指令：

export RISCV\_ROOT\_PATH=/home/nihui/osd/riscv64-linux-x86\_64-20210512

cmake -DCMAKE\_TOOLCHAIN\_FILE=../c906.toolchain.cmake ..

make –j4

编译后可以在，build目录下看到carddetect可执行程序。

客户端代码编译：

在编译客户端代码之前一定要确认自己的tina固件编译完成，且找到编译的out 目录，更改变量为相应的变量目录

LIB\_LINK\_NAPI = /home/gaojies/workspace/d1-tina-open/out/

WORKSPACE = /home/gaojies/workspace/d1-tina-open/

make –j16

编译完成后可以在output 目录看到card222222可执行程序

两个都编译好后就可以推向小机端开始跑了，不过单有这两个可执行程序可不能跑，还需要“模型”文件。

/mnt/SDCARD/yolo\_model/best-opt.param

/mnt/SDCARD/yolo\_model/best-opt.bin

将两个模型文件放到这个目录，就可以开始跑了。

先运行服务端，再运行客户端。效果如1.上手效果所示。

./card22222

./carddetect 2 #2这个参数表示从socket 接收参数识别

# 当参数为 1 时，其后面可以跟 图片路径

# 开发过程及修改教程

## 项目初衷

本项目是我来到全志科技实习期间，使用全志科技开D1开发板所做的具有可扩展性的项目，首先是能够熟练使用D1平台，然后就是能够在D1已有工具的基础上，将所有的可扩展demo有机的结合起来构建成一个综合的使用例程。D1开发板上有着众多外设以供使用，可以通过外接构建出属于自己的“智能”应用产品。同时D1又是为数不多的RISCV架构的可以运行Linux系统的SOC。不仅可以用于个人DIY,同时也能用于产品制作。D1芯片其性能算不上强，但是其外设可谓十分丰富。整个开发过程我将其分为三个阶段，分别为外设模块的demo开发验证，神经网络模型训练推理验证，应用APP开发。我接下来将从下面三个阶段分别展示如何开发以及开发思路。

## 验证demo及外设

Demo的验证过程有摄像头验证、按键验证、网络连接验证、MIPI屏幕验证。其中摄像头验证的过程中，首先尝试使用Opencv来进行摄像头画面采集，故移植了Opencv，相关的帖子见链接：

移植好的Opencv因为libjpeg库不能过编，故将该库舍弃。成功过编后，在板子上验证了一个demo，可以保存一张png图片，jpg经过测试不可以。不知道是出于什么原因，Opencv打不开系统的摄像头。总会返回打开失败的标志。

然后学习了V4L2框架，使用V4L2打开进行摄像头画面的捕捉，可以成功进行，岁有了这个摄像头的demo。

本来我制作数据集的思路本来是获取摄像头数据—拍视频—ffmeg进行帧提取制作数据集。其中视频数据需要编码，我对硬件的编码目前还不是很了解，所以我就使用了比较方便的方法，我直接编译一个ffmpeg上去，ffmpeg没有进行任何设置，所以使用的是软件编码，软件编码的效率仅仅只有5帧（某些同学的流畅级别？？？一帧能玩，两帧不卡，三帧流畅？）由于效率实在是比较低，最终我还是放弃了使用视频来制作数据集的方法。选择了使用按键来一张or几张的拍摄方法。

其中关于我移植好的Opencv和FFmpeg（虽然只有软件的方法）我都会放在Github，有想法的同学，或者可以改良的同学可以自取。

其中Demo验证的代码再github开源。

### 摄像头demo

摄像头demo为文档中的demo示例，参考文档即可。

相较于文档，有一点需要意，这也是在app开发过程中的所遇到的bug，V4L2框架中的摄像头视频流buffer需要不间断的进行更新，不能直接卡在原地。否则就会出现拍的照片为上几秒钟的照片。在本demo中更新buffer的方法为反复调用v4l2Grab函数。

### 按键demo

按键demo的制作为中作模型训练数据集使用，使用按键来进行图片的拍摄。

其主要思路为打开系统的imput\enent0 节点，然后判断当前节点的数据来进行按键状态的判断。

## 自己的模型训练及部署

自己训练模型的方法有很多种，这里使用的是网络上常用的目标检测框架Yolo框架，由于目前分类网络还是比较复杂，不适合在嵌入式设备上跑，就假装使用这个目标检测框架来实现效果。

最终拍好的照片分类使用数据标注工具进行标注，制作成为训练数据集，在数据标注的过程中，其中包括了将图片重新编号（使用python脚本做批量处理），标注好之后使用两个Python脚本对标注好的标签进行批处理转换成可以被Yolov5识别的txt标签数据集。

Yolov5在下载的时候一定要选择Relase版本的下载，否则在后面对模型的操作中就会出现问题。（相信问题可以看论坛记录）

主要推理部分还是参考nihui的方法，手动实现输入节点。修改网络模型（反正就是很骚的操作，最终能够实现模型的推理）

如果需要自己使用yolov5训练模型可以参考这个文章：<https://www.pythonf.cn/read/161104>

其中我个人修改的脚本会上传至github。

针对模型转换的问题，可以看<https://zhuanlan.zhihu.com/p/275989233>

推理代码：<https://github.com/Tencent/ncnn/blob/master/examples/yolov5.cpp>

参考：https://github.com/nihui/ncnn-android-yolov5

## 基于此桌面的APP开发

首先我使用的是LVGL 7.11，针对原本韦老师在这里我使用了G2D来进行图像数据传输，具体实现方法为：

添加相关头文件路径，移植sunxifb.c来替换fbdev 在此文件中使能了G2D的图像传输。应用层的代码不用改，可以看到界面刷新效率额显著提升，当然我也会在后面提供没有使用G2D的可执行文件。

进入桌面后的每个app都会分配一个回调函数，添加入口：

imx6ull-lvgl-master\lv\_100ask\src\lv\_100ask\_demo\lv\_100ask\_demo.c

192~240行：若添加APP图片，先声明转换后数组。然后将其添加在app\_icon数组中，其回调函数也添加在lv\_100ask\_event\_cb数组中。

然后在866行开始添加自己的APP回调。

进入回掉函数后，关闭主程序使能，进入分支程序入口：

in\_desktop\_page = false;

分支程序就可按照标准框架增加控件进行APP开发。

APP开发参考LVGL手册，绘制控件及其标准。https://docs.lvgl.io/latest

## 整体总结

卡牌识别机初衷基本上已经完成了，可以通过在哪吒上部署神经网络算法，能够使用摄像头采集图像进行识别，并将识别结果显示在界面上。从问题的提出，到我解决方案的查阅，到我针对不同问题的解决方法，到中间发现实现方案的缺陷，并针对问题寻找解决方法。从在github 上提问作者，到在论坛上和坛友们讨论，再到直接去找部门师兄相关方向的大牛去咨询，去了解，自己也学会了解遇到的不同问题。

顺带提一嘴，找到了LVGL中显示摄像头画面的程序：<https://github.com/Hehesheng/k210/blob/93030e1f4f11fae9a9fbfa9ef99edce6f2bc6f98/applications/gui/lvgl_camera.c>

但是自行实现会报，类型错误的问题。