# 信息科学与工程 学院课程设计成绩评价表

课程名称：嵌入式开发综合课程设计

设计题目：嵌入式系统 OpenCV 光学 OCT 成像的研究

专业： 物联网 班级： 1603 姓名： 郭治洪 学号：201616070320

| 序号 | 评审项目 | 分 数 | 满分标准说明 |
|---|---|---|---|
| 1 | 内 容 | | 10 分：思路清晰，语言表达准确，概念清楚，论点正确；设计方法科学，分析归纳合理；结论严谨，设计有应用价值。任务饱满，工作量适中 |
| 2 | 创 新 | | 10 分：内容新颖，设计能反映新技术，对前人工作有改进或突破，或有独特见解 |
| 3 | 完整性、实用性 | | 10 分：整体构思后合理，理论依据充分，设计完整，实用性强 |
| 4 | 数据准确、可靠 | | 10 分：数据准确，算法设计合理 |
| 5 | 规 范 性 | | 20 分：设计格式、绘图、实验数据、标准的运用等符合有关标准和规定 |
| 6 | 纪 律 性 | | 20 分：遵守课程设计纪律，听从指导教师安排，设计过程态度认真 |
| 7 | 答 辩 | | 20 分：准备充分，思路清晰、论点正确、对设计方案理解深入，问题回答有理有据，简明正确 |
| 总分 | | | |
| 综合意见 | 指导教师 2019 年 1 月 12 日 | | |

河南工業大學

# 课　程　设　计

课程设计名称：嵌入式开发综合课程设计

专 业 班 级 ：物联网 1603

学 生 姓 名 ：郭治洪

学　　号 ：201616070320

指 导 教 师 ：麦欢欢

课程设计时间：2019.1.7-2019.1.12

# 物联网工程 专业课程设计任务书

| | | | | | |
|---|---|---|---|---|---|
| **学生姓名** | 郭治洪 | **专业班级** | **物联网 1603** | **学号** | 201616070320 |
| **题　目** | 嵌入式系统 OpenCV 光学 OCT 成像的研究 | | | | |
| **课题性质** | **A.工程设计** | | **课题来源** | 老师制定 | |
| **指导教师** | 麦欢欢 | | **同组姓名** | - | |
| **主要内容** | 本次课程设计，在嵌入式系统（Raspberry Pi，内核 ARMv7+）安装 OpenCV 依赖库环境，使用 CodeBlocks 调用 gcc/g++ 在系统中编译出由光学 OCT 扫描数据成像的程序，并且改进成像速度，调用 GPU 加速，改进计算算法，又采用 Wifi 连接两台设备进行传输，连接手机进行传输绘制。 | | | | |
| **任务要求** | Phase 0.编译环境的配置，CodeBlocks，OpenCV 环境的安装，编译链接的设置。<br><br>Phase 1.调用 OpenCV 绘制由光学 OCT 扫描数据的图像，并将图像翻转。<br><br>Phase 2.调用 GPU-FFT 库，使用 GPU 加速成像。<br><br>Phase 3.改进成像算法，加速程序运行<br><br>Phase 4.使用两台设备通过 Wifi 传输成像好的图片，并且显示。<br><br>Phase 5.使用手机连接 Wifi 接收成像好的图片，并且显示。 | | | | |
| **参考文献** | [1] 嵌入式开发综合课程设计指导书，自编，2019。<br>[2]物联网开发平台（ FS-11C14 开发板）用户手册-V4.0，清华远见，北京华清远见研发中心，2012。 | | | | |
| **审查意见** | **指导教师签字：**<br><br>**教研室主任签字：** **2019 年 1 月 6 日** | | | | |

## 1. 需求分析

本次课程设计，在嵌入式系统（Raspberry Pi，内核 ARMv7+）安装 OpenCV 依赖库环境，使用 CodeBlocks 调用 gcc/g++ 在系统中编译出由光学 OCT 扫描数据成像的程序，并且改进成像速度，调用 GPU 加速，改进计算算法，又采用 Wifi 连接两台设备进行传输，连接手机进行传输绘制。

## 2. 概要设计

Phase 0.编译环境的配置，CodeBlocks，OpenCV 环境的安装，编译链接的设置。
Phase 1.调用 OpenCV 绘制由光学 OCT 扫描数据的图像，并将图像翻转。
Phase 2.调用 GPU-FFT 库，使用 GPU 加速成像。
Phase 3.改进成像算法，加速程序运行
Phase 4.使用两台设备通过 Wifi 传输成像好的图片，并且显示。
Phase 5.使用手机连接 Wifi 接收成像好的图片，并且显示。

## 3. 运行环境

Model:Raspberry Pi 3B+
Kernel:4.14.79-v7+
OS:Raspbian Stretch

## 4. 开发工具和编程语言

Compiler:GNU GCC 6.3.0/GNU G++ 6.3.0
IDE:CodeBlock17.02
Library:OpenCV 3.2
Software:hostapd,dnsmasq

## 5. 详细设计

Phase 0.编译环境的配置，CodeBlocks，OpenCV 环境的安装，编译链接的设置。

```
$ sudo apt-get install vim nano gedit gcc g++ codeblocks opencv
```

root 模式打开 CodeBlocks，设置链接选项 Settings-Compiler-links

```
`pkg-config --cflags --libs opencv` -ldl
```

设置其它编译选项 other-flags

```
`pkg-config --cflags --libs opencv`
```

执行测试程序 OpenCV_practice

```
$ cd /home/pi/OpenCV_practice
$ sudo chmod a+x build.sh
$ ./build.sh
```

成功的话可以看出人脸识别，和图形拼接显示

Phase 1. 调用 OpenCV 绘制由光学 OCT 扫描数据的图像，并将图像翻转。

成像部分核心代码：

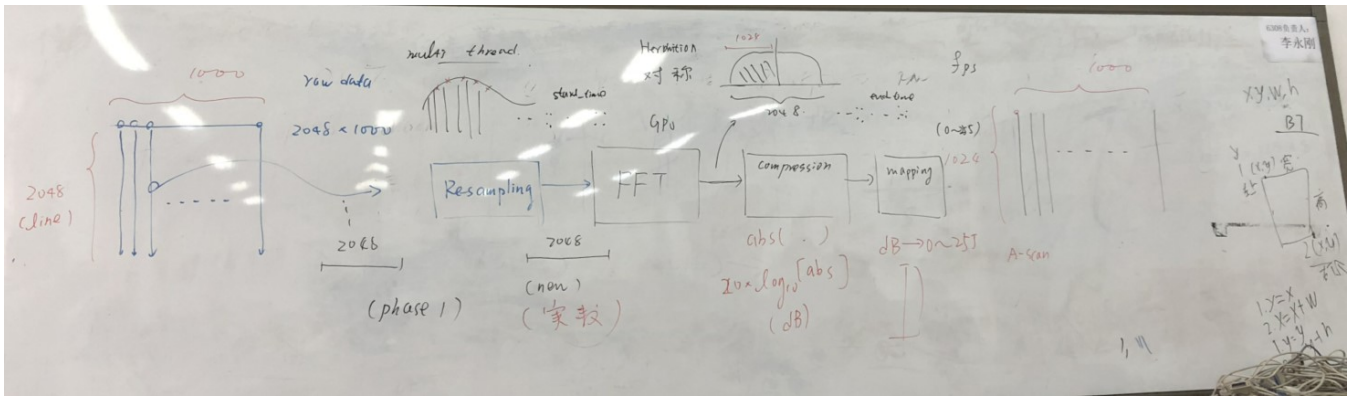
```
Mat img, img2;

float img_show[1000][1024];

img2.create(1024, 1000, CV_8U);
int numImgSize = 0;
fstream fin2;

for (int i = 0; i < img2.cols; i++) {
    for (int j = 0; j < img2.rows; j++, numImgSize++) {
        img2.at<uchar>(i, j) = Img[numImgSize];

    } // for
} //for

namedWindow("Image2", WINDOW_AUTOSIZE);
namedWindow("Image", WINDOW_AUTOSIZE);
resize(img2,img, Size(512,500),(0.0),(0.0), INTER_LINEAR);
imshow("Image", img);
```

执行测试程序 OCT_CPU

```
$ cd /home/pi/OCT_CPU/bin/Release/OCT_CPU
$ sudo chmod a+x OCT_CPU
$ ./OCT_CPU
```

Phase 2. 调用 GPU-FFT 库，使用 GPU 加速成像。

进行实验分析，查阅相关资料。




调用自定义的 FFT 函数进行快速转换 FFT(D2_Resamp,im,1000);

核心代码：

```
//myFFT.C
#include <stdlib.h>
#include <unistd.h>
#include <stdio.h>
#include <math.h>
#include <time.h>
#include "mailbox.h"
#include "gpu_fft.h"

void FFT( float re[][1024], float im[][1024], int jobs) {

  int ret, loops, freq, log2_N, N, mb = mbox_open() ;
  //ret:catch return value of GPU_prepare, log2_N:the Length of fft ;  jobs:e.g 1024 258

  log2_N = 10 ; // 1024

 //loops = 1 ;

  unsigned t[2];
  double tsq[2];

  struct GPU_FFT_COMPLEX *base;
  struct GPU_FFT *fft;

  ret = gpu_fft_prepare(mb,log2_N,GPU_FFT_REV,jobs,&fft); // call once

  N = 1 << log2_N; // FFT length

    int j = 0 ;
     while( j < jobs ) {
      base = fft->in + j*fft->step; // input buffer
      int i = 0 ;
      while( i < N )  {
        base[i].re = re[j][i] ;
        base[i].im = im[j][i];
        i++ ;
      }

      j++;
    }

   usleep(1) ;
  gpu_fft_execute(fft); // call one or many times
```

```
  j = 0 ;
  while( j < jobs ) {
    base = fft->out + j*fft->step; // output buffer
    freq = j + 1 ;

    int i =  0 ;
    while(  i < N ) {
      re[j][i] = base[i].re ;
      im[j][i] = base[i].im;
      i++ ;
    }

    j++;
  }

    gpu_fft_release(fft); // Videocore memory lost if not freed !
}
```

执行测试程序 OCT_GPU

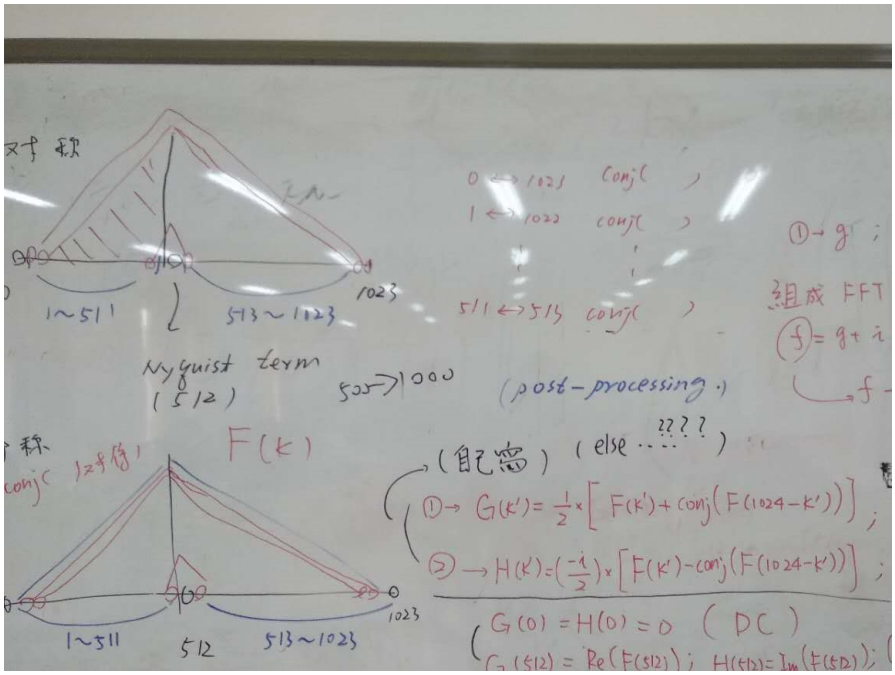
```
$ cd /home/pi/OCT_CPU/bin/Release/OCT_GPU
```

```
$ sudo chmod a+x OCT_GPU
```

```
$ ./OCT_GPU
```

Phase 3.改进成像算法，加速程序运行

进行建模分析




核心代码：

```
int line = 0 ;
    for ( int k = 0 ;   k < 500 ; k++ ) {
       for(int j = 0 , j2 = 1023 ;  j < 512 ; j ++) {
         if(j == 0 )  { // DC
             D4_absFFT[line][j] = 0  ;
             D4_absFFT[line+1][j] = 0  ;
         } // if

         else if( j == 512 ) {  //Nyqust
             D4_absFFT[line][j] = D2_Resamp[k][j] ;
             D4_absFFT[line+1][j] = im[k][j] ;
             j2-- ;
         } // else if

         else {
                double
temp1=0.5*sqrt((D2_Resamp[k][j]+D2_Resamp[k][1024-j])*(D2_Resamp[k][j]+D2_Resamp[k]
[1024-j])+(im[k][j]-im[k][1024-j])*(im[k][j]-im[k][1024-j]));
                D4_absFFT[line][j] = temp1;
                D4_absFFT[line+1][j] = -temp1;
                double
temp2=0.5*sqrt((D2_Resamp[k][j]+D2_Resamp[k][j])*(D2_Resamp[k][j]+D2_Resamp[k][j])+
(im[k][j]-im[k][j])*(im[k][j]-im[k][j]));
                D4_absFFT[line][j+512] = temp2;
                D4_absFFT[line+1][j+512] = -temp2;
            }
       } // for

       line+=2 ;
} // for
```

Phase 4.使用两台设备通过 Wifi 传输成像好的图片，并且显示。

手工建立 WIFI AP:

```
$ sudo apt-get update
$ sudo apt-get upgrade
$ sudo apt-get install dnsmasq hostapd
$ sudo nano /etc/dhcpcd.conf
```

```
# set virtual network card static IP
interface uap0
    static ip_address=192.168.4.1/24
```

```
nohook wpa_supplicant
```

```
$ mv /etc/dnsmasq.conf /etc/dnsmasq.conf.orig
```

```
$ sudo nano /etc/dnsmasq.conf
```

```
# set set virtual network card dhcp server info
interface=uap0
dhcp-range=192.168.4.2,192.168.4.20,255.255.255.0,24h
```

```
$ sudo nano /etc/hostapd/hostapd.conf
```

```
interface=uap0
driver=nl80211
ssid=RPi3-AP                    // SSID NAME
hw_mode=g
channel=7
wmm_enabled=0
macaddr_acl=0
auth_algs=1
ignore_broadcast_ssid=0
wpa=2
wpa_passphrase=123456789      // password
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP
```

```
$ sudo sed -i 's/#DAEMON_CONF=""/DAEMON_CONF="\/etc\/hostapd\/hostapd.conf"/g'
/etc/default/hostapd
```

```
$ sudo service dhcpcd restart
```

```
$ sudo reboot
```

```
$ sudo iw dev wlan0 interface add uap0 type __ap
```

```
$ sudo ifconfig uap0 up
```

```
$ service dnsmasq start
```

```
$ sudo hostapd /etc/hostapd/hostapd.conf
```

然后开启服务端

```
$ cd /home/pi/Phase4Server/bin/Release
$ sudo chmod a+x OCT_GPU_1024
$ ./OCT_GPU_1024
```

选择 I 进行监听，同时在客户端运行以下指令接收，设置服务器 IP：192.168.4.1

```
$ cd /home/pi/Phase4Client/APClientTset/bin/Release
$ sudo chmod a+x APClientTset
$ ./APClientTset
```

然后即可接收图像。

Phase 5.使用手机连接 Wifi 接收成像好的图片，并且显示。

设置热点见上部分

安卓手机安装程序后连接 Wifi 热点，打开程序设置热点信息：

IP:192.168.1.4

Port:9999

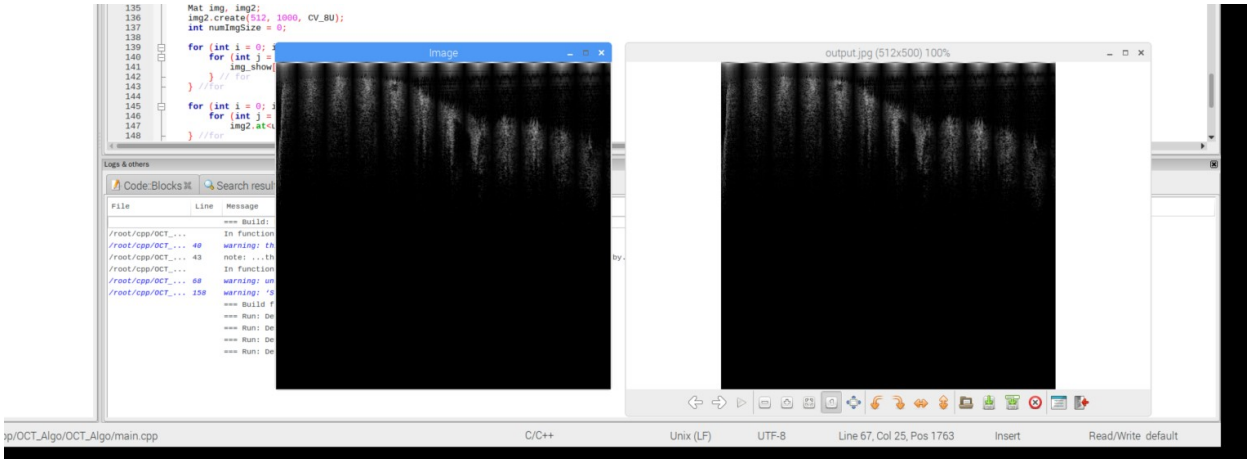
```
$ cd /home/pi/Phase5Server/bin/Release
$ sudo chmod a+x OCT_GPU_1024
$ ./OCT_GPU_1024
```

然后即可接收到 50 张图像

## 6. 调试分析

请见详细分析。

## 7. 测试结果



## 8. 心得体会

感谢老师对我们这次课设的准备和认真讲解！

虽然这次课设时间很短，但是收获很多，也学了很多，自己动手查资料，自己推公式，自己debug 程序，自己不断测试摸索正确的方向，最后终于成功了！

老师尤其给我特别大帮助和支持，这里要再次感谢他！

不过，估计折腾太猛了，实验最后一点时候，我的树莓派已经熟了（哭）。

最后在寇涵同学设备下成功完成了所有实验，这里也感谢他。

还是觉得时间太短不过瘾啊，我一定会加油更深入的!!!

## 参考文献


【1】 Pi 3 as wiireless client and wireless AP?
https://www.raspberrypi.org/forums/viewtopic.php?t=138730

【2】 Setting up a Raspberry Pi as an access point in a standalone network (NAT)
https://www.raspberrypi.org/documentation/configuration/wireless/access-point.md

【3】 Accelerating Fourier transforms using the GPU
https://www.raspberrypi.org/blog/accelerating-fourier-transforms-using-the-gpu/

【4】 FFT 节省资源的思路
https://www.cnblogs.com/xingshansi/p/6298391.html

【5】 GPU FFT 源码
https://github.com/adafruit/rpi-firmware/tree/8f305cad54cf725a14551b59a2e33c6fd37d7492/vc/sdk/opt/vc/src/hello_pi/hello_fft

【6】 INSTALL AND CONFIGURE OPENCV IN CODE BLOCKS
https://importgeek.wordpress.com/2016/08/27/install-and-configure-opencv-in-code-blocks/

【7】 以及老师和学长给的参考程序