厦門大學

信息学院软件工程系

《计算机网络》实验报告

题　　目 实验一　庾晓萍

班　　级 软件工程 2020 级卓越班

姓　　名 庾晓萍

学　　号 20420192201952

实验时间 2022 年 3 月 2 日

2022 年 3 月 2 日

填写说明

1、本文件为 Word 模板文件，建议使用 Microsoft Word 2019 打开，在可填写的区域中如实填写；

2、填表时勿破坏排版，勿修改字体字号，打印成 PDF 文件提交；

3、文件总大小尽量控制在 1MB 以下，最大勿超过 5MB；

4、应将材料清单上传在代码托管平台上；

5、在实验课结束 14 天内，按原文件发送至课程 FTP 指定位置。

1 实验目的

通过完成实验，掌握物理层传输的原理；了解传输过程中的编解码、噪声、分辨率、波特率、调制和误码等通信概念；理解奈氏定理和香农定理。

2 实验环境

操作系统：Windows 10；

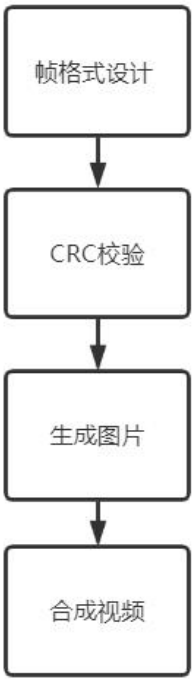
IDE：Visual Studio 2019

编程语言：C++；

3 实验结果

一、编码

0、主要流程



1、帧格式设计

（1）图片采用彩色编码，通过红、黄、蓝、绿四种颜色表示比特信息。（黄表示 00，红色表示 01……）。

（2）二维码包含四个定位矩形，其他区域为数据区域。每张二维码包含 5184（72^2）个像素格，其中 576（$4*12^2$）格被定位矩形占据。也就是说数据区的像素格为 4608 格，每格传递 2bits 信息，其中有 32bits 是最后加上的 0。故每张二维码的信息为：2*4608-32=9184bits。

（3）默认帧率为 10 帧/秒，可以在 Main.cpp 中更改。

2、CRC 效验核心代码

使用模二除法，算法类似高精度除法，但是不进位，每步的运算是异或。

```
//CRC为除数，原始数据data为被除数，模二除法
string CalculateCRC(string data, string& CRC) {
    data.append(string(CRC.size(), '0'));//在原始数据data后面加上校验码位数（k）个0
    //模2除法
    for (int i = 0; i < data.size() - CRC.size(); i++) {
        //如果是0可以跳过，0与其他数异或还是其他数
        if (data[i] == '1') {
            for (int j = 0; j < CRC.size(); j++) {
                //异或运算，相同0，不同1
                (data[i + j] == CRC[j]) ? data[i + j] = '0' : data[i + j] = '1';
            }
        }
    }
    return data.substr(data.size() - CRC.size(), CRC.size());//返回余数(后k位)
}
```

3、图片生成核心代码

使用 opencv 的 rectangle 函数涂色。每两个比特信息填成一种颜色。

```
void Help_Draw(string code, int& R, int& G, int& B, int& key) {
    if (code[key] == '0' && code[key + 1] == '0') { R = 255; G = 255; B = 0; }//黄色:00
    if (code[key] == '0' && code[key + 1] == '1') { R = 255; G = 0; B = 0; }//红色:01
    if (code[key] == '1' && code[key + 1] == '0') { R = 0; G = 255; B = 0; }//绿色:10
    if (code[key] == '1' && code[key + 1] == '1') { R = 0; G = 0; B = 255; }//蓝色:11
    key = key + 2;//跳两个比特，继续往下读取code数组
}
```

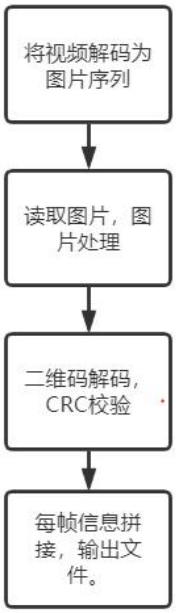
4、将生成的图片序列合成为视频

使用 opencv 的 VideoWriter 函数，将视频保存在 videoname 路径。

```
int frame_rate = 10;//视频帧率默认为10
cout << "Video frame rate defaults to 10" << endl;
VideoWriter video(videoname, CAP_ANY, frame_rate, Size(780, 780));//将视频保存成videoname
for (size_t i = 0; i < img.size(); i++)//i:long long unsigned int
{
    Mat image = img[i].clone();
    video << image;// 流操作符，把图片传入视频
}
```

二、解码

0、主要流程



1、将视频解码为图片序列

使用 opencv 的 VideoCapture 函数，将得到的图片序列储存在 srcImages 中。

```
void Read_Video(string videopath, vector<Mat>& srcImages) {
    VideoCapture capture(videopath);//使用cv2库函数读取视频,储存在srcImages中
    Mat frame;

    while (1) {//读取视频帧
        capture >> frame;
        if (frame.empty()) break;//停止读取
        else {
            resize(frame, frame, Size(720, 720), 0, 0, INTER_NEAREST);//调整大小（使用最近邻插值）
            srcImages.push_back(frame.clone());//将该帧克隆到srcImages中
            frame.release();//释放视频流
        }
    }
}
```

2、读取图片，图片处理

先进行简单处理，去除部分噪声（彩色转灰度、高斯滤波、二值化。）

```
string Code_Translate(Mat& srcImage, ofstream& verify) {
    Mat midImage, dstImage;//中间生成图像，最终图像
    string code = "";

    //处理原始图片
    midImage = Handle_Img(srcImage);//简单初步处理，去噪
    Get_ROI(midImage, srcImage, dstImage);//提取图片中的二维码存入dstImage中
    code = Decode(dstImage, verify);//将图片解码
    return code;//返回图片编码
}
```

3、二维码解码，CRC 校验

将识别到的颜色转化为对应的比特编码。

```
void Help_Decode(Scalar& color, string& code) {
    if (color[1] > 100 && color[2]>100 && color[0]<1.5*color[1]&& color[0] < 1.5 * color[2]) {
        code += "00";//黄色
    }
    else if (color[0] < color[2] / 1.5 && color[1] < color[2] / 1.5) code += "01";//红色
    else if (color[0] < color[1] / 1.5 && color[2] < color[1] / 1.5) code += "10";//绿色
    else if (color[1] < color[0] / 1.5 && color[2] < color[0] / 1.5) code += "11";//蓝色
}
```

4、将每帧信息拼接，输出二进制文件与校验文件。

```
string temp;
for (i = 0; i < srcImages.size(); i++) {
    temp = Code_Translate(srcImages[i], verify);//将图片转换为代码
    if (temp.empty())  continue;//第一次读时，如果当该张图片读空，读取下一张
    else {
        code_array.push_back(temp);//将读到的编码存入编码数组中
        break;//跳出循环
    }
}
```

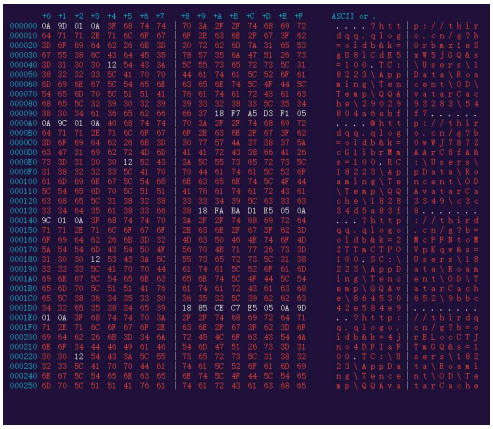
三、实验结果

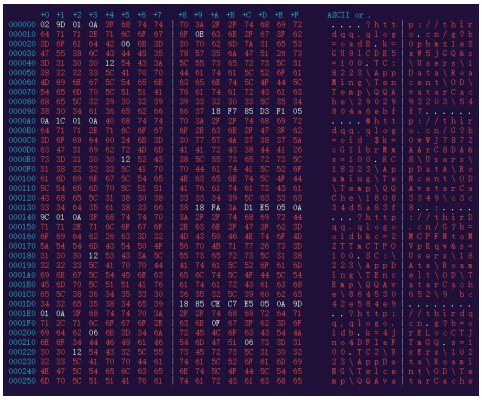
（1）传输速度

实验中，彩色二维码的分辨率是 5184（72^2）个像素方块。帧率 10 帧/秒，每张二维码包含 9184 bits 信息。故波特率以及传输速率为 $10*9184*10^{-3}$=91.94Kb/s。

（2）准确度

下面第一幅图为输入的二进制文件，第二幅图为输出的二进制文件，信息大致可以完整传输，但是仍然有传输错误的部分。





（3）输出文件说明

在附加文件包“EA1-1952-庾晓萍”中，包含了附加的文件。其中 input.bin 是输入的二进制文件，output.mp4 是输出的彩色二维码视频文件，byphone.mp4 是手机拍摄下的视频，output.bin 是由手机拍摄的视频解码得到的二进制文件，verify.bin 是作为校验的二进制文件。

4 实验代码

本次实验的代码已上传于以下代码仓库：

https://github.com/ryanregal/ExpOne_ComputerNetwork

5 实验总结

一、实验课后问题

1、在实验中的，编解码算法是什么？

编码算法：核心部分时将字符转化为比特，具体如下图。先将文件内容读入字符数组 input_string 中，将字符转换为二进制储存在字符串 data 中。

```
for (int i = 0; i < length_char; i++) {
    Binary_Code((unsigned char)input_string[i]);//将字符转换为二进制
    for (int j = 0; j < LEN; j++) {//储存在data中
        if (binary_digit[j] == 1) data.append(1, '1');
        else if (binary_digit[j] == 0) data.append(1, '0');
    }
}
data += "\0";//结束符
cout << "Finish encode" << endl;
```

解码算法：解码算法先调用 Read_Video 函数读取视频，将图片储存在 srcImages 数组中。再调用 Code_Translate 函数将图片转换为代码。

```
string videopath(videoname);//字符串videopath取值videoname
Read_Video(videopath, srcImages);//读取视频，将图片储存在srcImages数组中

string outfile(outname);//字符串outfile取值outname
ofstream out(outfile, ios::binary);//二进制打开outfile写入out
ofstream verify(verifyname, ios::binary);//二进制打开verifyname写入verify

int i = 0;
string temp;
for (i = 0; i < srcImages.size(); i++) {
    temp = Code_Translate(srcImages[i], verify);//将图片转换为代码
    if (temp.empty())  continue;//第一次读时，如果当该张图片读空，读取下一张
    else {
        code_array.push_back(temp);//将读到的编码存入编码数组中
        break;//跳出循环
    }
}
```

2、在实验中的，调制和解调算法是什么？其中，载体信号、调制信号是什么？使用的算法属于调频、调幅还是调相？

答：调制是将信息搭载到载体上，本次实验中，就是把二进制文件写到一张图片上，调制算法对应绘制二维码数据区的部分。在本次实验中，解调就是从二维码中获取数据，得到二进制文件。载体信号是二维码，调制信号是从文件得到的比特流。在实验中，是通过可见光的不同频率（像素块的不同颜色）来表达不同信息的，所以算法属于调频算法。

调制算法：调制算法将比特流转化为二维码，主要是在比特流末尾加上 CRC 校验码，然后调用绘制二维码。调制算法具体如下图：

```
//读取待测文件字符串，加入crc校验码，用来校验数据传输
void Encode_Crc(string code, int flag) {
    string CRC = "10000010011000001000111011011011";
    code.append(CalculateCRC(code, CRC));//向code中加入计算得到的余数
    QR_Draw(code, flag);//绘制二维码
}
```

解调算法：解调算法先获取图片，获取图片后进行灰度处理、二值化，然后根据定位码定位，校正图片，最后得到一张二维码的数据区数据。解调算法在 Code_Translate(Mat& srcImage, ofstream& verify)，具体如下图：

```
//将srcImage图片转换为编码
string Code_Translate(Mat& srcImage, ofstream& verify) {
    Mat midImage, dstImage;//中间生成图像，最终图像
    string code = "";

    //处理原始图片
    midImage = Handle_Img(srcImage);//简单初步处理，去噪
    Get_ROI(midImage, srcImage, dstImage);//提取图片中的二维码存入dstImage中
    code = Decode(dstImage, verify);//将图片解码
    return code;//返回图片编码
}
```

3、在实验中的，主要的噪声强度有多大，噪声来自哪些因素？

主要的噪声来自拍摄视频时手机的色差，背景的光线，以及拍摄图片变形。

4、你的编码算法分辨率是多少？（作为实验，不要求分辨率太大）

实验中，彩色二维码的分辨率是每张图 5184（722*722）个像素。

5、你的编码波特率是多少？传输率是多少？

默认帧率 10 帧/秒，每张二维码包含 9184 bits 信息。故波特率以及传输速率为 10*9184*10-3=91.94Kb/s。

6、按奈氏定理和香农定理，通信率上限是多少？

319.2Kb /s

二、实验思考与反思

通过这一次的计网实验，学习到了很多关于通信的知识，在实验过程中遇到了一些问题，首先是 opencv 库配置问题，出现了 dll 报错，最后发现是链接器配置问题。然后是像素块的颜色判断，识别手机拍摄视频时出现了错误，没有成功编码，最后修改了 RPG 的判断条件。还有在对图片处理时，没有采用掩模的，可能导致背景图片对解码的干扰，可以再对代码进行改进。

另外也和老师讨论了二维码的定位矩形问题。实验中采用了四个定位矩形。四个定位矩形的优点在于可以对变形的拍摄图形更好地处理，代码中用 opencv 的 getPerspectiveTransform 函数对图片进行了透视变换。但是四个定位矩形也存在不能识别二维码方向的问题，实际中会在右下角的中间用一个小矩形替代。