安

卓

开

发

项目总结

项目名称：自定义美颜相机

星网智慧科技研发一部——刘永彬

### **一、开发任务**

自定义美颜相机

任务要求：

使用OPENGLES。

修改摄像头采集到的视频数据，添加相对应的滤镜功能，达到美颜磨皮效果。

要求性能达到720p且帧数达到30FPS。

### **二、项目流程**

|  |
| --- |
|  |
| 图一：程序和底层Camera框架图 |
|  |
| 图二：camera2使用流程图 |

### 三、详细设计

1.使用Camera API 2开启相机

在onCreate方法中创建并启动Camera子线程，后面Camera开启、预览、拍照都放在这个子线程中。

2.设置相机参数

在setupCamera方法中获取摄像头管理者CameraManager，由其遍历设备可用的摄像头，

获取StreamConfigurationMap，它是管理摄像头支持的所有输出格式和尺寸，根据屏幕尺寸匹配最适合的预览尺寸。

3.开启摄像头

在openCamera方法中开启摄像头，当相机成功开启后回调onOpened方法，这里可以拿到CameraDevice对象，也就是具体的摄像头设备，至此，Camera 2已经打开。

4.初始化GLSurfaceView

实例化一个GLSurfaceView，mGLSurfaceView = new GLSurfaceView(this).

配置OpenGL ES，主要是版本设置和设置Renderer，Renderer用于执行OpenGL的绘制，mGLSurfaceView.setEGLContextClientVersion(2);mGLSurfaceView.setRenderer(new GLSurfaceView.Renderer()).

在屏幕上显示GLSurfaceView.

5.创建Renderer渲染器

在Renderer的onSurfaceCreated方法中创建一个OES纹理，之后根据OES纹理Id创建SurfaceTexture，接着在onDrawFrame中更新SurfaceTexture绑定的外部纹理图像，使其获取的是最新的预览数据。

6.编译Shader和链接program

shader代码主要是对顶点着色器和片段着色器进行编译，首先是定义顶点和纹理坐标，将顶点和纹理坐标数据使用FloatBuffer来存储，防止内存回收。关联顶点数据和顶点属性，将顶点坐标和纹理坐标传输给Shader，在onDrawFrame方法中执行。

**四、主要算法**

1.camera V2

cameraV2的比较重要的类：

CameraManager： 管理手机上的所有摄像头设备，它的作用主要是获取摄像头列表和打开指定的摄像头。

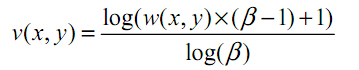
CameraDevice： 具体的摄像头设备，它有一系列参数（预览尺寸、添加相对应的滤镜功能拍照尺寸等），可以通过CameraManager的getCameraCharacteristics()方法获取。它的作用主要是创建CameraCaptureSession和CaptureRequest。

CameraCaptureSession： 相机捕获会话，用于处理拍照和预览的工作。

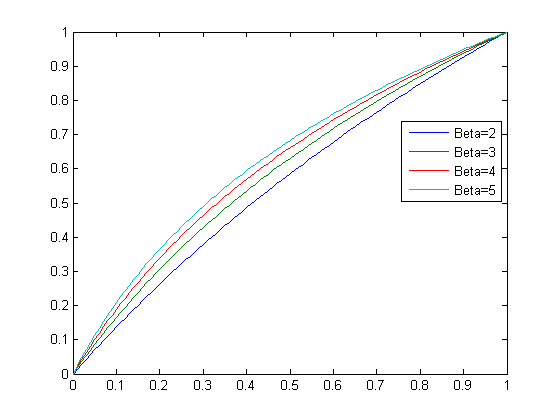
CaptureRequest： 捕获请求，定义输出缓冲区以及显示界面（TextureView或SurfaceView）等。

2.美白算法：使用logarithmic Curve

美白其实主要使皮肤变白变亮，因此，如果能有个合适的映射表，满足使得原图在色阶上有所增强，并且在亮度两端增强的稍弱，中间稍强，则也是个不错的选择。在研究一篇图像增强论文《A Two-Stage Contrast Enhancement Algorithm for Digital Images 》时发现其中的一个公式非常适合这个情况：



其中w(x,y) 表示输入图像数据，v(x,y)表示输入结果，beta为调节参数，我绘制了Beta分别为[2,3,4,5]时的曲线，如下图：



可见，Beta越大，美白的程度越强。

3.磨皮算法：对green通道进行高反差保留

将原图进行高斯模糊，把整个图片进行模糊化处理，然后使用该反差将重要的五官等进行保留处理。高反差保留 = 原图 – 高斯模糊图，然后将原图和高反差保留图进行叠加，从而得到最终磨皮效果图像。对green通道进行高反差保留主要做法是取出像素的green通道值，对其进行高反差保留，进行强光处理，计算像素灰度值，将灰度值进行次方，将原图像加上高反差保留图像乘次方值。

**五、个人总结**

本次项目是通过android-studio运用opengles实现美颜相机。在项目最开始，由于我对android开发经验比较欠缺，所以我的进度比较缓慢，第一周基本就是用来编写最简单的android项目来了解android开发，通过自己编写简单android工程以及在CSDN上浏览别人的博客，逐渐开始对android有一个初步的了解。

在对android有一个初步了解之后，在第二周开始分析项目，与组员一起讨论，最后制定了项目基本思路。我将项目分为三个部分去实现，首先是实现自定义相机的实现，制作一个可以拍照的自定义相机。随后就是在照相机的基础上增加滤镜功能，比如黑白、浮雕、分屏以及马赛克等。最后在这些基础上加上磨皮美白，并且实现性能达到720p以及帧数达到30FPS。

在实现自定义相机部分，目前主流的实现方法有camera1和camera2两种，由于camera2的API架构更加先进、并且可以获取更多的帧信息以及可以手动控制每一帧的参数等优势，我选择了使用camera2作为了我自定义相机开发。在开发初期，我查阅了github上的一些demo，但是我陷入一个误区，因为每一个人的demo都不同，而我一开始没有做相应的总结，导致出现了对整体代码的不熟悉，盲目使用别人的结构，这给后面对滤镜功能的实现埋下了隐患。在实现自定义相机功能的阶段，现在进行回想总结，感觉最错误的地方就是急功近利了，没有对整个功能代码上进行详细的总结，没有真正意义上将别人的demo转化成自己的。

第三周进行了对滤镜功能的完善，由于在自定义相机上没有进行深度的总结，在完善这个功能时一下子就卡住了，由于没有将别人的demo彻底的转化成自己的demo，导致在原有代码上对新功能的添加无从下手。在请教了组员后，我重新对自定义相机进行结构上的修改，并且对整个自定义相机的流程进行深入总结，才重新进行了对滤镜功能的完善工作。

整个滤镜功能上就是使用的opengl es2.0实现，首先创建GLSurfaceView和开启相机，之后创建一个外部纹理，根据此纹理ID创建一个SurfaceTexture，Camera将预览数据输出至此SurfaceTexture上，执行SurfaceTexture.updateTexImage()就会将一帧预览数据推送给外部纹理上。之后OpenGL ES就可以操作此纹理，比如加滤镜，滤镜就是对纹理的RGBA通道进行处理，处理后的数据就通过OpenGL ES绘制出来，其实对于OpenGL ES来说，最终就是画两个三角形，三角形的颜色取自纹理对应的位置。

在磨皮美白功能上，我们使用了对green通道进行高反差保留，因为green含有的信息量最多，因此直接处理green通道，降低计算，就能实现磨皮功能。美白则是使用logarithmic Curve算法，因为美白其实主要使皮肤变白变亮，因此，如果能有个合适的映射表，满足使得原图在色阶上有所增强，并且在亮度两端增强的稍弱，中间稍强，从而实现美白效果。

至此，本次android项目开发基本完成，在测试上基本实现功能要求，但是在细节上仍然不够完善，比如磨皮程度的滑动条改变磨皮效果不明显，对摄像头没有设置可切换以及磨皮美白功能较为单一等，这些都是需要改进的地方。这次开发过程中，我得到了组员的热心指导，在他们的帮助下，我快速的从误区里走出来，其次，导师在我遇到的各种奇奇怪怪的编译问题上耐心教导，引导我去尝试不同的解决方案。这都让我深深体会到一个团队的重要性。