实习总结

美颜相机--BeautyCamera

研发一部 实训生 陈馨鸣

一、开发任务：

编写 APK，使用 OPENGLES 方式，修改摄像头采集的视频数据，达到美颜磨皮效果，要求性能达到 720P@30FPS

二、实现功能：

1、滤镜

2、磨皮

3、美白

4、红润

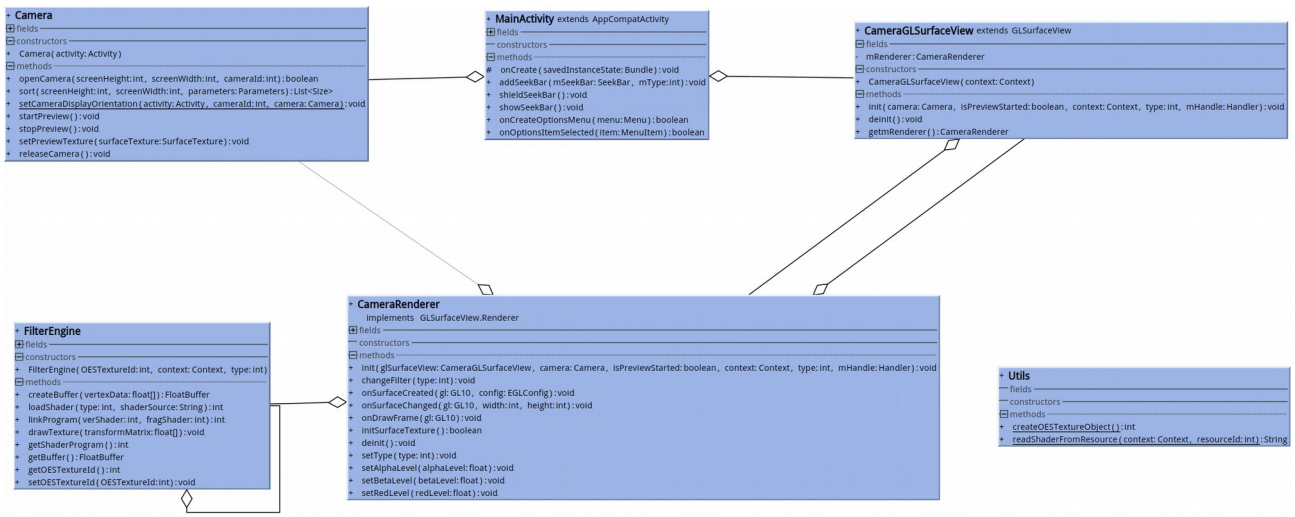
三、开发环境：

1、ubuntu 18.0.4

2、android studio 3.4.2

3、Gradle 5.1.1

四、类图



1、Camera 类

实现打开摄像头、设置摄像头参数、设置预览画面大小

获取设备支持的预览大小列表

public List<android.hardware.Camera.Size> sort

设置预览角度

public static void setCameraDisplayOrientation

开启预览

public void startPreview

停止预览

public void stopPreview

设置预览输出纹理

public void setPreviewTexture

释放摄像头

public void releaseCamera

2、CameraGLSurfaceView 类

CameraGLSurfaceView 继承 GLSurfaceView，使用 GLSurfaceView.Renderer 所提供的接口将图形绘制

初始化 CameraGLSurfaceView，配置 OpenGL ES,主要是版本设置和设置 Renderer

public void init

获取 CameraRenderer

public CameraRenderer getmRenderer

3、CameraRenderer 类

初始化渲染器

public void init

改变片着色器

public void changeFilter

当 surface 创建的时候调用

public void onSurfaceCreated

当 surface 尺寸发生改变的时候调用

public void onSurfaceChanged

实现每一帧渲染绘画

public void onDrawFrame

初始化 SurfaceTexture

public boolean initSurfaceTexture

4、FilterEngine 类

用来创建 Progress，链接，绑定顶点着色器和片着色器，告诉 OpenGL ES 使用此 program

FilterEngine 有参构造函数，初始化类内属性

public FilterEngine(int OESTextureId, Context context, int type)

将顶点和纹理坐标数据使用 FloatBuffer 来存储,防止内存回收

public FloatBuffer createBuffer

加载着色器，GL_VERTEX_SHADER 代表生成顶点着色器，GL_FRAGMENT_SHADER 代表生成片段着色器

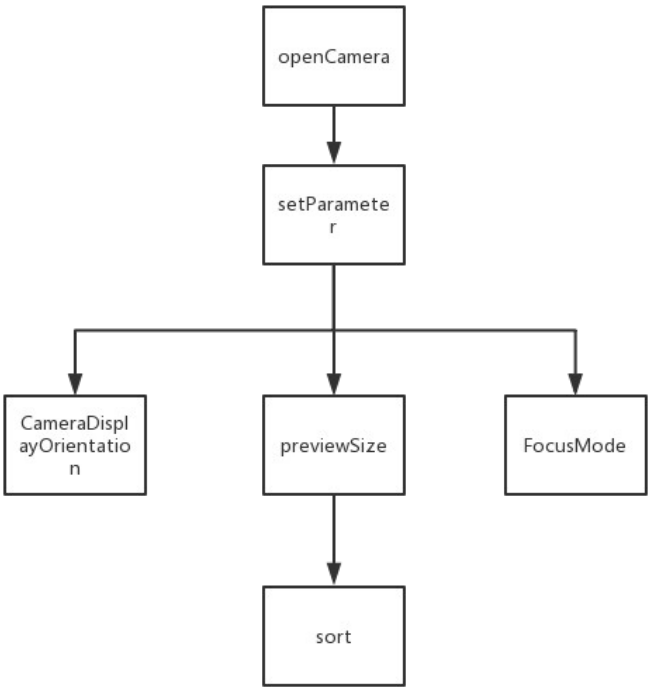
public int loadShader

将两个 Shader 链接至 program 中

public int linkProgram

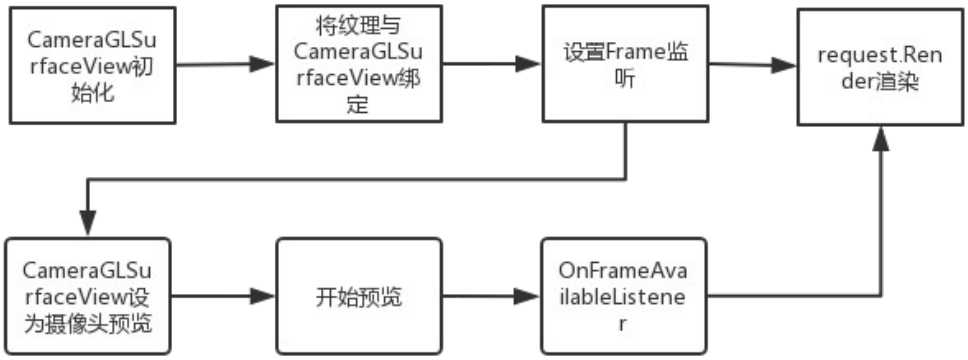
五、应用实现

1、Camera 参数设置



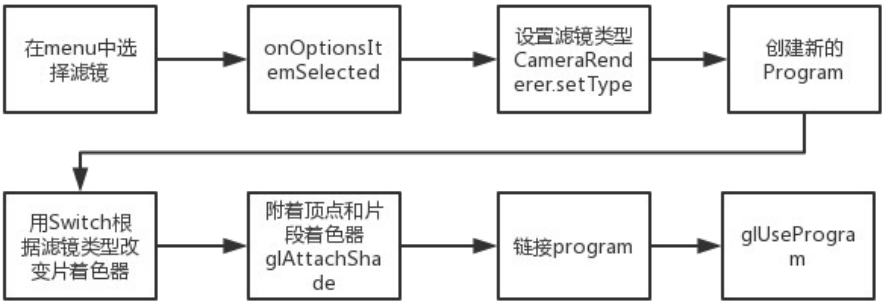
通过 Parameter 设置坐标旋转角度，预览大小及聚焦。

2、渲染



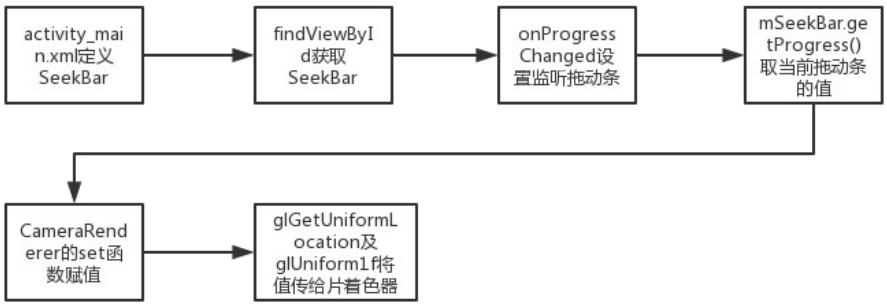
创建 CameraGLSurfaceView 和纹理，使用 SurfaceTexture 有参构造函数绑定纹理，设置监听 setOnFrameAvailableListener，设置摄像头预览 setPreviewTexture，开始预览 startPreview，每当产生 Frame 就会被 setOnFrameAvailableListener 监听到，触发 requestRender 进行渲染。

3、切换滤镜



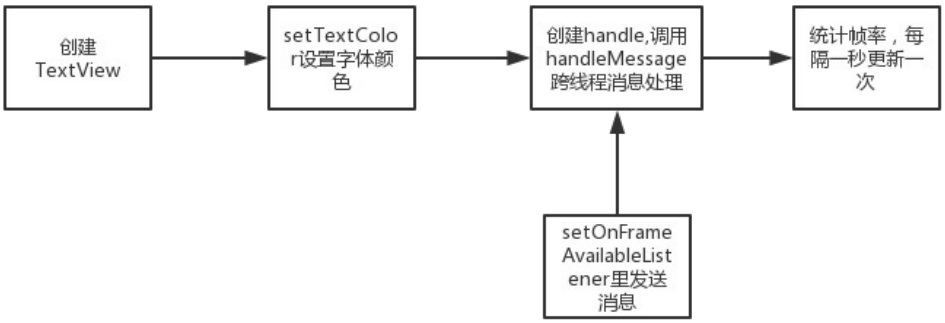
当点击菜单对应的选项，触发 onOptionsItemSelected，通过 Switch 设置相应的滤镜类型值，传给 CameraRenderer，CameraRenderer 创建新的 FilterEngine，FilterEngine 创建新的 Progress 并根据传来的值选择相应片着色器，将顶点着色器和片着色器附在 Progress 上，链接 Progress 并告诉 OpenGL ES 使用此 program。

4、Android 拖动条(SeekBar)



在 activity_main.xml 布局中定义拖动条，通过 findViewById 获取 SeekBar，设置 setOnSeekBarChangeListener 监听拖动条，onProgressChanged 调用 getProgress 传递拖动条的值给 CameraRenderer，CameraRenderer 将值传递给片着色器。

5、fps 实时帧率显示



六、磨皮算法

双边滤波磨皮：

双边滤波是一种可以保边去噪的滤波器。之所以可以达到此去噪效果，是因为滤波器是由两个函数构成，一个函数是由几何空间距离决定滤波器系数，另一个由像素差值决定滤波器系数。输出像素的值依赖于邻域像素的值的加权组合，权重系数取决于定义域核和值域核的乘积且同时考虑了空间域与值域的差别，

因此它相对于高斯模糊而言，不会模糊边缘。

将用双边滤波处理后的图像与原图叠加，线性叠加参数 beta 从拉条当前数值获取（beta*原图+（1-beta）*处理后图像）。经过测试，通过双边滤波处理实现美颜效果明显，但是每一帧处理时间太长，达到 250ms 一帧，无法实现实时美颜，因此我查询网上资料，发现有一种快速双边滤波，是在双边滤波的基础上进行的改进，克服了双边滤波处理时间长的缺点。

对 green 通道进行高反差保留：

高反差保留：

由于高斯模糊会把需要突出的五管等也模糊掉，因此需要使用高反差保留。

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，可以得到最终图像

由于 green 含有的信息量最多，可以直接对其进行处理，降低计算，实现磨皮。

首先取出像素的 green 通道值，对其进行高反差保留，进行强光处理处理，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。

这种处理可以满足帧率要求，且测试中美颜效果不错，基本满足要求。

七、感悟

这次在公司实习将近一个月时间，学到不少知识，android 开发知识、opengl 渲染、摄像头预览等等，这些知识都是在学校教学计划中没有，对我提升很大。在进行 opengl 渲染的时候，对 opengl 渲染架构还不太了解，刘宇剑导师都会很热心帮助我，给发一些相应的文档帮助我理解，真的很感谢他。

其次，我将美颜分为三部分：美白、磨皮和红润，其中磨皮算法相对于其他来讲，应该是比较难的了，我第一次实现磨皮的想法就是直接作高斯模糊然后锐化，效果一般且锐化效果明显边缘突出导致整个画面不自然，我周围实习的同学还吐槽说你这个美颜乔碧罗都哭了。后面我就去网上查找资料，发现大部分磨皮算法都用到了双边滤波，由于其模糊保留边缘，因此磨皮效果好。不过我尝试了一下，由于处理时间过长，250ms 一帧，导致画面很卡，实时帧率在 4-5 之间，无法满足 30fps 的要求，不过可以个人感觉用来处理照片还是不错的。我就想能不能改进一下，经过尝试发现还不如原来的，去网上搜了下，发现有个双边滤波的改进版快速双边滤波，解决了处理时间长的问题，不过没看懂就没用了。最后尝试的是高反差保留，这个算法最大的特点用与原图像减去高斯模糊和强光处理的图像得到的只留下边缘信息，且满足 30fps，美颜效果不错，就是去痘效果不太明显。所以了解到 green 中含有信息较多，对 green 信息做些次方处理，会不会去痘效果好些，尝试了一下，果然还不错，就是感觉拉拖动条的时候变化不明显。