短视频

视频录制

产品文档

腾讯云

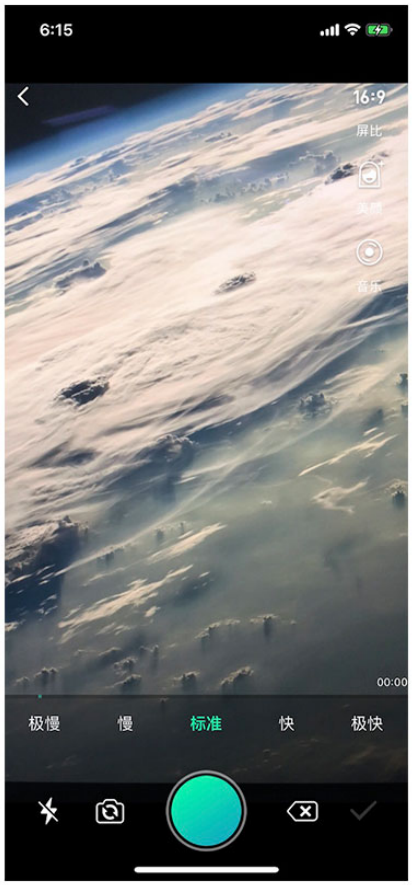
文档目录

视频录制
视频录制（iOS）

最近更新时间：2018-08-23 20:06:57

对接攻略

短视频录制即采集摄像头画面和麦克风声音，经过图像和声音处理后，进行编码压缩最终生成期望清晰度的 MP4 文件。
可以通过开发包中的DEMO工程体验录制的功能



接口介绍

腾讯云 UGC SDK 提供了相关接口用来实现短视频的录制，其详细定义如下：

接口文件	功能

接口文件	功能
TXUGCRecord.h	小视频录制功能
TXUGCRecordListener.h	小视频录制回调
TXUGCRecordEventDef.h	小视频录制事件回调
TXUGCRecordTypeDef.h	基本参数定义
TXUGCPartsManager.h	视频片段管理类，用于视频的多段录制，回删等

1. 画面预览相关

TXUGCRecord（位于 TXUGCRecord.h ）负责小视频的录制功能，我们的第一个工作是先把预览功能实现。startCameraSimplePreview函数用于启动预览。由于启动预览要打开摄像头和麦克风，所以这里可能会有权限申请的提示窗。

```
TXUGCRecord *record = [TXUGCRecord sharedInstance];
record.recordDelegate = self; //设置录制回调, 回调方法见TXUGCRecordListener

//配置相机及启动预览
TXUGCSimpleConfig * param = [[TXUGCSimpleConfig alloc] init];
//param.videoQuality = TXRecordCommon.VIDEO_QUALITY_LOW; // 360p
//param.videoQuality = TXRecordCommon.VIDEO_QUALITY_MEDIUM; // 540p
param.videoQuality = TXRecordCommon.VIDEO_QUALITY_HIGH; // 720p
param.frontCamera = YES; //使用前置摄像头
param.minDuration = 5; //视频录制的最小时长5s
param.maxDuration = 60; //视频录制的最大时长60s
param.enableBFrame = YES; // 开启B帧，相同码率下能获得更好的画面质量

//在self.previewView中显示照相机预览画面
[recorder startCameraSimple:param preview:self.previewView];

//结束画面预览
[[TXUGCRecord shareInstance] stopCameraPreview];
```

如果在相机启动后，可以通过以下方法修改:

```
// 切换视频录制分辨率到540p
[recorder setVideoResolution: VIDEO_RESOLUTION_540_960];

// 切换视频录制码率到6500Kbps
[recorder setVideoBitrate: 6500];
```

```
// 设置焦距为3, 当为1的时候为最远视角（正常镜头），当为5的时候为最近视角（放大镜头）
[recorder setZoom: 3];

// 切换到后置摄像头 YES 切换到前置摄像头 NO 切换到后置摄像头
[recorder switchCamera: NO];

// 打开闪光灯 YES为打开， NO为关闭.
[recorder toggleTorch: YES];

// 设置自定义图像处理回调
recorder.videoProcessDelegate = delegate;
```

TXVideoCustomProcessDelegate回调接口：

```
/**
* 在OpenGL线程中回调，在这里可以进行采集图像的二次处理
* @param texture 纹理ID
* @param width 纹理的宽度
* @param height 纹理的高度
* @return 返回给SDK的纹理
* 说明：SDK回调出来的纹理类型是GL_TEXTURE_2D，接口返回给SDK的纹理类型也必须是GL_TEXTURE_2D;
该回调在SDK美颜之后. 纹理格式为GL_RGBA
*/
- (GLuint)onPreProcessTexture:(GLuint)texture width:(CGFloat)width height:(CGFloat)height;

/**
* 人脸数据回调（商业版接口）
* @prama points 人脸坐标
* 说明：使用了人脸识别的相关功能如人脸识别贴纸、大眼或者瘦脸等。此回调在onPreProcessTexture:widt
h:height:之前会被调用
*/
- (void)onDetectFacePoints:(NSArray *)points;

/**
* 在OpenGL线程中回调，可以在这里释放创建的OpenGL资源
*/
- (void)onTextureDestoryed;
```

2. 画面截图

```
// 截图/拍照，startCameraSimplePreview或者startCameraCustomPreview 之后调用有效
[recorder snapshot:^(UIImage *image) {
// image为截图结果
}];
```

3. 录制相关

```
// 设置横竖屏录制
[recorder setHomeOrientation:VIDOE_HOME_ORIENTATION_RIGHT];

// 设置视频预览方向
// rotation：取值为 0，90, 180, 270（其他值无效）表示视频预览向右旋转的角度
// 注意：需要在startRecord 之前设置，录制过程中设置无效
[recorder setRenderRotation:rotation];

// 设置录制的宽高比
// VIDEO_ASPECT_RATIO_9_16 宽高比为9:16
// VIDEO_ASPECT_RATIO_3_4 宽高比为3:4
// VIDEO_ASPECT_RATIO_1_1 宽高比为1:1
// 注意：需要在startRecord 之前设置，录制过程中设置无效
[recorder setAspectRatio:VIDEO_ASPECT_RATIO_9_16];

// 设置视频录制速率
// VIDEO_RECORD_SPEED_SLOWEST, 极慢速
// VIDEO_RECORD_SPEED_SLOW, 慢速
// VIDEO_RECORD_SPEED_NOMAL, 正常速
// VIDEO_RECORD_SPEED_FAST, 快速
// VIDEO_RECORD_SPEED_FASTEST, 极快速
[recorder setRecordSpeed:VIDEO_RECORD_SPEED_NOMAL];

// 设置麦克风的音量大小，播放背景音混音时使用，用来控制麦克风音量大小
// 音量大小,1为正常音量,建议值为0~2,如果需要调大音量可以设置更大的值.
[recorder setMicVolume:volume];

// 设置录制是否静音 参数isMute代表是否静音，默认不静音
[recorder setMute:isMute];

// 开始录制
[recorder startRecord];

// 开始录制，可以指定输出视频文件地址和封面地址
[recorder startRecord:videoFilePath coverPath:coverPath];

// 开始录制,可以指定输出视频文件地址、封面地址、视频分片存储地址
[recorder startRecord:videoFilePath videoPartsFolder:videoPartFolder coverPath:coverPath];

// 暂停录制
[recorder pauseRecord];

// 继续录制
```

```
[recorder resumeRecord];

// 结束录制
[recorder stopRecord];
```

录制的过程和结果是通过 TXUGCRecordListener（位于 TXUGCRecordListener.h 中定义）协议进行回调：

- onRecordProgress 用于反馈录制的进度，参数millisecond表示录制时长，单位毫秒:

```
@optional
- (void)onRecordProgress:(NSInteger)milliSecond;
```

- onRecordComplete 反馈录制的结果，TXRecordResult 的 retCode 和 descMsg 字段分别表示错误码和错误描述信息，videoPath 表示录制完成的小视频文件路径，coverImage 为自动截取的小视频第一帧画面，便于在视频发布阶段使用。

```
@optional
- (void)onRecordComplete:(TXUGCRecordResult*)result;
```

- onRecordEvent 录制事件回调预留的接口，暂未使用

```
@optional
- (void)onRecordEvent:(NSDictionary*)evt;
```

4. 录制效果相关

在视频录制的过程中，您可以给录制视频的画面设置各种特效

```
// 设置全局水印
// normalizationFrame : 水印相对于视频图像的归一化值，sdk内部会根据水印宽高比自动计算height
// 比如视频图像大小为（540，960） frame设置为（0.1，0.1，0.1, 0）
// 水印的实际像素坐标为
// (540*0.1, 960*0.1, 540*0.1, 540*0.1*waterMarkImage.size.height / waterMarkImage.size.width）
[recorder setWaterMark:waterMarkImage normalizationFrame:frame)

//设置风格滤镜
// 设置颜色滤镜：浪漫、清新、唯美、粉嫩、怀旧...
// filterImage：指定滤镜用的颜色查找表。注意：一定要用png格式
// demo用到的滤镜查找表图片位于FilterResource.bundle中
[recorder setFilter:filterImage];
```

```
// 用于设置滤镜的效果程度，从0到1，越大滤镜效果越明显，默认取值0.5
[recorder setSpecialRatio:ratio];

// 设置组合滤镜特效
// mLeftBitmap 左侧滤镜
// leftIntensity 左侧滤镜强度
// mRightBitmap 右侧滤镜
// rightIntensity 右侧滤镜强度
// leftRadio 左侧图片占的比例大小
// 可以此接口实现滑动切换滤镜的效果，详见demo。
[recorder setFilter:leftFilterImgage leftIntensity:leftIntensity rightFilter:rightFilterImgage rightIntensity:rightIntensity leftRatio:leftRatio];

// 设置美颜风格、级别、美白及红润的级别
// beautyStyle的定义如下:
// typedef NS_ENUM(NSInteger, TXVideoBeautyStyle) {
// VIDOE_BEAUTY_STYLE_SMOOTH = 0, // 光滑
// VIDOE_BEAUTY_STYLE_NATURE = 1, // 自然
// VIDOE_BEAUTY_STYLE_PITU = 2, // pitu美颜, 需要购买商业版
// };
// 级别的范围为0-9 0为关闭， 1-9值越大，效果越明显
[recorder setBeautyStyle:beautyStyle beautyLevel:beautyLevel whitenessLevel:whitenessLevel ruddinessLevel:ruddinessLevel];

// 设置大眼效果 建议0~9，如果需要更明显可以设置更大值
[recorder setEyeScaleLevel:eyeScaleLevel];

// 设置瘦脸效果 建议0~9，如果需要更明显可以设置更大值
[recorder setFaceScaleLevel:faceScaleLevel];

// 设置V脸效果 建议0~9，如果需要更明显可以设置更大值
[recorder setFaceVLevel:level];

// 设置下巴拉伸或收缩效果 建议0~9，如果需要更明显可以设置更大值
[recorder setChinLevel:scale];

// 设置缩脸效果 建议0~9，如果需要更明显可以设置更大值
[recorder setFaceShortLevel:level];

// 设置小鼻效果 建议0~9，如果需要更明显可以设置更大值
[recorder setNoseSlimLevel:scale];

// 设置绿幕文件:目前图片支持jpg/png，视频支持mp4/3gp等Android系统支持的格式并支持循环播放
[recorder setGreenScreenFile:file];
```

```
// 设置动效贴纸 tmplDir 动效文件路径, tmplName为动效文件名, tmplName为nil则取消动效
[recorder selectMotionTmpl:tmplName inDir:tmplDir];

// 设置动效贴纸是否静音
[recorder setMotionMute:YES];
```

5. 录制BGM相关

在视频录制的过程中，您可以给视频的添加喜欢的BGM

```
// 设置BGM文件路径
[recorder setBGMAsset:path];

// 设置BGM，从系统媒体库loading出来的音乐，可以直接传入对应的AVAsset
[recorder setBGMAsset:asset];

// 播放BGM
[recorder playBGMFromTime:beginTime
toTime:endTime
withBeginNotify:^(NSInteger errCode) {
// 播放开始回调, errCode0为成功其它为失败
} withProgressNotify:^(NSInteger progressMS, NSInteger durationMS) {
// progressMS: 已经播放的时长， durationMS: 总时长
} andCompleteNotify:^(NSInteger errCode) {
// 播放结束回调, errCode0为成功其它为失败
}];

// 停止播放BGM
[recorder stopBGM];

// 暂停播放BGM
[recorder pauseBGM];

// 继续播放BGM
[recorder resumeBGM];

// 设置麦克风的音量大小，播放背景音乐混音时使用，用来控制麦克风音量大小
// volume: 音量大小，1为正常音量，建议值为0~2，如果需要调大音量可以设置更大的值
[recorder setMicVolume:1.0];

// setBGMVolume 设置背景音乐的音量大小，播放背景音乐混音时使用，用来控制背景音音量大小
// volume: 音量大小，1为正常音量，建议值为0~2，如果需要调大背景音量可以设置更大的值
[recorder setBGMVolume:1.0];
```

6. 录制声音特效相关

```
// 设置混响
// TXRecordCommon.VIDOE_REVERB_TYPE_0 关闭混响
// TXRecordCommon.VIDOE_REVERB_TYPE_1 KTV
// TXRecordCommon.VIDOE_REVERB_TYPE_2 小房间
// TXRecordCommon.VIDOE_REVERB_TYPE_3 大会堂
// TXRecordCommon.VIDOE_REVERB_TYPE_4 低沉
// TXRecordCommon.VIDOE_REVERB_TYPE_5 洪亮
// TXRecordCommon.VIDOE_REVERB_TYPE_6 金属声
// TXRecordCommon.VIDOE_REVERB_TYPE_7 磁性
[recorder setReverbType:VIDOE_REVERB_TYPE_1];

// 设置变声
// TXRecordCommon.VIDOE_VOICECHANGER_TYPE_0 关闭变声
// TXRecordCommon.VIDOE_VOICECHANGER_TYPE_1 熊孩子
// TXRecordCommon.VIDOE_VOICECHANGER_TYPE_2 萝莉
// TXRecordCommon.VIDOE_VOICECHANGER_TYPE_3 大叔
// TXRecordCommon.VIDOE_VOICECHANGER_TYPE_4 重金属
// TXRecordCommon.VIDOE_VOICECHANGER_TYPE_6 外国人
// TXRecordCommon.VIDOE_VOICECHANGER_TYPE_7 困兽
// TXRecordCommon.VIDOE_VOICECHANGER_TYPE_8 死肥仔
// TXRecordCommon.VIDOE_VOICECHANGER_TYPE_9 强电流
// TXRecordCommon.VIDOE_VOICECHANGER_TYPE_10 重机械
// TXRecordCommon.VIDOE_VOICECHANGER_TYPE_11 空灵
[record setVoiceChangerType:VIDOE_VOICECHANGER_TYPE_1];
```

7. 多段录制及回删

在视频录制的过程中，您可以录制多段视频，并且管理这些视频，最终合成一个完整的视频

```
// 开始录制
[recorder startRecord];

// 调用 pauseRecord 后会生成一段视频，视频可以在 TXUGCPartsManager 里面获取管理
[recorder pauseRecord];

// 继续录制视频
[recorder resumeRecord];

// 停止录制，将多段视频合成为一个视频输出
[recorder stopRecord];

//获取视频分片管理对象
TXUGCPartsManager *partsManager = recorder.partsManager;
```

```
//获取当前所有视频片段的总时长
[partsManager getDuration];

//获取所有视频片段路径
[partsManager getVideoPathList];

// 删除最后一段视频
[partsManager deleteLastPart];

// 删除指定片段视频
[partsManager deletePart:1];

// 删除所有片段视频
[partsManager deleteAllParts];

//合成所有片段视频
[partsManager joinAllParts: videoOutputPath];
```

8. 文件预览

使用 视频播放 即可预览刚才生成的 MP4 文件，需要在调用 startPlay 时指定播放类型为 PLAY_TYPE_LOCAL_VIDEO 。

9. 获取 license 信息

参考 短视频license集成

视频录制（Android）

最近更新时间：2018-08-22 21:15:44

对接攻略

短视频录制即采集摄像头画面和麦克风声音，经过图像和声音处理后，进行编码压缩最终生成期望清晰度的 MP4 文件。
可以通过开发包中的DEMO工程体验录制的功能

Android录制功能的代码位置：com.tencent.liteav.demo.videorecord 包名下面，其中 TCVideoSettingActivity 是录制设置界面，TCVideoRecordActivity 是录制界面，另外需要拷贝界面中所需的资源文件，就可以实现录制的界面效果和功能了。

接口介绍

腾讯云 UGC SDK 提供了相关接口用来实现短视频的录制，其详细定义如下：

接口文件	功能
TXUGCRecord.java	实现小视频的录制功能

接口文件	功能
TXRecordCommon.java	基本参数定义，包括了小视频录制回调及发布回调接口
TXUGCPartsManager.java	视频片段管理类，用于视频的多段录制，回删等
ITXVideoRecordListener.java	小视频录制回调

1. 画面预览相关

TXUGCRecord（位于 TXUGCRecord.java ）负责小视频的录制功能，我们的第一个工作是先把预览功能实现。startCameraSimplePreview函数用于启动预览。由于启动预览要打开摄像头和麦克风，所以这里可能会有权限申请的提示窗。

```
mTXCameraRecord = TXUGCRecord.getInstance(this.getApplicationContext());
mTXCameraRecord.setVideoRecordListener(this); //设置录制回调
mVideoView = (TXCloudVideoView) findViewById(R.id.video_view); //准备一个预览摄像头画面的 view
mVideoView.enableHardwareDecode(true);
TXRecordCommon.TXUGCSimpleConfig param = new TXRecordCommon.TXUGCSimpleConfig();
//param.videoQuality = TXRecordCommon.VIDEO_QUALITY_LOW; // 360p
param.videoQuality = TXRecordCommon.VIDEO_QUALITY_MEDIUM; // 540p
//param.videoQuality = TXRecordCommon.VIDEO_QUALITY_HIGH; // 720p
param.isFront = true; //是否前置摄像头，使用
param.minDuration = 5000; //视频录制的最小时长ms
param.maxDuration = 60000; //视频录制的最大时长ms
param.needEdit = false; // 录制完是否直接进入编辑界面编辑视频。false:生成指定参数的视频；true:生成高码率的视频，可快速进入编辑界面使用全部编辑功能
mTXCameraRecord.startCameraSimplePreview(param,mVideoView);

// 停止摄像头预览
stopCameraPreview();

// 切换视频录制分辨率
mTXCameraRecord.setVideoResolution(TXRecordCommon.VIDEO_RESOLUTION_540_960);

// 切换视频录制码率
mTXCameraRecord.setVideoBitrate(6500);

// 获取摄像头支持的最大焦距
mTXCameraRecord.getMaxZoom();

// 设置焦距
mTXCameraRecord.setZoom(value);

// 切换前后摄像头 参数 mFront 代表是否前置摄像头 默认前置
```

```
mTXCameraRecord.switchCamera(mFront);

// 是否打开闪光灯 参数 mFlashOn 代表是否打开闪关灯 默认关闭
mTXCameraRecord.toggleTorch(mFlashOn);

// 设置自定义图像处理回调
mTXCameraRecord.setVideoProcessListener(customProcessListener);
```

VideoCustomProcessListener回调接口：

```
/**
在OpenGL线程中回调，在这里可以进行采集图像的二次处理
Parameters:
textureId - 纹理ID
width - 纹理的宽度
height - 纹理的高度
Returns:
返回给SDK的纹理ID，如果不做任何处理，返回传入的纹理ID即可
说明：SDK回调出来的纹理类型是GLES20.GL_TEXTURE_2D，接口返回给SDK的纹理类型也必须是GLES20.GL_TEXTURE_2D
*/
int onTextureCustomProcess(int textureId, int width, int height);

/**
商业版回调人脸坐标
Parameters:
points - 归一化人脸坐标，每两个值表示某点P的X,Y值。值域[0.f,1.f]
*/
void onDetectFacePoints(float[] points);

/**
在OpenGL线程中回调，可以在这里释放创建的OpenGL资源
*/
void onTextureDestroyed();
```

2. 画面截图

```
// 截图/拍照，startCameraSimplePreview或者startCameraCustomPreview 之后调用有效
mTXCameraRecord.snapshot(new TXRecordCommon.ITXSnapshotListener() {
@Override
public void onSnapshot(Bitmap bmp) {
// 保存或者显示截图
}
});
```

3. 录制相关

```
// 设置横竖屏录制
mTXCameraRecord.setHomeOrientation(TXLiveConstants.VIDEO_ANGLE_HOME_DOWN);

// 设置视频预览方向
mTXCameraRecord.setRenderRotation(TXLiveConstants.RENDER_ROTATION_PORTRAIT);

// 设置录制的宽高比
// TXRecordCommon.VIDEO_ASPECT_RATIO_9_16 宽高比为9:16
// TXRecordCommon.VIDEO_ASPECT_RATIO_3_4 宽高比为3:4
// TXRecordCommon.VIDEO_ASPECT_RATIO_1_1 宽高比为1:1
mTXCameraRecord.setAspectRatio(TXRecordCommon.VIDEO_ASPECT_RATIO_9_16);

// 设置视频录制速率
mTXCameraRecord.setRecordSpeed(TXRecordCommon.RECORD_SPEED_NORMAL);

// 设置麦克风的音量大小，播放背景音混音时使用，用来控制麦克风音量大小
// 音量大小,1为正常音量,建议值为0~2,如果需要调大音量可以设置更大的值.
mTXCameraRecord.setMicVolume(x);

// 设置录制是否静音 参数isMute代表是否静音，默认不静音
mTXCameraRecord.setMute(isMute);

// 开始录制
mTXCameraRecord.startRecord();

// 开始录制，可以指定输出视频文件地址和封面地址
mTXCameraRecord.startRecord(videoFilePath, coverPath)

// 开始录制,可以指定输出视频文件地址、封面地址、视频分片存储地址
mTXCameraRecord.startRecord(videoFilePath, videoPartFolder, coverPath);

// 暂停录制
mTXCameraRecord.pauseRecord();

// 继续录制
mTXCameraRecord.resumeRecord();

// 结束录制
mTXCameraRecord.stopRecord();
```

录制的过程和结果是通过 TXRecordCommon.ITXVideoRecordListener（位于 TXRecordCommon.java 中定义）接口反馈出来的：

- onRecordProgress 用于反馈录制的进度，参数millisecond表示录制时长，单位毫秒:

```
@optional
void onRecordProgress(long milliSecond);
```

- onRecordComplete 反馈录制的结果，TXRecordResult 的 retCode 和 descMsg 字段分别表示错误码和错误描述信息，videoPath 表示录制完成的小视频文件路径，coverImage 为自动截取的小视频第一帧画面，便于在视频发布阶段使用。

```
@optional
void onRecordComplete(TXRecordResult result);
```

- onRecordEvent 录制事件回调，包含事件id和事件相关的参数(key,value)格式

```
@optional
void onRecordEvent(final int event, final Bundle param);
```

4. 录制效果相关

在视频录制的过程中，您可以给录制视频的画面设置各种特效

```
// 设置全局水印
//TXRect-水印相对于视频图像的归一化值，sdk内部会根据水印宽高比自动计算height
// 比如视频图像大小为（540，960）TXRect三个参数设置为0.1，0.1，0.1,
// 水印的实际像素坐标为（540 * 0.1，960 * 0.1，540 * 0.1 ，
// 540 * 0.1 * watermarkBitmap.height / watermarkBitmap.width ）
mTXCameraRecord.setWatermark(watermarkBitmap, txRect)

// 设置美颜类型
mTXCameraRecord.setBeautyStyle(style);

// 设置美颜、美白、红润效果程度
mTXCameraRecord.setBeautyDepth(int style, int beautyDepth, int whiteningDepth, int ruddyDepth);

// 设置颜色滤镜：浪漫、清新、唯美、粉嫩、怀旧...
// filterBitmap：指定滤镜用的颜色查找表。注意：一定要用png格式！！！
// demo用到的滤镜查找表图片位于RTMPAndroidDemo/app/src/main/res/drawable-xxhdpi/目录下。
mTXCameraRecord.setFilter(filterBitmap);

// 设置组合滤镜特效
// mLeftBitmap 左侧滤镜
// leftIntensity 左侧滤镜程度
// mRightBitmap 右侧滤镜
// rightIntensity 右侧滤镜程度
// leftRadio 左侧图片占的比例大小
```

```
// 可以此接口实现滑动切换滤镜的效果，详见demo。
mTXCameraRecord.setFilter(mLeftBitmap, leftIntensity, mRightBitmap, rightIntensity, leftRatio);

// 用于设置滤镜的效果程度，从0到1，越大滤镜效果越明显，默认取值0.5
mTXCameraRecord.setSpecialRatio(0.5);

// 设置大眼效果 建议0~9，如果需要更明显可以设置更大值
mTXCameraRecord.setEyeScaleLevel(eyeScaleLevel);

// 设置瘦脸效果 建议0~9，如果需要更明显可以设置更大值
mTXCameraRecord.setFaceScaleLevel(faceScaleLevel);

// 设置V脸效果 建议0~9，如果需要更明显可以设置更大值
mTXCameraRecord.setFaceVLevel(level);

// 设置下巴拉伸或收缩效果 建议0~9，如果需要更明显可以设置更大值
mTXCameraRecord.setChinLevel(scale);

// 设置缩脸效果 建议0~9，如果需要更明显可以设置更大值
mTXCameraRecord.setFaceShortLevel(level);

// 设置小鼻效果 建议0~9，如果需要更明显可以设置更大值
mTXCameraRecord.setNoseSlimLevel(scale);

// 设置绿幕文件:目前图片支持jpg/png，视频支持mp4/3gp等Android系统支持的格式并支持循环播放
mTXCameraRecord.setGreenScreenFile(path, isLoop);

// 设置动效贴纸 motionTmplPath 动效文件路径： 空String "" 则取消动效
mTXCameraRecord.setMotionTmp(motionTmplPath);

// 设置动效贴纸是否静音: true:动效贴纸静音；false：动效贴纸不静音
mTXCameraRecord.setMotionMute(true);
```

5. 录制BGM相关

在视频录制的过程中，您可以给视频的添加喜欢的BGM

```
// 设置BGM路径
mTXCameraRecord.setBGM(path);

// 设置BGM播放回调 TXRecordCommon.ITXBGMNotify
mTXCameraRecord.setBGMNofify(notify);

// 播放BGM
mTXCameraRecord.playBGMFromTime(startTime, endTime)
```

```
// 停止播放BGM
mTXCameraRecord.stopBGM();

// 暂停播放BGM
mTXCameraRecord.pauseBGM();

// 继续播放BGM
mTXCameraRecord.resumeBGM();

// 设置背景音乐的音量大小，播放背景音乐混音时使用，用来控制背景音音量大小
// 音量大小,1为正常音量,建议值为0~2,如果需要调大背景音量可以设置更大的值.
mTXCameraRecord.setBGMVolume(x);

// 设置背景音乐播放的开始位置和结束位置
mTXCameraRecord.seekBGM(startTime, endTime);
```

6. 录制声音特效相关

```
// 设置混响
// TXRecordCommon.VIDOE_REVERB_TYPE_0 关闭混响
// TXRecordCommon.VIDOE_REVERB_TYPE_1 KTV
// TXRecordCommon.VIDOE_REVERB_TYPE_2 小房间
// TXRecordCommon.VIDOE_REVERB_TYPE_3 大会堂
// TXRecordCommon.VIDOE_REVERB_TYPE_4 低沉
// TXRecordCommon.VIDOE_REVERB_TYPE_5 洪亮
// TXRecordCommon.VIDOE_REVERB_TYPE_6 金属声
// TXRecordCommon.VIDOE_REVERB_TYPE_7 磁性
mTXCameraRecord.setReverb(TXRecordCommon.VIDOE_REVERB_TYPE_1);

// 设置变声
// TXRecordCommon.VIDOE_VOICECHANGER_TYPE_0 关闭变声
// TXRecordCommon.VIDOE_VOICECHANGER_TYPE_1 熊孩子
// TXRecordCommon.VIDOE_VOICECHANGER_TYPE_2 萝莉
// TXRecordCommon.VIDOE_VOICECHANGER_TYPE_3 大叔
// TXRecordCommon.VIDOE_VOICECHANGER_TYPE_4 重金属
// TXRecordCommon.VIDOE_VOICECHANGER_TYPE_6 外国人
// TXRecordCommon.VIDOE_VOICECHANGER_TYPE_7 困兽
// TXRecordCommon.VIDOE_VOICECHANGER_TYPE_8 死肥仔
// TXRecordCommon.VIDOE_VOICECHANGER_TYPE_9 强电流
// TXRecordCommon.VIDOE_VOICECHANGER_TYPE_10 重机械
// TXRecordCommon.VIDOE_VOICECHANGER_TYPE_11 空灵
mTXCameraRecord.setVoiceChangerType(TXRecordCommon.VIDOE_VOICECHANGER_TYPE_1);
```

7. 多段录制及回删

```
// 开始录制
mTXCameraRecord.startRecord();

// pauseRecord 后会生成一段视频，视频可以在 TXUGCPartsManager 里面获取
mTXCameraRecord.pauseRecord();

// 继续录制视频
mTXCameraRecord.resumeRecord();

// 停止录制，将多段视频合成为一个视频输出
mTXCameraRecord.startRecord();

// 获取片段管理对象
mTXCameraRecord.getPartsManager();

// 获取当前所有视频片段的总时长
mTXUGCPartsManager.getDuration();

// 获取所有视频片段路径
mTXUGCPartsManager.getPartsPathList();

// 删除最后一段视频
mTXUGCPartsManager.deleteLastPart();

// 删除指定片段视频
mTXUGCPartsManager.deletePart(index);

// 删除所有片段视频
mTXUGCPartsManager.deleteAllParts();
```

8. 文件预览

使用 视频播放 即可预览刚才生成的 MP4 文件，需要在调用 startPlay 时指定播放类型为 PLAY_TYPE_LOCAL_VIDEO 。

9. 获取 license 信息

参考 短视频license集成