客户端实时多媒体 SDK
性能优化实例分享

Piasy 许建林

有多少人做过相机相关开发?

发 Intent 拍照? Camera API 预览、获取数据、编码、推流?

自我介绍

- Piasy，许建林，Powerinfo
- https://github.com/Piasy
- https://blog.piasy.com
- 拆轮子：必备开源库源码导读
- Advanced RxJava 系列翻译
- RTC fullstack

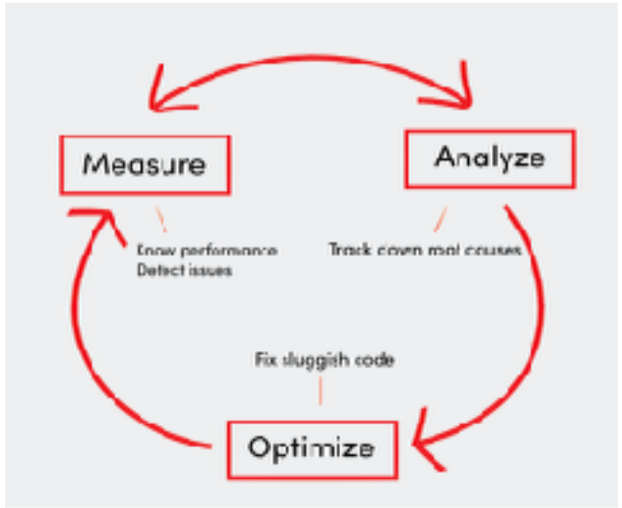
目录

- 背景介绍
- 实例分享

背景介绍

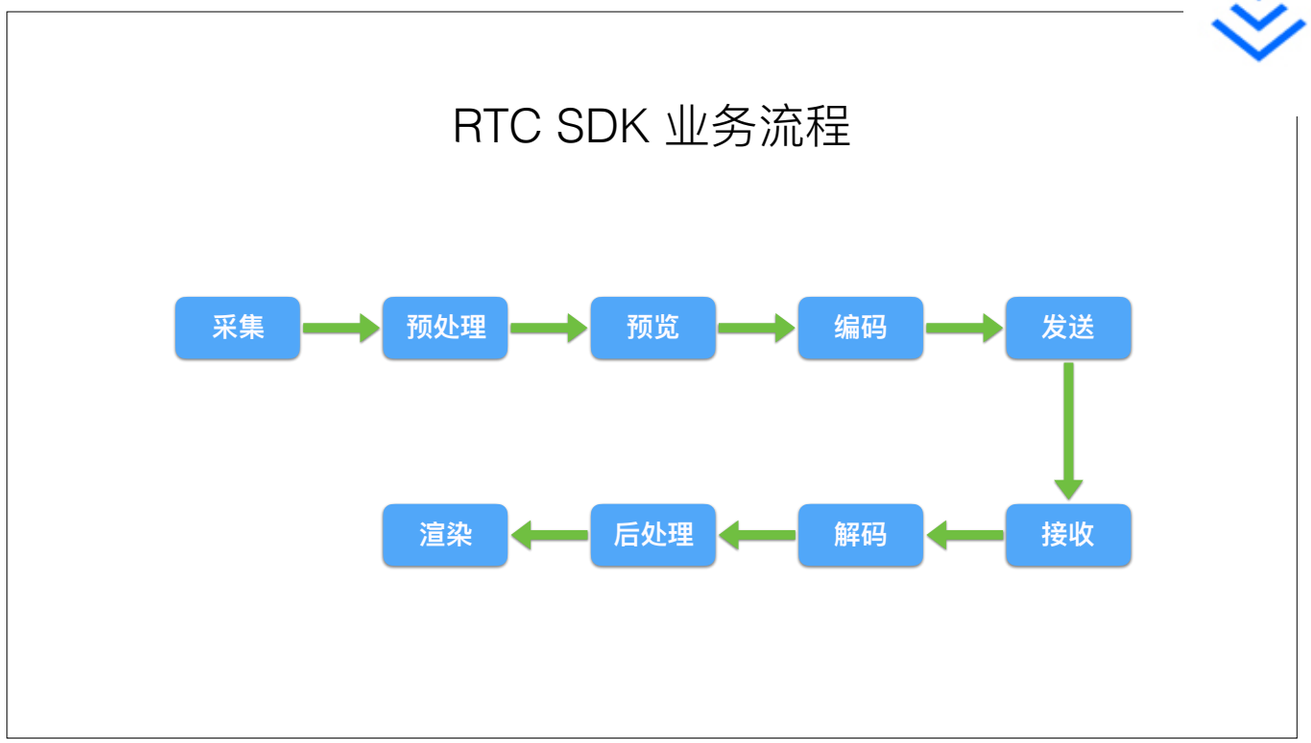
- 性能优化流程
- RTC SDK 业务流程

性能优化流程



http://blog.nimbledroid.com/2015/09/17/how-to-make-your-application-fluid.html

一切优化，始于测量
测量-分析 可能是反复的
优化之后，一定要测量验证



为什么要了解业务流程?
始于测量没错，但我们可能无法无差别测量。
而且不同流程涉及不同模块，是有分工的。

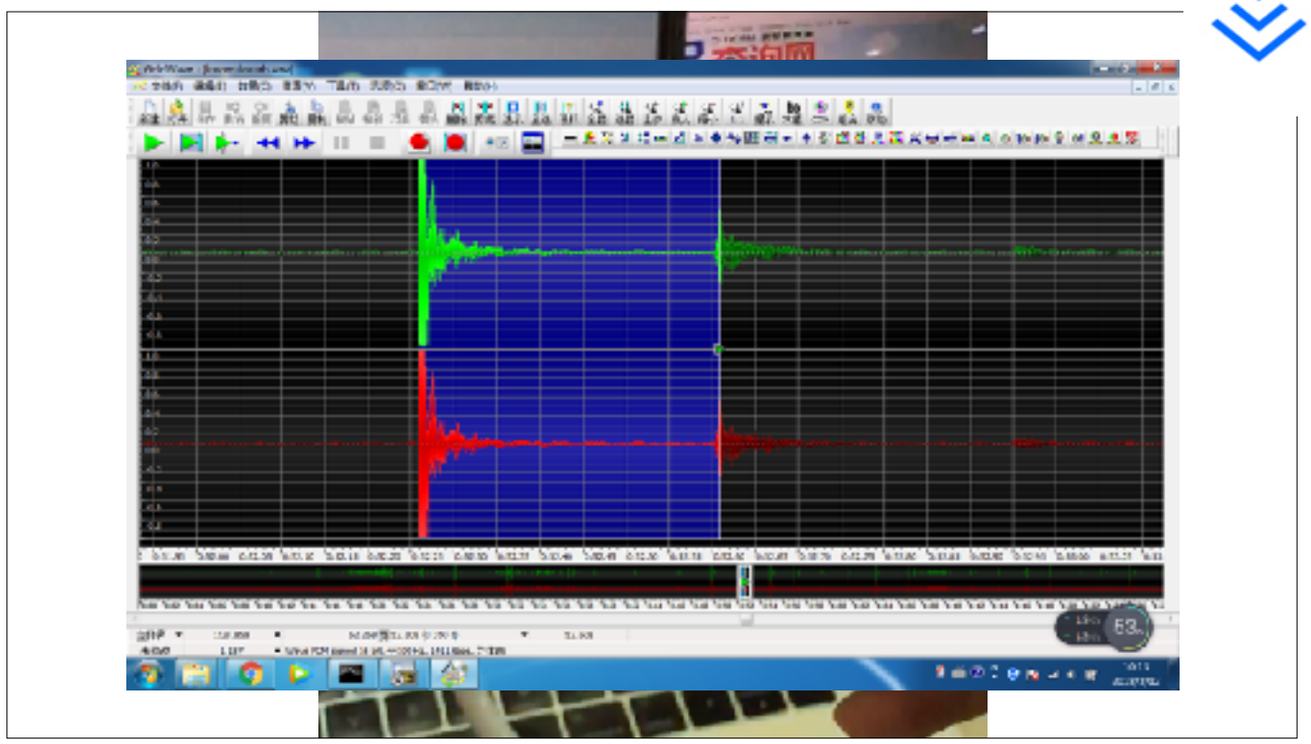
紧扣场景做优化

脱离场景谈优化，都是耍流氓

紧扣场景做优化

- 推流首屏时间
- 屏到屏延迟
- CPU 占用
- 内存抖动

不包括传输模块



秒开：按秒表？拍秒表+慢动作！ => 代码统计

视频延迟：拍秒表+拍照！

声音延迟：goldwave！

内存&CPU：Android Studio monitor！

推流首屏优化实例

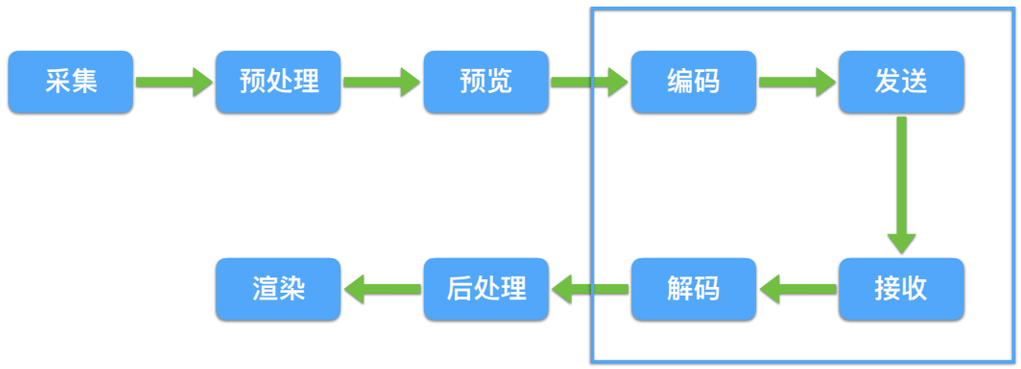
- 目标：点击按钮~输出首个音视频帧
- 操作
  - 开启视频采集
  - 开启音频采集
  - 开启视频编码
  - 开启音频编码
  - 开启推流底层

并行！

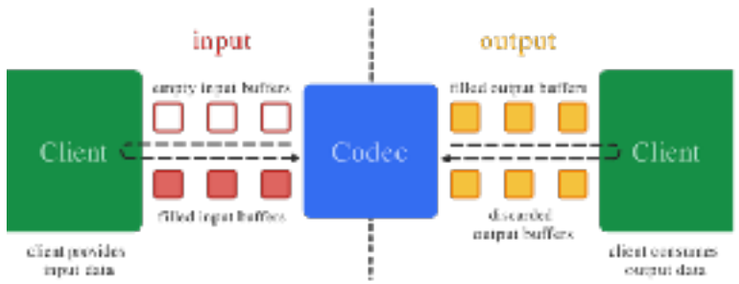
提前！

300ms -> 100ms

屏到屏延迟优化实例



MediaCodec 简介



三种角色，两个队列

屏到屏延迟优化实例

```
private void handleFrameAvailable(float[] transform, long timestampNanos) {
    mVideoEncoder.drainEncoder(false);
    mFullScreen.drawFrame(mTextureId, transform);
    mInputWindowSurface.setPresentationTime(timestampNanos);
    mInputWindowSurface.swapBuffers();
}
```

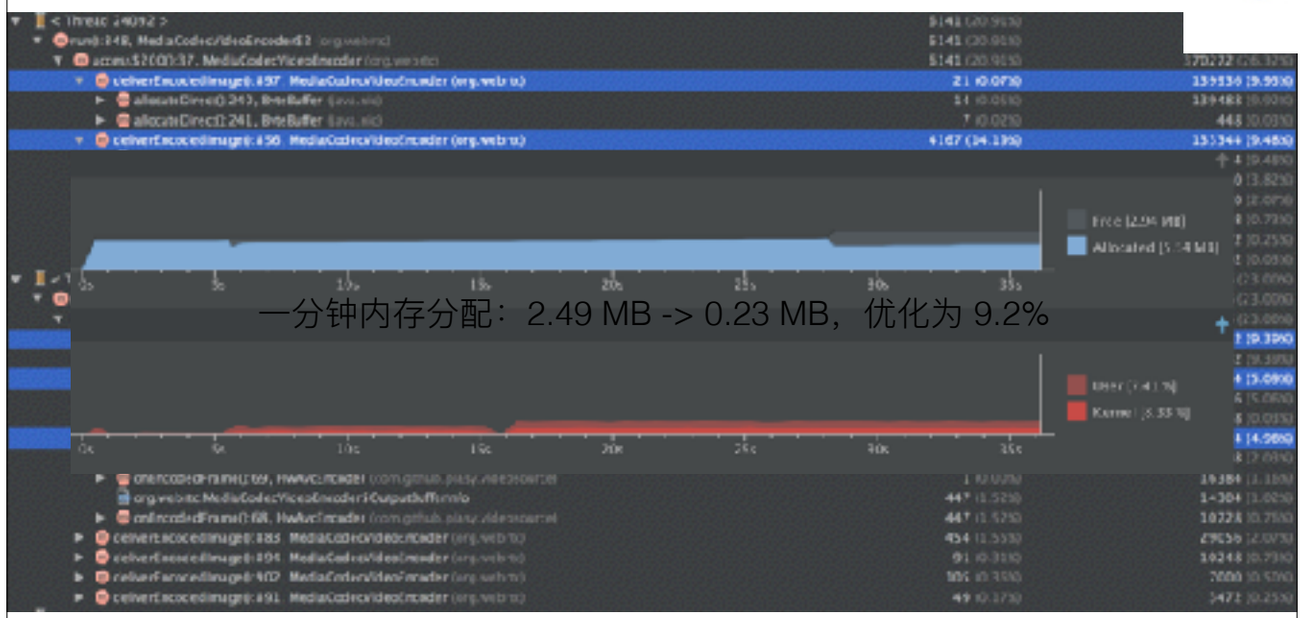
单独线程消费输出：减小 1~2 帧（40~80ms）

https://github.com/google/grafika/blob/c747398a8f0d5c8ec7be2c66522a80b43dfc9a1e/app/src/main/java/com/android/grafika/TextureMovieEncoder.java#L325-L334

CPU 占用优化实例

- 主播推流 800*448
- 辅播推流 256*144
- 两个视频合成一个视频后分发给观众

AS 的 monitor 可以查看 CPU 占用，但只包括 APP 进程，还应关注相关系统进程，例如 mediaserver，adb shell + top



https://github.com/Piasy/VideoCRE/commit/e48dd49b20205fb80ea0aa9602269f4ccd4e2369

60% 在 BufferInfo，dequeueOutputBuffer 传入的 timeout 为 0，15s 内发生了 2.6 万次，1.7/ms => timeout 3ms

字符串拼接 14%，不要以为在日志工具函数内通过变量控制是否打日志就够了，即便日志最终没有打印出来，但拼接日志字符串就可能已经成为瓶颈。

BufferInfo 19% 和 ByteBuffer 19%，不必每次分配，初始化/扩容时分配即可。

Thanks!

Q&A