

VideoDecoder 库

简介

`VideoDecoder` 库提供了一个简单的接口，用于解码视频文件并将其转换为RGB格式的帧。此库基于FFmpeg，支持FFmpeg支持的所有视频格式。

编译和链接


安装FFMEPG

需要安装libavcodec-dev libavformat-dev libswscale-dev三个库。以下指令为Ubuntu系统的参考指令，如果使用其他系统，请自行搜索相关教程。

```
1 sudo apt install libavcodec-dev libavformat-dev libswscale-dev
```


编译

- `libvideodecoder.a` 为封装好的静态库




libvideodecoder.a

10.29KB




- `video_decoder.h` 为对应头文件



video_decoder.h

365 B



编译你的程序时，确保链接 `VideoDecoder` 库和 `ffmpeg` 相关库：

```
1 gcc main.c -o main -L. -lvideodecoder -lavformat -lavcodec -lavutil -lswscale
```

结构体

Frame

```
1 typedef struct _Frame{
2     int width;
3     int height;
4     int linesize;
5     unsigned char** data;
6 } Frame;
```

- `data`: 这是一个指向数据平面数组的指针。对于RGB图像，`data[0]` 将包含一个指针，指向包含整个帧的RGB像素数据的内存区域。每个像素由**3个字节**组成，分别对应红色、绿色和蓝色。
- `linesize`: 每个数据平面，**每行**数据的字节数。
- `width`, `height`: 视频帧的宽度和高度。

函数

```
int decoder_init(const char *filename)
```

初始化视频解码器并打开指定的视频文件。请在执行以下任何函数前，先初始化decoder。

- `const char *filename`: 要打开的视频文件路径。
- **返回值**: 成功时返回 `0`，失败时返回 `-1`。

```
Frame decoder_get_frame()
```

从视频中获取下一帧。

- **返回值**: 成功时返回帧数据，没有更多帧或获取失败时返回内容为空的Frame结构体。

```
void decoder_close()
```

关闭视频解码器并释放所有资源。

```
double get_fps()
```

获取视频的帧率。注意：当视频为可变帧率时，该函数获取的帧率值可能错误。

- **返回值**: 视频的帧率。

```
int get_frame_index()
```

获取当前帧的索引。

- **返回值:** 当前帧的索引。

```
int get_total_frames()
```

获取视频的总帧数。注意：当视频为可变帧率时，该函数获取的总帧数可能错误。

- **返回值:** 视频的总帧数。