

# 百度行驶证 C++离线识别 SDK1.1

## 用户接入文档

百度在线网络技术（北京）有限公司  
(版权所有, 翻版必究)

## 修改记录

No	修改后版本号	修改内容简介	修改日期	修改人
1	V1.1	1.1 定稿	2023.10.18	

## 目 录

百度行驶证 C++离线识别 SDK1.1 .....	1
用户接入文档.....	1
1 背景.....	1
2 SDK 简介.....	1
2.1 鉴权.....	1
2.1.1 自动激活授权 .....	1
2.1.2 官网离线激活授权 .....	1
2.2 支持平台 .....	2
2.3 开发工具.....	2
2.4 依赖库及运行环境.....	2
2.5 SDK 包结构.....	2
3 SDK 接入.....	3
4 功能接口.....	7
4.1 正页.....	7
4.2 副页.....	9
4.3 正页推理.....	11
4.4 副页推理.....	12
5 常见问题.....	12
5.1 关于 IDE.....	12
5.2 激活后是否可以把激活文件 license.ini 和 license.key 拷贝到其他设备运行? .....	12
5.3 是否支持 Debug 模式? .....	12
只支持 Release 模式，不支持 debug 模式.....	12
5.4 SDK 支持 X64 模式。 .....	12
SDK 仅支持 X64 模式，支持运行在 win7 或 win10 系统中。 .....	12

# 1 背景

为使客户、第三方开发者等能够更快速、方便的接入使用百度行驶证离线识别 SDK、促进百度 OCR 产品赋能更多客户，特设计支持 C++语言的行驶证离线识别 SDK。

## 2 SDK 简介

本 SDK 适应于 Windows 平台下的操作系统, 开发者可在 VS2015 下面进行开发（推荐使用，不保证其他版本 vs 都兼容）。SDK 采用 C++的动态库 DLL 的方式, 通过提供动态库 DLL、LIB, 头文件的方式进行 SDK 的二次开发。

### 2.1 鉴权

#### 2.1.1 自动批量激活授权

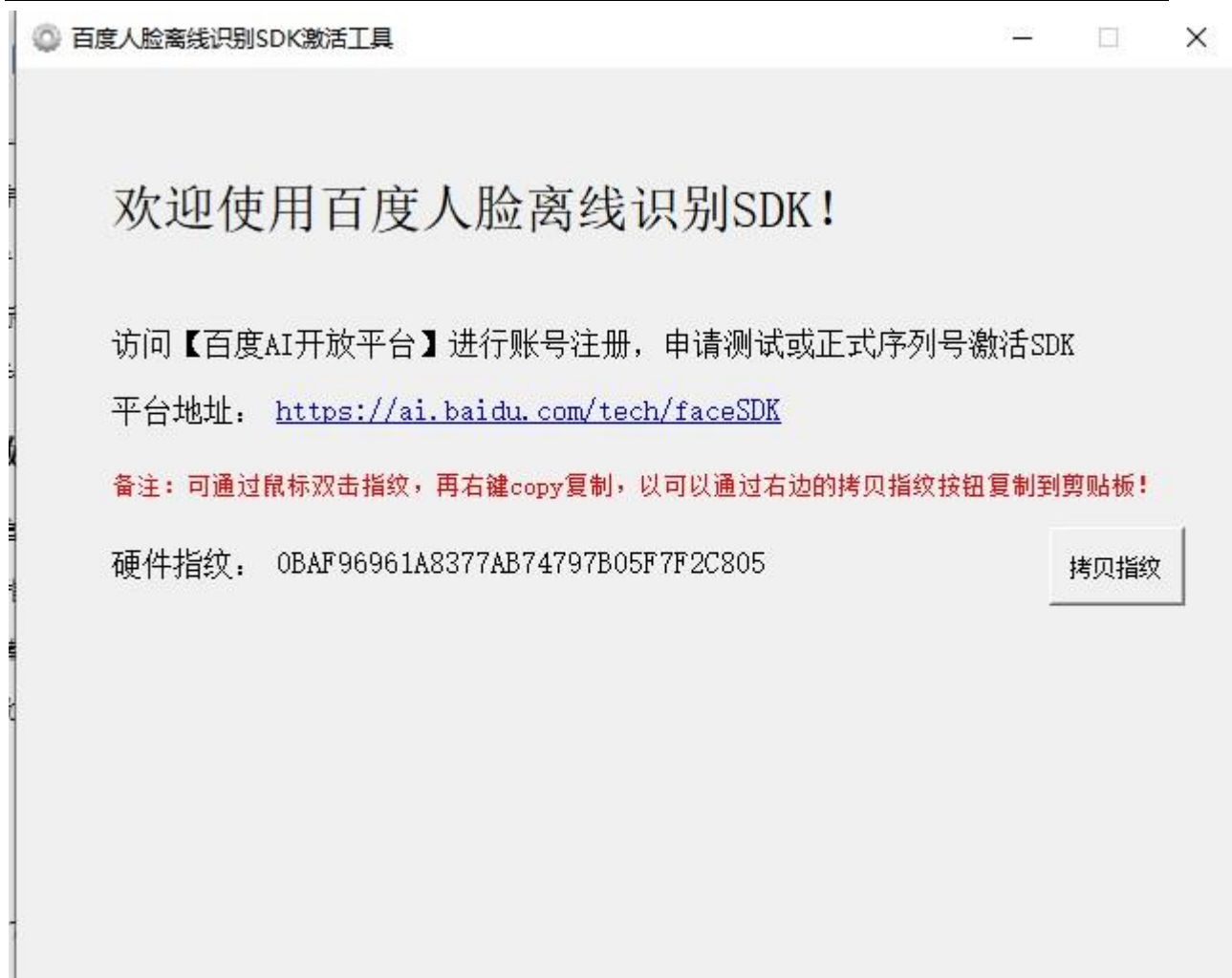
鉴权采用通过自动鉴权的方式实现，可参考 SDK 的示例代码 `IOcrgveEngine::init_license` 这块，通过传入字符串（从 license.key 文件读取内容）和 license.ini 文件的路径，进行在线拉取授权文件激活（license.key 里面的字符串可在百度官网申请，获取后可用获取的字符串替换 license.key 文件里面的内容，license.ini 可参考 SDK 示例命名一个空文件，联网拉取后会自动更新），其中 license.key 和 license.ini 可参考 SDK 示例，放置在如 license 文件夹,license.key 内容修改完毕后，运行 SDK，就会自动更新 license.ini 文件并通过授权。

该方式适应于批量激活（仅仅支持批量的激活序列号）。

#### 2.1.2 官网离线激活授权

鉴权还支持在百度官网进行离线授权激活，通过 SDK 中 tools 文件夹下的 license\_tool，执行里面的 exe，即可打开设备指纹获取工具，点击拷贝，把设备指纹拷贝到百度 AI 官网进行离线激活，激活后把授权 zip 文件下载解压成 license.ini 和 license.key，替换 SDK 中的 license.ini 和 license.key 文件，运行 SDK，即可通过激活。

获取设备指纹的工具如下图所示：



## 2.2 支持平台

SDK 支持 windows7、windows10、windows11 等平台，支持 64 位模式（32 位不支持）。

## 2.3 开发工具

SDK 支持在微软的 VS2015 上编译及运行，不保证在 VS 其他版本上正确运行，建议采用 VS2015 community 或 professional 版本。

## 2.4 依赖库及运行环境

本 SDK 在 VS 环境下编译会在 x64 下生成 exe 可执行文件。依赖的其他 DLL 库文件包括如 opencv 等都在该 exe 目录，请勿删除。若 exe 执行提示找不到依赖 dll 库文件，可在 SDK 的 tools 目录执行 vc\_redist.x64.exe 进行安装。

## 2.5 SDK 包结构

| - - - XingshizhengSdk

--- XingshizhengSdk.sln	vs 工程文件
--- readme.txt	sdk 说明文件
--- include	include 包含文件
--- lib	lib 库文件
--- x64	
--- x64 --- -Release	
--- x64 --- -Release --- -back_images	行驶证副页示例图文件夹
--- x64 --- -Release --- -front_images	行驶证正页示例图文件夹
--- x64 --- -Release --- -license	授权文件夹
--- x64 --- -Release --- -resource	SDK 模型文件夹
--- x64 --- -Release --- *.dll	SDK 依赖库文件
--- x64 --- -Release --- *.exe	SDK 编译的可执行示例文件
--- XingshizhengSDK----main.cpp	SDK demo Main()函数文件

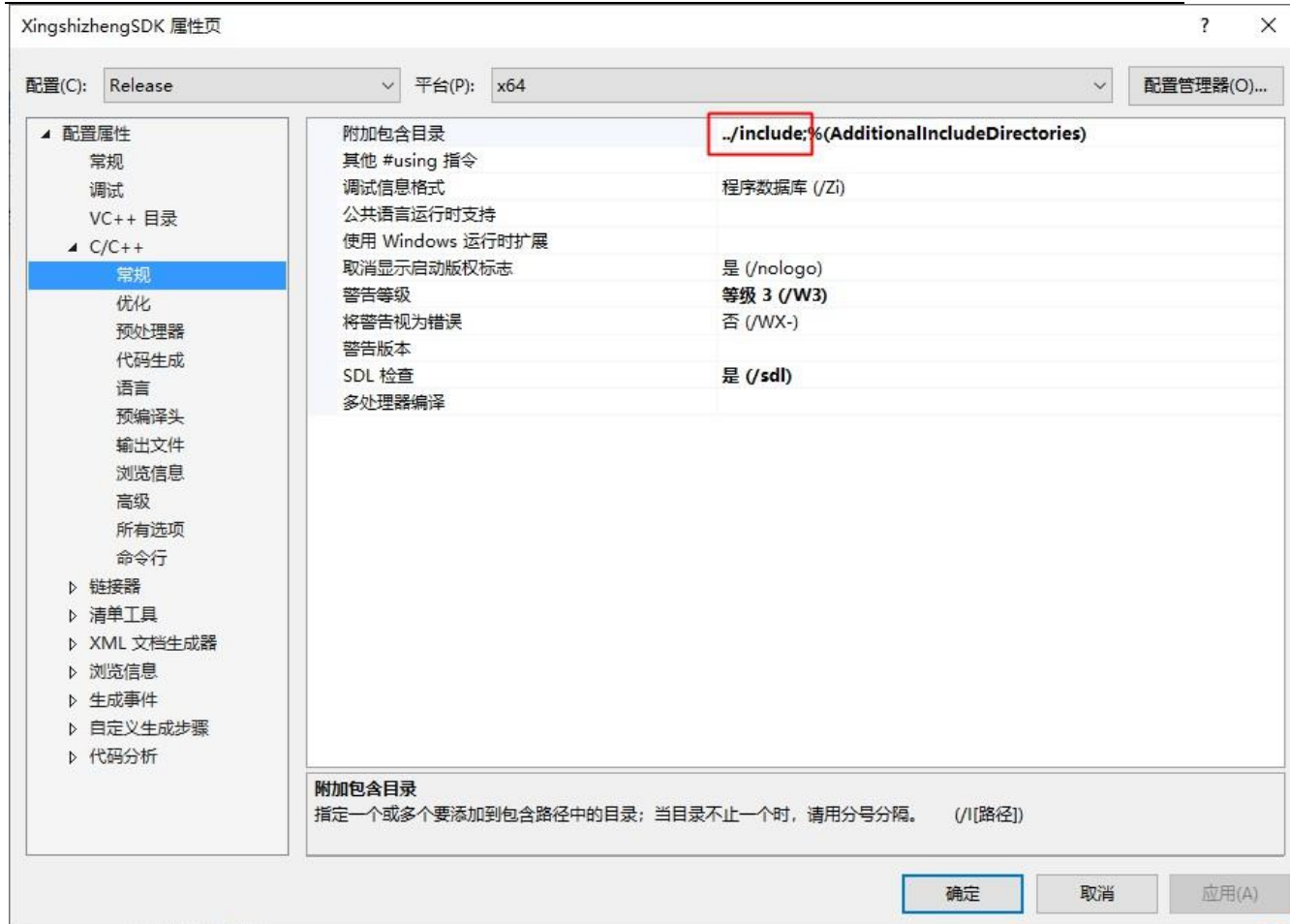
## 3 SDK 接入

如前所述，SDK支持用 VS2015 开发，集成 SDK 可参考示例工程 XingshizhengSDK，示例工程的入口文件为 main.cpp。

集成到项目中需要引入 include 到工程。

引入行驶证识别接口头文件，头文件如图一所示：

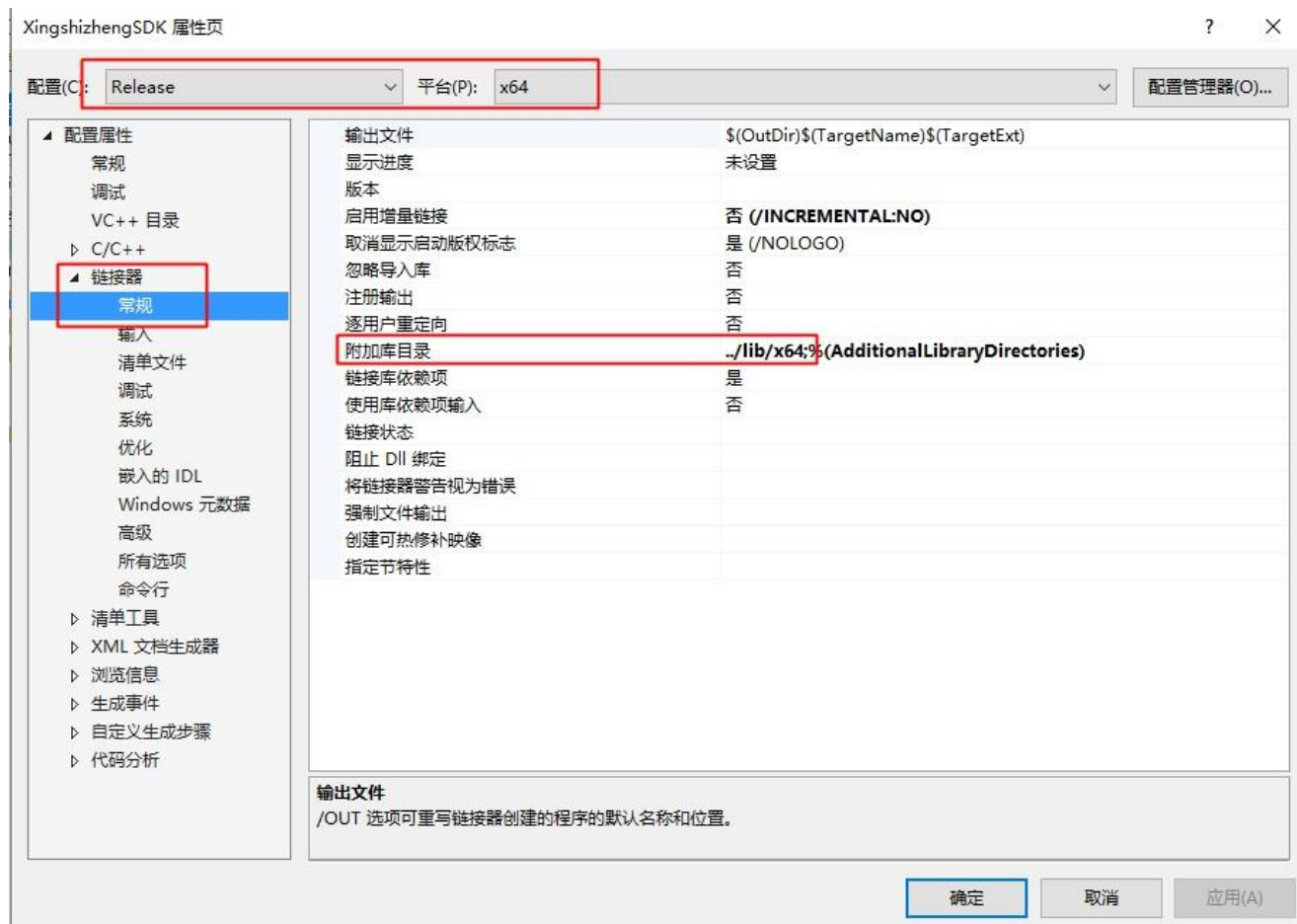
C/C++附加包含目录：



图一

引入 lib 如图二所示：

链接器->常规->附加库目录：

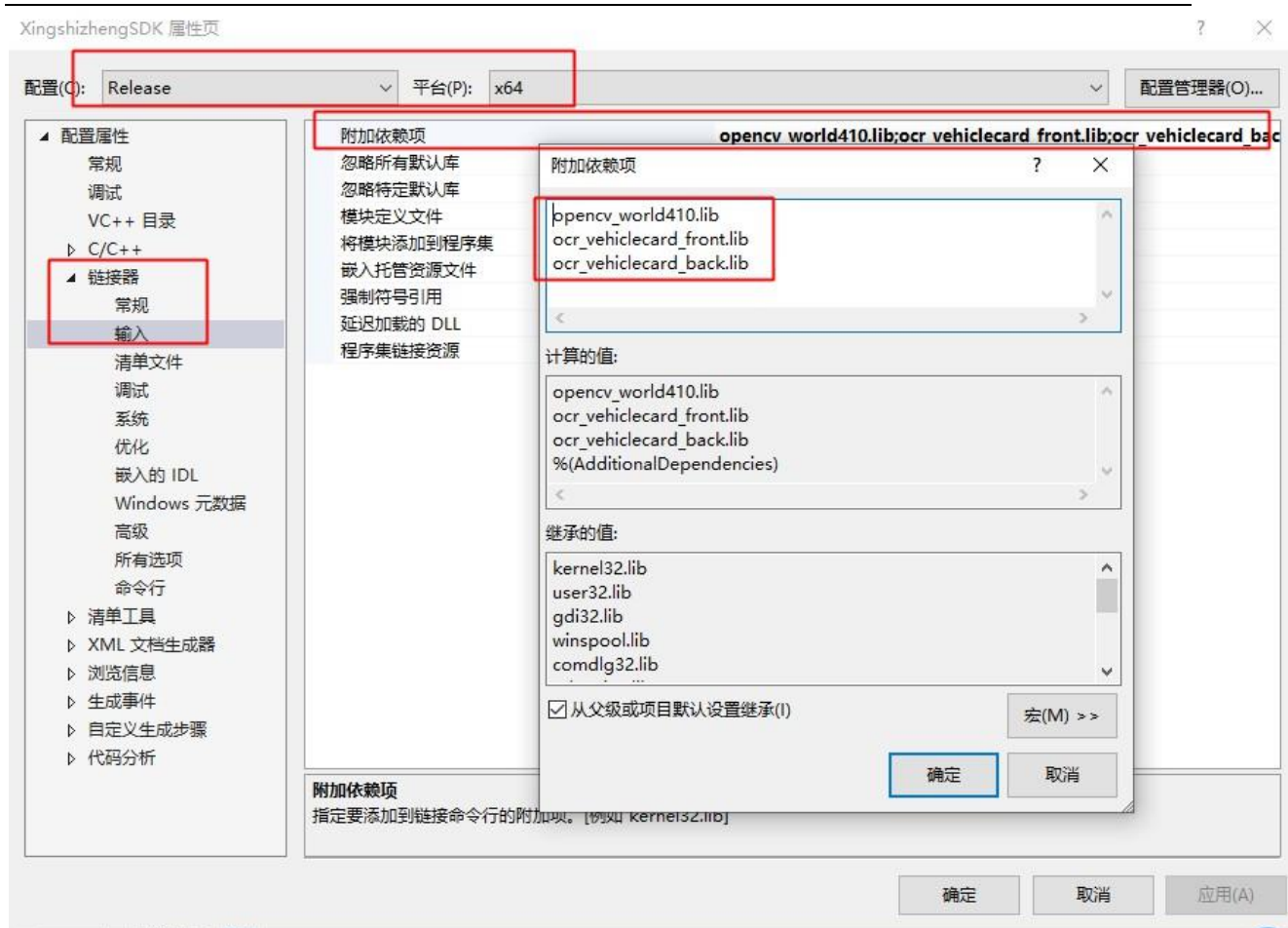


图二

lib 文件如图三所示:

链接器->输入->附加依赖项:





图三

最后，编译出的 exe 所在目录需要放如下图四的圈起来的文件。



















XingshizhengSDK > x64 > Release				
名称	修改日期	类型	大小	
 mkl_dnn.dll	2023/6/7 15:14	应用程序扩展	27,556 KB	
 mklml.dll	2023/6/7 15:14	应用程序扩展	90,478 KB	
 model-encrypt.dll	2021/12/10 19:48	应用程序扩展	54 KB	
 ocr_vehiclecard_back.dll	2023/8/10 15:28	应用程序扩展	843 KB	
 ocr_vehiclecard_front.dll	2023/8/10 15:03	应用程序扩展	1,244 KB	
 onnxruntime.dll	2023/6/7 15:13	应用程序扩展	7,715 KB	
 openblas.dll	2022/8/4 12:31	应用程序扩展	2,540 KB	
 opencv_ffmpeg320_64.dll	2021/11/9 14:57	应用程序扩展	16,990 KB	
 opencv_world320.dll	2021/11/9 14:57	应用程序扩展	40,873 KB	
 opencv_world410.dll	2021/11/17 22:22	应用程序扩展	26,718 KB	
 paddle_inference.dll	2023/2/15 18:16	应用程序扩展	59,231 KB	
 paddle2onnx.dll	2023/2/15 17:56	应用程序扩展	3,808 KB	
 ssleay32.dll	2022/4/18 14:46	应用程序扩展	374 KB	
 vka_license.dll	2021/12/10 19:48	应用程序扩展	90 KB	
 XingshizhengSDK.exe	2023/10/17 20:18	应用程序	81 KB	
 XingshizhengSDK.ioobj	2023/10/17 20:18	IOBJ 文件	3,396 KB	
 XingshizhengSDK.ipdb	2023/10/17 20:18	IPDB 文件	7,097 KB	
 XingshizhengSDK.pdb	2023/10/17 20:18	Program Debug...	3,700 KB	

图 四

sdk 调用分为三步 1.授权 2.初始化 3.识别。

首先需要引入头文件，如示例 main.cpp

接着 init\_license 通过授权。（授权方式见 2.1 鉴权部分描述）

然后：

第一步获取 SDK engine

第二步调用 SDK 识别接口，参考 main.cpp 示例

第三步结束时候销毁 engine；

行驶证副页和正页类似。

## 4 功能接口

百度行驶证 SDK 分为正页及副页 SDK，两个 SDK 结构及接口相似，使用不同的 namespace 区分。

### 4.1 正页

```
namespace vis_vehiclecard_front {

class DLL_API IOcrgveEngine {
public:
    /**
     * @brief 创建Engine实例
     *
     * @return not nullptr : Engine指针
     *         nullptr      : 创建失败
     */
    static IOcrgveEngine* create();

    /**
     * @brief 销毁Engine实例
     * @param p: 要销毁实例指针的地址
     *
     * @return none
     */
    static void destroy(IOcrgveEngine** p);

    /**
     * @brief 设置SDK的日志级别和输出路径
     * @param level: 日志级别
     * @param path : 文件输出路径, 如果为空则表示不输出日志文件
     * @param is_console : 是否控制台输出,
     *                     如果为True, 在Android下为Logcat输出;
     *                     其他平台是终端控制台输出
     *
     * @return OCSuccess : 设置成功
     *         Other      : 错误码
     */
    static void set_log(
        LogLevel level,
        const std::string& path,
        bool is_console);

    /**
     * @brief 初始化授权
     * @param license_key : 授权key
     * @param license_file : 授权文件
     *
     * @return OCSuccess : 设置成功
     *         Other      : 错误码
     */
};
```

```
/**
 * @brief 初始化授权
 * @param license_key : 授权key
 * @param license_file : 授权文件
 *
 * @return OCSuccess : 设置成功
 *         Other      : 错误码
 */
static VISStatus init_license(
    const std::string& license_key,
    const std::string& license_file,
    bool is_remote = false);

/**
 * @brief 初始化Engine
 * @param resource_path : 资源文件目录，包括配置文件、模型等
 *
 * @return OCSuccess : 初始化成功
 *         Other      : 错误码
 */

virtual VISStatus init(const std::string& resource_path) = 0;

/**
 * @brief 逆初始化Engine
 *
 * @return OCSuccess : 逆初始化成功
 *         Other      : 错误码
 */
virtual VISStatus uninit() = 0;

/**
 * @brief 预测 ocr_gvesdk
 *
 * @param frame          : 处理图像帧
 * @param orientation    : 图像方向
 * @param [Out]response  : 返回response
 *
 * @return OCSuccess : 预测成功
 *         Other      : 错误码
 */
virtual VISStatus process_ocr_gvesdk(
    const std::vector<ImageFrame>& frames,
    ImageOrientation orientation,
    std::vector<general_vertical_kv_ret>& response) = 0;
}; // class IOcr_gveEngine

} // namespace vis_vehiclecard_front
```

## 4.2 副页

```
namespace vis_vehiclecard_back {

class DLL_API IOcrgveEngine {
public:
    /**
     * @brief 创建Engine实例
     *
     * @return not nullptr : Engine指针
     *         nullptr      : 创建失败
     */
    static IOcrgveEngine* create();

    /**
     * @brief 销毁Engine实例
     * @param p: 要销毁实例指针的地址
     *
     * @return none
     */
    static void destroy(IOcrgveEngine** p);

    /**
     * @brief 设置SDK的日志级别和输出路径
     * @param level: 日志级别
     * @param path : 文件输出路径, 如果为空则表示不输出日志文件
     * @param is_console : 是否控制台输出,
     *                     如果为True, 在Android下为Logcat输出;
     *                     其他平台是终端控制台输出
     *
     * @return OCSuccess : 设置成功
     *         Other      : 错误码
     */
    static void set_log(
        LogLevel level,
        const std::string& path,
        bool is_console);
};
```

```
/**
 * @brief 初始化授权
 * @param license_key : 授权key
 * @param license_file : 授权文件
 *
 * @return OCSuccess : 设置成功
 *         Other      : 错误码
 */
static VISStatus init_license(
    const std::string& license_key,
    const std::string& license_file,
    bool is_remote = false);

/**
 * @brief 初始化Engine
 * @param resource_path : 资源文件目录，包括配置文件、模型等
 *
 * @return OCSuccess : 初始化成功
 *         Other      : 错误码
 */

virtual VISStatus init(const std::string& resource_path) = 0;

/**
 * @brief 逆初始化Engine
 *
 * @return OCSuccess : 逆初始化成功
 *         Other      : 错误码
 */
virtual VISStatus uninit() = 0;

/**
 * @brief 预测 ocr_gvesdk
 *
 * @param frame          : 处理图像帧
 * @param orientation    : 图像方向
 * @param [Out]response  : 返回response
 *
 * @return OCSuccess : 预测成功
 *         Other      : 错误码
 */
virtual VISStatus process_ocr_gvesdk(
    const std::vector<ImageFrame>& frames,
    ImageOrientation orientation,
    std::vector<general_vertical_kv_ret>& response) = 0;
}; // class IOcr_gveEngine

} // namespace vis_vehiclecard_back
```

## 4.3 正页推理

```
std::vector<vis_vehiclecard_front::general_vertical_kv_ret>
vehiclefront_response;
engine_front->process_ocrgvesdk(frames,
vis_vehiclecard_front::IMAGE_ORIENTATION_UP, vehiclefront_response);
```

## 4.4 副页推理

```
std::vector<vis_vehiclecard_back::general_vertical_kv_ret>
vehicleback_response;
engine_back->process_ocrgvesdk(frames,
vis_vehiclecard_back::IMAGE_ORIENTATION_UP, vehicleback_response);
```

# 5 常见问题

## 5.1 关于 IDE

SDK 推荐使用 vs2015community 或 professional 版本进行开发。若未安装该 ide 想运行 SDK 中的 exe, 可能需要安装 vc 运行环境, 请参考 tools 目录的 vc\_redist 文件夹的 exe 双击安装, 根据平台选择 x64。

## 5.2 激活后是否可以把激活文件 license.ini 和 license.key 拷贝到其他设备运行?

不能, 离线 sdk 和设备绑定, 每个设备对应一个 key 和一个 license 文件, 换设备无法运行。但对同一台设备, 可把 Release 下的 license.ini 和 license.key 拷贝到本电脑的另外 sdk, 该设备也等同于激活, 可以使用。

## 5.3 是否支持 Debug 模式?

只支持 Release 模式, 不支持 debug 模式

## 5.4 SDK 支持 X64 模式。

SDK 仅支持 X64 模式, 支持运行在 win7 或 win10 系统中。