iANPR SDK

Версия 1.7

Документация

СОДЕРЖАНИЕ

Введение

iANPR SDK – это комплект средств разработки для распознавания автомобильных номеров. Основная цель – обеспечить автоматизированное распознавание автомобильных номеров на основе библиотеки компьютерного зрения OpenCV. Возможности библиотеки включают обработку изображений. Основной язык использования библиотеки – C/C++.

Версия 1.7 откомпилирована с версиями OpenCV 3.4.0, при необходимости (по просьбе Клиента для FULL версий) может быть откомпилирована под другую версию.

Виды лицензий

iANPR FREE Данный вид лицензии предназначен для бесплатного использования библиотеки с ограниченными возможностями распознавания: скорость работы искусственно существенно замедлена. Данный вид лицензии можно использовать только в ознакомительных и/или академических целях. Не допускается распространение программного продукта совместно с данной библиотекой.

iANPR RUS PRO LIMITED Это платная лицензия, которая предоставляет все возможности по распознаванию стандартных и транзитных российских номеров. Допускается использование только для собственных нужд внутри организации (или только физическим лицом), принявшей данную лицензию.

iANPR RUS PRO EXTENDED LIMITED Это платная лицензия, которая предоставляет все возможности по распознаванию стандартных и транзитных российских номеров, а также остальных типов номеров, присутствующих в данной версии. Допускается использование только для собственных нужд внутри организации (или только физическим лицом), принявшей данную лицензию.

iANPR RUS PRO FULL Это платная лицензия, которая предоставляет все возможности по распознаванию всех типов номеров Российской Федерации, присутствующих в данной версии. Допускается использование для распространения собственного программного продукта.

iANPR KAZ PRO LIMITED Это платная лицензия, которая предоставляет все возможности по распознаванию частных и номеров организаций Казахстана (только прямоугольные однострочные).

Допускается использование только для собственных нужд внутри организации (или только физическим лицом), принявшей данную лицензию.

iANPR KAZ PRO FULL Это платная лицензия, которая предоставляет все возможности по распознаванию частных и номеров организаций Казахстана (только прямоугольные однострочные). Допускается использование для распространения собственного программного продукта.

iANPR TM PRO LIMITED Это платная лицензия, которая предоставляет все возможности по распознаванию частных и номеров организаций Туркменистана (только прямоугольные однострочные). Допускается использование только для собственных нужд внутри организации (или только физическим лицом), принявшей данную лицензию.

iANPR TM PRO FULL Это платная лицензия, которая предоставляет все возможности по распознаванию частных и номеров организаций Туркменистана (только прямоугольные однострочные). Допускается использование для распространения собственного программного продукта.

iANPR BY PRO LIMITED Это платная лицензия, которая предоставляет все возможности по распознаванию номеров Беларуси. Допускается использование только для собственных нужд внутри организации, принявшей данную лицензию.

iANPR BY PRO FULL Это платная лицензия, которая предоставляет все возможности по распознаванию номеров Беларуси. Допускается использование для распространения собственного программного продукта.

iANPR PL PRO LIMITED Это платная лицензия, которая предоставляет все возможности по распознаванию стандартных номеров Польши. Допускается использование только для собственных нужд внутри организации, принявшей данную лицензию.

iANPR PL PRO FULL Это платная лицензия, которая предоставляет все возможности по распознаванию стандартных номеров Польши. Допускается использование для распространения собственного программного продукта.

iANPR LV PRO LIMITED Это платная лицензия, которая предоставляет все возможности по распознаванию стандартных номеров Латвии. Допускается использование только для собственных нужд внутри организации, принявшей данную лицензию.

iANPR LV PRO FULL Это платная лицензия, которая предоставляет все возможности по распознаванию стандартных номеров Латвии. Допускается использование для распространения собственного программного продукта.

iANPR LT PRO LIMITED Это платная лицензия, которая предоставляет все возможности по распознаванию стандартных номеров Литвы. Допускается использование только для собственных нужд внутри организации, принявшей данную лицензию.

iANPR LT PRO FULL Это платная лицензия, которая предоставляет все возможности по распознаванию стандартных номеров Литвы. Допускается использование для распространения собственного программного продукта.

iANPR EST PRO LIMITED Это платная лицензия, которая предоставляет все возможности по распознаванию стандартных номеров Эстонии. Допускается использование только для собственных нужд внутри организации, принявшей данную лицензию.

iANPR EST PRO FULL Это платная лицензия, которая предоставляет все возможности по распознаванию стандартных номеров Эстонии. Допускается использование для распространения собственного программного продукта.

iANPR UA PRO LIMITED Это платная лицензия, которая предоставляет все возможности по распознаванию номеров Украины. Допускается использование только для собственных нужд внутри организации, принявшей данную лицензию.

iANPR UA PRO FULL Это платная лицензия, которая предоставляет все возможности по распознаванию номеров Украины. Допускается использование для распространения собственного программного продукта.

iANPR MD PRO LIMITED Это платная лицензия, которая предоставляет все возможности по распознаванию номеров Молдовы. Допускается использование только для собственных нужд внутри организации, принявшей данную лицензию.

iANPR MD PRO FULL Это платная лицензия, которая предоставляет все возможности по распознаванию номеров Молдовы. Допускается использование для распространения собственного программного продукта.

iANPR PRO FULL ALL Это платная лицензия, которая предоставляет все возможности по распознаванию номеров библиотеки iANPR для всех поддерживаемых на настоящий момент

стран. Допускается использование для распространения собственного программного продукта.

ARM – данное обозначение характеризует отдельный продукт, на который распространяется тоже лицензионное соглашение, но другие правила ценообразования.

Допускается совмещение лицензий на распознавание номеров разных стран. Подробнее об этом и о ценах на продукцию на странице:

http://intbusoft.com/iANPR/

1. Модули

В версии 1.7 библиотека была поделена на 3 модуля:

- модуль распознавания;
- модуль интерфейсов;
- потоковый модуль.

В модуле распознавания остались базовые функции для распознавания автомобильных номеров.

Модуль интерфейсов предназначен для облегчения доступа к функциям распознавания и предоставляет различные варианты доступа.

Потоковый модуль предназначен для объединения результатов распознавания с нескольких кадров.

1.1. Модуль распознавания – iANPR

В данном модуле (iANPR.h) реализовано распознавание одного изображения.

anprPlate

Функция поиска автомобильных номеров на изображении формата OpenCV.

```
int anprPlate(
    IplImage* Image,
    ANPR_OPTIONS Options,
    int* AllNumber, CvRect* Rects,
    char** Texts,
    void* param = NULL
);
```

Параметры:
Image – входное изображение в формате OpenCV (8-битное 1-канальное или 8-битное 3-канальное в зависимости от параметров Options.type_number);
Options – настройки режима распознавания в формате структуры ANPR_OPTIONS;
AllNumber – количество найденных номеров. До вызова функции должно быть равно количеству выделенных строк памяти **`Texts`**;

Rects – указатель на массив структур CvRect (это структура из библиотеки OpenCV), куда будут записаны зоны нахождения номеров;
Texts – указатель на массив указателей символьного типа, в которые для каждого номера будет возвращаться текст, указатели должны указывать на ранее выделенные области памяти;
param – пока не используется.

Функция anprPlate возвращает 0 при успешном нахождении хотя бы одного номера. 1 – не детектировано ни одного кандидата на автомобильный номер, 2 – не найдено ни одного номера. Помимо этого может быть возвращена одна из следующих ошибок, определенных в iANPRerror.h:

IMAGE_EMPTY (-2) Изображение пустое;

ERROR_TYPE_PLATE (-100) Неподдерживаемый для данной конфигурации тип номера. Например, лицензия iANPR RUS PRO LIMITED не поддерживает флаг типа номера ANPR_RUSSIAN_PUBLIC. Поэтому его использование будет возвращать ошибку.

ERROR_TYPE_FOR_COLOR (-101) Не соответствие типа изображения и флага типа номера в структуре ANPR_OPTIONS.

Структура ANPR_OPTIONS

Данная структура определяет режимы распознавания.

```
struct ANPR_OPTIONS
{
    char sign1 = 'i', sign2 = 'a', sign3 = '1';     //
служебная информация, не должна модифицироваться

    int min_plate_size;  // Минимальная площадь номера
    int max_plate_size;  // Максимальная площадь номера
    int Detect_Mode;     // Режимы детектирования
    int   max_text_size;//   Максимальное   количество
символов номера + 1
    int type_number;     // Тип автомобильного номера
    int flags;           // Дополнительные опции
    void* custom;        // Заполняется только для типа
ANPR_CUSTOM_TYPE, иначе пусто
    char* vers = "1.7"; // Используемая версия iANPR
SDK. Если не задано (vers  = 0), то считается 1.5.
    double alpha = 90.0;  // Поворот вокруг оси X
    double beta = 90.0;   // Поворот вокруг оси Y
    double gamma = 90.0; // Поворот вокруг оси Z
```

```
    int max_threads = 1;  // Количество потоков;
}
```

Минимальная и максимальная площади номеров ограничивают поиск кандидатов на автомобильные номера. Площадь номера определяется произведением ширины номера на его высоту. Если необходимо задавать номера Российской Федерации через ширину, то можно использовать следующий пересчет:

min_plate_size = min_plate_width * min_plate_height;

max_plate_size = max_plate_width * max_plate_height;

Где min_plate_width – минимальная ширина номера, max_plate_width – максимальная ширина номера, min_plate_height – минимальная высота номера, max_plate_height – максимальная высота номера.

Detect_Mode определяет режимы детектирования автомобильного номера. Их можно, даже нужно использовать совместно. В данной версии режимов детектирования 4:

- ANPR_DETECTMODE1,
- ANPR_DETECTMODE2,
- ANPR_DETECTMODE3,
- ANPR_DETECTMODE4.

Они отличаются настройками при поиске номеров и их можно использовать одновременно (при этом правда несколько снижается производительность).

ANPR_DETECTMODE1 – Метод, основанный на детектировании номера целиком с простой адаптивной обработкой изображения.

ANPR_DETECTMODE2 – Метод, основанный на детектировании номера целиком с адаптивной обработкой изображения, основанной на удалении мелких перемычек. Включает в себя практически 100% номеров детектированных с помощью ANPR_DETECTMODE1, а также номера, которые ANPR_DETECTMODE1 не детектируются. Поэтому ANPR_DETECTMODE1 не рекомендуется использовать.

ANPR_DETECTMODE3 – Метод, основанный на детектировании номера целиком с блочной обработкой изображения.

ANPR_DETECTMODE4 – Метод, основанный на выделении частей номера с простой адаптивной обработкой изображения. Не рекомендуется использовать отдельно от других методов, поскольку дает низкие показатели детектированных номеров и не всегда точное детектирование. Однако, его особенности таковы, что он детектирует те номера, которые не детектируются другими методами. *При этом могут в значительном количестве возникать дополнительные ложные срабатывания, например, на плакатах.*

Для качественного распознавания рекомендуется использовать комбинации методов ANPR_DETECTMODE2 и ANPR_DETECTMODE3,

или ANPR_DETECTMODE2 + ANPR_DETECTMODE3 + ANPR_DETECTMODE4, последнюю комбинацию методов можно получить одним флагом ANPR_DETECTCOMPLEXMODE.

Определение в iANPR.h:

```
#define ANPR_DETECTMODE1          0x01
#define ANPR_DETECTMODE2          0x02
#define ANPR_DETECTMODE3          0x04
#define ANPR_DETECTMODE4          0x08
```

Максимальное количество символов номера должно совпадать с максимальным размером, заданным в Texts функции anprPlate. Конечно максимальное количество символов + символ конца строки (0) равно 10, но если поставить больше размер буфера, например, 20, то это ошибкой не будет.

type_number определяет тип номеров для распознавания.

flags определяет дополнительные режимы распознавания. Нужно пока устанавливать 0, а если возникает необходимость выводить номера даже с низкой достоверностью распознавания отдельных символов (в том числе с символами, замененными знаком '?'), то установить флаг DEBUG_RECOGNITION_MODE, который равен 1. Флаг NO_LOW_RELIABILITY предназначен для снижения ложных срабатываний, при этом некоторые номера с низкой достоверностью могут быть отброшены.

Флаг RETURN_TYPE_NUMBER позволяет выводить после распознанного номера через двоеточие его тип. Например X111XX11:0.

Флаг IR_LIGHTING_CAMERA указывает, что для съемки используется инфракрасная камера, то есть на изображении фон номера белый, а цифры чёрные.

В версии 1.7 определены следующие возвращаемые типы:

Российские номера

Тип номера	Код	Описание возвращаемого типа
TYPE_RUSSIAN_BASE	0	Базовый номер России
TYPE_RUSSIAN_TRANSIT	1	Транзитный номер России
TYPE_RUSSIAN_TRAILER	2	Номер прицепа России
TYPE_RUSSIAN_PUBLIC	3	Общественный транспорт России
TYPE_RUSSIAN_POLICE	4	Номер полиции России
TYPE_RUSSIAN_ARMY	5	Военный номер России
TYPE_RUSSIAN_SQUARE_BASE	6	Номер трактора или мотоцикла России
TYPE_RUSSIAN_DIPLOMAT	7	Дипломатический номер

Номера Казахстана

Тип номера	Код	Описание возвращаемого типа
TYPE_KAZ_PRIVATE1993	21	Частные номера стандарта 1993 года
TYPE_KAZ_ORGANIZATION1993	22	Номера организаций стандарта 1993 года

TYPE_KAZ_PRIVATE2012	23	Частные номера стандарта 2012 года
TYPE_KAZ_ORGANIZATION2012	24	Номера организаций стандарта 2012 года

Номера Республики Беларусь

Тип номера	Код	Описание возвращаемого типа
TYPE_BY_2004_BASE	30	Базовый номер стандарта 2004 года
TYPE_BY_TRANSIT	31	Транзитные номера стандарта СТБ 914-99 (тип 12 и 12а в редакции 2011 года)
TYPE_BY_2004_TRAILER	32	Номера прицепов стандарта 2004 года
TYPE_BY_PUBLIC	33	Номера общественного транспорта
TYPE_BY_POLICE	34	Номера МВД
TYPE_BY_ARMY	35	Военные номера
TYPE_BY_SQUARE_BASE	36	Задние базовые двухстрочные номера стандарта 2004 года
TYPE_BY_2004_TRUCK	37	Номера грузовых автомобилей и автобусов стандарта 2004 года
TYPE_BY_1992_TRUCK	38	Номера грузовых автомобилей и автобусов стандарта 1992 года

Номера Польши

Тип номера	Код	Описание возвращаемого типа
TYPE_PL_BASE	40	Базовые номера стандарта 2000 и 2006 года

Номера Латвии

Тип номера	Код	Описание возвращаемого типа
TYPE_LV_BASE	50	Базовые номера стандарта 1992 и 2004 года

Номера Литвы

Тип номера	Код	Описание возвращаемого типа
TYPE_LT_BASE	60	Базовые номера стандарта 2004 года

Номера Эстонии

Тип номера	Код	Описание возвращаемого типа
TYPE_EST_BASE	70	Базовые номера стандарта 2004 года

Номера Украины

Тип номера	Код	Описание возвращаемого типа
TYPE_UA_BASE	80	Базовые номера стандарта 2015 года (легковые автомобили, прицепы, автобусы)
TYPE_UA_TRANSIT	81	Номера для разовых поездок на автомобилях, прицепах к ним и автобусах стандарта 2015 года
TYPE_UA_DIPLOMAT	82	Номера дипломатов и техперсонала стандарта 2013 года

Номера Молдовы

Тип номера	Код	Описание возвращаемого типа
TYPE_MD_2011_BASE	90	Номера физических и юридических лиц стандарта 2011-2015 года
TYPE_MD_2011_TRAILER	91	Номера прицепов и полуприцепов стандарта 2011-2015 года
TYPE_MD_2011_SQUARE_BASE	92	Номера мототехники стандарта 2011-2015 года

Нужно помнить, чтобы размеры буфера для текста были больше номера с возвращаемым типом.

Типы распознаваемых номеров

Типы распознавания Российских номеров

Тип номера	Код	Тип изображения	Описание поддерживаемых номеров
ANPR_RUSSIAN_BASE	0	8bit, 1 channel	Базовые
ANPR_RUSSIAN_BASE2	1	8bit, 1 channel	Базовые и транзитные
ANPR_RUSSIAN_EXTENDED	2	8bit, 1 channel	Базовые, транзитные и номера прицепов
ANPR_RUSSIAN_PUBLIC	3	8bit, 1 channel	Только общественный транспорт
ANPR_RUSSIAN_POLICE	5	8bit, 1 channel	Только номера полиции
ANPR_RUSSIAN_ARMY	6	8bit, 1 channel	Только военные номера
ANPR_RUSSIAN_EXTENDED2	4	8bit, 3 channel	ANPR_RUSSIAN_EXTENDED + ANPR_RUSSIAN_PUBLIC
ANPR_RUSSIAN_FULL	7	8bit, 3 channel	ANPR_RUSSIAN_EXTENDED2 + ANPR_RUSSIAN_POLICE + ANPR_RUSSIAN_ARMY + ANPR_RUSSIAN_DIPLOMAT
ANPR_RUSSIAN_SQUARE_BASE	8	8bit, 1 channel	Только квадратные номера
ANPR_RUSSIAN_FULL_WITH_SQUARE	9	8bit, 3 channel	ANPR_RUSSIAN_FULL + ANPR_RUSSIAN_SQUARE_BASE
ANPR_RUSSIAN_DIPLOMAT	10	8bit, 1 channel	

8bit, 1 channel – изображение в градациях серого; 8bit, 3 channel – цветное изображение.

Базовые номера [ГОСТ Р 50577-93]:

Рисунок А.1 – Регистрационный знак типа 1 с двухзначным кодом региона регистрации

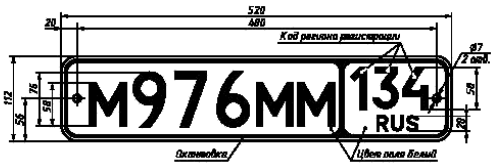


Рисунок А.2 – Регистрационный знак типа 1 с трехзначным кодом региона регистрации

Транзитные российские номера (ММ976М34).

Номера прицепов:

Рисунок А.5 – Регистрационный знак типа 2

Общественный транспорт – желтые номера формата ММ11122.
Номера полиции – синие номера формата М111122.
Военные номера –черные номера формата 1111ММ22.
Квадратные номера мотоциклов и тракторов типа:
1111
ММ22

Типы распознавания номеров Казахстана

Тип номера	Код	Тип изображения	Описание поддерживаемых номеров
ANPR_KAZ_1993_PRIVATE	100	8bit, 1 channel	Частные номера стандарта 1993 года
ANPR_KAZ_1993_ORGANIZATION	101	8bit, 1 channel	Номера организаций стандарта 1993 года
ANPR_KAZ_2012_PRIVATE	102	8bit, 1 channel	Частные номера стандарта 2012 года
ANPR_KAZ_2012_ORGANIZATION	103	8bit, 1 channel	Номера организаций стандарта 2012 года
ANPR_KAZ_BASE	104	8bit, 1 channel	Частные и номера организации стандартов 1993 и 2012

Типы распознавания номеров Туркменистана

Тип номера	Код	Тип изображения	Описание поддерживаемых номеров

ANPR_TM_2009	201	8bit, 1 channel	Частные номера стандарта 2009 года
ANPR_TM_PRIVATE_BEFORE_2009	202	8bit, 1 channel	Частные номера стандарта до 2009 года
ANPR_TM_BASE	203	8bit, 1 channel	Все частные номера

Типы распознавания номеров Республики Беларусь

Тип номера	Код	Тип изображения	Описание поддерживаемых номеров
ANPR_BY_TRUCK	300	8bit, 1 channel	Все грузовые номера
ANPR_BY_2004_TRUCK	301	8bit, 1 channel	Грузовые номера стандарта 2004 года
ANPR_BY_1992_TRUCK	302	8bit, 1 channel	Грузовые номер стандарта 1992 года
ANPR_BY_2004_TRAILER	303	8bit, 1 channel	Задний знак для прицепов и полуприцепов с 2004 года
ANPR_BY_2004_BASE	304	8bit, 1 channel	Легковые автомобили, начиная с 2004 года
ANPR_BY_SQUARE_BASE	305	8bit, 1 channel	Задние двухстрочные знаки, начиная с 2004 года
ANPR_BY_2004_BASE2	306	8bit, 1 channel	ANPR_BY_2004_TRUCK + ANPR_BY_2004_BASE + ANPR_BY_2004_TRAILER
ANPR_BY_TRANSIT	307	8bit, 1 channel	Транзитные номера стандарта СТБ 914-99 (тип 12 и 12а в редакции 2011 года)
ANPR_BY_PUBLIC	308	8bit, 1 channel	Общественный транспорт
ANPR_BY_POLICE	309	8bit, 1 channel	Номера МВД
ANPR_BY_FULL	310	8bit, 3 channel	ANPR_BY_2004_BASE2 + ANPR_BY_TRANSIT + ANPR_BY_PUBLIC + ANPR_BY_POLICE
ANPR_BY_FULL_WITH_SQUARE	311	8bit, 3 channel	ANPR_BY_FULL + ANPR_BY_SQUARE_BASE
ANPR_BY_2004_BASE3	312	8bit, 1 channel	ANPR_BY_2004_BASE2 + ANPR_BY_TRANSIT

Типы распознавания номеров Польши

Тип номера	Код	Тип изображения	Описание поддерживаемых номеров
ANPR_PL_BASE	400	8bit, 1 channel	Стандартные номера Польши ANPR_PL_BASE_7 + ANPR_PL_BASE_8

ANPR_PL_BASE_7	401	8bit, 1 channel	Стандартные номера Польши 7 символов
ANPR_PL_BASE_8	402	8bit, 1 channel	Стандартные номера Польши 8 символов

Типы распознавания номеров Латвии

Тип номера	Код	Тип изображения	Описание поддерживаемых номеров
ANPR_LV_BASE	500	8bit, 1 channel	Стандартные номера 1992 и 2004 года

Типы распознавания номеров Литвы

Тип номера	Код	Тип изображения	Описание поддерживаемых номеров
ANPR_LT_BASE	600	8bit, 1 channel	Стандартные номера с 2004 года

Типы распознавания номеров Эстонии

Тип номера	Код	Тип изображения	Описание поддерживаемых номеров
ANPR_EST_BASE	700	8bit, 1 channel	Стандартные номера 2004 года

Типы распознавания номеров Украины

Тип номера	Код	Тип изображения	Описание поддерживаемых номеров
ANPR_UA_BASE	800	8bit, 1 channel	Стандартные номера 2015 года (автомобили, прицепы, автобусы)
ANPR_UA_TRANSIT	801	8bit, 1 channel	Номера для разовых поездок на автомобилях, прицепах к ним и автобусах, 2015 год
ANPR_UA_DIPLOMAT	802	8bit, 1 channel	Номера дипломатов и техперсонала, 2013 год
ANPR_UA_BASE2	803	8bit, 1 channel	ANPR_UA_BASE + ANPR_UA_DIPLOMAT
ANPR_UA_BASE3	804	8bit, 1 channel	ANPR_UA_BASE2 + ANPR_UA_TRANSIT

Типы распознавания номеров Молдовы

Тип номера	Код	Тип изображения	Описание поддерживаемых номеров
ANPR_MD_2011_BASE	900	8bit, 1 channel	Номера физических и юридических лиц 2011-2015 года + номера такси
ANPR_MD_2011_TRAILER	901	8bit, 1 channel	Номера прицепов и полуприцепов 2011-2015 года
ANPR_MD_2011_SQUARE_BASE	902	8bit,	Номера мототехники 2011-

		1 channel	2015 года
ANPR_MD_2011_BASE2	903	8bit, 1 channel	ANPR_MD_2011_BASE + ANPR_MD_2011_TRAILER
ANPR_MD_2011_FULL_WITH_SQUARE	911	8bit, 3 channel	ANPR_MD_2011_BASE2 + ANPR_MD_2011_SQUARE_BASE

Тип ANPR_CUSTOM_TYPE

Данный тип является настраиваемым типом пользователя и имеет значение -1. При этом поле custom в структуре должно быть заполнено. Для всех остальных типов распознавания, поле пустое. Изображение на вход может подаваться как полноцветное, так и серое.

Настраиваемый тип определяется в iANPRCustom.h следующим образом:

```
const int max_ianpr_custom_in_types = 20;
struct iANPRCustomElement
{
	int all_types;
	float probability; // Вероятность страны
	int types[max_ianpr_custom_in_types];// Типы берутся в iANPR
	float	probability_types[max_ianpr_custom_in_types];	// Вероятность каждого типа, можно не заполнять
};
struct iANPRCustom
{
	int all_countries;
	iANPRCustomElement* Elements;
	int flags; // По умолчанию 0
};
```

В текущей версии вероятность каждого типа не работает. Вероятность страны по сути не вероятность, а значимость. Т.е. полученная вероятность номера умножается на значение значимости. Возьмем, например, Оренбургскую область, где попадаются номера из Казахстана довольно часто. Определим уровень значимости для России 1.2, а для Казахстана 1. Тогда если идентифицировался номер из России, тогда его вероятность будет умножена на 1.2. Если в тоже время определился номер из Казахстана, то его вероятность будет умножена на 1. Т.е. номер из России значимее из Казахстана. Можно считать это некоторой формой априорной вероятности.

Если вы пишите программу на языке C/C++, то можете заполнить структуру самостоятельно, например, так:

```
iANPRCustom* custom = NULL;
custom = new iANPRCustom;
custom->all_countries = 2;
custom->Elements = new iANPRCustomElement[2];
custom->Elements[0].all_types = 5;
custom->Elements[0].probability = 1.2f;
custom->Elements[0].types[0] = CUSTOM_RUSSIAN_BASE_EXTENDED;
custom->Elements[0].types[1] = CUSTOM_RUSSIAN_PUBLIC;
custom->Elements[0].types[2] = CUSTOM_RUSSIAN_POLICE;
custom->Elements[0].types[3] = CUSTOM_RUSSIAN_ARMY;
custom->Elements[0].types[4] = CUSTOM_RUSSIAN_SQUARE;
custom->Elements[1].all_types = 1;
custom->Elements[1].probability = 1.0f;
custom->Elements[1].types[0] = CUSTOM_KAZ_2012_PRIVATE;
```

Однако в этом случае нужно помнить, что удалить выделенные элементы вы должны самостоятельно. Для других языков программирования можно использовать функции создания и удаления из iANPRCustom.h:

void* CreateiANPRCustom(char* xml_buffer, int buffer_size);

void DeleteiANPRCustom(void* xml);

Пример xml файла, в котором записана информация, указанная в коде выше, здесь:

```
<?xml version="1.0"?>
  <countries value="2">
    <country all_types="5" probability="1.2" comment="First
country - Russia">
      <type value="2"/>
      <type value="3"/>
      <type value="5"/>
      <type value="6"/>
      <type value="8"/>
    </country>
    <country all_types="1" probability="1" comment="Second
country - Kazakhstan">
      <type value="102"/>
    </country>
  </countries>
```

Флаги в структуре iANPRCustom предназначены для дополнительных возможностей. На настоящий момент поддержан только один флаг FLAG_CUSTOM_MULTI_RESULT, который позволяет возвращать не один наиболее вероятный номер, а обнаруженные вероятные номера не более 3. Причем первый номер наиболее вероятен. Возвращаемые номера находятся в той же текстовой строке, поэтому ее необходимо сделать большего размера (не менее 40 байт), номера будут возвращены в следующем виде:

Номер 1|Номер 2|Номер 3

Для прописывание флага в xml, нужно изменить строку с количеством стран так:

```
<countries value="2" multi="1">
```

Поворот входного изображения (alpha, beta, gamma)

iANPR SDK предполагает: если автомобильный номер на изображении повёрнут, то на малый угол (подробнее в разделе требований к алгоритму распознавания). Для корректировки значительного угла наклона можно использовать параметры alpha, beta, gamma из IANPR_OPTIONS. Alpha задаёт поворот вокруг оси X, beta задаёт поворот вокруг оси Y, gamma задаёт поворот вокруг оси Z. Значение 90.0 означает отсутствие поворота. Для определения углов поворота вдоль осей воспользуйтесь утилитой persptrans, которую можно найти в составе iANPR SDK.

anprPlateRect

Функция поиска автомобильных номеров на регионе изображения формата OpenCV.

```
int anprPlateRect(
    IplImage* Image,
    CvRect Rect,
    ANPR_OPTIONS Options,
    int* AllNumber,
    CvRect* Rects,
    char** Texts,
    void* param = NULL
);
```

Параметры аналогичны функции **anprPlate**, дополнительный параметр **Rect** определяет обрабатываемую на изображении область.

Возвращаемые значения такие же, но добавляется ошибка:
ERROR_RECT (-1) – неправильно заданная область.

LicenseValue

Функция активации лицензионной версии iANPR. Вызов этой функции необходим только для лицензионной версии библиотеки и только один раз перед первым распознаванием. В случае использования функции LicenseValue с демо-версией iANPR, она никак не повлияет на работу программы.

```
void LicenseValue(
    char* lic
);
```

Параметры:
lic – массив, содержащий лицензионный ключ.

Функция не возвращает значений.

1.2. Модуль интерфейсов – iANPRinterface

Модуль интерфейсов расширяет возможности подключения к библиотеке. Определения функций представлены в iANPRinterface.h.

anprPlateMemory

Функция предназначена для распознавания графического файла форматов BMP, JPEG, PNG, TIFF форматов, который находится в памяти. К примеру, с жесткого диска в память читается BMP файл, а функции передается указатель на него.

```
int anprPlateMemory(
    char* in_buffer,
    int size_buffer,
    ANPR_OPTIONS Options,
    int* AllNumber,
    CvRect* Rects,
    char** Texts
);
```

Параметры:
in_buffer – указатель на входное изображение;
size_buffer – размер буфера изображения;
Остальные параметры аналогичны функции **anprPlate**.

Возвращаемые значения такие же, как и в **anprPlate**.

anprPlateMemoryRect

Назначение функции аналогично **anprPlateMemory**, только также как и в **anprPlateRect** добавляется область поиска.

```
int anprPlateMemoryRect(
    char* in_buffer,
```

```
    int size_buffer,
    CvRect Rect,
    ANPR_OPTIONS Options,
    int* AllNumber,
    CvRect* Rects,
    char** Texts
);
```

Назначение параметров и возвращаемых значений аналогично **anprPlateMemory** и **anprPlateRect**.

anprPlateMat

Функция аналогичная **anprPlate**, за исключением того, что первый параметр – это изображение в формате cv::Mat C++ интерфейса изображений в OpenCV.

```
int anprPlateMat(
    cv::Mat Image,
    ANPR_OPTIONS Options,
    int* AllNumber,
    CvRect* Rects,
    char** Texts
);
```

Назначение параметров и возвращаемых значений аналогично **anprPlate**.

anprPlateMatRect

Назначение функции аналогично **anprPlateMat**, только также как и в **anprPlateRect** добавляется область поиска.

```
int anprPlateMatRect(
    cv::Mat Image,
    CvRect Rect,
    ANPR_OPTIONS Options,
    int* AllNumber,
    CvRect* Rects,
    char** Texts
);
```

Назначение параметров и возвращаемых значений аналогично **anprPlate**.

anprPlateXML

Функция, аналогичная **anprPlate**, за исключением того, что найденные номера возвращаются в формате XML.

```
int anprPlateXML(
     IplImage* Image,
     ANPR_OPTIONS Options,
     char* xml_buffer,
     int* size_xml_buffer
);
```

Параметры:
Image – входное изображение в формате OpenCV (8-битное 1-канальное или 8-битное 3-канальное в зависимости от параметров Options.type_number);
Options – настройки режима распознавания в формате структуры ANPR_OPTIONS;
xml_buffer – указатель на выделенный в памяти перед вызовом функции буфер, в который будет возвращаться XML-строка следующего формата:

```
<?xml version="1.0" encoding="windows-1251"?>
<action_result version='1.0'>
     <allnumbers value='1'>
          <number value='M976MM134' coord='243,256,177,53'>
          </number>
     </allnumbers>
</action_result>
```

allnumbers показывает количество найденных автомобильных номеров. А для каждого найденного номера возвращается тег number, у которого value – текст номера, а coord – его координаты (X,Y,ширина, высота).
size_xml_buffer – размер выделенного буфера, после вызова функции будет содержать размер записанной XML-строки/

Возвращаемые значения такие же, как и в **anprPlate**, добавляется возможная ошибка:
ERROR_SIZE_XML_BUF (-3) Недостаточный размер буфера XML.

anprPlateRectXML, anprPlateMemoryXML, anprPlateMemoryRectXML, anprPlateMatXML, anprPlateMatRectXML

Варианты предыдущих функций с возвращением параметров через XML.

```
int anprPlateRectXML(
    IplImage* Image,
    CvRect Rect,
    ANPR_OPTIONS Options,
    char* xml_buffer,
    int* size_xml_buffer
);

int anprPlateMemoryXML(
    char* in_buffer,
    int size_buffer,
    ANPR_OPTIONS Options,
    char* xml_buffer,
    int* size_xml_buffer
);

int anprPlateMemoryRectXML(
    char* in_buffer,
    int size_buffer,
    CvRect Rect,
    ANPR_OPTIONS Options,
    char* xml_buffer,
     int* size_xml_buffer
);

int anprPlateMatXML(
    cv::Mat Image,
    ANPR_OPTIONS Options,
    char* xml_buffer,
    int* size_xml_buffer
);

int anprPlateMatRectXML(
    cv::Mat Image,
    CvRect Rect,
    ANPR_OPTIONS Options,
    char* xml_buffer,
    int* size_xml_buffer
);
```

Параметры и возвращаемые значения аналогично вышеописанным функциям.

1.3. Потоковый модуль - iANPRcapture

Потоковый модуль предназначен для повышения достоверности распознавания автомобильных номеров на видеопотоке за счет объединения результатов распознавания из нескольких кадров.

Функции данного модуля определены в iANPRcapture.h. ВНИМАНИЕ! Мультитип (несколько номеров для ANPR_CUSTOM_TYPE) на настоящий момент нельзя использовать.

CreateiANPRCapture

Создание iANPR-потока.

```
iANPRCapture CreateiANPRCapture(
    int max_frames,
    ANPR_OPTIONS Options,
    CvRect Rect
);
```

Параметры:
max_frames – количество кадров, из которых объединяется результат;
Options – настройки распознавания (заполненная структура ANPR_OPTIONS);
Rect – область распознавания в формате CvRect.

Возвращает выделенный в памяти объект iANPRCapture. Если создать не удастся, то возвращает NULL.

ReleaseiANPRCapture

Очистка памяти от объекта iANPRCapture.

```
void ReleaseiANPRCapture(
    iANPRCapture *Capture
);
```

Параметры:
Capture – указатель на объект iANPRCapture.

AddFrameToiANPRCapture

Функция добавляет текущий кадр в поток iANPRCapture и возвращает распознанные автомобильные номера.

```
int AddFrameToiANPRCapture(
    iANPRCapture Capture,
    IplImage* Image,
    int* AllNumber,
    CvRect* Rects,
    char** Texts
);
```

Параметры:
Capture – объект iANPRCapture.
Image – входное изображение.
Остальные параметры аналогичны **anprPlate**.

Возвращаемые значения аналогичны **anprPlate**.

AddFrameToiANPRCaptureMat

Функция добавляет текущий кадр (cv::Mat) в поток iANPRCapture и возвращает распознанные автомобильные номера.

```
int AddFrameToiANPRCaptureMat(
    iANPRCapture Capture,
    cv::Mat Mat,
    int* AllNumber,
    CvRect* Rects,
    char** Texts
);
```

Параметры аналогичны AddFrameToiANPRCapture за тем исключением, что текущий кадр передаётся в структуре cv::Mat.

Возвращаемые значения аналогичны **anprPlate**.

CreateMemoryForiANPRCapture

Функция выделяет память для дополнительной функциональности потокового распознавания. Необходимо вызывать только если предполагается использование далее функции **GetNumbersInMemory**.

```
int CreateMemoryForiANPRCapture(
    iANPRCapture Capture,
    int min_frames_with_plate,
```

```
    int frames_without_plate,
    int max_plates_in_mem
);
```
Параметры:
Capture – объект iANPRCapture.
min_frames_with_plate – количество кадров между первым и последним распознанным номером, после которого можно считать, что это действительно номер.
frames_without_plate – количество кадров без распознанного ранее номера, после которого можно выдавать результат.
max_plates_in_mem – максимальное количество номеров в памяти.

Функция возвращает 0 при успешном вызове, иначе ошибка.

GetNumbersInMemory

Возвращает суммированный результат нахождения номера из памяти. Он возвращается только после frames_without_plate кадров, установленных в **CreateMemoryForiANPRCapture**.

```
int GetNumbersInMemory(
    iANPRCapture Capture,
    int* AllNumber,
    char** Texts,
    int Size_Texts,
    CvPoint* Points,
    int* all_point
);
```
Параметры:
Capture – объект iANPRCapture.
AllNumber – возвращается количество распознанных номеров на последний кадр из памяти (первоначально содержит размер буфера (количество номеров) Texts).
Texts – содержимое номера.
Size_Texts – размер каждого элемента (строки) массива **Texts**.
Points – указатель на выделенный перед вызовом массив CvPoint для траектории (если NULL, то траектория не возвращается).
all_points – сюда передается размер массива **Points**, возвращается количество найденных точек, а если массив меньше (рекомендуется 1000 элементов), чем реально найдено точек, то будет возвращено только то количество точек, для которых есть место.

Функция возвращает 0 при успешном вызове, иначе ошибка. Если возвращаются одновременно 2 номера в кадре, что маловероятно, по причине суммирования и отсрочки, то будет возвращаться траектория только для первого.

CreateLineIntersection

Функция создает линию для фиксации пересечения. На самом деле линия состоит из двух линий – и пересечение фиксируется только тогда, когда пересекаются обе (сделано для исключения ложных срабатываний).

```
int CreateLineIntersection(
    iANPRCapture Capture,
    CvPoint p1a,
    CvPoint p2a,
    CvPoint p1b,
    CvPoint p2b
);
```

Параметры:
Capture – объект iANPRCapture.
p1a и **p2a** – точки характеризующие верхнюю линию (линия не может быть вертикальной – максимум дельта x может быть в 3 раза меньше дельта y. Иначе ошибка ERROR_SLOPE_LINE).
p1b и **p2b** – точки характеризующие нижнюю линию.

Функция возвращает 0 при успешном вызове, иначе ошибка. Линии должны быть параллельны друг другу. Иначе ERROR_NO_PARALLEL_LINES.

LicenseCapture

Функция активации лицензионной версии iANPRcapture. Вызов этой функции необходим только для лицензионной версии библиотеки и только один раз перед первым распознаванием. В случае использования функции LicenseCapture с демо-версией iANPRcapture, она никак не повлияет на работу программы.

```
void LicenseCapture(
    char* lic
);
```

Параметры:
lic – массив, содержащий лицензионный ключ.

Функция не возвращает значений.

2. Инсталляция и использование

Каких-либо особых требований к инсталляции не существует.

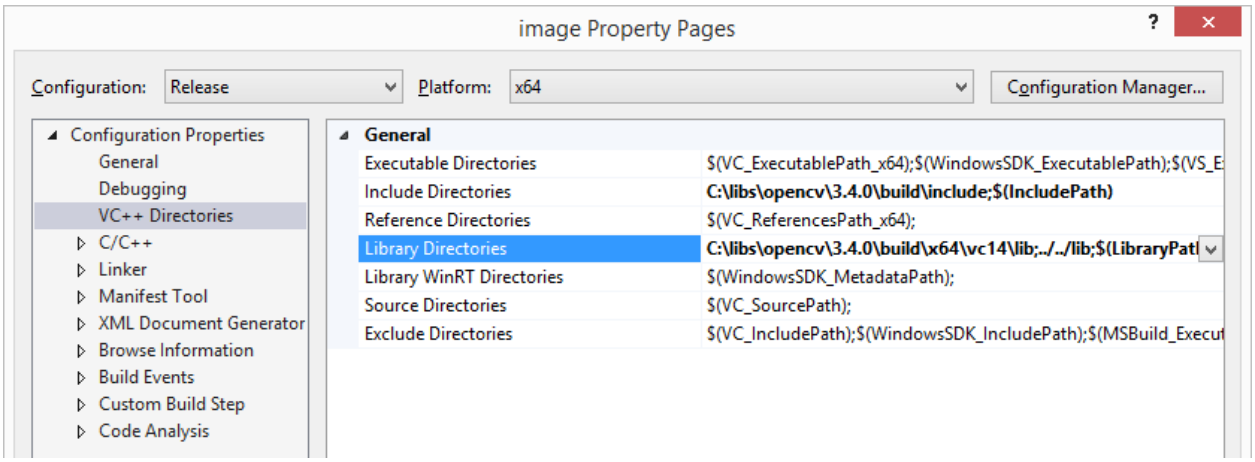
2.1. Windows

Для того, чтобы iANPR SDK заработало, на компьютер необходимо установить:
Для Microsoft Visual C++ 2015 Redistributable Package https://www.microsoft.com/ru-ru/download/details.aspx?id=48145

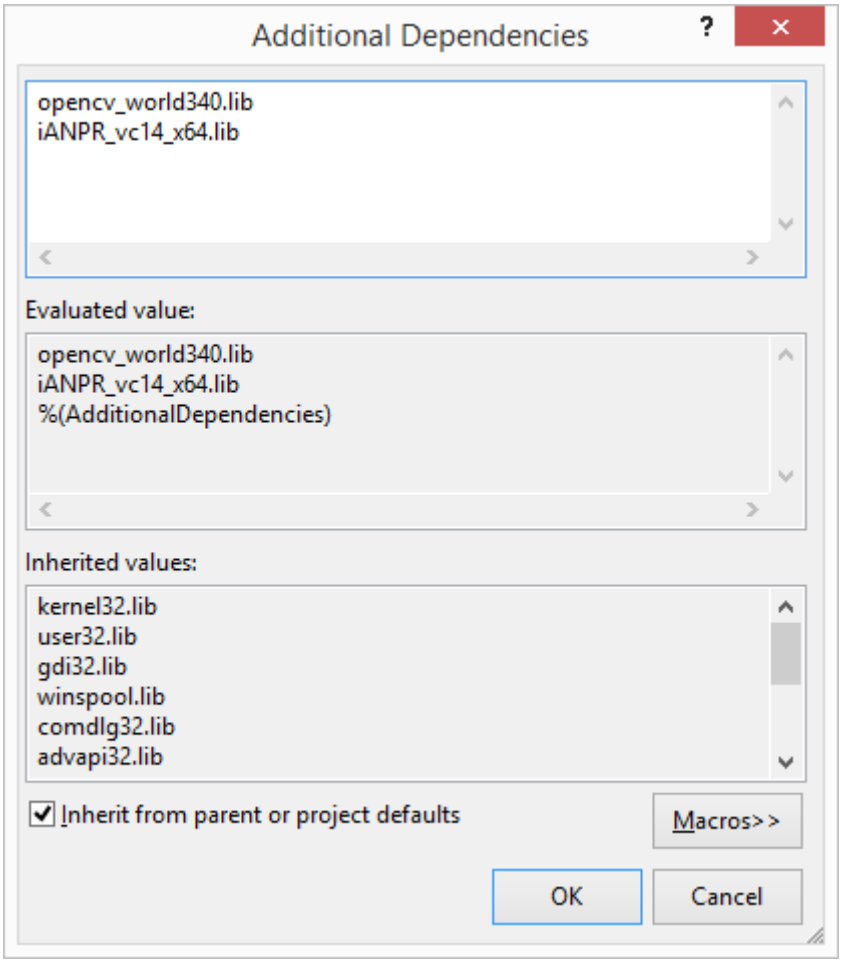
Скачайте OpenCV 3.4.0
https://sourceforge.net/projects/opencvlibrary/files/opencv-win/3.4.0/opencv-3.4.0-vc14_vc15.exe/download
Хотя нужные библиотеки OpenCV идут вместе с iANPR SDK, но для разработки программ вам вероятно понадобится подключение заголовочных файлов. В iANPR использовались библиотеки, откомпилированные vc14.

Далее осуществляете подключение в виде обычных динамических библиотек.

Если вы реализуете проект на C/C++ на Visual Studio, то пропишите пути до h и lib для OpenCV и места, где находится iANPR:



Добавьте подключаемые библиотеки (в свойствах Компоновщика):



После этого обратите внимание, чтобы все dll из папки x86 или x64 находились или в папке с вашим исполняемым файлом, или в папке, прописанной в переменной PATH.

2.2. Linux

Использованная ОС – Ubuntu 16.04 desktop i386

1. Руководствуясь

http://docs.opencv.org/doc/tutorials/introduction/linux_install/linux_install.html

устанавливаем дополнительное программное обеспечение для OpenCV.

2. Скачиваем версию OpenCV 3.4.0 для linux отсюда

http://sourceforge.net/projects/opencvlibrary/

И компилируем ее.

3. Скачиваем и распаковываем iANPR SDK

4. Необходимо сделать видимой библиотеку для программ. Для этого можно прописать путь до libianpr_86.so или просто скопировать этот файл туда же, где лежат библиотеки OpenCV:

/usr/local/lib/

5. Обновить пути до библиотек
ldconfig

6. В папке с примером для Linux открываем makefile и исправляем lianpr_64 на lianpr_86 (поскольку компилируем в 32 битной версии linux).

7. В той же папке вызываем
make

Должен создаться исполняемый файл.

8. Проверяем работу программы.

3. Примеры на C/C++ для Windows

3.1. Image

Исходный код примеров постоянно совершенствуется. В описании ниже отражены наиболее значимые участки кода. Однако полный исходный код примеров может отличаться. Полный исходный код примера image находится в папке samples, распространяемой с iANPR SDK.

```
#include "opencv2/highgui/highgui_c.h"
#include "../../Include/iANPR.h"
#include <stdio.h>
#include <Windows.h>

int main( int argc, char** argv)
{
	IplImage* Img = 0;
	IplImage* grayImg = 0;


	// filter input
	if (argc < 2)
	{
		printf ("Too few arguments. For help print %s /?", argv [0]);
		return -1;
	}
	else if (!strcmp (argv [1], "help") || !strcmp (argv [1], "-help") || !strcmp
(argv [1], "--help") || !strcmp (argv [1], "/?"))
	{
		printHelp (argv [0]);
		return 0;
	}
	else if (argc < 3)
	{
		printf ("Too few arguments. For help print %s /?", argv [0]);
		return -2;
	}
	else // argc >= 3
		Img = cvLoadImage( argv[2], CV_LOAD_IMAGE_COLOR  );

	if (!Img)
	{
		printf( "Can't load file!\n");
		return -4;
	}

	CvRect Rects[100];
	int all = 100;
	char** res = new char*[all];
	for(int j=0;j<all;j++) res[j] = new char[20];
	ANPR_OPTIONS a;
	a.Detect_Mode = ANPR_DETECTCOMPLEXMODE;
```

```
	a.min_plate_size = 500;
	a.max_plate_size = 50000;
	a.max_text_size = 20;
	a.type_number = atoi (argv [1]);
	a.flags = 0;
	a.max_threads = 1;

	bool isFullType = false;
	for (size_t i = 0; i < anprFullTypesCount; i++)
	if (anprFullTypes[i] == a.type_number)
		isFullType = true;

	// Вызов LicenseCapture необходим только для платных версий
	// И только один раз, перед первым распознаванием.
	char* key = new char[8001]; memset(key, 0, 8001);
	FILE* f = fopen("lic.key", "rb");
	if (f != NULL)
	{
		fread((void*)key, 8000, 1, f);
		fclose(f);
	}
	else
		puts("WARNING! File lic.key not found. This may crash program if you use
license version of iANPR SDK dlls");

	LicenseValue(key);
	delete [] key; key = 0;

	int i = -9999;
	if (isFullType)
		i = anprPlate( Img,  a, &all, Rects, res );
	else
	{
		grayImg = cvCreateImage (cvGetSize (Img), 8, 1);
		cvCvtColor (Img, grayImg, CV_BGR2GRAY);
		i = anprPlate( grayImg,  a, &all, Rects, res );
	}

	if ( i == 0 )
		for( int j = 0; j < all; j++ )
		{
			printf( "%s\n", res[j] );
		}
	else
		printf( "Error:%d\n", i );

	for(int j=0;j<100;j++) delete [] res[j];
	delete [] res;
	cvReleaseImage ( &Img );
	cvReleaseImage ( &grayImg );

	return 0;
	}
```

В примере загружается изображение с помощью функции cvLoadImage. Далее выделяется память для хранения 100 номеров (предполагаем, что будут найдено не более 100 номеров, конечно, можно устанавливать меньше).

Заполняется структура ANPR_OPTIONS, при заполнении указывается режим детектирования дорожных знаков ANPR_DETECTCOMPLEXMODE. Устанавливается диапазон площади

(в пикселях в квадрате) номеров, 1 поток для распознавания. Значение 20 соответствует определенным выше буферам для хранения номеров. Тип детектирования номера установлен базовым, что означает, что при попадании в кадр, к примеру, номеров прицепа, они могут распознаваться неправильно.

Считывается лицензионный ключ из файла lic.key с помощью функции fread, а затем ключ передаётся в функцию LicenseValue. Вызов этой функции необходим только для лицензионной версии iANPR SDK и только перед первым распознаванием.

Вызывается функция **anprPlate** из iANPR SDK. После чего найденные номера в кадре изображения выводятся в консоль.

В конце осуществляется уничтожение ранее выделенной памяти.

Пример вызова (для текущей версии – отличается от примера выше):

image.exe 0 ..\images\image.bmp

Информация выведется на консоль, в файл:

image.exe 0 ..\images\image.bmp > res.txt

3.2. Image_new

В данном примере показано, как пользоваться различными интерфейсами для работы с библиотекой. В отличие от предыдущего примера подключается дополнительный заголовочный файл iANPRinterface.h.

Функция Memory в примере показывает возможность чтения JPEG, BMP, PNG, TIFF файлов из памяти. Т.е. можно прочитать файл в память, а потом передать указатель в **anprPlateMemory**.

В функции WithMat приведен C++ интерфейс доступа на основе Mat. Изображение распознается через функцию **anprPlateMatRect**.

В функциях XMLWork и XMLWork2 приведены примеры работы с функциями **anprPlateRectXML** и **anprPlateMatRectXML** соответственно. Результат в виде XML-строки выводится в консоль.

3.3. Image_omp

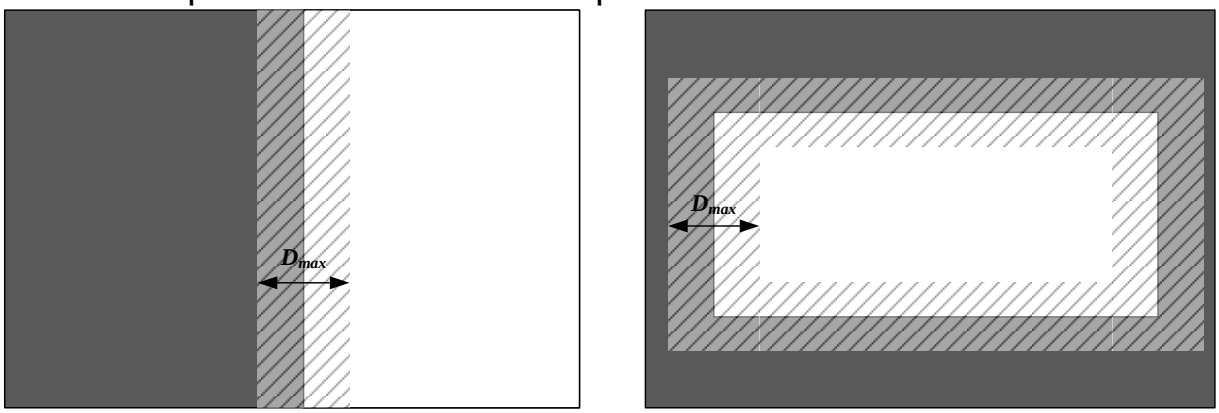
В версии iANPR 1.6 появилось встроенное распараллеливание. Этот пример может быть использован для сравнения результатов работы. В этом примере распараллеливание достигается за счёт разбиения входного изображения на отдельные части.

Когда производится работа с большими изображениями, например, 1920x1080, время распознавания на обычном ПК может быть недостаточно для работы в реальном времени. Для решения данной проблемы можно разбить изображение на части и анализировать их параллельно. Здесь, однако, следует помнить, что в случае попадания номера на пересечение частей, то он не будет распознан. Поэтому нужно внести некоторую избыточность, используя пересекающиеся части. Причем уровень пересечения определяется максимально возможными размерами объекта распознавания.

Предположим, что объект распознавания в форме круга, и его максимальный диаметр на изображении составляет D_{max}. Тогда ширина области пересечения сегментов должна быть больше D_{max}. В этом случае площадь несколько раз анализируемой области (в разных сегментах) будет рассчитываться по следующей формуле:

$$S = L * D_{max},$$

где L – длина границ между всеми сегментами. Понятно, что в этом случае S будет зависеть не только от количества сегментов, но и от их формы. На следующем рисунке показано два примера расположения сегментов. Один сегмент серого цвета, а другой белого. Область пересечения показана штрихами.

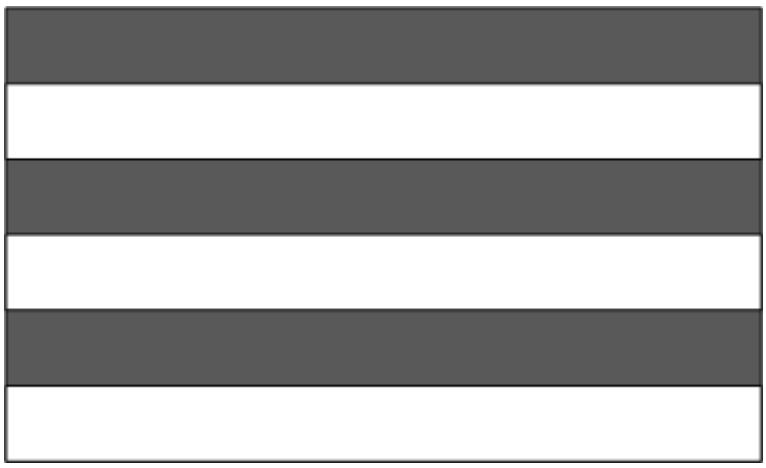


(а) (б)

Декомпозиция с пересекающимися границами: с небольшой областью пересечения (а), со значительной областью пересечения (б)

Недостатком является то, что при полном анализе изображения будет анализироваться не Width*Height площадь изображения, а Width*Height+S.

Предположим, что мы знаем количество ядер процессора, например, 6. Разделим изображение на 6 частей так:



Естественно нужно помнить об области пересечения частей. 1080/6 = 180

Возьмем в качестве максимальной высоты номера высоту в 80 пикселей. Тогда части вверх и вниз увеличатся на 40 пикселей, кроме самого верхнего и нижнего, где увеличение только в одну сторону.

В примере распараллеливание происходило на основе библиотеки OpenMP.

Для каждого потока создавалось собственное изображение, после чего запускался параллельный цикл:

```
#pragma omp parallel
		{
#pragma omp for
			for( i = 0; i < max; i++ )
			{
				CvRect Rects[100];
				int all = 100;
				char** res = new char*[all];
				for(int j=0;j<all;j++) res[j] = new char[20];
				ANPR_OPTIONS a;
				a.Detect_Mode = ANPR_DETECTCOMPLEXMODE;
				a.min_plate_size = 500;
				a.max_plate_size = 50000;
				a.max_text_size = 20;
				a.type_number = ANPR_RUSSIAN_BASE;
				a.flags = 0;

				int i2 = anprPlate( Images[i],  a, &all, Rects, res );
				if ( i2 == 0 )
					for( int j = 0; j < all; j++ ) {
#pragma omp critical
						{
							printf( "%s\n", res[j] );
						}
					}

				for(int j=0;j<100;j++) delete [] res[j];
				delete [] res;
			}
		}
```

Первоначально сравним насколько велика избыточность, откомпилировав программу с выключенным OpenMP. Успешное

распознавание было в обоих случаях. Время распознавания на AMD-FX(tm)-6100 Six-cores 3.3GHz (на одном ядре), 8Гб ОЗУ, Windows 7 64:

Последовательный блок 0.297с

Параллельный блок 0.369с

Т.е. избыточность вычислений примерно в 1.24 раз. Во столько замедлилось вычисление.

После включения OpenMP параллельный блок отработал за 0.1с. Т.е. прирост производительности в 2.9 раз.

Почему такой малый прирост? Ответ: 1) избыточность; 2) вычисления в блоках неравномерные – где обнаружился номер, а где нет, поэтому время работы – это время распознавания блока с максимальной информацией.

Но в принципе, даже такой прирост производительности – почти в 3 раза. А это позволяет за 1 секунду обработать 10 кадров, что для реального времени может быть достаточно при данных настройках.

3.4. Capture

Пример Capture работает с Web-камерой, но если в качестве аргумента командной строки указать видео файл, то будет обрабатывать его. Информация о распознавании выводится прямо в кадр следующим образом:

Особенностью данного примера потокового распознавания номера является то, что выдается результат распознавания не каждого кадра, а производится проверка – есть ли в предыдущем

кадре такой же найденный номер. Если есть, то только тогда выводится результат распознавания.

Данный подход позволяет отбросить значительную часть ложных срабатываний, возникающих в отдельном кадре.

3.5. Capture_(iANPRcapture)

Данный пример показывает, как работать с модулем **iANRcapture**.

Данный модуль позволяет добиться более высокой достоверности распознавания, чем в примере Capture, по причине того, что проверяется не совпадение номеров в обоих кадрах, а вычисляется общая достоверность распознанных символов в кадре с уровнем глубины (количества кадров), заданным программистом:

```
i_capture = CreateiANPRCapture( 10, a, cvRect( 0, 0, frame->width, frame->height ) );
```

Каждый кадр добавляется в потоковый объект

```
i1 = AddFrameToiANPRCapture( i_capture, object, &all, Rects, res );
```

Где вытесняется старый кадр и проверяется – не было ли похожего номера на предыдущих кадрах. Если номер, найденный в текущем кадре похож на номера в одном или нескольких предыдущих, то результаты распознавания суммируются.

3.6. (iANPRcapture_motion)

Данный пример является улучшение предыдущего. Позволяет вычислять траекторию движения номера и детектировать пересечение линии, т.е. можно реализовать функционал детектирования въезда-выезда авто.

Отличие данного примера в следующем.

Первоначально создается дополнительная память для дополнительной функциональности (удалять ее потом не надо, она очистится вместе с удалением объекта).

```
CreateMemoryForiANPRCapture( i_capture, 10, 15, 100 );
```

Максимальное количество номеров в памяти – 100. Если 15 кадров не было номера, распознанного ранее, то выдается результат. Номер считается распознанным, только если его начальное детектирование было раньше, чем 10 кадров в конечном.

Далее создается линия пересечения:

```
Lines[0].x = int( frame->width * 0.1f ); Lines[0].y = int( frame->height * 0.6f );
Lines[1].x = int( frame->width * 0.3f ); Lines[1].y = int( frame->height * 0.6f );
Lines[2].x = int( frame->width * 0.1f ); Lines[2].y = int( frame->height * 0.7f );
Lines[3].x = int( frame->width * 0.3f ); Lines[3].y = int( frame->height * 0.7f );
CreateLineIntersection( i_capture, Lines[0], Lines[1], Lines[2], Lines[3] );
```

Для получения такого уточненного результата нужно каждый раз (в каждом кадре, после вызова функции AddFrameToiANPRCapture вызывать функцию, возвращающую траекторию:

```
GetNumbersInMemory( i_capture, &all, res , 20, Points, &all_points );
```

Если значение all_points будет больше 0, то значит номер нашелся. И на экран выводится траектория номера и сам номер:

```
for(int j = 0; j < all_points; j++ )
{
	cvCircle( frame, Points[j], 5, CV_RGB(0,0,255), 3 );
	if ( j > 0 ) cvLine( frame, Points[j], Points[j-1], CV_RGB(0,0,255), 2 );
}

CvFont font;
float aa=0.001f*frame->width;
cvInitFont( &font, CV_FONT_HERSHEY_SIMPLEX, aa,
                            aa,0,1, 8 );
CvSize size;
int b;
cvGetTextSize( res[0], &font, &size, &b );
cvRectangle( frame, cvPoint(0, 0 ), cvPoint( size.width + 2,
	50 ), CV_RGB( 255, 255, 255 ), CV_FILLED );

CvPoint pp2,pp1;
pp2.x=0;
pp2.y=40;
pp1.x=1;
pp1.y=41;
cvPutText( frame, res[0], pp1, &font, CV_RGB(0,0,0) );
cvPutText( frame, res[0], pp2, &font, CV_RGB(0,255,0) );
cvResize( frame, image );
cvShowImage( "frame", image);
```

3.7. Persptrans

Утилита persptrans предназначена для устранения поворота автомобильного номера на исходном изображении. Угол поворота задаётся в структуре ANPR_OPTIONS с помощью параметров alpha, beta, gamma. Угол alpha задаёт поворот вокруг оси X, угол beta – вокруг оси Y, угол gamma – вокруг оси Z. Текущие значения параметров alpha, beta и gamma выводятся в стандартный поток

вывода (консоль). Полученные значения могут быть использованы в структуре ANPR_OPTIONS, передаваемой в одну из функций распознавания, например в anprPlate.

Для управления программой используются следующие клавиши:

- w, s – поворот вокруг оси X;
- a, d – поворот вокруг оси Y;
- q, e – поворот вокруг оси Z;
- r – выключить/включить распознавание;
- g – показать/скрыть сетку;
- v, b – увеличить / уменьшить количество клеток сетки;
- esc (escape) – выход.

Нажимать управляющие клавиши необходимо в окне «Transformed». Поворачивайте изображение до тех пор, пока не достигните требуемый результат в распознавании. Как правило, наилучшее распознавание происходит, если грани автомобильного номера (или хотя бы нижняя грань) параллельны сетке. Помните, что иногда меньший поворот лучше большего.

Пример использования:
persptrans.exe image.bmp 0

В данном примере использования для поворота будет загружено изображение image.bmp, распознаваться будут базовые российские номера (ANPR_RUSSIAN_BASE, то есть тип 0).

4. Примеры на других языках для Windows

4.1. C#

На C# рекомендуется использовать XML интерфейс для работы с библиотекой. В библиотеке – это пример platereader.

Подключается функция из библиотеки следующим образом:

```
[StructLayout(LayoutKind.Sequential, Pack = 0)]
public struct ANPR_OPTIONS
{
        public byte sign1, sign2, sign3;
        public int min_plate_size;
        public int max_plate_size;
        public int Detect_Mode;
        public int max_text_size;
        public int type_number;
        public int flags;
        public IntPtr custom;

        public IntPtr vers;

        public double alpha;
        public double beta;
        public double gamma;

        public int max_threads;
};

#if WIN32
        [DllImport("iANPRinterface_vc12_x86.dll", CallingConvention =
CallingConvention.StdCall)]
        unsafe public static extern int anprPlateMemoryXML(byte[] in_buffer, int
size_buffer, ANPR_OPTIONS Options,
            StringBuilder xml_buffer, int[] size_xml_buffer);
        [DllImport("iANPR_vc12_x86.dll", CallingConvention = CallingConvention.StdCall)]
        unsafe public static extern void LicenseValue(sbyte[] key);
#elif WIN64
        [DllImport("iANPRinterface_vc12_x64.dll", CallingConvention =
CallingConvention.StdCall)]
        unsafe public static extern int anprPlateMemoryXML(byte[] in_buffer, int
size_buffer, ANPR_OPTIONS Options,
            StringBuilder xml_buffer, int[] size_xml_buffer);
        [DllImport("iANPR_vc12_x64.dll", CallingConvention = CallingConvention.StdCall)]
        unsafe public static extern void LicenseValue(sbyte[] key);
#endif
```

При этом в настройках проекта необходимо разрешить небезопасный код и задать символы условной компиляции WIN32 и WIN64 для конфигураций сборки x86 и x64 соответственно.

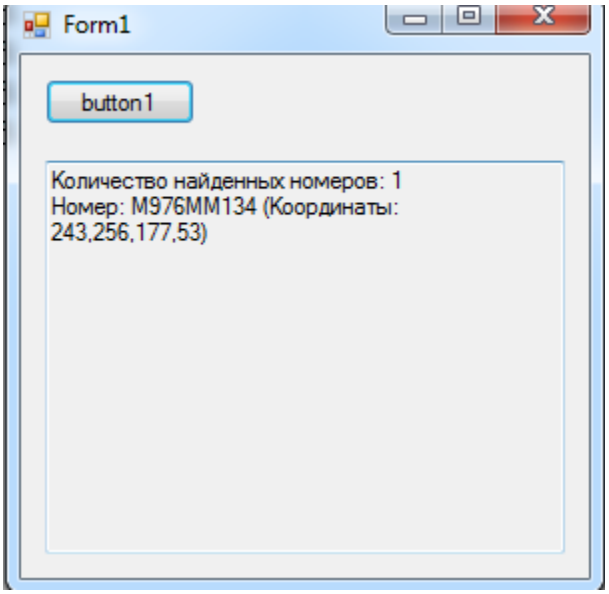
В примере, файл читается полностью в память и передается в функцию библиотеки:

```
ANPR_OPTIONS anpr;
anpr.Detect_Mode = 14;
anpr.min_plate_size = 500;
anpr.max_plate_size = 25000;
anpr.max_text_size = 20;
anpr.type_number = 0;
anpr.flags = 1;

anpr.sign1 = (byte)'i'; anpr.sign2 = (byte)'a'; anpr.sign3 = (byte)'1';

anpr.vers = Marshal.AllocHGlobal(4);
Marshal.WriteByte (anpr.vers, (byte) '1');
Marshal.WriteByte (anpr.vers + 1, (byte) '.');
Marshal.WriteByte (anpr.vers + 2, (byte) '6');
Marshal.WriteByte(anpr.vers + 3, 0);

anpr.alpha = 90;
anpr.beta = 90;
anpr.gamma = 90;

anpr.max_threads = 1;

StringBuilder buffer_builder = new StringBuilder(10000);
int[] size_builder = new int[1];
size_builder[0] = 10000;
int result = anprPlateMemoryXML(buffer, size, anpr, buffer_builder, size_builder);
```

Чтобы узнать результат распознавания, необходимо просмотреть возвращенный XML:

```
StringBuilder output = new StringBuilder();
using (XmlReader reader = XmlReader.Create(new StringReader(buffer_builder.ToString())))
{
	reader.ReadToFollowing("allnumbers");
	reader.MoveToFirstAttribute();
	string numbers = reader.Value;
	output.AppendLine("Количество найденных номеров: " + numbers);
	int all = Convert.ToInt32(numbers);
	for (int i = 0; i < all; i++)
	{
		reader.ReadToFollowing("number");
		reader.MoveToFirstAttribute();
		string num = reader.Value;
		reader.MoveToNextAttribute();
		string num_coord = reader.Value;
		output.AppendLine("Номер: " + num + " (Координаты: " + num_coord + ")");
	}
}
```

В результате в форму выведутся найденные номера и их координаты:



4.1.1. Пример iANPRcapture_motion на C# для iANPR SDK

Программа iANPRcapture_motion_CShrp предназначена для демонстрации возможностей iANPR SDK в вычислении траектории движения автомобильного номера и детектирования пересечения номером заранее заданных линий, т.е. для демонстрации возможности реализации функционала детектирования въезда-выезда автомобилей с помощью iANPR SDK. Эта программа написана на языке C# и является аналогом программы iANPRcapture_motion, написанной на языке C++. Эти и другие примеры использования распространяются в составе iANPR SDK.

Текущая версия iANPRcapture_motion_CShrp используется так:
iANPRcapture_motion_CShrp.exe 7 D:/video.avi
Где 7 – тип распознаваемого номера (российские номера),
D:/video.avi – полное имя видео файла.

4.2. Delphi

Конечно, в Delphi также можно использовать XML для возвращения результатов, но здесь приведен пример, как вызвать

функции из демо-версии iANPR без XML с использованием Delphi 7. Для вызова функций из платной версии iANPR необходимо перед первым использованием функций распознавания загрузить лицензионный ключ с помощью функции LicenseValue или LicenseCapture для активации iANPR.dll или iANPRcapture.dll соответственно.

Определение типов:

```
type
  ANPR_OPTIONS = Record
    min_plate_size:integer;
        max_plate_size:integer;
      Detect_Mode:integer;
      max_text_size:integer;
        type_number:integer;
    flags:integer;
  end;

type
  CvRect = Record
    x:integer;
    y:integer;
    width:integer;
    height:integer;
  end;
type
  PRect = ^CvRect;
```

Подключение функции:

```
function anprPlateMemoryRect( in_buffer: PChar; size_buffer: integer; Rect: CvRect;
Options: ANPR_OPTIONS; AllNumber: PInt; Rects: PRect; Texts: PPChar ): integer;
stdcall; external 'iANPRinterface_vc12_x86.dll'
  name 'anprPlateMemoryRect';
```

Чтение из файла и вызов функции:

```
with TFileStream.create(File_, fmOpenReadWrite) do
 try
 GetMem(p, Size);
 read(p^, Size);
 s := Size;
 finally
 free;
 end;
 all := 100;
 anpr.min_plate_size := 500;
 anpr.max_plate_size := 25000;
 anpr.Detect_Mode := 6; // ANPR_DETECTMODE2 | ANPR_DETECTMODE3;
 anpr.max_text_size := 20;
 anpr.type_number := 0; //      ANPR_RUSSIAN_BASE
 anpr.flags := 0;
 pr := @rect;
 GetMem( ptext, all * sizeof(pchar));
 for i := 0 to all-1 do
  GetMem( ptext[i], 20 *sizeof( char ) );
 RectArea.x := 0; RectArea.y := 0;
 RectArea.width := 640; RectArea.height := 480;
 r := anprPlateMemoryRect( p, s, RectArea, anpr, @all, pr, @ptext[0] );
```

```
FreeMem(p);
```

5. Рекомендации к использованию

ТРЕБОВАНИЯ К АЛГОРИТМУ РАСПОЗНАВАНИЯ

Номер автомобиля должен размещаться в кадре целиком.
Угол вертикального наклона видеокамеры не более 40°.
Угол наклона вглубь – не более 30°.
Изображения должны быть четкими и не размытыми.
Размер символов для надежного распознавания должен быть не менее 14 пикселей в высоту.
Расстояние до автомобиля камеры определяется фокусным расстоянием, установленным на камере и должно удовлетворять требованиям к высоте символов и четкости изображения. Это может быть и 3 метра, и 7 метров в зависимости от используемой камеры и ее настроек.

СРАВНЕНИЕ ПРОИЗВОДИТЕЛЬНОСТИ ВЕРСИЙ

Компьютер: Intel Core i7-6700K 4ГГц (на одном ядре), 16Гб ОЗУ, Windows 8.1 64 в режиме ANPR_RUSSIAN_FULL (библиотека x86)
Средняя обработка изображения 640x480: 0.023c (PRO), 0.4c (FREE)
Средняя обработка изображения 1920x1080: 0.171c (PRO), 3.6c (FREE)
При внутреннем распараллеливании 1920x1080: 0.098c (PRO)

РЕКОМЕНДАЦИИ К ПАРАМЕТРАМ РАСПОЗНАВАНИЯ
1. Для систем автоматизации въезда на парковку по списку белых номеров.
Detect_Mode = ANPR_DETECTCOMPLEXMODE;
flags = DEBUG_RECOGNITION_MODE;
Для того, чтобы наблюдать максимальное количество номеров, объединять результаты и потом отсеивать по списку белых номеров. Т.е. номер с одним ошибочно распознанным символом или вообще нераспознанным Вы можете на основе совпадения остальных символов с номером из белого списка (Вы сравниваете самостоятельно) отнести к правильно распознанному.
2. Для систем надежного распознавания номера при отсутствии белого списка.
Detect_Mode = ANPR_DETECTMODE2 | ANPR_DETECTMODE3;
flags = NO_LOW_RELIABILITY;
Для минимального количества ложных срабатываний.

6. Как пользоваться Демо-версией iANPR SDK

Демо версия не предназначена для распознавания в реальном времени, а лишь позволяет оценить функции распознавания автомобильных номеров. Технология распознавания реализована в библиотеке iANPR_vc12_x86.dll и существенно замедлена (примерно в 25-35 раз в зависимости от режима распознавания и процессора) для ограничения использования. Проверить работу SDK можно по готовым примерам. Например, image.exe.

image 0 image.bmp > image.txt

После отработки в image.txt будут результаты работы алгоритма распознавания (то же самое достигается bat-файлом run_read_image_bmp.bat). Если вы хотите распознать группу файлов в каталоге, то для этого можно воспользоваться bat-файлом run_read_in_dir.bat. Например, так:

run_read_in_dir.bat c:\im

Результаты распознавания выведутся в консоль.

Для тестирования распознавания в потоке, что должно повышать достоверность распознавания за счет анализа не одного кадра, а последовательности, нужно воспользоваться примером capture. По умолчанию пример работает с подключенной к компьютеру камерой, однако здесь следует помнить, что поскольку скорость существенно замедлена, то достоверность даже понизится, а не повысится. Поэтому если вы хотите проверить реальную достоверность распознавания, то запишите сначала видео в файл, а потом вызовите пример с параметром, например так:

capture.exe 0 100media\AMBA2826.MOV

Заключение

Библиотека постоянно развивается и совершенствуется, в том числе и с точки зрения качества распознавания.

Будет добавляться распознавание номеров из других стран.

Планируются дополнительные модули с вспомогательными функциями.

При возникновения ошибок, некорректного распознавания и т.п. обращайтесь на support@intbusoft.com, указав настройки алгоритма, которые вы используете, и приложив изображение, с которым возникают проблемы.