

iANPR SDK

Версия 1.21
Документация

СОДЕРЖАНИЕ

Введение

1. Модули

1.1. Модуль распознавания – iANPR

1.2. Модуль интерфейсов – iANPRinterface

1.3. Поточковый модуль - iANPRcapture

2. Инсталляция и использование

2.1. Windows

2.2. Linux

3. Примеры на C/C++ для Windows

3.1. Image

3.2. Image_new

3.3. Image_omp

3.4. Capture

3.5. Capture (iANPRcapture)

3.6. (iANPRcapture_motion)

4. Примеры на других языках для Windows

4.1. C#

4.2. Delphi

5. Рекомендации к использованию

6. Как пользоваться Демо-версией iANPR SDK

Заключение

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. Основным языком использования библиотеки – C/C++.

Версия 1.21 откомпилирована с версией OpenCV 2.4.8, при необходимости (по просьбе Клиента) может быть откомпилирована под другую версию.

Виды лицензий

iANPR RUS 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. Поставляется в виде динамических библиотек (Dll).

iANPR RUS PRO HOME Это платная лицензия, которая предоставляет все возможности по распознаванию стандартных и транзитных российских номеров. Предназначена для физических лиц. Поставляется в виде динамических библиотек (Dll).

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, принявшей данную лицензию. Поставляется в виде динамических библиотек (Dll).

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, принявшей данную лицензию. Поставляется в виде динамических библиотек (Dll).

iANPR RUS PRO FULL Это платная лицензия, которая предоставляет все возможности по распознаванию стандартных, транзитных российских номеров, а также остальных типов номеров, присутствующих в данной версии. Допускается использование для распространения собственного программного продукта.

1. Модули

В версии 1.2 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. Всего в модуле 2 функции.

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;

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
{
    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
};
```

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:

$\text{min_plate_size} = \text{min_plate_width} * \text{min_plate_height};$

$\text{max_plate_size} = \text{max_plate_width} * \text{max_plate_height};$

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:

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_PUBLI
ANPR_RUSSIAN_FULL	7	8bit, 3 channel	ANPR_RUSSIAN_EXTENDED2 + ANPR_RUSSIAN_POLICE + ANPR_RUSSIAN_ARMY

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

Базовые номера [ГОСТ Р 50577-93]:

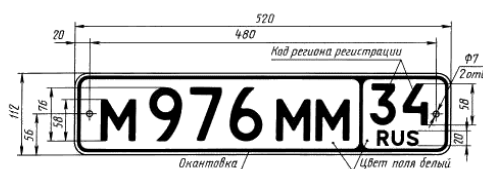


Рисунок А.1 – Регистрационный знак типа 1 с двухзначным кодом региона регистрации

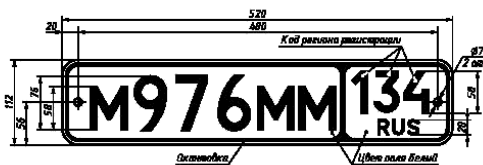


Рисунок А.2 – Регистрационный знак типа 1 с трехзначным кодом региона регистрации

Транзитные российские номера (ММ976М34).

Номера прицепов:

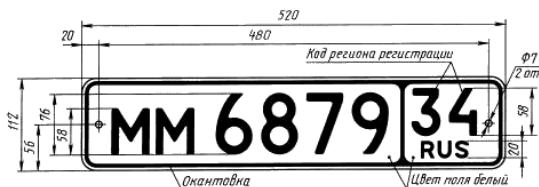


Рисунок А.5 – Регистрационный знак типа 2

Общественный транспорт – желтые номера формата MM11122.
Номера полиции – синие номера формата M111122.
Военные номера – черные номера формата 1111MM22.

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.

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.

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. Поточный модуль - iANPRcapture

Поточный 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.

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](#).

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.

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 x86 Microsoft Visual C++ 2010 SP1 Redistributable Package (x86)
<http://www.microsoft.com/en-us/download/details.aspx?id=8328>

Для x64 Microsoft Visual C++ 2010 SP1 Redistributable Package (x64)
<http://www.microsoft.com/ru-ru/download/details.aspx?id=13523>

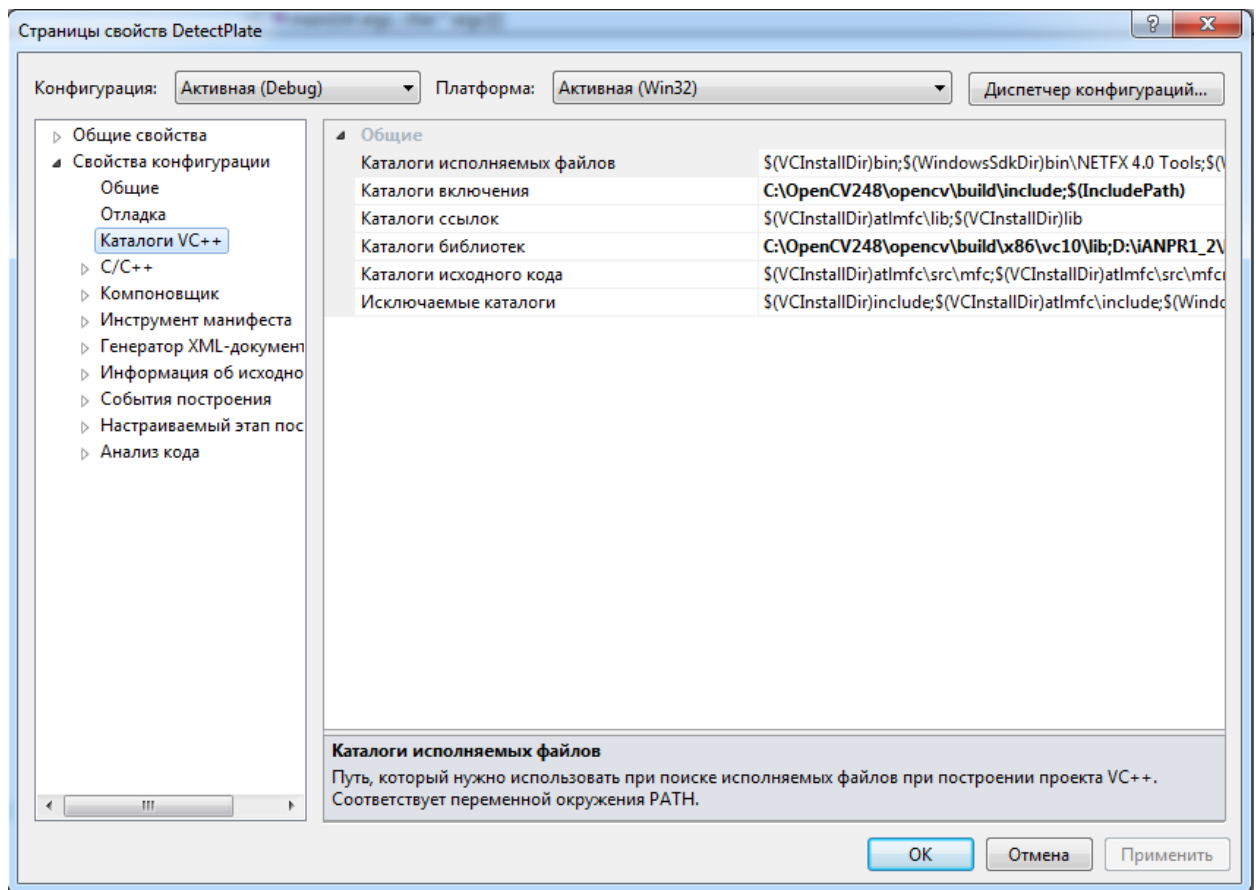
Скачайте OpenCV 2.4.8

<https://sourceforge.net/projects/opencvlibrary/files/opencv-win/2.4.8/opencv-2.4.8.exe/download>

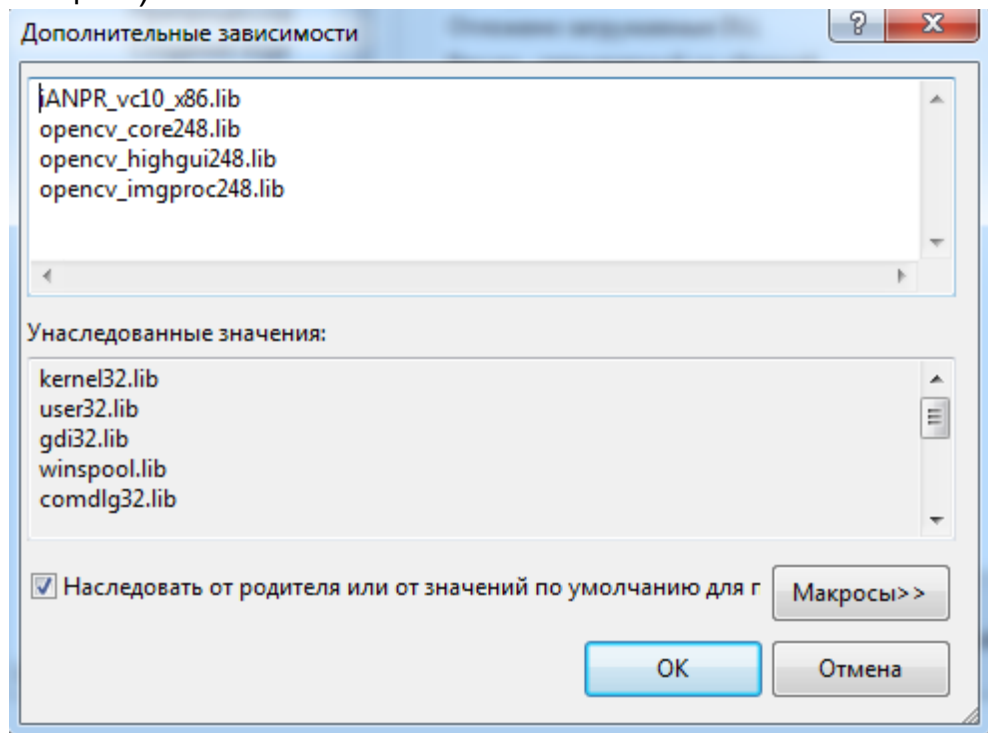
Хотя нужные библиотеки OpenCV идут вместе с iANPR SDK, но для разработки программ вам вероятно понадобится подключение заголовочных файлов. В iANPR использовались библиотеки, откомпилированные vs10.

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 13.10 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 2.4.8 для linux отсюда

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

И компилируем ее.

3. Скачиваем и распаковываем iANPR SDK

<http://intbusoft.com/rus/products/iANPR/>

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 файл sample.

8. Проверяем работу программы.

3. Примеры на C/C++ для Windows

3.1. Image

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

int main( int argc, char** argv)
{
    if ( argc > 1 )
    {
        IplImage* Img = cvLoadImage( argv[1], CV_LOAD_IMAGE_GRAYSCALE );
        if ( Img == NULL ){
            printf( "Cann't load file!");
            return 1;
        }
        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 i;
        __try
        {
            i = anprPlate( Img, a, &all, Rects, res );
        }
        __except(EXCEPTION_EXECUTE_HANDLER)
        {
            ExitProcess( 1 );
        }
        if ( i == 0 )
            for( int j = 0; j < all; j++ ) {
                printf( "%s\n", res[j] );
            }

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

В примере загружается изображение с помощью функции `cvLoadImage`. Далее выделяется память для хранения 100 номеров (предполагая, 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. 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.

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:

```
image.exe ..\images\image.bmp
```

Информация выведется на консоль, в файл:

```
image.exe ..\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 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

Встроенного распараллеливания в iANPR нет, однако его можно легко реализовать, разбивая входное изображение на отдельные части.

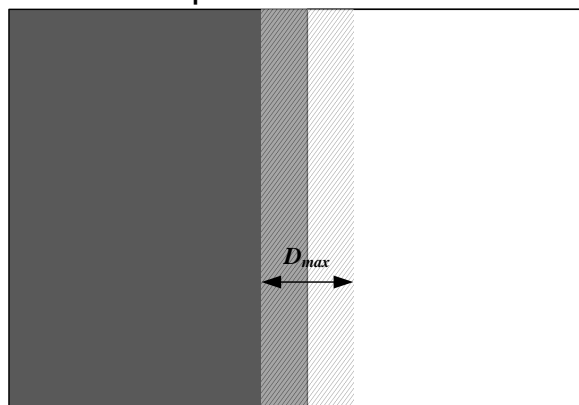
Когда производится работа с большими изображениями, например, 1920x1080, время распознавания на обычном ПК может быть недостаточно для работы в реальном времени. Для решения данной проблемы можно разбить изображение на части и анализировать их параллельно. Здесь, однако, следует помнить, что в

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.

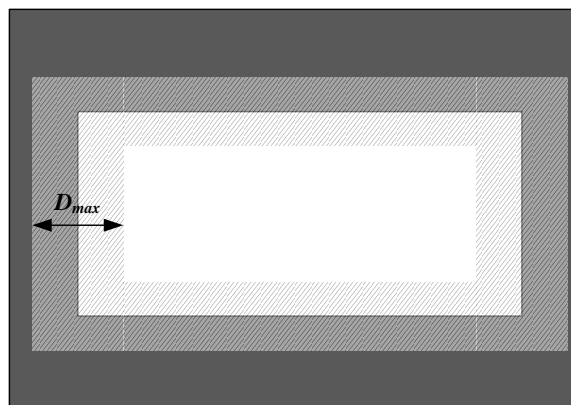
Предположим, что объект распознавания в форме круга, и его максимальный диаметр на изображении составляет D_{max} . Тогда ширина области пересечения сегментов должна быть больше D_{max} . В этом случае площадь несколько раз анализируемой области (в разных сегментах) будет рассчитываться по следующей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



(a)

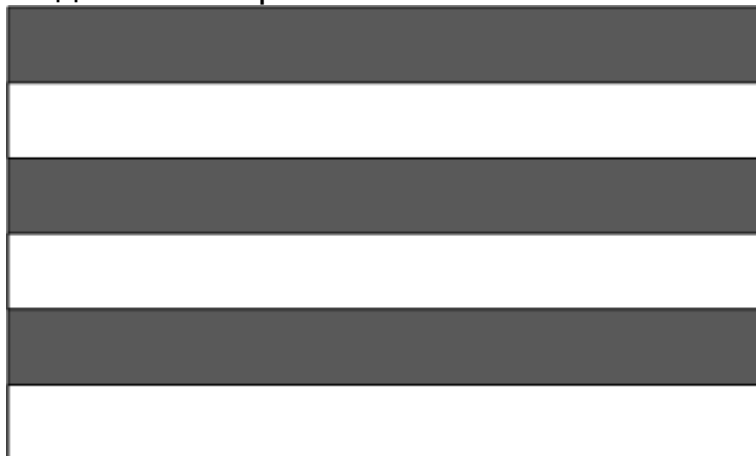


(б)

Декомпозиция с пересекающимися границами: с небольшой областью пересечения (a), 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++ ) {
                {
                    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.

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](#), появившимся в версии библиотеки 1.2.

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 найден. 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);
```

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:

```
public struct ANPR_OPTIONS
{
    public int min_plate_size; // Минимальная площадь номера
    public int max_plate_size; // Максимальная площадь номера
    public int Detect_Mode; // Режимы детектирования
    public int max_text_size; // Максимальное количество символов номера + 1
    public int type_number; // Тип автомобильного номера
    public int flags; // Дополнительные опции
};
[DllImport("iANPRinterface_vc10_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);
```

При этом в настройках проекта необходимо разрешить небезопасный код.

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;
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:

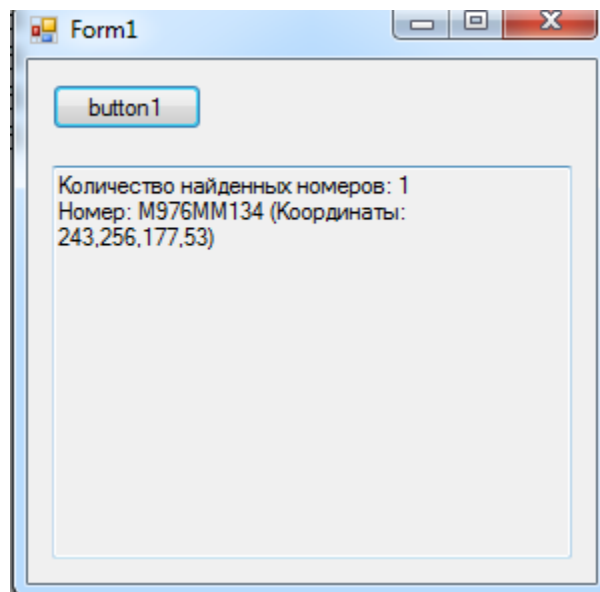
```
StringBuilder output = new StringBuilder();
using (XmlReader reader = XmlReader.Create(new StringReader(buffer_builder.ToString())))
{
    reader.ReadToFollowing("allnumbers");
    reader.MoveToFirstAttribute();
    string numbers = reader.Value;
    output.AppendLine("Количество найденных номеров: " + 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:



4.2. Delphi

Конечно, в Delphi также можно использовать XML для возвращения результатов, но здесь приведен пример, как вызвать функции iANPR без XML с использованием Delphi 7.

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 'iANPR_vc10_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 ) );
    end;
    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: AMD-FX(tm)-6100 Six-cores 3.3GHz (на одном ядре), 8Гб ОЗУ, Windows 7 64 в режиме ANPR_RUSSIAN_BASE

Средняя обработка изображения 640x480: 0.025с (PRO), 0.8с (FREE)

Средняя обработка изображения 1920x1080: 0.26с (PRO), 9с (FREE)

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 = 0;

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_vc10_x86.dll и существенно замедлена (примерно в 32 раза) для ограничения использования. Проверить работу SDK можно по готовым примерам. Например, image.exe.

```
image 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 100media\AMBA2826.MOV
```

Заключение

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

Будут добавляться распознавание номеров из других стран.

Планируются дополнительные модули с вспомогательными функциями.

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