TS-ANPR

TS-ANPR 은 딥러닝 기반의 대한민국 차량 번호 인식 엔진입니다.

- 차번 인식 데모: http://tsnvr.ipdisk.co.kr/

- 최신 엔진 다운로드: https://github.com/bobhyun/TS-ANPR/releases/

- 응용 프로그램 개발 가이드: https://github.com/bobhyun/TS-ANPR/blob/main/DevGuide.md

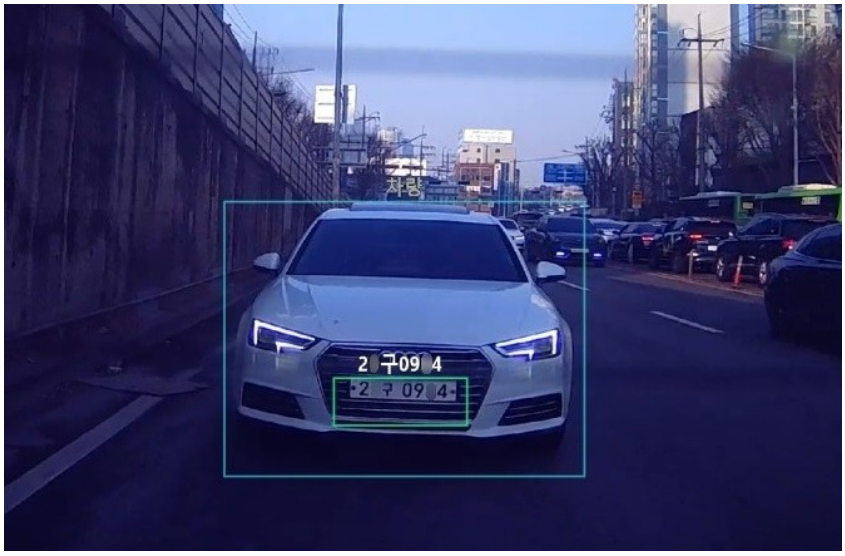
  - DLL entry points
  - 입력 이미지 파일 형식 (bmp, jpg, png, pnm, pbm, pgm, ppm, jfif, webp)
  - 입력 이미지 픽셀 형식 (GRAY, BGRA, RGBA, RGB, BGR, BGR555, BGR565, HSV, YCrCb, I420, YV12, IYUV, NV12, NV21)
  - 결과 출력 형식 (text, json, yaml, xml)
  - 프로그래밍 언어별 예제 소스 코드 (C/C++, C#, Visual Basic, Python, JavaScript/Node.js, Go, Pascal/Delphi, Perl, Ruby)

개발 문의: bobhyun@gmail.com
구매 문의: skju3922@naver.com
📞 전화: 02-6084-3922

특징

1. 차량 장착 검사 (Vehicle Mounted)

차체가 보이는 이미지에서 차량에 장착된 번호판인지 구분합니다.
차량 장착(v) 옵션을 사용하면 차량에 장착된 번호판만 인식합니다.



아래 이미지처럼 차량없이 번호판만 있거나 바이크 번호판 등은 무시합니다.

[이미지 출처: 연합뉴스]

[이미지 출처: 바이커즈랩]

번호판만 근접 촬영된 경우는 차량 인식이 안되는 경우가 있는데, 이런 경우 **차량 장착**(v) 옵션을 사용하지 않으면 차량 번호를 인식할 수 있습니다.

2. 다중 인식 (Multiple Recognition)

다중 인식(m) 옵션을 사용하면 이미지에 차량이 여러 대 있으면 모두 인식합니다.

다중 인식(m) 옵션을 사용하지 않으면 여러 대 차량 중 가장 번호판 신뢰도가 높은(잘 보이는) 것 하나만 인식합니다.

3. 서라운드 인식 (Surround Recognition)

서라운드 인식(s) 옵션을 사용하면 전복된 차량 또는 어안 렌즈 카메라로 촬영한 차량 등 이미지 내의 차량이 사방으로 기울어져 있거나 넘어져 있는 경우도 차량 번호를 인식할 수 있습니다.

[이미지 출처: KBS]


차량
4 서95 2
차량
2 모97 0
차량
3 부02 4
차량
4 버20 1
5 도81 7
B3

라이선스

1. 예제 소스 코드

제공되는 예제 소스 코드는 MIT 라이선스를 따릅니다.

```
The MIT License (MIT)
Copyright © 2022 TS-Solution Corp.

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to all conditions.

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.
```

2. TS-ANPR 엔진 바이너리

TS-ANPR 엔진 바이너리는 상용 라이선스로 제공됩니다.
아래 세 종류의 라이선스를 사용할 수 있습니다.

구분	기능	기간 제한
TS-ANPR 무료 평가판	차량 장착(v), 다중 인식(m), 서라운드 인식(s)	30 일
TS-ANPR 기본	차량 장착(v), 단일 인식	라이선스에 준함
TS-ANPR 프로	차량 장착(v), 다중 인식(m), 서라운드 인식(s)	라이선스에 준함

응용 프로그램 개발 가이드

목차

1. DLL entry points
    1.1. anpr_initialize
    1.2. anpr_read_file
    1.3. anpr_read_pixels
2. Output Format
    2.1. text
    2.2. json
    2.3. yaml
    2.4. xml
3. 오류 코드표
4. 예제

1. DLL entry points

1.1. anpr_initialize

라이브러리를 초기화 합니다.
라이브러리를 사용하기 위해 다른 함수보다 먼저 한 번 호출해야 합니다.

```
__declspec(dllimport)
const char* WINAPI anpr_initialize(const char* outputFormat); // [IN] 오류 발생시 출력 데이터 형식
```

Parameters:

- outputFormat:
    - 출력 데이터 형식
    - 지원하는 데이터 형식: text, json, yaml, xml (기본값: text)

Return value:

- 정상 처리된 경우 NULL terminated string (0x00)을 반환합니다.
- 오류가 발생한 경우는 outputFormat 에 지정한 데이터 형식의 문자열(utf-8 인코딩)로 오류 내용을 반환합니다.

1.2. anpr_read_file

이미지 파일에서 차량 번호를 인식합니다.

```
__declspec(dllimport)
const char* WINAPI anpr_read_file(
  const char* imgFileName,  // [IN] 입력 이미지 파일명
  const char* outputFormat, // [IN] 출력 데이터 형식
  const char* options);     // [IN] 기능 옵션
```

Parameters:

- imgFileName:
  - 입력 이미지 파일명 (utf-8 인코딩)
  - 지원하는 이미지 파일 형식: bmp, jpg, png, pnm, pbm, pgm, ppm, jfif, webp
- outputFormat:
  - 출력 데이터 형식
  - 지원하는 데이터 형식: text, json, yaml, xml (기본값: text)
- options:
  - 아래 문자를 조합하여 번호 인식 알고리즘의 옵션을 지정합니다.
  - vehicle-mounted (차량에 장착된 번호판만 인식)
  - multiple recognition (다중 인식)
  - surround recognition (사방으로 기울어진 번호판 인식)

  사용 예)

```
""    = 차량 장착 검사(X) + 단일 인식
"v"   = 차량 장착 검사(O) + 단일 인식
"m"   = 차량 장착 검사(X) + 다중 인식
"vm"  = 차량 장착 검사(O) + 다중 인식
"vs"  = 차량 장착 검사(O) + 서라운드 인식
"vms" = 차량 장착 검사(O) + 다중 인식 + 서라운드 인식
```

Return value:

- outputFormat 에 지정한 데이터 형식의 문자열(utf-8 인코딩)로 번호 인식 결과를 반환합니다.

Remarks:

- Return value 에 사용되는 문자열 버퍼는 dll 내부에서 관리되며 응용 프로그램에서는 문자열 버퍼를 참조하기만 하면 됩니다.
- 이 문자열 버퍼는 thread-safe 하며 각 thread 별로 다음 호출 전까지 결과 값이 유지됩니다.
- 참고 사이트
  - https://docs.microsoft.com/ko-kr/windows/win32/medfound/image-stride
  - https://docs.microsoft.com/ko-kr/windows/win32/medfound/video-fourccs

1.3. anpr_read_pixels

로딩된 이미지의 메모리 버퍼에서 차량 번호를 인식합니다.

```
__declspec(dllimport)
const char* WINAPI anpr_read_pixels(
  const unsigned char* pixels,  // [IN] 이미지 픽셀 시작 주소
  const unsigned long width,    // [IN] 이미지 가로 픽셀 수
  const unsigned long height,   // [IN] 이미지 세로 픽셀 수
  const unsigned long stride,   // [IN] 이미지 한 라인의 바이트 수
  const char* pixelFormat,      // [IN] 이미지 픽셀 형식
  const char* outputFormat,     // [IN] 출력 데이터 형식
  const char* options);         // [IN] 기능 옵션
```

Parameters:

- pixels: 이미지 픽셀 시작 주소
- width: 이미지 가로 픽셀 수
- height: 이미지 세로 픽셀 수
- stride: 이미지 한 라인의 바이트 수 (0 이면 padding 영역이 없는 것으로 간주하고 자동 계산)
- pixelFormat: 이미지 픽셀 포멧
  - 지원하는 픽셀 포멧:
    - GRAY: 흑백 이미지 (8bpp)
    - BGRA: BGRA (32bpp)
    - RGBA: RGBA (32bpp)
    - RGB: RGB (24bpp)
    - BGR: BGR (24bpp)
    - BGR555: BGR (16bpp)
    - BGR565: BGR (16bpp)
    - HSV: HSV (32bpp)
    - YCrCb: YUV444 (32bpp)
    - I420: YUV420 (12bpp)
    - YV12: YUV420 (12bpp)
    - IYUV: YUV420 (12bpp)
    - NV12: YUV420 (12bpp)
    - NV21: YUV420 (12bpp)
- outputFormat: (anpr_read_file 과 동일)
- options: (anpr_read_file 과 동일)

Return value: (anpr_read_file 과 동일)

Remarks: (anpr_read_file 과 동일)

2. Output Format

2.1. text

차량 번호 텍스트만 출력합니다.
번호판이 여러 개인 경우는 줄바꿈 문자 CR (0x0d)로 구분합니다.

```
01 가 2345
67 나 8901
```

차량 번호가 인식되지 않은 경우는 빈 텍스트 NULL terminated string (0x00)를 출력합니다.

오류가 반환되는 경우는 아래와 같은 텍스트 형식으로 출력합니다.

```
error: (1) Invalid parameters
```

2.2. json

차량 번호와 속성을 json 형식으로 출력합니다.

```
[
  {                          // 첫번째 번호판
    "text": "01 가 2345",     // 차량 번호
    "area": {                // 번호판 사각형 영역 (픽셀 단위)
      "x": 1217,             // 좌측 상단 좌표
      "y": 2083,
      "width": 92,
      "height": 175,
      "angle": 12.45         // 번호판 기울기
    },
    "conf": {                // 신뢰도 (범위: 0 ~ 1)
      "ocr": 0.75,           // 문자 인식 신뢰도
      "plate": 0.84          // 번호판 인식 신뢰도
    },
    "elapsed": 0.27,         // 소요 시간 (초 단위)
  },
  {                          // 두번째 번호판
    "text": "67 나 8901",
    "area": {
      "x": 1108,
      "y": 1317,
      "width": 67,
      "height": 217,
```

```
      "angle": 12.45
    },
    "conf": {
      "ocr": 0.76,
      "plate": 0.89
    },
    "elapsed": 0.14
  }
]
```

차량 번호가 인식되지 않은 경우는 아래와 같이 빈 데이터를 출력합니다.

```
[]
```

오류가 반환되는 경우는 아래와 같은 json 형식으로 출력합니다.

```
{
  "error": {
    "code": 1,
    "message": "Invalid parameters"
  }
}
```

2.3. yaml

차량 번호와 속성을 yaml 형식으로 출력합니다.

```
- text: 01 가 2345          # 첫번째 번호판
  area:                     # 번호판 사각형 영역 (픽셀 단위)
    x: 1217                 # 좌측 상단 좌표
    y: 2083
    width: 92
    height: 175
    angle: 12.45            # 번호판 기울기
  conf:                     # 신뢰도 (범위: 0 ~ 1)
    ocr: 0.75               # 문자 인식 신뢰도
    plate: 0.83             # 번호판 인식 신뢰도
  elapsed: 0.20             # 소요 시간 (초 단위)
- text: 67 나 8901          # 두번째 번호판
  area:
    x: 1108
    y: 1317
    width: 67
    height: 217
    angle: 12.45
  conf:
```

```
    ocr: 0.76
    plate: 0.89
  elapsed: 0.10
```

차량 번호가 인식되지 않은 경우는 아래와 같이 빈 데이터를 출력합니다.

```
```

오류가 반환되는 경우는 아래와 같은 yaml 형식으로 출력합니다.

```
error
  code: 1
  message: Invalid parameters
```

2.4. xml

차량 번호와 속성을 xml 형식으로 출력합니다.

```
<?xml version="1.0" encoding="utf-8"?>
<data>
  <license-plate text="01 가 2345" elapsed="0.20">   <!-- 첫번째 번호판,
      text: 차량 번호, elapsed: 소요 시간 (초 단위) -->
    <area x="1217" y="2083" width="92" height="175" angle="12.45"/>   <!--
      번호판 사각형 영역 (픽셀 단위), x, y: 좌측 상단 좌표, angle: 번호판 기울기 -->
    <conf ocr="0.75" plate="0.83"/>                                   <!--
      신뢰도 (범위: 0 ~ 1), ocr: 문자 인식 신뢰도, plate: 번호판 인식 신뢰도 -->
  </license-plate>
  <license-plate text="67 나 8901" elapsed="0.11">   <!-- 두번째 번호판 -->
    <area x="1108" y="1317" width="67" height="217"/>
    <conf ocr="0.76" plate="0.89"/>
  </license-plate>
</data>
```

차량 번호가 인식되지 않은 경우는 아래와 같이 빈 데이터를 출력합니다.

```
<?xml version="1.0" encoding="utf-8"?>
<data />
```

오류가 반환되는 경우는 아래와 같은 xml 형식으로 출력합니다.

```
<?xml version="1.0" encoding="utf-8"?>
<error code="1" message="Invalid parameters" />
```

3. 오류 코드표

전체 오류 목록은 아래 표와 같습니다.

code	message	설명
1	Invalid parameters	함수 호출 인자가 잘못된 경우
2	File not found	입력 이미지 파일이 존재하지 않는 경우
3	Invalid image	입력 이미지 메모리가 형식에 맞지 않는 경우
4	Unsupported image format	입력 이미지가 지원되지 않는 형식인 경우
100	License expired	라이선스가 만료된 경우
101	Corrupted library	라이브러리 구성 파일 중 일부가 없거나 손상된 경우
102	Not initialized	엔진이 초기화되지 않은 상태
103	Too many workers	라이브러리 호출 쓰레드 수가 한계를 초과한 경우 (최대 256 개)
104	Resource exhausted	더 이상 자원을 할당할 수 없는 경우
200	Unknown	기타 정의되지 않은 오류

4. 예제

디렉토리 구성

```
/examples
  /bin                    # ANPR 엔진 및 컴파일된 바이너리
    /x64                  # 64bit 바이너리 (amd64)
    /x86                  # 32bit 바이너리
  /img                    # 테스트용 샘플 이미지
  /cpp                    # C++ 예제
  /csharp                 # C#  예제
  /vb                     # Visual Basic 예제
  /javascript/nodejs      # JavaScript 예제
  /python                 # Python 예제
  /go                     # Golang 예제
  /pascal/delphi          # Pascal/Delphi 예제
  /perl                   # Perl 예제
  /ruby                   # Ruby 예제
```

TS-ANPR 엔진 디렉토리를 /examples/bin 디렉토리에 복사해 넣고 예제를 실행하면 됩니다.

언어	호출방식	예제
C/C++	Importlib	examples/cpp/anprCpp1
''	LoadLibrary	examples/cpp/anprCpp2
C#	.Net	examples/csharp/anprCsharpDotnet1
Visual Basic	.Net	examples/vb/anprVbDotnet1
Python	ctypes	examples/python
JavaScript	Node.js, ffi	examples/javascript/nodejs
Go	C, syscall	examples/go
Pascal	Delphi	examples/pascal/delphi
Perl	Win32::API	examples/perl
Ruby	ffi	examples/ruby