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

개발 문의: bobhyun@gmail.com
구매 문의: skju3922@naver.com
📞 전화: 02-6084-3920

버전 출시 이력

v1.7.1 출시🎉 (2024-02-07)

1.저사양 CPU 용 lite 버전 출시

●인식 속도는 일반 버전에 비해 약 2 배 빠름

●인식률은 일반 버전에 비해 근거리 촬영 이미지는 차이가 거의 없고 원거리 촬영이나 다중 인식시는 소폭 떨어짐

❍주차장 입출구 용으로 사용 권장

●엔진 파일만 교체하고 라이선스 변경없이 사용 가능

2.인식률 향상

●30 도 이상 기울어진 번호판 이미지에서 인식률 향상

3.2003 년까지 사용되던 주한 미군 개인 승용차 번호판 지원

[이미지 출처: 나무위키]

인식 속도 비교표

CPU	코어	쓰레드	클럭	운영체제	실행 환경	일반 버전	lite 버전
인텔 i7-12700	12	20	2.1GHz	윈도우즈, 리눅스	64 비트	0.040 초	0.021 초
인텔 i5-6500	4	4	3.2GHz	윈도우즈, 리눅스	64 비트	0.094 초	0.031 초
				윈도우즈	32 비트	0.203 초	0.094 초
인텔 i3-8100	4	4	3.6GHz	윈도우즈, 리눅스	64 비트	0.099 초	0.042 초
				윈도우즈	32 비트	0.204 초	0.096 초
인텔 셀러론 J4005	2	2	2.0GHz	윈도우즈, 리눅스	64 비트	1.017 초	0.398 초
				윈도우즈	32 비트	1.571 초	0.649 초
인텔 셀러론 1037U *(32 비트 전용)*	2	2	1.8GHz	윈도우즈	32 비트	1.186 초	0.484 초
Rockchip RK3588S	8	8	1.5GHz	리눅스 (ARM)	64 비트	0.534 초	0.247 초
BCM2711 *(라즈베리 파이 4)*	4	4	1.8GHz	리눅스 (ARM)	64 비트	1.213 초	0.528 초

[2023-12-14] v1.6.4

- 인식률 향상

[2023-11-22] v1.6.3

- 인식률 향상

[2023-11-7] v1.6.1

- Bugfix: v1.6.0 에서 임시번호판의 '임'자가 출력되지 않는 문제 수정

[2023-11-6] v1.6.0

- 차번 인식 결과 출력 개선
  - csv 형식 지원
  - 문자 인식 신뢰도 (conf.ocr) 값이 높은 순으로 정렬
- 인식률 향상

[2023-9-4] v1.5.1

- 인식률 향상

[2023-7-13] v1.5.0

- 인식률 향상
- `sync` 모드 지원
  - `anpr_initialize("sync;json")`와 같이 초기화시 `sync`를 지정하고 사용하면 됨
  - 미리 생성된 고정 갯수의 쓰레드풀 형태가 아니고 쓰레드가 계속 새로 생성되는 구조의 응용 프로그램에서 쓰레드 관리가 어려운 경우 사용 가능
  - 쓰레드 락(lock)을 사용하는 방식이므로 복잡한 비동기 쓰레드 관리를 신경쓰지 않아도 되는 반면 성능은 다소 떨어질 수 있음
  - `103: Too many workers`, `104: Resource exhausted` 오류가 발생하는 경우 대안으로 사용 가능

[2023-5-9] v1.4.0

- 리눅스🐧(linux-x86_64, linux-aarch64) 지원 (라이브러리 인터페이스는 기존 윈도우즈와 동일)
- 인식 속도 향상 (v1.2.0 대비 약 2.5 배 빠름🚀)
- Server, IoT 라이선스 추가
- 원거리의 여러 차량을 인식시 일부 차량 번호가 누락되는 현상 개선

[2023-2-28] v1.2.0

- 다중 차량 인식시 일부 번호판이 미인식되는 현상 개선
- 화질이 좋지 않은 번호판 이미지에서 문자 오인식 개선

[2022-1-18] v1.1.0

- 덤프트럭, 중장비 번호판 인식률 개선

[이미지 출처: 헤럴드경제]

[이미지 출처: 부동산미래]

[이미지 출처: jumbocar.tistory.com]

- 친환경 전기 자동차 인식 기능 추가
  - Output Format 에 ev 항목 추가됨 (개발 가이드 참고)

[이미지 출처: 부울경뉴스]

[2022-12-19] v1.0.6

- 1996 년 개정 후 폐지된 구형 번호판 (녹색 바탕 명조체 한글) 인식률 개선

[이미지 출처: 보배드림]

[2022-10-28] v1.0.5

- 응용에서 멀티스레드로 병렬 호출시 메모리 사용량 및 스레드 스위칭 오버헤드 개선
- 영업용 번호판 한글 문자 (바, 사, 아, 자) 오인식 개선

특장점

1. 차번 인식 능력

아래와 같은 다양한 환경 요인에 대해 뛰어난 적응력을 보입니다.

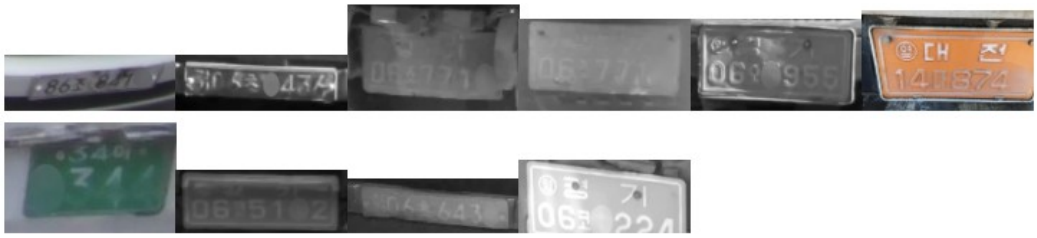
- 반사 필름

- 야간 노이즈

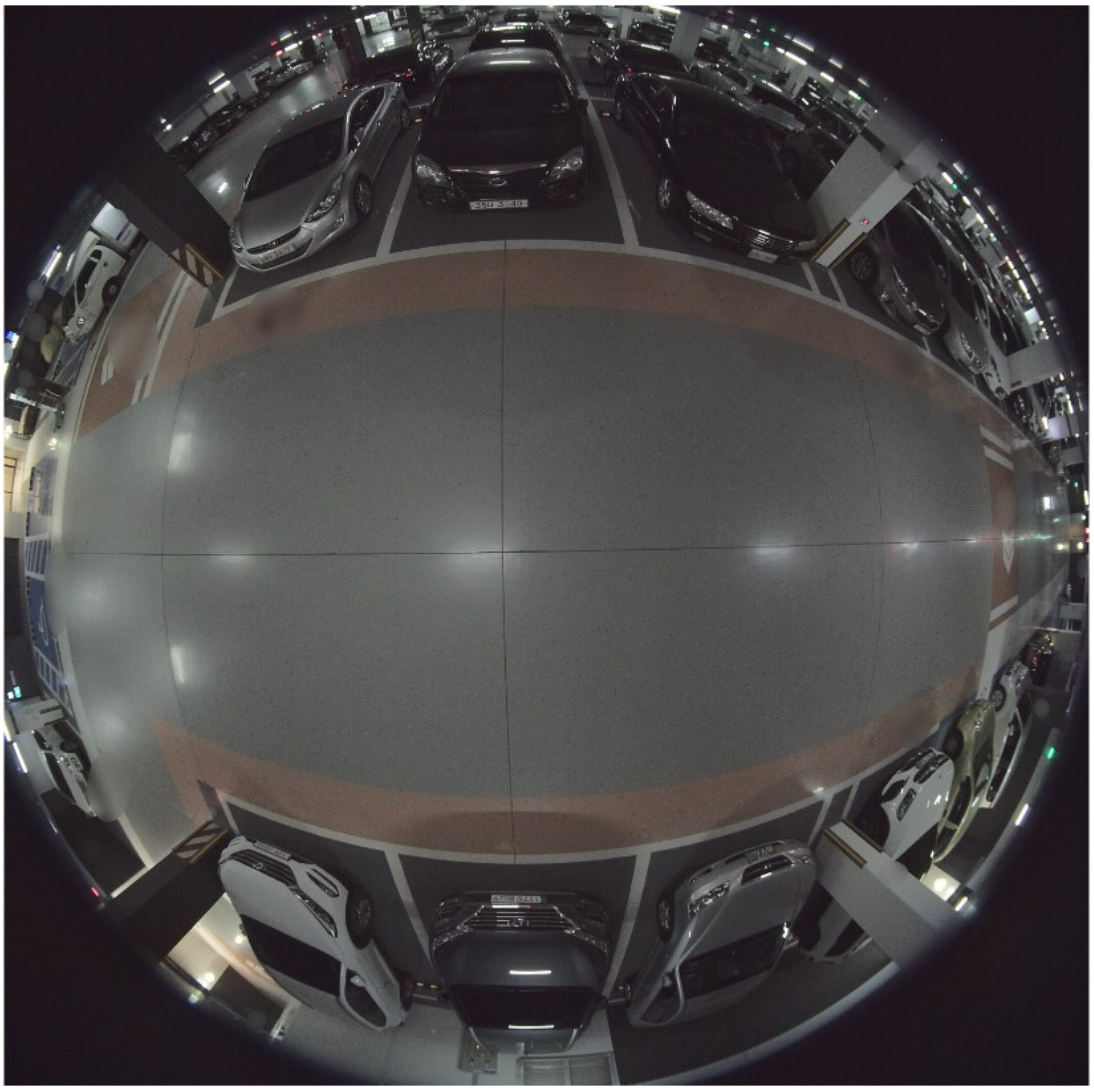
- 촬영 각도

- 날씨 / 조명

- 오염 / 훼손



- 360 도 어안 카메라 이미지
  - *이미지를 펼치지 않고 원본 이미지에서 여러 대의 차량 번호를 인식합니다.*



2. 각종 번호판 지원

아래와 같은 다양한 번호판 규격을 지원합니다.

- 덤프트럭, 중장비 번호판

- 특수 번호판 (임시, 외교, 군용)

- 친환경 전기차 번호판
  - *차번 인식 결과 데이터의 ev 항목에 true 또는 false 로 구분합니다.*
  - *단, 영업용 차량 번호판처럼 번호판 규격상 내연기관 차량과 구분되지 않는 경우는 판단이 불가능합니다.*

- '80, '90 년대 구형 번호판
  - *1996 년도 번호판 규격 개정 이전에 사용되던 처, 퍼, 차, 파, 추 ~ 후, 그 ~ 흐 문자를 지원합니다.*

3. 주요 운영체제 / CPU 아키텍처 지원

- 윈도우즈
  - 인텔 계열 64 비트(`windows-x86_64`), 32 비트(`windows-x86`)
  - 윈도우즈 7 이상 호환
- 리눅스
  - 인텔 계열 64 비트(`linux-x86_64`),
  - ARM 계열 64 비트(`linux-aarch64`)
  - 배포판에 관계없이 `glibc 2.27` 이상 호환

4. 다양한 개발 환경 지원

- 특정 프로그래밍 언어에 종속되지 않는 범용 라이브러리 인터페이스
- 프로그래밍 언어별 예제 제공 (C, C++, C#, Visual Basic, Python, JavaScript/Node.js, Go, Pascal/Delphi, Perl, Ruby)

- 입력 이미지 파일 형식 (bmp, jpg, png, pnm, pbm, pgm, ppm, jfif, webp)
- 입력 이미지 메모리 버퍼 픽셀 형식 (GRAY, BGRA, RGBA, RGB, BGR, BGR555, BGR565, HSV, YCrCb, I420, YV12, IYUV, NV12, NV21)
- 인식 결과 출력 형식 (text, json, yaml, xml, csv)

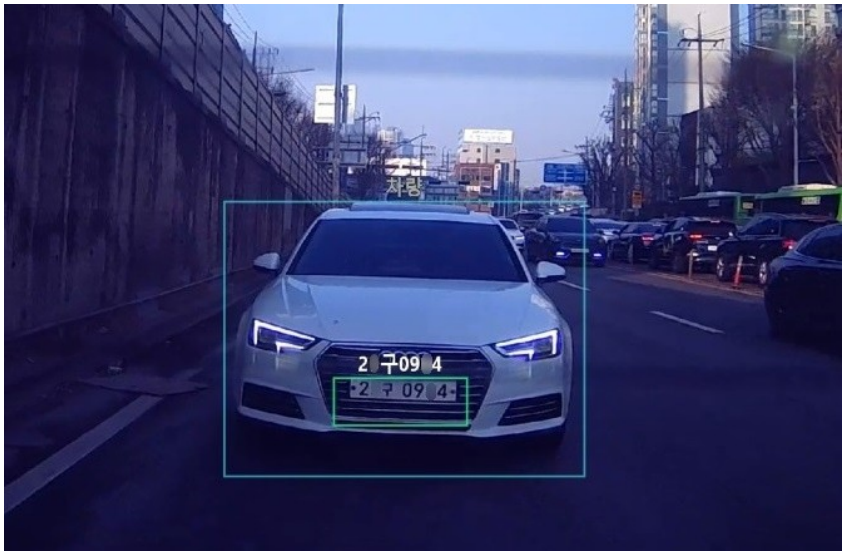
5. 다양한 라이선스 제공

- 무료 평가판 라이선스
  - 개발 및 데모용으로 시스템당 설치 이후 30 일간 무료 사용 기간 제공
- 상용 라이선스
  - 매체별: USB 동글, 또는 소프트웨어 라이선스 중 선택
  - 기능 및 성능별: `IoT`, `Basic`, `Pro`, `Server` 중 응용 소프트웨어 요구사항에 따라 선택

인식 옵션

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

엔진 및 라이선스 설치

윈도우즈용

anpr 엔진 파일을 설치할 디렉토리에 압축을 해제합니다.

x86 64 비트 윈도우즈 (windows-x86_64)

```
/windows-x86_64          # 윈도우즈용 64bit (amd64) 바이너리 디렉토리
  tsanpr.dll             # 공유 라이브러리 (API 제공)
  lpvr-xxxxx.eon         # 딥러닝 모델 #1
  lpocr_kr-xxxxx.eon     # 딥러닝 모델 #2
  tshelper.exe           # 도우미 앱 (라이선스 관리, Github 링크 제공)
```

x86 32 비트 윈도우즈 (windows-x86)

```
/windows-x86             # 윈도우즈용 32bit 바이너리 디렉토리
  tsanpr.dll             # 공유 라이브러리 (API 제공)
  lpvr-xxxxx.eon         # 딥러닝 모델 #1
  lpocr_kr-xxxxx.eon     # 딥러닝 모델 #2
  tshelper.exe           # 도우미 앱 (라이선스 관리, Github 링크 제공)
```

tshelper.exe 파일을 실행하면 해당 컴퓨터에 30 일 평가판 라이선스가 자동으로 설치되며, 이 후 30 일 동안 TS-ANPR 프로의 모든 기능을 사용해 볼 수 있습니다.

리눅스용

1. 설치

1.1 파일 압축 해제

anpr 엔진 파일을 설치할 디렉토리에 압축을 해제합니다.

x86 64 비트 리눅스 (linux-x86_64)

```
tar -xvf ts-anpr-v*-linux-x86_64.tar.gz
```

위의 명령을 실행하여 압축 해제된 파일 목록입니다.

```
  /linux-x86_64
    libtsanpr.so          # 공유 라이브러리 (API 제공)
    lpvr-xxxxx.eon        # 딥러닝 모델 #1
    lpocr_kr-xxxxx.eon    # 딥러닝 모델 #2
    tshelper              # 도우미 앱 (라이선스 관리, Github 링크 제공)
```

ARM 64 비트 리눅스 (linux-aarch64)

```
tar -xvf ts-anpr-v*-linux-aarch64.tar.gz
```

위의 명령을 실행하여 압축 해제된 파일 목록입니다.

```
  /linux-aarch64
    libtsanpr.so          # 공유 라이브러리 (API 제공)
    lpvr-xxxxx.eon        # 딥러닝 모델 #1
    lpocr_kr-xxxxx.eon    # 딥러닝 모델 #2
    tshelper              # 도우미 앱 (라이선스 관리, Github 링크 제공)
```

1.2. 라이선스 설치

`Tshelper` 프로그램으로 라이선스 관리를 합니다.
시스템에 최초로 실행하는 경우 현재 라이선스가 `미설치`로 표시됩니다.

```
./tshelper

TS-ANPR v1.5.0 (linux-x86_64)
(C) 2022-2023. TS Solution Corp. all rights reserved.

https://github.com/bobhyun/TS-ANPR

현재 라이선스:
  미설치

사용법:
  --lang, -l [LANG_ID]            # 표시 언어 선택 [ko, en]
```

```
  --trial, -t                    # 평가판 라이선스 설치하기
  --req, -r                      # 라이선스 요청서 파일 만들기
  --cert, -c [CERT_FILENAME]     # 라이선스 인증서 파일 설치하기
  --dongle, -d                   # USB 라이선스 동글 환경 설정하기

사용예:
  tshelper -l ko                 # 한국어 표시
  sudo ./tshelper -t             # 평가판 라이선스 설치하기 (root 권한 필요)
  sudo ./tshelper -r             # 라이선스 요청서 파일 만들기 (root 권한 필요)
  sudo ./tshelper -c sample.cert # 라이선스 인증서 파일 설치하기 (root 권한 필요)
  sudo ./tshelper -d             # USB 라이선스 동글 환경 설정하기 (root 권한 필요)
```

[참고] 표시되는 언어는 터미널 설정 기본값을 따릅니다. 만약 영문으로 표시되는 경우 아래와 같이 언어를 한글로 표시할 수 있습니다.

```
./tshelper -l ko
```

1.2.1 평가판 라이선스 설치

```
sudo ./tshelper -t

TS-ANPR v1.5.0 (linux-x86_64)
(C) 2022-2023. TS Solution Corp. all rights reserved.

https://github.com/bobhyun/TS-ANPR

새 라이선스가 설치되었습니다.

현재 라이선스:
  TS-ANPR 서버, 30 일 평가판 (30 일 남음)
```

1.2.2 정품 라이선스 (소프트웨어 라이선스)

소프트웨어 라이선스는 시스템별 고유 ID 를 식별하여 해당 시스템에서만 동작하는 라이선스를 발급받아 설치하는 방식으로 관리합니다.

1) 라이선스 요청서 만들기

아래 명령으로 생성된 라이선스 요청서 파일을 보내주시면 해당 시스템용 인증서 파일을 발급해 드립니다.

```
sudo ./tshelper -r

TS-ANPR v1.5.0 (linux-x86_64)
(C) 2022-2023. TS Solution Corp. all rights reserved.

https://github.com/bobhyun/TS-ANPR
```

```
현재 라이선스:
  TS-ANPR 서버, 30 일 평가판 (30 일 남음)

라이선스 요청서 파일이 저장되었습니다.

파일명: O20230414-TS-ANPR-8f5b0de4e9eabab6d727ab5c0d4c97e3.req
```

2) 인증서 설치하기

발급받은 정품 인증서 파일은 아래 명령으로 설치합니다.

```
sudo ./tshelper -c C20230414-TS-ANPR-8f5b0de4e9eabab6d727ab5c0d4c97e3.cert

TS-ANPR v1.5.0 (linux-x86_64)
(C) 2022-2023. TS Solution Corp. all rights reserved.

https://github.com/bobhyun/TS-ANPR

새 라이선스가 설치되었습니다.
현재 라이선스:
  TS-ANPR 서버, 정품 라이선스
```

1.2.3 정품 라이선스 (USB 동글 라이선스)

USB 동글 라이선스는 시스템에 USB 동글을 장착하면 즉시 적용됩니다.
단, 리눅스이 경우는 USB 동글을 최초로 삽입하기 전에 아래 명령을 한 번 실행한 후부터 자동 인식됩니다.

```
sudo ./tshelper -d
TS-ANPR v1.5.0 (linux-x86_64)
(C) 2022-2023. TS Solution Corp. all rights reserved.

https://github.com/bobhyun/TS-ANPR

USB 라이선스 동글 인식 환경이 설정되었습니다.
이제부터 USB 라이선스 동글을 삽입하면 자동으로 인식됩니다.
USB 동글이 이미 삽입되어 있으면 뺐다가 다시 삽입하세요.
```

2. 지원하는 리눅스 배포판

리눅스 시스템의 `glibc 2.27` 이상이면 특정 배포판에 무관하게 호환됩니다.

시스템의 glibc 버전은 아래 명령으로 확인할 수 있습니다.

```
ldd --version

ldd (Ubuntu GLIBC 2.35-0ubuntu3.1) 2.35
Copyright (C) 2022 Free Software Foundation, Inc.
```

```
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
Written by Roland McGrath and Ulrich Drepper.
```

호환되는 주요 리눅스 배포판 목록은 다음과 같습니다.

- Ubuntu 18.04 이후 버전
- Debian 10 이후 버전
- CentOS / RHEL / Oracle Linux 8 이후 버전
- Fedora 28 이후 버전

[참고] 기타 배포판들의 호환 여부는 distrowatch.com 에서 대부분 확인할 수 있습니다.
Slackware 를 예를 들면;
https://distrowatch.com/table.php?distribution=slackware
위 링크 화면 가운데 쯤 패키지 목록에서 glibc 항목에서 2.27 이후 버전을 찾으면 됩니다.

패키지	current	15.0	14.2	14.1	14.0
abiword (3.0.5)	--	--	--	--	--
alsa-lib (1.2.8)	1.2.8	1.2.5.1	1.1.1	1.0.27.2	1.0.26
amdgpu (23.0.0)	--	--	--	--	--
bash (5.2.15)	5.2	5.1	4.3	4.2	4.2
bind (9.18.13)	9.18.13	9.16.25	9.10.4-P1	9.9.3-P2	9.9.1-P3
chromium (112.0.5615.49)	--	--	--	--	--
cups (2.4.2)	2.4.2	2.3.3	2.1.4	1.5.4	1.5.4
dhcp (4.4.3-P1)	4.4.3-P1	4.4.2-P1	4.3.4	4.2.5-P1	4.2.4-P2
e2fsprogs (1.47.0)	1.47.0	1.46.5	1.43.1	1.42.8	1.42.6
firefox (112.0)	112.0	91.5.1	45.2.0	24.1.0	15.0.1
freetype (2.13.0)	2.13.0	2.11.1	2.6.3	2.5.0.1	2.4.10
gcc (12.2.0)	12.2.0	11.2.0	5.3.0	4.8.2	4.7.1
gimp (2.10.34)	2.10.34	2.10.30	2.8.16	2.8.6	2.8.2
glibc (2.37)	2.37	2.33	2.23	2.17	2.15
gnome-shell (44.0)	--	--	--	--	--

- 응용 프로그램 개발 전 단계의 기본적인 성능 테스트는 온라인 데모 사이트 http://tsnvr.ipdisk.co.kr/ 를 이용하실 수 있습니다.
- 응용 프로그램 개발 단계에서는 응용 프로그램 개발 가이드 와 포함된 프로그래밍 언어별 예제들을 참고하시기 바랍니다.
- 개발 관련 질문이나 요청 사항들은 Issues 에 등록해 주시면 적극적으로 지원하겠습니다.

응용 프로그램 개발 가이드

목차

1. Entry points
    1.1. 라이브러리 초기화
        1.1.1. anpr_initialize
    1.2. 차량 번호 인식
        1.2.1. anpr_read_file
        1.2.2. anpr_read_pixels
        1.2.3. 번호 인식 결과 출력 형식
            1.2.3.1. text
            1.2.3.2. csv
            1.2.3.3. json
            1.2.3.4. yaml
            1.2.3.5. xml
2. 오류 코드표
3. 예제

1. Entry points

모든 함수 원형은 아래와 같습니다.

```
#ifdef WIN32
#define TS_ANPR_ENTRY extern "C" __declspec(dllexport) const char* WINAPI
#else
#define TS_ANPR_ENTRY extern "C" const char*
#endif
```

- 장황해지지 않도록 이하는 `TS_ANPR_ENTRY`로 표기합니다.

1.1. 라이브러리 초기화

1.1.1. anpr_initialize

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

```
TS_ANPR_ENTRY anpr_initialize(const char* mode); // [IN] 라이브러리 동작 방식 설정
```

Parameters:

`mode`
: 라이브러리 동작 방식을 지정하는 목적으로 사용 (기존 `outputFormat`에서 용도 확장)
세미콜론(;) 문자로 구분하여 여러 설정을 표현할 수 있음 (예: `json;sync`)
지정 가능한 항목

outputFormat: 출력 데이터 형식

지원하는 데이터 형식: text, json, yaml, xml, csv(기본값: text)

outputFormat 생략하고 간단히 text, json 으로 사용 가능

sync: 동기 모드로 실행 (쓰레드 lock 을 걸고 호출한 순서대로 처리)

미리 생성된 고정 갯수의 쓰레드풀 형태가 아니고 쓰레드가 계속 새로 생성되는 구조의 응용 프로그램에서 호출하는 경우, 아래와 같은 오류 코드 발생시 사용을 고려할 수 있음

103: Too many workers 라이브러리 호출 쓰레드 수가 한계를 초과한 경우 (최대 256 개)

104: Resource exhausted 더 이상 자원을 할당할 수 없는 경우

복잡한 비동기 쓰레드 관리를 신경쓰지 않아도 되는 반면 쓰레드 락(lock)을 사용하는 방식이므로 성능은 다소 떨어질 수 있음

sync=true 또는 sync=false 로 표현할 수 있으며, 간단히 sync 만 사용해도 됨 (지정안하면 기본값 sync=false 로 동작)

Return value:

정상 처리된 경우 빈 텍스트 NULL terminated string (0x00)을 반환합니다.

오류가 발생한 경우는 mode 의 outputFormat 으로 지정한 데이터 형식의 문자열(utf-8 인코딩)로 오류 내용을 반환합니다.

1.2. 차량 번호 인식

1.2.1. anpr_read_file

이미지 파일에서 차량 번호를 인식합니다.

```
TS_ANPR_ENTRY 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, csv (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. [참고: 1.2.3. 번호 인식 결과 출력 형식]

Remarks:

Return value 에 사용되는 문자열 버퍼는 라이브러리 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

1.2.2. anpr_read_pixels

로딩된 이미지의 메모리 버퍼에서 차량 번호를 인식합니다.

```
TS_ANPR_ENTRY 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)

1.2.3. 번호 인식 결과 출력 형식

1.2.3.1. 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
```

1.2.3.2. csv

차량 번호와 속성을 csv 형식으로 출력합니다.

인식된 차량 번호 당 한 라인 씩으로 구성되며 각 컬럼은 콤마 문자(,)로 구분됩니다.

```
01 가 2345,1217,2083,92,175,12.45,0.75,0.83,0.20,ev
67 나 8901,1108,1317,67,217,12.45,0.76,0.89,0.10
```

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

```

```

오류가 반환되는 경우는 아래와 같은 csv 형식으로 출력합니다.

```
Error,1,Invalid parameters
```

1.2.3.3. 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
    },
```

```
    "ev": true,              // 친환경 전기자동차 여부
    "elapsed": 0.27,         // 소요 시간 (초 단위)
  },
  {                          // 두번째 번호판
    "text": "67나8901",
    "area": {
      "x": 1108,
      "y": 1317,
      "width": 67,
      "height": 217,
      "angle": 12.45
    },
    "conf": {
      "ocr": 0.76,
      "plate": 0.89
    },
    "ev": false,             // 친환경 전기자동차 여부
    "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"
  }
}
```

1.2.3.4. yaml

차량 번호와 속성을 yaml 형식으로 출력합니다.

```
- text: 01가2345          # 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
  ev: true             # 친환경 전기자동차 여부
  elapsed: 0.20        # 소요 시간 (초 단위)
- text: 67나8901        # 두번째 번호판
  area:
    x: 1108
    y: 1317
    width: 67
    height: 217
    angle: 12.45
  conf:
    ocr: 0.76
    plate: 0.89
  ev: false            # 친환경 전기자동차 여부
  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
```

1.2.3.5. xml

차량 번호와 속성을 `xml` 형식으로 출력합니다.

```
<?xml version="1.0" encoding="utf-8"?>
<data>
  <!-- 첫번째 번호판, text: 차량 번호, ev: 친환경 전기자동차 여부, elapsed: 소요 시간 (초 단위)
-->
  <license-plate text="01가2345" ev="true" elapsed="0.20">
    <!-- 번호판 사각형 영역 (픽셀 단위), x, y: 좌측 상단 좌표, angle: 번호판 기울기 -->
    <area x="1217" y="2083" width="92" height="175" angle="12.45"/>
    <!-- 신뢰도 (범위: 0 ~ 1), ocr: 문자 인식 신뢰도, plate: 번호판 인식 신뢰도 →
    <conf ocr="0.75" plate="0.83"/>
  </license-plate>
  <!-- 두번째 번호판 -->
  <license-plate text="67나8901" ev="false" elapsed="0.11">
    <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" />
```

2. 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
105	`License not installed`	라이선스가 설치되지 않은 상태 (리눅스에서 무료 평가판 라이센스가 설치되지 않은 경우 발생함)
106	`USB dongle I/O error`	USB 라이선스 동글 읽기 실패시 발생
107	License required	해당 기능을 사용하기 위한 라이선스가 없음
200	Unknown	기타 정의되지 않은 오류

3. 예제

디렉토리 구성

```
/examples
  /bin                  # 각 플랫폼별 ANPR 엔진
    /windows-x86_64
    /windows-x86
    /linux-x86_64
    /linux-aarch64
  /img                  # 테스트용 샘플 이미지
  /cpp                  # C++ 예제
  /csharp               # C#  예제
  /vb                   # Visual Basic 예제
  /javascript/nodejs    # JavaScript/Node.js 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

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 라이선스**로 제공됩니다.
아래 세 종류의 라이선스를 사용할 수 있습니다.

구분	기능	성능	기간 제한
TS-ANPR 무료 평가판	vms	무제한	30 일
TS-ANPR 서버	vms	무제한	라이선스에 준함
TS-ANPR 프로	vms	최대 16 CPU 코어	라이선스에 준함
TS-ANPR 기본	v	최대 8 CPU 코어	라이선스에 준함
TS-ANPR IoT	v	최대 4 CPU 코어	라이선스에 준함

- 기능: v(차량 장착), m(다중 인식), s(서라운드 인식)
- 성능: CPU 코어 일부분만 사용하는 방식으로 차번인식 성능을 제한