Project 제안서

OpenCV를 이용한 번호판 인식기

- 작성자 : 27기 180이기진
- 작성일 : 2020. 04. 02

BARAM KW univ
New generation of Robotics

목 차

BARAM KW univ
New generation of Robotics

1. Project 소개

(1) 선정 동기

특정 로봇들은 어떠한 행동을 시키려면 단순히 명령을 내리기만 해도 된다. 하지만 일상에서 일어나는 상황들에 대해 하나하나 대처하며 조금 더 섬세하고, 정교한 작업들을 수행하려면, 로봇도 인간과 마찬가지로 눈이 필요하다. 눈은 신체의 기본이 되는 기관중 하나이다. 눈을 통해 우리는 주변의 상황을 판단하고, 받아들인 시각 정보를 가지고 뇌를 통해 그에 대한 행동을 수행할 수 있다. 주어진 작업들만 반복적으로 시행하는 것이 아니고, 우리가 보는 것과 같이 주변을 보고 판단을 내릴 수 있어야 한다는 것이다. 이를 위해서 Open CV를 통한 영상 처리로 통해 미래에 만들 로봇의 눈에 대해 미리 알아보고자 한다. 로봇의 눈은 주변 물체와의 거리를 계산하거나, 정보를 읽어내는 능력 등이 필요하다. 이번 작품에서는 번호판 인식기를 만들어 주변 숫자 및 글자 정보에 대해 파악할 수 있도록 만들어 보고자 한다.

(2) 목표

가. 웹캠 영상 불러오기

-웹캠 앞에서 번호판을 들고 돌아다님

나. 번호판 부분을 감지

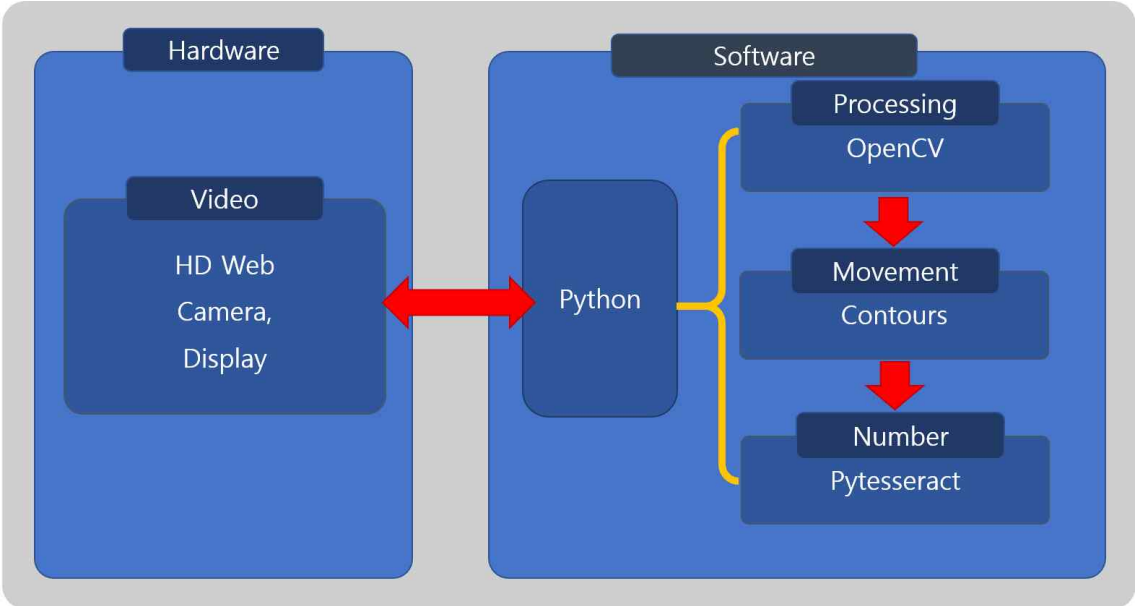
-번호판 숫자부분을 직사각형으로 표시하고, 숫자간의 거리 및 높이 등을 계산하여 번호판이라는 것을 알아냄

다. 번호판의 숫자를 인식

-광학 문자 인식 라이브러리를 통하여 번호판의 숫자 및 문자 인식

2. Project 내용

(1) System Architecture



사람이 웹캠 앞에서 숫자(폰트)가 써져 있는 번호판을 들고 있면 종이의 숫자 영역을 인식하고, 그 숫자를 인식하여 출력한다.

(2) 목표 구현 내용

A. 웹캠 영상 실시간 출력

VideoCapture 함수를 이용하여 웹캠의 영상을 실시간으로 불러온다.

`cv2.VideoCapture(0)`
괄호 안에 0번은 노트북의 기본 카메라, 1번부터는 외장 연결된 웹캠이 할당된다.

B. 영상 처리

Opencv에서 번호판 영역 및 숫자의 높은 인식률을 위해, 각 부분을 구별할 수 있도록 해주는 몇 가지 영상 처리가 필요하다.

1. GrayScale

웹캠이 받아오게 되는 이미지는 컬러이미지이며, BGR포맷으로 부른 경우 0~255의 총 256가지 색상값으로 색을 표현한다. 이 이미지를 회색으로 만들어주기 위해 Grayscale을 사용하며, 이 때 색상정보는 없이 밝기정보로만 구성되어 0~255까지 밝기 단계로 이미지를 표현하게 된다. 검은색은 0, 흰색은 255이다. 사용되는 코드

는 다음과 같다.

```
cv2.cvtColor(video, cv2.COLOR_BGR2GRAY)
```

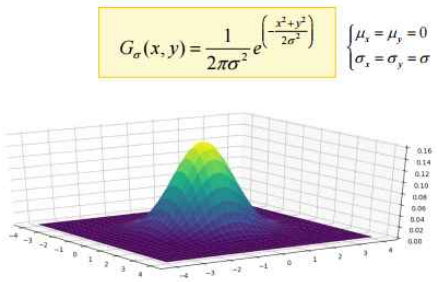
2. Gaussian Blur

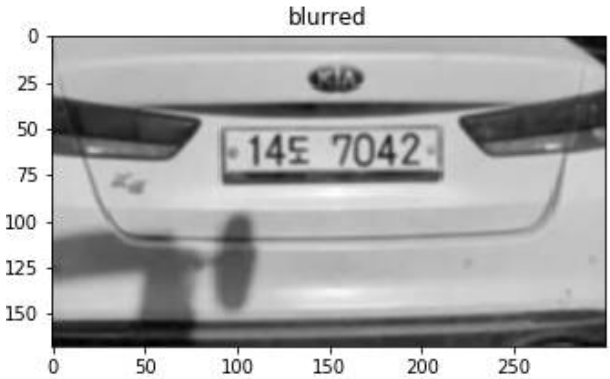
이미지에 커널을 적용하여 이미지를 블러 처리 하여 이미지의 노이즈 및 손상을 줄인다. 번호판을 포함한 이미지 전체를 조금씩 뿌옇게 만든다. (하단 사진 참고)

OpenCV에서 GaussianBlur를 사용하기 위한 함수 코드는 다음과 같다.

```
dst =cv2.GaussianBlur(src, ksize, sigmaX[, dst[, sigmaY[, borderType=BORDER_DEFAULT]]] )
```

src는 입력 이미지, ksize는 커널의 크기, sigmaX,Y는 각 축에 따른 표준 편차이다. 과도하게 노이즈를 제거하려다 이미지가 너무 흐릿해 질 수도 있으므로, 적당한 커널의 크기와 시그마값을 찾아야 한다.





3. 이진화(Binarization)

영상 이진화란 적, 녹, 청등의 RGB값으로 다양하게 분포되어 있는 색상 값들을 0과 1만으로 표현하는 것이다. 즉, 검은색, 흰색으로만 나타내겠다는 말과 같다. Gray로 바뀐 RGB영상은 하나의 픽셀당 0에서 255의 값중 하나를 갖게 된다. 그리고 사용자가 임계값(Threshold)을 설정해주면 그 값을 초과하는 픽셀값은 1로, 그 값 이하의 값은 0으로 변환하게 된다. 함수 코드는 다음과 같다.

```
cv2.threshold(gray, 100, 255, cv2.THRESH_BINARY)
```

괄호 안의 내용은 순서대로 이미지변수, 임계치, 총 색상 수 이다.

C. 번호판 부위 인식

웹캠 사람이 번호판 모형(폰트 글씨)를 들고 돌아다니면 표지판 부분을 인식한다.

1. Countours (윤곽선)

Countour는 특정 영역의 경계를 따라 같은 픽셀 강도를 갖는 지점을 연결하는 선이다. 모양 분석이나 객체 검출에 사용된다. 지금까지의 과정을 통해 이진화까지 마친 이미지 중에서 번호판 부분을 Countour를 통하여 검출후 직사각형 표시한다.

```
cv2.findContours(thr, cv2.RETR_TREE,
cv2.CHAIN_APPROX_SIMPLE)
```

위 함수를 통하여 이진화된 이미지에서 윤곽선을 찾는다. thr는 소스 이미지, cv2.RETR_TREE는 contour 추출모드, cv2.CHAIN_APPROX_SOIMPLE는 contour 근사 방법이다. //원본 이미지를 변경시키기 때문에 나중에 원본 이미지를 활용하려면 복사본을 가지고 Contour를 찾아야 한다.

```
cv2.drawContours(img, contours, -1, (0, 255, 0), 3)
```

findContours를 통해 찾아낸 윤곽선을 실제로 그리는 함수이다. 인자는 순서대로 img는 대상 이미지, contours, -1은 contour의 인덱스 파라미터(-1일땐 윤곽선을 다 그려준다), (0, 225, 0)은 윤곽선의 BGR색상값, 3은 선의 두께이다.

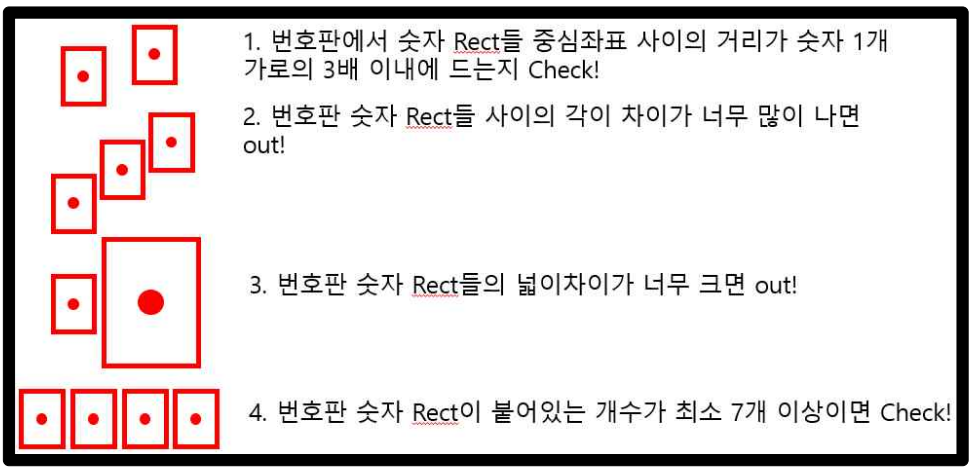
윤곽선이 그려진 영상에서, 번호판이 어디에 있는지 찾아내기 위하여 번호판의 숫자영역을 인식하도록 한다.

```
x, y, w, h = cv2.boundingRect(contour)
```

번호판 부분을 수월히 인식하기 위해서 번호판을 감싸는 사각형을 찾아낸다 x,y좌표와 너비, 높이 정보를 리스트에 저장한다.

또한 이때 각 사각형의 중심좌표도 다음과 같이 저장한다.
(xcenter, ycenter) = (x+(w/2), y + (h/2))

여러 BoundingRect들 중에서 번호판의 모양새를 한 Rect를 찾아낸다. 번호판 숫자 및 문자 폰트는 크기가 일정하다.
아래와 같은 기준을 적용한다.



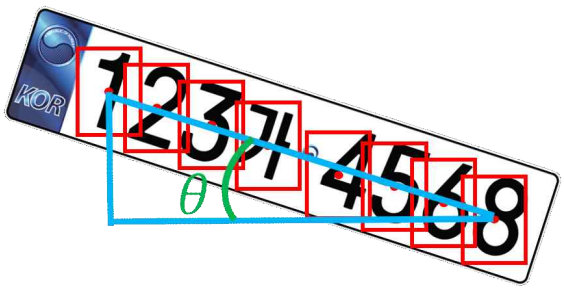
이를 통해 번호판과는 거리가 먼 BoundingRect들은 걸러지고 번호판의 숫자와 근접한 사각형들만 남게 된다. 번호판의 숫자 및 글자는 서로 붙어 있고, 최소 3개 이상 일정한 간격으로 붙어 있으므로 위치 및 배열 관계를 파악하여 최종적으로 숫자 부분을 도출한다.

2. 이미지 정렬(Affine Transform)

번호판이 항상 정면을 향하고 있을수는 없으므로 번호판이 비대칭으로 보이거나 삐뚤게 보일 때도 인식을 할 수 있어야 한다.

가. 방법1

번호판은 세로로 들고 있고, 왼쪽부터 읽게 되므로 x방향에 순차적으로 정렬을 해준다. 숫자 사각형들이 삐뚤어져 있다면, 수평과의 사이각을 구한다(arctan함수 사용).



위의 과정을 통해 각 숫자부분의 테두리를 검출하고, 그 중심점들의 좌표를 입력하였다. 번호판 영상 영상 혹은 이미지가 삐뚤어져 있다면, 위와같이 처음 숫자와 마지막 숫자의 중심좌표가 수평면과 이루는 각을 구해서 그 각만큼 이미지를 회전시키면 올바른 이미지를 구할 수가 있다.

cv2.getRotationMatrix2D(center, angle, scale)

위의 함수를 사용하여 영상을 회전시킨다. center에는 BoundingRect의 중심 좌표, 각은 수평과의 사이각이다. 사각형의 중심을 기준으로 회전시키는 행렬을 얻는다.

cv2.getRectSubPix(img_rotated, box_size, center)

회전된 이미지를 바탕으로 번호판 부분 이미지를 자른다. 수평정렬된 번호판의 이

미지가 반환된다.

정렬된 번호판의 이미지를 다시한번 이진화 및 처리를 한다.

나. 방법2
번호판의 맨 첫 번째 숫자의 왼쪽 두 꼭짓점과 맨 마지막 숫자의 오른쪽 두 꼭짓점을 4개의 점으로 잡고 아래의 함수를 사용한다.

```
cv2.getPerspectiveTransform()
cv2.warpPerspective()
```

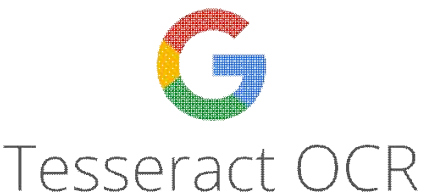
위에서 얻은 행렬을 사용하여 영상(이미지)을 회전시킨다.

D. Pytesseract를 이용한 숫자 감지 및 출력

Tesseract는 OCR 이미지로부터 텍스트를 인식 및 추출하는 소프트웨이다, 번호판 이미지가 깔끔하게 정렬 되면 Tesseract-OCR을 통해서 그 영역의 글자가 무엇인지 인식 및 추출한다. 코드는 다음과 같다.

```
pytesseract.images_to_string(img, lang='kor',
config='--psm7 --oem')
```

이미지를 문자열로 바꿔주며, 언어는 한국어, 설정값은 psm7로 문자열이 한줄로 나열되어 있음을 의미한다. 뒤의 oem은 테서렉트 엔진 버전이다.



(3) Project 시나리오

1. 사람이 번호판 모형을 들고 카메라 앞을 돌아다닌다.
2. 번호판의 각각 숫자와 글자에 직사각형 테두리가 쳐진다.
3. 인식한 번호가 모니터에 출력된다.

번호판 모형은 하드보드지에 붙여서 만든다.

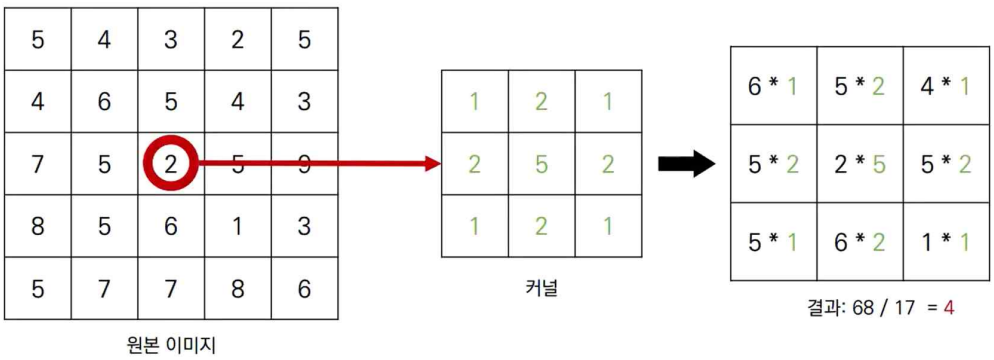
3. Project 진행 일정

진행 목표	3월	4월					5월		
	5	1	2	3	4	5	1	2	3
실시간 영상 받아오기			중간고사						
Gray scale, 필터, 이진화 처리 등									
Contours 로 숫자부분 Bindary Rectangle 인식									
패키지로 글자 인식 및 출력									
디버깅 및 테스트									

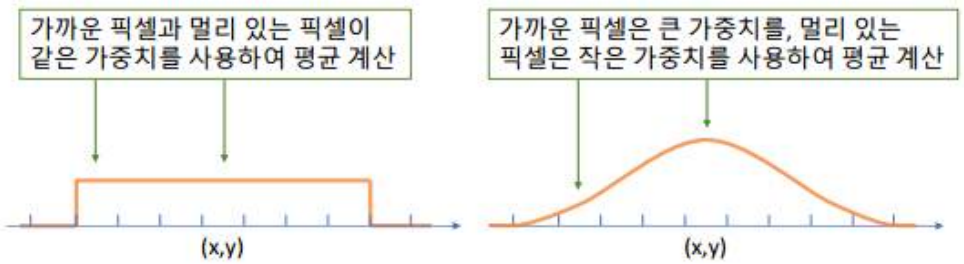
4. 용어 정리

1. Gaussian Blur

Convolution 계산이란, 임의의 픽셀값들의 이미지에서 한 픽셀값과 그 주변값으로 이루어진 행렬에 커널 행렬을 곱한 후, '(그 곱셈값들의 합) / (커널 값들의 합)' 으로 원래의 중심 픽셀 값을 조정하는 방식이다. 이 계산을 통해 커널을 곱하고 원 픽셀 값을 조정하므로써 경계가 모호해지고, 노이즈가 제거되는 블러 효과를 얻을 수 있다.



씌울 수 있는 여러 커널중 두가지 예시는 Basic Kernel과 Gaussian Kernel 이다. Basic Kernel은 모든 정사각행렬의 값들이 같은 커널이다. Gaussian Kernel은 정규분포(Normal Disturibition)을 따르기에 중간이 가장 높은 값을, 밖으로 퍼져 나갈수록 작은 값들을 가지는 커널이다.



2. Numpy(넘파이)

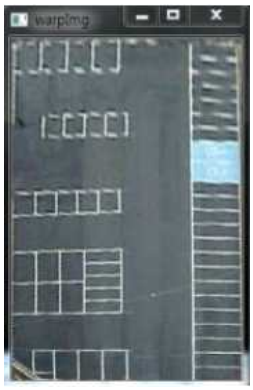
넘파이는 수학 및 과학 연산을 위한 파이썬 패키지이다. C언어로 구현된 파이썬 라이브러리로써, 고성능의 수치계산을 위해 제작되었다. 벡터 및 행렬 연산에서 매우 편리한 기능을 제공한다. 작품을 만들 때, Numpy의 기본 데이터 관리 단위인 array를 많이 사용할 예정이다.

3. 이미지 워핑(Warping)

이미지 와핑(Warping)이란, 기하학적 변형(Geometric Transformation)의 한 종류로써, 간단하게 한 (x, y)의 위치에 있는 픽셀을 (x', y')으로 대응시키는 작업을 의미한다. 예를 들어 스마트폰 앱 중에 문서를 카메라로 촬영한 후, 이를 PDF 파일로 변환할 때 이런 와핑을 주로 사용하게 된다.

OpenCV에서도 이런 Warping을 쉽게 할 수 있도록 하는 함수들이 구현 되어 있다. 앞서 말했듯이 Warping이란 (x, y) 위치의 픽셀을 (x', y')으로 대응하는 것으로, 대응시키기 위한 행렬이 필요한데 이 행렬이 바로 Transform Matrix이다. OpenCV에는 이 Transform Matrix를 구하는 함수도 정의 되어 있다. Warping을 하는 방법은 아래와 같다.

(1) (x,y)에 해당하는 4점의 좌표를 지정
(2) (x', y')에 해당하는 4점의 좌표를 지정
(3) getPerspectiveTransform()함수를 사용하여 Transform Matrix를 구함
(4) warpPerspective()함수를 이용하여 실제 Warping.



4. Tesseract-OCR

OCR(Optical Character Recognition)은 광학 문자 인식이라는 뜻으로, 사람이 쓰거나 기계로 인쇄한 문자의 영상을 이미지 스캐너로 획득하여 기계가 읽을 수 있는 문자로 변환하는 것이다. Tesseract는 구글에서 개발을 후원하는 무료 OCR엔진 소프트웨어이다. 이 테서렉트를 파이썬에서 사용할 수 있도록 구현해놓은 패키지가 pytesseract이다.

5. 참고 자료

https://hyongdoc.tistory.com/411
가우시안 블러

https://www.youtube.com/watch?v=PpTl7xxGXh4&t=32s
이미지 프로세싱

https://opencv-python.readthedocs.io/en/latest/doc/10.imageTransformation/imageTransformation.html
Affine Transformation

https://www.youtube.com/watch?v=j101FuFbRfE&t=5s
https://m.blog.naver.com/samsjang/220516697251
컨투어

https://www.youtube.com/watch?v=Y7XBsFzByTQ
https://www.youtube.com/watch?v=PpTl7xxGXh4
파이테서렉트

파이썬 기본: 점프 투 파이썬 (지은이: 박응용)